InfiniteKitchen: Cross-environment Cooperation for Zero-shot Multi-agent Coordination

Kunal Jha, Max Kleiman-Weiner[*], Natasha Jaques[*]
Department of Computer Science
University of Washington
Seattle, WA 98105, USA
{kj,maxkw,nj}@cs.washington.edu

Abstract

Zero-shot coordination (ZSC) is an important challenge for developing adaptable AI systems that are capable of collaborating with humans in unfamiliar tasks. While prior work has mainly focused on adapting to new partners [13, 28], generalizing cooperation across different environments is equally important. This paper investigates training AI agents in self-play (SP) to achieve zero-shot collaboration with novel partners in novel tasks. We introduce a new Jax-based, procedurally generated environment for multi-agent reinforcement learning, Infinite Kitchen. Our rule-based generator creates billions of solvable kitchen configurations that enable the training of a single, generalizable agent that can adapt to new levels. Our results show that exposure to diverse levels in self-play consistently improves generalization to new partners, with graph neural network (GNN) based architectures achieving the highest performance across many layouts. Our findings suggest that learning to collaborate across a multitude of unique scenarios encourages agents to develop maximally general norms, which prove highly effective for collaboration with different partners when combined with appropriate inductive biases.

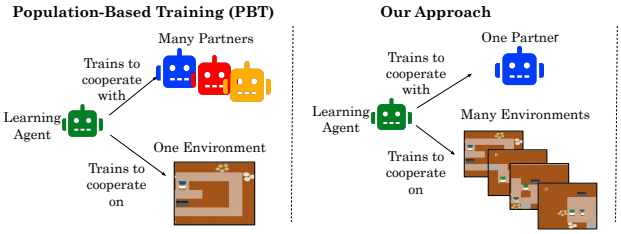
1 Introduction

Humans excel at ad-hoc cooperation, readily adapting to new partners and environments by jointly attending to relevant cues, reasoning about shared intentions, and playing their role within an implicit collective plan [30, 14, 25, 33]. This ability to *compositionally* represent collective tasks allows skills to transfer across domains. For instance, after mastering a family recipe with their parents, a novice chef can seamlessly transition to cooking that dish and more at home with their spouse. While understanding these cognitive mechanisms is crucial for building AI that can coordinate in novel scenarios, current reinforcement learning methods have yet to address this challenge. Developing AI capable of zero-shot coordination (ZSC) with new partners in unfamiliar tasks is essential for creating adaptable, human-compatible, AI agents [16].

Prior work on ZSC has mainly focused on adapting to novel partners, using methods like population-based training and self-play (SP) algorithms. These approaches either simulate diverse partner strategies during training [35, 4, 27, 23] or adjust the training objectives to explore broader strategy spaces [13]. However, they often suffer from high computational costs, poor scalability, or brittle coordination strategies. While generalizing to new partners is important, existing research has largely overlooked the equally important challenge of generalizing across different environments. Neglecting this aspect can lead to a false sense of progress in building AI that reliably coordinates with humans. Each agent can only cooperate on the specific problem instance it was trained on and thus lacks a

[*]Denotes equal contribution.

Accepted at the Intrinsically Motivated Open-ended Learning workshop at NeurIPS 2024.



Evaluation

Coordination Task	PBT	Our Approach
Novel Partners, Same Environment	Zero-shot Coordination ✅	Zero-shot Coordination ✅
Same Partner, *Novel* Environments	Coordination Failure ❌	Zero-shot Coordination ✅
Novel Partners, *Novel* Environments	Coordination Failure ❌	Zero-shot Coordination ✅

Figure 1: Overview of learning general cooperation through Infinite Kitchen. By training agents in self-play on a large distribution of environments, we find that agents develop the ability to coordinate with novel partners and novel problems, contrasting prior work which suggests self-play is insufficient for learning general norms for cooperation.

more general notion of cooperation. In this work, we investigate the following question: how can we train AI agents in self-play capable of zero-shot collaboration with *novel partners* in *novel tasks*? We focus on the game Overcooked, a 2-player, cooperative 2D cooking game where an AI agent collaborates with an AI or human partner to prepare a recipe [4, 33, 28, 37, 23]. We introduce a procedurally generated Overcooked environment called Infinite Kitchen. Infinite Kitchen is implemented in Jax and is highly performant; we achieve processing speeds of 10 million steps per minute on a single GPU. Unlike previous work that studies at most five handcrafted levels [4, 27, 23], our system generates billions (4.60×10^{28} possible initial states) of solvable kitchen configurations, enabling the training of a single, generalizable agent that adapts to diverse scenarios rather than memorizing specific solutions. In contrast to many unsupervised environment design (UED) approaches that struggle with unsolvable or trivial scenarios, our rule-based generator ensures each layout presents a legitimate coordination challenge, such as those depicted in Figure 2.

