GRATR: Zero-Shot Evidence Graph Retrieval-Augmented Trustworthiness Reasoning

Anonymous ACL submission

Abstract

Trustworthiness reasoning aims to enable agents in multiplayer games with incomplete information to identify potential allies and adversaries, thereby enhancing decision-making. In this paper, we introduce the graph retrieval-augmented trustworthiness reasoning (GRATR) framework, which retrieves observable evidence from the game environment to inform decision-making by large language models (LLMs) without requiring additional training, making it a zero-shot approach. Within the GRATR framework, agents first observe the actions of other players and evaluate the resulting shifts in inter-player trust, constructing a corresponding trustworthiness graph. During decision-making, the agent performs multi-hop retrieval to evaluate trustworthiness toward a specific target, where evidence chains are retrieved from multiple trusted sources to form a comprehensive assessment. Experiments in the multiplayer game *Werewolf* demonstrate that GRATR outperforms the alternatives, improving reasoning accuracy by 50.5% and reducing hallucination by 30.6% compared to the baseline method. Additionally, when tested on a dataset of Twitter tweets during the U.S. election period, GRATR surpasses the baseline method by 10.4% in accuracy, highlighting its potential in real-world applications such as intent analysis.

1 Introduction

In multiplayer games with incomplete information, trustworthiness reasoning is critical for evaluating the intentions of players, who may conceal their true motives through actions, dialogue, and other observable behaviors. Autonomous agents analyze the trustworthiness of players based on observable actions to identify potential allies and adversaries (Fig. 1). Current methods supporting such reasoning include symbolic reasoning, evidential theory (Liu et al., 2021), Bayesian reasoning (Wojtowicz and DeDeo, 2020; Sohn and Narain, 2021; Wan and Du, 2021), and reinforcement learning (Wan et al., 2021; Wang et al., 2020; Tiwari et al., 2021). While effective, these methods struggle to address the complexity of natural language interactions, the ambiguity of player behavior, and the dynamic nature of strategic decision-making in such environments.

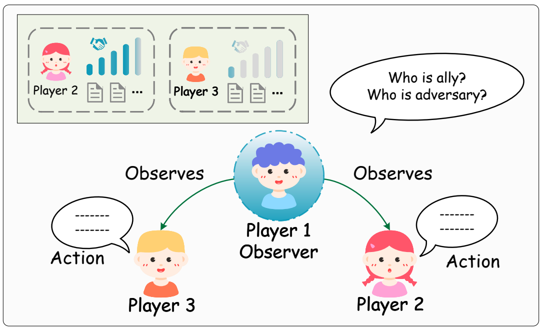


Figure 1: Illustration of trustworthiness reasoning. Agent observes the actions of other players to gather evidence, and then evaluates inter-player trust and informs decision-making.

To address these limitations, large language models (LLMs) offer a promising approach for trustworthiness reasoning in multiplayer games, owing to their advanced natural language understanding and generation capabilities (Brown et al., 2020; Kenton and Toutanova, 2019; Radford et al., 2019). LLMs utilize these capabilities to interpret complex dialogues, infer latent intentions, and detect deceptive behaviors from contextual cues. However, LLMs face inherent challenges, including the risk of hallucination and knowledge obsolescence (Ji et al., 2023; Maynez et al., 2020). To mitigate these issues, techniques such as supervised fine-tuning and reinforcement learning have been proposed to enhance their reasoning performance (Ouyang et al., 2022; Stiennon et al., 2020). Nonetheless, these approaches often require extensive historical data and well-defined reward signals, which may be scarce or unavailable in real-world game scenarios.

To enhance the capabilities of LLMs in

dynamic, knowledge-intensive environments, retrieval-augmented generation (RAG) (Gao et al., 2024; Zhao et al., 2024) has emerged as a promising alternative. RAG addresses the limitations of LLMs by integrating an external retrieval mechanism that dynamically fetches relevant information to augment the generation process. In the RAG framework, a retriever first indexes and retrieves pertinent data chunks, which are then combined with an input query to refine the generation process. This approach mitigates issues such as knowledge obsolescence and hallucination by incorporating up-to-date and contextually relevant information, making it a promising solution for trustworthiness reasoning in multiplayer games with incomplete information.

However, trustworthiness reasoning in multiplayer games presents additional challenges that exceed the capabilities of current RAG methods. Specifically, it requires the real-time collection and analysis of statements and actions as evidence exhibited by players. Due to the complexity of player interactions, trustworthiness reasoning for a given player must consider the actions of other players toward that target. This necessitates multi-hop retrieval and synthesis of evidence, which becomes computationally intensive and time-consuming, particularly in scenarios involving many players.

Contributions. We propose a novel method, the graph retrieval-augmented trustworthiness reasoning (GRATR) framework, which constructs a dynamic trustworthiness graph to model player interactions in real time, thus avoiding the computational overhead of retrieving information from large text corpora repeatedly. During the observation phase, agents collect observable evidence to dynamically update the graph's nodes (representing players) and edges (representing trust relationships). During decision-making, GRATR performs multi-hop retrieval to evaluate the trustworthiness of a specific target player, leveraging evidence chains from multiple trusted sources to form a comprehensive assessment. This approach enhances reasoning and decision-making without additional training, making it a zero-shot solution. We validate GRATR in the multiplayer game *Werewolf*, comparing it to baseline LLMs and LLMs with state-of-the-art RAG techniques. The experimental results demonstrate its ability to model dynamic trust relationships and support informed decision-making in complex, incomplete information scenarios. Furthermore, GRATR enhances transparency and traceability by visualizing temporal evidence and evidence chains through the trustworthiness graph, overcoming the limitations of previous methods. Beyond multiplayer games, we also apply GRATR to real-world scenarios, i.e., analyzing the intent behind social media tweets, showcasing its broader applicability.

2 Preliminary

In a multi-player game with incomplete information, the game can be described by the following components:

- **Players**: $P = \{p_1, p_2, \dots, p_n\}$, where p_i represents the i-th player, and each player p_i has a private type $\theta_i \in \Theta_i$, where Θ_i is the set of possible types for player p_i.
- **Actions**: In each round t, player p_i chooses an action $a_i^t \in A_i$, where A_i is the set of available actions for player p_i. A_i is assumed to be finite to simplify the analytical and computational complexity.
- **Observations**: After all players choose their actions, each player p_i receives an observation $o_i^t \in O_i$, where O_i is the set of possible observations for p_i. The observation o_i^t depends on the joint actions $\mathbf{a}^t = (a_1^t, a_2^t, \dots, a_n^t)$ and possibly other public or private signals. Public signals can be observed by all players, while private signals are unique to the particular players, such as the results of a seer's check in the *Werewolf* Game.
- **Objective**: Each player p_i aims to maximize their utility function $U_i(\mathbf{a_i}, \theta_i, \sigma_{-i}, T)$, where σ_{-i} is the strategy distribution of others, and T is the trustworthiness judgment. Greater accuracy in T (better trustworthiness reasoning) leads to higher utility. θ and $\mathbf{a}$ of traditional methods are complex, but our algorithm simplifies this by classifying characters as enemies/ allies and actions as protective/ aggressive. While theoretical analysis is complex due to model evaluation challenges, experimental results indirectly validate strong trustworthiness reasoning via action scores.

