Plan Explainability and Predictability for Cobots

Yu Zhang
Arizona State University
Arizona, USA
yzhan442@asu.edu

Hankz Hankui Zhuo
Arizona State University
Arizona, USA
hankz@asu.edu

Subbarao Kambhampati
Arizona State University
Arizona, USA
rao@asu.edu

arXiv:1511.08158v1 [cs.AI] 25 Nov 2015

Abstract

Robots are becoming pervasive in human populated environments. A desirable capability of these robots (cobots) is to respond to goal-oriented commands by autonomously constructing plans. However, such autonomy can add significant cognitive load and even potentially introduce safety risks to the humans when robots choose their plans unexpectedly. As a result, for cobots to be more helpful, one important requirement is for them to synthesize plans that do not *surprise* the humans. While there are previous works that studied socially acceptable robots which discuss "natural ways" for cobots to interact with humans, there still lacks a general solution, especially for cobots that can construct their own plans. In this paper, we introduce the notions of plan *explainability* and *predictability*. To compute these measures, first, we postulate that humans understand robot plans by associating high level tasks with robot actions, which can be considered as a labeling process. We learn the labeling scheme of humans for robot plans from training examples using conditional random fields (CRFs). Then, we use the learned model to label a new plan to compute its explainability and predictability. These measures can be used by cobots to proactively choose plans, or directly incorporated into the planning process to generate plans that are more explainable and predictable. We provide an evaluation on a synthetic dataset to demonstrate the effectiveness of our approach.

Introduction

Robots are becoming pervasive in human populated environments. Examples include robots for elderly care, housekeeping, search and rescue, just to name a few. To make such robots (cobots) more capable, significant research efforts have been invested in providing more autonomy to cobots. A desirable capability is to enable cobots to respond to goal-oriented commands instead of basic motor commands (e.g., in teleoperation). This capability requires cobots to autonomously synthesize plans for goals and execute plans to achieve them. However, such autonomy can increase cognitive load and even potentially introduce safety risks to humans when the robots choose their plans unexpectedly.

As a result, one important requirement for cobots is to ensure that the synthesized plans do not surprise the humans. This means that instead of synthesizing plans based solely on the model of the robots (which allows cobots to synthesize a plan to achieve a given goal), this synthesis must also consider the interpretation of the robot model from the human's perspective. This interpretation can capture, for example, the human's understanding of the robot model (potentially partial and inaccurate) as well as the human's preferences. For example, to darken a room where there is too much light, a robot can choose to either adjust the window blinds, switch off the lights, or break the light bulbs in the room. While breaking the light bulbs may well be the least costly plan to the robot under certain conditions (e.g., when the robot cannot easily move in the environment), it is clear that the other two options are more desirable in the context of cobots. One of the challenges here is that how the human interprets the robot's model (which determines how the robot chooses its plan) is inherently hidden. Furthermore, this interpretation can be arbitrarily different from the robot's own model. While there are previous works that studied socially acceptable robots (Hoffman and Breazeal 2007a; Hoffman and Breazeal 2007b; Shah et al. 2011; Narayanan et al. 2015), which discuss "natural ways" for cobots to interact with humans, there lacks a general solution, especially for cobots that can construct their own plans.

In this paper, we introduce the notions of plan *explainability* and *predictability*. Our problem settings are as follows: a robot is given a goal by a human working in the same environment, and it needs to synthesize a plan to achieve the goal. Given that most humans only have a limited understanding of robot model, they would naturally interpret a plan as achieving high level tasks or subgoals, which are the functional interpretations of sequences of robot actions in the plan. For example, a robot that executes a sequence of motor actions to pick up a cup may be interpreted as the task of "$pick\ up\ cup$". Intuitively, given any prefix of a plan (as a sub-plan), the more explainable the plan is, the easier it is (for humans) to associate tasks to actions in this sub-plan; similarly, the more predictable the plan is, the easier it is to predict the next task to be achieved in the full plan. In this regard, explainability concerns the association between high level human-interpreted tasks and robot actions, while predictability concerns the connections between these tasks. One observation is that plan predictability is partially relying on plan explainability. Intuitively, it is difficult to predict the next task of the robot if we do not even understand what it is currently trying to achieve.

Since the association between tasks and robot actions can

be considered as a labeling process, we learn the labeling scheme of humans for robot plans from training examples using conditional random fields (CRFs). We then use the learned model to label a new plan to compute its explainability and predictability. These measures can be used by cobots to proactively choose plans, or directly incorporated into the planning process to generate plans that are more explainable and predictable without affecting the quality too much. Since we use a learning method, our approach does not assume any prior knowledge on the representation of the human's interpretation of the robot model. The learned CRF can keep evolving through human feedbacks. We further provide an evaluation on a synthetic dataset to demonstrate the effectiveness of our approach.

