"Other-Play" for Zero-Shot Coordination

Hengyuan Hu [*1] **Adam Lerer** [1] **Alex Peysakhovich** [1] **Jakob Foerster** [*1]

Abstract

We consider the problem of zero-shot coordination - constructing AI agents that can coordinate with novel partners they have not seen before (e.g. humans). Standard Multi-Agent Reinforcement Learning (MARL) methods typically focus on the self-play (SP) setting where agents construct strategies by playing the game with themselves repeatedly. Unfortunately, applying SP naively to the zero-shot coordination problem can produce agents that establish highly specialized conventions that do not carry over to novel partners they have not been trained with. We introduce a novel learning algorithm called *other-play* (OP), that enhances self-play by looking for more robust strategies, exploiting the presence of known symmetries in the underlying problem. We characterize OP theoretically as well as experimentally. We study the cooperative card game Hanabi and show that OP agents achieve higher scores when paired with independently trained agents. In preliminary results we also show that our OP agents obtains higher average scores when paired with human players, compared to state-of-the-art SP agents.

1. Introduction

A central challenge for AI is constructing agents that can coordinate and cooperate with partners they have not seen before (Kleiman-Weiner et al., 2016; Lerer & Peysakhovich, 2017; Carroll et al., 2019; Shum et al., 2019). This is particularly important in applications such as cooperative game playing, communication, or autonomous driving (Foerster et al., 2016; Lazaridou et al., 2016; Sukhbaatar et al., 2016; Resnick et al., 2018). In this paper we consider the question of zero-shot coordination where agents are placed into a cooperative situation with a novel partner and must quickly coordinate if they wish to earn high payoffs.

Our setting is a partially observed cooperative Markov game (MG) which is commonly known among both agents. The agents are able to construct strategies separately in the training phase but cannot coordinate on the strategies that they construct. They must then play these strategies when paired together one time. We refer to this as zero-shot coordination.

A popular way of constructing strategies for MG with unknown opponents is self-play (or "self-training"), (Tesauro, 1994). Here the agent controls both players during training and iteratively improves both players' strategies. The agent then uses this strategy at test time. If it converges, self-play finds a Nash equilibrium of the game and yields superhuman AI agents in two-player zero-sum games such as Chess, Go and Poker (Campbell et al., 2002; Silver et al., 2017; Brown & Sandholm, 2018). However, in complex environments self-play agents typically construct 'inhuman' strategies (Carroll et al., 2019). This may be a benefit for zero-sum games, but is less useful when it is important to coordinate with, not trick, one's partner.

Our main contribution is "other-play" (OP), an algorithm for constructing good strategies for the zero-shot coordination setting. We assume that with every MG we are provided with a set of symmetries, i.e. arbitrary relabelings of the state/action space that leave trajectories unchanged up to relabeling. One source of miscoordination in zero-shot settings is that agents have no good way to break the symmetries (e.g. should we drive on the left or the right?). In most MDPs, there are classes of strategies that require more or less coordinated symmetry breaking. OP's goal is to find a strategy that is *maximally robust* to partners breaking symmetries in different ways while still playing in the same class. OP works as follows: it uses RL to maximize reward when matched with agents playing the same policy under a random relabeling of states and actions under the known symmetries.

To show the intuition behind OP consider the following game: you need to coordinate with an unknown stranger by independently choosing one from a set of 10 different levers (Figure 1a). If both of you pick the same lever a reward of 1 point is paid out, otherwise you leave the game empty-handed. Clearly, without any prior coordination the only option is to pick one of the levers at random, leading to an expected reward of $1/10 = 0.1$.

Next we consider a game that instead only pays 0.9 for one

[*]Equal contribution [1]Facebook AI Research, USA. Correspondence to: Hengyuan Hu <hengyuan@fb.com>, Jakob Foerster <jnf@fb.com>.

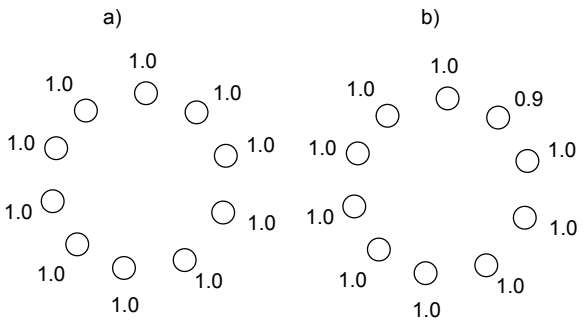

Figure 1. The *lever coordination game* illustrates the counter intuitive outcome of zero-shot coordination.

of the levers, keeping all other levers unchanged (Figure 1b). How does this change the coordination problem?

From the point of view of the MG the 1 payoff levers have no labels and so are symmetric. Since agents cannot coordinate on how to break symmetries, picking one of the 1.0 levers leads to 0.11 expected return. By contrast, OP suggests the choice of the 0.9 lever.

We note that this example illustrates another facet of OP: it is an equilibrium in meta-strategies. That is, neither agent wishes to deviate from OP as a reasoning strategy if the other agent is using it.

Note that OP does not use any action labels. Instead OP uses only features of the problem description to coordinate. Furthermore, note that the OP policy in this setting is the only policy that would *never* be chosen by the types of algorithms that try to use self-play to optimize team performance, e.g. VDN (Sunehag et al., 2018) or SAD (Hu & Foerster, 2019).

The main contributions of this work are: 1) we introduce OP as a way of solving the zero-shot coordination problem, 2) we show that OP is the highest payoff meta-equilibrium for the zero-shot coordination problem, 3) we show how to implement OP using deep reinforcement learning (deep RL) based methods, and 4) we evaluate OP in the cooperative card game Hanabi (Bard et al., 2020).

2. Related Work

2.1. Self-Play in Cooperative Settings

There is a large body of research on constructing agents to do well in positive-sum games. Self-play, if it converges, converges to an equilibrium of the game and so in purely cooperative games SP agents will be able to coordinate. Here the main problem is that SP may reach inefficient equilibria and so there is a large literature on pushing self-play toward higher payoff equilibria using various algorithmic innovations (Babes et al., 2008; Devlin et al., 2011; Devlin & Kudenko, 2016; Peysakhovich & Lerer, 2018). However, the setting where agents play with the same agents they have been trained with (aka. centralized training with decentralized execution) is quite different from the zero-shot coordination one which we study.

