Selecting Source Tasks for Transfer Learning of Human Preferences

Heramb Nemlekar, Naren Sivagnanadasan, Subham Banga, Neel Dhanaraj, Satyandra K. Gupta, and Stefanos Nikolaidis

***Abstract*—We address the challenge of transferring human preferences for action selection from simpler source tasks to complex target tasks. Our goal is to enable robots to support humans proactively by predicting their actions — without requiring demonstrations of their preferred action sequences in the target task. Previous research has relied on human experts to design or select a simple source task that can be used to effectively learn and transfer human preferences to a known target. However, identifying such source tasks for new target tasks can demand substantial human effort. Thus, we focus on automating the selection of source tasks, introducing two new metrics. Our first metric selects source tasks in which human preferences can be accurately learned from demonstrations, while our second metric selects source tasks in which the learned preferences, although not as accurate, can match the preferred human actions in the target task. We evaluate our metrics in simulated tasks and two human-led assembly studies. Our results indicate that selecting high-scoring source tasks on either metric improves the accuracy of predicting human actions in the target task. Notably, tasks chosen by our second metric can be simpler than the first, sacrificing learning accuracy but preserving prediction accuracy.**

***Index Terms*—Assembly, human-robot collaboration, transfer learning.**

I. Introduction

WE CONSIDER the problem of learning human preferences for action selection without the need for lengthy and laborious human demonstrations of the target task. Specifically, we want to enable robots to anticipate human actions and offer proactive assistance in complex tasks, such as a model airplane assembly, based on demonstrations of their preferred action sequences in much simpler tasks (see Fig. 1).

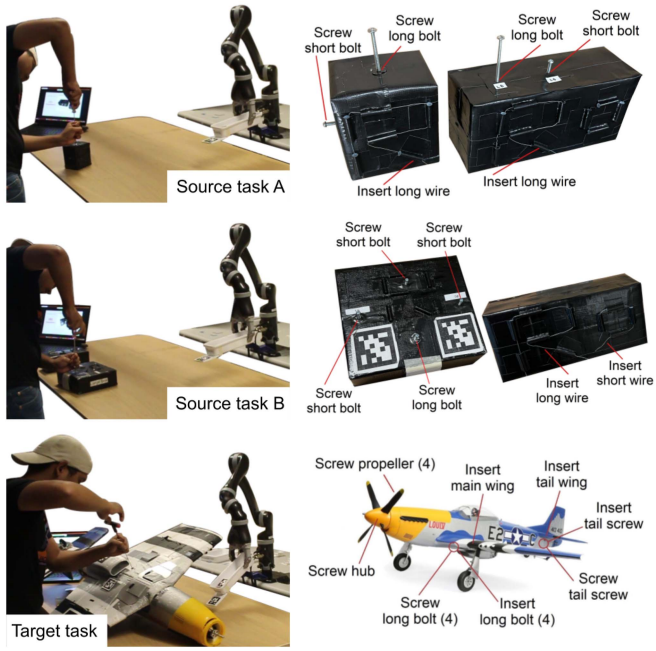

Fig. 1. We generate a set of source tasks with just 6 actions (e.g., A and B) and select the best source task (i.e., A) based on our *behavioral equivalence classes similarity* metric for accurately transferring human preferences to the target task of assembling a model airplane, consisting of 17 actions.

Previous research suggests that we can transfer the preferences of users from a short source task to a longer target task by representing them as a function of *task-agnostic features*, such as the effort required for user actions [1] or graph-based features of the user positions [2]. These features are common to both the source and target tasks, enabling the robot to learn user preferences from demonstrations in the source task and apply them for predicting user actions in the target task. For instance, users who started with the least-effort actions in a source assembly task also exhibited the same preference in the target assembly [3].

However, in these approaches, the source tasks are either pre-specified or manually designed based on human intuition and trial and error. For example, to learn user preferences for starting with actions that required the least effort, the source

Manuscript received 20 November 2023; accepted 20 May 2024. Date of publication 17 June 2024; date of current version 24 June 2024. This letter was recommended for publication by Associate Editor N. Yamanobe and Editor G. Venture upon evaluation of the reviewers' comments. This work was supported by NSF under Grant #2024936 and in part by Agilent Early Career Professor Award. *(Corresponding author: Heramb Nemlekar.)*

This work involved human subjects or animals in its research. Approval of all ethical and experimental procedures and protocols was granted by the University of Southern California Institutional Review Board (IRB) under Application No. UP-21-00695, and performed in line with the 45 CFR 46.111.

Heramb Nemlekar is with the Viterbi School of Engineering, Thomas Lord Department of Computer Science, University of Southern California, Los Angeles, CA 90007 USA, and also with the Department of Mechanical Engineering, Virginia Tech, Blacksburg, VA 24061 USA (e-mail: nemlekar@usc.edu).

Naren Sivagnanadasan, Subham Banga, Neel Dhanaraj, and Stefanos Nikolaidis are with the Viterbi School of Engineering, Thomas Lord Department of Computer Science, University of Southern California, Los Angeles, CA 90007 USA (e-mail: nsivagna@usc.edu; sbanga@usc.edu; dhanaraj@usc.edu; nikolaid@usc.edu).

Satyandra K. Gupta is with the Viterbi School of Engineering, Thomas Lord Department of Computer Science, University of Southern California, Los Angeles, CA 90007 USA, and also with the Viterbi School of Engineering, Aerospace and Mechanical Engineering Department, University of Southern California, Los Angeles, CA 90007 USA (e-mail: guptask@usc.edu).

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2024.3415432, provided by the authors.

Digital Object Identifier 10.1109/LRA.2024.3415432

assembly task included actions with varying levels of effort and allowed the user to choose between these actions when providing demonstrations. Designing such a source task can require significant human effort and there is limited research on selecting a suitable source task without human guidance. Thus, in this work, we focus on the research question:

How can we automatically select a good source task such that human preferences transferred from that task can enable accurate anticipation of the human actions in the target task?

