Compute-Aware Mixture-of-Agents: Verifier-Gated Adaptive Aggregation under a Fixed Token Budget

Shine Gupta [1] **S Akash** [2]

Abstract

Mixture-of-Agents (MoA) aggregates responses from several LLMs in a fixed number of layers, spending the same compute on every query. We extend MoA with a self-consistency verifier that decides per query when to stop aggregating; we call the result Compute-Aware MoA (CA-MoA). On GSM8K with a Llama-3.2-3B + Qwen2.5-3B pool, CA-MoA reaches 85% accuracy—tied with 4-sample self-consistency, $+5$ over MoA-1 at $1.4\times$ MoA-1's tokens. We add a regime-spanning τ-sweep that characterizes the operating frontier and makes the $n{=}20$ ± 15-pt sampling band visible. Two stress tests bound the regime: a homogeneous Llama$\times 3$ pool where MoA-1 saturates at 92%, and a mixed-capability pool where a weak Gemma2-2B corrupts aggregation. Across all three, the verifier signal tracks correctness: $\ell{=}1$ terminations are correct 89–100%, $\ell{=}3$ exhaustions 0–40%. The within-MoA gain moves $-8 \to +13.3 \to +5$ as the pool becomes aggregation-appropriate. We pre-register a $\sim$7B similar-capability follow-up at $n{=}200$.

1. Introduction

Many recent results on hard reasoning benchmarks come from *multi-agent* setups where several LLMs work on the same query (Wang et al., 2024; Wu et al., 2023; Hong et al., 2023). The most carefully studied of these is Mixture-of-Agents (Wang et al., 2024): a query passes through L layers of K aggregating models, and at $L{=}3$, $K{=}4$ MoA matches GPT-4-class scores on AlpacaEval and MT-Bench with open weights only.

The cost is paid uniformly. MoA spends the same $L \cdot K$ calls on a one-step arithmetic question that any single agent would have solved correctly, and on a multi-step word problem that arguably needs the full pipeline. Both extremes matter for agentic deployments where token budgets, memory footprint, and latency are real constraints. In single-model inference this kind of static allocation has been steadily eroded: early exit (Schwartz et al., 2020), speculative decoding (Leviathan et al., 2023), depth-adaptive transformers (Elbayad et al., 2020), and budget-conditioned sampling (Snell et al., 2024) all spend more compute on harder inputs and less on easier ones. The multi-agent analogue, where "depth" is the number of MoA layers rather than transformer blocks, has been left alone.

Existing work touches the boundary but not the interior. Cascades (Chen et al., 2023; Madaan et al., 2023a) retry at full-rollout granularity, so they cannot make use of MoA's intra-layer aggregation. Vanilla Self-Consistency (Wang et al., 2023) varies width with no verifier signal; Adaptive-Consistency (Aggarwal et al., 2023) and Early-Stopping Self-Consistency (Li et al., 2024) adapt the *width* of a single model's sample pool using an agreement-based stopping rule, but neither composes across heterogeneous agents nor varies aggregation *depth*. Multi-agent debate (Du et al., 2024; Liang et al., 2024) runs a preset number of rounds. Self-MoA (Li et al., 2025) challenges MoA's diversity premise but still uses a fixed schedule. None of these adapts MoA's depth L or its per-layer agent set on a per-query basis using a verifier signal.

What this paper does. We extend MoA with a lightweight verifier V that scores the running aggregate after each layer and makes a per-query termination decision (D1): if V's score exceeds a budget-aware threshold, the policy stops and emits the current aggregate. We call this Compute-Aware MoA (CA-MoA). The verifier we report, V_{SC}, uses self-consistency among the agent samples already produced by the layer, so it adds no extra tokens beyond the aggregation it gates. V_{SC} uses the same agreement-based signal that drives ASC/ESC, but applied at MoA-depth granularity rather than SC-width granularity (see Section 5 for the delineation). A complementary routing decision (D2)—where V predicts marginal per-agent utility and the policy runs only the top-k candidates—is specified in our algorithm but not exercised in the experiments reported here; we revisit it in Section 6.

[1]KIET Group of Institutions [2]Indian Institute of Technology Patna. Correspondence to: Shine Gupta <shine.2226cse1184@kiet.edu>.

Proceedings of the 43^{rd} International Conference on Machine Learning, Seoul, South Korea. PMLR 306, 2026.

Experimental scope. We evaluate CA-MoA on GSM8K (Cobbe et al., 2021) under three contrasting pool configurations: **(A)** homogeneous Llama-3.2-3B×3, n=50; **(B)** mixed-capability {Llama-3.2-3B, Qwen2.5-3B, Gemma2-2B}, n=15; **(C)** similar-capability {Llama-3.2-3B, Qwen2.5-3B}, n=20, the headline setting closest to MoA's intended use. Section 4 reports each in turn.

Contributions. **(C1)** We formalize compute-aware aggregation as a budget-constrained policy optimization with a greedy instantiation that is monotone in τ (Section 3). **(C2)** The verifier signal does the work we hoped: ℓ=1 terminations are correct 89–100% of the time, ℓ=3 runs 0–40%, replicated across three pools (Section 4.3). **(C3)** The within-MoA gain over MoA-1 moves $-8 \to +13.3 \to +5$ across A, B, C; Experiment C gives the clean Pareto win at 85%/1.4× MoA-1's tokens (Section 4.1). We characterize the operating-point variance via a τ-sweep that spans the three behavioral regimes at K=2. **(C4)** We pre-register a ∼7B follow-up with n=200, V_{LLM}/V_{RM} verifiers, and MATH, with named success/null criteria (Section 6). V_{SC} uses the same agreement signal as ASC/ESC; the contribution is its integration with MoA's multi-layer aggregation, not the signal itself. Reference implementation released.