Our experiments intriguingly reveal that exposure to diverse *environments* during training consistently improves generalization to novel *partners* (see Figure 1). From these results, we theorize that learning to collaborate with a single partner across many levels during training encourages agents to develop a minimal set of maximally general norms. We introduce a novel model architecture based on Graph Neural Networks (GNNs). We show that when Infinite Kitchen training is paired with relational inductive biases [1], the resulting learned norms prove highly effective for collaboration with different partners across multiple grids. Our results suggest that the diversity of problems encountered, rather than partner diversity alone, plays a crucial role in fostering general cooperative behaviors.

2 Related Work

The ZSC paradigm arises when generating data and retraining an AI for new tasks is costly or inconvenient, and its solution has the potential for broad impacts across robotics [2, 24], digital assistants [12, 18, 36], and other scientific domains [11, 19, 5].

Self-Play and Population-based Training ZSC has been approached through self-play and population-based training. Self-play has been successful in many games [34, 26, 32, 38, 4], but often leads to inflexible strategies that struggle with unfamiliar partners [28, 16]. Population-based methods, which train an AI with diverse partners and then evaluate it with humans, generally outperform self-play in zero-shot Human-AI coordination. In Overcooked, previous works [4, 28, 37, 23] introduced diversity through variations in initial conditions, action trajectories, or rewards, but these approaches are computationally expensive and fail to produce reusable strategies across tasks. Our approach systematically varies the environment to increase state diversity, enabling cross-task generalization while reducing the need to train multiple policies by remaining in the self-play setting.

Procedural Environment Generation Recent work has demonstrated that procedurally generated environments can improve the generalizability of reinforcement learning (RL) methods in single and multi-agent settings [7, 8, 10, 3, 6, 22]. These studies show that exposure to a large and diverse set of samples enhances generalization [7]. However, they typically evaluate agents with the same team seen during training, which doesn't address the core challenge of zero-shot coordination (ZSC). Related to our work, Ruhdorfer et. al. [20] study unsupervised environment design (UED) in a in the context of Overcooked. However, their work does not prevent the generation of impossible coordination challenges [9, 17, 15] and their results reveal poor cross-play performance on held-out levels. In

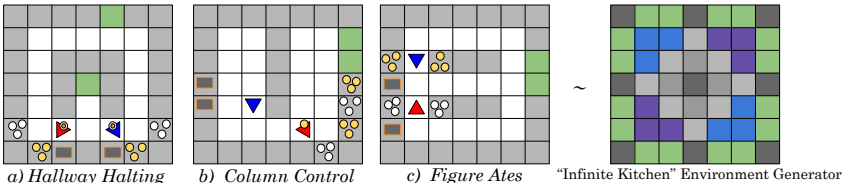


Figure 2: A few New Coordination Challenges from "Infinite Kitchen" - a procedural level generator capable of creating billions of layouts in *Overcooked*. 2a shows symmetric advantages with one optimal delivery location, requiring agents to break symmetry in who delivers first. Rigid conventions, like always having blue go first, can fail with novel partners, especially if red takes the lead, potentially trapping the other agent. 2b and 2c illustrate asymmetric advantages, where one agent is closer to the pot and the other to remaining items. These layouts are challenging due to navigational complexity and efficiency issues, particularly if agents develop brittle strategies. 2c is especially difficult due to the higher variability in conventions, increasing the risk of learning arbitrary rules that don't generalize to new partners.

contrast, we show for the first time how training on a vast number of novel, solvable coordination challenges improves the generalization of AI to novel partners.

3 Methods

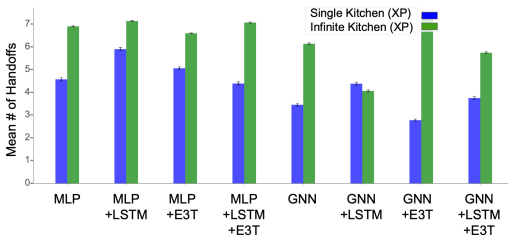
Procedurally Generated Overcooked We extend the Overcooked environment from the JaxMARL project to support a wider variety of levels while keeping observation and action spaces consistent [21]. We observe three main sources of variation that contribute to a range of coordination challenges in Overcooked: (1) barriers that hinder movement, (2) asymmetric advantages encourage self-play agents to memorize a single role, and (3) crowded objects restricting an agent's access to an item while it is being used by another.