We assume that human actions are guided by a reward function, which is a weighted sum of relevant task-agnostic features [3]. The feature weights specify a user's preference and we employ inverse reinforcement learning (IRL) [4] to learn the user's weights from their demonstrations in a source task. However, there are often many weights that can explain the user's demonstrations, making it difficult to pinpoint the actual user weights [5]. If the weights learned in the source task differ from the actual user weights, they may not produce the same actions as the user in the target task. Therefore, our first approach is to select the most informative source task such that user preferences (i.e., weights) can be accurately learned from their demonstrations.

Just as multiple weights can explain the user demonstrations in a source task, there can be many weights that produce the same actions as the user in the target task. Hence, to accurately predict user actions, our key insight is that the *weights learned in the source task can differ from the actual user weights as long as they are behaviorally equivalent (i.e., produce the same actions) in the target task*. Therefore, our second approach is to select the source task where the learned weights are most likely to be behaviorally equivalent to the actual user weights in the target task.

In summary, we propose two metrics for automatically selecting the best source task from a set of candidate tasks:

1) *Information Gain:* measures the reduction in uncertainty when learning the actual weights of any user from their demonstrations in a given source task.
2) *Behavioral Equivalence Classes Similarity (BECS):* measures the likelihood of the weights learned for any user in the source task being behaviorally equivalent to the actual weights of the same user in the target task.

We evaluate the benefit of selecting source tasks using our proposed metrics in simulation and across two studies: one with 22 users playing an assembly game online, and another involving 19 users performing real-world assembly tasks in person. We find that, for certain target tasks, using *BECS* allows us to find shorter source tasks compared to *information gain* while achieving similar accuracy in predicting user actions. Overall, our results show that preferences transferred from short source tasks with high scores on either metric lead to improved prediction accuracy in larger target tasks.

II. Related Work

Prior research in transfer learning has primarily focused on efficiently training simulated agents [6], [7], [8] and robots [9], [10] to optimally perform the target tasks. The problem of transferring human preferences is distinct from this body of work because it focuses on using the transferred knowledge to predict human behavior for offering robot assistance. Yet, much like transfer learning for synthetic agents [11], we can divide the problem of transferring human preferences into three steps: (i) selecting an appropriate source task to learn human preferences, (ii) establishing the relationship between the source and target tasks, and (iii) effectively transferring the preferences to the target task.

Similar to recent work in transferring human preferences [1], [2], prior work in transfer learning for synthetic agents has also relied on human experts to select the source task and provide a mapping to the target task. In robot soccer [6], the weights for the new states and actions in a 4-vs-3 game are initialized by duplicating the weights learned for similar states and actions in a 3-vs-2 game. Similarly, preferences of a simulated human from a block stacking task with 2 red and 3 blue blocks are used to expedite learning in a target task with 1 extra red block [12].

More generally, the source and target tasks can differ along various dimensions [13] like the objectives, states, actions, constraints, or dynamics of the two tasks. They may have the same states and actions but differ only on the reward function [10] or have the same reward structure but different state and action spaces [14]. While there is some work on learning the mapping between the states [15] and objects [16] of the source and target tasks, there is limited research on generating and selecting the source tasks automatically.

Prior work suggests that a good source task can be selected from a set of candidate tasks by measuring their similarity with the target task [17]. Their similarity metrics require computing the performance of the learned policy in the target task or computing the number of states where the rewards or values of the two tasks are similar. The similarity between tasks can also be computed based on the likelihood of observing samples from the target task in the candidate source tasks [18]. However, these metrics assume that the tasks have similar state and action spaces or that the robot has access to data samples in the target task.

In this work, we allow for differences in the states and actions of the source and target tasks. We assume that the only shared aspect is the reward function, which is represented by a common set of task-agnostic features. Unlike prior work, our goal is to select the best source task without having access to any user demonstrations. Most importantly, all of the prior research in selecting source tasks was for training synthetic agents to optimally perform the target task. To our knowledge, ours is the first to do so for transferring human preferences. The key distinction here is that we want to select a source task that can facilitate the transfer of a broad range of user preferences, as opposed to a single optimal policy.

III. Problem Formalism

A. Task Definition

We model each task as a Markov Decision Process (MDP) $M := (S, A, T, R, S_{start}, S_{end})$; where S is a finite set of states in the given task, A is a finite set of discrete human actions, $T(s_{t+1}|s_t, a_t)$ is the transition probability function, S_{start} and S_{end} are the sets of starting and terminal states, and $R(s_t)$ is the human's reward function. We assume that S, A, T, S_{start}, and S_{end} are known for a given task, while R captures the unknown user preference.

The user policy, $\pi : S \times A \mapsto [0, 1]$, is a mapping from the states to a probability distribution over actions. We assume that users choose actions to optimize for their preferences, i.e., maximize their long-term reward. In a state s, users will choose

action $a^\star$ based on their optimal Q-value function.

$$a^\star = \arg\max_{a \in A} \max_{\pi} Q^\pi(s,a) \tag{1}$$

$$Q^\pi = R^w(s) + \gamma \mathbb{E}_{s' \sim T(\cdot|s,a)}[V^\pi(s')] \tag{2}$$

$$V^\pi(s) = \mathbb{E}\left[\sum_{t=0}^{N} \gamma^t R^w(s_t) \,\middle|\, \pi, s_0 = s\right] \tag{3}$$

Here, V^π is the value of being in a state s and N is the time step at which a terminal state is reached. We assume that the users can optimize their sequence of actions for the entire task without discounting future rewards ($\gamma = 1$).

B. Transfer Learning

Our goal is to estimate the user policy in a target task $M_\mathcal{T}$ based on their demonstrations Ξ in a source task $M_\mathcal{S}$. For this, we represent the user's reward function as a weighted sum of d *task-agnostic features*, $\phi : \{S_\mathcal{S}, S_\mathcal{T}\} \mapsto \mathbb{R}^d$ [1].

$$R^w(s) = w^T \phi(s) \quad \forall s \in \{S_\mathcal{S}, S_\mathcal{T}\} \tag{4}$$

Here, $S_\mathcal{S}$ and $S_\mathcal{T}$ are the state spaces of the source and target task, respectively.