2.2. Cooperation and Coordination

A closely related problem to zero-shot coordination is ad-hoc teamwork (Stone et al., 2010; Barrett et al., 2011). For example: a robot agent joining another existing group of agents to play soccer (Barrett et al., 2011). Ad-hoc teamwork differs from the zero-shot coordination problem in that it is typically framed as a learning problem of learning the policies/capabilities of other agents during interaction whereas the pure zero-shot coordination scenario is one where there is no time to update a fixed policy that is constructed during training. These problems are closely linked and incorporating ideas from this literature into algorithms like OP is an interesting question for future research. However, another difference is that zero-shot agents only need to coordinate well with teams of agents that are optimized for the zero-shot setting, rather than arbitrary teams self-play of agents.

There is recent work looking at the situation where the one RL agent, trained separately, must join a group of new AI agents or humans (Lerer & Peysakhovich, 2018; Tucker et al., 2020; Carroll et al., 2019). These papers focus on using small amounts of observed behavior from partnered test time agents to either guide self-play to selecting the equilibrium (or "social convention") of the existing agents (Lerer & Peysakhovich, 2018; Tucker et al., 2020) or allow building a human model which can be used to learn an approximate best response using RL (Carroll et al., 2019). This setting is related, but zero-shot coordination gives no behavioral data to either agent to guide self-play or allow building a model of the other agent. Instead, zero-shot makes the assumption that test-time agents being themselves are optimized for the zero-shot setting (rather than the SP setting).

2.3. Game Theory and Tacit Coordination

Within behavorial game theory a large body of work considers coordination based on "focal points" or other shared grounding such as the famous "you lost your friend in New York City, where are you going to meet?" coordination problem (Schelling, 1980; Mehta et al., 1994). However, such focal points typically come from the fact that these coordination problems are not just abstract but grounded in *exogenous* features, *action labels*, that are meaningful due to a prior shared context. The zero-shot coordination setting thus is a special form of the tacit coordination problem

in which there are no shared exogenous features between the different agents and OP can be thought of as a way to coordinate in this setting.

There is also a large theoretical literature on learning and evolving coordination (Nowak, 2006). However, as with the self-play literature, it focuses on long run outcomes within a single group of agents learning or evolving together and does not typically focus on the question of engineering agents as we do.

2.4. Predicting Human Decision Making

Clearly, if we were able to accurately predict how our human counterparts are going to act in any given situation, the zero-shot coordination with human counterparts would reduce to learning a best response to those predicted actions. There is a large body of work using formal models to predict and understand human decision making (Camerer, 2011) and recent work that incorporates machine learning into this question (Wright & Leyton-Brown, 2010; Hartford et al., 2016; Peysakhovich & Naecker, 2017; Kleinberg et al., 2017; Fudenberg & Liang, 2019). However, the majority of this research focuses on extremely simple settings such as small normal form games (Wright & Leyton-Brown, 2010; Hartford et al., 2016; Fudenberg & Liang, 2019) or single decision problems (Peysakhovich & Naecker, 2017; Kleinberg et al., 2017) rather than complex cooperative settings with partial observability.

2.5. Domain Randomization

Our work is also related to the idea of domain randomization (Tobin et al., 2017). In RL and supervised learning domain randomization tries to make the realized model invariant to some feature of the environment. For example, an object detector should be invariant to the exact camera angle from which a view of an object is captured. OP applies a similar idea: a policy should be invariant to how an agent's partner breaks symmetries in the underlying game.

2.6. Exploiting Symmetries in Other Contexts

In the single agent context, it is harder to plan in MDPs that have more states. The idea of abstraction is to use underlying symmetries to 'compress' a large MDP into a simpler one, solve for the optimal strategy in the abstraction, and then lift the strategy to the original MDP. One set of such methods are MDP homomorphisms (van der Pol et al., 2020; Ravindran & Barto, 2004). These, like OP, use underlying symmetries but their goal is different: they want to find payoff maximizing policies for single agent decision problems, while OP seeks to find robust policies for zero-shot coordination. Note that as the lever game illustrates robust policies are not necessarily the payoff maximizing ones. In addition, these methods do not solve the problem of equilibrium selection among 'symmetric' policies in games, because the symmetry in the MDP just becomes a symmetry in the homomorphism.

A similar technique (compress, solve, then lift) is also used for finding Nash equilibria in large games like poker (Gilpin & Sandholm, 2007). In this case the abstraction treats 'isomorphic' states equally and thus reduces the effective number of states in the game. Again, the goal is different - poker abstractions are trying to find Nash equilibrium strategies in the original game while OP uses symmetries to select among a set of possible equilibria.

3. Zero-Shot Coordination

In this paper we study fully cooperative Markov games. To construct this environment we start out with a Dec-POMDP (Nair et al., 2003) with states $s_t \in \mathcal{S}$. There are $i = 1, \cdots N$ agents who each choose actions, $a_t^i \in \mathcal{A}$ at each time step.

The game is partially observable, $o_t^i \sim O(o|i, s_t)$ being each agent's stochastic observation function. At time t each agent has an action-observation history $\tau_t^i = \{o_0^i, a_0^i, r_0^i, \cdots, o_t^i\}$ and selects action a_i^t using stochastic policies of the form $\pi_\theta^i(a^i|\tau_t^i)$. The transition function, $P(s'|s, \mathbf{a})$, conditions on the joint action, $\mathbf{a}$.

The game is fully cooperative, agents share the reward r_t which is conditioned on the joint action and the state. Thus, the goal is to maximize the expected return $J = \mathbb{E}_\tau R(\tau)$, where $R(\tau) = \sum_t \gamma^t r_t$ is calculated using the discount factor γ.

Most work on cooperative MARL focuses on a setting where agents are trained together, although they must execute their policies independently at least at test time , *e.g.* (Lowe et al., 2017; Foerster et al., 2018a;b). The goal is to construct *learning rules*, i.e. functions that map Markov games to (joint) policies that select policies for each agent that together maximize expected discounted return. Because agents are trained together, these policies may be arbitrarily complex.

We are instead interested in achieving high returns with partners that were not trained together with our agent. Instead, we will frame the problem as follows: suppose that multiple independent AI designers will construct agents that have to interact in various but ex-ante unknown Dec-POMDPs without being able to coordinate beforehand, what learning rule should these designers agree on? To make this even more concrete, consider the case of independent autonomous vehicles made by multiple firms which have to interact in novel traffic situations on a daily basis.

The first key concept we introduce is the class of *equivalence mappings*, Φ, for a given Dec-POMDP.

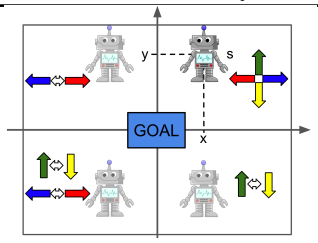

Figure 2.

