Evaluating Creativity and Deception in Large Language Models: A Simulation Framework for Multi-Agent Balderdash

Parsa Hejabi[*†] **Elnaz Rahmati**[*†] **Alireza S. Ziabari**[†] **Preni Golazizian**[†]

Jesse Thomason[†] **Morteza Dehghani**[†]

[†] University of Southern California

{hejabi, erahmati, salkhord, golazizi, jessetho, mdehghan}@usc.edu

Abstract

Large Language Models (LLMs) have shown impressive capabilities in complex tasks and interactive environments, yet their creativity remains underexplored. This paper introduces a simulation framework utilizing the game Balderdash to evaluate both the creativity and logical reasoning of LLMs. In Balderdash, players generate fictitious definitions for obscure terms to deceive others while identifying correct definitions. Our framework enables multiple LLM agents to participate in this game, assessing their ability to produce plausible definitions and strategize based on game rules and history. We implemented a centralized game engine featuring various LLMs as participants and a judge LLM to evaluate semantic equivalence. Through a series of experiments, we analyzed the performance of different LLMs, examining metrics such as True Definition Ratio, Deception Ratio, and Correct Guess Ratio. The results provide insights into the creative and deceptive capabilities of LLMs, highlighting their strengths and areas for improvement. Specifically, the study reveals that infrequent vocabulary in LLMs' input leads to poor reasoning on game rules and historical context.[1]

1 Introduction

Large Language Models (LLMs) have recently been employed as agents in various complex tasks, showcasing their potential in dynamic, interactive environments (Dorbala et al., 2024; Singh et al., 2024). This has led to a growing interest in LLM-based multi-agent systems (LLM-MA), particularly within the realm of gaming (Mukobi et al., 2024; Xu et al., 2023). Games offer a structured yet flexible platform to analyze and understand LLM behavior under diverse scenarios (Light et al., 2023).

Currently, LLMs are typically evaluated through static tasks (Lee et al., 2023; Zhao et al., 2024; Gómez-Rodríguez and Williams, 2023). Traditional games like Avalon (Wang et al., 2023) and Werewolf (Xu et al., 2024) have also been used to benchmark LLMs, focusing on logical reasoning and strategic interaction. These games require players to engage in deception, deduction, and negotiation, providing valuable insights into LLMs' decision-making processes. However, these studies often overlook the assessment of creativity.

To address this gap, we introduce a simulation framework for the game *Balderdash*. In this game, players generate plausible yet fictitious definitions for obscure terms, aiming to deceive other players while identifying the correct definitions. We argue that Balderdash can be used to evaluate both the creativity and logical reasoning of LLMs, challenging the models to balance these two crucial aspects and providing a comprehensive assessment of their capabilities.

In this paper, we aim to assess the creativity of LLMs by evaluating their ability to generate plausible definitions for obscure words in Balderdash. We will further examine their logical reasoning skills by observing how effectively they deceive opponents and identify correct definitions in the context of the game. Finally, we will investigate the performance of these models in a multi-agent setting where both creativity and logical deduction are crucial for success.

2 Related Work

LLMs have demonstrated remarkable success in planning and reasoning capabilities, resulting in the automation of numerous tasks, such as science experiments (Zheng et al., 2023) and software development (Qian et al., 2023; Hong et al., 2023; Dong et al., 2023). The advancement of using an LLM as a planning or decision-making agent has led to significant progress in complex problem-solving and world simulation within LLM-MA

[*]Equal contribution.

[1]https://github.com/ParsaHejabi/Simulation-Framework-for-Multi-Agent-Balderdash

systems. One example of world simulation is using memory-based adjustment for LLM agents in games with cooperative or competitive communication paradigms, with either centralized or decentralized communication structures (Guo et al., 2024). For instance, Mukobi et al. (2024) use the Welfare game, where LLM agents balance investing in military units and improving their nations' welfare to evaluate the cooperative capabilities of LLMs.

Avalon (Wang et al., 2023; Light et al., 2023) and Werewolf (Xu et al., 2024, 2023) are two other games used in this paradigm, both with two groups of roles, good and evil, and the winner is the team that succeeds in eliminating the other. The evil group members have the advantage of knowing each other, while the good group members should rely on behavioral patterns to find other members in their group. The most important capabilities examined in these types of games are deceiving other players and distinguishing between the behavioral patterns of good and evil.

Wang et al. (2023) compare the performance of Recursive Contemplation (ReCon) and Chain-of-Thought (CoT) (Wei et al., 2022) for LLM reasoning in the Avalon game, where agents are evaluated by the `gpt-4-0613` model (OpenAI et al., 2024) using six binary labels (concealment, logic, contribution, persuasiveness, information, and creativity), showing the superiority of ReCon. Light et al. (2023) also use Avalon for benchmarking LLMs based on their win rate, showing that while LLMs can deduce information from their discussions with other players, they are not able to strategize accordingly. Xu et al. (2023) use Werewolf to examine the effect of memory (experience pool) and its size on agent adjustment in the game, where models are shown to improve over rounds based on win rate. Xu et al. (2024) also use the Werewolf game to evaluate LLMs combined with reinforcement learning to examine agent adjustment.

Outside of game simulations, Lee et al. (2023) and Orwig et al. (2024) use Divergent Thinking (DT) to evaluate LLMs' creativity by calculating semantical differences among multiple responses for a specific topic, e.g., describing a new feasible use case for a typical object. DT is defined as a thought process that enables people to explore and think in multiple directions (Guilford, 1967), which aligns with the objectives of the Balderdash game, explained in the next Section.

3 The Original Balderdash Game

Balderdash is a word game where players aim to create plausible-sounding definitions for rare and unusual words. The game has two objectives: 1. to deceive other players into believing an invented definition is the correct one, and 2. to correctly identify the true definition among those presented. The game also includes a competitive aspect where players advance on a board towards a finish line.

In each round of the game, the *Dasher* (the leader of each round) draws a card from Balderdash's deck of cards, which contains obscure, rare words along with their definitions. The Dasher announces the chosen word to all players, who then write a definition down on their sheets. Players can either write down the true definition (if they know it) or invent a plausible definition they think will convince others.