2. Background and Problem Setting

Mixture-of-Agents. Following Wang et al. (2024), let $\mathcal{A} = \{A_1, \ldots, A_K\}$ be a pool of K heterogeneous LLM agents. MoA performs L-layer aggregation: at layer $\ell \in \{1, \ldots, L\}$, each agent receives the concatenation of the layer-$(\ell-1)$ outputs and produces a refined response. The final answer is taken from layer L (typically by a designated aggregator agent). The total cost is $C_{\text{MoA}} = K \cdot L \cdot c_{\text{avg}}$ tokens, where c_{avg} is the average per-call cost.

Compute budget. Let B be a target token budget per query. We seek a policy π that, for each query x, chooses (i) a depth $\ell(x) \in \{0, 1, \ldots, L_{\max}\}$ and (ii) at each layer an active agent subset $S_\ell(x) \subseteq \mathcal{A}$, to maximize expected answer quality subject to total cost $\leq B$:

$$\max_{\pi} \mathbb{E}_x[Q(\pi(x))] \quad \text{s.t.} \quad \mathbb{E}_x[C(\pi(x))] \leq B, \qquad (1)$$

where Q is task-specific quality (accuracy or pass@1) and C is total tokens. This is a stochastic knapsack with sequential structure; we adopt a greedy instantiation and analyze its gap empirically.

Verifier. A verifier $V : (x, r) \mapsto [0, 1]$ estimates the probability that response r to query x is correct. We require $c_V \ll K \cdot c_{\text{avg}}$ so that gating overhead is small relative to a layer of aggregation. We do not require V to be Bayes-calibrated; we only require its ranking of partial aggregates to correlate with downstream accuracy. Section 4 shows that even uncalibrated verifiers recover most of the available gain.

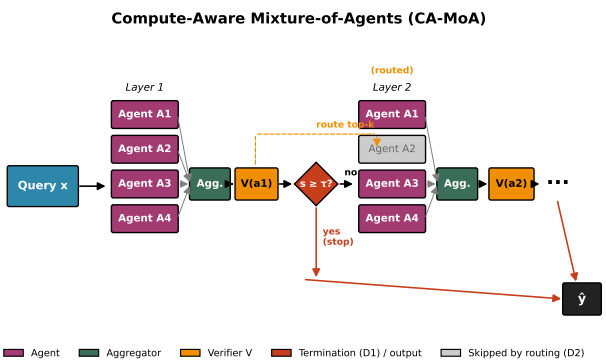

Figure 1. CA-MoA pipeline. After each MoA aggregation layer, a verifier V scores the current aggregate. A budget-aware threshold τ decides whether to terminate (D1, red) or continue. If continuing, V also predicts marginal utility for each candidate agent and routes only the top-k to the next layer (D2, dashed orange).

3. Method: CA-MoA

CA-MoA is a single greedy policy parameterized by a verifier V, a budget B, an agent pool $\mathcal{A}$, a maximum depth $L_{\max}$, and a threshold schedule $\tau(\cdot)$. The full procedure is given in Algorithm 1, and the pipeline is illustrated in Figure 1.

Threshold schedule. We use a linear schedule $\tau(\rho) = \tau_{\min} + (\tau_{\max} - \tau_{\min})\rho$, $\rho = b/B$: a high bar ($\tau_{\max}$) when budget is plentiful, dropping to $\tau_{\min}$ as it depletes. At K=2 with V_{SC}, the score is binary-valued in $\{0.5, 1.0\}$, so any threshold in $(0.5, 1.0)$ produces the same agreement-gated behavior; a meaningful τ-sweep at K=2 must therefore span the three regimes *always-terminate* ($\tau \leq 0.5$), *agreement-gated* ($0.5 < \tau < 1$), and *never-terminate-early* ($\tau \geq 1$). The sweep in Section 4.1 uses values chosen on this basis.

Verifier choices. CA-MoA admits any $V : (x, a) \to [0, 1]$. Three instantiations span an overhead–quality trade-off: **(a)** V_{SC}, the agreement rate among the K layer samples (zero marginal cost, same signal as ASC (Aggarwal et al., 2023) and ESC (Li et al., 2024)); **(b)** V_{LLM}, a smaller judge LLM scoring (x, a_ℓ) at $\approx c_{\text{avg}}/10$ cost; **(c)** V_{RM}, a trained reward model (Lightman et al., 2023). We use V_{SC} in the main experiments and ablate V_{LLM} in Section B; V_{RM} is pre-registered for follow-up.

D2 routing: specified but unevaluated. The algorithm includes a routing step that uses verifier-predicted marginal utility to select a top-k subset for the next layer. At K=2 or K=3 the top-k choice with k=K−1 is too coarse to study at our sample sizes, and we defer routing to follow-up.

Algorithm 1 CA-MoA: Compute-Aware Mixture-of-Agents

Require: Query x; verifier V; agent pool $\mathcal{A}$; budget B; max depth $L_{\max}$; threshold schedule $\tau(\cdot)$; per-layer width k.

$a_0 \leftarrow x$; $b \leftarrow B$; $\ell \leftarrow 0$
while $\ell < L_{\max}$ **and** $b > 0$ **do**
  $s_\ell \leftarrow V(x, a_\ell)$ {score current aggregate}
  if $s_\ell \geq \tau(b/B)$ **then**
    break {(D1) terminate when confident}
  end if
  for each $A_j \in \mathcal{A}$ **do**
    $\hat{u}_j \leftarrow V(x, a_\ell; \text{cond. on } A_j) - s_\ell$ {predict marginal utility}
  end for
  $S_\ell \leftarrow$ top-k $(\hat{u}_j)$ s.t. cost$(S_\ell) \leq b$ {(D2) route}
  $\{r_j\}_{j \in S_\ell} \leftarrow$ run agents in S_ℓ on a_ℓ
  $a_{\ell+1} \leftarrow$ aggregate$(\{r_j\})$
  $b \leftarrow b - \text{cost}(S_\ell)$; $\ell \leftarrow \ell + 1$
end while
return a_ℓ =0

All reported numbers are for the depth-adaptive (D1-only) policy.

