A. Proofs of Theorems

We first prove two lemmas that will be necessary for the proofs of the main theorems.

Lemma 1. *For any policies* (π_0, π_1, π)*,*

$$\sum_{\tau_t \in T_t} P(\tau_t|\pi_0) \sum_{a_t} \pi(a_t|\tau_t^i) \left(R(s_t, a_t) + \sum_{s_{t+1}} \mathcal{T}(s_{t+1}|s_t, a_t) V^{\pi_1}(\tau_{t+1}) \right)$$
$$= \sum_{\tau_t^i \in T_t^i} P(\tau_t^i|\pi_0) \sum_{a_t} \pi(a_t|\tau_t^i) Q^{\pi_0 \to \pi_1}(a_t|\tau_t^i) \quad (7)$$

This Lemma shows an equivalence between an expected value integrated over all *trajectories* of length t, and expected value integrated over all *AOHs* of length t.

Proof.

$$\sum_{\tau_t \in T_t} P(\tau_t|\pi_0) \sum_{a_t} \pi(a_t|\tau_t^i) \left(R(s_t, a_t) + \sum_{s_{t+1}} \mathcal{T}(s_{t+1}|s_t, a_t) V^{\pi_1}(\tau_{t+1}) \right) \quad (8)$$
$$= \sum_{\tau_t^i \in T_t^i} P(\tau_t^i|\pi_0) \sum_{\tau_t \in \tau_t^i} P(\tau_t|\tau_t^i) \sum_{a_t} \pi(a_t|\tau_t^i) \left(R(s_t, a_t) + \sum_{s_{t+1}} \mathcal{T}(\tau_{t+1}|\tau_t, a_t) V^{\pi_1}(\tau_{t+1}) \right) \quad (9)$$
$$= \sum_{\tau_t^i \in T_t^i} P(\tau_t^i|\pi_0) \sum_{a_t} \pi(a_t|\tau_t^i) \sum_{\tau_t \in \tau_t^i} P(\tau_t|\tau_t^i) \left(R(s_t, a_t) + \sum_{s_{t+1}} \mathcal{T}(s_{t+1}|s_t, a_t) V^{\pi_1}(\tau_{t+1}) \right) \quad (10)$$
$$= \sum_{\tau_t^i \in T_t^i} P(\tau_t^i|\pi_0) \sum_{a_t} \pi(a_t|\tau_t^i) Q^{\pi_0 \to \pi_1}(a_t|\tau_t^i) \quad (11)$$

□

Lemma 2. *The softmax policy with temperature* T *is worse than the optimal policy by at most* T/e*. Formally, for any* $x_1 \ldots x_N \in \mathbb{R}^N$,

$$\frac{\sum_{i=1}^N \exp(x_i/T) x_i}{\sum_{j=1}^N \exp(x_j/T)} \geq \max_i x_i - T/e \quad (12)$$

Proof. Let $x_1 \geq x_2 \geq \ldots \geq x_N$ w.l.o.g. So $\max_i x_i = x_1$. Let $y_i = x_i - x_1$.

$$\frac{\sum_{i=1}^N \exp(x_i/T) x_i}{\sum_{j=1}^N \exp(x_j/T)} = \frac{\sum_{i=1}^N \exp(x_1/T) \exp(y_i/T)(y_i + x_1)}{\sum_{j=1}^N \exp(x_1/T) \exp(y_j/T)} \quad (13)$$
$$= x_1 + \frac{\sum_{i=1}^N \exp(y_i/T) y_i}{\sum_{j=1}^N \exp(y_j/T)} \quad (14)$$
$$\geq x_1 + \frac{\sum_{i=1}^N \exp(y_i/T) y_i}{N} \qquad \text{since } y \leq 0 \quad (15)$$
$$\geq x_1 + \min_{z \leq 0} \exp(z/T) z \quad (16)$$

To compute the minimum of $\exp(z/T)z$ in $(-\infty, 0]$,

$$f(z) = \exp(z/T)z \tag{17}$$
$$\frac{df}{dz} = \exp(z/T)(1 + z/T) = 0 \tag{18}$$
$$z = -T \tag{19}$$
$$f(z) = -\exp(-1)T = -T/e \tag{20}$$

The value of $f(z)$ at both endpoints is 0. Therefore, $\min_{z\leq 0} \exp(z/T)z = -T/e$, which when substituted into Equation 16 proves the theorem.