The task-agnostic features $\phi(s) = [\phi_1(s), \ldots, \phi_d(s)]$ are aspects shared by the states of both tasks that dictate how users prefer to sequence their actions. These features can be extracted from abstract representations of the task states, such as the graph-based features of user locations [2], or hand-crafted based on domain knowledge [3]. We follow the latter approach in this work. The weights $w = [w_1, \ldots, w_d]$ indicate how each user prioritizes these features and encodes their individual preference. With this representation, we can learn the feature weights $\hat{w}$ (i.e., preference) from user demonstrations in the source task and use them to compute the rewards and estimate the user policy $\hat{\pi}_\mathcal{T}$ in the target task.

C. Inverse Reinforcement Learning

We assume that each user operates according to a ground-truth weight w and provides demonstrations Ξ^w by executing the optimal policy based on these weights. Every demonstration, $\xi \in \Xi^w$, is a sequence of task states and user actions, i.e., $[(s_0, a_0, s_1), \ldots, (s_t, a_t, s_{t+1}), \ldots, (s_{N-1}, a_{N-1}, s_N)]$.

We use inverse reinforcement learning (IRL) [19] to learn the weights of a given user from their demonstrations in the source task, $\hat{w} = IRL(\Xi)$. Specifically, we assume an IRL approach that maps user demonstrations to a weight $\hat{w}$ in a space of user preferences W, by matching the expected feature count of the learner's optimal policy, μ_π, to the mean feature count of the user demonstrations, μ_Ξ.

$$\mu_\pi = \mathbb{E}\left[\sum_{t=0}^{N} \phi(s_t) \,\middle|\, \pi\right] \qquad \mu_\Xi = \frac{1}{|\Xi|} \sum_{\xi \in \Xi} \sum_{s_t \in \xi} \phi(s_t) \tag{5}$$

In practice, we can compute μ_π by simulating the optimal demonstrations, $\Xi^{\hat{w}}$, for the learner's current estimate.

D. Problem Definition

Our focus is on selecting an appropriate source task for the transfer learning of human preferences. Specifically, given the target task $M_\mathcal{T}$, the task-agnostic features ϕ, and a set of source tasks $\mathcal{M}_\mathcal{S}$, we want to find the best source task, $M_\mathcal{S}^\star \in \mathcal{M}_\mathcal{S}$, such that the policy $\hat{\pi}_\mathcal{T}$ computed with weights learned from user demonstrations in $M_\mathcal{S}^\star$, accurately predicts the users' actions in that target task.

IV. Source Task Selection

Consider the example shown in Fig. 2, where a robot has to guide users to their end locations in the target task. Each user can have their own preference, $w = [w_1, w_2]$, for minimizing the total time (ϕ_1) and money (ϕ_2) required to reach the end. For example, users that have a ground-truth weight of $w = [0.8, 0.2]$ will follow the blue path shown in the target task because they have a lower weight for spending money than time. Given a set of source tasks, such as A, B, and C, we want to quantify the effectiveness of each source task in facilitating accurate prediction of user actions in the target task based on user demonstrations in that source task.

A. Information Gain

If the weights learned from user demonstrations in a source task match the ground-truth weights, i.e., $\hat{w} = w$, the robot can accurately reproduce the user's actions in the target task. However, IRL is an ill-posed problem [19]. Often, there can be many weights, $\hat{w} \in W$, for which the optimal policy produces similar demonstrations as the ground-truth weights.

This uncertainty in determining the user's weights can be representational, e.g., W may contain scalar multiples of w, or experimental, e.g., if every $w \in W$ results in the same demonstrations in the source task [5]. For example, source task A in Fig. 2 forces every user to follow the same path, while in source task B, every demonstration has the same total feature count, i.e., costs the same amount of time and money. Thus, we cannot gain any information about the user's preference from demonstrations in such tasks.

On the other hand, in source task C, users can choose to spend either time or money. The demonstrations, $\xi \in \Xi^w$, of any user with $w = [0.8, 0.2]$ will follow the path shown in blue, avoiding the grey cells. When learning from the user's demonstrations, the IRL approach would map to $\hat{w} \in W$, for which the optimal demonstrations $\Xi^{\hat{w}}$ have the same mean feature count as Ξ^w.

However, in the space of weights for source task C, any weight in the shaded blue region, say $\hat{w} = [0.8, 0.9]$, would also result in the same set of demonstrations and thus have the same mean feature count as Ξ^w. Therefore, for the IRL approach, any weight in this blue region is a valid solution for $\hat{w}$. We refer to the set of all such weights as the *behavioral equivalence class* (BEC) of w in the source task $M_\mathcal{S}$ [20].

$$BEC(w|M_\mathcal{S}) = \{\hat{w} \in W \,|\, \mu_{\Xi^{\hat{w}}} = \mu_{\Xi^w} \text{ in } M_\mathcal{S}\} \tag{6}$$

The volume of $BEC(w|M_\mathcal{S})$ measures the uncertainty in pinpointing the ground-truth weights w. In the worst case, all weights belong to the same BEC, as observed in source tasks A and B, while in the most ideal case, each weight results in a unique set of demonstrations and belongs to a separate BEC. Hence, our first approach is to select a source task in which fewer weights are behaviorally equivalent to one another, allowing the IRL approach to estimate the ground-truth weights with greater certainty. We use entropy to measure the amount of information that can be obtained from user demonstrations in a source

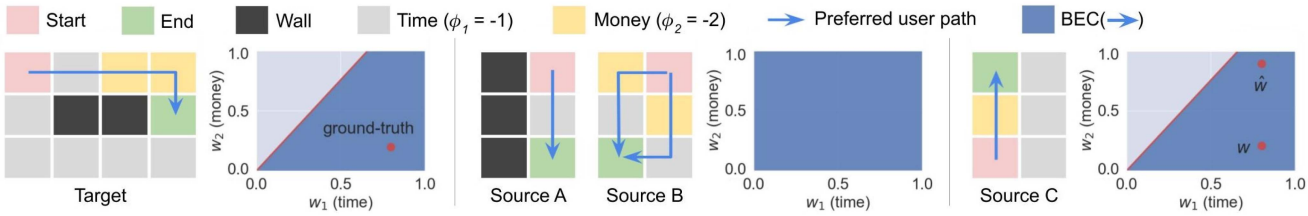

Fig. 2. Navigation assistance: We want the robot to anticipate the user's preferred path in a target map. The user can move up, down, left, or right, unless prevented by a wall. Each yellow cell requires the user to spend 2 units of money, while every other cell requires 1 unit of time. The shaded blue regions in the space of user weights represent the behavioral equivalence class of the user's demonstration in the corresponding task.

