Ranked Voting based Self-Consistency of Large Language Models

Weiqin Wang, Yile Wang*, Hui Huang
College of Computer Science and Software Engineering, Shenzhen University
here1swqw@gmail.com, wangyile@szu.edu.cn, hhzhiyan@gmail.com

Abstract

Majority voting is considered an effective method to enhance chain-of-thought reasoning, as it selects the answer with the highest "self-consistency" among different reasoning paths (Wang et al., 2023). However, previous chain-of-thought reasoning methods typically generate only a *single answer* in each trial, thereby ignoring the possibility of other potential answers. As a result, these alternative answers are often overlooked in subsequent voting processes. In this work, we propose to generate *ranked answers* in each reasoning process and conduct *ranked voting* among multiple ranked answers from different responses, thereby making the overall self-consistency more reliable. Specifically, we use three ranked voting methods: Instant-runoff voting, Borda count voting, and mean reciprocal rank voting. We validate our methods on six datasets, including three multiple-choice and three open-ended question-answering tasks, using both advanced open-source and closed-source large language models. Extensive experimental results indicate that our proposed method outperforms the baselines, showcasing the potential of leveraging the information of ranked answers and using ranked voting to improve reasoning performance. The code is available at https://github.com/szu-tera/RankedVotingSC.

1 Introduction

Large language models (LLMs) have shown strong performance in recent years (Ouyang et al., 2022; OpenAI, 2023; Dubey et al., 2024; Yang et al., 2024; DeepSeek-AI, 2024). Chain-of-thought prompting (Wei et al., 2022) further improves the performance of LLMs in commonsense (Talmor et al., 2019) and mathematical (Cobbe et al., 2021) reasoning tasks. Building on these advancements, Wang et al. (2023) propose a majority voting based self-consistency approach, which leverages multiple reasoning paths through sampling to identify the most self-consistent answer, thereby improving the reasoning performance of LLMs.

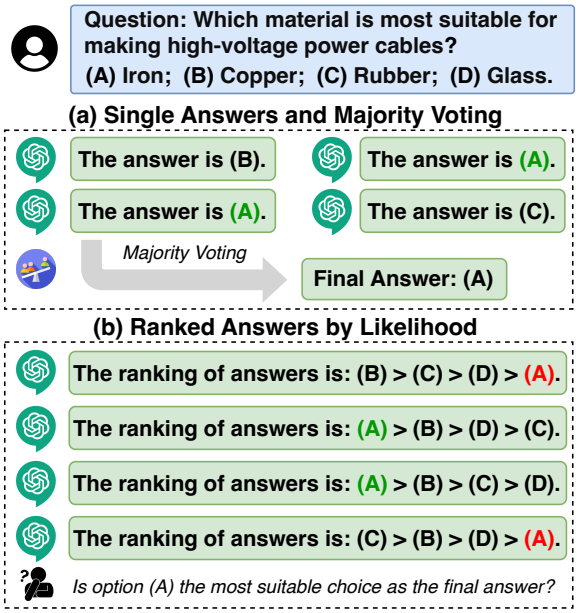

Figure 1: Example of (a) majority voting based self-consistency among single answers (Wang et al., 2023) and (b) ranked answers in four responses by models.

An example of majority voting based self-consistency is shown in Figure 1(a). In four responses by the model, option (A) was answered twice, (B) and (C) were each answered once, therefore (A) is considered the answer with the highest "self-consistency". In each response, the model replies with only one option, thereby omitting the possibility and priority of other options, which may introduce biases in the following majority voting process. In this work, we consider obtaining ranked answers instead of only a single answer in each response and employ ranked voting based self-consistency from multiple ranked answers, which we hope can lead to more reliable self-consistency and better reasoning performance.

*Corresponding author.

Findings of the Association for Computational Linguistics: ACL 2025, pages 14410–14426
July 27 - August 1, 2025 ©2025 Association for Computational Linguistics

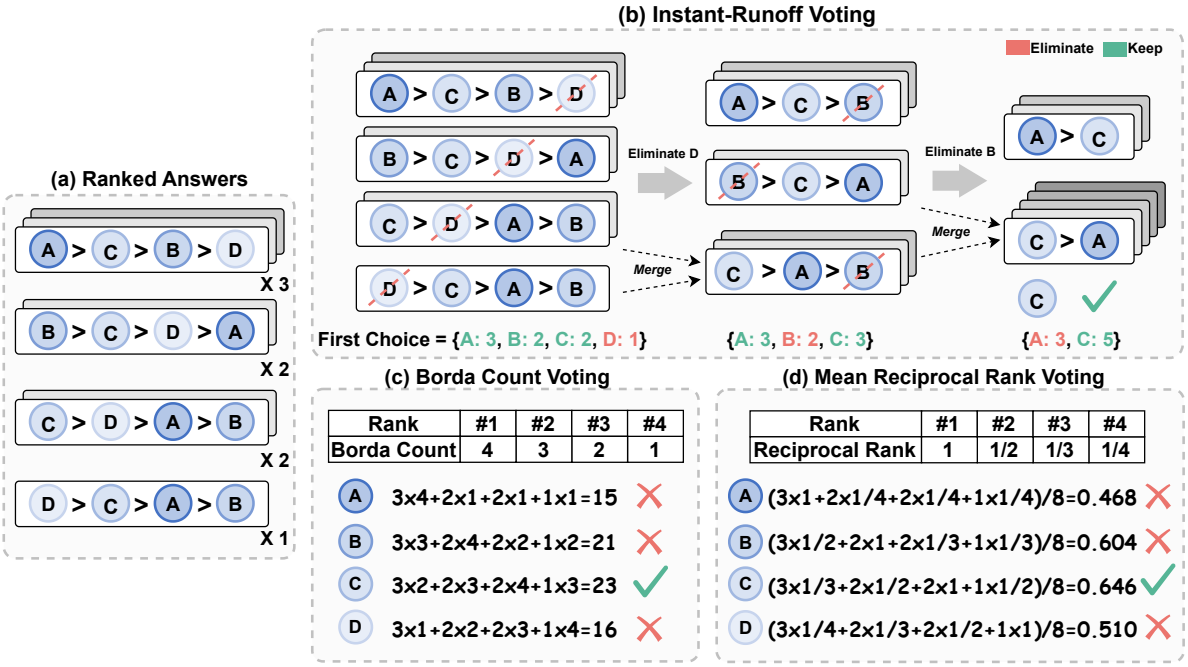

Figure 2: Examples of the procedures for three ranked voting methods. (a) The obtained ranked answers ($k = 8$). (b) Instant-runoff voting. (c) Borda count voting. (d) Mean reciprocal rank voting.

We show the example of ranked answers in Figure 1(b). The model generates the possibility ranking of all options each time. In this case, we find that option (B) ranks in the top two positions in all four responses, while option (A) ranks last in two responses. Considering the overall results, option (A) may not be the best suitable final answer.

To further decide the best answer according to the ranking information, we attempt to use ranked voting to determine the most appropriate answer based on the obtained ranked answers in responses. Ranked voting has been widely used in election systems. Specifically, we use three ranking voting methods to facilitate the final answer, including instant-runoff voting (Cary, 2011), Borda count voting (Emerson, 2013), and mean reciprocal rank voting. The first two methods are widely used in elections, while the latter one is related with the ranking-based MRR metric. We consider these ranked voting based approaches, compared with majority voting on single answers, can provide more reliable final answer by leveraging the ranking information of ordered answers.

We validate our ranked voting based self-consistency by using advanced LLMs, including four open-source models LLaMa-3 (Dubey et al., 2024), Qwen-2.5 (Yang et al., 2024), Gemma-2 (Team Gemma et al., 2024), Phi-3 (Abdin et al., 2024), and two closed-source models GPT-3.5 (OpenAI, 2022) and GPT-4 (OpenAI, 2023). Empirical results show that our method consistently outperforms baselines on three multiple-choice and three open-ended question-answering datasets.

2 Method

We first introduce the majority voting based self-consistency baseline (§2.1) and propose obtaining the ranked answers in each LLMs' response (§2.2). Then we provide detailed descriptions to three ranked voting methods as shown in Figure 2, including instant-runoff voting (§2.3), Borda count voting (§2.4), and mean reciprocal rank voting (§2.5). In the final we briefly present overall comparison and how to handle tie votes (§2.7).

2.1 Majority Voting based Self-Consistency

Given a model $\mathcal{M}$ and question $\mathcal{Q}$, following chain-of-thought reasoning (Wei et al., 2022), we have

$$\mathcal{R}, \mathcal{A} = \mathcal{M}(\mathcal{Q}_1, r_1, a_1 ..., \mathcal{Q}_n, r_n, a_n, \mathcal{Q}), \quad (1)$$

where $\{\mathcal{Q}_j, r_j, a_j\}_{j=1}^{n}$ are questions, reasoning paths, and answers in few-shot examples, respectively. $\mathcal{R}$ and $\mathcal{A}$ are generated reasoning path and answer of the question $\mathcal{Q}$.