Once all definitions are submitted, the Dasher examines the answers and immediately awards three points to any player whose invented definition closely resembles the true definition. These players do not continue participating in that round. The Dasher then mixes all the remaining invented definitions with the true definition of the word and reads them aloud. Players must vote for the definition they believe is the correct one. Correct guesses are awarded two points, and one point is awarded for each vote a player's definition receives. Additionally, the Dasher receives three points if no player guesses the correct definition. The game continues with a new Dasher each round until one player reaches the finish line on the game board.[2]

4 LLM-MA Balderdash

We propose a framework where LLMs play the Balderdash game, enabling the benchmarking of their capabilities in generating and evaluating creative content. This framework includes a centralized game engine featuring various LLMs as participants, multiple datasets as the game's word decks, an LLM as the Dasher, and a review of previous rounds given to players as history. These features are discussed in more detail below.

4.1 LLMs as Participants

In our framework, LLMs play Balderdash against one another. To incorporate a range of different LLMs, we include four open-source, small, instruct-tuned models loaded locally and one large

[2] See `https://www.hasbro.com/common/instruct/balderdash.pdf` for more detailed instructions.

Model Name	Abbreviation	Reference	# Parameters
meta-llama/Meta-Llama-3-8B-Instruct	Llama	AI@Meta (2024)	8 billion
microsoft/Phi-3-small-8k-instruct	Phi	Abdin et al. (2024)	7 billion
google/gemma-1.1-7b-it	Gemma	Gemma Team et al. (2024)	7 billion
mistralai/Mistral-7B-Instruct-v0.3	Mistral	Jiang et al. (2023)	7 billion
gpt-3.5-turbo-0125	GPT	OpenAI (2024)	Not specified

Table 1: Summary of the LLMs used in the framework. The models are referenced using their abbreviated names throughout the paper.

Dataset	# Words	Avg. Frequency
All Balderdash	225	1.8e-8
Llama-Known	84	3.6e-8
Phi-Known	88	3.7e-8
Gemma-Known	35	5.7e-8
Mistral-Known	88	3.7e-8
GPT-Known	131	2.6e-8
Basic English	2865	6.3e-5

Table 2: Summary of datasets used in this work. The "Known" datasets are named using the abbreviated names of the models.

API-based model (see Table 1). Each game consists of multiple rounds, with the same set of players participating in each round. Since there are no boards implemented in the framework (and hence no finish line), the players' objective is to maximize their points.

4.2 Word Deck

We created two different datasets used as the Word Decks in our framework. First, we rely on the set of words originally used in the Balderdash game, containing rare and infrequent English words to simulate the actual game. We also created multiple subsets of this dataset containing the words known by each model. According to Kang and Choi (2023), LLMs are biased towards frequent words and co-occurrences, making them vulnerable and unpredictable when infrequent words are used in the input. Therefore, we created another dataset containing the most frequent English words to evaluate LLMs on both frequent and infrequent decks of words. Table 2 shows a summary of the datasets, along with their average frequency calculated using the NGRAMS Dataset (Trenkmann, 2023).

4.2.1 Balderdash Words

We created the "All Balderdash" dataset containing 225 distinct Balderdash words sourced from the Wordnik dictionary's list of Balderdash game words[3], complete with all their different definitions and their part of speech tags.

Known Balderdash Words: Following Jhirad et al. (2023), we created datasets of words understood by each LLM by inputting every word along with its part of speech from the "All Balderdash" dataset into each model five times, using a temperature value of 0.9. Each model is prompted to act as a universal dictionary and provide a definition of each word. Subsequently, we used `Llama` as a semantic equivalence judge to determine whether each definition was semantically equivalent to the word's actual definition. We explain this choice in Section 4.3. The prompts provided no context about the Balderdash game (all prompts are detailed in Appendix D). If the model affirmed the semantic equivalence of the majority (three or more out of five) of the definitions, the word is labeled as a "known" word for that model.

4.2.2 Basic Frequent English Words

We use the Oxford 3000 word list (Oxford University Press, 2024), containing the most frequent English words. Using the NLTK package (Bird and Loper, 2004), English stopwords are removed from this list, resulting in 2895 words. Then, the Merriam-Webster dictionary API (Merriam-Webster, 2024) is used to obtain the various definitions and part of speech tags of these words. Words that do not have any definitions in the Merriam-Webster dictionary are discarded, resulting in 2865 words. The gathered data is cleaned with regular expressions to remove special tokens as defined in the API's documentation. Given that the words

[3] https://www.wordnik.com/lists/balderdash-game-words

in this dataset are the most frequently used English words and thus likely present in the training data of these LLMs, it is expected that even under high-temperature conditions during LLM inference, they will be able to generate accurate definitions for these words.

4.3 Dasher (Judge)

The main responsibility of the Dasher is to act as a judge and examine participants' definitions. Following Zheng et al. (2024), where an LLM is used to evaluate open-domain question-answering, we use an LLM as the judge to determine whether each generated definition is semantically equivalent to the reference dictionary definition.

We created a dataset ("Judge Evaluation Data") to evaluate the best LLM for the Dasher role in the game. This dataset consists of 40 randomly selected words from the "All Balderdash" dataset. For each word, `GPT` was prompted once to provide an accurate definition and again to generate a deceiving definition within the context of the Balderdash game. A human annotator then labeled the `GPT`-generated definitions (including both correct and deceiving definitions) as "True" if they were equivalent to the dictionary definition and "False" otherwise. Each LLM was then prompted to do the same task and respond with either "True" or "False." The specific prompts used are detailed in Appendix D.

Based on the alignment of human labels and each LLM's labels (see Table 3), `Llama` was chosen as the judge of the game in all experiments. Surprisingly, `GPT` performed the worst. Further investigation revealed that the "Judge Prompt" described in Section 4.5 led `GPT` to become a very strict judge, resulting in generating "False" even for small differences in details. We acknowledge that LLMs might have a self-enhancement bias toward their own output or other machine-generated outputs (Chen et al., 2024; Zheng et al., 2024), resulting in a slightly unfair evaluation.

It is also worth mentioning that BERTScore (Song et al., 2021) is another method for calculating semantic distance used in machine translation. However, our experiments detailed in Appendix A demonstrate that it is not feasible to use BERTScore for the judge component.

