Confidence Improves Self-Consistency in LLMs

Anonymous ACL submission

Abstract

Self-consistency decoding enhances LLMs' performance on reasoning tasks by sampling diverse reasoning paths and selecting the most frequent answer. However, it is computationally expensive, as sampling many of these (lengthy) paths is required to increase the chances that the correct answer emerges as the most frequent one. To address this, we introduce *Confidence-Informed Self-Consistency (CISC)*. CISC performs a *weighted* majority vote based on confidence scores obtained directly from the model. By prioritizing high-confidence paths, it can identify the correct answer with a significantly smaller sample size. When tested on nine models and four datasets, CISC outperforms self-consistency in nearly all configurations, reducing the required number of reasoning paths by over 40% on average. In addition, we introduce the notion of *within-question confidence evaluation*, after showing that standard evaluation methods are poor predictors of success in distinguishing correct and incorrect answers to the same question. In fact, the most calibrated confidence method proved to be the least effective for CISC. Lastly, beyond these practical implications, our results and analyses show that LLMs can effectively judge the correctness of their own outputs, contributing to the ongoing debate on this topic.

1 Introduction

Modern large language models (LLMs) demonstrate strong reasoning capabilities (Bubeck et al., 2023; Guo et al., 2025), driven in part by their capacity to generate a sequence of intermediate reasoning steps that lead them toward a final answer (Wei et al., 2022; Jaech et al., 2024). Self-consistency (Wang et al., 2022) is a popular decoding strategy that further improves LLMs' reasoning performance by sampling a diverse set of reasoning paths and selecting the most frequent answer as the final output. Despite its effectiveness, this approach is also computationally expensive, as it requires generating a large number of (long) reasoning paths to increase the chances that the correct answer emerges as the most frequent one.

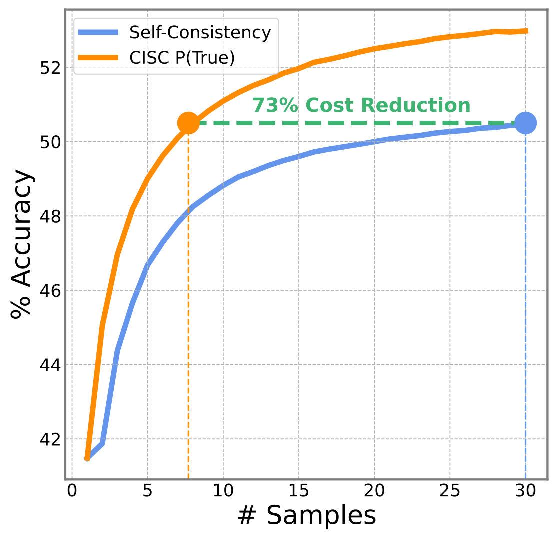

Figure 1: Accuracy as a function of the number of sampled responses for self-consistency vs CISC, using Gemma2-9B on the MATH dataset. CISC achieves higher overall accuracy while significantly reducing computational costs. With just 8 samples, it surpasses the performance of 30-sample self-consistency.

Motivated by recent evidence that LLMs possess the ability to judge the correctness of their own outputs (Kadavath et al., 2022; Zhang et al., 2024), we hypothesize that self-consistency could be made significantly more efficient if the model could *review* each generated reasoning path before selecting a final answer. We therefore introduce **Confidence-Informed Self-Consistency** (CISC), a lightweight extension of self-consistency. As illustrated in Figure 2, CISC uses the model to generate a self-assessment score for each path and employs these scores in a weighted majority vote.

We conducted a comprehensive comparison of

[*] Equal contribution.

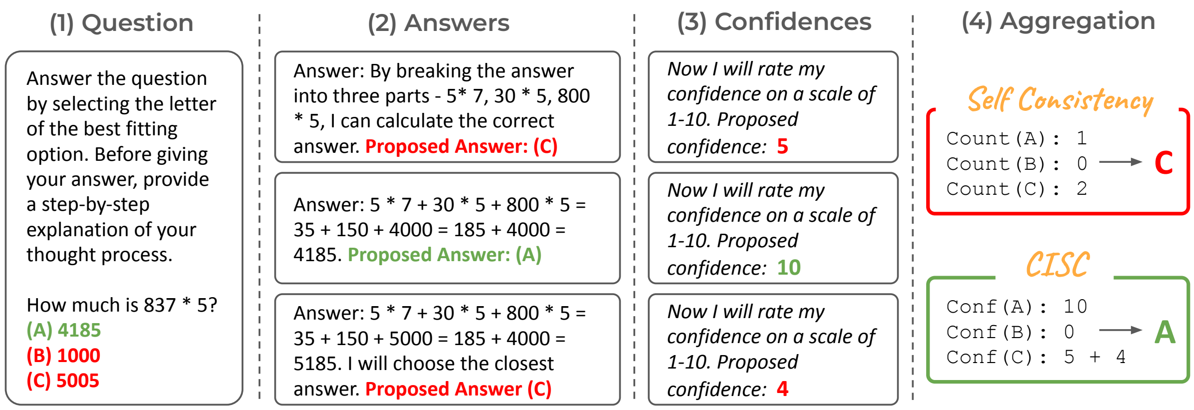

Figure 2: A simplified example comparing self-consistency vs CISC. (1) Given an input question, (2) both methods first sample multiple reasoning paths. (4, top) Self-consistency then simply selects the most frequent answer. Conversely, (3) CISC adds a self-assessment step, where a confidence score is assigned to each path (see §4.1 for more advanced methods). Then, (4, bottom) it selects the final answer via a weighted majority vote.

CISC and self-consistency, spanning nine LLMs of various sizes, four datasets covering a wide range of mathematical and commonsense reasoning tasks, and three popular methods for deriving self-assessment confidence scores from the model. Our results demonstrate that CISC outperforms self-consistency in virtually all the examined configurations. Using the best-performing confidence estimation method, CISC achieves comparable performance to self-consistency while reducing the required number of reasoning paths by over 40% on average (See Figure 1 for an example).

Surprisingly, the most calibrated confidence method is actually the least useful for CISC. We offer a potential explanation: existing confidence evaluation metrics measure the usefulness of confidence scores for comparing answers across different questions, while CISC requires distinguishing correct and incorrect answers for the same question. To address this, we propose the Within-Question Discrimination (WQD) metric that specifically measures this ability, and demonstrate that it can predict the relative performance of CISC with different confidence methods.

Finally, we conduct a qualitative-analysis and find a significant agreement between model confidence scores and human assessments of the reasoning-paths' quality. Specifically, responses identified by the model as low-confidence were also significantly more likely to be flagged by human evaluators as exhibiting signs of low-quality reasoning patterns.

To summarize, we contribute practical methods and foundational insights:

- We propose CISC, a decoding strategy that can be used as a drop-in replacement to self-consistency, achieving comparable accuracy at a significantly lower computational cost.
- We introduce the concept of within-question confidence evaluation, after showing that standard evaluation methods are poor predictors of success in distinguishing correct and incorrect answers to the same question.
- We present empirical evidence supporting the idea that LLMs are capable of self-assessing their responses, contributing to the ongoing debate regarding this capability (Gero et al., 2023; Huang et al., 2023; Li et al., 2024a; Stechly et al., 2024)

2 Notations

We consider an auto-regressive language model M with parameters θ. We use $p_\theta(\cdot|x)$ to denote M's distribution over the next token given the provided context x. Given a question q (e.g., "*Jane had 4 apples and ate half of her apples. How many apples she has now?*"), we denote the model's response as $(\mathbf{r}, \mathbf{a})$, where $\mathbf{a}$ is the answer (e.g., "2") and $\mathbf{r}$ is a *reasoning path* (or chain-of-thought), a sequence of logical steps supposedly leading up to this answer (e.g., "*If Jane ate half her apples, this means she ate 2 apples. 4 minus 2 is 2.*").