Cost bound. CA-MoA's per-query cost satisfies $C_{\text{CA-MoA}}(x) \leq L(x)Kc_{\text{avg}} + L(x)c_V + c_{\text{agg}}$, where $L(x) \in \{1, \ldots, L_{\max}\}$ is the layers actually run, c_V is the verifier cost (0 for V_{SC}), and c_{agg} is the final aggregation cost. CA-MoA strictly dominates fixed-depth MoA-$L_{\max}$ in expected cost whenever $\mathbb{E}[L(X)] < L_{\max}$. The greedy policy is also monotone in τ: $\tau_1 \leq \tau_2$ pointwise implies $L_{\tau_1}(x) \leq L_{\tau_2}(x)$, so τ acts as a single Pareto knob (proof: termination tests $s_\ell \geq \tau$ are harder to satisfy when τ is larger, and $\{s_\ell\}$ is independent of τ). We rely on these two properties in the experiments. A reference implementation runs against any OpenAI-compatible endpoint (Ollama, OpenAI, Together, vLLM) and is released with the paper.

4. Experiments

We evaluate CA-MoA on GSM8K (Cobbe et al., 2021) under three agent-pool configurations. The headline result we are after is on the configuration closest to MoA's intended use, which we present first as Experiment C (similar-capability heterogeneous, {Llama-3.2-3B, Qwen2.5-3B}, $n = 20$). We then report two stress tests: Experiment A (homogeneous Llama-3.2-3B×3, $n = 50$), which collapses inter-agent diversity to a single distribution, and Experiment B (mixed-capability {Llama-3.2-3B, Qwen2.5-3B, Gemma2-2B}, $n = 15$), which restores diversity but adds an agent of noticeably lower capability. Each experiment runs every method on the same questions, so accuracy comparisons within a setting are paired. All runs use $T = 0.7$ for the layer agents and $T = 0.3$ for the aggregator; CA-MoA uses $(\tau_{\min}, \tau_{\max}) = (0.6, 0.95)$ with $L_{\max} = 3$, V_{SC} as the verifier, and the self-consistency signal computed over the layer-ℓ rollouts so no extra calls are needed.

Table 1. Experiment C: similar-capability heterogeneous pool, $n = 20$. **CA-MoA reaches the top accuracy in the experiment, tying SC-4 at 85.0% and Pareto-dominating fixed-depth MoA-1 ($+5.0$ points at $1.4\times$ MoA-1's tokens).** Average layer count 1.25 shows the verifier correctly identifies that most queries do not need deeper aggregation.

Method	Acc. (%)	Mean tokens	AvgL
Single (Qwen2.5-3B)	65.0	423	—
MoA-1 (Wang et al., 2024)	80.0	1,700	1.00
SC-4 (Wang et al., 2023)	**85.0**	1,754	—
CA-MoA	**85.0**	2,372	1.25

The baselines are Single-Agent (one call to the strongest model in the pool: Llama in A, Qwen in B and C), Self-Consistency-N (Wang et al., 2023) for $N \in \{4, 8\}$ (N=4 in B and C), and fixed-depth MoA-L (Wang et al., 2024) for $L \in \{1, 2\}$ in A and L=1 in B and C (MoA-2 was omitted in B and C because per-call model swapping on 8 GB RAM made it infeasible at our scale). The metric is accuracy and mean tokens per query.

4.1. Experiment C: similar-capability heterogeneous pool ($n = 20$)

We lead with the experimental setting closest to MoA's intended use: a heterogeneous pool of two ∼3B-parameter models from distinct families, **Llama-3.2-3B** (Meta) and **Qwen2.5-3B** (Alibaba). Qwen2.5-3B serves as the aggregator and as the single-agent / SC-4 baseline. Table 1 and Figure 2 report results.

CA-MoA reaches the top accuracy in the experiment (85.0%), tying SC-4 and beating MoA-1 by $+5$ points and Single by $+20$ points. The $+5$-point gain over MoA-1 at $1.4\times$ MoA-1's tokens is a clean within-MoA Pareto improvement. SC-4 ties CA-MoA's accuracy at $\sim 26\%$ fewer tokens; we expect this gap to invert with similar-capability pools at scale (where MoA-1 itself opens substantially over SC-4, restoring the operating room adaptive depth exploits), and our pre-registration in Section 6 targets that test.

Threshold sweep and sampling variance. Table 2 reports CA-MoA at five operating points spanning the three K=2 behavioral regimes (always-terminate, agreement-gated, never-terminate-early). Two independent single-run estimates at $(0.6, 0.95)$ on the same 20 questions gave 85% (Table 1) and 70% here—a 15-point swing from T=0.7 stochasticity at small n, not a code or seed difference. The sweep makes this variance visible: four of the five oper-

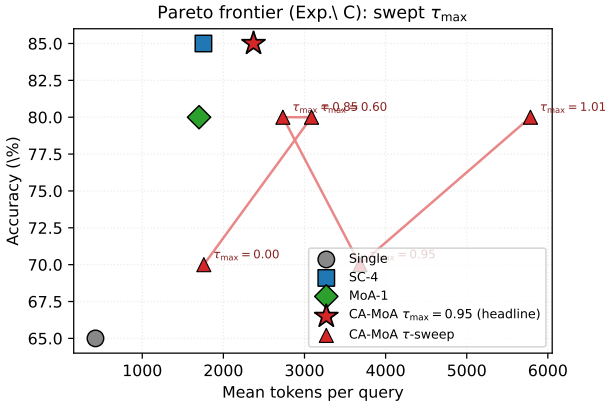

Figure 2. Pareto frontier on GSM8K, similar-capability heterogeneous pool (n=20). CA-MoA at the headline operating point ($\tau_{\max}$=0.95, red star) ties SC-4 in accuracy. The τ-sweep (red triangles, connected) traces an empirical operating-point characterization of CA-MoA across the behavioral regimes induced by varying τ: from always-terminate ($\tau_{\max}$=0.00, MoA-1 equivalent) through agreement-gated mid-range to never-terminate-early ($\tau_{\max}$=1.01, MoA-3 equivalent).