□

Theorem 1. *For any $T > 0$ and starting policy π_0, OBL computes a unique policy π_1.*

Proof. Since AOHs cannot repeat (a successor AOH is always longer than its predecessor) and the game has bounded length, the AOHs of acting players form a DAG with edges from each AOH to all possible successor AOHs for the next acting player. Ordering AOHs topologically s.t. successor AOHs precede their predecessors, we prove by induction.

In the base case, the first AOH in the topological ordering is a terminal AOH τ_ϵ, so $Q^{\pi_0\to\pi_1}(\cdot|\tau_\epsilon) = 0$.

For the inductive case, we must show that if $Q^{\pi_0\to\pi_1}(a|\tau^j_{t+1})$ is unique for every τ^j_{t+1} that is a successor of τ^i_t, then $Q^{\pi_0\to\pi_1}(a|\tau^i_t)$ is unique (where j is the player to act at the successor trajectory).

$\pi_1(a|\tau^i_t)$ is a function of $Q^{\pi_0\to\pi_1}(\cdot|\tau^i_t)$.

$$Q^{\pi_0\to\pi_1}(a|\tau^i_t) = \sum_{\tau_t,\tau_{t+1}} R(s_t, a) + \mathcal{B}_{\pi_0}(\tau_t|\tau^i_t)\mathcal{T}(\tau_{t+1}|\tau_t)V^{\pi_0\to\pi_1}(\tau^j_{t+1}) \tag{21}$$

The RHS of Eq. 21 only depends on π_1 at successor AOHs, which are uniquely defined. □

Theorem 2. *For every policy π_1 generated by OBL from π_0, $J(\pi_1) \geq J(\pi_0) - t_{max}T/e$, i.e. OBL is a policy improvement operator except for a term that vanishes as $T \to 0$.*

Proof. Let $J(\pi_0 \xrightarrow{t} \pi_1)$ be the expected return of playing π_0 for the first $t - 1$ timesteps and π_1 subsequently.

We prove by induction backwards in t that

$$J(\pi_0 \xrightarrow{t} \pi_1) \geq J(\pi_0) - T(t_{max} - t)/e \tag{22}$$

for all t, which implies the theorem for $t = 0$.

The base case of $t = t_{max}$ is trivially true because $(\pi_0 \xrightarrow{t_{max}} \pi_1) \equiv \pi_0$.

For the inductive case, suppose that Eq. (22) holds for all $t > t'$. Let T_t and T^i_t be the set of all trajectories and AOHs of length t, respectively.

$$J(\pi_0) - e(t_{max} - (t' + 1))T \leq J(\pi_0 \xrightarrow{t'+1} \pi_1) \tag{23}$$

$$J(\pi_0 \xrightarrow{t'+1} \pi_1) = \sum_{\tau_{t'+1} \in T_{t'+1}} P(\tau_{t'+1}|\pi_0) V^{\pi_1}(\tau_{t'+1}) \tag{24}$$

$$= \sum_{\tau_{t'} \in T_{t'}} P(\tau_{t'}|\pi_0) \sum_{a_{t'}} \pi_0(a_{t'}|\tau^i_{t'}) \left(R(s_{t'}, a_{t'}) + \sum_{s_{t'+1}} \mathcal{T}(s_{t'+1}|s_{t'}, a_{t'}) V^{\pi_1}(\tau_{t'+1}) \right) \tag{25}$$

$$= \sum_{\tau^i_{t'} \in T^i_{t'}} P(\tau^i_{t'}|\pi_0) \sum_{a_{t'}} \pi_0(a_{t'}|\tau^i_{t'}) Q'_{\pi_0}(a_{t'}|\tau^i_{t'}) \qquad \text{(Lemma 1)} \tag{26}$$

$$\leq \sum_{\tau^i_{t'} \in T^i_{t'}} P(\tau^i_{t'}|\pi_0) \max_{a_{t'}} Q'_{\pi_0}(a_{t'}|\tau^i_{t'}) \tag{27}$$

$$\leq \sum_{\tau^i_{t'} \in T^i_{t'}} P(\tau^i_{t'}|\pi_0) \sum_{a_{t'}} \pi_1(a_{t'}|\tau^i_{t'}) Q'_{\pi_0}(a_{t'}|\tau^i_{t'}) + T/e \qquad \text{(Lemma 2)} \tag{28}$$