The game proceeds as follows:

- At the beginning of each round t, each player p_i observes h_i^t and selects an action $a_i^t = s_i(h_i^t, \theta_i)$, where h means the history and s means the strategy function for choosing actions based on history and private type.
- After all actions $\mathbf{a}^t$ are chosen, players receive observations o_i^t.

- Players update their beliefs $\sigma_i^{t+1}(\theta_{-i} \mid h_i^{t+1})$ based on the new history h_i^{t+1} that includes o_i^t and a_i^t.
- The game continues for a fixed number of rounds T, or until a stopping condition is met.

3 Methodology

To enhance the effectiveness of LLM reasoning, especially in environments where trust and strategic interactions are crucial, it is essential to retrieve the most relevant evidence from historical data. This motivated us to develop a framework where the information observed by agents is structured into a graph-based evidence base. By maintaining this evidence graph, we can retrieve related evidence chains, augment LLM reasoning, and mitigate the issues of hallucination and opacity. This methodology forms the foundation of our proposed GRATR system. Figure 2 presents the framework of GRATR. The process begins with the initialization of a trustworthiness graph when an agent participates in the game. Observations made by the agent are analyzed using the LLM to extract evidence and assess its credibility, which is then used to update the trustworthiness graph G. Through multi-hop retrieval on G, evidence chains are constructed to evaluate the trustworthiness of other players. Finally, the system updates trustworthiness relationships among players based on the gathered evidence, leveraging the graph structure to provide a transparent and well-grounded reasoning process.

3.1 The Trustworthiness Graph Initialization

Assume an agent participates as a player in a multiplayer game with incomplete information, maintaining a directed graph G^t to record historical observations h^t up to round t as a dynamic evidence base. This graph G^t serves as the foundation for the agent's reasoning process, enabling the retrieval and use of real-time evidence. The graph consists of two core components: nodes and edges.

Nodes: Each node in the graph G^t represents a player p_i and stores two parameters.

- **Trustworthiness of Nodes** $T^t(p_i) \in [-1, 1]$: The perceived trustworthiness of player p_i by the agent at time t. When $T^t(p_i) > \epsilon$, the agent regards player p_i as an ally, when $T^t(p_i) < -\epsilon$, the agent regards player p_i an adversary; otherwise, the agent regards player p_i as indifferent.
- **Historical Observations** $h^t(p_i)$: The history of observations gathered by the agent about player p_i up to round t, serving as the evidence base.

Edges: Each directed edge $e^t(p_i, p_j)$ connects player node p_i to player node p_j and stores two parameters.

- **Evidence List** $D^t(p_i, p_j)$: This list contains a set of evidence items $d^t(p_i, p_j)$ that record the actions of player p_i towards player p_j as observed by the agent. Each evidence item includes the specific action taken and its associated credibility $c^t(p_i, p_j)$, indicating the significance of this action in assessing trustworthiness.
- **Trustworthiness of Edges** $T^t(p_i, p_j)$: This weight reflects the trustworthiness of p_i in p_j from the agent's perspective, determined by the accumulated evidence in the evidence list.

GRATR initializes a directed graph where nodes represent players and edges denote trustworthiness. At the initial time $t = 0$, the edge weight is set to zero, i.e., $T^0(p_i, p_j) = 0$, and the evidence list $D^0(p_i, p_j)$ is empty, indicating no prior observations or assessments. The graph structure is fixed, but edge weights and evidence lists are dynamically updated during interactions.

3.2 The Trustworthiness Graph Update

When the agent receives a new observation $o^t(p_i)$ following an action by player p_i, the evidence graph G^t must be updated to incorporate this new information. This ensures that G^t accurately represents the current state of trustworthiness among the players at time t.

The agent uses the LLM to extract evidence items $d^t(p_i, p_j)$ and their corresponding weights $c^t(p_i, p_j)$ from the observation $o^t(p_i)$ (the related prompt used for LLM interactions is provided in Appendix 1.1). For each directed edge $e^t(p_i, p_j)$ in the graph, the evidence list $D^{t+1}(p_i, p_j)$ associated with the edge $e^t(p_i, p_j)$ is updated by adding the new evidence $d^t(p_i, p_j)$:

$$D^{t+1}(p_i, p_j) = D^t(p_i, p_j) \cup \{d^t(p_i, p_j)\}. \quad (1)$$

The sign of $c^t(p_i, p_j)$ indicates the nature of p_i's intention towards p_j: negative for hostility and positive for support, with $|c^t(p_i, p_j)|$ reflecting its strength. Note that the evidence list $D^t(p_i, p_j)$ is updated with the new observation, and the edge weight $T^t(p_i, p_j)$ is adjusted accordingly during retrieval to maintain an accurate representation of trustworthiness.

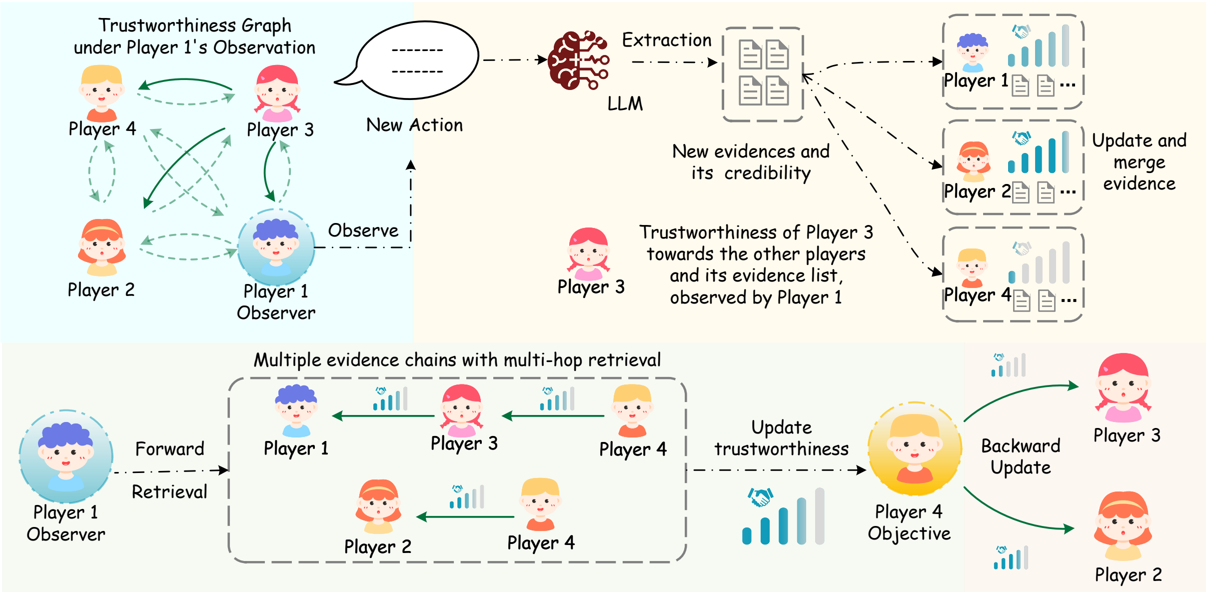


Figure 2: The overall framework of GRATR: Step 1. An agent participates in the game as player 1, and initializes a trustworthiness graph G. Step 2. When player 1 receives a new observation $o_1^t(p_3)$ following an action by a_3^t at time t, it uses an LLM to extract the action into new evidence and its credibility and then updates and merges evidence on the graph G^t. Step 3. Player 1 obtains multiple evidence chains by multi-hop retrieval and updates the trustworthiness of player 4. Step 4. Update the trustworthiness of player 4 towards player 2 and player 3.

