SYMMETRY-BREAKING AUGMENTATIONS FOR AD HOC TEAMWORK

Ravi Hammond [1 2 3], **Dustin Craggs** [3], **Mingyu Guo** [3], **Jakob Foerster** [1] **& Ian Reid** [3 4]
[1]Foerster Lab for AI Research, University of Oxford
[2]Autonomous Intelligent Machines and Systems, University of Oxford
[3]Australian Institute for Machine Learning, University of Adelaide
[4]Mohamed bin Zayed University of Artificial Intelligence
ravi.hammond@gmail.com

ABSTRACT

In dynamic collaborative settings, for artificial intelligence (AI) agents to better align with humans, they must adapt to novel teammates who utilise unforeseen strategies. While adaptation is often simple for humans, it can be challenging for AI agents. Our work introduces *symmetry-breaking augmentations* (SBA) as a novel approach to this challenge. By applying a symmetry-flipping operation to increase *behavioural diversity* among training teammates, SBA encourages agents to learn robust responses to unknown strategies, highlighting how social conventions impact human-AI alignment. We demonstrate this experimentally in two settings, showing that our approach outperforms previous ad hoc teamwork results in the challenging card game Hanabi. In addition, we propose a general metric for estimating symmetry dependency amongst a given set of policies. Our findings provide insights into how AI systems can better adapt to diverse human conventions and the core mechanics of alignment.

1 INTRODUCTION

Humans and AI agents alike employ a diverse range of *conventions* when interacting with one another. These conventions facilitate communication and coordination, which are crucial for effective teamwork in many multi-agent settings. They range in complexity from knowing which side of the road to drive on to coordinating using a shared language. For agents to effectively coordinate with others—particularly in contexts where strategic conventions vary—they must develop an understanding of these conventions, especially when coordination failures can lead to severe consequences such as vehicle collisions.

Figure 1: Augmenting conventions of other agents. The driver stops at red and drives on green (top), but with SBA, our agent *sees* the driver stopping and starting with many colours (bottom).

The challenge of aligning to previously unseen teammates has been formalised as ad hoc teamwork (AHT) (Stone et al., 2010). One method of training and evaluating an AHT agent is to use reinforcement learning (RL) to learn a *best response* (BR) to a training population of teammates (Fudenberg & Tirole, 1991) (more formally defined in Section 2.2) and then evaluate against a test set of held-out agents. The key challenge, therefore, is *generalising* to unseen policies after only being exposed to a subset of possible strategies during training, a problem that reflects the practical difficulties of coordinating effectively in diverse human-AI scenarios.

Furthermore, due to symmetries, the space of possible conventions is often combinatorial even in simple environments, making it computationally challenging to compute the *best response* even if the training population were sufficiently large to cover the distribution. Children reduce this computational burden by transferring existing knowledge to equivalent symmetries through higher-level reasoning (Beasty, 1987). In contrast, chimpanzees have been observed to fail these symmetry

tests (Dugdale & Lowe, 2000). AI agents also struggle with this, from computer vision models that fail to generalise to different coloured images (Galstyan et al., 2022) to cooperative agents that cannot recognise when teammates are using symmetry-equivalent conventions (Hu et al., 2020).

To address this within a human-AI alignment framework, we introduce *symmetry-breaking augmentations* (SBA), a policy augmentation technique that alters the behaviour of agents in the training pool by making them break symmetries in various ways. SBA acts as an operator that can be applied to other agents in an environment, flipping their behaviours along environmental symmetries. When symmetries are present, this combinatorially amplifies the range of conventions to which the ad hoc agent is exposed. Thus, even with a relatively small training population, SBA enables RL agents to learn to adapt to a much more diverse set of conventions during training. This technique not only improves test-time performance but also ensures the agent acts predictably in relation to environmental symmetries, making it easier for humans to adapt to the its behaviour.

For example, consider a traffic conductor learning to direct drivers. Initially, the conductor is unaware of the colour conventions drivers may use for stop and go. As shown in Figure 1, SBA can be used to create new training experiences by altering the observed colours. This prevents the conductor from overfitting to the potentially limited conventions of its training partners and provides experiences that enable adaptation to different conventions in the future.

SBA is closely related to zero-shot coordination (ZSC) approaches such as Other Play (Hu et al., 2020) and Equivariant Networks (Muglich et al., 2022a). However, since these approaches aim to learn policies that are invariant to environmental symmetries, they are not directly applicable to the AHT setting, where an agent must be able to coordinate with teammates that *do* use symmetry-breaking conventions. SBA also differs from population-based ZSC and AHT approaches (Lupu et al., 2021; Rahman et al., 2023a) in that it aims to generalise from a provided set of partners rather than generating a sufficiently diverse set of partners from scratch. SBA could be applied in conjunction with these population-based approaches to further increase the diversity of policies.

Since, SBA is most effective when the agents in the training population use symmetry-breaking conventions, we introduce the *Augmentation Impact* (AugImp) metric, which measures the extent to which a specific augmentation alters policies within a population. This enables us to analyse a population prior to training to predict how effective SBA will be at improving AHT performance.

We demonstrate how SBA improves performance in both a simple matrix game and the card game Hanabi. In Hanabi, independently trained agents frequently develop incompatible conventions, even when trained using the same algorithm (Hu et al., 2020). Since Humans also use a diverse range of conventions when playing Hanabi, a successful AHT Hanabi agent needs to quickly infer and adapt to the conventions of its teammates while avoiding triggering unexpected responses. We experiment by training an AHT agent with existing Hanabi populations of simplified action decoder (SAD) (Hu & Foerster, 2019) and independent Q-learning (IQL) (Tan, 1993) policies. We show that in Hanabi, SBA leads to improvements of up to 17% in game score when adapting to previously unseen teammates from the same population. Additionally, we show that SBA improves performance when generalising outside of the training distribution to populations of Other Play (OP) (Hu et al., 2020) and Off-Belief Learning (OBL) agents (Hu et al., 2021).

To summarise, our contributions are:

- SBA, a general method of augmenting a training population for AHT that amplifies the diversity of conventions the agent is exposed to during training.
- A general metric that measures the effectiveness of policy augmentation techniques for AHT by assessing how much they diversify behaviours in a training population.
- Evaluation of SBA in Hanabi, demonstrating state-of-the-art performance for AHT.

2 BACKGROUND

2.1 DEC-POMDPS

We formalise the cooperative multi-agent setting as a decentralised partially-observable Markov Decision Process (Dec-POMDP) (Nair et al., 2003). The Dec-POMDP, G, is a 9-tuple $(\mathcal{N}, \mathcal{S}, \{\mathcal{A}^i\}_{i=1}^n, \{\mathcal{O}^i\}_{i=1}^n, \mathcal{T}, \mathcal{U}, \mathcal{R}, T, \gamma)$, with finite sets $\mathcal{N}, \mathcal{S}, \{\mathcal{A}^i\}_{i=1}^n, \{\mathcal{O}^i\}_{i=1}^n$, respectively de-

noting the set of agents, states, actions, and observations, where i denotes the set pertaining to agent $i \in \mathcal{N} = \{1, \dots, n\}$. $\mathcal{A}^i$ and $\mathcal{O}^i$ are the set of actions and observations for agent i, and $a^i \in \mathcal{A}^i$ and $o^i \in \mathcal{O}^i$ are a specific action and observation that agent i may take and observe. We also write $\mathcal{A} = \times_{i=1}^{n} A^i$ and $\mathcal{O} = \times_{i=1}^{n} O^i$, as the sets of joint actions and observations, respectively. $s_t \in \mathcal{S}$ is the state at time t, and $a_t \in \mathcal{A}$ is the joint action of all agents at time t, which changes the state according to the transition distribution $s_{t+1} \sim \mathcal{T}(\cdot|s_t, a_t)$. The subsequent joint observation of the agents, $o_{t+1} \in \mathcal{O}$, is distributed according to $o_{t+1} \sim \mathcal{U}(\cdot|s_{t+1}, a_t)$, where $\mathcal{U} = \times_{i=1}^{n} \mathcal{U}^i$. At time t, the joint observation o_t is appended to the trajectory $\tau_t = (o_1, a_1, \dots, o_{t-1}, a_{t-1}, o_t)$, and each agent i individually decides its own action a_t^i based on its policy $\pi^i(a_t^i|\tau_t^j)$, which is conditioned on its action-observation history (AOH) $\tau_t^i = (o_1^i, a_1^i, \dots, o_{t-1}^i a_{t-1}^i, o_t^i)$. π^i represents agent i's component of the decentralized joint policy $\pi \in \Pi$, where Π is the set of all possible joint-policies in the environment G. When G transitions to state s_{t+1}, all agents receive a common reward $r_{t+1} \in \mathbb{R}$ according to the distribution $r_{t+1} \sim \mathcal{R}(\cdot|s_{t+1}, a_t)$. The behaviour of a joint-policy π is characterised by the distribution of trajectories τ it produces, and, taking into account the time horizon T and discount factor $\gamma \in [0, 1]$, is optimal if it maximises the expected return:

$$J(\pi) = \mathbb{E}_{\tau \sim \pi}[\sum_{t=1}^{T} \gamma^{t-1} r_t]. \tag{1}$$

2.2 Ad Hoc Teamwork

Ad hoc teamwork (AHT) is the problem of creating an agent that is able to collaborate effectively with a group of novel teammates. This has been a long-standing challenge in the field of artificial intelligence (Stone et al., 2010; Bard et al., 2020).

