Evaluating Interdependent Collaboration in Multi-Agent LLM Systems

Rosanna Vitiello[1] **Yonatan Bisk** **Carolyn Rosé**[1]
[1] Language Technologies Institute, Carnegie Mellon University
{rvitiell, ybisk, cprose}@cs.cmu.edu

Abstract

Advances in LLMs offer hope of corresponding advances in agent participation in teamwork, while also posing new challenges in designing collaborative benchmarks for evaluating and integrating agents effectively into hybrid teams in real-world situations. While prior work has demonstrated that LLMs can operate in multi-agent settings, many evaluate on problems that do not require collaboration and can be solved with a single agent. Moreover, existing benchmarks often oversimplify collaboration on several dimensions by restricting evaluation to single episode tasks, homogeneous teams, and domains in which LLMs likely have extensive world knowledge. To bridge this gap, we propose a new collaborative multi-agent benchmark, Collaboration Rush, which includes interdependent tasks with multi-episode evaluation across three different domains. Additionally, we evaluate a multi-agent baseline on groups of varied LLM compositions. Our findings reveal that our evaluation paradigm exposes gaps in multi-agent LLM performace in collaborative tasks. By identifying these weaknesses, we motivate the need for future work in improving hybrid multi-agents systems for out-of-domain, multi-episode collaborative tasks.

1 Introduction

Human-human collaboration is challenging and has been demonstrated to benefit from automated support, such as agent-based support (Adamson et al., 2014; Naik et al., 2024). The language capabilities of state-of-the-art Large Language Models (LLMs) offer hope for advances in this space (Brown et al., 2020; Driess et al., 2023), but also raise new questions about how to design such agents and eventually introduce them into hybrid teams in real world scenarios. Simulation studies offer a means to generate synthetic data and to narrow down the space of designs, strategies, and behavior practices, or troubleshoot novel agent capabilities in highly controlled environments prior to running more realistic but costly user studies with human participants. This paper contributes a simulation study paradigm involving LLM agents that can play the role of collaborators or supporters of collaboration within these kinds of human user study design spaces.

In the same way that collaboration benefits humans, strong collaborative skills in multi-agent systems may enhance problem solving and decision making. Collaboration creates the opportunity for agents to learn from other output and increase the abilities of weaker models to complete tasks. In recent years, there has been increased interest in developing collaborative multi-agent systems using LLMs (Hong et al., 2023; Sun et al., 2023). However, while these works confirm multi-agent systems can improve reasoning performance, many only evaluate on tasks that do not necessitate collaboration, such as arithmetic reasoning benchmarks which could be solved by single agent systems. Consequently, these systems are not explicitly evaluated on interdependent collaborative group capabilities, but rather the collective ensemble of their individual contributions.

Additionally, existing interdependent collaboration benchmarks (Carroll et al., 2019; Gong et al., 2023; Zhang et al., 2024; Zhou et al., 2024) are limited in their ability to inform design work for agents that participate in hybrid teams because of their narrow scope of task contexts. In order for models to seamlessly collaborate with humans, they must learn how to adapt their support to different partner needs over time. In human-human collaboration, this process depends on the makeup of the teams and their familiarity with task context. For example, we expect that a new team with a range of individual capabilities coordinating on an unfamiliar task would collaborate successfully in distinct ways and have different needs compared to a veteran team of experts working on a long-standing project in their field. Prior benchmarks often oversimplify or fail to evaluate this collaborative process by restrict-

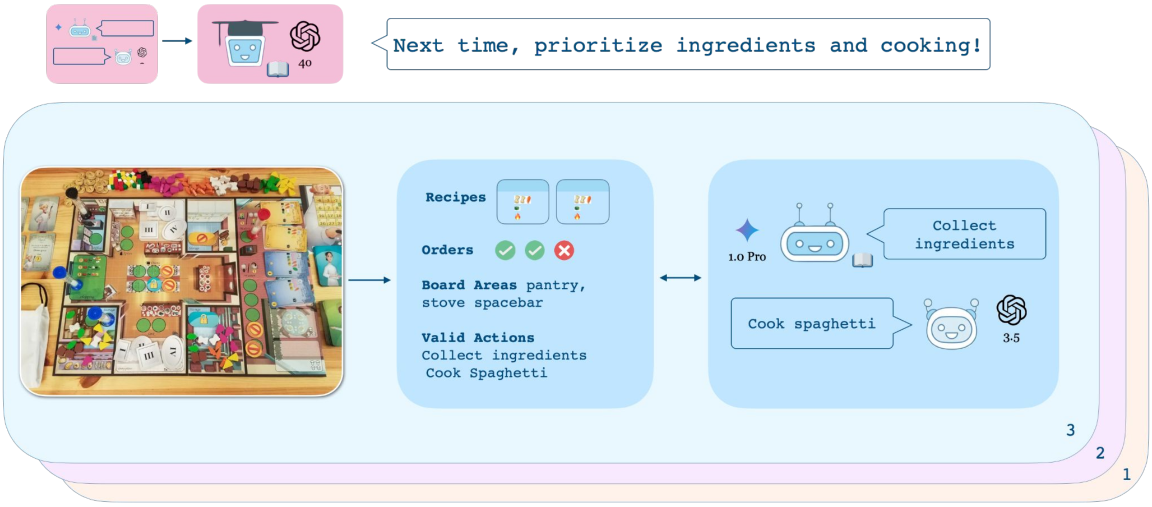

Figure 1: **Kitchen Rush**: A new evaluation benchmark for LLM collaboration over three episodes. In each episode, the game state is initialized and encoded into a textual observation. From observations with manual or teacher scaffolding, agents choose actions and update the game state.

ing evaluation to homogeneous teams of the same model, single episode tasks, and domains in which LLMs likely have extensive world knowledge (e.g., cooking (Carroll et al., 2019), coding (Hong et al., 2023). This motivates evaluation benchmarks for multi-agent LLM teams with varied group composition across multiple episodes and domains.

Our main contributions are a paradigm for advancing work in multi-agent systems and support for collaborative tasks:

- **Collaborative Benchmark** We propose a new multi-agent task, Collaboration Rush. In contrast to prior work, our task consists of multiple episodes and includes three task domains with which LLM agents would have varying world knowledge.
- **Multi-Agent Baseline Evaluation** We evaluate agents as collaborators and agents as supporters of collaboration through a teacher-feedback agent framework consisting of varied group LLM compositions.

Our findings reveal that LLM agents in this framework struggle to perform interdependent tasks across different domains and fail to consistently increase group performance over multiple episodes. Further, in teams with members of varying individual strength, stronger partners contribute to the the majority of performance – lacking the collaborative behavior necessary to succeed in tasks that require interdependent collaboration.

2 Related Work

Collaboration Benchmarks for LLMs range across multiplayer games (Carroll et al., 2019; Bard et al., 2019; Light et al., 2023; Qi et al., 2024; Gong et al., 2023), virtual embodied household tasks (Zhang et al., 2024; Guo et al., 2024), multi-robot collaboration (Mandi et al., 2023), and social scenarios (Zhou et al., 2024). These tasks often require pure collaboration with shared goals, such as cooking in OvercookedAI (Carroll et al., 2019), the investigation of Theory of Mind abilities in Hanabi (Bard et al., 2019), and coordination in virtual embodied environments (Zhang et al., 2024; Mandi et al., 2023). Other multi-agent benchmarks have mixed objectives in which groups compete (Light et al., 2023; Xu et al., 2024) or negotiate conflicting social goals (Zhou et al., 2024).

Multi-Agent LLM Systems have emerged as a popular approach to enhancing performance through collaboration and communication between models. These mutli-agent approaches have shown to increase benchmark performance by modularly breaking down complex tasks through assigned roles and team organization (Hong et al., 2023; Guo et al., 2024; Zhao et al., 2024), hierarchical reasoning (Liu et al., 2023), and subtask planning (Mandi et al., 2023). In a similar vein, other works in multi-agent debate have shown that negotiation between an ensemble of LLMs improves factuality (Khan et al., 2024; Du et al., 2023) and reasoning (Sun et al., 2023; Liang et al., 2023) over single model reasoning approaches.

