EMS: Multi-Agent Voting via Efficient Majority-then-Stopping

Anonymous ACL submission

Abstract

Majority voting is the standard for aggregating multi-agent responses into a final decision. However, traditional methods typically require all agents to complete their reasoning before aggregation begins, leading to significant computational overhead, as many responses become redundant once a majority consensus is achieved. In this work, we formulate the multi-agent voting as a reliability-aware agent scheduling problem, and propose an Efficient Majority-then-Stopping (EMS) to improve reasoning efficiency. EMS prioritizes agents based on task-aware reliability and terminates the reasoning pipeline the moment a majority is achieved from the following three critical components. Specifically, we introduce Agent Confidence Modeling (ACM) to estimate agent reliability using historical performance and semantic similarity, Adaptive Incremental Voting (AIV) to sequentially select agents with early stopping, and Individual Confidence Updating (ICU) to dynamically update the reliability of each contributing agent. Extensive evaluations across six benchmarks demonstrate that EMS consistently reduces the average number of invoked agents by 32%.

1 Introduction

Large Language Models (LLMs) have demonstrated remarkable success in complex tasks such as mathematical reasoning, code generation, and commonsense question answering (Kojima et al., 2022; Wan et al., 2023; Li et al., 2026). Despite these advancements, a single LLM often exhibits limited reasoning diversity and struggles with problems requiring dynamic, task-specific strategies (Mirzadeh et al., 2024; Jiang et al., 2024; Guo et al., 2024). Recent research has sought to address these limitations by leveraging multi-agent systems (MAS) (Li et al., 2023; Chen et al., 2024; Wu et al., 2023), where multiple LLM-based agents collaborate to tackle complex tasks. By aggregating responses from a diverse set of agents, MAS can exploit a variety of reasoning paths, leading to more robust and accurate outcomes than those achieved by individual models. A critical aspect of MAS is the aggregation of multiple agent responses into a final decision (Wang et al., 2023; Jiang et al., 2023; Liang et al., 2024; Du et al., 2023), which is typically achieved through a decision-making operation.

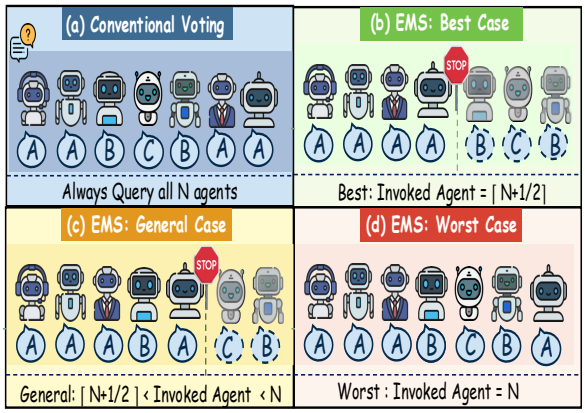

Figure 1: Intuitive comparison of different multi-agent voting processes. There exists a significant difference in terms of *Number of Invoked Agents* among different voting sequences.

Among
jority voting has become one of the most widely used methods due to its simplicity and effectiveness(Kaesberg et al., 2025; Zhao et al., 2024). In the standard majority voting setting, each agent performs reasoning independently, and the final decision is made based on the most frequent response. Several extensions of this basic paradigm have been proposed to enhance performance, such as reliability-based weighting considering agents' past accuracy, and confidence-based soft voting integrating model-reported confidence scores. However, these approaches still follow the same fundamental paradigm: *reasoning first, aggregation*

later. In this setup, all agents must complete their reasoning before a final decision can be made. This paradigm, while effective, introduces a significant inefficiency. In many scenarios, the majority decision can be reached before all agents have completed their reasoning, making the contributions of the remaining agents redundant. This leads to unnecessary computational overhead, especially when large numbers of agents or expensive LLM inferences are involved. This inefficiency raises a key challenge: *how can we reduce redundant reasoning in multi-agent LLM systems while preserving the accuracy benefits of majority voting?*

Inspired by principles from Byzantine Fault Tolerance(Lamport et al., 1982), a reasonable strategy is to adopt a majority-then-stop mechanism. In a system with N agents, once a majority of $[(N+1)/2]$ agents reach consensus, the final decision is determined, and the remaining reasoning can be terminated. This strategy has two primary advantages: it preserves the decision accuracy of standard majority voting while potentially reducing inference costs by avoiding unnecessary reasoning. However, its efficiency critically depends on the order in which agents are queried. As shown in Figure 1, if the most reliable agents are consulted first, a consensus can be reached more quickly, whereas querying less reliable agents early may require waiting for more agents to vote. Therefore, achieving efficient early stopping requires prioritizing agents that are most likely to produce a response for majority voting.

To address the above issue, we formulate the majority voting as a reliability-aware agent scheduling problem in multi-agent LLM systems, and propose an Efficient Majority-then-Stopping (EMS) to improve reasoning efficiency without sacrificing decision-making accuracy. The main idea of EMS is to estimate the task-aware reliability of each agent and prioritize those with higher expected reliability during voting, enabling early majority consensus and reducing redundant inference. EMS consists of three key components: Agent Confidence Modeling (ACM) First, Agent Confidence Modeling (ACM) constructs a comprehensive confidence profile for each agent by considering historical reliability or semantic similarity to the query task. Second, the Adaptive Incremental Voting (AIV) dynamically queries agents in descending order of estimated reliability and stops the voting process once a majority consensus is reached. Third, Individual Confidence Updating (ICU) is proposed to update the reliability of the individual agent that is contributing to voting.