$$= \sum_{\tau_{t'} \in T_{t'}} P(\tau_{t'}|\pi_0) \sum_{a_{t'}} \pi_1(a_{t'}|\tau^i_{t'}) \left(R(s_{t'}, a_{t'}) + \sum_{s_{t'+1}} \mathcal{T}(\tau_{t'+1}|\tau_{t'}, a_{t'}) V^{\pi_1}(\tau_{t'+1}) \right) + T/e \text{ (Lemma 1)} \tag{29}$$

$$= \sum_{\tau_{t'} \in T_{t'}} P(\tau_{t'}|\pi_0) V^{\pi_1}(\tau_{t'}) + T/e \tag{30}$$

$$= \mathbb{E}_{T_{t'}} \left[V^{\pi_1}(\tau_{t'}) | \pi_0 \right] + T/e \tag{31}$$

$$= J(\pi_0 \xrightarrow{t'} \pi_1) + T/e \tag{32}$$

Lemmas 1 and 2 can be found above. □

Notably, unlike standard MARL, the fixed points of the OBL learning rule *are not* guaranteed to be equilibria of the game.

Theorem 3. *If repeated application of the OBL policy improvement operator converges to a fixed point policy π, then π is an ϵ-subgame perfect equilibrium of the Dec-POMDP, where $\epsilon = t_{max}T/e$.*

Proof. We'll start with a proof sketch for a "temperature 0" policy $\pi_1(\tau^i) = \operatorname{argmax}_a Q^{\pi_0 \to \pi_1}(a|\tau^i)$ and then deal with the softmax policy.

Suppose π is a fixed point of the temperature-0 OBL operator. Then at every AOH τ^i,

$$\pi(\tau^i) = \operatorname*{argmax}_a Q^{\pi \to \pi}(a|\tau^i) \tag{33}$$

$$= \operatorname*{argmax}_a Q^{\pi}(a|\tau^i) \tag{34}$$

By the one-shot deviation principle (Hendon et al., 1996), π must be a subgame-perfect equilibrium of G.

Now, we will need to reiterate the proof of the one-shot deviation principle in order to modify it for softmax policies.

Suppose that π is a fixed point of the OBL operator at temperature T. We will show that for any π'_i,

$$V^{\pi}(\tau^i) \geq V^{\pi'_i, \pi_{-i}}(\tau^i) - T/e(|\tau^i| - t_{max}). \tag{35}$$

which for $\tau^i = \emptyset$ reduces to $J(\pi) \geq J(\pi'_i, \pi_{-i}) - t_{max}T/e$, proving the theorem.

We prove by induction over i's AOHs, treating successor AOHs before predecessors as before.

The base case at terminal AOHs holds trivially. Now suppose that (35) is true for all successors of τ^i. We know from Lemma 2 that

$$\sum_a \pi(\tau^i) Q^{\pi \to \pi}(a|\tau^i) \geq \max_a Q^{\pi \to \pi}(a|\tau^i) - T/e \tag{36}$$

$$V^{\pi}(\tau^i) = \sum_a \pi(\tau^i) Q^{\pi}(a|\tau^i) \geq \max_a Q^{\pi}(a|\tau^i) - T/e \tag{37}$$

Let $P(\tau^{i'}|\tau^i, a, \pi_{-i})$ be the probability that player i's next AOH is $\tau^{i'}$ after playing a at τ^i, and assuming other players play π_{-i}.[2] Then expanding out Q^π in (37) leads to

$$V^\pi(\tau^i) \geq \max_a \sum_{\tau^{i'}} P(\tau^{i'}|\tau^i, a, \pi_{-i}) V^\pi(\tau^{i'}) - T/e \tag{38}$$

$$\geq \max_a \sum_{\tau^{i'}} P(\tau^{i'}|\tau^i, a, \pi_{-i}) \left(V^{\pi'_i, \pi_{-i}}(\tau^{i'}) - T/e - T/e(|\tau^{i'}| - t_{max}) \right) \tag{39}$$

$$\geq \max_a \sum_{\tau^{i'}} P(\tau^{i'}|\tau^i, a, \pi_{-i}) \left(V^{\pi'_i, \pi_{-i}}(a|\tau^{i'}) - T/e(|\tau^i| - t_{max}) \right) \tag{40}$$

$$\geq V^{\pi'_i, \pi_{-i}}(\tau^i) - T/e(|\tau^i| - t_{max}) \tag{41}$$

Eq. (38) just expands out the expectation for Q^π over all possible next AOHs that player i may reach given the other players' policies π_{-i}.

