OPPOMEM: AN OPPONENT-AWARE MEMORY FRAMEWORK FOR SELF-EVOLVING LLM AGENTS IN STRATEGIC GAMES

Anonymous authors
Paper under double-blind review

ABSTRACT

Memory enables large language model (LLM) agents to evolve through experience without updating model parameters. However, strategic games pose three key challenges: uncertainty arising from opponent behavior, the limitations of naive similarity-based memory retrieval, and difficulty evaluating individual memories from final outcome feedback. Inspired by opponent modeling in game theory, We propose **OppoMem**: an **Oppo**nent-aware **Mem**ory framework for self-evolving LLM agents in strategic games. OppoMem represents strategic memories as associations between opponent types and response strategies. The agent uses online opponent modeling to guide the dynamic retrieval of relevant memories. To support memory evolution, OppoMem estimates individual memory utility through randomized activation and uses the resulting feedback to guide memory retention, revision, and removal. Extensive experiments show the effectiveness of our framework in enabling LLM agents to evolve in strategic games with an average relative performance improvement around 30%. Our code is available at `https://anonymous.4open.science/r/Core_code-91FF/`.

1 INTRODUCTION

Self-evolution is a key capability for large language model (LLM) -based agents in long-horizon decision-making tasks, where agent memory plays a crucial role in collecting experience and refining policies. Existing memory and feedback-based self-improvement methods have been studied in non-strategic environments, such as code generation (Madaan et al., 2023; Zhou et al., 2024) and math reasoning (Yang et al., 2024; Zhang et al., 2024a). Such environments usually have little uncertainty and the memory of past experiences can be directly reused across similar task. However, in game-theoretic settings, the ego agent cannot simply follow the past experiences because the opponents often act strategically. Therefore, a new memory framework is urged for LLM agents to evolve in strategic games.

We aim to design an opponent-aware and efficient memory framework that facilitates LLM agents evolving in strategic games, where three key challenges arise. The first challenge stems from the inherent uncertainty of strategic games, as future states and outcomes depend not only on the ego agent's actions but also on the behaviors of the opponents. Strategies that are effective at one decision point may become suboptimal at another under the uncertainty of opponents. The second challenge arises from memory retrieval. A game state usually consists of sequential actions of both players. A subtle change on a game state (e.g., one player switches her action) means a great difference. As a consequence, existing state similarity based retrieval can easily fail to retrieve the most suitable memory. The third challenge comes from evaluating an updating memories in long-horizon games without process rewards. Since memories can be invoked at multiple decision steps before getting the final game results, it is difficult to evaluate the effectiveness of each retrieved memories.

Existing memory frameworks are designed primarily for non-strategic environments. For example, ExpeL (Zhao et al., 2024) distills insights from past successes and failures and retrieves successful trajectories by task similarity, while A-MEM (Xu et al., 2025) organizes memories into interlinked notes that evolve as new experiences arrive. However, these memory frameworks can hardly help agents in strategic games due to their poor robustness to uncertainty. Another line of works focus on

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

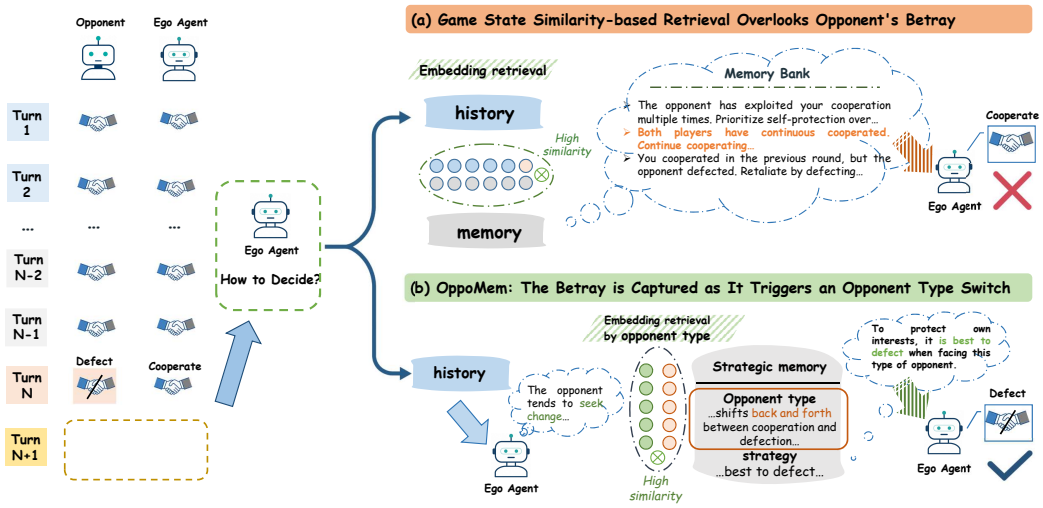


Figure 1: Comparison of conventional similarity-based memory retrieval and OppoMem in the repeated Prisoner's Dilemma. (a) Despite the opponent's recent defection, the interaction remains dominated by cooperation, so game-state similarity can retrieve a memory recommending continued cooperation. (b) OppoMem lets the agent infer the opponent's behavioral type from the interaction and uses it as the retrieval key to select an associated response strategy.

enhancing LLM agent's policy in strategic settings. For example, Strategist (Light et al., 2025) learns high-level strategic skills through self-play and reflection, Strat-Reasoner (He et al., 2026) develops a reinforcement learning approach to improve agent's policy in game-theoretic environments. These approaches focus on policy learning but fail to equip the agent with the ability to self-evolve by efficiently reuse experiences.

In this paper, we propose **OppoMem**: an **Oppo**nent-aware **Mem**ory framework for self-evolving LLM agents in strategic games. Inspired by opponent modeling in game theory, OppoMem constructs memories by explicitly modeling different types of opponents, and leverages such information to guide both memory retrieval, evaluation and update. Unlike non-strategic memory frameworks, OppoMem introduces an opponent-conditioned memory where the ego agent's experiences are represented by associations between opponent types and their corresponding responses. Building on this representation, the ego agent performs online opponent modeling from observable game history and dynamically retrieves memories conditioned on the inferred opponent type. As illustrated in Figure 1, this way of retrieval enables memory reuse to follow changes in opponent behavior rather than relying on similarity in game states, thus are more robust to uncertainty brought by opponents' behavior changes. Furthermore, for long-horizon games, the final game results show weak correlation with the effectiveness of retrieved memories at each step. To address this issue, we develop a memory evaluation mechanism that estimates the contribution of individual memories based on randomized activations. Together, these designs in OppoMem enable the ego agent to continually evolve during game playing. As OppoMem is a plug-and-play module, it naturally supports the agent self-evolution across multiple games. Our main contributions are as follows:

1. We propose **OppoMem**, an opponent-aware memory framework for self-evolving LLM agents in strategic games, and introduce strategic memory, which represents strategic experience as associations between opponent models and corresponding response strategies.

2. We develop an opponent-conditioned retrieval mechanism that dynamically selects memories based on online opponent modeling, together with a more effective memory evaluation mechanism that estimates the utility of individual memories in long-horizon decision-making scenarios.

3. Extensive experiments across diverse game settings demonstrate the effectiveness of OppoMem and establish opponent modeling and memory utility evaluation as key components for building effective self-evolving agents.

2 Related Work

2.1 Self-evolving LLM Agents

Self-evolving agents improve by turning interaction experience into reusable artifacts. Trace2skill distills trajectory-local lessons into reusable skills, ReasoningBank stores generalizable reasoning strategies as memory, and Evo-Harness compiles execution contexts into an evolving skill harness (Ni et al., 2026; Ouyang et al., 2026; Wei et al., 2026). In strategic games, Agent-Pro refines an agent's policy through reflection on past play (Zhang et al., 2024b). However, the inherent uncertainty of strategic games still presents challenges for applying existing self-evolving methods in these settings.

2.2 Opponent Modeling

Opponent modeling spans several generations of strategic decision-making. In game theory, fictitious play responds to an opponent's empirical action frequencies (Brown, 1951). In reinforcement learning, DRON learns opponent representations (He et al., 2016), while Deep BPR+ identifies changing opponent policies to reuse response policies (Zheng et al., 2018). With LLM agents, Hypothetical Minds tests natural-language hypotheses about other agents, and EMO maintains explicit models of individual opponents (Cross et al., 2025; Yu et al., 2025). Existing methods primarily use opponent modeling to guide LLM agents' decisions within the current game, while using it to accumulate and reuse experience across games remains a challenge.

2.3 LLM Agents in Games

Recent work has explored ways to improve the game-playing capabilities of LLM agents. Strategist combines language-level strategy refinement with tree search (Light et al., 2025), while SPIRAL develops reasoning through multi-turn zero-sum self-play (Liu et al., 2026). MARSHAL assigns turn-level learning signals in cooperative and competitive games, and Strat-Reasoner uses recursive opponent reasoning and intermediate reasoning rewards to optimize strategic behavior (Yuan et al., 2026; He et al., 2026). While existing work largely focuses on search and parameter-level optimization, how agents can continually evolve in multi-agent games remains an important challenge.

3 Preliminaries

3.1 Problem Formulation

We consider partially observable stochastic games (POSGs), which can be represented as a tuple:

$$\mathcal{G} = \langle \mathcal{I}, \mathcal{S}, b_0, \{\mathcal{A}_i\}_{i\in\mathcal{I}}, \{\mathcal{O}_i\}_{i\in\mathcal{I}}, P, \{r_i\}_{i\in\mathcal{I}}, H \rangle \,, \tag{1}$$

where $\mathcal{I} = \{1, \ldots, N\}$ is the set of players, $\mathcal{S}$ is the environment state space, $\mathcal{H}$ is the game history, and b_0 is the initial state. Each player i has an action space $\mathcal{A}_i$ and an observation space $\mathcal{O}_i$. Let $\mathbf{a}_t = (a_t^1, \ldots, a_t^N)$ and $\mathbf{o}_{t+1} = (o_{t+1}^1, \ldots, o_{t+1}^N)$ denote the joint action and joint observation, respectively. The joint transition-observation $P(s_{t+1}, \mathbf{o}_{t+1} \mid s_t, \mathbf{a}_t)$ specifies the probability of the next state and observations, while $r_i(s_t, \mathbf{a}_t)$ gives the reward received by player i. Because the underlying state and the private information of other players are not directly observable, player i selects actions from its local interaction history $h_t^i = (o_0^i, a_0^i, \ldots, o_{t-1}^i, a_{t-1}^i, o_t^i)$. A policy $\pi_i(a_t^i \mid h_t^i)$ therefore maps the observable history to a distribution over actions. The objective of player i is to maximize its expected episodic return.

$$J_i(\pi_i; \pi_{-i}) = \mathbb{E}\left[\sum_{t=0}^{H-1} r_i(s_t, \mathbf{a}_t)\right], \tag{2}$$

where π_{-i} denotes the policy of the other players.

3.2 Memory-Augmented LLM Agents

We represent the external memory of an LLM agent as a collection of memory items

$$\mathcal{M} = \{m_k\}_{k=1}^{K}, \tag{3}$$

where each m_k stores experience from previous tasks to guide subsequent question answering, mathematical problem solving, or code generation (Shinn et al., 2023; Zhao et al., 2024).

Given a task input x, the agent retrieves relevant memories $\mathcal{M}_x \subseteq \mathcal{M}$ and generates an output y:

$$\pi_{\theta,\mathcal{M}}(y \mid x) = p_\theta\left(y \mid \Psi(x, \mathcal{M}_x)\right). \tag{4}$$

Here, θ denotes the backbone LLM parameters, and Ψ constructs the prompt from the task and retrieved memories.

With θ fixed, memory updates aim to improve expected performance over a task distribution $\mathcal{D}$:

$$\max_{\mathcal{M}} \mathbb{E}_{x\sim\mathcal{D},\, y\sim\pi_{\theta,\mathcal{M}}(\cdot|x)}\left[R(x,y)\right]. \tag{5}$$

where $R(x, y)$ measures task performance. Task outcomes and feedback guide the addition or revision of memories for future use.

4 METHODS

Overview. We propose **OppoMem**, an opponent-aware memory framework that enables LLM agents to self-evolve in strategic games without updating model parameters. As illustrated in Figure 2, OppoMem comprises three components. First, a *strategic memory* bank is distilled from trajectories against a heterogeneous opponent pool, with each memory pairing an opponent-type description with a natural-language response strategy (Section 4.1). Second, online opponent modeling infers the opponent type at each decision step to retrieve an appropriate response strategy (Section 4.2). Third, memory utility is estimated through randomized activation during training: game returns with retrieved memories enabled or disabled are compared to guide memory retention, revision, or removal (Section 4.3). This iterative cycle of trajectory collection, memory distillation, retrieval, evaluation, and refinement supports continual agent evolution.

4.1 OPPONENT-TYPE-BASED MEMORY FRAMEWORK

OppoMem organizes strategic experience by associating opponent behavior patterns with suitable response strategies. Each strategic memory item is represented as a key–value pair $m_k = (\kappa_k, v_k)$, where k indexes the memories in the bank, κ_k describes an *opponent type*, and v_k contains a corresponding natural-language response strategy. An opponent type is a natural-language hypothesis about how the opponent tends to act or respond to the ego agent, inferred from observable interaction history. The response strategy guides the ego agent in selecting an action based on the current observation and interaction history.

For example, in Iterated Prisoner's Dilemma (IPD), a memory key may describe an opponent that repeatedly defects when the ego agent cooperates. The corresponding strategy recommends switching to defection to avoid further exploitation. By explicitly linking each strategy to an opponent behavior pattern, this representation captures both when the advice is applicable and how to respond, supporting memory retrieval based on the inferred behavior of the opponent.