The major contributions can be summarized as follows:

1. We analyze the inefficiency of the widely used reasoning-first aggregation paradigm in Multi-Agent Voting and formulate it as a reliability-aware agent scheduling problem to improve inference efficiency.

2. We propose an Efficient Majority-then-Stopping (EMS), a novel decision-making framework that enables early termination of multi-agent reasoning while preserving the accuracy of majority voting.

3. Extensive evaluations across six benchmarks demonstrate that EMS consistently reduces the average number of invoked agents by 32%.

2 Related Work

Multi-Agent Systems. Recent advances in large language models (LLMs) have stimulated increasing interest in multi-agent systems (MAS) (Kojima et al., 2022; Wan et al., 2023; Mirzadeh et al., 2024), where multiple LLM-based agents collaborate to solve complex tasks. Representative frameworks such as AutoGen (Wu et al., 2023) enable structured collaboration among multiple agents through role assignment, communication, and task decomposition. Other interaction-based approaches, including multi-agent debate (Chan et al., 2023; Liang et al., 2024), further encourage agents to critique and refine one another's intermediate reasoning, thereby improving the quality of final outputs. While these studies demonstrate the effectiveness of collaborative reasoning for improving task performance, the efficiency of multi-agent inference, particularly the reduction of redundant agent calls during decision making, remains relatively underexplored.

Efficient Multi-Agent Inference. To improve efficiency, recent work has explored adaptive routing and selective invocation mechanisms (Yue et al., 2025). Another work improves the efficiency of debate-style reasoning by reducing redundant communication rather than changing the final aggregation rule. For example, GroupDebate (Liu et al., 2024) reduces debate token cost through grouped discussion and intermediate result sharing across debate groups. Similarly, DOWN (Eo et al.,

2025)activates debate only when initial confidence suggests that additional deliberation is necessary, thereby avoiding unnecessary interaction on easy instances. These approaches aim to selectively allocate computation based on intermediate confidence signals or structured communication. However, they typically focus on optimizing interaction efficiency rather than directly leveraging the majority consensus structure for early termination.

Multi-Agent Voting and Aggregation. Aggregating outputs from multiple agents is a fundamental component of MAS (Wang et al., 2025), and majority voting remains one of the most widely adopted decision rules because of its simplicity and strong performance. Recent work demonstrates that majority voting accounts for the vast majority of performance gains typically attributed to complex multi-agent debate, suggesting that expensive inter-agent communication rounds are often unnecessary (Kaesberg et al., 2025; Choi et al., 2025). Furthermore, standard voting protocols are particularly optimal for reasoning-based tasks, significantly outperforming other decision-making structures (Zhang et al., 2025). Despite these advances, most existing approaches follow a *reasoning-first-aggregation-later* paradigm, which introduces significant computational waste as the final decision is often reachable before all agents complete their reasoning. Our work addresses this gap by formulating majority voting as a reliability-aware agent scheduling problem.

3 Methodology

3.1 Problem Formulation

Consider a multi-agent system (Li et al., 2023; Kim et al., 2025) composed of N agents $\mathcal{A} = \{\boldsymbol{a}_1, \boldsymbol{a}_2, \ldots, \boldsymbol{a}_N\}$, where each agent $\boldsymbol{a}_j$ is instantiated by a specific Large Language Model (LLM). Let $\mathcal{Q} = \{\boldsymbol{q}_1, \boldsymbol{q}_2, \ldots, \boldsymbol{q}_M\}$ denote a dataset consisting of M queries, where each query $\boldsymbol{q}_i$ is associated with a ground-truth answer y_i^*. During reasoning, each agent $\boldsymbol{a}_j$ can be modeled as an inference function $\mathcal{F}_j : \mathcal{Q} \to \mathcal{Y}$, where $\mathcal{Y}$ denotes the predicted answer space. Given a query $\boldsymbol{q}_i$, the agent $\boldsymbol{a}_j$ independently performs reasoning to produce the predicted answer $y_{i,j} = \mathcal{F}_i(\boldsymbol{q}_j)$. The goal of multi-agent decision making is to aggregate a final prediction $\hat{y}_i$ for each query based on all responses $\{y_{i,j}\}_{j=1}^N$.

Vanilla Majority-Voting. In the vanilla majority voting scheme, all N agents are queried for each query $\boldsymbol{q}_j$, resulting in a collection of agent responses $\mathcal{Y}_i = \{y_{i,j}\}_{j=1}^N$. Let $\mathcal{U}_i$ denote the set of unique answers appearing in $\mathcal{Y}_i$. The final prediction is determined by selecting the most frequent answer,

$$\hat{y}_i = \arg\max_{y \in \mathcal{U}_i} \sum_{j=1}^{N} \mathbb{I}\big[y_{i,j} = y\big], \tag{1}$$

where $\mathbb{I}[\cdot]$ is the indicator function. The answer that obtains the maximal vote count is selected as the final result. Although it is a simple and effective strategy to aggregate the response, it requires every agent to perform inference before majority-voting, leading to unnecessary computational cost when the majority decision could already be determined earlier.

