OrchDebate: Adaptive Orchestration for Multi-Agent Debate via Self-Evolving Skills

Anonymous ACL submission

Abstract

Multi-Agent Debate (MAD) has emerged as a promising paradigm for eliciting stronger inference-time reasoning, yet most existing systems still use fixed interaction protocols across instances. Such static designs can suffer from anchoring bias, overthinking on easy problems, and rigid coordination across tasks of varying difficulty. We present **OrchDebate**, a framework that recasts multi-agent debate as structured-state orchestration over reusable debate-control skills. Instead of following a pre-defined template, OrchDebate uses an orchestrator to examine the current debate state and select the next coordination action, such as introducing agents under different visibility settings or terminating deliberation with an adaptive readout rule. This skill-based view makes debate control modular and state-dependent. The orchestrator operates through a structured control interface rather than directly generating reasoning content, helping reduce reliance on early answers. OrchDebate further maintains a self-evolving skill bank, allowing coordination strategies to improve online from the system's own debate trajectories, without additional external data. Across diverse benchmarks, OrchDebate delivers a better accuracy–cost trade-off than fixed debate protocols by allocating debate compute more selectively.

1 Introduction

Multi-Agent Debate (MAD), where multiple large language model (LLM) instances deliberate over a shared query before producing a final answer, has emerged as a promising paradigm for achieving stronger collective intelligence in inference-time reasoning (Xi et al., 2025; Luo et al., 2025; Guo et al., 2024; Tran et al., 2025; Yin et al., 2023; Chen et al., 2024; Zeng et al., 2025; Sun et al., 2025; Becker et al., 2025; Du et al., 2024; Xiong et al., 2023; Xu et al., 2023; Li et al., 2024a). Yet a central limitation remains: most MAD systems still operate under a pre-specified and largely fixed interaction protocol, with fixed agent counts, interaction patterns, debate rounds, and decision rules (Guo et al., 2024; Tran et al., 2025; Tillmann, 2025; Cui et al., 2025; Zhang et al., 2026c).

However, this static design is increasingly mismatched to the heterogeneous nature of reasoning tasks. Recent evidence points to three recurring failure modes under fixed debate protocols: i) anchoring bias, where agents gravitate toward an initial answer instead of correcting it (Weng et al., 2025; Wynn et al., 2025; Cui et al., 2025; Wu et al., 2025b); ii) overthinking collapse, where unnecessary deliberation flips correct answers and wastes tokens on easy instances (Wu et al., 2025a; Zhang et al., 2025; Yang et al., 2025); and iii) protocol rigidity, where a one-size-fits-all interaction strategy fails to match the varying difficulty and structure of different problems (Eo et al., 2025; Kaesberg et al., 2025; Choi et al., 2025; Zhang et al., 2026b,c; Li et al., 2024a). These observations suggest that the core challenge in MAD is no longer simply how to make agents debate, but how to orchestrate debate adaptively.

Recent work has taken important steps toward more adaptive debate, including confidence-triggered invocation (Eo et al., 2025), dynamic role assignment (Zhang et al., 2026b; Fan et al., 2025; Zhang et al., 2026c), budget-aware structuring (Yang et al., 2026a), sparse or structured communication (Zeng et al., 2025; Sun et al., 2025; Li et al., 2024b; Liu et al., 2024), and diversity-oriented debate designs (Liu et al., 2025; Li et al., 2026; Cui et al., 2025; Wu et al., 2025b). However, these approaches typically adapt only one dimension of coordination while leaving the overall debate protocol largely fixed. In parallel, work on evolving skills in LLM agent systems has shown that agent capabilities can be externalized, reused, and refined over time (Yang et al., 2026b; Pei et al., 2025; Qiu et al., 2026; Zhang et al., 2026a), but

these ideas have not been brought into debate coordination itself. However, two key gaps remain: existing MAD frameworks lack multi-axis, state-dependent control over how debate unfolds, and they do not let the coordination policy itself improve online from debate experience.

To address these gaps, we introduce **OrchDebate**, a framework that reconceptualizes multi-agent debate as structured-state orchestration over a bank of reusable debate-control skills. Instead of committing to a fixed protocol before deliberation begins, OrchDebate treats debate as a sequential coordination problem. At each step, an orchestrator examines a compact control-facing Structured Debate State (SDS) and selects the next debate-control skill. The action space is expressed through a skill bank of reusable debate-control skills, such as spawning a verifier, critic, or diversifier under different visibility policies, or terminating debate with an adaptive readout rule. This turns debate control from a monolithic protocol into a modular, compositional, and state-aware policy. Moreover, the orchestrator is functionally distinct from the reasoning agents and operates through a structured control interface rather than directly generating reasoning content. This design gives the orchestrator a more independent view of debate flow, which is designed to mitigate reliance on early answers.

OrchDebate further maintains a self-evolving skill bank that improves coordination strategies online from its own debate trajectories, allowing the orchestration policy to become progressively stronger over time without requiring additional external data. In our main experiments, we compare OrchDebate against competitive prompting and representative debate baselines (Liang et al., 2024; Zeng et al., 2025; Li et al., 2024b; Liu et al., 2025; Li et al., 2026; Liu et al., 2024) on six diverse benchmarks spanning scientific reasoning, logical reasoning, broad knowledge, mathematics, and multi-step reasoning (Rein et al., 2024; Liu et al., 2023; Hendrycks et al., 2021a; Cobbe et al., 2021; Hendrycks et al., 2021b; Sprague et al., 2024). Across these benchmarks, OrchDebate achieves the best or tied-best accuracy on all six tasks under GPT-5-nano and on five of the six tasks under both Claude-Haiku-4.5 and Qwen-3-8B, while remaining more token-efficient than fixed debate protocols.

Taken together, our contributions are:

- We formulate multi-agent debate as structured-state orchestration over reusable debate-control skills, replacing fixed debate protocols with instance-adaptive, multi-axis coordination.
- We propose an adaptive orchestrator, functionally distinct from reasoning agents, that selects debate-control skills based on the Structured Debate State and includes mechanisms designed to limit reliance on early intermediate answers.
- We introduce a self-evolving skill bank that improves debate coordination online from the system's own debate trajectories, without requiring additional external data.

2 Related Work

2.1 Multi-Agent Debate

MAD studies how multiple LLM agents can deliberate, critique, and aggregate reasoning to improve final-answer quality. Early work establishes the basic debate paradigm and its variants (Du et al., 2024; Liang et al., 2024; Xiong et al., 2023; Xu et al., 2023), while later studies extend it to cross-model communication and consensus-oriented interaction (Yin et al., 2023; Chen et al., 2024). More recent work improves debate efficiency or robustness through sparse communication, structured interaction, budget-aware design, and diversity-oriented deliberation (Zeng et al., 2025; Li et al., 2024b; Liu et al., 2024; Yang et al., 2026a; Liu et al., 2025; Li et al., 2026; Cui et al., 2025). At the same time, several analyses show that debate gains depend heavily on when debate is invoked, how agents are coordinated, and how final answers are selected, and they further identify anchoring, brittle self-correction, and overthinking as recurring failure modes (Choi et al., 2025; Kaesberg et al., 2025; Zhang et al., 2026c; Li et al., 2024a; Weng et al., 2025; Wynn et al., 2025; Wu et al., 2025a; Yang et al., 2025; Zhang et al., 2025; Wu et al., 2025b). In contrast to prior MAD variants that mainly redesign a fixed interaction template, OrchDebate treats debate as structured-state orchestration over reusable debate-control skills.

2.2 Adaptive Coordination and Evolving Skills

Recent work on LLM agents shows that useful behaviors can be externalized as skills and improved over time through refinement, memory, and self-improvement mechanisms (Yang et al., 2026b; Qiu et al., 2026; Zhang et al., 2026a; Pei et al., 2025). This line of research suggests that agent capabili-