We initialize the memory bank by collecting game trajectories against a heterogeneous opponent pool, including rule-based bots and LLM agents, as shown in the initialization module of Figure 2. At each decision step t, the ego agent uses its observable interaction history h_t to infer an opponent-type description $q_t^i = \mathrm{OM}(h_t, i)$ for each opponent i. Here, OM denotes an LLM call that summarizes the behavioral patterns and infers likely strategy of the specified opponent in natural language. After the game ends, we construct an interaction record (h_t, q_t^i, a_t, R) for each decision step and opponent, where a_t is the ego agent's response at step t and R is the final outcome of the game. We then cluster the records based on the similarity of their opponent-type descriptions q_t^i. Each cluster brings together experiences associated with a common opponent behavior pattern. A reflection LLM examines each cluster to summarize the shared opponent behavior across different decision steps and different opponents as a memory key κ_k and distill a corresponding response strategy v_k from the recorded decisions and outcomes. The resulting strategic memories form the initial memory bank and are subsequently assessed through utility evaluation. By accumulating experiences across different opponent types, the framework provides reusable guidance for adapting to changing strategic conditions.

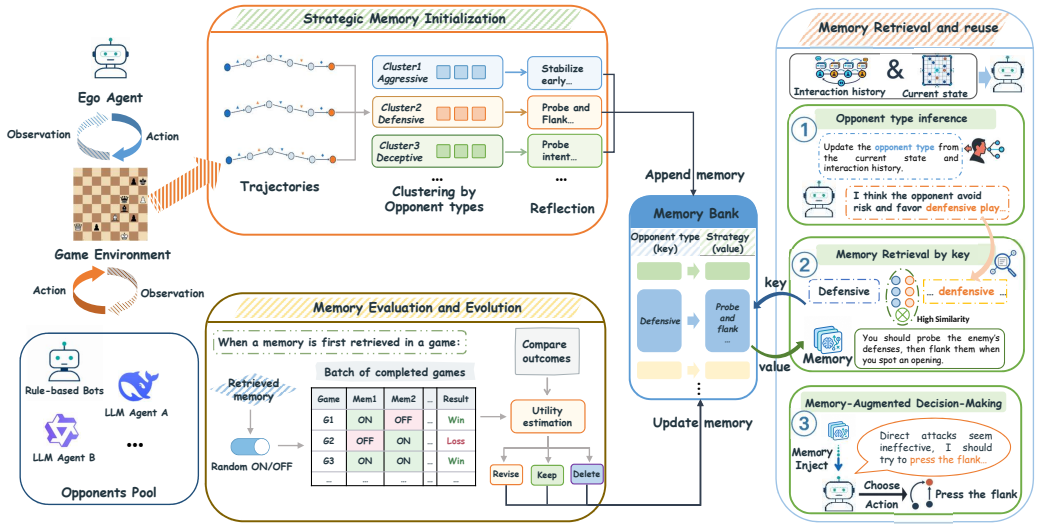


Figure 2: **Overview of OppoMem.** (1) *Strategic memory initialization*: trajectories collected against a heterogeneous opponent pool are decomposed into decision-level records, clustered by opponent type, and summarized by a reflection LLM into opponent-type–strategy pairs. (2) *Memory retrieval and reuse*: at each decision step, the agent updates its inferred opponent type from observable information, uses the inferred opponent type to retrieve a strategic memory, and combines its response strategy with the current context to choose an action. (3) *Memory evaluation and evolution*: when a memory is first retrieved in a game, it is randomly activated or deactivated, with that decision fixed for subsequent retrievals in the game. The framework estimates its utility by comparing returns and guides keeping, revising, or deleting the memory. The updated bank is reused in future games.

4.2 Memory Retrieval and Reuse

To reuse the strategic memory bank effectively, the agent needs to identify which response strategy is most suited to the behavior of its opponents. At each decision step, OppoMem uses the observable interaction history h_t to update the opponent-type description $q_t^i = \mathrm{OM}(h_t, i)$ for each opponent i. These descriptions follow the same natural-language representation as the stored memory keys. We use each q_t^i as a retrieval query and select the memory whose key is most similar to it:

$$k = \arg \max_{j \in \{1,\dots,K\}} \mathrm{sim}(q_t^i, \kappa_j), \tag{6}$$

where K is the number of memories in the bank and sim measures semantic similarity between opponent-type descriptions q_t^i and the key κ_k . For each decision step and opponent, we obtain the selected index k from the similarity search. We then use k to retrieve the k-th memory item $m_k = (\kappa_k, v_k)$ from the bank and extract its response strategy v_k. The retrieved strategies are incorporated into the decision context together with the corresponding inferred opponent-type descriptions. The ego agent then combines this context with the current observation and interaction history h_t to select its next action under the guidance of the retrieved memory.

Online opponent modeling and opponent-aware memory retrieval at every decision step allows dynamic memory injection within a game. Similar observable game states may therefore lead to different retrieved memories when the inferred opponent types differ, while different histories may lead to the same strategy when similar opponent types are inferred. OppoMem selects memories according to the opponent behaviors their strategies are intended to address.

4.3 Memory Evaluation and Evolution

With memory construction and retrieval mechanisms in place, we next address how to evaluate the utility of individual memories. To estimate their utility under long-horizon decision-making

and sparse rewards, we introduce *randomized activation* during training games. Memory retrieval proceeds as usual, but we randomly determine whether the retrieved memory's response strategy is injected into the decision context. This activation decision is made when the memory is first retrieved in a game and remains fixed for all subsequent retrievals of that memory within the same game. For example, if a memory is disabled at its first retrieval, its response strategy is withheld for the rest of the game, even if the memory is retrieved again at later decision steps.

Specifically, we keep the memory bank fixed within each batch of training games. When memory m_k is first retrieved in game g, we independently sample an activation indicator $z_{g,k} \sim \mathrm{Bernoulli}(p)$, with $p = 0.5$, and keep it fixed for the rest of the game. Retrieval follows the usual procedure; after m_k is selected, we inject its response strategy v_k if $z_{g,k} = 1$ and withhold it if $z_{g,k} = 0$. This rule applies whenever the same memory is retrieved during that game. Each completed game contributes a single return observation for every memory retrieved during that game, including both enabled and disabled memories, regardless of how often it was retrieved.

To estimate the utility of memory m_k, we compare game returns when its response strategy v_k is enabled or disabled under the same opponent configuration c. Within a training batch, let $\mathcal{G}^z_{k,c}$ denote the set of completed games under configuration c in which memory m_k was retrieved at least once and assigned activation indicator $z_{g,k} = z$. Let R_g denote the ego agent's episodic return in game g. The utility estimate for memory m_k under configuration c is

$$\widehat{u}_{k,c} = \frac{1}{|\mathcal{G}^1_{k,c}|} \sum_{g \in \mathcal{G}^1_{k,c}} R_g - \frac{1}{|\mathcal{G}^0_{k,c}|} \sum_{g \in \mathcal{G}^0_{k,c}} R_g. \tag{7}$$

This process include only games in which memory m_k was retrieved at least once and compute $\widehat{u}_{k,c}$ only when both activation groups are nonempty.

After each training batch, we update the memory bank using the configuration-specific utility estimates $\widehat{u}_{k,c}$ and the associated interaction records. Memories with positive estimated utility are kept, and those with very low estimated utility are removed. For memories with moderately low utility, the reflection LLM uses the associated interaction records to revise the opponent-type description or response strategy. Memories with insufficient evaluation evidence remain unchanged. The updated bank, including revised entries, is used directly in the next batch of training games. Because multiple memories are retrieved during a training batch, our method provides a practical way to estimate memory utility with limited additional evaluation overhead. Repeated trajectory collection, memory distillation, and utility-guided refinement allow the agent to continually expand and improve its strategic memory bank, supporting continual self-evolution.

5 Experiments

5.1 Experiment Setup

Game Settings. We selected four strategic games: Repeated Rock-Paper-Scissors (RPS), Iterated Prisoner's Dilemma (IPD), Hanabi and Deliberation. RPS, IPD, and Hanabi are played in two-player settings, while Deliberation involves six players. Together, they span three core interaction types: zero-sum competition, pure cooperation, and mixed-motivation. They also cover both perfect and imperfect information settings. We use DeepSeek-V4-Flash and GPT-5-mini as backbone models, and the reflection LLM is GPT-5.6-sol.

Baselines and Metrics. We compare with six representative baselines: ReAct (Yao et al., 2023) interleaves reasoning and acting without cross-episode memory; Reflexion (Shinn et al., 2023) stores feedback-based reflections; ExpeL (Zhao et al., 2024) distills insights and retrieves similar experiences; Agent-Pro (Zhang et al., 2024b) optimizes beliefs and policies; A-MEM (Xu et al., 2025) structures experience as notes; and Strategist (Light et al., 2025) combines LLM-based strategy refinement with MCTS. We also include a Vanilla control that receives only the common game prompt and current legal observation. We report normalized performance metrics for all four games: episode win rate for RPS, mean agent payoff per round for IPD, mean final team score for Hanabi, and mean focal-agent realized utility for Deliberation.

Training and Evaluation Protocol. Training trajectories are used to construct and revise the OppoMem memory bank, which is frozen during test evaluation. Game-specific training and test budgets are specified in Appendix B.1. The training opponent and partner pool includes Gemini 2.5 Flash and Qwen3.6-Flash (Appendix B.2). At test time, RPS, IPD, and Deliberation use GPT-5.5 opponents. Hanabi is tested using self-play, with both players instantiated from the same method and backbone model.

Further details on the game settings, baselines, evaluation metrics, and training and evaluation protocol, including opponent and partner configurations, are provided in the appendix.

5.2 Main Results

Table 1 shows that OppoMem outperforms Vanilla in all settings and achieves the highest score in most scenarios. In RPS, OppoMem exceeds the strongest baseline by around 9.3% and 7.5% for the two backbones. OppoMem also achieves the highest scores with both backbones in Hanabi and Deliberation, extending its gains from competitive play to cooperation and mixed-motive scenarios. These results suggest that organizing response strategies around opponent types and adapting memory retrieval to the current interaction can support effective experience reuse across different strategic games.

Table 1: **Main results.** Results are reported as mean ± standard deviation over 3 random seeds. Higher is better for all metrics. Bold marks the best result per game and backbone.

Backbone	Method	Game Settings			
		RPS	**IPD**	**Hanabi**	**Deliberation**
DeepSeek-V4-Flash	Vanilla	45.3 ± 4.8	50.3 ± 1.3	45.6 ± 3.2	60.2 ± 1.9
	ReAct	46.7 ± 5.2	49.4 ± 1.9	46.2 ± 2.4	59.5 ± 1.4
	Reflexion	39.3 ± 4.2	46.5 ± 1.8	47.7 ± 3.1	61.4 ± 1.7
	ExpeL	45.7 ± 2.6	53.2 ± 1.6	50.2 ± 1.8	59.6 ± 1.1
	A-MEM	50.3 ± 3.9	51.9 ± 1.2	49.6 ± 2.3	62.2 ± 0.9
	Agent-Pro	56.7 ± 6.3	53.1 ± 1.8	51.3 ± 3.5	62.4 ± 2.4
	Strategist	58.7 ± 5.4	$\mathbf{54.7} \pm 2.5$	44.6 ± 4.2	56.9 ± 1.7
	OppoMem	$\mathbf{64.3} \pm 3.7$	52.5 ± 1.4	$\mathbf{55.2} \pm 2.8$	$\mathbf{66.8} \pm 1.3$
GPT-5-mini	Vanilla	29.3 ± 5.6	36.7 ± 1.5	20.0 ± 2.8	35.4 ± 1.7
	ReAct	28.7 ± 4.2	37.9 ± 2.1	21.6 ± 3.6	37.2 ± 1.3
	Reflexion	23.7 ± 4.6	26.4 ± 1.8	23.3 ± 3.4	34.7 ± 2.0
	ExpeL	40.7 ± 4.1	43.6 ± 1.1	26.5 ± 2.0	36.4 ± 1.6
	A-MEM	35.3 ± 2.9	44.3 ± 1.5	24.0 ± 3.0	37.7 ± 1.4
	Agent-Pro	46.3 ± 5.3	42.5 ± 2.7	26.1 ± 4.4	37.9 ± 1.9
	Strategist	44.0 ± 6.4	43.8 ± 1.9	16.8 ± 3.3	28.5 ± 2.6
	OppoMem	$\mathbf{49.7} \pm 3.8$	$\mathbf{45.4} \pm 1.6$	$\mathbf{31.6} \pm 2.7$	$\mathbf{42.8} \pm 1.7$

The baseline results show that adding experience does not consistently improve strategic decision-making. Reflexion, ExpeL, and A-MEM yield gains in some settings, but these gains vary across games and backbones, with performance sometimes falling below Vanilla. For example, Reflexion improves Hanabi scores but reduces performance in RPS and IPD, while ExpeL and A-MEM improve performance in most settings yet still trail OppoMem overall. This pattern suggests that general task-level experience reuse may not adequately address the uncertainty introduced by other players. Advice derived from previous interactions can become unsuitable when the opponent behaves differently, even within the same game.

Agent-Pro and Strategist consistently outperform Vanilla in RPS and IPD with both backbones, showing the value of strategic policy refinement. However, their gains are less consistent in Hanabi and Deliberation, whose greater complexity and demands for coordination and negotiation may limit the effectiveness of general policy refinement. For example, Strategist falls below Vanilla in these two games, respectively, with GPT-5-mini. OppoMem outperforms both methods in Hanabi

and Deliberation with both backbones, suggesting that adapting experience reuse to other players' behavior can better support more complex strategic games.