Each element of Φ is a bijection of each of $\mathcal{S}$, O, and $\mathcal{A}$ onto itself, such that it leaves the Dec-POMDP unchanged:

$$\begin{aligned}\phi \in \Phi \iff & P(\phi(s')|\phi(s),\phi(a)) = P(s'|s,a) \\ & \wedge R(\phi(s'),\phi(a),\phi(s)) = R(s',a,s) \\ & \wedge O(\phi(o)|\phi(s),\phi(a),i) = O(o|s,a,i) \\ & \text{where equalities apply } \forall s', s, a'\end{aligned}$$

In other words, Φ describes the symmetries in the underlying Dec-POMDP. We note that our notation is heavily overloaded since each ϕ can act on actions, states and the observation function, so ϕ shorthand for $\phi = \{\phi_{\mathcal{S}}, \phi_{\mathcal{A}}, \phi_O\}$.

Next, we extend ϕ to also act on trajectories:

$$\phi(\tau_t^i) = \{\phi(o_0^i), \phi(a_0^i), \phi(r_0), \cdots, \phi(o_t^i)\}.$$

At this point an example might be helpful: consider a grid-world with a robot, shown in Figure 2, that can move in the 4 cardinal directions. In our example the goal is in the middle of the room, which leaves two axis of symmetry: We can invert either the x-axis, the y-axis or both, as long as we make the corresponding changes to the action space, for example mapping "up" to "down" and vice versa when inverting the y-axis.

In a similar way, we can extend ϕ to act on policies π, as follows:

$$\pi' = \phi(\pi) \iff \pi'(\phi(a)|\phi(\tau)) = \pi(a|\tau), \forall \tau, a$$

These symmetries are the "payoff irrelevant" parts of the Dec-POMDP. They come from the fact that the actions and states in the Dec-POMDP do not come with labels and so taking a policy and permuting it with respect to these symmetries does not change the outcome of interest: the trajectory and the reward.

It is precisely these symmetries that can cause problems for self-play trained agents. Since agents are trained together, they can coordinate on how to break symmetries. However, there is no guarantee that multiple SP agents trained separately will break symmetries in the same way. In this case when they are paired together their policies may fail spectacularly.

The goal of OP, then, will be to build policies which are maximally robust to this failure mode.

4. Other Play

We consider the 2 agent case for ease of notation with π^1, π^2 denoting each agent's component of the policy and π denoting the joint policy.

First, consider self-play (SP) learning rule. This is the learning rule which tries to optimize the following objective:

$$\pi^* = \arg\max_{\pi} J(\pi^1, \pi^2) \tag{1}$$

When the Dec-POMDP is tabular we can solve this via various methods. When it is not, deep reinforcement learning (deep RL) can be used to apply function approximation and gradient based optimization of this objective function. Though there is a large literature focusing on various issues in multi-agent optimization (Busoniu et al., 2006; Hernandez-Leal et al., 2019), our paper is agnostic to the precise method used.

These policies can be arbitrary and in complicated Dec-POMDPs multiple maxima to Equation 1 will often exist. These multiple policies can (and, as we will see in our experiments, often will) use coordinated symmetry breaking to receive high payoffs. Therefore, 2 matched, separately trained, SP agents will not necessarily receive the same payoff with each other as they receive with themselves.

To alleviate this issue, we need to make the optimization problem more robust to the symmetry breaking.

Let us consider the point of view of constructing a strategy for agent 1 where agent 2 will be the unknown novel partner. The *other-play* (OP) objective function for agent 1 maximizes expected return when randomly matched with a symmetry-equivalent policy of agent 2 rather than with a particular one. In other words, we perform a version of self-play where agents are not assumed to be able to coordinate on exactly how to break symmetries.

$$\pi^* = \arg\max_{\pi} \mathbb{E}_{\phi\sim\Phi}\ J(\pi^1, \phi(\pi^2)) \tag{2}$$

Here the expectation is taken with respect to a uniform distribution on Φ. We call this expected return J_{OP}. We will now consider what policies maximize J_{OP}.

Lemma 1.

$$J(\pi_A, \pi_B) = J(\phi(\pi_A^1), \phi(\pi_B^2)), \quad \forall \phi \in \Phi, \pi_A, \pi_B$$

This Lemma follows directly from the fact that the MDP is invariant to any $\phi \in \Phi$.

Lemma 2.

$$\{\phi \cdot \phi' : \phi' \in \Phi\} = \Phi\ ,\ \forall \phi \in \Phi$$

This Lemma follows from the fact that Φ is a bijection.

Proposition 1. *The expected OP return of π is equal to the expected return of each player independently playing a policy π^i_Φ which is the uniform mixture of $\phi(\pi^i)$ for all $\phi \in \Phi$.*

Proof.

$$J_{OP}(\pi) = \mathbb{E}_{\phi\sim\Phi}\ J(\pi^1, \phi(\pi^2)) \tag{3}$$
$$= \mathbb{E}_{\phi_1\sim\Phi,\phi_2\sim\Phi}\ J(\phi_1(\pi^1), \phi_1(\phi_2(\pi^2))) \tag{4}$$
$$= \mathbb{E}_{\phi_1\sim\Phi,\phi_2\sim\Phi}\ J(\phi_1(\pi^1), \phi_2(\pi^2)) \tag{5}$$
$$= J(\pi_\Phi) \tag{6}$$

(4) follows from Lemma 1, (5) follows from Lemma 2.

□

Corollary 1. *The distribution π^*_{OP} produced by OP will be the uniform mixture π_Φ with the highest return $J(\pi_\Phi)$.*

Let $\mathcal{L}_i$ be the set of learning rules which input a Dec-POMDP and output a policy for agent i.

A meta-equilibrium is a learning rule for each agent such that neither agent can improve their expected payoff by unilaterally deviating to a different learning rule.

Proposition 2. *If agent i uses OP as their learning rule then OP is a payoff maximizing learning rule for the agent's partner. Furthermore both agents using OP is the best possible meta-equilibrium.*

Proof. Since the Dec-POMDP has no labels for actions and states, $\mathcal{L}_i$ must choose all $\phi(\pi)$ with equal probability. Among these possible outputs, π^*_{OP} maximizes the return by Corollary 1. □

5. Implementing Other Play via Deep RL

We now turn to optimizing the OP objective function. In many applications of interest the Dec-POMDP is not tabular. Thus, deep RL algorithms use function approximation for the state space and attempt to find local maxima of Equation 1 using self-play reinforcement learning.

We show how to adapt this method to optimize the other-play objective (Equation 2). This amounts to applying a very specific kind of asymmetric domain randomization (Tobin et al., 2017) during training.

