Adaptive Agent Architecture for Real-time Human-Agent Teaming

Tianwei Ni[1*], Huao Li[2*], Siddharth Agrawal[1*], Suhas Raja[3], Fan Jia[1], Yikang Gui[2], Dana Hughes[1], Michael Lewis[2], Katia Sycara[1]

[1]Carnegie Mellon University, [2]University of Pittsburgh, [3]University of Texas at Austin
tianwein@cs.cmu.edu, hul52@pitt.edu, siddhara@cs.cmu.edu

Abstract

Teamwork is a set of interrelated reasoning, actions and behaviors of team members that facilitate common objectives. Teamwork theory and experiments have resulted in a set of states and processes for team effectiveness in both human-human and agent-agent teams. However, human-agent teaming is less well studied because it is so new and involves asymmetry in policy and intent not present in human teams. To optimize team performance in human-agent teaming, it is critical that agents infer human intent and adapt their polices for smooth coordination. Most literature in human-agent teaming builds agents referencing a learned human model. Though these agents are guaranteed to perform well with the learned model, they lay heavy assumptions on human policy such as optimality and consistency, which is unlikely in many real-world scenarios. In this paper, we propose a novel adaptive agent architecture in human-model-free setting on a two-player cooperative game, namely Team Space Fortress (TSF). Previous human-human team research have shown complementary policies in TSF game and diversity in human players' skill, which encourages us to relax the assumptions on human policy. Therefore, we discard learning human models from human data, and instead use an adaptation strategy on a pre-trained library of exemplar policies composed of RL algorithms or rule-based methods with minimal assumptions of human behavior. The adaptation strategy relies on a novel similarity metric to infer human policy and then selects the most complementary policy in our library to maximize the team performance. The adaptive agent architecture can be deployed in real-time and generalize to any off-the-shelf static agents. We conducted human-agent experiments to evaluate the proposed adaptive agent framework, and demonstrated the suboptimality, diversity, and adaptability of human policies in human-agent teams.

1 Introduction

Multi-agent systems have recently seen tremendous progress in teams of purely artificial agents, especially in computer games (Vinyals et al. 2019; Guss et al. 2019; OpenAI et al. 2019). However, many real-world scenarios like autonomous driving (Sadigh et al. 2018; Fisac et al. 2019), assisted robots (Agrawal and Williams 2017; Li et al. 2019), and Unmanned Aerial System (McNeese et al. 2018; Demir, McNeese, and Cooke 2017) do not guarantee teams of homogeneous robots with shared information - more often, it involves interaction with different kinds of humans who may have varying and unknown intents and beliefs. Understanding these intents and beliefs is crucial for robots to interact with humans effectively in this scenario. **Human-agent teaming** (HAT) (Scholtz 2003; Chen and Barnes 2014), an emerging form of human-agent systems, requires teamwork to be a set of interrelated reasoning, actions and behaviors of team members that combine to fulfill team objectives (Morgan Jr et al. 1986; Salas, Sims, and Burke 2005; Salas, Cooke, and Rosen 2008). In this paper, we focus on the setting of two-player human-agent teaming in a computer game, where the agent should cooperate with the human in real-time to achieve a common goal on one task. The human playing that role may be any person with any policy at any time, and potentially not be an expert in the task at hand.

One of the fundamental challenges for an artificial agent to work with a human, instead of simply another artificial agent, is that humans may have complex or unpredictable behavioral patterns and intent (Chen and Barnes 2012; Green and Bavelier 2006). In particular, they may misuse or disuse the multi-agent system based on their perception, attitude and trust towards the system (Parasuraman and Riley 1997). This difference becomes very critical in scenarios where an agent interacts with a diverse population of human players, each of which might have different intents, beliefs, and skills (ranging from novices to experts) (Kurin et al. 2017). To succeed, cooperative agents must be able to infer human intent or policy to inform their action accordingly.

Capabilities of adapting to humans are essential for a human-agent team to safely deploy and collaborate in a real-time environment (Bradshaw, Feltovich, and Johnson 2011). Real-time adaptation is critical in practical deployments where robots are not operating unilaterally in a controlled environment, such as urban driving environments for autonomous vehicles (Fisac et al. 2019). The ability to respond to real-time observations improves an agent's ability to perform in the face of various kinds of team structures or situations. Accomplishing real-time agent adaptation requires that agents are able to capture the semantics of the observed human behavior, which is likely volatile and noisy,

*indicates equal contribution. In AAAI 2021 Workshop on Plan, Activity, and Intent Recognition.

and then infer a best response accordingly. The challenge of capturing human behavior is further increased since the agent only observes a small snapshot of recent human actions, among players with varying play styles or skill levels. Finally, we note that humans may adjust their behavior in response to a changing agent policy, which can make stable adaptation difficult to achieve (Haynes and Sen 1996; Fisac et al. 2019; Sadigh et al. 2016). Real-time environments like computer games also require agents to perform both sufficiently fast estimation of the teammate's policy, as well as planning, while ensuring flexibility for unexpected strategic game states (Vinyals et al. 2017).

Past research on real-time adaptation in HAT can be divided into two forms of adaptive agent training. In the first *human-model-free* form: the agent does not build a model of human policy, but instead infers human types from the match between current observations and an exemplar and then take corresponding best actions. This setting is adopted by our approach and can be found in the psychology literature (Kozlowski and Chao 2018; Kozlowski et al. 2015; Kozlowski and Klein 2000). The second form widely adopted in robotics research *human-model-based*: first trains a model of the human to learn humans' policies or intent, then integrates the human model into the environment, and finally trains the agent upon the integrated environment. This setting requires much more computational resources than *human-model-free* form to learn the human model and deploy the agent in real-time, and inevitably imposes heavy assumptions on human policies (Zakershahrak et al. 2018; Sadigh et al. 2018; Fisac et al. 2019) like: optimal in some unknown reward function, single-type or consistent among different humans, and time-invariant for one human, etc. However these assumptions deviate from real-world human policies, especially coming from a diverse population in different skill levels and intent on a relatively hard task. On the contrary, *human-model-free* setting imposes minimal assumptions on human policies and can be deployed in real-time teaming efficiently.

