Language Agents with Reinforcement Learning for Strategic Play in the Werewolf Game

Zelai Xu [1] **Chao Yu** [1] **Fei Fang** [2] **Yu Wang** [*1] **Yi Wu** [*1 3]

Abstract

Agents built with large language models (LLMs) have shown great potential across a wide range of domains. However, in complex decision-making tasks, pure LLM-based agents tend to exhibit intrinsic bias in their choice of actions, which is inherited from the model's training data and results in suboptimal performance. To develop *strategic language agents*, i.e., agents that generate flexible language actions and possess strong decision-making abilities, we propose a novel framework that powers LLM-based agents with reinforcement learning (RL). We consider Werewolf, a popular social deduction game, as a challenging testbed that emphasizes versatile communication and strategic gameplay. To mitigate the intrinsic bias in language actions, our agents use an LLM to perform deductive reasoning and generate a diverse set of action candidates. Then an RL policy trained to optimize the decision-making ability chooses an action from the candidates to play in the game. Extensive experiments show that our agents overcome the intrinsic bias and outperform existing LLM-based agents in the Werewolf game. We also conduct human-agent experiments and find that our agents achieve human-level performance and demonstrate strong strategic play.

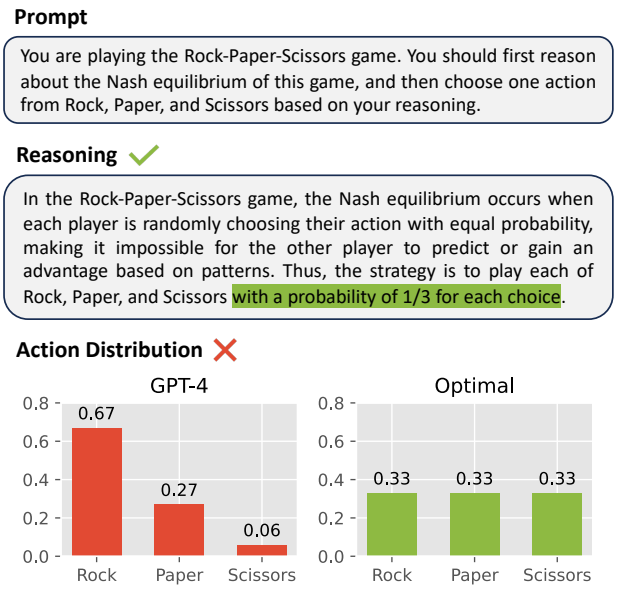


Figure 1: The intrinsic bias in the action distribution of pure LLM-based agents despite the correct reasoning result.

1. Introduction

Building agents that are capable of logical reasoning, strategic decision-making, and communication with humans has been a longstanding aspiration in the field of AI (Wooldridge & Jennings, 1995; Goodwin, 1995). With their extensive knowledge and emergent generalizability, large language models (LLMs) have shown great potential in constructing intelligent agents and have led to many recent advancements (Ouyang et al., 2022; Wei et al., 2022a). These LLM-based agents demonstrate proficiency in a wide range of tasks including web surfing (Nakano et al., 2021; Yao et al., 2022b; Zheng et al., 2024), complex video games (Wang et al., 2023a; Ma et al., 2023), and real-world applications (Ahn et al., 2022; Shah et al., 2023). Moreover, in multi-agent scenarios, LLM-based agents exhibit the ability to produce human-like interactions (Park et al., 2023; Williams et al., 2023), achieve zero-shot cooperation (Li et al., 2023; Chen et al., 2023), and compete with adversarial players (Meta et al., 2022; Wang et al., 2023b).

Despite the impressive achievements, we observe that in complex decision-making tasks such as multi-agent games, pure LLM-based agents built with prompting techniques tend to exhibit intrinsic bias in their choice of actions, resulting in suboptimal performance. Consider a simple example where we use GPT-4 to play the Rock-Paper-Scissors game. To generate the optimal strategy, we prompt the LLM to

[*]Equal advising [1]Tsinghua University, Beijing, China [2]Carnegie Mellon University, Pittsburgh, United States [3]Shanghai Qi Zhi Institute, Shanghai, China. Correspondence to: Zelai Xu <zelai.eecs@gmail.com>, Yi Wu <jxwuyi@gmail.com>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

first reason about the Nash equilibrium of this game and then choose an action based on the reasoning result. We let this pure LLM-based agent play 100 independent games and profile its reasoning result and action distribution in Fig 1. In all 100 games, the agent successfully identifies the optimal strategy as randomly choosing an action. However, even with this perfect reasoning result, the agent's actual action distribution shows a clear bias toward choosing Rock, which can be easily exploited by an opponent who always plays Paper. This intrinsic bias is inevitably inherited from the model's pretraining data and hinders pure LLM-based agents from strong strategic play in more complex scenarios. A recent work, Cicero (Meta et al., 2022), addresses this issue in the game of Diplomacy by learning a policy to choose from predefined actions that specify the agents' intents and then generating action-conditioned dialogues with a language model. Nevertheless, the action space of their policy is a fixed, finite set of game-specific actions, while real-world interactions between humans often feature free-form conversations with nuanced intents, which induces an unbounded action space for agents to play.

In this work, we aim to develop *strategic language agents*, i.e., agents that generate flexible language actions and possess strong decision-making abilities. We consider the Werewolf game as a challenging mixed cooperative-competitive multi-agent testbed. Werewolf is one of the most popular social deduction games where two teams of players with hidden roles interact in natural language to uncover and defeat their opponents. During gameplay, the Werewolves need to lie about their roles and secretly eliminate other players, while the Villagers need to share information and vote out the hidden Werewolves. The game's high demand for language proficiency necessitates the use of LLMs to build intelligent agents. However, pure LLM-based agents face the challenges of deducing hidden roles with deceptive information and overcoming intrinsic bias in language actions, which result in unsatisfactory performance.

To build strategic language agents for the Werewolf game, we propose a novel framework that powers LLM-based agents with reinforcement learning (RL) to address these challenges. Our framework consists of three components. The first component is hidden role deduction, which uses an LLM to distinguish between truths and lies and explicitly deduce the hidden role of each player to facilitate subsequent decision-making. The second component is diverse action generation, which prompts the LLM to generate a diverse set of action candidates instead of a single action to mitigate the intrinsic bias. The last component is population-based RL training, which learns an RL policy to optimize the distribution over the action candidates and improves policy robustness by playing against a population of various agents. The combination of the LLM and the RL policy enables our agents to deduce hidden information and overcome intrinsic bias, leading to strong strategic play in the Werewolf game.

We conduct extensive experiments to evaluate the performance of strategic language agents in Werewolf and demonstrate the benefit of our design. We first perform case studies and visualize the action distributions to validate that our agents overcome the intrinsic bias and learn better policy. Then we compare our agents with four state-of-the-art baselines in a round-robin tournament where our agents achieve the highest win rate against all agents. Moreover, we let our agents play with human players and find they achieve comparable win rates to average humans as both teammates and opponents. We also conduct systematic ablations to show the effectiveness of the key components in our framework.

2. Related Work

Building Agents with Large Language Models. There is a recent trend in developing agents with large language models for various domains including website scenarios (Nakano et al., 2021; Yao et al., 2022a; Deng et al., 2023), game and simulation (Huang et al., 2022a; Wang et al., 2023c;a; Ma et al., 2023), real-world embodiment (Ahn et al., 2022; Huang et al., 2022b; Vemprala et al., 2023), and multi-agent environments (Park et al., 2023; Li et al., 2023; Chen et al., 2023). A shared foundation of these works is to utilize the reasoning and in-context learning ability of LLMs to facilitate decision-making. Chain-of-Thought (CoT) (Wei et al., 2022b) is perhaps the most well-known work that unlocks the reasoning ability of LLMs by asking them to think step-by-step. To synergize reasoning and acting, ReAct (Yao et al., 2022b) uses LLMs to generate both reasoning traces and action plans and proves effective in various benchmarks. Follow-up work continues to improve performance by incorporating self-reflection (Shinn et al., 2023) and strategic reasoning (Gandhi et al., 2023). However, even with the perfect reasoning results, these pure LLM-based agents could suffer from the model's intrinsic bias and make suboptimal decisions in more complex scenarios. Our work takes a step further to address this issue by using the LLM to generate a diverse set of action candidates with minimal bias and training an RL policy to optimize decision-making.

Some other works also augment LLMs with external modules to enhance the agents' decision-making ability. One representative work is Cicero (Meta et al., 2022) which combines LLMs with RL and achieves human-level play in the game of Diplomacy. The main difference from our work is that their policy can choose from a fixed finite set of game-specific actions because Diplomacy is a board game with negotiation. By contrast, Werewolf is a language game without predefined actions, requiring our agents to generate language actions in an unbounded space. Another work is Tree-of-Thought (ToT) (Yao et al., 2023) which generalizes CoT by producing multiple thoughts at each reasoning step

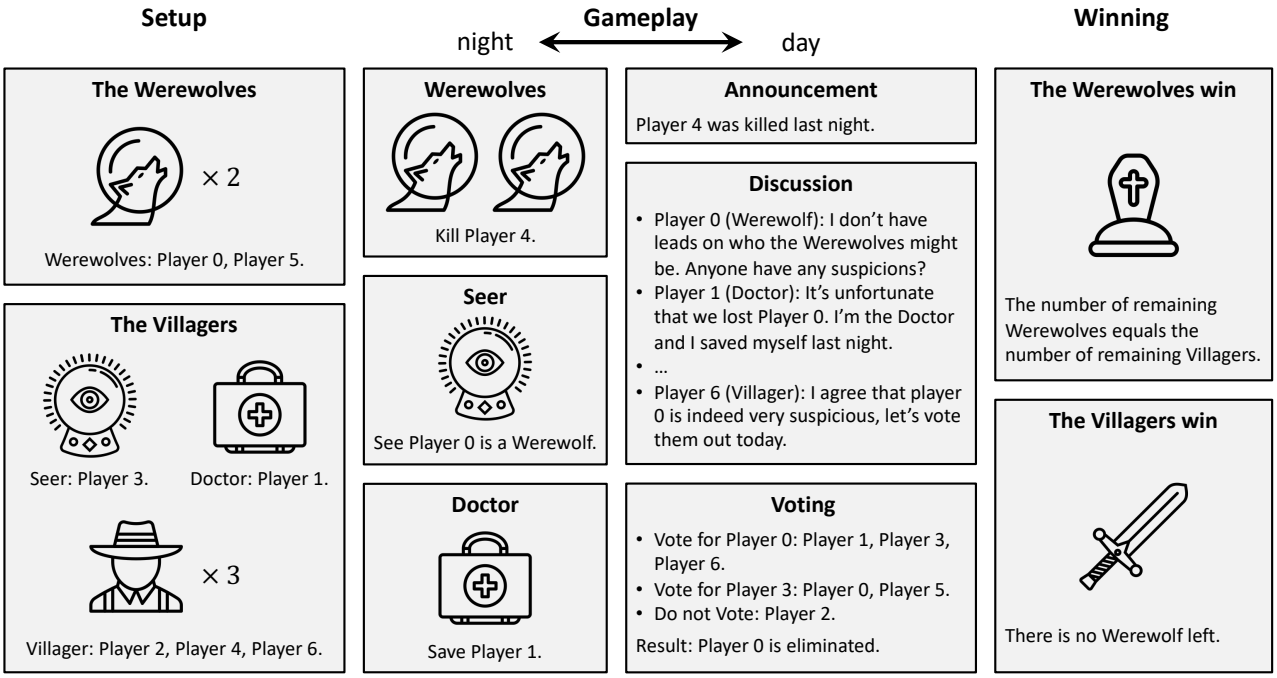


Figure 2: An example of the Werewolf game. Players are randomly assigned a hidden role and divided into the Werewolves and the Villagers. The game alternates between night and day rounds until one team achieves the winning condition.

to create a tree structure and search for optimal plans. The extensive tree search could be inefficient in complex environments with prohibitively large or even infinite game trees. LLM+P (Liu et al., 2023) incorporate classic planners by converting natural language task descriptions into planning domain definition language (PDDL). However, PDDL is limited to classic planning problems and cannot be applied to non-cooperative language games like Werewolf.

Reinforcement Learning in Non-Cooperative Games. Applying reinforcement learning to non-cooperative games has achieved great success in the game of Go (Silver et al., 2016; 2018), poker (Moravčík et al., 2017; Brown & Sandholm, 2018; 2019), and video games (Vinyals et al., 2019; Berner et al., 2019). The most popular method underlying these achievements is self-play and its variants (Heinrich et al., 2015; Heinrich & Silver, 2016; Hennes et al., 2020; Xu et al., 2023b), which learn a policy by training against itself and its past checkpoints. Population-based training (PBT) methods like policy-space response oracles (PSRO) (Lanctot et al., 2017; Muller et al., 2019) and league training (Vinyals et al., 2019) generalize self-play by maintaining a pool of various policies and training against the population. Another notable line of work is based on regret minimization techniques such as counterfactual regret minimization (CFR) (Zinkevich et al., 2007; Lanctot et al., 2009; Tammelin, 2014; Brown et al., 2019) that iteratively refine the strategies by minimizing the difference between actual outcomes and potential better choices. Most prior works apply these methods to typical RL environments with vectorized actions, while we combine LLM with RL to generate language actions and learn in the Werewolf game.