During each episode of MARL training, for each agent i a random permutation $\phi_i \in \Phi$ is chosen uniformly iid from Φ, and agent i observes and acts on $\phi(\mathcal{S}, O, \mathcal{A})$. Importantly, the agents act in different permutations of the same environment.

This environment randomization is equivalent to other-play, because the MDP remains constant under ϕ_i while the effect of agent i's policy on the environment is $\phi_i(\pi_i)$. The fixed points of independent optimization of π under this learning rule will be joint policies where each π_i is a BR to the uniform mixture of permutations of partner policies, i.e. precisely the permutation-invariant equilibria that are the solutions of other-play.

We note that OP is fundamentally compatible with any type of optimization strategy and can be applied whenever there are symmetries in the underlying MDP.

6. Experiments

We evaluate OP in two different settings. In each setting we will compare standard SP against OP. We will perform comparisons of agents trained together to agents trained separately that are placed into a zero-shot coordination test game.

6.1. Lever Game

We begin with the "lever game" mentioned in the introduction. This environment is tabular, there are only 10 actions possible per player. Here, during training, we use simple joint action learning, compute the true gradient with respect to the current policy and update. We show training time (i.e. expected reward with itself) and test time (zero-shot) coordination performance for both SP (optimizing equation 1) and OP (optimizing equation 2). The code is available as a notebook online here and can be executed online without downloading: `https://bit.ly/2vYkfI7`.

Figure 3 shows the results. As expected, OP agents coordinate on the unique option of 0.9 points both during the training phase and at test time. As a consequence, OP agents can carry out successful zero-shot coordination when paired with other OP agents.

In contrast, SP agents achieve higher rewards of 1.0 points during the training phase but entirely fail to coordinate with other, independently trained, SP agents.

6.2. Hanabi with AI Agents

We now turn to a much more complex environment. We construct agents for the cooperative card game Hanabi, which has recently been established a benchmark environment for multi-agent decision making in partially observable settings (Bard et al., 2020).

Hanabi is a cooperative card game with the interesting twist that players cannot see their own cards and hence have to rely on receiving information from the other player (who can see their hand). In Hanabi, there are two main ways of exchanging information: first of all, players can take

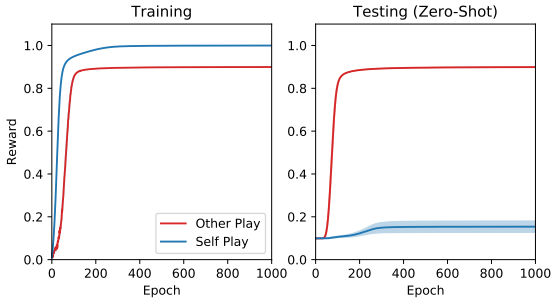

Figure 3. Train and test performance of self-play and other-play algorithms on the lever coordination game. Shown is the mean, shading is the standard error of the mean (s.e.m.), across 30 different seeds.

costly "hint" actions that point out subsets of cards based on rank or color. For example, hinting for "blue" reveals the color of all blue cards. Secondly, observing the actions themselves can be informative, in particular when players have pre-established conventions. The goal in Hanabi is to play cards in a legal order completing stacks of cards, one for each color. There are 5 color and 5 cards and the maximum points is 25. Players will lost a life token if they play a card out of order. Once they exhaust the deck or lose all 3 lives ("bomb out"), the game will terminate.

As noted, the vast majority of research on Hanabi has been in the self-play setting, in which a group of agents is trained to jointly obtain the highest possible score. To apply OP in Hanabi we note that assuming no side information, a permutation of the colors of the cards leaves the game unchanged. We use this as our class of symmetries.

6.3. MARL Training Details

OP can be applied on top of any SP algorithm. In Hanabi the Simplified Action Decoder (SAD) (Hu & Foerster, 2019) method achieves SOTA performance for RL agents. We use SAD as a base algorithm onto which we add OP. We use the open-sourced implementation of SAD as well as most of its hyper-parameters but with two major modifications. First, we use 2 GPUs for simulation instead of 1 as in the original paper. This doubles the data generation speed and has a profound effect on reducing the wall clock time required to achieve competitive performance. Second, we introduce extra hyper-parameters that control the network architecture to add diversity to the model capacity in order to better demonstrate the effectiveness of OP. Specifically, the network can have either 1 or 2 fully connected layers before 2 LSTM layers and can have an optional residual connection to by-pass the LSTM layers. For SP, we re-train the base SAD and the SAD + AUX variant proposed in Hu & Foerster (2019). SAD + AUX is specifically engineered for Hanabi by adding an auxiliary task to predict whether a card is playable, discardable, or unknown. We train agents with the aforementioned 4 different network architectures. We run each hyper-parameter configuration with 3 different seeds and thus 12 models are produced for each category of {SAD, SAD + AUX, SAD + OP, SAD + AUX + OP}.

6.4. Evaluation

We evaluate the models within the same category by pairing different models together to play the game, a process we refer to as *cross-play*. Clearly, if independent training runs ("seeds") from the same training method fail to coordinate with each other at test time it is unlikely they will coordinate with agents optimized by through a different process, let alone humans. As such, cross-play is a cheap proxy to evaluate whether a training method has potential for zero-shot coordination with human players. Figure 4 shows the scores obtained between all pairs of agents. Table 1 shows the average within-pair and cross-play scores. We see that SAD coordinates with itself but fails to coordinate with any other SAD agent. SAD with OP, however, significantly improves the cross-play. The effect is especially profound when the model has limited representation power. The top left corner of the graph, which corresponds to the simplest models that have only 1 fully connected layers, 2 LSTM layers and no residual connection, shows almost perfect cooperation scores. With the network growing more complicated, different strategies start to emerge and the cross-play performance drops. Auxiliary task implicitly improves cross-play scores by encouraging all agents to act basing on grounded information and confident predictions. Nonetheless, adding OP to SAD + AUX further improves performance and achieves the highest cross-play payoffs.

We can further study the policies resulting from these learning algorithms. Figure 5 picks the agent with highest cross-play performance in each category (top row) as well as their worst possible partner (bottom row) and presents $P(a^i_t \mid a^j_{t-1})$ over a subset of actions averaged over timesteps in 1000 episodes generated through self-play. In other words, we ask, do the agents respond very differently to possible actions of their partner? A large difference indicates that what an agent would do in a situation is very different from what their partner would do: a recipe for miscoordination!