4.4 History

To provide the players with a memory-based review of previous rounds' outcomes and a sense of their performance, we give each player a history of each round in the form of a CSV file.

LLM	F_1	Recall	Precision	Accuracy
Llama	**0.74**	0.74	0.74	**0.82**
Phi	0.72	0.74	0.71	0.81
Gemma	0.73	**0.77**	0.70	0.81
Mistral	0.70	0.70	0.70	0.80
GPT	0.19	0.11	**0.75**	0.68

Table 3: Evaluation of each LLM as the Dasher using 80 manually labeled data points.

4.5 Game Engine

We implemented a game engine capable of simulating Balderdash within a multi-agent environment with centralized communication. In this game engine, LLMs are given five categories of prompts (technical details of the game engine and prompts are available in Appendix C and Appendix D):

Game Rules Prompt: Describes the game rules, scoring rules, and the player's objective, given as a "system" prompt. For models that do not support the "system" role, this prompt is placed at the beginning of the "user" prompt.

History Prompt (Optional): Provides a review of a moving window of the previous rounds, given as a "user" prompt. This approach is to simulate how a human might recall and adapt their strategy over time. The history is available in two versions: 1. Full History includes detailed information for each round, namely round ID, player rank up to that round, score, word, reference definition, generated definition, semantic equivalence, correct guess indicator, deception ratio, and round winners' strategies. 2. Mini History includes a concise version with round ID, player rank up to that round, score, word, and generated definition.

Generate Definition Prompt: Asks the player to generate a definition for a given word based on the game rules, concatenated with the optional history prompt.

Vote on Definitions Prompt: Asks the player to choose the reference dictionary definition for a word among all given definitions during the voting phase, concatenated with the optional history prompt.

Judge Prompts: Consist of a "system" and a "user" prompt, asking the judge LLM to evaluate whether a reference dictionary definition and a given definition capture the same core concept.

Upon each run, the results of each round are stored in a MongoDB database with collections for games, rounds, and players storing game configurations, player details, and round-specific data:

Games Collection: Stores overall game configurations, such as game description, number of rounds, judge LLM model name, random seed, scoring rules, history window size, LLMs' temperature, game's word deck, and prompt files used.

Rounds Collection: Stores round-specific data, including the announced word, its definition, round players' received scores, cast votes, generated definitions, and the judge's evaluation on two aspects: whether each definition is semantically equivalent to the reference definition, and whether it matches at least one of the different meanings of the word (for words with multiple definitions).

Players Collection: Stores player details, including the LLM name, cumulative score over each round, and rank history in each round.

5 Evaluation

Each Balderdash game, denoted as G_m, consists of N rounds (R_n^m). In each G_m, a constant set of K players participate ($P = \{p_1, ..., p_k\}$). The set of all players using the l^{th} LLM is denoted as LLM_l. Therefore, each player (p_k) is a member of one and only one LLM_l. R_n^m contains information about all players participating in the n^{th} round of G_m, including "`judge decision`", "`llm knows one`", "`votes`", and "`scores`". The first two are mappings between each p_k and a binary value, indicating whether p_k's generated definition was semantically equal to the first reference dictionary definition of R_n^m's word and whether p_k's output was semantically equal to at least one of the various definitions of R_n^m's word, respectively. "`votes`" is another mapping containing information on each p_k's vote in the voting phase, either for another player ($p_{k'}$) or for "-1", representing the reference dictionary definition. "`scores`" is a mapping between each p_k and an integer value indicating p_k's score in R_n^m.

5.1 Metrics

We define five metrics for each round (R_n^m): 1. True Definition Ratio (TDR), 2. LLM Knows Ratio (LKR), 3. Deception Ratio (DR), 4. Correct Guess Ratio (CGR), and 5. Average Score (AS). $TDR_n^m(LLM_l)$ represents the ratio of true definitions generated for the announced word in the m^{th} game and the n^{th} round for all players in LLM_l.

$$TDR_n^m(LLM_l) = \frac{\sum_{p_k \in LLM_l} R_n^m(\texttt{judge decision})[p_k]}{|LLM_l|} \quad (1)$$

LKR measures the ratio of instances where the LLM aims to generate the true definition.

$$LKR_n^m(LLM_l) = \frac{\sum_{p_k \in LLM_l} R_n^m(\texttt{llm knows one})[p_k]}{|LLM_l|} \quad (2)$$

The metrics DR and CGR are designed to evaluate the performance of each LLM in the voting phase. DR measures the success ratio of LLMs in deceiving other players.

$$DR_n^m(LLM_l) = \frac{1}{|LLM_l|} \sum_{p_k \in LLM_l} \frac{\sum_{v \in R_n^m(\texttt{votes})} \delta(v, p_k)}{|R_n^m(\texttt{votes})| - 1} \quad (3)$$

CGR reflects the LLMs' ability to identify the reference dictionary definition amidst deceiving ones.

$$CGR_n^m(LLM_l) = \frac{\sum_{p_k \in LLM_l} \delta(R_n^m(\texttt{votes})[p_k], -1)}{|LLM_l|} \quad (4)$$

AS is the average score achieved by an LLM. This metric also represents a weighted summation of TDR, DR, and CGR, where the weights are determined by the game's scoring rules.

$$AS_n^m(LLM_l) = \frac{\sum_{p_k \in LLM_l} R_n^m(\texttt{scores})[p_k]}{|LLM_l|} \quad (5)$$

The above metrics are used to assess the overall performance of LLMs in the LLM-MA Balderdash game. In cases where there is a dominant strategy that allows players to get the most points, such as generating the correct definition when the correct definition score is set to a high value, we define convergence to assess the LLM's strategy. The goal of convergence is to determine if the model

can find and continuously use the most rewarding strategy. Convergence is defined as follows:

$$\overline{LKR}_n > 1 - \epsilon, \quad \forall n > T \tag{6}$$

6 Experiments & Results

To evaluate the LLMs' performance and strategy, we conduct three experiments. The first experiment provides a leaderboard of LLMs based on their proficiency in playing the original Balderdash game. The second experiment investigates whether LLMs learn from their history and converge to follow the most rewarding strategy. The final experiment targets LLMs' ability to reason over game rules and choose the best greedy choices.