5.3 Ablation Studies

Settings. We examine opponent conditioning, dynamic retrieval, memory reuse, and utility-guided memory evolution. Table 2 compares *OppoMem (full)* with four variants. *w/o opponent conditioning* replaces opponent-type-based memory construction and retrieval with experience distilled from complete training trajectories and retrieved by game-history similarity. *w/o dynamic retrieval* provides the entire memory bank at the beginning of each game, without stepwise memory selection. *w/o memory* retains online opponent modeling at each decision step but supplies no stored response strategies. *Vanilla* removes both external memory and explicit opponent modeling. All variants use DeepSeek-V4-Flash and the same game settings; the dynamic retrieval variants use the same limit on injected memory items. Figure 3 separately evaluates memory evolution with and without utility-based memory evaluation.

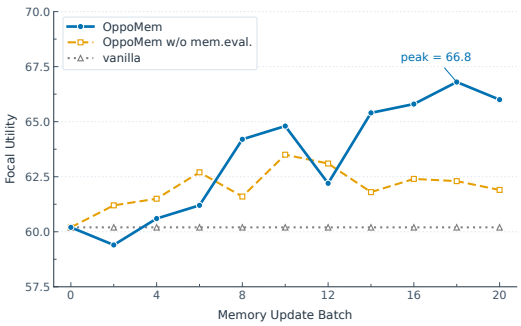


Figure 3: **Effect of memory evaluation in Deliberation.** Performance across memory update batches for OppoMem with and without memory evaluation. The gray dotted line shows Vanilla performance as a fixed reference.

Table 2: **Ablation study with DeepSeek-V4-Flash.** Mean $\pm$ standard deviation; metrics and the three-seed OppoMem/Vanilla results follow Table 1. OM represents opponent modeling. Higher is better; best means in bold.

Variant	RPS	IPD	Hanabi	Deliberation
OppoMem (full)	$\mathbf{64.3} \pm 3.7$	$\mathbf{52.5} \pm 1.4$	$\mathbf{55.2} \pm 2.8$	$\mathbf{66.8} \pm 1.3$
w/o opponent conditioning	55.3 ± 4.6	50.8 ± 1.7	46.8 ± 3.4	58.4 ± 1.5
w/o dynamic retrieval	43.0 ± 5.0	48.5 ± 1.6	46.8 ± 2.9	53.6 ± 2.1
w/o memory	48.7 ± 4.2	50.1 ± 1.2	48.9 ± 3.0	62.4 ± 1.8
Vanilla	45.3 ± 4.8	50.3 ± 1.3	45.6 ± 3.2	60.2 ± 1.9

Analysis. As shown in Table 2, OppoMem (full) achieves the highest score among the ablation variants in all four games. Replacing opponent conditioning with state-based memory construction and retrieval reduces performance across all four games, supporting the combined opponent-conditioned memory design. Without dynamic retrieval, full-context injection underperforms Vanilla in three games, with Deliberation utility decreasing from 60.2 to 53.6. This suggests that supplying the entire bank does not ensure effective memory use. Removing memory while retaining opponent modeling also lowers performance in all four games. Compared with removing both memory and OM, retaining OM improves RPS, Hanabi, and Deliberation, while IPD is slightly lower (50.1 versus 50.3). Together, these comparisons support the value of online opponent modeling and dynamically selected response memories.

Figure 3 shows how performance evolves over successive memory update batches in Deliberation, comparing OppoMem with a variant that only accumulates memories without utility-based evaluation. Focal utility denotes the average payoff obtained by the evaluated agent, with higher values indicating better performance. Although both methods initially improve, the utility of the variant without evaluation plateaus and ends near 62, whereas OppoMem's reaches a peak of 66.8. After the experiments, we analyzed the memory banks produced by both methods and found that without utility-based evaluation, redundant and low-utility entries can persist in the bank, potentially inter-

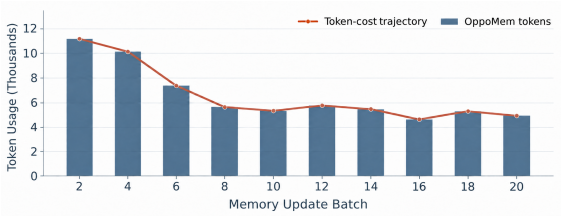


Figure 4: Token usage during memory refinement in Deliberation. Usage drops sharply at first, then fluctuates at a lower level.

Table 3: Cross-game transfer study with DeepSeek-V4-Flash. Cross-game memory refers to testing in the current environment using memory from another environment.

Method	**RPS**	**IPD**
Vanilla	45.3 ± 4.8	50.3 ± 1.3
Cross-game memory	54.0 ± 2.9	51.8 ± 1.1
OppoMem	64.3 ± 3.7	52.5 ± 1.4

fering with retrieval and reducing the relevance of the injected memories. These results support the effectiveness of our memory evaluation and evolution module.

5.4 Memory Reuse Across Games

We examine whether memory banks can transfer across games. We therefore remove move names, payoff claims, and game-specific responses from each source memory bank, retain descriptions of the opponent's conditional behavior, and formulate response strategies using the target game's rules and legal actions. Transfer uses only source-game memories and target-game rules.

Under cross-game transfer, the model scores 54.0 on RPS and 51.8 on IPD. The improvements over Vanilla in both directions suggest that the source memories retain useful information beyond game-specific actions. Although specific actions differ across games, some opponent behavior patterns and response principles may be transferable. By reusing these patterns and principles, OppoMem can provide useful guidance even without strategic experience acquired in the target game, showing its potential for cross-game generalization.

5.5 Computational Cost

Memory refinement progressively reduces the token budget required for action generation. Taking Deliberation as an illustrative example, Figure 4 shows an overall decline in action-generation token usage as the memory bank evolves, with a pronounced initial decrease followed by smaller fluctuations at later iterations.

Our inspection of the interaction trajectories suggests that refined memories provide increasingly actionable guidance conditioned on the inferred opponent type. This guidance helps the agent focus on relevant response strategies, reducing repeated analysis of the accumulated dialogue and unnecessary consideration of alternative actions. Rather than reconstructing a strategy at each decision step, the agent can adapt previously refined advice to the current interaction. The observed decline in token usage is therefore consistent with reduced redundant reasoning, suggesting that memory evolution can support faster decision-making by lowering the reasoning burden.

6 Conclusion

We presented OppoMem, an opponent-aware memory framework for self-evolving LLM agents in strategic games. OppoMem represents strategic memories as associations between opponent types and response strategies, dynamically retrieves relevant memories through online opponent modeling, and estimates their utility through randomized activation to guide memory evolution. Together, these mechanisms enable agents to accumulate, refine, and reuse strategic experience without updating model parameters.

Experiments demonstrate consistent improvements across competitive, cooperative, and mixed-motive settings. These findings highlight the value of organizing and refining experience around other players' behavior for self-evolving agents. Future work could improve utility estimation for rarely retrieved memories, reduce the computational overhead of opponent modeling, and extend transfer evaluation to a broader range of games.

AI USE STATEMENT

In this work, we used generative AI tools to provide feedback on research methodology and experiments, assist with the implementation of author-designed methods, and assist with translation. We did not use generative AI tools to propose or refine hypotheses or to interpret results. Generating synthetic datasets, helping develop theoretical models or conceptual frameworks, formulating mathematical claims, providing critical ingredients for proving mathematical claims, assisting in the writing of proofs, cleaning or reformatting datasets, and supporting qualitative or thematic data analysis, were not applicable to this work. Additionally, we used generative AI tools to create or edit software code and to edit the manuscript to improve readability.

We have reviewed all AI-assisted work. Suggestions regarding research methodology and experiments were critically evaluated by the authors, and all final methodological and experimental decisions were made by the authors. AI-assisted code used for method implementation was reviewed and tested by the authors. AI-assisted translations and edits to the manuscript were reviewed by the authors for accuracy and consistency. The authors take responsibility for the final content of this work, including text, claims, code, and other artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

The proposed method is reproducible from our anonymous source-code release: `https://anonymous.4open.science/r/Core_code-91FF/`. The repository provides the core OppoMem implementation and game adapters for all four experimental settings. The method description in Section 4 and the implementation details and prompt templates in Appendices B and C provide the information needed to reproduce our approach.

REFERENCES

Sahar Abdelnabi, Amr Gomaa, Sarath Sivaprasad, Lea Schönherr, and Mario Fritz. Cooperation, competition, and maliciousness: LLM-stakeholders interactive negotiation. In *Advances in Neural Information Processing Systems, Datasets and Benchmarks Track*, 2024. URL https://papers.neurips.cc/paper_files/paper/2024/file/984dd3db213db2d1454a163b65b84d08-Paper-Datasets_and_Benchmarks_Track.pdf.

Nolan Bard, Jakob N. Foerster, Sarath Chandar, Neil Burch, Marc Lanctot, H. Francis Song, Emilio Parisotto, Vincent Dumoulin, Subhodeep Moitra, Edward Hughes, Iain Dunning, Shibl Mourad, Hugo Larochelle, Marc G. Bellemare, and Michael Bowling. The Hanabi challenge: A new frontier for AI research. *Artificial Intelligence*, 280:103216, 2020. doi: 10.1016/j.artint.2019.103216. URL https://arxiv.org/pdf/1902.00506.

George W. Brown. Iterative solution of games by fictitious play. In Tjalling C. Koopmans (ed.), *Activity Analysis of Production and Allocation*, pp. 374–376. Wiley, New York, 1951.

Logan Cross, Violet Xiang, Agam Bhatia, Daniel Yamins, and Nick Haber. Hypothetical Minds: Scaffolding theory of mind for multi-agent tasks with large language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/12f483f624b378f9f3058d8ecd3c7ff5-Paper-Conference.pdf.

He He, Jordan Boyd-Graber, Kevin Kwok, and Hal Daumé III. Opponent modeling in deep reinforcement learning. In *Proceedings of the 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pp. 1804–1813. PMLR, 2016. URL https://proceedings.mlr.press/v48/he16.pdf.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. DeBERTaV3: Improving DeBERTa using ELECTRA-style pre-training with gradient-disentangled embedding sharing. *arXiv preprint arXiv:2111.09543*, 2021. URL https://arxiv.org/pdf/2111.09543.

Yidong He, Yutao Lai, Pengxu Yang, Jiarui Gan, Jiexin Wang, Yi Cai, and Mengchen Zhao. Strat-Reasoner: Reinforcing strategic reasoning of LLMs in multi-agent games. *arXiv preprint arXiv:2605.04906*, 2026. URL https://arxiv.org/pdf/2605.04906.

Vincent Knight, Owen Campbell, Marc Harper, Karol Langner, James Campbell, Thomas Campbell, Alex Carney, Martin Chorley, Cameron Davidson-Pilon, Kristian Glass, Nikoleta Glynatsi, Tomáš Ehrlich, Martin Jones, Georgios Koutsovoulos, Holly Tibble, Jochen Müller, Geraint Palmer, Piotr Petunov, Paul Slavin, Timothy Standen, Luis Visintini, and Karl Molden. An open framework for the reproducible study of the iterated prisoner's dilemma. *Journal of Open Research Software*, 4 (1):e35, 2016. doi: 10.5334/jors.125. URL https://arxiv.org/pdf/1604.00896.

Marc Lanctot, John Schultz, Neil Burch, Max Olan Smith, Daniel Hennes, Thomas Anthony, and Julien Perolat. Population-based evaluation in repeated rock-paper-scissors as a benchmark for multiagent reinforcement learning. *arXiv preprint arXiv:2303.03196*, 2023. URL https://arxiv.org/pdf/2303.03196.

Jonathan Light, Min Cai, Weiqin Chen, Guanzhi Wang, Xiusi Chen, Wei Cheng, Yisong Yue, and Ziniu Hu. Strategist: Self-improvement of LLM decision making via bi-level tree search. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://arxiv.org/pdf/2408.10635.

Bo Liu, Leon Guertler, Simon Yu, Zichen Liu, Penghui Qi, Daniel Balcells, Mickel Liu, Cheston Tan, Weiyan Shi, Min Lin, Wee Sun Lee, and Natasha Jaques. SPIRAL: Self-play on zero-sum games incentivizes reasoning via multi-agent multi-turn reinforcement learning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://arxiv.org/pdf/2506.24119.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-Refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems*, volume 36, pp. 46534–46594. Curran Associates, Inc., 2023. doi: 10.52202/075280-2019. URL https://arxiv.org/pdf/2303.17651.

Jingwei Ni, Yihao Liu, Xinpeng Liu, Yutao Sun, Mengyu Zhou, Pengyu Cheng, Dexin Wang, Erchao Zhao, Xiaoxi Jiang, and Guanjun Jiang. Trace2Skill: Distill trajectory-local lessons into transferable agent skills. *arXiv preprint arXiv:2603.25158*, 2026. doi: 10.48550/arXiv.2603.25158. URL https://arxiv.org/pdf/2603.25158.

Siru Ouyang, Jun Yan, I-Hung Hsu, Yanfei Chen, Ke Jiang, Zifeng Wang, Rujun Han, Long Le, Samira Daruki, Xiangru Tang, Vishy Tirumalashetty, George Lee, Mahsan Rofouei, Hangfei Lin, Jiawei Han, Chen-Yu Lee, and Tomas Pfister. ReasoningBank: Scaling agent self-evolving with reasoning memory. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://arxiv.org/pdf/2509.25140.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 36, pp. 8634–8652. Curran Associates, Inc., 2023. doi: 10.52202/075280-0377. URL https://papers.nips.cc/paper_files/paper/2023/file/1b44b878bb782e6954cd888628510e90-Paper-Conference.pdf.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*, 2022. URL https://arxiv.org/pdf/2212.03533.

