Concept-based Understanding of Emergent Multi-Agent Behavior

Niko A. Grupen* Cornell University
Natasha Jaques Google Research
Been Kim Google Research
Shayegan Omidshafiei Google Research

Abstract

This work studies concept-based interpretability in the context of multi-agent learning. Unlike supervised learning, where there have been efforts to understand a model's decisions, multi-agent interpretability remains under-investigated. This is in part due to the increased complexity of the multi-agent setting—interpreting the decisions of multiple agents over time is combinatorially more complex than understanding individual, static decisions—but is also a reflection of the limited availability of tools for understanding multi-agent behavior. Interactions between agents, and coordination generally, remain difficult to gauge in MARL. In this work, we propose Concept Bottleneck Policies (CBPs) as a method for learning intrinsically interpretable, concept-based policies with MARL. We demonstrate that, by conditioning each agent's action on a set of human-understandable concepts, our method enables post-hoc behavioral analysis via concept intervention that is infeasible with standard policy architectures. Experiments show that concept interventions over CBPs reliably detect when agents have learned to coordinate with each other in environments that do not demand coordination, and detect those environments in which coordination is required. Moreover, we find evidence that CBPs can detect coordination failures (such as lazy agents) and expose the low-level inter-agent information that underpins emergent coordination. Finally, we demonstrate that our approach matches the performance of standard, non-concept-based policies; thereby achieving interpretability without sacrificing performance.

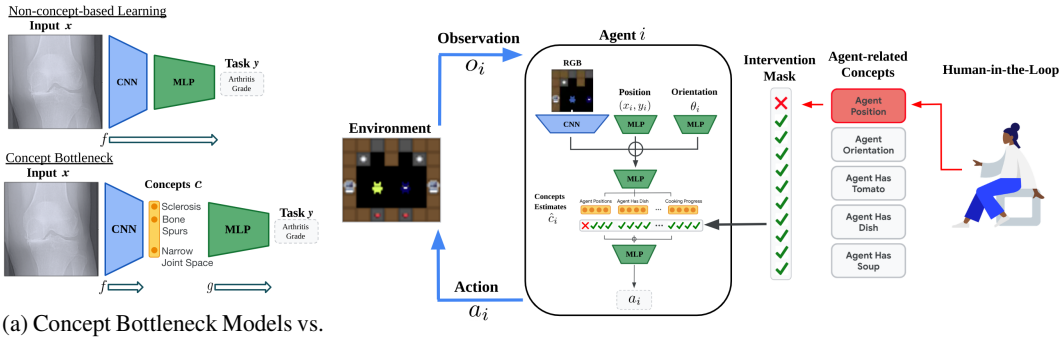
1 Introduction

Multi-agent learning techniques continue to play a crucial role in the development of scalable and generally-capable AI systems. In addition to well-known successes in board and card games (e.g., Go [1], Chess and Shogi [2], Poker [3], Stratego [4], Hanabi [5]) and massively-multiplayer online games (e.g., StarCraft [6], Dota2 [7]), multi-agent learning has enabled agents to develop a range of coordination capabilities, such as navigating game-theoretic social dilemmas [8], balancing low-level control with high-level strategy in football [9], allocating roles and conventions [10]; and even developing socioeconomic behaviors such as bartering [11] and tax policy design [12].

For all its success, multi-agent learning still lacks in a critical area: interpretability. Understanding emergent multi-agent behavior is challenging—for one, it is difficult to decipher the nature of a learned coordination strategy (or whether agents have learned to coordinate at all)—and these issues are only compounded by the inherent opacity of neural networks, which state of the art multi-agent methods rely upon heavily. For this reason, recent work has shifted focus from traditional measures of performance (reward, evaluations against human experts, etc) to better understanding emergent behaviors [13]. Related methods perform behavioral analysis in a *post-hoc* manner, either by visualizing trajectories [14], quantifying basic statistics related to agent behavior [9], or measuring the predictability of game-related concepts from a network's activations [15]. In the supervised learning

*Work done while an intern at Google Research. Correspondence to: `niko@cs.cornell.edu`

36th Conference on Neural Information Processing Systems (NeurIPS 2022).



(a) Concept Bottleneck Models vs. Regular Models.

(b) Concept Bottleneck Policies.

Figure 1: Concept bottleneck models force neural network classifiers to make decisions through an intermediate set of human-understandable concepts. Concept Bottleneck Policies increase the interpretability of emergent multi-agent behavior in a similar manner—distilling agent strategies into concepts—yielding interpretable, concept-based policies for MARL. CBPs also enable post-hoc behavioral analysis via concept intervention, which reveals the inter-agent factors driving coordination.

literature, an alternative to post-hoc analysis is *intrinsic interpretability*, in which a model's decisions incorporate human-understandable concepts directly, rather than extracting them from models after training. Intrinsically interpretable models can be preferable when concept-based explanations cannot be learned automatically, or if extracted concepts are unreliable [16]. Of particular relevance is the work of Koh et al. [17], which showed that it is possible to train an end-to-end neural network that is "bottlenecked" by human-understandable concepts, enabling intrinsic, concept-based interpretability while maintaining high performance in image classification tasks.

In this work, we bridge the multi-agent learning and concept-based interpretability domains by introducing Concept Bottleneck Policies (CBPs)—a class of intrinsically-interpretable, concept-based models for MARL. CBPs force agents to make decisions by first predicting an intermediate set of human-understandable concepts, then using those concepts to select actions. CBPs increase interpretability while also targeting important facets of the multi-agent learning problem. For example, if a subset of the concepts available to each agent correspond to other agents in the environment, learning through a concept bottleneck incentivizes agents to model each other, which has been shown in prior work to benefit performance [18, 19]. Moreover, by intervening on concepts related to other agents, we can obtain in-depth analysis of emergent multi-agent behaviors, and specifically learned coordination, that is much more difficult to attain from reward or qualitative analysis alone.