AI Agents for Social Deduction Games. Social deduction games like Werewolf require both strong communication skills and strategic decision-making. Earlier agents (Wang & Kaneko, 2018) lack language proficiency and usually rely on predefined protocols instead of natural language to play a simplified game. DeepRole (Serrino et al., 2019) integrates deductive reasoning into CFR to play another social deduction game named Avalon without communication. With the presence of LLMs, more agents have been developed to play these games in natural language. A concurrent study (Xu et al., 2023a) builds a Werewolf agent by heuristic information retrieval and past experience reflection. Another work proposes ReCon (Wang et al., 2023b) to play Avalon by thinking from both the agent's and the opponents' perspective. Despite their deliberate designs, these pure LLM-based agents are not strong enough in decision-making as they still face the issue of intrinsic bias. We use them as baselines to compare with our agents in the experiments.

3. The Werewolf Game

We aim to build strong agents for the seven-player Werewolf game with two Werewolves, one Seer, one Doctor, and three

Villagers. An example of the game is shown in Fig. 2 and the detailed rules can be found in Appendix A.

3.1. Text-Based Environemnt

We implement a pure text-based Werewolf environment that does not consider external factors like the players' tone or facial expressions. The observations and actions are text in natural language and the game proceeds as follows.

Setup. At the beginning of the game, each player is randomly assigned a hidden role, which divides them into the Werewolves and the Villagers (Seer, Doctor, and Villagers). The two Werewolves know each other's role and hence also know which players are the Villagers. Their goal is to "kill" the other players without being discovered. The Villagers do not know the hidden roles of other players and their goal is to identify and eliminate the Werewolves.

Gameplay. The game alternates between night and day rounds, starting with the night. In the night round, everyone closes their eyes to let the Werewolves, Seer, and Doctor take secret actions. The Werewolves choose one player to kill, the Seer chooses one player to check if they are a Werewolf, and the Doctor chooses one player to save without knowing who is the target of the Werewolves. If the Doctor chooses the same player as the Werewolves, the player is successfully saved and no one is killed in the night round. Otherwise, the player is eliminated from the game.

In the day round, an announcement is first made to every player about who was killed or no one was killed last night. Then the remaining players take turns to speak in a discussion about who might be the Werewolves. Players can claim or lie about their roles, share or withhold information they have discovered, and accuse or defend other players to achieve their purposes. After all players have participated in the discussion, a vote is held to eliminate one suspicious player. Each player can vote for one player or choose not to vote, and the player with the most votes is eliminated. Then the game continues to the next night round until the Werewolves or the Villagers win the game.

Winning. The Werewolves win if the number of remaining Werewolves is equal to the number of remaining Villagers. The Villagers win if both Werewolves are eliminated.

The observation of each player is a list of information in natural language that describes the current game history. This includes their ID, hidden role, secret actions at night (if any) as well as the announcements, discussions, and voting results in the day rounds. Werewolves also have their teammate's ID and secret actions in the observation. The players' actions are also in natural language and can be divided into three types. The first type is secret actions at night including killing, seeing, and saving that choose a specific player as the target. The second type is statement actions during discussion that share information and convey the player's opinion. The last type is voting actions that vote for one surviving player or choose not to vote.

3.2. Challenges for LLM-Based Agents

Werewolf is a challenging mixed cooperative-competitive game that involves deceptive communication and strategic gameplay. Its high demand for language proficiency necessitates the use of LLMs to build intelligent agents. We identify two major challenges of Werewolf that pose higher requirements on the ability of LLM-based agents.

The first challenge is to deduce hidden information from deceptive communication. With the existence of unknown opponents, agents could be misled by manipulative statements and fail to make accurate judgments about other players. This requires the agents to distinguish between truths and lies and deduce the hidden roles of other players.

The second challenge is to overcome the intrinsic bias in LLMs and optimize decision-making. As illustrated by the Rock-Paper-Scissor example in Fig 1, pure LLM-based agents could have suboptimal biases in their action distributions, hindering their ability to make stronger decisions. This issue becomes more pronounced in competitive games like Werewolf because opponents can exploit the biased distributions to predict the agents' behaviors and counteract to gain advantages. Therefore, agents must overcome intrinsic bias to reduce exploitation and achieve strong performance.

4. Strategic Language Agents

To address the aforementioned challenges, we propose a novel framework that powers LLM-based agents with RL training to build agents that generate flexible language actions and possess strong decision-making ability, which we call *strategic language agents*. As shown in Fig. 3, our agents consist of three components: hidden role deduction, diverse action generation, and population-based RL training.

4.1. Hidden Role Deduction

The first component aims to improve our agents' deductive reasoning ability in the presence of deceptive information. we use an LLM to convert the raw observation into an organized information record and a structured deduction result. The information record extracts key information and distinguishes between truthful and deceptive statements, while the deduction result infers the hidden role of each player and rates their reliability. These structured and informative data serve as inputs for the next two components and lay the foundation for subsequent decision-making.

Concretely, the information record categorizes the raw observation, which is a list of information in natural language,

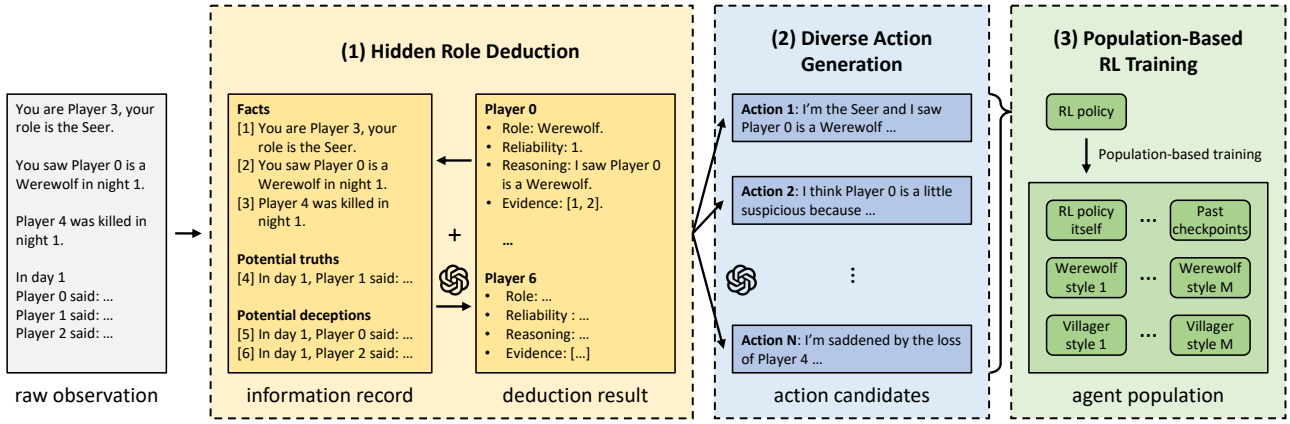


Figure 3: Overview of strategic language agents. (1) Hidden role deduction: categorize key information and deduce hidden roles with the LLM to facilitate subsequent decision-making. (2) Diverse action generation: prompt the LLM for a diverse set of action candidates to mitigate intrinsic bias. (3) Population-based RL training: learn a separate RL policy to optimize the distribution over action candidates and enhance decision-making by playing against a population of various agents.

into three types including facts, potential truths, and potential deceptions. All information except the statements in discussion is categorized as facts, which encompass the player's role, secret actions, announcements, and voting results. The statements are categorized according to the speaker's reliability in the deduction result. In our implementation, the reliability is rated on a scale from 1 to 10. If a player's reliability is larger than 6, their statements are regarded as potential truths, otherwise as potential deceptions.

With the organized information record, we then prompt the LLM to deduce the hidden roles of other players. For each player, the LLM is asked to generate four attributes including role, reliability, reasoning, and evidence. The role attribute infers the most likely hidden role of the player and the reliability attribute rates the credibility of their statements from 1 to 10. The reasoning attribute is an auxiliary one that explicitly shows the deduction process. The last evidence attribute is a list of integers that cite items from the information record to support the current deduction. The evidences are used to extract key information that contributes to the deduction of hidden roles. If a statement is never cited as evidence, it is regarded as an uninformative item and removed from the information record. More detailed designs and prompts can be found in Appendix B.

4.2. Diverse Action Generation

To mitigate the intrinsic bias of LLMs and reduce exploitation from adversarial players, agents should be able to consider a broader range of reasonable actions instead of a single biased action. We propose the diverse action generation component to prompt the LLM for a set of action candidates with strategic diversity. More specifically, given the information record and deduction result as input, we consider two approaches to generate N action candidates.

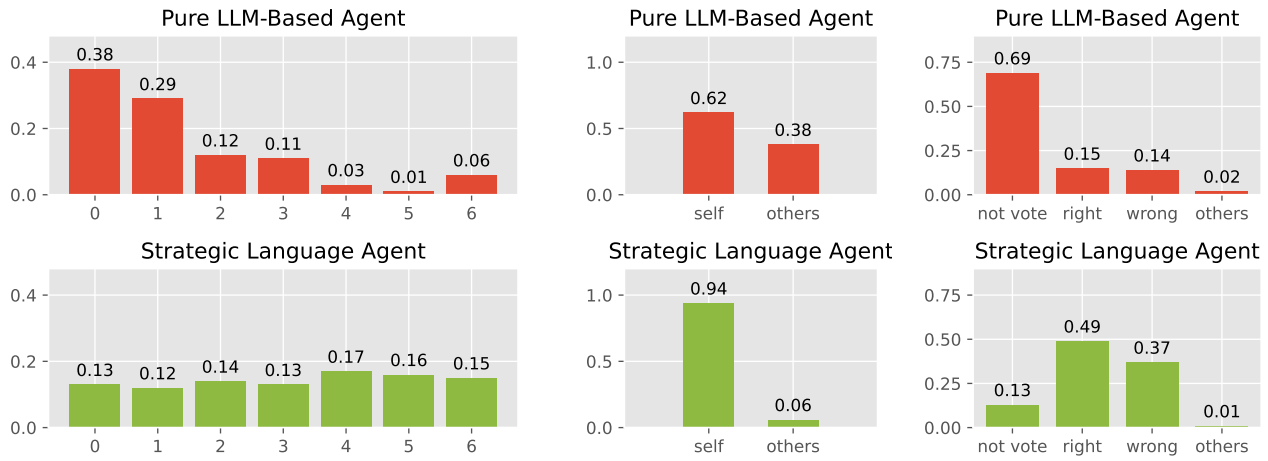
Vanilla Prompting. The first approach generates all action candidates in a single inference by asking the LLM to propose N diverse action candidates that correspond to different strategies. This approach is simple yet effective for easier actions like secret actions and voting actions, which only need to choose a player as the target.

Iterative Prompting. The second approach prompts the LLM for N times and iteratively generates one action candidate at a time. In each inference, we add the existing action candidates to the input and ask the LLM to propose a new action candidate that is strategically different from existing ones. By having more interactions with the LLM, this approach is empirically found to produce better performance for more complex actions like statement actions.

In our implementation, we use the first approach to generate secret actions and vote actions for fewer inferences, and the second approach to generate statement actions for higher quality. We also ask the LLM to output their reasoning for each action candidate for interpretability and better performance. The detailed prompts can be found in Appendix B.

4.3. Population-Based RL Training

With the diverse set of action candidates, agents can avoid fixed patterns and choose from a variety of actions to take. Although random sampling already leads to unpredictable play, the optimal policy in most cases is a non-uniform distribution over the action candidates. Directly fine-tuning the LLM to refine the distribution requires a substantial amount of data and computation. We take an alternative approach that efficiently learns a smaller RL policy to optimize the action distribution and enhance decision-making ability by



(a) Werewolf first night action. (b) Doctor first night action. (c) Villager voting action.

Figure 4: Comparison of the action distributions of the pure LLM-based agent and our strategic language agent.

playing against a population of various agents.

The main difference between our setting and classic RL environments is that the action space of our policy is a set of language actions generated during the game. Because the action space is not predefined, we cannot use typical policy networks that only take the observation as input and produce a distribution over the fixed action set. Instead, we adopt a self-attention network (Vaswani et al., 2017) that takes the observation and all action candidates as input to produce the action distribution. Concretely, the observation and action candidates in natural language are converted to vector embeddings using the LLM embedding API. We also use a vector to include player-specific information like their ID, role, etc. and pass it through an encoder to produce a player embedding of the same dimension. The player embedding, observation embedding, and action candidate embeddings are passed through a residual self-attention block, and the probability to sample an action candidate is proportional to the scaled dot-product attention between the observation embedding and action candidate embedding.