Meanwhile, the agent updates the trustworthiness of p_j in response to the evidence items extracted from the LLM. The update depends on the two factors: the perceived trustworthiness of p_i, the credibility $c^t(p_i, p_j)$ of the $d^t(p_i, p_j)$, which also represents the p_i's confidence of the p_j's current role classification $\mathcal{R}^t(p_j)$. The updated trustworthiness $T_i^t(p_j)$ is computed as follows:

$$u^t(p_j) = T^t(p_i) \cdot c^t(p_i, p_j), \tag{2}$$

$$T^{t+1}(p_j) = \begin{cases} T^t(p_j), & \text{if } |u^t(p_j)| \leq |T^t(p_j)|, \\ u^t(p_j), & \text{if } |u^t(p_j)| > |T^t(p_j)|. \end{cases} \tag{3}$$

$u^t(p_j)$ represents the inference of p_j's trustworthiness through the observation $o^t(p_i)$.

3.2.1 Evidence Merging

In this phase, the objective is to aggregate and evaluate the various evidence collected by the agent over time, specifically related to the interactions between players p_i and p_j. Assume that the agent has n pieces of evidence $d^t(p_i, p_j)$ towards player p_j in the evidence list $D^t(p_i, p_j)$ associated with the directed edge $e^t(p_i, p_j)$. The evidence is sorted in chronological order, with each piece of evidence having an associated weight $c^t(p_i, p_j)$ and a temporal importance factor ρ. The updated edge weight $T^{t+1}(p_i, p_j)$ is computed as follows:

$$T^{t+1}(p_i, p_j) = \tanh\left(\textstyle\sum_{k=1}^{n} \rho^{n-k} \cdot c^t(p_i, p_j)\right). \tag{4}$$

The impact of evidence decreases over time, with more recent evidence having greater influence. The $\tanh$ function is used to constrain the edge weight $T^{t+1}(p_i, p_j)$ within the interval $[-1, 1]$, providing a bounded measure of the trustworthiness between players. The motivation and intuition behind the difference between Eq. (4) and Eq. (2) (3) lie in the distinct roles of $T^{t+1}(p_j)$ and $T^{t+1}(p_i, p_j)$. Updating $T^{t+1}(p_j)$ with a single piece of evidence (Eq. (2) (3)) reflects real-time adjustments based on immediate observations, focusing on simplicity and responsiveness. In contrast, updating $T^{t+1}(p_i, p_j)$ by merging all evidence (Eq. (4)) accounts for the chronological accumulation and potential conflicts of past observations, aiming to provide a comprehensive and accurate trustworthiness assessment over time. This distinction ensures both adaptability to new evidence and robustness in reasoning about long-term confidence.

3.3 Graph Retrieval Augmented Reasoning

During the agent's turn, particularly when deciding on an action involving player p_o, the reasoning process is augmented by retrieving and leveraging relevant evidence from the evidence graph G^t. This graph-based retrieval augments the player's trustworthiness assessment by incorporating historical evidence into the reasoning process. The retrieval process is divided into three key phases: evidence merging, forward retrieval, backward update, and reasoning.

3.3.1 Forward Retrieval

Given that the agent holds a trustworthiness value $T^t(p_1)$ towards player p_1, if there exists a evidence chain $\mathcal{C}_n : p_o \to p_{o-1} \to \cdots \to p_1$, the value $V_{\mathcal{C}_n}$ of this evidence chain and the cumulative trustworthiness update $u_n^t(p_o)$ towards player p_o are computed as follows:

$$V_{\mathcal{C}_n} = \sum_{k=1}^{o-1} T^t(p_{k+1}) \cdot T^t(p_{k+1}, p_k), \quad (5)$$

$$u_n^t(p_o) = T^t(p_1) \cdot \prod_{k=1}^{o-1} T^t(p_{k+1}, p_k). \quad (6)$$

The uncertainty associated with the chain $\mathcal{C}_n$ is defined by:

$$H(\mathcal{C}_n) = -|u_n^t(p_o)| \log_2 |u_n^t(p_o)|. \quad (7)$$

For the player p_o with m related evidence chains $\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_m$, the updated trustworthiness $T_i^t(p_o)$ is given by:

$$T^{t+1}(p_o) = \frac{\sum_{n=1}^{m}(V_{\mathcal{C}_n} - H(\mathcal{C}_n)) \cdot u_n^t(p_o)}{\sum_{n=1}^{m}(V_{\mathcal{C}_n} - H(\mathcal{C}_n))}. \quad (8)$$

The trustworthiness update is a weighted sum of the relevant evidence chains, where each chain's weight is determined by its value and associated uncertainty.

3.3.2 Backward Update

Once $T^t(p_o)$ is updated, the edge weights associated with the relevant evidence chains need to be updated in reverse:

$$T^{t+1}(p_o, p_{o-1}) = \gamma \cdot \frac{T^{t+1}(p_o)}{T^{t+1}(p_{o-1})} + T^t(p_o, p_{o-1}). \quad (9)$$

Here, γ represents the learning rate for the backward update, and p_{o-1} is the preceding player in the evidence chain $\mathcal{C}_n$ $(n = 1, 2, \ldots m)$.

3.3.3 Reasoning

After updating the trustworthiness of the agent towards p_o, a summary and reasoning are made based on the trustworthiness of player p_o and the relevant evidence chains retrieved. Specifically, the trustworthiness of the agent towards p_o and the evidence chains are combined into a prompt sent to LLM, which ultimately returns the summary and reasoning of the player p_o. The prompt used is shown in Appendix 1.2.

4 Experiments

In this section, we evaluate the enhancement of LLMs' reasoning and intent analysis capabilities with GRATR, testing it on both the *Werewolf* game and the Twitter dataset from the 2024 U.S. election. We use pure LLMs as the baseline, alongside state-of-the-art algorithms, including NativeRAG (Lewis et al., 2020), RerankRAG (Sun et al., 2023), and LightRAG (Guo et al., 2024), for comparison.

