A CoMeDi Algorithm Details

A.1 Mixed-play Buffer Collection

Mixed-play consists of two phases in each episode: mixed-state generation and self-play. The "input" policies are the policy for the convention we are currently training π_n and the partner policy used for cross-play optimization π^* (using the same notation from Eq 7). First, we choose a random timestep within the episode that represents the length of the first phase (Line 2). Until this timestep occurs, we randomly sample the action from self-play or cross-play for both players (Lines 5-9). We do not store any of these transitions in the training buffer. Instead, we use the state at the last timestep and pretend that this is the initial state of the environment. For the rest of the timesteps, we perform the second phase, by taking self-play actions and store that in the buffer (Lines 10-12). When optimizing, we treat this new buffer the same as we would treat self-play, but modified with a positive weight hyperparameter, β, representing the importance of mixed-play.

Algorithm 1: Generating Mixed Play Buffer

Input: policies π_n, π^*, MDP M
Output: Replay buffer from running mixed-play.
1 $s_0, R \leftarrow s, 0$ `// start state, reward`
2 $t \sim$ uniform$(1, T)$ `// length of mixed-state generation phase`
3 **for** $i \leftarrow [0, T)$ **do**
4   $o^1, o^2 \leftarrow o(s_i)$
5   **if** $i < t$ **then**
6     $\pi_1^m \leftarrow$ Randomly choose π_n or π^*
7     $\pi_2^m \leftarrow$ Randomly choose π_n or π^*
8     $a_i^1, a_i^2 \leftarrow \pi_1^m(o^1), \pi_2^m(o^2)$
9     $s_{i+1}, - \leftarrow$ Step in M with(a_i^1, a_i^2)
10   **else**
11     $a_i^1, a_i^2 \leftarrow \pi_n(o^1), \pi_n(o^2)$ `// self-play`
12     $s_{i+1}, r_i \leftarrow$ Step in M with(a_i^1, a_i^2) `// keep reward in phase2`
Return: ReplayBuffer$(s_{t:T}, a_{t:T}^1, a_{t:T}^2, r_{t:T})$

A.2 Full CoMeDi Algorithm

Simplified pseudocode for the CoMeDi algorithm is presented below. Note that conventions are generated in a sequential order, with π_1 being trained with standard self-play and each π_i being trained with the awareness of prior conventions, $D_{1:i-1}$. The arg max operation in line 4 is estimated empirically by simulating a fixed number of rounds of cross-play in the environment with each existing convention and selecting the convention with the highest cross-play as π^*.

Algorithm 2: Diverse Conventions with CoMeDi

Input: Number of policies to generate n
Output: Diverse set of conventions D
1 $D \leftarrow (\pi_1, \ldots, \pi_n)$, parameterized by $(\theta_1, \ldots, \theta_n)$
2 Train π_1 with standard self-play
3 **for** $i \in \{2, \ldots, n\}$ **do**
4   **while** policy π_i has not converged **do**
5     $\pi^* \leftarrow \arg\max_{\pi^* \in D_{1:i-1}} J(\pi_i, \pi^*)$
6     $\tau_{SP} \leftarrow \texttt{GetRollout}(\pi_i, \pi_i)$
7     $\tau_{XP} \leftarrow \texttt{GetRollout}(\pi_i, \pi^*)$
8     $\tau_{MP} \leftarrow \texttt{MixedPlayRollout}(\pi_i, \pi^*)$
9   Estimate $J(\pi_i, \pi_i), J(\pi_i, \pi^*), J_M(\pi_i, \pi^*)$ with $\tau_{SP}, \tau_{XP}, \tau_{MP}$
10     $\theta_i \leftarrow \theta_i + \nabla_{\theta_i}[-J(\pi_i, \pi_i) + \alpha J(\pi_i, \pi^*) - \beta J_M(\pi_i, \pi^*)]$
Return: D

A.3 Implementation Details

We base the implementation of our algorithm on the Multi-Agent PPO algorithm (MAPPO) [8]. MAPPO is an actor-critic method which, in standard self-play, trains a single actor network for the policy and a single critic network for the value function [7]. To adapt MAPPO to train a pool of n conventions using our proposed mixed-play algorithm, we train n actor networks, n self-play critic networks, and $n^2 - n$ cross-play critic networks, each representing a cross-play pairing between the n conventions. We also use the PantheonRL library [6] to design our environments and training algorithms since it is designed to handle dynamic training interactions like cross-play and mixed-play. We have also integrated CoMeDi with a new GPU-accelerated simulation framework, which enables the collection of large batches of cross-play and mixed-play buffers in parallel with the collection of self-play buffers (more details to be revealed after the GPU simulation framework is released from double-blind review).