6.1 Leaderboard Experiment

In this experiment, we aim to create a leaderboard for LLMs by having these models play Balderdash against each other. To keep the game fair, only models of comparable size (namely `Llama`, `Phi`, `Gemma`, and `Mistral`) are used. Using more advanced models would disrupt the game flow, as smaller models wouldn't be able to rise in the rankings and consequently learn from their history. Each game with four players representing four LLMs is run five times using five different subsets of words to ensure that the chosen set of words does not affect the results. This experiment is conducted with three types of history (none, mini, and full) and two datasets ("Basic Frequent English Words" and "All Balderdash") to examine the models' performance on both frequent and infrequent English words.

The results for "Basic Frequent English Words," shown in Table 4, indicate a considerable improvement for all models as the history becomes more informative. The only metric that decreases is CGR. As LKR approaches 1.0 for all models with increasing history, the ratio of rounds with more than one correct definition in the voting phase also increases. This could lead to confusion for all players and possibly result in a drop in CGR because the definitions in the voting phase are true definitions of the word but not the reference one used by the judge.

The results for "All Balderdash" are shown in Table 5. Contrary to the "Basic Frequent English Words" results, consistent improvement is not observed for all LLMs. A possible reason could be the infrequency of the words in this dataset. In almost all settings, `Phi` performs strongly in finding the correct definition during the voting phase, suggesting its potential for detecting disinformation. Furthermore, `Mistral` shows the best overall performance in deceiving its opponents, possibly due to greater creativity in generating deceptive definitions, as deception in Balderdash is an iterative process requiring creativity to avoid pattern recognition.

None of the models dominate the others across all game settings. However, when using the "Basic Frequent English Words" dataset, `Mistral` has the most wins, whereas when using the "All Balderdash" dataset, `Phi` performs best overall. It is worth mentioning that `Mistral` was the only model that failed to conform to the specified format in the voting prompt in two games.

6.2 Convergence Experiment

Although the leaderboard provides some insight into LLMs' performance, evaluating their strategies and understanding their behavior remains challenging. Therefore, this experiment aims to evaluate LLMs' reasoning and strategy in an environment where a dominant method for maximizing scores exists based on the history of past rounds. The dataset used in this experiment is limited to "Known Balderdash Words" for each LLM, and the game is run with three players using the same LLM (including `GPT`). Considering that the players know the definitions of the announced words, we hypothesize that in each game, the LLMs' LKR will converge since generating the true definition is the most rewarding strategy. Similar to the first experiment, each game is run five times with five different subsets of the dataset. Only two types of history (mini and full) are used in this experiment.

Figure 1 depicts $\overline{LKR}_n$ over rounds, showing that none of the models converge, contrary to our hypothesis. The plots show a reduction in fluctuations for the full history setting compared to the mini history, but still, there is no improvement or trend for any of the models over rounds. This phenomenon could be due to the infrequency of the words in the dataset or a weakness of these LLMs in finding or repeatedly using the best strategy.

6.3 Game Rules Experiment

The final experiment aims to assess LLMs' ability to understand and reason over the game rules without providing history. This experiment is conducted with one player, using the "Known Balderdash Words" dataset for each LLM, and two distinct rule sets: 1. awarding fifty points for generat-

HT	LLM	LKR	TDR	DR	CGR	AS
none	Llama	0.59 ± 0.10	0.40 ± 0.04	0.19 ± 0.12	0.56 ± 0.11	2.08 ± 0.19
	Phi	0.49 ± 0.12	0.33 ± 0.10	0.24 ± 0.13	**0.77 ± 0.12**	2.21 ± 0.30
	Gemma	**0.93 ± 0.02**	**0.77 ± 0.08**	**0.25 ± 0.13**	0.47 ± 0.19	**2.65 ± 0.13**
	Mistral	0.59 ± 0.13	0.48 ± 0.17	0.15 ± 0.07	0.74 ± 0.13	2.35 ± 0.29
mini	Llama	0.89 ± 0.12	0.78 ± 0.14	0.28 ± 0.18	0.49 ± 0.20	2.68 ± 0.15
	Phi	0.74 ± 0.18	0.60 ± 0.18	0.30 ± 0.17	**0.74 ± 0.07**	2.52 ± 0.22
	Gemma	0.92 ± 0.09	0.78 ± 0.12	0.30 ± 0.28	0.43 ± 0.16	2.63 ± 0.21
	Mistral	**0.94 ± 0.07**	**0.83 ± 0.07**	**0.34 ± 0.18**	0.35 ± 0.32	**2.76 ± 0.06**
full	Llama	0.93 ± 0.06	0.78 ± 0.16	0.61 ± 0.07	**0.52 ± 0.31**	2.72 ± 0.18
	Phi	0.98 ± 0.04	0.85 ± 0.05	0.53 ± 0.34	0.42 ± 0.26	2.79 ± 0.07
	Gemma	0.97 ± 0.04	0.84 ± 0.07	0.52 ± 0.29	0.44 ± 0.34	2.69 ± 0.21
	Mistral	**1.00 ± 0.00**	**0.90 ± 0.05**	**0.62 ± 0.35**	0.05 ± 0.10	**2.81 ± 0.06**

Table 4: Leaderboard experiment results on "Basic Frequent English Words," evaluating each LLM in three different settings based on history type (HT) using the average of LKR, TDR, DR, CGR, and AS metrics over all rounds and games. The highest value of each metric for different game settings is in bold. However, based on the standard deviation, this does not represent absolute superiority.