We use π_A to represent the AHT agent, and π_j for the teammate's joint-policy. To evaluate the performance of our AHT joint-policy $\pi_A = (\pi_A^1, \dots, \pi_A^n)$ and the teammate joint-policy π_j, using Equation 1, we obtain the average expected AHT return by matching each individual component of π_A, i.e. π_A^i, with all other $n-1$ components of π_j, i.e. π_j^i. This objective is formalised by:

$$J_{AHT}(\pi_A, \pi_j) = \frac{1}{n}\big(J(\pi_A^1, \pi_j^2, \dots, \pi_j^n) + \dots + J(\pi_j^1, \dots, \pi_j^{n-1}, \pi_A^n)\big), \tag{2}$$

Our AHT agent π_A learns a best-response π_A^* to a training set $\Pi^{train} \in \Pi$ by interacting with each policy $\pi_j \in \Pi^{train}$, and maximising Equation 2. We formally define this objective as:

$$\pi_A^*(\Pi^{train}) = \underset{\pi_A}{\text{argmax}}\, \mathbb{E}_{\pi_j \sim \Pi^{train}}[J_{AHT}(\pi_A, \pi_j)], \tag{3}$$

where π_j is sampled uniformly from Π^{train}. The learned AHT policy, $\pi_A^*(\Pi^{train})$, is then evaluated using the robustness measure, $M_{\Pi^{eval}}(\pi_A^*(\Pi^{train}))$, which evaluates Equation 2 while interacting with previously unseen policies from the evaluation policy set $\Pi^{eval} \in \Pi$. This measure is formally given by:

$$M_{\Pi^{eval}}(\pi_A^*(\Pi^{train})) = \mathbb{E}_{\pi_j \sim \Pi^{eval}} J_{AHT}(\pi_A^*(\Pi^{train}), \pi_j), \tag{4}$$

where π_j is sampled uniformly from Π^{eval}.

2.3 Equivalence Mappings

To improve coordination with unseen teammates in the Dec-POMDP setting, domain knowledge can be exploited to increase the variety of conventions present in the training set. To achieve this we use a class of *equivalence mappings* (symmetries) (Hu et al., 2020), Φ, for a given Dec-POMDP G, such that each $\phi \in \Phi$ is an automorphism of $\mathcal{S}$, $\mathcal{A}$, and $\mathcal{O}$ onto itself, and leaves G unchanged up to relabeling such that the environment dynamics and rewards function stay the same:

$$\begin{aligned} \phi \in \Phi \iff & \mathcal{T}(\phi(s_{t+1})|\phi(s_t), \phi(a_t)) = \mathcal{T}(s_{t+1}|s_t, a_t) \\ & \wedge \mathcal{R}(\phi(r_{t+1})|\phi(s_{t+1}), \phi(a_t)) = \mathcal{R}(r_{t+1}|s_{t+1}, a_t) \\ & \wedge \mathcal{U}^i(\phi(o_{t+1}^i)|\phi(s_{t+1}), \phi(a_t)) = \mathcal{U}^i(o_{t+1}^i|s_{t+1}, a_t) \\ & \text{where equalities apply } \forall s_{t+1}, s_t \in \mathcal{S}, a_t \in \mathcal{A}, i \in \mathcal{N}. \end{aligned} \tag{5}$$

For ease of notation, ϕ is shorthand for

$$\phi \in \Phi = \{\phi_{\mathcal{S}}, \phi_{\mathcal{A}}, \phi_{\mathcal{O}}\}, \tag{6}$$

where each $\phi \in \Phi$ acts on trajectories as

$$\phi(\tau_t) = (\phi(o_0), \phi(a_0), \ldots, \phi(a_{t-1}), \phi(o_t)), \tag{7}$$

and acts on policies as

$$\hat{\pi} = \phi(\pi) \iff \hat{\pi}(\phi(a)|\phi(\tau)) = \pi(a|\tau). \tag{8}$$

Policies π, $\hat{\pi}$ in Equation 8 are said to be symmetry-equivalent to one another with respect to ϕ. For every symmetry operator ϕ in the automorphism group Φ, there exists an inverse operator $\phi^{-1} \in \Phi$ such that $\phi \circ \phi^{-1} = \phi^{-1} \circ \phi = e$, where e is the identity automorphism of Φ, and $\circ$ denotes function composition. Illustrated in Figure 2, using ϕ, the augmented policy $\hat{\pi}$ experiences a symmetrically-equivalent version of τ_t, and its actions are converted back to their original mapping with ϕ^{-1}. While OP uses ϕ to prevent symmetry-breaking conventions for ZSC, our work applies ϕ to the AHT setting, improving robustness (Equation 10) by increasing the variety of conventions present in the training set Π^{train}.

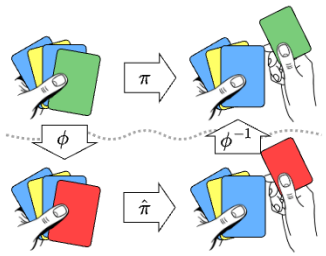

Figure 2: The ϕ operator converts green observations to red (left), and ϕ^{-1} inversely converts red actions back to green (right). In this game red and green are symmetrically-equivalent, so the application of ϕ and ϕ^{-1} leaves the game unchanged up to relabelling.

3 Symmetry-Breaking Augmentations

One of the biggest challenges in AHT is predicting the conventions that test-time policies will use and determining how a training population should be selected so that its best response is robust to these conventions. In the following, we introduce SBA, a method that addresses this problem by augmenting the training population through the random matching of the AHT agent with symmetry-equivalent policies of training teammates. We will discuss SBA both as a formal method and as a scalable algorithm-agnostic framework applicable to the deep RL setting.

3.1 SBA Objective

We start by introducing the SBA learning rule which uses the set of equivalence mappings $\phi \in \Phi$ from Section 2.3 to diversify an existing training population Π^{train} such that the best-response is robust to a larger variety of conventions.

The intuitive approach is to apply a different ϕ to each teammate π^i_j to maximise the variety of teams encountered. However, in a fully-collaborative Dec-POMDP, using different ϕ's will break the coordination between each $n-1$ components, π^i_j, so the same ϕ needs to be applied instead. Moreover, if π^i_j is a physical agent acting in the real world, then ϕ can't easily be applied as it requires us to modify its actions and observations.

Lemma 1. $J(\pi) = J(\phi(\pi)) \forall \phi \in \Phi, \pi \in \Pi$

This Lemma shows that the expected return of a joint-policy π is equal to the expected return when ϕ is applied π.

Proof in Appendix A.2.

Proposition 1. The expected AHT return when ϕ is applied to π is equal to the expected AHT return when the inverse ϕ^{-1} is applied to each of the π^i_j teammate policies.

Proof in Appendix A.2.

By applying ϕ to π_A, we're guaranteed to always be able to alter π_j's perceived conventions, and its application is of order $O(1)$. With this, our AHT agent π_A learns the optimal best-response

(SBA^*) to Π^{train} that has been augmented with Φ, by interacting with each policy $\pi_j \in \Pi^{train}$, applying $\phi \in \Phi$ to π_A, and maximising the expected AHT return (Equation 2). We formally define this objective as:

$$\pi_A^*(\Pi^{train}) = \underset{\pi}{\text{argmax}}\, \mathbb{E}_{\pi_j \sim \Pi^{train}, \phi \sim \Phi}[J_{AHT}(\phi(\pi_A), \pi_j)], \tag{9}$$

where π_j and ϕ are uniformly sampled from Π^{train} and Φ respectively. To evaluate the robustness when interacting with an unseen evaluation set Π^{eval}, we use the same robustness measure from Equation 10, $M_{\Pi^{eval}}(\pi_A^*(\Pi^{train}))$, and also apply equivalence mappings to the evaluation policies to reduce the evaluation variance (effectively generating a larger test-population). This measure is formally given by:

$$M_{\Pi^{eval}}(\pi_A^*(\Pi^{train})) = \mathbb{E}_{\pi_j \sim \Pi^{eval}, \phi \sim \Phi} J_{AHT}(\phi(\pi_A^*(\Pi^{train})), \pi_j), \tag{10}$$

where π_j and ϕ are sampled uniformly from Π^{eval} and Φ.

3.2 Algorithm

The idea behind SBA is simple: As shown in Figure 3, each of the AHT agents observations o_t^i are mapped with ϕ to an equivalent state with relabelled symmetries, π_A^i chooses an action a_t^i, and the action is inversely relabelled with ϕ^{-1} before being applied to the environment. The permuted observations and actions are appended to π_A^i's AOH τ^i, which is used to update the model. Notice that ϕ influences the agent's actions and observations, not the environment dynamics, and therefore any standard RL learning algorithm can be used to update the model, like DQN (Mnih et al., 2015), DDPG (Lillicrap et al., 2015), A3C (Mnih et al., 2016), or PPO (Schulman et al., 2017).

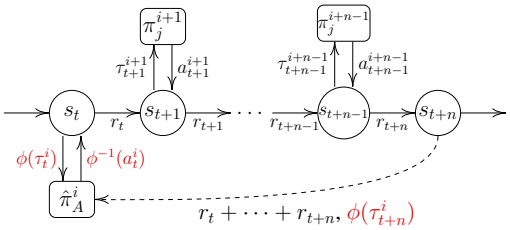

Figure 3: *Symmetry-breaking augmentations* for an n-player Dec-POMDP. The equivalence map ϕ is only applied to the observations and actions of our AHT agent π_A, not the teammate policy π_j.