In our experiments, we show that concept intervention reveals key information underlying learned multi-agent coordination. Specifically, concept interventions can distinguish agents who have learned to coordinate from those that act independently, expose the distinct inter-agent factors that drive coordination between agents (when it emerges), and even diagnose common multi-agent failure modes such as lazy agents. Finally, beyond behavioral analysis, we demonstrate experimentally that Concept Bottleneck Policies perform comparably to non-concept based policies on difficult coordination tasks; meaning that CBPs achieve concept-based interpretability without sacrifing performance.

In sum, our contributions are as follows: (i) We introduce Concept Bottleneck Policies as an interpretable, concept-based learning architecture for MARL; (ii) We highlight the role CBPs can play in better understanding multi-agent behavior (and specifically coordination); (iii) We show that CBPs can learn effective policies that match the performance of non-concept-based network architectures.

2 Related Work

Our work lies at the intersection of interpretability and MARL. In interpretability, computer vision works have used saliency maps to help provide local explanations (at the pixel level) for model predictions [20–23]. Interpretability using grounded concepts has been previously explored to provide more meaningful, human-understandable explanations of model decisions [24–30]. As described earlier, concept bottleneck models [17] constrain the networks through such grounded concepts to enable *direct* interpretability of the model itself.

RL interpretability techniques have included those that either directly increase agents' model transparency [31, 32], where we take transparency loosely to mean the ability of a human observer to understand the underlying decision-making process of the agent; or analyze agent behaviors from a post-hoc perspective [13, 15, 33–38]. Approaches that directly increase transparency include those that use model compression in combination with decision trees to yield more interpretable agent policies [31], or rely upon causal decision trees with limited depth to explore causality of agent decisions [32], However, these approaches have not yet been extended to MARL settings involving complex observations (e.g., images), where function approximators are typically needed to solve underlying tasks. Post-hoc RL interpretability methods have used computer vision-based saliency techniques to better attribute decisions to image observation regions [33], highlighting states that lead to major differences in agent behaviors [34], and summarize key agent behaviors using measures such as action uncertainty [35]. Increasingly, techniques making use of natural language have been used to enable human understanding of agents through instructions [39] or policy explanations [40].

Interpretability of MARL agents has received limited attention in prior works, with primary investigations gauging agents' internal representations by making predictions about future outcomes [36], visualizing latent clusterings of agents' neural activation vectors [37, 41], visualizing emergent role assignments [42], and most recently using offline behavioral analysis to learn behavioral spaces over agents [13]. Recent works have also conducted analysis of decision-making in two-player games such as Chess [15], and Hex [38]. In contrast to our work, these approaches focus on post-hoc interpretability, whereas ours makes agent decisions directly transparent, thus enabling causal analysis of their decisions (as illustrated in our experiments).

3 Background

Partially-Observable Markov Games A partially-observable Markov game [43] of N agents is defined by the tuple $\mathcal{M} = (\mathcal{S}, \boldsymbol{\mathcal{A}}, \mathcal{T}, \boldsymbol{\mathcal{R}}, \boldsymbol{\Omega}, \mathcal{O})$ where $\mathcal{S}$, $\boldsymbol{\Omega}$, and $\boldsymbol{\mathcal{A}}$ define the game's global state-space, joint observation-space, and joint action-space, respectively. In each state, each agent i selects an action $a_i \in \mathcal{A}_i$, yielding a joint action $\boldsymbol{a} = (a_1, ..., a_N)$ for all agents. Following action selection, the environment transitions to a new state according to the transition function $\mathcal{T} : \mathcal{S} \times \boldsymbol{\mathcal{A}} \rightarrow \mathcal{S}$, produces new observations for each agent according to the observation function $\mathcal{O} : \mathcal{S} \times \boldsymbol{\mathcal{A}} \rightarrow \boldsymbol{\Omega}$, and emits a reward to each agent defined by the reward function $\boldsymbol{\mathcal{R}} : \mathcal{S} \times \boldsymbol{\mathcal{A}} \rightarrow \mathbb{R}^N$. The action selection of each agent is dictated by a policy $\pi_i : \Omega_i \rightarrow \mathcal{A}_i$ and the collection of all individual policies $\boldsymbol{\pi} = (\pi_1, ..., \pi_N)$ is referred to as the joint policy.

Concept Bottleneck Models Concept bottleneck models refer to a class of intrinsically-interpretable neural network architectures for supervised learning [17]. The network makes predictions by first estimating a set of human-understandable concepts, then producing an output based on those concept estimates–i.e. the network is "bottlenecked" by concepts. Formally, given a dataset $\{(x^{(j)}, y^{(j)}, c^{(j)})\}_{j=1}^{n}$ consisting of inputs $x \in \mathbb{R}^d$, outputs $y \in \mathbb{R}$, and human-understandable concepts $c \in \mathbb{R}^k$ (where k is the number of unique concepts), a concept bottleneck model learns two mappings – $f : \mathbb{R}^d \rightarrow \mathbb{R}^k$ from input-space to concept-space, and $g : \mathbb{R}^k \rightarrow \mathbb{R}$ from concept-space to output-space. The model can then make predictions $\hat{y} = f(g(x))$ as a composition of those mappings.

Figure 1a highlights the increased interpretability achieved by concept bottlenecks in the context of the arthritis classification task posed by Koh et al. [17]. Unlike standard networks that learn an uninterpretable mapping from MRI images to arthritis severity labels, the concept bottleneck's output is conditioned on an intermediate set of human-understandable concepts. Since these concepts correspond to real-world features—e.g., the presence of bone spursin the arthritis task—they can be used by human observers to understand model failures or misclassifications. Moreover, because the model's predictions are conditioning on these concepts, we can manually change the model's output (without retraining or fine-tuning) by intervening on its concept estimates.

4 Concept Bottleneck Policies for MARL

In this section, we introduce Concept Bottleneck Policies as an intrinsically interpretable, concept-based policy learning method for MARL. First, we bridge concept-based interpretability and MARL by extending the Markov game formalism to include the components necessary for concept-based

learning. Then, we introduce the concept bottleneck architecture and show how it can be learned within any existing policy learning scheme. Finally, we demonstrate our method's potential as a tool for interpreting emergent multi-agent behavior. In particular, we introduce a behavioral analysis technique using concept intervention and show that, using this technique, we can both detect successful coordination and expose coordination failures (e.g., lazy agents).