Related Work

There exists previous work on building autonomous cobots. One important capability is for cobots to respond to goal-oriented commands via automated planning. A planning capability allows cobots to autonomously synthesize a plan to achieve a goal instead of following basic motor commands, thus significantly reducing the human's cognitive load. Meanwhile, for cobots to work along side of humans, they must also be aware of the humans while synthesizing their plans. Previous work has considered human-aware planning (Sisbot et al. 2007; Cirillo, Karlsson, and Saffiotti 2009; Chakraborti et al. 2015) in which cobots take the human's activities into account while constructing their own plans. A prerequisite to achieve this capability in human-aware planning is a plan recognition component, which is used to infer the human goals and plans. This information can then be used by cobots to avoid interference, and plan for serendipity and teaming with humans. There exists a rich literature on plan recognition (Kautz and Allen 1986; Charniak and Goldman 1993; Ramírez and Geffner 2010; Levine and Williams 2014). Many recent works propose how to use these techniques for human-aware planning and human-robot teaming (Talamadupula et al. 2014; Chakraborti et al. 2015; Zhang et al. 2015).

While our work on plan explainability and predictability falls within the scope of human-aware planning, it also differs significantly from the previous work. First, in our context, the human knows the goal of the robot. This distinguishes our work from goal recognition. Furthermore, compared to previous work on human-aware planning in general, there, it is assumed that the robot (R) has a model of the human (H). While this model is arguably obtainable, for example, via prior knowledge, communication or learning while assuming a specific model representation (Zhuo and Kambhampati 2013), it is difficult to argue the same in our context. The model here is one level deeper: it is about the interpretation of the robot model from the human's perspective. In other words, R needs to understand the model of *itself* in H's eyes. This information is inherently hidden, difficult to convey, and can be arbitrarily different (e.g., having different model representation) from R's own model.

In the human-robot interaction community, there exists prior works that discuss how to enable natural and fluent human-robot interaction (Hoffman and Breazeal 2007a; Hoffman and Breazeal 2007b; Shah et al. 2011; Narayanan et al. 2015) to create more socially acceptable robots (Fong, Nourbakhsh, and Dautenhahn 2003). These works, however, apply only to specific behaviors and application domains, and thus cannot be generalized. While our approach also has connections to methods for model learning via expert teaching, such as inverse reinforcement (Abbeel and Ng 2004) and tutoring systems (Murray 1999), there are no experts (or teachers) involved in our context. More specifically, although the robot knows more than the human at the low level, it needs to consider the human's high level "preferences". In this sense, our work is also connected to preferred grammar in HTN (Li et al. 2014). There are also works on excuse (Gbelbecker et al. 2010) and explanation generation (Hanheide et al. 2015) which emphasize on explaining planning failures. Finally, while our learning approach appears to be similar to information extraction (Peng and McCallum 2006), we use the learned model to proactively guide planning instead of passively extracting information.

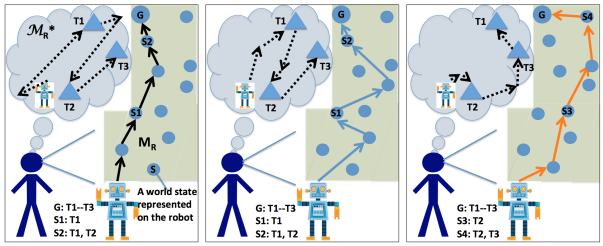


Figure 1: General problem settings. The left part shows that a robot plan based solely on M_R has parts that are unexplainable and unpredictable from the human's perspective (i.e., in $\mathcal{M}_R^*$). The middle part shows a plan that is more explainable but still not very predictable. The right part shows a plan that is both explainable and predictable. The human's interpretation ($\mathcal{M}_R^*$) is assumed to be associated with tasks (i.e., $T_1 - T_3$). The robot model (M_R) is assumed to be specified as transitions between states. The relationships between tasks and states are shown at the bottom. The human interprets the robot goal G as achieving the three tasks $T_1 - T_3$.

Explainability and Predictability

In our settings, a robot R needs to achieve a goal given by a human in the same environment. The robot has a model of itself (referred to as M_R), which is used to autonomously construct plans to achieve goals. In this paper, we assume that this model is based on PDDL (Fox and Long 2003), which is a general planning domain definition language. As we discussed, for a cobot to generate explainable and predictable plans, it must not only consider M_R but also $\mathcal{M}_R^*$, which is the interpretation of M_R from the human's perspective. Next, we first present a general view of the problem and then discuss our solution.