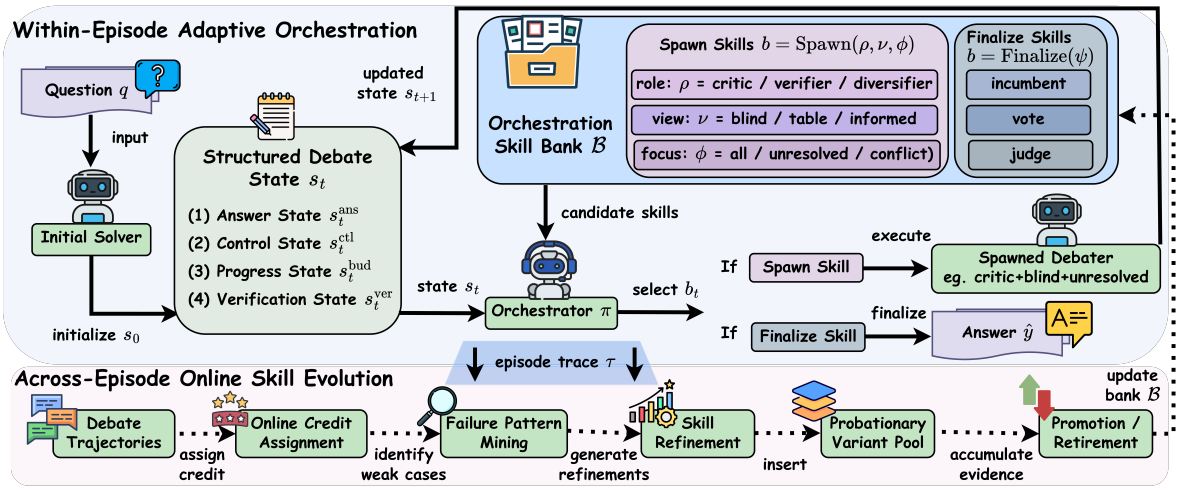

Figure 1: Overall framework of OrchDebate. Within each episode, the orchestrator reads SDS, selects the next debate-control skill from the skill bank, executes the resulting spawn or finalize action, and updates SDS. Across episodes, debate trajectories provide feedback for online skill evolution, which refines the skill bank over time.

ties need not be encoded solely in static prompts or monolithic policies, but can instead be represented, selected, and updated as modular skills. However, these ideas have rarely been brought into debate coordination itself. This gap motivates our view of debate as a state-conditioned orchestration problem over reusable control skills. Meanwhile, existing debate methods still largely rely on fixed protocols even when they introduce local forms of adaptivity, such as selective triggering, dynamic role assignment, or context-aware coordination (Eo et al., 2025; Zhang et al., 2026b; Fan et al., 2025; Sun et al., 2025; Zhou et al., 2025; Wang et al., 2026). OrchDebate bridges these directions by representing debate control as reusable skills and allowing the orchestration policy itself to improve online from its own debate experience over time.

3 Methodology

OrchDebate reframes multi-agent debate as adaptive orchestration rather than a fixed protocol. The key idea is to maintain a structured SDS, use it to decide the next debate-control skill, and continuously improve the self-evolving skill bank online. Figure 1 provides an overview of the OrchDebate framework. We first introduce SDS in §3.1, then describe adaptive debate control in §3.2, next present online skill evolution in §3.3, and finally summarize the overall OrchDebate procedure in §3.4.

3.1 Structured Debate State

State abstraction. To decide what the orchestrator should observe, OrchDebate maintains a Structured Debate State (SDS), a compact control-facing summary that avoids conditioning on full multi-agent transcripts and retains only the signals needed for routing, verification, escalation, and stopping decisions.

Formally, at debate step t, we represent the state as

$$s_t = \left(s_t^{\text{ans}},\ s_t^{\text{ver}},\ s_t^{\text{ctl}},\ s_t^{\text{bud}}\right), \tag{1}$$

where the four components summarize the answer landscape, verification progress, control-side risks, and resource budget, respectively. Concretely, s_t^{ans} records signals such as the current leading answer, agreement level, answer margin, answer changes, and stability; s_t^{ver} tracks unresolved issues, verification coverage, and verification debt; s_t^{ctl} captures orchestration cues such as anchoring risk, response diversity, and stopping or escalation hints; and s_t^{bud} records the current debate step and remaining budget, with $\beta_t = B - C_t$ denoting the remaining token budget.

Leading answer and consensus dynamics. Let $\mathcal{A}_t = \{a_t^{(1)}, \ldots, a_t^{(n_t)}\}$ denote the set of candidate answers currently produced by active reasoning agents. We define the leading answer by majority support:

$$a_t = \arg\max_{a} \sum_{i=1}^{n_t} \mathbf{1}[a_t^{(i)} = a], \tag{2}$$

and track its dynamics through

$$\delta_t = \mathbf{1}[a_t \neq a_{t-1}]. \tag{3}$$

Together with entropy, margin, and stability, these quantities tell the orchestrator how strong and how settled the current leader is.

3.2 Adaptive Debate Control

Debate-control skills. Instead of committing to a fixed debate template, OrchDebate operates over a reusable bank of debate-control skills, $\mathcal{B} = \{b_1, \ldots, b_K\}$. A skill is a structured control primitive that specifies both what debate operation to perform and how it should be realized:

$$b = (\mathrm{op}, r, \nu, \phi, \psi), \tag{4}$$

where $\mathrm{op} \in \{\mathrm{spawn}, \mathrm{finalize}\}$ is the operation type, r is the role assigned to the next reasoning action (e.g., critic, verifier, diversifier), ν is the visibility policy (e.g., blind, table-only, informed), ϕ specifies the focus scope (e.g., all claims, unresolved points, conflict set, top-k issues), and ψ denotes the readout rule used at termination. For spawn skills, OrchDebate uses (r, ν, ϕ) to determine how the next reasoning step is constructed; for finalize skills, it uses ψ to determine how the final answer is read out from the current debate state.

Selecting the next control skill. Given the current SDS, the orchestrator adaptively determines how debate should proceed by selecting the next legal skill from the bank. Let $\mathcal{L}(s_t) \subseteq \mathcal{B}$ denote the set of legal skills under state s_t. The control decision is implemented as

$$\tilde{b}_t = \pi(s_t), \quad b_t = \mathrm{Resolve}(\tilde{b}_t, \mathcal{L}_t). \tag{5}$$

Here, π is an LLM-based controller conditioned on a structured textual rendering of SDS, $\tilde{b}_t$ is the raw controller proposal, and $\mathcal{L}_t = \mathcal{L}(s_t)$ denotes the legal skill set under state s_t. The resolver maps the proposal to a legal skill or triggers a deterministic fallback when the proposal is malformed or illegal. The orchestrator does not decide the answer directly; it decides the next debate-control skill, which governs whether debate should continue, diversify, verify, or terminate.

Legal actions and fallback. Not every skill is legal at every step. In particular, finalization is allowed only after a minimum number of reasoning steps:

$$\mathcal{L}(s_t) = \mathcal{B}_{\mathrm{spawn}} \cup \{\, b \in \mathcal{B}_{\mathrm{final}} : t \geq t_{\min} \,\}. \tag{6}$$

Because the orchestrator is LLM-based, its output may occasionally be malformed or map to an illegal skill. OrchDebate therefore uses a deterministic fallback resolver π_{rule} whenever the proposed action cannot be matched to a valid skill in $\mathcal{L}(s_t)$.

Anchoring-aware control. An especially important case is blind spawning, which hides the current leading answer from the next agent and encourages an independent challenge rather than inheritance of the incumbent. This is particularly useful when the current leader is weak, unstable, or insufficiently verified, since the newly spawned agent can examine the problem without inheriting the incumbent answer.

3.3 Online Skill Evolution

Skill variants. To avoid relying on a fixed set of initial prompt realizations, OrchDebate allows each control skill to improve online through a black-box evolution loop driven by the system's own debate trajectories, without requiring additional external data. For every base skill b_k, the system maintains one validated incumbent realization $v_k^\star$ and a set of probationary candidates $\mathcal{P}_k$, yielding the variant pool

$$\mathcal{V}_k = \{v_k^\star\} \cup \mathcal{P}_k. \tag{7}$$

These variants share the same semantic control function but may differ in wording, realization details, or execution configuration.

Probationary exploration. When base skill b_k is selected, OrchDebate usually executes the incumbent but occasionally explores a probationary candidate:

$$P(v \mid b_k) = \begin{cases} 1 - p, & v = v_k^\star, \\ \dfrac{p}{|\mathcal{P}_k|}, & v \in \mathcal{P}_k. \end{cases} \tag{8}$$

This bounded exploration strategy limits the risk of weak candidates while still collecting evidence about potentially better realizations. After each episode, OrchDebate assigns lightweight online credit to the executed skills using the final debate outcome relative to the direct-solve baseline, while penalizing unnecessary extra cost and answer degradation. For intermediate debate actions, this episode-level signal is complemented by local improvements in the structured debate state, so skills are rewarded not only for improving final outcomes but also for helping the debate become more verifiable and less incumbent-bound. When a recurring diagnostic pattern persists, a lightweight mutator proposes local edits to the corresponding realization. New variants are inserted into probation and

then evaluated through normal online usage on the evolution stream, rather than through a separate offline prompt-tuning procedure.

Promotion and replacement. Let $\hat{A}(v)$ denote the average online advantage of variant v estimated on the evolution stream, and let $\hat{D}(v)$ denote its degradation rate relative to the direct-solve baseline, with τ_D denoting the maximum allowed degradation rate. A probationary variant is promoted only if it has been evaluated sufficiently often, achieves positive accumulated advantage, and does not introduce excessive degradation:

$$N(v) \geq n_{\min}, \quad \hat{A}(v) \geq \epsilon, \quad \hat{D}(v) \leq \tau_D. \tag{9}$$

Otherwise, the candidate is retired. This yields a two-level adaptation scheme: adaptive debate control determines which skill to use within an episode, while online skill evolution improves how that skill is realized across episodes. Additional evolution details are provided in Appendix B.3.