Moreover, instead of training the whole batch of n diverse agents in parallel, in practice we sequentially grow the set of agents one at a time, keeping the previous agents fixed. We find that sequential generation leads to more stable training: since the previous agents are fixed, the diversity regularization term becomes a reward shaping term that is only a function of the policy of the current agent.

A.4 Practical Guidelines for Hyperparameter Tuning

There are some safe choices for hyperparameters that work well in general, which we used to tune the hyperparameters for our experiments. First, we observe that directly using the best MAPPO hyperparameters for the particular environment, like learning rate and the model architecture, transfers well to CoMeDi. To find the cross-play weight (α), fix the mixed-play weight to 0 and find the lowest value for the cross-play weight such that increasing it further does not significantly increase the self-play score or decrease the cross-play score. If the cross-play weight is too high, this may cause training instabilities since the updates to increase the self-play score would be directly counteracted by the updates to decrease the cross-play score. Finally, choose the value of the mixed-play parameter such that the average mixed-play score (for the second half) is slightly less than half of the self-play score, which would indicate that self-play is able to smoothly continue from any mixed-play state.

The guidelines for choosing hyperparameters works well in general, but domain knowledge of the environments also helps. If your specific environment also has some indicators of handshakes, you can also use those to determine if handshakes are still happening. Furthermore, environments where partners' actions are not visible, like Blind Bandits, do not require mixed-play at all because handshakes are impossible. In practice, we have seen that CoMeDi is relatively robust to hyperparameters and it gives reasonable policies with a cross-play weight of 0.5 and a mixed-play weight of 1, even if they are not perfectly optimal.

Choosing a population size is also an art, but due to the sequential nature of CoMeDi, prior conventions are unaffected by the generation of later conventions. The choice of algorithm for generating a "convention-aware agent" would likely influence the number of conventions to use for the diverse set.

A.5 Extending to Larger Teams

When using CoMeDi for cooperative games with more than 2 players, we can follow the same algorithm presented in 2, but we have to be a bit careful when collecting rollouts.

To collect the cross-play buffer between π_i and an existing π^* in a k-player game, we can randomly assign each player to one of the conventions but keep those assignments consistent throughout the duration of the episode. The same logic regarding the minimization of cross-play rewards still applies since semantically similar conventions would result in a high reward even when the team contains a mix of the conventions.

To collect the mixed-play buffer, we would randomly choose between the two conventions for each player at each timestep during the mixed-state generation phase. We would still treat self-play as normal by using the convention being trained as the convention for all players.

B Experiments

B.1 Choice of Baselines

Throughout this work, we compare the performance of CoMeDi against pure MAPPO, a modified version of ADAP, and a pure cross-play minimization baseline (CoMeDi with $\beta = 0$). We do not directly use ADAP because we find that its generative capabilities from parameter sharing often limits the diversity of its agents, resulting in very high cross-play between agents in the population. By adding its diversity loss to the MAPPO algorithm, we eliminate the confounding variables of the base PPO implementation and the parameter sharing in the original algorithm. We also do not directly compare against TrajeDi because a fundamental aspect of its algorithm is the concurrent generation of a common best response agent, which implicitly optimizes for high cross-play within the "diverse set" of policies. However, upon analyzing TrajeDi's diversity loss, we note that it is very similar in practice to ADAP's loss. Therefore, we believe that our modified version of ADAP is the fairest representative of statistical diversity approaches. Concurrent work that implements a technique similar to our pure cross-play minimization provides some other benchmarks with MAVEN and TrajeDi which show similar results to what we experienced with our modification of ADAP.

We do not compare our work to those in section 2.3, because they require more assumptions regarding the training domain. In particular, the strength of reward shaping methods (and other domain engineering techniques) is highly dependent on the manual design of the appropriate environment parameter space. However, CoMeDi can be interpreted as an *automatic* technique for reward shaping to form diverse conventions, which can potentially eliminate the need to engineer the environment parameter space for domain randomization.