□

Theorem 4. *Application of OBL to any constant policy $\pi_0(a|\tau^i) = f(a)$ - or in fact any policy that only conditions on public state - yields an optimal grounded policy in the limit as $T \to 0$.*

Proof. We need only show that any policy π_0 that conditions only on public state yields a state distribution at each AOH that matches the grounded beliefs $\mathcal{B}_G$; then the optimal grounded policy follows directly from the definition of OBL (Eq. 3)

The true state distribution induced by π_0 at τ^i are

$$P(\tau|\tau^i, \pi_0) = \frac{P(\tau) \prod_t P(o_t^i|\tau) \pi_0(a_t|\tau_t^{-i})}{\sum_{\tau'} P(\tau') \prod_t P(o_t^i|\tau') \pi_0(a_t|\tau_t'^{-i})} \tag{42}$$

If π_0 is constant, i.e. $\pi_0(a_t|\tau^i) = f(a_t)$, then π_0 in the numerator and denominator immediately cancel, yielding the grounded beliefs

$$P(\tau|\tau^i) = \frac{P(\tau) \prod_t P(o_t^i|\tau)}{\sum_{\tau'} P(\tau') \prod_t P(o_t^i|\tau')}. \tag{43}$$

If π only depends on the public state, then the numerator and denominator still cancel for all trajectories τ which contribute to the sum, since any τ that doesn't share a common public state with τ' will lead to different observations, by definition. □

B. Experimental Details for Hanabi

B.1. Reinforcement Learning

We use a highly scalable setup as illustrated in Figure 5 to efficiently train RL models in Hanabi using moderate computational resources (three GPUs). It includes four major components. The first one is a large number of parallel thread workers that handle the interactions between environments and the multiple players in each environment. The player needs to invoke one or more neural network inferences at each step to compute an action, which we run on GPUs. Here, we take an asynchronous approach that instead of waiting for the neural network calls to return, the thread worker immediately moves on to execute the next set of environment and players after sending the neural network request to the inference loop. This allows us to run multiple environments in a single thread worker. On top of this, we also run multiple thread workers in parallel. The combination of these two techniques allows us to run a massive amount of environments and therefore generates a large amount of neural network inference requests simultaneously. These requests are then batched together before sending to an everlasting inference loop. The batched inference is executed on GPUs and the inference loop may use multiple GPUs to distribute workloads. Each player collects observations, actions and rewards at each step and aggregates them into a trajectory at the end of an episode. The trajectory is padded to a fixed length of 80 time-steps and then stored into the

[2]Crucially, this transition probability is not dependent on **player i's** policy to reach τ^i, because AOH τ^i already specifies all of player i's actions to reach τ^i.