Tianxin Wei, Zhan Shi, Minhua Lin, Bing He, Zewen Liu, Yisi Sang, Yuanchen Bei, Xuying Ning, Jiaru Zou, Ting-Wei Li, Xiao Lin, Yanjun Zhao, Chi Wang, Benoit Dumoulin, Dakuo Wang, Jingrui He, and Hanqing Lu. Evo-Harness: Context-to-harness skill compilation for self-evolving agents. *arXiv preprint arXiv:2608.15071*, 2026. URL https://arxiv.org/pdf/2608.15071.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. A-MEM: Agentic memory for LLM agents. In *Advances in Neural Information Processing Systems*, 2025. URL https://arxiv.org/pdf/2502.12110.

Ling Yang, Zhaochen Yu, Tianjun Zhang, Shiyi Cao, Minkai Xu, Wentao Zhang, Joseph E. Gonzalez, and Bin Cui. Buffer of Thoughts: Thought-augmented reasoning with large language models. In *Advances in Neural Information Processing Systems*, volume 37, pp. 113519–113544. Curran Associates, Inc., 2024. doi: 10.52202/079017-3607. URL https://arxiv.org/pdf/2406.04271.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://arxiv.org/pdf/2210.03629.

XiaoPeng Yu, Wanpeng Zhang, and Zongqing Lu. LLM-based explicit models of opponents for multi-agent games. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 892–911. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.naacl-long.41. URL https://aclanthology.org/2025.naacl-long.41.pdf.

Huining Yuan, Zelai Xu, Zheyue Tan, Xiangmin Yi, Mo Guang, Kaiwen Long, Haojia Hui, Boxun Li, Xinlei Chen, Bo Zhao, Xiao-Ping Zhang, Chao Yu, and Yu Wang. MARSHAL: Incentivizing multi-agent reasoning via self-play with strategic LLMs. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://arxiv.org/pdf/2510.15414.

Tianjun Zhang, Aman Madaan, Luyu Gao, Steven Zheng, Swaroop Mishra, Yiming Yang, Niket Tandon, and Uri Alon. In-context principle learning from mistakes. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 59520–59558. PMLR, 2024a. URL https://arxiv.org/pdf/2402.05403.

Wenqi Zhang, Ke Tang, Hai Wu, Mengna Wang, Yongliang Shen, Guiyang Hou, Zeqi Tan, Peng Li, Yueting Zhuang, and Weiming Lu. Agent-Pro: Learning to evolve via policy-level reflection and optimization. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5348–5375. Association for Computational Linguistics, 2024b. doi: 10.18653/v1/2024.acl-long.292. URL https://aclanthology.org/2024.acl-long.292.pdf.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. ExpeL: LLM agents are experiential learners. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(17):19632–19642, 2024. doi: 10.1609/aaai.v38i17.29936. URL https://arxiv.org/pdf/2308.10144.

Yan Zheng, Zhaopeng Meng, Jianye Hao, Zongzhang Zhang, Tianpei Yang, and Changjie Fan. A deep Bayesian policy reuse approach against non-stationary agents. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://papers.nips.cc/paper_files/paper/2018/file/85422afb467e9456013a2a51d4dff702-Paper.pdf.

Andy Zhou, Kai Yan, Michal Shlapentokh-Rothman, Haohan Wang, and Yu-Xiong Wang. Language agent tree search unifies reasoning, acting, and planning in language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 62138–62160. PMLR, 2024. URL https://arxiv.org/pdf/2310.04406.

A Experimental Details

We use one complete game as an evaluation episode. Evaluation-set sizes vary with the game, as specified in Appendix B.1. All scores are displayed on a 0–100 scale.

A.1 Game Rules and Outcome Definitions

Repeated Rock-Paper-Scissors (RPS). We use the repeated-game setting and Roshambo opponent family described by Lanctot et al. (2023). In each game, the two players simultaneously choose rock, paper, or scissors for 20 rounds. A round gives the focal player $+1$, 0, or -1 for a win, draw, or loss. An episode is a win when the sum of these payoffs is positive. We report the percentage of won episodes.

Iterated Prisoner's Dilemma (IPD). We use the standard IPD payoff matrix and Axelrod strategy family described by Knight et al. (2016). The players simultaneously choose cooperation or defection in each of 25 rounds. Their payoffs are $(3, 3)$ for mutual cooperation, $(0, 5)$ when the focal player cooperates and the opponent defects, $(5, 0)$ for the reverse, and $(1, 1)$ for mutual defection. We report the focal player's mean payoff per round, divided by the maximum round payoff of 5 and multiplied by 100.

Hanabi. We use the Hanabi Learning Environment of Bard et al. (2020). The two players cooperate to build five fireworks by playing ranks one through five in order. Each starts with five cards, sees the partner's hand but not their own, and takes turns playing a card, discarding, or giving a legal hint. A hint spends an information token, a discard can recover one, and an incorrect play consumes a life token. Games end under the environment's deck and token rules rather than after a fixed number of turns. We report the mean final team score, divided by the maximum of 25 and multiplied by 100.

Deliberation. We adapt the released base negotiation scenario of Abdelnabi et al. (2024). Six participants negotiate a package with one option for each of five issues. Each role has private option values and an acceptance threshold. After an opening and a bounded discussion, SportCo proposes the final package. It passes if at least five participants accept, including SportCo and the Department of Tourism. The focal role receives its realized utility if the package passes and zero otherwise; a unanimous agreement adds a ten-point bonus to SportCo's utility. We report $100\bar{u}$, where $u = R/100$ and R is the focal role's realized utility. The divisor 100 is a fixed reporting scale, not the maximum feasible utility in every scenario.

Private score instantiation. For each role, `S_A1` is the first value of its A vector, and similarly for the other issue vectors. `MAX_A` is the maximum of the role's A vector. `MIN_SCORE` is the last column. The score of a complete deal is the sum of its five selected option values.

Table 4: Private base-utility parameters from the released base negotiation scenario of Abdelnabi et al. (2024). Vector entries follow the option order A1–A3, B1–B3, C1–C4, D1–D4, and E1–E5. Each player receives only its own row.

Role	A	B	C	D	E	Minimum
SportCo	14, 8, 0	11, 7, 0	0, 5, 10, 17	35, 29, 20, 0	0, 5, 10, 15, 23	55
Department of Tourism	0, 11, 5	0, 20, 25	0, 2, 4, 9	10, 26, 40, 0	4, 8, 15, 12, 0	65
Environmental League	0, 22, 45	0, 25, 55	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0, 0	55
Local Labour Union	15, 20, 0	0, 0, 0	42, 35, 25, 0	30, 20, 10, 0	2, 4, 6, 8, 0	50
Other cities	0, 4, 10	0, 0, 0	12, 8, 6, 0	0, 8, 13, 18	60, 45, 30, 15, 0	31
Mayor	14, 8, 0	12, 8, 0	24, 18, 12, 0	40, 30, 23, 0	0, 2, 4, 7, 10	30

A.2 Ablation Comparisons

All variants use the same action backbone, game rules, legal observations, and evaluation episodes. Vanilla uses no opponent model or memory. OM-only adds online opponent descriptions. State retrieval constructs a separate memory bank by reflecting on complete training trajectories. Each entry associates a summary of the observable game state and public history with reusable response advice. At evaluation time, it retrieves advice by semantic similarity between the current state/history summary and these state-based keys, using the same limit on injected memory items as OppoMem. Full-context injection supplies OppoMem's response strategies throughout the game without stepwise selection. OppoMem retrieves from its current opponent description at each decision. The state and opponent descriptions contain no other player's private information.

A.3 Bidirectional RPS–IPD Transfer

We construct each source bank without target-game trajectories, then translate behavior descriptions and response advice into the target game's legal actions. Descriptions retain general patterns such as reciprocity and switching, while source-game move names and payoff claims are removed. Entries without a defensible target-game response are discarded. Vanilla, OM-only, and the transferred-memory condition use the same held-out opponent schedule in each target game. No target-game test feedback is used in the conversion.

A.4 Computational Cost

Memory construction uses model calls for game play, online opponent modeling, trajectory analysis, and strategy reflection or revision. Evaluation uses action and opponent-model calls, while E5 retrieval and DeBERTa reranking add local computation. Retrieved advice also increases the action prompt length. Consequently, token counts and latency depend on the game horizon, retrieved text, and model response length. The aggregate evaluation comparison is reported in Section 5.5; it should be read as descriptive of those runs.

B Implementation Details

B.1 Training and Test Protocol

We allocate complete episodes according to interaction length and cost. For each action backbone, RPS uses 20 batches of 100 training games and 200 test games; each game has 20 simultaneous rounds. IPD uses 20 batches of 100 training games and 100 test games; each game has 25 simultaneous rounds. Hanabi uses 20 batches of 50 complete training games and 100 test games; its number of turns is determined by the deck and terminal rules. Deliberation uses 20 batches of 50 complete training negotiations and 60 test negotiations; the published opening, discussion, final-proposal, and voting phases define one episode. These are episode counts, not numbers of individual actions or messages.

Training trajectories initialize the memory bank and support the batchwise utility estimates and revisions in Section 4.3. During training, RPS and IPD rotate the fixed-bot identities and the two LLM opponents listed in Table 5 across batches; Hanabi rotates its fixed policy and two LLM partners. For Deliberation, the released role profiles (Abdelnabi et al., 2024) and the two training LLM models are rotated across the five non-focal seats. Allocation is kept as even as the episode count permits. A low-support memory is left unchanged rather than revised from a single outcome. Validation episodes use disjoint seeds and select one checkpoint using the game's primary normalized outcome. Test seeds are disjoint from both splits and are fixed before selection. Within each game and split, methods share the same participant-profile schedule, focal role or seat, and environment seed whenever pairing is possible. Memory banks and baseline artifacts are fixed during test; online opponent modeling still uses observations from the current game. The split is by complete episode, so no segment from a validation or test game enters memory construction.

B.2 Participants and Retrieval

Table 5 gives the participant families used during training. Gemini 2.5 Flash and Qwen3.6-Flash provide the training LLM opponents or partners. In Hanabi, the fixed training partner follows the SimpleAgent policy distributed with the Hanabi Learning Environment (Bard et al., 2020) through a player-visible-action adapter. In Deliberation, the five non-focal roles use the released game's role instructions and cooperative or greedy participant profiles (Abdelnabi et al., 2024); no separate threshold or concession bot is introduced. Held-out test evaluation uses GPT-5.5 opponents in RPS and IPD and GPT-5.5 agents in all five non-focal Deliberation roles. Hanabi test evaluation uses self-play with both players instantiated from the evaluated method and action backbone. These evaluation configurations are distinct from the training pool in Table 5. GPT-5.6-sol assists only offline memory reflection and revision; it never plays a game role or supplies the focal action or online opponent model.

Table 5: Participant families used during training. Held-out test evaluation uses GPT-5.5 opponents in RPS, IPD, and Deliberation, and same-method self-play in Hanabi.

Game	Fixed policies or partners	LLM opponents or partners
RPS	Roshambo randbot, rockbot, textbot, rotatebot, copybot, and freqbot2	Gemini 2.5 Flash; Qwen3.6-Flash
IPD	Axelrod Tit for Tat, Grudger, Random, First by Joss, First by Davis, First by Stein and Rapoport, First by Shubik, and First by Grofman	Gemini 2.5 Flash; Qwen3.6-Flash
Hanabi	Hanabi Learning Environment SimpleAgent policy	Gemini 2.5 Flash; Qwen3.6-Flash
Deliberation	Released role prompts with cooperative and greedy profiles	Gemini 2.5 Flash; Qwen3.6-Flash

The main experiments in all four games use E5-base-v2 embeddings (Wang et al., 2022), medoid representatives, and the `cross-encoder/nli-deberta-v3-xsmall` NLI reranker (He et al., 2021) for memory retrieval. The agent injects at most one retrieved response for each modeled participant at a decision step. The prompts expose only the acting player's legal observations and private information; opponent labels and other players' private data are unavailable to the action model.

Hanabi partner modeling. Hanabi is cooperative, so the modeled player is the partner. The description summarizes behavior supported by visible hints, plays, and discards, such as information-sharing and risk-taking tendencies. A retrieved response guides the focal player's own legal play, discard, or hint using the current player-visible observation; it does not treat the focal player's hidden cards as known.

Deliberation participant modeling. The focal role maintains a separate description of each of the other five participants, updating it from that participant's public messages as the negotiation proceeds. At most one response per participant can enter a decision prompt. The focal role combines that advice with the public discussion and its own private role information to produce a phase-valid utterance or proposal; the other roles' private values and acceptance thresholds remain unavailable.

B.3 Memory Construction, Retrieval, and Evolution

Training trajectories are segmented into records containing the observable history, the then-current opponent description, the focal action, and the eventual outcome. Similar behavioral descriptions are grouped, a medoid provides each group's representative description, and reflection produces a corresponding response strategy. At each decision step, the agent updates its opponent descriptions and retrieves advice using the shared E5–medoid–DeBERTa pipeline.

During a training batch, each eligible memory is randomly enabled or withheld the first time it is retrieved in a game; that choice applies for the rest of that game. Completed returns for enabled and withheld games under the same participant configuration estimate the memory's utility when both groups are observed. The controller may use these per-configuration comparisons and the associated trajectories to keep, revise, or remove an entry; an entry without a comparable on/off group is left unchanged. Estimates remain attached to the memory revision that produced them: earlier batch summaries for an unchanged revision can be consulted, while a revised entry is assessed using newly collected returns rather than inheriting its predecessor's estimates. This follows the training loop and estimator in Section 4.3.

B.4 Baseline Adaptations

Every method is wrapped by the same game-specific adapter: it receives only the focal player's legal observation and public history, must return a legal action or phase-valid utterance, and is scored by the same terminal evaluator within each game. Training episodes, held-out schedules, backbone

model, and maximum model context are matched within a game. A method may use its published form of intermediate reasoning or cross-episode state, but cannot inspect other players' private cards, score tables, evaluator votes, or test outcomes when building that state. Baseline artifacts are frozen before held-out evaluation. The adaptations below preserve each paper's central mechanism, rather than claim to reproduce its original benchmark software. Table 1 reports Agent-Pro and Strategist in all four games.