B.2 Hyperparameters

Table 1: Common hyperparameters for agents in Blind Bandits and Balance Beam

hyperparameters	value
fc layer dim	512
num fc	2
activation	ReLU
network	mlp
ppo epochs	15
mini batch	1

Table 2: Hyperparameters in Blind Bandits

hyperparameters	value
buffer length	200
environment timesteps	10000
actor/critic lr	2×10^{-5}
linear lr decay	False
entropy coef	0.01
ADAP α	0.2
CoMeDi α	1.0
CoMeDi β	0.0

Table 3: Hyperparameters in Balance Beam

hyperparameters	value
buffer length	1250
environment timesteps	50000
actor/critic lr	2.5×10^{-5}
linear lr decay	True
entropy coef	0.01
ADAP α	0.05
CoMeDi α	0.3
CoMeDi β (ablation)	0.0, 0.25, 0.5, 1.0

Table 4: Hyperparameters in Overcooked (only training set)

hyperparameters	value
CNN Kernel Size	3×3
fc layer dim	64
num fc	2
activation	ReLU
rollout threads	50
buffer length (per thread)	200
environment timesteps	1000000
ppo epochs	10
actor/critic lr	1×10^{-2}
linear lr decay	True
entropy coef	0.0
ADAP α	0.025
CoMeDi α	0.5
CoMeDi β	0.0, 1.0

Table 5: Hyperparameters in Overcooked (convention-aware agents)

hyperparameters	value
CNN Kernel Size	3×3
fc layer dim	64
num fc	2
activation	ReLU
rollout threads	50
buffer length (per thread)	200
environment timesteps (SP phase)	200000
ppo epochs (SP phase)	100
lr	1×10^{-2}
entropy coef	1×10^{-3}

B.3 Compute Resources

We conducted our experiments with our lab's internal cluster. We only used the Intel Xeon Silver 4214R CPU for training in Blind Bandits and Balance Beam. For the Overcooked experiments, we used an additional NVIDIA TITAN RTX, which required around 3 hours per configuration. However, this time can be reduced significantly by disabling deterministic behavior in CUDA. Current work on optimizing performance with the GPU-accelerated simulator also shows that this time can be reduced.

B.4 Blind Bandits Environment Description

The Blind Bandits environment is a collaborative two-player Dec-POMDP where each player takes k steps, $k \geq 2$, and at each step they have to choose to go left (L) or right (R). Each player is given their own past history of actions in the episode, but cannot see the others' actions. There are two ways to get a positive score. To get a score of S, the first player's first step must be L, and the second player's last step must be L. To get a score of $G > S$, the first player's first step must be R, but all following steps by all players must be L except for the second player's last step, which must be R. If the players fail to coordinate, they get a score of 0.

The policies that converge to S are all compatible with one another since all of them have agent 1 start with L and have agent 2 end with L (shown in blue). If two agents are playing randomly, there is a 0.25 probability that they will get a reward of S. Meanwhile, only one trajectory will result in a score of G (shown in orange), so there is only a 2^{-2k} chance that random agents will get a reward of G. Notice how these two conventions (S and G) are entirely incompatible with one another because their first and last moves must be different, and are therefore different equilibria. Ideally, we want to find a representative policy for both conventions using a technique that finds a diverse set of conventions.

B.5 Blind Bandits Baseline Results

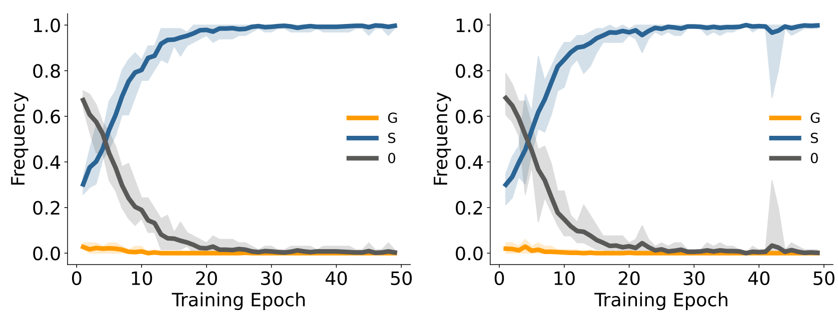

Figure 1: Frequency of self-play scores during the optimization of two conventions using statistical diversity (the ADAP loss) from 10 independent seeds. The blue line indicates an S convention, the orange line indicates an G convention, and the black line indicates a score of 0 (all other paths that lead to 0). With ADAP, neither convention converges to G. We choose $k = 3$ as the number of steps in the environment.