4.1 Concept-based Markov Games

To support concept learning, we extend the definition of Markov games from Section 3 in two important ways. First, we assume that, in addition to its state space $\mathcal{S}$, the environment maintains an interpretable concept-state space $\mathcal{C}$, where each concept-state $\boldsymbol{c}\in\mathcal{C}$ is a vector of human-understandable concepts that are extracted from the environment and describe key features of the environment, such as the position of agents or objects (we note that this assumption holds in popular RL settings like Atari [44] and the multi-agent games we study here). Second, though agents never observe all of $\boldsymbol{c}$ directly (only the concepts related to themselves), we allow them to generate estimates $\hat{\boldsymbol{c}}$ of the environment's underlying concept state. Thus, at each time-step t, the environment is described by both its state s_t and concept-state $\boldsymbol{c}_t$, and each agent i produces both an action $a_{i,t}$ and a set of concept estimates $\boldsymbol{c}_{i,t}$.

4.2 Concept Bottleneck Architecture

In principle, there are a number of ways in which an agent's concept estimates $\hat{\boldsymbol{c}}$ can be modeled within the RL framework. Inspired by concept bottleneck models, we are most interested in examining policy architectures in which an agent's action is conditioned entirely on it's own concept estimates. To this end, we factorize an agent i's policy $\pi_i(a_i|o_i)$ into two sub-policies: $\pi_i^{\text{conc}} : \mathcal{O}_i \rightarrow \mathcal{C}_i$ mapping observations o_i to concept estimates $\hat{\boldsymbol{c}}_i$; and $\pi_i^{\text{act}} : \mathcal{C}_i \rightarrow \mathcal{A}_i$ mapping concept estimates $\hat{\boldsymbol{c}}_i$ to actions a_i. The composition of π_i^{conc} and π_i^{act} yields a standard policy mapping observations to actions:

$$\pi_i(a_i|o_i) = \pi_i^{\text{act}}(\pi_i^{\text{conc}}(\cdot|o_i))$$

More concretely, given an observation $o_{i,t}$ for some agent i and time-step t, the concept bottleneck first produces concept estimates $\hat{\boldsymbol{c}}_{i,t}\sim\pi_i^{\text{conc}}$, then uses those estimates alone to select an action $a_{i,t}\sim\pi_i^{\text{act}}$. Importantly, structuring the policy network such that actions are conditioned on concept estimates creates an intrinsically interpretable policy—the policy is forced to provide a human-understandable rendering of the factors driving its own decision making. Figure 1b shows the proposed architecture.

Importantly, $\hat{\boldsymbol{c}}$ can be thought of as an agent's *beliefs about the underlying game state*, which has implications for behavioral analysis. For example, if two agents i and j collide while moving in the environment, we can examine both $\hat{\boldsymbol{c}}_{i,t}$ and $\hat{\boldsymbol{c}}_{j,t}$ to identify which of the two agents incorrectly modeled the location its teammate.

4.3 Concept Bottleneck Learning

Under the concept bottleneck framing, learning a strong but interpretable policy requires both (i) learning how to predict concepts accurately, and (ii) learning how to select actions from those concepts effectively. To achieve the former, we introduce the following concept loss:

$$L_C(\boldsymbol{c}, \hat{\boldsymbol{c}}) = \sum_j L_{C_j}(c_j, \hat{c}_j) \tag{1}$$

where each component L_{C_j} measures the concept prediction error between the j'th predicted concept and its concept label. Note that the definition of each L_{C_j} is dictated by the values that both c_j and $\hat{c}_j$ take on—mean-squared error if c_j is a scalar, log loss if c_j is binary, cross entropy if c_j is categorical, etc. In practice, each pair of concepts $\boldsymbol{c}$ (from the environment's concept state) and concept estimates $\hat{\boldsymbol{c}}$ are stored in an agent's replay buffer during training alongside standard (s, a, r, s') tuples.

To learn a policy from concept estimates, we can attach Equation (1) as an auxiliary loss to the reward-based loss defined by any base RL algorithm (e.g., PPO [45], TD3 [46], etc). In general, if L_{RL} is a generic reward-based loss defined by some RL algorithm, we can construct the following joint concept bottleneck policy (CBP) objective:

$$L_{CBP} = L_{\text{RL}} + \lambda L_C \,, \tag{2}$$

where the concept loss coefficient λ weights the relative importance of concept prediction. Later, we examine λ's impact on training and, in particular, the learned behavior of a multi-agent team.

4.4 Behavioral Analysis via Concept Intervention

In addition to its intrinsic interpretability, a key feature of our method is its support of test-time intervention analysis. Once a CBP is trained, we can freeze the policy network, roll out a trajectory, and at each time-step replace an individual concept estimate (e.g., the j-th concept estimate $\hat{c}_j$) with a replacement value $\bar{c}_j$. In the simplest case, we can mask out the estimate completely by setting $\bar{c}_j = 0$. We can then observe the effect, if any, that $\bar{c}_j$ has on the agent's behavior.

Such test-time intervention is a powerful tool for analyzing multi-agent behavior. In multi-agent settings, it is often difficult to determine from reward alone if agents have learned to coordinate or if they are simply acting independently [37]. Using concept intervention, we propose a direct test for coordination. For an agent i to coordinate with another agent j, i must condition its policy on some information about j (either j's position, orientation, etc). It follows that if the agents are coordinating and we mask out i's concept estimates pertaining to j, we should observe a decrease in team performance. Conversely, if we do not see performance degrade, then i and j must not be explicitly coordinating (i.e. directly using signals from each other). By masking out the concepts pertaining to an agent's teammates, therefore, we can identify whether or not agents are coordinating.

