Learning to Infer Final Plans in Human Team Planning

Joseph Kim[1], **Matthew E. Woicik**[1], **Matthew C. Gombolay**[2], **Sung-Hyun Son**[2], **Julie A. Shah**[1]
[1] Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology
[2] MIT Lincoln Laboratory*, Lexington, MA
[1]{joseph_kim, mwoicik, julie_a_shah}@csail.mit.edu
[2]{matthew.gombolay, sson}@ll.mit.edu

Abstract

We envision an intelligent agent that analyzes conversations during human team meetings in order to infer the team's plan, with the purpose of providing decision support to strengthen that plan. We present a novel learning technique to infer teams' final plans directly from a processed form of their planning conversation. Our method employs reinforcement learning to train a model that maps features of the discussed plan and patterns of dialogue exchange among participants to a final, agreed-upon plan. We employ planning domain models to efficiently search the large space of possible plans, and the costs of candidate plans serve as the reinforcement signal. We demonstrate that our technique successfully infers plans within a variety of challenging domains, with higher accuracy than prior art. With our domain-independent feature set, we empirically demonstrate that our model trained on one planning domain can be applied to successfully infer team plans within a novel planning domain.

1 Introduction

Intelligent decision support (IDS) systems are increasingly being introduced to work in concert with human planners in mission-critical and high-intensity domains. IDS systems are designed to reduce cognitive load and help their human partners produce high-quality plans. There is also a growing need for IDS systems that can effectively participate in human *team planning*, as plans in high-intensity domains are often negotiated between two or more responders prior to execution [Casper and Murphy, 2003]. In this paper, we focus on the development of an IDS agent that can perform *plan inference* by monitoring human teams' planning conversations and inferring their final plans.

An IDS agent with plan inference technology can enable an automatic generation of a plan summary. Furthermore, plan inference can serve as a prerequisite step for an IDS agent providing plan evaluation (i.e., visualizing the current plan's performance metrics) and critique (i.e., suggesting a more optimal course of actions). Plan inference can also reduce the translation burden of having human operators manually transcribe the deployment plans of autonomous systems. With reliable inference, robot plans can be implemented directly at the end of team planning sessions, potentially reducing execution times during time-sensitive missions.

Plan inference, also known as plan recognition in automated planning, is the problem of inferring users' intended goals and their corresponding plans through observed actions. In multi-agent plan recognition (MAPR), joint plans are inferred by observing the coordinated activities of several teaming agents. MAPR generally assumes access to a plan library, where each candidate plan fully specifies the temporally aligned activities of all agents [Banerjee *et al.*, 2010]. Recent works allow for noisy observations as input [Zhuo and Li, 2011], and even replace the need for a plan library with domain models [Zhuo *et al.*, 2012]. Most prior work related to MAPR, however, focuses on observing teams' *execution* of a predetermined plan; little work explores monitoring the process of how the plan was formed during team *planning sessions*.

Monitoring team planning sessions (i.e., collaborative planning dialogues) is challenging because the observations are highly stochastic: Actions are proposed, accepted, rejected, and modified intermittently, and not everything that team members say during a meeting appears in their final plan [Di Eugenio *et al.*, 2000]. Furthermore, team members alter their goals and preferences dynamically [Carberry, 1990].

In this paper, we present a novel plan inference technique to infer final plans from human team planning sessions. We employ a set of dialogue features to capture the context behind speakers' uttered actions and reason about the team's overall agreement process. We also employ planning domain models to efficiently search the large space of possible plans and use the costs of candidate plans as a reinforcement signal to train our plan inference model. (Such an approach forgoes the need for a plan library.)

We evaluate our technique on human team planning datasets across four different planning domains. We demonstrate that our technique infers teams' final plans with higher accuracy than prior art. With our domain-independent feature set, we empirically demonstrate that our learned model can be applied to successfully infer team plans within a novel plan-

*DISTRIBUTION STATEMENT A. Approved for public release: distribution unlimited.

ning domain, and that performance is robust to incomplete planning domain models.

2 Related Work

Plan inference, or plan recognition (PR), can be considered the inverse to traditional planning. While planning aims to determine a sequence of actions in order to achieve a given goal, PR attempts to determine the goal and its corresponding plan through observed actions. PR is an active research area in the fields of automated planning and human-computer interaction, where observed actions can correspond to users' motion, uttered speech, or activities within interactive computing environments [Freedman *et al.*, 2014; Amir and Gal, 2013]. Earlier models for PR assume access to plan recipes that encode how the user might perform a set of actions to accomplish one of several goals [Carberry, 2001]. Other approaches assume access to a large database of users' previously executed plans (i.e., a *plan library*) and select the most similar plan during testing [Zhuo *et al.*, 2013]. Recent works have transformed PR into a modified planning problem, utilizing domain theory and off-the-shelf planning algorithms for an efficient search [Ramırez and Geffner, 2009; Sohrabi *et al.*, 2016; Pereira *et al.*, 2017].

As an extension to PR, multi-agent plan recognition (MAPR) is the problem of inferring a joint plan from observing coordinated activities of a team of agents. MAPR is challenging because it requires reasoning about each agent's contribution to a particular portion of a collaborative plan. Similar to PR, MAPR models generally perform plan inference through the use of a plan library [Banerjee *et al.*, 2010]. Even with access to a plan library, Banerjee et al. [2010] proved the NP-complete hardness of MAPR. Several authors have developed approaches to mitigate the complexity by shrinking the hypothesis space, by imposing temporal dependencies and constraints within the plan library [Kaminka *et al.*, 2002; Sukthankar and Sycara, 2006]. More recent works allow noisy observations and replace the need for a plan library with compact action models [Zhuo and Li, 2011; Zhuo *et al.*, 2012].