4.1 Experiment on Werewolf Game

We implemented our GRATR method using the classic multiplayer game *Werewolf* (Xu et al., 2024). The game consists of 8 players, including three leaders (the witch, the guard, and the seer), three werewolves, and two villagers. The history message window size K is set to 15. We use the GPT-3.5-turbo, GPT-4o, GPT-4o-mini, Qwen-Max, and DeepSeek-V3 models as the backend LLMs, with their temperatures set to 0.3 according to the original paper's setup. In each game, four players are assigned to each algorithm, with three players randomly assigned to the leader and werewolf roles, and the remaining player assigned to the village side. The algorithm corresponding to the winning side is considered the winner of the game. Each algorithm participated in 50 games with different backend LLMs.

4.1.1 Win Rate Analysis

Table 1 presents the win rates of LLMs with GRATR in pairwise comparisons against the baseline and LLMs with NativeRAG, RerankRAG, and LightRAG in the *Werewolf* game. The win rates include total win rate (TWR), werewolf win rate (WWR), and leader win rate (LWR).

From the mean TWR in Table 1, it is clear that GRATR significantly outperforms both pure LLMs and LLMs with advanced RAG methods. Except for the match against NativeRAG, where the win rate is 78.4%, GRATR achieves win rates above 80% in all other pairwise competitions. The experimental results support the claim that GRATR effectively enhances LLM reasoning in incomplete information games and improves win rates. More specifically, the results show that GRATR achieves the highest win rate when playing against pure LLMs, followed by LightRAG, RerankRAG, and NativeRAG. This suggests that external retrieval-based techniques are beneficial for enhancing LLM reasoning. Furthermore, while LightRAG, as a graph-based retrieval-augmented generation method, excels at summarization rather than reasoning, and RerankRAG, though a commendable variant, fails to capture the causal relationships of player actions in multi-hop retrieval, which results in its lower performance compared

GRATR vs.	Model	TWR	WWR	LWR
Baseline	GPT-3.5-turbo	76.0%	72.0%	80.0%
	GPT-4o	88.0%	84.0%	92.0%
	GPT-4o-mini	84.0%	76.0%	92.0%
	Qwen-max	94.0%	88.0%	100.0%
	DeepSeek-v3	92.0%	88.0%	96.0%
	Mean	86.8%	81.6%	92.0%
NativeRAG	GPT-3.5-turbo	66.0%	60.0%	72.0%
	GPT-4o	80.0%	76.0%	84.0%
	GPT-4o-mini	78.0%	72.0%	84.0%
	Qwen-max	80.0%	76.0%	84.0%
	SeepAeek-v3	88.0%	80.0%	96.0%
	Mean	78.4%	72.8%	84.0%
RerankRAG	GPT-3.5-turbo	72.0%	76.0%	68.0%
	GPT-4o	90.0%	84.0%	96.0%
	GPT-4o-mini	80.0%	80.0%	80.0%
	Qwen-max	92.0%	84.0%	100.0%
	DeepSeek-v3	90.0%	84.0%	96.0%
	Mean	84.8%	81.6%	88.0%
LightRAG	GPT-3.5-turbo	80.0%	76.0%	84.0%
	GPT-4o	90.0%	96.0%	84.0%
	GPT-4o-mini	84.0%	80.0%	88.0%
	Qwen-max	88.0%	84.0%	90.0%
	DeepSeek-v3	88.0%	96.0%	80.0%
	Mean	86.0%	86.4%	85.2%

Table 1: The total, werewolf, and leader win rates (TWR, WWR, LWR) of GRATR in pairwise comparisons against the baseline and LLMs with NativeRAG, RerankRAG, and LightRAG in the *Werewolf* game.

to NativeRAG.

Further analysis of the win rates when the agent plays as a werewolf or leader reveals that the agent performs significantly better as a leader. Notably, when using Qwen-Max, the win rate reaches 100% against both Baseline and RerankRAG. This can be attributed to the game dynamics of *Werewolf*, where the werewolf must deceive other players to conceal their identity, whereas the leader only needs to reason out who the werewolf is. The high win rates for the leader role provide evidence that GRATR enhances the reasoning ability of LLMs, enabling them to identify the concealed werewolf. Although GRATR also performs well when the agent plays as a werewolf, the deception required for this role presents a greater challenge for LLMs.

The experiment shows that different LLMs have a significant impact on the results. As shown in Table 1, GRATR achieves better performance with GPT-4o, Qwen-Max, and DeepSeek-V3, with win rate improvements of 4%, 2%, 10%, and 4%, respectively. Publicly available evidence (Chiang et al., 2024; Contributors, 2023) indicates that GPT-4o, Qwen-Max, and DeepSeek-V3 exhibit stronger reasoning capabilities compared to GPT-3.5-turbo and GPT-4o-mini. Therefore, we conclude that stronger LLMs further amplify the performance advantages of GRATR.

4.1.2 Action Scores

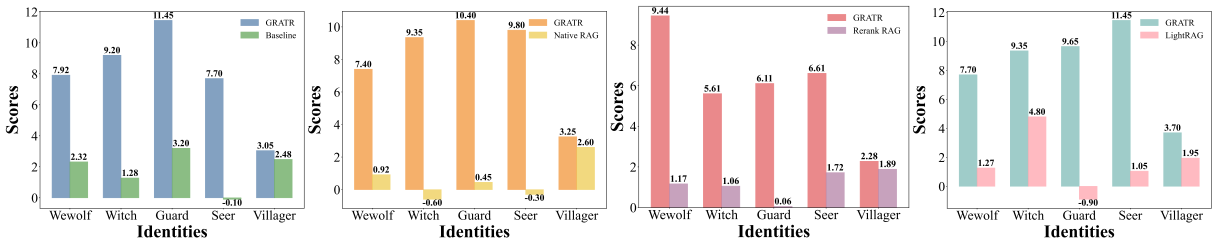
In the *Werewolf* game, win rate alone evaluates the overall performance of the team, but it does not fully reflect the individual agent's actual performance. Therefore, this section further analyzes the agent's action scores in each game to highlight its superiority in reasoning, social interaction, role identification, and other aspects. Table 2 presents the detailed scoring breakdown for agents under different identities, including scores for correct and incorrect votes. Additionally, each winning player is pre-allocated a base score of 5 points.

	Werewolf	**Witch**	**Guard**	**Seer**	**Villager**
Correct	0.5	1.5	1.5	1.5	1.0
Incorrect	-0.5	-1.5	-1.5	-1.5	-1.0

Table 2: Correct and incorrect action scores for different identities in Werewolf game.

Fig. 3 presents the action scores of GRATR vs. baseline LLM, LLM with NativeRAG, RerankRAG, and LightRAG in the *Werewolf* game. Overall, when the agent plays as a villager, the score differences are minimal, generally under 2 points, and even less than 1 point when compared to baseline LLM and LLMs with NativeRAG or RerankRAG. This indicates that, on average, the agent makes fewer than one error per game round. It is important to note that in the *Werewolf* game, villagers have no prior information other than their own identity, so all reasoning is based on the inconsistencies and consistencies in players' actions rather than validating with prior knowledge. Therefore, the superior behavior scores of GRATR when the agent plays as a villager demonstrate the algorithm's multi-hop retrieval capability and its advantage in causal reasoning.