Sequential Majority Voting. To reduce redundant reasoning, we reformulate the above parallel voting process as a sequential decision procedure. Instead of querying all agents simultaneously, agents are activated in an ordered sequence until a majority consensus is reached. To order the sequence of all agents, we maintain a global confidence state for all agents,

$$\Theta = \{\Phi_1, \Phi_2, \ldots, \Phi_N\}, \tag{2}$$

where $\Phi_j = (c_j, v_j, \mathcal{H}_j)$ denotes the state associated of $\boldsymbol{a}_j$ agent. c_j denotes the number of queries on which agent $\boldsymbol{a}_j$ agreed with the final voting consensus, v_j denotes the number of queries in which $\boldsymbol{a}_j$ participated, and $\mathcal{H}_j$ is a semantic history buffer storing the embeddings of queries on which $\boldsymbol{a}_j$ agreed with the consensus.

After that, one critical problem is how to measure the task-aware reliability between all agents $\mathcal{A}$ and the given query $\boldsymbol{q}_i$. Formally, a confidence score between $\boldsymbol{q}_i$ and $\boldsymbol{a}_j$ is computed with:

$$S_{i,j} = \mathcal{S}(\boldsymbol{q}_i, \Phi_j), \tag{3}$$

where $\mathcal{S}(\cdot, \cdot)$ is the function to estimates the reliability of agent $\boldsymbol{a}_j$ on query $\boldsymbol{q}_i$. Finally, $\boldsymbol{S}_i = \{S_{i,j}\}_{j=1}^N$ denote the set of confidence scores of all agents.

After that, sorting these scores in descending order produces a permutation of all agents,

$$\Psi_i = \mathrm{argsort}_{\downarrow}(\boldsymbol{S}_i), \tag{4}$$

where Ψ_i is a permutation of sorted index $\{1, \ldots, N\}$. Then, the resulting prioritized agent

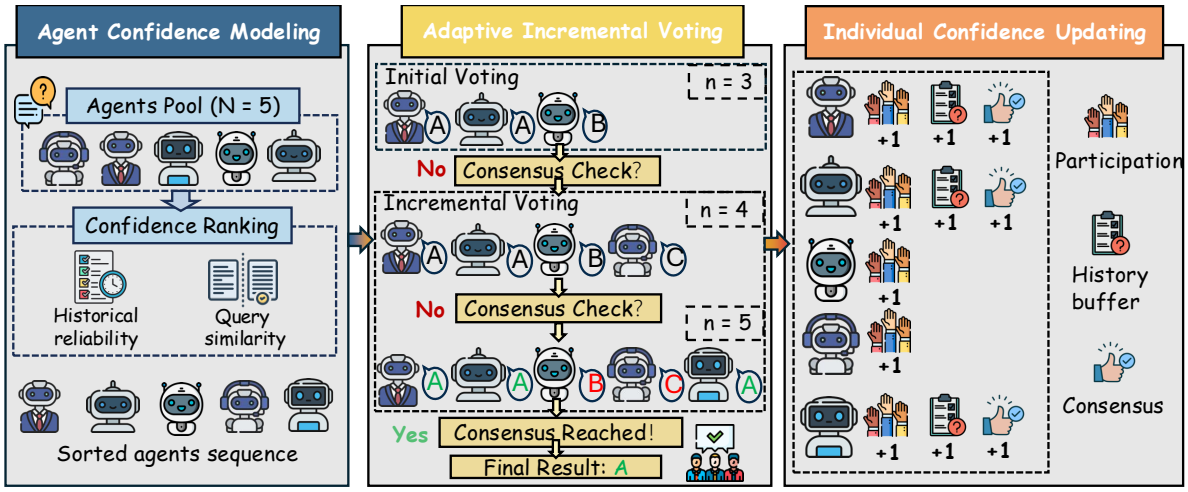

Figure 2: Overview of the proposed Efficient Majority-then-Stopping (EMS) framework. For each query, EMS first uses the Agent Confidence Modeling (ACM) to determine a reliability-aware voting order. It then performs incremental voting via Adaptive Incremental Voting (AIV), where agents are invoked sequentially according to the estimated order. Finally, Individual Confidence Updating (ICU) updates the confidence state of the agents contributing to the majority-voting.

sequence is $\hat{\mathcal{A}}_i$,

$$\begin{aligned}\hat{\mathcal{A}}_i &= (\hat{\boldsymbol{a}}_1, \hat{\boldsymbol{a}}_2, \ldots, \hat{\boldsymbol{a}}_N) \\ &= (\boldsymbol{a}_{\Psi_i(1)}, \boldsymbol{a}_{\Psi_i(2)}, \ldots, \boldsymbol{a}_{\Psi_i(N)}),\end{aligned} \tag{5}$$

where $\hat{\boldsymbol{a}}_j$ denotes the j-th sorted agent, and $\Psi_i(j)$ is the index in the sorted index of Ψ_i.

Majority-then-Stop Mechanism. In a voting mechanism containing N agents, the minimum number of agents required to vote to ensure the correctness of the outcome is $\lceil (N+1)/2 \rceil$, which can be treated as the *majority threshold*. Formally, we define the majority threshold as $\tau = \lceil (N+1)/2 \rceil$. Based on the above obtained ordered sequence $\hat{\mathcal{A}}_i$ of the query $\boldsymbol{q}_i$, all agents are queried sequentially. To obtain the final results with majority voting, we first query the first n agents ($\tau \leq n \leq N$), and obtain the partial response set,

$$\mathcal{Y}_{i,n} = \{y_{i,j}\}_{j=1}^{n}, \tag{6}$$

where $y_{i,j} = \mathcal{F}_{\hat{\boldsymbol{a}}_j}(\boldsymbol{q}_i)$ is the prediction produced by the j-th agent in the prioritized sequence.