Most of the work related to MAPR, however, has focused on observing teams' plan *execution*. In our MAPR setting, we focus on monitoring human team *planning sessions* – a natural deliberation process that occurs among multiple decision makers to achieve consensus and generate a plan. Note that the discussants may not be the agents involved in the actual plan execution. We propose a *plan inference* technique that utilizes features from teams' dialogue patterns, in concert with the use of planning domain models.

Our work is closely related to the plan inference model proposed by Kim et al. [2015]. In this work, the authors designed a probabilistic generative model to infer final plans from team conversations. The model employs Gibbs sampling with logical plan validation to inform its prior over the space of possible plans. However, candidate plans were proposed by randomly inserting/deleting actions, where the model suffered from scalability in sampling a valid plan as the number of actions increased. Furthermore, there was no learning component to the model – in order to infer plans, a computationally expensive sampling procedure was repeated for every conversation datapoint. Instead of only relying upon plan validation, our plan inference model leverages a richer amount of planning information, including the estimated plan cost, action consistency, and landmark analysis. We propose a learning model that can efficiently learn team planning behavior from training data and successfully apply it to infer plans in novel domains.

Our work is also closely related to the Sohrabi et al. [2016] approach for performing PR with unreliable observations. In this work, the authors approach PR for STRIPS problem by designing a transformed planning problem that incorporates additional "explain" and "discard" operators, and modifies the overall plan cost to incorporate penalties for discarding observations. A cost-optimal planner is then utilized to produce the a set of most likely plans with respect to the overall cost metric. Our approach adopts the same overall cost metric, but uses plan cost as the signal to train a learning model. The learning approach is advantageous, as compared to the Sohrabi et al.'s planning approach, because it can readily incorporate a variety of dialogue and planning features that capture additional information about the most likely plan from the team conversation, thereby providing robustness to incomplete model specifications and some interpretability of feature importance in producing most likely plan. Finally, our approach is also readily applied to numerical planning and discrete optimization problems, in addition to the propositional planning problems addressed in Sohrabi et al.

Rather than working with raw, natural language, our IDS agent takes as input a structured form of conversation with semantically parsed actions. Automatic parsing and grounding of raw utterances is an important area of research in natural language processing [Artzi and Zettlemoyer, 2013]; however, we view this as a separate problem and do not focus on it in this paper. Our form of input still preserves the natural patterns of group decision-making and presents the challenge of inferring plans from noisy observations.

3 Preliminaries and Problem Formulation

Our task is to infer the final plan from a processed form of the human team's planning conversation. We assume that the planning problem under the human team's deliberation can be represented using Planning Domain Description Language (PDDL). A *plan* is an action sequence $\pi = (a_o, ...a_n)$ that is identified as *valid* when the resulting state achieves the goal conditions. *Invalid* plans do not satisfy goal conditions, or contain action pairs that violate the precondition and effect rules described in the planning problem. In propositional or STRIPS-based planning, the cost of a valid plan is represented by its number of actions, $|\pi|$. In numerical planning, numeric-valued state variables can be defined (e.g. total fuel consumption, energy usage), and the cost of plans may be expressed as an arithmetic function of these numerical state variables. Planning files that describe the dynamics of the domain and a problem instance are assumed to be available to the IDS agent.

The primary input to our plan inference problem is a list of observed (uttered) actions from the human team's plan-

Speaker	Raw Utterance	O: **Actions**	ϕ_2: DAs	ϕ_3: APs
A	Heading back to base is required for recharge	**Goto(base)**	Inform	
A	Let's go pick up P2	**Pickup(P2)**	Suggest	
B	How many steps is that?		Ques. - Inform	
A	11 units		Inform	
B	Ok nice		Assess	Positive
B	Then we should strike enemy UAV2	**Target(UAV2)**	Suggest	
A	Hmm. After that we could run out of fuel on way back		Inform	Partial

Table 1: A sample conversation segment from the human team planning dataset (domain = UAV mission planning). Raw utterances are parsed into a format of planning actions, O. Our feature set includes the use of dialogue acts (ϕ_2: DAs) and adjacency pairs (ϕ_3: APs).

ning conversation (see the third column in Table 1). Actions are presented in a structured, machine-understandable format. These observed actions provide evidence toward the formation of the final plan the team intends to execute. Observations are noisy, in part due to the natural tendency to discuss several potential options before converging onto a final plan. Additionally there are missing observations (i.e., actions that appear in the final plan are not explicitly mentioned during the team conversation). Such actions may be implicitly suggested or deemed obvious within the shared understanding of the team. Thus, simple concatenation of all actions will most likely result in an invalid plan.

In many real-world planning domains, the number of possible plans to address a given problem can be large [Gerevini and Long, 2005]. Finding even a valid plan is PSPACE-complete [Bylander, 1991]. It is therefore intractable for the agent to enumerate all possible plans and perform similarity matching; the agent must infer plans without using a plan library.

We also desire for the agent to learn a computational model without having access to a traditional, supervised dataset. That is, even without knowing the actual plans (e.g. plan execution traces, post-meeting reports) the team in question has ultimately committed to, the agent should be able to learn plan formation behavior through conversation data alone.

4 Our Plan Inference Model

The main idea of our approach is to leverage the context from team dialogue and planning features to infer the team's final plan. Our model iterates according to the following main steps (a visual illustration of these steps is provided in Figure 1):

1. **Action selection policy**: From the set of uttered actions (observations), the agent picks a subset that the team will likely commit to.
2. **Low-level planner execution**: The low-level planner generates a plan that navigates through the selected actions.
3. **Policy update**: Policy weights are updated using the generated plan's "overall" cost as the reinforcement signal. Overall cost combines the original plan cost with the cost accounting for unselected actions.