While these works show multi-agent systems can improve reasoning performance, many only evaluate on tasks that do not require interdependent collaboration and can be solved with a single system. A growing area of interest in multi-agent systems lies not only in their ability to improve general reasoning, but also in their ability to simulate and improve the dynamics and interactions observed in human-human and human-AI teams (Naik et al., 2024; Ma et al., 2024), the nature of which may change depending on task characteristics and group composition. Thus, we extend prior literature by evaluating a multi-agent LLM system on interdependent collaboration tasks that requires more than one agent to successfully complete, and investigating how the performance of these systems changes across different collaborative dimensions, such as varied group composition and familiarity with task context.

3 Collaboration Rush (CRUSH)

We introduce a new evaluation benchmark, Collaboration Rush (CRUSH), inspired by the board game Kitchen Rush (Turczi and Bagiartakis, 2017). CRUSH contains a family of multi-agent coordination tasks which include three parallel versions in different domains: the original Kitchen Rush, Legal Rush, and Alien Rush. The representation of each CRUSH domain is instantiated in text prompts through an observation of the board state (Fig. 3) and manual (Fig. 4). To evaluate the extent to which implicit world knowledge of domain affects group performance on collaborative tasks, we use a word mapping to transfer Kitchen Rush prompts to parallel and equivalent versions, Legal and Alien Rush. We describe the original Kitchen Rush domain and detail its translation to parallel domains in the following sections.

3.1 Kitchen Rush

In Kitchen Rush, the goal of each episode is to complete all assigned recipes within the specified number of turns. Each turn, agents control a specific number of workers, each of which can be used to complete one high-level action per turn, such as taking ingredients, or cooking a meal on a stove. An episode of a scenario ends when all assigned recipes are either finished or ruined, or the group exceed the allotted number of turns for the episode.

Agents work in teams to complete each recipe (Fig 2). Every recipe requires (1) **ingredients** (pasta, carrots, lettuce, meat, or bread), (2) **spices** (white, black, green, red, and yellow), and (3) **cook time** to complete. If an agent is not careful or a group does not coordinate their actions properly, they may ruin an order (e.g., cooking a meal before collecting ingredients, exceeding the cook time of the order). Ruined orders cannot be completed.

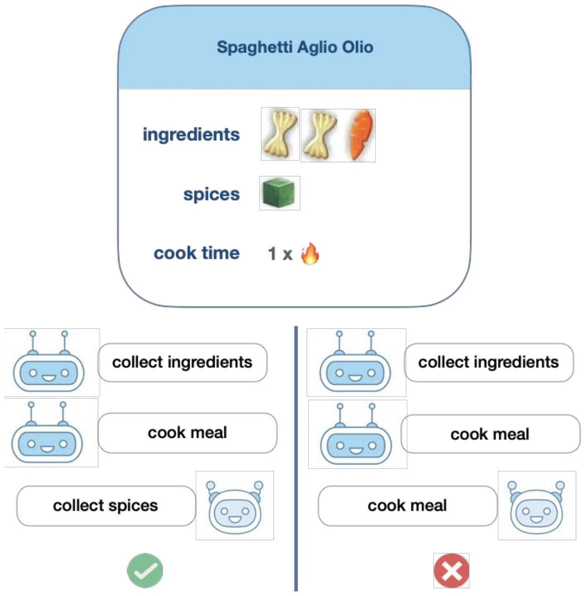

Figure 2: Example of a Kitchen Rush Recipe specifying ingredients, spices, and length of cook time. Agents must coordinate their actions in a valid order to complete the recipe (left). Otherwise, agents may ruin the recipe, such as cooking the meal too many times (right).

In addition to coordinating actions to complete dishes, groups must also appropriately share resources. Every action must be carried out in a specific location of the kitchen (e.g., ingredients from the **pantry**, spices from the **spicebag**, and orders are cooked on the **stoves**). Locations have a limited number of action spaces, restricting how many of the same action can be completed every turn. For example, a kitchen with only two stoves means there can only be two cook actions every turn. In this case, if the recipes require long cook times, then it would be imperative that agents share stoves carefully and plan to cook earlier than later.

3.2 Parallel Domains: Alien & Legal Rush

We additionally evaluate on parallel versions of Kitchen Rush: Legal Rush and Alien Rush. Construction of parallel versions involves replacing all cooking related entities with legal entities and pseudoword alien entities for Legal and Alien Rush respectively. In each Kitchen Rush prompt, we manually identify all words that refer to cooking. This

```
Recipes:
Fettucine Alfredo:
    ingredients: {'pasta': 2, 'cheese': 1, 'meat': 2}
    spices: {'white': 1, 'black': 1}
    cook time: 1

Orders:
Fettucine Alfredo:
    ingredients: collected
    spices: collected
    cook time: 1
    finished: True
    ruined: False

Board Areas:
pantry:
    area_type: pantry
    area_id: pantry
    ingredients: {'meat': 16, 'carrot': 20, 'lettuce': 20,
'bread': 20, 'pasta': 17, 'cheese': 17}
    available_action_spaces: 4

Valid Actions:
{'action type': 'TAKE INGREDIENTS', 'area id': 'pantry1',
```

Figure 3: Abbreviated Kitchen Rush observation

```
The game Kitchen Rush has the following rules:
- The goal of the game is to finish all the orders. You
score points for finishing orders and lose points for
ruining orders.
- Players work together to finish orders. To finish an order
properly, players must follow its corresponding recipe.
Orders are finished when its ingredients, spices and cook
time matches the corresponding recipe.
- There are 6 kinds of ingredients (meat, carrot, lettuce,
bread, pasta, and cheese) and 5 kinds of spices (green,
black, white, yellow, and red).

***Actions***
1. Take Ingredients: Collect ingredients for an order from
the specified pantry. Removes one available action space
from pantry.

*** The game ends when: ***
- All the orders are either finished or ruined

These conventions will help when playing the game:
1. ***Take Ingredients before Cooking Order***
- Only cook an order if the order already has its
ingredients to avoid ruined orders
```

Figure 4: Abbreviated Kitchen Rush manual

includes cooking related terms such as ingredients, spices, recipe names, and stems of cooking-related verbs. Using a lexical mapping dictionary, we translate Kitchen Rush words to Alien pseudowords or legal themed Legal words. The same one-to-one mapping is used to translate all Kitchen Rush text, such as manuals about the game. Consequently, all three domains are functionally equivalent tasks; however, agents decreasingly benefit from their implicit world knowledge of the task in the Kitchen, Legal, Alien domains. We manually verified that all verbs, tenses and nouns of each prompt is translated property. Appendix A contains all prompts and translations.

3.3 Multi-Episode Scenarios

Our environment allows the customization of scenarios by changing the initial game state represented in the observation prompt. Each scenario's recipes and resources are easily modifiable. Similar to work by Carroll et al. (2019), this flexibility enables design of scenarios that can target evaluation of game understanding versus high-level coordination strategies. This allows evaluation that can pinpoint why models fail, such as due to lack of understanding of game rules or inability to interdependently collaborate

To this end, we evaluate on two scenarios: an easy game knowledge scenario and a hard interdependent scenario. Each scenario consists of three episodes. The goal of each episode is to complete five assigned recipes within ten turns. Within the same difficulty scenario, each episode is strategically equivalent with the same initial state of resources and recipe composition but relevant entities are swapped, and initial resources are adjusted such that the same challenges occur in each episode, such as ensuring one resource is severely limited. This prevents success across episodes that arises from memorizing recipes but maintains the same difficulty and flavor of coordination challenges.