Let $\mathcal{U}_{i,n}$ denote the set of unique answers in $\mathcal{Y}_{i,n}$. The voting process terminates once a majority consensus emerges,

$$\exists\, \hat{y} \in \mathcal{U}_{i,n} \quad \text{s.t.} \quad \sum_{j=1}^{n} \mathbb{I}\big[y_{i,j} = \hat{y}\big] \geq \tau. \tag{7}$$

If Eq. (7) is satisfied, the consensus answer is immediately returned as $\hat{y}_i$. Otherwise, the next agent in $\hat{\mathcal{A}}_i$ is queried until a majority is reached or all N agents are used, in which case the decision reduces to the standard majority vote in Eq. (1).

The main insight of EMS is that it treats the multi-agent voting process as a sequential problem, where agents vote one after another. The latter agent only reasons and votes after the former agent has finished voting, and adaptively decides whether the subsequent agents need to reason and vote based on the current voting results. Consequently, designing an appropriate multi-agent voting order becomes the key to this problem. As shown in Figure 2, the proposed Efficient Majority-then-Stopping (EMS)first calculates each agent's confidence score with Agent Confidence Modeling for each query to decide the voting order, and then performs incremental voting based on the adaptive voting order with Adaptive Incremental Voting. The Individual Confidence Updating is finally used to update the confidence state of the agent contributing to the voting.

3.2 Agent Confidence Modeling

Agent Confidence Modeling(ACM) is proposed to compute the confidence score $S_{i,j}$ by quantifying how reliably agent $\boldsymbol{a}_j$ is expected to answer query $\boldsymbol{q}_i$. Drawing inspiration from human voting processes, we have designed two confidence computation strategies for agents. One strategy is agent-centric, aiming to assign higher confidence to agents whose historical voting has consistently aligned with the outcome, named the *historical reliability*. The other strategy is problem-centric,

prioritizing voting by agents who possess expertise in the specific problem domain, named the *query-adaptive semantic reliability*.

The historical reliability $S^h_{i,j}$ measures how consistently agent $\boldsymbol{a}_j$ has aligned with the voting consensus across all prior queries, serving as a global indicator of the agent's overall reasoning quality. Formally, it calculates the proportion of correct votes in the voting history participated in by the agent,

$$S^h_{i,j} = \frac{c_j}{v_j}, \tag{8}$$

where c_j and v_j denote the correct and total number of agents $\boldsymbol{a}_j$ after (i-1) voting step, respectively.

Otherwise, the query-adaptive semantic reliability aims to measure each agent's potential ability to solve a query problem. Formally, for each agent $\boldsymbol{a}_j$, it maintains the semantic history buffer $\mathcal{H}_j$ to store the embeddings of queries that $\boldsymbol{a}_j$ previously agreed with the consensus. After that, the query-adaptive semantic reliability is the average similarity between the current query's embedding and all stored semantics in the history buffer $\mathcal{H}_j$,

$$S^q_{i,j} = \frac{1}{|\mathcal{H}_j|} \sum_{\mathbf{h} \in \mathcal{H}_j} \frac{\phi(\boldsymbol{q}_i) \cdot \mathbf{h}}{\|\phi(\boldsymbol{q}_i)\| \, \|\mathbf{h}\|}, \tag{9}$$

where $\phi(\cdot)$ is the pre-trained sentence encoder, and $|\mathcal{H}_j|$ denotes the number of set $\mathcal{H}_j$.

Once obtaining the historical reliability $S^h_{i,j}$ or the query-adaptive semantic reliability $S^q_{i,j}$ of all N agents, $\boldsymbol{S}^*_i = \{S^*_{i,j}\}^N_{j=1} (* \in \{h, q\})$ can be used to obtain the task-adaptive agent sequence via Eq. (4), and generate the sorted agents sequence,

$$\hat{\mathcal{A}}_i = (\hat{\boldsymbol{a}}_{i,1},\ \hat{\boldsymbol{a}}_{i,2},\ \ldots,\ \hat{\boldsymbol{a}}_{i,N}), \tag{10}$$

where $\hat{\boldsymbol{a}}_{i,1}$ is the agent with the highest confidence for query $\boldsymbol{q}_i$, followed by agents in decreasing order of expected proficiency. This ordering is then passed to the following Adaptive Incremental Voting stage.

3.3 Adaptive Incremental Voting

Since EMS treats the multi-agent voting process as a sequential problem, the Adaptive Incremental Voting (AIV) is proposed to conduct the voting process based on the previously obtained agent voting order $\hat{\mathcal{A}}_i$ for the candidate query $\boldsymbol{q}_i$. Drawing on the quorum principle of Byzantine Fault Tolerance (BFT), EMS firstly sets the initial quorum size to $\tau = \lceil (N+1)/2 \rceil$ to conduct the initial majority voting. By initiating the process with τ agents, EMS balances the need for a reliable initial consensus with the potential for significant computational savings.

Let $\hat{\mathcal{A}}_i = (\hat{\boldsymbol{a}}_{i,1}, \hat{\boldsymbol{a}}_{i,2}, \ldots, \hat{\boldsymbol{a}}_{i,N})$ denote the ordered list of N agents for query $\boldsymbol{q}_i$. These agents are arranged in a predefined order based on their reliability, as determined by the global confidence state Θ. The objective of AIV is to sequentially select agents from this list and aggregate their votes until a consensus is reached.