Note that the predictor (step 2) must output a valid plan. It is possible that human teams form invalid plans at the end of their planning sessions; this can occur when team members overlook constraints or lose situational awareness [Miller, 2004]. However, if invalid plans are allowed for prediction, the number of possible plans significantly and unnecessarily increases. We designed the agent to only sample the valid plan space not only to keep the search manageable, but also due to the notion that valid plans are ultimately desirable for execution. We assume that human mistakes, if they occur, result in only small perturbations from a valid plan.

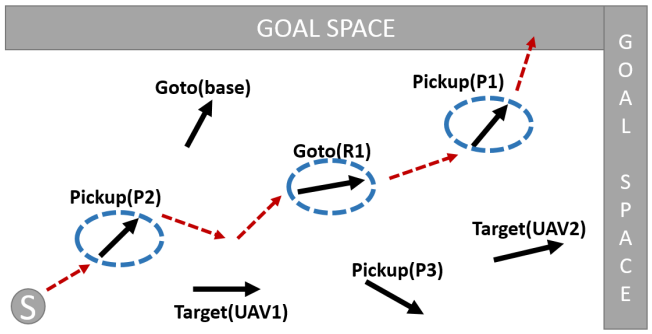

Figure 1: From the set of observed actions (↗), the action selection policy picks a subset that the team will likely commit to (highlighted by ⬭). The low-level planner then generates a valid plan by navigating through the selected actions (⇢). The overall cost of the plan is used to update the policy. Ⓢ is the initial problem state.

4.1 Action Selection Policy

We define a planning conversation as a document d, and a set of all observed actions in that document as O. For each action $o_i \in O$, a binary decision variable a_i is assigned, denoting whether or not the action o_i will be included in subsequent planning. We assume that $\{a_i\}$ from $i = 1, .., n$ ($n = |O|$, number of unique actions) are conditionally independent given the document (Equation 1). Such an assumption is required to make our policy learning computationally tractable. While the initial action selection is carried out independently, the dependency of selected actions will be addressed through their effect on plan generation.

$$P(A|d;\theta) = P(a_1, a_2, .., a_n|d;\theta) = \prod_{i=1}^{n} P(a_i|d;\theta) \quad (1)$$

We model individual action selection decisions via a log linear distribution [Lafferty *et al.*, 2001], allowing for a diverse range of document-action features:

$$P(a_i|d;\theta) = \frac{\exp(\theta \cdot \phi(d, a_i))}{\sum_{a_i'} \exp(\theta \cdot \phi(d, a_i'))}, \quad (2)$$

$\phi(d, a_i) = \langle \phi_1, ..., \phi_k \rangle$ is a k-dimensional feature representation, and θ is the corresponding weight vector to be learned. We use the following set of features to capture context from both the dialogue and the planning information:

- ϕ_1 *(Action counts)*: Repetition of information is one way of ensuring that a group continues to attend to the information and consider it when developing a plan [Larson *et al.*, 1996]. Actions that are more frequently addressed during team discussion may be more likely to appear in the final plan.
- ϕ_2 *(Dialogue acts)*: Dialogue acts (DAs) are semantic labels that define the functional roles of utterances. An action expressed in the form of a deliberate "suggestion" can have a different likelihood of appearing in the final plan compared with an action mentioned during the course of regular information exchange. DAs have been successfully utilized as features for text summarization and monitoring shared understanding [Qin *et al.*, 2017; Kim and Shah, 2016]. We use a DA label set designed for monitoring group decision-making [Carletta, 2007].
- ϕ_3 *(Adjacency pairs)*: While DAs capture the speaker's intent behind an uttered action, adjacency pairs (APs) denote the listener's sentiment on that action. We use the following AP labels: positive assessment, negative assessment, and partial support [Somasundaran and Wiebe, 2007].
- ϕ_4 *(Relative position)*: Sentence location (relative to the document length) is a common feature used in text summarization [Gupta and Lehal, 2010]. Group decision-making is described as a "balance - propose - confirm" process [Di Eugenio *et al.*, 2000], wherein teams initially spend time exchanging information and then propose and commit to a plan. Actions proposed near the end of the meeting may be more likely to appear in the final plan than actions introduced at the beginning. This feature captures a notion the temporal ordering of actions.

ϕ_1—ϕ_4 are *dialogue features* that essentially store information about the conversation dynamics of team planning. Next, we describe the *planning features* that store information about an action's effect on plan generation:

- ϕ_5 *(Action feasibility)*: This feature determines whether an action (not the entire plan) is infeasible with respect to the planning problem. This could occur when speakers mention actions that are parametrized with incorrect/unavailable resources or objects.
- ϕ_6 *(Landmark analysis)*: Landmarks are states that all valid plans must pass through during their execution [Hoffmann *et al.*, 2004]. Landmark-based heuristics have been successfully utilized for both planning and goal recognition [Pereira *et al.*, 2017; Vered *et al.*, 2018]. We apply a feature function that checks whether an action's effect list contains one or more landmarks; it can serve as a guidance to determine if an action represents a potential stepping stone toward the goal.
- ϕ_7 *(Action consistency)*: This feature detects whether the current action is an outlier with respect to other actions in O, using the cost difference of a plan having to traverse through the entire action set versus a plan traversing the set excluding the current action: $C(\pi|O) - C(\pi|O \setminus \{o_{current}\})$. An inconsistent action with respect to other actions in O can incur a high cost difference.
- ϕ_8 *(Action similarity)*: An IDS agent may possess its own "optimal" plan irrespective of human discussion. This feature determines if the current action is a part of the agent's plan. Since our IDS agent possesses full planning information, π_{agent} is cost-optimal; however, note that there may be multiple optimal plans, and that the plans generated by the human team may not be cost-optimal.