For identities other than the villager, such as the werewolf and leader, the action score differences are significantly larger. A major portion of this difference stems from the win rate, as the winning side is awarded a base score of 5 points. The remaining differences are due to the correctness of the agent's actions. For example, when the agent plays as a werewolf, the score difference is greater than 5 points, indicating that the agent made incorrect actions. However, in the *Werewolf* game, the werewolf has prior knowledge of all teammates and opponents, so any errors in actions are primarily



(a) GRATR vs. baseline. (b) GRATR vs. NativeRAG. (c) GRATR vs. RerankRAG. (d) GRATR vs. LightRAG.

Figure 3: Action scores of GRATR vs. baseline LLM, LLM with NativeRAG, RerankRAG, and LightRAG in Werewolf game

attributed to LLM hallucination. While there may be potential deception and disguise involved, this section does not delve further into this aspect, as no additional supporting information is available.

4.1.3 Hallucination Detection

This section detects LLM hallucination by analyzing the agent's thinking and actions. If the agent's reasoning aligns with their true stance, the process is correct. If the agent's actions conflict with their reasoning, it indicates strategic deception. If the agent's thinking deviates from their real stance, it signals cognitive bias or hallucination. We manually labeled the consistency of the agent's thinking and actions, with the results shown in Table 3.

Method	Identity	Correct Reasoning	Deception	Hallucination
Baseline	Werewolf	61.9%	1.5%	36.6%
	Leader	69.7%	0.5%	29.8%
GRATR	Werewolf	**85.1%**	**11.4%**	**3.5%**
	Leader	**97.0%**	1.3%	**1.7%**
NativeRAG	Werewolf	79.1%	3.1%	17.8%
	Leader	84.4%	1.7%	13.9%
RerankRAG	Werewolf	74.6%	9.4%	16.0%
	Leader	83.9%	**6.0%**	10.1%
LightRAG	Werewolf	76.2%	10.1%	13.7%
	Leader	86.2%	4.5%	9.3%

Table 3: Correct Reasoning, deceive, and hallucination rates of different methods in different identities.

The data in the table shows that the GRATR method outperforms others in both correct reasoning and hallucination mitigation. It improves correct reasoning by at least 6% and 12.6% for the Werewolf and Leader identities, respectively. It also exhibits an 11.4% deception rate for the Werewolf identity. However, the deception rate is lower for the Leader identity, as the Leader typically needs to reveal their identity to guide the villagers to victory, with deception used only for strategic purposes. Most importantly, GRATR significantly mitigates LLM hallucination, reducing them by a factor of 10 for the Werewolf identity and 17 for the Leader identity compared to the baseline. These results strongly support GRATR's effectiveness in enhancing LLM reasoning capabilities and reducing hallucination.

4.2 Experiment on Intent Analysis

In this section, we utilize a public dataset of Twitter tweets at the time of the U.S. election (Balasubramanian et al., 2024) to evaluate GRATR for their intent analysis capability. This dataset comprehensively captures large-scale social media discourse related to the 2024 U.S. presidential election. The dataset includes approximately 27 million publicly available political tweets collected between May 1 and November 1, 2024. Each tweet is accompanied by detailed metadata, including precise timestamps and multi-dimensional user engagement metrics (such as reply count, retweet count, like count, and view count).

4.2.1 Experimential Results

We define five possible tweet intents: Anti-Democrat, Anti-Republican, Pro-Democrat, Pro-Republican, and Neutral (Ibrahim et al., 2024). To evaluate the accuracy of the algorithm, we manually annotated the intents of 26,523 valid tweets (i.e., tweets that are not garbled and are meaningful). For this, we generalized each tweet to the individual who sent it (since a person may have sent multiple tweets) and followed the timeline to simulate a real Twitter discussion environment. In this setup, we applied LLMs with GRATR, baseline LLMs, LLMs with NativeRAG, RerankRAG, and LightRAG to analyze the stance of each individual and further analyze the intent of their tweets. Table 5 presents the accuracy and macro F1-score of all comparison algorithms for intent analysis of the tweets.

	Baseline	GRATR	NativeRAG	RerankRAG	LightRAG
Accuracy	0.818	**0.922**	0.868	0.879	0.891
Macro F1	0.809	**0.914**	0.869	0.878	0.893

Table 4: Accuracy and Macro F1-score of baseline LLMs, LLMs with GRATR, NativeRAG, RerankRAG, and LightRAG on intent analysis of the tweets.

Among all the methods, LLMs enhanced with

GRATR achieve the highest accuracy of 0.922 and a macro F1-score of 0.914, demonstrating superior performance. The accuracy metric reflects the proportion of correctly classified tweets out of the total. However, accuracy alone may not fully capture performance when dealing with imbalanced data, such as political tweets, where certain intents (e.g., Pro-Democrat or Anti-Republican) are more prevalent than others. In these cases, the macro F1-score provides a more balanced evaluation by considering both precision and recall for each intent category individually, ensuring equal weight for less frequent categories. The significantly higher macro F1-score of LLMs with GRATR (0.914), compared to the baseline models (0.809), indicates that GRATR enhances the model's ability to accurately predict all intents, especially the subtle or less frequent ones, in politically charged discourse. This result highlights GRATR's capacity to integrate contextual and temporal information, which is critical for understanding the nuanced intents of tweets, particularly in dynamic environments like social media during a presidential election. Additionally, LLMs with RAG, including NativeRAG, RerankRAG, and LightRAG, all outperform the baseline LLMs, underscoring the effectiveness of RAG in improving intent analysis.

5 Related Work

Reasoning Task. In incomplete information games, players enhance decision-making by reasoning through observed data and analyzing behaviors in real time, despite misleading information (Wu et al., 2024; Zhang et al., 2024; Cheng et al., 2024; Qin et al., 2024; Costarelli et al., 2024). Traditional methods like Bayesian approaches (Zamir, 2020), evolutionary game theory (Deng et al., 2015), and machine learning techniques such as Monte Carlo tree search (Cowling et al., 2012) and reinforcement learning (RL) (Heinrich and Silver, 2016) have been used, with RL gaining prominence for its inference capabilities. However, RL's reliance on domain-specific data limits generalizability. Large language models (LLMs) offer an alternative with extensive knowledge and language capabilities, as shown by Xu et al. (Xu et al., 2023), who combined LLMs and RL for strategic language agents. Yet, LLMs face challenges like high training costs, inability to update data in real time, and hallucination, hindering real-time reasoning in multiplayer games. RAG addresses these limitations, enhancing LLMs' reasoning in dynamic game environments.