To achieve strong and robust performance with various teammates and opponents, we train the RL policy with a population of diverse agents. This is because real-world games are usually non-transitive (Czarnecki et al., 2020), i.e., the policies may cycle like Rock-Paper-Scissors, and population-based training can help agents get over the endless cycles and achieve a higher level of play. Our agent population consists of two types of policies. The first type is the RL policy itself and its past checkpoints. Training with these policies resembles self-play and fictitious play. The second type is pure LLM-based agents with predefined play styles, which introduce more strategic diversity. These agents only have the first hidden role deduction component and generate one action according to their predefined styles. In our implementation, we consider three common styles for the Werewolves and three styles for the Villagers. More implementation and training details can be found in Appendix C.

5. Experiments

We conduct experiments from four different aspects to comprehensively evaluate the performance of strategic language agents. We first perform case studies and visualize the action distributions of our agents to check if they address the intrinsic bias issue. Then we compare our agents with four existing LLM-based agents in a round-robin tournament to assess their decision-making ability. Furthermore, we conduct human-agent experiments to evaluate their performance with and against human players. We also perform ablation studies to investigate the contribution of the key components in our design. More experiments, discussions, and emergent behaviors can be found in Appendix D, E, F.

5.1. Case Studies

To validate that our method overcomes the intrinsic bias in LLMs, we investigate three typical situations in the Werewolf game and compare the action distributions of our agents with a pure LLM-based agent that only has the hidden role deduction component and follows *ReAct* (Yao et al., 2022b) to generate one single action to play.

Werewolf First Night Action. On the first night, the Werewolves need to choose a player to kill without any information. Their optimal policy is to randomly choose a player other than themselves so that the Doctor cannot predict and save the victim. However, as shown in Fig. 4(a), the pure LLM-based agent has a clear bias toward killing Player 0,

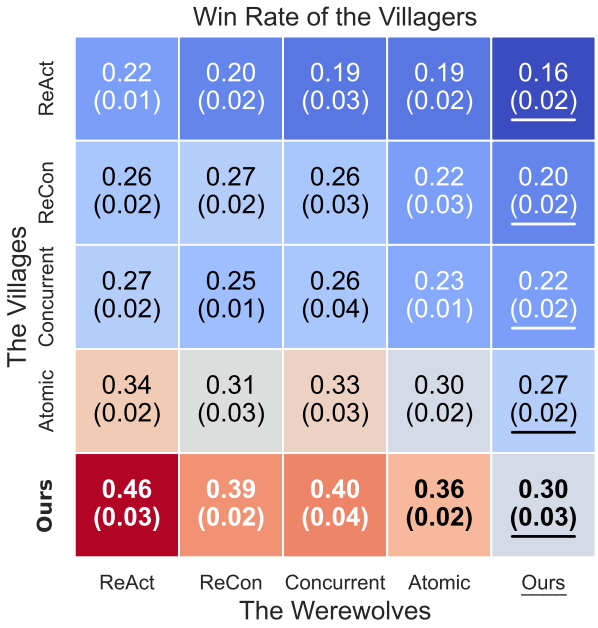


Figure 5: Win rate matrix of the round-robin tournament.

which could be exploited by a Doctor who always saves Player 0. By contrast, our agents produce an almost uniform distribution which is close to the optimal policy.

Doctor First Night Action. The Doctor also needs to choose a player to save without knowing who is the victim on the first night. From the Doctor's perspective, any player other than themselves could be a Werewolf, and saving others may waste the action on a Werewolf. Therefore, the optimal policy for the Doctor is to always save themselves on the first night. As shown in Fig. 4(b), the pure LLM-based agent exhibits a suboptimal policy that saves others with a probability of 0.38, while our agents learn to save themselves with a high probability of 0.94.

Villager Voting Action with Two Self-Proclaimed Seers. Another common situation is when two players claim to be the Seer and the Villagers need to choose their actions in the voting phase. The pure LLM-based agent tends to make conservative decisions and chooses not to vote with a high probability of 0.69, as shown in Fig 4(c). However, one of the self-proclaimed Seers must be a Werewolf and not voting for any player only makes it easier for the Werewolves to control the voting result and eliminate the real Seer. In comparison, our agents have a much lower probability for choosing not to vote and learn to identify and vote out the right Werewolf who pretends to be the Seer.

5.2. Round-Robin Tournament

We evaluate the performance of our agents in the Werewolf game by playing against existing LLM-based agents. Because there is only one agent (Xu et al., 2023a) that works out of the box for Werewolf, we make our best effort to select four representative methods as baselines and make them compatible with our environment. We implement the *ReAct* (Yao et al., 2022b) agent by directly prompting the LLM with the raw observation to generate both reasoning and action. *ReCon* (Wang et al., 2023b) is originally designed for Avalon and generates actions by planning from their own perspective and refining the plan from others' perspective. The *Concurrent* agent (Xu et al., 2023a) considers a different version of the Werewolf game and employs heuristic information retrieval and reflection on past experiences. We make necessary changes to the prompts of *ReCon* and *Concurrent* to apply them in our setting. The *Atomic* agent is inspired by Cicero (Meta et al., 2022) and has a predefined set of 13 atomic actions like "target Player 0", "claim to be the Seer", etc. This agent learns an RL policy to select an atomic action based on the raw observation and then prompts the LLM to follow the atomic action and generate the language action used in actual gameplay.

We let our agents and these four baseline agents participate in a round-robin tournament that runs head-to-head competitions between each pair of agents. For a pair of agents (A, B), we run 100 Werewolf games with five A agents being the Villagers and two B agents being the Werewolves. This leads to a 5×5 cross-play matrix in Fig. 5 that records the win rates of the Villagers. For example, the value in the first row and last column corresponds to the win rate of *ReAct* when it plays as the Villagers against our agents as the Werewolves. The row in the matrix represents the agent's performance as the Villagers against different opponents, and a row with higher (red) values means stronger performance as the Villagers. As shown by the bold numbers in the last row, our agents achieve the highest win rates as the Villagers against all agents. Similarly, the column represents the agent's performance as the Werewolves and smaller (blue) values mean lower win rates of their opponents, indicating stronger performance as the Werewolves. The last column with underlined numbers shows that our agents also achieve the best performance as the Werewolves.

The strong decision-making ability of our agents comes from the combination of an LLM to generate flexible language actions and an RL policy to overcome intrinsic bias. The *React*, *ReCon*, and *Concurrent* agents directly generate language actions with an LLM. Even if the latter two have specific designs for social deduction games, they still face the intrinsic bias issue and have unsatisfactory performance. On the other hand, although the *Atomic* agent also combines an LLM with an RL policy, its language generation is conditioned on the predefined action set which is hard, if not impossible, to cover the unbounded language space. More examples and analysis can be found in Appendix E.

Win Rate	Play with Six Copies of Our Agents			Play with Six Human Players		
	Pure LLM-Based	**Ours**	Human	Pure LLM-Based	**Ours**	**Human**
As the Villagers	0.25 (0.03)	**0.30 (0.03)**	0.28 (0.06)	0.29 (0.06)	0.36 (0.05)	**0.38 (0.07)**
As the Werewolves	0.61 (0.05)	**0.70 (0.03)**	0.67 (0.08)	0.54 (0.08)	**0.67 (0.06)**	0.66 (0.08)
Overall	0.42 (0.04)	**0.50 (0.03)**	0.48 (0.07)	0.42 (0.07)	**0.52 (0.06)**	**0.52 (0.08)**

Table 1: Win rate of one AI agent or human when playing with six copies of our agents and playing with six human players.

Win Rate	As the Villagers	As the Werewolves
Ours	**0.30 (0.03)**	**0.70 (0.03)**
- RL Policy	0.23 (0.03)	0.61 (0.04)
- Diversity	0.22 (0.04)	0.61 (0.05)
- Deduction	0.16 (0.02)	0.54 (0.04)

Table 2: Ablation on key components.

Win Rate	As the Villagers	As the Werewolves
Ours	**0.30 (0.03)**	**0.70 (0.03)**
- Evidence	0.28 (0.04)	0.67 (0.03)
- Reasoning	0.25 (0.03)	0.66 (0.05)
- Reliability	0.23 (0.02)	0.62 (0.04)

Table 3: Ablation on deduction attributes.

5.3. Human Evaluation

We further conduct human-agent experiments to evaluate the performance of our agents. Concretely, we compare our agents with human players and the pure LLM-based agent used in Section 5.1 in two setups of the seven-player Werewolf game. The first setup is to let one human or one AI agent play with six copies of our agents, and the second setup is to let one AI agent or one human play with six human players. We recruited 160 human players and randomly assigned each player to one of the two setups to play 10 consecutive games. The human players know who are the AI agents and the hidden roles are assigned randomly in each game. Additional details of the human evaluation procedure can be found in Apendix D.7.

We report the win rate of the one AI agent and one human player both as the Werewolves and as the Villagers in Table 1 and each entry is averaged over 100 games. In both setups, comparing the results in the "Pure LLM-Based" column and "Ours" column shows that our agents achieve higher win rates than the pure LLM-based agent when playing with six copies of our agents and when playing with six human players. This result validates that our framework to combine the LLM with the RL policy leads to stronger performance in the Werewolf game. In addition, when playing with six copies of our agents, the results in the "Ours" column and the "Human" column show that our agents achieve higher win rates than human players, both as the Werewolves and as the Villagers. More interestingly, when playing with six human players, our agents also achieve comparable win rates to human players. Note that our agents are not trained with real humans, this result shows the strong robustness of our agents to both cooperate with and compete against unseen human players and achieve human-level performance.

Win Rate	As the Villagers	As the Werewolves
Ours (PBT)	**0.30 (0.03)**	**0.70 (0.03)**
Self-Play	0.26 (0.04)	0.66 (0.02)

Table 4: Ablation on agent population.

5.4. Ablation Study

Key Components. We consider three ablated agents by gradually removing the three key components from our agents and evaluating their performance by playing against our agents. The "- RL policy" agent uses the LLM to choose from the action candidates instead of the RL policy. The "- Diversity" agent further removes the diverse action generation component and directly prompts the LLM to produce one action. The "- Deduction" agent removes all three components and directly generates an action based on the raw observation. The results in Table 2 show that all three components contribute to the improvement in performance. It is also worth noting that, although the "- RL policy" agent generates diverse action candidates, its win rates are comparable to the "- Diversity" agent. This is because using the LLM to choose from the action candidates still suffers from intrinsic bias. This issue is fully solved by adding the RL policy to optimize decision-making.

Deduction Attributes. We study the contribution of the four attributes in hidden role deduction by gradually removing the evidence, reasoning, and reliability attributes and evaluating against our agents. The results in Table 3 show that using all four attributes gives the best performance.

Agent Population. We also learn an agent using self-play instead of population-based training (PBT) and compare it with our agents. The results in Table 4 show that our agents achieve higher win rate both as the Villagers and as the Werewolf. This aligns with the results in other real-world games that introducing more diversity to the population makes the policy more robust (Czarnecki et al., 2020).

6. Conclusion

We propose a novel framework that combines an LLM with an RL policy to build strategic language agents that overcome the intrinsic bias of pure LLM-based agents. Our agents deduce hidden information by categorizing key information and inferring hidden roles with the LLM. To address the issue of intrinsic bias, our agents generate a diverse set of action candidates and learn an RL policy to choose from the action candidates. By powering LLM-based agents with an RL policy, our agents outperform existing LLM-based agents and achieve human-level play in the Werewolf game.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (No.62325405, U19B2019, M-0248), Tsinghua University Initiative Scientific Research Program, Tsinghua-Meituan Joint Institute for Digital Life, Beijing National Research Center for Information Science, Technology (BNRist), Beijing Innovation Center for Future Chips and 2030 Innovation Megaprojects of China (Programme on New Generation Artificial Intelligence) Grant No. 2021AAA0150000.

Impact Statement

While the intent of our work is to develop LLM-based agents with strong decision-making ability in the Werewolf games, it also raises important ethical concerns regarding AI agents' deceptions to human participants and the potential misuse of our approach. We strictly follow principles for ethical research in our study and make our best efforts to reduce any potential harm to participants in our human-AI experiment and the broader human society.

To ensure minimal negative impact on human participants in our study, we conduct the human-agent experiment with department approval and allow each participant to take part in the study after agreeing to the consent form. We mention explicitly in the consent form that the Werewolf game contains deceptive communications and the AI agents may have dishonest statements in the game. The participant can choose to stop participating at any time during the experiment.

We are also committed to prevent the misuse of our approach in ways that could harm human society. On the one hand, our implementation of strategic language agents focuses on the pure text-based environment of Werewolf and cannot be directly applied to other real-world situations. On the other hand, our experiment results show that our agents have stronger ability to identify and counteract deceptions than existing LLM-based agents. This make it possible to use our agents to help humans recognize manipulative content and prevent potential malicious use with harmful intent.

References

Ahn, M., Brohan, A., Brown, N., Chebotar, Y., Cortes, O., David, B., Finn, C., Fu, C., Gopalakrishnan, K., Hausman, K., et al. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022.

Berner, C., Brockman, G., Chan, B., Cheung, V., Debiak, P., Dennison, C., Farhi, D., Fischer, Q., Hashme, S., Hesse, C., et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019.

Brown, N. and Sandholm, T. Superhuman ai for heads-up no-limit poker: Libratus beats top professionals. *Science*, 359(6374):418–424, 2018.