Wang et al. (2023) further propose finding the most "consistent" answer among multiple responses. In particular, by setting a high decoding temperature (e.g., $\tau = 0.7$), the final answer is the

Multiple-Choice Question-Answering

/* Few-Shot Examples */
Question: [...] ($\mathcal{Q}_j$)
Answer: [...] ($a_j^{r_1} \succ a_j^{r_2} \succ a_j^{r_3} \succ a_j^{r_4}$)
/* Test Question */
Question: George wants to warm his hands quickly by rubbing them. Which skin surface will produce the most heat? (A) dry palms. (B) wet palms. (C) palms covered with oil. (D) palms covered with lotion. ($\mathcal{Q}$)
Answer: Heat is generated by friction when two surfaces rub together. The amount of friction depends on the dryness and texture of the surfaces. Dry palms have [...]. Wet palms reduce [...]. Palms covered with oil lower [...], and palms covered with lotion reduce [...]. The ranking of options by likelihood is: A > B > D > C. ($\mathcal{A}^{r_1} \succ \mathcal{A}^{r_2} \succ \mathcal{A}^{r_3} \succ \mathcal{A}^{r_4}$)

Open-Ended Question-Answering

/* Few-Shot Examples */
Question: [...] ($\mathcal{Q}_j$)
Answer: [...] ($a_j^{r_1} \succ a_j^{r_2} \succ a_j^{r_3} \succ a_j^{r_4} \succ a_j^{r_5}$)
/* Test Question */
Question: The language used in Cameroon is ($\mathcal{Q}$)
Answer: Cameroon is a multilingual country located in Central Africa. However, the official languages of the country are French and English. While many indigenous languages are also spoken, they do not have the same status as the official languages. The ranking of answers by likelihood is: French > English > Pidgin English > Fulfulde > Hausa. ($\mathcal{A}^{r_1} \succ \mathcal{A}^{r_2} \succ \mathcal{A}^{r_3} \succ \mathcal{A}^{r_4} \succ \mathcal{A}^{r_5}$)

Table 1: The examples to obtain ranked answers $\mathcal{A}^r$ (colored in blue) on multiple-choice and open-ended question-answering tasks in few-shot settings.

majority voting results across k possible solutions $\{\mathcal{R}_i, \mathcal{A}_i\}_{i=1}^k$ in totally k responses:

$$\mathcal{A}_{\text{final}}^{\text{majority}} = \text{argmax}_{\mathcal{A}} \sum_{i=1}^{k} \mathbb{1}(\mathcal{A}_i = \mathcal{A}). \quad (2)$$

2.2 From Single Answer to Ranked Answers

In Eq. 1, $\mathcal{A}$ usually indicates a *single answer* such as a specific option in multiple-choice question-answering. However, it is difficult to reflect the possibility of other options from the single answer $\mathcal{A}$. Thus, we consider obtaining the *ranked answers* $\mathcal{A}^r$, which contains multiple ranked candidates according to the preference of LLMs:

$$\mathcal{R}, \mathcal{A}^r = \mathcal{M}(\mathcal{Q}_1, r_1, a_1^r ..., \mathcal{Q}_n, r_n, a_n^r, \mathcal{Q}), \quad (3)$$

Algorithm 1 Instant-Runoff Voting

Input: Ranked answers ANS $= \{\mathcal{A}_1^r, ..., \mathcal{A}_k^r\}$
Output: Final answer WINNER
while True **do**
  CURRWINNER = MostFirstChoice(ANS)
  if Count(CURRWINNER) $> k/2$ **then**
    return CURRWINNER
  end if
  for j in $1, ..., k$ **do**
    $\mathcal{A}_j^r$ = EliminateTheLastOne($\mathcal{A}_j^r$)
  end for
  ANS $= \{\mathcal{A}_1^r, ..., \mathcal{A}_k^r\}$
end while

where $\{\mathcal{Q}_j, r_j, a_j^r\}_{j=1}^n$ are questions, reasoning paths, and ranked answers in few-shot examples, respectively. $\mathcal{R}$ and $\mathcal{A}^r$ are generated reasoning path and ranked answers of the question $\mathcal{Q}$. The ranked answers $\{a_j^r\}_{j=1}^n$ and $\mathcal{A}^r$ includes m ranked candidate answers:

$$\begin{aligned} a_j^r &= a_j^{r_1}, a_j^{r_2}, ..., a_j^{r_m}, \\ \mathcal{A}^r &= \mathcal{A}^{r_1}, \mathcal{A}^{r_2}, ..., \mathcal{A}^{r_m}, \end{aligned} \quad (4)$$

where $a_j^{r_1} \succ a_j^{r_2} \succ ... \succ a_j^{r_m}$ and $\mathcal{A}^{r_1} \succ \mathcal{A}^{r_2} \succ ... \succ \mathcal{A}^{r_m}$ indicate that $a_j^{r_1}$ or $\mathcal{A}^{r_1}$ is the most possible answer, $a_j^{r_m}$ or $\mathcal{A}^{r_m}$ is the least possible answer in corresponding responses, respectively.

Examples of Ranked Answers. For obtaining the ranked answers $\mathcal{A}^r$, we design the demonstrations in few-shot settings and show two examples in Table 1 on both multiple-choice and open-ended question-answering scenarios.

Ranked Voting. Instead of *majority voting*, we leverage the information of ranked answers and get the final answer according to k possible ranked solutions $\{\mathcal{A}_i^r\}_{i=1}^k = \{\mathcal{A}_i^{r_1}, \mathcal{A}_i^{r_2}, ..., \mathcal{A}_i^{r_m}\}_{i=1}^k$:

$$\mathcal{A}_{\text{final}}^{\text{ranked}} = \text{RANKVOTE}(\mathcal{A}_1^r, \mathcal{A}_2^r, ..., \mathcal{A}_k^r), \quad (5)$$

where RANKVOTE indicates three ranked voting methods we used, as shown in Figure 2. We describe them in detail in the following subsections.

2.3 Instant-Runoff Voting

Instant-runoff voting (IRV) is a voting system that allows voters to rank candidates in order of preference (Cary, 2011). The main idea of IRV is to eliminate the candidate with the fewest votes in each round until a candidate receives a majority (more than 50%) of the votes. In a situation where the number of votes is relatively balanced, this voting

method can determine the most suitable candidate through multiple rounds of selection. We provide the IRV procedure in Algorithm 1 and a specific example below.

For example, consider the following ranked answers: "$a \succ b \succ c$" appearing on 3 responses, "$b \succ c \succ a$" appearing on 2 responses, and "$c \succ a \succ b$" appearing on 3 responses, respectively. In the initial round, candidate "a" and "c" each secure 3 first-choice votes (37.5%), while "b" garners only 2 first-choice votes (25%). Since no candidate achieves a majority of votes, the candidate "b" is eliminated due to the fewest first-choice votes. The second-choice votes from the responses that originally ranked "b" first are redistributed to "c", resulting in "c" accumulating a total of 5 first-choice votes (62.5%). Meanwhile, "a" retains its 3 first-choice votes (37.5%). Consequently, "c" emerges as the winner with a clear majority.

2.4 Borda Count Voting

Borda count voting (BCV) is a positional voting rule that gives each candidate a number of points (i.e., Borda count) based on their ranking (Emerson, 2013). Suppose we have m ranked answers $\mathcal{A}^{r_1}, \mathcal{A}^{r_2}, ..., \mathcal{A}^{r_m}$, the Borda count for candidate $\mathcal{A}$ is calculated as follows:

$$\text{BordaCount}(\mathcal{A}) = \sum_{i=1}^{k} (m - \text{rank}_{\mathcal{A}}(\mathcal{A}_i^r) + 1), \quad (6)$$

where $\text{rank}_{\mathcal{A}}(\mathcal{A}_i^r)$ is the rank of candidate $\mathcal{A}$ in the ranked answers $\mathcal{A}_i^r = \mathcal{A}_i^{r_1}, \mathcal{A}_i^{r_2}, ..., \mathcal{A}_i^{r_m}$ from the i-th response. For example, for ranked answers "$a \succ b \succ c$" and "$b \succ c \succ a$", the Borda count for candidates "a", "b", and "c" are 3+1=4, 2+3=5, and 1+2=3, respectively. The final answer $\mathcal{A}_{\text{final}}^{\text{BCV}}$ is the select candidate with the largest Borda count:

$$\mathcal{A}_{\text{final}}^{\text{BCV}} = \text{argmax}_{\mathcal{A}} \text{BordaCount}(\mathcal{A}). \quad (7)$$

2.5 Mean Reciprocal Rank Voting

The mean reciprocal rank (MRR) is a metric used in information retrieval to evaluate the effectiveness of search algorithms. For a sample of queries, it is calculated as the average of the multiplicative inverse of the rank of the first correct candidate: 1 for the first place, 1/2 for the second place, etc. Inspired by such rank-aware metrics, we consider evaluating the answer according to its place in multiple ranked answers, and obtain the final answer according to the MRR scores in k responses, which we called mean reciprocal rank voting (MRRV). In particular, for a candidate answer a, we calculate the corresponding MRR score as follows:

$$\text{MRR}(\mathcal{A}) = \frac{1}{k} \sum_{i=1}^{k} \frac{1}{\text{rank}_{\mathcal{A}}(\mathcal{A}_i^r)}, \quad (8)$$

where $\text{rank}_{\mathcal{A}}(\mathcal{A}_i^r)$ is the rank of candidate $\mathcal{A}$ in the ranked answers $\mathcal{A}_i^r = \mathcal{A}_i^{r_1}, \mathcal{A}_i^{r_2}, ..., \mathcal{A}_i^{r_m}$ from the i-th response. For example, for ranked answers "$a \succ c \succ b$" and "$b \succ a \succ c$" in two responses, the MRR scores for candidates "a", "b", and "c" are $\frac{1}{2}(\frac{1}{1}+\frac{1}{2})$=0.75, $\frac{1}{2}(\frac{1}{3}+\frac{1}{1})$=0.66, $\frac{1}{2}(\frac{1}{2}+\frac{1}{3})$=0.42, respectively. MRR score of 1.00 indicates that the corresponding candidate is always ranked first. The final answer $\mathcal{A}_{\text{final}}^{\text{MRRV}}$ is the selected candidate with the highest MRR score:

$$\mathcal{A}_{\text{final}}^{\text{MRRV}} = \text{argmax}_{\mathcal{A}} \text{MRR}(\mathcal{A}). \quad (9)$$

2.6 Construction of Few-Shot Examples

Constructing few-shot examples for our method is straightforward. The primary criterion we follow is to ensure strong semantic relevance between the question and its relevant candidates. To demonstrate how to construction an examples, we show one example in Table 2. A high-quality example should not only justify the correct answer, but also assess the plausibility of the other candidates in addressing the question, as well as their semantic relevance to it. We refer to such an example as a template. Once created, the template can be reused to construct additional examples for other questions. To scale up the few-shot example set, we begin by using existing examples as few-shot, and

An Example Prompt for Constructing Few-Shot
Question: What home entertainment equipment requires cable? (A) radio shack (B) substation (C) television (D) cabinet **Answer:** The most likely answer is a television, as it commonly requires cable for broadcasting or streaming services. A substation and radio shack are unrelated to home entertainment equipment, and a cabinet is purely for storage, not requiring a cable for its function. The most likely answer is (C). The ranking of options by likelihood is: C > B > A > D.