3.4 Orchestrated Multi-Agent Debate

Initialization. Given a question q, OrchDebate first obtains an initial answer using the reasoning model $\mathcal{M}$ and initializes the SDS from the resulting debate state. This initial state gives the orchestrator a compact view of the current answer landscape, verification status, control-relevant cues, and remaining budget.

Iterative orchestration. At each step, the controller selects the next legal skill, executes the corresponding reasoning action, and updates the SDS accordingly. Formally, if v_t is the selected realization of skill b_t, the episode transition is

$$r_t = \mathrm{Exec}(v_t, q, s_t; \mathcal{M}), \; s_{t+1} = U(s_t, r_t) \tag{10}$$

where r_t denotes the resulting reasoning action or finalization outcome. The system repeats this loop until a legal finalize skill is selected, the step limit is reached, or the budget is exhausted. In this way, debate is not governed by a predetermined template; its trajectory is shaped online by a sequence of state-dependent control decisions.

Finalization. When a legal finalize skill is selected, OrchDebate commits to a final answer using the readout rule associated with that skill. If the step or budget limit is reached first, the system falls back to a readout from the current debate state. Algorithm 1 summarizes the full single-episode procedure.

Algorithm 1: OrchDebate: Orchestrated Multi-Agent Debate for One Episode

Input : Question q, reasoning model $\mathcal{M}$, orchestrator π, skill bank $\mathcal{B}$, maximum steps T, minimum debate steps $t_{\min}$
Output : Final answer $\hat{y}$

```
a_0 ← M(q);
s_0 ← InitSDS(q, a_0);
for t = 0, 1, ..., T − 1 do
    propose b_t ← π(s_t);
    if b_t ∉ L(s_t) then
        b_t ← π_rule(s_t);
    if b_t = Finalize(ψ) then
        if t ≥ t_min then
            return Readout(ψ, s_t);
        b_t ← π_rule(s_t);
    sample variant v_t ~ P(· | b_t);
    execute r_t ← Exec(v_t, q, s_t; M);
    update s_{t+1} ← U(s_t, r_t);
return FallbackReadout(s_T)
```

4 Experiments

4.1 Experimental Setup

Benchmarks. We evaluate on six benchmarks spanning complementary task types: GPQA (Rein et al., 2024) for graduate-level science question answering, LogiQA2.0 (Liu et al., 2023) for logical reasoning, MMLU (Hendrycks et al., 2021a) for broad knowledge and reasoning, GSM8K (Cobbe et al., 2021) for grade-school mathematics, MATH (Hendrycks et al., 2021b) for competition-style mathematical problem solving, and MuSR (Sprague et al., 2024) for multi-step reasoning. These benchmarks cover both settings where multi-agent deliberation is often beneficial and settings where unnecessary interaction can induce overthinking or answer drift. For OrchDebate, each benchmark is further split into an evolution stream and a held-out evaluation set; details are provided in Appendix C.1.

Baselines. We compare OrchDebate against single-agent prompting baselines, including direct chain-of-thought decoding (CoT) (Wei et al., 2022) and self-consistency (CoT-SC) (Wang et al., 2023), and representative multi-agent debate baselines, including MAD (Liang et al., 2024), S-MAD (Li et al., 2024b), S^2-MAD (Zeng et al., 2025), GroupDebate (Liu et al., 2024), DMAD (Liu et al., 2025), and DynaDebate (Li et al., 2026). We further study OrchDebate through internal ablations in Section 4.3, including w/o Orchestrator, w/o SDS, w/o Skills Bank, and w/o Self-Evolving. Detailed implementation settings of all compared baselines are

Model	Method	GPQA		LogiQA2.0		MMLU		GSM8K		MATH		MuSR	
		Acc(%)	Tok(k)	Acc(%)	Tok(k)	Acc(%)	Tok(k)	Acc(%)	Tok(k)	Acc(%)	Tok(k)	Acc(%)	Tok(k)
GPT-5-nano	CoT	66.9	8.8	72.2	4.4	84.5	3.5	94.5	2.8	79.3	4.4	65.3	6.8
	CoT-SC	65.6	94.1	72.8	46.2	84.5	37.4	94.8	28.6	79.5	44.0	65.7	75.2
	MAD	63.1	18.1	69.5	9.6	80.2	8.1	89.2	6.1	72.5	9.5	57.7	16.0
	S-MAD	66.9	10.9	73.2	9.7	84.2	11.0	95.5	12.6	72.2	11.6	68.3	21.7
	S^2-MAD	65.0	31.3	**76.2**	17.0	84.5	14.0	94.5	9.5	80.7	14.3	69.0	27.2
	DMAD	67.5	23.0	74.5	14.4	85.5	12.2	89.8	10.2	77.5	13.9	66.7	24.4
	GroupDebate	66.9	38.9	74.5	22.1	86.7	18.4	93.8	12.7	79.8	19.9	65.0	40.3
	DynaDebate	59.4	17.1	70.5	11.0	81.8	9.3	94.3	11.6	78.5	11.0	69.0	17.2
	OrchDebate	**71.2**	9.6	**76.2**	8.8	**87.3**	8.1	**96.8**	5.9	**81.6**	8.3	**73.3**	12.6
Claude-Haiku-4.5	CoT	56.2	2.2	77.8	1.4	86.2	1.1	95.5	0.9	64.2	1.7	68.7	3.4
	CoT-SC	60.0	25.1	81.3	17.5	87.8	14.3	96.8	10.4	70.5	18.6	68.0	39.7
	MAD	58.8	9.9	78.2	4.1	84.5	3.4	92.5	2.7	65.2	4.4	63.7	10.8
	S-MAD	66.2	37.5	80.5	26.1	80.8	21.6	96.2	9.7	67.7	17.4	70.0	48.1
	S^2-MAD	57.5	11.6	79.5	11.0	**88.8**	11.9	95.8	6.5	76.8	12.3	68.3	13.7
	DMAD	56.9	15.1	80.5	9.7	86.5	12.9	90.2	9.6	69.8	11.2	73.7	18.9
	GroupDebate	63.1	21.3	82.8	15.0	85.5	13.3	96.8	10.0	75.5	17.2	75.0	29.2
	DynaDebate	66.2	12.2	79.2	12.3	86.5	10.5	94.7	8.0	75.2	13.1	74.3	14.6
	OrchDebate	**69.4**	7.8	**83.3**	8.1	87.4	6.4	**97.2**	4.9	**78.6**	9.5	**75.7**	10.2
Qwen-3-8B	CoT	53.8	9.9	79.2	4.2	77.8	3.0	96.3	1.8	77.5	5.1	65.7	5.6
	CoT-SC	53.8	98.9	81.3	38.6	79.2	26.4	**97.0**	18.7	79.3	56.7	70.7	54.2
	MAD	52.5	27.6	78.7	11.8	80.2	9.0	95.5	6.7	79.3	18.7	65.3	16.9
	S-MAD	58.8	48.1	80.5	21.1	79.0	14.6	95.5	9.8	70.8	16.1	69.7	27.0
	S^2-MAD	56.2	41.9	82.2	14.3	79.8	10.4	96.0	11.7	80.8	25.9	65.0	18.9
	DMAD	54.4	45.2	79.8	22.6	80.2	17.9	95.5	14.6	79.8	28.8	71.3	28.4
	GroupDebate	57.5	60.0	82.3	23.6	79.2	17.6	96.5	14.7	81.8	40.2	69.0	33.2
	DynaDebate	54.4	18.0	79.3	10.4	78.3	13.2	95.8	9.0	80.0	11.7	68.3	13.3
	OrchDebate	**59.4**	11.1	**84.0**	7.5	**81.3**	6.1	96.5	7.4	**83.5**	9.7	**73.0**	9.4

Table 1: **Comparison of Token Cost and Accuracy Between OrchDebate and Other Methods.** The highest-accuracy results are shown in **bold**. For each dataset, we underline the result pair for the highest accuracy and the result pair for the lowest token cost, excluding CoT.

provided in Appendix C.3.

Model Settings. Each LLM setting uses a single model family throughout OrchDebate, including orchestration, debate actions, and final judging when invoked. We use a bounded debate budget and enable online skill evolution only on the evolution split; the resulting skill bank is then frozen for all held-out evaluations. All methods are evaluated on the same held-out benchmark splits under a matched pipeline. Full configuration details are provided in Appendix C.

Evaluation Metrics. We evaluate methods along two primary axes: accuracy and token cost. We use task accuracy as the quality metric and the average total number of tokens consumed per instance as the cost metric. Unless otherwise stated, token counts include the full interaction cost under a unified accounting protocol, including debate turns, orchestration decisions, and judging steps when used.