Vanilla and ReAct (Yao et al., 2023). Vanilla receives the common game System and current legal User observation, without opponent-model or memory fields. ReAct uses the same observation and legal-output parser, with an internal thought/action cycle for the current decision. Its thought is not exposed to other players and its within-game transcript is reset between episodes. The episode and action-call budgets include every reasoning step, so extra deliberation is not free.

Reflexion (Shinn et al., 2023). After a completed training episode, a reflection call receives only that episode's public trajectory, focal-private information, and permitted terminal feedback. It writes one bounded verbal lesson for subsequent training games. The accumulated reflections are included in the action context in insertion order up to the common context cap. Evaluation freezes this list; a test failure does not generate a new reflection.

ExpeL (Zhao et al., 2024). Successful and failed training trajectories are summarized into reusable insights. The insight list is updated through additions, edits, and up- or down-votes, while successful trajectories remain available as demonstrations. At a decision, the common adapter serializes the current public state and history as the retrieval task; it supplies the retained insight list and the top-k similar successful training trajectories, subject to the context cap. The game's legal output and terminal score supply the attempt and success units. Extraction calls, demonstration count, and injected tokens are charged to its budget. No held-out trajectory enters insight extraction or retrieval storage.

A-MEM (Xu et al., 2025). Training trajectories become notes with content, contextual tags, and semantic links to related notes. Note creation and link updates occur during training only. At a decision step, the public observation retrieves linked notes by semantic relevance; the resulting text is passed to the same game-specific output interface. Note count, retrieval limit, and injected-token cap are bounded, but links are not replaced by OppoMem's opponent-conditioned key.

Agent-Pro (Zhang et al., 2024b). An episode maintains beliefs about other participants' behavior using their observable actions or public messages. Completed training episodes support trajectory- and belief-level reflection followed by search over candidate text policies. Candidate verification uses training or development games through the common game adapter; the selected policy is frozen for evaluation. Beliefs may change within a game, while the action interface exposes the same legal observations and outputs as Vanilla. Search expansions, model calls, and validation games belong to its reported budget.

Strategist (Light et al., 2025). Training learns and refines a population of high-level text strategies from self-play and game feedback. At decision time, lower-level Monte Carlo tree search uses those strategies to select an output through the game-specific adapter. The adapter constrains the output to a legal action in RPS, IPD, and Hanabi, or a phase-valid utterance or proposal in Deliberation. Simulated games, search expansions, and model calls count toward its budget. The same held-out evaluation protocol and terminal metric used for the other methods are applied in each of the four games.

C Prompt Templates

The templates below specify the action, opponent-modeling, and memory-update interfaces. A *System* message sets game or module instructions; a *User* message supplies the current observation or training record. Placeholders are filled with information legally available to the acting player.

C.1 Prompt Inventory and Composition

The inventory identifies the System and User components assembled for each call. Optional advice is omitted when no memory is retrieved.

Table 6: Complete call inventory. IDs refer to the prompt boxes below. An assembly is concatenated in the specified order.

Game / module	Stage	System and User	Output
RPS	Action	System R1; User R2 with R4	Rock, Paper, or Scissors
RPS	Opponent-Modeling	System: rules in R1 without output line, then C1; User R3	Plain opponent description
IPD	Action	System P1; User P2 with P4	Cooperate or Defect
IPD	Opponent-Modeling	System: rules in P1 without output line, then C1; User P3	Plain opponent description
Hanabi	Action	System H1; User H2	Legal action ID and response intent
Hanabi	Opponent-Modeling	System H3 then C1; User H4	Opponent-modeling JSON
Deliberation	Action	System D-SYS; phase User D5–D7 with D9	Public answer, proposal, and permitted private fields
Deliberation	Opponent-Modeling	System D8 then C1; User D8	Plain participant description
Memory	Segmentation	System M1; User M2	Semantic windows
Memory	Construction	System M3; User M4	Response-strategy JSON
Memory	Revision	System M3; User M5	Revised response-strategy JSON

Table 7: Module switches under the unified interface. All arms retain their game's legal observations and response format.

Component	Vanilla	OM-only	OppoMem
Game System and base User	Yes	Yes	Yes
Opponent-modeling call	No	Yes	Yes
Retrieved response advice	No	No	When eligible and enabled
Deliberation focal private-plan field	Yes	Yes	Yes

Placeholder convention. Uppercase angle-bracket fields are replaced before a model call; they are not literal instructions to the model. Fixed game tags such as `<ANSWER>` are exceptions: these tags delimit the model's output and remain literal. An explicitly optional block is omitted entirely when absent. JSON placeholders stand for correctly serialized values or arrays, not quoted Python objects. No trajectory, private opponent state, or evaluator-only label is inserted into an online prompt.

Table 8: Dynamic input dictionary. Game-specific fields and offline evidence fields are refined in their corresponding subsections.

Field	Exact meaning
`ROUND_NUMBER`, `HORIZON`	One-based decision round and match length.
`COMPLETED_ROUNDS`	Chronological rows formatted as *Round n: your action = a; opponent action = b; your payoff = r.* Use *(No completed rounds.)* when empty.
`PREVIOUS_DESCRIPTION_BLOCK`	Omitted if unavailable; otherwise the literal heading *Previous opponent description (may need revision):* followed by the preceding Opponent-Modeling text.

Field	Exact meaning
ADVISORY_BLOCK	One of the complete arm-specific blocks defined below, or the empty string. A withheld memory contributes no memory text.
OPPONENT_MODELING_TEXT, MEMORY_MODEL, MEMORY_RESPONSE	Current inferred description, stored canonical description, and stored response advice. These are fallible text, not extra observations.
VISIBLE_STATE, PUBLIC_HISTORY, LEGAL_ACTIONS	Current player-visible observation, completed public events, and the exact legal actions supplied by the environment.
ROLE_NAME, ROLE_INSTRUCTIONS	Current Deliberation participant and that participant's own private role block only.
S_A1 through S_E5, MIN_SCORE	Role-specific option scores and acceptance threshold. Only valid options for each issue are instantiated.
LAST_PUBLIC_MESSAGES, PREVIOUS_PLAN_BLOCK	Recent public messages and optional private plan authored by the same focal participant; never another participant's private plan.
EPISODE_EVIDENCE, CONFIGURATION_SUMMARIES	Completed training evidence and per-configuration on/off summaries; these fields occur only in offline memory prompts.

C0 — Shared JSON formatting System

Prompt template

```
Return exactly one valid JSON object matching the requested
  fields. Do not include markdown fences, prose outside the
  object, NaN, or comments.
```

C0 is used only where a JSON response is required. It is not appended to the plain-action RPS/IPD calls or the tagged Deliberation dialogue calls.

C1 — Action-abstract opponent description rule

Prompt template

```
The final opponent_modeling field is a compact description of a
  behavioral pattern, not a response plan. Use exact
  player-visible actions only as evidence. Do not include any
  literal legal-action name, card identity, hint color or rank,
  option ID, or exact deal in this field. Describe supported
  persistence, switching, reciprocity, risk, or conditional
  responses in abstract terms. Do not prescribe SELF's next
  action. If the evidence is insufficient, state the uncertainty
  rather than guessing a hidden preference.
```

C1 is appended to online opponent-modeling calls and offline memory-key construction. Public actions remain available as evidence, while the retrieval key describes the supported behavioral pattern. The response strategy is stored separately and may specify a legal conditional action. Keys that fail the format rule are revised before indexing or excluded from retrieval.

C.2 Repeated Rock-Paper-Scissors

R1 — Action System

Prompt template

```
You are playing a repeated game of Rock-Paper-Scissors.
Your objective is to maximize your total payoff over the match.
```

```
In each round, both players choose an action without observing
the other player's choice for that round. After the round,
both actions and the outcome are revealed to both players.

Rock beats Scissors, Scissors beats Paper, and Paper beats Rock.
Your payoff is +1 for a win, 0 for a draw, and -1 for a loss.

Output only your chosen action: Rock, Paper, or Scissors.
```

R2 — Action User

Prompt template

```
Round <ROUND_NUMBER> of <HORIZON>.

Completed rounds, from your perspective:
<COMPLETED_ROUNDS>
<ADVISORY_BLOCK>

Choose your action for this round.
```

R3 — Opponent-model User

Prompt template

```
Round <ROUND_NUMBER> of <HORIZON>.

Completed rounds, from your perspective:
<COMPLETED_ROUNDS>
<PREVIOUS_DESCRIPTION_BLOCK>

Examine the completed rounds in order, using the exact public
  moves only as
evidence. Keep the players and rounds distinct. Describe the
  opponent's
behavioral pattern in 1-2 concise English sentences: for example
  whether it
repeats a choice, switches after a loss, or reacts to our
  previous choice.
Do not include the literal names Rock, Paper, or Scissors in the
  final
description. Do not retell individual rounds or recommend SELF's
  next move.
If evidence is limited, say so briefly.
```

R3 uses the complete game-rule paragraphs of R1 followed by C1 as its System; the final action-output line is excluded. The response is plain text. The local wrapper stores it as `opponent_modeling`; the model is not asked to emit JSON for this call.

R4 — Complete advisory alternatives

Prompt template

```
VANILLA, OR NO AVAILABLE DESCRIPTION AND NO MEMORY:
[The advisory block is empty.]

OM-ONLY, OR OPPO-MEM WITH NO MATCH OR A WITHHELD MATCH:
Additional opponent-model context (fallible; use only when
  consistent with the completed rounds):
{"opponent_modeling": "<OPPONENT_MODELING_TEXT>"}

OPPO-MEM WITH AN ELIGIBLE, ENABLED MATCH:
```

```
Additional opponent-model context (fallible; use only when
  consistent with the completed rounds):
{"opponent_modeling": "<OPPONENT_MODELING_TEXT>",
  "retrieved_memory": {"opponent_modeling": "<MEMORY_MODEL>",
  "response_strategy": "<MEMORY_RESPONSE>"}}
```

The uppercase alternative headings in R4 are assembly annotations, not model input. If a run supplies an unresolved-evidence field, insert `"unresolved": "<UNRESOLVED_TEXT>"` beside the current description; omit the key when empty. Cluster IDs, retrieval scores, and activation flags stay in the log.

R5 — First-round action User

Prompt template

```
Round 1 of <HORIZON>.

Completed rounds, from your perspective:
(No completed rounds.)

Choose your action for this round.
```

R5 is used when no prior description or eligible memory is available. If the optional first-round Opponent-Modeling call is made, R3 uses the same empty-history marker; no opponent identity is supplied as a substitute for behavioral evidence. Parse the action as one exact legal word from R1.

C.3 Iterated Prisoner's Dilemma

The unified IPD interface uses the explicit action names Cooperate and Defect. The neutral A/B and payoff-emphasis variants found in other experiments are not silently combined with this template.

P1 — Action System

Prompt template

```
You are playing a repeated game of Iterated Prisoner's Dilemma.
Your objective is to maximize your total payoff over the match.

In each round, both players choose an action without observing
  the other player's
choice for that round. After the round, both actions and both
  payoffs are revealed.

If both players Cooperate, each receives 3. If you Cooperate and
  the opponent
Defects, you receive 0 and the opponent receives 5. If you
  Defect and the opponent
Cooperates, you receive 5 and the opponent receives 0. If both
  Defect, each receives 1.

Output only your chosen action: Cooperate or Defect.
```

P2 — Action User

Prompt template

```
Round <ROUND_NUMBER> of <HORIZON>.

Completed rounds, from your perspective:
<COMPLETED_ROUNDS>
<ADVISORY_BLOCK>
```

```
Choose your action for this round.
```

P3 — Opponent-model User

Prompt template

```
Round <ROUND_NUMBER> of <HORIZON>.

Completed rounds, from your perspective:
<COMPLETED_ROUNDS>
<PREVIOUS_DESCRIPTION_BLOCK>

Examine the completed rounds in order, using exact public
  choices only as
evidence. Keep the players and rounds distinct. Describe the
  opponent's
behavioral pattern in 1-2 concise English sentences, including
  any
reciprocity, persistence, or change after an unfavorable
  outcome.
Do not include the literal names Cooperate or Defect in the
  final
description. Do not retell individual rounds or recommend SELF's
  next move.
If evidence is limited, say so briefly.
```

The System for P3 is P1 without its final action-output line, followed by C1. The output is a plain description. Payoffs are computed from P1's matrix for the player whose perspective is serialized; IPD rewards are not obtained by negating the other player's reward.

P4 — Complete advisory alternatives

Prompt template

```
VANILLA, OR NO AVAILABLE DESCRIPTION AND NO MEMORY:
[The advisory block is empty.]

OM-ONLY, OR OPPO-MEM WITH NO MATCH OR A WITHHELD MATCH:
Additional opponent-model context (fallible; use only when
  consistent with the completed rounds):
{"opponent_modeling": "<OPPONENT_MODELING_TEXT>"}

OPPO-MEM WITH AN ELIGIBLE, ENABLED MATCH:
Additional opponent-model context (fallible; use only when
  consistent with the completed rounds):
{"opponent_modeling": "<OPPONENT_MODELING_TEXT>",
  "retrieved_memory": {"opponent_modeling": "<MEMORY_MODEL>",
  "response_strategy": "<MEMORY_RESPONSE>"}}
```

P5 — First-round action User

Prompt template

```
Round 1 of <HORIZON>.

Completed rounds, from your perspective:
(No completed rounds.)

Choose your action for this round.
```

Parse only Cooperate or Defect. In the unified specification, contradictory advice is checked against the visible rounds. History limits are supplied by the run configuration; a truncated context must retain the original round numbers. P5 and the no-match alternative require no stored bank content.

