Discriminatively Reranking Abductive Proofs for Plan Recognition

Discriminatively Reranking Abductive Proofs for Plan Recognition

Sam Wiseman and **Stuart M. Shieber**
School of Engineering and Applied Sciences
Harvard University
Cambridge, MA 02138
{swiseman,shieber}@seas.harvard.edu

Sam Wiseman and Stuart M. Shieber. Discriminatively reranking abductive proofs for plan recognition. In *Proceedings of the 24th International Conference on Automated Planning and Scheduling (ICAPS)*, Portsmouth, New Hampshire, 21–26 June 2014.

Abstract

We investigate the use of a simple, discriminative reranking approach to plan recognition in an abductive setting. In contrast to recent work, which attempts to model abductive plan recognition using various formalisms that integrate logic and graphical models (such as Markov Logic Networks or Bayesian Logic Programs), we instead advocate a simpler, more flexible approach in which plans found through an abductive beam-search are discriminatively scored based on arbitrary features. We show that this approach performs well even with relatively few positive training examples, and we obtain state-of-the-art results on two abductive plan recognition datasets, outperforming more complicated systems.

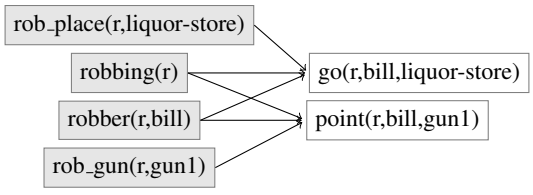
Introduction

Logical abduction has long been recognized as an attractive approach to plan recognition, since it provides a formal means of allowing an agent to make assumptions about its observations in an effort to explain them. Consequently, many earlier plan recognition systems were primarily abductive (for example that of Ng and Mooney (1992)). While the abductive approach is very powerful, one of its drawbacks is that the ability to make assumptions results in there being many valid abductive proofs for any set of observations, and no clear method for distinguishing the good from the bad. Early abductive systems dealt with this issue either by formulating various heuristics for choosing among candidate proofs (Ng and Mooney 1992), or by manually defining costs and weights on atoms and axioms, and attempting to find a least-cost proof (Hobbs et al. 1993). Another problem with the classical formulation is that it does not easily handle uncertainty in the axioms or plan-recipes. Recently, there have been a number of proposals to deal with these issues (primarily in the plan recognition domain) by adapting formalisms that integrate logic and probability, such as Markov Logic Networks, to be abductive (Singla and Mooney 2011; Raghavan and Mooney 2011). In addition to naturally handling uncertainty, these approaches provide a principled way of selecting one abductive explanation from others, since we can, for instance, select the most probable explanation. In what follows, we argue that there is a simpler, discriminative approach to addressing these two issues, and we show that this approach outperforms these more complicated systems on two datasets commonly used to evaluate abductive systems.

Review of Abduction for Plan Recognition

Formally, given a knowledge base of first-order formulae T, and a conjunction of (existentially quantified) observations O, an abductive proof is a set of atomic assumptions A that is consistent with T, and that together with T implies O (Ng and Mooney 1992). In plan recognition, the A we seek generally includes the instantiation of one or more logical predicates that correspond to high-level actions. To take an example from the dataset of Ng and Mooney (described below), if we observe, as in Figure 1, actions O corresponding to Bill going to the liquor store and pointing a gun at the owner, we seek a set of atomic assumptions A to adequately explain O (given the background knowledge in T). A might include, for instance, that Bill is engaging in the high-level action of *robbing*, and that he goes to the liquor store for this purpose. Examples of the sort of background knowledge (contained in T) that we might appeal to also appear in Figure 1.