4.2 Main Results

Table 1 reports results on six benchmarks under three LLM settings. Overall, OrchDebate achieves the strongest accuracy–efficiency trade-off among debate methods. It reaches the best or tied-best accuracy on all six benchmarks under GPT-5-nano, and the best accuracy on five of six benchmarks under both Claude-Haiku-4.5 and Qwen-3-8B. The gains are consistent on GPQA, MATH, and MuSR, where OrchDebate outperforms the strongest competing debate baseline under all three LLM settings. Exceptions are MMLU under Claude-Haiku-4.5, where S^2-MAD remains slightly stronger (88.8 vs. 87.4), and GSM8K under Qwen-3-8B, where CoT-SC obtains the highest accuracy while OrchDebate ties the strongest debate baseline.

These gains do not simply reflect higher debate cost. Under GPT-5-nano, OrchDebate reaches the best or tied-best accuracy on all six benchmarks while using only 5.9k to 12.6k tokens, whereas strong fixed-topology baselines such as S^2-MAD, DMAD, and GroupDebate typically require larger budgets. Similar patterns under Claude-Haiku-4.5 and Qwen-3-8B suggest that adaptive orchestration allocates debate computation more selectively.

OrchDebate also improves over direct CoT, particularly on harder benchmarks. Under GPT-5-nano, it improves over CoT by +4.3 on GPQA, +2.3 on MATH, and +8.0 on MuSR; the corresponding gains are +13.2, +14.4, and +7.0 under Claude-Haiku-4.5, and +5.6, +6.0, and +7.3 under Qwen-3-8B. On relatively easier settings such as GSM8K, the margin is smaller, and CoT-SC can

Method	GPT-5-nano		Claude-Haiku-4.5		Qwen-3-8B	
	Acc(%)	Tok(k)	Acc(%)	Tok(k)	Acc(%)	Tok(k)
MAD	63.1	18.1	58.8	9.9	52.5	27.6
OrchDebate	**71.2**	**9.6 (-47.0%)**	**69.4**	**7.8 (-21.2%)**	**59.4**	**11.1 (-59.8%)**
OrchDebate w/o Orchestrator	62.5	11.9 (-34.3%)	61.9	12.2 (+23.2%)	51.9	14.8 (-46.4%)
OrchDebate w/o SDS	67.5	16.5 (-8.8%)	60.6	14.3 (+44.4%)	52.5	18.9 (-31.5%)
OrchDebate w/o Skills Bank	66.2	13.2 (-27.1%)	62.5	9.8 (-1.0%)	54.4	16.3 (-40.9%)
OrchDebate w/o Self-Evolving	66.9	11.5 (-36.5%)	65.6	8.5 (-14.1%)	56.9	11.9 (-56.9%)

Table 2: Module ablation on GPQA across three LLM settings. We report accuracy and token cost, with token change in parentheses computed relative to MAD under the same model.

still achieve the best absolute result at much higher cost. Overall, these results show that OrchDebate improves answer quality while avoiding the unnecessary computation often incurred by static multi-agent protocols. Additional supplementary results for GPT-4o-mini are reported in Appendix D.

4.3 Ablation Study

To assess the contribution of each component in OrchDebate, we conduct module ablations on GPQA under three LLM settings, as shown in Table 2. Additional ablation results on MuSR and MMLU are provided in Appendix D.2. We consider four variants: w/o Orchestrator replaces adaptive orchestration with a fixed debate protocol with early stopping; w/o SDS keeps only a solver-confidence signal; w/o Skills Bank removes skill-based actions; and w/o Self-Evolving disables online skill evolution.

Full OrchDebate performs best across all three LLMs, reaching 71.2%, 69.4%, and 59.4% under GPT-5-nano, Claude-Haiku-4.5, and Qwen-3-8B, respectively, while reducing token usage by 47.0%, 21.2%, and 59.8% relative to MAD. Removing either adaptive orchestration or SDS leads to clear performance drops, showing that effective debate control depends on both adaptive decisions and informative state signals. Removing the skills bank also hurts performance, suggesting that reusable debate-control skills provide more than simple continuation control. Finally, disabling self-evolution reduces accuracy from 71.2% to 66.9%, from 69.4% to 65.6%, and from 59.4% to 56.9%, showing that online evolution further improves the initial skill set.

4.4 Orchestration Behavior Analysis

To examine how OrchDebate adapts its control behavior, we analyze the distribution of controller

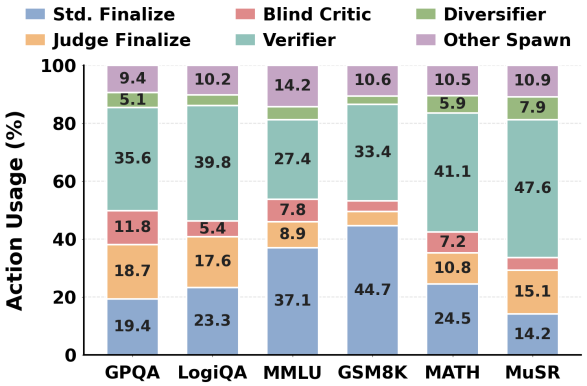

Figure 2: Distribution of OrchDebate controller actions across benchmarks. Each bar reports the percentage of decisions assigned to one of six action categories, summing to 100%.

actions across benchmarks. For analysis, controller decisions are grouped into six categories: Standard Finalize, Judge Finalize, Blind Critic, Verifier, Diversifier, and Other Spawn. All results discussed in this subsection are based on GPT-5-nano. Figure 2 shows a clear relationship between task difficulty and the orchestrator's action profile. On relatively easier benchmarks where the base solver is already strong, such as GSM8K and MMLU, the controller relies more on Standard Finalize, accounting for 44.7% and 37.1% of decisions, respectively, which suggests that OrchDebate avoids extending deliberation when evidence stabilizes early.

On harder benchmarks, the controller shifts more decisions toward continued scrutiny. Verifier becomes the dominant action on GPQA, LogiQA2.0, MATH, and MuSR, accounting for 35.6% to 47.6% of decisions, while Judge Finalize also rises to 15%–19% on GPQA, LogiQA2.0, and MuSR, indicating a greater reliance on arbitrated readout when evidence remains contested. Anchoring-aware actions are used more selectively: Blind Critic peaks

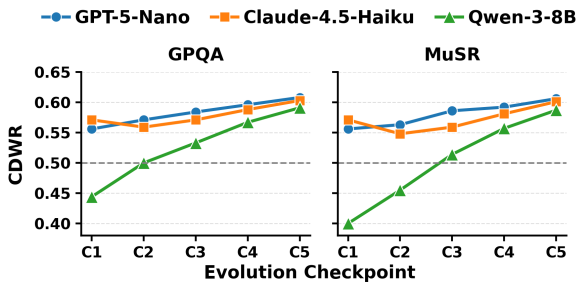

Figure 3: CDWR of OrchDebate against a fixed initial skill bank on GPQA and MuSR across five evolution checkpoints. The dashed line at 0.5 denotes parity.

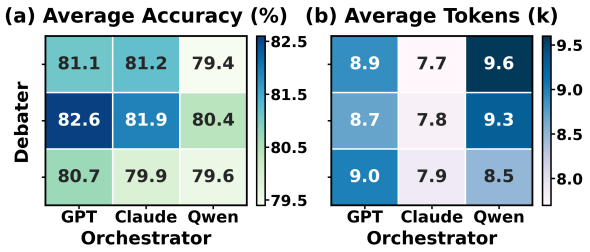

Figure 4: Cross-model OrchDebate results averaged over six benchmarks. Rows denote debaters and columns denote orchestrators. Left: accuracy (%); right: token cost (k).

at 11.8% on GPQA, and Diversifier is most visible on MuSR (7.9%) and MATH (5.9%). Taken together, these patterns suggest that OrchDebate does not uniformly prolong debate across tasks, but instead adapts its coordination strategy to the difficulty and disagreement structure of each benchmark.

4.5 Impact of Skill Evolution

As shown in Section 4.3, removing self-evolution hurts final performance. We next examine how this gain develops over time. Figure 3 compares the self-evolving skill bank with a fixed-bank counterpart on two representative hard benchmarks, GPQA and MuSR, under three LLM settings. We report cumulative disagreement win rate (CDWR) over five evolution checkpoints on the evolution stream, denoted by C_1–C_5, where the dashed line at 0.5 marks parity on the subset of instances where the two variants disagree.