task $M_{\mathcal{S}}$.

$$H(M_{\mathcal{S}}) = -\sum_{i=1}^{k} P_i \log_2(P_i) \tag{7}$$

where, k is the number of behavioral equivalence classes (BECs) of weights in the source task, and P_i is the proportion of weights in the i-th BEC. The number of BECs in a source task can range from 1 to $|W|$. For example, in task C, the space of preferences is divided into two BECs. We can see that the value of $H(M_{\mathcal{S}})$ increases as we increase the number of BECs in the space of weights.

In our experiments, we compute P_i by uniformly sampling the space of weights, simulating their demonstrations in the source task, and comparing their feature counts. If all weights belong to the same BEC, $H(M_{\mathcal{S}}) = 0$. Conversely, if each weight belongs to a different BEC, $H(M_{\mathcal{S}}) = \log_2 |W|$.

Given a set of source tasks $\mathcal{M}_{\mathcal{S}}$, we select the best source task $M^{\star}_{\mathcal{S}}$, by maximizing the *information gain*.

$$M^{\star}_{\mathcal{S}} = \arg\max_{M_{\mathcal{S}} \in \mathcal{M}_{\mathcal{S}}} H(M_{\mathcal{S}}) \tag{8}$$

B. Behavioral Equivalence Classes Similarity

A key motivation in transfer learning of human preferences is to have a short source task for reducing the time and effort required to obtain user demonstrations. However, with our first approach, the ideal source task would have to allow for as many unique demonstrations as the number of distinct weights in W. Such a task may have a large state and action space, requiring lengthy user demonstrations.

Here, we make an observation that, even in the target task, there can be many weights that can produce demonstrations with the same mean feature count as the actual user demonstrations. For example, any weight in the shaded blue region of the target task in Fig. 2, including the ground-truth weight, will produce the same demonstration shown in blue. We refer to the set of these weights as the behavioral equivalence class of the ground-truth weights in the target task, $BEC(w|M_{\mathcal{T}})$.

If the weights learned in a source task, $\hat{w}$, are behaviorally equivalent to the user's ground-truth weights, w, in a target task, $M_{\mathcal{T}}$, the learned weights would produce demonstrations that visit the same features as the user in the target task. In our example, the optimal demonstrations for $\hat{w}$ would spend the same amount of time and money as w in the target task, if $\hat{w} \in BEC(w|M_{\mathcal{T}})$.

For instance, the demonstration shown in source task C can be provided by a user with $w = [0.8, 0.2]$, but mapped to a different weight $\hat{w} = [0.8, 0.9]$. Even if $\hat{w} \neq w$, we can see that $\hat{w}$ will produce the same demonstration as w in the target task. This illustrates that the learned weights only need to be as accurate as necessary to reproduce the user's behavior in the target task.

Thus, our key insight is the *weights learned in the source task need to only be behaviorally equivalent to the ground-truth weights in the target task* for anticipating user actions. We introduce a new metric called *Behavioral Equivalence Classes Similarity (BECS)*, to measure the likelihood of the weights learned in a source task, $M_{\mathcal{S}}$, belonging to the same BEC as the ground-truth weights in a target task, $M_{\mathcal{T}}$.

$$BECS(M_{\mathcal{S}}, M_{\mathcal{T}}) = \frac{1}{|W|} \sum_{w_i \in W} P(w_i, M_{\mathcal{S}}, M_{\mathcal{T}}) \tag{9}$$

$P(w_i, M_{\mathcal{S}}, M_{\mathcal{T}})$ measures the proportion of weights in $BEC(w_i|M_{\mathcal{S}})$ that also belong to $BEC(w_i|M_{\mathcal{T}})$. In our experiments, we compute P by uniformly sampling weights from the BECs corresponding to the weight w_i.

$$P(w_i, M_{\mathcal{S}}, M_{\mathcal{T}}) = \frac{\sum_{\hat{w}_j \in BEC(w_i|M_{\mathcal{S}})} \delta\left(\hat{w}_j \in BEC(w_i|M_{\mathcal{T}})\right)}{|BEC\left(w_i|M_{\mathcal{S}}\right)|} \tag{10}$$

where δ is an indicator function.

Given a set of source tasks $\mathcal{M}_{\mathcal{S}}$, we select the best source task $M^{\star}_{\mathcal{S}}$ with the objective of maximizing the $BECS$.

$$M^{\star}_{\mathcal{S}} = \arg\max_{M_{\mathcal{S}} \in \mathcal{M}_{\mathcal{S}}} BECS(M_{\mathcal{S}}, M_{\mathcal{T}}) \tag{11}$$

If $BECS(M_{\mathcal{S}}, M_{\mathcal{T}}) = 1$, any weights learned from user demonstrations in the source task are behaviorally equivalent to the user's ground-truth weight in the target. As *BECS* reduces, the probability of accurately anticipating the user's behavior in the target task based on their source task demonstrations also decreases.

In summary, while *information gain* focuses on accurately learning the weights for the task-agnostic features, *BECS* directly focuses on accurately reproducing the user's actions in a specific target task. Thus, if the target task is fixed, we select the shortest source task that receives the maximum *BECS* score. Conversely, if the target task is changing but the task-agnostic features remain the same, we can select the shortest source task that is the most *informative*.

V. Simulation Experiments

We start by evaluating the proposed metrics with simulated users. Specifically, we measure the accuracy of predicting the (simulated) user actions in randomly generated target tasks, using weights learned from their (simulated) demonstrations in source tasks having varying scores for *informativeness* and *BECS*. Our aim is to assess whether the prediction accuracy

improves when we utilize source tasks with higher scores on the proposed metrics. We make the following hypotheses:

H1. Transferring user preferences from more *informative* source tasks would improve the accuracy of predicting user actions in the target tasks.