We see that two paired SAD agents have very different policies and thus miscoordinate a lot. They also learn "inhuman" conventions that are hard for human to understand. For example, the agent hints Color5 to indicate discarding the 1st card while its partner interprets that as playing the 2nd card. OP eliminates these type of conventions. From the plot and our experience of playing with the SAD + OP agent, we find that it tends to use color hints to indicate either that the partner should save the card, or to disambiguate with a

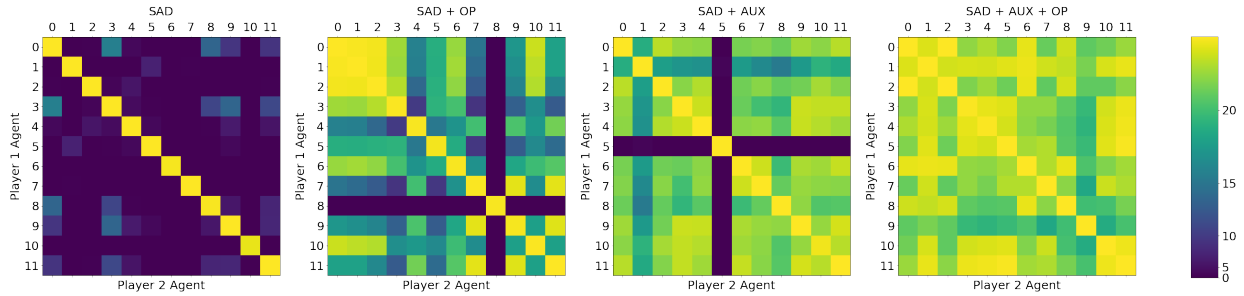

Figure 4. Cross-Play Matrix. Visualization of paired evaluation of different agents trained under the same method. y-axis represents the agent index of player 1 (first mover) and x-axis represents that of player 2. Agents 0-2: 1-layer FC, without residual connection; Agents 3-5: 1-layer FC, with residual connection; Agents 6-8: 2-layer FC, without residual connection; Agents 9-11: 2-layer FC, with residual connection. All agents have 2-layer LSTMs after the FC layers. Each block in the grid is obtained by evaluating the pair on 10K games with different seeds. Please refer to Table 1 for numeric results.

Method	Cross-Play	Cross-Play(*)	Self-Play
SAD	2.52 ± 0.34	3.02 ± 0.39	23.97 ± 0.04
SAD + OP	15.32 ± 0.65	18.28 ± 0.36	23.93 ± 0.02
SAD + AUX	17.65 ± 0.69	21.09 ± 0.18	$\mathbf{24.09 \pm 0.03}$
SAD + AUX + OP	$\mathbf{22.07 \pm 0.11}$	$\mathbf{22.49 \pm 0.18}$	24.06 ± 0.02

Table 1. Cross-Play Performance. The average performance of pairs of agents that are train with the same method but different network architecture and/or seeds. Please refer to Figure 4 for visualization of performance for each individual pair. Cross-Play score is non-diagonal mean of each grid. Cross-Play(*) is the cross-play score after removing the worst model from the grid. Self-Play score is the score attained when agents play with the partner they are trained with.

subsequent rank hint. This is not a typical strategy played by seasoned human players but is easy understand and thus makes the agent easier to cooperate with. However, due to the way we implement OP in Hanabi, it is still possible to form secretive conventions such as using all color hints to indicate a specific move. For example, the worst partner of SAD + OP uses all color hints to indicate playing the 5th card.

6.5. Hanabi with Humans

So far we have focused on AI agents that play other AI agents and have shown that OP is a meta-equilibrium with respect to learning rules in the zero-shot setting.

We now ask: do human strategies in Hanabi also have an OP-like quality? In other words, do OP agents perform well with humans?

To begin to answer this question we recruited 20 individuals from a board game club. These individuals were familiar with Hanabi but not expert players. We asked each individual to play a game of Hanabi with two bots, in random order, using the user interface open-sourced by (Lerer et al., 2019). We note that we did not provide the participants with any information about the bots, either regarding their strategy or the method through which they were trained.

For testing we selected our best SAD + AUX + OP agent based on the cross-play performance (henceforth OP bot). We also have individuals play with the SOTA self-play agent from (Hu & Foerster, 2019) (henceforth SP bot). We download models from their GitHub repo and pick the model based on cross-play scores. For reference, the SP model used here gets 23.99 in self-play and 20.99 in cross-play with other agents where the only difference among them is seed.

Since in Hanabi the exact deck being used can make a huge difference (for example, some hands are unwinnable), to reduce the variance of our results we play each seed by two different players, one for our OP agent, and one for the control. Importantly, to prevent any adaptation advantages, we alternate the order between which bot came first across different participants.

Humans achieved an average score of 15.75 (s.e.m. 1.23) with the OP bot and "bombed out" 45% of games. Thus the OP bot, which has high cross-play scores with other OP bots is also able to play with humans. Note, in our counting convention players keep the current score when the bomb out, which be believe is more appropriate for the zero-shot setting.

By comparison, humans paired with the SP bot achieve an average score of 9.15 (s.e.m. 1.18) and an 85% bomb