To address these challenges, agents need to flexibly coordinate roles, anticipate each other's actions, and navigate obstacles. We randomly generate features like walls, plates, pots, and onions within the grid, using wall-predicates to structure the generation process and introduce adjustable complexity. This ensures solvable layouts by placing essential objects in reachable areas, as shown in Figure 6 and detailed in the Appendix, but also varied environments, since we sample additional delivery locations, plate piles, pots, and onion piles on unoccupied walls. In the worst case where a wall divides both agents and all items are located on one side of the divider, a single agent will be able to achieve the task on their own. If the items are spread across both sides of the divider, then this is a coordination problem similar to the *Forced Coordination* Overcooked layout [4]. In the remaining cases, since a single agent is able to complete the task on its own, it is achievable by a pair of agents but still presents challenging navigation and coordination problems.

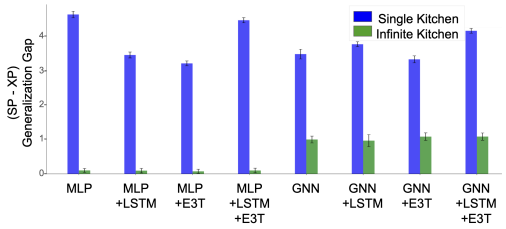
Our generator creates new coordination challenges in Overcooked, as illustrated in Figure 2. Moreover, Jax allows us to run the entire training and evaluation pipeline, from the environment generation to the neural network updating of agents, at 10 million steps per minute on a single GPU. We leverage this speed to train our agents on Infinite Kitchen for 2 billion steps.

Graph Attention Networks Previous work in Overcooked has relied on Convolutional Neural Networks (CNNs) for feature extraction from pixel observations. However, human cognition heavily relies on relational bias and attention, which facilitate complex social reasoning and generalization by forming abstract representations and attributing intentions to agents [30]. To incorporate these cognitive aspects, we extend the use of Graph Attention Networks [31] to Overcooked as a feature extractor, replacing the traditional MLP or CNN approach in Overcooked.

Following [29], we represent each cell's sparse 26-channel symbolic encoding from the original JaxMARL implementation [21] as a node in a graph, connecting all cells to all agents. This allows agents to reason about their environment in relation to other objects and each other. After processing this graph through a single graph attention layer, we concatenate the node-level information into a cohesive scene embedding using a Linear layer with ReLU activation. The scene embedding is then passed through an LSTM and three Linear layers to create the actor and critic networks. We refer to this architecture as GNN+LSTM, and also introduce a variant without the LSTM called GNN.



(a) Average Cross Play Agent Performance across 5 original Overcooked Levels



(b) Generalization Gap Between Self Play and Cross Play Performance

Figure 3: Comparison of agent performance across different architectures and training regimes. 3a shows the performance of Single Kitchen agents within their training environment (blue bars), whereas for Infinite Kitchen agents (green bars) we show generalization to a novel, held-out test environment. 3b shows the generalization gap between self play and cross play performance for Single Kitchen and Infinite Kitchen agents, where lower values indicate a better ability to generalize to novel partners on the 5 original grids.

4 Results

In the following section, we evaluate the effect of training on our Procedurally Generated Overcooked environment, Infinite Kitchen, for ZSC with AI baselines, although future work will evaluate the performance of these agents playing with humans.

Held-out test environments. For evaluation, we generate a set of 100 grids from the procedural environment generator and prevent them from being sampled during training.We also keep the expanded version of the 5 original Overcooked grids shown in Figure 5 held out of the training data. We train 6 teams of agents using IPPO on 2 billion steps split across 32 parallel environments per team. We compare the performance of a single model trained across multiple grids (which we refer to as Infinite Kitchen) to models trained only on one grid (which we refer to as Single Kitchen).

Baselines. In keeping with conventional architecture for Overcooked with symbolic observations [21], we train a single Linear layer feature extractor, a Recurrent Neural Network to encode historical context within the scene, and three additional Linear layers followed by ReLU activations on the outputs of first two for both the actor and critic modules required by IPPO. We call this baseline (MLP+LSTM), and call a variant with the same architecture except no Recurrent layers (MLP).All network architectures are trained in both Single Kitchen and Infinite Kitchen. We also train and evaluate the State-of-the-Art Self-Play algorithm for cross-play generalization in Overcooked, E3T [35], on both Single and Infinite Kitchen, using either an MLP or GNN feature extractor but keeping all other architectural decisions true to the original work.