Brown, N. and Sandholm, T. Superhuman ai for multiplayer poker. *Science*, 365(6456):885–890, 2019.

Brown, N., Lerer, A., Gross, S., and Sandholm, T. Deep counterfactual regret minimization. In *International conference on machine learning*, pp. 793–802. PMLR, 2019.

Chen, W., Su, Y., Zuo, J., Yang, C., Yuan, C., Qian, C., Chan, C.-M., Qin, Y., Lu, Y., Xie, R., et al. Agentverse: Facilitating multi-agent collaboration and exploring emergent behaviors in agents. *arXiv preprint arXiv:2308.10848*, 2023.

Czarnecki, W. M., Gidel, G., Tracey, B., Tuyls, K., Omidshafiei, S., Balduzzi, D., and Jaderberg, M. Real world games look like spinning tops. *Advances in Neural Information Processing Systems*, 33:17443–17454, 2020.

Deng, X., Gu, Y., Zheng, B., Chen, S., Stevens, S., Wang, B., Sun, H., and Su, Y. Mind2web: Towards a generalist agent for the web. *arXiv preprint arXiv:2306.06070*, 2023.

Gandhi, K., Sadigh, D., and Goodman, N. D. Strategic reasoning with language models. *arXiv preprint arXiv:2305.19165*, 2023.

Goodwin, R. Formalizing properties of agents. *Journal of Logic and Computation*, 5(6):763–781, 1995.

Heinrich, J. and Silver, D. Deep reinforcement learning from self-play in imperfect-information games. *arXiv preprint arXiv:1603.01121*, 2016.

Heinrich, J., Lanctot, M., and Silver, D. Fictitious self-play in extensive-form games. In *International conference on machine learning*, pp. 805–813. PMLR, 2015.

Hennes, D., Morrill, D., Omidshafiei, S., Munos, R., Perolat, J., Lanctot, M., Gruslys, A., Lespiau, J.-B., Parmas, P., Duéñez-Guzmán, E., et al. Neural replicator dynamics: Multiagent learning via hedging policy gradients. In

Proceedings of the 19th international conference on autonomous agents and multiagent systems, pp. 492–501, 2020.

Huang, W., Abbeel, P., Pathak, D., and Mordatch, I. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In *International Conference on Machine Learning*, pp. 9118–9147. PMLR, 2022a.

Huang, W., Xia, F., Xiao, T., Chan, H., Liang, J., Florence, P., Zeng, A., Tompson, J., Mordatch, I., Chebotar, Y., et al. Inner monologue: Embodied reasoning through planning with language models. *arXiv preprint arXiv:2207.05608*, 2022b.

Lanctot, M., Waugh, K., Zinkevich, M., and Bowling, M. Monte carlo sampling for regret minimization in extensive games. *Advances in neural information processing systems*, 22, 2009.

Lanctot, M., Zambaldi, V., Gruslys, A., Lazaridou, A., Tuyls, K., Pérolat, J., Silver, D., and Graepel, T. A unified game-theoretic approach to multiagent reinforcement learning. *Advances in neural information processing systems*, 30, 2017.

Li, G., Hammoud, H. A. A. K., Itani, H., Khizbullin, D., and Ghanem, B. Camel: Communicative agents for" mind" exploration of large scale language model society. *arXiv preprint arXiv:2303.17760*, 2023.

Liu, B., Jiang, Y., Zhang, X., Liu, Q., Zhang, S., Biswas, J., and Stone, P. Llm+ p: Empowering large language models with optimal planning proficiency. *arXiv preprint arXiv:2304.11477*, 2023.

Ma, W., Mi, Q., Yan, X., Wu, Y., Lin, R., Zhang, H., and Wang, J. Large language models play starcraft ii: Benchmarks and a chain of summarization approach. *arXiv preprint arXiv:2312.11865*, 2023.

Meta, Bakhtin, A., Brown, N., Dinan, E., Farina, G., Flaherty, C., Fried, D., Goff, A., Gray, J., Hu, H., et al. Human-level play in the game of diplomacy by combining language models with strategic reasoning. *Science*, 378(6624):1067–1074, 2022.

Moravčík, M., Schmid, M., Burch, N., Lisỳ, V., Morrill, D., Bard, N., Davis, T., Waugh, K., Johanson, M., and Bowling, M. Deepstack: Expert-level artificial intelligence in heads-up no-limit poker. *Science*, 356(6337):508–513, 2017.

Muller, P., Omidshafiei, S., Rowland, M., Tuyls, K., Perolat, J., Liu, S., Hennes, D., Marris, L., Lanctot, M., Hughes, E., et al. A generalized training approach for multiagent learning. *arXiv preprint arXiv:1909.12823*, 2019.

Nakano, R., Hilton, J., Balaji, S., Wu, J., Ouyang, L., Kim, C., Hesse, C., Jain, S., Kosaraju, V., Saunders, W., et al. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*, 2021.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744, 2022.

Park, J. S., O'Brien, J. C., Cai, C. J., Morris, M. R., Liang, P., and Bernstein, M. S. Generative agents: Interactive simulacra of human behavior. *arXiv preprint arXiv:2304.03442*, 2023.

Serrino, J., Kleiman-Weiner, M., Parkes, D. C., and Tenenbaum, J. Finding friend and foe in multi-agent games. *Advances in Neural Information Processing Systems*, 32, 2019.

Shah, D., Osiński, B., Levine, S., et al. Lm-nav: Robotic navigation with large pre-trained models of language, vision, and action. In *Conference on Robot Learning*, pp. 492–504. PMLR, 2023.

Shinn, N., Labash, B., and Gopinath, A. Reflexion: an autonomous agent with dynamic memory and self-reflection. *arXiv preprint arXiv:2303.11366*, 2023.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018.

Tammelin, O. Solving large imperfect information games using cfr+. *arXiv preprint arXiv:1407.5042*, 2014.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Vemprala, S., Bonatti, R., Bucker, A., and Kapoor, A. Chatgpt for robotics: Design principles and model abilities. *Microsoft Auton. Syst. Robot. Res*, 2:20, 2023.

Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., Choi, D. H., Powell, R., Ewalds, T., Georgiev, P., et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575 (7782):350–354, 2019.

Wang, G., Xie, Y., Jiang, Y., Mandlekar, A., Xiao, C., Zhu, Y., Fan, L., and Anandkumar, A. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*, 2023a.

Wang, S., Liu, C., Zheng, Z., Qi, S., Chen, S., Yang, Q., Zhao, A., Wang, C., Song, S., and Huang, G. Avalon's game of thoughts: Battle against deception through recursive contemplation. *arXiv preprint arXiv:2310.01320*, 2023b.

Wang, T. and Kaneko, T. Application of deep reinforcement learning in werewolf game agents. In *2018 Conference on Technologies and Applications of Artificial Intelligence (TAAI)*, pp. 28–33. IEEE, 2018.

Wang, Z., Cai, S., Liu, A., Ma, X., and Liang, Y. Describe, explain, plan and select: Interactive planning with large language models enables open-world multi-task agents. *arXiv preprint arXiv:2302.01560*, 2023c.

Wei, J., Tay, Y., Bommasani, R., Raffel, C., Zoph, B., Borgeaud, S., Yogatama, D., Bosma, M., Zhou, D., Metzler, D., et al. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*, 2022a.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., Le, Q. V., Zhou, D., et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35: 24824–24837, 2022b.

Williams, R., Hosseinichimeh, N., Majumdar, A., and Ghaffarzadegan, N. Epidemic modeling with generative agents. *arXiv preprint arXiv:2307.04986*, 2023.

Wooldridge, M. and Jennings, N. R. Intelligent agents: Theory and practice. *The knowledge engineering review*, 10(2):115–152, 1995.

Xu, Y., Wang, S., Li, P., Luo, F., Wang, X., Liu, W., and Liu, Y. Exploring large language models for communication games: An empirical study on werewolf. *arXiv preprint arXiv:2309.04658*, 2023a.

Xu, Z., Liang, Y., Yu, C., Wang, Y., and Wu, Y. Fictitious cross-play: Learning global nash equilibrium in mixed cooperative-competitive games. In *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems*, pp. 1053–1061, 2023b.

Yao, S., Chen, H., Yang, J., and Narasimhan, K. Webshop: Towards scalable real-world web interaction with grounded language agents. *Advances in Neural Information Processing Systems*, 35:20744–20757, 2022a.

Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., and Cao, Y. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022b.

Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T. L., Cao, Y., and Narasimhan, K. Tree of thoughts: Deliberate problem solving with large language models. *arXiv preprint arXiv:2305.10601*, 2023.

Yu, C., Velu, A., Vinitsky, E., Gao, J., Wang, Y., Bayen, A., and Wu, Y. The surprising effectiveness of ppo in cooperative multi-agent games. *Advances in Neural Information Processing Systems*, 35:24611–24624, 2022.

Zheng, B., Gou, B., Kil, J., Sun, H., and Su, Y. Gpt-4v (ision) is a generalist web agent, if grounded. *arXiv preprint arXiv:2401.01614*, 2024.

Zinkevich, M., Johanson, M., Bowling, M., and Piccione, C. Regret minimization in games with incomplete information. *Advances in neural information processing systems*, 20, 2007.

Our project website is at https://sites.google.com/view/strategic-language-agents/

A. Detailed Rules of the Game

Setup. At the beginning of each game, seven roles including two Werewolves, one Seer, one Doctor, and three Villagers are randomly assigned to the seven players. The Werewolves know each other's identity, while the Seer, Doctor, and Villagers only know their own identity. The players are denoted as "player_0", "player_1", ... "player_6" in the game.

Night Round. In the night round, the alive Werewolves, Seer, and Doctor can use their ability and take secret actions. These actions are only known to themselves and their teammates.

- Werewolf: choose a player to kill. If there are two Werewolves alive, the Werewolf with a smaller ID first proposes a player to kill. Then the proposal is added to the observation of the other Werewolf and this Werewolf decides the final kill target. For example, if player_0 and player_2 are the Werewolves, player_0 first proposes to kill player_i, then player_2 knows this information and decides to kill player_j. The final kill target is player_j. If there is only one Werewolf alive, then this Werewolf's action is the final kill target. The Werewolf is not allowed to kill a dead player or kill themselves or kill their teammate.
- Seer: choose a player to see if they are a Werewolf. The Seer is not allowed to see the identity of a dead player or themselves. The Seer is allowed to see the same player in different nights, though it is a waste of action.
- Doctor: choose a player to save without knowing who is the target of the Werewolves. The Doctor is not allowed to save a dead player and is allowed to save themselves.

Day Round Announcement. An announcement about last night's result is announced to all remaining players. If a player is killed, they are immediately moved out of the game and cannot reveal their role or communicate with other players. The announcement is as follows.

- If the Werewolves choose to kill player_i and the Doctor chooses to save a different player_j (or the Doctor is dead), then player_i is killed and the announcement will be "player_i was killed last night".
- If the Werewolves choose to kill player_i and the Doctor also chooses to save player_i, then no player is killed and the announcement will be "no player was killed last night".

Day Round Discussion. All remaining players take turns to speak only once in an open discussion. For example, if the remaining players are player_0, player_2, and player_5, then the discussion will start with player_0, continue to player_2, and end with player_5.

Day Round Voting. All remaining players simultaneously vote for one player or choose not to vote. Players are not allowed to vote for a dead player or themselves. The player with the most votes will be eliminated without revealing their role. If multiple players have the most votes, one player is randomly chosen and eliminated. The voting result is public and can be observed by all players.

Winning. The Werewolves win the game when the number of remaining Werewolves is equal to the number of other remaining players. The Werewolves do not have to eliminate all other players to win the game. The Villagers win the game when both Werewolves are eliminated.

B. Detailed Prompts

Since any prompting technique can be combined with our diverse action generation component and the population-based RL training component, we do not optimize every detail of the prompting choices as long as they produce reasonable results. It is possible to further improve the performance of our agents by using better prompting techniques.