Easy Scenario evaluates an agent's understanding of the game. This scenario is not interdependent and can be completed by one agent within the allotted turns. There are no resource conflicts, so agents have unlimited ingredients, spices, and actions spaces at each location. Groups that can complete all five recipes signal that they have mastery of the basic task completion. However, groups that ruin recipes in this scenario suggest they lack the ability to play the game at its simplest level.

Hard Scenario evaluates group ability to interdependently coordinate resources and strategies. The pantry and stoves are severely limited, such that all five recipes can only be completed by the final turn. Success requires agents to divide tasks between collecting ingredients and cooking recipes. Without task division, groups cannot complete all five recipes within the allotted time. Thus, groups that complete less than five recipes, ruin recipes, or exceed the allotted time of the episode may indicate that they did not effectively collaborate.

4 Methods and Experimental Design

The success of a collaboration depends on group composition, their familiarity with the task context, and their ability to dynamically adapt to partner skills and group strategies over time. As we aim to better understand how these factors effect the performance of multi-agent teams, we incorporate these dimensions in our experimental design to address the following questions: (1) how does prior world knowledge or explicit task knowledge of the task domain affect multi-agent group collaboration? (2) to what extent does group composition affect the nature of multi-agent group performance? and (3) to what extent can multi-agent groups utilize feedback from prior interactions to improve future interactions?

4.1 LLM Agent Prompt Variations

As simple baselines, we implement an LLM agent similar to those in prior LLM agent tasks (Yao et al., 2022; Agashe et al., 2024). In this approach, LLMs are given a prompt which contains state obtained from the game environment during the agent's turn and a list of valid candidate actions (`obs`). Along with observations, additional explicit knowledge about the game (`manual`) or feedback from prior rounds (`group` or `roles` teacher feedback) may be appended to the prompt. With this textual information, agents pick the next best action from the candidate actions to update the environment. We include full details of prompt implementations in Appendix A and briefly describe the different prompt baselines below:

Observation only (`obs`) Every worker turn, we provide agents with a text observation of the board state. (e.g., recipes, order status, resource availability). Agents are prompted to choose the next action within a list of valid candidate actions. In this condition, we examine the extent to which implicit world knowledge about the domain and the task structure helps model performance.

Manual (`+manual`) Along with observations, we provide agents with a manual of the game rules. Agents may use provided explicit knowledge of the game to choose an action from the observation. By controlling whether agents have access to a manual, we investigate how explicit task knowledge may affect group performance.

Episodic Teacher Feedback (`+group` or `+roles`) To investigate the extent to which LLM agents in this context can utilize feedback from prior interaction to improve future interactions, we implement a teacher agent to provide relevant plans and strategies to agents from prior episode history. Similar to the player prompt, in order to generate feedback, we provide the teacher with a game manual as well as action history and results from the previous episode. From this information, we prompt teachers to generate new strategies for the next episode or adjust prior strategies given in the previous episode. Note that in the first episode of a scenario, no teacher feedback is given because there is no previous episode history to generate feedback. We evaluate two teacher prompts: (1) `group strategy` prompt that gives the same feedback and strategies to every agent, and (2) an `individual roles` feedback prompt that gives individualized roles and feedback to each player.

4.2 Experimental Design

Task We evaluate multi-agent pairs on our collaboration benchmark. We run each experimental condition three times on each scenario, easy and hard, described in Section 3.3. Each scenario contains three episodes which groups play through sequentially. The order of the versions of each episode are balanced amongst the three runs with a Latin square to mitigate ordering effects.

Experimental Variables We manipulate the following experimental variables in each simulation and analyze their interactions:

Group Composition. To manipulate the extent to which group composition effects performance, we experiment with the LLM compositions of each group pairing. We vary along five models across three LLM families. We refer to pairs with the same LLM backbone as **homogeneous** and pairs with different LLM backbones as **heterogeneous**. We categorize larger LLM models as **strong** (`GPT-4o` and `Gemini 1.5 Flash`) and smaller models as **weak** (`GPT-3.5-Turbo`, `Gemini 1.0 Pro`, `llama-3-8b-Instruct`).

Explicit Task Knowledge. We vary explicit task knowledge by controlling the presence of a game manual in prompts (`obs` or `obs+manual`).

Domain. We evaluate across domains to assess the impact on groups of implicit world knowledge (`KRush`, `LRush`, and `ARush`).

Episodic Feedback. We evaluate each group with varying teacher feedback conditions: no teacher, individual `roles` teacher, or `group` strategy feedback from a strong model.

Metrics We evaluate each simulation on group and individual metrics in Table 1.

Group Recipes/Turns		
Completed	(CR)	# completed recipes by group (i.e. score)
Ruined	(RR)	# ruined recipes completed by group
Turns		# of turns in the episode
Individual Actions		
Positive	(A^+)	Action contributed to completing recipes
Negative	(A^-)	Action contributed to ruining recipes
Neutral	$(A^=)$	Neutral action (e.g. no action) by an agent
Total		$A^+ + A^- + A^=$

Table 1: Evaluation Metrics

5 Results and Discussion

To conduct a thorough analysis considering effects of all experimental variables and their interactions, we conduct an ANOVA (Analysis of Variance), a statistical method used to evaluate which independent variables significantly influence dependent performance metrics. We define our independent variables as those mentioned in Section 4.2, and the dependent variable as number of completed recipes, which we refer to as score. We also investigate two-way and three-way interactions between experimental variables. A multi-variate ANOVA with interaction terms allows investigation of interaction dependencies while carefully controlling for the added risk of a Type I error in the face of multiple comparisons. We observe that the model indicates certain effects depend on other experimental variables, as detailed in the full ANOVA results provided in Appendix B. We present a summary of the key findings below.

5.1 Domain Findings

First, we analyze effects of domain across different pairings. In doing so, we investigate how prior world knowledge or explicit task knowledge of the domain affects multi-agent group collaboration.

Domain affects LLM group task performance. We find a significant main effect of domain on task performance: $F(2, 7960) = 340.17$, $p < .0001$. A student-t post-hoc analysis reveals that groups perform significantly better on tasks in which they have more implicit world knowledge, performing best to worst in KRush, LRush and ARush. This trend holds across strong-strong, strong-weak, and weak-weak `obs` group performance in the easy scenario across domains (Fig 5). Overall, strong-strong groups perform best and weak-weak groups perform worse. The high performance by strong models in the easy task indicates that they have understanding of basic game rules (≥ 4 recipes completed) while the low performance by weak models suggest they do not (≤ 2 recipes completed). However, even in the easy scenario, strong-strong groups cannot achieve basic task performance in non-KRush settings without a manual, in which they only complete 1-3 recipes.

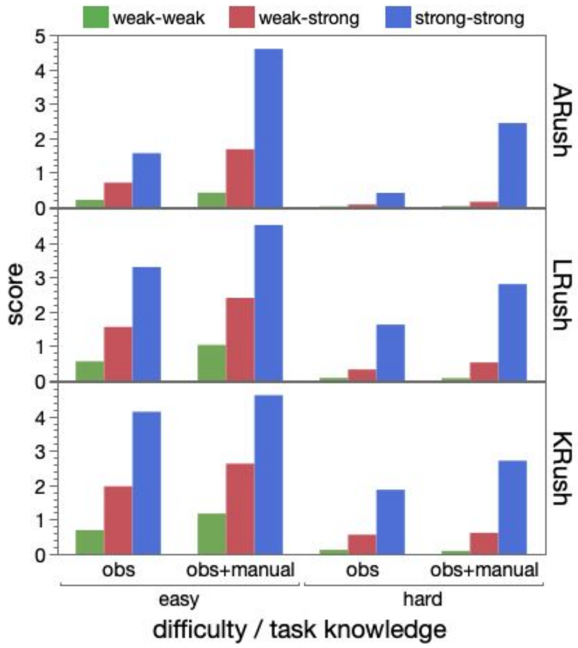

Figure 5: Average completed recipes (score) across domain and LLM strength composition. We compare results in simulations run with and without a manual in the easy and hard condition.