Cross-Play (XP) Evaluation Metric. We compare the performance of all agents and architectures in self-play (SP) and cross-play (XP). We assess the mean number of handoffs (successful deliveries) in two settings. The SP setting is when an agent plays with the same partner it saw during training but does not do any learning. We generate 100 rollouts of agent behavior per grid per pair of agents trained together (with 6 seeds per architecture/training regime this becomes 600 data points per model evaluated. For XP, we assess the mean number of handoffs for agents playing with partners they have not seen during training but rely on the same algorithm and architecture. This results in 3000 datapoints per model evaluated given 6 seeds all playing with each other.

Findings We compare the zero-shot coordination performance between AI baselines using different architectures in Figure 3. The results show that even for simple architectures such as an MLP without any recurrent networks, a single model training on Infinite Kitchen significantly improves XP performance on the conventional 5 layouts than models which only trained on one layout. This finding holds true even when compared to the state-of-the-art method for ZSC coordination on a single kitchen (E3T). When Infinite Kitchen agents are trained using the E3T algorithm, we observe that the addition of a Graph Network feature extractor boosts cross-play performance further. Moreover, Figure 3b demonstrates that this improved generalization is a direct product of training on Infinite Kitchen: there is a lower generalization gap across the board between SP and XP performance for

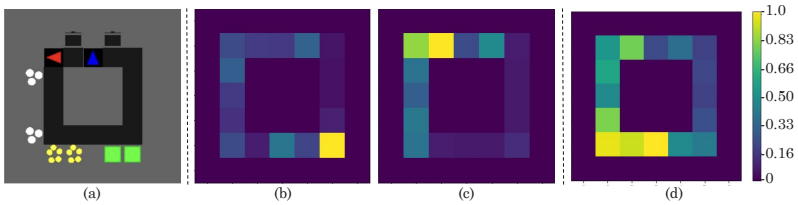


Figure 4: Heatmap of agent trajectories in an episode of *Forced Coordination* (grid illustrated in 4a). (4b, 4c) show the normalized frequency of visited locations by Single Kitchen agents in Self Play. 4c shows the normalized frequency of visited locations by an Infinite Kitchen agent. Brighter regions indicate the location has been visited more often. The highly concentrated regions of cells visited for SK seeds in comparison to the IK seed indicates that SK agents learn brittle and rigid strategies which do not generalize to new partners. In contrast, IK agents show more diverse and adaptable behavior by increasing its state coverage.

Infinite Kitchen agents than for Single Kitchen agents. This indicates that the norms learned during training across varied environments are enabling transfer to novel partners as well as problems.

We validate our findings in Figure 7 in the Appendix, which shows the cross-play performance of Infinite Kitchen models on 100 held-out levels created by the procedural environment generator. Infinite Kitchen agents consistently cooperate with their training partner to solve new tasks, and this ability generalizes to other partners (green bars). Figure 7 also shows the performance of Single Kitchen agents trying to generalize to novel environments and partners. None of these agents could solve a single task created by the procedural generator because they have overfit to the environment they trained on, forming brittle visual representations and cooperation strategies. Thus, the mean number of handoffs on cross-level evaluations for existing methods is 0 even in SP, compared to the 9 deliveries achieved by IK agents in XP settings using simple architectures such as MLPs. GNNs provide an additional boost to this cross-level, cross-partner generalization, boosting the number of handoffs to 10 on some IK generated grids.

Qualitative Analysis of Learned Norms We provide empirical intuition for the success of IK methods performing in XP with novel partners. After being exposed to many diverse grids, IK agents have gained a richer representation of the compositional nature of a task and the consequences of movement and interaction in a grid. This means they can explore more of the state space with confidence, so if a novel partner takes an action they were not expecting during training, they can still optimally respond. This can be seen in Figure 4, where we compare the frequency of different locations visited by SK and IK agents in *Coordination Ring*. The prevalence of darker regions for the SK agents indicates that they follow a fixed route when completing the task that can be brittle if users try going the opposite way around the center block and force agents into locations they do not have experience acting in. We found that the seed in 4b is unable to play with seed 4c because they have learned different modes of the cooperative policy. In contrast, the IK agents have a much more uniform distribution over the cells they visit, meaning they will be more capable of adapting to users' actions should they be forced into a route they would not traditionally take.

5 Discussion

This paper has shown how learning cooperative strategies across many tasks with a single partner is a powerful method to generalize to many tasks with novel partners. Whereas prior work on ZSC has focused on increasing the entropy of *strategies* a learning agent is exposed to during training, we propose increasing the entropy of the *task space* as a means to coordinate across partners. When paired with methods to learn rich relational representations of scenes, we find this approach enables robust transfer across a wide variety of coordination problems and partners.