Initial Voting: The voting process begins with an initial subset of agents. Specifically, the first $\tau = \lceil (N+1)/2 \rceil$ agents are invoked to cast their votes. The reason for starting with τ agents is to satisfy the majority rule with the minimum number of agents, ensuring that the remaining $N - \tau$ agents cannot form a larger opposing bloc. Let $\mathcal{Y}_i = \{y_{i,(1)}, \ldots, y_{i,(\tau)}\}$ denote the answer set of query $\boldsymbol{q}_i$, where $\mathcal{U}_i$ denote the set of distinct answers in $\mathcal{Y}_i$. At the initial stage, the condition for achieving a majority consensus among the first τ agents is:

$$y_{i,1} = y_{i,2} = \cdots = y_{i,\tau}. \tag{11}$$

If the above condition holds, the process terminates immediately with $\hat{y}_i = y_{i,1}$, and outputs the agent sets $\mathbb{A}_i = \{\hat{\boldsymbol{a}}_{i,1}, \ldots\ \hat{\boldsymbol{a}}_{i,\tau}\}$ that contributing the majority-voting.

Incremental Voting: If the initial responses from Top-τ agents do not form a consensus, the process proceeds by incrementally adding more agents to the voting process. At the incremental voting of n-th step ($\tau + 1 \leq n \leq N$), the corresponding historical results set is defined as $\mathcal{Y}_i = \{y_{i,1}, \ldots, y_{i,n-1}\}$, and the voted agents as $\mathbb{A}_i = \{\hat{\boldsymbol{a}}_{i,1}, \ldots\ \hat{\boldsymbol{a}}_{i,n-1}\}$. Therefore, the Adaptive Incremental Voting invokes the next agent $\hat{\boldsymbol{a}}_{i,n}$ for majority-voting, and its response is added to the answer set,

$$\mathbb{A}_i = \mathbb{A}_i \cup \{\hat{\boldsymbol{a}}_{i,n}\}, \tag{12}$$

$$\mathcal{Y}_i = \mathcal{Y}_{i,n-1} \cup \{y_{i,n}\}, \tag{13}$$

where $y_{i,n}$ is the results generated by the invoked agent $\hat{\boldsymbol{a}}_{i,n}$.

At each step n, the updated set of answers $\mathcal{Y}_i$ is checked to justify whether a majority condition has been satisfied. Formally, AIV checks whether any answer set $\mathcal{Y}_i$ has accumulated at least τ votes, *i.e.,* whether the following condition is satisfied:

$$\exists y \in \mathcal{U}_i, \quad \sum_{j=1}^{n} \mathbb{I}[y_{i,j} = y] \geq \tau, \tag{14}$$

where $\mathbb{I}[A]$ is the indicator function that equals 1 if the condition A holds, and 0 otherwise.

If a majority consensus is reached after invoking n-th agent, the final answer is $\mathcal{Y}_i$ and the process terminates. If no majority is reached after all N agents have voted, the procedure degenerates to vanilla majority voting. In this case, EMS applies the plurality rule over $\mathcal{Y}_j$:

$$\hat{y}_i = \text{Plurality}(\mathcal{Y}_{i,N}), \tag{15}$$

and outputs the most frequent answer. Moreover, the invoked agent sets that contributing to the voting is $\mathbb{A}_i$.

3.4 Individual Confidence Updating

After that, the Individual Confidence Updating (ICU) is proposed to update the confidence states Θ, on which the agents are involved in voting. Formally, ICU updates the state $\Phi_j = (c_j, v_j, \mathcal{H}_j)$ for each agent $\boldsymbol{a}_j \in \mathbb{A}_i$ that participated in the voting on query $\boldsymbol{q}_j$. Formally, the query embedding can be put into the semantic memory when the agent answers correctly, following the same sliding-window policy of capacity. Moreover, the number of c_i and v_j is updated to add 1 for the agents who are involved in voting. Agents that were *not* invoked for $\boldsymbol{q}_i$ retain their previous state $\Phi_{i,j}$. The update rules are:

$$\begin{aligned} v_j &\leftarrow v_j + 1, \\ c_j &\leftarrow c_j + 1, \\ \mathcal{H}_j &\leftarrow \mathcal{H}_j \cup \{\phi(\boldsymbol{q}_i)\}, \quad \text{if } y_{i,j} = \hat{y}_i. \end{aligned} \tag{16}$$

4 Experiments

4.1 Experimental Setup

Datasets. We evaluate the performance of EMS across six benchmarks: 1) Mathematical Reasoning: *AQuA* (Ling et al., 2017), *Math500* (Hendrycks et al., 2021b), and *GSM8K* (Cobbe et al., 2021). 2) General Knowledge: *MMLU* (Hendrycks et al., 2021a), *GPQA Diamond* (Rein et al., 2024), *CommonsenseQA* (Talmor et al., 2019). All datasets utilize standard settings.

Agent Configuration. Our framework utilizes a heterogeneous pool of $N = 9$ large language models to ensure diverse reasoning perspectives, which includes: 1) *OpenAI* (GPT-4.1, GPT-4o, GPT-4o-mini); 2) *Google* (Gemini-1.5-pro, Gemini-2.5-flash); 3) *Anthropic* (Claude-3-5-Sonnet); 4) *DeepSeek* (DeepSeek-V3); 5) *Meta* (Llama-3.1-70B-Instruct); and 6) *Alibaba* (Qwen2.5-32B-Instruct). All model inferences are conducted via the API aggregation service provided by Yunwu API platform (`https://yunwu.ai/`). For semantic query feature extraction, we employ the *paraphrase-multilingual-MiniLM-L12-v2* encoder to map queries into a shared latent space for similarity-based retrieval.