H2. Transferring user preferences from source tasks with higher *BECS* scores would improve the accuracy of predicting user actions in the corresponding target task.

A. Implementation

To select the best source tasks using our proposed metrics, we first need access to the set of candidate source tasks $\mathcal{M}_S$, the task-agnostic feature function ϕ, and the space of user weights W. Here, we briefly describe how we acquire these inputs in our experiments.

As stated in Section III-A, each source task $M_S \in \mathcal{M}_S$ is an MDP with a fixed set of actions A, a set of states S, and a transition function T. We use the following procedure to generate each source task in $\mathcal{M}_S$:

Actions — We assume that users need to perform each action $a \in A$ to complete the source task but have the flexibility to perform these actions in their preferred sequence. We let A contain N actions. Here, N is chosen to be smaller than the number of actions in the target task to ensure that the source tasks are shorter. When generating a source task, we start with $A = \emptyset$ and randomly sample N actions from a space P that describes the properties of each action, such as the type of motion (e.g., screwing or inserting), the objects that it uses (e.g., screwdriver), or the effort it takes to perform.

Transitions — Next, to generate the transition function T, we assume that each action in the source task is deterministic. We let users perform the first sampled action at any step of the source task. For each subsequent action that we sample, we randomly decide if users can perform it in any sequence or only after performing the previously sampled action when executing the task. In this way we impose constraints on how the actions can be sequenced. Since we assign the constraints at random, the number of all possible action sequences in a source task can range from 1 to N factorial.

States — We let each state $s \in S$ in the source task be a vector of N booleans; where each boolean corresponds to an action in A and indicates whether it has been performed. For example, in a source task with $N = 4$ actions, the state after the user has performed the second and third sampled actions is $s = [0, 1, 1, 0]$. The starting state when no actions have been performed is $\vec{0}$, while the terminal state is $\vec{1}$.

In this manner, we generate the states S, actions A, and transitions T of each source task in $\mathcal{M}_S$. We now design the task-agnostic features that constitute the users' rewards.

Task-agnostic features — We compute the feature values $\phi(s)$ for every state $s \in S$ based on the properties of the actions performed by users to reach that state. For example, to design a feature that measures the total effort spent in reaching the state $s = [0, 1, 1, 0]$, we sum the efforts required for the second and third sampled actions.

Space of weights — We restrict the weights W to positive values on a d-dimensional sphere of unit radius. This ensures that W does not include weights that are scalar multiples of each other and mitigates the representational uncertainty (mentioned in Section IV-A). The dimensionality of W aligns with the dimensionality of the task-agnostic features.

We emphasize that our metrics are independent of these implementations and can be applied to any $\mathcal{M}_S$, ϕ, and W, as defined in Section III.

B. Selecting Informative Tasks

We generate a set of 512 candidate source tasks in which half of the tasks have $N = 2$ actions and the rest have $N = 4$ actions. We randomly sample the actions from a 2-D space $P = [0., 0.5, 1.0]^2$ and manually design a 2-D task-agnostic feature function ϕ based on the action properties. One such source task C, shown in Fig. 3, has two actions $a_1 = [0.5, 0.5]$ and $a_2 = [0, 1]$. Starting from $s_{start} = [0, 0]$, users can sequence these actions in any order to reach $s_{end} = [1, 1]$. The task-agnostic features for each state are the sum of the action properties, e.g., $\phi(s_{end}) = [0.5, 1.5]$. We do not assign any meaning to the properties in our simulations.

Following the same procedure, we generate 32 target tasks having 4, 6, 8, and 10 actions, each. For each source task, we measure the *information gain* and compute the *BECS* with each target task. We then measure the mean prediction accuracy for that source task using 64 newly sampled users (i.e., actual weights). We simulate the demonstrations of each user in the source task, learn their weights via maximum entropy IRL [4], and use the learned weights to predict their simulated actions in each of the target tasks.

Fig. 5 shows the scores for each generated source task on the *information gain* metric. We see that the difference between the learned weights and the actual weights is smaller when learning from informative source tasks. A Pearson correlation coefficient showed a strong negative correlation between the information gain score and the error in learning the weights, $r(510) = -0.98, p < 0.001$. Accordingly, we see that the mean prediction accuracy in target tasks improves as we use source tasks that score higher on the information gain metric. We perform a linear regression analysis to test if the metric scores affected the mean prediction accuracy across the 128 target tasks. Our results indicated that the information gain scores explained 93% of the variance and significantly affected the mean prediction accuracy ($R^2 = 0.93, F(1, 510) = 6783, p < 0.001$). This supports **H1**.

The prediction accuracy is high, even for less informative source tasks, for two reasons. First, since the space of user weights was small (2-D), many sampled users had identical action sequences in the target tasks, particularly in those with just 4 actions. Second, even if the sequences were different overall, users had matching actions in certain states because of the sequencing constraints. As the number of actions in the target tasks increases, the difference between the prediction accuracy of the least and most informative source tasks also increases. This is because larger target tasks can allow the users to express a wider range of distinct preferences, and thus we need more informative source tasks to learn their preferences and predict their actions accurately.