In this paper, we propose a human-model-free adaptive agent architecture based on a pre-trained static agent library. The adaptive agent aims to perform well in a nontrivial real-time strategic game, Team Space Fortress (TSF) (Agarwal, Hope, and Sycara 2018). TSF is a two-player cooperative computer game where the players control spaceships to destroy the fortress. TSF presents a promising arena to test intelligent agents in teams since it involves heterogeneous team members (bait and shooter) with adversary (fortress), and it has sparse rewards which makes model training even more difficult. TSF is a nontrivial testbed to solve as it requires some real-time cooperation strategy for the two players without communication and control skills for human players. Before constructing exemplar policies, we first evaluate the nature of this testbed through previous research in *human-human* teams. The results (Li et al. 2020b) show that different human-human pairs demonstrate significantly diverse performance and the team performance was affected by both individual level factors such as skill levels and team level factors such as team synchronization and adaptation.

The diverse team performance and complicated team dynamics in *human-human* teams inspired us to build a real-time adaptive agent to cooperate with any human player in *human-agent* teams. The methodology of our real-time adaptive agent is quite straightforward. First, we design a diverse policy library of rule-based and reinforcement learning (RL) agents that can perform reasonably well in TSF when paired with each other, i.e. *agent-agent* teams. We record the self-play performance of each pair in advance. Second, we propose a novel similarity metric between any human policy and each policy in the library from observed human behavior, namely cross-entropy method (CEM) adapted from behavior cloning (Bain 1995). The adaptive agent uses the similarity metric to find the most similar policy in the exemplar policies library to the the current human trajectory. After this, the adaptive agent switches its policy to the best complementary policy to the predicted human policy in real-time. Using this adaptive strategy, it is expected to outperform any static policy from the library. Our approach is directly built upon single-agent models, thus can generalize to any off-the-shelf reinforcement learning/imitation learning algorithms.

We evaluated our approach online by having human players play against both agents with exemplar static policy and adaptive policy. These players were sourced through Amazon's Mechanical Turk (MTurk)[1] program and played TSF through their internet browsers. Each human player was assigned one role in TSF and played with all the selected agents for several trials, but was not told which agents they were playing with and was rotated through random sequences of the agents to ensure agent anonymity and reduce learning effect.

Based on the collected game data from these human-agent teams, we are interested in the three key questions: (1)How are human players' policies compared to agent policies in our library? (2) Is our adaptive agent architecture capable of identifying human policy and predicting team performance for human-agent teams? (3) Do our adaptive agents perform better than static policy agents in human-agent teams? We answer these questions in the experimental section.

2 Related Work

In the multi-agent system domain, researchers have been focusing on how autonomous agents model other agents in order to better cooperate with each other in teams, which is termed as *human-model-based* methods when applied to human-agent system in the introduction of this paper. Representative work includes *ad-hoc teamwork* that the agent is able to use prior knowledge about other teammates to cooperate with new unknown teammates (Barrett and Stone 2015; Albrecht and Stone 2018).

This is a reasonable number of work in human-robot interaction that attempts to infer human intent from observed behaviors using inverse planing or inverse reinforcement learning (Bajcsy et al. 2018; Sadigh et al. 2017, 2018; Reddy, Dragan, and Levine 2018; Fisac et al. 2019). However, these work impose ideal assumptions on human policy, e.g. optimal under some unknown reward, consistent

[1] https://www.mturk.com/

through time, and with unique type among humans, which does not hold in many complicated real-world applications, where the human-agent systems are required to generalize to various kinds of team scenarios.

In human-agent teaming, past research (Fan et al. 2009; Harbers, Jonker, and Van Riemsdijk 2012; van Zoelen et al. 2020; Levine and Williams 2018; Chen et al. 2018) has established a variety of protocols within small teams. However, these approaches often rely on some degree of explicit communication on humans' observation or intent.

The alternative setting of agent design in human-agent system is *human-model-free*, rarely discussed in the robotics literature. Some psychology literature in this setting learns to infer human intent from retrospective teammate reports, where software analyzes historical observations of humans to inform behavior in the present (Kozlowski and Chao 2018; Kozlowski et al. 2015; Kozlowski and Klein 2000). These historical behaviors may fail to capture potential changes in teammate policies a real-time environment and limit the ability of software to best adapt to a situation.

Our approach opens the door of *human-model-free* setting in human-agent system for robotics literature, significantly different from previous *human-model-based* methods. We make least assumptions on human, and use proposed architecture to realize adaptation, which involves similarity metric to infer human policy. The least assumptions and straightforward architecture enable our approach to deploy in a real-time human-agent environment with various kinds of human players.

3 Team Space Fortress

We have adapted Space Fortress (Mané and Donchin 1989), a game which has been used extensively for psychological research, for teams. Team Space Fortress (TSF) is a co-operative computer game where two players control their spaceships to destroy a fortress in a low friction 2D environment. The player can be either human or (artificial) agent, thus there are three possible combinations in teams: human-human, human-agent, and agent-agent.

