A Equivariance Property Proofs

Here we write the proofs for the Symmetric and Fixing Properties of our methodology introduced in Section 3, which are analogous to those of [35]:

Proposition. *(Symmetric Property) $S(\psi) \in \mathbf{\Psi}_{equiv}$, for all $\psi \in \mathbf{\Psi}$; that is, S maps neural networks to equivariant neural networks.*

Proof. Without loss of generality take $g' \in G$ and $\psi \in \mathbf{\Psi}$. Then,

$$\begin{aligned}
\mathbf{K}_{g'}^{-1} S(\psi)(\mathbf{L}_{g'}\mathbf{x}) &= \mathbf{K}_{g'}^{-1} \frac{1}{|G|} \sum_{g \in G} \mathbf{K}_g^{-1} \psi(\mathbf{L}_g \mathbf{L}_{g'} \mathbf{x}) \\
&= \frac{1}{|G|} \sum_{g \in G} \mathbf{K}_{g'}^{-1} \mathbf{K}_g^{-1} \psi(\mathbf{L}_g \mathbf{L}_{g'} \mathbf{x}) \\
&= \frac{1}{|G|} \sum_{g \in G} \mathbf{K}_{gg'}^{-1} \psi(\mathbf{L}_{gg'} \mathbf{x}) \\
&= \frac{1}{|G|} \sum_{g'^{-1} m \in G} \mathbf{K}_m^{-1} \psi(\mathbf{L}_m \mathbf{x}) \\
&= \frac{1}{|G|} \sum_{m \in g' G} \mathbf{K}_m^{-1} \psi(\mathbf{L}_m \mathbf{x}) \\
&= \frac{1}{|G|} \sum_{m \in G} \mathbf{K}_m^{-1} \psi(\mathbf{L}_m \mathbf{x}) \\
&= S(\psi)(\mathbf{x}).
\end{aligned}$$

Thus $\mathbf{K}_{g'} S(\psi)(\mathbf{x}) = S(\psi)(\mathbf{L}_{g'}\mathbf{x})$. □

Proposition. *(Fixing Property) $\mathbf{\Psi}_{equiv} = Ran(S)$; that is, the range of S covers the entire equivariant subspace.*

Proof. By Proposition 1, we have $\mathrm{Ran}(S) \subset \mathbf{\Psi}_{\mathrm{equiv}}$. It thus suffices to show $\mathbf{\Psi}_{\mathrm{equiv}} \subset \mathrm{Ran}(S)$. Take $\psi \in \mathbf{\Psi}_{\mathrm{equiv}}$. Then,

$$\begin{aligned}
S(\psi) &= \frac{1}{|G|} \sum_{g \in G} \mathbf{K}_g^{-1} \psi(\mathbf{L}_g) \\
&= \frac{1}{|G|} \sum_{g \in G} \mathbf{K}_g^{-1} \mathbf{K}_g \psi \\
&= \frac{1}{|G|} \sum_{g \in G} \psi \\
&= \psi,
\end{aligned}$$

where we have used Equation 2 since $\psi \in \mathbf{\Psi}_{\mathrm{equiv}}$. Thus, ψ is a fixed point of S, and so $\psi \in \mathrm{Ran}(S)$. □

B Optimal Meta-Equilibrium Proof

Proposition. *Assume $G = \Phi$. If an agent chooses either the naive approach or the G-OP learning rule, then choosing either of these is payoff maximizing for the agent's partner. In addition, both players choosing either the first approach or the G-OP learning rule is the best possible meta-equilibrium.*

Proof. If an agent π^1 has an architecture $\pi^1 \in \mathrm{Ran}(S)$ or uses the G-OP learning rule, then they necessarily have the objective to maximize

$$\frac{1}{|G|}\sum_{\phi \in G} J(\pi^1, \phi(\pi^2)),$$

because either of these choices forces consideration of all $\phi \in G$ equally. Since $G = \Phi$, we have

$$\frac{1}{|\Phi|}\sum_{\phi \in \Phi} J(\pi^1, \phi(\pi^2)) = \mathbb{E}_{\phi \in \Phi} J(\pi^1, \phi(\pi^2)),$$

where the expectation is taken with respect to the uniform distribution of Φ. We have thus derived the OP objective, and so what we sought so show is a corollary of [22] (Proposition 2 in their paper). □

C Averaging Over Very Different Distributions

This passage concerns why if given a policy that acts very differently in different symmetries, which we then symmetrize with respect to a large group G, the resulting policy would be one that selects actions approximately uniformly at random.