3 Confidence-Informed Self-Consistency

In this section we present *Confidence-Informed Self-Consistency* (CISC). When designing CISC, we hypothesized that it is possible to reduce self-consistency's computational costs by generating a *confidence score* for each reasoning path, and performing a weighted majority vote.

As an intuitive example, consider a hypothetical setting where there exist only two possible answers, one correct and one incorrect. For a model that responds with the correct answer 60% of the time, standard majority voting will require *40 samples* to reach 90% accuracy[1]. However, a weighted majority vote that weights correct answers twice as much as incorrect ones, will achieve 90% accuracy with less than *10 samples*.

With this motivation in mind, we build on recent findings suggesting that LLMs are capable of judging the correctness of their own outputs (Kadavath et al., 2022; Tian et al., 2023b; Zhang et al., 2024), and incorporate the model's self-assessment of its reasoning paths into the final answer selection:

Definition 3.1 (Confidence-Informed Self-Consistency)**.** *Given a question q and responses $\{(\boldsymbol{r}_1, \boldsymbol{a}_1), \ldots, (\boldsymbol{r}_m, \boldsymbol{a}_m)\}$, CISC involves:*

- ***Confidence Extraction**: A self-assessed confidence score $c_i \in \mathbb{R}$ is derived for each $(\boldsymbol{r}_i, \boldsymbol{a}_i)$.*
- ***Confidence Normalization**: The confidence scores are normalized using Softmax: $\tilde{c}_i = \frac{\exp\left(\frac{c_i}{T}\right)}{\sum_{j=1}^{m} \exp\left(\frac{c_j}{T}\right)}$, where T is a tunable temperature hyper-parameter (see discussion below).*
- ***Aggregation**: The final answer is selected using a confidence-weighted majority vote: $\hat{a}_{CISC} = \arg\max_a \sum_{i=1}^{m} \mathbb{1}[\boldsymbol{a}_i = a] \cdot \tilde{c}_i$.*

The temperature parameter T controls the relative importance of the answer frequency versus the confidence scores. Namely, as $T \to \infty$, the distribution of normalized confidence scores approaches the uniform distribution, and CISC collapses to vanilla self-consistency. Conversely, as $T \to 0$, the softmax normalization approaches the hard maximum function, prioritizing the single response with the highest confidence and disregarding the overall frequency of answers. This may lead CISC to select a different answer than self-consistency (see Figure 2).

[1] Calculated using the binomial distribution. All the technical details are included in Appendix A

4 Experimental Setup

We compare CISC and self-consistency across a range of confidence extraction methods (§4.1), reasoning tasks (§4.2) and models (§4.3).

4.1 Confidence Extraction Methods

We use the following methods:

- **Response Probability** (Wang et al., 2022): The confidence in a response $(\mathbf{r}, \mathbf{a})$ is taken to be the model's (length-normalized) probability of generating $(\mathbf{r}, \mathbf{a}) = (x_1, \ldots, x_n)$ given the question:

$$p_\theta(\mathbf{r}, \mathbf{a}) = [\Pi_{i=1}^{n} p_\theta(x_i | x_1 \ldots x_{i-1}, q)]^{\frac{1}{n}}$$

- **Verbal Confidence** (Lin et al., 2022): After sampling $(\mathbf{r}, \mathbf{a})$ from the model, we prompt it to rate its confidence in its previously generated output. We implement two variants: (1) **Verbal Binary** instructs the model to output either 0 or 1, and (2) **Verbal 0-100** instructs the model to output a score on a scale of 0-100.
- **P(True)** Kadavath et al. (2022): We prompt the model to rate its confidence in $(\mathbf{r}, \mathbf{a})$ in binary format (either 0 or 1), and compute the probability that the model assigns to the token 1.

Efficient and Consistent Confidence Prompting. Our implementation of the prompt-based methods employs a *two-step* prompting procedure (as depicted in Figure 2). Given a question prompt q, we first use the model to generate the reasoning chain and answer (r, a). We then concatenate a confidence extraction prompt e (e.g., "*Now I will rate my confidence...*"), and continue the generation on (q, r, a, e). This serves two important purposes. First, it ensures that when comparing self-consistency and CISC, the reasoning chains are identical. Second, the fact that the prefix (q, r, a) remains unchanged after concatenating the confidence extraction prompt e means it does not require reprocessing by the LLM. Consequently, the additional cost of the confidence extraction step consists only of encoding len$(e) \approx 20$ tokens and generating a single token. Since a single (q, r, a) typically contains hundreds of tokens, the confidence extraction step adds only a negligible computational overhead to self-consistency. Further overhead reduction can be achieved through prompt optimization or by using the single-step procedure described in Appendix B. The precise prompts used and additional technical details are also provided in Appendix B.

4.2 Datasets

We used four large reasoning benchmarks:[2]

- **GSM8K** (Cobbe et al., 2021a): A dataset of grade-school level math word problems. We evaluate on the entire validation set (1320 questions).
- **MATH** (Hendrycks et al., 2021): A more challenging dataset of math word problems. We used the entire test set (5K questions).
- **MMLU-Pro** (Wang et al., 2024c): A more challenging version of the Multitask Language Understanding (MMLU) benchmark, testing language models' general knowledge and reasoning abilities with a wide range of topics such as science and history. We randomly sampled 5K questions.
- **Big-Bench-Hard** (Suzgun et al., 2022): A challenging selection of tasks from the big-bench benchmark (bench authors, 2023), comprises a variety of reasoning tasks that pose challenges to LLMs, such as counting objects. We selected 20 out of 23 tasks (5,761 examples), eliminating three sub-tasks that required designated answer extraction methods.

4.3 Models

We use nine instruction-tuned open-weights LLMs from 3 different families:

- **GEMMA2** (Team et al., 2024): A Google AI model family, including 2B, 9B, and 27B parameter models.
- **QWEN2.5** (Yang et al., 2024): A model family from Alibaba AI, with 7 models ranging from 0.5B to 72B parameters. We selected three models: 3B, 14B, and 72B.
- **Mistral** (Mistral-AI, 2024): We used three of the latest models available - Ministral-8B-Instruct-2410, Mistral-Small-Instruct-2409, mistralai/Mistral-Large-Instruct-2411 - with 8B, 22B, 123B parameters respectively.

4.4 Metrics

We compare CISC against self-consistency using the following metrics:

- **% Cost Reduction**: The percentage of computational cost saved by using CISC. We fix the compute budget for CISC (5 or 10 model responses) and measure the number of responses[3] required for self-consistency to achieve equivalent accuracy:

$$100 \times \left(1 - \frac{\text{CISC budget}}{\text{\# Comparable SC responses}}\right)$$

- **% Accuracy Improvement**: The relative accuracy gain of CISC over self-consistency when both methods utilize the same number of responses per question:

$$100 \times \left(\frac{\text{CISC Acc}}{\text{SC Acc}} - 1\right)$$

[2] Other than the popular GSM8K, the other datasets were chosen as the three largest datasets in the Hugging Face Leaderboard (Hugging-Face, 2024b) (as of December 1st, 2024).