Our findings raise many questions, such as how much do agents which learn to coordinate on many tasks with a single partner benefit from additional exposure to diverse partner strategies during the course of learning? Future research will address these questions and evaluate the generalizability of IK agents compared to SOTA population-based methods when cooperating with humans.

References

[1] P. W. Battaglia, J. B. Hamrick, V. Bapst, A. Sanchez-Gonzalez, V. Zambaldi, M. Malinowski, A. Tacchetti, D. Raposo, A. Santoro, R. Faulkner, C. Gulcehre, F. Song, A. Ballard, J. Gilmer, G. Dahl, A. Vaswani, K. Allen, C. Nash, V. Langston, C. Dyer, N. Heess, D. Wierstra, P. Kohli, M. Botvinick, O. Vinyals, Y. Li, and R. Pascanu. Relational inductive biases, deep learning, and graph networks, 2018.

[2] C. Breazeal, C. Kidd, A. Thomaz, G. Hoffman, and M. Berlin. Effects of nonverbal communication on efficiency and robustness in human-robot teamwork. pages 708 – 713, 09 2005.

[3] N. Carion, G. Synnaeve, A. Lazaric, and N. Usunier. A structured prediction approach for generalization in cooperative multi-agent reinforcement learning, 2019.

[4] M. Carroll, R. Shah, M. K. Ho, T. L. Griffiths, S. A. Seshia, P. Abbeel, and A. Dragan. On the utility of learning about humans for human-ai coordination, 2020.

[5] C. Castelfranchi. The theory of social functions: challenges for computational social science and multi-agent learning. *Cognitive Systems Research*, 2(1):5–38, 2001.

[6] Y. Chen, C. Tang, R. Tian, C. Li, J. Li, M. Tomizuka, and W. Zhan. Quantifying agent interaction in multi-agent reinforcement learning for cost-efficient generalization, 2023.

[7] K. Cobbe, C. Hesse, J. Hilton, and J. Schulman. Leveraging procedural generation to benchmark reinforcement learning, 2020.

[8] K. Cobbe, O. Klimov, C. Hesse, T. Kim, and J. Schulman. Quantifying generalization in reinforcement learning. In K. Chaudhuri and R. Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 1282–1289. PMLR, 09–15 Jun 2019.

[9] M. Dennis, N. Jaques, E. Vinitsky, A. Bayen, S. Russell, A. Critch, and S. Levine. Emergent complexity and zero-shot transfer via unsupervised environment design, 2021.

[10] M. C. Fontaine, Y.-C. Hsu, Y. Zhang, B. Tjanaka, and S. Nikolaidis. On the importance of environments in human-robot coordination, 2021.

[11] S. S. Ghazimirsaeid, M. S. Jonban, M. W. Mudiyanselage, M. Marzband, J. L. R. Martinez, and A. Abusorrah. Multi-agent-based energy management of multiple grid-connected green buildings. *Journal of Building Engineering*, 74:106866, 2023.

[12] T. Guo, X. Chen, Y. Wang, R. Chang, S. Pei, N. V. Chawla, O. Wiest, and X. Zhang. Large language model based multi-agents: A survey of progress and challenges, 2024.

[13] H. Hu, A. Lerer, B. Cui, D. Wu, L. Pineda, N. Brown, and J. Foerster. Off-belief learning, 2021.

[14] M. Kleiman-Weiner, M. K. Ho, J. L. Austerweil, L. Michael L, and J. B. Tenenbaum. Coordinate to cooperate or compete: abstract goals and joint intentions in social interaction. In *Proceedings of the 38th Annual Conference of the Cognitive Science Society*, 2016.

[15] W. Li, P. Varakantham, and D. Li. Generalization through diversity: Improving unsupervised environment design. In E. Elkind, editor, *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI-23*, pages 5411–5419. International Joint Conferences on Artificial Intelligence Organization, 8 2023. Main Track.

[16] M. Ma, J. Liu, S. Sokota, M. Kleiman-Weiner, and J. N. Foerster. Learning intuitive policies using action features. In *International Conference on Machine Learning*, pages 23358–23372. PMLR, 2023.

[17] I. Mediratta, M. Jiang, J. Parker-Holder, M. Dennis, E. Vinitsky, and T. Rocktäschel. Stabilizing unsupervised environment design with a learned adversary. In S. Chandar, R. Pascanu, H. Sedghi, and D. Precup, editors, *Proceedings of The 2nd Conference on Lifelong Learning Agents*, volume 232 of *Proceedings of Machine Learning Research*, pages 270–291. PMLR, 22–25 Aug 2023.

[18] S. Poddar, Y. Wan, H. Ivison, A. Gupta, and N. Jaques. Personalizing reinforcement learning from human feedback with variational preference learning, 2024.