Retrieval Augmented Generation. RAG enhances LLMs by integrating external knowledge retrieval. NativeRAG (Lewis et al., 2020) involves document chunking/encoding, vector-based semantic retrieval, and prompt construction. While efficient, it often retrieves low-relevance chunks. RerankRAG improves accuracy by adding a reranking step (e.g., transformer-based cross-encoders) to prioritize relevant chunks (Sun et al., 2023). GraphRAG uses knowledge graphs, modeling entities as nodes and relationships as edges, supporting multi-hop reasoning, and capturing complex dependencies for deeper queries (Edge et al., 2024). Both Rerank and GraphRAG increase computational complexity. LightRAG (Guo et al., 2024) mitigates this with lightweight strategies like heuristic filtering, balancing efficiency and relevance. Retrieval-Augmented Reasoning (RAR) (Tran et al., 2024) integrates dynamic knowledge retrieval with reasoning modules, improving temporal relevance but facing challenges in multi-step inference and trustworthiness verification.

6 Conclusion

This paper introduces GRATR, a novel framework that enhances agent reasoning in multiplayer games with incomplete information through trustworthiness reasoning. Unlike the existing RAG works, GRATR addresses the limitations in handling temporal and causal evidence in long-term games by implementing a dynamic trustworthiness graph that updates in real-time with new evidence. The framework consists of two main phases. During the agent observation phase, evidence is collected to update the nodes and edges of the graph. During the agent's turn, relevant evidence chains are retrieved to assess the trustworthiness of the player's actions, thereby improving reasoning and decision-making. Experiments conducted in the multiplayer game *Werewolf* demonstrate that GRATR outperforms existing methods in terms of game winning rate, overall performance, and reasoning ability, while mitigating LLM hallucination. Additionally, GRATR enables the traceability and visualization of the reasoning process through time-based evidence and evidence chains. Furthermore, GRATR's application to the U.S. election Twitter dataset highlights its effectiveness in intent analysis, showcasing its potential for real-world applications.

Limitations

While the GRATR framework demonstrates promising performance in our experiments, there are several limitations. First, the computational complexity of the framework increases with the number of players and game progression, particularly during multi-hop retrieval, which may affect real-time performance. Second, the framework's performance relies on the reasoning capabilities of LLMs, with significant variations across different models. Finally, in real-world scenarios, the framework may encounter more uncertainties and noise. In future work we will further optimize the graph structure of the framework and improve the robustness.

Ethics Statement

This research involves several ethical considerations that we have carefully addressed:

- **Data Privacy and Security**: In our experiments with the Twitter dataset, we only used publicly available tweets and ensured that all data collection and processing complied with Twitter's terms of service. We did not collect or store any personal information beyond what was publicly accessible.

- **AI Safety and Fairness**: Our framework is designed to enhance reasoning capabilities while maintaining transparency and accountability. The trustworthiness graph structure allows for clear traceability of decision-making processes, helping to prevent potential biases or unfair outcomes.

- **Social Impact**: While our framework demonstrates potential applications in social media analysis, we acknowledge the importance of responsible deployment. The technology should not be used to manipulate public opinion or interfere with democratic processes.

- **Transparency**: We have made our methodology and experimental results fully transparent, including limitations and potential risks. This transparency helps ensure that the technology can be properly evaluated and used responsibly.

- **Research Ethics**: All experiments were conducted with appropriate safeguards and ethical guidelines in place. We ensured that our research did not cause harm to any individuals or groups.

We believe that these ethical considerations are crucial for the responsible development and deployment of AI technologies in social and political contexts.

References

Ashwin Balasubramanian, Vito Zou, Hitesh Narayana, Christina You, Luca Luceri, and Emilio Ferrara. 2024. A public dataset tracking social media discourse about the 2024 us presidential election on twitter/x. *arXiv preprint arXiv:2411.00376*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Pengyu Cheng, Tianhao Hu, Han Xu, Zhisong Zhang, Yong Dai, Lei Han, and Nan Du. 2024. Self-playing Adversarial Language Game Enhances LLM Reasoning. *arXiv preprint*. ArXiv:2404.10642 [cs].

Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios Nikolas Angelopoulos, Tianle Li, Dacheng Li, Hao Zhang, Banghua Zhu, Michael Jordan, Joseph E Gonzalez, and 1 others. 2024. Chatbot arena: An open platform for evaluating llms by human preference. *arXiv preprint arXiv:2403.04132*.

OpenCompass Contributors. 2023. Opencompass: A universal evaluation platform for foundation models. `https://github.com/open-compass/opencompass`.

Anthony Costarelli, Mat Allen, Roman Hauksson, Grace Sodunke, Suhas Hariharan, Carlson Cheng, Wenjie Li, and Arjun Yadav. 2024. GameBench: Evaluating Strategic Reasoning Abilities of LLM Agents. *arXiv preprint*. ArXiv:2406.06613 [cs].

Peter I Cowling, Edward J Powley, and Daniel Whitehouse. 2012. Information set monte carlo tree search. *IEEE Transactions on Computational Intelligence and AI in Games*, 4(2):120–143.

Xinyang Deng, Deqiang Han, Jean Dezert, Yong Deng, and Yu Shyr. 2015. Evidence combination from an evolutionary game theory perspective. *IEEE transactions on cybernetics*, 46(9):2070–2082.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, and Jonathan Larson. 2024. From local to global: A graph rag approach to query-focused summarization. *arXiv preprint arXiv:2404.16130*.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang. 2024. Retrieval-Augmented Generation for Large Language Models: A Survey. *arXiv preprint*. ArXiv:2312.10997 [cs].

Zirui Guo, Lianghao Xia, Yanhua Yu, Tu Ao, and Chao Huang. 2024. Lightrag: Simple and fast retrieval-augmented generation.

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

Hazem Ibrahim, Farhan Khan, Hend Alabdouli, Maryam Almatrooshi, Tran Nguyen, Talal Rahwan, and Yasir Zaki. 2024. Analyzing political stances on twitter in the lead-up to the 2024 us election. *arXiv preprint arXiv:2412.02712*.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of naacL-HLT*, volume 1. Minneapolis, Minnesota.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Peide Liu, Mengjiao Shen, Fei Teng, Baoying Zhu, Lili Rong, and Yushui Geng. 2021. Double hierarchy hesitant fuzzy linguistic entropy-based todim approach using evidential theory. *Information Sciences*, 547:223–243.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. On faithfulness and factuality in abstractive summarization. *arXiv preprint arXiv:2005.00661*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Zhanyue Qin, Haochuan Wang, Deyuan Liu, Ziyang Song, Cunhang Fan, Zhao Lv, Jinlin Wu, Zhen Lei, Zhiying Tu, Dianhui Chu, Xiaoyan Yu, and Dianbo Sui. 2024. UNO Arena for Evaluating Sequential Decision-Making Capability of Large Language Models. *arXiv preprint*. ArXiv:2406.16382 [cs].

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, and 1 others. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Hansem Sohn and Devika Narain. 2021. Neural implementations of bayesian inference. *Current Opinion in Neurobiology*, 70:121–129.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. 2020. Learning to summarize with human feedback. *Advances in Neural Information Processing Systems*, 33:3008–3021.