4.5 Temperature Scaling

As discussed in §3, CISC re-scales the confidence values using a softmax transformation, parameterized by a temperature $T > 0$. We tune the temperature separately for each model and confidence extraction method using a 10% held-out set, aggregated across all four datasets (§4.2). More details and the optimal temperature values for each configuration are in appendix D.

4.6 Bootstrap

To compute the performance of a decoding strategy s (either self-consistency or a variant of CISC) with a sample budget of $b \in [1, ..., 30]$, we perform bootstrap sampling. We first sample 30 different reasoning paths from the model. Next, we draw $n = 500$ sets of b paths for each question, apply s to each set, and compute the accuracy per set. We then average the results across all bootstrap samples to obtain the final score.

5 Main Results

This section demonstrates CISC's (§3.1) substantial performance advantage over self-consistency. We compare CISC, using fixed compute budgets of 5 and 10 responses per question, based on the metrics defined in §4.4.

CISC outperforms self-consistency across virtually all models and datasets. Table 1 presents the *Cost Reduction* and *Accuracy Improvement* (see §4.4) achieved by CISC with each confidence method. The results are macro-averaged across all models and datasets. CISC outperforms self-consistency with every confidence method.

[3] If self-consistency failed to reach CISC's accuracy using up to 30 responses, we use a maximal value of 31 for this metric.

	Cost Reduction		Acc Improvement	
Confidence Method	Budget 5	Budget 10	Budget 5	Budget 10
Verbal Binary	18% (6.1)	10% (11.1)	0.4%	0.2%
Verbal 1-100	22% (6.4)	30% (14.4)	0.8%	0.4%
Response Probability	22% (6.5)	31% (14.6)	1.1%	0.8%
P(True)	**41% (8.4)**	**46% (18.6)**	**1.6%**	**1.1%**

Table 1: **CISC performance (macro-averaged over all datasets and models) per confidence method.** CISC performs better than standard self-consistency in terms of both efficiency gains and accuracy improvements across all confidence methods. Specifically, the **P-True** method achieves the best results. For instance, self-consistency must use 18.6 sampled responses on average to match the accuracy obtained by CISC using only 10 samples, representing a 46% reduction in computational costs.

The *P(True)* method yields the best results, achieving an average Cost Reduction of 41% and 46% with budgets of 5 and 10 responses, respectively. Figure 3 presents a detailed breakdown of CISC's performance using *P(True)* across all models and datasets. Notably, CISC is effective across nearly all configurations, with some exceeding 67% cost reduction.

We provide additional results in Appendix C. In particular, Table 6 shows a per-dataset breakdown of Table 1, and Table 7 shows the Accuracy Improvement metric micro-averaged across configurations, which enables the computation of confidence intervals. These demonstrate that the observed improvements of CISC (for each confidence method examined) are strongly statistically significant.

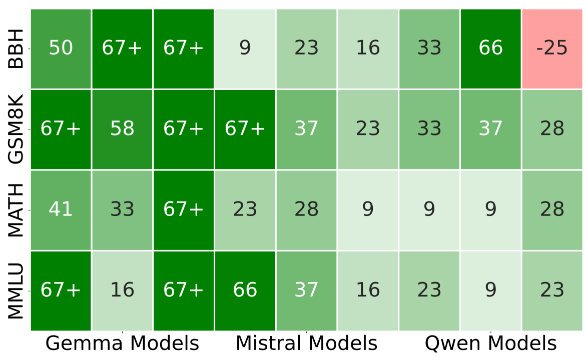

Figure 3: Results breakdown for CISC using the P(True) method and a budget of 10 responses per question. Each cell is annotated with the Cost Reduction (Percentage; §4.4) of CISC compared to self-consistency. As can be seen, CISC improves performance across almost all model families and datasets. In many cases, even 30 samples are not enough for self-consistency to reach CISC performance, leading to Cost Reduction of over 67%.

Confidence Normalization improves CISC's performance. We drill down into the importance of the within-question confidence normalization step in CISC. In Table 2, we compare CISC's performance with and without confidence normalization. We see that for every confidence method examined, CISC with normalization (softmax with a tunable temperature value) outperforms its unnormalized counterpart. However, as shown in Supplementary Table 8, normalization is effective only when using appropriate temperature hyperparameters. Because different confidence extraction methods produce scores on different scales, their optimal temperatures vary considerably (values are provided in Supplementary Figure 7). For instance, the P(True) method yields confidence values with high similarity, thus requiring lower temperatures to distinguish between them.

Confidence Method	Cost Reduction @ 10
P(True) (w/o normalization)	32% (14.8)
P(True) (w/ normalization)	**46% (18.6)**
SP (w/o normalization)	24% (13.1)
SP (w/ normalization)	**31% (14.6)**
Verbal (w/o normalization)	20% (12.5)
Verbal (w/ normalization)	**30% (14.4)**

Table 2: **CISC performance with and without confidence normalization** (bottom and top rows, respectively). We see that while CISC demonstrates substantial cost reductions even without normalization, employing normalization (Softmax and temperature scaling) significantly enhances performance, across all three confidence methods.

Confidence Method	ECE-t ↓	Brier-t ↓	WQD ↑	CISC Cost Reduction ↑
Verbal Binary	**0.005**	0.187	52.2%	10%
Verbal 0-100	0.046	**0.173**	56.1%	30%
Response Prob.	0.090	0.192	59.0%	31%
P(True)	0.030	0.182	**62.3%**	**46%**

Table 3: **Comparison of different confidence extraction methods in terms of between-question and within-question confidence evaluation metrics**. We see that between-question metrics (ECE-t, Brier-t) are poor indicators of effective confidence extraction for CISC, while our novel WQD metric (6.1) effectively predicts which confidence extraction method yields the best CISC performance.

6 Within-Question Confidence Evaluation

Recent work demonstrated that verbal confidence methods significantly outperform P(True) in terms of *calibration* (Tyen et al., 2023), which is the de-facto approach to evaluate the quality of confidence measures. Yet, perhaps surprisingly, CISC is more effective with P(True) than with verbal confidence methods (Table 1). In this section we settle these differences, and explain why well-calibrated confidence measures can still be less useful for CISC.

We argue that existing evaluation metrics, whether *calibration* based (Kadavath et al., 2022; Tian et al., 2023b) or *discrimination* based (Kuhn et al., 2023; Nguyen et al., 2024) examine the confidence behavior *between* the input questions. However, for CISC to work well, we want the confidence scores to be able to distinguish correct and incorrect responses *to the same question*.

To gain an intuition for the difference between *within-question* and *between-question* confidence evaluation, consider the following simple example. Imagine a model M and a dataset with two types of questions: questions that M finds "*easy*" (e.g., answers correctly 95% of the time) and questions that M finds "*hard*" (e.g., answers correctly 5% of the time). Consider a confidence measure that assigns every answer to an "*easy*" question a confidence of 0.95 and every answer to a hard question a confidence of 0.05. This confidence signal is useless for CISC, as it does not make any distinctions between answers to the same question. On the other hand, it scores well under existing metrics (e.g., it is perfectly calibrated).