Baselines. We compare EMS instantiated with different confidence strategy, namely *EMS-Rel* and *EMS-Sim*, against the following baselines: 1) *Individual Baselines* include Single Agent with Chain-of-Thought (CoT) (Wei et al., 2022) using the top-performing model and Self-Consistency (SC) (Wang et al., 2023) via repeated sampling. 2) *Voting Baselines* include the Simple Majority Voting (Simple MV) using the full agent pool, and Weighting methods comprise Historical-based Weighted MV(Li et al., 2024) and a Soft-Weighted MV(Conf-MV)(Xiong et al., 2024), where agent votes are weighted by their self-reported confidence scores. 3) *EMS-Rel* and *EMS-Sim* use the historical reliability and query-adaptive semantic reliability, respectively.

Evaluation Metrics. Average Accuracy measures the correctness of the final consensus, denoted as "*Avg.Acc.*". Moreover, we measure the efficiency of a multi-agent system by the *Average Number of Invoked Agents*, defined as the mean number of agents actually called to produce responses per query. Formally, given a query set $\mathcal{Q}$, the metric is computed as

$$\text{Avg.\#Agents} = \frac{1}{|\mathcal{Q}|} \sum_{q_i \in \mathcal{Q}} \mathcal{L}(\boldsymbol{q}_i), \tag{17}$$

where $\mathcal{L}(\boldsymbol{q}_i)$ denotes the number of agents invoked for query $\boldsymbol{q}_i$. This metric serves as a proxy for inference cost, as each agent invocation typically corresponds to one LLM call.

4.2 Main Results

Table 1 presents a comprehensive comparison between the proposed EMS framework and existing methods across six benchmarks. First, all majority-voting approaches consistently outperform single-agent inference, highlighting the effectiveness of multi-agent aggregation. Importantly, both EMS variants largely preserve the accuracy of the full ensemble, with the best variant achieving 86.38%, only slightly below the highest accuracy

Table 1: Comparison on six benchmarks. “Avg.Acc." (%) and efficiency metrics are reported. “Avg.#Agents" denotes the mean number of agents actually called to produce responses per query.

Methods	GSM8K	MATH500	GPQA	CSQA	AQuA	MMLU	Avg. Acc.	Avg.#Agents
Single Agent	95.72	80.40	64.65	84.95	87.66	90.11	83.92	1.00
Self-Consistency	96.28	82.00	66.16	85.88	88.52	90.63	84.91	10.00
Simple MV	97.04	84.80	68.69	87.47	90.16	90.09	86.38	9.00
Weighted MV	97.12	85.00	69.19	87.63	90.23	90.47	86.61	9.00
Conf-MV	96.96	83.60	67.17	87.24	89.92	89.86	85.79	9.00
EMS-Sim	97.04	84.80	68.69	87.47	90.16	90.09	86.38	6.19
EMS-Rel	97.04	84.80	68.69	87.47	90.16	90.09	86.38	**6.10**

of 86.61% obtained by *Simple MV*. Meanwhile, EMS substantially reduces computational cost. For instance, EMS-Rel and EMS-Sim decrease the average number of agent invocations from 9.00 to 6.10 and 6.19, respectively. These results indicate that confidence-guided agent ordering enables the system to reach a stable consensus early, thereby eliminating redundant reasoning while maintaining high decision quality. Overall, EMS achieves performance comparable to the *Simple MV* baseline while significantly reducing the number of agent calls through its dynamic, confidence-driven early-stopping mechanism.

4.3 Ablation Study

In this section, we give a comprehensive analysis of the proposed method to verify its effectiveness.

Effect of Early Stopping. The core insight of EMS is that an early stopping mechanism based on dynamic consensus checking can eliminate redundant computations without compromising reasoning quality. To validate this hypothesis, we compare our dynamic agent ordering with several heuristic baselines. Specifically, *Random ES* randomly shuffles the agent order while applying consensus-based early stopping, whereas *Fixed Random-5* simply selects five random agents without any consensus check. As shown in Table 2, *Random ES* achieves an average accuracy of 86.38%, identical to the *Full MV* baseline, while reducing the average number of invoked agents (*Avg.#Agents*) from 9.0 to 6.31 (nearly 30% reduction). However, naive truncation without consensus verification can be harmful. The *Fixed Random-5* strategy, which lacks a consensus check, suffers a substantial performance degradation, with the average accuracy dropping to 84.37%. These results highlight that consensus-based stopping is a critical safeguard for preserving the performance ceiling while improving computational efficiency.

Table 2: Effect of different Ordering and Stopping Strategies.

Methods	Avg. Acc.	Avg.#Agents
Full MV	86.38	9.00
Random ES	86.38	6.52
Fixed Random-5	84.37	5.00
Fixed Top-5	85.26	5.18
EMS-Sim	86.38	6.19
EMS-Rel	86.38	6.10

Effect of Agent Sorting Strategies. The ordering of agents is a critical factor in determining which agents are queried first during the voting process. We therefore analyze the impact of different sorting strategies on the efficiency of the voting procedure, and summarize the related results in Table 2. Compared with *Random ES*, which relies on a random ordering of agents, *EMS-Sim* prioritizes agents according to their query-adaptive semantic reliability. This strategy reduces the average number of invoked agents (*Avg.#Agents*) from 6.52 to 6.19, indicating that a more informed agent ranking can effectively lower the number of agent calls in multi-agent voting. This observation highlights the importance of reliability-aware scheduling, which forms a key component of the proposed EMS framework. Furthermore, *EMS-Rel*, which prioritizes agents based on historical reliability, achieves an even lower average of 6.10 invoked agents (*Avg.#Agents*), demonstrating the additional efficiency gains obtainable through stronger reliability signals.