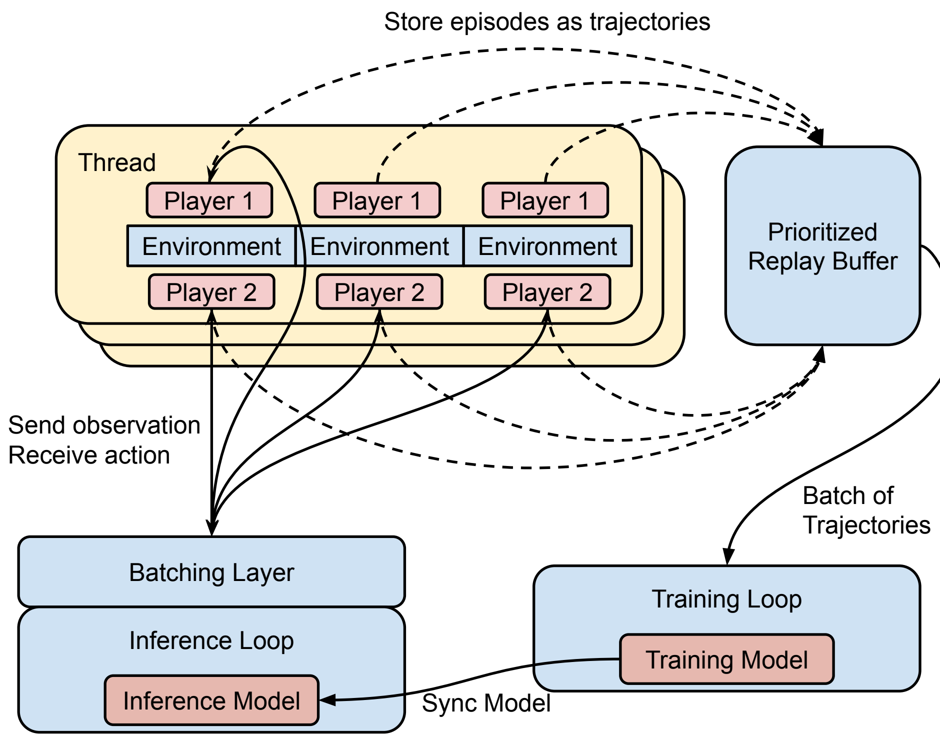

Figure 5. Illustration of RL training setup. Some arrows linking *player 1* and *batching layer* are omitted for legibility.

prioritized replay buffer. A training loop, which runs completely in parallel to all the procedures mentioned above, keeps sampling batches from the replay buffer and updates the model with TD-error. The training loop sends a new copy of the model to the inference loop to have the inference model synced every 10 gradient steps. We also follow best Q-learning practices such as dueling architecture (Wang et al., 2016), double DQN (van Hasselt et al., 2016) and prioritized experience replay (Schaul et al., 2016) with priority of the trajectory computed the same way as (Kapturowski et al., 2019). In all of our RL experiments, i.e. both training OBL agents and reproducing Other-Play agents, we run 80 thread workers and 80 environments in each thread worker on 40 CPU cores. We use 2 GPUs for the inference loop and 1 GPU for the training loop. For OBL, we use 1 or 2 additional GPUs for belief model inference. We list out important hyper-parameters in Table 2.

Notably, our implementation runs much faster in terms of both data generation due to a more efficient parallelization and thus can train on more data given a fixed, reasonable amount of time. After training for 40 hours, we are able to significantly outperform the state-of-the-art self-play performance in Hanabi using existing methods. As shown in Table 3, a simple VDN baseline is sufficient to achieve a remarkable 24.27 and 24.32 on average for 2-player and 3-player Hanabi, reducing the distance to perfect (25 points) by 26% and 47% respectively. We also find that with abundant data and fast data generation enabled by our infrastructure, the VDN baseline is strong enough that additional methods such as simplified action decoder (SAD) or auxiliary task (AUX) are no longer helpful for further improving self-play scores. However, we note that AUX still improves cross-play scores significantly.

B.2. Belief Learning

Instead of creating a training/validation dataset, we use a similar setting as the reinforcement learning training, where mini-batches are sampled from an ever-changing replay buffer populated by thread workers. There are four main differences. First, the inference loop uses a pretrained fixed policy instead of syncing with a training policy periodically. Second, we store the true hand of the player alongside the trajectories which will be used as training targets for our belief model. Third, a normal experience replay buffer without priority is used. Finally, the training loop trains an auto-regressive model that predicts cards in hand one-by-one from the oldest to the latest with supervised learning. The network architecture is the same as the one used in (Hu et al., 2021).

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`
`# optimization`	
`optimizer`	`Adam (Kingma & Ba, 2015)`
`lr`	`6.25e-05`
`eps`	`1.5e-05`
`grad_clip`	`5`
`batchsize`	`128`
`# Q learning`	
`n_step`	`1 (OBL), 3 (non-OBL)`
`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-Parameters for Reinforcement Learning

Agent	2 Players		3 Players	
	Mean	Max	Mean	Max
VDN (Hu & Foerster, 2020)	23.83 ± 0.03	23.96	23.71 ± 0.06	23.99
SAD (Hu & Foerster, 2020)	23.87 ± 0.03	24.01	23.69 ± 0.05	23.93
SAD+AUX (Hu & Foerster, 2020)	24.02 ± 0.01	24.08	23.56 ± 0.07	23.81
VDN (ours)	$\mathbf{24.27 \pm 0.01}$	**24.33**	$\mathbf{24.32 \pm 0.02}$	24.39
SAD (ours)	24.26 ± 0.01	**24.33**	24.24 ± 0.03	**24.42**
VDN+AUX (ours)	24.26 ± 0.01	24.32	24.08 ± 0.01	24.17

Table 3. Self-play of various methods rerun using our infrastructure, comparing with previous published best results. For each method, we run 10 independent training with different seeds and shown the mean and max of the final model evaluated on 10K games. VDN is value-decomposition network that construct the joint Q-value with the sum of Q-values of each player. SAD is the simplified action decoder that includes additional representations for the greedy action beside the actual action that chosen by the agent via ϵ-greedy or sampling from a softmax function. AUX means the addition of an auxiliary task that predict whether each card is playable/discardable/unknown.