Table 2: A few-shot example for question answering, showing the rationale and a candidate ranking based on semantic relevance (explanation highlighted in blue).

then leverage large language models to automatically generate additional demonstrations, requiring only minimal human verification.

2.7 Overall Comparison and Tiebreaker Rule

Comparison of Methods. The ranked voting methods IRV, BCV, and MRRV are different from each other. Specifically, IRV can be regarded as an elimination-based ranked voting, while BCV and MRRV can be seen as ranked voting with different weighting schemes. In Figure 2, all three methods arrive at the same final answer (C). However, in practice, their results may vary due to the specific outcomes of ranked answers.

Tiebreaker Rules. Although ties are rare in voting outcomes, they may occur particularly for complex questions. Compared with self-consistency baseline, our ranking-based approach mitigates this issue by incorporating preference ranking information (see details in §4). To ensure rigor, we handle tie scenarios through confidence scoring for all methods, including the baseline: when multiple candidates receive equal votes, we compute each answer's confidence score $\mathcal{S}_i = \sum_{t=1}^{n} \log(p(\mathcal{C}_{i,t}))$, where $\mathcal{C}_{i,t}$ denotes the t-th token's predicted probability in the i-th candidate. The final selection chooses the candidate with the highest sum of logarithmic probabilities. For closed-source models, we defer to the model itself to select the best answer from the candidates.

3 Experiments

3.1 Settings

Datasets. We validate our method on six tasks across different domains. The three multiple-choice QA tasks include: **CommonsenseQA** (Talmor et al., 2019) is a multiple-choice question answering dataset that requires different types of commonsense knowledge. **ARC-Challenge** (Clark et al., 2018) includes genuine grade-school level, multiple-choice science questions. **AQUA-RAT** (Ling et al., 2017) is a multiple-choice QA dataset which involves five-options algebraic word problems with rationales. We also evaluate on three open-ended QA tasks form Big-bench (Srivastava et al., 2023): **WikiData** involves performing open-domain cloze-style question answering. **Date Understanding** aims to measure models' ability to understand date-related information. **Word Unscrambling** asks models to unscramble the given letters to form an English word. We list examples of each dataset in Appendix A.

Models. We compare with widely-used LLMs, including open-source models LLaMA-3 (Dubey et al., 2024), Qwen-2.5 (Yang et al., 2024), Gemma-2 (Team Gemma et al., 2024), and Phi-3 (Abdin et al., 2024), as well as closed-source GPT-3.5-turbo (OpenAI, 2022) and GPT-4-turbo (OpenAI, 2023) models. For open-source models, we use both lightweight (2B∼4B) and medium-sized (7B∼9B) models for comprehensive evaluations, and the checkpoints are listed in Appendix B.

Baseline Methods. We compare with the following baselines: **Few-Shot-CoT** (Wei et al., 2022) with chain-of-thought reasoning. **Best-of-N** (Stiennon et al., 2020) involves sampling multiple solutions and selecting the best one based on the scores of a fine-tuned Qwen2.5-7B (Yang et al., 2024) reward model. **Few-Shot-CoT-SC** (Wang et al., 2023) uses majority voting for aggregating multiple answers. **Adaptive-SC** (Aggarwal et al., 2023) uses dynamic number of responses instead of fixed ones.

Implementation Details. For all tasks, we use LM-evaluation-harness (Gao et al., 2024) for fair comparison. Following Wei et al. (2022), we set the number of few-shot examples $n = 8$ and decoding temperature $\tau = 0.7$ for baselines and our methods. Considering the budget constraints, we set the number of responses $k = 8$ in main experiments.

3.2 Main Results

2B∼4B Open-Source LLMs. The results are shown in Table 3. Overall, our method demonstrates improved average performance across various datasets and models, surpassing all baseline methods. Compared to the Few-Shot-CoT-SC, our three ranked voting methods achieve an average improvement of 3.32% with LLaMA-3.2-3B and 4.95% with Qwen-2.5-3B. Our method also outperforms Best-of-N, which relies on a reward model to enhance generation quality and incurs additional computational overhead.

Across different tasks, we achieve the largest improvement when performing data understanding tasks using Qwen-2.5-3B. Compared to Few-Shot-CoT-SC, the improvement can reach 10.84% to 12.46%. This suggests that, in contrast to providing a single answer for majority voting, offering multiple candidate answers along with their ranking and conducting ranked voting can offer more useful context.

We also find that the majority voting based self-

Method	Multiple-Choice QA (Accuracy)			Open-Ended QA (Exact Match)			Average
	CommonsenseQA	ARC-Challenge	AQUA-RAT	WikiData	Date Understanding	Word Unscrambling	
				(LLaMA-3.2-3B)			
Few-Shot-CoT	71.99	78.67	58.27	66.67	57.72	25.33	59.78
Best-of- N	74.61	80.63	67.72	70.83	63.69	26.50	64.00
Few-Shot-CoT-SC	73.46	80.54	61.81	73.67	60.98	24.33	62.47
Adaptive-SC	74.20	80.89	64.17	73.33	61.25	26.33	63.36
Instant-Runoff Voting	**74.86**$_{+1.40}$	81.31$_{+0.77}$	69.29$_{+7.48}$	74.00$_{+0.33}$	**64.23**$_{+3.25}$	27.67$_{+3.34}$	65.23$_{+2.76}$
Borda Count Voting	74.69$_{+1.23}$	80.55$_{+0.01}$	67.32$_{+5.51}$	**77.00**$_{+3.33}$	60.16$_{-0.82}$	28.33$_{+4.00}$	64.67$_{+2.20}$
Mean Reciprocal Rank Voting	74.45$_{+0.99}$	**81.40**$_{+0.86}$	**71.26**$_{+9.45}$	76.00$_{+2.32}$	62.60$_{+1.62}$	**29.00**$_{+4.67}$	**65.79**$_{+3.32}$
				(Qwen-2.5-3B)			
Few-Shot-CoT	77.15	71.42	70.87	69.00	42.55	6.67	56.28
Best-of-N	77.97	75.90	75.39	70.67	50.14	8.00	59.68
Few-Shot-CoT-SC	77.89	76.96	77.95	72.67	47.97	7.33	60.13
Adaptive-SC	77.48	78.07	77.17	72.67	48.78	7.67	60.31
Instant-Runoff Voting	**78.95**$_{+1.06}$	83.45$_{+6.49}$	79.13$_{+1.18}$	74.67$_{+2.00}$	**60.70**$_{+12.73}$	11.67$_{+4.34}$	64.76$_{+4.63}$
Borda Count Voting	78.38$_{+0.49}$	83.19$_{+6.23}$	**80.31**$_{+2.36}$	**76.33**$_{+3.66}$	58.81$_{+10.84}$	11.33$_{+4.00}$	64.72$_{+4.59}$
Mean Reciprocal Rank Voting	78.79$_{+0.90}$	**83.70**$_{+6.74}$	79.92$_{+1.97}$	75.33$_{+2.66}$	60.43$_{+12.46}$	**12.33**$_{+5.00}$	**65.08**$_{+4.95}$
				(Gemma-2-2B)			
Few-Shot-CoT	69.53	71.33	36.61	72.67	36.04	**24.33**	51.75
Best-of-N	70.76	73.59	42.32	72.17	37.40	20.83	52.84
Few-Shot-CoT-SC	69.29	71.42	36.22	73.67	37.67	21.67	51.66
Adaptive-SC	70.76	73.46	33.46	74.33	37.13	23.00	52.02
Instant-Runoff Voting	**71.50**$_{+2.21}$	**74.74**$_{+3.32}$	**44.49**$_{+8.27}$	75.67$_{+2.00}$	37.94$_{+0.27}$	23.00$_{+1.33}$	**54.56**$_{+2.90}$
Borda Count Voting	70.93$_{+1.67}$	73.12$_{+1.70}$	42.52$_{+6.30}$	**77.67**$_{+4.00}$	37.40$_{-0.27}$	22.00$_{+0.33}$	53.94$_{+2.28}$
Mean Reciprocal Rank Voting	71.42$_{+2.13}$	72.18$_{+0.76}$	42.91$_{+6.69}$	77.33$_{+3.66}$	**39.02**$_{+1.35}$	22.67$_{+1.00}$	54.26$_{+2.60}$
				(Phi-3-4B)			
Few-Shot-CoT	74.53	86.86	66.54	76.00	62.87	24.67	65.25
Best-of-N	76.29	86.35	74.41	71.50	60.70	22.33	65.26
Few-Shot-CoT-SC	75.84	90.13	73.62	77.33	66.12	23.33	67.73
Adaptive-SC	75.76	88.57	71.26	77.67	65.31	24.00	67.09
Instant-Runoff Voting	78.54$_{+2.70}$	**90.70**$_{+0.57}$	74.41$_{+0.79}$	79.00$_{+2.34}$	**66.40**$_{+0.28}$	**27.00**$_{+3.67}$	69.34$_{+1.61}$
Borda Count Voting	78.71$_{+2.87}$	88.23$_{-1.90}$	**75.20**$_{+1.58}$	78.67$_{+1.34}$	63.96$_{-2.16}$	**27.00**$_{+3.67}$	68.63$_{+0.90}$
Mean Reciprocal Rank Voting	**78.95**$_{+3.11}$	90.44$_{+0.31}$	**75.20**$_{+1.58}$	**80.00**$_{+2.67}$	65.58$_{-0.54}$	**27.00**$_{+3.67}$	**69.53**$_{+1.80}$

Table 3: Comparison between baselines and our method (in gray) across open-source LLMs with 2B∼4B parameters. In each column, the best results are **in bold**, and the second-best results are underlined. The subscript values represent the differences from Few-Shot-CoT-SC baseline.

consistency does not always help. For example, the performance drops (25.33→24.33, 24.33→21.67, and 24.67→23.33) on word unscrambling task. This indicates that for some tasks, due to the difficulty of problems and the inherent limitations of LLMs, majority voting based self-consistency does not always lead to improvements. Overall, our ranked voting based self-consistency generally outperforms majority voting based methods such as Few-Shot-CoT-SC and Adaptive-SC.