In Figure 1, we see that using ADAP to train a set of two agents almost always leads to them both converging to S. For this result, we used the ADAP diversity weight of 0.2, the most common value used in the original paper. Increasing this weight sometimes results in discovering a few G conventions, but this is inconsistent and results in more unstable training.

In fact, training an agent to converge to the G convention is more difficult that it appears. Off-the-shelf MARL algorithms typically converge to the S convention as well. At the first training epoch, the actor chooses random actions, so it will get a score of S 1/4 of the time while it gets a score of G with a probability of 2^{-2k}. The critic network (in both the decentralized and centralized setting) will learn that the value of choosing to go left in the first state is $S/2$ while the value of choosing to go to the right is $G/2^k$. Therefore, the policy updates will favor the S convention in the first epoch as long as $S > G/2^{k-1}$, and this separation gets larger until it fully converges to an S convention.

Most existing zero-shot coordination (ZSC) algorithms would also converge to the S convention. The TrajeDi algorithm finds a common best response to a set of conventions that have a wide diversity in trajectory, but the G convention has only one trajectory so it will never be chosen. The Off Belief Learning (OBL) algorithm would also only converge to the S convention because it acts on the belief that its partner is a random agent. Higher levels of OBL will be stuck in the S convention since they assume that their partner is from a lower level of OBL. Note that the purpose of ZSC is to have a high cross-play between independent runs of the same algorithm, so both of these techniques succeed in that manner, but this does not imply that these algorithms would have high cross-play with humans.

B.6 Balance Beam Environment Description

At the start of the game, the location of each player is randomly initialized. At each timestep, both players take simultaneous actions, and can move 1 or 2 locations to the left or right, but cannot stay still. If an action leads to them falling off the line, they get a score of -1 multiplied by the number of remaining timesteps. Otherwise, they get a score of $-d(s_1, s_2)/5$ where d is the distance between the two player's locations. Finally, if both players are on the same spot at the end of their turn, they get an extra point. Each episode lasts two timesteps, and the players see the result of the first step's actions before making their second move.

A perfect convention would always get a score of 2.0, because players can score +1 in each of the two timesteps. The worst score is -2.0, which occurs when one player moves off of the line at the first time step.

We also hand-code some conventions to see if CoMeDi discovers conventions that are similar to those that humans follow. There are two simple conventions: a left-biased convention and a right-biased convention. These dictate how ties are broken when multiple actions are equally optimal. For example, if the distance between two players is 1 step, like in Figure 6 of the main text, the player from a left-biased convention would want to move to the open space to the left since staying in one spot is not an option.

B.7 Balance Beam Baseline Results

We designed the Balance Beam environment to distill the issue of handshakes when minimizing cross-play, which is why the main text only emphasizes the impact of the β hyperparameter in CoMeDi. However, we also trained ADAP in this environment to see how it compares to the other approaches.

The first trained convention gets a score of 2.0, and gets an expected score of 0.768 and 1.112 when paired with the left and right-biased agents respectively.

The second trained convention gets a score of 1.808, and gets an expected score of 1.048 and 0.176 when paired with the left and right-biased agents respectively. Its cross play score with the first trained convention is 0.392.

When training the ADAP agents, we observed a very unstable training process resulting in very low self-play scores with typical values of the diversity weight parameter. For this reason, we had to choose a relatively low value for the loss parameter (0.05) in order to make a fair comparison with our technique.

B.8 Overcooked Agent Generation

For our experiments, we generated 2 baseline agents and 2 convention-aware agents using CoMeDi. The first baseline agent, which we refer to as "SP", was trained with pure self-play, and we tuned the hyperparameters to maximize its self-play score. For all other generated agents, we maintained the same hyperparameters that are inherent to MAPPO, but we would tune the diversity weights specific to each algorithm. Our second baseline, which we refer to as "ADAP", is a convention-aware agent to a population of 8 agents trained with ADAP with no parameter sharing and a diversity weight of 0.025. The XP agent was also a convention-aware agent to a population of 8 agents trained with $\alpha = 0.5$ and $\beta = 0.0$. The CoMeDi agent was the same as XP, but its population was trained with $\alpha = 0.5$ and $\beta = 1.0$.