The overall trend is consistently positive. By C_5, all six curves lie above parity, indicating that evolution more often helps than hurts once it changes the final outcome. The gains also accumulate over time rather than coming from isolated updates. This pattern is especially clear for Qwen-3-8B, whose CDWR rises from 44.4% to 59.1% on GPQA and from 40.0% to 58.7% on MuSR, while GPT-5-nano shows a nearly monotonic improvement on both benchmarks. Claude-Haiku-4.5 exhibits a small early dip, but recovers steadily and ends above 0.6 on both tasks. Since the evolving and fixed-bank variants often make the same final prediction, raw accuracy alone can understate the effect of bank updates. CDWR isolates the cases actually changed by evolution and shows that these changes become increasingly favorable as updates accumulate.

4.6 Orchestrator–Debater Cross-Model Study

Finally, we examine whether OrchDebate's gains come only from stronger debaters, or also from better orchestration. Figure 4 reports average accuracy and token cost across the six benchmarks under different orchestrator–debater pairs.

Cross-model composition remains effective: the orchestrator and debater need not come from the same model family. However, the choice of orchestrator clearly matters. Averaged over all debaters, GPT-5-nano gives the highest accuracy as orchestrator (81.5%), while Claude-Haiku-4.5 is the most efficient, reaching comparable accuracy (81.0%) with the lowest token cost (7.8k). Qwen-3-8B is weaker in this role, with lower accuracy (79.8%) and higher cost (9.1k). The debater also matters: Claude-Haiku-4.5 is the strongest debater on average (81.6%), and the best pair uses Claude-Haiku-4.5 as debater with GPT-5-nano as orchestrator, reaching 82.6%. Overall, OrchDebate benefits from both capable debaters and effective state-conditioned orchestration.

5 Conclusion

We presented **OrchDebate**, a framework that reframes multi-agent debate as state-conditioned orchestration over reusable debate-control skills rather than a fixed interaction protocol. By combining a structured debate state, adaptive skill routing, and a self-evolving skill bank, OrchDebate allocates debate computation more selectively and improves coordination from its own debate experience. Across six reasoning benchmarks and three LLM families, it consistently achieves a stronger trade-off between accuracy and efficiency than fixed-protocol debate baselines. These results suggest that future MAD systems should focus on orchestrating debate online rather than scaling static interaction structures.

6 Limitations

Although OrchDebate is designed to improve the trade-off between accuracy and efficiency in debate-based reasoning, it still incurs additional inference overhead relative to single-pass prompting, since solving a query may require multiple rounds of orchestration and agent interaction. This added cost may limit its applicability in highly latency-sensitive scenarios. In addition, while our experiments cover six representative benchmarks, they remain focused on structured reasoning tasks with relatively well-defined answers. The extent to which the same orchestration mechanism transfers to broader settings, such as open-ended generation, tool-using agents, or other long-horizon tasks, remains to be established in future work.

References

Jonas Becker, Lars Benedikt Kaesberg, Niklas Bauer, Jan Philip Wahle, Terry Ruas, and Bela Gipp. 2025. Mallm: Multi-agent large language models framework. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 418–439, Suzhou, China. Association for Computational Linguistics.

Justin Chen, Swarnadeep Saha, and Mohit Bansal. 2024. Reconcile: Round-table conference improves reasoning via consensus among diverse llms. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7066–7085, Bangkok, Thailand. Association for Computational Linguistics.

Hyeong Kyu Choi, Jerry Zhu, and Sharon Li. 2025. Debate or vote: Which yields better decisions in multi-agent large language models? In *Advances in Neural Information Processing Systems*. NeurIPS 2025 Spotlight.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *Preprint*, arXiv:2110.14168.

Yu Cui, Hang Fu, Haibin Zhang, Licheng Wang, and Cong Zuo. 2025. Free-mad: Consensus-free multi-agent debate. *Preprint*, arXiv:2509.11035. ICLR 2026 withdrawn submission.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B. Tenenbaum, and Igor Mordatch. 2024. Improving factuality and reasoning in language models through multiagent debate. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 11733–11763. PMLR.

Sugyeong Eo, Hyeonseok Moon, Evelyn Hayoon Zi, Chanjun Park, and Heuiseok Lim. 2025. Debate only when necessary: Adaptive multiagent collaboration for efficient llm reasoning. *Preprint*, arXiv:2504.05047.

Wei Fan, JinYi Yoon, and Bo Ji. 2025. imad: Intelligent multi-agent debate for efficient and accurate llm inference. *Preprint*, arXiv:2511.11306.

Taicheng Guo, Xiuying Chen, Yaqi Wang, Ruidi Chang, Shichao Pei, Nitesh V. Chawla, Olaf Wiest, and Xiangliang Zhang. 2024. Large Language Model based multi-agents: A Survey of Progress and Challenges. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, pages 8048–8057. Survey Track.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021a. Measuring massive multitask language understanding. In *International Conference on Learning Representations (ICLR)*.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021b. Measuring mathematical problem solving with the MATH dataset. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1.

Lars Benedikt Kaesberg, Jonas Becker, Jan Philip Wahle, Terry Ruas, and Bela Gipp. 2025. Voting or consensus? decision-making in multi-agent debate. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 11640–11671, Vienna, Austria. Association for Computational Linguistics.

Junyou Li, Qin Zhang, Yangbin Yu, Qiang Fu, and Deheng Ye. 2024a. More agents is all you need. *Transactions on Machine Learning Research*. Accepted by TMLR.

Yunxuan Li, Yibing Du, Jiageng Zhang, Le Hou, Peter Grabowski, Yeqing Li, and Eugene Ie. 2024b. Improving multi-agent debate with sparse communication topology. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 7281–7294, Miami, Florida, USA. Association for Computational Linguistics.

Zhenghao Li, Zhi Zheng, Wei Chen, Jielun Zhao, Yong Chen, Tong Xu, and Enhong Chen. 2026. Dynadebate: Breaking homogeneity in multi-agent debate with dynamic path generation. *Preprint*, arXiv:2601.05746.

Tian Liang, Zhiwei He, Wenxiang Jiao, Xing Wang, Yan Wang, Rui Wang, Yujiu Yang, Shuming Shi, and Zhaopeng Tu. 2024. Encouraging divergent thinking in large language models through multi-agent debate. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 17889–17904, Miami, Florida, USA. Association for Computational Linguistics.

Hanmeng Liu, Jian Liu, Leyang Cui, Zhiyang Teng, Nan Duan, Ming Zhou, and Yue Zhang. 2023. Logiqa 2.0 — an improved dataset for logical reasoning in natural language understanding. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 31:2947–2962.

Tongxuan Liu, Xingyu Wang, Weizhe Huang, Wenjiang Xu, Yuting Zeng, Lei Jiang, Hailong Yang, and Jing Li. 2024. Groupdebate: Enhancing the efficiency of multi-agent debate using group discussion. *Preprint*, arXiv:2409.14051.

Yexiang Liu, Jie Cao, Zekun Li, Ran He, and Tieniu Tan. 2025. Breaking mental set to improve reasoning through diverse multi-agent debate. In *The Thirteenth International Conference on Learning Representations*, Singapore. ICLR 2025 Poster.

Junyu Luo, Weizhi Zhang, Ye Yuan, Yusheng Zhao, Junwei Yang, Yiyang Gu, Bohan Wu, Binqi Chen, Ziyue Qiao, Qingqing Long, Rongcheng Tu, Xiao Luo, Wei Ju, Zhiping Xiao, Yifan Wang, Meng Xiao, Chenwu Liu, Jingyang Yuan, Shichang Zhang, and 7 others. 2025. Large Language Model Agent: A Survey on Methodology, Applications and Challenges. *Preprint*, arXiv:2503.21460.

Zehua Pei, Hui-Ling Zhen, Shixiong Kai, Sinno Jialin Pan, Yunhe Wang, Mingxuan Yuan, and Bei Yu. 2025. Scope: Prompt evolution for enhancing agent effectiveness. *Preprint*, arXiv:2512.15374.

Libin Qiu, Zhirong Gao, Junfu Chen, Yuhang Ye, Weizhi Huang, Xiaobo Xue, Wenkai Qiu, and Shuo Tang. 2026. Autorefine: From trajectories to reusable expertise for continual llm agent refinement. *Preprint*, arXiv:2601.22758.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. 2024. Gpqa: A graduate-level google-proof q&a benchmark. In *Proceedings of the First Conference on Language Modeling (COLM)*, Philadelphia, Pennsylvania, United States.

Zayne Rea Sprague, Xi Ye, Kaj Bostrom, Swarat Chaudhuri, and Greg Durrett. 2024. Musr: Testing the limits of chain-of-thought with multistep soft reasoning. In *The Twelfth International Conference on Learning Representations (ICLR)*. Spotlight.

Yiliu Sun, Zicheng Zhao, Sheng Wan, and Chen Gong. 2025. Cortexdebate: Debating sparsely and equally for multi-agent debate. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 9503–9523, Vienna, Austria. Association for Computational Linguistics.

Arne Tillmann. 2025. Literature review of multi-agent debate for problem-solving. *Preprint*, arXiv:2506.00066.