Abduction can also be used for the recognition of plans not explicitly defined in first-order logic. For instance, if we have a Hierarchical Task Network (HTN) (Erol, Hendler, and Nau 1994) with a high-level *robbery* task that is refined by an *armed-robbery* method, then for the purposes of abductive plan recognition we can view the decomposition of *armed-robbery* into its first subtask, which might involve going to the place to be robbed, as being specified by the horn-clause axiom $\forall_{x,y}\ \textit{armed-rob}(x,y) \rightarrow \textit{go}(x,y)$, and so on for the other subtasks and their decompositions. Thus, abduction in the HTN framework amounts to "proving" the observed primitive actions in terms of higher-level tasks, using the axioms as just described, and subject to precondition satisfaction. (Indeed, the preconditions can be viewed as a tractable approximation of the consistency-checking technically required for abduction.) Because a proof of this form implies a particular plan, there is a natural correspondence between logical atoms and instantiated actions, and we will often speak of a "proof" and a "plan" interchangeably. Throughout, we will think of plans and proofs as directed acyclic graphs (DAGs), where there is a directed



Axioms

$$\text{robbing}(r) \wedge \text{robber}(r, x) \wedge \text{rob\_place}(r, y) \rightarrow \text{go}(r, x, y)$$
$$\text{robbing}(r) \wedge \text{robber}(r, x) \wedge \text{rob\_gun}(r, z) \rightarrow \text{point}(r, x, z)$$

Figure 1: A simplified robbery plan DAG and the (implicitly universally quantified) axioms it uses; observations are white and assumptions grey

edge between instantiated antecedent and instantiated consequent. We show a very simplified such DAG representing a (non-HTN) first-order logic representation of the robbery plan mentioned above in Figure 1, along with the axioms (which are implicitly universally quantified) used in its composition. There, white-shaded atoms are observed, and grey-shaded atoms are assumed. Together, the grey-shaded atoms constitute an abductive proof of the observed, white atoms.

In Figure 1, every observed atom is explained (i.e., not assumed). In general, however, observations may also be assumed, and non-observations may also be explained; that is, there may be *internal*, non-observed nodes in the DAG that are explained by *other* assumed nodes.

A Discriminative Approach

Whereas statistical relational formalisms such as Markov Logic Networks and Bayesian Logic Programs are useful for specifying distributions over random variables that are described by first-order relationships (Richardson and Domingos 2006; Kersting and De Raedt 2001), we argue that this is unnecessary for doing effective plan recognition. Rather, we hypothesize that (structural) features of the plan or proof itself (and perhaps features of the knowledge base too) are sufficient to discriminate better plans from worse. To continue our robbery example, whereas a statistical relational approach might, given observations about Bill pointing a gun, predict that the high-level goal is robbing because the marginal probability of this event is high given the observations, we suggest that it may instead be sufficient to merely score the plan DAG itself for plausibility. That is, perhaps we can learn to discriminate between plan DAGs by merely looking at their properties, such as how connected certain observations are in the DAG (a feature highlighted by Ng and Mooney (1992) and elaborated on below), or whether a pair of subgoals (like pointing a weapon and taking cash) appear together in the DAG. Such an approach allows us to avoid modeling unnecessary parts of the (potentially complicated) domain in question, and to focus our modeling on the discriminative task we care about. Moreover, if our features are sensitive to the axioms used in obtaining a particular proof, we can additionally learn that certain axioms are more reliable than others, thus handling uncertainty in the domain as well.

Review of Reranking

Reranking is a technique popular in natural-language processing, which typically involves training a discriminative model to select from among the top K structured predictions made by a generative model. For instance, if we have a probabilistic parser that can produce the K most probable parse trees for a given sentence, we can train a discriminative model to select the best parse from among these candidates, and use this as our final prediction. This technique is useful because we typically require some local, factorized model to generate predictions in a structured space, but reranking allows our final predictions to also depend on global features of the predicted structures (Collins and Koo 2005). We can apply this idea to abductive plan recognition by employing a (heuristic) beam-search through the proof space as our "generative model" for abductive proofs, and then training a reranker to discriminatively score the returned candidate proofs. This approach can be quite effective if our search is able to reliably generate a (not-too-large) superset of the correct proof(s) for a given recognition problem. (We return to this issue below.)