The above thought experiment shows that the fact that well-calibrated confidence scores can be derived from a model does not necessarily imply the model possesses a capacity to self-assess its own responses. To isolate this specific ability, we design a metric that measures whether the confidence scores can distinguish correct and incorrect responses to the same question:

Definition 6.1 (Within-Question Discrimination)**.** *Given a dataset of questions, for each question q, denote the sampled responses by $R_q = \{(\boldsymbol{r}_i, \boldsymbol{a}_i)\}_{i=1}^m$, and let $R_q^+, R_q^- \subseteq R_q$ be the subsets of correct and incorrect responses respectively. We evaluate the Within-Question Discrimination (WQD) of a confidence method $c : (r, a) \mapsto \mathbb{R}$ as:*

$$\mathit{WQD}(c) \equiv$$
$$\frac{1}{N} \cdot \sum_q \sum_{(r,a)\in R_q^+} \sum_{(r',a')\in R_q^-} [c(r, a) > c(r', a')]$$

where $N = \sum_q |R_q^+| \cdot |R_q^-|$.

That is, we compute the fraction of cases where the higher confidence response is indeed the correct response, out of pairs of responses *to the same question* (exactly one of which is correct). In our work, we use $m = 30$ (as described in §4.6).

To emphasize the importance of *within-question evaluation*, we test if WQD is more predictive of CISC's success than standard *between-question* confidence metrics. We compare each confidence method from §4.1 in terms of: (i) standard metrics, such as ECE (Guo et al., 2017) and Brier Score (Brier, 1950), (ii) WQD, (iii) CISC performance at a budget of 10 samples. We follow previous work (Tyen et al., 2023) and report the standard metrics after applying temperature scaling (Ovadia et al., 2019), a technique that fits a single temperature parameter T to the model's confidences to minimize the negative log-likelihood on the data. We use ECE-t and Brier-t to denote the scaled scores.

The results of this comparison, averaged across all datasets (§4.2) and models (§4.3), are summarized in Table 3. Indeed, we see that the verbal confidence methods obtain the best ECE-t and Brier-t scores while also achieving the worst performance in CISC. On the other hand, the WQD metric is able to perfectly predict the relative scores of each confidence method in CISC. This emphasizes the limitations of relying solely on traditional confidence evaluation methods for evaluating the models ability to self-assess its reasoning.

The WQD metric prioritizes interpretability, focusing on the discrimination ability of the confi-

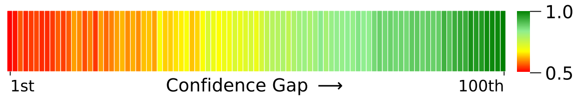

Figure 4: Within-Question Discrimination score (indicated by color) increases smoothly as a function of the confidence gap (percentiles, x-axis). Here we use the P(True) method, Gemma2-9B and the MATH dataset.

dence scores irrespective of the relative magnitude of the confidence values $c(r, a)$ and $c(r', a')$. However, examining the relationship between WQD and the confidence gap $|c(r, a) - c(r', a')|$ offers additional insights. Figure 4 illustrates a near monotonic relationship: the within-question discrimination ability (indicated by color) smoothly increases with the confidence gap (x-axis). These findings suggest a fine-grained self-assessment mechanism, where even small differences in confidence scores reflect significant variations in the probability of a response being correct

Taken together, our findings provide a compelling evidence that LLMs indeed posses an intrinsic ability to reassess their own responses.

7 Qualitative Analysis

In §5 we showed that CISC has clear performance advantages over standard self-consistency, and argued that this suggests LLMs are capable of self-assessing their confidence in responses to the same question. To facilitate a better understanding of this phenomenon, we asked human evaluators to identify indicators of *low-quality model responses* (i.e., logical patterns that reduced the evaluators' confidence in the correctness of the LLM response). Our analysis revealed a strong correlation between the prevalence of these indicators and lower confidence scores assigned by the LLM.

Sampling Process. We performed the analysis on MMLU-Pro (§4.2), using three representative models, one from each model family.

To reduce the evaluation burden we limited it to three LLM responses per question. We selected these triplets based on two criteria: (1) CISC and SC produced different results, where one method yielded a correct answer and the other did not, and (2) the final answers of the three responses were not all distinct, which would otherwise degenerate self-consistency's majority voting.

Out of the remaining triplets, we randomly chose 45 for which SC was correct and 45 where SC was wrong. Then, for each triplet, we randomly took either the response with highest relative-confidence or the response with lowest relative-confidence. This ensured an equal number of low relative-confidence responses that were correct and incorrect, mitigating potential bias of answer correctness on our analysis. The process resulted in 90 responses for human evaluation.

Human Evaluation. Two human evaluators (NLP Phd students), unaware of both the model's confidence scores and the ground truth labels, reviewed 90 samples. The evaluators' task was to identify logical patterns in the LLM reasoning-chain which reduce their confidence that the LLM has reached a correct answer; we call these patterns low-quality-indicators. Also, the evaluators were asked to briefly describe each identified pattern.

Results. Our evaluation demonstrated a significant correlation in confidence assessments: 67% of the samples assessed as relative-low confidence by the model were also judged to contain low-quality indicators by human evaluators, while only 33% of the samples assessed as relative-high confidence by the model contained the human identified low-quality-indicators. This strong correlation suggests that LLMs are adept at assessing their own reasoning processes and identifying patterns that humans consider indicative of low quality.

In addition, we categorized these low-quality indicators. Three primary categories emerged: (1) the LLM's final answer was not among the provided options; (2) the LLM deliberated between multiple options; and (3) the LLM omitted necessary calculations. Of these, only categories (1) and (3) showed a strong correlation with the LLM's low-confidence scores. Further details regarding these categories and their correlation statistics are available in the Appendix E.

8 Related Work

Confidence signals for LLMs. There is a long line of work on deriving confidence measures from LLMs. Popular approaches use the agreement across multiple samples (Kuhn et al., 2023; Manakul et al., 2023; Tian et al., 2023a; Lyu et al., 2024), the model's internal representations (Azaria and Mitchell, 2023; Burns et al., 2022) or directly prompting the model to verbalize its confidence (Tian et al., 2023b; Kadavath et al., 2022). All papers in this line of work focused on fact-seeking tasks, so confidence is typically derived based on the final answer alone. To the best of our knowledge, our work is the first to apply these approaches to scoring the entire reasoning path.

Reasoning verification. While learned verifiers have been demonstrated to significantly improve performance on math word problems (Cobbe et al., 2021b; Lightman et al., 2023; Li et al., 2022), the ability of LLMs to perform *self*-verification and *self*-correction is still heavily contested, with some works providing positive evidence for such capabilities (Weng et al., 2022; Gero et al., 2023; Madaan et al., 2024; Liu et al., 2024; Li et al., 2024a) and others arguing that the gains can mostly be attributed to clever prompt design, unfair baselines, data contamination and using overly simple tasks (Tyen et al., 2023; Valmeekam et al., 2023; Hong et al., 2023; Huang et al., 2023; Stechly et al., 2024; Zhang et al., 2024). This work contributes to this ongoing discussion by presenting multiple lines of evidence supporting LLM self-verification. In particular, we demonstrate clear benefits from a simple confidence-based self-verification approach.

Improving self-consistency's efficiency. Numerous attempts (Chen et al., 2024) have been made to reduce SC computational overhead while maintaining quality. However, none have matched the widespread adoption of self-consistency. This can be largely attributed to several limitations: (1) a trade-off where throughput is reduced while latency increases, for example by sampling chains sequentially until reaching a certain condition (Li et al., 2024b) or running expensive LLM calls instead of the cheap majority voting (Yoran et al., 2023), (2) the need for manual feature crafting and tuning tailored to each dataset (Wan et al., 2024), (3) promising results on specialized setups (Wang et al., 2024a) which did not generalize to standard benchmarks (Table 9), and (4) as highlighted by Huang et al. (2023), many of the more sophisticated methods that appear promising actually don't outperform self-consistency when evaluated with a thorough analysis of inference costs. Our approach is different in that CISC adds minimal complexity to self-consistency, and improves throughput without compromising latency.