Manuals address domain gaps. Presence of explicit task knowledge in the `obs+manual` condition enables LLM groups to achieve similar task performance across domains. This interaction is most evident in strong-strong groups, who reduce performance gaps when a manual is present across all three domains in the easy and hard scenarios. Similar trends can be seen in weak-strong groups which achieve equivalent performance with a manual across LRush and KRush in both difficulties but perform worse in the most foreign domain, ARush. Weak-weak groups benefit from a manual across LRush and KRush in the easy scenario but perform too poorly in the hard scenario to conclude a meaningful comparison. These results indicate providing explicit task knowledge through prompts is effective to reduce task-performance gaps across domains for LLMs with proficient context length and model size.

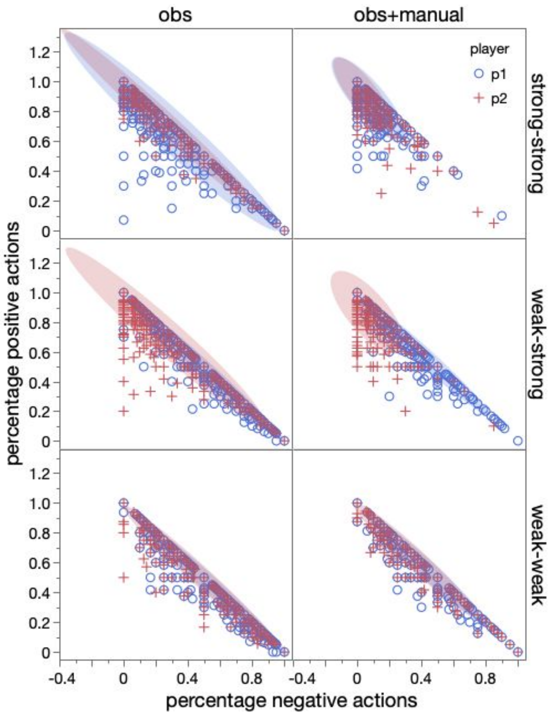

Figure 6: Positive to negative action distribution comparison of p1-p2 partners with varying individual strengths

LLMs fail to collaborate when interdependent. This is evident as no group averages more than three completed recipes in the hard task across any domain. In the hard scenario, all five recipes can only be completed by the tenth and final turn. Groups that complete less than five recipes either (1) could not complete five recipes by the last turn or (2) ruined all five recipes and terminated the episode early. Groups that time out have reasonable task performance, but did not interdependently collaborate with their partner to efficiently complete the task before ten turns. Groups that terminate prematurely are not individually skilled enough to complete the hard scenario.

We investigate the nature of strong-strong group interactions in the hard scenario. When examining the performance of GPT-4o pairs with a manual in the hard scenario (Table 2), the majority of simulation runs ruin few recipes which indicates capable individual task performance. However, they fail to complete more than four recipes before the turn limit because they cannot interdependently coordinate with a partner. Thus, manual prompting is insufficient to grant LLMs the ability to collaborate in this task. This motivates future work on approaches to improve multi-agent groups in interdependent settings.

Turns		**< 10**	**= 10**					
					CR			**Ave**
Model	**Domain**		**1**	**2**	**3**	**4**	**5**	**RR**
GPT-4o	Kitchen	7	0	9	27	58	0	0.4
	Courtroom	9	0	4	33	53	0	0.5
	Alien	11	9	13	36	31	0	0.8

Table 2: Percentage of GPT-4o groups with a manual that terminated early (<10) or reached turn limit (=10) and the average ruined recipes (Ave RR) across runs that reached turn limit. For groups that reached turn limit, percentages are broken down by the number of completed recipes (CR) by the limit.

5.2 Group Composition Findings

We examine how group composition, in terms of homogeneity and strength, changes the nature of collaboration and performance on the task.

Groups need a strong model. There is a significant main effect of group composition on group performance: $F(4, 7960) = 1855.35$, $p < .0001$. A student-t post-hoc analysis indicates that the relative strengths of each partner significantly affects their ability to complete recipes with strong-strong, weak-strong, and weak-weak groups performing best to worst, respectively. The post-hoc analysis also reveals that there is no significant difference between homogeneous and heterogeneous strong-strong groups; however, homogeneous weak-weak groups perform significantly better than heterogeneous weak-weak groups.

Weaker models benefit from stronger partners through unbalanced individual contributions. When group composition varies by individual strength, an often desired characteristic of collaboration is the ability of stronger partners to improve the performance of their weaker counterparts. In contrast, stronger pairings may outperform weaker ones simply due to superior individual skills, rather than enhancement of each other's performance at the group level. In interdependent tasks, the former is more desired than the latter because a single agent has limited individual ability to improve task performance and must rely on the competencies of the entire group to complete tasks.

Significant performance gains relative to model strength are explained by stronger individual contribution rather than enhanced group collaboration. We examine the distributions of positive and negative actions across players in different group compositions (Fig. 6). We observe that

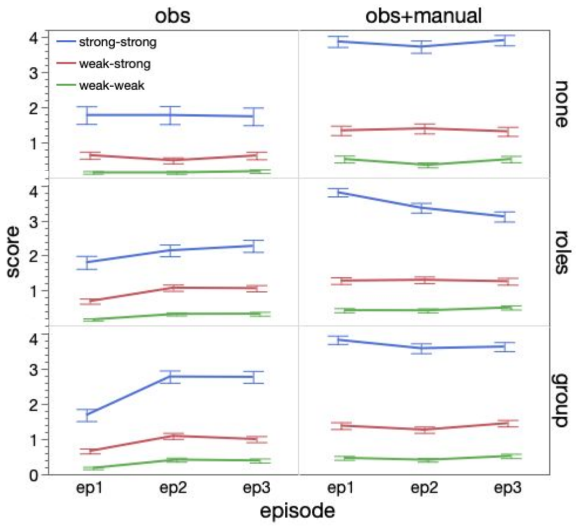

Figure 7: Average score on each episode across groups of varied individual strengths (strong-strong, weak-strong, weak-weak), task knowledge (obs, obs+manual), and feedback (none, roles, group).

the distributions of contribution in strong-strong groups and weak-weak groups overlap. That is, partners of similar abilities individually contribute at same rate. Furthermore, when groups of the same strength are given a manual, their distributions improve equally. Despite improvement, we argue that this performance increase is still explained entirely by stronger individual contribution rather than enhanced group collaboration because they do not successfully complete the interdependent hard scenario.

In weak-strong groups, the distribution of partner contribution overlaps less. Strong models tend to contribute to more positive actions and less negative actions. Inversely, weak models contribute to more negative actions and less positive actions. Moreover, when these groups are given a manual, only the strong model distribution benefits. We similarly conclude that increased performance in weak-strong groups is not explained by stronger models increasing weaker model contribution, but rather by unbalanced individual contributions.

5.3 Multi-Episode and Teacher Findings

Lastly, we evaluate the extent to which LLM agents in this context can utilize teacher feedback to improve interactions over time.

Feedback improves performance but plateaus. There is a significant main effect of teacher feedback on group performance: $F(2, 7960) = 17.36$, $p < .0001$. Student-t post-hoc analysis indicates that the presence of a group teacher is better than no teacher, but there is no significant difference between the roles teacher and no teacher. We find these trends are consistent across episodes which indicate that teacher feedback significantly improves performance in the second episode after which performance plateaus in the third episode. In contrast, without a teacher, static prompts in the obs and obs+manual baseline do not show any significant difference across episodes (Fig 7).