B.3. Other Models

To test the performance of OBL when paired with unseen partners (*ad-hoc* teamplay), we train three types of policies with existing methods.

Rank Bot. The first is trained with the Other-Play (OP) (Hu et al., 2020) technique. The majority of policies trained with OP use a rank-based convention where they predominantly hint for ranks to indicate a playable card. For example, in a case where "Red 1" and "Red 2" have been played and the partner just draw a new "Red 3", the other agent will hint 3 and then partner will play that card deeming that 3 being a red card based on that convention. We refer to this category of agents as "Rank Bot".

Color Bot. Equivalently, one can also expect a color-based policy that will hint red for that latest card instead. In fact, such color-based policy was also appeared in the original Other-Play paper as the worst partner of the reset of the Other-Play policies. However, they are generally hard to reproduce as only 1 out of the 12 their training runs ended up with such policy. We use a simple reward shaping technique to produce similar policies reliably where we give each "hint color" move an extra reward during the first half of the training process and then disable the extra reward in the second half to wash out any artifacts. However, the reward shaping introduces undesired side effects that lead to arbitrary conventions. In practice,

	2 Players	3 Players
w/ Other-Play	19.93 ± 0.09	13.07 ± 0.15
w/o Other-Play	$\mathbf{21.01} \pm \mathbf{0.07}$	$\mathbf{17.46} \pm \mathbf{0.12}$

Table 4. Results of Clone Bot. Here Other-Player refer to the color shuffling technique first proposed in the Other-Player paper. We adapt the same technique here but use it for data-augmentation purpose. The trained models are evaluated on 5000 games by greedily selecting the action of the highest probability at test time. Errors shown are standard error of mean (s.e.m.)

we find that hiding the *last action* field of the input observation will make training outcomes more consistent. The reward shaping and feature engineering techniques are only used for producing this specific policy, which we refer to as "Color Bot".

Clone Bot. One of the goals of the Hanabi challenge (Bard et al., 2020) is to develop artificial agents that can collaborate with humans. To understand how well our models can collaborate with humans without resorting to costly experiments, we train a behavior clone bot mimicking human behaviors to serve as a proxy. We acquire 240,954 2-player games and 113,900 3-player games from the online game platform "Board Game Arena"[3] and convert it to a dataset for supervised learning. The model takes in the trajectory of an entire game from the perspective of a single player and predicts its action at each time-step. This is similar to the independent Q-learning setting in the multi-agent RL context. The network consists of one fully layer, followed by ReLU activation, a two-layer LSTM and an output fully connected layer with softmax. Dropout is applied before the output layer to reduce overfitting. The best models use 512 hidden units for each layer in the network and 0.5 dropout rate. We adapt the color shuffling technique of Other-Play (Hu et al., 2020) as a data augmentation tool. At each training step, we randomly shuffle the color space for both observation and action of each trajectory in the mini-batch before feeding them to the network. We find this technique significantly improves the performance of the supervised model, especially for 3-player Hanabi where we have much less data. The results are shown in Table 4. The agent is trained by minimizing the cross-entropy loss. At test time, the agent selects the action with the highest probability at each step. We pick the model that achieves the highest self-play score during training as our final "Clone Bot".

C. Qualitative Analysis of Learned Hanabi Agents

In this section we share more insights about the policies learned by different methods through a series of qualitative analysis.