Reranking Model

To train a reranker, it is necessary to specify a loss function to be minimized over the training set. One popular approach is MaxEnt reranking (Charniak and Johnson 2005), which models the conditional probability of a particular candidate being the best candidate with a MaxEnt (i.e., multinomial logistic regression) model. More formally, given a set $\mathcal{Y}$ of candidate proofs/plans for a set of observations O, we can model the conditional probability that $y \in \mathcal{Y}$ is the best proof in the set as $p(y \mid \mathcal{Y}) = \frac{\exp(\mathbf{w}^\mathsf{T}\phi(y))}{\sum_{y' \in \mathcal{Y}} \exp(\mathbf{w}^\mathsf{T}\phi(y'))}$, where we have omitted the conditioning on the observations O for brevity. Above, ϕ maps proof/plan DAGs (such as that depicted in Figure 1) into feature space (i.e., to $\mathbb{R}^d$, where d is the number of features), and $\mathbf{w}$ is a (learned) weight vector. The features extracted by ϕ are elaborated on below.

Given training data $\mathcal{D} = \langle y^{(1)}, \mathcal{Y}^{(1)} \rangle \ldots \langle y^{(N)}, \mathcal{Y}^{(N)} \rangle$ consisting of N correct plans paired with a set of candidate plans generated by our search algorithm (which we assume to contain the correct plan) for a particular set of observations O, we can then train $\mathbf{w}$ so as to minimize the negative log-likelihood:

$$-\ln p(\mathcal{D} \mid \mathbf{w}) = -\sum_{i=1}^{N} \left[\mathbf{w}^\mathsf{T}\phi(y^{(i)}) - \ln \sum_{y' \in \mathcal{Y}^{(i)}} \exp(\mathbf{w}^\mathsf{T}\phi(y')) \right] \quad (1)$$

The gradient for Equation (1) takes a familiar form (see Riezler et al. (2002) for the derivations), and this convex function can be minimized using an off-the-shelf minimizer.

Because there is no guarantee that all the sets of plans $\mathcal{Y}^{(i)}$ are of the same size, we see that minimizing Equation (1) will cause our learned $\mathbf{w}$ to be biased in favor of minimizing the scores of the incorrect proofs/plans in the *biggest* sets. This is particularly problematic when using a search

procedure to generate our candidate plans, since owing to constraints (such as preconditions) in the domain, certain observations may generate many more candidate plans than others. To avoid this bias, we modify Equation (1) to instead be a weighted sum over training examples. In particular, we weight each example by $\gamma^{(i)} = \frac{\max_{1 \leq j \leq N} |\mathcal{Y}^{(j)}|}{|\mathcal{Y}^{(i)}|}$, giving the modified objective

$$\sum_{i=1}^{N} \gamma^{(i)} \left[-\mathbf{w}^{\mathsf{T}} \phi(y^{(i)}) + \ln \sum_{y' \in \mathcal{Y}^{(i)}} \exp(\mathbf{w}^{\mathsf{T}} \phi(y')) \right] + \lambda \mathbf{w}^{\mathsf{T}} \mathbf{w}, \tag{2}$$

where we have also added on an ℓ_2 regularization term to prevent overfitting. Since the $\gamma^{(i)}$ are all positive, Formula (2) is a positive weighted sum of convex functions, and so is still convex.

Assuming we have access to a training set consisting of lists of observations paired with the full plan that generated them, the procedure for training is, for each list of observations, to use a beam-search to generate a set of candidate plans assumed to include the correct plan, and then to train the reranker over these sets using Formula (2). For prediction, a set of candidate plans is generated by the same search for a new list of observations, and we predict the candidate plan scored most highly by the trained reranker.

One nice thing about this set-up is that it effectively makes use of *negative* training examples, which are cheap to generate. That is, we can generally create many negative training examples in our search by simply expanding our beam-width. While having too many negative training examples can cause classifiers to classify almost everything negatively, we do not have this problem in our case because we will always be selecting the plan scored most highly *from the candidate set*.