Concept bottlenecks also provide a means for detecting common MARL failures, such as lazy agents. Here we define a lazy agent as one that does not contribute to increasing team reward through its own actions. For an agent i to contribute, it must therefore be conditioning its policy on information about its own interactions with the environment. We propose to define a lazy agent i as one that has learned a sub-optimal policy that does not encode such information, leading to unproductive behavior. We can therefore test for the *degree of laziness* of an agent by masking out its concept estimates *about itself* and examining the extent to which team performance degrades as a result. If team performance remains the same with agent i thus incapacitated, we conclude that i is a lazy agent.

5 Experiments

In this section, we evaluate the following hypotheses pertaining to Concept Bottleneck Policies:

H1: Identifying Emergent Coordination Can post-hoc concept intervention identify the level of coordination required by an environment? To what extent can it identify emergent coordination from policies that act independently? Moreover, if agents *are* coordinating, can intervention expose the specific inter-agent features that underlie that coordination? **H2: Identifying Lazy Agents** Can post-hoc concept intervention identify common failure modes such as lazy agents and, in general, measure an agent's contribution to the larger multi-agent system? **H3: Bottleneck Performance** Can CBPs match the performance of traditional, non-concept-based neural network policies; thereby achieving intrinsic interpretability without sacrificing performance?

Environment We use Melting Pot Collaborative Cooking as an experimental domain [10]. In Collaborative Cooking, agents must work together to cook and deliver soups in a kitchen-like environment. Solving the cooking task requires sophisticated coordination, involving both task partitioning—splitting a recipe into parts—and role assignment—distributing sub-tasks among agents. For these reasons, Collaborative Cooking is investigated in a number of prior works [47–49] and is emerging as a strong benchmark for multi-agent learning. We consider four variants of the game for $N{=}2$ agents. Each environment supports the following agent- and object-oriented concepts: (i) agent position; (ii) agent orientation; (iii) if an agent has a tomato, dish or soup; (iv) cooking pot position; (v) soup cooking progress; (vi) cooking pot tomato count; (vii) tomato position; (viii) dish position. Further details are provided in Appendix A.

Training and architecture For each of our experiments, we use PPO [45] as a backbone algorithm, which has been shown to be a state-of-the-art multi-agent algorithm [50, 51]. We augment the PPO objective with the concept loss defined in Equation (1) (we refer to this combination as ConceptPPO moving forward). Each agent's policy network consists of CNN and MLP encoders (for image and position/orientation inputs, respectively), followed by a two-layer MLP and a linear mapping that compresses the encoded inputs into concept predictions. Concept estimates are fed through a two-layer MLP, which produces the final action. ReLU activation is used throughout (except in the bottleneck layer itself). As a baseline, we use vanilla PPO (no concept loss) with the same architecture. We train 10 individual policies across each of the following values of λ:

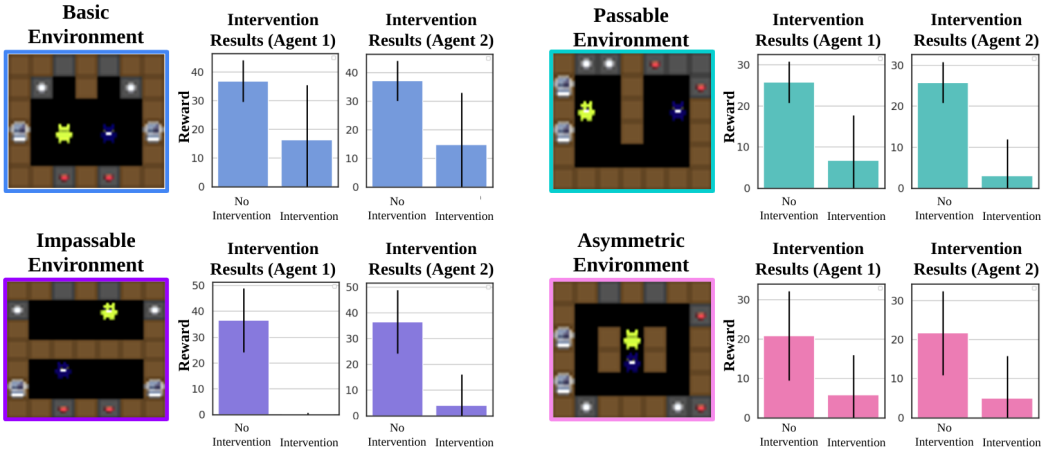


Figure 2: **Does the environment require coordination?** Averaging the impact of concept intervention over all policies trained in an environment reveals the extent to which coordination is required by that environment. In the impassable environment, agents cannot solve the task without coordinating, leading to consistent performance drops under intervention. In the basic environment, policies that coordinate (and therefore fail under intervention) are averaged with policies that act independently (and are uninterrupted by intervention), so the overall impact of intervention is less severe.

$\{0.01, 0.1, 0.25, 0.5, 0.75, 1.0.2.0, 5.0, 10.0\}$. Additional training details, including hyperparameter sweeps, are provided in Appendix A.1.

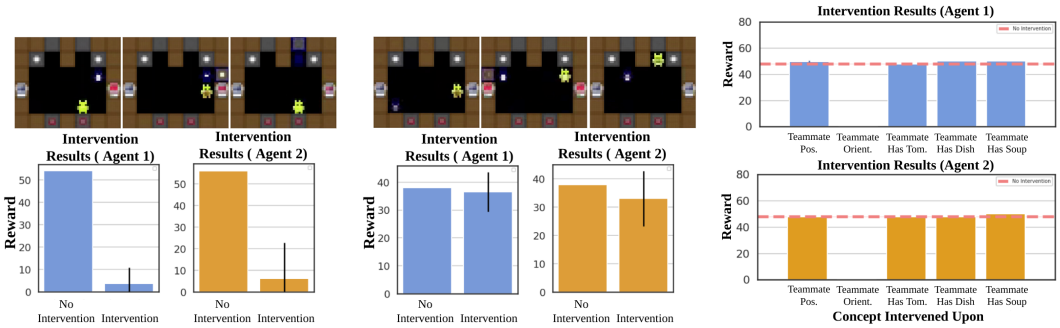
5.1 H1: Identifying Emergent Coordination

We evaluate the efficacy of Concept Bottleneck Policies as a mechanism for understanding emergent multi-agent behavior using the post-hoc intervention technique introduced in Section 4.4.