[19] B. Roche, J.-F. Guégan, and F. Bousquet. Multi-agent systems in epidemiology: a first step for computational biology in the study of vector-borne disease transmission. *BMC Bioinformatics*, 9(1):435, Oct. 2008.

[20] C. Ruhdorfer, M. Bortoletto, A. Penzkofer, and A. Bulling. The overcooked generalisation challenge. 2024.

[21] A. Rutherford, B. Ellis, M. Gallici, J. Cook, A. Lupu, G. Ingvarsson, T. Willi, A. Khan, C. S. de Witt, A. Souly, S. Bandyopadhyay, M. Samvelyan, M. Jiang, R. T. Lange, S. Whiteson, B. Lacerda, N. Hawes, T. Rocktaschel, C. Lu, and J. N. Foerster. Jaxmarl: Multi-agent rl environments in jax. *arXiv preprint arXiv:2311.10090*, 2023.

[22] M. Samvelyan, A. Khan, M. Dennis, M. Jiang, J. Parker-Holder, J. Foerster, R. Raileanu, and T. Rocktäschel. Maestro: Open-ended environment design for multi-agent reinforcement learning, 2023.

[23] B. Sarkar, A. Shih, and D. Sadigh. Diverse conventions for human-ai collaboration, 2023.

[24] T. B. Sheridan. Human–robot interaction: Status and challenges. *Human Factors*, 58(4):525–532, 2016. PMID: 27098262.

[25] M. Shum*, M. Kleiman-Weiner*, M. L. Littman, and J. B. Tenenbaum. Theory of minds: Understanding behavior in groups through inverse planning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 6163–6170, 2019.

[26] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. van den Driessche, T. Graepel, and D. Hassabis. Mastering the game of Go without human knowledge. *Nature*, 550(7676):354–359, Oct. 2017.

[27] D. Strouse, M. Kleiman-Weiner, J. Tenenbaum, M. Botvinick, and D. Schwab. Learning to share and hide intentions using information regularization. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

[28] D. Strouse, K. R. McKee, M. Botvinick, E. Hughes, and R. Everett. Collaborating with humans without human data, 2022.

[29] A. Tacchetti, H. F. Song, P. A. M. Mediano, V. Zambaldi, N. C. Rabinowitz, T. Graepel, M. Botvinick, and P. W. Battaglia. Relational forward models for multi-agent learning, 2018.

[30] M. Tomasello. *The cultural origins of human cognition.* The cultural origins of human cognition. Harvard University Press, Cambridge, MA, US, 1999. Pages: vi, 248.

[31] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio. Graph attention networks, 2018.

[32] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev, J. Oh, D. Horgan, M. Kroiss, I. Danihelka, A. Huang, L. Sifre, T. Cai, J. P. Agapiou, M. Jaderberg, A. S. Vezhnevets, R. Leblond, T. Pohlen, V. Dalibard, D. Budden, Y. Sulsky, J. Molloy, T. L. Paine, C. Gulcehre, Z. Wang, T. Pfaff, Y. Wu, R. Ring, D. Yogatama, D. Wünsch, K. McKinney, O. Smith, T. Schaul, T. Lillicrap, K. Kavukcuoglu, D. Hassabis, C. Apps, and D. Silver. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575(7782):350–354, Nov. 2019.

[33] S. A. Wu*, R. E. Wang*, J. A. Evans, J. B. Tenenbaum, D. C. Parkes, and M. Kleiman-Weiner. Too many cooks: Bayesian inference for coordinating multi-agent collaboration. *Topics in Cognitive Science*, 13(2):414–432, 2021.

[34] G.-S. Xia, X. Bai, J. Ding, Z. Zhu, S. Belongie, J. Luo, M. Datcu, M. Pelillo, and L. Zhang. Dota: A large-scale dataset for object detection in aerial images. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3974–3983, 2018.

[35] X. Yan, J. Guo, X. Lou, J. Wang, H. Zhang, and Y. Du. An efficient end-to-end training approach for zero-shot human-AI coordination. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[36] L. Ying, K. Jha, S. Aarya, J. B. Tenenbaum, A. Torralba, and T. Shu. Goma: Proactive embodied cooperative communication via goal-oriented mental alignment, 2024.

[37] R. Zhao, J. Song, Y. Yuan, H. Haifeng, Y. Gao, Y. Wu, Z. Sun, and Y. Wei. Maximum entropy population-based training for zero-shot human-ai coordination, 2022.

[38] Y. Zhou, J. Li, and J. Zhu. Posterior sampling for multi-agent reinforcement learning: solving extensive games with imperfect information. In *International Conference on Learning Representations*, 2020.