Analysis of the Adaptive Incremental Voting. The core idea of EMS is that *Adaptive Incremental Voting* (AIV) controls the voting process through a dynamic consensus check, enabling early termination once a stable majority emerges and thereby avoiding unnecessary reasoning. To better understand this mechanism, we analyze the distribu-

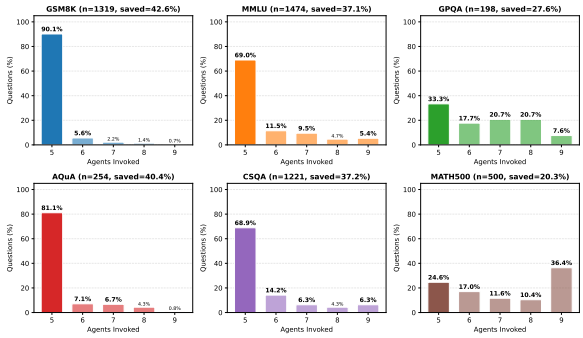

Figure 3: Analysis of the Adaptive Incremental Voting. Each bar represents the proportion of questions that achieve the final results based on the "Agents Invoked".

tion of the number of agents invoked per query across different benchmarks. As shown in Figure 3, under a majority-voting setup with 9 agents, the *Initial Voting* stage—consisting of the first five agents—already reaches consensus for a large portion of queries on relatively easier benchmarks. For instance, consensus is achieved at this stage for 81.1% and 90.1% of the queries on GSM8K and AQuA, respectively, while the corresponding proportions are 68.9% on CSQA and 69.0% on MMLU. In contrast, MATH500 and GPAQ exhibit the lowest consensus rates during the initial stage, indicating that these more challenging tasks require additional agents to reach agreement. Overall, the *Initial Voting* stage with five agents substantially reduces the number of agent invocations, particularly for relatively simple tasks. After this stage, the *Incremental Voting* procedure progressively incorporates additional agents only when necessary. Consequently, the proportion of queries requiring many agents decreases rapidly over time; for most benchmarks, only a small fraction of queries require all N agents to complete the majority voting process, with MATH500 being a notable exception. These observations demonstrate that *Adaptive Incremental Voting* provides an effective strategy for achieving efficient majority voting while preserving decision quality.

Sensitivity Analysis of Agent Pool Size N To investigate the scalability and upper-bound performance of EMS, we evaluate its behavior under different agent pool sizes $N \in \{5, 7, 9, 11, 13\}$ on the MMLU and GPQA benchmarks. The results are summarized in Table 3. As the candidate pool size N increases, we observe a consistent improvement in reasoning accuracy, accompanied by a gradual increase in the average number of agents participating in the voting process. However, the marginal benefit of enlarging the agent pool diminishes as N grows. For example, on the GPQA dataset, increasing N from 9 to 13 requires invoking an additional 2.93 agents on average, yet yields only a modest accuracy improvement of 1.12%. This observation suggests that continually increasing the number of agents provides limited returns and may not be an efficient strategy for multi-agent voting. Furthermore, we observe that as N increases, the average agent call rate of EMS remains relatively stable, indicating that the proposed early-stopping mechanism effectively controls the growth of inference cost even when the agent pool becomes larger.

Table 3: Performance scaling with respect to the maximum agent pool size N. Accuracy (%) and "Avg.#Agents" are reported for MMLU and GPQA.

N	MMLU		GPQA	
	Acc. (%)	Avg.#Agents	Acc. (%)	Avg.#Agents
5	88.50	3.12	64.20	3.58
7	89.65	4.21	67.45	5.43
9	90.09	5.12	68.69	6.12
11	90.25	6.18	69.52	7.84
13	**90.31**	7.34	**69.81**	9.05

5 Conclusion

Majority voting is the standard for aggregating multi-agent responses into a final decision. To address the inefficiency of conventional majority voting in multi-agent LLM systems, we propose the Efficient Majority-then-Stopping (EMS) framework, which prioritizes agents based on their task-aware reliability and terminates the reasoning process once a majority consensus is reached. EMS integrates three key components to enable efficient and reliable decision making: Agent Confidence Modeling (ACM), Adaptive Incremental Voting (AIV), and Individual Confidence Updating (ICU). Extensive experiments across multiple benchmarks demonstrate that EMS significantly reduces inference costs in multi-agent voting while maintaining competitive accuracy.

Notably, the efficiency of EMS critically depends on the quality of the agent ordering. While the current framework leverages relatively simple historical and semantic signals to construct the voting sequence, designing more advanced and expressive agent scheduling strategies remains an important direction for future work.

Limitations

Despite the effectiveness of EMS, our work has two key limitations. (1) The evaluation is merely conducted on the mathematical reasoning and general knowledge; more challenging and field tasks can be applied to verify the generalization of the proposed method. (2) The essence of this algorithm is to improve efficiency by reducing the votes of unnecessary agents. However, based on human voting experience, the choice of voting agents is actually a very complex issue, which is also a key bottleneck.

Ethical Considerations

The proposed Efficient Majority-then-Stopping is an algorithm for general reasoning tasks, and the algorithm itself does not have any ethical risks.

References

Chi-Min Chan, Weize Chen, Yusheng Su, Jianxuan Yu, Wei Xue, Shanghang Zhang, Jie Fu, and Zhiyuan Liu. 2023. Chateval: Towards better llm-based evaluators through multi-agent debate. *arXiv preprint arXiv:2308.07201*.