A sample screen from the game is shown in Fig. 1. At the center of the stage lies a rotating fortress. The fortress and two spaceships can all fire missiles towards each other at a range. The first spaceship entering the hexagon area will be locked and shot by the fortress. However, the fortress becomes vulnerable when it is firing. Players die immediately whenever they hit any obstacles (e.g. boundaries, missiles, the fortress). The game resets every time either fortress or both players are killed. Once the fortress has been killed, both players must leave the activation region (outer pink boundaries) before the fortress respawns.

The team performance is measured by the number of fortresses that players kill. The action space is 3-dimensional discrete space, including TURN (turn left, right, or no turn), THRUST (accelerate the speed or not), and FIRE (emit one missile or not). The frame per second (FPS) is 30, thus in a 1-minute game, there are around 1800 frames.

In order to test a common instance of teamwork, players were instructed in a common strategy and assigned roles of either *bait* or *shooter*. The *bait* tries to attract the fortress's attention by entering the inner hexagon where it is vulnerable to the fortress. When the fortress attempts to shoot at the bait, its shield lifts making it vulnerable. The other player in the role of *shooter* can now shoot at the fortress and destroy it.

There are some difference in observations and actions between human players and agent players. Human players observe the game screen (RGB image) at each frame, then hit or release the keys on keyboard to take actions. Agent players instead observe an array composed of the states (position, velocity, angle) of all the entities including the fortress and two players, and communicate their actions directly through the game engine.

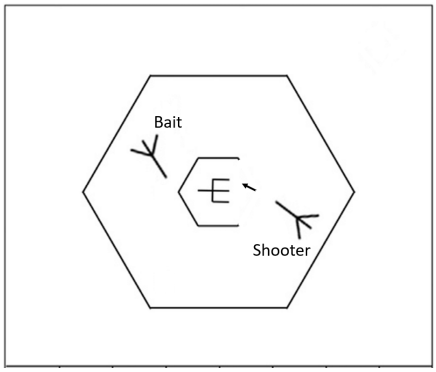

Figure 1: Sample TSF game screen (line drawing version, original screen is in black background). Spaceships are labeled as shooter and bait. Entity at center is the rotating fortress with the boarder around it as the shield. Activation region is the hexagon area around players' spaceships. Black arrow is a projectile emitted from the shooter towards the fortress. All the entities are within the rectangle map borders.

4 Adaptive Agent Architecture

In this section, we formulate our method for an adaptive agent architecture. First we introduce the exemplar policies library pre-trained by reinforcement learning or designed by rules for TSF. This library will be used as a standard baseline to identify human policies. Next, we introduce the similarity metric adopted in the architecture (i.e. cross-entropy metric) that measures the distance between human trajectory and exemplar policies in the library. Finally, we define the adaptive agent architecture given the estimated human policy according to the similarity metric.

4.1 Exemplar Policies Library

The exemplar policies library $\mathcal{L} = \mathcal{L}_{\mathcal{B}} \cup \mathcal{L}_{\mathcal{S}}$ consists of two sets of policies in bait ($\mathcal{B}$) and shooter ($\mathcal{S}$) roles, $\mathcal{L}_{\mathcal{B}}$ and $\mathcal{L}_{\mathcal{S}}$ respectively. Both bait policies and shooter policies are trained using a combination of RL and rule based behavior.

These exemplar policies can be divided into several main types: baits can be divided into three types (B1-B3, B4-B7, B8-B9) and shooters can be divided into two types (S1-S3, S4-S7). Below are the technical details of each type.

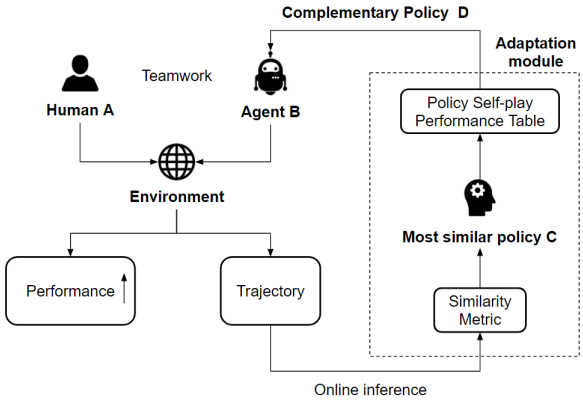

Figure 2: The flowchart of the proposed adaptive agent architecture. The adaptation module (in dotted boarder) takes the input of the trajectory at current timestamp, and then assigns the adaptive agent with new policy at next timestamp. The adaption procedure can be deployed in real-time (online).

Bait policy library $\mathcal{L}_\mathcal{B}$ To make these different bait policies diverse, we train them using different reward functions, inspired from human-human experiments where there were multiple ways to achieve good performance. The reward functions attempt to encode the desirable behavior of a bait agent. The bait agent is then trained using an RL algorithm to achieve an optimal behavior with respect to the given reward function. The bait library $\mathcal{L}_\mathcal{B}$ are composed of 9 bait policies. The goal for bait is to keep alive inside the activation region to make the fortress vulnerable from behind so that the shooter can grasp the opportunity to destroy the fortress from behind. In general, the bait has two *conflicting* objectives which it tries to balance. If the bait is inside the activation more time, it is more vulnerable and prone to getting killed by the fortress. However, bait's presence inside the activation region gives an opportunity for the shooter to attack the fortress from behind. Different bait agents try to balance these two conflicting objectives in different ways.