7B∼9B Open-Source LLMs. The results are shown in Table 4. Similarly, our methods achieved the best and second-best results, indicating that they are still effective on the medium-size LLMs of ranked voting. Across different scenarios, compared to Few-Shot-CoT-SC, LLaMa-3-8B achieves an average performance improvement of 2.68% to 3.51%. Compared to the 2B∼4B lightweight LLMs, the overall improvement has slightly decreased, possibly due to the increase in the LLMs' inherent capacity.

Closed-Source LLMs. The results are shown in Table 5. Our method achieves an average accuracy of 76.69% and 87.41% on gpt-3.5-turbo-0125 and gpt-4-turbo-2024-04-09, respectively, outperforming the best baseline methods by 3.4% and 0.48%. These two results reflect the differences in GPT-3.5 and GPT-4. Similar to the open-source models, we see large improvements on GPT-3.5, while on the state-of-the-art GPT-4 model, the gains from voting across multiple responses decrease. Nevertheless, our ranking voting method still performs slightly better than the baselines.

4 Analyses

We further analyze our proposed method from different perspectives. Unless otherwise specified, we conduct experiments on CommonsenseQA and WikiData using LLaMa-3-8B.

The Impact of k. Figure 3 shows the performance of different number of samples k. For CommonsenseQA, under different settings of k, ranked voting methods consistently outperform majority voting based self-consistency. Moreover, compared with baseline with $k = 12$ or more, our method achieves better results when $k = 8$, and its performance improves as k increases. In contrast, the baseline shows limited improvement as k increases.

Method	Multiple-Choice QA (Accuracy)			Open-Ended QA (Exact Match)			Average
	CommonsenseQA	ARC-Challenge	AQUA-RAT	WikiData	Date Understanding	Word Unscrambling	
				(LLAMA-3-8B)			
Few-Shot-CoT	78.13	84.13	62.99	78.33	70.19	25.67	66.67
Best-of-N	79.03	85.15	72.44	77.83	69.92	26.83	68.53
Few-Shot-CoT-SC	78.71	86.77	66.93	80.00	71.82	24.00	68.04
Adaptive-SC	78.95	86.95	68.50	79.67	71.00	25.33	68.40
Instant-Runoff Voting	79.12 +0.41	87.29 +0.52	74.02 +7.09	80.00 +0.00	**72.09** +0.27	35.00 +11.00	71.25 +3.21
Borda Count Voting	**79.69** +0.98	87.12 +0.35	70.87 +3.94	80.33 +0.33	71.00 −0.82	**35.33** +11.33	70.72 +2.68
Mean Reciprocal Rank Voting	79.36 +0.65	**87.46** +0.69	**75.20** +8.27	**80.67** +0.67	71.27 −0.45	**35.33** +11.33	**71.55** +3.51
				(QWEN-2.5-7B)			
Few-Shot-CoT	83.29	87.80	80.31	68.00	68.29	17.00	67.45
Best-of-N	84.32	89.04	82.87	69.00	70.05	23.17	69.74
Few-Shot-CoT-SC	84.28	89.16	84.65	70.33	70.73	21.00	70.03
Adaptive-SC	84.68	**89.25**	83.46	71.33	70.46	22.33	70.25
Instant-Runoff Voting	84.77 +0.49	89.16 +0.00	85.04 +0.39	77.67 +7.34	**71.27** +0.54	**24.00** +3.00	71.98 +1.95
Borda Count Voting	84.52 +0.24	**89.25** +0.09	84.25 −0.40	**79.33** +9.00	70.73 +0.00	23.33 +2.33	71.90 +1.87
Mean Reciprocal Rank Voting	**84.93** +0.65	87.80 −1.36	**85.83** +1.18	78.67 +8.34	**71.27** +0.54	**24.00** +3.00	**72.08** +2.05
				(GEMMA-2-9B)			
Few-Shot-CoT	80.75	88.05	63.39	78.00	77.24	45.67	72.18
Best-of-N	80.22	88.52	71.46	77.50	76.29	43.17	72.86
Few-Shot-CoT-SC	80.10	89.59	67.32	81.33	78.05	44.33	73.45
Adaptive-SC	81.41	86.77	66.93	80.67	77.78	45.67	73.20
Instant-Runoff Voting	**83.21** +3.11	**89.68** +0.09	**72.83** +5.51	82.67 +1.34	78.59 +0.54	**46.33** +2.00	**75.55** +2.10
Borda Count Voting	82.96 +1.86	89.33 −0.26	70.87 +3.55	**83.33** +2.00	**79.40** +1.35	46.00 +1.67	75.31 +1.86
Mean Reciprocal Rank Voting	83.05 +2.95	89.25 −0.34	72.44 +5.12	83.00 +1.67	78.32 +0.27	**46.33** +2.00	75.40 +1.95
				(PHI-3-7B)			
Few-Shot-CoT	80.43	91.47	65.75	79.33	68.56	27.67	68.87
Best-of-N	80.79	91.30	**77.17**	77.17	73.04	29.67	71.52
Few-Shot-CoT-SC	81.24	91.89	73.62	79.33	73.71	28.33	71.35
Adaptive-SC	81.16	91.98	74.02	79.00	73.17	29.67	71.50
Instant-Runoff Voting	81.98 +0.74	**92.24** +0.35	**77.17** +3.55	80.00 +0.67	75.34 +1.63	**30.33** +2.00	**72.84** +1.49
Borda Count Voting	**82.31** +1.07	91.81 −0.08	75.20 +1.58	80.00 +0.67	**75.61** +1.90	28.33 +0.00	72.21 +0.86
Mean Reciprocal Rank Voting	82.15 +0.91	91.98 +0.09	**77.17** +3.55	**81.00** +1.67	75.34 +1.63	28.67 +0.34	72.72 +1.37

Table 4: Comparison between baselines and our method (in gray) across open-source LLMs with 7B∼9B parameters.

Method	Multiple-Choice QA (Accuracy)			Open-Ended QA (Exact Match)			Average
	CommonsenseQA	ARC-Challenge	AQUA-RAT	WikiData	Date Understanding	Word Unscrambling	
				(GPT-3.5-TURBO-0125)			
Few-Shot-CoT	79.85	86.65	58.27	78.00	58.27	49.50	68.42
Best-of-N	80.34	87.86	**72.44**	79.83	61.11	58.17	73.29
Few-Shot-CoT-SC	79.61	87.54	69.49	79.00	57.99	54.50	71.36
Adaptive-SC	79.77	88.25	70.67	79.33	58.40	54.50	71.82
Instant-Runoff Voting	80.84 +1.23	89.33 +1.79	70.67 +1.18	82.00 +3.00	**68.97** +10.98	66.50 +12.00	76.30 +4.94
Borda Count Voting	80.92 +1.31	**89.59** +2.05	68.90 −0.59	81.50 +2.50	68.70 +10.71	**67.67** +13.17	76.21 +4.85
Mean Reciprocal Rank Voting	**81.00** +1.39	89.25 +1.71	71.06 +1.57	**82.33** +3.33	68.83 +10.84	**67.67** +13.17	**76.69** +5.33
				(GPT-4-TURBO-2024-04-09)			
Few-Shot-CoT	86.32	93.69	80.31	82.00	88.08	78.67	84.84
Best-of-N	87.39	94.70	85.43	82.33	86.99	81.67	86.42
Few-Shot-CoT-SC	87.39	95.30	86.22	83.00	88.35	81.33	86.93
Adaptive-SC	87.22	95.21	86.22	82.67	**88.62**	81.00	86.82
Instant-Runoff Voting	**87.47** +0.08	**97.01** +1.71	**86.61** +0.39	83.00 +0.00	88.35 +0.00	**82.00** +0.67	**87.41** +0.48
Borda Count Voting	87.31 −0.08	96.84 +1.54	85.04 −1.18	**83.33** +0.33	88.35 +0.00	81.67 +0.34	87.09 +0.16
Mean Reciprocal Rank Voting	87.31 −0.08	**97.01** +1.71	86.22 +0.00	83.00 +0.00	**88.62** +0.27	81.67 +0.34	87.30 +0.37

Table 5: Comparison between baselines and our method (in gray) across closed-source GPT-3.5 and GPT-4 models.

For WikiData, the results of instant-runoff voting and baseline are similar, while the other two methods consistently achieve an accuracy improvement of 0.5∼1.0 with different settings of k.

Robustness Analysis. The performance of LLMs is relatively sensitive to the order of examples in few-shot setting, such as exhibiting some recency bias (Zhao et al., 2021). We randomly shuffled the order of few-shot ($n = 8$) examples and conducted four experiments, and the results are shown in Figure 4. For both CommonsenseQA and WikiData, the performance of our method surpasses that of the baseline consistently, and it exhibits relatively smaller variance, indicating that our approach leverages additional ranking information to yield more robust results.

Compare the Impact of Ranked Answers and the Impact of Ranked Voting. To investigate whether the improvement stems from the ranking of answers or the ranked voting, we conducted experiments with the settings where only a single response with ranked answers is provided, taking

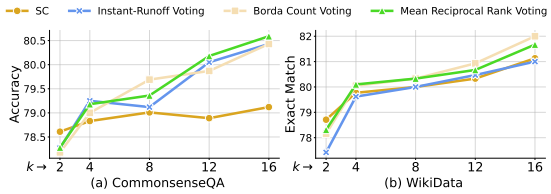

Figure 3: Comparison of methods with different number of responses k.

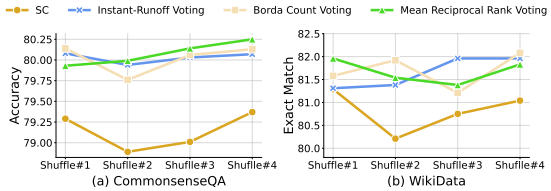

Figure 4: Comparison of methods with different shuffled examples for few-shot learning.