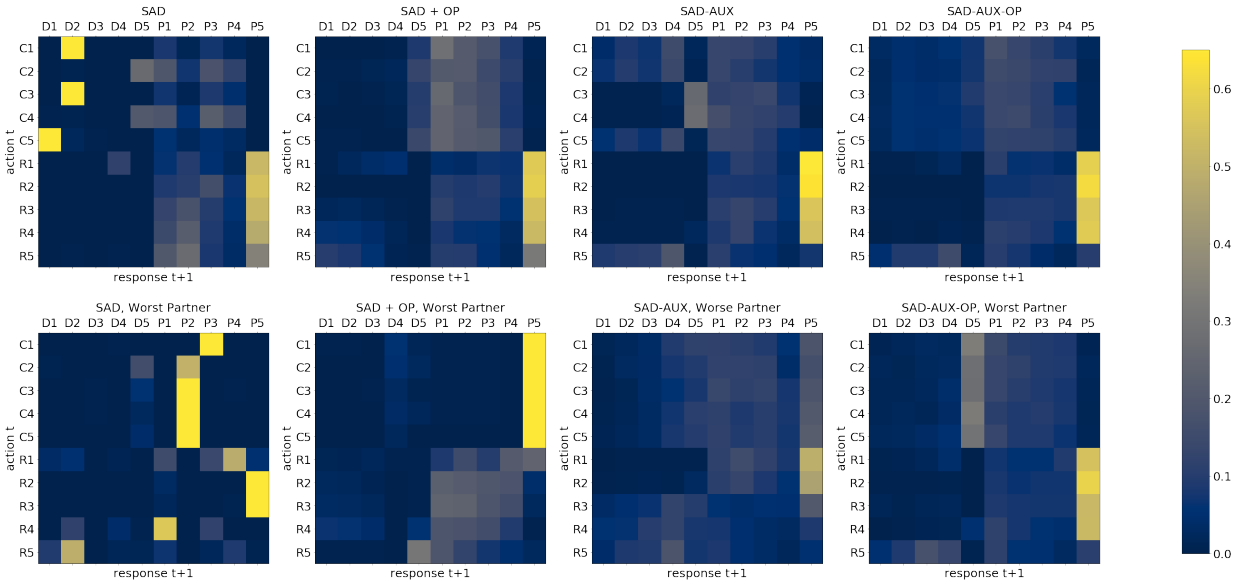

Figure 5. $P(a_t^i \mid a_{t-1}^j)$ Matrices. Each subplot can be used as a proxy to roughly decipher conventions of an agent. The y-axis means the action taken at time-step t and x-axis means the percentage of each action as response at time-step $t+1$. Here we only show a quarter of the matrix that corresponds to the interaction between color/rank hint and play/discard positions. C1-C5 and R1-R5 mean hinting 5 different colors and ranks respectively while D1-D5 and P1-P5 mean discarding and playing 1st-5th cards of the hand. For each plot, we take an agent and run 1000 episodes of self-play to compute statistics. The agents that achieved the highest cross-play scores in Figure 4 are used to generate the top row and their worst partners are chosen to render the bottom row.

rate. To the best of our knowledge, the only other research involving human-AI collaboration in Hanabi is (Eger et al., 2017). Here, a hand-coded AI agent, designed to play well with humans is used and achieves an average of around 15.0 points when paired with humans. Beyond the average scores and "bomb-out" rates, we thus also have access to pairwise comparisons for the two bots when playing two different people on the same deck.

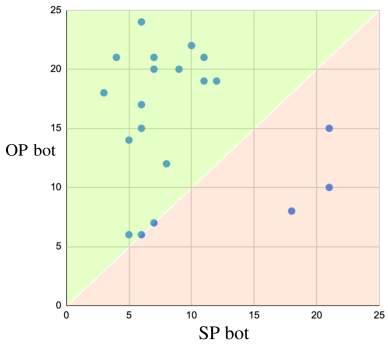

Figure 6. Shown are all scores obtained by human testing. Each blue dot is one seed, when humans are matched with the SP bot (x-axis) and the OP bot (y-axis).

These preliminary numbers confirm that our OP bot significantly outperformed the SOTA self-play bot from (Hu & Foerster, 2019) when paired with humans. In particular, OP won 15 out of the 20 per-seed comparisons and tied in 2 cases, losing to the control group for 3 seeds (p=0.004[1]).

Of course, these results do not suggest that OP will work in *every* zero-shot coordination where AI agents need to cooperate with humans. However, they are encouraging and suggest that OP is a fruitful research direction for the important problem of human-AI coordination.

7. Other Attempts

While attempting to make progress on the zero-shot coordination problem in Hanabi we tried a variety of approaches. Here we discuss some other approaches that seemed promising but did not yield agents that were able to coordinate with agents they were not trained with. While this does not necessarily mean these approaches are doomed to failure, we report these results as information for other researchers interested in this problem.

In particular we tried multi-agent RL adaptations of cognitive hierarchies (Stahl, 1993), k-level reasoning (Costa-Gomes et al., 2001) and training a population of agents. Our original inspiration was that both cognitive hierarchies and k-level reasoning should reduce the tendency towards arbitrary symmetry breaking and have been shown to pro-

[1] exact binomial test of the null hypothesis being that P(OP higher score) $\leq$ P(control higher score).

duce human like decision making in other settings (Wright & Leyton-Brown, 2010). Similarly, population based approaches are gaining popularity for regularizing communication protocols in the field of emergent communication, see e.g. (Fitzgerald, 2019; Tieleman et al., 2018; Lowe et al., 2019). We found that none of these approaches produced high cross-play performance in Hanabi, which we now consider a necessary condition for high zero-shot performance with humans. In hind-sight, considering that all of these approaches would necessarily fail in the matrix game example, this is not at all surprising. Still, to help future researchers learn from our endeavours, we are adding all results to the supplementary material and will open-source the corresponding agents.

8. Conclusion

We have shown that a simple expansion of self-play which we call *other-play* can construct agents that are better able to zero-shot coordinate with partners they have not seen before. We have proven theoretical properties of the OP strategy, shown how to implement it with deep RL, and shown in experiments with the cooperative card game Hanabi that OP can construct robust agents that can play with other AIs as well as with humans.

We do not claim that OP is a silver bullet for all zero-shot coordination problems. In particular, because OP is a modification of the SP algorithm, it can be combined with many of the algorithmic innovations that have been developed to improve SP in various games (Lanctot et al., 2017; Foerster et al., 2018a; Lowe et al., 2017; Foerster et al., 2017). Thus, we believe that this represents an exciting research direction for those interested in moving deep RL beyond two-player, zero-sum environments to ones involving coordination and cooperation. Currently we are assuming that the symmetries Φ are given to the algorithm. However, in principle, discovering the symmetries of an MDP is another optimization problem, which opens interesting avenues for future work.

Acknowledgements

We would like to thank Noam Brown for encouraging discussions and Pratik Ringshia for help with the UI for human testing. We would also like to thank our human participants for offering to help.

References

Babes, M., Munoz de Cote, E., and Littman, M. L. Social reward shaping in the prisoner's dilemma. *International Conference on Autonomous Agents and Multiagent Systems*, pp. 1389–1392, 2008.

Bard, N., Foerster, J. N., Chandar, S., Burch, N., Lanctot, M., Song, H. F., Parisotto, E., Dumoulin, V., Moitra, S., Hughes, E., et al. The hanabi challenge: A new frontier for ai research. *Artificial Intelligence*, 280:103216, 2020.

Barrett, S., Stone, P., and Kraus, S. Empirical evaluation of ad hoc teamwork in the pursuit domain. In *AAMAS*, pp. 567–574, 2011.

Brown, N. and Sandholm, T. Superhuman ai for heads-up no-limit poker: Libratus beats top professionals. *Science*, 359(6374):418–424, 2018.

Busoniu, L., Babuska, R., and De Schutter, B. Multi-agent reinforcement learning: A survey. In *2006 9th International Conference on Control, Automation, Robotics and Vision*, pp. 1–6. IEEE, 2006.

Camerer, C. F. *Behavioral game theory: Experiments in strategic interaction*. Princeton University Press, 2011.

Campbell, M., Hoane Jr, A. J., and Hsu, F.-h. Deep blue. *Artificial intelligence*, 134(1-2):57–83, 2002.