Experimental Evaluation

We evaluate our approach on two datasets commonly used to evaluate abductive plan recognition systems, to facilitate comparison with other systems.

Datasets

The first dataset we consider is that originally used by Ng and Mooney to evaluate their ACCEL plan recognition system. This dataset contains logical representations of "stories" involving agents and their actions, and the high-level plans (such as robbing) that give rise to these actions must be predicted. Our robbery example above is a simplified adaptation of one rather short such story. The dataset includes a background knowledge base of approximately 180 axioms (some of which are recursive), a 25-plan training set, and a 25-plan test set (Ng and Mooney 1992).

The second dataset we consider is the Monroe corpus. This corpus consists of 5000 hierarchical plans relating to emergency response artificially generated by a modified HTN planner. Unlike the plans in the previous dataset, which can be arbitrary DAGs, these HTN-type plans are all trees. The plans in the corpus involve 10 top-level goal schemas and 38 sub-goal schemas, with an average of 9.5 action-observations per plan and an average subgoal-depth of 3.8 per plan (Blaylock and Allen 2005).

Features

We use the same features for both datasets. These features attempt to capture various structural features of predicted plans, as follows:

- **Action/Predicate Counts:** we count the occurrences of each action/predicate appearing, respectively, in the observed, assumed, and explained sets of atoms appearing in the DAG, as well as the total atoms in each set. For example, for the DAG in Figure 1, the feature 'observed-go' would have value 1, as would the feature 'assumed-robber'; the total counts for the observed and assumed sets are 2 and 4, respectively. (Note that in the Monroe domain the top-level goal will be the only atom ever assumed, since the plans are trees.)
- **Simplicity and Coherence Features:** these features are motivated by the "simplicity" and "coherence" metrics that are observed by Ng and Mooney to be useful in plan recognition (1992). The former counts the number of explained atoms minus the number of assumed atoms, and the latter is the proportion of pairs of observed atoms that share a common parent atom in the induced DAG. In Figure 1, these values are -2 and 1/1, respectively, since there is only one pair of observed atoms, and they share the 'robbing' node as a parent. In addition, we use an unnormalized version of the coherence score, and the ratio of the numbers of explained and assumed atoms.
- **Generalized Coherence Features:** we count the number of observation pairs (by action/predicate) that share a common parent, the number of internal atom pairs (by action/predicate) that share a common parent, and the number of internal atom pairs (by action/predicate) that merely co-occur in a plan. In Figure 1, then, the value of the feature 'observed-share_parent-go-point' would be 1, and similarly for any internal nodes that co-occur or that share a parent.
- **Graph Features:** we additionally have features for the number of edges and nodes in the induced proof/plan DAG.

ACCEL Experiments While the ACCEL system originally developed for the story understanding data scores itself on the precision and recall of the atoms it assumes in proving the observed actions, Raghavan recasts the problem in terms of pure plan recognition by defining high-level actions in terms of the plan-related atoms, and scoring her system in terms of its precision and recall on these high-level plans (Raghavan 2012). Because the other machine-learning based approaches to abductive plan recognition we consider also adopt this evaluation framework, we will adopt it too to facilitate comparison.

As a first experiment, we use reranking in a fairly traditional way, namely, we learn to rerank the top-K candidates produced by a black-box plan recognizer. In particular, we use the original ACCEL system, which uses a beam search to search through the proof space, and which selects the set

of assumptions that score most highly according to the "coherence" metric mentioned above (Ng and Mooney 1992). Clearly, under this scheme our prediction for a given test-example cannot be any better than the best candidate plan among the candidates selected by ACCEL. However, if ACCEL fails to predict the best plan of the candidates it has generated, we may be able to learn to select the best candidate more reliably. To generate training data, we set ACCEL's beam-widths so that it would output a large number of candidate proofs (typically around 90) for each example. We supplemented these candidates with additional negative examples we generated by searching through the proof space, so that each training example had approximately 150 candidate proofs. We then trained our reranker on this supplemented training corpus. To test, we took the candidates produced by ACCEL under its normal settings (typically around 10 per test example), and selected the highest scorer using our learned reranker. We compare these results with those achieved by ACCEL when it selects the best proof itself. In Table 1 we show the result of these experiments, where precision and recall are macro-averaged over the 25 test examples. We also compare with Raghavan and Mooney (2011) and Singla and Mooney (2011) ('R&M' and 'S&M' respectively). We see that reranking improves ACCEL's results slightly, as we had hoped.