As we can see from Figure 4 in the main paper, the OBL level 1 agent plays a significantly higher percentage of grounded cards, *i.e.* cards of which both color and rank are known. This supports our claim that OBL learns a grounded policy given a constant π_0. However, it is worth mentioning that OBL level 1 still plays a fair number of cards when it only knows their rank. At first this may seem contradictory to the grounded policy claim. However, as we analyze the games, we find that in many cases the best grounded policy is to play those cards even if the color is not known. For example, at an early stage of the game when only a few cards have been played, knowing a card is 1 is sufficient to play it. Moreover, the agent often knows that the card is of a "safe color". For example, when both red 2 and green 2 have been played, it will be safe to play a card if we know that it is either a red 3 or a green 3. Playing such a card will be classified as "only rank" in the figure. Furthermore, there are multiple lives in Hanabi and sometimes it is worth taking a risk if the odds are good. We observe that the OBL level 1 agent sometimes plays a card blindly without knowing any information about the card, as reflected in the figure by the slightly higher percentage of "none" category than other OBL levels and other agents except for SAD. This is because Hanabi is not designed to be a game that can be mastered by a grounded policy, and therefore the number of the remaining hint tokens are often low due to the aggressive hinting strategy used by the policy. In this case, OBL level 1 resorts to maximizing the utility of the life tokens by playing one or two cards blindly in later stages of the game when the chances that a newly drawn card could be useful is high and remaining life tokens are abundant. Once there is only one life token left, the agent becomes conservative and stops the gambling behavior.

The conventions in the high levels of OBL agents stem from the fact that OBL level 1 agents give hints about a card in certain orders based on the situations. In some cases the OBL level 1 agent first hints at the color of a card it wants the partner to play while in other cases it hints rank first. The order depends on the grounded probability of having a playable card in the partner's hand after giving the hint assuming only the public information. If the partner knowing the color of some cards leads to a higher chance that they will have a playable card in hand, then we should hint color first and vice

[3] https://en.boardgamearena.com/

versa. This information gets picked up by the belief model and then taken advantage of by the OBL in the next level to form conventions. The conventions gradually get reinforced and lead to more nuanced conventions in the higher level of OBL agents.

For reference, we can also see from the same Figure 4 that the Other-Play Rank Bot and Color Bot have strong preferences to use only rank and only color to exchange information respectively, making it difficult for them to coordinate with each other. The Clone Bot uses both color and rank depending on the situation, which is similar to the highest level OBL agent. The SAD agent, which is trained in the typical self-play setting, seems to play "blindly" a lot. However, the fact that such an agent achieves a high self-play score indicates that it must be using some form of secretive conventions that are not grounded at all, e.g. conventions such as "hinting red means play the second card", which completely disregard the grounded information revealed by the hint actions. Since these conventions assign arbitrary and unpredictable meanings to actions that vary between every independent training run, they fail under the zero-shot coordination setting and are similarly poor for human-AI coordination.

D. Off-Belief Learning on 3 Player Hanabi

Method	Self-Play	Cross-Play	w/ Clone Bot
Other-Play	23.98 ± 0.03	17.36 ± 0.19	12.18 ± 0.25
OBL (level 1)	20.52 ± 0.05	20.41 ± 0.01	13.10 ± 0.42
OBL (level 2)	22.71 ± 0.04	22.50 ± 0.01	14.09 ± 0.48
OBL (level 3)	23.19 ± 0.03	22.85 ± 0.01	13.96 ± 0.47
OBL (level 4)	23.38 ± 0.04	23.02 ± 0.01	13.88 ± 0.47

Table 5. Performance in 3 player Hanabi. We run 10 independent training with different seeds for each method. Self-Play indicates play between an agent and itself. Cross-Play indicates play between agents from different independently-trained runs of the same algorithm. For 3 player Hanabi, the Cross-Play is computed by pairing agents from 3 different training runs. When evaluating agents with clone bot, we average the results of 2 copies of the same agents with 1 clone bot and 1 agent with 2 copies of the same clone bots. Each combination of agents are evaluated on 5000 games.

To better demonstrate the generalizability of off-belief learning. we apply it to 3-player Hanabi. The results are shown in Table 5. The cross-Play between 3 players can either consist of two copies of one agent and one copy of another or three different agents. We find the latter to be more challenging and therefore the cross-Play in the table is computed that way, except for Clone Bot. The score with Clone Bot is the average of two Clone Bots with one other bot and one Clone Bot with two other bots. We run 10 independent training seeds for each method. Despite high Self-Play score, Other-Play agents suffer greatly in the cross-Play, even more than they do in 2 Player Hanabi. However, the gap between cross-Play and Self-Play scores for each OBL level are significantly smaller. Therefore, even the OBL level 1 agent that does not use any conventions is able to collaborate better in the zero-shot coordination setting. As we apply OBL for more levels, the cross-Play score grows on the same pace as Self-Play score. Notably, all OBL levels perform better than Other-Play when collaborating with Clone Bot, indicating OBL as a promising method for human-AI coordination.