Self-consistency with confidence. Related approaches to CISC's confidence-weighted majority vote were previously explored in both the original self-consistency paper Wang et al. (2022), that considered a weighted majority using Sequence Probability (§4.4), and in Miao et al. (2023), that concluded that verbally "*asking the LLM to check its own reasoning is largely ineffective*" for improving self-consistency. In both cases, these failures are attributed to the confidence scores being too similar to one another. Our work shows that despite this, the scores contain a useful signal (reflected in the WQD scores; Table 3) that can be utilized by a normalization step prior to aggregation to significantly improve the efficiency of self-consistency. Furthermore, the P(True) method, which achieves the highest WQD scores, has not been previously used for attempting to improve self-consistency.

9 Discussion

In this work we introduced CISC, a lightweight extension of self-consistency. Across diverse models, datasets, and confidence extraction methods, CISC consistently outperformed self-consistency, reducing computation costs by over 40% on average.

The performance gains achieved by using model-derived confidence scores provide a practical evidence that LLMs can effectively judge the quality of their own outputs, contributing to the ongoing debate on this topic (Huang et al., 2023; Li et al., 2024a). This is further strengthened by our qualitative evaluation, revealing significant agreement between model confidence and human assessments of response quality.

Complementing our investigation of LLM self-assessment, we address the crucial aspect of evaluating confidence methods. Traditional calibration metrics, which assess confidence across different questions, fail to capture a model's ability to distinguish between high and low quality responses to the same question. To overcome this, we introduce the Within-Question Discrimination (WQD) metric and demonstrate its effectiveness.

We encourage future research to explore the integration of model self-confidence into more sophisticated reasoning frameworks like Tree of Thoughts (Yao et al., 2024) or Graph of Thoughts (Besta et al., 2024), believing that harnessing this inherent capability can further boost performance. Another promising avenue is training models to produce more accurate intrinsic or verbal confidence (Lin et al., 2022; Chaudhry et al., 2024), which would directly improve CISC and related methods. Conversely, CISC and WQD can be used to assess the impact of advancements in confidence generation.

10 Limitations

Confidence Prompting. Our confidence extraction prompting approach minimizes the computational overhead (§4.1) by using short confidence prompts (less than 5% of the input and reasoning chain length) that, unlike other works, are appended after the reasoning chain. This allows us to continue to use the auto-regressive cache that was used when the models generated the answer. While this approach is readily implementable within frameworks like HuggingFace (Hugging-Face, 2024a), it may not be universally supported. An alternative one-step prompting approach, which does not rely on prefix caching, is discussed in Appendix B. We opted for the two-step approach in this study to ensure a clear and robust evaluation of CISC, fully mitigating the impact of confidence integration on the generated reasoning paths.

Access to the model's probabilities. The preferred CISC approach calculates P(True) (as described in §4.1) by examining the model's assigned probability to the verbal confidence token. This method is available in both popular open-weights frameworks (e.g., Hugging-Face (2024a)) and closed-weights frameworks (e.g., OpenAI (2025)). However, this feature may not be universally available across all frameworks.

Human Evaluation. The qualitative human evaluation presented in Section 7 provides further support for our claims regarding LLMs' ability to self-assess the correctness of their responses. This evaluation was conducted on the MMLU dataset, which offers a diverse set of single-choice questions. Extending this analysis to other datasets could offer additional insights.

Additional ablations. We examined the performance of CISC across several key aspects, focusing on the impact of the choice of confidence extraction method and the impact of the confidence normalization step. Additional ablations could include examining the effect of zero-shot vs few-shot prompting, different choices of normalization techniques, and using trainable confidence methods (Lin et al., 2022; Chaudhry et al., 2024) to improve the performance of CISC.

11 Ethics Statement

This work improves LLM reasoning efficiency by introducing a new decoding strategy (CISC). While CISC itself introduces no new ethical issues, LLMs can perpetuate biases and have societal impacts. Responsible LLM development and deployment, including bias mitigation, are crucial.

References

Amos Azaria and Tom Mitchell. 2023. The internal state of an llm knows when its lying. *arXiv preprint arXiv:2304.13734*.

BIG bench authors. 2023. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research*.

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

Glenn W Brier. 1950. Verification of forecasts expressed in terms of probability. *Monthly weather review*, 78(1):1–3.

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. 2023. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*.

Collin Burns, Haotian Ye, Dan Klein, and Jacob Steinhardt. 2022. Discovering latent knowledge in language models without supervision. *arXiv preprint arXiv:2212.03827*.

Arslan Chaudhry, Sridhar Thiagarajan, and Dilan Gorur. 2024. Finetuning language models to emit linguistic expressions of uncertainty. *arXiv preprint arXiv:2409.12180*.

Wenqing Chen, Weicheng Wang, Zhixuan Chu, Kui Ren, Zibin Zheng, and Zhichao Lu. 2024. Self-para-consistency: Improving reasoning tasks at low cost for large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 14162–14167, Bangkok, Thailand. Association for Computational Linguistics.

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. 2021a. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

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

Zelalem Gero, Chandan Singh, Hao Cheng, Tristan Naumann, Michel Galley, Jianfeng Gao, and Hoifung Poon. 2023. Self-verification improves few-shot clinical information extraction. *arXiv preprint arXiv:2306.00024*.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1321–1330. PMLR.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *NeurIPS*.

Ruixin Hong, Hongming Zhang, Xinyu Pang, Dong Yu, and Changshui Zhang. 2023. A closer look at the self-verification abilities of large language models in logical reasoning. *arXiv preprint arXiv:2311.07954*.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. 2023. Large language models cannot self-correct reasoning yet. *arXiv preprint arXiv:2310.01798*.

Hugging-Face. 2024a. Hugging face. `https://huggingface.co/`.

Hugging-Face. 2024b. Hugging face leaderboard. `https://huggingface.co/spaces/open-llm-leaderboard/open_llm_leaderboard#/`.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720*.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, et al. 2022. Language models (mostly) know what they know. *arXiv preprint arXiv:2207.05221*.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. 2023. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. *arXiv preprint arXiv:2302.09664*.

Loka Li, Zhenhao Chen, Guangyi Chen, Yixuan Zhang, Yusheng Su, Eric Xing, and Kun Zhang. 2024a. Confidence matters: Revisiting intrinsic self-correction capabilities of large language models. *arXiv preprint arXiv:2402.12563*.

Yifei Li, Zeqi Lin, Shizhuo Zhang, Qiang Fu, Bei Chen, Jian-Guang Lou, and Weizhu Chen. 2022. Making large language models better reasoners with step-aware verifier. *arXiv preprint arXiv:2206.02336*.

Yiwei Li, Peiwen Yuan, Shaoxiong Feng, Boyuan Pan, Xinglin Wang, Bin Sun, Heda Wang, and Kan Li. 2024b. Escape sky-high cost: Early-stopping self-consistency for multi-step reasoning. *arXiv preprint arXiv:2401.10480*.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. *arXiv preprint arXiv:2305.20050*.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. Teaching models to express their uncertainty in words. *arXiv preprint arXiv:2205.14334*.