For each layout of Overcooked, we can determine the expected number of dishes to be served by each agent in self-play. In the Cramped Room, SP averages 4.36 dishes, ADAP averages 2.75 dishes, XP averages 4.68 dishes, and CoMeDi averages 5.52 dishes. Meanwhile, in the Coordination Ring, SP averages 3.47 dishes, ADAP averages 1.90 dishes, XP averages 3.06 dishes, and CoMeDi averages 5.36 dishes.

We can also calculate the expected reward for the best and worst performing agents in the training sets of ADAP, XP, and MP. In Cramped Room, ADAP's average rewards spanned 0 to 5.98, XP's rewards spanned 4.12 to 5.96, while MP's rewards spanned 5.0 to 5.88. In Coordination Ring, ADAP's average rewards spanned 0 to 4.965, XP's rewards spanned 2.75 to 4.56, while MP's rewards spanned 4.92 to 5.99.

When training agents with ADAP, we would frequently see a few policies with very high expected returns with the remaining policies having low scores. We attempted to tune ADAP's diversity weight to enable more balanced generation, but this issue continued to persist even with a final diversity weight significantly lower than the values presented in the original paper.

B.9 Overcooked User Study Setup

The human-AI interaction portion of this research was approved by our IRB.

Our total population consisted of 25 paid participants with varying prior experiences in Overcooked, recruited through Prolific. We did not impose any conditions on participation through Prolific except for requiring "proficiency in English language." We paid $10.33 (US dollars) per participant for approximately 40 minutes of time, equivalent to $15.50 per hour. We titled the study "Playing with AI Agents in the Overcooked Video Game" with the following description:

"In this study, participants will play with 4 different AI agents in 2 settings of the Overcooked game. Our goal is to understand how well trained agents can work with humans in tasks that require coordination. Your task is to try to work with the AI to cook many "dishes of onion soup" within a 40-second time limit. You will also fill out short surveys to judge the quality of the AI agents. You will play 20 games in total.

To play the game, you need to use a keyboard with arrow keys and a space bar. Desktops, laptops, and tablets with keyboards may be used. The AI agents runs on your browser and are designed to be lightweight so they should be compatible with most hardware.

The games will be fast-paced, and we may reject submissions that are consistently unable to score points in the game. As long as we can see that you are trying to score points, your submission will be accepted.

NOTE: Please only accept this study if you have at least 8 GB RAM. The game will take up to 2 GB of RAM since we are loading models onto your computer so you have a smoother experience. If the game freezes, please let us know and refresh the page. Please use Firefox, Chrome, or Safari."

To ensure that participants across the world are able to complete the study without excessive latency, we run all models on the users' devices directly through tensorflow-js.

Each user played 2 games in a layout independently before playing each AI agent for two 40-second rounds in a random order. The users first played with all agents in the Cramped Room environment before playing with all agents in the Coordination Ring environment.

We also asked the users to fill out a short survey with qualitative questions about each partner after playing both rounds in any configuration using the 7-point Likert scale.

We presented the following 8 statements (in order):

1. The AI followed my lead when making decisions.
2. The AI agent frequently blocked my progress.
3. The AI was consistent in its actions.
4. The AI always made reasonable actions throughout the game.
5. I would like to collaborate with this AI in future Overcooked tasks.
6. The AI's actions were human-like.
7. I trusted the AI agent in making good decisions.
8. The AI agent was better than me at this game.

Note that a higher Likert score is better for all of the prompts except for the second question.

We also presented an optional free-response section with "Other comments or observations?" that users could fill out at the end of each survey.

To determine statistical significance, we use the paired Student t-test between the scores across different AI agents for each layout.

We also asked a smaller set of users (8 users) to play with an expert human player in-person after playing with all AI agents to determine what human-level performance would entail without allowing explicit communication. This expert human player was trained to adapt to different play styles, and users reported that this human player was "very good" at the game, noting fast reflexes. This set of users was mutually exclusive from the Prolific set, since the games needed to be played in person. We determine if there is a statistically significant difference between the AI agents and the human expert through the unpaired t-test.