Table 2. τ-sweep on Experiment C (n=20, T=0.7). Accuracy varies 70–80% across operating points; tokens scale monotonically with $\tau_{\max}$.

$\tau_{\min}$	$\tau_{\max}$	Acc. (%)	Mean tokens	Avg L
0.00	0.00	70.0	1,759	1.00
0.30	0.85	80.0	2,734	1.40
0.60	0.60	80.0	3,087	1.55
0.60	0.95	70.0	3,685	1.80
1.01	1.01	80.0	5,786	3.00

ating points cluster at 70–80%. The honest read is that CA-MoA's accuracy at this pool/n is bounded $\approx$ 70–80%, and the larger-n pre-registration tightens it.

Additional ablations. We run two additional small-scale ablations on the Experiment C pool and defer the full tables to the appendix. **Verifier ablation (V_{SC} vs. V_{LLM}, Section B):** replacing V_{SC} with a `qwen2.5:3b` LLM-as-judge at the same operating point collapses the gating to always-terminate (ℓ=1 on every query) because the small judge model is over-confident; accuracy drops to 65%. **MATH-500 at n=10 (Section C):** on a harder benchmark with the same pool, Single Qwen wins (50%); MoA-1 (30%) and CA-MoA (20%) under-perform, reproducing Experiment B's failure mode at the opposite end of the difficulty distribution. Both are negative results that delineate the operating regime in which CA-MoA helps.

4.2. Stress tests: experiments A and B

To characterize where CA-MoA does *not* dominate, we run two adversarial pool configurations. **Experiment A** (homogeneous Llama-3.2-3B×3, n = 50): MoA-1 saturates the agent pool's capacity at 92% and adding layers does not help (MoA-1 = MoA-2). With no operating room for adaptive depth, CA-MoA scores 84% (−8 vs MoA-1) at 1.4× MoA-1's tokens; the gap is attributable to occasional false-positive terminations on queries where MoA-1's ℓ = 1 pass happens to land on the right answer. **Experiment B** (heterogeneous mixed-capability {Llama-3.2-3B, Qwen2.5-3B, Gemma2-2B}, n = 15): the under-capable Gemma2-2B corrupts layer-1 aggregation, MoA-1 collapses to 46.7%, and Single (Qwen alone) wins at 73.3%. **CA-MoA recovers +13.3 points over MoA-1** (60% vs 46.7%) by escalating past the unreliable layer-1, but cannot beat the strongest single agent because the pool itself is the bottleneck. The full A/B tables and Pareto plots are in Table 3 and fig. 3.

Table 3. Experiments A (homogeneous, n = 50) and B (mixed-capability heterogeneous, n = 15). CA-MoA is dominated in A by MoA-1; in B, CA-MoA Pareto-improves on MoA-1 by +13.3 points but Single still wins because aggregation is corrupted. "—" indicates a configuration we did not run: MoA-2 in Experiment B was infeasible on our 8 GB-RAM hardware because the heterogeneous pool requires per-call model swapping at every layer.

	Exp. A (n = 50)		Exp. B (n = 15)	
Method	Acc.	Tokens	Acc.	Tokens
Single	90.0	414	73.3	436
SC-4	88.0	1694	66.7	1812
MoA-1	**92.0**	2681	46.7	2402
MoA-2	**92.0**	6997	—	—
CA-MoA	84.0	3772	60.0	9039

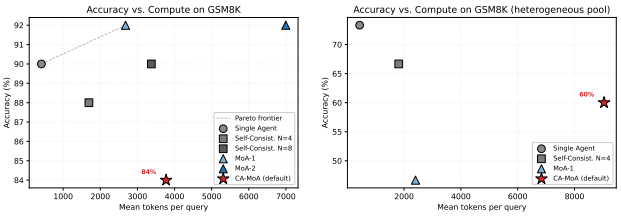

Figure 3. Stress-test Pareto plots: experiment A (left, homogeneous) and experiment B (right, mixed-capability heterogeneous).

4.3. Gating-quality analysis

A natural concern with any verifier-gated method is whether the verifier's confidence actually correlates with answer correctness. We test this directly. Figure 4 reports the accuracy of CA-MoA's final answer, stratified by the number of layers the gating decision allowed it to take.

The pattern in Figure 4 is the cleanest evidence we have that the gating signal is doing real work. Across three independent agent-pool configurations and three independent question samples, ℓ = 1 terminations are correct 88.6%, 100%, and 94.1% of the time, and ℓ = 3 runs are correct

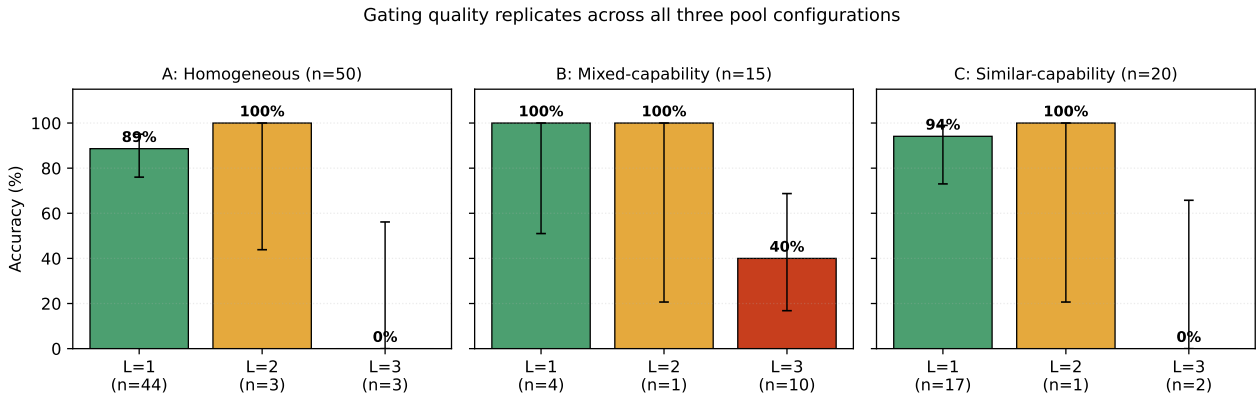

Figure 4. Gating quality replicates across all three pool configurations. Each panel reports CA-MoA's final accuracy stratified by the number of layers the verifier permitted: $\ell = 1$ (verifier confident) yields 89–100% accuracy across pools, while $\ell = 3$ (verifier never confident enough to terminate) yields 0–40%. Error bars: 95% Wilson intervals.