Feedback is less effective and inconsistent. Manuals are more effective than teacher feedback in improving overall performance. When a manual is present, groups with a teacher do not perform significantly better than groups with no teacher. In fact, results show that groups with a manual perform worse with a roles teacher than with none. We also note that the effectiveness of feedback is inconsistent across group strengths. Similar to trends with manuals, weak-weak groups benefit less from feedback than weak-strong and strong-strong groups. While manuals do not improve task performance over multiple episodes, they do contribute to the majority of performance gains across and teams and domains. We hypothesize that LLMs benefit more from manuals than high-level teacher feedback because they are better aligned to their instructional pretraining. This motivates future work in developing dynamic approaches which can improve multi-agent LLM task performance over multiple episodes.

6 Conclusion

We introduce a new family of collaboration tasks, CRUSH. Despite prior success of multi-agent systems, our paradigm exposes gaps that suggest prior work misses key factors in multi-agent evaluation. In particular, we show that current LLMs in our framework fail to consistently increase group performance over multiple episodes, particularly for teams of mixed strengths in interdependent tasks. These failures must be addressed to better understand and improve human-AI collaboration. That is, to support the dynamics and interactions of human teams, it is critical that LLM agents adapt their support to different partner needs over time. This work not only underscores the necessity of integrating these key factors into collaborative evaluations, but also sets the stage for future research aimed at developing adaptive multi-agent approaches that strengthen long-term human-AI collaboration.

7 Limitations

While this work contributes and expands on prior literature in collaborative multi-agent tasks for LLMs, it has several limitations.

Limitations of Simulation We limit our current benchmark in several ways that reduce the scope of our evaluation. Firstly, we restrict our task to text-only simulation. In reality, collaboration is a complex, multimodal interaction that cannot be captured solely through written communication. Effective collaboration includes verbal and non-verbal cues, such as body language, tonality of speech, and joint attention, all of which ground communication and coordination. Secondly, our task is constrained to a fixed turn-based interaction, which cannot capture aspects of multi-agent tasks such as interjection and simultaneous coordination. And lastly, our simulation does not include human interaction, which limits our benchmark's ability to transfer findings to human collaboration applications. While we did not implement other modalities in the scope of this work, we choose to base our simulation on a real board game, Kitchen Rush, to allow for future work on this dimension.

Lack of Finetuning We do not evaluate the effect of fine-tuning in our task. While we do assess the foundational capability of LLMs which is necessary in evaluating the implicit knowledge of models in this task, this approach limits our understanding of an LLM's potential performance with training.

References

David Adamson, Gregory Dyke, Hyeju Jang, and Carolyn Penstein Rosé. 2014. Towards an agile approach to adapting dynamic collaboration support to student needs. *International Journal of Artificial Intelligence in Education*, 24:92–124.

Saaket Agashe, Yue Fan, Anthony Reyna, and Xin Eric Wang. 2024. Llm-coordination: Evaluating and analyzing multi-agent coordination abilities in large language models. *Preprint*, arXiv:2310.03903.

Nolan Bard, Jakob N. Foerster, A. P. Sarath Chandar, Neil Burch, Marc Lanctot, H. Francis Song, Emilio Parisotto, Vincent Dumoulin, Subhodeep Moitra, Edward Hughes, Iain Dunning, Shibl Mourad, H. Larochelle, Marc G. Bellemare, and Michael H. Bowling. 2019. The hanabi challenge: A new frontier for ai research. *ArXiv*, abs/1902.00506.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeff Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *ArXiv*, abs/2005.14165.

Micah Carroll, Rohin Shah, Mark K. Ho, Thomas L. Griffiths, Sanjit A. Seshia, Pieter Abbeel, and Anca Dragan. 2019. *On the utility of learning about humans for human-AI coordination*. Curran Associates Inc., Red Hook, NY, USA.

Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, Yevgen Chebotar, Pierre Sermanet, Daniel Duckworth, Sergey Levine, Vincent Vanhoucke, Karol Hausman, Marc Toussaint, Klaus Greff, Andy Zeng, Igor Mordatch, and Pete Florence. 2023. Palm-e: An embodied multimodal language model. In *arXiv preprint arXiv:2303.03378*.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B Tenenbaum, and Igor Mordatch. 2023. Improving factuality and reasoning in language models through multiagent debate. *arXiv preprint arXiv:2305.14325*.

Ran Gong, Qiuyuan Huang, Xiaojian Ma, Hoi Vo, Zane Durante, Yusuke Noda, Zilong Zheng, Song-Chun Zhu, Demetri Terzopoulos, Li Fei-Fei, and Jianfeng Gao. 2023. Mindagent: Emergent gaming interaction. *Preprint*, arXiv:2309.09971.

Xudong Guo, Kaixuan Huang, Jiale Liu, Wenhui Fan, Natalia Vélez, Qingyun Wu, Huazheng Wang, Thomas L Griffiths, and Mengdi Wang. 2024. Embodied llm agents learn to cooperate in organized teams. *arXiv preprint arXiv:2403.12482*.

Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Ceyao Zhang, Jinlin Wang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, Chenglin Wu, and Jürgen Schmidhuber. 2023. Metagpt: Meta programming for a multi-agent collaborative framework. *Preprint*, arXiv:2308.00352.

Akbir Khan, John Hughes, Dan Valentine, Laura Ruis, Kshitij Sachan, Ansh Radhakrishnan, Edward Grefenstette, Samuel R. Bowman, Tim Rocktäschel, and Ethan Perez. 2024. Debating with more persuasive llms leads to more truthful answers. *Preprint*, arXiv:2402.06782.

Tian Liang, Zhiwei He, Wenxiang Jiao, Xing Wang, Yan Wang, Rui Wang, Yujiu Yang, Zhaopeng Tu, and Shuming Shi. 2023. Encouraging divergent thinking in large language models through multi-agent debate. *ArXiv*, abs/2305.19118.

Jonathan Light, Min Cai, Sheng Shen, and Ziniu Hu. 2023. Avalonbench: Evaluating LLMs playing the game of avalon. In *NeurIPS 2023 Foundation Models for Decision Making Workshop*.

Jijia Liu, Chao Yu, Jiaxuan Gao, Yuqing Xie, Qingmin Liao, Yi Wu, and Yu Wang. 2023. Llm-powered hierarchical language agent for real-time human-ai coordination. *Preprint*, arXiv:2312.15224.

Qianou Ma, Hua Shen, Kenneth Koedinger, and Tongshuang Wu. 2024. How to teach programming in the ai era? using llms as a teachable agent for debugging. *Preprint*, arXiv:2310.05292.

Zhao Mandi, Shreeya Jain, and Shuran Song. 2023. Roco: Dialectic multi-robot collaboration with large language models. *Preprint*, arXiv:2307.04738.

Atharva Naik, Jessica Ruhan Yin, Anusha Kamath, Qianou Ma, Sherry Tongshuang Wu, Charles Murray, Christopher Bogart, Majd Sakr, and Carolyn P. Rose. 2024. Generating situated reflection triggers about alternative solution paths: A case study of generative ai for computer-supported collaborative learning. *Preprint*, arXiv:2404.18262.

Siyuan Qi, Shuo Chen, Yexin Li, Xiangyu Kong, Junqi Wang, Bangcheng Yang, Pring Wong, Yifan Zhong, Xiaoyuan Zhang, Zhaowei Zhang, Nian Liu, Wei Wang, Yaodong Yang, and Song-Chun Zhu. 2024. Civrealm: A learning and reasoning odyssey in civilization for decision-making agents. *Preprint*, arXiv:2401.10568.

Qiushi Sun, Zhangyue Yin, Xiang Li, Zhiyong Wu, Xipeng Qiu, and Lingpeng Kong. 2023. Corex: Pushing the boundaries of complex reasoning through multi-model collaboration. *Preprint*, arXiv:2310.00280.

David Turczi and Vangelis Bagiartakis. 2017. Kitchen rush. Board game. Illustrated by Gong Studios.

Yuzhuang Xu, Shuo Wang, Peng Li, Fuwen Luo, Xiaolong Wang, Weidong Liu, and Yang Liu. 2024. Exploring large language models for communication games: An empirical study on werewolf. *Preprint*, arXiv:2309.04658.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2022. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*.