B.10 Cramped Room Overcooked Results

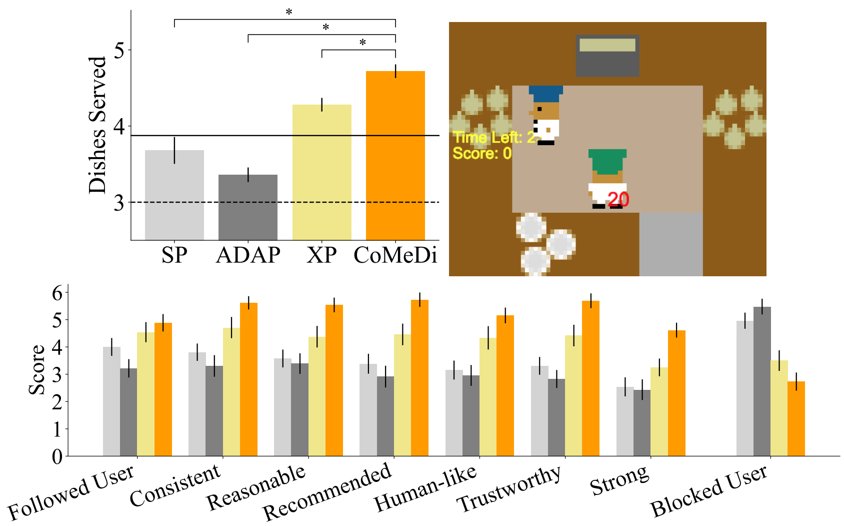

Figure 2: Results for the Cramped Room: average scores (top left), environment visualization (top right), and user survey feedback using the 7-point Likert scale(bottom). Higher average scores are better. The dashed horizontal line indicates the average score when a player is working alone while the solid horizontal line indicates the average score when paired with an expert human. Higher is better for all but the last feedback score. Error bars represent the standard error.

In the Cramped Room Environment, players have a very small area to move around in, and they need to coordinate around the management of ingredients and the usage of plates. Given the simplicity of the setting (relative to the Coordination ring), humans might be able to adapt quickly to AI players since the patterns of movement can be very clear after interacting for a short time. In particular, we note that this layout has lower convention dependence than the Coordination Ring.

The results of the user study in the Cramped Room environment are in Figure 2. In terms of scores, we see that CoMeDi (score of 4.72) performs the best, followed by XP (4.28), SP (3.68), and ADAP (3.36). In terms of statistical significance, CoMeDi outperforms all AI agents ($p < 10^{-3}$) and the expert human ($p < 10^{-4}$). XP also outperforms ADAP, SP, and the expert human ($p < 0.01$ for each).

B.11 User Free Response Feedback

User feedback for SP in the Cramped Room environment:

- This AI trusted me more than the others. He waited for me to get the plate and deliver the soup. The other bots rushed to do it themselves.
- I could not understand what the AI was doing
- The AI was slow in decision making
- Ai was a bit hesitant sometimes

User feedback for ADAP in the Cramped Room environment:

- The AI sometimes blocked me. He held the onion in his hands, and it was already cooked, and I had the plate in my hands, but I couldn't get to pick up the soup because he was blocking it. I think he was trying to place the onion even tho there was no space to place it in.
- AI was blocking me whole time
- i didnt like this one...
- It was somehow worse than S, it blocked the plate for like 7 seconds
- the ai seemed very confused
- He tried to do things right, but sometimes he seemed confused.

User feedback for XP in the Cramped Room environment:

- This AI was more lazy. He only held the plate in his hand and waited for me to cook the soup. The other AI wasn't that way (AI D). He cooked and adapted in getting the plate and everything. This bot did the exact opposite.
- I did not like the fact that it could not path find a different way to deliver the full plate.
- The AI seems to be faster
- ai was very good at this game
- Once he picked up a plate after I got one plate, then he never stopped that action until I put the next 3 onions in the pot.

User feedback for CoMeDi in the Cramped Room environment:

- This bot was by far the best. He was very consistent in everything he did, his moves were all correct and I felt very good, working with him.
- this one was not so bad
- This AI could work by himself, literally, and get the same 100 score, I believe.