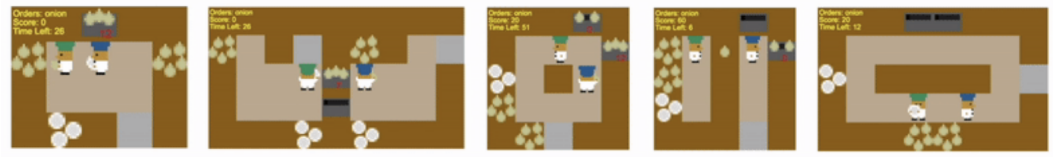


Figure 5: 5 original Overcooked layouts from [4]. From left to right, we have *Cramped Room*, *Asymmetric Advantages*, *Coordination Ring*, *Forced Coordination*, *Counter Circuit*.

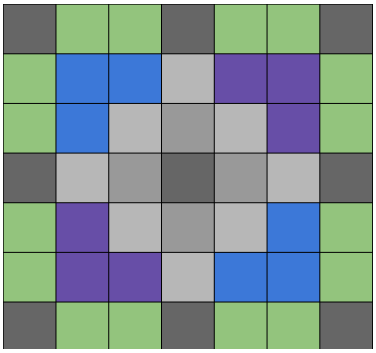


Figure 6: Color map of Procedural Overcooked Grid Generation. Green regions show reachable areas for delivery locations, pots, plates, and onions. Agents spawn in opposite blue or purple regions. Dark gray indicates potential wall locations that may block access to items. The grid shows all wall predicates (horizontal, vertical, middle block, no interior walls) superimposed, though only one type is sampled during actual grid generation.

A Appendix / supplemental material

A.1 Wall Generation Method

The generation process was carefully designed to ensure that every layout created is both solvable and presents significant coordination challenges from one or more of the 3 tasks. We begin by creating a continuous wall around the border of the grid, making the maximal space a player can move in be a 5×5 grid. We then have the option to generate a large, 3×3 block in the middle of this space with a probability p_m or a continuous wall stretching through the middle row or column of the grid with a probability p_c, or leaving the 5×5 grid empty with a probability p_e. If we generate a middle wall, each cell in the 3×3 middle block has the probability of being converted back into a free space with a probability p_p. If we chose to generate a continuous wall, we select f cells from the 5 cells that the wall is comprised with a probability of $\frac{1}{\binom{5}{f}} * (1 - p_f)$ of those f of cells being converted into a free space, where p_f is the probability of the wall remaining intact.

We note that this wall generation process leaves the potential for being unsolvable if items are naively sampled from all possible walls, since delivery locations could end up in the corners of the grid, on a wall in the middle of the grid that is inaccessible, or at T-junctions where the border wall connects with a wall going through the middle row or column of the grid (the dark gray regions in Figure 6). As such, we begin by first sampling a single delivery location, plate pile, pot, and onion pile to be on the border wall but not within the corners or potential T-junction locations (the green regions within Figure 6). We then spawn agents on opposite quadrants of the 5×5 grid, but make sure none of them are located on a wall position if the middle wall block, horizontal continuous wall, and vertical continuous wall are superimposed onto of each other (opposite blue regions or opposite purple regions in Figure 6). When taken with our previous wall generation scheme, this enables agents to complete the task independent of where the items apart of the recipe are generated. In the worst case where a wall divides both agents and all items are located on one side of the divider, a single agent will be able to achieve the task on their own. If the items are spread across both sides of the divider, then this is a coordination problem similar to the *Forced Coordination* Overcooked

layout. In all other cases, since a single agent is able to complete the task on its own, it is achievable by a pair of agents but still presents challenging navigation and coordination problems.

By structuring our generator to target the mentioned three sources of variation, and ensuring solvability, we introduce a number of different coordination challenges for future research on collaboration in Overcooked to address. A few of them are detailed in Figure 2.