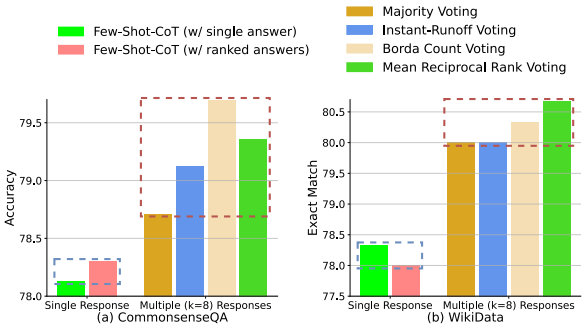

Figure 5: The impact of ranked answers (in blue dashed box) and ranked voting (in red dashed box). Ranked voting is more effective and capable of delivering substantial and reliable improvements.

the top-ranked answer as the final result. In this setting, the voting mechanisms are not employed. The results are shown in Figure 5. We find that the ranked answers along do not lead to consistent improvement, even making the results worse on WikiData. The gap between ranked voting and majority voting is much larger than that between Few-Shot-CoT (w/ ranked answers) and Few-Shot-CoT (w/ single answer), indicating that ranked voting can indeed enhance the model's performance.

The Impact of the Number of Candidates. We observed that the number of candidates can significantly affect the performance of ranked voting methods. To systematically examine this effect, we select a set of values $\mathcal{C} = 1, 2, 3, 4, 5$ and conduct experiments using our approach under each candidate setting.

As shown in Figure 6, we find that the performance of our methods is positively correlated with

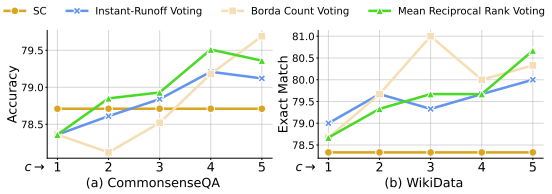

Figure 6: Performance comparison of SC and ranked voting methods (IRV, BCV, MRRV) under different candidate numbers c (Since SC does not incorporate the concept of ranking candidates, its performance remains constant and thus appears as a flat line).

Datasets	Majority Voting	Ranked Voting		
		IRV	BCV	MRRV
CommonsenseQA	5.08%	3.77%	4.42%	2.29%
WikiData	4.33%	4.33%	4.00%	3.67%

Table 6: The tie rates for different voting methods.

the number of candidates c: as c increases, the performance also improves. On CommonsenseQA, our methods surpass the baseline with only $c = 2$ candidates, and by $c = 4$, all ranked voting strategies consistently outperform SC. On WikiData, even with $c = 1$, ranked methods already achieve better performance than SC, and their accuracy continues to rise as more candidates are considered. These results demonstrate that incorporating a richer set of candidate responses allows our ranked approaches to extract stronger reasoning signals, leading to improved robustness and generalization across tasks. Moreover, increasing the number of candidates effectively encourages the model to explore a broader solution space, promoting deeper reasoning and more diverse perspectives during the generation process.

The Impact of Ranking Information on Tie Situations. Both majority voting and ranked voting may result in ties among multiple candidates, which can lead to confusing and incorrect answers. The ranked voting methods leverage ranking information, which effectively reduces the occurrence of ties, as confirmed by our experiments.

The results are shown in Table 6, we find that there are 4.33%∼5.08% ties in majority voting based self-consistency, while our methods reduce this to 2.29%∼4.33%. Among the three ranked voting methods, the weighting-based methods (BCV and MRRV) are less likely to result in ties compared to the elimination-based method (IRV). The difference arises from the vote aggregation mechanisms. Weighting-based methods assign varying

confidence scores to candidate answers, reducing the likelihood of ties. In contrast, IRV iteratively eliminates lower-ranked options, which can lead to ties when multiple answers have similar rankings.

5 Related Work

Chain-of-Thought Reasoning. Chain-of-thought (CoT) have shown effectiveness for eliciting the reasoning ability of LLMs through generating rationales in contexts (Wei et al., 2022; Kojima et al., 2022; Chen et al., 2023; Zhang et al., 2023). Although the generated rationales improve the performance, previous studies focus on designing prompts for certain tasks. For example, Yao et al. (2023) and Besta et al. (2024) further propose tree-of-thought (ToT) and graph-of-thought (GoT) for solving search-style or elaborate problems, respectively. These works usually generate a single final answer in each trial and there is a lack of study on prompting the model to generate multiple possible answers and corresponding priority relationships.

Self-Consistency of LLMs. Wang et al. (2023) propose self-consistency to improve chain-of-thought reasoning, a decoding method that improves the accuracy of LLMs by taking the majority vote among multiple answers generated through diverse reasoning paths. Aggarwal et al. (2023) further propose adaptive self-consistency to reduce the computational cost for obtain diverse reasoning paths, which dynamically adjusts the number of responses based on the model's confidence using a stopping criteria. Xiong et al. (2023) introduce inter-consistency among LLMs for multi agents collaboration. Wang et al. (2024) introduce soft self-consistency in long-duration interactive tasks for language model agents. Huang et al. (2024) enhance the coding ability of LLMs through multiple-perspective self-consistency. As a comparison, these methods rely on majority voting for specific tasks and ignore the possibility of multiple potential answers, which could offer useful information for obtaining the final correct answer in general reasoning and question-answering tasks.

Ranked Voting. Ranked voting expresses the preferences of voters by ranking multiple candidates, organizing selections on an ordinal scale, which has been widely applied in elections, competitions, and recommendation systems (Myatt, 2007; Colomer, 2013). Similar ideas have also been applied to enhance the performance of neural models. Schwartz (2021) propose using an ensemble method to aggregate predictions from multiple MRR and NDCG based models. Liu et al. (2024) proposes pairwise-preference search to address bias in text evaluation by constructing a global ranking. Zhao et al. (2024) introduces general electoral decision-making interface to enhance collaboration among multiple LLM agents through ordinal preferential voting. In recent years, ranking systems have also been valued and proposed to be introduced into more NLP benchmarks (Colombo et al., 2022; Rofin et al., 2023) and Chatbot arenas (Min et al., 2025). Tang et al. (2024) similarly leverage multiple generations via permutation self-consistency to enhance ranking robustness, though their focus is on mitigating positional bias rather than vote-based aggregation.

6 Conclusion

We introduce ranked voting based self-consistency to enhance the reasoning performance of large language models. Our method outperforms traditional majority voting techniques by generating and incorporating ranking information among multiple candidate answers during the reasoning process. The proposed method improves the reliability of self-consistency and boosts the performance. Extensive experiments demonstrate the effectiveness of our method, as it consistently outperforms baselines across a variety of multiple-choice and open-ended question-answering datasets, using both open-source and closed-source LLMs.

Limitations

First, ranked voting has a rich history and comes in many forms. In our main experiments, we demonstrate three effective ranked voting strategies for large language models, however, there are some other ranked voting methods show limited improvement in datasets we used. For example, in the Appendix C, we introduce approval voting and find that this method performs similarly to majority voting when applied to ranked answers from models. Second, as shown in Figure 6, the number of candidates in the ranked answers may have an impact on the performance. Increasing the number of candidates allows a broader range of potential options to be considered, which could improve the accuracy of the final decision. In this work, we enable the model to autonomously rank a small set of options or open-ended generated candidates (usually 4$\sim$5), without imposing additional constraints by designing prompts to expand the candidate pool.

Acknowledgments

This work was supported in parts by NSFC (62306161, U21B2023), Guangdong Basic and Applied Basic Research Foundation (2023B1515120026), DEGP Innovation Team (2022KCXTD025), and Scientific Development Funds from Shenzhen University.

References

Marah Abdin, Jyoti Aneja, Hany Awadalla, Ahmed Awadallah, Ammar Ahmad Awan, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Jianmin Bao, Harkirat Behl, et al. 2024. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*.

Pranjal Aggarwal, Aman Madaan, Yiming Yang, and Mausam. 2023. Let's sample step by step: Adaptive-consistency for efficient reasoning and coding with LLMs. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12375–12396, Singapore. Association for Computational Linguistics.

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, and Torsten Hoefler. 2024. Graph of thoughts: Solving elaborate problems with large language models. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(16):17682–17690.

David Cary. 2011. Estimating the margin of victory for instant-runoff voting. In *Proceedings of the 2011 Conference on Electronic Voting Technology/Workshop on Trustworthy Elections*, EVT/WOTE'11, page 3, USA. USENIX Association.

Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W. Cohen. 2023. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *Transactions on Machine Learning Research*.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Pierre Colombo, Nathan Noiry, Ekhine Irurozki, and Stephan Clémençon. 2022. What are the best systems? new perspectives on nlp benchmarking. In *Advances in Neural Information Processing Systems*, volume 35, pages 26915–26932. Curran Associates, Inc.

Josep M Colomer. 2013. Ramon llull: from 'ars electionis' to social choice theory. *Social Choice and Welfare*, 40(2):317–328.

DeepSeek-AI. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Peter Emerson. 2013. The original borda count and partial voting. *Social Choice and Welfare*, 40.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. 2024. A framework for few-shot language model evaluation.

Baizhou Huang, Shuai Lu, Xiaojun Wan, and Nan Duan. 2024. Enhancing large language models in coding through multi-perspective self-consistency. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1429–1450, Bangkok, Thailand. Association for Computational Linguistics.

Takeshi Kojima, Shixiang (Shane) Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *Advances in Neural Information Processing Systems*, volume 35, pages 22199–22213. Curran Associates, Inc.

Jean-François Laslier and Karine Vander Straeten. 2003. Approval voting: an experiment during the french 2002 presidential election. *Laboratoire d'Econométrie, Ecole Polytechnique, Paris*.

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, Vancouver, Canada. Association for Computational Linguistics.

Yinhong Liu, Han Zhou, Zhijiang Guo, Ehsan Shareghi, Ivan Vulic, Anna Korhonen, and Nigel Collier. 2024. Aligning with human judgement: The role of pairwise preference in large language model evaluators. *arXiv preprint arXiv:2403.16950*.

Rui Min, Tianyu Pang, Chao Du, Qian Liu, Minhao Cheng, and Min Lin. 2025. Improving your model ranking on chatbot arena by vote rigging. *arXiv preprint arXiv:2501.17858*.

David P Myatt. 2007. On the theory of strategic voting. *The Review of Economic Studies*, 74(1):255–281.