C. BECS vs Information Gain

Fig. 6 (Right) shows the $BECS$ scores and mean prediction accuracy for each source task with respect to a specific target

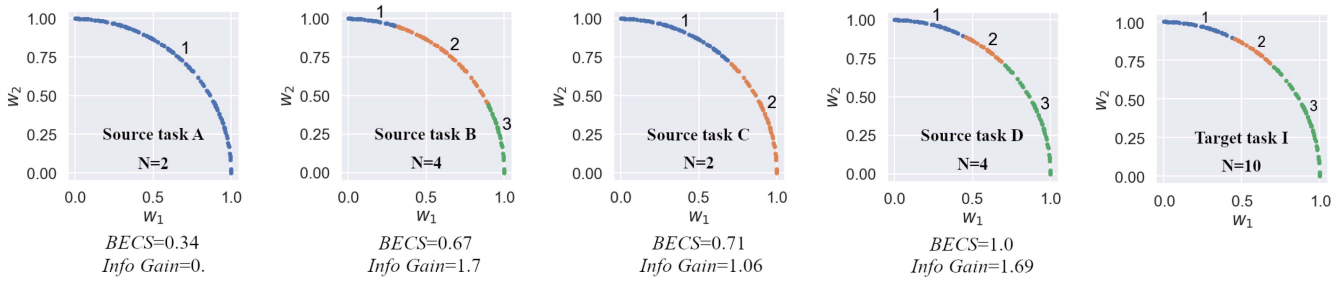

Fig. 3. Behavioral equivalence classes (BECs) of weights in different source tasks and the chosen target task I. Each BEC in a task is numbered and represented by a different color, e.g., the target task has 3 BECs. The prediction accuracy improves as the *BECS* score increases from A to D.

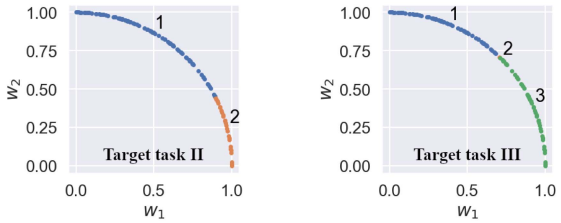

Fig. 4. Behavioral equivalence classes of weights in two other target tasks with 10 actions, differing from those in target task I.

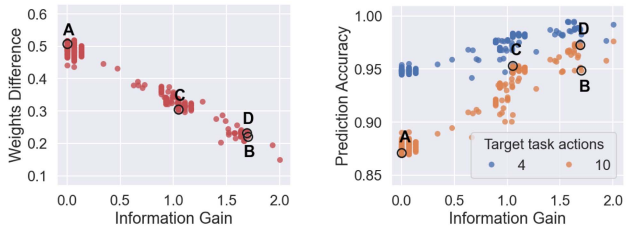

Fig. 5. Information Gain: (Left) Mean difference between the weights learned in the source task and the actual weights. (Right) mean accuracy of predicting actions in the target tasks based on learned weights.

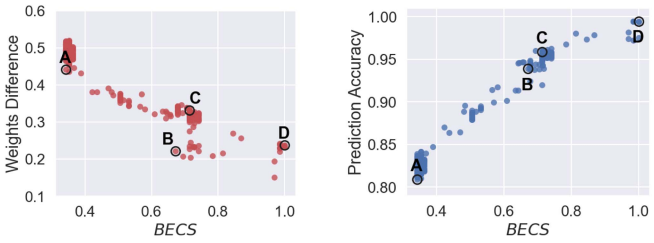

Fig. 6. BECS: (Left) Average difference between the weights learned in the source task and the actual weights, and (Right) mean accuracy of predicting actions in a specific target task based on learned weights.

task having 10 actions. We specifically choose a target task that divides the space of user weights into multiple behavioral equivalence classes (BECs).

A Pearson correlation coefficient showed a strong positive correlation between *BECS* and the mean prediction accuracy, $r(510) = 0.98$, $p < 0.001$. The best-scoring source task with $BECS = 1$, i.e. task D, achieves maximum prediction accuracy. Fig. 3 shows the behavioral equivalence classes of weights in specific source tasks and the chosen target task. We see that the best source task D divides the space of weights into similar BECs as the target task I. In contrast, in the worst-scoring source task A, all weights are behaviorally equivalent and lead to the same set of demonstrations. Consequently, task A achieves the lowest prediction accuracy. The prediction accuracy increases with decreasing differences between the BECs in the source and target tasks as we move from task A to task D. This result supports **H2**.

An interesting observation is that source task B is as informative as the best source task D since both tasks divide the space of weights into the same number of classes with a similar proportion of users in each class. Yet, it achieves lower prediction accuracy because it has a lower *BECS* score. Likewise, task C divides the weights into just two BECs and has a lower score on the information gain metric than task B. Still, it achieves a higher prediction accuracy than task B because the second class in its BECs overlaps with the third class in the target task I BECs, resulting in higher $BECS$.

A multiple regression analysis indicated that both metrics explained 97% of the variance ($R^2 = 0.97$, $F(2, 509) = 8619$, $p < 0.001$), with $BECS$ ($\beta = 0.25, t = 24.04, p < 0.001$) having a greater effect on the prediction accuracy than *information gain* ($\beta = 0.02, t = 6.71, p < 0.001$).

Another important observation is that task C has only 2 actions while task B, which is more informative, has 4 actions. Therefore, when considering both prediction accuracy and the human effort required to provide demonstrations, task C is a better source task than task B for target task I. For a different target task II with BECs shown in Fig. 4 (Left), task B ($BECS = 0.98$) is a better source task than tasks C ($BECS = 0.77$) and D ($BECS = 0.79$). Similarly, task C is the shortest, best-scoring source task ($BECS = 1.$) for the target task III shown in Fig. 4 (Right), compared to task B ($BECS = 0.94$) and D ($BECS = 1$).

Key takeaway: These results demonstrate that while information gain effectively finds source tasks that perform well across all target tasks in general, *BECS* is better at finding shorter source tasks tailored to a specific target task.

VI. Assembly Game Experiment

We now test our proposed approach with real users playing an assembly game on a computer. Specifically, we measure the accuracy of predicting user actions in a target assembly task based on their demonstrations in the best-scoring source tasks on the *information gain* metric.

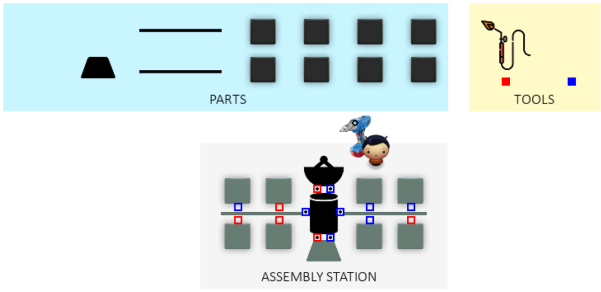

Fig. 7. Target Task: The parts are initially placed in the blue section on the left and the tools are placed in the yellow region on the right, at the top of the screen. The grey shapes in the assembly station indicate where the parts must be placed, while the small red and blue squares represent the assembly actions of welding and screwing, respectively.