Dancheng Liu, Amir Nassereldine, Ziming Yang, Chenhui Xu, Yuting Hu, Jiajie Li, Utkarsh Kumar, Changjae Lee, and Jinjun Xiong. 2024. Large language models have intrinsic self-correction ability. *arXiv preprint arXiv:2406.15673*.

Qing Lyu, Kumar Shridhar, Chaitanya Malaviya, Li Zhang, Yanai Elazar, Niket Tandon, Marianna Apidianaki, Mrinmaya Sachan, and Chris Callison-Burch. 2024. Calibrating large language models with sample consistency. *arXiv preprint arXiv:2402.13904*.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. 2024. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems*, 36.

Potsawee Manakul, Adian Liusie, and Mark JF Gales. 2023. Selfcheckgpt: Zero-resource black-box hallucination detection for generative large language models. *arXiv preprint arXiv:2303.08896*.

Ning Miao, Yee Whye Teh, and Tom Rainforth. 2023. Selfcheck: Using llms to zero-shot check their own step-by-step reasoning. *arXiv preprint arXiv:2308.00436*.

Mistral-AI. 2024. Mistral models. `https://mistral.ai/news/ministraux/`.

Minh-Vuong Nguyen, Linhao Luo, Fatemeh Shiri, Dinh Phung, Yuan-Fang Li, Thuy-Trang Vu, and Gholamreza Haffari. 2024. Direct evaluation of chain-of-thought in multi-hop reasoning with knowledge graphs. *arXiv preprint arXiv:2402.11199*.

OpenAI. 2025. Openai api. https://platform.openai.com/docs/api-reference/chat/create.

Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, David Sculley, Sebastian Nowozin, Joshua Dillon, Balaji Lakshminarayanan, and Jasper Snoek. 2019. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. *Advances in neural information processing systems*, 32.

Kaya Stechly, Karthik Valmeekam, and Subbarao Kambhampati. 2024. On the self-verification limitations of large language models on reasoning and planning tasks. *arXiv preprint arXiv:2402.08115*.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V Le, Ed H Chi, Denny Zhou, , and Jason Wei. 2022. Challenging big-bench tasks and whether chain-of-thought can solve them. *arXiv preprint arXiv:2210.09261*.

Gemma Team, 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*.

Katherine Tian, Eric Mitchell, Huaxiu Yao, Christopher D Manning, and Chelsea Finn. 2023a. Fine-tuning language models for factuality. *arXiv preprint arXiv:2311.08401*.

Katherine Tian, Eric Mitchell, Allan Zhou, Archit Sharma, Rafael Rafailov, Huaxiu Yao, Chelsea Finn, and Christopher D Manning. 2023b. Just ask for calibration: Strategies for eliciting calibrated confidence scores from language models fine-tuned with human feedback. *arXiv preprint arXiv:2305.14975*.

Gladys Tyen, Hassan Mansoor, Peter Chen, Tony Mak, and Victor Cărbune. 2023. Llms cannot find reasoning errors, but can correct them! *arXiv preprint arXiv:2311.08516*.

Karthik Valmeekam, Matthew Marquez, and Subbarao Kambhampati. 2023. Can large language models really improve by self-critiquing their own plans? *arXiv preprint arXiv:2310.08118*.

Guangya Wan, Yuqi Wu, Jie Chen, and Sheng Li. 2024. Dynamic self-consistency: Leveraging reasoning paths for efficient llm sampling. *arXiv preprint arXiv:2408.17017*.

Han Wang, Archiki Prasad, Elias Stengel-Eskin, and Mohit Bansal. 2024a. Soft self-consistency improves language model agents. *arXiv preprint arXiv:2402.13212*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. 2024b. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *arXiv preprint arXiv:2406.01574*.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, et al. 2024c. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *arXiv preprint arXiv:2406.01574*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Yixuan Weng, Minjun Zhu, Fei Xia, Bin Li, Shizhu He, Shengping Liu, Bin Sun, Kang Liu, and Jun Zhao. 2022. Large language models are better reasoners with self-verification. *arXiv preprint arXiv:2212.09561*.

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, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2024. Tree of thoughts: Deliberate problem solving with large language models. *Advances in Neural Information Processing Systems*, 36.

Ori Yoran, Tomer Wolfson, Ben Bogin, Uri Katz, Daniel Deutch, and Jonathan Berant. 2023. Answering questions by meta-reasoning over multiple chains of thought. *arXiv preprint arXiv:2304.13007*.

Yunxiang Zhang, Muhammad Khalifa, Lajanugen Logeswaran, Jaekyeom Kim, Moontae Lee, Honglak Lee, and Lu Wang. 2024. Small language models need strong verifiers to self-correct reasoning. *arXiv preprint arXiv:2404.17140*.

A Quantitative example from §3

Consider a simplified *binary* setting in which there are two possible answers: correct and incorrect. Given a number of samples n and a probability $p = 0.6$ of generating the correct answer, the number of samples with the correct answer follows the Binomial distribution $X \sim \text{Binomial}(n, p)$. For such distribution, the majority vote is accurate whenever $X > \frac{n}{2}$ and it has 50% chance to be accurate when $X = \frac{n}{2}$ (i.e., a random choice).

Now, to illustrate how the self-assessment score of LLMs can be helpful, consider that we have an oracle that assigns twice the weight for answers that are correct. In this case, a weighted majority vote would be accurate whenever $X > \frac{n}{3}$ and it has 50% chance to be accurate when $X = \frac{n}{3}$.

In Figure 5 we plot the relationship between, (x-axis) the number of samples, and (y-axis) the accuracy of the *weighted* majority vote over these samples. The graph features two lines: (blue) each sample gets an equal weight, and (orange) correct answers are assigned twice the weight of incorrect ones.

While this intuition about cost-saving also applies to the general case of an *arbitrary* set of answers, this setting is trickier to analyze in closed-form because the specific distribution of incorrect answers impacts the majority vote. E.g., an answer that appears only 20% of the time can still be correct under majority vote if all the other 80% incorrect answers are different from one another. This could be obtained by placing additional distributional assumptions on the sampled answers. The analysis of the binary case can be thought of as a worst-case analysis of the general case, since in the worst case, all the incorrect answers are identical and the majority will be accurate if and only if more than half the sampled answers are correct.

B Prompting Techniques

As described in Section 4.1, for our prompt based confidence extraction techniques (Verbal Confidence, P(True)), we used a two-step approach: First, we prompted the model to answer benchmark questions using the prompts shown in Table 4. Then, we extracted confidence by concatenating the prompts shown in Table 5 and running the model again. This two-step process allowed using the same answers when comparing self-consistency and CISC.

While a simpler single-step implementation (outputting both answer and confidence in a single response) is possible, we did not explore it in this study. For research purposes, we prioritized a clean setup that ensured requesting confidence scores did not influence the generated answers and chain-of-thoughts.