Weize Chen, Yusheng Su, Jingwei Zuo, Cheng Yang, Chenfei Yuan, Chi-Min Chan, Heyang Yu, Yaxi Lu, Yi-Hsin Hung, Chen Qian, Yujia Qin, Xin Cong, Ruobing Xie, Zhiyuan Liu, Maosong Sun, and Jie Zhou. 2024. Agentverse: Facilitating multi-agent collaboration and exploring emergent behaviors. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Hyeong Kyu Choi, Xiaojin Zhu, and Sharon Li. 2025. Debate or vote: Which yields better decisions in multi-agent large language models? *arXiv preprint arXiv:2508.17536*.

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. *CoRR*, abs/2110.14168.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B Tenenbaum, and Igor Mordatch. 2023. Improving factuality and reasoning in language models through multiagent debate. In *Forty-first International Conference on Machine Learning*.

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

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. *arXiv preprint arXiv:2402.01680*.

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*.

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. *arXiv preprint arXiv:2103.03874*.

Bowen Jiang, Yangxinyu Xie, Zhuoqun Hao, Xiaomeng Wang, Tanwi Mallick, Weijie J Su, Camillo Jose Taylor, and Dan Roth. 2024. A peek into token bias: Large language models are not yet genuine reasoners. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 4722–4756.

Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. 2023. Llm-blender: Ensembling large language models with pairwise ranking and generative fusion. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14165–14178.

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*. Association for Computational Linguistics.

Yubin Kim, Ken Gu, Chanwoo Park, Chunjong Park, Samuel Schmidgall, A. Ali Heydari, Yao Yan, Zhihan Zhang, Yuchen Zhuang, Mark Malhotra, Paul Pu Liang, Hae Won Park, Yuzhe Yang, Xuhai Xu, Yilun Du, Shwetak N. Patel, Tim Althoff, Daniel McDuff, and Xin Liu. 2025. Towards a science of scaling agent systems. *CoRR*, abs/2512.08296.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Leslie Lamport, Robert E. Shostak, and Marshall C. Pease. 1982. The byzantine generals problem. *ACM Trans. Program. Lang. Syst.*, 4(3):382–401.

Huao Li, Yu Chong, Simon Stepputtis, Joseph P Campbell, Dana Hughes, Charles Lewis, and Katia Sycara. 2023. Theory of mind for multi-agent collaboration via large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 180–192.

Jiawei Li, Yang Gao, Yizhe Yang, Yu Bai, Xiaofeng Zhou, Yinghao Li, Huashan Sun, Yuhang Liu, Xingpeng Si, Yuhao Ye, Yixiao Wu, Yiguan Lin, Bin Xu, Ren Bowen, Chong Feng, and Heyan Huang. 2026. Fundamental capabilities and applications of large language models: A survey. *ACM Comput. Surv.*, 58(2):38:1–38:42.

Junyou Li, Qin Zhang, Yangbin Yu, Qiang Fu, and Deheng Ye. 2024. More agents is all you need. *Trans. Mach. Learn. Res.*, 2024.

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.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. Program induction by rationale generation: Learning to solve and explain algebraic word problems. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 158–167.

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. *CoRR*, abs/2409.14051.

Iman Mirzadeh, Keivan Alizadeh, Hooman Shahrokhi, Oncel Tuzel, Samy Bengio, and Mehrdad Farajtabar. 2024. Gsm-symbolic: Understanding the limitations of mathematical reasoning in large language models. *arXiv preprint arXiv:2410.05229*.

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 *First Conference on Language Modeling*.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158.

Xingchen Wan, Ruoxi Sun, Hanjun Dai, Sercan Arik, and Tomas Pfister. 2023. Better zero-shot reasoning with self-adaptive prompting. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 3493–3514.

Siyuan Wang, Zhuohan Long, Zhihao Fan, Xuanjing Huang, and Zhongyu Wei. 2025. Benchmark self-evolving: A multi-agent framework for dynamic LLM evaluation. In *Proceedings of the 31st International Conference on Computational Linguistics, COLING 2025, Abu Dhabi, UAE, January 19-24, 2025*, pages 3310–3328. Association for Computational Linguistics.

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*.

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 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Shaokun Zhang, Erkang Zhu, Beibin Li, Li Jiang, Xiaoyun Zhang, and Chi Wang. 2023. Autogen: Enabling next-gen LLM applications via multi-agent conversation framework. *CoRR*, abs/2308.08155.

Miao Xiong, Zhiyuan Hu, Xinyang Lu, Yifei Li, Jie Fu, Junxian He, and Bryan Hooi. 2024. Can llms express their uncertainty? an empirical evaluation of confidence elicitation in llms. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Yanwei Yue, Guibin Zhang, Boyang Liu, Guancheng Wan, Kun Wang, Dawei Cheng, and Yiyan Qi. 2025. Masrouter: Learning to route llms for multi-agent systems. *arXiv preprint arXiv:2502.11133*.

Qian Zhang, Yan Zheng, Jinyi Liu, Hebin Liang, and Lanjun Wang. 2025. Key decision-makers in multi-agent debates: Who holds the power? *CoRR*, abs/2511.11040.

Xiutian Zhao, Ke Wang, and Wei Peng. 2024. An electoral approach to diversify llm-based multi-agent collective decision-making. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, EMNLP 2024, Miami, FL, USA, November 12-16, 2024*, pages 2712–2727. Association for Computational Linguistics.