Problem Formulation

The general problem settings are shown in Fig. 1, in which we present $\mathcal{M}_R^*$ and M_R graphically. $\mathcal{M}_R^*$ is presented as

thought clouds and M_R is presented in the polygonal area. Given a domain, the problem is to find a plan for a given problem that satisfies the following:

$$\underset{\pi_{M_R}}{\operatorname{argmin}}\, cost(\pi_{M_R}) + \alpha \cdot dist(\pi_{M_R}, \pi_{\mathcal{M}_R^*}) \tag{1}$$

where π_{M_R} is a plan that is constructed using M_R (i.e., the plan constructed by the robot), $\pi_{\mathcal{M}_R^*}$ is a plan that is constructed using $\mathcal{M}_R^*$ (i.e., human's anticipation of the robot's plan), $cost$ returns the cost of a plan, $dist$ returns the distance between two plans, and α is the relative weight. The goal of Eq. (1) is to find a plan that minimizes a weighted sum of the cost of the robot plan and the distance (i.e., difference) between the two plans. Since the robot model M_R is often given, the challenge is on the second part in Eq. (1).

An observation is that when $\mathcal{M}_R^*$ and $dist$ are provided, we can solve Eq. (1). However, as discussed previously, $\mathcal{M}_R^*$ is inherently hidden, difficult to convey, and can be arbitrarily different from M_R. Hence, we use a learning method to directly learn the $dist$ function. We postulate that humans understand robot plans by associating tasks with actions, which can be considered as a labeling process. Thus, we assume that $dist$ can be functionally decomposed as:

$$dist(\pi_{M_R}, \pi_{\mathcal{M}_R^*}) = F \circ \mathcal{L}^*(\pi_{M_R}) \tag{2}$$

where F is a domain specific function that takes plan labels as input, and $\mathcal{L}^*$ is the labeling scheme of the human for robot plans based on $\mathcal{M}_R^*$. As a result, Eq. (1) now becomes:

$$\underset{\pi_{M_R}}{\operatorname{argmin}}\, cost(\pi_{M_R}) + \alpha \cdot F \circ \mathcal{L}^*_{CRF}(\pi_{M_R} | \{S_i | S_i = \mathcal{L}^*(\pi^i_{M_R})\}) \tag{3}$$

where $\{S_i\}$ is the set of training examples and $\mathcal{L}^*_{CRF}$ is the learned model of $\mathcal{L}^*$. We can now formally define plan explainability and predictability in our context. Given a plan of robot R (i.e., π_{M_R}, simplified below as π for clarity) as a sequence of actions:

$$\pi = \langle a_0, a_1, a_2, ...a_N \rangle \tag{4}$$

where a_0 is a null action that denotes plan starting. Given the domain, we assume that a set of task labels T is provided to label robot actions:

$$T = \{T_1, T_2, ...T_M\} \tag{5}$$

Explainability Labeling: Explainability concerns the association between tasks and robot actions. Each action in a plan is associated with an action label. The set of action labels for explainability is the power set of the task labels:

$$L = 2^T \tag{6}$$

When an action label includes multiple task labels, the action may be interpreted as contributing to multiple tasks; when an action label is the empty set, the action may be interpreted as unexplainable. When a plan is labeled, we can compute its explainability measure based on its action labels in a domain specific way. More specifically, we define:

Definition 1 (Plan explainability) *Given a domain, the explainability θ_π of a robot plan π is computed by a mapping, $F_\theta : \boldsymbol{L}_\pi \rightarrow [0, 1]$ (with 1 being the most explainable).*

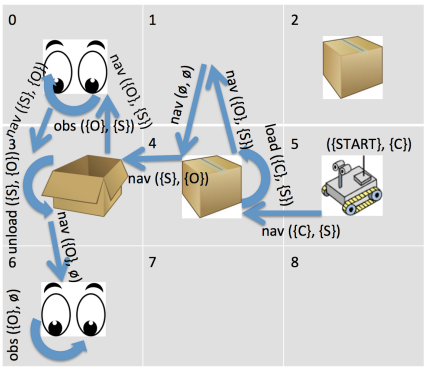


Figure 2: Example for plan explainability and predictability.

$\mathbf{L}_\pi$ above denotes the sequence of action labels for π. An example of F_θ used in our evaluation is given below:

$$F_\theta(\mathbf{L}_\pi) = \frac{\sum_{i \in [1,N]} \mathbf{1}_{L(a_i) \neq \emptyset}}{N} \tag{7}$$

where N is the plan length, $L(a_i)$ returns the action label of a_i, and $\mathbf{1}_{formula}$ is an indicator function that returns 1 when the $formula$ holds or 0 otherwise.

Predictability Labeling: Predictability concerns the connections between tasks in a plan. An action label for predictability is composed of two parts: a current label and a next label (i.e., $L \times L$). The current label is also the action label for explainability. The next label is used to specify the tasks that are anticipated to be achieved next in the plan. A next label with multiple task labels may be interpreted as the possible candidate tasks to be achieved next; when this label is the empty set, it may be interpreted as that the next task is unpredictable, or that there are no more tasks to be achieved.