Carroll, M., Shah, R., Ho, M. K., Griffiths, T., Seshia, S., Abbeel, P., and Dragan, A. On the utility of learning about humans for human-ai coordination. In *Advances in Neural Information Processing Systems*, pp. 5175–5186, 2019.

Costa-Gomes, M., Crawford, V. P., and Broseta, B. Cognition and behavior in normal-form games: An experimental study. *Econometrica*, 69(5):1193–1235, 2001.

Devlin, S. and Kudenko, D. Plan-based reward shaping for multi-agent reinforcement learning. *The Knowledge Engineering Review*, 31(1):44–58, 2016.

Devlin, S., Kudenko, D., and Grześ, M. An empirical study of potential-based reward shaping and advice in complex, multi-agent systems. *Advances in Complex Systems*, 14 (02):251–278, 2011.

Eger, M., Martens, C., and Córdoba, M. A. An intentional ai for hanabi. In *2017 IEEE Conference on Computational Intelligence and Games (CIG)*, pp. 68–75. IEEE, 2017.

Fitzgerald, N. To populate is to regulate. *arXiv preprint arXiv:1911.04362*, 2019.

Foerster, J., Assael, I. A., De Freitas, N., and Whiteson, S. Learning to communicate with deep multi-agent reinforcement learning. In *Advances in neural information processing systems*, pp. 2137–2145, 2016.

Foerster, J., Nardelli, N., Farquhar, G., Afouras, T., Torr, P. H., Kohli, P., and Whiteson, S. Stabilising experience replay for deep multi-agent reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 1146–1155. JMLR. org, 2017.

Foerster, J. N., Farquhar, G., Afouras, T., Nardelli, N., and Whiteson, S. Counterfactual multi-agent policy gradients. In *Thirty-second AAAI conference on artificial intelligence*, 2018a.

Foerster, J. N., Song, F., Hughes, E., Burch, N., Dunning, I., Whiteson, S., Botvinick, M., and Bowling, M. Bayesian action decoder for deep multi-agent reinforcement learning. *arXiv preprint arXiv:1811.01458*, 2018b.

Fudenberg, D. and Liang, A. Predicting and understanding initial play. *American Economic Review*, 109(12):4112–41, 2019.

Gilpin, A. and Sandholm, T. Lossless abstraction of imperfect information games. *Journal of the ACM (JACM)*, 54 (5):25–es, 2007.

Hartford, J. S., Wright, J. R., and Leyton-Brown, K. Deep learning for predicting human strategic behavior. In *Advances in Neural Information Processing Systems*, pp. 2424–2432, 2016.

Hernandez-Leal, P., Kartal, B., and Taylor, M. E. A survey and critique of multiagent deep reinforcement learning. *Autonomous Agents and Multi-Agent Systems*, 33(6):750–797, 2019.

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

Kleiman-Weiner, M., Ho, M. K., Austerweil, J. L., Littman, M. L., and Tenenbaum, J. B. Coordinate to cooperate or compete: abstract goals and joint intentions in social interaction. In *CogSci*, 2016.

Kleinberg, J., Liang, A., and Mullainathan, S. The theory is predictive, but is it complete? an application to human perception of randomness. In *Proceedings of the 2017 ACM Conference on Economics and Computation*, pp. 125–126, 2017.

Lanctot, M., Zambaldi, V., Gruslys, A., Lazaridou, A., Tuyls, K., Pérolat, J., Silver, D., and Graepel, T. A unified game-theoretic approach to multiagent reinforcement learning. In *Advances in Neural Information Processing Systems*, pp. 4190–4203, 2017.

Lazaridou, A., Peysakhovich, A., and Baroni, M. Multi-agent cooperation and the emergence of (natural) language. *arXiv preprint arXiv:1612.07182*, 2016.

Lerer, A. and Peysakhovich, A. Maintaining cooperation in complex social dilemmas using deep reinforcement learning. *arXiv preprint arXiv:1707.01068*, 2017.

Lerer, A. and Peysakhovich, A. Learning social conventions in markov games. *arXiv preprint arXiv:1806.10071*, 2018.

Lerer, A., Hu, H., Foerster, J., and Brown, N. Improving policies via search in cooperative partially observable games. *arXiv preprint arXiv:1912.02318*, 2019.

Lowe, R., Wu, Y., Tamar, A., Harb, J., Abbeel, O. P., and Mordatch, I. Multi-agent actor-critic for mixed cooperative-competitive environments. In *Advances in neural information processing systems*, pp. 6379–6390, 2017.

Lowe, R., Gupta, A., Foerster, J., Kiela, D., and Pineau, J. Learning to learn to communicate, 2019.

Mehta, J., Starmer, C., and Sugden, R. The nature of salience: An experimental investigation of pure coordination games. *The American Economic Review*, 84(3): 658–673, 1994.

Nair, R., Tambe, M., Yokoo, M., Pynadath, D., and Marsella, S. Taming decentralized pomdps: Towards efficient policy computation for multiagent settings. In *IJCAI*, volume 3, pp. 705–711, 2003.

Nowak, M. A. *Evolutionary dynamics: exploring the equations of life*. Harvard University Press, 2006.

Peysakhovich, A. and Lerer, A. Prosocial learning agents solve generalized stag hunts better than selfish ones. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 2043–2044. International Foundation for Autonomous Agents and Multiagent Systems, 2018.

Peysakhovich, A. and Naecker, J. Using methods from machine learning to evaluate behavioral models of choice under risk and ambiguity. *Journal of Economic Behavior & Organization*, 133:373–384, 2017.

Ravindran, B. and Barto, A. G. Approximate homomorphisms: A framework for non-exact minimization in markov decision processes. 2004.

Resnick, C., Kulikov, I., Cho, K., and Weston, J. Vehicle community strategies. *arXiv preprint arXiv:1804.07178*, 2018.

Schelling, T. C. *The strategy of conflict*. Harvard university press, 1980.

Shum, M., Kleiman-Weiner, M., Littman, M. L., and Tenenbaum, J. B. Theory of minds: Understanding behavior in groups through inverse planning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 6163–6170, 2019.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):354–359, 2017.

Stahl, D. O. Evolution of smartn players. *Games and Economic Behavior*, 5(4):604–617, 1993.

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

Sukhbaatar, S., Fergus, R., et al. Learning multiagent communication with backpropagation. In *Advances in neural information processing systems*, pp. 2244–2252, 2016.

Sunehag, P., Lever, G., Gruslys, A., Czarnecki, W. M., Zambaldi, V., Jaderberg, M., Lanctot, M., Sonnerat, N., Leibo, J. Z., Tuyls, K., et al. Value-decomposition networks for cooperative multi-agent learning based on team reward. In *Proceedings of the 17th international conference on autonomous agents and multiagent systems*, pp. 2085–2087. International Foundation for Autonomous Agents and Multiagent Systems, 2018.