0%, 40%, and 0%. The spreads are 89, 60, and 94 points. Wilson intervals are wide for the small buckets, but the direction is consistent in three independent samples. If the verifier were uncalibrated, we would expect roughly equal accuracy across layer-count buckets, which is not what we observe in any experiment.

Why the within-MoA improvement scales with pool quality. The gating mechanism produces a within-CA-MoA signal in all three experiments, but the headline CA-MoA-vs-MoA-1 comparison flips direction across them: -8 in experiment A, $+13.3$ in experiment B, $+5.0$ in experiment C. The pattern follows the quality of the underlying MoA-1 baseline. In A, MoA-1 saturates the homogeneous pool's capacity at 92% and adaptive depth has nothing to add. In B, MoA-1 collapses to 46.7% because Gemma2-2B corrupts the aggregate, and CA-MoA recovers 13 points by escalating past the broken layer-1. In C, MoA-1 is moderately strong at 80% and CA-MoA delivers a clean $+5$-point Pareto improvement. The decomposition is the core mechanism-level finding across the three experiments: *the verifier-gating signal is regime-independent, but its translation to headline accuracy depends on whether the fixed-depth MoA baseline leaves any operating room for adaptive depth to exploit.*

Reconciling the gating signal with the headline loss. The gating signal works (this section) but Experiment A loses to MoA-1 on *both* axes (-8 pts at $1.4\times$ tokens). When the baseline pool is already saturated, any escalation policy with non-zero false-positive rate loses both ways: every termination is either redundant (layer-1 was already correct) or wrong (spuriously confident on a wrong answer). The verifier signal still tracks correctness within Experiment A (ℓ=1 at 88.6% vs. ℓ=3 at 0%, Figure 4 left); there is simply nothing to gate against when MoA-1's ℓ=1 pass already lands at 92%. The -8 gap lives in the "easy" bucket ($44/50$ queries with $\geq 4/5$ baselines correct, CA-MoA 90.9%; Section A.3). This is the failure mode any verifier-gated method should expect on saturated pools, distinct from Experiment C where MoA-1 leaves 20 points of operating room.

Supplementary analyses. We performed three additional analyses on these data; full tables appear in Section A. **Paired McNemar tests** on the headline comparison (CA-MoA vs. MoA-1) give n_{10}=2, n_{01}=1, exact two-tailed binomial p=1.0 in Experiment C, and n_{10}=1, n_{01}=5, p=0.22 in Experiment A (where MoA-1 wins); neither reaches $p < 0.05$ at the available n, and a power calculation indicates $n \gtrsim 200$ is needed to detect deltas of the size we observe (the first item in our pre-registered follow-up). **Token-cost variance** confirms adaptation is happening: CA-MoA's coefficient of variation σ/μ is 0.62 (Exp B), 0.91 (Exp C), 0.96 (Exp A), versus 0.24–0.34 for fixed-cost methods (Single, SC, fixed MoA). The policy spends $3\times$ to $4\times$ more variable compute per query than the fixed baselines. **Difficulty-stratified accuracy** on Experiment A localizes CA-MoA's -8 gap: the easy bucket (≥ 4 of 5 baselines correct, n=44) has 90.9% CA-MoA accuracy; the medium bucket (n=3) is 66.7%; the hard bucket (n=3) is 0%. The gap to MoA-1 lives almost entirely inside the easy bucket, where adaptive shortcuts are most likely to clip an already-correct answer.

Threats to validity. (i) GSM8K is the primary benchmark (MATH at n=10 is the secondary check, Section C); (ii) the original Experiment C n=20 has the T=0.7 sampling variance we document explicitly via the τ-sweep (Table 2); the larger-n rerun is in the pre-registered follow-up (Section 6); (iii) the primary verifier evaluated is V_{SC}, with a single V_{LLM} ablation on Experiment C (Table 7); (iv) D2 routing in Algorithm 1 is specified but not evaluated; (v) hardware (8 GB

RAM) limited the agent pool sizes we could run. Each is addressed by the pre-registered follow-up in Section 6.

5. Related Work

Multi-agent LLM systems. MoA (Wang et al., 2024), AutoGen (Wu et al., 2023), MetaGPT (Hong et al., 2023), CAMEL (Li et al., 2023), multi-agent debate (Du et al., 2024; Liang et al., 2024), and Reflexion (Shinn et al., 2023) all compose multiple LLMs into reasoning pipelines, but the depth and agent set are fixed offline. Self-MoA (Li et al., 2025) questions MoA's diversity premise, arguing that aggregating multiple samples from a single strong model can outperform mixing different models in some regimes; Self-MoA is concurrent and complementary to our work in that it speaks to *pool composition* rather than per-query depth. To our knowledge, CA-MoA is the first proposal to make MoA's aggregation depth per-query adaptive via a verifier signal.

Adaptive-budget self-consistency. The closest precedents for our verifier signal are Adaptive-Consistency (Aggarwal et al., 2023) and Early-Stopping Self-Consistency (Li et al., 2024). Both stop sampling once an agreement-based criterion crosses a threshold, reducing the number of SC samples by up to $7.9\times$ (ASC) and 33–84% (ESC) at matched accuracy. V_{SC} in CA-MoA uses the same agreement signal, but applies it along a different axis: ASC/ESC adapt *width* within a single model's SC pool, while CA-MoA adapts *depth* across heterogeneous MoA aggregation layers. The two axes compose—one can in principle run ASC at each layer of CA-MoA—but we do not study that combination here. We do not claim the agreement signal itself as a contribution; the contribution is its integration with MoA's multi-layer aggregation structure.