In the simplest version of our algorithm, all of the AHT agents' transitions, $\mathbb{T} = (\tau_t^i, a_t^i, r_{t:t+n}, \tau_{t+n}^i)$ are stored, where $r_{t:t+n} = \sum_{t'=t}^{t+n} r_{t'}$ is the sum of all rewards received between time step t and $t+n$, and $\mathbb{T}$ is used to update the model. See Algorithm 1 for a more formal description.

Algorithm 1: Symmetry-Breaking Augmentations

Input: algorithm $\mathbb{A}$, Dec-POMDP G, population Π^{train}
Initialise: $\mathbb{A}$, equivalence mappings Φ from G
for *each episode* **do**
  $\pi_{-j} \leftarrow$ teammate policy sampled from Π^{train}
  $\phi \leftarrow$ equivalence mapping sampled from Φ
  $s_0, \tau_0 \leftarrow$ initial state and history
  for *each step* t **do**
    append observation o_t^i from s_t to AOH τ_t^i
    $a_t^i \leftarrow$ sample action using $\mathbb{A}$: $\phi^{-1}(\pi_j^i(\cdot|\phi(\tau_t^i)))$
    append action a_t^i to AOH τ_t^i
    for *each teammate component* π_{-j}^{-i} **do**
      append observation o_t^{-i} from s_t to AOH τ_t^{-i}
      $a_t^{-i} \leftarrow$ sample action from $\pi_{-j}^{-i}(\cdot|\tau_t^{-i})$
      append action a_t^{-i} to AOH τ_t^{-i}
    end
    take joint-action a_t, observe r_t, and s_{t+1}
  end
  for *each AHT agent turn* t **do**
    $r_{t:t+n} \leftarrow$ sum rewards from r_t to r_{t+n}
    $\mathbb{T} \leftarrow$ transition $(o_t^i, a_t^i, r_{t:t+n}, o_{t+n}^i)$
    perform one step of optimisation using $\mathbb{A}$ and $\mathbb{T}$
  end
end

3.3 Augmentation Impact

We aim to increase the diversity of training partners by applying SBA to each member of the training population, and hypothesize that this will lead to better generalisation in the AHT setting. However, if the training agents barely rely on symmetry-based conventions, SBA will have little effect, i.e. $\phi(\pi_j) \approx \pi_j, \forall \pi_j \in \Pi^{train}, \forall \phi \in \Phi$.

Since training an AHT agent can be expensive, it is useful to quantify how much SBA will diversify a population prior to training. For this, we introduce *Augmentation Impact* (AugImp), a metric that takes

a population Π, a set of equivalence mappings Φ, and for each pair of policies $\pi_1, \pi_2 \in \Pi$, measures the expected absolute difference of the crossplay scores (Lupu et al., 2021) with and without each augmentation $\phi \in \Phi$ being applied to one of the policies, π_1. AugImp is formalised by:

$$\text{AugImp}(\Pi, \Phi) = \mathbb{E}_{\pi_1 \sim \Pi, \pi_2 \sim \Pi, \phi \sim \Phi}[|J_{\text{XP}}(\phi(\pi_1), \pi_2) - J_{\text{XP}}(\pi_1, \pi_2)|]$$

where π_1, π_2, and ϕ are uniformly sampled from Π and ϕ respectively. The bigger the AugImp score, the more Π is diversified by Φ, and the better effect that SBA will have when training an AHT agent with that population.

4 Iterated Lever Game Experiments

We first test SBA in a simple fully cooperative environment where agents are tasked to coordinate by pulling one of ten possible levers. As shown in Figure 4, a reward of 1 is paid out if both players pick the same lever, otherwise they get nothing. The game is played twice, but in the second round the players are able to see what lever their partner previously pulled. If agents could coordinate beforehand they would always pull the same lever, but when playing with an unknown teammate there is no way to coordinate on the first round. To apply SBA to the lever game, since a permutation of the levers leaves the game unchanged, we use this as our class of symmetries.

We train our AHT agent with a population of five different teammates that each deterministically pull one of the levers, and evaluate with ten policies that pull all ten levers. We refer to Appendix B for more details on the implementation. The code is available online without downloading: `https://bit.ly/lever-game-sba`.

The results are shown in Figure 5, where, as expected, all agents randomly choose a lever in the first round. In the second round, during training the baseline (BR) always successfully switches to the correct lever for a return of 1.2, but only scores 0.6 at test time because it doesn't expect the other five levers to be pulled. Our SBA agent, however, experiences all levers during training, so is able to adapt to all teammates for a total score of 1.1 in both training and testing.

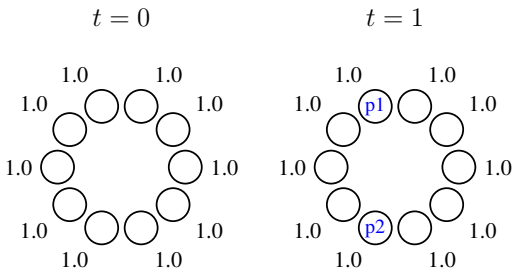

Figure 4: In the *iterated lever coordination game* agents can see what actions were previously taken. The game highlights the difficulty of adapting to conventions not seen during training.

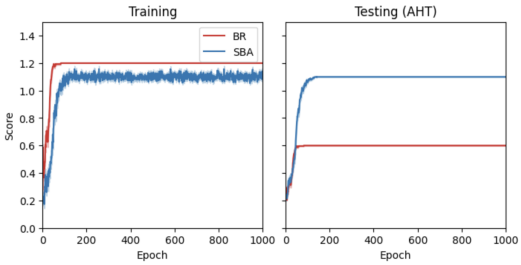

Figure 5: Training curves for the *iterated lever coordination game*. Shown is the mean, shading is the standard error of the mean, across 30 different seeds. SBA improves test performance because it exposes the agent to more conventions during training.

5 Hanabi Experiments

We now test SBA in Hanabi. Hanabi is a fully-cooperative, partially-observable card game (Bard et al., 2020) for MARL, theory of mind, and AHT research. In Hanabi, players cannot see their own cards and must rely on limited clues from teammates to cooperatively build five coloured decks in ascending order without triggering bombs. For the full rules of the game, please see Appendix C. For a full description of our experiment setup, please see Appendix D.

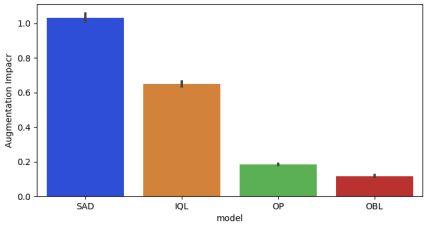

Figure 6: *Augmentation Impact* (AugImp) for Hanabi populations. SAD and IQL populations have a larger AugImp than OP and OBL, because they contain policies with more symmetry-breaking conventions.

5.1 TEAMMATE SELECTION

Before training our agent, a teammate policy population needs to be selected. Since there exists a range of pre-trained Hanabi agent populations online, we use AugImp (Equation 11) to guide the selection. We analyse the AugImp scores for four populations: 13 simplified action decoder (SAD), 12 independent Q-learning (IQL), 12 other-play (OP) models, and 5 off-belief learning (OBL) models (pre-trained weights for these models are available on GitHub[1][2]).

To calculate the crossplay scores for each pair of policies π_1 and π_2 and each augmentation ϕ, we take the mean score over 1000 games. Figure 6 shows the AugImp scores for each population. The scores for SAD and IQL are much larger than OP and OBL, which is expected because OP and OBL are designed to use conventions that don't break symmetries. For a complete breakdown of the AugImp score distributions for each policy pairing, see Appendix E. Since SBA increases the diversity of SAD and IQL, we use these populations for training because we expect the largest AHT performance improvement.

5.2 BEST RESPONSE AGENTS

We train AHT agents using the pre-trained SAD and IQL populations, and evaluate their ad hoc generalisation to held-out partners. We create a number of testing and training splits for each population. Each population is randomly divided into *small*, *medium*, or *large* training/test splits: *small* splits have 1 training policy, *medium* splits have 6, and *large* training sets contain all but 2 policies from the population. The remaining policies form the test set for that split. Smaller split sizes are more challenging, as our AHT agents are exposed to a more limited set of partners during training. We randomly sample 10 different partitions for *medium* and *large* training sets, and run all possible partitions for *small* (13 for SAD and 12 for IQL). See Appendix D.2 for details on train-test splits.

We train AHT agents using SBA on pre-existing populations rather than generating our population from scratch as in Lupu et al. (2021); Rahman et al. (2023a). While this approach is limited in that it requires pre-existing training policies to be available, it has the advantage of being a natural way for us to specify a prior over strategies that we want our AHT agent to specialise in. This is important in the context of Hanabi, where the space of possible strategies is large. We also believe that this approach is sufficient to demonstrate the effectiveness of SBA at allowing AHT agents to generalise to symmetry-equivalent held-out partners.

5.3 AD HOC TEAMWORK RESULTS

Here we examine the impact of SBA on AHT performance to the held-out test agents for SAD and IQL. For each of the train/test splits outlined above, we train a standard best response with and without SBA. Each agent is evaluated on the held out test agents from its training split.