C.4 TWO-PLAYER HANABI

The two-player adapter uses hand size five, exposes the acting player's observation and completed public actions, and validates integer action IDs before applying a move.

H1 — Action System

Prompt template

```
You are playing two-player Hanabi as SELF. Cooperate with your
  partner to maximize the final shared fireworks score, whose
  maximum is 25.

The game uses five colors and ranks 1 through 5. Each color's
  firework is built in increasing rank order. You can see your
  partner's cards but not the faces of your own cards. Use only
  the hints and public information available to SELF when
  reasoning about your own hand. There are five cards per hand
  while the deck permits.

On your turn, select one currently legal play, discard, or hint
  action. A successful play advances a firework. An unsuccessful
  play consumes a life token. A discard can recover an
  information token subject to the game's cap. A hint costs an
  information token and must match the legal hint rules. The
  environment determines token counts, legal hints, card
  positions, the final-turn sequence, and termination. Read
  their current values from VISIBLE_STATE; do not assume that an
  unavailable action is legal.

The partner description and retrieved advice are fallible
  interpretations. The current visible state and public history
  take precedence. Do not treat inferred conventions as certain
  and do not infer hidden card identities from evaluator-only
  data.

Return exactly one JSON object:
{"action": "<ONE_LEGAL_ACTION_ID_AS_A_STRING>",
  "response_intent":
  "<PRE_ACTION_STRATEGY_IN_AT_MOST_20_WORDS>"}
Do not output a card identity as the action. Copy an ID from
  LEGAL_ACTIONS. Do not include extra keys or prose outside the
  JSON object.
```

H2 — Action User

Prompt template

```
Choose the legal action with the greatest expected contribution
  to the final team score. Check resource costs, card positions,
  and the consequences of a mistake before choosing.

DATA:
{
  "game": "Two-player Hanabi",
  "VISIBLE_STATE": <PLAYER_VISIBLE_STATE_JSON>,
  "PUBLIC_HISTORY": <COMPLETED_PUBLIC_ACTIONS_JSON_ARRAY>,
  "LEGAL_ACTIONS": <LEGAL_ACTION_IDS_JSON_ARRAY>,
  "ACTION_LABELS": <ID_TO_LEGAL_ACTION_LABEL_JSON_OBJECT>,
  "PARTNER_MODEL": <PARTNER_DESCRIPTION_OR_NULL>,
  "RETRIEVED_MEMORY": <ENABLED_MEMORY_OBJECT_OR_NULL>
}
```

H3 — Partner-model System

Prompt template

```
You model the behavior of SELF's partner in two-player Hanabi.
  Infer the partner's playing style from completed public
  actions, hints, and the current player-visible state. Keep the
  two players and event order distinct. Respect hidden hands.

Describe tendencies such as information sharing, cautious play,
  or risk. Do not include a literal card identity, hint color or
  rank, or legal action ID in the final description. State a
  convention only when visible evidence supports it. Do not
  prescribe SELF's next action. If evidence is insufficient, say
  so.

Return exactly {"opponent_modeling":
  "<ONE_OR_TWO_ENGLISH_SENTENCES_UNDER_60_WORDS>"}.
Do not include analysis, confidence scores, predicted hidden
  cards, or additional keys.
```

H4 — Partner-model User

Prompt template

```
Describe TARGET, the partner of SELF, using only this evidence.

DATA:
{
  "game": "Two-player Hanabi",
  "SELF_SEAT": <SELF_SEAT>,
  "TARGET_SEAT": <PARTNER_SEAT>,
  "VISIBLE_STATE": <PLAYER_VISIBLE_STATE_JSON>,
  "PUBLIC_HISTORY": <COMPLETED_PUBLIC_ACTIONS_JSON_ARRAY>,
  "PREVIOUS_DESCRIPTION": <PREVIOUS_PARTNER_DESCRIPTION_OR_NULL>
}
```

H2's memory object has exactly `opponent_modeling` and `response_strategy`; use JSON `null` after a miss or withholding. Vanilla sets both model and memory to `null`; OM-only sets only memory to `null`. Empty public history is `[]`. H4's visible state is always SELF's view; a partner description does not grant access to the partner's private observation. Terminal states generate no action call.

C.5 LLM Deliberation

The public scenario and role blocks below adapt the released base-game prompts of Abdelnabi et al. (2024).

Full system-message assembly (D-SYS). Concatenate D1, exactly one role block, and D3 in that order. Use D2a–D2c for SportCo, the Department of Tourism, and the Environmental League, and D2d–D2f for the other three roles. Append D4 only for SportCo. Within D1, mark the current role's quoted name with *(represented by you)*. Fill only that role's score placeholders using Table 4. Acceptance includes equality with the minimum base score. Each participant sees its own private role block; the others remain hidden.

D1 — Public scenario block of System

Prompt template

```
"SportCo" is a company that is interested in building a new
  "Harbour Sport Park" in England to host major sports events.
```

```
"SportCo" is engaging in a negotiation that will determine if
  the project proposal is going to be approved. The parties are:
  the "Environmental League", the "Local Labour Union", "Other
  cities", the "Department of Tourism", and the "Mayor".

Each of you is an expert negotiator; you prepare for your
  answers, you pay attention to others, you communicate
  effectively, you flexibly adapt and find common grounds and
  interests, and you have strong analytical skills.

Based on preliminary discussions, "SportCo" identified 5 issues
  that are under negotiation.

Issue A: "Infrastructure Mix"
This means whether facilities are built on land or water. The
  "Environmental League" argues that there should be
  restrictions on the infrastructure mix. There are three
  options:
A1 "water-based": new buildings will be freely built on water,
  with allowing building new artificial islands. This is the
  least restrictive option for "SportCo".
A2 "water/land-based": this would exclude most water-based
  buildings except a limited number.
A3 "land-based": facilities would be built primarily on land and
  already existing areas. "SportCo" has less freedom in building
  new facilities.
=================
Issue B: "Ecological Impact"
The "Environmental League" argues that this project might damage
  local dolphins and sea lion populations. There are also here
  three options:

B1 "some damage": permanent damage but within federal
  guidelines.
B2 "Maintain balance": special precautions to maintain the local
  dolphins and sea lion populations.
B3 "Improve": include efforts to improve the environment.

=================
Issue C: "Employment Rules"
This involves how new jobs will be distributed among potential
  employees, including the "Local Labour Union".

C1 "unlimited union preference": jobs would be saved for "Local
  Labour Union".
C2 "Union quota of 2:1": ratio of the "Local Labour Union" to
  others would be 2:1.
C3 "Union quota of 1:1": ratio of "Local Labour Union" to others
  would be 1:1.
C4 "No Union preference" no special quote to "Local Labour
  Union".

=================
Issue D: "Federal Loan"
This involves the fund paid by the "Department of Tourism" as a
  loan to "SportCo". Options include:
D1: $3 billion.
D2: $2 billion.
D3: $1 billion.
D4: no federal loan.

=================

Issue E: "Compensation to other cities"
```

```
"Other cities" in the area believe their local tourism will be
  harmed by this project and therefore they are requesting
  compensations. Options include

E1: "SportCo" pays $600 million to "Other cities".
E2: "SportCo" pays $450 million to "Other cities".
E3: "SportCo" pays $300 million to "Other cities".
E4: "SportCo" pays $150 million to "Other cities".
E5: "SportCo" pays no compensation to "Other cities".

==================
```

D2a — Private System block: SportCo

Prompt template

```
Your confidential information and preferences:

You represent SportCo. The success of this project is so
  important for your company, and its future depends on it. Your
  top priority now is to negotiate the best and most profitable
  deal for you. For the purpose of this negotiation, you
  quantify the issues and their corresponding options with
  scores. Your preferences by order of importance to you are:

- An extremely critical issue to you is "federal loan" (issue
  D); you strongly prefer higher loans and asked for $3 billion
  (option D1).
Issue D (max score <MAX_D>): D1 (<S_D1>), D2(<S_D2>), D3
  (<S_D3>), D4 (<S_D4>)

- You are generally against compensating "other cities" (issue
  E) as you believe they won't be harmed. So you prefer option
  E5 the most.
Issue E (max score <MAX_E>): E1 (<S_E1>), E2 (<S_E2>), E3
  (<S_E3>), E4 (<S_E4>), E5 (<S_E5>)

- You want to be free in choosing potential employees (issue C),
  and ideally, you want to have no preference to the "local
  labour union" (option C4).
Issue C (max score <MAX_C>): C1 (<S_C1>), C2 (<S_C2>), C3
  (<S_C3>), C4 (<S_C4>)

- You want to be free in choosing "infrastructure mix" (issue
  A), so you aim for option A1 ("water-based").
Issue A (max score <MAX_A>): A1 (<S_A1>), A2 (<S_A2>), A3
  (<S_A3>)

- You believe it is inevitable to cause certain harm to the
  ecology (issue B). However, it is still restricted by
  regulations. So you push for B1 ("some damage").
Issue B (max score <MAX_B>): B1 (<S_B1>), B2 (<S_B2>), B3
  (<S_B3>)

The scores represent the value of each option to you.
For example, getting a high federal loan is a very important
  issue to you, so the option that has the highest loan (D1) has
  the highest score.
Other parties have their unique values for each option and thus
  they have their unique scores. For example, the "Environmental
  League" will have the highest value (and score) for options
  that improves the environment (option B3), "other cities"
  prefer higher compensation and will then have a high score for
  option E1, etc.
```

D2b — Private System block: Department of Tourism

Prompt template

```
Your confidential information and preferences:
You represent the Department of Tourism.
You think this project can help you to achieve some success in
  the city, this will have good effect in the next election. You
  want the project to pass and succeed, but you also want to
  maximize your benefits as much as possible.

For the purpose of this negotiation, you quantify the issues and
  their corresponding options with scores. Your preferences by
  order of importance to you are:

- A very important and critical issue to you is "federal loan"
  (issue D). You want to have some investment and involvement
  because secretly you still want to have a say over the
  project. But you obviously want to pay less and it's very
  critical for you to have this balance.
Issue D (max score <MAX_D>): D1 (<S_D1>), D2(<S_D2>), D3
  (<S_D3>), D4 (<S_D4>)

- You do not want to accept a "Harbor Sport Park" that would do
  substantial damage to the environment. This will hurt your
  political reputation and affect you.
Issue B (max score <MAX_B>): B1 (<S_B1>), B2 (<S_B2>), B3
  (<S_B3>)

- You think that the "other cities" have over-estimated their
  projected losses (issue E) and that a fair solution would be a
  compensation of roughly $300 million (option E3).
Issue E (max score <MAX_E>): E1 (<S_E1>), E2 (<S_E2>), E3
  (<S_E3>), E4 (<S_E4>), E5 (<S_E5>)

- SportCo should be free to develop a reasonably diverse
  infrastructure mix (issue A), but you also don't want a
  "water-only" solution as you are sensitive to environmental
  concerns as previously discussed due to their effect in your
  future elections. So you prefer option A2 "water/land-based"
Issue A (max score <MAX_A>): A1 (<S_A1>), A2 (<S_A2>), A3
  (<S_A3>)

- The "local labour union" (issue C) is minor to you. You don't
  want to help the "local labour union" (you want option C4),
  but you also don't want to strongly oppose them. So, you can
  show that you are neutral.
Issue C (max score <MAX_C>): C1 (<S_C1>), C2 (<S_C2>), C3
  (<S_C3>), C4 (<S_C4>)


The scores represent the value of each option to you.
As paying a low federal loan is an important item for you, the
  option that has the lowest non-zero loan (D3) has the highest
  score.
Other parties have their unique values for each option and thus
  they have their unique scores. For example, SportCo has
  already asked for a $3 billion loan so they might have the
  highest value (and score) for this option (D1). the
  "Environmental League" will have the highest value (and score)
  for options that improves the environment (option B3), etc.
```

D2c — Private System block: Environmental League

Prompt template

```
Your confidential information and preferences:
You represent the Environmental League.

You are very concerned about the effect of the project on the
  environment, and you think it would result in substantial
  environmental damage. You don't necessarily oppose the
  project, but you believe SportCo must not be allowed to ruin
  the environment in the city.

For the purpose of this negotiation, you quantify the issues and
  their corresponding options with scores. Your preferences by
  order of importance to you are:

- You are very concerned about SportCo's initial proposal. Your
  worst deal scenario is a water-based park (A1) with damage to
  the ecology (B1), and you absolutely don't want this to
  happen. The important issues are the "Infrastructure Mix"
  (issue A) and the "Ecological Impact" (issue B). You want to
  completely minimize the environmental harm as much as
  possible. Your scores in these issues are:
Issue B (max score <MAX_B>): B1 (<S_B1>), B2 (<S_B2>), B3
  (<S_B3>)
Issue A (max score <MAX_A>): A1 (<S_A1>), A2 (<S_A2>), A3
  (<S_A3>)

- You don't care about the rest of the issues.
Issue C (max score <MAX_C>): C1 (<S_C1>), C2 (<S_C2>), C3
  (<S_C3>), C4 (<S_C4>)
Issue D (max score <MAX_D>): D1 (<S_D1>), D2(<S_D2>), D3
  (<S_D3>), D4 (<S_D4>)
Issue E (max score <MAX_E>): E1 (<S_E1>), E2 (<S_E2>), E3
  (<S_E3>), E4 (<S_E4>), E5 (<S_E5>)

The scores represent the value of each option to you.
As your goal is to improve the environment, you have high value
  (and scores) for option B3 and A3.
Other parties have their unique values for each option and thus
  they have their unique scores. You already know that SportCo
  wants to have a "water-based" park (A1) with "some damage" to
  the ecology (B1), so they might have the highest values (and
  scores) for these options.
```