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

Tieleman, O., Lazaridou, A., Mourad, S., Blundell, C., and Precup, D. Shaping representations through communication. *OpenReview*, 2018.

Tobin, J., Fong, R., Ray, A., Schneider, J., Zaremba, W., and Abbeel, P. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pp. 23–30. IEEE, 2017.

Tucker, M., Zhou, Y., and Shah, J. Adversarially guided self-play for adopting social conventions. *arXiv preprint arXiv:2001.05994*, 2020.

van der Pol, E., Kipf, T., Oliehoek, F. A., and Welling, M. Plannable approximations to mdp homomorphisms: Equivariance under actions. *arXiv preprint arXiv:2002.11963*, 2020.

Wright, J. R. and Leyton-Brown, K. Beyond equilibrium: Predicting human behavior in normal-form games. In *Twenty-Fourth AAAI Conference on Artificial Intelligence*, 2010.

A. Details on Other Attempts

k	Cross-Play	Self-Play
1	1.06 ± 0.04	1.04 ± 0.06
2	0.95 ± 0.18	0.99 ± 0.33
3	1.49 ± 0.11	2.63 ± 0.34
4	2.48 ± 0.22	5.89 ± 0.75
5	2.04 ± 0.35	7.22 ± 0.56

Table 2. Cognitive Hierarchies Performance. We train CH for 5 levels with 3 seeds. The cross-play and self-play results are computed by averaging scores of intra-level pairing of agents trained with different seeds. Cross-play score is averaged over 6 pairs and self-play score is averaged over 3 pairs for each cell.

In this section we provide more details on other attempts mentioned previously.

The core idea behind cognitive hierarchies (CH) (Stahl, 1993) and k-level reasoning (Costa-Gomes et al., 2001) is to train a sequence of K agents of different capabilities. The final agents may hopefully learn strategies that cross-play well through such explicit route of evolution. In our implementation of CH, the first agent $a^{(0)}$ in the sequence is a random agent that pick actions uniformly regardless of the state. The kth agent $a^{(k)}$ is trained to be the "best response" to the pool of agents $\{a^{(0)}, ..., a^{(k-1)}\}$. Intuitively, this means that the first agent will learn to play based only on the hinted facts, i.e. ground information, to play Hanabi because $a^{(0)}$'s actions contain no intentions nor conventions. $a^{(2)}$ can then learn to give more useful hints and the subsequent agents may learn more complicated behaviors. In k-level, the kth agent $a^{(k)}$ only learns to best respond $a^{(k-1)}$ and other aspects remain the same. Because the agents are trained with a random agent $a^{(0)}$ and they will "bomb out" inevitably, we alter the reward scheme so that the agents receive reward 0, instead of the negative of current score, when they lose all life tokens.

The performance of CH is shown in Table 2. The most prominent phenomenon is that CH converges quite slowly, due to the fact that it needs to cooperate with a pool of different yet primitive policies. For each level, we train the models until convergence and 5 levels normally take several days to complete. For reference, it roughly takes less than 20 hours for SAD and our other-play agents to reach 23 points in self-play under the same settings and hardware. The prohibitive cost in time and computation make CH unsuitable for complicated tasks like MARL in Hanabi. Moreover, even though the self-play score is low, we can already see a clear performance gap between self-play and cross-play, making it safe to assume that CH will not work well in zero-shot coordination.

In Table 3, we show results of K-Level method trained for 10 levels. Despite the gap between cross-play and self-

k	Cross-Play	Self-Play
1	0.73 ± 0.18	0.95 ± 0.36
2	0.50 ± 0.02	0.54 ± 0.13
3	2.99 ± 0.11	3.24 ± 0.24
4	1.71 ± 0.12	2.57 ± 0.11
5	6.14 ± 0.58	7.27 ± 1.48
6	2.08 ± 0.22	4.52 ± 1.05
7	6.28 ± 0.92	8.82 ± 2.17
8	1.82 ± 0.25	4.89 ± 1.95
9	6.87 ± 0.91	10.26 ± 2.52
10	2.05 ± 0.28	6.54 ± 2.65

Table 3. K-Level Performance. We train K-Level with $K = 10$ and 3 seeds. The cross-play and self-play scores are computed by intra-level pairing of agents trained with different seeds.

play being smaller, this method suffers from non-monotonic improvements between levels and the same high cost in time and sample complexity.

	Population1	Population2
Population1	23.46 ± 0.01	19.97 ± 0.06
Population2	20.16 ± 0.06	23.44 ± 0.01

Table 4. Performance of Population Based Method. Each cell is computed by pairing all agents from one population with those from the other population and then average the scores. Diagonal can be seen as self-play score and non-diagonal can be seen as cross-play score under a population setting.

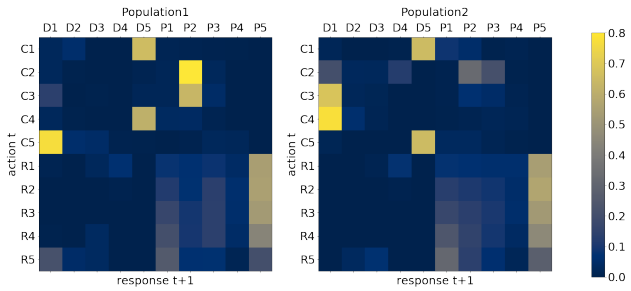

Figure 7. $P(a_t^i \mid a_{t-1}^j)$ matrices of one model from each population. The semantic of the visualization is identical to Figure 5.

Different from CH and K-Level where agents are trained sequentially, population based approaches (Fitzgerald, 2019; Tieleman et al., 2018; Lowe et al., 2019) train agents simultaneously by pairing distinct agents together to generate samples. We briefly experimented with a simple population setting where we initialize N different agents with their private replay buffers. They are uniformly paired together with each other to generate samples and write their observation action sequences into the their own buffer. Each agent is optimized independently at each training step. We train 2 populations with different seeds. Each of them contains 4 agents initialized differently. The numerical results are shown in Table 4. This method can achieve decent cross-play scores. It is worth noting that the diversity between the hyper-parameters of the two populations is much smaller than that of the experiments shown in Figure 4 so that they are not directly comparable. However, a closer look at their respective policy through $P(a_t^i \mid a_{t-1}^j)$ matrices reveals the problem. The way they use color hints not only differs greatly from each other but also breaks the color symmetry of the game, which is the exact problem other-play tries to solve. Qualitatively they are hard for human to play with. They manage to achieve good cross-play scores because the agents seldom use color hints.