Table 7 outlines the mean performance for each of the populations and split sizes. In all cases, SBA improves performance over the baseline. For the *medium* SAD splits we also compare to the best result for this population from Generalized Beliefs (see Section 6.1) (Muglich et al., 2022b). Our method improves upon this previous Hanabi AHT state-of-the-art by an average of 2.93 points[3].

Train Size	Agent	SAD ↑	IQL ↑
small	BR	8.15 ± 1.28	11.52 ± 1.08
	SBA (ours)	$\mathbf{9.12 \pm 1.42}$	11.84 ± 0.99
medium	Gen. Belief	12.47 ± 1.02	-
	BR	13.09 ± 0.49	15.04 ± 0.37
	SBA (ours)	$\mathbf{15.40 \pm 0.49}$	$\mathbf{16.08 \pm 0.42}$
large	BR	14.69 ± 1.05	15.34 ± 0.80
	SBA (ours)	$\mathbf{16.34 \pm 1.29}$	15.95 ± 0.71
	OP	3.26 ± 1.20	12.00 ± 0.51

Figure 7: SBA performance in Hanabi. The reported score for generalized beliefs (Gen. Belief) on the SAD *medium* split size. Shown is the standard error of the mean (s.e.m) across the *small*, *medium*, and *large* training train-test splits.

We perform two-tailed Monte Carlo permutation tests (Dwass, 1957) to estimate whether there is a statistically significant difference between the baseline and SBA. For medium and large split sizes,

[1] https://github.com/facebookresearch/hanabi_SAD

[2] https://github.com/facebookresearch/off-belief-learning

[3] Our baseline (BR) also outperforms Generalised Beliefs because we base our work on the newer and more fine-tuned OBL implementation.

we find that SBA confers a statistically significant advantage over the baseline to the level $\alpha = 0.01$. For small split sizes, we find that the improvement is statistically significant to the level $\alpha = 0.05$. This shows that applying SBA while training an AHT agent in this context consistently improves performance, making it essential for AHT when symmetry-based conventions exist within the training population.

We also examine the effect of SBA when applied to OP agents. We train baseline and SBA agents on *medium* OP train/test splits. Intuitively, OP trains agents to be equivariant under symmetries which should render SBA ineffective. Indeed, the baseline agents achieve an average score of 19.27 ± 0.42 with the held-out partners, while our SBA agents score 19.39 ± 0.42. We find that this difference is *not statistically significant,* indicating that SBA does not degrade performance even when applied to a population with lower AugImp variance under colour permutation.

To gain insight into why SBA improves performance (whenever it does work!), we analyse how often each agent gives *colour hints* (Figure 8). When an SBA agent is trained with SAD and IQL splits, it hints colours significantly less often compared to a baseline agent. Due to random symmetry breaking, hinting colours risks eliciting an unexpected reaction from the teammate, and SBA learns to avoid this. Instead it learns to use other means of hinting, such as using rank. When trained with OP, however, given OP's low AugImp variance (Figure 6), it hints colours roughly as often as the baseline.

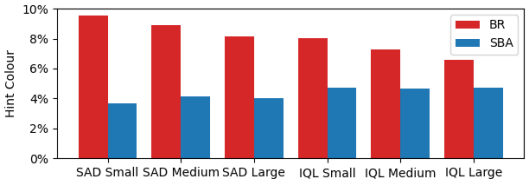

Figure 8: Frequency of the *hint colour* action played. SBA hints colours less often when the training set uses symmetry breaking conventions.

5.4 Generalisation to Other Populations

The previous experiment examines how SBA affects generalisation to held-out teammates from the same population as those used for training (either SAD or IQL). Here, we investigate whether SBA is effective at creating policies that can transfer to entirely different populations. We take the *medium* split size agents from the previous experiment, and evaluate them when paired with teammates from these different populations.

Train Set	Agent	OP ↑	OBL ↑
SAD	BR	15.69 ± 0.26	$\mathbf{4.51 \pm 0.21}$
	SBA (ours)	$\mathbf{17.72 \pm 0.26}$	3.85 ± 0.16
IQL	BR	15.71 ± 0.24	$\mathbf{5.73 \pm 0.24}$
	SBA (ours)	$\mathbf{16.42 \pm 0.16}$	5.50 ± 0.20

Figure 9: *Symmetry-Breaking Conventions* performance in Hanabi for *medium* training set sizes, cooperating with out-of-distribution populations. Shown is the standard error of the mean (s.e.m) across 13, 10, and 10 training splits respectively.

Table 9 outlines our results. SBA agents trained on SAD exhibit improved transfer performance to the IQL and OP populations. Applying the same significance testing as in the previous experiment, we find that these results are significant to the level $\alpha = 0.01$. Similarly for agents trained with IQL partners, we find that SBA confers a statistically significant advantage when playing with OP agents (to the level $\alpha = 0.05$). Additional results with all split sizes and OP-trained SBA and baseline agents are available in Appendix F.

We also see that SBA can *harm* performance when transferring to the OBL population. For SBA agents trained with SAD, we find this detrimental effect to be statistically significant to the level $\alpha = 0.01$. OBL policies require explicit colour information approximately 65% of the time for cards that they play (Hu et al., 2021), and thus rely on colour hints. SBA agents exhibit much lower frequency of providing these colour hints to the partner (Figure 8). We hypothesize that this is one of the main reasons why SBA harms performance in this case.

6 Related Work

6.1 Ad hoc Teamwork

There have been a number of works that address the Ad Hoc Teamwork (AHT) problem. One such technique is Generalised Beliefs that uses belief models to assist with generalisation (Muglich et al., 2022b). Belief models are used to provide latent representations of trajectories to a policy model and assist with search rollouts for action selection. They show that this leads to improvements in AHT scores in Hanabi. This technique, however, requires an additional step to train the belief model prior to AHT agent training.

Several approaches achieve generalisation to held-out policies by training a best response to a diverse population (Lowe et al., 2017; Charakorn et al., 2020; McKee et al., 2022), often by training this population using an approach that encourages diversity. Approaches include minimising performance between policies while maximising individual self-play scores (Charakorn et al., 2022; Cui et al., 2022; Rahman et al., 2023b) and finding minimum coverage sets that span the policy space (Rahman et al., 2023a; Lauffer et al., 2023). Canaan et al. (2022) use hand-crafted rules to find diverse agents that have been trained using genetic algorithms and Yu et al. (2023) introduce sub-optimal biases into the reward function. While this ensures diversity in the training teammates, training large enough populations can be prohibitively expensive given the range of possible conventions.

Some previous works train an AHT agent using a population that is generated from scratch to be maximally diverse (Rahman et al., 2023b; Charakorn et al., 2022) or to approximate the minimum coverage set of possible best-response policies (Rahman et al., 2023a). Our approach differs from these in that it can be applied to any population, whether pre-existing or generated from scratch. SBA could be used in combination with a population generation-based approach to further increase diversity with a smaller training population size. In the presence of environmental symmetries, this could reduce the time required to train (and load) the AHT training population.

For zero-shot coordination relate work, see Appendix G.1, and for social convention related work, see Appendix G.2.

7 Conclusion

In this work we have shown that by applying a simple augmentation to the basic AHT learning framework, which we call *symmetry-breaking augmentations*, we can construct agents that are better able to coordinate in the AHT setting with partners they have not seen before. Our method achieves state-of-the-art performance when evaluated with a diverse collection of policies, including SAD policies that were previously unable to collaborate well in cross-play due to their high degree of symmetrical conventional specialisation. We have demonstrated that SBA always improves performance, regardless of training set size, and have defined SBA generally, shown its implementation with deep RL, and provided evidence from experiments in Hanabi that SBA yields robust agents capable of playing well with unfamiliar artificial partners.

One limitation of our approach is that symmetries must exist in both the environment and in teammate strategies, and may require expert knowledge to define. However, we expect that SBA could be applied to partial or imperfect symmetries, or extended to more general augmentations that are not strictly symmetry-based. Methods to automatically detect these symmetries could also be developed, and we leave this for future work.

In our experiments we assumed that a population of training agents is available whose strategies serve as a reasonable prior for the teammates our AHT agent will encounter; nevertheless, SBA could also be applied in settings where this training population is generated from scratch (Lupu et al., 2021; Rahman et al., 2023b).

In future work we will investigate how SBA can be utilised in combination with other AHT improvements, such as search, for instance by shuffling symmetries in search rollouts (Sutton & Barto, 2018) to better predict teammate actions. We will also apply SBA to a wider range of Dec-POMDPs, including those with disjoint sets of equivalent states and agents that observe and act in the real world. Given the prevalence of (potentially imperfect) symmetries in the real world, we believe that the key SBA ideas can be used to augment the experiences of real-world agents.

ETHICS STATEMENT

We have carefully considered the ethical implications of our work and have adhered to all relevant institutional, national, and international guidelines. No experiments involving human or animal subjects were conducted, and all data used were either publicly available or obtained with the necessary permissions and anonymised. We welcome further discussion on the ethical aspects of our research.

ACKNOWLEDGMENTS

Ravi gratefully acknowledges the funding provided by the Australian Government Research Program (RTP) Scholarship during his time at Adelaide University whilst working on this project. He also acknowledges funding from Autonomous Intelligent Machines and Systems, as well as from Rosebud, during his time at Oxford University.