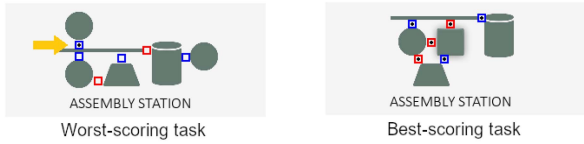

Fig. 8. Information Gain: The dot inside the red and blue squares indicates that the sequencing constraints for that action have been satisfied. Only one action (marked with an arrow) is available at the start of the worst-scoring task and each subsequent action is constrained to the previous action, forcing users to follow the same sequence of actions. On the other hand, the actions in the best-scoring task can be performed in any sequence.

Our hypothesis is that the prediction accuracy in the target task will be higher when transferring the preferences of real users from the most informative source task (**H3**).

A. Task Description

In the assembly game, users control a character to move parts to an assembly station and connect them using tools of the appropriate color (see Fig. 7). The target task has 14 actions and resembles a satellite assembly. The only sequencing constraint is that the horizontal rods must be connected to the satellite before the square panels can be placed. In a pilot study of users playing the game, we found that user preferences depended on the number of times they wanted to change the tools and parts during the task. Based on the user responses, we manually designed a 4-d feature function, $\phi = [\text{keep\_part}, \text{keep\_tool}, \text{change\_part}, \text{change\_tool}]$ to measure whether user kept the same part or tool when moving to the next task state. We select the best-scoring source task on the *information gain* metric from 256 candidate tasks; each having 6 actions and generated similarly to our simulations. We sample the action properties from known sets of parts and tools, $P = [\{\text{panel}, \text{rod}, \ldots\}, \{\text{welding}, \text{screwing}\}]$. As the baseline, we select the worst-scoring source task on our metric.

B. Online Study

We recruited 22 participants from Amazon Mechanical Turk (MTurk). Each participant played the assembly game on the Trinket app and performed all three tasks: the best-scoring source task, the worst-scoring source task, and the target assembly task. For each task, the participants were first provided a practice round to understand the states, actions, and transitions in that task, and determine their preference. After each practice round, the participants demonstrated their preferred sequence of actions in the task. Fig. 8 shows the best and worst scoring tasks for the information gain metric.

For each user, we use their demonstrations in the source task to learn their preference (i.e., weights) and then measure the accuracy of predicting their demonstration in the target task based on the learned weights. A two-tailed paired t-test showed a statistically significant difference ($t(21) = 2.09, p = 0.048$) in the mean prediction accuracy for the best ($M = 0.42, SE = 0.025$) and worst ($M = 0.36, SE = 0.011$) scoring tasks. We noticed that a fair number of MTurk participants provided sub-optimal demonstrations and often changed their preferences across tasks, resulting in lower prediction accuracy overall. Yet, this result supports hypothesis **H3** and shows the benefit of selecting informative source tasks with real users.

Key takeaway: We found that the least informative task constrains users to identical demonstrations, while the most informative task allows users to express their preferences distinctly, leading to improved predictions in the target task.

VII. Human-Robot Assembly

Finally, we conduct a user study where real users perform actual assembly tasks in collaboration with a Kinova robotic arm. While the assembly game study showed the benefit of selecting the best-scoring tasks on our first metric, the worst-scoring tasks chosen as the baseline were straightforward. Even without assessing the source tasks using our metrics, it is evident that a source task that forces every user to provide the same demonstration would not be ideal for the transfer learning of user preferences. Thus, in this study, we compare a high-scoring source task on our second metric (*BECS*) to a low-scoring source task that allows users to perform the actions in any sequence they prefer. Our hypothesis is that the accuracy of predicting user actions in a real-world assembly task would be higher when transferring preferences from a high-scoring source task, compared to a low-scoring source task on the *BECS* metric (**H4**).

A. Task Description

The target task is a model airplane assembly (see Fig. 1) with 17 actions. Users perform the assembly actions with the help of a robot that supplies the tools and parts required for each user action. In a pilot study of users performing the target task, we observed that users prefer to sequence their actions in increasing or decreasing order of effort, along with keeping or changing the parts and tools. Based on the user responses in our pilot study, we designed 6 task-agnostic features $\phi = \{\text{increasing\_effort}, \text{decreasing\_effort}, \text{keep\_part}, \text{change\_part}, \text{keep\_tool}, \text{change\_tool}\}$ by referring prior work on transfer of human preferences in assembly tasks [3].

We generate 600 source tasks with only 6 actions each. To ensure that the tasks can be designed in the real world, we randomly sample the actions from a space of properties that includes 4 primitive actions and a set of generic parts — $P = [\{\text{screw\_short\_bolt}, \text{insert\_short\_wire}, \text{screw\_long\_bolt}, \text{insert\_long\_wire}\}, \{\text{box1}, \text{box2}, \text{box3}\}]$. The primitive actions require increasing levels of effort and were obtained from [3]. Here, we do not impose any constraints on action sequencing. The high-scoring source task A, with $BECS = 0.47$, and low-scoring source task B, with $BECS = 0.15$, are shown in

Fig. 1. The low-scoring task featured at least one instance of each primitive action, while interestingly, the high-scoring task omitted one of the primitive actions.

B. Offline Evaluations

We recruited 19 participants from the student population at the University of Southern California (USC). The study was approved by USC's Institutional Review Board (UP-21-00695). Each participant performed both the source tasks and the target task, using a graphic interface to request parts and tools from the robot. We offered a practice round for each task so that participants could determine their preferred action sequence before demonstrating it in the task. We also counterbalanced the order of the source tasks to mitigate sequencing effects.

We use the demonstrations of each user in the source task to learn their weights and compute the accuracy of predicting their demonstration in the target task. A two-tailed paired t-test showed a statistically significant difference ($t(18) = 2.542$, $p = 0.020$) in the prediction accuracy for the high ($M = 0.81$, $SE = 0.025$) and low ($M = 0.75$, $SE = 0.012$) scoring tasks on *BECS*, supporting **H5**.