B1-B3 type policies are trained by A2C algorithm (Konda and Tsitsiklis 2000) to learn TURN action and use rules on THRUST action. The reward function of TURN learning is binary, encouraging the baits to stay inside the activation region. For the observation space of the bait policy, we convert the original Cartesian coordinate system to a new one, where the new origin is still at the fortress while the new positive Y-axis goes through the bait, which is shown to ease the training in RL for TSF. Then, we use the converted coordinates of bait position and two nearest missile positions to train the agent. The intuition is that bait is sensitive to the nearest shells to keep itself alive. The rule in THRUST action limits the maximum speed of the bait agents. By tuning the threshold in speed, we have policy B1-B3. B4-B7 policies are trained by RL in both TURN and THRUST actions. They share same reward function as B1-B3 using the same transformation in coordinate system. By using different RL algorithms from A2C (Konda and Tsitsiklis 2000), PPO (Schulman et al. 2017), to TRPO (Schulman et al. 2015), and different observation space (whether to perceive the shooter's position and velocity), we have policies B4-B7.

B8-B9 belong to the another set of policy using a different reward structure, learning TURN and THRUST by PPO algorithm (Schulman et al. 2017). B8 and B9 are designed by aggressive and defensive objectives, respectively. The reward structure used to train these 2 agents is composed of three parts: (1) "border reward" to encourage the agents to keep far away from the fortress, (2) "bearing reward" to encourage the agents to align itself directly towards the fortress when inside the activation region, (3) "death penalty" to discourage the agents from being killed by the fortress or hitting the border. Border reward encourages risk-averse behavior, while bearing reward risk-seeking behavior, and by controlling the coefficients among these three parts, we have agents B8-B9.

Shooter policy library $\mathcal{L}_\mathcal{S}$ The shooter policy library $\mathcal{L}_\mathcal{S}$ are composed of 7 shooter policies, with 4 of them mirror shooters that are purely rule-based, and 3 of them RL shooters that learn the TURN action by RL.

The mirror shooters are based on the prior knowledge of TSF game that a good shooter should have an *opposite* position against the Bait, which was observed in many successful human-human teams. Thus the mirror shooter tries to keep at opposite position to the current position of the bait (termed as *target position*) until it finds itself having a good chance to fire to destroy the fortress. By controlling the threshold of distance to the target position, we have agents S4-7.

The RL shooters' reward function takes the same team strategy of opposite positions, trained by DDQN (Van Hasselt, Guez, and Silver 2015) on TURN action. Specifically, the reward is designed to encourage the shooter to keep close to the outside the activation region when bait does not enter the region, and keep at the rear of the fortress when bait enters the region. S1-3 are different in max speed.

Self-play performance We evaluate the performance of each shooter-bait pairs in the exemplar policy library by self-play in TSF environment, and record the results in self-play performance table $\mathcal{P}$ in advance. The table $\mathcal{P}$ has rows with the number of bait policies in $\mathcal{L}_\mathcal{B}$ and columns with the number of shooter policies in $\mathcal{L}_\mathcal{S}$, with each entry the average performance of the bait-shooter pair. When applied to our policy library, table $\mathcal{P}$ is showed in Table 1.

On average, teams that consist of two static agents show significantly better performance (6.03) than human-human teams (2.60) reported in previous research (Li et al. 2020b). This indicates that most of our agent pairs, including both RL-based and rule-based agents, have a super-human performance in TSF which benefits from the design of reward function and rules.

Similar to human-human teams, agent-agent teams also show complementary policy pairs that work extremely well with each other. An example would be S4-S7 (mirror shooters) who yield a dominant performance when pairing with most of the baits except for B8 and B9. While for specific bait policies such as B8 and B9, the best teammate would be S2 or S3 (RL shooters) in stead of the more "optimal" S4-S7. We could tell from the self-play table that the space

of reasonable policies in TSF game is indeed diverse, and there are more than one path towards good team dynamics and team performance. This confirms again the necessity of introducing real-time adaptive agents in human-agent teams. Thus we build the adaptive agent framework based upon this self-play table in the next subsections.

	S1	S2	S3	**S4**	**S5**	**S6**	**S7**
B1	5.1	5.7	5.0	5.1	4.9	4.5	3.9
B2	6.5	7.0	6.0	7.6	7.6	7.5	6.8
B3	5.9	6.7	5.8	8.0	8.1	8.2	7.9
B4	6.4	7.1	5.9	7.5	7.6	7.3	7.3
B5	6.3	7.1	6.2	6.8	6.8	6.4	6.2
B6	6.2	7.1	6.1	7.8	7.8	7.4	7.0
B7	6.2	7.0	6.2	7.8	7.8	8.0	7.8
B8	4.3	5.3	**5.4**	3.1	2.8	3.0	3.0
B9	4.9	5.7	5.5	2.3	2.1	2.1	1.8

Table 1: Self-play agent performance table $\mathcal{P}$. Each row is for one bait agent named Bi in $\mathcal{L}_\mathcal{B}$ (i=1 to 9), and each column is for one shooter agent named Sj in $\mathcal{L}_\mathcal{S}$ (j=1 to 7). Each entry is computed by per-minute team performance (number of fortress kills) of the corresponding pair. We segment the tables to group same type of agents, and mark the "optimal" bait and shooter agents in bold.

4.2 Similarity Metric

Now we introduce the **cross-entropy metric** (CEM) as the similarity metric used in this architecture. Cross-entropy, well-known in information theory, can measure the (negative) distance between two policies π_1, π_2:

$$\mathrm{CEM}(\pi_1, \pi_2) := \mathbb{E}_{s,a\sim\pi_1}[\log \pi_2(a|s)] \tag{1}$$

where $\pi_1(\cdot|s), \pi_2(\cdot|s)$ are action distributions given state s. This is actually the training objective of behavior cloning (Bain 1995) to expert policy π_1, i.e., $\max_{\pi_2} \mathrm{CEM}(\pi_1, \pi_2)$, which is to maximize the log-likelihood of expert actions in agent policy π_2 given a collection of expert state-actions. Thus the larger the $\mathrm{CEM}(\pi_1, \pi_2)$, the more similar π_1 is to π_2.