Khanh-Tung Tran, Dung Dao, Minh-Duong Nguyen, Quoc-Viet Pham, Barry O'Sullivan, and Hoang D. Nguyen. 2025. Multi-Agent Collaboration Mechanisms: A Survey of LLMs. *Preprint*, arXiv:2501.06322.

Chao Wang, Han Lin, Huaze Tang, Huijing Lin, and Wenbo Ding. 2026. Rumad: Reinforcement-unifying multi-agent debate. *Preprint*, arXiv:2602.23864.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*, Kigali, Rwanda. ICLR 2023 Poster.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837. NeurIPS 2022 Main Conference Track.

Zhiyuan Weng, Guikun Chen, and Wenguan Wang. 2025. Do as we do, not as you think: the conformity of large language models. In *International Conference on Learning Representations*. ICLR 2025 Oral.

Haolun Wu, Zhenkun Li, and Lingyao Li. 2025a. Can llm agents really debate? a controlled study of multi-agent debate in logical reasoning. *Preprint*, arXiv:2511.07784.

Jing Wu, Suiyao Chen, Inseok Heo, Alexander Gutfraind, Shengjie Liu, Chen Li, Bharathi Srinivasan, Xian Zhang, and Michael Sharps. 2025b. Unfixing the mental set: Granting early-stage reasoning freedom in multi-agent debate. ICML 2025 Workshop on Multi-Agent Systems in the Era of Foundation Models.

Andrea Wynn, Harsh Satija, and Gillian K. Hadfield. 2025. Talk isn't always cheap: Understanding failure modes in multi-agent debate. *Preprint*, arXiv:2509.05396.

Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, Rui Zheng, Xiaoran Fan, Xiao Wang, Limao Xiong, Yuhao Zhou, Weiran Wang, Changhao Jiang, Yicheng Zou, Xiangyang Liu, and 10 others. 2025. The Rise and Potential of Large Language Model Based Agents: A Survey. *Science China Information Sciences*, 68(2):121101.

Kai Xiong, Xiao Ding, Yixin Cao, Ting Liu, and Bing Qin. 2023. Examining inter-consistency of large language models collaboration: An in-depth analysis via debate. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 7572–7590, Singapore. Association for Computational Linguistics.

Zhenran Xu, Senbao Shi, Baotian Hu, Jindi Yu, Dongfang Li, Min Zhang, and Yuxiang Wu. 2023. Towards reasoning in large language models via multi-agent peer review collaboration. *Preprint*, arXiv:2311.08152.

Liming Yang, Junyu Luo, Xuanzhe Liu, Yiling Lou, and Zhenpeng Chen. 2026a. BAMAS: Structuring budget-aware multi-agent systems. In *Proceedings of the AAAI Conference on Artificial Intelligence*. Oral paper.

Yongjin Yang, Euiin Yi, Jongwoo Ko, Kimin Lee, Zhijing Jin, and Se-Young Yun. 2025. Revisiting multi-agent debate as test-time scaling: A systematic study of conditional effectiveness. *Preprint*, arXiv:2505.22960.

Yutao Yang, Junsong Li, Qianjun Pan, Bihao Zhan, Yuxuan Cai, Lin Du, Jie Zhou, Kai Chen, Qin Chen, Xin Li, Bo Zhang, and Liang He. 2026b. Autoskill: Experience-driven lifelong learning via skill self-evolution. *Preprint*, arXiv:2603.01145.

Zhangyue Yin, Qiushi Sun, Cheng Chang, Qipeng Guo, Junqi Dai, Xuanjing Huang, and Xipeng Qiu. 2023. Exchange-of-thought: Enhancing large language model capabilities through cross-model communication. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 15135–15153, Singapore. Association for Computational Linguistics.

Yuting Zeng, Weizhe Huang, Lei Jiang, Tongxuan Liu, XiTai Jin, Chen Tianying Tiana, Jing Li, and Xiaohua Xu. 2025. S^2-mad: Breaking the token barrier to enhance multi-agent debate efficiency. 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)*, pages 9393–9408, Albuquerque, New Mexico. Association for Computational Linguistics.

Hangfan Zhang, Zhiyao Cui, Jianhao Chen, Xinrun Wang, Qiaosheng Zhang, Zhen Wang, Dinghao Wu, and Shuyue Hu. 2025. Stop overvaluing multi-agent debate — we must rethink evaluation and embrace model heterogeneity. *Preprint*, arXiv:2502.08788.

Haozhen Zhang, Quanyu Long, Jianzhu Bao, Tao Feng, Weizhi Zhang, Haodong Yue, and Wenya Wang. 2026a. Memskill: Learning and evolving memory skills for self-evolving agents. *Preprint*, arXiv:2602.02474.

Miao Zhang, Junsik Kim, Siyuan Xiang, Jian Gao, and Cheng Cao. 2026b. Dynamic role assignment for multi-agent debate. *Preprint*, arXiv:2601.17152.

Qian Zhang, Jinyi Liu, Yan Zheng, Hebin Liang, and Lanjun Wang. 2026c. Key decision-makers in multi-agent debates: Who holds the power? *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(35):29883–29891.

Zeqi Zhou, Fang Wu, Shayan Talaei, Haokai Zhao, Meixin Cheng, Tinson Xu, Amin Saberi, and Yejin Choi. 2025. When to trust context: Self-reflective debates for context reliability. *Preprint*, arXiv:2506.06020.

A Notation

We summarize the core symbols used throughout the method. q denotes an input question drawn from distribution $\mathcal{D}$. $\mathcal{M}$ denotes the LLM used in OrchDebate. In our main experiments, the same model family is used throughout the framework to instantiate debate agents, the controller π, and the final judge when needed. The controller π selects debate-control skills from the skill bank $\mathcal{B}$ based on a structured rendering of the current Structured Debate State (SDS). At step t, the debate is summarized by the Structured Debate State (SDS)

$$s_t = \left(s_t^{\text{ans}},\, s_t^{\text{ver}},\, s_t^{\text{ctl}},\, s_t^{\text{bud}}\right),$$

where s_t^{ans} summarizes the current answer landscape, s_t^{ver} summarizes the current verification status, s_t^{ctl} collects control-relevant coordination cues, and s_t^{bud} tracks progress and remaining budget. In particular, a_t denotes the current leading answer, δ_t indicates whether the leader changed relative to the previous step, and $\beta_t = B - C_t$ is the remaining budget after cumulative cost C_t. A control skill is denoted by $b_k \in \mathcal{B}$, and a concrete executable realization of that skill is a variant v. The final prediction is $\hat{y}$, the gold answer is y^*, and $C(q)$ is the total inference cost at termination.

B Extended Method Details

B.1 Structured Debate State Details

The main text defines the Structured Debate State (SDS) as a compact control-facing summary:

$$s_t = \left(s_t^{\text{ans}},\, s_t^{\text{ver}},\, s_t^{\text{ctl}},\, s_t^{\text{bud}}\right), \tag{11}$$

where the four components correspond to answer status, verification status, control cues, and resource budget.

In implementation, s_t^{ans} records the current leading answer, agreement strength, answer margin, recent answer shifts, and short-term stability. s_t^{ver} tracks unresolved checks, verification coverage, and accumulated verification debt. s_t^{ctl} contains control-relevant signals such as anchoring risk, response diversity, and stop-or-escalate cues. Finally, s_t^{bud} records the debate step and the remaining budget $\beta_t = B - C_t$.

SDS is realized as a compact state dictionary grouped around these four components. The controller operates on a structured rendering of SDS rather than on the full debate transcript. This keeps orchestration cheaper and more stable while preserving the signals needed for routing, verification, escalation, and stopping.

B.2 Debate-Control Skills

As in the main text, OrchDebate operates over a bank of debate-control skills

$$b = (\mathrm{op}, r, \nu, \phi, \psi), \tag{12}$$

where $\mathrm{op} \in \{\mathrm{spawn}, \mathrm{finalize}\}$ is the operation type, r is the assigned reasoning role, ν is the visibility policy, ϕ is the focus scope, and ψ is the readout rule at termination.

Spawn skills determine how the next reasoning action is constructed, such as whether the next agent acts as a critic, verifier, or diversifier, and whether it is blind, table-only, or informed with respect to the current debate state. Finalize skills terminate the episode through a specified readout rule, such as incumbent, vote-based, or judge-based readout. This factorization allows OrchDebate to adapt along multiple coordination axes without committing to a fixed protocol.

B.3 Variant Sampling and Online Skill Evolution

For each base skill b_k, OrchDebate maintains one incumbent realization $v_k^\star$ and a probation set $\mathcal{P}_k$, yielding the variant pool

$$\mathcal{V}_k = \{v_k^\star\} \cup \mathcal{P}_k. \tag{13}$$

When b_k is selected, the executed realization is sampled using the same bounded-exploration rule as in the main text:

$$P(v \mid b_k) = \begin{cases} 1-p, & v = v_k^\star, \\ \dfrac{p}{|\mathcal{P}_k|}, & v \in \mathcal{P}_k. \end{cases} \tag{14}$$

This limits the exposure of weak candidates while still collecting evidence about potentially stronger realizations. When $\mathcal{P}_k$ is empty, the incumbent realization $v_k^\star$ is selected deterministically.