User feedback for SP in the Coordination Ring environment:

- The AI did the correct thing, but when it came to pressing the space bar, he struggled. Even tho he just needed to press the space bar, and he stood right in front of it, and it was just one space bar away, he decided to wait in front of it for some seconds and then press it, which concluded us to make less soup overall.
- the first soup was ok and then AI like "freeze"...it was weird.
- ok so this one completly stood in my way while i was trying to get the plate he just stood there with an onion on his hand. Aside from the ai i really likes this study and experiment and hope one day ill be able to participate in one of them again!
- At first it looked like the cooperation was working until when it just stopped doing anytihng and occasionally just blocked the way to the stoves. Towards the end it just standed in the corner doing nothing.
- Yes it was consistenly bad
- He didn't know what to do when picking a plate and there were onions missing in a pot.

User feedback for ADAP in the Coordination Ring environment:

- This bot blocked me a lot of times. He also just stood around the onions and did nothing, which stopped us both. He sometimes got a plate in his hand and just went up and down all the time until he decided to play normally again. Would not play with this bot.
- i also didnt like this ai.
- This one was very bad at making decisions and pathfinding
- the AI slowed the process

- AI was better than the other ones so far, but still got alot in the way which is frustrating to play with.
- He was in the way a lot and almost never knew what to do next, or where to go.

User feedback for XP in the Coordination Ring environment:

- He did very good in the first round, we always went clockwise. It was perfect. In the second round, he decided to be the lazy chef. He was too lazy to pick up an onion or the plate.
- It's moves felt very scattered and didn't make sense to me.
- The AI kept getting in the way and got nothing done.
- The AI was helpful
- This one was doing great decisions, but sometimes he seemed confused. In general I liked it.

User feedback for CoMeDi in the Coordination Ring environment:

- This bot is GODLIKE! He did everything correct, he adapted to my moves like he can see into the future. Just great playing with him. The most efficient by far.
- it seems sometimes you get into a flow, which the AI breaks after a while
- He knew exactly what to do next.
- The experiment was very cool because every AI was unique. The best AI was AI M by far, my most favorite. If I was a chef, I would definetly hire him!

C Related Work

Concurrent to our work, methods for training diverse agents with respect to reward have been proposed.

In LIPO [3], they also consider cross-play as a diversity metric, but do not address the critical issue of handshakes that arise when minimizing cross-play. Their results reinforce our observations when the mixed-play weight, β, is set to 0. However, as we have noticed in our Balance Beam simulation results and the Overcooked user study, mixed-play has a large impact on creating good-faith conventions.

The ADVERSITY [4] work considers a zero-shot coordination framework in the game of Hanabi. They propose a belief reinterpretation model to address a similar "sabotaging" behavior that we experienced. This model is designed to tackle the issue of handshakes by finding a plausible distribution of self-play states that would result in the same observation received from cross-play and use this while training so agents cannot discriminate between self-play and cross-play observations. However, it is unclear whether belief reinterpretation would help in the games we examine in this paper since cross-play can potentially encounter observations that are completely impossible under self-play. Specifically, in Overcooked, all agents have access to the state from the prior frame, but conventions exist as a way to manage the workload of tasks between partners. In Hanabi, a core part of the game is predicting the underlying state of the game given the observation. As such, conventions are based around communicating information about the state implicitly through actions.

ADVERSITY uses the fact that multiple trajectories can generate the same action-observation history, which enables it to gain very strong results in Hanabi. However, this technique would fail in our settings because there are observations that are only possible under cross-play and not self-play, so belief reinterpretation would not be helpful. Therefore, although ADVERSITY and CoMeDi both attempt to address the problem of handshakes, their core assumptions are entirely different. Since CoMeDi does not perform explicit belief reinterpretation, it will not be competitive with ADVERSITY on Hanabi, but it would still be able to train a sequence of agents.

In games where implicit communication to predict the underlying state is the core task, ADVERSITY is a strong choice for training a diverse set of agents. However, CoMeDi would be more effective in tasks where a team needs to divide a workload or commit to a particular strategy for effective coordination. We believe that these scenarios are more similar to the tasks that one would encounter in a robotics domain or typical video game setting.