B.1. Prompt for the Rock-Paper-Scissor Example

The prompt used in the Rock-Paper-Scissor Example in Fig 1 is listed below. We use "gpt-4-1106-preview" model to produce the result.

```
You are playing the Rock-Paper-Scissors game.  You should first reason about the Nash equilibrium
strategy of this game, and then choose one action from Rock, Paper, and Scissors based on your reasoning.

You should only respond in JSON format as described below Response Format
{
    "reasoning": reason about the Nash equilibrium strategy of this game,
    "action": choose one action from ["Rock", "Paper", "Scissors"] based on your reasoning
}
```

B.2. System Prompt

The system prompt used in our method is listed below.

```
You are an expert in playing the social deduction game named Werewolf.  The game has seven roles
including two Werewolves, one Seer, one Doctor, and three Villagers. There are seven players including
player_0, player_1, player_2, player_3, player_4, player_5, and player_6.

At the beginning of the game, each player is assigned a hidden role which divides them into
the Werewolves and the Villagers (Seer, Doctor, Villagers). Then the game alternates between the night
round and the day round until one side wins the game.

In the night round:  the Werewolves choose one player to kill;  the Seer chooses one player to
see if they are a Werewolf; the Doctor chooses one player including themselves to save without knowing
who is chosen by the Werewolves; the Villagers do nothing.

In the day round:  three phases including an announcement phase, a discussion phase, and a
voting phase are performed in order.
In the announcement phase, an announcement of last night's result is made to all players. If player_i
was killed and not saved last night, the announcement will be "player_i was killed"; if a player was
killed and saved last night, the announcement will be "no player was killed"
In the discussion phase, each remaining player speaks only once in order from player_0 to player_6 to
discuss who might be the Werewolves.
In the voting phase, each player votes for one player or choose not to vote. The player with the most
votes is eliminated and the game continues to the next night round.

The Werewolves win the game if the number of remaining Werewolves is equal to the number of
remaining Seer, Doctor, and Villagers. The Seer, Doctor, and Villagers win the game if all Werewolves
are eliminated.
```

B.3. Prompt for Secret Actions

The prompt for secret actions in our method is listed below.

```
Now it is night <n_round> round, you (and your teammate) should choose one player to kill/see/save. As
player_<id> and a <role>, you should first reason about the current situation, then choose from the
following actions: <action_0>, <action_1>, ..., .

You should only respond in JSON format as described below.
Response Format:
```

```
{
    "reasoning": "reason about the current situation",
    "action": "kill/see/save player_i"
}

Ensure the response can be parsed by Python json.loads
```

B.4. Prompt for Statement Actions

The prompt for statement actions in our method is listed below.

```
Now it is day <n_round> discussion phase and it is your turn to speak. As player_<id> and a <role>,
before speaking to the other players, you should first reason the current situation only to yourself,
and then speak to all other players. You should only respond in JSON format as described below.
Response Format:
{
    "reasoning": "reason about the current situation only to yourself",
    "statement": "speak to all other players"
}

Ensure the response can be parsed by Python json.loads
```

B.5. Prompt for Voting Actions

The prompt for voting actions in our method is listed below.

```
Now it is day <n_round> voting phase, you should vote for one player or do not vote to maximize the
Werewolves' benefit (for the Werewolves) / you should vote for one player that is most likely to be a
Werewolf or do not vote (for the Villagers). As player_<id> and a <role>, you should first reason about
the current situation, and then choose from the following actions: do no vote, <action_0>, <action_1>,
..., .

You should only respond in JSON format as described below.
Response Format:
{
    "reasoning": "reason about the current situation",
    "action": "vote for player_i"
}

Ensure the response can be parsed by Python json.loads
```

B.6. Prompt for Deductive Reasoning

For clarity of the main text, we introduce four attributes including role, reliability, reasoning, and evidence. However, in practice, the word "reliability" can be confusing to LLMs because it can refer to the reliability of the player or refer to the reliability of the deduction result, which leads to completely rating results. To avoid this problem, we use "confidence" to replace "reliability" in our implementation to prompt the LLM for deduction result. Concretely, "confidence" is an integer ranging from 5 to 10 that rates the certainty of the deduction result of the current player's role. A confidence of 5 means a random guess of role and a confidence of 10 means absolutely sure about the role.

After the LLM produces the confidence for each player's role, we then use confidence to compute the player's reliability. If a player is deduced to be a Werewolf and the confidence for this deduction is high, then this player is likely to be a real Werewolf and should have a low reliability. Therefore, the reliability of a player deduced to be a Werewolf is calculated as 11 - confidence. Likewise, if a player is deduced to be a non-Werewolf and the confidence is high, then this player should have a high reliability. We let the reliability of such players to be equal to the confidence. We also allow the LLM to deduce the role of a player as "Uncertain". The confidence for an "Uncertain" player is set to a fixed value of 6.

The prompt for deductive reasoning in our method is listed below.

```
As player_<id> and a <role>, you should reflect on your previous deduction and reconsider the hidden
roles of <remaining players>. You should provide your reasoning, rate your confidence, and cite all
key information as evidence to support your deduction.

You should only respond in JSON format as described below.
Response Format:
{
    "player_i": {
        "role": select the most likely hidden role of this player from
                ["Werewolf", "Seer", "Doctor", "Villager", "Uncertain"],
        "reasoning": your reflection and reasoning,
        "confidence": rate the confidence of your deduction from 5 (pure guess)
                      to 10 (absolutely sure),
        "evidence": list of integers that cite the key information
    }
}
Ensure the response can be parsed by Python json.loads
```

B.7. Prompt for Diverse Action Generation

We use two kinds of methods to get N action candidates.

- For the secret actions and voting actions, we produce all actions in a single inference by adding the following prompt in the action prompt: "propose N diverse actions that correspond to different strategies."

- For the statement actions, we iteratively asks the LLMs to produce one new action at a time by adding the following prompt in the action prompt: "consider a new action that is strategically different from existing ones."

Below are two examples of the diverse action candidates generated by the LLM.

Our agent (player_0) as the Doctor in the night round.

Action candidate 1.

- Reasoning: Based on the facts and deductions, player_2 is likely to be a Seer. If I save player_2 tonight, it will prevent the Werewolf from eliminating another player. This strategy focuses on protecting the remaining non-Werewolf players
- Action: save player_2.

Action candidate 2.

- Reasoning: Considering that player_3 may be a Non-Werewolf, saving player_3 tonight could benefit the non-Werewolf team. It would increase the chances of eliminating the Werewolf in the upcoming day round. This strategy prioritizes potential collaboration with player_3.
- Action: save player_3.

Action candidate 3.

- Reasoning: As player_0, I have information that I am the Doctor. Protecting myself could be a wise choice to ensure the non-Werewolf team has a higher chance of winning. This strategy prioritizes self-preservation.
- Action: save player_0.

Our agent (player_4) as a Werewolf in the discussion phase.

Action candidate 1.

- Reasoning: As a Werewolf, my goal is to eliminate the Seer, Doctor, and Villagers in order to win the game. Since player_1 is likely to be a Seer, it is in my best interest to cast suspicion on them and try to get them eliminated. This will make it easier for me to target the remaining players in the following nights.
- Discussion: I think we should start by discussing player_1's role. As a Seer, I saw that they are the Werewolf last night and we should vote to eliminate them. I encourage everyone to share their thoughts on player_1 and provide any information or suspicions they may have.

Action candidate 2.

- Reasoning: Another possible strategy is to divert suspicion away from player_1 and instead focus on player_2 or player_3. By doing this, we can create confusion among the remaining players and make it harder for them to trust each other. This can potentially lead to miscommunication and mistakes in their decision-making process.
- Discussion: I suggest we consider the possibility that player_2 or player_3 might be the Werewolf. It's important for us to thoroughly discuss and analyze their behavior, actions, and any information we may have about them. Let's hear from each player and gather as much information as possible.

Action candidate 3.

- Reasoning: A third strategy is to remain silent and not actively participate in the discussion. By doing so, I can create doubt and suspicion among the other players. They may start suspecting each other and questioning why I am not participating. This can potentially lead to a split in their trust and unity, making it easier for me to manipulate the situation in my favor.
- Discussion: I would like to observe the discussion for now and hear what each player has to say. It's important for us to thoroughly analyze all the information and opinions shared. Let's have an open and honest discussion where everyone gets a chance to voice their thoughts and suspicions.

B.8. Prompt for Different Styles

We consider the following three different play styles for the pure LLM-based Werewolf agent in our population-based training. The prompt for each style is listed below.

1. Quiet follower: "As a Werewolf, you should be a quiet follower that lays low and follow others' opinion to avoid drawing attention to yourself."
2. Active contributor: "As a Werewolf, you should be an active contributor that pretends to be one of the Villagers by actively engaging in discussion and looking for Werewolves."
3. Aggressive accusor: "As a Werewolf, you should be an aggressive accuser that accuses others to create chaos and divert suspicion from yourself."

We also consider the three different play styles for the pure LLM-based Villager agent in our population-based training. The prompt for each style is listed below.

1. Secretive player: "As the Seer / the Doctor / a Villager, you should be a secretive player that hides your role to gather more information."
2. Proactive player: "As the Seer / the Doctor / a Villager, you should be a proactive player that reveals you identity once you obtain crucial information."
3. Default player: no additional prompt.

B.9. Text Embedding

We use OpenAI's "text-embedding-ada-002" model to get the text embeddings.

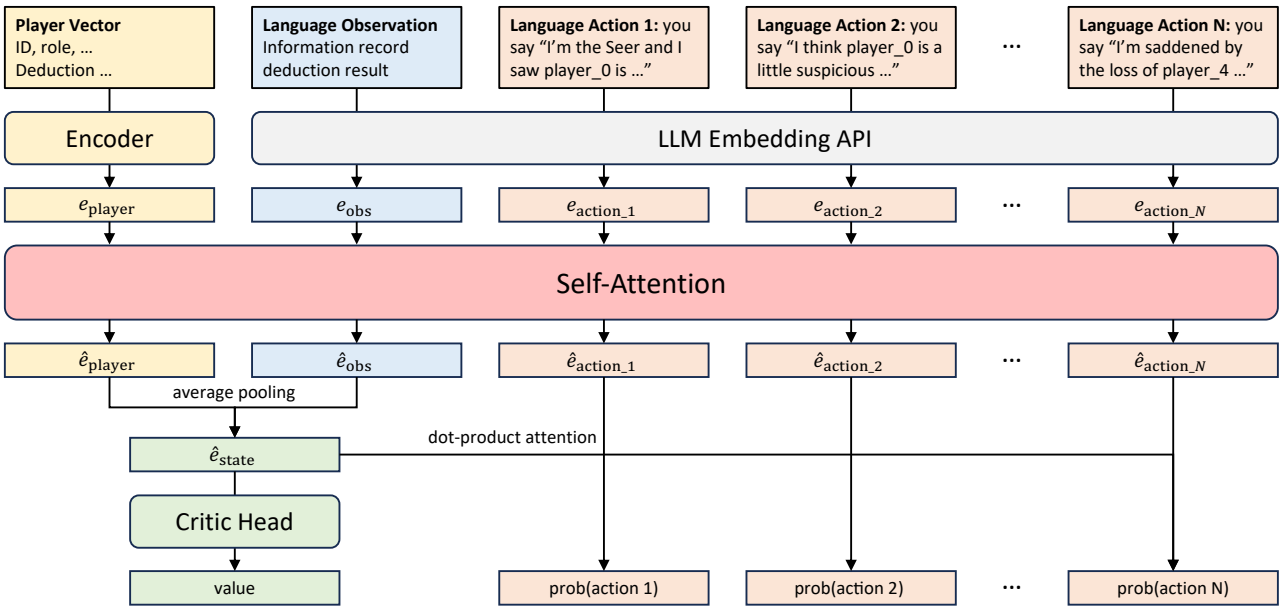


Figure 6: Self-attention policy architecture.

C. Implementation Details

C.1. Self-Attention Policy Architecture

We use a self-attention policy and the architecture is shown in Fig. 6. The inputs are divided into three types and their embeddings are produced as follows.

Player Vectors (Yellow). We first generate a player vector that includes information like the player's ID, role, deductions, etc. by one-hot encoding. The detail of the player vector is listed in Table 5. Then the player vector is fed into an MLP encoder to get the player embedding for self-attention input.

Language Observation (Blue). The raw observation input is in the form of natural language and is the concatenation of the information record and deduction result described in Section 4.1. This language observation input is then converted into vector embedding using LLM embedding API. In our case, we use OpenAI's "text-embedding-ada-002" and the length of the embedding is 1536.

Language Action Candidates (Orange). For each action candidate, the raw input is also in the form of natural language and is the concatenation of the reasoning and action as described in Section 4.2. Each raw input is then converted into vector embedding using the same LLM embedding API as the language observation.

We use a residual self-attention network without position embeddings to generate contextualized player embedding, observation embedding, and action embeddings. Then we average pool the player embedding and the observation embedding to get the state embedding. The state embedding is fed into an MLP critic head to produce the predicted value. The probability to sample an action candidate is proportional to the dot-product attention between the state embedding and the corresponding action embedding.

C.2. Reward Design

The reward for the Werewolf environment is mainly the winning reward. We also designed several shaping rewards to accelerate training. More specifically, we consider the following reward.

- **Winning Reward:** all winners +100, all losers -100.
- **Werewolf Killing Reward:** if the Werewolves successfully kill a player at night, the Werewolves +5, the Villagers (Seer, Doctor, Villagers) -5.

		Length	Description
ID		7	one hot encoding of ID.
Role		4	one hot encoding of role, ["Werewolf", "Seer", "Doctor", "Villager"].
Round		1	current round.
Phase		3	one hot encoding of current phase, ["night", "discussion", "voting"].
Alive players		7	alive flag for each player.
For each round (3 rounds)	secret action	7	one hot encoding of the target player, (all zero if do not act).
	announcement	7	one hot encoding of the dead player, (all zero if no player is dead).
	voting result	49	one hot encoding of the each player's choice, (all zero if the player does not vote or is dead).
For each player (7 players)	role	4	one hot encoding of deduced role, ["Werewolf", "Seer", "Doctor", "Villager"].
	confidence	1	confidence of deduction on scale 5-10.