With $\phi(d, a_i)$ and the current θ, the policy samples an action set using Equation 2, where $O_{select} = \{o_i \in O | a_i = 1\}$.

4.2 Low-level Planner Execution

With the action set sampled from the policy above, a low-level planner generates a plan that navigates through each action in O_{select}. The low-level planner attempts to produce a valid plan by "filling in the gaps" between actions in O_{select}. Inconsistent pairs of actions can make the problem unsolvable or trigger a plan with a large cost. Thus, the notion of the dependency of actions is represented by the resulting plan's validity and cost. We follow Ramırez and Geffner's [2009] method of compiling the execution of observed actions (but without enforcing their sequencing) as part of the goal state. The compilation allows the low-level planner to be treated as a black-box, where any off-the-shelf planning system can be integrated. For propositional planning problems, we employed LAMA [Richter and Westphal, 2010] and Metric-FF [Hoffmann, 2003] for numerical planning problems. Both are fast, satisficing planners and two of the top-performing planners at International Planning Competitions.

The planner returns a valid plan, $\pi|O_{select}$ and its cost, $C(\pi|O_{select})$. If the problem is unsolvable or the planner times out, a large cost, M, is returned.

4.3 Policy Update

Policy weights, θ, are updated using the following signal adopted from Sohrabi et al. [2016][1]:

$$C_{overall} \triangleq C(\pi|O_{select}) + \lambda \cdot |O \setminus O_{select}|, \qquad (3)$$

$|O \setminus O_{select}|$ is the number of *unselected* actions, and λ is a non-negative real number. $C_{overall}$ is a linear combination of the generated plan's original cost and the cost accounting for treating some of the observed actions as noise. λ is a hyperparameter that controls the penalty for excluding an action. Figure 2 depicts how as $\lambda \to \infty$, a plan navigating through the full O would be generated. On the other hand, as $\lambda \to 0$, the agent would disregard the observations and simply output its own optimal plan, π_{agent}. With cost function $C_{overall}$, the policy will learn to select actions to produce a low-cost plan while attempting to keep as many observations as possible.

[1]The original cost definition by Sohrabi et al. includes a separate penalty term for the number of missing actions, but is excluded in our cost signal. Generally, the number of missing actions will be reduced as $C(\pi|O_{select})$ is minimized.

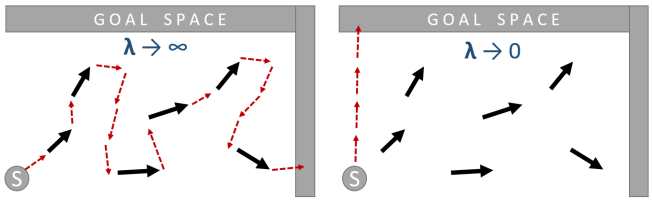

Figure 2: As $\lambda \to \infty$, the cost function in Equation 3 would train the policy to navigate through full O (left). As $\lambda \to 0$, the policy would be biased to disregard all observations (right).

We hypothesize that such a cost signal positively correlates with the process by which human teams form their plans (i.e., observe all the options given thus far, then work with a viable subset, leading to a high-quality plan). We test this hypothesis using real human team planning datasets across four different planning domains.

During learning (step 3), we aim to identify model parameters, θ, that minimize the expected value function based on the overall cost function defined in Equation 3:

$$J_\theta(A) = \mathbb{E}_{p(A|d;\theta)}[C_{overall}].$$

Although there is no closed-form solution, policy gradient algorithms [Sutton *et al.*, 2000] are capable of estimating θ by performing stochastic gradient descent. The gradient is calculated as follows:

$$\nabla_\theta J(A) = \mathbb{E}_{p(A|d;\theta)}[C_{overall} \cdot \nabla_\theta logP(A|d;\theta)] \quad (4)$$

$$\nabla_\theta logP(A|d;\theta) = \sum_{i=1}^{n} \nabla_\theta logP(a_i|d;\theta)$$

$$= \sum_{i=1}^{n} \phi(d,a_i) - \sum_{a_i'} p(a_i'|d;\theta)\phi(d,a_i') \quad (5)$$

We denote $\delta = \nabla_\theta logP(A|d;\theta)$, a gradient in which θ is updated in proportion to $C_{overall}$:

$$\theta \leftarrow \theta - \alpha \cdot C_{overall} \cdot \delta, \quad (6)$$

α is the learning rate. δ is computed using Equation 5 given our feature set $\phi(d,a_i)$ and θ. The full policy gradient algorithm for plan inference is summarized in Algorithm 1.

Algorithm 1 Policy gradient algorithm for plan inference.

Input: set of documents D, number of training iterations T
Output: θ

```
Initialize θ with random values
for t = 1 to T do
    for each document d ∈ D do
        O_select ← ∅
        for each observed action o ∈ O do
            Sample action based on policy in Eqn. 2
            if a_i = 1 then O_select = O_select ∪ {o}
        Execute low-level planner to obtain π|O_select
        Compute C_overall using Eqn. 3
        Update θ using Eqn. 6
```

As depicted in Algorithm 1, each training iteration cycles through a full set of documents, D. The total training runtime complexity is $\mathcal{O}(T \cdot |D| \cdot |O|)$, and the memory complexity is $\mathcal{O}(k)$, scaling in the number of features. The test phase involves a sampling of O_{select} is from the learned policy (Equation 2), and a single low-level planner call to output $\pi|O_{select}$ for prediction.