Weiwei Sun, Lingyong Yan, Xinyu Ma, Shuaiqiang Wang, Pengjie Ren, Zhumin Chen, Dawei Yin, and Zhaochun Ren. 2023. Is chatgpt good at search? investigating large language models as re-ranking agents. *arXiv preprint arXiv:2304.09542*.

Prayag Tiwari, Hongyin Zhu, and Hari Mohan Pandey. 2021. Dapath: Distance-aware knowledge graph reasoning based on deep reinforcement learning. *Neural Networks*, 135:1–12.

Hieu Tran, Zonghai Yao, Junda Wang, Yifan Zhang, Zhichao Yang, and Hong Yu. 2024. Rare: Retrieval-augmented reasoning enhancement for large language models. *arXiv preprint arXiv:2412.02830*.

Guojia Wan and Bo Du. 2021. Gaussianpath: a bayesian multi-hop reasoning framework for knowledge graph reasoning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 4393–4401.

Guojia Wan, Shirui Pan, Chen Gong, Chuan Zhou, and Gholamreza Haffari. 2021. Reasoning like human: Hierarchical reinforcement learning for knowledge graph reasoning. In *International Joint Conference on Artificial Intelligence*. International Joint Conference on Artificial Intelligence.

Qi Wang, Yongsheng Hao, and Jie Cao. 2020. Adrl: An attention-based deep reinforcement learning framework for knowledge graph reasoning. *Knowledge-Based Systems*, 197:105910.

Zachary Wojtowicz and Simon DeDeo. 2020. From probability to consilience: How explanatory values implement bayesian reasoning. *Trends in Cognitive Sciences*, 24(12):981–993.

Shuang Wu, Liwen Zhu, Tao Yang, Shiwei Xu, Qiang Fu, Yang Wei, and Haobo Fu. 2024. Enhance Reasoning for Large Language Models in the Game Werewolf. *arXiv preprint*. ArXiv:2402.02330 [cs].

Yuzhuang Xu, Shuo Wang, Peng Li, Fuwen Luo, Xiaolong Wang, Weidong Liu, and Yang Liu. 2024. Exploring Large Language Models for Communication Games: An Empirical Study on Werewolf. *arXiv preprint*. ArXiv:2309.04658 [cs].

Zelai Xu, Chao Yu, Fei Fang, Yu Wang, and Yi Wu. 2023. Language agents with reinforcement learning for strategic play in the werewolf game. *arXiv preprint arXiv:2310.18940*.

Shmuel Zamir. 2020. *Bayesian games: Games with incomplete information*. Springer.

Bin Zhang, Hangyu Mao, Jingqing Ruan, Ying Wen, Yang Li, Shao Zhang, Zhiwei Xu, Dapeng Li, Ziyue Li, Rui Zhao, Lijuan Li, and Guoliang Fan. 2024. Controlling Large Language Model-based Agents for Large-Scale Decision-Making: An Actor-Critic Approach. *arXiv preprint*. ArXiv:2311.13884 [cs].

Penghao Zhao, Hailin Zhang, Qinhan Yu, Zhengren Wang, Yunteng Geng, Fangcheng Fu, Ling Yang, Wentao Zhang, and Bin Cui. 2024. Retrieval-Augmented Generation for AI-Generated Content: A Survey. *arXiv preprint*. ArXiv:2402.19473 [cs].

A Pseudocode of GRATR

The pseudocode of GRATR's process are shown in Alg. 1, 2.

Algorithm 1 Graph Update Process

1: **Input**: Graph G^t, observations o^t.
2: Query the LLM to extract $\{\mathcal{R}^t(p_i, p_j), d^t(p_i, p_j), c^t(p_i, p_j)\}$ from the observation o^t;
3: **for** each intention $d^t(p_i, p_j)$ **do**
4: Update the evidence list $D^{t+1}(p_i, p_j)$ using Eq. (1);
5: **end for**
6: **for** each player p_j connected by an edge $e^t(p_i, p_j)$ **do**
7: Update the trustworthiness $T^{t+1}(p_j)$ using Eqs. (2), (3);
8: **end for**

B Complexity Analysis and Practical Deployment

The time complexity is $O(1)$ for real-time updates and $O(n \log n)$ for retrieval and reasoning, where n is the number of nodes, ensuring acceptable computational performance even with large datasets. Through testing, the comparison algorithms average 0.21s for updates, 0.61s for retrieval/reasoning, and 6.94 MB for storage. GRATR (8 nodes) averages 0.35s for updates, 1.33s for retrieval/reasoning, and 2.27 MB for storage, reflecting a 0.67x increase in update time, a 1.18x increase in reasoning time, and a 1/3 reduction in space. For larger n, time is mainly spent on LLM reasoning, while space is used for evidence storage. In future work, we will optimize graph structures and retrieval methods to reduce costs.

C Ablation Studies

We conduct ablation studies to demonstrate the effectiveness of different components in our algorithm. As shown in Table 1, we evaluate three variants of GRATR: GRATR-1 (without evidence merging), GRATR-2 (without multi-hop retrieval), and GRATR-3 (without backward update). The results reveal that all components contribute positively to the model's performance. The full GRATR model achieves the best performance with 92.2% accuracy and 91.4% macro F1-score. Removing multi-hop retrieval (GRATR-2) leads to the most significant performance drop (12-13 percentage points), indicating its crucial role in the model. The evidence merging mechanism (GRATR-1) also shows substantial impact, with an 8-9 percentage point decrease in performance when removed. While the backward update mechanism (GRATR-3) has a relatively smaller contribution, its removal still results in a 2-3 percentage point performance drop. These results validate the necessity of each component in our proposed architecture, with multi-hop retrieval being the most essential feature for the model's effectiveness.

	GRATR	GRATR-1	GRATR-2	GRATR-3
Accuracy	**0.922**	0.834	0.793	0.901
Macro F1	**0.914**	0.837	0.792	0.905

Table 5: Accuracy and Macro F1-score of GRATR, GRATR-1 (without evidence merging), GRATR-2 (without multi-hop retrieval), and GRATR-3 (without backward update).

D LLM Prompts

D.1 Extract Evidence

Your task is to analyze a given player's statement and determine its type based on the context of a role-playing deduction game. Based on the statement provided, determine which of the following types it belongs to:

- **Attack**: The player attempts to question or accuse another character, suggesting they might be suspicious, or provide evidence against another character.

- **Defend**: The player tries to defend a character, suggesting they are not suspicious. Note that character A and character B must be members of [Player 1, Player 2, ...], and might be the same, meaning the statement might be self-defense.

- **Deceive**: The player attempts to mislead other players with false information.

Additionally, provide a score indicating the strength or certainty of the statement's intent on a scale of 0 to 10, where 0 is very weak/uncertain and 10 is very strong/certain.

You must also determine the relationship between the players involved in the statement, categorizing it as one of the following:

- **Ally**: The player is supporting or aligning with another player.

- **Adversary**: The player is opposing or accusing another player.