D Limitations

Although our technique of creating a convention-aware agent using CoMeDi was able to surpass human-level performance in Overcooked, this technique has some drawbacks. Since the policy has no memory, training a convention-aware agent on a diverse set may lead to the AI breaking conventions established with humans earlier in the game, which is why one user reported that the CoMeDi agent sometimes breaks the established flow in Coordination Ring. Also, the BC-based algorithm for generating the convention-aware agent is often unable to account for human suboptimalities or transitions between conventions. For instance, some agents in Cramped Room would pick up an onion and expect the human to pick up a plate. If the human does not comply, it simply stands around instead of dropping off the onion and picking up a plate itself, because this type of action is never experienced under self-play for any convention.

On a theoretical level, CoMeDi does not provide any guarantees regarding the quality of agents. This is not unique to our algorithm, as statistical diversity techniques and reward shaping often changes the environment to the extent that "equilibrium conventions" are no longer guaranteed. Also, our solution to handshakes, mixed-play, implicitly assumes that re-establishing handshakes will come at some expense to the self-play scores. If this assumption is violated, as is the case with cheap-talk signals that aren't necessary for coordination, handshakes can be re-established at every timestep, effectively bypassing the mixed-play optimization. The issue of cheap-talk is a very tricky case in general when attempting to define diversity or robustness, because signals have no implicit meaning. In these cases, environment designers can remove extraneous cheap-talk signals or add nonuniform cost to communication, which has been effective in the realm of zero-shot communication [1].

E Broader Impact

We believe that CoMeDi can have a positive impact on game design and human-AI interaction in general. Being able to generate diverse conventions can allow game designers to understand the different strategies that players might try to use before extensive play-testing. Effective zero-shot coordination techniques would also help reduce the risk of misaligned conventions. We observe that, with proper tuning of the mixed-play weight, the convention-aware agent trained with CoMeDi learns to follow the lead of the human player, as indicated by the "followed user" section of the user survey. This is important for safety-critical applications like human-robot interaction tasks, because an overly assertive robot could unintentionally harm a human.

As a tool, it can directly be used for harmful ends, such as making it easier to cheat in multi-player games or generally conduct harm on others. Another potential effect of developing effective artificial zero-shot collaborators is that it could lead to more social withdrawal. In particular, if people who play video games start to strongly prefer playing with super-human AI collaborators over other humans, we may see people play less games with other people, which could counteract the prosocial benefits of cooperative gaming [5]. We therefore urge potential game designers and publishers who want to use CoMeDi to generate AI partners to evaluate the impact that artificial agents would have on their community.

F Creative Assets

Custom assets for this paper were digitally created by the authors without the assistance of AI image generation models. Stylistic inspiration was drawn from the Overcooked figures in [2] under the MIT license.

References

[1] K. Bullard, F. Meier, D. Kiela, J. Pineau, and J. N. Foerster. Exploring zero-shot emergent communication in embodied multi-agent populations. *CoRR*, abs/2010.15896, 2020. URL https://arxiv.org/abs/2010.15896.

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

[3] R. Charakorn, P. Manoonpong, and N. Dilokthanakul. Generating diverse cooperative agents by learning incompatible policies. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=UkU05GOH7_6.

[4] B. Cui, A. Lupu, S. Sokota, H. Hu, D. J. Wu, and J. N. Foerster. Adversarial diversity in hanabi. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=uLE3WF3-H_5.

[5] T. Greitemeyer and D. O. Mügge. Video games do affect social outcomes: A meta-analytic review of the effects of violent and prosocial video game play. *Personality and Social Psychology Bulletin*, 40(5):578–589, 2014. doi: 10.1177/0146167213520459. URL https://doi.org/10.1177/0146167213520459. PMID: 24458215.

[6] B. Sarkar, A. Talati, A. Shih, and S. Dorsa. Pantheonrl: A marl library for dynamic training interactions. In *Proceedings of the 36th AAAI Conference on Artificial Intelligence (Demo Track)*, 2022.

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

[8] C. Yu, A. Velu, E. Vinitsky, Y. Wang, A. Bayen, and Y. Wu. The surprising effectiveness of ppo in cooperative, multi-agent games. *arXiv preprint arXiv:2103.01955*, 2021.