Definition 2 (Plan Predictability) *Given a domain, the predictability β_π of a plan π is computed by a mapping, $F_\beta : \boldsymbol{L}^2_\pi \rightarrow [0, 1]$ (with 1 being the most predictable).*

$\mathbf{L}^2_\pi$ denotes the sequence of action labels for predictability. An example of F_β is given below which is used in our evaluation when assuming that the current and next labels are associated with at most one task label:

$$F_\beta(\mathbf{L}^2_\pi) = \frac{\sum_{i \in [0,N]} \mathbf{1}_{|L(a_i)|=1} \wedge (\mathbf{1}_{L^2(a_i)=L(a_j)} \vee \mathbf{1}_{L^2(a_{i:N})=\emptyset})}{N+1} \tag{8}$$

where $a_j (j > i)$ is the first action that has a different current label as a_i or the last action in the plan if unfound, $L^2(a_i)$ returns the next label of a_i and $\mathbf{1}_{L^2(a_{i:N})=\emptyset}$ returns 1 only if the next labels for all actions after a_i (including a_i) are $\emptyset$.

A Concrete Example

Before we discuss how to learn the labeling scheme of the human from training examples, we provide a concrete example to connect the previous concepts and how training examples may be obtained. In this example, there is a rover in a grid environment working remotely with a human. An illustration of this example is presented in Fig. 2. There are

resources to be collected, which are represented as boxes. There is one storage area that can store one resource, which is represented as an open box. The rover can also make observations at locations. The rover actions include $\{navigate\ l_1\ l_2\}$, $\{observe\ l\}$ $\{load\ l\}$, and $\{unload\ l\}$, each representing a set of actions. Each l ($l \in \{0-8\}$) with or without a subscript denotes a location. $navigate$ (or nav) can move the rover from a location to one of its adjacent locations; $load$ can be used to pick up a resource when the rover is not already loaded; $unload$ can be used to unload a resource at a storage area if the area is empty; $observe$ (or obs) can be used to make an observation. Once a location is observed, it remains observed. The goal in this example is for the rover to make the storage area non-empty and observe two locations that contain the eye symbol.

In this example, we assume that there are three tasks that may be used by the human to interpret the rover's plans, which are COLLECT (C), STORE (S) and OBSERVE (O). Note that we do not specify any arguments for these tasks (e.g., which resource the rover is collecting) since this information may not be specifically captured by humans. This also illustrates that M_R and $\mathcal{M}_R^*$ can be arbitrarily different. In Fig. 2, we present a plan of the rover as connected arrows starting from the its initial location.

Human Interpretation as Training Examples: Let us now discuss how humans may interpret this plan (i.e., associating labels with actions) as the actions are observed *incrementally*: when labeling a_i, we only have access to the plan prefix $\langle a_0, ..., a_i \rangle$. At the beginning (labeling for a_0), the rover starts at l_5. Given knowledge of the environment and the rover's goal, we may infer that the first task should be COLLECT (the resource from l_4). Hence, we may choose to label a_0 as ($\{$START$\}$, $\{$C$\}$). The first action of the rover (i.e., $nav\ l_5\ l_4$) seems to match with our prediction. Furthermore, given that the storage area is closest to the rover's location after completing COLLECT, the next task is likely to be STORE. Hence, we may label a_1 as ($\{$C$\}$, $\{$S$\}$) as shown in the figure. The second action (i.e., $load\ l_4$) also matches with our expectation. Hence, we label a_2 too as ($\{$C$\}$, $\{$S$\}$). The third action, $nav\ l_4\ l_1$, however, is unpredictable since we are expecting STORE from the previous steps. Nevertheless, we can still explain it as contributing to OBSERVE (at location l_0). Hence, we may label this navigation action (a_3) as ($\{$O$\}$, $\{$S$\}$). For the fourth action, the rover moves back to l_4, which is somewhat unexplainable since the rover's behavior seems to be oscillating without reasons. Hence, we may choose to label this action as ($\emptyset$, $\emptyset$). The labeling of the rest of the plan happens in a similar manner. This thought process reflects how training examples are provided by human labelers. Note that when humans are to label a plan in real-world applications, they may also be given access to the actual plan execution by the robot (e.g., as video streams).