OpenAI. 2022. ChatGPT.

OpenAI. 2023. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Gray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*.

Mark Rofin, Vladislav Mikhailov, Mikhail Florinsky, Andrey Kravchenko, Tatiana Shavrina, Elena Tutubalina, Daniel Karabekyan, and Ekaterina Artemova. 2023. Vote'n'rank: Revision of benchmarking with social choice theory. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 670–686, Dubrovnik, Croatia. Association for Computational Linguistics.

Idan Schwartz. 2021. Ensemble of MRR and NDCG models for visual dialog. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3272–3363, Online. Association for Computational Linguistics.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, et al. 2023. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research*. Featured Certification.

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

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, Minneapolis, Minnesota. Association for Computational Linguistics.

Raphael Tang, Crystina Zhang, Xueguang Ma, Jimmy Lin, and Ferhan Ture. 2024. Found in the middle: Permutation self-consistency improves listwise ranking in large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 2327–2340, Mexico City, Mexico. Association for Computational Linguistics.

Team Gemma, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. 2024. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*.

Han Wang, Archiki Prasad, Elias Stengel-Eskin, and Mohit Bansal. 2024. Soft self-consistency improves language models agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 287–301, Bangkok, Thailand. 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 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. Curran Associates, Inc.

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.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. 2024. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik R Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Yifan Zhang, Jingqin Yang, Yang Yuan, and Andrew Chi-Chih Yao. 2023. Cumulative reasoning with large language models. *arXiv preprint arXiv:2308.04371*.

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*, pages 2712–2727, Miami, Florida, USA. Association for Computational Linguistics.

CommonsenseQA (1221) Question: What must someone do before they shop? Options: (A) get money (B) have money (C) bring cash (D) go to market (E) bring cash Answer: (A).
ARC-Challenge (1172) Question: There is a very slight change in gravity at the top of a mountain. Which property would most likely be less at the top of a mountain? Options: (A) mass (B) weight (C) body density (D) body temperature Answer: (B).
AQUA-RAT (254) Question: I have a money pouch containing Rs. 700. There are equal number of 25 paise coins, 50 paise coins and one rupee coins. How many of each are there? Options: (A) 453 (B) 651 (C) 400 (D) 487 (E) 286 Answer: (C).
WikiData (300) Question: The language of The Rise and Fall of Ziggy Stardust and the Spiders from Mars is Answer: English.
Date Understanding (369) Question: Jane got her job in 2016. Today is her 3-year work anniversary. She still remember that on Dec 2, her second day at work, she spilled coffee on her laptop. What is the date today in MM/DD/YYYY? Answer: 12/01/2019.
Word Unscrambling (300) Question: The word eptnerrreenu is a scrambled version of the English word Answer: entrepreneur.

Table 7: Examples of each dataset, the numbers in brackets denote the number of evaluation questions.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 12697–12706. PMLR.

A Examples of Datasets

We list examples of different datasets in Table 7. For CommonsenseQA, we evaluated all instances from the validation set since the test set does not provide answers. For ARC-Challenge, AQUA-RAT, and Date Understanding, we tested on all instances from the test set. For WikiData and Word Unscrambling, we randomly sampled 300 instances from the test set for evaluation.

B Checkpoints of Models

The checkpoints of open-source models in our experiments are shown in Table 8.

C More Results of Other Ranked Voting based Methods

Approval Voting. In addition to the three voting methods discussed in the main pages, we introduce an additional approach called approval voting (Laslier and Vander Straeten, 2003). This voting system enables participants to express their stance on an issue—whether to approve or disapprove. To streamline this process, approval voting assumes that higher-ranked answers are more likely to be accepted as correct. Therefore, we establish a threshold t: answers ranked before this threshold are considered approved, while those ranked after are deemed disapproved:

$$\begin{aligned} \mathcal{A}^r_{1:t} &= \mathcal{A}^{r_1}, \mathcal{A}^{r_2}, \ldots, \mathcal{A}^{r_t}, \\ \mathcal{A}^r_{\text{approved}} &= \mathcal{A}^r_{1:t}. \end{aligned} \tag{10}$$

Next, we conduct majority voting to select the answer with the most supporters, where each approval vote carries equal weight in the majority voting process. In the experiments, we set the threshold $t = 2$.

Results. Table 9 shows the performance of approval voting. It is evident that in certain cases, approval voting outperforms self-consistency and can even achieve the best performance. For example, in the WikiData task, LLaMA-3-8B attains the highest performance using approval voting. However, this method has its limitations as well, since it is not particularly stable. Although it performs well in some situations, it may sometimes underperform compared to Few-Shot-CoT.

D Results of Multiple Experimental Runs

The standard deviation results for multiple experimental runs are shown in Table 10. The comparable and relatively small standard deviations between baseline and ours indicate the stability of the implementation and our ranked voting methods.

E Case Study

We show cases of the ranked voting on CommonsenseQA in Table 11, Table 12, and Table 13.

Model	Checkpoints
LLaMA-3.2-3B	https://huggingface.co/meta-llama/Llama-3.2-3B-Instruct
LLaMA-3-8B	https://huggingface.co/meta-llama/Meta-Llama-3-8B-Instruct
Qwen-2.5-3B	https://huggingface.co/Qwen/Qwen2.5-3B-Instruct
Qwen-2.5-7B	https://huggingface.co/Qwen/Qwen2.5-7B-Instruct
Gemma-2-2B	https://huggingface.co/google/gemma-2-2b-it
Gemma-2-9B	https://huggingface.co/google/gemma-2-9b-it
Phi-3-4B	https://huggingface.co/microsoft/Phi-3-mini-4k-instruct
Phi-3-7B	https://huggingface.co/microsoft/Phi-3-small-8k-instruct

Table 8: Checkpoints of open-source models in our experiments. Note that LLaMA-3 series only include 8B and 70B models, while LLaMA-3.2 series include 1B, 3B, 11B, and 90B models. Therefore, in our experiments, we use LLaMA-3-8B and LLaMA-3.2-3B for validating on the LLaMA models.

Method	Multiple-Choice QA (Accuracy)			Open-Ended QA (Exact Match)			Average
	CommonsenseQA	ARC-Challenge	AQUA-RAT	WikiData	Date Understanding	Word Unscrambling	
(LLaMA-3-8B)							
Few-Shot-CoT	78.13	84.13	62.99	78.33	70.19	25.67	66.67
Few-Shot-CoT-SC	**78.71**	**86.77**	**66.93**	80.00	**71.82**	24.00	68.04
Approval Voting	77.48$_{-1.23}$	83.96$_{-2.81}$	65.35$_{-1.58}$	**80.67**$_{+0.67}$	69.38$_{-2.44}$	**34.67**$_{+10.67}$	**68.58**$_{+0.54}$
(Qwen-2.5-7B)							
Few-Shot-CoT	83.29	87.80	80.31	68.00	68.29	17.00	67.45
Few-Shot-CoT-SC	**84.28**	**89.16**	**84.65**	70.33	**70.73**	21.00	**70.03**
Approval Voting	83.46$_{-0.82}$	87.20$_{-1.96}$	76.77$_{-7.88}$	**78.33**$_{+8.00}$	69.65$_{-1.08}$	**22.33**$_{+1.33}$	69.62$_{-0.41}$
(Gemma-2-9B)							
Few-Shot-CoT	80.75	88.05	63.39	78.00	77.24	45.67	72.18
Few-Shot-CoT-SC	80.10	**89.59**	**67.32**	81.33	**78.05**	44.33	73.45
Approval Voting	**82.64**$_{+2.54}$	88.99$_{-0.60}$	**67.32**$_{+0.00}$	**82.00**$_{+0.67}$	76.15$_{-1.90}$	**46.33**$_{+2.00}$	**73.91**$_{+0.46}$
(Phi-3-7B)							
Few-Shot-CoT	80.43	91.47	65.75	79.33	68.56	27.67	68.87
Few-Shot-CoT-SC	**81.24**	**91.89**	**73.62**	79.33	**73.71**	**28.33**	**71.35**
Approval Voting	80.92$_{-0.32}$	90.36$_{-1.53}$	67.72$_{-5.90}$	**80.67**$_{+1.34}$	70.73$_{-2.98}$	27.67$_{-0.66}$	69.68$_{-1.67}$

Table 9: Results for approval voting using open-source LLMs, where the overall improvement is limited.