	Precision	Recall	F-Score
ACCEL + Rerank	**91.43**	**90.19**	**90.81**
ACCEL	89.39	89.39	89.39
Beam + Rerank	87.94	85.10	86.50
R&M	72.07	85.57	78.24
S&M (Best)	69.13	78.94	73.73

Table 1: Story Understanding results

While this is an encouraging result, it is presumably much easier to rerank the (limited) output of a system that already does quite well on the recognition task than to rerank a much noisier and larger set of predictions. Moreover, it is not realistic to assume we would have access to as effective a search procedure as that provided by ACCEL on an arbitrary plan-recognition task. We therefore experiment with reranking the results of a much less targeted beam search. We found that the union of the results of 3 beam-searches, each using one of the following simple heuristics, was sufficient to recover candidate sets containing the correct plan for all plans in the training set when using a beam-width of size 50 and 25 breadth-first expansions: **(1)** prefer plans that leave the fewest number of unexplained observations; **(2)** in addition prefer plans where no observation is explained by another observation and where no observation is explained merely by a renaming; **(3)** in addition prefer plans that use fewer recursive axioms. This resulted in an average of approximately 2195 candidate plans per training example, and we trained the reranker over these. The results of prediction on the test plans (using candidates generated by the same search procedure, which generated approximately 2137 candidates per text example on average) are also in Table 1 (as "Beam + Rerank"). While the results of this scheme are slightly below those of the ACCEL system, we see that they outperform the other machine-learning based approaches on all metrics, with the exception of the system of Raghavan and Mooney on recall. In addition, while it can be difficult to train these latter systems on the mere 25 training examples, thus requiring manual tuning of certain parameters (Raghavan and Mooney 2011), we are able to effectively train using cheaply generated negative examples.

Monroe Experiments Whereas the abductive plan recognition systems of Raghavan and Mooney (2011) and Singla and Mooney (2011) evaluate themselves on predicting the top-level goals of plans in the Monroe corpus, we additionally focus on predicting the subgoals. We tackle this more difficult task because given the relatively small number of top-level goals and their preconditions (which abductive systems presumably have access to), observed actions are frequently unambiguous with respect to the top-level goal. Subgoals, however, are both more ambiguous and often generated recursively, and so predicting the goal hierarchy can be challenging. We therefore follow the later work of Blaylock and Allen (2006) in attempting to predict this hierarchy level by level, with top-level goal prediction equivalent to predicting the 0'th level of the hierarchy.

Following Blayock and Allen (2006), we randomly selected 500 plans from the corpus for testing, and we report the "convergence" percentage for subgoal prediction at different levels of the hierarchy. (We also follow Blaylock and Allen in only reporting convergence results for subgoals that correspond to at least two observed actions.) This convergence percentage refers to the percentage of subgoal schemas correctly identified at different levels of the subgoal hierarchies across the plans in the test set.

For these experiments, we used 2500 randomly selected plans for training, and the rest as development data. To search, for each top level goal we ran a depth-first beam search from the observations, and pruned candidate plans that could not be explained in terms of this goal, finally taking the union of the results. This yielded an average of approximately 434 candidate proofs per training example. In Table 2 we compare our 0'th level results with those of the other abductive systems considered above, and our results at all levels with those of Blaylock and Allen, who use a Cascading HMM approach to subgoal prediction (Blaylock and Allen 2006). Table 2 also shows (under the '#' column) for each level how many plans in the test-set contain sub-goals corresponding to at least two observations. We see that our approach performs best on all levels except the 0'th.