Learning Approach

To compute θ_π and β_π from Defs. (1) and (2) for a given plan π, the challenge is to provide a label for each action. This requires us to learn the labeling scheme of humans (i.e., $\mathcal{L}^*$ in Eq. (2)) from training examples and then apply the learned model to π (i.e., $\mathcal{L}^*_{CRF}$ in Eq. (3)). To formulate a learning method, we consider the sequence of labels as hidden variables. The plan that is executed by the robot constitutes the observations. The graphical model that we choose for the learning method is conditional random fields (CRFs) (Lafferty, McCallum, and Pereira 2001). The distributions that are captured by CRFs have the following form:

$$p(\mathbf{x}, \mathbf{y}) = \frac{1}{Z} \Pi_A \Phi(\mathbf{x}_A, \mathbf{y}_A) \tag{9}$$

in which Z is a normalization factor that satisfies:

$$Z = \sum_{\mathbf{x},\mathbf{y}} \Pi_A \Phi(\mathbf{x}_A, \mathbf{y}_A) \tag{10}$$

In the equations above, $\mathbf{x}$ represents the sequence of observations, $\mathbf{y}$ represents the sequence of hidden variables. $\Phi(\mathbf{x}_A, \mathbf{y}_A)$ represents a factor that is related to a subgraph in the CRF model associated with variables $\mathbf{x}_A$ and $\mathbf{y}_A$. In our context, $\mathbf{x}$ are the observations made during the execution of the action sequence of a plan; $\mathbf{y}$ are the action labels. Each factor is associated with a set of features observed during the plan execution. Next, we discuss the features that can be used for plan explainability and predictability.

Features for Learning

Given a robot plan, the immediate set of features that we have access to is from the robot model, which captures the state changes as each action in the plan is executed. Although the human may not be able to understand the complete specification of these changes as in the robot model, these features can capture environment changes that humans can interpret. Furthermore, when the dynamics of the robot is known, we can derive the sequence of primitive motor commands that implement the robot motion given the plan. This information can be used to extract motion related features. Moreover, when the robot is executing actions, it may also have various sensor information available. For example, this may include onboard video streams and skeleton tracking information for the human. Sensor information can be recorded to extract features for training examples. During plan synthesis (i.e., testing phase), since our goal is to generate explainable and predictable plans proactively before they are executed, sensor information will be missing. However, the associated features may still be estimated (e.g., distances to the human). In this work, we use a linear chain CRF. However, our formulation is easily extensible to more general types of CRFs. Given a robot plan $\pi = \langle a_0, a_1, a_2, ... \rangle$, each action is associated with a set of features. Hence, each training example is of the following form:

$$\langle (F_0, L_0^2), (F_1, L_1^2), (F_2, L_2^2), ... \rangle \tag{11}$$

where L_i^2 is the action label for predictability (and explainability) for a_i. F_i is the set of features for a_i which includes:

Plan Features: Given the robot model (specified as an action model), the set of plan features for a_i includes the action description, as well as state variables after executing the sequence of actions $\langle a_0, ..., a_i \rangle$ from the initial state. This information can be easily extracted given the model. For example, in our rover example in Fig. 2, this set of features for

a_0 includes: $at\ rover\ l_5$, $at\ resource0\ l_2$, $at\ resource1\ l_4$, $at\ storage0\ l_3$.

Action Features: The set of action features for a_i describes the motion (e.g., dynamics) of this action. These features can be used to capture, for example, smoothness of action execution as well as transition between actions. These features serve as important cognitive cues for humans to understand robot actions. For example, an action that enables a robot to cross a river may be interpreted as *swimming*, *pedaling*, or *propelling* depending on how the robot crosses the river. Action features can be extracted for a plan before execution (for testing) given the dynamics of the robot.

Interaction Features: Interaction features take into account of a_i's influence on the human. For example, it can include how far the human is from the robot, and what the human is performing when a_i is being executed. This set of features captures how the robot interacts with the human, and how these interactions influence the human's interpretation of the robot's plan under different situations. This set of features are more conveniently extracted from sensor information. During plan synthesis or the testing phase when such information is unavailable, this set of features may need to be estimated, for example, based on the inferred goal and plan of the human.

Using the Learned Model

Given a set of training examples in the form of Eq. (11), we can train the CRF model in Eq. (3). During plan synthesis (or testing), the robot needs to synthesize a plan that is explainable and predictable. The most straightforward method is for it to consider a set of candidate plans. For each plan, the robot must first extract the features of the actions as we discussed earlier. It then uses the trained model (i.e., $\mathcal{L}^*_{CRF}$) to produce the labels for the actions in the plan. θ and β can then be computed given the mappings in Defs. (1) and (2). These measures can then be used to choose a plan that is more explainable and predictable. A more efficient way is to incorporate these measures into heuristic values in the planning process. We plan to consider this in future work.