During the evolution stream, OrchDebate assigns lightweight online credit to executed variants using terminal correctness feedback relative to the direct-solve baseline, SDS-level improvements, and token cost. For an episode e, let z_e be the terminal correctness of OrchDebate, z_e^0 be the correctness of the direct-solve baseline, and c_e be the normalized token cost. For an executed variant v, we define

$$a_e(v) = (z_e - z_e^0) + \alpha \Delta_{\mathrm{SDS}}(e) - \lambda c_e, \tag{15}$$

where $\Delta_{\mathrm{SDS}}(e)$ summarizes diagnostic improvements such as increased verification coverage, fewer unresolved issues, and reduced vote instability. We estimate

$$\begin{aligned} \hat{A}(v) &= \frac{1}{N(v)} \sum_{e: v \in e} a_e(v), \\ \hat{D}(v) &= \frac{1}{N(v)} \sum_{e: v \in e} \mathbf{1}[z_e^0 = 1 \wedge z_e = 0]. \end{aligned} \tag{16}$$

Candidate variants are generated from recurring diagnostic patterns observed in previous evolution episodes, such as degradation relative to direct solve, unresolved verification debt, unstable votes, or excessive cost. After insertion, a candidate starts with $N(v) = 0$, and its $\hat{A}(v)$ and $\hat{D}(v)$ are updated only from subsequent evolution episodes in which it is executed. After the evolution stream, the skill bank is frozen and no further variant promotion or mutation is performed on held-out evaluation examples.

A probationary variant is promoted only if it has been evaluated sufficiently often, achieves positive accumulated advantage, and its degradation rate is bounded by τ_D:

$$N(v) \geq n_{\min}, \quad \hat{A}(v) \geq \epsilon, \quad \hat{D}(v) \leq \tau_D. \tag{17}$$

Otherwise, the candidate is retired. The mutator is restricted to local realization-level edits, such as wording, visibility phrasing, focus emphasis, or execution configuration, while preserving the semantic identity of the underlying control skill.

C Additional Experimental Details

C.1 Evolution Stream and Held-out Evaluation

For each benchmark, we use a 60/40 split between an evolution stream and a held-out evaluation set. The evolution stream is used to update the OrchDebate skill bank, where terminal correctness feedback and debate-internal diagnostics are used for skill credit assignment, variant promotion, and retirement. After this stage, the skill bank is frozen, and all main results are reported on the held-out

Algorithm 2: OrchDebate: Online Skill Evolution

Input : Episode batch $\mathcal{E}$, skill bank $\mathcal{B}$, mutator μ, thresholds $n_{\min}, \epsilon, \tau_D$

foreach *recurring diagnostic pattern g observed in previous evolution episodes* **do**
  identify source skill b_k;
  $\tilde{v} \leftarrow \mu(g, b_k)$;
  add $\tilde{v}$ to probation pool $\mathcal{P}_k$ with $N(\tilde{v}) = 0$;
foreach *skill $b_k \in \mathcal{B}$* **do**
  foreach *probation variant $v \in \mathcal{P}_k$* **do**
    update $N(v)$, $\hat{A}(v)$, and $\hat{D}(v)$ from subsequent evolution episodes where v is executed;
    if $N(v) \geq n_{\min}$ **then**
      if $\hat{A}(v) \geq \epsilon$ ***and*** $\hat{D}(v) \leq \tau_D$ **then**
        promote v to incumbent and retire $v_k^\star$;
      else
        retire v;

evaluation set without further skill updates. Gold answers in the held-out set are used only for offline scoring, and all baselines are evaluated on the same held-out examples under the matched pipeline.

C.2 Model Configuration

In the main experiments, each LLM setting uses a single model family throughout OrchDebate. That is, GPT-5-nano, Claude-Haiku-4.5, and Qwen-3-8B are each evaluated with the same model family used for orchestration, debate actions, and judging when required. For fair comparison, all baselines are run under matched LLM settings whenever their original protocols permit.

In our implementation, all calls are issued through a unified chat-style interface, and structured outputs are enforced through JSON-only prompting rather than native response-format APIs. In the current configuration, solver, orchestrator, and judge calls use deterministic decoding, while debate-agent calls use a higher decoding temperature to encourage more diverse counterarguments. The implementation also maintains separate output caps for solver, debate, orchestration, and judge calls, rather than forcing a single budget across all modules.

C.3 Baseline Implementation Details

All baselines are implemented under the same evaluation pipeline as OrchDebate. For each model family, they use the same underlying LLM, held-out evaluation split, answer extraction procedure, scoring rules, and token accounting protocol as the main experiments. Since these methods are designed around different interaction structures, we do not artificially force them to use the same number of agents or debate rounds. Instead, each baseline is instantiated with a fixed and transparent configuration that preserves its characteristic mechanism, such as sparse communication, dynamic participation, or judge-based aggregation, while keeping the comparison under comparable model access and evaluation rules. For fairness, baseline hyperparameters are selected only on the same evolution split used before OrchDebate evaluation, by comparing a small set of fixed configurations and choosing the configuration with a favorable accuracy–token-cost trade-off. Once selected, all baseline configurations are frozen and applied unchanged to the held-out evaluation set; no baseline is tuned using held-out labels. By reporting both accuracy and token cost, our evaluation compares not only final task performance but also the efficiency trade-offs induced by different debate-control strategies. This protocol gives both OrchDebate and the compared baselines access to pre-evaluation development feedback, while reserving the held-out split exclusively for final reporting.

CoT (Wei et al., 2022). Chain-of-Thought (CoT) uses a single reasoning call that asks the model to reason step by step before producing a final answer. This serves as the standard single-agent reasoning baseline.

CoT-SC (Wang et al., 2023). CoT with self-consistency samples 20 independent Chain-of-Thought responses at temperature 0.7 and returns the majority answer among the sampled outputs. We use a standard self-consistency configuration that improves robustness without introducing a disproportionately large cost relative to the debate baselines.

MAD (Liang et al., 2024). MAD is implemented as a two-agent debate with an affirmative debater, a negative debater, and a moderator. The debate runs for up to 2 rounds. After each round, the moderator determines whether the discussion has reached a sufficiently clear preference; otherwise, the debate continues, and a final judging step is used when needed. This configuration preserves the core adversarial debate structure of MAD while keeping the interaction depth controlled.

S-MAD (Li et al., 2024b). S-MAD uses 6 agents connected by a sparse communication topology

and runs for 5 rounds. At each round, each agent only observes the responses of its graph neighbors rather than the full set of agents. The final answer is determined by majority vote. This setting preserves the sparse-communication design of the method while avoiding the much higher token cost of fully connected large-agent debate.

S^2-MAD (Zeng et al., 2025). S^2-MAD uses 5 agents organized into two groups. The debate lasts for up to 4 rounds, with 2 rounds per stage. Agents first answer independently, then debate within groups, and exchange group summaries across stages. Early stopping is enabled once group-level summaries become sufficiently aligned. This staged configuration retains the core hierarchical discussion pattern of S^2-MAD while bounding total interaction cost.

DMAD (Liu et al., 2025). DMAD runs three fixed reasoning variants in parallel at each round: direct answering, context-aware chain-of-thought reasoning, and decomposition-based chain-of-thought reasoning. A selector model then chooses the best candidate among the three outputs. We use 2 rounds in total. This preserves the diversity-and-selection principle of DMAD without making the method disproportionately expensive.

DynaDebate (Li et al., 2026). DynaDebate first generates 3 distinct reasoning paths and assigns one path to each agent. The agents then debate for up to 2 rounds. When disagreement persists, the method may invoke one verification step between rounds. The verifier can choose code-style symbolic checking, problem-context checking, or no additional checking, and the returned evidence is used in the subsequent round. This configuration captures the dynamic path-and-verification mechanism of DynaDebate while keeping auxiliary verification calls bounded.

GroupDebate (Liu et al., 2024). GroupDebate uses 5 agents split into 2 groups and runs for 4 rounds with 2 intra-group rounds per stage. Agents first debate within groups, after which each group produces a summary. These summaries are then used for inter-group information exchange in the next stage. The final answer is selected by majority vote. This preserves the method's group-structured deliberation pattern while maintaining a controlled token budget across stages.

C.4 Runtime Configuration

OrchDebate is run under a bounded inference budget and a bounded debate horizon. In practice, the controller operates in LLM mode throughout the main runs, while each example is constrained by fixed limits on total inference budget, debate depth, and the number of spawned debate actions. Verification is also restricted to a small set of focused issues rather than an open-ended checklist.