Policy gradient algorithms optimize a non-convex objective and are guaranteed to only produce the local optimum. However, they scale well to large state spaces, and have previously been successfully utilized for various planning tasks (e.g., learning action models and precondition relations from text [Zhuo and Kambhampati, 2013; Branavan *et al.*, 2012]).

5 Applying the Model

5.1 Human Team Planning Dataset

We tested our model on a human team planning dataset collected by Kim and Shah [2016], wherein teams of two participants each conversed via web chat and generated plans for a hypothetical emergency response scenario. The goal was to transport a number of scattered patients to hospitals, with each team deciding on the priority of patients and the utilization of transport vehicles. Teams were allotted 20 minutes of chat time in order to simulate the time-critical nature of emergency response. The dataset contains conversation data for two planning domains, including the final plans manually reported by each team. These ground truth plans were used to evaluate the plan inference model.

In addition to the pre-existing dataset, we collected our human team planning dataset from two additional domains: (1) UAV mission planning and (2) missile defense. In the UAV mission planning scenario, teams had to generate plans to maximize the number of task completions (e.g. package delivery, surveillance, and enemy strike) using a given amount of fuel. In the missile defense domain, teams solved a task assignment problem involving a set of countermeasures to incoming missiles in order to maximize the survival of ships. The design was inspired by the gaming scenarios used in existing military applications [Gombolay *et al.*, 2017]. Similar to the data collection design by Kim and Shah, teams of two participants each were supplied with identical scenario information and conversed via web chat. The planning sessions were restricted to 20 minutes, and the final plans were reported upon completion of the sessions.

For our test domains, we clarify that goals in terms of scenario objectives of "save patient(s)", "maximize number of subtasks" or "maximize ship survival" are known to the IDS agent. However, it is not a fully specified goal in terms of identifying who to save, what subtasks to fulfill, or what missiles to assign. It is the job of the agent to uncover those human priorities and preferences through evidence in the team's planning conversation.

While two emergency response domains by Kim and Shah were modeled as propositional planning problems, our UAV and missile defense domains are modeled as numerical planning and discrete optimization (i.e., mixed integer program) problems, respectively. Our plan inference model can be readily applied to planning problems of varying complexity (propositional vs. numerical) and varying solution techniques (i.e., generative planners vs. mixed-integer program solvers).

Domain	Optim.	N_{doc}	N_{utter}	$NoiseR$	$MissR$
emergency1	Prop. Plan	57	101	0.46	0.49
emergency2	Prop. Plan	58	82	0.23	0.50
UAV	Num. Plan	22	90	0.34	0.43
Missile	IP	23	48	0.39	0.05

Table 2: A summary of the team planning dataset used in our evaluation. N_{doc} is the number of documents and N_{utter} is the average utterance count within each document. $NoiseR$ and $MissR$ are the average ratios of noisy and missing observations, respectively.

Given information about the dynamics of the planning environment and the ability to estimate the cost of a candidate plan, our inference model can be readily implemented.

The preprocessed form of input requires annotations of parsed actions, ϕ_2: DAs, and ϕ_3: APs. These were provided by two annotators with initial inter-rater agreements of 0.81, 0.75, and 0.83. Afterwards, annotators were asked to resolve their differences and produce a final set of annotations. The summary of the full human planning dataset is provided in Table 2. For each domain, we analyzed the ratios of noisy and missing observations, which ranged from 0.23-0.46 and 0.05-0.49, respectively.

5.2 Test Setup

Evaluation metrics: To evaluate plan inference performance, we used the following plan similarity metrics. Recall represents the ratio of actions correctly recovered from the true final plan, $|\pi_{pred} \cap \pi_{true}|/|\pi_{true}|$. Precision represents the ratio of predicted actions that appear in the final plan, $|\pi_{pred} \cap \pi_{true}|/|\pi_{pred}|$. F1-score is the harmonic mean between recall and precision, the primary plan similarity metric that we sought to maximize. We also investigated P_{valid}, the ratio of valid plans out of all predictions.

Baselines: We compared our model against the following planning baselines: 1) $AllObs$, which outputs a plan navigating through all observations, 2) $NoObs$, which disregards all observations and outputs the agent's plan, 3) the generative model proposed by Kim et al. [2015], which was designed to monitor human team planning, and 4) the planning approach described by Sohrabi et al. [2016], which was designed to handle noisy observations.

Experimental details[2]: All evaluation metrics were averaged over 25 independent runs. Each of these trials used T=500 and α of 0.005. Our policy gradient algorithm uses the standard ϵ-greedy exploration, with ϵ decay rate of $1/(10 + t)$. We normalized all plan costs with respect to the expected average plan cost for each domain. λ was set by maximizing F1-score using 10% of the documents as a validation set. For each low-level planner call, search time was limited to 20 sec.

6 Results and Discussion

Plan prediction performance: Table 3 depicts the average F1-scores (over 25 runs) of our model compared with the baselines. $NoObs$ performed poorly where π_{agent} disregarded all observed actions. Employing alternative off-the-shelf planners may yield different cost-optimal plans that, by chance, align more similarly to π_{true}. However, average similarity over a diverse set of team plans would likely be low. $AllObs$ resulted in higher F1-scores for the emergency response domains, but not for the UAV and missile domains. Emergency response domains did not feature any hard resource constraints, which generally allowed $AllObs$ the flexibility necessary to produce valid plans while incorporating all observed actions ($P_{valid} = 0.86$). The UAV and missile domains contained more-complicated sets of numerical and resource constraints, such that incorporating all observations often rendered the planning problem unsolvable ($P_{valid} = 0.13$).