REFERENCES

Stéphane Airiau, Sandip Sen, and Daniel Villatoro. Emergence of conventions through social learning. *Autonomous Agents and Multi-Agent Systems*, 28(5):779–804, 2014.

Nolan Bard, Jakob N Foerster, Sarath Chandar, Neil Burch, Marc Lanctot, H Francis Song, Emilio Parisotto, Vincent Dumoulin, Subhodeep Moitra, Edward Hughes, et al. The hanabi challenge: A new frontier for ai research. *Artificial Intelligence*, 280:103216, 2020.

Allan Beasty. *The role of language in the emergence of equivalence relations: A developmental study.* PhD thesis, University College of North Wales, 1987.

Rodrigo Canaan, Xianbo Gao, Julian Togelius, Andy Nealen, and Stefan Menzel. Generating and adapting to diverse ad-hoc partners in hanabi. *IEEE Transactions on Games*, 2022.

Micah Carroll, Rohin Shah, Mark K Ho, Tom Griffiths, Sanjit Seshia, Pieter Abbeel, and Anca Dragan. On the utility of learning about humans for human-ai coordination. *Advances in neural information processing systems*, 32, 2019.

Rujikorn Charakorn, Poramate Manoonpong, and Nat Dilokthanakul. Investigating partner diversification methods in cooperative multi-agent deep reinforcement learning. In *Neural Information Processing: 27th International Conference, ICONIP 2020, Bangkok, Thailand, November 18–22, 2020, Proceedings, Part V 27*, pp. 395–402. Springer, 2020.

Rujikorn Charakorn, Poramate Manoonpong, and Nat Dilokthanakul. Generating diverse cooperative agents by learning incompatible policies. In *The Eleventh International Conference on Learning Representations*, 2022.

Brandon Cui, Andrei Lupu, Samuel Sokota, Hengyuan Hu, David J Wu, and Jakob Nicolaus Foerster. Adversarial diversity in hanabi. In *The Eleventh International Conference on Learning Representations*, 2022.

Neil Dugdale and C Fergus Lowe. Testing for symmetry in the conditional discriminations of language-trained chimpanzees. *Journal of the experimental analysis of behavior*, 73(1):5–22, 2000.

Meyer Dwass. Modified randomization tests for nonparametric hypotheses. *The Annals of Mathematical Statistics*, pp. 181–187, 1957.

Jakob Foerster, Francis Song, Edward Hughes, Neil Burch, Iain Dunning, Shimon Whiteson, Matthew Botvinick, and Michael Bowling. Bayesian action decoder for deep multi-agent reinforcement learning. In *International Conference on Machine Learning*, pp. 1942–1951. PMLR, 2019.

Drew Fudenberg and Jean Tirole. *Game theory*. MIT press, 1991.

Tigran Galstyan, Hrayr Harutyunyan, Hrant Khachatrian, Greg Ver Steeg, and Aram Galstyan. Failure modes of domain generalization algorithms. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19077–19086, 2022.

Alex Graves. Long short-term memory. *Supervised sequence labelling with recurrent neural networks*, pp. 37–45, 2012.

Michael Hechter and Karl-Dieter Opp. Social norms. 2001.

Dan Horgan, John Quan, David Budden, Gabriel Barth-Maron, Matteo Hessel, Hado Van Hasselt, and David Silver. Distributed prioritized experience replay. *arXiv preprint arXiv:1803.00933*, 2018.

Hengyuan Hu and Jakob N Foerster. Simplified action decoder for deep multi-agent reinforcement learning. *arXiv preprint arXiv:1912.02288*, 2019.

Hengyuan Hu, Adam Lerer, Alex Peysakhovich, and Jakob Foerster. "other-play" for zero-shot coordination. In *International Conference on Machine Learning*, pp. 4399–4410. PMLR, 2020.

Hengyuan Hu, Adam Lerer, Brandon Cui, Luis Pineda, Noam Brown, and Jakob Foerster. Off-belief learning. In *International Conference on Machine Learning*, pp. 4369–4379. PMLR, 2021.

Steven Kapturowski, Georg Ostrovski, John Quan, Remi Munos, and Will Dabney. Recurrent experience replay in distributed reinforcement learning. In *International conference on learning representations*, 2018.

Niklas Lauffer, Ameesh Shah, Micah Carroll, Michael D Dennis, and Stuart Russell. Who needs to know? minimal knowledge for optimal coordination. In *International Conference on Machine Learning*, pp. 18599–18613. PMLR, 2023.

Adam Lerer and Alexander Peysakhovich. Learning existing social conventions via observationally augmented self-play. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, pp. 107–114, 2019.

David Lewis. *Convention: A philosophical study*. John Wiley & Sons, 2008.

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Ryan Lowe, Yi I Wu, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in neural information processing systems*, 30, 2017.

Andrei Lupu, Brandon Cui, Hengyuan Hu, and Jakob Foerster. Trajectory diversity for zero-shot coordination. In *International Conference on Machine Learning*, pp. 7204–7213. PMLR, 2021.

Mingwei Ma, Jizhou Liu, Samuel Sokota, Max Kleiman-Weiner, and Jakob Nicolaus Foerster. Learning intuitive policies using action features. In *International Conference on Machine Learning*, pp. 23358–23372. PMLR, 2023.

Kevin R McKee, Joel Z Leibo, Charlie Beattie, and Richard Everett. Quantifying the effects of environment and population diversity in multi-agent reinforcement learning. *Autonomous Agents and Multi-Agent Systems*, 36(1):21, 2022.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pp. 1928–1937. PMLR, 2016.

Darius Muglich, Christian Schroeder de Witt, Elise van der Pol, Shimon Whiteson, and Jakob Nicolaus Foerster . Equivariant networks for zero-shot coordination. In *NeurIPS 2022*, November 2022a.

Darius Muglich, Luisa M Zintgraf, Christian A Schroeder De Witt, Shimon Whiteson, and Jakob Foerster. Generalized beliefs for cooperative ai. In *International Conference on Machine Learning*, pp. 16062–16082. PMLR, 2022b.

Ranjit Nair, Milind Tambe, Makoto Yokoo, David Pynadath, and Stacy Marsella. Taming decentralized pomdps: Towards efficient policy computation for multiagent settings. In *IJCAI*, volume 3, pp. 705–711, 2003.

Arrasy Rahman, Jiaxun Cui, and Peter Stone. Minimum coverage sets for training robust ad hoc teamwork agents. *arXiv preprint arXiv:2308.09595*, 2023a.

Arrasy Rahman, Elliot Fosong, Ignacio Carlucho, and Stefano V. Albrecht. Generating teammates for training robust ad hoc teamwork agents via best-response diversity, 2023b.

Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. *arXiv preprint arXiv:1511.05952*, 2015.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Peter Stone, Gal A Kaminka, Sarit Kraus, and Jeffrey S Rosenschein. Ad hoc autonomous agent teams: Collaboration without pre-coordination. In *Twenty-Fourth AAAI Conference on Artificial Intelligence*, 2010.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

Ming Tan. Multi-agent reinforcement learning: Independent vs. cooperative agents. In *Proceedings of the tenth international conference on machine learning*, pp. 330–337, 1993.

Gerald Tesauro. Td-gammon, a self-teaching backgammon program, achieves master-level play. *Neural computation*, 6(2):215–219, 1994.

Johannes Treutlein, Michael Dennis, Caspar Oesterheld, and Jakob Foerster. A new formalism, method and open issues for zero-shot coordination. In *International Conference on Machine Learning*, pp. 10413–10423. PMLR, 2021.

Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016.

Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Hasselt, Marc Lanctot, and Nando Freitas. Dueling network architectures for deep reinforcement learning. In *International conference on machine learning*, pp. 1995–2003. PMLR, 2016.

Chao Yu, Jiaxuan Gao, Weilin Liu, Botian Xu, Hao Tang, Jiaqi Yang, Yu Wang, and Yi Wu. Learning zero-shot cooperation with humans, assuming humans are biased. *arXiv preprint arXiv:2302.01605*, 2023.

A PROOFS

A.1 LEMMA 1

$J(\pi) = J(\phi(\pi)) \forall \phi \in \Phi, \pi \in \Pi$

This Lemma shows that the expected return of a joint-policy π is equal to the expected return when ϕ is applied π.