Algorithm 2 Graph Retrieval Augmented Reasoning

Input: Number of the selected top trustworthiness nodes w, the target player p_o;
Initialization: Players $p_1, \dots, p_n$; Nodes N in G_i^t; Evidence chains list $C \leftarrow [\mathcal{C}_1, \mathcal{C}_2, \dots, \mathcal{C}_w]$ for player p_o (initially empty); Priority Queue $Q \leftarrow \emptyset$;
$N \leftarrow \text{Sort}(N, T^t(n))$;
$\{n_1, n_2, \dots, n_w\} \leftarrow \text{Top-}w(N)$;
$Q \leftarrow \{n_1, n_2, \dots, n_w\}$;
for $j = 1, 2, \dots, w$ **do**
  $\mathcal{C}_j \leftarrow \{n_1\}$;
end for
while $Q \neq \emptyset$ **do**
  $n_c, \mathcal{C}_c \leftarrow \mathbf{argmax}_{n \in Q}\, T^t(n)$;
  $Q \leftarrow Q \setminus \{n_c\}$;
  for each $n_k \in \text{Neighbors}(n_c)$ **do**
    Merge evidence $e^t(p_k, p_c)$ to update $T^{t+1}(p_k, p_c)$ based on the Eq. (4); // p_k, p_c are the players corresponding to the nodes n_k, n_c.
  end for
  $n_{k^*} \leftarrow \mathbf{argmax}_{n_k \in \text{Neighbors}(n_c)} T^t(p_k)$;
  $\mathcal{C}_c \leftarrow \mathcal{C}_c \cup \{n_{k^*}\}$;
  $Q \leftarrow Q \cup \{n_{k^*}\}$;
end while
Use C to update $T^{t+1}(p_o)$ based on the Eqs. (5), (6), (7), (8);
Update $T^{t+1}(p_o, p_{o-1})$ based on the Eq. (9);
Summarize and reason based on $T^t(p_o)$ and the evidence chains retrieved C;

- **Indifferent**: The player's statement does not clearly indicate support or opposition toward another player.

Carefully read the following statement and determine its type based on its content and tone: [Player's statement]

Please choose the appropriate type, relationship, and briefly explain your reasoning in the following format: [Role 1][Type][Role 2][Reason][Score][Relationship].

Please note that the statement might address multiple players simultaneously. In such cases, list each relevant result separately instead of in one line.

Here are some examples:

1. **Statement**: [(Player 1): I think Player 2's behavior was very strange. He kept avoiding important discussions. I believe Player 4 is innocent because he has been helping us.]

   Answer: [Player 1][Attack][Player 2][The Player 1 is questioning Player 2's behavior, implying they might be suspicious.][6][Adversary][Player 1][Defend][Player 4][The Player 1 is defending Player 4, suggesting they are not suspicious.][7][Ally]

2. **Statement**: [(Player 4): I observed that Player 3 was identified as suspicious by the moderator. I believe we should carefully consider the roles of the remaining players and gather more information before making any decisions.]

   Answer: [Player 4][Attack][Player 3][The current player indirectly accuses Player 3 of being suspicious by mentioning the moderator's identification, influencing others' perceptions.][9][Adversary]

3. **Statement**: [(Player 7): I believe Player 4 is innocent. He has been helping us analyze the situation.]

   Answer: [Player 7][Defend][Player 4][The Player 7 is defending Player 4, suggesting they are not suspicious.][7][Ally]

4. **Statement**: [(Player 1): I choose to eliminate Player 3.]

   Answer: [Player 1][Attack][Player 3][The Player 1 is strongly attacking Player 3.][10][Adversary]

5. **Statement**: [(Player 2): I choose to protect Player 3.]

   Answer: [Player 2][Defend][Player 3][The Player 2 is strongly protecting Player 3.][10][Ally]

(End of Example)

Now, given the statement: **Statement**: [statement]

D.2 Summary

Task: Summarize and reason about the trustworthiness of player (p_o) based on the updated trustworthiness score and the relevant evidence chains retrieved. The summary should include an analysis of the player's inferred identity, trustworthiness score, and the evidence supporting the inference.

Input:

- **Trustworthiness Score**: [Insert trustworthiness score of the player]

- **Evidence chains**: [Insert supporting evidence]

Output Format:

1. **Summary**: Provide a concise summary of the inferred identity of the player, the trustworthiness score, and the evidence supporting the inference.

2. **Reasoning**: Explain the reasoning behind the inferred identity and trustworthiness score, incorporating the retrieved evidence chains.

Example Output: [Player 1] is inferred to be a [identity], my [judge]. My level of trust in him is [confidence] [evidence].

E Werewolf Game

E.1 Introduction

The Werewolf game is a classic social deduction game typically played by 8 to 18 players. The game is divided into two main factions: the Good Faction and the Werewolf Faction. The goal of the Good Faction is to identify and eliminate all Werewolves, while the Werewolf Faction aims to hide their identities and eliminate all members of the Good Faction. Below is a detailed introduction to the game.

E.2 Game Roles

The game features various roles, each with unique abilities and objectives. Common roles include:

- **Villager (Ordinary Villager)**: No special abilities; they rely on deduction and voting to identify Werewolves.
- **Werewolf**: Can kill one player each night and disguises as a Villager during the day.
- **Seer**: Can check the identity of one player each night to determine if they are a Werewolf or Villager.
- **Witch**: Possesses a healing potion and a poison potion. The healing potion can revive a player killed by Werewolves, while the poison potion can kill a player.
- **Hunter**: When killed by Werewolves or voted out, the Hunter can shoot and eliminate another player.
- **Guard**: Can protect one player each night, preventing them from being killed by Werewolves.

E.3 Game Flow

The game alternates between Night and Day phases.

Night Phase

- Werewolf Action: The Werewolf team discusses and selects a player to kill.
- Seer Action: The Seer chooses a player to check their identity.
- Witch Action: The Witch can choose to use the healing potion to save a player killed by Werewolves or use the poison potion to kill a player.
- Guard Action: The Guard selects a player to protect from Werewolf attacks.
- Hunter Action: If the Hunter is killed by Werewolves or voted out, they can shoot and eliminate another player.

Day Phase

- Discussion: All players discuss the events of the previous night and deduce who the Werewolves are.
- Voting: Players vote to eliminate one player. The player with the most votes is eliminated.
- Reveal: The eliminated player reveals their role, and the game proceeds to the next night.

E.4 Victory Conditions

- Good Faction Victory: All Werewolves are eliminated.
- Werewolf Faction Victory: The number of Werewolves equals or exceeds the number of members in the Good Faction.

E.5 Game Strategies

- Deduction and Deception: Members of the Good Faction must use logic to identify Werewolves, while Werewolves must disguise themselves and mislead other players.
- Role Coordination: Roles like the Seer, Witch, and Guard must use their abilities strategically to help the Good Faction gain an advantage.
- Psychological Play: Players must use language and behavior to influence others' judgments, create confusion, or guide votes.