If we know the policy π_2, and are able to obtain state-action samples from π_1, then we can estimate cross-entropy $\mathrm{CEM}(\pi_1, \pi_2)$ by Monte Carlo sampling. That is to say, under the assumption above, policy π_1 can be unknown to us. In human-agent teaming, human policy π_H cannot be observed but the state-action pairs generated by the human policy can be easily obtained, and agent policy π_A is designed by us, as programmers, thus known to us.

Therefore, we can leverage CEM as the similarity metric: given a sliding window of frames that record the observed behavior of the human policy π_H, we can estimate the cross-entropy between a human policy π_H and any known agent policy π_A by the following formula:

$$\frac{1}{T}\sum_{t=1}^{T} \log \pi_A(a_t|s_t), \quad \text{where } (s_t, a_t)_{t=1}^{T} \sim \pi_H \tag{2}$$

where $(s_t, a_t)_{t=1}^{T}$ are the sequential state-action pairs from human policy, T is the window size, which is a hyperparameter to be tuned.

4.3 Adaptive Agent Architecture

The prerequisite for the architecture is the exemplar policies library $\mathcal{L}$ introduced in the Sec. 4.1 and the self-play table $\mathcal{P}$ of the library to translate human-agent performance in the adaptation process.

Figure 2 shows the overall flowchart of our adaptive agent framework. When the game starts and a new human player A starts to play as one pre-specified role $R_1 \in \{\mathcal{B}, \mathcal{S}\}$ in TSF, the adaptive agent framework will first randomly assign a policy B from the library $\mathcal{L}_{R_2}$ in teammate role R_2 such that $\{R_1, R_2\} = \{\mathcal{B}, \mathcal{S}\}$, and keep track of the joint trajectories (state-action sequences) and record them into memory.

The adaptation process is as follows. As we maintain the latest human trajectories of a pre-specified window size, and we first use the data to compute the similarity by cross-entropy metric between the human trajectory and any of exemplar policies in the library $\mathcal{L}_{R_1}$ with same role. Then we figure out the most similar policy $C \in \mathcal{L}_{R_1}$ to the human trajectory, and look up the performance table $\mathcal{P}$ to find the optimal complementary policy $D \in \mathcal{L}_{R_2}$ to the predicted human policy type C. Finally, we assign the agent D as the complementary policy at next timestamp with the human player.

The adaptation process on the exemplar policies selection is based on the following assumption: if the human policy A with role R_1 is similar to one exemplar policy $C \in \mathcal{L}_{R_1}$ within some threshold, then the human policy A will have similar team performance with teammates as C, i.e., if C performs better with $D \in \mathcal{L}_{R_2}$ than $E \in \mathcal{L}_{R_2}$, so does A. This enables us to adapt the agent policy in real-time by the recent data without modeling the human policy directly.

5 Human-Agent Teaming Experiments

In this section, we first introduce our experiment design for human-agent teaming, then evaluate the human-agent performance when paired with static policy agents (introduced in Sec. 4.1) and proposed adaptive agents (introduced in Sec. 4.3).

By analyzing the collected human-agent data, we aim to answer the following motivated questions:

1. How are human players' policies compared to agent policies in our library?
2. Is our adaptive agent architecture capable of identifying human policies and predicting team performance for human-agent teams?
3. Do our adaptive agents perform better than static policy agents in human-agent teams?

5.1 Experimental Design

We recruited participants from Amazon Mechanical Turk for our human-agent experiments. They were paid USD 2 for participating in the 15-min online study. Participants were randomly assigned a role of either shooter or bait and then teamed with artificial agents in the corresponding role to

play Team Space Fortress. Each participant would need to complete five sessions of data collection with three 1-min game trials in each session. Participants teamed with different agent variants between sessions in a random sequence. The five variants were selected from our static agent library $\mathcal{L}$. When selecting these designated agents, we balanced the performance in self-play table and the diversity by considering different training methods and reward functions. Specifically, we select $\{$B3, B6, B7, B8, B9$\}$ as tested static baits and $\{$S1, S2, S3, S4, S7$\}$ as tested static shooters. In the dataset of human and static agent teams, we got 25 valid data points from human shooters and 29 valid data from human baits.

5.2 Results

Policy space representation To quantify the relationship between real human policies in the experiments and agent policies in the library, we leveraged a similarity embedding by comparing the distance between the collected human trajectories and agent policies using CEM measurement (see Sec. 4.2). This provides us with a high-dimensional policy space based on agent policies in our library. Specifically, CEM was employed to generate the average log-probability of state-action pairs in a human trajectory coming from a certain agent policy. We could then construct a similarity vector for each trajectory with the dimensions equal to the number of policies in the library. The value in each dimension represents the similarity distance from human trajectories to a certain agent policy. Then, we applied a principal component analysis (PCA) based on the log-probability dataset to project the high-dimensional policy space into a 2D plane for a better visualization. The two primary components left explain more than 99% of the variance.

Fig. 3 illustrates the human policies in static agent dataset. We could get following qualitative insights from the illustration: 1) the learnt similarity embedding separates different human policies well, 2) reinforcement learning policies are homogeneous (red nodes in the bottom-left corner) while the rule-based policies are a bit off (red nodes in the upper-left corner). 3) the distribution of human policies correlates with their team performance in that players to the left tend to have better team performance (colored nodes to the left are larger in size). Those findings align with our expectations and validate the proposed adaptive agent architecture. In the following analysis, we will quantify them based on the CEM measurement and the similarity embedding.