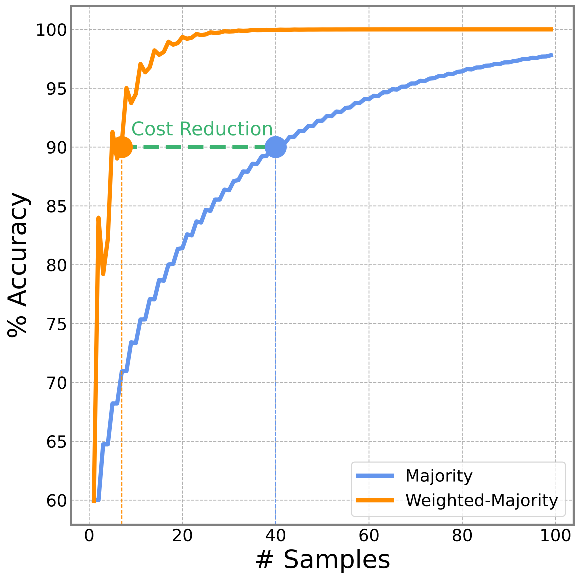

Figure 5: The relationship between the number of samples (x-axis) and the accuracy of majority vote over these samples (y-axis), for two different hypothetical cases: (blue) Each sample receives an equal weight in majority voting, and (orange) Correct answers are assigned double the weight of incorrect ones. Adding this additional weighting information translated into $4X$ reduction in the number of samples required for the majority vote to reach 90% accuracy.

As shown in Table 5, all the confidence extraction prompts that we used are extremely lightweight. We deliberately avoided methods that significantly increase the number of generated tokens like generating k guesses with associated probabilities (Tian et al., 2023b).

For the P(True) method, we modified the prompts from Kadavath et al. (2022) in two ways: (1) We changed the format to allow concatenation after the model provided its answer, ensuring that prefix caching could be re-used between the two steps. (2) We changed the prompt format to 0/1 instead of True/False, as some benchmarks are using True/False as ground truth labels, and we observed that it might confuse the model when extracting confidence scores.

C Additional Results

For each confidence method, Table 1 shows the macro-average results across all models and datasets. A more detailed version of this table, with a per dataset breakdown, is given at Table 6.

In addition, Table 7 shows micro-averaged aggregated results with confidence intervals, demonstrating the strong statistical significance of our findings. These bootstrap confidence intervals were calculated as follows: (1) For each confidence method, results from all datasets and models were combined into a single dataset of approximately $n \approx 150,000$ rows. (2) 10,000 bootstrap sets were generated by repeatedly sampling n elements with replacement. (3) The procedure described in 4.6 was applied to each set, yielding 10,000 estimates of the mean accuracy difference. (4) We used these estimates to calculate the 95% interval.

Table 8 is an extended version of table 2. One important insight that can be derived from the extended table, is that using softmax normalization without temperature scaling is strongly discouraged for CISC.

We also add Figure 6 featuring additional graphs similar to Figure 1, but with all the confidence methods.

Finally, in Table 9, we include ablations comparing CISC's weighted majority mechanism to more simple methods like selecting the max confidence (Wang et al., 2024a) or using the confidence values as a tie-breaker for self-consistency.

D Temperature Scaling Results

As discussed in §4.5, a single optimal temperature, $T^{\star}$, was determined for each model and confidence extraction method by using a 10% held-out set, aggregated across all datasets. Fitting is done using grid search on values between 10^{-4} and 10^{4}. The temperatures for each configuration are presented in Figure 7. As can be seen, each of the confidence extraction method work with a different temperature magnitude because it produce confidence values on a different scale.

E Qualitative Appendix

The qualitative analysis presented in §7 involved sampling the reasoning paths using three models: Qwen2.5 3B, Gemma2 9B and Mistral Large (123B). To broaden our evaluated sample pool, we employed a bootstrap process, sampling three distinct traces per question multiple times. Then, we first filtered these samples so that each of them arrived from a different question, and continued with the sampling process described in §7.

Human evaluators were asked to identify logical patterns in the LLMs' reasoning paths that reduced the evaluators' confidence in the correctness of the LLMs' answers. Importantly, the MMLU dataset requires significant domain knowledge and unspecialized humans achieved only 34.5% accuracy (Hendrycks et al., 2020), compared to a random baseline of 25%. The MMLU-pro dataset is based on the MMLU dataset, but is considered much harder. This means that our evaluators, which lacked specialized knowledge, could not easily how to solve each question. Instead, we instructed them to focus on identifying low-quality reasoning errors in the responses of the LLMs. This approach aligns with findings from a prior analysis on GPT-4o (Wang et al., 2024b), which attributed 39% of its errors to reasoning flaws that do not rely on specialized domain knowledge.

Following this review, we aggregated the indicators of low quality into high-level categories. Three main categories encompassed 49% of the samples. The remaining samples either lacked low-quality indicators (50%) or had indicators that did not fit into a sizable category (1%). The different categories and their prevalence are presented in Table 11.

Two of these three categories show a strong association with relative-low confidence scores from the model: (1) The model arrived at solutions not present among the available options, and (2) The model only conducted partial calculations necessary. Interestingly, the pattern where the model explores several plausible solutions without identifying a definitive "correct" one was not specifically associated with either high or low confidence in the model's reasoning paths, underscoring that not all human-identified patterns significantly influence the model's assessment.

Overall, the alignment of human-identified low-quality indicators with low-confidence scores provides another evidence of the ability of LLMs to self-assess and prioritize high confidence solutions. An ability that is leveraged by CISC.

F Compute

For each model (§4.3), we generated approximately 500,000 responses - 17,000 questions (§4.2), with 30 samples (§4.6). As a reference, inference with Gemma2-2-Billion (1K token context length) required an order of 100 Nvidia H100 GPU hours.