Consider l categorical distributions $\mu_1, \dots, \mu_l$ sampled uniformly from the collection of all categorical distributions over k categories. Denoting $\mu_{i,j}$ as the probability of the j^{th} category of distribution i, by symmetry in the categories and by assumption we have that $\mu_{a,r}$ and $\mu_{b,s}$ are equally distributed for all $1 \leq r, s \leq k$, $1 \leq a, b \leq l$. Therefore, by the law of large numbers, $\frac{1}{l}\sum_{i=1}^{l} \mu_{i,r} \approx \frac{1}{l}\sum_{i=1}^{l} \mu_{i,s}$ when l is large, for all $1 \leq r, s \leq k$. But since $\frac{1}{l}\sum_{i=1}^{l} \mu_i$ is a probability distribution, this implies $\frac{1}{l}\sum_{i=1}^{l} \mu_{i,j} \to \frac{1}{k}$ as l becomes large, for all $1 \leq j \leq k$; i.e., $\frac{1}{l}\sum_{i=1}^{l} \mu_i$ approaches the uniform distribution over k values.

The reader may identify $l = |G|$, k as the number of legal actions, and each μ_i as the policy's action distribution under symmetry i, where we took each μ_i uniformly at random to mimic a policy that acts entirely differently per Dec-POMDP symmetry. While the uniform distribution over actions is indeed symmetry-equivariant (and invariant), it is not in general useful.

D Hanabi

Hanabi is a cooperative card game that can be played with 2 to 5 people. Hanabi is a popular game, having been crowned the 2013 "Spiel des Jahres" award, a German industry award given to the best board game of the year. Hanabi has been proposed as an AI benchmark task to test models of cooperative play that act under partial information [5]. To date, Hanabi has one of the largest state spaces of all Dec-POMDP benchmarks.

The deck of cards in Hanabi is comprised of five colors (white, yellow, green, blue and red), and five ranks (1 through 5), where for each color there are three 1's, two each of 2's, 3's and 4's, and one 5, for a total deck size of fifty cards. Each player is dealt five cards (or four cards if there are 4 or 5 players). At the start, the players collectively have eight information tokens and three fuse tokens, the uses of which shall be explained presently.

In Hanabi, players can see all other players' hands but their own. The goal of the game is to play cards to collectively form five consecutively ordered stacks, one for each color, beginning with a card of rank 1 and ending with a card of rank 5. These stacks are referred to as fireworks, as playing the cards in order is meant to draw analogy to setting up a firework display[4].

[4] Hanabi (花火) means 'fireworks' in Japanese.

We call the player whose turn it is the active agent. The active agent must conduct one of three actions:

- **Hint** - The active agent chooses another player to grant a hint to. A hint involves the active agent choosing a color or rank, and revealing to their chosen partner all cards in the partner's hand that satisfy the chosen color or rank. Performing a hint exhausts an information token. If the players have no information tokens, a hint may not be conducted and the active agent must either conduct a discard or a play.
- **Discard** - The active agent chooses one of the cards in their hand to discard. The identity of the discarded card is revealed to the active agent and becomes public information. Discarding a card replenishes an information token should the players have less than eight.
- **Play** - The active agent attempts to play one of the cards in their hand. The identity of the played card is revealed to the active agent and becomes public information. The active agent has played successfully if their played card is the next in the firework of its color to be played, and the played card is then added to the sequence. If a firework is completed, the players receive a new information token should they have less than eight. If the player is unsuccessful, the card is discarded, without replenishment of an information token, and the players lose a fuse token.