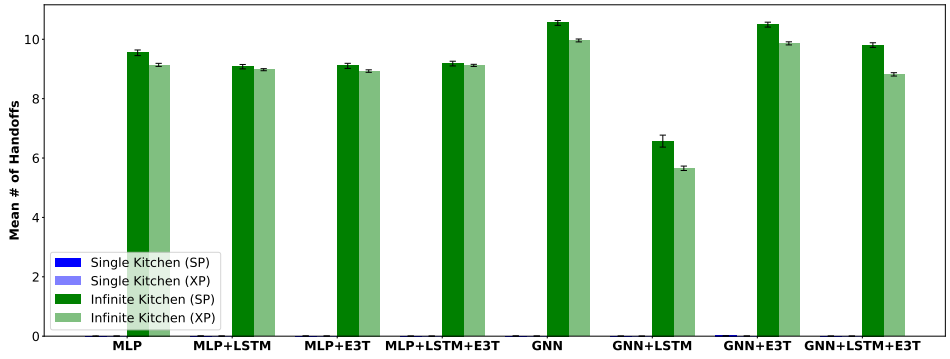
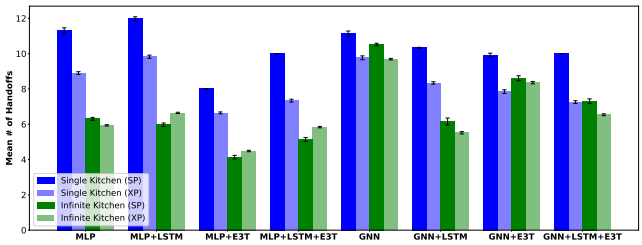


Figure 7: Comparison of agent performance on 100 held-out procedurally generated levels using different architectures and training regimes. The blue bars indicate Single Kitchen agent performance and the green bars indicate Infinite Kitchen agent performance. Hashed bars show performance from self play, while solid bars show performance in cross play.

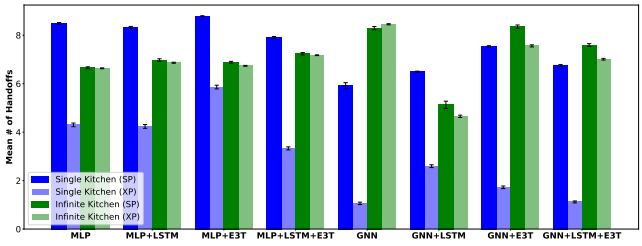
A.2 Additional Results

We demonstrate the cross-play performance of different seeds and architectures of models trained on our procedurally generated environment playing 100 novel levels together in Figure 7. Just as in Figure 3, we find that the GNNs support generalization to novel environments and novel partners, and that strategies learned for cooperation in self-play are flexible enough to transfer over into the cross-play setting, even with simple architectures such as an MLP.

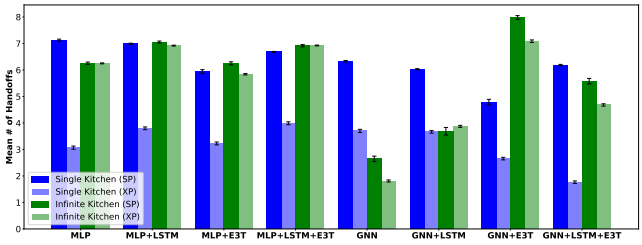
Moreover, we provide a breakdown of the cross-play performance between Single Kitchen and Infinite Kitchen methods on each of the five canonical levels (Asymmetric Advantages, Forced Coordination, Coordination Ring, Counter Circuit, and Cramped Room) in Figure 8. Again, we find that a single model training on a procedurally generated environment is able to cooperate with novel partners better than agents who were only exposed to one environment, with the added benefit of their learned coordination strategies being reusable when faced with novel challenges.



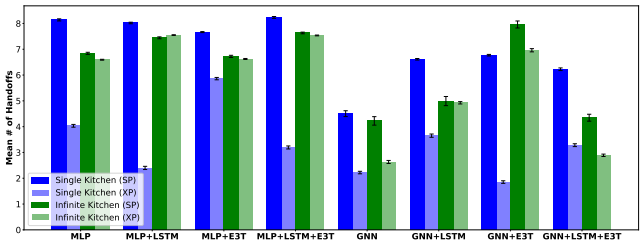
(a) *Asymmetric Advantages*



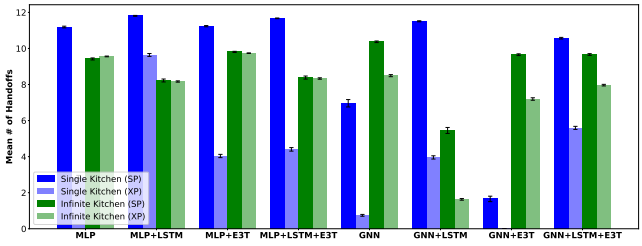
(b) *Coordination Ring*



(c) *Counter Circuit*



(d) *Cramped Room*



(e) *Forced Coordination*

Figure 8: Comparing Infinite Kitchen vs Single Kitchen across the 5 original Overcooked layouts. Infinite Kitchen is a single model we reevaluate across each layout, and Single Kitchen is when we retrain a new model for every layout. We observe that Infinite Kitchen typically produces better Cross-play performance despite never having seen the levels or partners it is tested with during training.