Human policy identification In the proposed adaptive agent architecture, our model infers human policy by classifying it as the most similar policy in the library based on CEM measurement, then assigns the agent with the corresponding complementary policy in the self-play table. One way of verifying this method is to see if human-agent teams performed better when the predicted human policy was closer to the complementary match in the self-play table $\mathcal{P}$. Assuming each human maintains a consistent policy over the course of interaction when paired with a specific teammate with static policy, we could then calculate, for each human-agent pair, the similarity between human policy and

Figure 3: Policy representations of each human baits (left) and shooters (right) in the static agent dataset (after PCA dimension reduction). Each colored node in the figures represents the average policy of a human player, while the size of which indicates his average team performance. Red nodes are reference points of baseline agent polices.

the optimal agent policy for the agent that the human was playing with.

This "similarity to optimal" quantifies the degree to which a human player is similar to the optimal policy given an agent teammate in our architecture. Correlation analysis shows that "similarity to optimal" is positively correlated with team performance in both bait ($r = 0.636, p = .0002$) and shooter ($r = 0.834, p < .0001$) groups. This result indicates that the complementary policy pairs we found in agent-agent self-play can be extended to human-agent teams, and our proposed architecture is able to accurately identify human policy types and predict team performance.

Furthermore, our model could also infer human policies in real-time. This is to say, even within the same team, humans might also take different sub-policies as their mental model of the team state evolves over time (Salas, Cooke, and Rosen 2008). We could take the log-probabilities generated by CEM as time series data to capture the online adaptation process of humans over the course of interaction at each timestamp.

An example visualization is shown in Fig. 4 where curves represent the log-probabilities of each agent policy over the course of interaction. We can tell from the graph that in the segments of a specific trial, the human trajectories were inferred to reflect different policies, although the average log-probability would still be in favor of B8. Those findings motivate us to test an online adaptive agent using a sliding time window to capture the human policy shifts in real-time.

5.3 Pilot experiment with adaptive agents

In previous experiment and analysis, we validated our proposed architecture on the static agent dataset. Finally, we conducted a pilot experiment to measure the performance of adaptive agents in HATs by pairing them with human players.

In the adaptive agent experiment, adaptive agent uses the CEM similarity metric (Sec. 4.2) to identify the policy most similar to the human behavior over a fixed number of recent preceding game frames. The frames were tracked us-

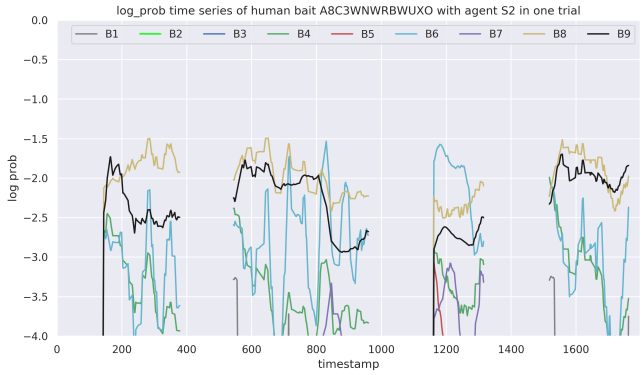

Figure 4: The log-probability curves of one human policy generated by CEM. Data is from one specific 1-min trial of a human bait paired with agent S2. We segment the trial into several episodes, each of which starts with the bait entering the activation region and ends with the team killing the fortress. The curves with same color represent the same agent policy for inference.

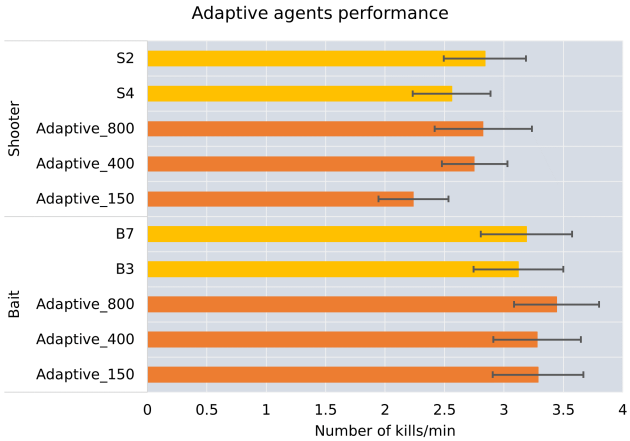

Figure 5: Human-agent team performance when humans paired with adaptive or static agent policies. Error bars represent one standard error away from the mean.

ing a sliding window, the size of which was adjusted during the hyperparameter tuning phase of experimentation. To perform the adaptation procedure, in each frame, after identifying the most similar agent to the human teammate, the agent referenced the self-play table to select the policy that would best complement the teammate's estimated policy.

In this round of experiment, the five variants including three values of the window size hyperparameter (T in Eq. 2) for the adaptive agent (150, 400, 800 frames) and two best-performed static agent policy (representing the extreme condition of 0 window size where the adaptive agent becomes static). Besides that, all experimental settings are the same as in static agent experiment. We got in total, 22 valid data points from human shooters and 25 valid data from human baits.

Fig. 5 shows the average team performance of HATs when human players were paired with either static or adaptive agents. We could see from the figure that adaptive agents (marked in orange) have slightly better performance than static agents (marked in yellow), although not statistically significant. In addition, adaptive agents with longer time window (e.g. 800 frames) tend to have better performance in HATs since they accumulate more evidence for human policy inference. However, a larger sample size and and better hyper-parameter tuning might be necessary for future research to confirm the advantage of adaptive agents in HATs.