We additionally enforce a mini-debate constraint before nontrivial finalization. Concretely, judge-style or other non-incumbent finalization paths are enabled only after at least one debate action has been executed. Stable-stop decisions are further governed by explicit conditions on answer stability, verification progress, and disagreement signals, which prevents trivial zero-debate finalization and makes stopping behavior reproducible across runs.

C.5 Controller Observation and Action Execution

At inference time, the orchestrator does not consume the full raw transcript. Instead, it operates on the SDS together with a compact rendering of recent debate state. In practice, this view summarizes local signals such as recent vote patterns, unresolved verification items, and short recent-agent traces, while avoiding full transcript conditioning. This design keeps orchestration efficient and stable while preserving the information needed for routing, verification, escalation, and stopping.

The executable action space is discretized in implementation. Spawn actions are parameterized by agent role (critic, verifier, or diversifier), visibility policy (blind, table-only, or informed), and focus scope (all, unresolved-only, conflict-only, or top-k). Finalize actions are parameterized by the readout rule, including incumbent, vote-based, and judge-based variants. These implementation choices are consistent with the multi-axis control view described in the main methodology.

C.6 Issue Construction and Benchmark Processing

For the released configuration, issue construction is intentionally lightweight. Rather than always invoking a separate issue-miner model, the implementation can derive verification issues directly from the solver brief. In this setting, the resulting issue set is designed to preserve both local verification and answer-challenging signals, without

introducing an additional expensive call at every example.

Across benchmarks, the evaluation pipeline normalizes tasks into three interfaces: standard multiple-choice, open-answer math, and entity-choice reasoning. Answer extraction and grading are then made task-specific. In particular, the implementation applies stronger normalization for open-answer math and separately canonicalizes entity-choice predictions against the available candidate options. These rules are shared across direct-solve and debate-style methods so that gains are not driven by benchmark-specific extraction artifacts.

D Extended Results

The appendix additionally reports extended benchmark tables and supplementary analyses that are omitted from the main text for space reasons. In particular, we include additional main-table-style results for extra LLM settings beyond the three primary models in the main text, using the same benchmark suite, evaluation pipeline, and reporting format whenever complete runs are available.

D.1 GPT-4o-mini

Table 3 reports the corresponding results under GPT-4o-mini. Although these results are presented only as supplementary evidence, they provide a useful reference point for cross-model consistency under the same external baselines and benchmark protocol.

Under GPT-4o-mini, OrchDebate attains the highest accuracy on five of the six benchmarks, including GPQA, LogiQA2.0, GSM8K, MATH, and MuSR, while S-MAD remains strongest on MMLU. At the same time, MAD continues to be the lowest-token debate baseline across all six benchmarks. These results are broadly consistent with the main findings in the paper: adaptive orchestration yields a stronger balance between answer accuracy and inference cost than fixed debate topologies, while the strongest static baseline can still vary across tasks.

D.2 Additional Ablation Results on MuSR and MMLU

To examine whether the component effects observed on GPQA also generalize to other task types, we further report module ablations on MuSR and MMLU under the same three LLM settings, as shown in Table 4. The overall trends remain broadly consistent with the main ablation. On MuSR, full OrchDebate performs best across all three LLMs, reaching 73.3%, 75.7%, and 73.0% under GPT-5-nano, Claude-Haiku-4.5, and Qwen-3-8B, while also reducing token usage by 21.3%, 5.6%, and 44.4% relative to MAD. Removing the orchestrator again causes the largest degradation, dropping accuracy to 66.7%, 69.7%, and 66.7%, which suggests that adaptive orchestration remains especially important when problems require sustained multi-step coordination. Removing SDS or the skills bank also consistently hurts performance, indicating that effective control on MuSR still depends on both structured state and reusable debate-control actions. Finally, disabling self-evolution reduces accuracy from 73.3% to 70.3%, from 75.7% to 74.3%, and from 73.0% to 71.0%, showing that the gains from online evolution extend beyond GPQA.

On MMLU, the pattern is similar but the margins are smaller, which is consistent with the fact that broader knowledge-and-reasoning questions often require less extended deliberation than harder reasoning-focused benchmarks. Even so, full OrchDebate still performs best among the ablation variants under all three LLM settings, reaching 87.3%, 87.4%, and 81.3%. Removing the orchestrator again leads to the most substantial drop, reducing accuracy to 84.4%, 84.8%, and 79.2%, while removing SDS or the skills bank also weakens performance in most cases. Disabling self-evolution yields a smaller but still consistent decline, from 87.3% to 86.8%, from 87.4% to 86.9%, and from 81.3% to 81.0%. Compared with MuSR, the cost advantage on MMLU is less pronounced and can even reverse under some settings, but the full system still provides the strongest overall balance between answer quality and inference cost among the ablation variants in this comparison.

Model	Method	GPQA		LogiQA2.0		MMLU		GSM8K		MATH		MuSR	
		Acc(%)	Tok(k)	Acc(%)	Tok(k)	Acc(%)	Tok(k)	Acc(%)	Tok(k)	Acc(%)	Tok(k)	Acc(%)	Tok(k)
GPT-4o-mini	CoT	38.1	0.9	61.8	0.7	75.2	0.6	92.2	0.5	60.5	0.5	62.7	2.7
	CoT-SC	40.6	9.5	61.3	7.9	76.5	6.3	92.8	5.0	62.8	5.3	63.0	28.2
	MAD	41.2	4.4	58.8	2.8	72.7	2.5	91.7	2.3	58.0	3.8	66.3	8.2
	S-MAD	46.9	25.7	64.5	17.1	**80.2**	14.5	94.2	11.6	62.3	11.2	67.0	37.0
	S^2-MAD	41.9	9.1	64.8	8.5	79.5	6.9	94.8	5.8	64.0	12.8	63.7	12.1
	DMAD	40.6	12.6	64.2	8.6	79.5	7.9	93.2	7.0	61.7	12.5	66.3	13.1
	GroupDebate	40.6	15.6	67.2	11.1	78.2	9.4	95.5	13.2	62.3	14.7	66.0	23.0
	DynaDebate	43.1	11.3	63.5	8.8	78.5	8.8	92.7	9.4	63.7	8.9	66.7	13.2
	OrchDebate	**48.1**	7.5	**67.5**	6.8	79.3	5.9	**95.7**	5.1	**65.8**	7.1	**67.3**	11.7

Table 3: **Additional main-table-style results under GPT-4o-mini.** We report the same benchmarks and metrics as in Table 1. The highest-accuracy results are shown in **bold**. For each dataset, we underline the result pair for the highest accuracy and the result pair for the lowest token cost, excluding CoT.

Benchmark	Method	GPT-5-nano		Claude-Haiku-4.5		Qwen-3-8B	
		Acc(%)	Tok(k)	Acc(%)	Tok(k)	Acc(%)	Tok(k)
MuSR	MAD	57.7	16.0	63.7	10.8	65.3	16.9
	OrchDebate	**73.3**	**12.6 (-21.3%)**	**75.7**	**10.2 (-5.6%)**	**73.0**	**9.4 (-44.4%)**
	OrchDebate w/o Orchestrator	66.7	15.6 (-2.5%)	69.7	13.4 (+24.1%)	66.7	14.0 (-17.2%)
	OrchDebate w/o SDS	69.3	16.0 (0.0%)	72.0	14.0 (+29.6%)	68.7	14.8 (-12.4%)
	OrchDebate w/o Skills Bank	68.0	14.1 (-11.9%)	72.3	12.1 (+12.0%)	69.0	12.3 (-27.2%)
	OrchDebate w/o Self-Evolving	70.3	13.6 (-15.0%)	74.3	11.0 (+1.9%)	71.0	10.8 (-36.1%)
MMLU	MAD	80.2	8.1	84.5	3.4	80.2	9.0
	OrchDebate	**87.3**	**8.1 (0.0%)**	**87.4**	**6.4 (+88.2%)**	**81.3**	**6.1 (-32.2%)**
	OrchDebate w/o Orchestrator	84.4	9.5 (+17.3%)	84.8	7.6 (+123.5%)	79.2	8.2 (-8.9%)
	OrchDebate w/o SDS	85.8	11.0 (+35.8%)	85.6	8.4 (+147.1%)	79.9	9.1 (+1.1%)
	OrchDebate w/o Skills Bank	85.1	9.2 (+13.6%)	85.0	6.7 (+97.1%)	80.1	7.8 (-13.3%)
	OrchDebate w/o Self-Evolving	86.8	9.2 (+13.6%)	86.9	6.9 (+102.9%)	81.0	7.4 (-17.8%)

Table 4: Additional module ablation results on MuSR and MMLU across three LLM settings. We report accuracy and token cost, with token change in parentheses computed relative to MAD under the same model.