Table 5: Details of the player vector.

- **Seer Seeing Reward:** if the Seer successfully identifies a Werewolf at night, the Werewolves -2, the Seer +2.
- **Doctor Saving Reward:** if the Doctor successfully saves a player at night, the Werewolves -5, the Doctor +5.
- **Voting Result Reward:**
  - If a Werewolf is voted out, the Werewolves -5, the Villagers (Seer, Doctor, Villagers) -5.
  - If a non-Werewolf is voted out, the Werewolves +5, the Villagers -5.
- **Individual Voting Reward:**
  - If the current player votes for a Werewolf, the Werewolves -1, the current player +1.
  - If the current player votes for a non-Werewolf, the Werewolves +1, the current player -1.
  - If the current player chooses not to vote, no additional reward for any player.

C.3. Population-Based RL Training and Hyperparameters

We use MAPPO (Yu et al., 2022) as the RL algorithm and use population-based training to learn the policy. The population is initialized to a set of six pure LLM-based agents including a quiet follower (Werewolf), an active contributor (Werewolf), an aggressive accusor (Werewolf), a secretive player (non-Werewolf), a proactive player (non-Werewolf), and a default player (non-Werewolf). As training progresses, we gradually add checkpoints of the RL policy into the population.

More specifically, at the beginning of each episode, we randomly select four players to be the learning agents who use the current policy and three players to be the fixed agents who use policies sampled from in the population. For each fixed agent, it randomly samples one policy from the population and uses this policy till the end of the game. In this way, we make the RL policy play with a wide range of policies both as teammates and opponents The rollout data of the four learning agents are then collected and used to train the RL policy. In our implementation, we use a distributed RL training system to run parallel environments and utilize multiple API accounts to maximize the throughput of LLM inference calls. To make the RL training start with a good initial policy, we first apply behavior cloning with data generated by pure LLM-based agents with hidden role deduction only. Then we use this BC policy as the initial policy and run MAPPO for 10M steps. The hyperparameters for RL training are listed in Table 7.

Hyper-parameters	Value
Learning rate	5e-4
Discount rate (γ)	0.95
GAE parameter (λ_{GAE})	0.95
Gradient clipping	10.0
Adam stepsize	1e-5
Value loss coefficient	1
Entropy coefficient	0.01
PPO clipping	0.2
PPO epochs	10
MLP encoder layer num	3
MLP encoder layer size	1536
Attention layer head num	12
Attention layer size	128
Critic head layer num	1
Weight decay coefficient	1e-6
Action candidate num N	3

Figure 7: Training hyperparameters.

Win Rate	As the Villagers	As the Werewolves
ReAct	0.16 (0.02)	0.54 (0.03)
ReCon	0.20 (0.02)	**0.61 (0.02)**
Concurrent	**0.22 (0.02)**	0.60 (0.04)
Hidden Role Deduction	**0.22 (0.04)**	**0.61 (0.05)**

Figure 8: Comparison with other prompting techniques.

Win Rate	w.o. communication	**w. communication**
ReAct	0.05 (0.01)	**0.16 (0.02)**
ReCon	0.04 (0.01)	**0.20 (0.02)**
Concurrent	0.04 (0.01)	**0.22 (0.02)**
Atomic	0.05 (0.02)	**0.27 (0.02)**
Ours	0.06 (0.02)	**0.30 (0.03)**

Figure 9: Ablation on communication.

N	2	**3**	4	5
increased diversity	0.35	**0.37**	0.23	0.16

Table 6: Ablation on the number of action candidates N.

D. Experiment Details and Additional Results

D.1. Compare Hidden Role Deduction with Other Prompting Techniques

Any prompting technique can be combined with the diverse action generation component and the population based RL training component of our framework. Although we do not optimize every detail of our prompting choice, we compare our current design of hidden role deduction with other prompting methods to show that it produces reasonable performance. We compare our agents with the hidden role deduction component only and compare it with *ReAct* (Yao et al., 2022b), *ReCon* (Wang et al., 2023b), and *Concurrent* (Xu et al., 2023a) by playing against our agents. As shown in Table 8, our agents using only the hidden role deduction components achieve the highest win rates both as the Villagers and as the Werewolves. It is possible to further improve the performance of our agents by using better prompt techniques, but the focus of our work is to overcome the intrinsic bias in LLM-based agents using reinforcement learning.

D.2. Ablation on Communication

Werewolf is a language game that heavily relies on natural language communication between agents. To show the importance of communication to achieve strong performance, we perform an ablation by removing the communication ability of the four agents in the round-robin tournament Section 5.2, i.e., the agents always return an empty string in the discussion phase. We evaluate these agents by playing 100 games against our agent and their mean win rate is shown in Table 9.

D.3. Ablation on Different Actions

We perform an ablation study to separate the influence of statement actions from secret and vote actions to better understand their relative contributions to overall performance. We consider the following three agents. (1) Ours: use the RL policy for all three kinds of actions (statement actions in discussions, secret actions at night, and voting actions). (2) Statement w.o. RL policy: use the LLM to generate one action for statement actions and use the RL policy for secret actions and voting actions. (3) All w.o. RL policy: use the LLM to generate one action for all three kinds of actions.

As shown in Table 7., using RL policy for both statement actions and strategic actions (secret and voting actions) contributes to the improvement in performance. Interestingly, using RL policy for secret and voting actions leads to more improvement as the Villagers, while using RL policy for statement actions leads to more improvement as the Werewolves. A possible

Win Rate	As the Villagers	As the Werewolves
Ours	**0.30 (0.03)**	**0.70 (0.03)**
Statement w.o. RL policy	0.28 (0.05)	0.66 (0.05)
All w.o. RL policy	0.22 (0.04)	0.61 (0.05)

Table 7: Ablation on Different Actions.

Win Rate	GPT-4	LLaMA-7B	ChatGLM-6B
w.o. policy	0.23 (0.02)	0.14 (0.01)	0.11 (0.01)
w. policy	**0.35 (0.04)**	**0.19 (0.02)**	**0.21 (0.03)**

Table 8: Zero-shot transfer to other LLMs.

explanation for this difference is that the Villagers can simply be honest with their statements and focus more on finding and voting out the Werewolves, while the Werewolves are required to make up convincing statements to deceive the Villagers and also choose the best killing and voting target.

D.4. Ablation on the Number of Action Candidates N

We perform an ablation on the number of action candidates N to investigate the effect of N and validate that the newly added action candidates increase the overall diversity. Given an action set $\mathcal{A}$, we define the increased diversity introduced by a new action a as $\mathrm{div}(\mathcal{A}, a) = \min_{a' \in \mathcal{A}} \|e(a) - e(a')\|$, where $e(a)$ is the embedding of a produced by the LLM. This metric uses the minimum Euclidean distance between the embeddings of the new action and the existing actions to compute the increased diversity. If the embedding of the new action is similar to any of the existing actions, the increased diversity will be small and close to zero. Note this is one possible way to define diversity in actions and there are many other reasonable definitions. A good way to decide if an action is diverse is to use human evaluation, and we provide two examples of the diverse action candidates proposed by the LLM in Appendix B.7.

We consider $N = 2, 3, 4, 5$ and evaluate the increased diversity introduced by the last action. As shown in Table 6, the increased diversity of the last action is relatively large when $N = 2, 3$ and becomes smaller when $N = 4, 5$. This aligns with the intuition that it is harder to propose new actions when there are already many actions. Another reason for small increased diversity when $N = 4, 5$ is that sometimes the number of possible actions is small. For example, when there are 5 players alive and 2 of them are Werewolves (a common situation on the second night), the Werewolves only have 3 possible actions at night and prompting for a 4th or 5th action cannot improve any diversity. In our implementation, we set $N = 3$.

D.5. Zero-Shot Transfer to Other LLMs

Since our RL policy takes natural language state and actions as input and is decoupled from the LLM used in previous steps, it can be directly combined with any other LLMs and improve the performance of the LLM-based agent. We evaluate this zero-shot transfer ability of our RL policy trained with gpt-3.5-turbo by applying it to unseen LLMs including GPT-4, LLaMA-7B, and ChatGLM-6B. We implemented two agents for each LLM, one using our RL policy learned with gpt-3.5-turbo and the other without the policy. The agent with our RL policy (w. policy) follows the design of our agent and uses the RL policy to select actions, while the agent without policy (w.o. policy) uses the LLM instead of the RL policy to select actions. These two agents are evaluated by playing against our agents for 100 games and their average win rates are shown in Table 8. Although not trained with any of these LLMs, the RL policy is shown to improve the performance of all these LLMs from stronger models like GPT-4 to weaker models like LLaMA-7B as shown by the bold numbers in the table. This is because we use natural language as a general interface between LLMs and the RL policy. As long as the LLMs can produce a set of language actions, the RL policy can be used to improve the strategic ability in a zero-shot way.

D.6. Generalization to Different Settings of Werewolf

For generalization to new forms of the Werewolf game, the first two components in our framework (hidden role deduction and diverse action generation) can be directly generalized to new settings like changing the number of players, adding new roles, and changing the winning conditions, with only slight changes in the prompt.

The population-based RL training method can also be directly used in different settings, but the policy needs to be retrained

Win Rate	6-player	8-player
w.o. the 7-player policy	0.18 (0.04)	0.27 (0.05)
w. the 7-player policy	**0.23 (0.05)**	**0.30 (0.06)**

Table 9: Generalization to Werewolf game with different numbers of players.

for a new setting to achieve the best performance. We expect an RL policy trained under one specific game setting to generalize to similar settings like adding or removing one player, but we do not expect it to generalize to games with significant changes like changing the winning conditions. To evaluate the transferability of a trained RL policy to slightly different game settings, we considered a 6-player and an 8-player Werewolf game and compared agents with and without the RL policy trained on the 7-player game. As shown in Table 9, the RL policy trained in 7-player setting can generalize to the 6-player and 8-player settings and improve the performance.

D.7. Human Evaluation

Experiment Procedure. We conduct the human-agent experiment with department approval and recruited 160 human players to participate in the evaluation. The participants are required to be fluent in English and over 18 years old. They are paid $10 per hour and are allowed to participate in the experiment after agreeing to our consent form, which explicitly explained that the Werewolf game contains deceptive communications and the AI agents may have dishonest statements in the game. The participant can choose to withdraw from the experiment at any time.

We randomly divide the 160 participants into 2 groups. The first group has 20 participants and each of the participants play with 6 copies of our agents for 10 consecutive games. In 5 of the 10 games, we let the human participant play as one of the Villagers. In the rest 5 games, we let the participant play as one of the Werewolves. The hidden roles of each players are randomly assigned and the order of 5 games as the Werewolves and 5 games as the Villagers are also randomly generated. This lead to 100 games of one human participant playing as the Villagers with six copies of our agents, and also 100 games of one human participant playing as the Werewolves with six of our agents. The win rates averaged over 100 games is shown in the "Human" column in the left part of Table 1.

The second group has 140 participants. We further split them to 20 teams of 7 players and each team plays 30 consecutive games. In 10 of the 30 games, we randomly sample 6 human players from the team and let them play with our agents. Similarly, in 10 of the 30 games, we randomly 6 human players from the team to play with the pure LLM-based agent. In the rest 10 games, we let these 7 human player play the game without AI agents. We also randomly generated the order of these 30 games and randomly assign the hidden roles that each AI agent plays as the Werewolf for 5 games and as the Villagers for 5 games. This lead to a total number of 600 games and gives the 6 averaged win rates in the right part of Table 1.

Consent Form.

Introduction: You are invited to participate in a research study that aims to evaluate the performance of an AI agent developed to play the Werewolf game. The purpose of this research is to assess the strategic capabilities of the AI agent in a social deduction game setting, where participants engage in discussions and try to identify hidden roles among the players.

Study Overview:

- The research involves playing the Werewolf game with an AI agent developed by the researcher.
- Participants will interact with the AI agent through text-based communication within the Werewolf game environment.
- The game involves deceptive communication and the AI agent may make dishonest statements to achieve their objectives in the game.

Voluntary Participation: Your participation in this study is entirely voluntary. You have the right to refuse to participate or withdraw from the experiment at any time without any consequences. If you choose to withdraw, it will not affect your relationship with the researcher or your access to any benefits.

Risks and Benefits:

- The primary risks associated with this study are related to the social and psychological aspects of playing the Werewolf game, including potential frustration or disappointment.
- The participant will be paid $10 per hour and your involvement will contribute to the understanding of AI agent in social deduction games.

Confidentiality:

- Your identity will be kept confidential, and your personal information will not be disclosed in any publications or presentations resulting from this research.
- The data collected will be anonymized, and any quotes used will be attributed without identifying details.

Deceptive Communications:

- The Werewolf game involves deceptive communication, and the AI agent may make dishonest statements as part of its strategy.
- It is essential to understand that the AI agent's behavior is not reflective of the researcher's personal values or intentions.

Questions and Concerns: If you have any questions or concerns about the study, you can contact the researcher at ...

Consent: I have read and understood the information provided in this consent form. I agree to participate voluntarily in the study and acknowledge that I can withdraw at any time without consequences.