Since both source tasks include actions with varying effort levels, the robot could learn whether users prefer to sequence actions in increasing or decreasing order of effort from their demonstrations in either task. However, in the low-scoring task, every action that requires the same tool (e.g., screwing) also requires the same part, preventing the robot from learning whether users would prefer keeping the 'same part' more than keeping the 'same tool' or vice versa. In contrast, since *BECS* tries to find a source task in which we can differentiate between weights that have distinct behavior in the target task, the high-scoring task included two pairs of actions that require the same tool but different parts. This allowed the robot to also learn user preferences for keeping the 'same part' or 'same tool' (see supplementary video).

VIII. Conclusion

In summary, we propose two metrics to select source tasks for effectively transferring human preferences. Our first metric selects the most informative source tasks to accurately learn the preferences, while our second metric selects source tasks with similar classes of user preferences to the target task. Through simulation experiments and two user studies, we show that short source tasks selected using our proposed metrics improve the accuracy of predicting user actions in longer target tasks. This can enable robots to assist users according to their preferences without any demonstrations in the target task, thus reducing human effort.

While our work focused on proposing metrics for automating the selection of source tasks, designing the source tasks is another challenge. In our user studies, we manually designed the selected source tasks in the real world, although doing so for multiple target tasks can be burdensome. One approach to tackle this issue is by selecting a generic source task using our *information gain* metric. This metric focuses only on accurately learning user preferences over the relevant task-agnostic features, without being limited to a particular target task. However, the problem of selecting a generic source task to transfer human preferences to multiple target tasks with distinct features remains unresolved and needs further investigation. Moreover, randomly generating the source tasks and robustly computing their metric scores can be challenging in complex domains, e.g., with high-dimensional feature spaces and continuous actions. We are excited about future research on the efficient generation and design of short source tasks for transferring human preferences to complex target tasks.

References

[1] H. Nemlekar, N. Dhanaraj, A. Guan, S. K. Gupta, and S. Nikolaidis, "Transfer learning of human preferences for proactive robot assistance in assembly tasks," in *Proc. ACM/IEEE Int. Conf. Hum.-Robot Interact.*, 2023, pp. 575–583.

[2] Y. Guo, R. Jena, D. Hughes, M. Lewis, and K. Sycara, "Transfer learning for human navigation and triage strategies prediction in a simulated urban search and rescue task," in *Proc. IEEE 30th Int. Conf. Robot Hum. Interactive Commun.*, 2021, pp. 784–791.

[3] H. Nemlekar, R. Guan, G. Luo, S. K. Gupta, and S. Nikolaidis, "Towards transferring human preferences from canonical to actual assembly tasks," in *Proc. IEEE 31st Int. Conf. Robot Hum. Interactive Commun.*, 2022, pp. 1161–1167.

[4] B. D. Ziebart, A. L. Maas, J. A. Bagnell, and A. K. Dey, "Maximum entropy inverse reinforcement learning," in *Proc. 23rd AAAI Conf. Artif. Intell.*, 2008, pp. 1433–1438.

[5] A. Kareem, N. Jiang, and S. Singh, "Repeated inverse reinforcement learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017.

[6] M. E. Taylor and P. Stone, "Behavior transfer for value-function-based reinforcement learning," in *Proc. 4th Int. Joint Conf. Auton. Agents Multiagent Syst.*, 2005, pp. 53–59.

[7] T. Brys, A. Harutyunyan, M. E. Taylor, and A. Nowé, "Policy transfer using reward shaping," in *Proc. 14th Int. Conf. Auton. Agents Multiagent Syst.*, 2015, pp. 181–188.

[8] O. Day and T. M. Khoshgoftaar, "A survey on heterogeneous transfer learning," *J. Big Data*, vol. 4, no. 1, pp. 1–42, 2017.

[9] I. Clavera, D. Held, and P. Abbeel, "Policy transfer via modularity and reward guiding," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2017, pp. 1537–1544.

[10] Z. Cao, M. Kwon, and D. Sadigh, "Transfer reinforcement learning across homotopy classes," *IEEE Robot. Automat. Lett.*, vol. 6, no. 2, pp. 2706–2713, Apr. 2021.

[11] M. E. Taylor and P. Stone, "Transfer learning for reinforcement learning domains: A survey," *J. Mach. Learn. Res.*, vol. 10, no. 7, pp. 1633–1685, 2009.

[12] T. Munzer, M. Toussaint, and M. Lopes, "Preference learning on the execution of collaborative human-robot tasks," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2017, pp. 879–885.

[13] T. Fitzgerald, A. Goel, and A. Thomaz, "Abstraction in data-sparse task transfer," *Artif. Intell.*, vol. 300, 2021, Art. no. 103551.

[14] B. Banerjee and P. Stone, "General game learning using knowledge transfer," in *Proc. 20th Int. Joint Conf. Artif. Intell.*, 2007, pp. 672–677.

[15] Y. Bengio, "Deep learning of representations for unsupervised and transfer learning," in *Proc. ICML Workshop Unsupervised Transfer Learn.*, 2012, pp. 17–36.

[16] T. Fitzgerald, A. Goel, and A. Thomaz, "Human-guided object mapping for task transfer," *ACM Trans. Hum.-Robot Interact.*, vol. 7, no. 2, pp. 1–24, 2018.

[17] J. L. Carroll and K. Seppi, "Task similarity measures for transfer in reinforcement learning task libraries," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, 2005, pp. 803–808.

[18] A. Lazaric, M. Restelli, and A. Bonarini, "Transfer of samples in batch reinforcement learning," in *Proc. 25th Int. Conf. Mach. Learn.*, 2008, pp. 544–551.

[19] A. Y. Ng and S. Russell, "Algorithms for inverse reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2000, pp. 663–670.

[20] M. S. Lee, H. Admoni, and R. Simmons, "Machine teaching for human inverse reinforcement learning," *Front. Robot. AI*, vol. 8, 2021, Art. no. 693050.