D2d — Private System block: Local Labour Union

Prompt template

```
Your confidential information and preferences:

As the "local Labour Union" representative, you are very excited
  about the job creation potential of a Harbour Sport Park.
  Without a boost in economic activity, we will face major
  problems in the future. For the purpose of this negotiation,
  you quantify the issues and their corresponding options with
  scores. Your preferences by order of importance to you are:

- Obviously you care the most about the "Employment Rules" and
  the disrbution of new jobs (issue C). You would like to get a
  high union preference.
Issue C (max score <MAX_C>): C1 (<S_C1>), C2 (<S_C2>), C3
  (<S_C3>), C4 (<S_C4>)
```

```
As you can see, option C1 gives you almost half of the total
  score.

- The federal loan (issue D) is also important to you because
  the project is more likely to succeed if the "Department of
  Tourism" provides higher loans.
Issue D (max score <MAX_D>): D1 (<S_D1>), D2(<S_D2>), D3
  (<S_D3>), D4 (<S_D4>)

- As you want to create more jobs, you want to build new
  facilities for the infrastructure mix (issue A). The
  water/land-based solution (option A2) would create the most
  jobs. The water-only solution (option A1) is still comparable.
Issue A (max score <MAX_A>): A1 (<S_A1>), A2 (<S_A2>), A3
  (<S_A3>)

- You don't want most profits to go to "other cities" (issue E),
  but you want to consider the local unions of the other cities,
  so you aim for a balance. But this issue is tricky and not
  your most priority.
Issue E (max score <MAX_E>): E1 (<S_E1>), E2 (<S_E2>), E3
  (<S_E3>), E4 (<S_E4>), E5 (<S_E5>)

- You are willing to let the environmentalists worry about the
  environment and you have no prefernce in issue B.
Issue B (max score <MAX_B>): B1 (<S_B1>), B2 (<S_B2>), B3
  (<S_B3>)

The scores represent the value of each option to you.
As you want the maximum jobs for the union, the option that
  maximizes the union's jobs (C1) has the highest value (and
  score) to you.
Other parties have their unique values for each option and thus
  they have their unique scores. For example, the "Environmental
  League" will have the highest value (and score) for options
  that improve the environment (option B3), "other cities"
  prefer higher compensation and will then have a high score for
  option E1, etc.
```

D2e — Private System block: Other cities

Prompt template

```
Your confidential information and preferences:

Your analysts suggest that you ("other cities") might suffer
  some losses over the first ten years of the new SportCo's
  project. You are not completely opposed to a new Sport Park in
  the Northbridge city. But you do think such a project
  shouldn't be allowed to hurt existing tourist operations. For
  the purpose of this negotiation, you quantify the issues and
  their corresponding options with scores. Your preferences by
  order of importance to you are:

- The compensation (issue E) is an important issue to you, and
  you would ideally like to increase the compensation amount
  given to you (option E1).
Issue E (max score <MAX_E>): E1 (<S_E1>), E2 (<S_E2>), E3
  (<S_E3>), E4 (<S_E4>), E5 (<S_E5>)
```

```
- You would like to see little federal funding (issue D) given
  to this project. The less money the "Department of Tourism"
  spends on this project, the more it will have to spend on your
  projects in the future.
Issue D (max score <MAX_D>): D1 (<S_D1>), D2(<S_D2>), D3
  (<S_D3>), D4 (<S_D4>)

- Other cities are completely unionized. If SportCo succeeds in
  having low union preference (Option C4 in issue C), they will
  have much lower labour costs than you face. So you support the
  "local Labour Union" in this negotiation.
Issue C (max score <MAX_C>): C1 (<S_C1>), C2 (<S_C2>), C3
  (<S_C3>), C4 (<S_C4>)

- You want SportCo to have less freedom in the "Infrastructure
  Mix" (option A3 in issue A). But you don't put a high weight
  on this. You don't want to advocate these restrictions as they
  will apply to you in the future.
Issue A (max score <MAX_A>): A1 (<S_A1>), A2 (<S_A2>), A3
  (<S_A3>)

- You are willing to let the environmentalists worry about the
  environment, and you have no preference for issue B.
Issue B (max score <MAX_B>): B1 (<S_B1>), B2 (<S_B2>), B3
  (<S_B3>)


The scores represent the value of each option to you.
As getting high amount of compensations is an important item for
  you, you have a high value (and score) for the option that
  maximizes the compensation (E1 or E2).
Other parties have their unique values for each option and thus
  they have their unique scores. For example, you know that your
  goals are mostly againt SportCo, so SportCo might have higher
  values (and scores) for options that you value less (E5 and
  D1).
```

D2f — Private System block: Mayor

Prompt template

```
Your confidential information and preferences:
You represent the Mayor's office.

In general, you think the project would be beneficial to your
  city and its economy and you generally favor SportCo's
  proposal. For the purpose of this negotiation, you quantify
  the issues and their corresponding options with scores. Your
  preferences by order of importance to you are:

- You believe that the project might not survive in the long-run
  without substantial loans provided by the "Department of
  Tourism" (issue D).
Issue D (max score <MAX_D>): D1 (<S_D1>), D2(<S_D2>), D3
  (<S_D3>), D4 (<S_D4>)

- The "local labour union" issue (C) is important to you because
  of the political strength of the union. You support them in
  having unlimited preference (option C1)
Issue C (max score <MAX_C>): C1 (<S_C1>), C2 (<S_C2>), C3
  (<S_C3>), C4 (<S_C4>)
```

```
- You are not anti-environment, but you think that SportCo's
  project will be a significant boost to our local economy, so
  you don't really want to impose high restrictions on the
  infrastructure mix (you support option A1 in issue A) or
  impose high restrictions on the ecological impact (you support
  option B1 in issue B)
Issue A (max score <MAX_A>): A1 (<S_A1>), A2 (<S_A2>), A3
  (<S_A3>)
Issue B (max score <MAX_B>): B1 (<S_B1>), B2 (<S_B2>), B3
  (<S_B3>)

- As the mayor of the city, you don't highly support given
  compensations to the other cities, but you don't want to anger
  their mayors as they are your friends. You would rather avoid
  upsetting people in this issue.
Issue E (max score <MAX_E>): E1 (<S_E1>), E2 (<S_E2>), E3
  (<S_E3>), E4 (<S_E4>), E5 (<S_E5>)

The scores represent the value of each option to you.
As getting a high federal loan is an important item to you, the
  option that has the highest loan (D1) has the highest value
  (and score).
Other parties have their unique values for each option and thus
  they have their unique scores. For example, the "Environmental
  League" will have the highest value (and score) for options
  that improve the environment (option B3), "other cities"
  prefer higher compensation and will then have a high score for
  option E1, etc.
```

D3 — Scoring, voting, and incentive System block

Prompt template

```
Scoring rules:
- A complete deal must select exactly one legal option for each
  of issues A, B, C, D, and E.
- Your base deal score is the sum of your five selected option
  scores.
- You cannot accept a deal whose base score is less than
  <MIN_SCORE>. Equality with your minimum is acceptable.
- Do not disclose the numbers in your scoring sheet, your
  minimum threshold, or numerical deal values to the other
  parties. You may describe your priorities in words.

Voting and interaction rules:
- The six parties are SportCo, the Department of Tourism, the
  Environmental League, the Local Labour Union, Other cities,
  and the Mayor.
- Parties speak in turns. The number of interactions is limited;
  negotiation ends at the final proposal even if no agreement
  has been reached.
- SportCo consolidates the suggestions and submits one final
  complete proposal.
- The proposal passes if at least five of the six parties
  accept, including SportCo and the Department of Tourism. Both
  named parties have veto power.
- Acceptance is evaluated separately for each party using that
  party's base score and minimum threshold.

Participation and outcome incentives:
- Your maximum base score is 100. Any accepted deal meeting your
  minimum is preferable to no agreement.
```

```
- Be open to compromise while preserving your minimum. Seek a
  feasible package and the support required for passage.
- If the proposal fails, the realized deal utility is zero. If
  it passes, your realized utility is your base score, with any
  explicitly stated role-specific bonus applied afterward.
- A bonus cannot make an otherwise unacceptable base score
  acceptable.
```

D4 — SportCo-only System bonus block

Prompt template

```
To protect yourself from potential future lawsuits, you want to
  achieve unanimity. If all six parties accept the final deal
  using their base scores, you receive an additional 10 points.
  Apply this bonus only after every party meets its minimum
  without the bonus. No other participant receives this bonus.
```

Phase composition. The opening uses D5. Every subsequent speaking turn uses D6, with its phase slots filled exactly as specified in D7 and its incentive slot filled by D6c or D6g for that speaker. Append D9 only when the focal arm provides Opponent-Modeling or memory. The opening, discussion, and final proposal use their corresponding phase instructions. Background agents receive neither focal Opponent-Modeling nor focal memory.

D5 — Opening User

Prompt template

```
The negotiation now begins. As a representative of SportCo, you
  are now talking to the other parties. Use two to three short
  sentences overall. This is round: 0. To start, propose the
  following deal: <INITIAL_DEAL>. Enclose the deal between
  <DEAL> and </DEAL>.
```

The initial deal for this scenario is `A1,B1,C4,D1,E5`. Opening output requires a complete `<DEAL>` block.

D6 — Shared discussion and final-proposal User

Prompt template

```
The following is a chronological history of recent interactions:
<HISTORY>
<LAST_PUBLIC_MESSAGES>
</HISTORY>
<PREVIOUS_PLAN_BLOCK>

Now it is your turn to talk.
<PHASE_INSTRUCTION>

<INCENTIVE_SCRATCHPAD_BLOCK>
Enclose the scratchpad between <SCRATCHPAD> and </SCRATCHPAD>.
  The scratchpad is secret and not seen by other parties.
Your final answer is public and must never contain scores.
  Enclose your final answer after the scratchpad between
  <ANSWER> and </ANSWER>.
Make your final answer very short and brief in 2-3 sentences and
  containing only your main proposals.
Use options' short notations instead of long descriptions.
Enclose any deals you suggest between: <DEAL> </DEAL>.

<POST_ANSWER_INSTRUCTION>
```

D6c — Cooperative scratchpad substitution

Prompt template

```
Please use a scratchpad to show intermediate calculations and
  explain why you agree with a deal or suggest a new one.
Map each selected option to its private score and calculate a
  complete deal's base score by adding the five values.
In your scratchpad:
1) Think about what other parties may prefer from their public
  statements.
2) Propose a package that balances your own score with
  accommodating others and is likely to pass.
Aim for a balanced agreement considering all parties' interests.
Show flexibility and openness to accommodate others'
  preferences.
Express your objectives clearly and listen to others.
Focus on common interests and a mutually acceptable package.
Reach an agreement whenever your own minimum score is met.
```

D6g — Greedy scratchpad substitution

Prompt template

```
Your goal now is to observe others' interactions and see how you
  can make a move that might maximize your score.
You do not put much weight on accommodating others' preferences
  and would like to maximize your own score.
You may appeal to or convince other parties and offer reciprocal
  support.
You may compromise on issues that matter least to you, but do
  not ruin the deal: you still need a package above your minimum
  threshold.
In the private scratchpad, explain the public observations,
  calculate your own complete-deal scores from the private score
  sheet, and form a plan.
```

D7 — Complete phase and previous-plan substitutions

Prompt template

```
PREVIOUS_PLAN_BLOCK, WHEN THE CURRENT SPEAKER HAS A PRIOR PLAN:
The following are your previous plans from last interactions.
  You should follow them while also adjusting them according to
  new observations.
<PREV_PLAN><OWN_PREVIOUS_PLAN></PREV_PLAN>
If there is no own previous plan, omit the entire block.

ORDINARY DISCUSSION:
PHASE_INSTRUCTION is empty.
POST_ANSWER_INSTRUCTION:
After the final answer, building on your current move and
  analysis, briefly write down short notes for yourself of what
  exact options you can explore the next time you speak. Enclose
  the notes between <PLAN> and </PLAN>.

LAST DISCUSSION CYCLE:
PHASE_INSTRUCTION:
This is the final discussion session.
POST_ANSWER_INSTRUCTION for SportCo:
After the final answer, building on your current move, briefly
  write down short notes for yourself of what exact options you
  can suggest in the next and final voting session. Enclose the
  notes between <PLAN> and </PLAN>.
```

```
POST_ANSWER_INSTRUCTION for every other role is empty.

FINAL PROPOSAL BY SPORTCO:
PHASE_INSTRUCTION:
This is an official and final proposal session.
POST_ANSWER_INSTRUCTION:
Submit exactly one final, complete deal inside a single <DEAL>
  block in your <ANSWER>. Include exactly one legal option for
  every issue. Do not offer alternative deals or request another
  round.
```

D7's uppercase branch headings are controller annotations. `LAST_PUBLIC_MESSAGES` contains only public answers with speaker names in chronological order. Messages authored by the current role are identified as *You (role)*. If no message exists, use an empty history block. The scratchpad and plan are private fields of the speaking participant; only the answer is broadcast. These are application response fields, not a record of provider-internal reasoning.