Method	Multiple-Choice QA			Open-Ended QA		
	CommonsenseQA	ARC-Challenge	AQUA-RAT	WikiData	Date Understanding	Word Unscrambling
(LLAMA-3.2-3B)						
Few-Shot-CoT-SC	$73.46_{\pm 0.3}$	$80.54_{\pm 0.4}$	$61.81_{\pm 0.3}$	$73.67_{\pm 0.2}$	$60.98_{\pm 0.5}$	$24.33_{\pm 0.6}$
Instant-Runoff Voting	$\mathbf{74.86}_{\pm 0.2}$	$81.31_{\pm 0.2}$	$69.29_{\pm 0.3}$	$74.00_{\pm 0.4}$	$\mathbf{64.23}_{\pm 0.4}$	$27.67_{\pm 0.7}$
Borda Count Voting	$74.69_{\pm 0.2}$	$80.55_{\pm 0.2}$	$67.32_{\pm 0.2}$	$\mathbf{77.00}_{\pm 0.4}$	$60.16_{\pm 0.8}$	$28.33_{\pm 0.7}$
Mean Reciprocal Rank Voting	$74.45_{\pm 0.1}$	$\mathbf{81.40}_{\pm 0.3}$	$\mathbf{71.26}_{\pm 0.3}$	$76.00_{\pm 0.4}$	$62.60_{\pm 0.9}$	$\mathbf{29.00}_{\pm 0.6}$
(QWEN-2.5-3B)						
Few-Shot-CoT-SC	$77.89_{\pm 0.2}$	$76.96_{\pm 0.18}$	$77.95_{\pm 0.2}$	$72.67_{\pm 0.2}$	$47.97_{\pm 0.4}$	$7.33_{\pm 0.02}$
Instant-Runoff Voting	$\mathbf{78.95}_{\pm 0.2}$	$83.45_{\pm 0.3}$	$79.13_{\pm 0.1}$	$74.67_{\pm 0.1}$	$\mathbf{60.70}_{\pm 0.5}$	$11.67_{\pm 0.5}$
Borda Count Voting	$78.38_{\pm 0.2}$	$83.19_{\pm 0.5}$	$\mathbf{80.31}_{\pm 0.4}$	$\mathbf{76.33}_{\pm 0.1}$	$58.81_{\pm 0.5}$	$11.33_{\pm 0.6}$
Mean Reciprocal Rank Voting	$78.79_{\pm 0.3}$	$\mathbf{83.70}_{\pm 0.4}$	$79.92_{\pm 0.1}$	$75.33_{\pm 0.1}$	$60.43_{\pm 0.4}$	$\mathbf{12.33}_{\pm 0.6}$
(GEMMA-2-2B)						
Few-Shot-CoT-SC	$69.29_{\pm 0.2}$	$71.42_{\pm 0.2}$	$36.22_{\pm 0.2}$	$73.67_{\pm 0.2}$	$37.67_{\pm 0.9}$	$21.67_{\pm 0.5}$
Instant-Runoff Voting	$\mathbf{71.50}_{\pm 0.2}$	$\mathbf{74.74}_{\pm 0.4}$	$\mathbf{44.49}_{\pm 0.1}$	$75.67_{\pm 0.4}$	$37.94_{\pm 0.2}$	$\mathbf{23.00}_{\pm 0.3}$
Borda Count Voting	$70.93_{\pm 0.2}$	$73.12_{\pm 0.3}$	$42.52_{\pm 0.2}$	$\mathbf{77.67}_{\pm 0.5}$	$37.40_{\pm 0.4}$	$22.00_{\pm 0.5}$
Mean Reciprocal Rank Voting	$71.42_{\pm 0.2}$	$72.18_{\pm 0.3}$	$42.91_{\pm 0.1}$	$77.33_{\pm 0.5}$	$\mathbf{39.02}_{\pm 0.4}$	$22.67_{\pm 0.4}$
(PHI-3-4B)						
Few-Shot-CoT-SC	$75.84_{\pm 0.3}$	$90.13_{\pm 0.2}$	$73.62_{\pm 0.5}$	$77.33_{\pm 0.2}$	$66.12_{\pm 0.5}$	$23.33_{\pm 0.5}$
Instant-Runoff Voting	$78.54_{\pm 0.3}$	$\mathbf{90.70}_{\pm 0.3}$	$74.41_{\pm 0.1}$	$79.00_{\pm 0.4}$	$\mathbf{66.40}_{\pm 0.4}$	$\mathbf{27.00}_{\pm 1.1}$
Borda Count Voting	$78.71_{\pm 0.3}$	$88.23_{\pm 0.2}$	$\mathbf{75.20}_{\pm 0.3}$	$78.67_{\pm 0.3}$	$63.96_{\pm 0.9}$	$\mathbf{27.00}_{\pm 1.3}$
Mean Reciprocal Rank Voting	$\mathbf{78.95}_{\pm 0.3}$	$90.44_{\pm 0.2}$	$\mathbf{75.20}_{\pm 0.3}$	$\mathbf{80.00}_{\pm 0.3}$	$65.58_{\pm 0.4}$	$\mathbf{27.00}_{\pm 1.2}$
(LLAMA-3-8B)						
Few-Shot-CoT-SC	$78.71_{\pm 0.2}$	$86.77_{\pm 0.2}$	$66.93_{\pm 0.3}$	$80.00_{\pm 0.3}$	$71.82_{\pm 0.8}$	$24.00_{\pm 0.6}$
Instant-Runoff Voting	$79.12_{\pm 0.2}$	$87.29_{\pm 0.2}$	$74.02_{\pm 0.3}$	$80.00_{\pm 0.2}$	$\mathbf{72.09}_{\pm 0.3}$	$35.00_{\pm 0.6}$
Borda Count Voting	$\mathbf{79.69}_{\pm 0.2}$	$87.12_{\pm 0.2}$	$70.87_{\pm 0.3}$	$80.33_{\pm 0.4}$	$71.00_{\pm 0.4}$	$\mathbf{35.33}_{\pm 1.0}$
Mean Reciprocal Rank Voting	$79.36_{\pm 0.2}$	$\mathbf{87.46}_{\pm 0.2}$	$\mathbf{75.20}_{\pm 0.3}$	$\mathbf{80.67}_{\pm 0.4}$	$71.27_{\pm 0.6}$	$\mathbf{35.33}_{\pm 1.1}$
(QWEN-2.5-7B)						
Few-Shot-CoT-SC	$84.28_{\pm 0.1}$	$89.16_{\pm 0.2}$	$84.65_{\pm 0.1}$	$70.33_{\pm 0.5}$	$70.73_{\pm 0.7}$	$21.00_{\pm 0.3}$
Instant-Runoff Voting	$84.77_{\pm 0.2}$	$89.16_{\pm 0.2}$	$85.04_{\pm 0.1}$	$77.67_{\pm 0.3}$	$\mathbf{71.27}_{\pm 0.3}$	$\mathbf{24.00}_{\pm 0.4}$
Borda Count Voting	$84.52_{\pm 0.3}$	$\mathbf{89.25}_{\pm 0.2}$	$84.25_{\pm 0.3}$	$\mathbf{79.33}_{\pm 0.3}$	$70.73_{\pm 0.4}$	$23.33_{\pm 0.3}$
Mean Reciprocal Rank Voting	$\mathbf{84.93}_{\pm 0.2}$	$87.80_{\pm 0.2}$	$\mathbf{85.83}_{\pm 0.2}$	$78.67_{\pm 0.3}$	$\mathbf{71.27}_{\pm 0.4}$	$\mathbf{24.00}_{\pm 0.3}$
(GEMMA-2-9B)						
Few-Shot-CoT-SC	$80.10_{\pm 0.1}$	$89.59_{\pm 0.2}$	$67.32_{\pm 0.2}$	$81.33_{\pm 0.3}$	$78.05_{\pm 0.2}$	$44.33_{\pm 0.4}$
Instant-Runoff Voting	$\mathbf{83.21}_{\pm 0.1}$	$\mathbf{89.68}_{\pm 0.2}$	$\mathbf{72.83}_{\pm 0.2}$	$82.67_{\pm 0.4}$	$78.59_{\pm 0.2}$	$\mathbf{46.33}_{\pm 0.4}$
Borda Count Voting	$82.96_{\pm 0.1}$	$89.33_{\pm 0.1}$	$70.87_{\pm 0.2}$	$\mathbf{83.33}_{\pm 0.1}$	$\mathbf{79.40}_{\pm 0.1}$	$46.00_{\pm 0.4}$
Mean Reciprocal Rank Voting	$83.05_{\pm 0.2}$	$89.25_{\pm 0.1}$	$72.44_{\pm 0.2}$	$83.00_{\pm 0.2}$	$78.32_{\pm 0.1}$	$\mathbf{46.33}_{\pm 0.4}$
(PHI-3-7B)						
Few-Shot-CoT-SC	$81.24_{\pm 0.4}$	$91.89_{\pm 0.1}$	$73.62_{\pm 0.3}$	$79.33_{\pm 0.2}$	$73.71_{\pm 0.5}$	$28.33_{\pm 0.8}$
Instant-Runoff Voting	$81.98_{\pm 0.2}$	$\mathbf{92.24}_{\pm 0.1}$	$\mathbf{77.17}_{\pm 0.2}$	$80.00_{\pm 0.4}$	$75.34_{\pm 0.4}$	$\mathbf{30.33}_{\pm 0.8}$
Borda Count Voting	$\mathbf{82.31}_{\pm 0.2}$	$91.81_{\pm 0.2}$	$75.20_{\pm 0.3}$	$80.00_{\pm 0.5}$	$\mathbf{75.61}_{\pm 0.5}$	$28.33_{\pm 0.7}$
Mean Reciprocal Rank Voting	$82.15_{\pm 0.2}$	$91.98_{\pm 0.1}$	$\mathbf{77.17}_{\pm 0.2}$	$\mathbf{81.00}_{\pm 0.4}$	$75.34_{\pm 0.3}$	$28.67_{\pm 0.7}$
(GPT-3.5-TURBO-0125)						
Few-Shot-CoT-SC	$79.61_{\pm 0.2}$	$87.54_{\pm 0.1}$	$69.49_{\pm 0.3}$	$79.00_{\pm 0.1}$	$57.99_{\pm 0.4}$	$54.50_{\pm 0.2}$
Instant-Runoff Voting	$80.84_{\pm 0.1}$	$89.33_{\pm 0.1}$	$70.67_{\pm 0.2}$	$82.00_{\pm 0.3}$	$\mathbf{68.97}_{\pm 0.5}$	$66.50_{\pm 0.3}$
Borda Count Voting	$80.92_{\pm 0.1}$	$\mathbf{89.59}_{\pm 0.1}$	$68.90_{\pm 0.2}$	$81.50_{\pm 0.2}$	$68.70_{\pm 0.5}$	$\mathbf{67.67}_{\pm 0.4}$
Mean Reciprocal Rank Voting	$\mathbf{81.00}_{\pm 0.1}$	$89.25_{\pm 0.1}$	$\mathbf{71.06}_{\pm 0.2}$	$\mathbf{82.33}_{\pm 0.3}$	$68.83_{\pm 0.5}$	$\mathbf{67.67}_{\pm 0.3}$
(GPT-4-TURBO-2024-04-09)						
Few-Shot-CoT-SC	$87.39_{\pm 0.1}$	$95.30_{\pm 0.1}$	$86.22_{\pm 0.2}$	$83.00_{\pm 0.1}$	$88.35_{\pm 0.3}$	$81.33_{\pm 0.0}$
Instant-Runoff Voting	$\mathbf{87.47}_{\pm 0.1}$	$\mathbf{97.01}_{\pm 0.1}$	$\mathbf{86.61}_{\pm 0.1}$	$83.00_{\pm 0.2}$	$88.35_{\pm 0.2}$	$\mathbf{82.00}_{\pm 0.0}$
Borda Count Voting	$87.31_{\pm 0.1}$	$96.84_{\pm 0.1}$	$85.04_{\pm 0.2}$	$\mathbf{83.33}_{\pm 0.2}$	$88.35_{\pm 0.2}$	$81.67_{\pm 0.1}$
Mean Reciprocal Rank Voting	$87.31_{\pm 0.1}$	$\mathbf{97.01}_{\pm 0.1}$	$86.22_{\pm 0.1}$	$83.00_{\pm 0.2}$	$\mathbf{88.62}_{\pm 0.2}$	$81.67_{\pm 0.1}$

Table 10: The standard deviation results in subscripts of multiple experimental runs.