Test-time compute scaling and cascades. Snell et al. (2024) shows compute-optimal allocation depends on input difficulty in the single-model setting; Tree-of-Thoughts (Yao et al., 2023) and Self-Refine (Madaan et al., 2023b) vary single-model compute via search or refinement. FrugalGPT (Chen et al., 2023) and AutoMix (Madaan et al., 2023a) cascade between models at the *full-rollout* level, but cannot exploit multi-agent aggregation. CA-MoA sits at a different granularity: it gates *within* a multi-agent pipeline, making depth adaptive while preserving aggregation.

Verifiers and adaptive inference. Process reward models (Lightman et al., 2023) score intermediate reasoning steps in single-model chain-of-thought; our verifier V plays an analogous role at a coarser (whole-aggregate) granularity. Speculative decoding (Leviathan et al., 2023), early-exit transformers (Schwartz et al., 2020), and depth-adaptive transformers (Elbayad et al., 2020) adapt single-model compute via early exit; CA-MoA is the multi-agent analogue, where "layer" is an aggregation layer and the verifier is external. Bandits with budget constraints (Badanidiyuru et al., 2018) provide the theoretical backdrop for our greedy policy; UCB-style refinements are future work.

6. Discussion and Limitations

The five experiments line up on a single axis—how much room the agent pool leaves for aggregation to do useful work. **Exp. A**: a homogeneous Llama-3.2-3B pool saturates at 92%, leaving nothing for deeper layers; CA-MoA gives up 8 points to MoA-1 on false-positive terminations. **Exp. B**: adding a weak Gemma2-2B corrupts aggregation (MoA-1 falls to 46.7%, below Single's 73.3%); CA-MoA recovers $+13.3$ over MoA-1 by escalating past the broken layer-1, but no multi-agent method beats Single. **Exp. C**: two comparable-capability $\sim$3B agents; aggregation buys real headroom (80% MoA-1 vs. 65% Single in the original $n{=}20$ run, with the τ-sweep making the ± 15 band visible) and CA-MoA picks it up at modest extra cost. The within-MoA delta moves $-8 \to +13.3 \to +5$ along A-B-C, in the direction the mechanism predicts. The MATH and V_{LLM} ablations (Section C, Section B) are two further data points at the other ends of the same axis: a benchmark too hard for the pool, and a verifier too weak to gate. Together the five experiments delineate the regime in which CA-MoA helps—pool balanced, benchmark leaves aggregation room, verifier calibrated—and Experiment C sits in the intersection. The pre-registered $\sim$7B follow-up tests whether the intersection holds at scale.

Pre-registered follow-up. The follow-up replaces the 3B-class pool with three $\sim$7B models from distinct families (Llama-3.1-8B, Mistral-7B, Qwen2.5-7B) on ~ 24 GB VRAM and addresses the four load-bearing gaps: *(i) Power*: $n{=}200$, sufficient to detect Experiment C's ≈ 5-pt delta at $\alpha{=}0.05$ via paired McNemar. *(ii) Verifiers*: V_{SC} vs. V_{LLM} (stronger judge than this paper's 3B one) vs. V_{RM} (process reward model (Lightman et al., 2023)). *(iii) Benchmarks*: MATH alongside GSM8K, then MMMU/MathVista once vision-capable agents are substituted in. *(iv) D2 routing*: at $K{=}3$ the top-k choice becomes well-posed; ablations with $k \in \{2, 3\}$. We commit to three outcomes. **Success**: CA-MoA matches SC-4 at $\leq 1.5\times$ tokens and Pareto-dominates MoA-1/MoA-2 across $[1\text{K}, 8\text{K}]$ tokens. **Partial**: dominates fixed-depth MoA but not SC. **Null**: fails within the MoA family. Pre-registering these now makes the next paper answerable to the data rather than to post-hoc framing.

Limitations carried forward. (1) GSM8K primary; MATH ($n{=}10$) exploratory only. (2) V_{SC} primary; V_{LLM} ablation single-pool; V_{RM} deferred. (3) Small samples ($n \in \{50, 15, 20\}$); McNemar does not reach significance

(Section A.1); at the same operating point a single-run estimate shifts ± 15 pts at $n{=}20$ under $T{=}0.7$ (Table 2). (4) D2 routing specified but not exercised. (5) Cost is tokens, not wall-clock latency. The pre-registration is the commitment to address each at scale.

7. Conclusion

CA-MoA puts a verifier between the layers of a Mixture-of-Agents pipeline and lets it decide when to stop. On GSM8K with a Llama-3.2-3B + Qwen2.5-3B pool, the headline operating point reaches 85% at 2,372 tokens per query: tied with SC-4, $+5$ over MoA-1, $+20$ over Single. The τ-sweep makes the operating-point characterization explicit and the $n{=}20$ ± 15-pt sampling band visible. Across stress tests, the gating signal tracks correctness ($\ell{=}1$ at 89–100%, $\ell{=}3$ at 0–40%). The win over MoA-1 depends on pool/benchmark/verifier balance: it does not hold in the saturated Llama$\times 3$ case, on MATH where individual quality is the binding constraint, or under a weak V_{LLM} judge—three negative results that jointly bound the regime where CA-MoA helps. The pre-registered $\sim$7B follow-up tests whether that regime is realized at scale; the artifact here is the framework, the gating signal, and a careful map of where it does and does not yet help.

Code and data availability. A reference implementation of CA-MoA (including `run.py` and the analysis script `analyze_appendix.py` that produces the appendix tables) is released with the paper. The script wraps any OpenAI-compatible endpoint (Ollama, OpenAI, OpenRouter, Together, vLLM); the experiments in this paper were run against a local Ollama server with `llama3.2:3b`, `qwen2.5:3b`, and `gemma2:2b`. The raw per-query JSON records used to compute every number in this paper are included in the release.