Hongxin Zhang, Weihua Du, Jiaming Shan, Qinhong Zhou, Yilun Du, Joshua B Tenenbaum, Tianmin Shu, and Chuang Gan. 2024. Building cooperative embodied agents modularly with large language models. *ICLR*.

Zhonghan Zhao, Kewei Chen, Dongxu Guo, Wenhao Chai, Tian Ye, Yanting Zhang, and Gaoang Wang. 2024. Hierarchical auto-organizing system for open-ended multi-agent navigation. *Preprint*, arXiv:2403.08282.

Xuhui Zhou, Hao Zhu, Leena Mathur, Ruohong Zhang, Haofei Yu, Zhengyang Qi, Louis-Philippe Morency, Yonatan Bisk, Daniel Fried, Graham Neubig, and Maarten Sap. 2024. SOTOPIA: Interactive evaluation for social intelligence in language agents. In *The Twelfth International Conference on Learning Representations*.

A LLM Agent Prompts

Player Prompts We provide detailed prompts for the LLM agents as both player agents and teacher agents. LLM players are provided with A.1 to generate an action for a worker. Depending on the condition, different parts of the prompt may be included or excluded. Specifically the `[manual]` is only included in the `obs+manual` condition and teacher feedback is only included during episode 2 and 3 in the `group` and `roles` teacher condition. After being provided an observation of the game board state, the agent generates an action from the valid action list provided in the observation.

Teacher Prompts To generate teacher feedback, LLM teachers are provided with A.2 in the `group` teacher condition and A.3 in the `roles` teacher condition after an episode is completed. Unlike the player prompt, to generate their feedback with proper task context, the prompt always includes a manual. The prompt additionally includes history from the prior round which contains the initial state of the round, the action history of the agents, and the final score. Examples of generated teacher feedback from our results can be found in A.10 and A.11.

Manuals and Observations In parallel domains, we pass each prompt through a dictionary mapping to change all cooking related entities to legal theme entities in Legal Rush and pseudowords in Alien Rush. To demonstrate this translation, we provide manuals and sample observations of each domain.

- KRush manual (A.4) & observation (A.7)
- LRush manual (A.5) & observation (A.8)
- ARush manual (A.6) & observation (A.9)

A.1 Player Prompt

System Prompt: You are a helpful assistant.
User Prompt:

`[manual]`

I am `[player]`. I am playing `[domain]` Rush Board Game with a partner. At each timestep, I will provide you with the relevant information of the game. I will also provide you with the legal action. Help me select the best next action. Format your response as Explanation: <brief explanation for selecting the move> Action:<selected move>. The <selected move> must be an integer. Do not say anything else. Got it?
Assistant Prompt: Got it.
User Prompt: It is my (`[player]`) turn. Use the following advice to reason and select the best move:

`[teacher feedback]`
`[observation]`

A.2 Group Teacher Prompt

System Prompt: You are a helpful assistant.
User Prompt:

`[manual]`

You are a teacher helping a group play `[domain]` rush. The group that you are helping completed a round of the game. I will provide you with relevant information from the last round, such as the history of actions played and the score of each round. Your goal is to help them create strategies to improve their collaboration for the next round using this history. If they already have strategies, you may keep them the same, add or alter them if there are other strategies needed due to the previous round results. Format your response as Explanation: <brief explanation for strategies> Strategies:<list of strategies>. Do not say anything else. Got it?
Assistant Prompt: Got it.
User Prompt:

`[prior round history]`

A.3 Roles Teacher Prompt

System Prompt: You are a helpful assistant.
User Prompt:

[manual]

You are a teacher helping a group play [domain] Rush. The group that you are helping completed a round of the game. I will provide you with relevant information from the last round, such as the history of actions played, the score of each round. Your goal is to assign them roles to help them improve performance. If they already have roles, you may keep them the same or alter them if you think it will improve group coordination. While writing roles, think step-by-step:

1. From the action history and results, summarize the behaviors, tendencies and mistakes of each player
2. Write roles and feedback for each player

Format your response as Explanation: <brief explanation for roles> [p1]: <list of roles & feedback> [p2]: <list of roles & feedback>
Assistant Prompt: Got it.
User Prompt:

[prior round history]

A.4 Kitchen Rush Manual

The game Kitchen Rush has the following rules:

- The goal of the game is to finish all the orders. You score points for finishing orders and lose points for ruining orders.
- Players work together to finish orders. To finish an order properly, players must follow its corresponding recipe. Orders are finished when its ingredients, spices and cook time matches the corresponding recipe.
- There are 6 kinds of ingredients (meat, carrot, lettuce, bread, pasta, and cheese) and 5 kinds of spices (green, black, white, yellow, and red).
- If an order is cooked before its ingredients are collected, it is ruined. If an order is ruined, it cannot be finished.
- There are 3 types of board areas: pantries, spicebags, and stoves. Each board area has a limited number of action spaces. If there are no available action spaces, the board area cannot be used. Action spaces reset to available after a turn is finished.
- Players have two workers which can carry out one action per turn.
- Players can only choose action for their workers from the Valid Actions

Actions

1. Take Ingredients: Collect ingredients for an order from the specified pantry. Removes one available action space from pantry.
2. Take Spices: Collect spices for an order from specified spicebag. Removes one available action space from spicebag.
3. Cook Order: Cook a meal by one cooking time on specified stove. Removes one available action space from stove.

The game ends when:

- All the orders are either finished or ruined
- You have played all 11 turns of the game.

These conventions will help when playing the game:

1. Take Ingredients before Cooking Order - Only cook an order if the order already has its ingredients to avoid ruined orders
2. Do Not Work on Ruined or Finished Orders - If an order is ruined or finished, you do not need to work on it. Do not do actions on ruined orders or finished orders
3. Avoid Ruining Orders - Do not cook an order before its ingredients are collected.

A.5 Legal Rush Manual

The game Legal Rush has the following rules:

- The goal of the game is to finish all the cases. You score points for finishing cases and lose points for mishandling cases.
- Players work together to finish cases. To finish a case properly, players must follow its corresponding brief. Cases are finished when its evidence, arguments and court time matches the corresponding brief.
- There are 6 kinds of evidence (documents, photographs, affidavits, contracts, depositions, and recordings) and 5 kinds of arguments (logical, ethical, emotional, procedural, and factual).
- If a case is presented before its evidence are collected, it is mishandled. If a case is mishandled, it cannot be finished.
- There are 3 types of board areas: evidence rooms, conference rooms, and courtrooms. Each board area has a limited number of action spaces. If there are no available action spaces, the board area cannot be used. Action spaces reset to available after a turn is finished.
- Players have two lawyers which can carry out one action per turn.
- Players can only choose action for their lawyers from the Valid Actions

Actions

1. Collect Evidence: Collect evidence for a case from the specified evidence room. Removes one available action space from evidence room.
2. Prepare Arguments: Collect arguments for a case from specified conference room. Removes one available action space from conference room.
3. Present Case: Present a case by one court time on specified courtroom. Removes one available action space from courtroom.

The game ends when:

- All the cases are either finished or mishandled
- You have played all 11 turns of the game.

These conventions will help when playing the game:

1. Collect Evidence before Presenting Case - Only present a case if the case already has its evidence to avoid mishandled cases
2. Do Not Work on Mishandled or Finished Cases - If a case is mishandled or finished, you do not need to work on it. Do not do actions on mishandled cases or finished cases
3. Avoid Mishandling Cases - Do not present a case before its evidence are collected.