Signature: _____

Example Input for Human Players.

Basic Information:

- you are player_5, your role is Doctor.
- current round and phase: night 2.
- remaining players: player_0, player_1, player_2, player_5, player_6.

Round 1:

- night 1: you chose to save player_5.
- day 1 announcement: player_4 was killed last night.
- day 1 discussion:
  - player_0 said: Good day, fellow players. As a Villager, my objective is to help identify and eliminate the Werewolves. Since player_4 was killed last night, we know there is at least one Werewolf among us. I would like to hear everyone's thoughts and suspicions about who might be the Werewolves. Let's work together to find the culprits and ensure the safety of our village.
  - player_1 said: ...
  - player_2 said: ...
  - player_3 said: ...
  - you said: ...
  - player_6 said: ...

```
  • day 1 voting result: player_3 had the most votes and was eliminated.
      – voted for player_3: player_1, player_6.
      – voted for player_1: player_3.
      – choose not to vote: player_0, player_2, player_5.

Now it is night 2 round and you should choose one player to save. As player_5 and the Doctor, you
should choose from the following actions: save player_0, save player_1, save player_2, save player_5,
save player_6.
```

E. Discussions

E.1. Application to Other Games

Our approach has three components: (1) hidden role deduction, (2) diverse action generation, and (3) population-based RL training. We believe our framework of using these three components is general enough to be adapted to other social deduction games. The main changes would be specific prompt designs but the general framework stays the same. Specifically, while some prompts used in the first component are specifically designed for the Werewolf game, the prompts for generating diverse actions, and the population-based reinforcement learning method in the second and third components are general to be applied in other tasks like social deduction games (e.g., *The Resistance: Avalon* (Wang et al., 2023b)) and board games with negotiations (e.g., *Diplomacy* (Meta et al., 2022)).

Take *The Resistance: Avalon* as an example. To apply our method in this game, we need to specialize the deduction template in the first component and maybe use additional prompting techniques like in (Wang et al., 2023b). We also need to change the prompt in the third component to generate reasonable agent pool for this game. Then our framework can be applied to this game without changing the diverse action generation and RL training components.

E.2. Discussion on the Atomic Agent

The atomic agent predefines a set of high-level atomic actions and trains an RL policy with this fixed action space. The RL policy takes the embeddings of the information record and deduction result as input and selects the atomic action based on this input. Then the natural language actions used in gameplay are generated by prompting the LLM to follow the selected atomic actions. In our case, the atomic action set consists of 13 actions including "idle", "target player_0", "target player_1", "target player_2", "target player_3", "target player_4", "target player_5", "target player_6", "claim to be a Werewolf", "claim to be a Seer", "claim to be a Doctor", "claim to be a Villager", and "do not reveal role".

As shown in the results of the round-robin tournament in Fig 5, although both our agent and the atomic agent combine RL with LLMs, our agent achieves much better performance. This is because the predefined atomic actions can be too general and fail to generate more fine-grained actions. Consider the situation where the agent is a Werewolf and their teammate is accused. The agent can choose to defend the teammate, avoid discussing the accusation, or support the accusation, but none of these actions can be stably generated by prompting LLMs with any of the predefined atomic actions. By contrast, our agent produces the action set during gameplay and can generate diverse actions at any granularity, which greatly improves the performance.

E.3. Failure Cases

Unintentional Disclosure of Hidden Role. One failure case of our agents is that the Werewolf agents could unintentionally reveal their identity in the discussion phase. This is because the Werewolf agents need to first reason as a Werewolf, and then pretend to be a non-Werewolf in discussion. This mismatch in their thoughts and words could make the LLM get confused and accidentally speak out their thought as a Werewolf. Fortunately, the probability of such a failure case is significantly reduced by generating multiple diverse actions and using an RL policy to choose the best one. A potential way to further mitigate this issue is to get the reasoning as a Werewolf and the statement as a non-Werewolf in two separate calls of LLM.

Inconsistent Behaviors. Another failure case is that our agents could produce inconsistent behaviors, especially when they are lying. For example, our agent is a Werewolf and claims to be the Doctor on the first day round. However, on the second day, our agent may claim to have no information in the night, which contradicts the previous claim as the Doctor. This issue

is mitigated by the diverse action generation and the RL policy learns to choose other better actions.

E.4. Limitations

Pure Text-Based Setting In this work, we focus on powering LLM-based agents with RL to overcome intrinsic bias and improve decision-making ability. Therefore, we only consider a simplified pure text-based setting of the Werewolf game. In real-world games, Werewolf also involves other non-verbal elements like tone of voice, facial expressions, and body language. These multi-modal data are also important for deducing hidden roles and making better decisions. It would be an interesting future direction to build strategic AI agents that utilize multi-modal information.

Without using Human Data Since human gameplay data are usually expensive and hard to collect, we take an alternative approach by generating human-like agents using LLM and training our agents with these agents to make the learned policy more compatible with human strategies. However, the language style and game strategy of the learned agent may still have some difference from real human players. An potential future direction is to collect human data and fine-tune the LLM to make the agents more human-like and even indistinguishable from human players.

F. Emergent Behaviors

To more intuitively demonstrate the performance and strategic behaviors of our agent, we provide a qualitative as well as a quantitative study on the emergent behaviors exhibited by our agent. We first show two common and basic behaviors, concealment and cooperation, that typically appear in mixed cooperative-competitive games with hidden identities. Then, we report two more advanced and strategic behaviors including bluffing and sacrificing that demonstrate the sophistication and strength of our agent.

Concealment. Concealment refers to the act of hiding or disguising the true identity or intention of the players, which is a fundamental skill for all roles. Werewolves should strike a balance between engagement and discretion to avoid detection, while roles with special abilities like the Seer should also consider hiding their identities to prevent being killed by Werewolves. Below is an example where our agent plays as the Seer and chooses to conceal their identity because they do not have any important information.

Our agent as the Seer in the discussion phase.

- **Reasoning:** Since I'm the Seer, I know player_0 was not a Werewolf. However, he was killed, so there are still two Werewolves among us. I need to be careful not to reveal my role too early, as Werewolves might target me next. Since I don't have any more information right now, I should not make any specific accusations, but encourage discussion and observe the reactions of others.
- **Discussion:** It's unfortunate that player_0 is dead. We need to work together to find out who is responsible for this. Pay attention to who is too eager to accuse or too quick to defend. Let's discuss.

Moreover, concealment is not just about what to say but also involves ensuring your actions align with your words. Werewolves sometimes need to give up advantageous actions like voting to eliminate their opponents which may reveal their true identities, as shown in the following example.

Our agent as a Werewolf in the voting phase.

- **Reasoning:** Since everyone is advocating for unity and careful observation, accusing someone outright could backfire and make me a target. Therefore, it might be best to go along with the majority and not vote in this round.
- **Action:** Do not vote.

Cooperation. It is crucial for players to cooperate in multi-agent environments to achieve their common goals. Werewolves should strive for a common approach when casting suspicion, defending themselves, or voting eliminations. Cooperation is even harder for the Villagers because they do not know the role of each other and need to first distinguish friend and foe and then achieve coordination. Below we give an example where the Doctor manages to cooperate with the Seer.

The Seer (player_2) revealed their identity and asked the Doctor for protection.
Our agent as the Doctor in the night round.

- **Reasoning:** In the first day discussion, player_2 claimed to be the Seer and identified player_0 as a Werewolf. Player_0 is indeed very suspicious because they accused me without valid reasons. Therefore, I think player_2 is probably the true Seer and is very likely to be the Werewolves' target. It would be best to save player_2 instead of myself tonight.
- **Action:** save player_2.

In this case, our agent successfully identifies the Seer and decides to save player_2 instead of themselves. This allows the Seer to survive for another round to provide more information, which leads to the final victory.

Bluffing. Bluffing is a more advanced strategy that deceives or misleads other players about one's true identity. It is often adopted by the Werewolves to pretend to be important roles like the Seer or the Doctor to misdirect the voting and eliminate innocent players. Bluffing is a high-risk, high-reward strategy that requires skill, timing, and keen awareness of other players' perceptions in order to be executed effectively, as demonstrated in the example below.

Our agent as a Werewolf in the discussion phase.

- **Reasoning:** Since I'm the last player to speak before voting and no one has come forward as the Seer, I should claim to be the Seer. This could take the heat off of me, additionally, it might guide other players to suspect and possibly vote out whoever I indicate as a Werewolf.
- **Discussion:** I believe it's important for us to become more open; I am the Seer. I checked player_4 last night, and he is a Werewolf. I understand it's a risky move to reveal my identity. But we need to unite and use all the resources we have to identify the Werewolves. I hope my move encourages the Doctor to protect me tonight, and we should vote player_4 out today.

Our agent astutely observes that no player has declared themselves as the Seer and no one can refute their claim because they are the last one to speak. This is the perfect time to pretend to be the Seer. Our agent seizes the chance to gain others' trust and persuades everyone to eliminate player_4, who is an innocent player.

Sacrificing. In most cases, Werewolves should secretly help each other to stay undercover. However, when their teammates are suspected by many players and are very likely to be voted out, it would be suspicious not to vote for the teammate. Therefore, agents sometimes need to sacrifice their fellow Werewolf to reduce suspicion of themselves, which is shown in the following example.

Our agent's Werewolf teammate (player_6) has been accused by the Seer.
Our agent as a Werewolf in the voting phase.

- **Reasoning:** While it would indeed be disadvantageous to lose player_6, as they are my Werewolf teammate, at this point in the game, the suspicion against them is quite high. Therefore, voting for player_6 may contribute to maintaining my disguise and trust among other players, which are crucial for us to win.
- **Action:** vote for player_6.

G. Game Log Examples

The Werewolves Win. This is an example game log where the Werewolves win the game.

role assignments:
* player_0: Werewolf.
* player_1: Villager.
* player_2: Villager.
* player_3: Villager.
* player_4: Werewolf.
* player_5: Doctor.
* player_6: Seer.

night 1:
* Werewolves: player_0 and player_4 chose to kill player_1.
* Seer: player_6 saw player_0 is a Werewolf.
* Doctor: player_5 chose to save player_5.

day 1 announcement: player_1 was killed last night.
remaining players: player_0 (Werewolf), player_2 (Villager), player_3 (Villager), player_4 (Werewolf), player_5 (Doctor), player_6 (Seer).

day 1 discussion:
* player_0 (Werewolf) said: "Good morning, everyone. As player_0, it is unfortunate that player_1 was killed last night. I think we should focus on finding the Seer and Doctor who can provide valuable information about who might be the Werewolves. Let's all share our thoughts and suspicions regarding the remaining players and try to figure out who might be the Werewolf."
* player_2 (Villager) said: "Good morning, everyone. As a Villager, I believe our priority should be to find the Werewolves among us. Since player_1 was killed last night, we can rule out the possibility of player_1 being the Werewolf. I would like to hear everyone's thoughts on who they suspect might be the Werewolves. Let's share our suspicions and try to gather as much information as we can to make an informed decision during the voting phase."
* player_3 (Villager) said: "Hello everyone, as a Villager, I don't have any special abilities to help us determine who the Werewolves might be. I will carefully listen to everyone's opinions and votes. Let's work together to find the Werewolves and eliminate them."
* player_4 (Werewolf) said: "Good morning, everyone. As a Villager, I agree with player_2 that our priority should be to find the Werewolves among us. Since player_1 was killed last night, we can rule out the possibility of player_1 being the Werewolf. I think it's important for us to share our suspicions and gather as much information as possible. Let's work together to identify the Werewolves and eliminate them. I'm interested to hear what everyone's thoughts are on who might be the Werewolves."
* player_5 (Doctor) said: "During the night, I chose to save myself as I am the Doctor. Unfortunately, player_1 was killed last night. This means that player_0, player_2, player_3, player_4, player_6, and I are the remaining players. Let's discuss and share our suspicions on who the Werewolves might be. It's crucial for us to work together to identify and eliminate the Werewolves to ensure the safety of our village."
* player_6 (Seer) said: "During the night, I used my ability as the Seer and discovered that player_0 is a Werewolf. This means we already have one Werewolf identified. We should focus our discussion on the remaining players to try and identify any other potential Werewolves. Let's hear everyone's thoughts and suspicions about player_2, player_3, player_4, and player_5. We need to work together to eliminate all the Werewolves and secure victory for the Seer, Doctor, and Villagers."

day 1 voting: player_0 had the most votes and was eliminated.
* voted for player_0: player_2, player_5, player_6.
* voted for player_2: player_4.
* voted for player_6: player_0.
* chose not to vote: player_3.
remaining players: player_2 (Villager), player_3 (Villager), player_4 (Werewolf), player_5 (Doctor), player_6 (Seer).

night 2:
* Werewolf: player_4 chose to kill player_2.
* Seer: player_6 saw player_2 is not a Werewolf.
* Doctor: player_5 chose to save player_5.

day 2 announcement: player_2 was killed last night.
remaining players: player_3 (Villager), player_4 (Werewolf), player_5 (Doctor), player_6 (Seer).

day 2 discussion:
* player_3 (Villager) said: "I have been listening carefully to everyone's opinions and observations. It is clear that player_0 was a Werewolf based on player_6's claim as the Seer. However, we should not solely rely on this information. I would like to hear from player_4 and player_5 about their suspicions and any potential insights they might have. Additionally, player_6, since you claimed to be the Seer, can you provide any further information or suspicions about the remaining players? Let's discuss and share our thoughts to make an informed decision for today's voting."
* player_4 (Werewolf) said: "Good morning, everyone. It's unfortunate that player_2 was killed last night. Based on the deductions and information shared yesterday, we have already identified player_0 as a Werewolf. Now, let's focus our attention on the remaining players - player_3, player_5, and player_6. We need to thoroughly discuss and analyze their behaviors and statements to find any inconsistencies or suspicious actions. It's crucial that we work together to eliminate all the Werewolves and secure victory for the Seer, Doctor, and Villagers. Let's start the discussion and share our thoughts on who we believe might be the Werewolves."
* player_5 (Doctor) said: "I have been consistently saving myself as the Doctor, so I can confirm that I am still alive. However, we lost player_2 last night, which means they were not saved. This raises suspicions on player_3 and player_4 as they have not been confirmed or suspected by anyone yet. We should discuss and share our suspicions to uncover the Werewolves among us. It's crucial for us to work together to identify and eliminate the Werewolves to ensure the safety of our village."