The game ends when all three fuse tokens are spent, when the players successfully complete all five fireworks, or when the last card in the deck is drawn and all players take one last turn. If the game finishes by depletion of all fuse tokens (i.e. by "bombing out"), the players receive a score of 0. Otherwise, the score of the finished game is the sum of the highest card ranks in each firework, for a highest possible score of 25.

E Experiment Details

Many of the hyperparameter choices are taken from [21, 22]. The main body of the network for each trained agent consists of 1 fully connected layer with a hidden dimension of 512, 2 LSTM layers of 512 units, and two output heads for value and advantage, respectively. The networks are updated using the Adam optimizer [26] with learning rate 6.25×10^{-5} and $\epsilon = 1.5 \times 10^{-5}$. We use a batchsize of 128 for training. The replay buffer size is 10^5 episodes, and is warmed up with 10^4 episodes before training commences. We use two GPUs for asynchronous actors to collect rollouts and record to an experience replay, and one GPU for computing gradients and model updates. The trainer sends its network weights to all actors every 10 updates and the target network is synchronized with the online network every 2.5×10^3 updates. Training was concluded after 10^3 epochs, where each model reached (or near reached) convergence; each epoch consisted of 10^3 updates. The compute we used were 3 NVIDIA GeForce RTX 2080 Ti GPUs and 40 CPU cores.

Our complete code for training and symmetrizing agents can be found in our GitHub repo: `https://github.com/gfppoy/equivariant-zsc`. This repo is based off the Off-Belief Learning codebase [23] and the Hanabi Learning Environment (HLE).

F Comparing LSTM Modification choices

Here we compare the two design choices mentioned in Section 3.1: namely 1) $h_t = \frac{1}{|G|}\sum_{g \in G} h_{t,g}$ and $c_t = \frac{1}{|G|}\sum_{g \in G} c_{t,g}$ (*averaging*); and 2) $h_t = h_{t,e}$ and $c_t = c_{t,e}$ (*identity*). Table 3 summarizes the results of symmetrizing the policies in Section 4.2 at test time using the dihedral group, using either the averaging or identity schemes. We can see the averaging scheme tends to slightly outperform the identity, an interesting finding given the counter-intuitive nature of the design choice.

Table 3: Cross-play (XP) scores and bombout rates of various diverse policy types symmetrized at test time. Agents are symmetrized with respect to the dihedral group, and we compare the averaging and identity schemes as described above. Each agent is trained with a different seed. Each pair of agents was evaluated over 5000 games, with the total averages compiled here. The error bars are the standard error of the mean.

XP Stats	SAD	IQL	OP	OBL-L5
W/o symmetrizer	2.52 ± 0.34	10.53 ± 0.78	15.32 ± 0.65	23.77 ± 0.06
$h_t = h_{t,e},\ c_t = c_{t,e}$	3.41 ± 0.41	13.32 ± 0.68	16.35 ± 0.54	23.88 ± 0.04
$h_t = \frac{1}{\lvert G\rvert}\sum_{g \in G} h_{t,g},\ c_t = \frac{1}{\lvert G\rvert}\sum_{g \in G} c_{t,g}$	3.61 ± 0.39	13.62 ± 0.65	16.48 ± 0.53	23.89 ± 0.04