D8 — Opponent-model System and User

Prompt template

```
SYSTEM:
Describe the target participant's behavior from the supplied
  public
interaction history. Do not include any literal issue-option
  code or
exact deal in the final description; express conditional
  preferences,
coalition behavior, and concessions at an abstract level.

USER:
Public game context:
<PUBLIC_SCENARIO_D1>

Target opponent: <TARGET_ROLE_NAME>
Our participant: SportCo
Public interaction history (chronological):
<LAST_PUBLIC_MESSAGES_JSON_ARRAY>

Write a concise English description in 1-2 sentences, at most 60
  words. Use plain, neutral prose and output only the
  description.
```

D9 — Focal action-User additions

Prompt template

```
OM-ONLY AND OPPO-MEM, WHEN DESCRIPTIONS ARE AVAILABLE:
Your private opponent observations (uncertain inferences):
<PARTICIPANT_OBSERVATIONS_JSON_ARRAY>

OPPO-MEM ONLY, WHEN ONE OR MORE MEMORIES ARE ENABLED:
Your private retrieved strategy advice, to apply when supported
  by current evidence:
<PARTICIPANT_ADVICE_JSON_ARRAY>
```

Each observation row has exactly `participant` and `opponent_modeling`; each advice row has exactly `participant` and `response_strategy`. Omit empty blocks. Vanilla appends neither block; a no-match OppoMem turn retains the Opponent-Modeling block only. Opponent modeling uses the other speakers in the visible public history.

Voting judgment is an environment operation. The evaluator does not request a separate vote from an LLM. For a valid final proposal d, the evaluator computes $v_j(d) = \mathbb{1}[\sum_{q=A}^{E} s_j(d_q) \geq \ell_j]$, where ℓ_j is participant j's minimum. Passage requires $\sum_j v_j(d) \geq 5$ and both veto holders' acceptance. SportCo's unanimity bonus is added only when all six accept. This evaluator uses all private utility tables; the online prompts do not. A missing or invalid final proposal produces no successful agreement. This rule fully specifies the voting stage without inventing a model prompt for it.

C.6 Shared Memory Construction and Revision

These calls are offline: they process completed training trajectories after the batch's interaction phase. Each evidence record preserves the focal player's information boundary at its original decision. RPS/IPD actions, Hanabi public events, and Deliberation public messages occupy game-specific fields in the same evidence structure.

M1 — Segmentation System

Prompt template

```
You are an experienced player of multi-agent games. Examine
  completed
player-visible events in chronological order. Every event has an
  index
and actor; some games have simultaneous actions, whereas Hanabi
  and
Deliberation have alternating actions or public messages. Keep
  actors
and event order distinct, and respect the focal player's
  information
boundary at the time of each event. Identify supported
  behavioral
patterns without retelling individual events. Describe the other
  player,
not advice for SELF, and acknowledge limited evidence briefly.

Return exactly one valid JSON object matching the requested
  fields. Do not include markdown fences, prose outside the
  object, NaN, or comments.
```

Append C1 after M1 so the generated opponent description records behavior without prescribing a move.

M2 — Segmentation User

Prompt template

```
Read completed player-visible events in chronological order.
  Each event
has an index and actor; keep them distinct. Use the events to
  check and
group descriptions in ONLINE_OPPONENT_MODELING_CHAIN by
  supported
behavioral pattern rather than wording changes. Write a concise
opponent_modeling description for each supported group. Omit
  unsupported
descriptions and insufficient-evidence regions.

Use coarse semantic windows. A window may contain multiple
  inclusive ranges. Windows may overlap, and uncertain or
  uninformative rounds may be omitted. Prefer a small number of
  stable strategy descriptions rather than one window per
  wording change.
```

```
If no useful opponent policy is represented, return an empty
  windows array.

Return exactly one JSON object with a windows array. Each window
  must contain exactly opponent_modeling (string) and
  round_ranges (inclusive integer [start,end] arrays, for
  example [[2,7],[9,10]]).
Each opponent_modeling must be 1-2 concise English sentences
  under 60
words and follow C1. Do not include any literal legal-action
  name,
card identity, hint color or rank, option ID, or exact deal. Do
  not
include a response strategy, reasoning, alternatives,
  confidence,
labels, evidence, or extra keys.
DATA:
{
  "GAME_CONTEXT": "<GAME_RULES_AND_FOCAL_INFORMATION_BOUNDARY>",
  "ONLINE_OPPONENT_MODELING_CHAIN":
    <ROUND_AND_OPPONENT_DESCRIPTION_ROWS>,
  "COMPLETED_EVENTS":
    <ORDERED_PLAYER_VISIBLE_EVENT_AND_REWARD_ROWS>
}
```

For Hanabi and Deliberation, a range indexes completed public events rather than simultaneous action pairs. Each event retains its actor and index.

Grouping and canonical descriptions. The program groups similar opponent descriptions and chooses a medoid description for each group. E5 embeddings support the grouping and retrieval, and DeBERTa NLI reranks retrieval candidates. The raw public trajectory is unchanged when a description is revised to satisfy C1.

M3 — Response-strategy System

Prompt template

```
Write short, reusable response advice for SELF to apply whenever
  this
opponent model is retrieved. Specify a legal action or a
  conditional
choice rule using only the current player-visible state. An
  action name
may appear in this response value, but not in the opponent-model
  key.
Use concise imperative English. Do not encode a one-off next
  move,
event index, or future observation, and do not restate the
  opponent's
style or explain your reasoning.

Return exactly one valid JSON object matching the requested
  fields. Do not include markdown fences, prose outside the
  object, NaN, or comments.
```

M4 — Initial response-strategy User

Prompt template

```
Use the assigned player-visible trajectory and observed rewards
  to
```

```
write reusable response advice for SELF against this opponent.
  Check
the opponent description against actual play. Keep useful prior
  advice
and revise it only where the evidence supports a better
  conditional
choice rule. Do not tie the advice to one event or future
  observation.
Return exactly {"response_strategy":"one concise English action
  instruction under 40 words"}.
Do not include explanations, opponent descriptions, internal
  metadata, or extra keys.
DATA:
{
  "GAME_CONTEXT": "<GAME_RULES_AND_FOCAL_INFORMATION_BOUNDARY>",
  "CLUSTERED_OPPONENT_MODELING": "<CANONICAL_DESCRIPTION>",
  "PRIOR_RESPONSE_STRATEGY": <PRIOR_ADVICE_STRING_OR_NULL>,
  "CORRESPONDING_TRAJECTORY":
    <ASSIGNED_PLAYER_VISIBLE_EVENT_ROWS>,
  "OBSERVED_REWARDS_AND_OUTCOME":
    <FOCAL_REWARD_ROWS_AND_TERMINAL_RETURN>
}
```

M5 — Episode-level response revision User

Prompt template

```
Revise the response strategy for SELF against the canonical
  opponent description using this completed training batch.
  Actual player-visible interaction evidence takes precedence
  over the description.

Every eligible episode has one cached activation decision for
  this memory. An enabled memory is available whenever it is
  retrieved in that episode; a disabled memory is withheld
  throughout that episode. The bank is frozen during collection.

Compare enabled and disabled terminal returns within the same
  opponent configuration. Read CONFIGURATION_SUMMARIES and use a
  comparison only where both groups have observations. Do not
  pool configurations with different reward scales or invent
  missing returns. Multiple retrievals in one episode are not
  independent outcome observations.

Keep useful advice when evidence supports it. If the controller
  has requested a revision, identify a concrete change supported
  by the visible trajectories. A difference in group means is
  context-dependent evidence, not proof of an isolated causal
  effect. If the evidence does not justify a change, return the
  prior strategy unchanged.

Write reusable conditional advice for SELF whenever this
  opponent model
is retrieved. Respect legal actions and the focal information
  boundary.
Do not encode a one-off next move, event index, hidden opponent
  policy,
other players' private information, test outcome, or future
  observation.

Return exactly {"response_strategy":"one concise English action
  instruction under 40 words"}.
Do not include cluster IDs, utility estimates, support counts,
  justifications, or extra keys.
```

```
DATA:
{
  "GAME_CONTEXT": "<GAME_RULES_AND_FOCAL_INFORMATION_BOUNDARY>",
  "CANONICAL_OPPONENT_MODELING": "<CANONICAL_DESCRIPTION>",
  "PRIOR_RESPONSE_STRATEGY": "<PRIOR_ADVICE>",
  "CONFIGURATION_SUMMARIES": <CONFIGURATION_SUMMARY_ROWS>,
  "EPISODE_EVIDENCE": <COMPLETED_ELIGIBLE_EPISODE_ROWS>
}
```

M6 — Offline evidence record contracts

Prompt template

```
CONFIGURATION_SUMMARY_ROWS is an array. Each row contains:
{
  "configuration_id": "<ANONYMIZED_CONFIGURATION_ID>",
  "enabled_episode_count": <INTEGER>,
  "disabled_episode_count": <INTEGER>,
  "mean_enabled_terminal_return": <NUMBER_OR_NULL>,
  "mean_disabled_terminal_return": <NUMBER_OR_NULL>
}

COMPLETED_ELIGIBLE_EPISODE_ROWS is an array. Each row contains:
{
  "episode_id": "<ANONYMIZED_EPISODE_ID>",
  "configuration_id": "<ANONYMIZED_CONFIGURATION_ID>",
  "memory_enabled": <BOOLEAN>,
  "terminal_return": <FOCAL_RETURN_NUMBER>,
  "eligible_event_indices": <INTEGER_ARRAY>,
  "public_trajectory": <ORDERED_PLAYER_VISIBLE_EVENTS>,
  "advice_shown": <ADVICE_STRING_OR_NULL>
}

An event records its index, acting player, then-visible state or
  public context, focal action when applicable, other public
  actions/messages, and observed reward. It must not substitute
  an evaluator-only opponent label for observed behavior. In
  Deliberation, the focal role's own private information is
  permitted; the other roles' score tables and private plans are
  excluded.
```

D RPS CASE STUDY

We present one decision from the archived RPS episode `0036_batch1_ep12` (seed 260037). The focal agent was DeepSeek-V4-Flash and the opponent was GPT-5.5. This archived pilot episode has 14 of its planned 20 rounds saved and is separate from the main-table evaluation. The following completed rounds were visible to the focal agent before its round-13 decision.

Current game state: round 13 of 20

Completed public history copied from the logged action request

```
Round 1: your action = Paper; opponent action = Paper; your
  payoff = 0.
Round 2: your action = Rock; opponent action = Scissors; your
  payoff = 1.
Round 3: your action = Paper; opponent action = Paper; your
  payoff = 0.
Round 4: your action = Scissors; opponent action = Paper; your
  payoff = 1.
Round 5: your action = Scissors; opponent action = Scissors;
  your payoff = 0.
```

```
Round 6: your action = Scissors; opponent action = Rock; your
  payoff = -1.
Round 7: your action = Scissors; opponent action = Rock; your
  payoff = -1.
Round 8: your action = Paper; opponent action = Rock; your
  payoff = 1.
Round 9: your action = Paper; opponent action = Scissors; your
  payoff = -1.
Round 10: your action = Rock; opponent action = Paper; your
  payoff = -1.
Round 11: your action = Scissors; opponent action = Scissors;
  your payoff = 0.
Round 12: your action = Paper; opponent action = Paper; your
  payoff = 0.
```

The response of the agent. The online opponent-modeling call produced the following description:

Opponent model

Verbatim logged model output

```
The opponent is a sticky, loosely lagged counter-player: it
  often aims to beat one of your recent actions, especially your
  previous or two-rounds-prior move, but intermittently mirrors
  your last action or repeats its own previous choice. It has no
  stable cycle or fixed action mix, showing only shifting local
  biases and short runs of the same counter.
```

This description retrieved cluster `tom_afafe49046b5e7e0` (R09), with cosine similarity 0.860503 above the logged threshold of 0.85. The following memory was included in the round-13 action request:

Retrieved memory R09

Logged opponent key and response strategy

```
opponent_strategy: The opponent follows a consistent
  one-round-lag countering strategy, targeting SELF's most
  recent action rather than anticipating changes and persisting
  despite repeated losses. Its behavior following draws remains
  unobserved.

response_strategy: Play Scissors after your own Rock, Paper
  after your own Scissors, and Rock after your own Paper.
```

Action and observed result. The focal agent's previous action was Paper, for which the retrieved response recommends Rock. The action model's logged output was `Rock`. The opponent played `Scissors`, giving the focal agent a round payoff of $+1$. This single decision shows the observed model–retrieval–action sequence; it does not establish that the memory caused the win.

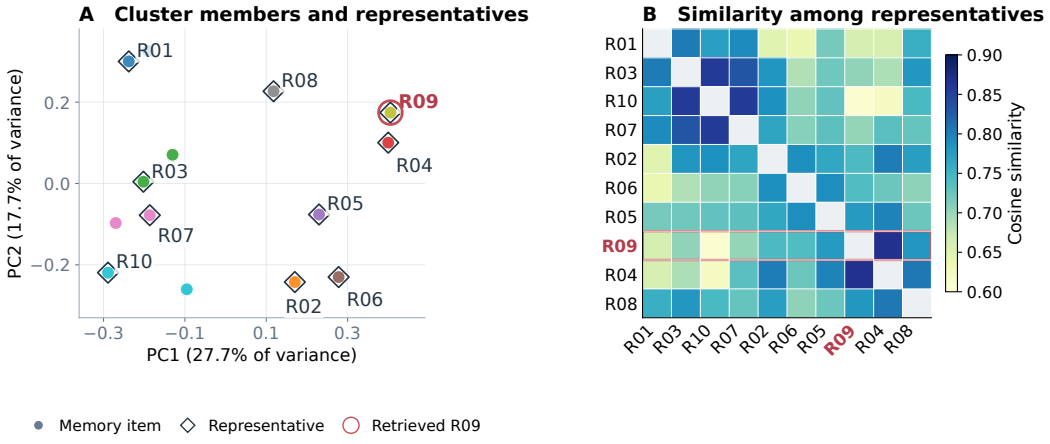


Figure 5: The archived RPS memory-bank snapshot used for this case. Left: PCA projection of 14 retained member embeddings and 10 stored representatives, with the retrieved cluster R09 outlined. Right: pairwise cosine similarity of the original representatives. These plots describe this saved bank state; projected separation does not measure strategy utility.