Model	Emerg.1	Emerg.2	UAV	missile
B1: $NoObs$	0.09	0.09	0.37	0.06
B2: $AllObs$	0.50	0.56	0.12	0.10
[Kim *et al.*, 2015]	0.37	0.44	0.42	0.47
[Sohrabi *et al.*, 2016] - Sequencing Enforced	0.39	0.52	n/a	n/a
[Sohrabi *et al.*, 2016] - Sequencing Relaxed	0.54	0.70	n/a	n/a
Our Model	**0.55**	**0.72**	**0.47**	**0.56**
SVM-linear	0.57	0.73	0.47	0.59
LOOCV	**0.58**	**0.73**	**0.47**	**0.48**

Table 3: A comparison of F1-scores across various techniques.

While the generative model proposed by Kim et al. [2015] outperformed $NoObs$, it demonstrated mediocre overall F1-scores, primarily due to its inefficiency in sampling a diverse set of valid plans ($P_{valid} = 0.02$). Since plans are formed by randomly inserting/removing actions, the model can get stagnant iterating over the perturbations of an invalid plan or fail to traverse to a different valid plan if one has been discovered. The runtime and memory use of the approach also preclude its use in online applications, for the moderately-sized problems evaluated in this work. Runtime grows as a function of the number of Gibbs samples and the time for plan validation. Average runtime at test, using the parameters reported in Kim et al. (2,000 Gibbs samples), was 1,970 sec. In contrast, our model required a single low-level planner call during test that took on average about 0.2 sec. Whereas our model stored feature weights, θ, Kim et al.'s model required storing a number of partial plans that scaled in the number of Gibbs samples, and was significantly greater than $|\theta|$.

Our plan inference model outperformed all aforementioned techniques, with an increase in F1-scores observed across all four domains. With our model, predicted plans are always valid ($P_{valid} = 1$). Each learning episode of our model lasted about 5.2 sec. Curves in Figure 3 demonstrates how our model was able to exceed Kim et al.'s max F1-score in less than 10 training episodes.

Comparison to a supervised baseline: We compared our model (reinforcement learning) against a supervised baseline that incorporates ground truth plans. For the supervised

[2] All tests were performed using an Intel(R) Xeon(R) E5-2630 v4 processor (2.20 GHz, 24 cores) with 48 GB RAM.

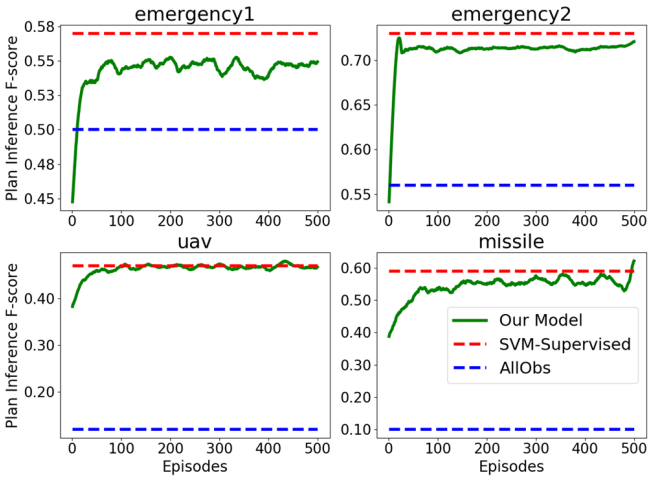

Figure 3: Plan inference F1-score of our model with respect to the number of training episodes.

Our Model	SVM-linear
ϕ_1: **Action counts (+)**	ϕ_1: **Action counts (+)**
ϕ_2: **DA = Inform (-)**	ϕ_2: DA = Transition (+)
ϕ_7: Action inconsistency (-)	ϕ_5: Action infeasibility (-)
ϕ_2: DA = Ques. - Suggest (-)	ϕ_4: **Relative position (+)**
ϕ_4: **Relative position (+)**	ϕ_2: **DA = Inform (-)**

Table 4: The top five ranked features (in descending order) learned by our model and by the SVM-linear model (overlaps in **bold**).

method, we used a linear SVM model as our model is likewise linear in the features. The action selection policy was replaced by a linear SVM classifier using our same feature set $\phi(d, a_i)$. Figure 3 shows that our model achieved performance on par with that of a supervised SVM-linear model. The supervised approach would be favorable when a database of plan traces or post-meeting summaries are made available; our analysis indicated a slight increase in F1-score and an improved runtime, with training and testing requiring less than 0.1 sec. However, our model is more advantageous than the supervised approach in that it does not require a labeled dataset of ground truth plans. Feature analysis in Table 4 indicates overlapping ranked features and consistency with regard to their signs between the two models. These empirical results support our hypothesis that the overall plan cost function (defined in Equation 3) can serve as a powerful source of supervision when learning to infer teams' final plans.

Generalization: We performed leave-one-out cross validation (LOOCV) across four domains to determine whether a learned model can infer plans within novel domains. For each LOOCV fold, the model was trained on a combined set of documents from three domains and tested on documents from the remaining domain. Average test F1-scores (depicted in the last row of Table 3) deviated only slightly compared with our original model (for the emergency1 domain, prediction performance slightly improved). These results support the utility of employing our domain-independent feature set for plan inference within novel domains.