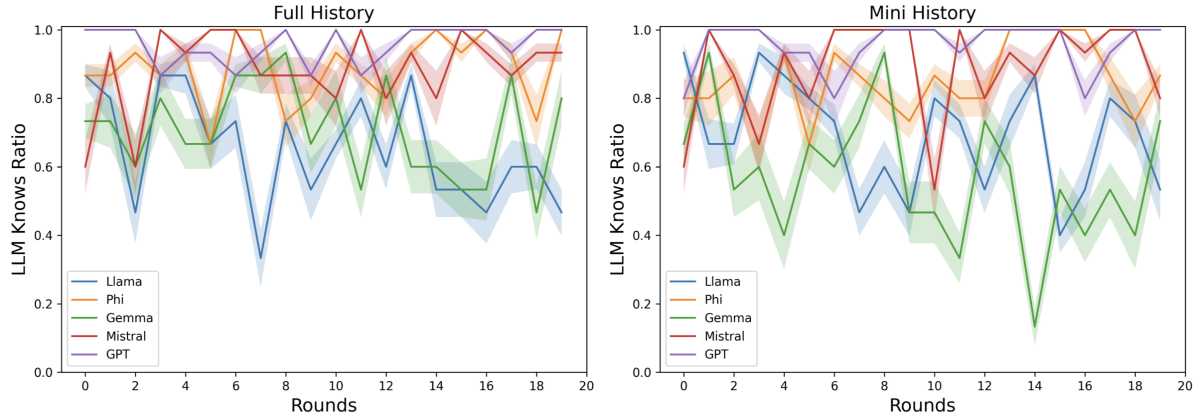

Figure 1: Convergence experiment on "Known Balderdash Words" with mini and full history types, examining changes in $\overline{LKR}_n$ over rounds. Note that for all LLMs, the standard deviation is down-scaled by a factor of 0.2 for presentation purposes.

ing the true definition, and 2. awarding zero points for the same task, both with the same scoring rules for correct guesses and receiving votes from other players in the voting phase. In this setting, our hypothesis is that LLMs will choose the most rewarding strategy in each setting in a greedy manner, which is generating the true definition and guessing the correct definition in the first and second experiment settings, respectively.

To assess our hypothesis, $\overline{TDR}$ and $\overline{LKR}$ were calculated in both settings (Table 6). Although there is a slight increase in $\overline{TDR}$ for `Mistral` and `GPT`, the results are still disappointing. Even with zero points for generating the true definition, models are still choosing this strategy, leading to zero points in each round.

7 Conclusion

Current LLM-MA game simulations overlook the assessment of creativity in LLMs. This study introduces a systematic framework through the Balderdash game to probe aspects of creativity, deception, and logical reasoning inherent in these models. Our initial assumption was that LLMs are familiar with most Balderdash words and can learn the patterns in machine-generated deceiving definitions, thereby enabling them to generate the correct definitions of words and choose the dictionary definition in the voting phase of Balderdash.

Contrary to our expectations, LLMs are not fa-

HT	LLM	LKR	TDR	DR	CGR	AS
none	Llama	0.27 ± 0.09	0.22 ± 0.11	0.26 ± 0.06	0.57 ± 0.12	2.00 ± 0.25
	Phi	0.31 ± 0.13	0.25 ± 0.11	0.19 ± 0.05	**0.62 ± 0.05**	**2.08 ± 0.22**
	Gemma	0.18 ± 0.10	0.15 ± 0.08	0.12 ± 0.03	0.35 ± 0.08	1.26 ± 0.15
	Mistral	**0.44 ± 0.17**	**0.33 ± 0.17**	**0.27 ± 0.04**	0.45 ± 0.08	2.05 ± 0.28
mini	Llama	0.29 ± 0.15	0.24 ± 0.12	**0.30 ± 0.07**	0.54 ± 0.12	2.04 ± 0.17
	Phi	**0.45 ± 0.16**	**0.35 ± 0.15**	0.26 ± 0.09	**0.56 ± 0.13**	**2.28 ± 0.19**
	Gemma	0.07 ± 0.07	0.05 ± 0.05	0.10 ± 0.06	0.32 ± 0.12	0.96 ± 0.35
	Mistral	0.44 ± 0.17	**0.35 ± 0.13**	**0.30 ± 0.07**	0.41 ± 0.11	2.05 ± 0.25
full	Llama	0.33 ± 0.16	0.24 ± 0.15	0.20 ± 0.06	0.39 ± 0.12	1.70 ± 0.27
	Phi	**0.40 ± 0.14**	**0.37 ± 0.14**	**0.33 ± 0.08**	**0.66 ± 0.13**	**2.52 ± 0.25**
	Gemma	0.17 ± 0.10	0.13 ± 0.10	0.19 ± 0.09	0.26 ± 0.13	1.19 ± 0.48
	Mistral	0.36 ± 0.18	0.31 ± 0.15	0.28 ± 0.06	0.36 ± 0.12	1.89 ± 0.27

Table 5: Leaderboard experiment results on "All Balderdash," evaluating each LLM in three different settings based on history type (HT) using the average of LKR, TDR, DR, CGR, and AS metrics over all rounds and games. The highest value of each metric for different game settings is in bold. However, based on the standard deviation, this does not represent absolute superiority.

LLM	LKR		TDR	
Correct Def. Points	**0**	**50**	**0**	**50**
Llama	0.60 ± 0.04	0.59 ± 0.08	0.55 ± 0.04	0.49 ± 0.09
Phi	0.64 ± 0.08	0.62 ± 0.12	0.55 ± 0.07	0.54 ± 0.09
Gemma	0.53 ± 0.09	0.50 ± 0.07	0.46 ± 0.14	0.46 ± 0.08
Mistral	0.79 ± 0.09	0.82 ± 0.09	0.65 ± 0.08	0.68 ± 0.11
GPT	0.92 ± 0.04	0.93 ± 0.07	0.86 ± 0.04	0.88 ± 0.05

Table 6: Examining LLM reasoning through the effect of game rules.

miliar with more than half of the Balderdash words and perform poorly during the voting phase. None of the models used in the experiments showed signs of correct reasoning based on game rules or strategy convergence derived from historical context. Interestingly, this phenomenon is more pronounced with Balderdash words (infrequent English words) compared to more frequent English words, suggesting that LLMs are more susceptible to failure in reasoning when encountering infrequent vocabulary.

The best judge among all LLMs we tested was `Llama`, which had the best alignment with human labels. Based on the leaderboard experiment, `Phi` performed strongly in finding the correct definition during the voting phase, suggesting its potential for detecting disinformation. Furthermore, `Mistral` showed the best overall performance in deceiving its opponents, likely due to its creativity in generating deceptive definitions.