Environmental Coordination Demands For each Concept PPO and PPO policy trained in our cooking environments, we mask out, for each agent, all of the concepts related to that agent's teammate. We measure the average cumulative reward attained over 100 trajectories each. The results of this intervention test are shown in Fig. 2. The level of coordination required by the environment is apparent from the severity of performance degradation that results from intervening on each agent's estimates of its teammate. Most notable is the contrast between the basic environment (blue) and the impassable environment (purple). The impassable environment requires strict coordination—agents can only access a subset of ingredients and must pass items to each other across the center divider. In this case, intervention performance drops to near-zero across all policies. In the basic environment, on the other hand, there are no obstacles and agents have access to their own supply of ingredients; and so both policies in which agents coordinate and policies in which agents act independently are successful. Consequently, the impact of intervention is much less severe, as the performance of policies that coordinate (and fail under intervention) is averaged in with independent policies that are unaffected by intervention. Altogether, these results indicate that our method accurately distinguishes environments that require coordination from those that do not.

Identifying Coordination vs. Independent Behaviors Our next intervention test aims to disentangle, within a single environment, policies that learn to coordinate from those that opt for independent action. We hone in on the basic environment—because it most clearly supports independent policies as a successful strategy—and plot the performance of each policy individually (rather than averaging across all policies). As shown in Fig. 3, we again find a stark contrast between coordination and non-coordination, but this time at the level of individual policies.

In the trajectory snapshot in Fig. 3a, we see an emergent strategy in which two agents coordinate through role assignment. In particular, the orange agent maneuvers in the bottom-right corner of the environment picking up tomatoes and bringing them to the cooking pot; while the blue agent stays in the top-right corner running dishes to and from the cooking pot to deliver soup. As expected, intervention over this strategy leads to a catastrophic drop in performance. Each agent's coordination hinges on an accurate modeling of its teammate. In the trajectory snapshot in Fig. 3b, we find a much



(a) Coordinating Agents. (b) Independent Agents. (c) Teammate Concepts.

Figure 3: **Identifying and Understanding Coordination**. In the basic environment, the impact of concept intervention (in terms of decreased reward) disentangles (a) policies that coordinate and (b) policies that act independently. Moreover, intervening over each agent's teammate-related concepts individually identifies that inter-agent factors driving the team's coordination strategy. Surprisingly, teammate orientation is relied upon heavily by both agents for coordination.

different strategy in which agents complete the cooking task independently on opposite sides of the environment. Here, intervention does not hurt performance (or hurts performance only marginally), as neither agent needs to closely monitor its teammate to complete the task. Finally, because intervention *does not* impact performance when agents are not coordinating, this result provides further evidence that coordination detection through intervention is accurate and reliable.

Factors of Coordination Next, we examine the policies that *have* learned to coordinate and attempt to pinpoint the specific inter-agent features that drive their coordination. To this end, rather than intervening over all concepts pertaining to an agent's teammate at once, we iterate over each of them individually. As we hypothesize, if we know that the agents are coordinating, then intervening over concepts related to the other agent should result in a sharp drop in performance. The results of this intervention test are shown in Fig. 3c.

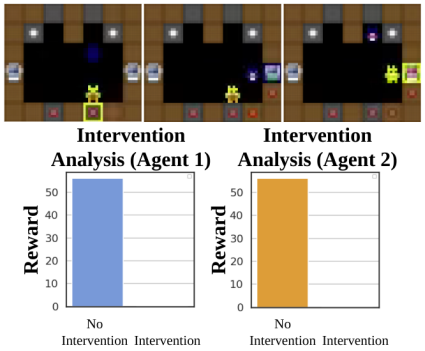
Interestingly, performance only drops when intervening over the orientation concept, indicating that agents are primarily using the orientation of the other agent as a signal to drive coordination. This is curious, as we would expect coordination to involve multiple individual sources of inter-agent information; or combinations thereof. We emphasize that identifying the specific factors that underlie an agent's coordination strategy would be much more difficult without the concept intervention enabled by our concept bottleneck policies.

For completeness, we conduct two additional experiments to support this analysis. First, we rule out the possibility that intervening with a fixed mask to zero-out orientation creates an OOD (or otherwise adversarial) input that the agents' policies cannot handle. We do this by manufacturing intervention masks that are both in- and out-of-distribution, using an empirical sample of the orientations experienced by each agent at test-time; and show that our results are consistent across both cases. Second, we train a new set of ConceptPPO policies without orientation as a concept and re-run this iterative intervention over concepts pertaining to each agent's teammate. These results show that, without orientation, agent coordination latches on to another single concept estimate as its driving signal. Detailed discussion of these supporting results is provided in Appendix B.

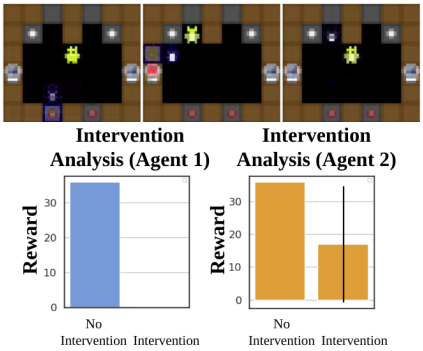
5.2 H2: Identifying Lazy Agents

In addition to coordination successes, the concept intervention technique enabled by our method allows us to test for *coordination failures*. Here we test for the presence of lazy agents by masking out each agent's concept predictions about itself—including the agents own position, orientation, etc. According to our hypothesis, if an agent is acting productively in the environment, removing this information will greatly hinder that agent's performance; and team performance as a whole. If performance does not decrease, the agent likely isn't contributing to the task.