Comparison to [Sohrabi *et al.*, 2016]: The plan recognition (PR) as planning approach by Sohrabi et al. transformed planning problem to incorporate additional "explain" and "discard" operators, and modified the overall plan cost to incorporate penalties for discarding observations. A cost-optimal planner was then utilized to produce the a set of most likely plans with respect to the overall cost function defined in Equation 3. The technique was designed for PR via goal recognition, utilizing a top-k planner [Katz *et al.*, 2018] to obtain a probability distribution over the candidate goals and their corresponding plans. In our PR setting, we instead specify a single abstract goal condition for each test domain (e.g. a mission objective of "saving one or more patient")[3] and direct the IDS agent to output a single predicted plan.

In order to make a head-to-head comparison, we implemented Sohrabi et al.'s approach using the same hyperparameter, λ, and the same low-level planner, LAMA, as used in our model. The approach presented by Sohrabi et al. differed from our approach in that it required the produced plans to reproduce of the sequencing of observations. In human team planning, however, actions are not necessarily uttered in the order of their planned execution. We observed that the sequencing constraint often rendered the modified planning problem unsolvable or led to the generation of convoluted plans with excess plan length. As shown in Table 3, this resulted in mediocre F1-scores for the Sohrabi et al. approach. To provide a fairer basis of comparison, we also implemented a variant of the approach that relaxed the sequencing constraint. The modified approach produced results that were comparable to our model for the STRIPS domains.

The modified Sohrabi et al. approach is, by design, a planning approach tailored to STRIPS-style planning problems. Our approach, in contrast, learns an action selection policy that is independent of the underlying representation of the planning problem, and therefore our plan inference model can be readily applied to planning problems of varying complexity, including numerical planning and mixed integer programs.

While pure planning-based approaches are directly impacted by the incomplete PDDL specifications, our learning model incorporates a variety of dialogue and planning features (e.g. action feasibility, action similarity) that capture additional information about the most likely plan from the team conversation, thereby providing robustness to incomplete model specifications. We compared plan prediction performance with incomplete PDDL files, generated by randomly removing a portion of original preconditions, effects, and object lists. The average F1-scores (over 25 independent runs), shown in Table 5, indicated that our model was on average relatively more robust to degraded PDDL specifications, than the modified Sohrabi et al. planning approach. The trend was observed across both propositional planning domains and for all variants of degraded PDDL tested.

[3] Pre-enumerating all possible goal states that meet the condition can be intractable for a subset of our mission planning domains.

		[Sohrabi *et al.*, 2016]		**Our Model**	
Condition	M	E1	E2	E1	E2
1) Pre/Effects	10%	0.47	0.50	0.51	0.55
2) Pre/Effects	20%	0.37	0.40	0.41	0.42
3) Objects	5%	0.28	0.48	0.36	0.51

Table 5: Comparison of models (F1-scores) in degraded PDDL specifications. Conditions 1) and 2) randomly remove preconditions and effects in the domain file, and condition 3) removes objects declared in the problem file. M represents the percentage of parameters removed from the original files. E1 and E2 stand for the two emergency response planning domains.

7 Conclusion and Future Work

In this paper, we presented a novel technique for inferring teams' final plans through a processed form of their planning conversations. Our plan inference model utilizes features from teams' dialogue patterns and planning domain models to efficiently search the large space of possible plans. While using planning feedback as its only source of supervision, our model achieved performance on par with that of a supervised baseline. We empirically demonstrated that our model learns to infer plans in novel domains, can be readily applied to planning problems of varying complexity, and that it is less sensitive to degraded model specifications than prior art.

Our aim is to eventually apply plan inference techniques on an IDS agent that can actively support human team planning. Integration of automatic parsing and dialogue act tagging tools would enable the development of an online plan inference agent. We also note the importance of exploring the effect of social roles and more complicated interactions during human team planning. Different types of expertise, authority, and group sizes represent possible additional indicator features to include and analyze in the plan inference model. Lastly, we seek to develop an active plan inference agent designed to query information from human teammates to better infer and support their plans.

References

[Amir and Gal, 2013] Ofra Amir and Ya'akov Kobi Gal. Plan recognition and visualization in exploratory learning environments. *ACM Trans. Interactive Intelligent Systems*, 3(3):16, 2013.

[Artzi and Zettlemoyer, 2013] Yoav Artzi and Luke Zettlemoyer. Weakly supervised learning of semantic parsers for mapping instructions to actions. *Trans. ACL*, 1:49–62, 2013.

[Banerjee *et al.*, 2010] Bikramjit Banerjee, Landon Kraemer, and Jeremy Lyle. Multi-agent plan recognition: formalization and algorithms. In *AAAI*, pages 1059–1064, 2010.

[Branavan *et al.*, 2012] SRK Branavan, Nate Kushman, Tao Lei, and Regina Barzilay. Learning high-level planning from text. In *ACL*, pages 126–135, 2012.

[Bylander, 1991] Tom Bylander. Complexity results for planning. In *IJCAI*, pages 274–279, 1991.

[Carberry, 1990] Sandra Carberry. *Plan Recognition in Natural Language Dialogue*. MIT Press, 1990.

[Carberry, 2001] Sandra Carberry. Techniques for plan recognition. *User Modeling and User-Adapted Interaction*, 11(1):31–48, 2001.

[Carletta, 2007] Jean Carletta. Unleashing the killer corpus: experiences in creating the multi-everything AMI meeting corpus. *Language Resources and Evaluation*, 41(2):181–190, 2007.

[Casper and Murphy, 2003] Jennifer Casper and Robin R. Murphy. Human-robot interactions during the robot-assisted urban search and rescue response at the world trade center. *IEEE Trans. SMC, Part B*, 33(3):367–385, 2003.