The capability to synthesize explainable and predictable plans is useful for cobots. For example, in domains where humans interact closely with robots (e.g., in an assembly warehouse), more preferences should be given to plans that are more explainable and predictable since there would be high risks if the robots act unexpectedly. Furthermore, the weights on explainability and predictability may vary significantly in different domains. For example, there are domains in which explainability matters the most (e.g., non-teaming tasks). We plan to investigate the integration of these measures and their impacts in various domains.

Evaluation

We evaluate our approach on a synthetic dataset based on our example rover domain. The aim is twofold: evaluate how well the learning approach can capture the labeling scheme of the human given the training data; evaluate planning with the consideration of the θ and β measures against a baseline.

Dataset Synthesis

To simplify the data synthesis process, we make the following assumptions: all rover actions have the same cost; all rover actions are associated with at most one task label (i.e., $L = T + \emptyset$ in Eq. (6)). To differ M_R from $\mathcal{M}^*_R$, we add "oscillations" to the plans of the rover. These oscillations are incorporated by randomly adding locations for the rover to visit as *hidden goals*. For these locations, the rover only needs to visit them. As a result, it may demonstrate "unexpected" behaviors given only the *public goal*, denoted by G, which is known to both the rover and human. We denote the goal that also includes the hidden goals as G'. Given a problem with a public goal G, we implement a labeling scheme of the human to automatically provide the "ground truth" of a rover plan, which is constructed by the rover to achieve G'.

For each action in this rover plan, we associate it with a current and a next label incrementally. These labels are chosen from $\{\{\text{COLLECT}\}, \{\text{STORE}\}, \{\text{OBSERVE}\}, \emptyset\}$. We denote the plan prefix $\langle a_0, ...a_i\rangle$ for a plan π as π_i, the state after applying π_i as s_i from the initial state, and a plan that is constructed from s_i to achieve G (i.e., using s_i as the initial state) as $P(s_i)$. For the current label of a_i:

1. If $|P(s_i)| \geq |P(s_{i-1})|$, we label a_i as $\emptyset$ (i.e., unexplainable). This rule means that humans may label an action as unexplainable if it does not contribute to achieving G.
2. If $|P(s_i)| < |P(s_{i-1})|$, we label a_i based on the distances from the current rover location to the targets (i.e., storage areas or observation locations), current state of the rover (i.e., loaded or not), and whether a_i moves the rover closer to these targets. For example, if the closest target is a storage area and the rover is loaded, we label a_i as $\{\text{STORE}\}$. When there are ties, we label a_i as $\emptyset$ (i.e., undetermined).

For the next label of a_i:

1. This label is determined by the target that is closest to the rover state after the current task is achieved. When there are ties, a_i is labeled as $\emptyset$ (i.e., undetermined). If the current label is $\emptyset$, we also label a_i as $\emptyset$ (i.e., unpredictable).
2. If the current task is also the last task, we label a_i as $\emptyset$ since there is no next task.

For evaluation, we define F_θ and F_β as in Eqs. (7) and (8). We randomly generate problems in a 4×4 environment. For each problem, we randomly generate $1-3$ resources as a set RE, $1-3$ storage areas as a set ST, $1-3$ observation locations as a set OB. The public goal G of a problem, first, includes making all storage areas non-empty. To ensure a solution, we force $|RE| = |ST|$ if $|RE| < |ST|$. Furthermore, the rover must make observations at the locations in OB. G' for the rover includes G above, as well as a set of hidden goals (details given later). Locations of the rover, RE, ST, OB and hidden goals are randomly generated in the environment and do not overlap in the initial state. Although seemingly simple, the state space of this domain is on the order of 10^{20}.

Evaluation Results

Since we use synthetic data, we extract only plan features in our evaluations. First, we evaluate the predication performance of the learned model (i.e., $\mathcal{L}^*_{CRF}$) as the number

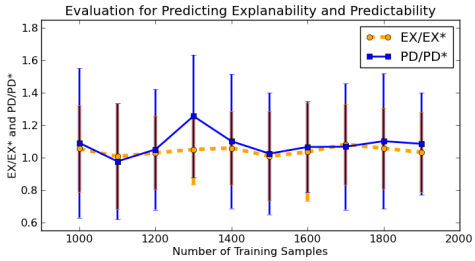


Figure 3: Evaluation for predicting θ and β as the number of training samples increases. EX denotes θ; PD denotes β. The superscript $*$ is used to denote θ and β computed from the ground truth ($\mathcal{L}^*$). The result shows the means and standard deviations at each setting for 100 testing samples.