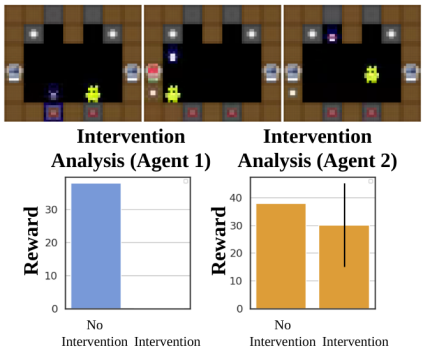
Figure 4 shows the results of this lazy agent test for four policies, each differing in the strength of their contribution to the team. As predicted by our hypothesis, the productivity of an agent can be exposed through the impact of the intervention test. When agents are both contributing to the task



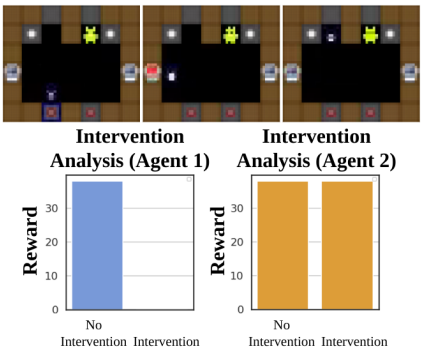
(a) Strong Coordination.



(b) Moderate Coordination



(c) No Coordination.



(d) No Movement.

Figure 4: **Identifying Lazy Agents** Intervening on an agent's concepts pertaining to itself exposes its level of involvement in the task. In cases where coordination is either (a) strong or (b) moderate, intervention has a large negative impact on the performance of both agents. In cases where an agent is (c) not coordinating or (d) not moving at all, intervention has a negligible impact on performance.

(top row), concept intervention has a strong negative impact on performance for both agents. When an agent is not contributing to the task (bottom row), concept intervention has a much smaller impact on performance. In the extreme case in which one agent does not move at all over the course of a trajectory (bottom right), performance is not impacted at all by intervention.

Crucially, the magnitude of performance degradation is a direct function of the productivity of each agent. Thus, these results not only demonstrate that our method can reliably diagnose lazy agents, but also suggest that it can quantify the *degree of laziness* as a function of performance degradation under intervention. We stress that diagnosing lazy agents and the degree to which an agent contributes to a multi-agent strategy would be more difficult without our concept bottleneck architecture.

5.3 H3: Bottleneck Performance

To evaluate the general performance of our method, we compare the performance of Concept PPO to the non-concept PPO baseline. We measure performance as the average cumulative reward obtained over 100 test-time trajectories (and five random seeds each). Figure 5 provides an overview of these results. We find evidence that ConceptPPO can match the performance of PPO across each of our environments for small values of the concept cost coefficient ($\lambda <= 0.5$). This is an important result from the perspective of interpretability. It demonstrates that, if λ is tuned appropriately, it is possible to train intrinsically-interpretable policy networks—where, notably, decisions are expressed in human-understandable concepts—without sacrificing in task performance.

Figure 5 also shows, though, that over-valuing the concept prediction loss causes performance to degrade. In particular, performance falls for $\lambda > 0.75$ and, in all but the basic environment, collapses to zero for larger values ($\lambda \geq 5.0$). In these cases, agent policies begin to overvalue concept prediction accuracy relative to reward, which changes the nature of their emergent coordination strategy. As λ increases, agents are more incentivized to place themselves in areas of the state-space where they

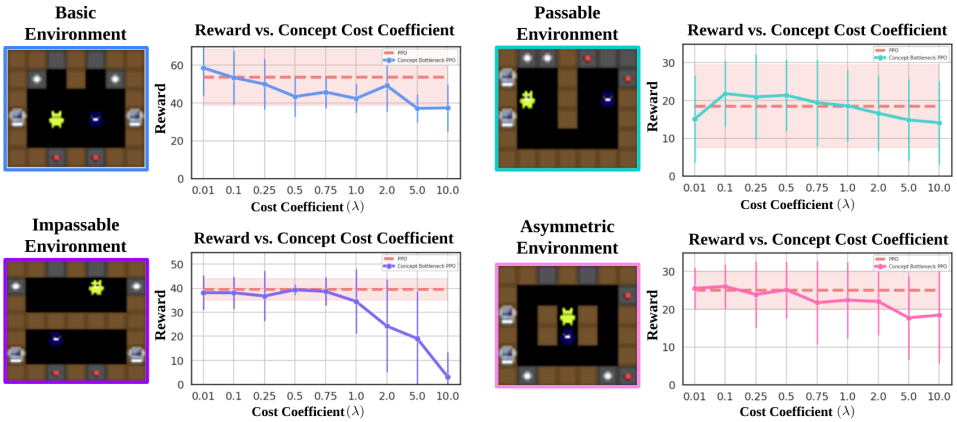
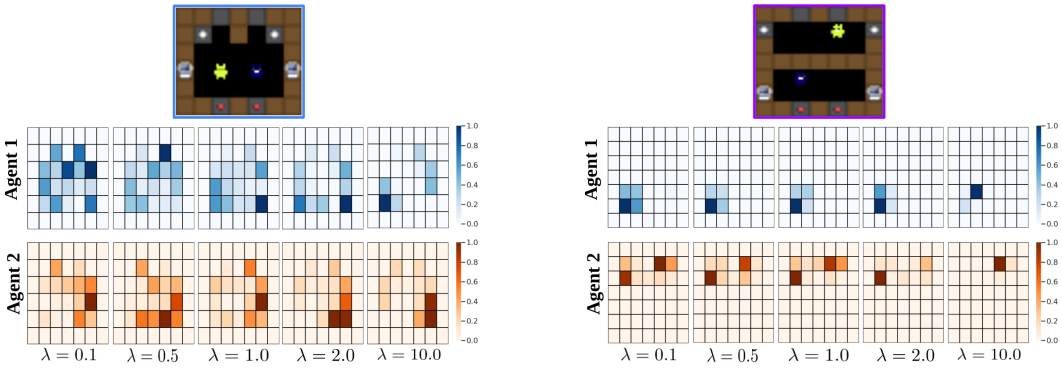


Figure 5: Performance of ConceptPPO and PPO in the cooking domains. With $\lambda \leq 0.5$, ConceptPPO matches the performance of non-concept-based PPO. This indicates that ConceptPPO achieves interpretability without sacrificing performance. For higher values of λ, agent behavior changes as a result of over-valuing the conecpt prediction loss, which in turn impacts performance.