6 Conclusion and Future Work

In this paper, we proposed a novel adaptive agent framework in human agent teaming (HAT) based on the cross-entropy similarity measure and a pre-trained static policy library. The framework was inspired by human teamwork research, which illustrates important characteristics of teamwork such as the existence of complementary policies, influence of adaptive actions on team performance, and the dynamic human policies in cooperation (Li et al. 2020a,b). Those findings motivate us to introduce an online adaptive agents into HATs in order to maximize the team performance even when given unknown human teammates. The proposed framework adopts a human-model-free method to reduce the computational cost in real-time deployment and make the pipeline more generalizable to diverse task settings and human constraints.

The specific task scenario studied in this paper, i.e. Team Space Fortress, is a nontrivial cooperative game which requires sequential decision-making and real-time cooperation with heterogeneous teammates. We evaluated the validity of proposed adaptive agent framework by running human-agent experiments. Results show that our adaptive agent architecture is able to identify human policies and predict team performance accurately. We constructed a high-dimensional policy space based on exemplar policies in a pre-trained library and leveraged it as a standard and reliable way to categorize and pair human policies. The distance between human policy and the optimal complementary for his/her teammate is shown to be positively correlated with team performance, which confirms the validity of our proposed framework. In additional, we found that human players showed diverse policies in HAT (1) when paired with different teammates (2) over the course of interaction within the same team. These findings point out that we cannot simply impose strong assumptions on humans, e.g. optimality, consistency, and unimodality, prevalent in human-model-based settings. Thus, we employed an online inference mechanism to identify the human policy shifting during the course of interaction and adapt the agent policy in real time.

As for future directions, we would like to enrich the static agent library by introducing novel policies such as imitation learning agents that learn from human demonstrations. A larger coverage in the policy space of exemplar policies library could lead to a more accurate estimation of human policy and a better selection of complementary policy.

Acknowledgments

This research was supported by a reward W911NF-19-2-0146 and AFOSR/AFRL award FA9550-18-1-0251.

References

Agarwal, A.; Hope, R.; and Sycara, K. 2018. Challenges of context and time in reinforcement learning: Introducing Space Fortress as a benchmark. *arXiv preprint arXiv:1809.02206* .

Agrawal, S.; and Williams, M.-A. 2017. Robot authority and human obedience: A study of human behaviour using a robot security guard. In *Proceedings of the companion of the 2017 ACM/IEEE international conference on human-robot interaction*, 57–58.

Albrecht, S. V.; and Stone, P. 2018. Autonomous agents modelling other agents: A comprehensive survey and open problems. *Artificial Intelligence* 258: 66–95.

Bain, M. 1995. A Framework for Behavioural Cloning. In *Machine Intelligence 15*, 103–129.

Bajcsy, A.; Losey, D. P.; O'Malley, M. K.; and Dragan, A. D. 2018. Learning from physical human corrections, one feature at a time. In *Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction*, 141–149.

Barrett, S.; and Stone, P. 2015. Cooperating with Unknown Teammates in Complex Domains: A Robot Soccer Case Study of Ad Hoc Teamwork. In *AAAI*, volume 15, 2010–2016. Citeseer.

Bradshaw, J. M.; Feltovich, P.; and Johnson, M. 2011. Human-agent interaction .

Chen, J. Y.; and Barnes, M. J. 2012. Supervisory control of multiple robots: Effects of imperfect automation and individual differences. *Human Factors* 54(2): 157–174.

Chen, J. Y.; and Barnes, M. J. 2014. Human–agent teaming for multirobot control: A review of human factors issues. *IEEE Transactions on Human-Machine Systems* 44(1): 13–29.

Chen, J. Y.; Lakhmani, S. G.; Stowers, K.; Selkowitz, A. R.; Wright, J. L.; and Barnes, M. 2018. Situation awareness-based agent transparency and human-autonomy teaming effectiveness. *Theoretical issues in ergonomics science* 19(3): 259–282.

Demir, M.; McNeese, N. J.; and Cooke, N. J. 2017. Team synchrony in human-autonomy teaming. In *International Conference on Applied Human Factors and Ergonomics*, 303–312. Springer.

Fan, X.; McNeese, M.; Sun, B.; Hanratty, T.; Allender, L.; and Yen, J. 2009. Human–agent collaboration for time-stressed multicontext decision making. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans* 40(2): 306–320.

Fisac, J. F.; Bronstein, E.; Stefansson, E.; Sadigh, D.; Sastry, S. S.; and Dragan, A. D. 2019. Hierarchical game-theoretic planning for autonomous vehicles. In *2019 International Conference on Robotics and Automation (ICRA)*, 9590–9596. IEEE.

Green, C. S.; and Bavelier, D. 2006. Enumeration versus multiple object tracking: The case of action video game players. *Cognition* 101(1): 217–245.

Guss, W. H.; Codel, C.; Hofmann, K.; Houghton, B.; Kuno, N.; Milani, S.; Mohanty, S.; Liebana, D. P.; Salakhutdinov, R.; Topin, N.; et al. 2019. The minerl competition on sample efficient reinforcement learning using human priors. *arXiv preprint arXiv:1904.10079* .

Harbers, M.; Jonker, C.; and Van Riemsdijk, B. 2012. Enhancing team performance through effective communication.

Haynes, T.; and Sen, S. 1996. Co-adaptation in a team. *International Journal of Computational Intelligence and Organizations* 1(4): 1–20.

Konda, V. R.; and Tsitsiklis, J. N. 2000. Actor-critic algorithms. In *Advances in neural information processing systems*, 1008–1014.

Kozlowski, S. W.; and Chao, G. T. 2018. Unpacking team process dynamics and emergent phenomena: Challenges, conceptual advances, and innovative methods. *American Psychologist* 73(4): 576.