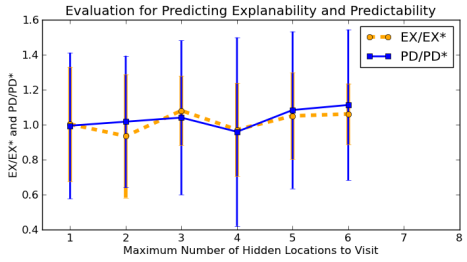


Figure 4: Evaluation for predicting θ and β measures as the difference between M_R and $\mathcal{M}_R^*$ increases (i.e., as the maximum number of hidden goals increases).

of training samples increases. Then, we evaluate the performance as the difference between M_R and $\mathcal{M}_R^*$ gradually increases (i.e., as the number of hidden goals increases). Finally, we compare the planning performance with a baseline.

In this evaluation, we randomly generate $1-3$ hidden goals to include in G'. The number of training samples is gradually increased from 1000 to 1900 with step size 100. After the model is trained, we evaluate it on 100 testing samples. The result is presented in Fig. 3. We can see that the prediction performance (i.e., the ratios between θ and β computed based on $\mathcal{L}_{CRF}^*$ and $\mathcal{L}^*$) is generally between $50\%-150\%$, which shows that our approach can capture $\mathcal{L}^*$ relatively well. We anticipate this performance to be further improved when using general types of CRFs and incorporating more complex features. Furthermore, the performance does not improve much as the sample size increases. This shows that our approach can capture $\mathcal{L}^*$ using a relatively small amount of training samples.

The number of training samples in this evaluation is fixed to be 1000. The other settings are kept the same as those in the previous evaluation except that we vary the maximum number of hidden goals from 1 to 6 with step size 1. The result is presented in Fig. 4. We can see that the oscillation levels do not seem to influence the prediction performance much. This shows that our approach is effective whether M_R and $\mathcal{M}_R^*$ are similar or largely different.

In this evaluation, we compare the performance of planning with the consideration of θ and β (denoted as EXPD)

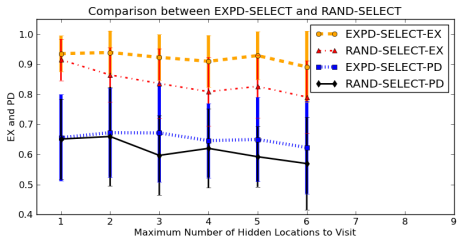


Figure 5: Comparison of EXPD and BASELINE

with a baseline approach that does not (denoted as BASELINE). For a given public goal G, we randomly construct 20 problems with a given level of oscillation as determined by the maximum number of hidden goals. Each such problem corresponds to a different G' and a plan is created for it. The set of plans for these 20 problems associated with the same G represents the set of candidate plans for G. For each level of oscillation, we randomly generate 50 different Gs and then construct the set of candidate plans for each G. Again, we gradually increase the level of oscillation as in our second evaluation. While EXPD chooses plans with the highest θ and β values based on a previously learned model (i.e., the model learned from 1900 samples from the first evaluation), BASELINE randomly chooses plans.

We compare the θ and β values computed from the ground truth labeling of the chosen plans. The result is provided in Fig. 5. When the oscillation is small, the performances of both approaches are similar. As the oscillation increases, the performances of the two approaches diverge. This is expected since BASELINE randomly chooses plans and hence its performance should decrease as the oscillation increases. On the other hand, EXPD is not influenced as much although its performance also tends to decrease. This is partly due to the fact that the model used in this evaluation is trained with samples having a maximum of 3 hidden goals.

Conclusion

In this paper, we introduced plan explainability and predictability for cobots that can autonomously construct plans given goals. The motivation is that for such robots to cohabit with humans, they must consider not only their own models, but also the human's interpretation of their models when constructing plans. This enables cobots to synthesize plans that do not surprise the humans. To the best of our knowledge, this is the first attempt to model and use plan explainability and predictability, which differs significantly from previous work on human-aware planning and human-robot teaming. The proposed measures have a variety of applications (e.g., achieving fluent human-robot interaction and human safety). To compute these measures, we learn the labeling scheme of humans for robot plans from training examples based on CRFs. We then use this learned model to label a new plan to compute its explainability and predictability. The proposed method is evaluated on a synthetic dataset to show that it is effective in capturing explainability and predictability, and producing explainable and predictable plans.

References

[Abbeel and Ng 2004] Abbeel, P., and Ng, A. Y. 2004. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the Twenty-first International Conference on Machine Learning*, ICML '04, 1–. New York, NY, USA: ACM.

[Chakraborti et al. 2015] Chakraborti, T.; Briggs, G.; Talamadupula, K.; Zhang, Y.; Scheutz, M.; Smith, D.; and Kambhampati, S. 2015. Planning for serendipity. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*.

[Charniak and Goldman 1993] Charniak, E., and Goldman, R. P. 1993. A bayesian model of plan recognition. *Artificial Intelligence* 64(1):53 – 79.

[Cirillo, Karlsson, and Saffiotti 2009] Cirillo, M.; Karlsson, L.; and Saffiotti, A. 2009. Human-aware task planning for mobile robots. In *Advanced Robotics, 2009. ICAR 2009. International Conference on*, 1–7.