(a) Basic Environment.

(b) Impassable Environment.

Figure 6: State visitation results. As concept cost coefficient λ increases, agents visit smaller regions of the state space. Eventually, behavior collapses and agents opt not to move at all.

can accurately predict concepts, including those related to the other agent. We hypothesize that this causes agents to develop behaviors that are less efficient from a reward perspective, but more effective from a predictability perspective, such as clustering themselves together spatially.

State Visitation Analysis To illustrate the behavioral changes induced by λ, we measure the distribution of states visited by each agent across a subset of the λ values used during training. The results for each agent are plotted as a heatmap in Fig. 6. As expected, we find that agents visit smaller and smaller regions of the state space as λ increases until, finally, behavior collapses. In the most extreme cases (e.g., $\lambda = 10.0$ in the impassable environment) agents opt not to move at all—after all, the easiest way to predict concepts accurately is to not change anything in the environment.

6 Conclusion

We proposed Concept Bottleneck Policies as an intrinsically interpretable, concept-based policy learning method for MARL. We demonstrated that our method is effective for understanding emergent multi-agent behavior. In particular, Concept Bottleneck Policies support test-time concept intervention, which can be used to identify when a multi-agent team has learned to coordinate, what inter-agent features drive that coordination, and to what extent coordination is required in a particular environment. Moreover, concept intervention can help diagnose coordination failures like lazy agents. Experimental results also show that our method achieves comparable levels of performance to traditional non-concept-based policy learning methods.

Acknowledgments and Disclosure of Funding

We would like to thank Yannick Assogba, Lucas Dixon, Asma Ghandeharioun, Adam Pearce, Ryan Mullins, Joel Z. Leibo, Edgar Duéñez Guzmán, Peter Hase, Srivatsan Krishnan, and Blair Bilodeau for their helpful feedback on this work.

References

[1] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

[2] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv preprint arXiv:1712.01815*, 2017.

[3] Noam Brown and Tuomas Sandholm. Superhuman ai for multiplayer poker. *Science*, 365 (6456):885–890, 2019.

[4] Julien Perolat, Bart de Vylder, Daniel Hennes, Eugene Tarassov, Florian Strub, Vincent de Boer, Paul Muller, Jerome T Connor, Neil Burch, Thomas Anthony, et al. Mastering the game of stratego with model-free multiagent reinforcement learning. *arXiv preprint arXiv:2206.15378*, 2022.

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

[6] Oriol Vinyals, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Georgiev, et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575(7782):350–354, 2019.

[7] Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysław Dębiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019.

[8] Joel Z Leibo, Vinicius Zambaldi, Marc Lanctot, Janusz Marecki, and Thore Graepel. Multi-agent reinforcement learning in sequential social dilemmas. *arXiv preprint arXiv:1702.03037*, 2017.

[9] Siqi Liu, Guy Lever, Zhe Wang, Josh Merel, S. M. Ali Eslami, Daniel Hennes, Wojciech M. Czarnecki, Yuval Tassa, Shayegan Omidshafiei, Abbas Abdolmaleki, Noah Y. Siegel, Leonard Hasenclever, Luke Marris, Saran Tunyasuvunakool, H. Francis Song, Markus Wulfmeier, Paul Muller, Tuomas Haarnoja, Brendan Tracey, Karl Tuyls, Thore Graepel, and Nicolas Heess. From motor control to team play in simulated humanoid football. *Science Robotics*, 7(69): eabo0235, 2022. doi: 10.1126/scirobotics.abo0235.

[10] Joel Z Leibo, Edgar A Dueñez-Guzman, Alexander Vezhnevets, John P Agapiou, Peter Sunehag, Raphael Koster, Jayd Matyas, Charlie Beattie, Igor Mordatch, and Thore Graepel. Scalable evaluation of multi-agent reinforcement learning with melting pot. In *International Conference on Machine Learning*, pages 6187–6199. PMLR, 2021.

[11] Michael Bradley Johanson, Edward Hughes, Finbarr Timbers, and Joel Z Leibo. Emergent bartering behaviour in multi-agent reinforcement learning. *arXiv preprint arXiv:2205.06760*, 2022.

[12] Stephan Zheng, Alexander Trott, Sunil Srinivasa, Nikhil Naik, Melvin Gruesbeck, David C Parkes, and Richard Socher. The ai economist: Improving equality and productivity with ai-driven tax policies. *arXiv preprint arXiv:2004.13332*, 2020.

[13] Shayegan Omidshafiei, Andrei Kapishnikov, Yannick Assogba, Lucas Dixon, and Been Kim. Beyond rewards: a hierarchical perspective on offline multiagent behavioral analysis. *arXiv preprint arXiv:2206.09046*, 2022.

[14] Bowen Baker, Ingmar Kanitscheider, Todor Markov, Yi Wu, Glenn Powell, Bob McGrew, and Igor Mordatch. Emergent tool use from multi-agent autocurricula. *arXiv preprint arXiv:1909.07528*, 2019.

[15] Thomas McGrath, Andrei Kapishnikov, Nenad Tomašev, Adam Pearce, Demis Hassabis, Been Kim, Ulrich Paquet, and Vladimir Kramnik. Acquisition of chess knowledge in alphazero. *arXiv preprint arXiv:2111.09259*, 2021.

[16] Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215, 2019.

[17] Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. Concept bottleneck models. In *International Conference on Machine Learning*, pages 5338–5348. PMLR, 2020.

[18] He He, Jordan Boyd-Graber, Kevin Kwok, and Hal Daumé III. Opponent modeling in deep reinforcement learning. In *International conference on machine learning*, pages 1804–1813. PMLR, 2016.