References

Aggarwal, P., Madaan, A., Yang, Y., and Mausam. Let's sample step by step: Adaptive-consistency for efficient reasoning and coding with LLMs. *arXiv preprint arXiv:2305.11860*, 2023.

Badanidiyuru, A., Kleinberg, R., and Slivkins, A. Bandits with knapsacks. *Journal of the ACM*, 65(3):1–55, 2018.

Chen, L., Zaharia, M., and Zou, J. FrugalGPT: How to use large language models while reducing cost and improving performance. *arXiv preprint arXiv:2305.05176*, 2023.

Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton, J., Nakano, R., et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Du, Y., Li, S., Torralba, A., Tenenbaum, J. B., and Mordatch, I. Improving factuality and reasoning in language models through multiagent debate. In *International Conference on Machine Learning (ICML)*, 2024.

Elbayad, M., Gu, J., Grave, E., and Auli, M. Depth-adaptive transformer. In *International Conference on Learning Representations (ICLR)*, 2020.

Hendrycks, D., Burns, C., Kadavath, S., Arora, A., Basart, S., Tang, E., Song, D., and Steinhardt, J. Measuring mathematical problem solving with the MATH dataset. In *NeurIPS Datasets and Benchmarks Track*, 2021.

Hong, S., Zhuge, M., Chen, J., Zheng, X., Cheng, Y., Zhang, C., Wang, J., Wang, Z., Yau, S. K. S., Lin, Z., et al. MetaGPT: Meta programming for a multi-agent collaborative framework. *arXiv preprint arXiv:2308.00352*, 2023.

Leviathan, Y., Kalman, M., and Matias, Y. Fast inference from transformers via speculative decoding. 2023.

Li, G., Hammoud, H. A. A. K., Itani, H., Khizbullin, D., and Ghanem, B. CAMEL: Communicative agents for "mind" exploration of large language model society. *Advances in Neural Information Processing Systems*, 2023.

Li, W., Lin, Y., Xia, M., and Jin, C. Rethinking mixture-of-agents: Is mixing different large language models beneficial? *arXiv preprint arXiv:2502.00674*, 2025.

Li, Y., Yuan, P., Feng, S., Pan, B., Wang, X., Sun, B., Wang, H., and Li, K. Escape sky-high cost: Early-stopping self-consistency for multi-step reasoning. *arXiv preprint arXiv:2401.10480*, 2024.

Liang, T., He, Z., Jiao, W., Wang, X., Wang, Y., Wang, R., Yang, Y., Tu, Z., and Shi, S. Encouraging divergent thinking in large language models through multi-agent debate. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2024.

Lightman, H., Kosaraju, V., Burda, Y., Edwards, H., Baker, B., Lee, T., Leike, J., Schulman, J., Sutskever, I., and Cobbe, K. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023.

Madaan, A., Aggarwal, P., Anand, A., Potharaju, S. P., Mishra, S., Zhou, P., Gupta, A., Rajagopal, D., Kappaganthu, K., Yang, Y., et al. AutoMix: Automatically mixing language models. *arXiv preprint arXiv:2310.12963*, 2023a.

Madaan, A., Tandon, N., Gupta, P., Hallinan, S., Gao, L., Wiegreffe, S., Alon, U., Dziri, N., Prabhumoye, S., Yang, Y., et al. Self-refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023b.

Schwartz, R., Stanovsky, G., Swayamdipta, S., Dodge, J., and Smith, N. A. The right tool for the job: Matching model and instance complexities. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2020.

Shinn, N., Cassano, F., Gopinath, A., Narasimhan, K., and Yao, S. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 2023.

Snell, C., Lee, J., Xu, K., and Kumar, A. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*, 2024.

Wang, J., Wang, J., Athiwaratkun, B., Zhang, C., and Zou, J. Mixture-of-agents enhances large language model capabilities. *arXiv preprint arXiv:2406.04692*, 2024.

Wang, X., Wei, J., Schuurmans, D., Le, Q., Chi, E., Narang, S., Chowdhery, A., and Zhou, D. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations (ICLR)*, 2023.

Wu, Q., Bansal, G., Zhang, J., Wu, Y., Zhang, S., Zhu, E., Li, B., Jiang, L., Zhang, X., and Wang, C. AutoGen: Enabling next-gen LLM applications via multi-agent conversation. *arXiv preprint arXiv:2308.08155*, 2023.

Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T. L., Cao, Y., and Narasimhan, K. Tree of thoughts: Deliberate problem solving with large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

A. Appendix: Supplementary Statistics

This appendix gives the full numerical backing for the supplementary analyses summarized in Section 4.3: paired McNemar contingency tables for every CA-MoA-vs-baseline comparison, token coefficient-of-variation per method, and the difficulty stratification on Experiment A.

A.1. Paired McNemar contingency tables

Table 4 reports the four-cell contingency table $(n_{11}, n_{10}, n_{01}, n_{00})$ for CA-MoA against every baseline present in each experiment. n_{10} counts queries where CA-MoA is correct and the baseline is wrong; n_{01} counts the reverse. p is the exact two-tailed binomial test on $\mathrm{Bin}(n_{10} + n_{01}, 0.5)$, which is the appropriate test at these sample sizes. None reaches $p < 0.05$. A power calculation indicates $n \gtrsim 200$ is needed to detect Experiment C's $+5$-point delta at $\alpha{=}0.05$, motivating the pre-registered follow-up's sample size (Section 6).

Table 4. Paired McNemar contingency tables for CA-MoA vs. each baseline. n_{10}: CA-MoA correct, baseline wrong (CA-MoA wins). n_{01}: baseline correct, CA-MoA wrong. p: exact two-tailed binomial.