Limitations

The judge in the LLM-MA Balderdash plays a crucial role in both running the game and evaluating its outcomes. Consequently, the accuracy of the judge is a critical factor in our work. In the current version of the game engine, an LLM serves as the judge. However, an alternative could involve replacing the LLM judge with a specialized model specifically trained to discriminate between true and deceiving definitions. This replacement would likely result in higher accuracy and a more reliable game simulation system.

Additionally, the possibility of self-enhancement bias should be considered when using an LLM as the judge. To evaluate this bias, we can assess each LLM as the judge on definitions generated not only by `GPT` (the model we're using) but also by all other LLMs employed in our work. By comparing error rates across different sets of generated definitions, we can gain insights into how biased these models are toward their own output.

We create subsets of words from the "All Balderdash" dataset understood by each LLM by probing the models' output using a prompt to generate a definition for each word. Given the low frequency of these words and the high temperature value in our setting, this might lead to false negatives. A better method would involve comparing the frequency of occurrence of "All Balderdash" words to that of "Basic Frequent English Words" in each LLM's training data (if available).

Currently, we use a high temperature during inference to create diversity in generated responses. Alternatively, using temperature scaling or diverse prompting methods for each player might result in more reliable outputs, especially for infrequent words. Furthermore, all evaluations of LLMs' strategy and reasoning in this work are based on game results and scores. Replacing current prompts with multi-stage reasoning and Chain-of-Thought (CoT) approaches might improve LLMs' performance in the game and provide better insights into the reasoning behind their strategies.

References

Marah Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Jianmin Bao, Harkirat Behl, Alon Benhaim, Misha Bilenko, Johan Bjorck, Sébastien Bubeck, Qin Cai, Martin Cai, Caio César Teodoro Mendes, Weizhu Chen, ..., and Xiren Zhou. 2024. Phi-3 technical report: A highly capable language model locally on your phone. *Preprint*, arXiv:2404.14219.

AI@Meta. 2024. Llama 3 model card.

Steven Bird and Edward Loper. 2004. NLTK: The natural language toolkit. In *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, pages 214–217, Barcelona, Spain. Association for Computational Linguistics.

Guiming Hardy Chen, Shunian Chen, Ziche Liu, Feng Jiang, and Benyou Wang. 2024. Humans or llms as the judge? a study on judgement biases. *arXiv preprint arXiv:2402.10669*.

Yihong Dong, Xue Jiang, Zhi Jin, and Ge Li. 2023. Self-collaboration code generation via chatgpt. *arXiv preprint arXiv:2304.07590*.

Vishnu Sashank Dorbala, James F. Mullen, and Dinesh Manocha. 2024. Can an embodied agent find your "cat-shaped mug"? llm-based zero-shot object navigation. *IEEE Robotics and Automation Letters*, 9(5):4083–4090.

Thomas Mesnard Gemma Team, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, Pouya Tafti, Léonard Hussenot, and et al. 2024. Gemma.

Carlos Gómez-Rodríguez and Paul Williams. 2023. A confederacy of models: a comprehensive evaluation of LLMs on creative writing. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 14504–14528, Singapore. Association for Computational Linguistics.

Sylvain Gugger, Lysandre Debut, Thomas Wolf, Philipp Schmid, Zachary Mueller, Sourab Mangrulkar, Marc Sun, and Benjamin Bossan. 2022. Accelerate: Training and inference at scale made simple, efficient and adaptable. `https://github.com/huggingface/accelerate`.

Joy Paul Guilford. 1967. *The nature of human intelligence.* McGraw-Hill.

Taicheng Guo, Xiuying Chen, Yaqi Wang, Ruidi Chang, Shichao Pei, Nitesh V. Chawla, Olaf Wiest, and Xiangliang Zhang. 2024. Large language model based multi-agents: A survey of progress and challenges. *Preprint*, arXiv:2402.01680.

Sirui Hong, Xiawu Zheng, Jonathan Chen, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, et al. 2023. Metagpt: Meta programming for multi-agent collaborative framework. *arXiv preprint arXiv:2308.00352*.

James Jhirad, Edison Marrese-Taylor, and Yutaka Matsuo. 2023. Evaluating large language models' understanding of financial terminology via definition modeling. In *Proceedings of the 13th International Joint Conference on Natural Language Processing and the 3rd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 93–100.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *Preprint*, arXiv:2310.06825.

Cheongwoong Kang and Jaesik Choi. 2023. Impact of co-occurrence on factual knowledge of large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 7721–7735, Singapore. Association for Computational Linguistics.

Hanmi Lee, Wenqing Zhou, HongHong Bai, Weiran Meng, Tianli Zeng, Kaiping Peng, Song Tong, and Takatsune Kumada. 2023. Natural language processing algorithms for divergent thinking assessment. In *2023 ieee 6th eurasian conference on educational innovation (ecei)*, pages 198–202. IEEE.

Jonathan Light, Min Cai, Sheng Shen, and Ziniu Hu. 2023. Avalonbench: Evaluating llms playing the game of avalon. *Preprint*, arXiv:2310.05036.

Merriam-Webster. 2024. Merriam-webster's dictionary api. https://dictionaryapi.com/. Accessed: 2024-04-18.

Gabriel Mukobi, Hannah Erlebach, Niklas Lauffer, Lewis Hammond, Alan Chan, and Jesse Clifton. 2024. Welfare diplomacy: Benchmarking language model cooperation.

OpenAI. 2024. New embedding models and api updates. https://openai.com/index/new-embedding-models-and-api-updates/. Accessed: 2024-05-16.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, ..., and Barret Zoph. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

William Orwig, Emma R Edenbaum, Joshua D Greene, and Daniel L Schacter. 2024. The language of creativity: Evidence from humans and large language models. *The Journal of Creative Behavior*.

Oxford University Press. 2024. Oxford learner's dictionaries: Oxford 3000 word list. Accessed: 2024-04-18.

Chen Qian, Xin Cong, Cheng Yang, Weize Chen, Yusheng Su, Juyuan Xu, Zhiyuan Liu, and Maosong Sun. 2023. Communicative agents for software development. *arXiv preprint arXiv:2307.07924*.