Proof.

$$J(\pi) = \mathbb{E}_{\tau_t \sim \pi} V^{\pi}(\tau_t) \tag{11}$$

$$= \sum_{\tau_t} P(\tau_t|\pi) \sum_{a_t} \pi(a_t|\tau_t) \sum_{r_t} \mathcal{R}(r_t|s_t, a_t) \left(r_t + \gamma \sum_{s_{t+1}} \mathcal{T}(s_{t+1}|s_t, a_t) \sum_{o_{t+1}} \mathcal{U}(o_{t+1}|s_{t+1}, a_t) V^{\pi}(\tau_t \oplus (a_t, o_{t+1})) \right) \tag{12}$$

$$= \sum_{\tau_t} P(\tau_t|\phi(\phi^{-1}(\pi))) \sum_{a_t} \pi(\phi(\phi^{-1}(a_t))|\phi(\phi^{-1}(\tau_t))) \sum_{r_t} \mathcal{R}(r_t|s_t, a_t) \left(r_t + \gamma \sum_{s_{t+1}} \mathcal{T}(s_{t+1}|s_t, a_t) \sum_{o_{t+1}} \mathcal{U}(o_{t+1}|s_{t+1}, a_t) V^{\phi(\phi^{-1}(\pi))}(\tau_t \oplus (a_t, o_{t+1})) \right) \tag{13}$$

Since ϕ is an automorphism.

$$= \sum_{\phi(\tau_t)} P(\phi(\tau_t)|\phi(\phi^{-1}(\phi(\pi)))) \sum_{\phi(a_t)} \pi(\phi(\phi^{-1}(\phi(a_t)))|\phi(\phi^{-1}(\phi(\tau_t)))) \sum_{\phi(r_t)} \mathcal{R}(\phi(r_t)|\phi(s_t), \phi(a_t)) \left(\phi(r_t) + \gamma \sum_{\phi(s_{t+1})} \mathcal{T}(\phi(s_{t+1})|\phi(s_t), \phi(a_t)) \sum_{\phi(o_{t+1})} \mathcal{U}(\phi(o_{t+1})|\phi(s_{t+1}), \phi(a_t)) \right.$$

$$\left. V^{\phi(\phi^{-1}(\phi(\pi)))}(\phi(\tau_t) \oplus (\phi(a_t), \phi(o_{t+1}))) \right) \tag{14}$$

$$= \sum_{\tau_t} P(\tau_t|\phi(\pi)) \sum_{a_t} \pi(\phi(a_t)|\phi(\tau_t)) \sum_{r_t} \mathcal{R}(r_t|s_t, a_t) \left(r_t + \gamma \sum_{s_{t+1}} \mathcal{T}(s_{t+1}|s_t, a_t) \sum_{o_{t+1}} \mathcal{U}(o_{t+1}|s_{t+1}, a_t) V^{\phi(\pi)}(\tau_t \oplus (a_t, o_{t+1})) \right) \tag{15}$$

$$= \mathbb{E}_{\tau_t \sim \phi(\pi)} V^{\phi(\pi)}(\tau_t) \tag{16}$$

$$= J(\phi(\pi)) \tag{17}$$

□

A.2 PROPOSITION 1

The expected AHT return when ϕ is applied to π is equal to the expected AHT return when the inverse ϕ^{-1} is applied to each of the π_j^i teammate policies.

Proof.

$$
\begin{aligned}
&J_{AHT}(\pi_A, \phi^{-1}(\pi_j)) \\
&= \frac{1}{n}\big(J(\pi_A^1, \phi^{-1}(\pi_j^2), \dots, \phi^{-1}(\pi_j^n)) + \\
&\quad \cdots + J(\phi^{-1}(\pi_j^1), \dots, \phi^{-1}(\pi_j^{n-1}), \pi_A^n)\big) \\
&= \frac{1}{n}\big(J(\phi(\pi_A^1), \phi(\phi^{-1}(\pi_j^2)), \dots, \phi(\phi^{-1}(\pi_j^n))) + \\
&\quad \cdots + J(\phi(\phi^{-1}(\pi_j^1)), \dots, \phi(\phi^{-1}(\pi_j^{n-1})), \phi(\pi_A^n))\big) \\
&= \frac{1}{n}\big(J(\phi(\pi_A^1), \pi_j^2, \dots, \pi_j^n) + \\
&\quad \cdots + J(\pi_j^1, \dots, \pi_j^{n-1}, \phi(\pi_A^n))\big) \\
&= J_{AHT}(\phi(\pi_A), \pi_j) \qquad (18)
\end{aligned}
$$

□

B ITERATED LEVER COORDINATION GAME DETAILS

To emphasise the necessity of augmenting a training policy population with symmetry-breaking augmentation, we have created the iterated lever-coordination game. This game underscores the importance of exposing AHT agents to conventions not initially present in the training population to facilitate generalisation to a broader range of conventions at test-time. The neural network employed in this experiment is a basic 2-layer fully connected network with one hidden layer, utilising the sigmoid function as the activation function. The training process takes place on a CPU with a single thread. We present the crucial hyper-parameters in Table 1.

Hyper-parameters	Value
`# Network`	
`hidden size`	`20`
`activation`	`sigmoid`
`layers`	`2`
`# Optimisation`	
`optimiser`	`Adam`
`lr`	`0.05`
`eps`	`0.9`
`batchsize`	`10`
`# Training`	
`epochs`	`1000`
`num runs`	`30`

Table 1: Hyper-Paramaters for *iterated lever coordination game* Reinforcment Learning.

C HANABI RULES

Hanabi is a co-operative card game where players work together to create five colour-coded stacks of cards, each stack arranged in ascending rank from one to five. The deck contains exactly fifty cards: five colours (red, yellow, green, blue, and white), each composed of three copies of rank one, two copies of ranks two, three, and four, and a single copy of rank five. The game is played with two

to five players. If there are two or three players, each starts with five cards; if there are four or five players, each starts with four. Players hold their cards facing away from themselves so they can see everyone else's hand, but not their own. The group shares eight information tokens and three fuse tokens. If the group ever loses all three fuse tokens, the game ends immediately and the final score is zero.

Each turn, a player must choose one of three actions. The first action is to **give information** to a teammate by spending one information token. This clue must focus on a single rank or a single colour, and the clue-giver must indicate every card in the teammate's hand that matches that choice. The second action is to **discard** a card from hand, which returns one information token to the pool (unless the group already has the maximum of eight). A new card is drawn from the deck to replace any discarded card if the deck has not yet been exhausted. The third action is to **play** a card from hand, attempting to place it on the appropriate stack. Each colour stack must begin with a rank one, followed by two, three, four, and five in ascending order. A card that is played correctly is added to its colour stack, and if that card is a rank five, the team gains one information token (up to a maximum of eight). If a played card cannot legally be placed (for example, it is the wrong rank for its colour stack), a bomb is triggered and one fuse token is removed.

Once the deck is empty, each player takes one final turn. The score is the total number of successfully placed cards across all colours, with a maximum of twenty-five if every card is played in perfect sequence. Communication is strictly limited to the "give information" action, so effective co-operation relies on careful deduction and subtle signalling to avoid bombs and achieve the highest possible score.

D Experiment Details for Hanabi

D.1 Reinforcement Learning

We employ a highly scalable training architecture illustrated in Figure 10, built upon the framework implemented in the Off-Belief Learning Github Repository (Hu et al., 2021). This architecture features multiple parallel thread workers responsible for managing interactions across various Hanabi environments and the agents operating within each environment. Each actor initiates multiple inference calls to neural networks at every time step, with each inference call executed on GPUs. When a player initiates an inference call, the worker thread promptly proceeds to the next agent in the thread, facilitating the simultaneous execution of multiple games and agents on a single worker thread. Inferences invoked by different players are batched together and processed on the GPU in parallel. This approach enables the concurrent execution of a substantial number of games and environments, generating a significant volume of data for training purposes.

At each time step, players collect observations, actions, and rewards, and aggregate them into episodes. These trajectories are padded to 80 time steps and stored in a priority replay buffer. The training loop, operating independently of the aforementioned worker threads, continually samples transitions from the buffer, using them to update the model. After every 10 gradient update steps, the new model synchronizes with all the models conducting inference on GPUs.

We extended the training architecture to accommodate Ad Hoc Teamwork (AHT) agents. In each game within every worker thread, one agent acts as our AHT agent, learning from the experience, while the other agent sends inference requests to a frozen set of pre-trained neural network weights. Each AHT agent interacts with a distinct pre-trained model. When agents collect observations, actions, and rewards for storage in the replay buffer, only the AHT agent's experience is used for optimization; teammates' experiences are discarded. At the episode's start, a different policy augmentation permutation is chosen, maintaining an unchanged teammate in the environment. To ensure an even distribution of teammate policies, the number of games is selected to perfectly divide the number of training agents, ensuring experiences are evenly distributed across teammate policies.

The architecture adheres to what was, at one point in time, a state-of-the-art model—Recurrent Replay Distributed Deep Q-Networks (R2D2) (Kapturowski et al., 2018)—which incorporates best practices such as double-DQN (Van Hasselt et al., 2016), dueling network architecture (Wang et al., 2016), prioritized experience replay (Schaul et al., 2015), distributed training with parallel environments (Horgan et al., 2018), and a recurrent network to handle partial observability (Graves,

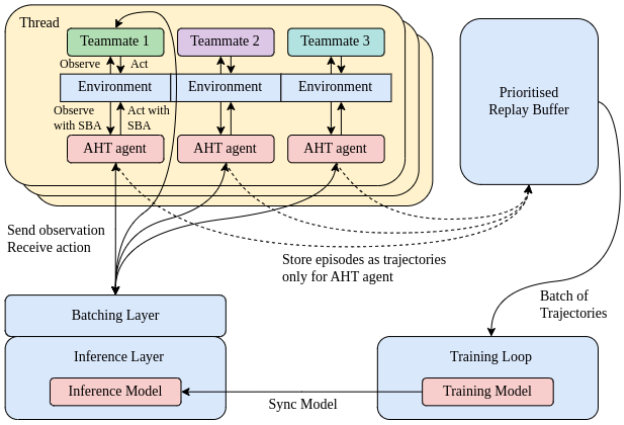

Figure 10: Illustration of RL training setup for AHT using SBA. Some arrows linking *player 1* and *batching layer* are omitted for legibility. Note that the experience of the teammates are not stored in the replay buffer, or used to update the model in the *training loop*.