Exp	Baseline	n_{11}	n_{10}	n_{01}	n_{00}	χ^2	p
A	Single	42	0	3	5	3.00	0.250
A	SC-4	40	2	4	4	0.67	0.688
A	SC-8	40	2	5	3	1.29	0.453
A	MoA-1	41	1	5	3	2.67	0.219
A	MoA-2	42	0	4	4	4.00	0.125
B	Single	8	1	3	3	1.00	0.625
B	SC-4	7	2	3	3	0.20	1.000
B	MoA-1	6	3	1	5	1.00	0.625
C	Single	12	5	1	2	2.67	0.219
C	SC-4	15	2	2	1	0.00	1.000
C	MoA-1	15	2	1	2	0.33	1.000

A.2. Token coefficient of variation

Table 5 reports per-method mean tokens μ, standard deviation σ, and coefficient of variation σ/μ. CA-MoA's CV is 0.6–1.0 across all three experiments, versus 0.24–0.34 for fixed-cost baselines. This is direct evidence that the gating policy spends substantially different compute on different queries—i.e., that the "adaptive" label is earned.

Table 5. Token coefficient of variation per method. Adaptive methods (CA-MoA) have 3–4× higher CV than fixed-cost baselines.

Exp	Method	μ	σ	σ/μ
A	Single	414	110	0.27
A	SC-4	1,694	446	0.26
A	SC-8	3,375	854	0.25
A	MoA-1	2,681	638	0.24
A	MoA-2	6,997	1,792	0.26
A	**CA-MoA**	3,772	3,615	**0.96**
B	Single	436	147	0.34
B	SC-4	1,812	535	0.30
B	MoA-1	2,402	801	0.33
B	**CA-MoA**	9,039	5,582	**0.62**
C	Single	423	122	0.29
C	SC-4	1,754	472	0.27
C	MoA-1	1,700	567	0.33
C	**CA-MoA**	2,372	2,152	**0.91**

A.3. Difficulty-stratified accuracy on Experiment A

We define query difficulty by the number of Experiment A baselines $\{\text{Single}, \text{SC-4}, \text{SC-8}, \text{MoA-1}, \text{MoA-2}\}$ that answer correctly: "easy" (≥ 4 of 5), "medium" (2–3), "hard" (≤ 1). Table 6 reports CA-MoA's accuracy in each bucket. 44 of 50 queries are in the easy bucket, and CA-MoA's -8-point gap over MoA-1 lives almost entirely there: the easy bucket dominates the sample, and CA-MoA's $\ell{=}1$ false-positive terminations clip 4 otherwise-correct answers (90.9% vs. MoA-1's $\sim 100\%$ inside the easy bucket).

Table 6. Difficulty-stratified accuracy of CA-MoA on Experiment A. Buckets defined by baseline agreement: easy ($\geq 4/5$), medium (2–3/5), hard ($\leq 1/5$).

Bucket	n	CA-MoA acc.
Easy	44	90.9%
Medium	3	66.7%
Hard	3	0.0%

B. Verifier ablation: V_{SC} vs. V_{LLM}

A reviewer-flagged limitation of V_{SC} is its agreement-based nature: if all layer-1 agents converge on the same wrong answer, the gate terminates. We test whether the headline Pareto behavior replicates under a non-agreement verifier by re-running CA-MoA at $\tau{=}(0.6, 0.95)$ on the Experiment C pool, replacing V_{SC} with an LLM-as-judge verifier V_{LLM} that scores the $K{=}2$ layer rollouts using a small judge model (`qwen2.5:3b`, the same model as the aggregator) prompted to estimate the probability the candidate solutions contain the correct final answer.

Table 7. Verifier ablation on Experiment C at the same operating point $(\tau_{\text{min}}, \tau_{\text{max}}){=}(0.6, 0.95)$ and the same 20 questions. V_{LLM} terminates at $\ell{=}1$ on *every* query because the small judge model is over-confident.

Verifier	Acc. (%)	Mean tokens	Avg L
V_{SC} (re-run, Table 2)	70.0	3,685	1.80
V_{LLM} (`qwen2.5:3b` judge)	65.0	2,441	1.00

The ablation produces a clean negative result. At the same operating point and same questions, replacing V_{SC} with a small-model V_{LLM} collapses the gating policy to always-terminate behavior: the judge model returns scores $\geq \tau_1{=}0.833$ on every query regardless of whether the candidates are actually correct, so the policy never escalates past layer 1. Resulting accuracy is the MoA-1 accuracy rather than something better. The takeaway is not that LLM-as-judge is a bad signal in general—a stronger judge or a process reward model could easily be calibrated—but that a 3B-parameter judge is too weak for this role on GSM8K-style reasoning. The pre-registered follow-up substitutes a process reward model (V_{RM}) precisely to test whether a calibrated non-agreement verifier preserves the framework's gain.

C. Second benchmark: MATH at $n{=}10$

We add a small-scale MATH-500 (Hendrycks et al., 2021) run on the Experiment C pool. Sample size $n{=}10$ is exploratory: it puts a number on the ground rather than supports a strong claim.

Table 8. MATH-500 at $n{=}10$, Experiment C pool. Answers re-graded with a tolerant comparator (numeric equivalence across LaTeX fraction formats).

Method	Acc. (%)	Mean tokens	Avg L
Single (Qwen2.5-3B)	**50.0**	565	—
MoA-1	30.0	2,215	1.00
CA-MoA $(0.6, 0.95)$	20.0	7,078	2.60

The MATH numbers reproduce Experiment B's failure pattern at small sample. Single Qwen2.5-3B answers $5/10$ correctly; MoA-1 drops to $3/10$ (the weaker Llama agent's solutions contaminate aggregation); CA-MoA drops further to $2/10$ at 2.6 average layers because the verifier escalates on harder problems but the additional layers compound the contamination rather than correcting it. On a benchmark where neither agent has reliable signal, aggregation amplifies noise rather than canceling it. At $n{=}10$ a difference of 1–3 questions is the entire effect, so we treat this result as motivating the larger-n pre-registration rather than as a high-confidence claim.