* player_6 (Seer) said: "Good morning, everyone. After the events of the previous night, we have confirmed that player_0 was a Werewolf. However, player_2, who was killed last night, was not a Werewolf as I saw them and they were innocent. This means that the remaining Werewolves are still among us. I believe we should focus our attention on player_3, player_4, and player_5 as potential suspects. I would like to hear from each of you about your observations and suspicions regarding these players. It's important that we work together to uncover the true identities of the Werewolves and secure victory for the Seer, Doctor, and Villagers. Let's discuss and share our thoughts on who we believe might be the Werewolves."

day 2 voting: player_5 had the most votes and was eliminated.
* voted for player_5: player_3, player_4.
* voted for player_4: player_5.
* chose not to vote: player_6.
remaining players: player_3 (Villager), player_4 (Werewolf), player_6 (Seer).

night 3:
* Werewolf: player_4 chose to kill player_6.
* Seer: player_6 saw player_4 is a Werewolf.

day 3 announcement: player_5 was killed last night.
remaining players: player_3 (Villager), player_4 (Werewolf).

game result: the Werewolves win the game.

The Villagers Win. This is an example game log where the Villagers win the game.

role assignments:
* player_0: Doctor.
* player_1: Seer.
* player_2: Werewolf.
* player_3: Werewolf.
* player_4: Villager.
* player_5: Villager.
* player_6: Villager.

night 1:
* Werewolves: player_2 and player_3 chose to kill player_0.
* Seer: player_1 saw player_0 is not a Werewolf.
* Doctor: player_0 chose to save player_0.

day 1 announcement: no player was killed last night.
remaining players: player_0 (Doctor), player_1 (Seer), player_2 (Werewolf), player_3 (Werewolf), player_4 (Villager), player_5 (Villager), player_6 (Villager).

day 1 discussion:
* player_0 (Doctor) said: "Good morning everyone! It's day 1, and fortunately, no one was killed last night. I think we should analyze the night 1 announcement and share our thoughts. Let's all discuss who we suspect might be the Werewolves and any suspicions we have based on the limited information we have so far."
* player_1 (Seer) said: "Hello everyone! As the Seer, I can confirm that player_0 is not a Werewolf. Unfortunately, I don't have any information about the other players yet. I would suggest that we all share any suspicions or observations we might have to narrow down our search for the Werewolves."

* player_2 (Werewolf) said: "Hello everyone! As we start the discussion, I want to remind everyone that we should consider all possibilities and not solely rely on player_1's claim as the Seer. While player_0 seems to be cleared for now, it doesn't mean they are completely innocent. We should keep an open mind and look for other clues or suspicions. Let's all share our observations and suspicions to help us narrow down the search for the Werewolves."
* player_3 (Werewolf) said: "I think we should focus our attention on player_1. While they claim to be the Seer and have cleared player_0, we should be cautious as they could be lying. It's possible that player_1 is a Werewolf trying to gain our trust. We need to consider other players' observations and suspicions as well. Let's share any information we have and work together to identify the true Werewolves among us."
* player_4 (Villager) said: "Hello everyone! As a Villager, I don't have any special abilities or information to share. However, player_1 claims to be the Seer and has cleared player_0. While we should remain skeptical and consider other possibilities, it might be beneficial to focus our attention on players who have not been cleared yet. Let's hear from player_2, player_3, player_5, and player_6. Do any of you have any suspicions or observations that could help us identify the Werewolves?"
* player_5 (Villager) said: "I think it's important for us to share any suspicions or observations we may have. I think it is plausible that player_1 is the true Seer and player_0 is innocent. As a Villager, I agree with player_4 that I should focus on players who have not been cleared yet. Player_6, we have not heard from you yet. Do you have any suspicions or observations that could help us identify the Werewolves?"
* player_6 (Villager) said: "I agree with player_1 that we should share any suspicions or observations we might have. Let's keep an open mind and encourage everyone to contribute to the discussion. It would be helpful if players could share any suspicions they might have based on their interactions or observations so far. Together, we can work towards identifying the true Werewolves among us."

day 1 voting: player_2 had the most votes and was eliminated.
* voted for player_2: player_1, player_4, player_5.
* voted for player_1: player_2, player_3.
* chose not to vote: player_0, player_6.
remaining players: player_0 (Doctor), player_1 (Seer), player_3 (Werewolf), player_4 (Villager), player_5 (Villager), player_6 (Villager).

night 2:
* Werewolf: player_3 chose to kill player_1.
* Seer: player_1 saw player_3 is a Werewolf.
* Doctor: player_0 chose to save player_1.

day 2 announcement: no player was killed last night.
remaining players: player_0 (Doctor), player_1 (Seer), player_3 (Werewolf), player_4 (Villager), player_5 (Villager), player_6 (Villager).

day 2 discussion:
* player_0 (Doctor) said: "I would like to share my reasoning with all the players. Based on the night 2 announcement, it is possible that I successfully saved the targeted player. Since I am the Doctor and I saved player_1, it means that the Werewolves targeted player_1 last night. Therefore, player_1, who received two votes in the previous round and was targeted, is likely to be the Seer. I encourage the other players to share their suspicions and any information they might have to help us identify the Werewolves."

* player_1 (Seer) said: "Hello everyone! As the Seer, I have important information to share. Last night, I discovered that player_3 is a Werewolf. This means we have at least one Werewolf among us. I also confirmed that player_0 is not a Werewolf based on my previous investigation. I encourage all of you to share any suspicions or observations you might have to help us identify the remaining Werewolves. Let's work together to eliminate them and secure a victory for the villagers."
* player_3 (Werewolf) said: "Hello everyone! In response to player_1's claim, I have reason to believe that they are either a Werewolf or made a mistake as the Seer. As a Villager myself, I know that player_1 is lying and is likely a Werewolf. We should not solely rely on player_1's information and should continue to gather more evidence. I encourage everyone to share any suspicions or observations they might have to help us identify the remaining Werewolves. Let's work together to eliminate them and secure a victory for the villagers."
* player_4 (Villager) said: "Hello everyone! As a Villager, I don't have any special abilities or information to share. However, we have conflicting claims from player_1 and player_3. Player_1 claims to be the Seer and has cleared player_0, but player_3 doubts their claim and suggests player_1 is not the Seer. It is essential that we gather more information and evidence before making any decisions. I encourage player_1 and player_3 to provide any additional information or reasoning that could help us resolve this conflict. Furthermore, I urge all players to share their suspicions or observations to help us identify the Werewolves. Let's work together to find the truth and make the right decision."
* player_5 (Villager) said: "Hello everyone! After considering the conflicting claims from player_1 and player_3, we need to gather more information to resolve this conflict. It is crucial for player_1 and player_3 to provide additional information or reasoning to support their claims. We should also keep player_4 in mind as they have not been cleared or suspected yet. I encourage all players to share their suspicions and observations to help us identify the Werewolves. Let's work together to find the truth and make the right decision."
* player_6 (Villager) said: "Hello everyone! It's interesting to see the developments from the previous day. Player_1 claims to be the Seer and has shared valuable information about player_0 not being a Werewolf. However, player_3 has raised concerns about the accuracy of player_1's claim, suggesting that they may be a Werewolf or made a mistake. We should consider both possibilities and gather more evidence before making any final judgments. I encourage all players to share their suspicions and observations to help us identify the remaining Werewolves. Let's keep an open mind and continue working together towards a victory for the villagers."

day 2 voting: player_3 had the most votes and was eliminated.
* voted for player_3: player_0, player_1, player_4, player_5, player_6.
* voted for player_1: player_3.
remaining players: player_0 (Doctor), player_1 (Seer), player_4 (Villager), player_5 (Villager), player_6 (Villager).

game result: the Villagers win the game.

Human Evaluation. This is an example game log where 6 humans play with 1 AI agent.

role assignments:
* player_0: Human as Werewolf.
* player_1: Human as Seer.
* player_2: Human as Villager.
* player_3: Human as Villager.
* player_4: Human as Werewolf.
* player_5: AI as Doctor.
* player_6: Human as Villager.

night 1:
* Werewolves: player_0 and player_4 chose to kill player_3.
* Seer: player_1 saw player_0 is a Werewolf.
* Doctor: player_5 chose to save player_5.

day 1 announcement: player_3 was killed last night.
remaining players: player_0 (Human as Werewolf), player_1 (Human as Seer), player_2 (Human as Villager), player_4 (Human as Werewolf), player_5 (AI as Doctor), player_6 (Human as Villager).

day 1 discussion:
* player_0 (Human as Werewolf) said: "I'm just a Villager and I have no information, the Seer should come out and give us information."
* player_1 (Human as Seer) said: "hhh I'm the Seer, and I saw player_0 is a Werewolf. We have lost player_3 and must vote out a Werewolf today, or the Werewolf will kill another player and we will lose."

* player_2 (Human as Villager) said: "Case 1: player 1 is the Seer and player 0 is the Werewolf. Case 2: player 1 is a Werewolf pretending to be the Seer and frames player 0. We need more information and vote out one of them today."
* player_4 (Human as Werewolf) said: "Nonono, only case 2 is true because I'm the Seer, I saw player_5 is not a Werewolf. player_1 is a Werewolf eager to vote out good players and we should all vote player_1. I'll keep an eye on the voting result and check identity tomorrow night."
* player_5 (AI as Doctor) said: "I agree with player_4 that we should vote out player_1. Player_1's claim of being the Seer is suspicious, especially since player_4 has confirmed being the actual Seer and saw that I am not a Werewolf. Let's eliminate player_1 and ensure the safety of our village."
* player_6 (Human as Villager): "1 say he is the Seer and 0 is a Werewolf, 4 then said he is the Seer and 5 is not a Werewolf. I don't know why 5 think 4 is true and 1 is suspicious, 4 can be a Werewolf pretending Seer and give you good credit. It seems that 5 and 4 is on the same side or just stupid. I think we should agree on the vote. If you believe 1, vote 0. If you believe 4, vote 1. I'll vote 1."

day 1 voting: player_0 had the most votes and was eliminated.
* voted for player_0: player_1, player_2, player_6.
* voted for player_1: player_0, player_4.
* voted for player_6: player_5.
remaining players: player_1 (Human as Seer), player_2 (Human as Villager), player_4 (Human as Werewolf), player_5 (AI as Doctor), player_6 (Human as Villager).

night 2:
* Werewolf: player_4 chose to kill player_6.
* Seer: player_1 saw player_4 is a Werewolf.
* Doctor: player_5 chose to save player_4.

day 2 announcement: player_6 was killed last night.
remaining players: player_1 (Human as Seer), player_2 (Human as Villager), player_4 (Human as Werewolf), player_5 (AI as Doctor).

day 2 discussion:
* player_0 (Human as Seer) said: "I think the situation is very clear. If I'm not a Seer, then 3 good players are dead and we have 2 Werewolves among the remaining 4 players and the game is over. But the game is not over, so player_0 must be a dead Werewolf and I'm the true Seer. Everyone should believe my words now. I checked player_4 and he is a Werewolf. Let's vote him and win the game."

* player_2 (Human as Villager) said: "Agree. player 1 is a confirmed Seer. We don't need to listen to player 4's lie. Just vote him."
* player_4 (Human as Werewolf) said: "" (empty)
* player_5 (AI as Doctor) said: "I agree with player_1 being the Seer. Player_4's confirmation of my identity as the Villager is misleading and player_1's correct identification of player_0 as a Werewolf on day 1 provide strong evidence. We should vote to eliminate player_4 as he claimed to be the Seer but provided false information about player_1 being a Werewolf."

day 2 voting: player_4 had the most votes and was eliminated.
* voted for player_4: player_1, player_2, player_5.
* chose not to vote: player_4.
remaining players: player_1 (Human as Seer), player_2 (Human as Villager), player_5 (AI as Doctor).

game result: the Villagers win the game.