2012). In all our experiments, we execute games and players on 23 worker threads, with 80 games per thread, and allocate 1 thread for training. For inferences, we employ three Nvidia GeForce RTX 3090 GPUs and one for training, and the worker threads are executed on an AMD Ryzen Threadripper 1920X 12-Core CPU. Our essential hyper-parameters are detailed in Table 2.

Hyper-parameters	Value
`# replay buffer related`	
`burn-in frames`	`10,000`
`replay buffer size`	`100,000`
`priority exponent`	`0.9`
`priority weight`	`0.6`
`max trajectory length`	`80`
`# Optimisation`	
`optimiser`	`Adam`
`lr`	`6.25e-05`
`eps`	`1.5e-05`
`grad clip`	`5`
`batchsize`	`128`
`# Q learning`	
`n step`	`3`
`discount factor`	`0.999`
`target network sync interval`	`2500`
`exploration` ϵ	$\epsilon_0 \dots \epsilon_n$, `where` $\epsilon_i = 0.1^{1+7i/(n-1)}, n = 80$

Table 2: Hyper-Paramaters for *Hanabi* Reinforcment Learning.

D.2 Hanabi Training Population Splits

To gain a more accurate understanding of SBA's performance when learning how to generalize a held-out sets of evaluation policies, it is useful to observe how much generalisation is affected by the size of the training set. Since SBA combinatorially increases the number of partners encountered during training, a performance improvement should still be observed when there is only one policy in the training set. As there is a varying number of pre-trained policies in each population, specifically 13 simplified action decoder (SAD), 12 independent Q-learning (IQL), and 12 other-play (OP), we have different split sizes for *small*, *medium*, and *large* splits. Table 3 provides the exact breakdown of these splits.

Population	*small* splits	*medium* splits	*large* splits
SAD	1 train/12 test	6 train/7 test	11 train/2 test
IQL	1 train/11 test	6 train/6 test	10 train/2 test
OP	1 train/11 test	6 train/6 test	10 train/2 test

Table 3: Breakdown of the train and test splits we use for Hanabi policy populations.

E AUGMENTATION IMPACT BREAKDOWN

Training an AHT agent in large Dec-POMDPs can be expensive, so it's important to determine whether an augmentation technique will meaningfully diversify a population before training commences. SBA is a technique that will only change the policies in the training population if they rely on symmetry-breaking conventions. Therefore, before training, we can use Augmentation Impact (AugImp) (Equation 19) to assess how much a given augmentation (in our case, SBA) will diversify the policies in a population.

The AugImp calculates the absolute difference across all pairs of agents and across all permutations. This metric combines all the information for a population into a single value, but the information about the max and min values is lost. To obtain a more thorough overview of the augmentations, we examine all Augmentation Differences (AD) individually for each policy pair. AD represents the Hanabi score difference between two agents from a population before and after the permutation has been applied to one agent, and it is defined as:

$$AD(\pi_1, \pi_2, \phi) = J_{\text{XP}}(\pi_1, \pi_2) - J_{\text{XP}}(\phi(\pi_1), \pi_2). \tag{19}$$

In Figure 11, we can observe the complete breakdown of all *Augmentation Differences* (AD) for all four Hanabi populations: SAD, IQL, OP, and OBL. The plot illustrates how different the augmentation scores are compared to no augmentation (depicted as a black 'x'). As expected, the plot demonstrates that SAD and IQL policy populations have a much larger spread than OP and OBL, with SAD frequently reaching an Augmentation Difference of $\pm$ 8 points. Interestingly, there exists one IQL pair that works together particularly well when a specific color permutation is applied.

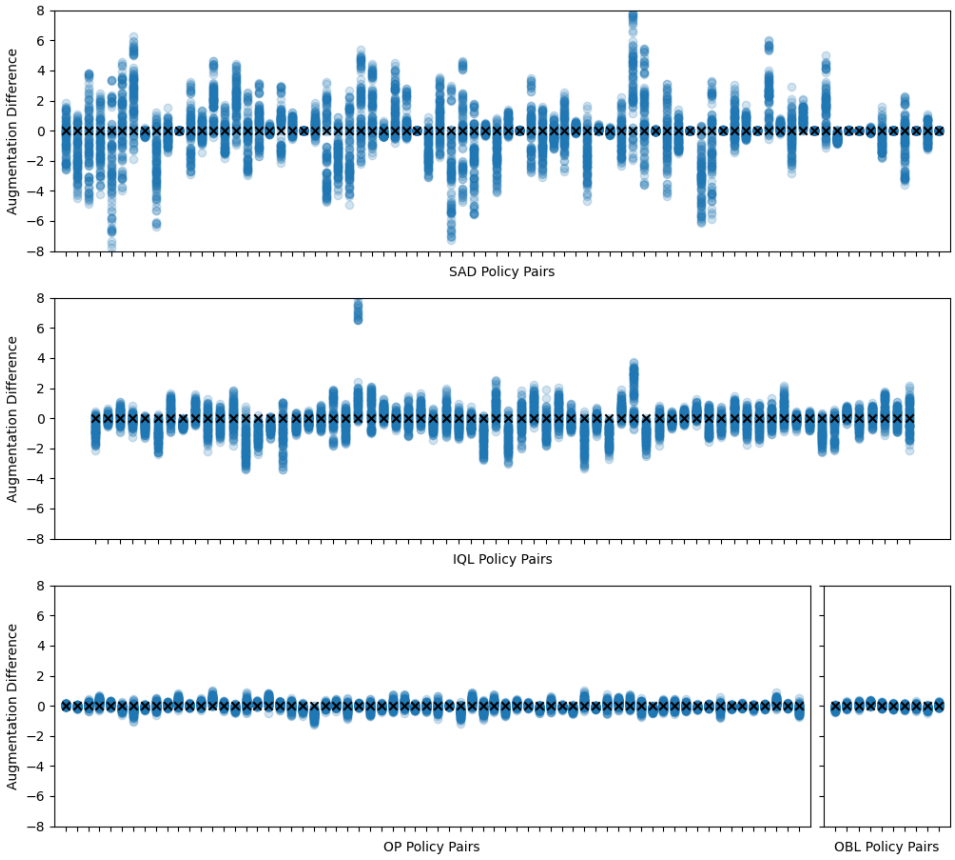

Figure 11: *Augmentation Difference* (Equation 19) scores for all pairs of SAD, IQL, OP, and OBL. Each column of blue contains the score differences between two policy pairs before and after applying an augmentation. The black x's represent the original non-augmented policies. It's clear that the SBA diversifies SAD and IQL populations much more than OP and OBL.

F AHT Results

F.1 SAD Ad Hoc Teamwork

		SAD (eval)		w/ IQL		w/ OP		w/ OBL	
Split	Agent	Score ↑	Bombout ↓	Score ↑	Bombout ↓	Score ↑	Bombout ↓	Score ↑	Bombout ↓
small	BR	8.15 ± 1.28	0.60 ± 0.03	10.78 ± 1.34	0.46 ± 0.06	10.85 ± 1.50	0.47 ± 0.07	**3.77 ± 0.39**	0.70 ± 0.01
	SBA (ours)	**9.12 ± 1.42**	**0.57 ± 0.03**	**11.53 ± 1.39**	**0.44 ± 0.06**	**12.19 ± 1.59**	**0.41 ± 0.07**	3.65 ± 0.34	0.71 ± 0.01
medium	Gen. Belief	12.36 ± 0.96	-	-	-	-	-	-	-
	BR	13.09 ± 0.49	0.39 ± 0.04	15.22 ± 0.25	0.27 ± 0.01	15.69 ± 0.26	0.26 ± 0.01	**4.51 ± 0.21**	**0.65 ± 0.01**
	SBA (ours)	**15.40 ± 0.49**	**0.28 ± 0.04**	**16.71 ± 0.22**	**0.20 ± 0.01**	**17.72 ± 0.26**	**0.17 ± 0.01**	3.85 ± 0.16	0.70 ± 0.01
large	BR	14.69 ± 1.05	0.31 ± 0.05	16.61 ± 0.12	0.21 ± 0.01	16.78 ± 0.22	0.21 ± 0.01	**4.56 ± 0.22**	0.64 ± 0.02
	SBA (ours)	**16.34 ± 1.29**	**0.24 ± 0.06**	**17.40 ± 0.07**	**0.17 ± 0.00**	**17.81 ± 0.21**	**0.16 ± 0.01**	4.36 ± 0.20	0.65 ± 0.02

Table 4: Mean scores and bombout rates for BR and SBA in Hanabi with a SAD population. Models are trained with 1 train and 12 test policies, 6-7, and 11-2. Compared are the reported 6-7 split scores for Generalized Beliefs (gen. belief) (Muglich et al., 2022b) (gen. belief). Shown is the standard error of the mean (s.e.m) across 13, 10, and 10 training splits respectively. An AHT agent trained on a SAD policies significantly improves performance over the baseline.