CommonsenseQA

Question: Dance can be elegant and specific, or you can just have fun and what?
Options: (A) falling down (B) trip (C) fall down (D) move around (E) celebrate.
Ground-Truth Label: D

Self-Consistency Answer through Majority Voting: A. ✗

Our Ranked Answers:
Response 1: The answer should focus on an action that represents a casual, carefree approach, contrasting with the elegance and precision of dance. "Trip" fits well because it conveys a playful or carefree moment that can happen when you're not focusing on being precise. This aligns with the idea of just having fun. The ranking of options by likelihood is: B > D > E > C > A.
Response 2: The answer should be the action that is the opposite of being elegant and specific, which implies a more casual or carefree approach. Of the above choices, the closest match is "just have fun and fall down", which implies a more playful and carefree attitude. The ranking of options by likelihood is: A > D > E > C > B.
Response 3: The answer should be the phrase that describes the opposite of elegance, which is a more casual and carefree way of dancing. Of the above choices, "falling down" implies a loss of control and a more playful approach to dancing. The ranking of options by likelihood is: A > E > D > C > B.
Response 4: The answer should be a phrase that describes an action that is carefree and fun, which is a common aspect of dancing. The ranking of options by likelihood is: E > D > A > C > B.
Response 5: The answer should be the action that is often associated with having fun while dancing. Of the above choices, "move around" is a general action that is often done while having fun and dancing, without any specific technique or elegance. The ranking of options by likelihood is: D > E > A > C > B.
Response 6: The answer should be the opposite of "elegant and specific", which implies a more carefree and casual attitude towards dancing. Of the above choices, "just have fun" is closely related to "move around" in a playful way. The ranking of options by likelihood is: D > E > A > C > B.
Response 7: The answer should be the action that is often associated with having fun while dancing. Of the above choices, "have fun" is often associated with "celebrate". The ranking of options by likelihood is: E > D > A > C > B.
Response 8: The answer should be the opposite of "elegant" and "specific", which implies a more casual or playful approach to dance. The ranking of options by likelihood is: A > E > D > C > B.

Instant-Runoff Voting: D. ✓

Instant-Runoff Voting Procedure:
Round 1:
B > D > E > C > A × 1
A > D > E > C > B × 1
A > E > D > C > B × 2
E > D > A > C > B × 2
D > E > A > C > B × 2
Count: A (3 times), B (1 times), C (0 times), D (2 times), E (2 times). No option gets more than half the votes, and C has the fewest, so C is eliminated.
Round 2:
B > D > E > A × 1
A > D > E > B × 1
A > E > D > B × 2
E > D > A > B × 2
D > E > A > B × 2
Count: A (3 times), B (1 times), D (2 times), E (2 times). No option gets more than half the votes, and B has the fewest, so B is eliminated.
Round 3:
D > E > A × 1 $\xrightarrow{merge}$ D > E > A × 3
A > D > E × 1 A > D > E × 1
A > E > D × 2 A > E > D × 2
E > D > A × 2 E > D > A × 2
D > E > A × 2
Count: A (3 times), D (3 times), E (2 times). No option gets more than half the votes, and E has the fewest, so E is eliminated.
Round 4:
D > A × 1 $\xrightarrow{merge}$ D > A × 5
A > D × 1 A > D × 3
A > D × 2
D > A × 2
D > A × 2
Count: A (3 times), D (5 times). D gets more than half the votes, so D is the winner.

Table 11: Example of instant-runoff voting on CommonsenseQA.

CommonsenseQA

Question: Danny found an old film in a sealed what?
Options: (A) clingfilm (B) disneyland (C) cave (D) cabinet (E) movie.
Ground-Truth Label: D

Self-Consistency Answer through Majority Voting: A. ✗

Our Ranked Answers:
Response 1: The answer should be the container where the film was found. Of the above choices, clingfilm and cabinet are both plausible containers, but sealed clingfilm is more likely to preserve an old film. The ranking of options by likelihood is: A > D > C > E > B.
Response 2: The answer should be the place where an old film would likely be stored. Of the above choices, a sealed cabinet makes the most sense for storing old films. The ranking of options by likelihood is: D > C > A > E > B.
Response 3: The answer should be a place where an old film could be found, likely in a sealed container. Of the above choices, a cabinet is a common place to store things, and it could be sealed. The ranking of options by likelihood is: D > C > A > E > B.
Response 4: The answer should be the item that is used to seal something. Of the above choices, clingfilm is used to seal things. The ranking of options by likelihood is: A > D > C > E > B.
Response 5: The answer should be a container or packaging that can be sealed. Of the above choices, clingfilm is a type of plastic wrap that can be sealed. The ranking of options by likelihood is: A > D > C > E > B.
Response 6: The answer should be the container that is typically sealed to keep things inside. Of the above choices, clingfilm is a type of plastic wrap used to seal things. The ranking of options by likelihood is: A > D > C > E > B.
Response 7: The answer should be a container or object that can be sealed. Of the above choices, clingfilm (also known as plastic wrap) is commonly used to seal items. The ranking of options by likelihood is: A > D > C > E > B.
Response 8: The answer should be the container where an old film could be stored and sealed. Of the above choices, the most likely container is a cabinet. The ranking of options by likelihood is: D > C > A > E > B.

Borda Count Voting: D. ✓

Borda Count Voting Calculation:
A: 5 + 3 + 3 + 5 + 5 + 5 + 5 + 3 = 34

B: 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 = 8

C: 3 + 4 + 4 + 3 + 3 + 3 + 3 + 4 = 27

D: 4 + 5 + 5 + 4 + 4 + 4 + 4 + 5 = 35 ✓

E: 2 + 2 + 2 + 2 + 2 + 2 + 2 + 2 = 16

Table 12: Example of Borda count voting voting on CommonsenseQA.

CommonsenseQA

Question: The man acted ridiculous at the funeral, what attitude should he have taken?
Options: (A) straightforward (B) serious (C) solemn (D) somber (E) funny.
Ground-Truth Label: C

Self-Consistency Answer through Majority Voting: B. ✗

Our Ranked Answers:
Response 1: The most likely attitude to take at a funeral is solemn. This word implies a sense of respect and seriousness that is appropriate for such a setting. The other options, such as straightforward, serious, and somber, are all related to seriousness and respect, but "funny" would be inappropriate and insensitive. The ranking of options by likelihood is: C > D > A > E > B.
Response 2: The most appropriate attitude for the man to take at a funeral would be "serious" or "somber". Funeral are solemn occasions requiring a display of respect for the deceased and the grieving process. Funny or straight-forward would be inappropriate. The ranking of options by likelihood is: B > C > D > E > A.
Response 3: The most appropriate demeanor at a funeral is one of solemnity, as it is a moment for grieving and honoring the departed. Words like straightforward, serious, and somber all reflect an attitude of respect and mourning, while funny would be seen as unsuitable and disrespectful. The ranking of options by likelihood is: C > D > E > A > B.
Response 4: The most likely answer is (B) serious, as a funeral is a somber occasion and a serious attitude is appropriate and respectful. The other options, like straightforward, solemn, and somber, are less likely to be the correct response to a funeral. The ranking of options by likelihood is: B > C > A > D > E.
Response 5: The most likely attitude to take at a funeral is (B) serious, as this is a time of mourning and respect, and a funeral is a solemn occasion. The other options, such as straightforward, somber, and funny, are less appropriate for such an event. The ranking of options by likelihood is: B > E > A > C > D.
Response 6: The most appropriate attitude for the man to have taken at a funeral would be "somber", as it reflects a sense of respect and sadness appropriate for such a solemn occasion. The other options, while potentially applicable in other contexts, are not the best match for a funeral, which is generally a time for mourning and reflection. The ranking of options by likelihood is: D > C > E > A > B.
Response 7: The most likely attitude to take at a funeral is solemn, as it is a time for mourning and respect for the deceased. Straightforward, serious, and somber are all synonyms for showing respect and mourning, while funny would be inappropriate and disrespectful. The ranking of options by likelihood is: C > B > D > A > E.
Response 8: The most likely attitude the man should have taken is (B) serious, as funerals are somber occasions requiring respect and solemnity. The other options, such as straightforward, solemn, and somber, are less appropriate for such an event. The ranking of options by likelihood is: B > C > D > E > A.

Mean Reciprocal Rank Voting: C. ✓

Mean Reciprocal Rank Voting Calculation:
A: $\frac{1}{8}(\frac{1}{3}+\frac{1}{5}+\frac{1}{4}+\frac{1}{3}+\frac{1}{3}+\frac{1}{4}+\frac{1}{4}+\frac{1}{5}) = 0.26875$

B: $\frac{1}{8}(\frac{1}{5}+1+\frac{1}{5}+1+1+\frac{1}{5}+\frac{1}{2}+1) = 0.6375$

C: $\frac{1}{8}(1+\frac{1}{2}+1+\frac{1}{2}+\frac{1}{4}+\frac{1}{2}+1+\frac{1}{2}) = 0.65625$ ✓

D: $\frac{1}{8}(\frac{1}{2}+\frac{1}{3}+\frac{1}{2}+\frac{1}{4}+\frac{1}{5}+1+\frac{1}{3}+\frac{1}{3}) = 0.43125$

E: $\frac{1}{8}(\frac{1}{4}+\frac{1}{4}+\frac{1}{3}+\frac{1}{5}+\frac{1}{2}+\frac{1}{3}+\frac{1}{5}+\frac{1}{4}) \approx 0.28958$

Table 13: Example of mean reciprocal rank voting on CommonsenseQA.