[Fong, Nourbakhsh, and Dautenhahn 2003] Fong, T. W.; Nourbakhsh, I.; and Dautenhahn, K. 2003. A survey of socially interactive robots. *Robotics and Autonomous Systems*.

[Fox and Long 2003] Fox, M., and Long, D. 2003. Pddl2.1: An extension to pddl for expressing temporal planning domains. *J. Artif. Int. Res.* 20(1):61–124.

[Gbelbecker et al. 2010] Gbelbecker, M.; Keller, T.; Eyerich, P.; Brenner, M.; and Nebel, B. 2010. Coming up with good excuses: What to do when no plan can be found. In *International Conference on Automated Planning and Scheduling*.

[Hanheide et al. 2015] Hanheide, M.; Gbelbecker, M.; Horn, G. S.; Pronobis, A.; Sj, K.; Aydemir, A.; Jensfelt, P.; Gretton, C.; Dearden, R.; Janicek, M.; Zender, H.; Kruijff, G.-J.; Hawes, N.; and Wyatt, J. L. 2015. Robot task planning and explanation in open and uncertain worlds. *Artificial Intelligence*.

[Hoffman and Breazeal 2007a] Hoffman, G., and Breazeal, C. 2007a. Cost-based anticipatory action selection for human–robot fluency. *Robotics, IEEE Transactions on* 23(5):952–961.

[Hoffman and Breazeal 2007b] Hoffman, G., and Breazeal, C. 2007b. Effects of anticipatory action on human-robot teamwork efficiency, fluency, and perception of team. In *Proceedings of the ACM/IEEE international conference on Human-robot interaction*, 1–8. ACM.

[Kautz and Allen 1986] Kautz, H. A., and Allen, J. F. 1986. Generalized Plan Recognition. In *National Conference on Artificial Intelligence*, 32–37.

[Lafferty, McCallum, and Pereira 2001] Lafferty, J. D.; McCallum, A.; and Pereira, F. C. N. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, 282–289. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

[Levine and Williams 2014] Levine, S. J., and Williams, B. C. 2014. Concurrent plan recognition and execution for human-robot teams. In *Twenty-Fourth International Conference on Automated Planning and Scheduling*.

[Li et al. 2014] Li, N.; Cushing, W.; Kambhampati, S.; and Yoon, S. 2014. Learning probabilistic hierarchical task networks as probabilistic context-free grammars to capture user preferences. *ACM Trans. Intell. Syst. Technol.* 5(2):29:1–29:32.

[Murray 1999] Murray, T. 1999. Authoring Intelligent Tutoring Systems: An analysis of the state of the art. *International Journal of Artificial Intelligence in Education (IJAIED)* 10:98–129. Part II of the Special Issue on Authoring Systems for Intelligent Tutoring Systems (editors: Tom Murray and Stephen Blessing).

[Narayanan et al. 2015] Narayanan, V.; Zhang, Y.; Mendoza, N.; and Kambhampati, S. 2015. Automated planning for peer-to-peer teaming and its evaluation in remote human-robot interaction. In *ACM/IEEE International Conference on Human Robot Interaction (HRI)*.

[Peng and McCallum 2006] Peng, F., and McCallum, A. 2006. Information extraction from research papers using conditional random fields. *Information Processing & Management* 42(4):963 – 979.

[Ramírez and Geffner 2010] Ramírez, M., and Geffner, H. 2010. Probabilistic plan recognition using off-the-shelf classical planners. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2010, Atlanta, Georgia, USA, July 11-15, 2010*.

[Shah et al. 2011] Shah, J.; Wiken, J.; Williams, B.; and Breazeal, C. 2011. Improved human-robot team performance using chaski, a human-inspired plan execution system. In *Proceedings of the 6th international conference on Human-robot interaction*, 29–36. ACM.

[Sisbot et al. 2007] Sisbot, E.; Marin-Urias, L.; Alami, R.; and Simeon, T. 2007. A human aware mobile robot motion planner. *Robotics, IEEE Transactions on* 23(5):874–883.

[Talamadupula et al. 2014] Talamadupula, K.; Briggs, G.; Chakraborti, T.; Scheutz, M.; and Kambhampati, S. 2014. Coordination in human-robot teams using mental modeling and plan recognition. In *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, 2957–2962.

[Zhang et al. 2015] Zhang, Y.; Narayanan, V.; Chakraborty, T.; and Kambhampati., S. 2015. A human factors analysis of proactive assistance in human-robot teaming. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*.

[Zhuo and Kambhampati 2013] Zhuo, H. H., and Kambhampati, S. 2013. Action-model acquisition from noisy plan traces. In *IJCAI*.