[Di Eugenio *et al.*, 2000] Barbara Di Eugenio, Pamela W Jordan, Richmond H Thomason, and Johanna D Moore. The agreement process: An empirical investigation of human–human computer-mediated collaborative dialogs. *Int. Jour. Hum.-Com. Studies*, 53(6):1017–1076, 2000.

[Freedman *et al.*, 2014] Richard G Freedman, Hee-Tae Jung, and Shlomo Zilberstein. Plan and activity recognition from a topic modeling perspective. In *ICAPS*, pages 360–364, 2014.

[Gerevini and Long, 2005] Alfonso Gerevini and Derek Long. Plan constraints and preferences in PDDL3. *Fifth IPC. Tech. Rep.*, 2005.

[Gombolay *et al.*, 2017] Matthew Gombolay, Reed Jensen, and Sung-Hyun Song. Machine learning techniques for analyzing training behavior in serious gaming. *IEEE Trans. on Computational Intelligence and AI in Games*, 2017.

[Gupta and Lehal, 2010] Vishal Gupta and Gurpreet Singh Lehal. A survey of text summarization extractive techniques. *Jour. Emerg. Tech. in Web Intelligence*, 2(3):258–268, 2010.

[Hoffmann *et al.*, 2004] Jörg Hoffmann, Julie Porteous, and Laura Sebastia. Ordered landmarks in planning. *JAIR*, 22:215–278, 2004.

[Hoffmann, 2003] Jörg Hoffmann. The Metric-FF planning system: Translating "ignoring delete lists" to numeric state variables. *JAIR*, 20:291–341, 2003.

[Kaminka *et al.*, 2002] Gal A Kaminka, David V Pynadath, and Milind Tambe. Monitoring teams by overhearing: A multi-agent plan-recognition approach. *JAIR*, 17(1):83–135, 2002.

[Katz *et al.*, 2018] Michael Katz, Shirin Sohrabi, Octavian Udrea, and Dominik Winterer. A novel iterative approach to top-k planning. In *ICAPS*, 2018.

[Kim and Shah, 2016] Joseph Kim and Julie A Shah. Improving team's consistency of understanding in meetings. *IEEE Trans. Human-Machine Systems*, 46(5):625–637, 2016.

[Kim *et al.*, 2015] Been Kim, Caleb M Chacha, and Julie A Shah. Inferring team task plans from human meetings: A generative modeling approach with logic-based prior. *JAIR*, 52:361–398, 2015.

[Lafferty *et al.*, 2001] John Lafferty, Andrew McCallum, and Fernando C.N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, pages 282–289, 2001.

[Larson *et al.*, 1996] James R Larson, Caryn Christensen, Ann S Abbott, and Timothy M Franz. Diagnosing groups: charting the flow of information in medical decision-making teams. *Jour. Personality and Social Psychology*, 71(2):315, 1996.

[Miller, 2004] C. Miller. Modeling human workload limitations on multiple UAV control. In *Human Factors*, pages 526–527, 2004.

[Pereira *et al.*, 2017] Ramon Fraga Pereira, Nir Oren, and Felipe Meneguzzi. Landmark-based heuristics for goal recognition. In *AAAI*, pages 3622 – 3628, 2017.

[Qin *et al.*, 2017] Kechen Qin, Lu Wang, and Joseph Kim. Joint modeling of content and discourse relations in dialogues. In *ACL*, pages 974–984, 2017.

[Ramırez and Geffner, 2009] Miquel Ramırez and Hector Geffner. Plan recognition as planning. In *IJCAI*, pages 1778–1783, 2009.

[Richter and Westphal, 2010] Silvia Richter and Matthias Westphal. The LAMA planner: Guiding cost-based anytime planning with landmarks. *JAIR*, 39:127–177, 2010.

[Sohrabi *et al.*, 2016] Shirin Sohrabi, Anton V Riabov, and Octavian Udrea. Plan recognition as planning revisited. In *IJCAI*, pages 3258–3264, 2016.

[Somasundaran and Wiebe, 2007] Swapna Somasundaran and Janyce Wiebe. Detecting arguing and sentiment in meetings. In *SIGdial Workshop on Discourse and Dialogue*, 2007.

[Sukthankar and Sycara, 2006] Gita Sukthankar and Katia Sycara. Simultaneous team assignment and behavior recognition from spatio-temporal agent traces. In *AAAI*, 2006.

[Sutton *et al.*, 2000] Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *NIPS*, pages 1057–1063, 2000.

[Vered *et al.*, 2018] Mor Vered, Ramon Fraga Pereira, Mauricio Magnaguagno, Felipe Meneguzzi, and Gal A. Kaminka. Towards online goal recognition combining goal mirroring and landmarks. In *AAMAS*, 2018.

[Zhuo and Kambhampati, 2013] Hankz Hankui Zhuo and Subbarao Kambhampati. Action-model acquisition from noisy plan traces. In *IJCAI*, pages 2444–2450, 2013.

[Zhuo and Li, 2011] Hankz Hankui Zhuo and Lei Li. Multi-agent plan recognition with partial team traces and plan libraries. In *IJCAI*, pages 484–489, 2011.

[Zhuo *et al.*, 2012] Hankz Hankui Zhuo, Qiang Yang, and Subbarao Kambhampati. Action-model based multi-agent plan recognition. In *NIPS*, pages 368–376, 2012.

[Zhuo *et al.*, 2013] Hankz Hankui Zhuo, Tuan Anh Nguyen, and Subbarao Kambhampati. Model-lite case-based planning. In *AAAI*, pages 1077–1083, 2013.