Kozlowski, S. W.; Grand, J. A.; Baard, S. K.; and Pearce, M. 2015. Teams, teamwork, and team effectiveness: Implications for human systems integration. .

Kozlowski, S. W.; and Klein, K. J. 2000. A multilevel approach to theory and research in organizations: Contextual, temporal, and emergent processes. .

Kurin, V.; Nowozin, S.; Hofmann, K.; Beyer, L.; and Leibe, B. 2017. The atari grand challenge dataset. *arXiv preprint arXiv:1705.10998* .

Levine, S. J.; and Williams, B. C. 2018. Watching and acting together: Concurrent plan recognition and adaptation for human-robot teams. *Journal of Artificial Intelligence Research* 63: 281–359.

Li, H.; Hughes, D.; Lewis, M.; and Sycara, K. 2020a. Individual adaptation in teamwork. In *Proceedings of the 42nd Annual Conference of the Cognitive Science Society*, In Press.

Li, H.; Milani, S.; Krishnamoorthy, V.; Lewis, M.; and Sycara, K. 2019. Perceptions of Domestic Robots' Normative Behavior Across Cultures. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, 345–351.

Li, H.; Ni, T.; Agrawal, S.; Hughes, D.; Lewis, M.; and Sycara, K. 2020b. Team Synchronization and Individual Contributions in Coop-Space Fortress. In *Proceedings of the 64th Human Factors and Ergonomics Society Annual Meeting*, In Press.

Mané, A.; and Donchin, E. 1989. The space fortress game. *Acta psychologica* 71(1-3): 17–22.

McNeese, N. J.; Demir, M.; Cooke, N. J.; and Myers, C. 2018. Teaming with a synthetic teammate: Insights into human-autonomy teaming. *Human factors* 60(2): 262–273.

Morgan Jr, B. B.; et al. 1986. Measurement of Team Behaviors in a Navy Environment. Final Report. .

OpenAI; Berner, C.; Brockman, G.; Chan, B.; Cheung, V.; Debiak, P.; Dennison, C.; Farhi, D.; Fischer, Q.; Hashme, S.; Hesse, C.; Józefowicz, R.; Gray, S.; Olsson, C.; Pachocki, J.; Petrov, M.; de Oliveira Pinto, H. P.; Raiman, J.; Salimans, T.; Schlatter, J.; Schneider, J.; Sidor, S.; Sutskever, I.; Tang, J.; Wolski, F.; and Zhang, S. 2019. Dota 2 with Large Scale Deep Reinforcement Learning. *arXiv preprint arXiv:1912.06680* .

Parasuraman, R.; and Riley, V. 1997. Humans and automation: Use, misuse, disuse, abuse. *Human factors* 39(2): 230–253.

Reddy, S.; Dragan, A.; and Levine, S. 2018. Where do you think you're going?: Inferring beliefs about dynamics from behavior. In *Advances in Neural Information Processing Systems*, 1454–1465.

Sadigh, D.; Dragan, A. D.; Sastry, S.; and Seshia, S. A. 2017. Active Preference-Based Learning of Reward Functions. In *Robotics: Science and Systems*.

Sadigh, D.; Landolfi, N.; Sastry, S. S.; Seshia, S. A.; and Dragan, A. D. 2018. Planning for cars that coordinate with people: leveraging effects on human actions for planning and active information gathering over human internal state. *Autonomous Robots* 42(7): 1405–1426.

Sadigh, D.; Sastry, S. S.; Seshia, S. A.; and Dragan, A. 2016. Information gathering actions over human internal state. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 66–73. IEEE.

Salas, E.; Cooke, N. J.; and Rosen, M. A. 2008. On teams, teamwork, and team performance: Discoveries and developments. *Human factors* 50(3): 540–547.

Salas, E.; Sims, D. E.; and Burke, C. S. 2005. Is there a "big five" in teamwork? *Small group research* 36(5): 555–599.

Scholtz, J. 2003. Theory and evaluation of human robot interactions. In *36th Annual Hawaii International Conference on System Sciences, 2003. Proceedings of the*, 10–pp. IEEE.

Schulman, J.; Levine, S.; Abbeel, P.; Jordan, M.; and Moritz, P. 2015. Trust region policy optimization. In *International conference on machine learning*, 1889–1897.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347* .

Van Hasselt, H.; Guez, A.; and Silver, D. 2015. Deep reinforcement learning with double q-learning. *arXiv preprint arXiv:1509.06461* .

van Zoelen, E. M.; Cremers, A.; Dignum, F. P.; van Diggelen, J.; and Peeters, M. M. 2020. Learning to Communicate Proactively in Human-Agent Teaming. In *International Conference on Practical Applications of Agents and Multi-Agent Systems*, 238–249. Springer.

Vinyals, O.; Babuschkin, I.; Czarnecki, W. M.; Mathieu, M.; Dudzik, A.; Chung, J.; Choi, D. H.; Powell, R.; Ewalds, T.; Georgiev, P.; et al. 2019. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature* 575(7782): 350–354.

Vinyals, O.; Ewalds, T.; Bartunov, S.; Georgiev, P.; Vezhnevets, A. S.; Yeo, M.; Makhzani, A.; Küttler, H.; Agapiou, J.; Schrittwieser, J.; et al. 2017. Starcraft ii: A new challenge for reinforcement learning. *arXiv preprint arXiv:1708.04782* .

Zakershahrak, M.; Sonawane, A.; Gong, Z.; and Zhang, Y. 2018. Interactive plan explicability in human-robot teaming. In *2018 27th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, 1012–1017. IEEE.