Table 4 showcases the performance results of SBA and baseline (BR) agents, both trained with SAD training populations of varying sizes: small, medium, and large. The robustness of each AHT agent is assessed by evaluating them with a held-out set of SAD agents, while their generalization is tested through cooperation with three distinct algorithms—namely, IQL, OP, and OBL—that were absent

Splits	SAD (eval)	IQL	OP	OBL
1-12	**0.021**	**0.019**	**0.002**	0.648
6-7	**0.002**	**0.002**	**0.002**	**0.006**
11-2	**0.002**	**0.002**	**0.002**	0.438

Table 5: Monte carlo paired permutation test comparing SBA to BR trained on SAD across different splits, and evaluating with different teammates. 100k samples are taken. Calculated is the two-sided p-value.

during training. In addition to the Hanabi scores, the bombout rate is presented, where lower rates indicate superior performance. The displayed standard error of the mean reflects the error across the training splits.

Notably, when the SBA agent is evaluated in coordination with other held-out SAD agents, as well as agents from IQL and OP populations, there is a significant improvement in scores across all scenarios. The most substantial improvement percentage is observed when SBA is trained with a medium-sized training set, resulting in up to a 17% score improvement. Remarkably, even when trained with just one agent in the training set, SBA still demonstrates performance enhancement, highlighting the efficacy of symmetry-breaking augmentations.

Interestingly, when SAD agents coordinate with OBL policies, the performance declines. This can be attributed to the fact that, in approximately 60% of instances, OBL plays a card only when it knows both the color and rank information. Since SBA tends to provide color hints less frequently, it fails to furnish OBL with sufficient information, resulting in a reduced frequency of card plays and, consequently, diminished performance.

In determining statistical significance, we employ a Paired Monte-Carlo Permutation test, and the corresponding results are presented in Table 5. Notably, SBA demonstrates a significant performance enhancement when assessed under SAD, IQL, and OP training policies. Conversely, its performance takes a noticeable dip when the AHT agent engages with OBL, particularly for medium-sized training sets. However, for both small and large training sets, the outcomes remain inconclusive.

F.2 IQL AD HOC TEAMWORK

		IQL (eval)		w/ SAD		w/ OP		w/ OBL	
Split	Agent	Score ↑	Bombout ↓	Score ↑	Bombout ↓	Score ↑	Bombout ↓	Score ↑	Bombout ↓
1-11	BR	11.52 ± 1.08	0.41 ± 0.05	8.49 ± 1.19	0.59 ± 0.05	11.52 ± 1.34	0.43 ± 0.06	5.19 ± 0.26	0.58 ± 0.02
	SBA (ours)	11.84 ± 0.99	0.40 ± 0.04	8.32 ± 0.96	0.59 ± 0.04	11.91 ± 1.08	0.40 ± 0.05	4.13 ± 0.33	0.66 ± 0.03
6-6	BR	15.04 ± 0.37	0.27 ± 0.02	13.23 ± 0.15	0.38 ± 0.01	15.71 ± 0.24	0.25 ± 0.01	5.73 ± 0.24	0.57 ± 0.02
	SBA (ours)	**16.08 ± 0.42**	**0.21 ± 0.02**	13.70 ± 0.19	0.34 ± 0.01	**16.42 ± 0.16**	**0.20 ± 0.01**	5.50 ± 0.20	0.56 ± 0.01
10-2	BR	15.34 ± 0.80	0.25 ± 0.03	13.81 ± 0.11	0.34 ± 0.00	16.06 ± 0.17	0.22 ± 0.01	5.97 ± 0.16	0.54 ± 0.01
	SBA (ours)	**15.95 ± 0.71**	**0.21 ± 0.03**	14.08 ± 0.16	0.33 ± 0.01	**16.72 ± 0.11**	**0.20 ± 0.01**	5.93 ± 0.16	0.55 ± 0.01

Table 6: Mean scores and bombout rates for BR and SBA in Hanabi with a IQL population. Models are trained with 1 train and 12 test policies, 6-7, and 11-2. Shown is the standard error of the mean across 13, 10, and 10 training splits respectively.

Splits	IQL (eval)	SAD	OP	OBL
1-11	0.187	0.713	0.156	0.117
6-6	**0.004**	0.156	**0.021**	0.330
10-2	**0.004**	0.235	**0.031**	0.284

Table 7: Monte carlo paired permutation test comparing SBA to BR trained on IQL across different splits, and evaluating with different teammates. 100k samples are taken. Calculated is the two-sided p-value.

Table 6 presents the performance results of SBA and baseline (BR) agents, both trained with IQL training populations of varying sizes: small, medium, and large. The robustness of each AHT agent is assessed by evaluating them with a held-out set of IQL agents, while their generalization is tested through cooperation with three distinct algorithms—specifically, SAD, OP, and OBL—that were absent during training. In addition to the Hanabi scores, the bombout rate is provided, with lower rates indicating superior performance. The displayed standard error of the mean reflects the error across the training splits.

In contrast to when SBA is trained with a SAD population, its performance does not exhibit the same level of strong improvement when trained with IQL. This is expected due to the lower Augmentation Impact score it receives (Section 5.1). Nevertheless, this AHT agent still demonstrates statistically significant performance on an IQL evaluation set for medium and large-sized training sets, as well as when evaluated with OP policies for medium and large training sets. However, when SBA is trained with a single IQL training partner, it never shows statistically significant performance over the baseline (BR). This is likely because the conventions present in a single IQL agent are not strong enough for SBA to meaningfully create a diverse training population.

Furthermore, as shown in Table 7, none of these SBA agents achieve statistically significant improvements over the baseline when playing with SAD agents. This is likely attributed to the fact that SAD conventions are much stronger than IQL, and learning to adapt to IQL conventions alone is not sufficient to adequately adjust to SAD.

F.3 OP AD HOC TEAMWORK

		w/ OP (eval)		w/ SAD		w/ IQL		w/ OBL	
Split	Agent	Score ↑	Bombout ↓	Score ↑	Bombout ↓	Score ↑	Bombout ↓	Score ↑	Bombout ↓
6-6	BR	19.27 ± 0.42	0.14 ± 0.02	12.58 ± 0.18	0.40 ± 0.01	15.44 ± 0.07	0.25 ± 0.00	7.27 ± 0.26	0.54 ± 0.01
	SBA (ours)	19.39 ± 0.42	0.14 ± 0.02	12.51 ± 0.24	0.40 ± 0.01	15.52 ± 0.13	0.24 ± 0.01	7.44 ± 0.39	0.53 ± 0.02

Table 8: Mean scores and bombout rates for BR and SBA in Hanabi with a OP population. Models are trained with 1 train and 12 test policies, 6-7, and 11-2. Shown is the standard error of the mean (s.e.m) across 13, 10, and 10 training splits respectively.

Splits	OP (eval)	SAD	IQL	OBL
6-6	0.164	0.641	0.408	0.629

Table 9: Monte carlo paired permutation test comparing SBA to BR trained on OP across different splits, and evaluating with different teammates. 100k samples are taken. Calculated is the two-sided p-value.

Table 8 presents the performance results of both SBA and baseline (BR) agents, trained with a medium-sized OP training population. The robustness of each AHT agent is evaluated by assessing them with a held-out set of OP agents. Generalization is tested by cooperation with three distinct algorithms—specifically, SAD, IQL, and OBL—that were absent during training. In addition to the Hanabi scores, the bombout rate is provided, where lower rates indicate superior performance. The displayed standard error of the mean (s.e.m) represents the error across the training splits.

Given that an agent trained with Other-Play aims to avoid symmetry-breaking conventions, as supported by its low Augmentation Impact score in Section 5.1, augmenting an OP policy with SBA is anticipated to have minimal impact on AHT performance. The results presented in this section substantiate this assertion, demonstrating that SBA does not exhibit any performance improvement over the baseline. In certain cases, due to variance, it even performs slightly worse than the baseline. This observation is further corroborated by the Monte-Carlo Permutation Tests in Table 9, where SBA does not show any significantly improved performance, with p-values ranging from 0.16 to 0.6.

G Related Work

G.1 Zero-shot Coordination

In zero-shot coordination (ZSC), agents must coordinate with new teammates that are also optimised for ZSC. Example solution approaches to this problem address symmetries in conventions by training agents in self-play with symmetry-equivalent versions of themselves (Hu et al., 2020; Treutlein et al., 2021), incorporating symmetrization into the network architecture (Muglich et al., 2022a), or using belief models to find optimal grounded policies that assume all previous actions were taken by the uniform random policy (Hu et al., 2021). While these techniques achieve high scores in ZSC, they are designed to avoid use of specialised conventions, and fail to coordinate with policies that do use these conventions.

G.2 Social Conventions

Like humans who use social conventions to facilitate coordination (Hechter & Opp, 2001; Lewis, 2008), artificial learning agents are also known to exploit conventions to cooperate (Airiau et al., 2014). The issue, however, is that in many collaborative settings there are multiple optimal strategies under self-play (Tesauro, 1994), but no guarantee that two independently trained agents will converge to policies with compatible conventions (Foerster et al., 2019; Hu & Foerster, 2019; Hu et al., 2020).

Solutions have been proposed to encourage learning agents to better converge to test-time conventions. Such as revealing test-time observations during training (Lerer & Peysakhovich, 2019), learning with human behavioral-cloned models to better coordinate with real humans (Carroll et al., 2019), exploiting the similarity between action and observation features to take human-like actions (Ma et al., 2023), and by training with teammates that have hidden biases to better coordinate with sub-optimal humans (Yu et al., 2023). While interesting directions, these methods make assumptions about what conventions the test-time policies will use, whereas our approach exploits and diversifies the conventions that already exist within a training population.