Ishika Singh, David Traum, and Jesse Thomason. 2024. Twostep: Multi-agent task planning using classical planners and large language models. *arXiv preprint arXiv:2403.17246*.

Yurun Song, Junchen Zhao, and Lucia Specia. 2021. SentSim: Crosslingual semantic evaluation of machine translation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3143–3156, Online. Association for Computational Linguistics.

Martin Trenkmann. 2023. NGRAMS – search the world's largest ngram dataset. Accessed on May 18, 2024.

Shenzhi Wang, Chang Liu, Zilong Zheng, Siyuan Qi, Shuo Chen, Qisen Yang, Andrew Zhao, Chaofei Wang, Shiji Song, and Gao Huang. 2023. Avalon's game of thoughts: Battle against deception through recursive contemplation. *Preprint*, arXiv:2310.01320.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837. Curran Associates, Inc.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Yuzhuang Xu, Shuo Wang, Peng Li, Fuwen Luo, Xiaolong Wang, Weidong Liu, and Yang Liu. 2023. Exploring large language models for communication games: An empirical study on werewolf. *Preprint*, arXiv:2309.04658.

Zelai Xu, Chao Yu, Fei Fang, Yu Wang, and Yi Wu. 2024. Language agents with reinforcement learning for strategic play in the werewolf game. *Preprint*, arXiv:2310.18940.

Yunpu Zhao, Rui Zhang, Wenyi Li, Di Huang, Jiaming Guo, Shaohui Peng, Yifan Hao, Yuanbo Wen, Xing Hu, Zidong Du, et al. 2024. Assessing and understanding creativity in large language models. *arXiv preprint arXiv:2401.12491*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. 2024. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36.

Zhiling Zheng, Oufan Zhang, Ha L Nguyen, Nakul Rampal, Ali H Alawadhi, Zichao Rong, Teresa Head-Gordon, Christian Borgs, Jennifer T Chayes, and Omar M Yaghi. 2023. Chatgpt research group for optimizing the crystallinity of mofs and cofs. *ACS Central Science*, 9(11):2161–2170.

A BERTScore as a Judge

Following Song et al. (2021), where semantic distance is used for evaluating machine translation, BERTScore is calculated between all true definitions of each word in the "Judge Evaluation Data" and its respective deceiving definition. The maximum F_1-score of the calculated BERTScores for each word is used. For 60% of the words in the "Judge Evaluation Data," the maximum F_1-score of

the true definition is higher than that of the deceiving definition. However, it is not possible to define a clear threshold for semantic equivalence using BERTScore, as scores for both true and deceiving definitions overlap, ranging between 0.8 and 0.9. Therefore, it was not possible to use BERTScore output for our judge component.

B Computational Resources

All the experiments were conducted on three `NVIDIA RTX A6000` GPUs with 48GB VRAM. In total, all the experiments took around 8 hours to complete.

C Game Engine Technical Details

This platform supports using different sets of LLMs, each with different chat templates, GPU mappings, and parameters such as temperature or maximum number of new tokens to generate. These LLMs can be assigned as individual players or the game judge, or a single LLM can represent all game roles.

We use Huggingface's transformers (Wolf et al., 2020) and accelerate (Gugger et al., 2022) libraries to load all local, open-source models in our game engine. This allows for loading variations of these models, such as different quantizations, or completely changing them by simply modifying the model names in the game engine configuration. For API-based models, such as `GPT`, we use OpenAI's python library. Other OpenAI models can be used by changing the model names in the game engine configurations.

D Detailed Prompts

User Prompt

```
define the word {word} in one sentence.
```

Figure 2: The prompt for generating a correct definition of words to create the "Judge Evaluation Data".

User Prompt

```
Suppose you are playing Balderdash, and you want to deceive your opponents with a deceiving definition. define {word} in one sentence.
```

Figure 3: The prompt for generating deceiving definitions of words to create the "Judge Evaluation Data".

System Prompt

```
You are a universal dictionary. Your task is to provide the definition of a word given to you along with its part of speech. Use at most one sentence.
```

User Prompt

```
{word} ({pos}):
```

Figure 4: The prompts for generating definitions to create the "Known Balderdash Words" dataset.

System Prompt

```
You are a semantical equivalence judge. Your task is to determine whether a given definition is semantically equal to the actual definition of a word. Give your answer as a single word, either 'true' or 'false'.
```

User Prompt

```
Actual definition: {definition} and generated definition: {llm_definition}. Your judgment:
```

Figure 5: The prompts for semantical equivalence judgment used to create the "Known Balderdash Words" dataset.

Judge - System Prompt

```
Please act as a judge and evaluate whether a reference dictionary definition of a word and a given definition by an assistant capture the same core concept. Consider the essence and key components in the reference dictionary definition. You will be given a word, its reference dictionary definition, and an assistant's definition of that word. Your job is to assess if the assistant's definition captures the essential meaning or core concept of the reference dictionary definition, even if there are minor differences in detail or scope. Begin your evaluation by comparing the assistant's definition with the reference dictionary definition. Maintain objectivity and ensure that the order in which the definitions were presented does not influence your decision. Do not allow the length of the definitions to influence your evaluation. Output your final verdict by strictly following this format: "true" if the assistant's definition captures the core concept of the reference dictionary definition, and "false" if the assistant's definition fails to capture the core concept of the reference dictionary definition.
```

Judge - User Prompt

```
You receive the word: "{word}", the reference dictionary definition: "{correct_definition}", and assistant's definition: "{definition}". Give your answer as a single word, either "true" or "false". Your verdict is:
```

Figure 6: The prompts for the semantical equivalence judge, used in the game engine.

Generate Definition - User Prompt

```
Please act as a competitive player in the Balderdash game and write a definition for the word "{word}". You should consider the game's scoring rules and the history of your performance in previous rounds. Begin by elaborating on this information and choose the strategy for writing your definition to maximize your points. Avoid explaining your strategy and only write your definition. Avoid rewriting the given word at the beginning of your definition. Use at most one sentence. Output your definition between double quotes strictly following this format.
```

Generate Definition (No History Version) - User Prompt

```
Please act as a competitive player in the Balderdash game and write a definition for the word "{word}". You should consider the game's scoring rules. Begin by elaborating on this information and choose the strategy for writing your definition to maximize your points. Avoid explaining your strategy and only write your definition. Avoid rewriting the given word at the beginning of your definition. Use at most one sentence. Output your definition between double quotes strictly following this format.
```

Figure 7: The generate definition prompt, in two versions for asking the player to generate a definition for a given word based on the game rules, concatenated with the optional history prompt.