A.6 Alien Rush Manual

The game Alien Rush has the following rules:

- The goal of the game is to finish all the irenkels. You score points for finishing irenkels and lose points for ruining irenkels.
- Players work together to finish irenkels. To finish an irenkel properly, players must follow its corresponding blintar. Irenkels are finished when its ayplixs, plurns and jantrox time matches the corresponding blintar.
- There are 6 kinds of ayplixs (zarvex, fluxin, shonix, flozix, grintal, and brentix) and 5 kinds of plurns (drikel, quentor, froxen, glivent, and plorix).
- If an irenkel is jantroxed before its ayplixs are collected, it is ruined. If an irenkel is ruined, it cannot be finished.
- There are 3 types of board areas: drovus, quelixths, and yorvexs. Each board area has a limited number of action spaces. If there are no available action spaces, the board area cannot be used. Action spaces reset to available after a turn is finished.
- Players have two workers which can carry out one action per turn.
- Players can only choose action for their workers from the Valid Actions

Actions

1. Take Ayplixs: Collect ayplixs for an irenkel from the specified drovus. Removes one available action space from drovus.
2. Take Plurns: Collect plurns for an irenkel from specified quelixth. Removes one available action space from quelixth.
3. Jantrox Irenkel: Jantrox a pokem by one jantroxing time on specified yorvex. Removes one available action space from yorvex.

The game ends when:

- All the irenkels are either finished or ruined
- You have played all 10 turns of the game.

These conventions will help when playing the game:

1. Take Ayplixs before Jantroxing Irenkel - Only jantrox an irenkel if the irenkel already has its ayplixs to avoid ruined irenkels
2. Do Not Work on Ruined or Finished Irenkels - If an irenkel is ruined or finished, you do not need to work on it. Do not do actions on ruined irenkels or finished irenkels
3. Avoid Ruining Irenkels - Do not jantrox an irenkel before its ayplixs are collected.

A.7 Sample Kitchen Rush Observation

Recipes:
Fettucine Alfredo:
  ingredients: 'pasta': 2, 'cheese': 1, 'meat': 2
  spices: 'white': 1, 'black': 1
  cook time: 1

Orders:
Fettucine Alfredo:
  ingredients: not collected
  spices: not collected
  cook time: 0
  finished: False
  ruined: False

Board Areas:
pantry:
  area_type: pantry
  area_id: pantry
  ingredients: 'meat': 40, 'carrot': 40, 'lettuce': 40, 'bread': 40, 'pasta': 40, 'cheese': 40
  available_action_spaces: 4

spicebag:
  area_type: spicebag
  area_id: spicebag
  spices: 'green': 12, 'black': 12, 'white': 12, 'red': 8, 'yellow': 8
  available_action_spaces: 4

stoves:
  area_type: stoves
  area_id: stoves
  available_action_spaces: 4

Valid Actions:

0. {'action_type': 'TAKE_INGREDIENTS', 'area_id': 'pantry', 'value': {'order': 'Fettucine Alfredo'}}
1. {'action_type': 'TAKE_SPICES', 'area_id': 'spicebag', 'value': {'order': 'Fettucine Alfredo'}}
2. {'action_type': 'COOK_MEAL', 'area_id': 'stoves', 'value': {'order': 'Fettucine Alfredo'}}

A.8 Sample Legal Rush Observation

Briefs:
Anderson V. Metro Transportation Corp.:
  evidence: 'depositions': 2, 'recordings': 1, 'documents': 2
  arguments: 'emotional': 1, 'ethical': 1
  court time: 1

Cases:
Anderson V. Metro Transportation Corp.::
  evidence: not collected
  arguments: not collected
  court time: 0
  finished: False
  mishandled: False

Board Areas:
evidence room:
  area_type: evidence room
  area_id: evidence room
  evidence: 'documents': 40, 'photographs': 40, 'affidavits': 40, 'contracts': 40, 'depositions': 40, 'recordings': 40
  available_action_spaces: 4

spicebag:
  area_type: conference room
  area_id: conference room
  spices: 'logical': 12, 'ethical': 12, 'emotional': 12, 'factual': 8, 'procedural': 8
  available_action_spaces: 4

stoves:
  area_type: courtrooms
  area_id: courtrooms
  available_action_spaces: 4

Valid Actions:

0. {'action_type': 'COLLECT_EVIDENCE', 'area_id': 'evidence room','value': {'case': 'Anderson V. Metro Transportation Corp.'}}
1. {'action_type':''PREPARE_ARGUMENTS', 'area_id': 'conference room', 'value': {'case': 'Anderson V. Metro Transportation Corp.'}}
2. {'action_type': 'PRESENT_CASE', 'area_id': 'courtrooms', 'value': {'case': 'Anderson V. Metro Transportation Corp.'}}

A.9 Sample Alien Rush Observation

Blintars:
Zarnorblat Alfrek::
    ayplixs: ‘grintal’: 2, ‘brentix’: 1, ‘zarvex’: 2
    plurns: ‘froxen’: 1, ‘quentor’: 1
    jantrox time: 1

Irenkels:
Zarnorblat Alfrek:
    ayplixs: not collected
    plurns: not collected
    jantrox time 0
    finished: False
    ruined: False

Board Areas:
drovus:
    area_type: drovus
    area_id: drovus
    evidence: ‘zarvex’: 40, ‘fluxin’: 40, ‘shonix’: 40, ‘flozix’: 40, ‘grintal’: 40, ‘brentix’: 40
    available_action_spaces: 4

quelixth:
    area_type: quelixth
    area_id: quelixth
    spices: ‘drikel’: 12, ‘quentor’: 12, ‘froxen’: 12, ‘plorix’: 8, ‘glivent’: 8
    available_action_spaces: 4

yorvexs:
    area_type: yorvexs
    area_id: yorvexs
    available_action_spaces: 4

Valid Actions:

0. {‘action_type’: ‘TAKE_AYPLIXS’, ‘area_id’: ‘drovus’, ’value’: {’irenkel’: ‘Zarnorblat Alfrek’}}
1. {‘action_type’: ‘TAKE_PLURNS’, ‘area_id’: ‘quelixth’, ’value’: {’irenkel’: ‘Zarnorblat Alfrek’}}
2. {‘action_type’: ‘JANTROX_POKEM’, ‘area_id’: ‘yorvexs’, ’value’: {’irenkel’: ‘Zarnorblat Alfrek’}}

A.10 Group KRush Generation Example

Explanation: Based on the action history and final results, it is clear that the group struggled with coordinating their actions effectively, leading to all orders being ruined. Key areas for improvement include ensuring ingredients are collected before cooking, prioritizing actions to avoid overlap, and maintaining clear communication.

Strategies:

1. Prioritize Ingredient Collection: Ensure all required ingredients for an order are collected before moving on to spices or cooking.
2. Sequence Actions Properly: Follow the sequence of collecting ingredients first, then spices, and finally cooking. Avoid cooking any order until all its ingredients and spices are collected.
3. Communicate and Coordinate: Players should communicate their planned actions to avoid overlapping and ensure efficient use of action spaces.
4. Assign Orders: Assign specific orders to each player to manage, reducing confusion and overlap in actions.
5. Monitor Action Spaces: Keep track of available action spaces in the pantry, spicebag, and stoves to avoid wasted actions or delays.
6. Avoid No-Action Turns: Plan ahead to ensure every turn is used productively, minimizing the need for No-Action decisions

A.11 Roles KRush Generation Example

Explanation: From the action history and final results, it is evident that there were several key mistakes made by both players. Player 2 repeatedly cooked orders when ingredients were not fully collected, leading to ruined orders. Player 1 generally followed the rule of taking ingredients first but sometimes took ingredients for the wrong order based on the game's prioritization. To improve coordination and performance, we can assign specific roles to each player to streamline the process and avoid repeating mistakes.

Roles & Feedback:
p1:

1. Ingredients Specialist: Focus on collecting ingredients for all orders.
2. Order Prioritizer: Prioritize orders based on the amount of ingredients and spices needed, ensuring that orders with fewer ingredients are collected first.