Level	#	B&A	Beam + Rerank	R&M	S&M
0	500	**100**	99.2	98.8	97.3
1	479	71.8	**94.6**	-	-
2	244	45.8	**81.1**	-	-
3	144	41.2	**84.0**	-	-
4	72	61.8	**88.9**	-	-
5	52	6.2	**96.1**	-	-
6	3	0	**100**	-	-

Table 2: Convergence percentage for different subgoal levels

Discussion and Future Work

The fact that the simple discriminative approach described above seems to lead to dramatically improved performance on a variety of plan recognition tasks should not be particularly surprising; it seems quite reasonable that structural features of plans and proofs would be useful in determining their plausibility. Indeed, similar intuitions about structural properties are exploited in the coherence and simplicity metrics of the ACCEL system and other such systems (Ng and Mooney 1992), and this presumably explains why ACCEL manages to outperform much more modern systems, which essentially ignore this information.

One drawback of the approach presented above is that it relies on the existence of a search procedure that can reliably produce a (not too big) superset of the correct plan/proof. While this may be easy to engineer in some domains, there are likely to be some domains where it is not. This issue is ameliorated somewhat, however, by the fact that there are a variety of approaches for learning search heuristics in structured spaces (Daumé III and Marcu 2005). The output of such a search could then be reranked as above. We leave experiments along these lines to future work.

It is also interesting to note here that there is intuitively a tradeoff involved in picking a beam width, since a larger beam will give the reranker more negative examples to learn from, but it will also require the reranker to select from more candidates at prediction time. We are unaware of any theoretical results characterizing this tradeoff precisely, but the results in Table 1 suggest that performance is hurt somewhat when moving from hundreds of candidate proofs to thousands. We leave further investigation of this to future work.

Acknowledgments

We are grateful to Sindhu Raghavan and Nate Blaylock for answering questions about their systems.

References

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

Blaylock, N., and Allen, J. 2006. Fast hierarchical goal schema recognition. In *Proceedings of the National Conference on Artificial Intelligence*, volume 21, 796.

Charniak, E., and Johnson, M. 2005. Coarse-to-fine N-best parsing and maxent discriminative reranking. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, 173–180. Association for Computational Linguistics.

Collins, M., and Koo, T. 2005. Discriminative reranking for natural language parsing. *Computational Linguistics* 31(1):25–70.

Daumé III, H., and Marcu, D. 2005. Learning as search optimization: Approximate large margin methods for structured prediction. In *Proceedings of the 22nd International Conference on Machine learning*, 169–176. ACM.

Erol, K.; Hendler, J.; and Nau, D. S. 1994. HTN planning: Complexity and expressivity. In *AAAI*.

Hobbs, J. R.; Stickel, M. E.; Appelt, D. E.; and Martin, P. 1993. Interpretation as abduction. *Artificial Intelligence* 63(1):69–142.

Kersting, K., and De Raedt, L. 2001. Bayesian logic programs. *arXiv preprint cs/0111058*.

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.

Raghavan, S., and Mooney, R. J. 2011. Abductive plan recognition by extending Bayesian logic programs. In *Machine Learning and Knowledge Discovery in Databases*. Springer. 629–644.

Raghavan, S. V. 2012. *Bayesian Logic Programs for Plan Recognition and Machine Reading*. Ph.D. Dissertation, University of Texas at Austin.

Richardson, M., and Domingos, P. 2006. Markov logic networks. *Machine learning* 62(1-2):107–136.

Riezler, S.; King, T. H.; Kaplan, R. M.; Crouch, R.; Maxwell III, J. T.; and Johnson, M. 2002. Parsing the Wall Street Journal using a lexical-functional grammar and discriminative estimation techniques. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, 271–278. Association for Computational Linguistics.

Singla, P., and Mooney, R. J. 2011. Abductive Markov logic for plan recognition. In *Proceedings of AAAI*.