[19] Roberta Raileanu, Emily Denton, Arthur Szlam, and Rob Fergus. Modeling others using oneself in multi-agent reinforcement learning. In *International conference on machine learning*, pages 4257–4266. PMLR, 2018.

[20] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

[21] Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. Smoothgrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*, 2017.

[22] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International conference on machine learning*, pages 3319–3328. PMLR, 2017.

[23] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014.

[24] Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, et al. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In *International conference on machine learning*, pages 2668–2677. PMLR, 2018.

[25] Amirata Ghorbani, James Wexler, James Y Zou, and Been Kim. Towards automatic concept-based explanations. *Advances in Neural Information Processing Systems*, 32, 2019.

[26] David Bau, Jun-Yan Zhu, Hendrik Strobelt, Agata Lapedriza, Bolei Zhou, and Antonio Torralba. Understanding the role of individual units in a deep neural network. *Proceedings of the National Academy of Sciences*, 117(48):30071–30078, 2020.

[27] Asma Ghandeharioun, Been Kim, Chun-Liang Li, Brendan Jou, Brian Eoff, and Rosalind W Picard. Dissect: Disentangled simultaneous explanations via concept traversals. *arXiv preprint arXiv:2105.15164*, 2021.

[28] Zhi Chen, Yijie Bei, and Cynthia Rudin. Concept whitening for interpretable image recognition. *Nature Machine Intelligence*, 2(12):772–782, 2020.

[29] Chih-Kuan Yeh, Been Kim, Sercan Arik, Chun-Liang Li, Tomas Pfister, and Pradeep Ravikumar. On completeness-aware concept-based explanations in deep neural networks. *Advances in Neural Information Processing Systems*, 33:20554–20565, 2020.

[30] Wolfgang Stammer, Patrick Schramowski, and Kristian Kersting. Right for the right concept: Revising neuro-symbolic concepts by interacting with their explanations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3619–3629, 2021.

[31] Osbert Bastani, Carolyn Kim, and Hamsa Bastani. Interpretability via model extraction. *arXiv preprint arXiv:1706.09773*, 2017.

[32] Prashan Madumal, Tim Miller, Liz Sonenberg, and Frank Vetere. Explainable reinforcement learning through a causal lens. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 2493–2500, 2020.

[33] Zhao Yang, Song Bai, Li Zhang, and Philip HS Torr. Learn to interpret atari agents. *arXiv preprint arXiv:1812.11276*, 2018.

[34] Dan Amir and Ofra Amir. Highlights: Summarizing agent behavior to people. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, pages 1168–1176, 2018.

[35] Pedro Sequeira and Melinda Gervasio. Interestingness elements for explainable reinforcement learning: Understanding agents' capabilities and limitations. *Artificial Intelligence*, 288:103367, 2020.

[36] Siqi Liu, Guy Lever, Zhe Wang, Josh Merel, SM Ali Eslami, Daniel Hennes, Wojciech M Czarnecki, Yuval Tassa, Shayegan Omidshafiei, Abbas Abdolmaleki, et al. From motor control to team play in simulated humanoid football. *Science Robotics*, 7(69):eabo0235, 2022.

[37] Max Jaderberg, Wojciech M Czarnecki, Iain Dunning, Luke Marris, Guy Lever, Antonio Garcia Castaneda, Charles Beattie, Neil C Rabinowitz, Ari S Morcos, Avraham Ruderman, et al. Human-level performance in 3d multiplayer games with population-based reinforcement learning. *Science*, 364(6443):859–865, 2019.

[38] Jessica Zosa Forde, Charles Lovering, George Konidaris, Ellie Pavlick, and Michael L Littman. Where, when & which concepts does alphazero learn? lessons from the game of hex. In *AAAI Workshop on Reinforcement Learning in Games*, volume 2, 2022.

[39] Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, et al. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022.

[40] Bradley Hayes and Julie A Shah. Improving robot controller transparency through autonomous policy explanation. In *2017 12th ACM/IEEE International Conference on Human-Robot Interaction (HRI*, pages 303–312. IEEE, 2017.

[41] Anuj Mahajan, Tabish Rashid, Mikayel Samvelyan, and Shimon Whiteson. Maven: Multi-agent variational exploration. *Advances in Neural Information Processing Systems*, 32, 2019.

[42] Tonghan Wang, Tarun Gupta, Anuj Mahajan, Bei Peng, Shimon Whiteson, and Chongjie Zhang. Rode: Learning roles to decompose multi-agent tasks. *arXiv preprint arXiv:2010.01523*, 2020.

[43] Michael L Littman. Markov games as a framework for multi-agent reinforcement learning. In *Machine learning proceedings 1994*, pages 157–163. Elsevier, 1994.

[44] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

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

[46] Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International conference on machine learning*, pages 1587–1596. PMLR, 2018.

[47] Sarah A Wu, Rose E Wang, James A Evans, Joshua B Tenenbaum, David C Parkes, and Max Kleiman-Weiner. Too many cooks: Bayesian inference for coordinating multi-agent collaboration. *Topics in Cognitive Science*, 13(2):414–432, 2021.

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

[49] DJ Strouse, Kevin McKee, Matt Botvinick, Edward Hughes, and Richard Everett. Collaborating with humans without human data. *Advances in Neural Information Processing Systems*, 34: 14502–14515, 2021.

[50] Christian Schroeder de Witt, Tarun Gupta, Denys Makoviichuk, Viktor Makoviychuk, Philip HS Torr, Mingfei Sun, and Shimon Whiteson. Is independent learning all you need in the starcraft multi-agent challenge? *arXiv preprint arXiv:2011.09533*, 2020.

[51] Chao Yu, Akash Velu, Eugene Vinitsky, Yu Wang, Alexandre Bayen, and Yi Wu. The surprising effectiveness of ppo in cooperative, multi-agent games. *arXiv preprint arXiv:2103.01955*, 2021.

[52] Ghost Town Games. Overcooked. https://store.steampowered.com/app/ 448510/Overcooked/, 2016.