Feedback:

- You did a good job focusing on collecting ingredients. However, make sure you are always prioritizing the orders that need fewer ingredients first to streamline the process.
- Double-check the status of each order before taking ingredients to ensure you are working on the correct one.

p2:

1. Spices Specialist: Focus on collecting spices for all orders.
2. Cooking Manager: Only cook orders after checking with p1 to confirm that all ingredients and spices are collected.

Feedback:

- Avoid cooking orders before confirming that all required ingredients and spices are collected to prevent ruining them.
- Coordinate more closely with p1 to ensure you are collecting spices for the orders that have all their ingredients first.
- Stay vigilant about the game's rules and ensure you are working on the orders that are ready to be cooked.

B ANOVA Analysis

The ANOVA model includes domain, difficulty, group composition, task knowledge, teacher condition, and episode as independent variables with score as the dependent variable (Table 3). This model explains 68% of the variance in the data.

Independent Variables	
domain	KRush, LRush, ARush
difficulty	hard, easy
group composition	hom-strong-strong, het-strong-strong, het-weak-strong, hom-weak-weak, het-weak-weak
task knowledge	obs, `obs+manual`
teacher	group teacher, roles teacher, no teacher
episode	episode 1, episode 2, episode 3
Dependent Variables	
score	# of completed recipes by the group ranging from 0-5.

Table 3: The list of independent and dependent variables and their possible values included in the ANOVA.

B.1 Main Effect and Interaction Details

We investigate each main effect of the independent variables, as well two-way and three-way interactions between reported in Table 4. We report post-hoc student-t analysis of the main effects and significant interactions below.

Domain Interaction The main effect of domain is significant ($F(2, 7960) = 340.17$, $p < .0001$). Post-hoc analysis using Student-t test indicates that groups perform from best to worst in the following order: KRush, LRush, and ARush. All two-way and three-way interactions between domain, composition, task knowledge, as well as between domain, task knowledge, and difficulty are significant. The significant three way interactions between domain, composition, and task knowledge reveal that in strong-strong and weak-weak `obs+manual` groups and weak-weak `obs` groups, the KRush and LRush condition are significantly greater than the ARush condition, but there is no significant difference between KRush and LRush domains. Similarly, the significant three way interaction between domain, condition, and difficulty indicates that in `obs+manual` groups in the hard difficulty, the KRush and LRush condition are significantly greater than the ARush condition, but there is no significant difference between KRush and LRush domains.

Difficulty Interaction The main effect of difficulty is significant ($F(1, 7960) = 2774.24$, $p < .0001$). Post-hoc analysis using the Student-t test indicates that the mean score in the easy condition is significantly greater than the hard condition. This effect holds in all significant two-way and three-way interactions. That is, groups of the same domain-condition, and condition-composition and domain-composition pairing perform significantly better in the easy difficulty than the hard difficulty.

Group Composition Interaction The main effect of composition is significant ($F(4, 7960) = 1855.35$, $p < .0001$). Post-hoc analysis using the Student-t test indicates that groups perform from best to worst in the following order: strong-strong, weak-strong, weak-weak. There is no significant difference between het-strong-strong and hom-strong-strong groups, but there is a significant difference between hom-weak-weak groups and het-weak-weak groups. All two-way interactions of composition with domain, task knowledge, teacher condition, or difficulty are significant. These two-way interaction indicate that there is no significant difference between hom-weak-weak group and het-weak-weak group for groups (1) in the LRush and ARush domains, (2) in the `obs` condition, (3) in any teacher condition, and (4) in the hard difficulty. The significant three-way interactions between these variables are consistent with two-way interaction findings. That is, the significant difference between hom-weak-weak and het-weak-weak groups only holds for `obs+manual` groups in KRush and `obs+manual` groups in the easy condition.

Task Knowledge Interaction The main effect of task knowledge is significant ($F(1, 7960) = 1030.67$, $p < .0001$). Post-hoc Student-t analysis reveals the mean score for groups in the `obs+manual` condition is significantly greater than the `obs` condition. This effect holds in all significant two-way interactions and most three-way interactions. However, The significant three-way interaction between task knowledge, composition and difficulty indicates for weak-weak groups in the hard difficulty, there is no significant difference between the `obs+manual` and `obs` condition.

Teacher Interaction The main effect of teacher condition is significant ($F(2, 7960) = 17.36$, $p < .0001$). Post-hoc analysis shows pairs with the group teacher perform significantly better than the roles and no teacher condition; however, there is no significant difference between roles teacher condition and no teacher condition.

The significant three-way interaction with composition and teacher condition indicates that the main effect of teacher condition does not consistently hold. In the `obs+manual` condition, groups do not see consistent significant difference between teacher conditions, and sometimes perform significantly worse with a teacher. In the `obs` condition: (1) hom-strong-strong groups perform best to worst in group, roles, and none, (2) het-weak-strong perform significantly better with the roles and group teacher than no teacher, but there is no significant difference between roles and group, (3) het-strong-strong and hom-weak-weak groups perform best with the group teacher, but there is no difference between group and roles teacher and between roles and no teacher, and (4) het-weak-weak groups do not benefit significantly in any teacher condition. In other words, we find that the teacher feedback inconsistently improves group performance, particularly when groups have weak partners. All two-way interactions with teacher condition are significant, but do not change the interpretation above.

Episode Interaction The main effect of episode is significant ($F(2, 7960) = 5.94$, $p = 0.0026$). Post-hoc analysis reveals that groups perform significantly better after episode 1; however, there is no significant difference between episode 2 and 3. The significant three-way interaction with composition reveals that the main effect does not hold in `obs+manual` groups, often with some groups performing significantly worse in later episodes. In `obs` groups, the episode main effect holds in the group and roles conditions; however, for weak-weak groups and groups with no teacher, performance across episodes does not consistently improve. That is, when groups already have manual or groups have weak partners, teacher feedback does not improve reliably performance over multiple episodes. The significant two-way interactions with episode do not change this interpretation.

Source	DF	Sum of Squares	F Ratio	p-value
Main Effects				
domain	2	562.05	340.17	$< .0001$*
difficulty	1	2291.85	2774.24	$< .0001$*
composition	4	6130.95	1855.35	$< .0001$*
task knowledge	1	851.46	1030.67	$< .0001$*
teacher	2	28.68	17.36	$< .0001$*
episode	2	9.82	5.94	0.0026*
Two-way Interactions				
episode x composition	8	2.29	0.35	0.9479
episode x task knowledge	2	49.51	29.97	$< .0001$*
episode x teacher	4	19.73	5.97	$< .0001$*
composition x task knowledge	4	456.19	138.05	$< .0001$*
composition x teacher	8	20.59	3.11	0.0016*
task knowledge x teacher	2	39.33	23.81	$< .0001$*
domain x composition	8	116.89	17.69	$< .0001$*
domain x task knowledge	2	148.29	89.75	$< .0001$*
domain x difficulty	2	68.90	41.70	$< .0001$*
composition x difficulty	4	478.04	144.66	$< .0001$*
task knowledge x difficulty	1	50.27	60.85	$< .0001$*
Three-way Interactions				
domain x composition x task knowledge	8	214.90	32.52	$< .0001$*
episode x composition x task knowledge	8	24.62	3.73	0.0002*
episode x composition x teacher	16	17.15	1.30	0.1884
episode x task knowledge x teacher	4	17.35	5.25	0.0003*
composition x task knowledge x teacher	8	21.84	3.31	0.0009*
domain x composition x difficulty	8	12.44	1.88	0.0581
domain x task knowledge x difficulty	2	10.06	6.09	0.0023*
composition x task knowledge x difficulty	4	21.54	6.52	$< .0001$*
episode x composition x task knowledge x teacher	16	22.75	1.72	0.0361*
domain x composition x task knowledge x difficulty	8	39.60	5.99	$< .0001$*

Table 4: Effect tests and interactions included in ANOVA. P-values with asterisks are significant ($p < 0.05$)