Full History - User Prompt

Your performance history in the previous rounds will be provided, helping you to learn from past performances, better understand your opponents' behaviors, and adapt your strategies to maximize your scoring potential in future rounds. History is provided in CSV format between triple backticks. Columns descriptions of the CSV: round_id: The id for the corresponding round. rank_among_players: An integer indicating your rank among all players up to that round. score: An integer indicating your score in that round. word: The announced word in that round. definition: The reference dictionary definition of the announced word. generated_definition: Your definition for the announced word. wrote_true_definition: A boolean showing whether the reference dictionary definition of your definition captures the same core concept. If the value of this column is True, you have not participated in the voting phase on that round, and thus, the 'guessed_correct_definiton' column will be False. guessed_correct_definiton: A boolean showing whether you have correctly guessed the reference dictionary definition in the voting phase. deception_ratio: The ratio of players who voted to your definition excluding yourself in the voting phase divided by the total number of players who participated in the voting phase. If the 'wrote_true_definition' is True, then this value will be -1. round_winners_strategies: A list of tuples containing the definition and that definition's outcome for each of the player(s) who got the highest scores in the corresponding round, in the format of [(definition_round_id, outcome_for_definition_round_id)]. "' {history_csv} "'

Mini History - User Prompt

Your performance history in the previous rounds will be provided, helping you to learn from past performances, better understand your opponents' behaviors, and adapt your strategies to maximize your scoring potential in future rounds. History is provided in CSV format between triple backticks. Columns descriptions of the CSV: round_id: The id for the corresponding round. rank_among_players: An integer indicating your rank among all players up to that round. score: An integer indicating your score in that round. word: The announced word in that round. generated_definition: Your definition for the announced word. "' {history_csv} "'

Figure 8: The full and mini history prompts, used for providing the performance history of each player in the game engine.

Game Rules - System Prompt

```
Please act as a competitive player in the Balderdash game. In each round of the game, a rare and unusual word will be given to all players. The players then write down a definition, which may be an honest attempt to supply the reference dictionary definition or, if they do not know or, for tactical reasons, decide not to, a fictitious definition for the word designed to sound convincing. Players submitting a definition that is semantically equal to the reference dictionary definition are immediately awarded {correct_definition_points} points, and they will not continue playing on that round. Then, the remaining definitions, including the reference dictionary definition, are given to each player in random order. Then, players write which definition they believe is the reference dictionary definition. Players are awarded {correct_vote_points} points if they guess the correct definition. Players are awarded {receiving_vote_points} points for each other player who incorrectly chooses the fake definition they wrote. Your goal is to maximize your points in each round by selecting the best strategy in writing a definition for the word and in the voting phase. You will be given a history of the previous rounds. Use the information in the history and pay attention to the scoring rules to choose the best strategy.
```

Game Rules (No History Version) - System Prompt

```
Please act as a competitive player in the Balderdash game. In each round of the game, a rare and unusual word will be given to all players. The players then write down a definition, which may be an honest attempt to supply the reference dictionary definition or, if they do not know or, for tactical reasons, decide not to, a fictitious definition for the word designed to sound convincing. Players submitting a definition that is semantically equal to the reference dictionary definition are immediately awarded {correct_definition_points} points, and they will not continue playing on that round. Then, the remaining definitions, including the reference dictionary definition, are given to each player in random order. Then, players write which definition they believe is the reference dictionary definition. Players are awarded {correct_vote_points} points if they guess the correct definition. Players are awarded {receiving_vote_points} points for each other player who incorrectly chooses the fake definition they wrote. Your goal is to maximize your points in each round by selecting the best strategy in writing a definition for the word and in the voting phase. Pay attention to the scoring rules to choose the best strategy.
```

Figure 9: The game rules prompt, in two versions for providing instructions on playing the game and the game's scoring rules to each player used in the game engine.

Vote on Definitions - User Prompt

```
Please act as a competitive player in the Balderdash game and choose the reference dictionary definition index. You will be given the word for this round, your given definition, and the other definitions, excluding the definitions that were semantically equal to the reference dictionary definition, including the reference dictionary definition in random order. You should consider the game's scoring rules and the history of your performance in previous rounds. Begin by elaborating on this information and choose the reference dictionary definition, which will maximize your points. Avoid explaining your strategy. Choose your vote among the allowed choice(s) and only write your vote. Your definition for "{word}" was "{definition}". All definitions, including the reference dictionary definition, are given to you in the format: 1. definition_1 2. definition_2 3. definition_3 Definitions: “‘ {definitions} “‘ As a Balderdash player, your task is to choose the reference dictionary definition index {all_indexes_excluding_player_descriptive} and write it without any explanation. Your allowed choice(s): {all_indexes_excluding_player} Use at most one character, which is a single digit.
```

Vote on Definitions (No History Version) - User Prompt

```
Please act as a competitive player in the Balderdash game and choose the reference dictionary definition index. You will be given the word for this round, your given definition, and the other definitions, excluding the definitions that were semantically equal to the reference dictionary definition, including the reference dictionary definition in random order. You should consider the game's scoring rules. Begin by elaborating on this information and choose the reference dictionary definition, which will maximize your points. Avoid explaining your strategy. Choose your vote among the allowed choice(s) and only write your vote. Your definition for "{word}" was "{definition}". All definitions, including the reference dictionary definition, are given to you in the format: 1. definition_1 2. definition_2 3. definition_3 Definitions: “‘ {definitions} “‘ As a Balderdash player, your task is to choose the reference dictionary definition index {all_indexes_excluding_player_descriptive} and write it without any explanation. Your allowed choice(s): {all_indexes_excluding_player} Use at most one character, which is a single digit.
```

Figure 10: The vote on definitions prompt, in two versions for asking the player to choose the reference dictionary definition for a word among all given definitions during the voting phase, concatenated with the optional history prompt.