G Conditional Action Matrices

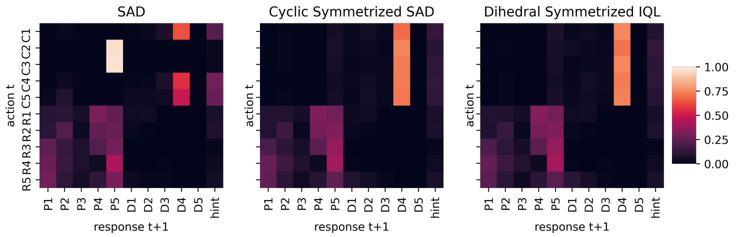

Figure 3: Conditional action matrices of SAD, i.e. $P(a_t^i \mid a_{t-1}^j)$, unsymmetrized (left) and symmetrized at test time (middle is C_5-symmetrized and right is D_{10}-symmetrized). The y-axis represents the action taken at timesetep t and the x-axis shows the proportion of each action as response at timestep $t+1$. The matrices show the interactions between color/rank hinting and play/discarding. C1-5 and R1-5 mean hinting the 5 different colors and ranks respectively, and P1-5 and D1-5 mean playing and discarding the 1st-5th cards in the hand. We selected a random agent, and each plot is thereby computed by running 1000 episodes of self-play with the agent to compute the statistics.

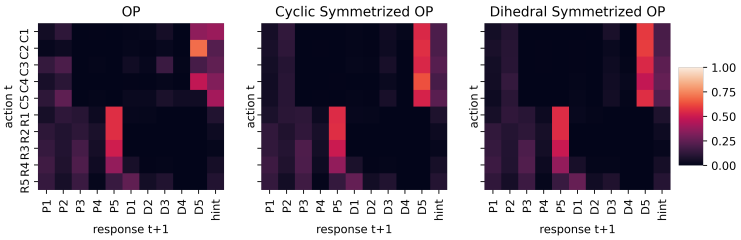

Figure 4: Conditional action matrices of OP, i.e. $P(a_t^i \mid a_{t-1}^j)$, unsymmetrized (left) and symmetrized at test time (middle is C_5-symmetrized and right is D_{10}-symmetrized). The y-axis represents the action taken at timesetep t and the x-axis shows the proportion of each action as response at timestep $t+1$. The matrices show the interactions between color/rank hinting and play/discarding. C1-5 and R1-5 mean hinting the 5 different colors and ranks respectively, and P1-5 and D1-5 mean playing and discarding the 1st-5th cards in the hand. We selected a random agent, and each plot is thereby computed by running 1000 episodes of self-play with the agent to compute the statistics.

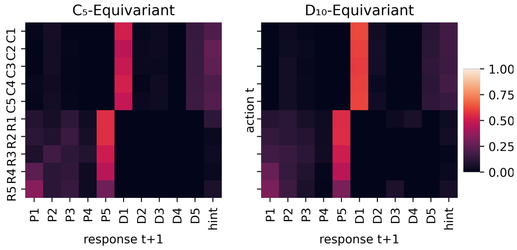

Figure 5: Conditional action matrices of G-equivariant agents, i.e. $P(a_t^i \mid a_{t-1}^j)$, C_5 on left and D_{10} on right. The y-axis represents the action taken at timesetep t and the x-axis shows the proportion of each action as response at timestep $t+1$. The matrices show the interactions between color/rank hinting and play/discarding. C1-5 and R1-5 mean hinting the 5 different colors and ranks respectively, and P1-5 and D1-5 mean playing and discarding the 1st-5th cards in the hand. We selected a random agent C_5-equivariant agent and a random D_{10}-equivariant agent, and each plot is thereby computed by running 1000 episodes of self-play with each agent to compute the statistics.

H On the High Variance of Cross-Play Scores

Reinforcement learning is notoriously sensitive to hyperparameter settings and seeds, making reproducibility of results challenging[5]. Correspondingly, in our experimentation we have found that average cross-play scores of groups of trained agents can vary greatly from group to group, where adjusting such hyperparameters as batch size, number of GPUs used for simulation, network architecture and seeds used can all lead to different average cross-play scores. This should be borne in mind for future research that aims to compare new results with existing baselines.

I Broader Impact

We have found that equivariant networks can effectively solve symmetry breaking and encourage play suited for cooperative settings. No technology is safe from being used for malicious purposes, which equally applies to our research. However, fully-cooperative settings target benevolent applications.

[5] https://media.neurips.cc/Conferences/NIPS2018/Slides/jpineau-NeurIPS-dec18-fb.pdf