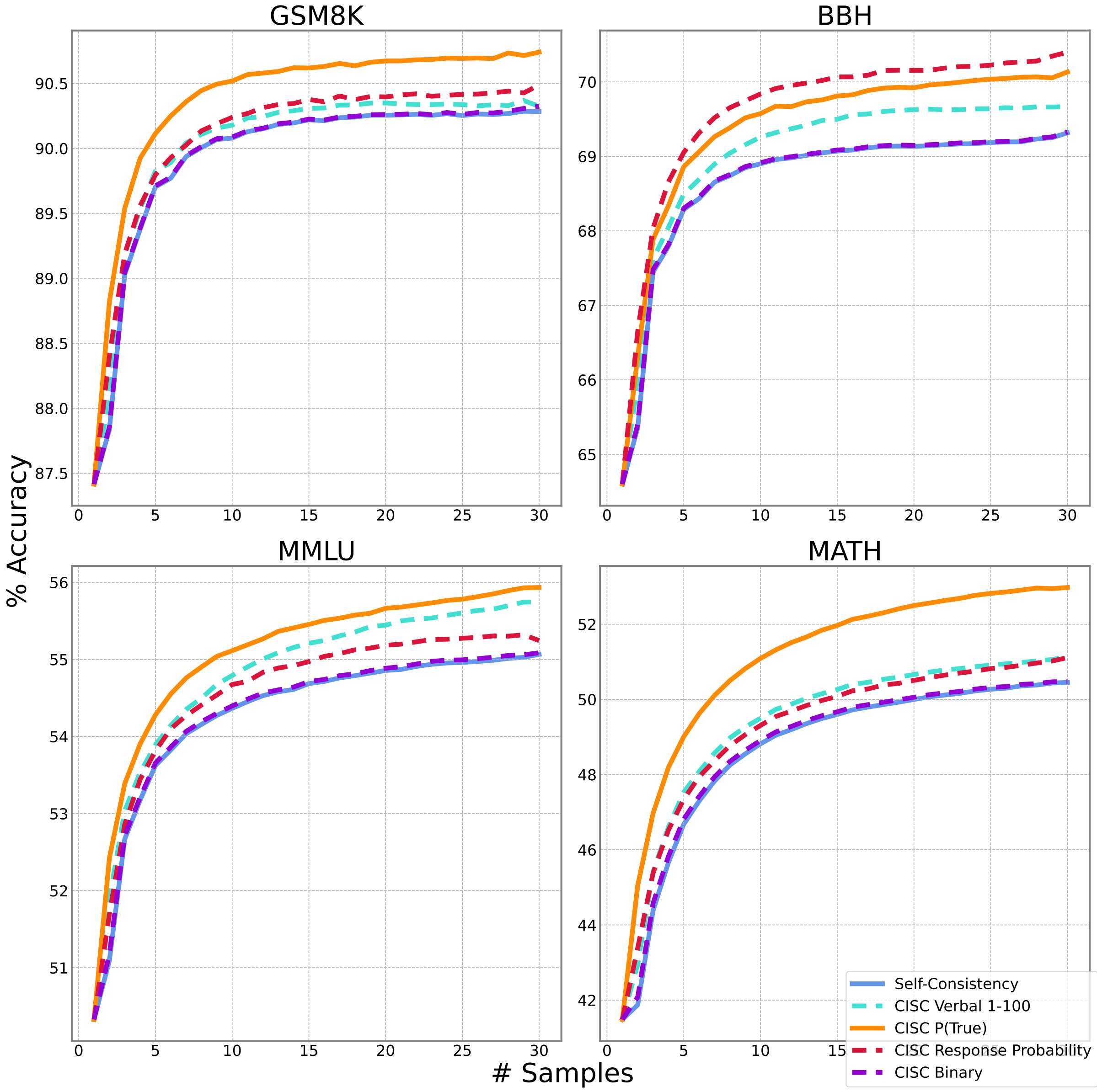

Figure 6: Comparison between different confidence extraction methods using Gemma2-9B model and four datasets (§4.2). CISC with P(True) outperforms Self-Consistency and is the best of all the CISC variants.

General Instructions
Before giving your answer, provide a step-by-step explanation of your thought process. Then on a new line, give your proposed answer adhering to this precise format: 'Proposed answer: (X).', where X is your proposed answer.

Dataset	Prompt
MMLU-Pro	You will be given a single-choice question. Answer the question by selecting the letter of the best fitting option. *[General Instructions]* The answer MUST ALWAYS be the letter of one of the available options; it CANNOT be "None of the Above".
MATH	You will be given a question and your goal is to answer it correctly.\nYour proposed answer should be a TeX expression, such as '5', '3.14', or '$\\sqrt{8}$ *[General Instructions]*
BBH (no options)	You will be given a question and your goal is to answer it correctly. *[General Instructions]*
BBH (with options)	You will be given a question and your goal is to answer it correctly. *[General Instructions]* Select the letter of the best fitting option. The answer CANNOT be "None of the Above".
GSM8K	You will be given a question and your goal is to answer it correctly. *[General Instructions]*

Table 4: The prompts used to generate model responses for benchmark questions. For all datasets, we used the *General Instructions* (shown at the top) asking the model to solve each question step-by-step and provide its final answer in a specified format. In addition, for each dataset we briefly explained the expected questions format. All prompts were zero-shot; few-shot experiments are reserved for future work.

Confidence Method	Prompt
Verbal 0-100	Now I will rate my confidence in the proposed answer on a scale of 0-100. Proposed confidence: (
Verbal Binary	Now I will rate my confidence in the proposed answer as either 0 or 1. Proposed confidence: (

Table 5: The prompts used to extract the model confidence in its response. As explained in section B, these prompts are concatenated as a second step, after the model already answers the question. For the P(True) method, we used the Verbal Binary prompt and looked at the probably the model assigns to the token 1. Importantly, in all the models evaluated in this work, "(0" and "(1" are tokenized as two separate tokens.

		Comparable SC Samples		Acc Improvement (%)	
Dataset	Confidence Method	Budget 5	Budget 10	Budget 5	Budget 10
MMLU	Verbal Binary	18% (6.1)	12% (11.3)	0.4	0.2
	Verbal 1-100	25% (6.7)	32% (14.6)	0.9	0.7
	Response Probability	17% (6.0)	23% (13.0)	0.7	0.6
	P(True)	**37% (7.9)**	**47% (18.8)**	**1.4**	**1.0**
MATH	Verbal Binary	18% (6.1)	11% (11.2)	0.8	0.5
	Verbal 1-100	17% (6.0)	12% (11.3)	1.3	0.6
	Response Probability	19% (6.2)	17% (12.0)	2.2	1.2
	P(True)	**32% (7.3)**	**34% (15.2)**	**3.0**	**2.0**
GSM8K	Verbal Binary	18% (6.1)	7% (10.8)	0.2	0.1
	Verbal 1-100	22% (6.4)	32% (14.6)	0.3	0.1
	Response Probability	21% (6.3)	33% (14.9)	0.7	0.5
	P(True)	**43% (8.8)**	**53% (21.2)**	**0.9**	**0.6**
BBH	Verbal Binary	17% (6.0)	10% (11.1)	0.2	0.1
	Verbal 1-100	22% (6.4)	41% (17.0)	0.5	0.4
	Response Probability	32% (7.3)	45% (18.3)	0.7	0.8
	P(True)	**48% (9.7)**	**47% (19.0)**	**1.0**	**0.9**

Table 6: Aggregated results across all models for each dataset and confidence extraction method. All methods demonstrate better performance than standard self-consistency, with the P-True method achieving the best results and leading to an computational cost reduction of up to 53%

	Acc Improvement	
Confidence Method	Budget 5	Budget 10
Verbal Binary	0.35 (0.34-0.37)	0.20 (0.18-0.21)
Verbal 1-100	0.68 (0.64-0.72)	0.46 (0.40-0.51)
Response Probability	0.88 (0.84-0.92)	0.69 (0.63-0.74)
P(True)	**1.38 (1.32-1.43)**	**1.03 (0.96-1.10)**

Table 7: **Micro-averaged Aggregated Results.** This table presents the micro-averaged aggregated results with confidence intervals for each confidence method. Each confidence method demonstrates statistically significant improvements over self-consistency, and **P(True)** method exhibits significant superiority over other methods. This detailed view supplements the macro-average results shown in Table 1 and provides statistical verification of the efficiency gains and accuracy improvements attributed to CISC methods.

Confidence Method	% Cost Reduction 5	10	% Acc Improvement 5	10	30
P(True) - No Normalization	29% (7.0)	32% (14.8)	1.4	0.8	0.4
P(True) - Softmax T=1	27% (6.8)	30% (14.2)	1.3	0.8	0.3
P(True) - Softmax T=Tuned	**41% (8.4)**	**46% (18.6)**	**1.6**	**1.1**	**0.9**
Sequence Probability - No Normalization	21% (6.3)	24% (13.1)	1.1	0.6	0.3
Sequence Probability - Softmax T=1	20% (6.3)	23% (13.0)	1.1	0.6	0.2
Sequence Probability - Softmax T=Tuned	**22% (6.5)**	**31% (14.6)**	**1.1**	**0.8**	**0.7**
Verbal 0 - 100 - No Normalization	20% (6.3)	20% (12.5)	0.7	0.4	0.1
Verbal 0 - 100 - Softmax T=1	12% (5.7)	-1% (9.9)	-0.3	-1.4	-2.6
Verbal 0 - 100 - Softmax T=Tuned	**22% (6.4)**	**30% (14.4)**	**0.8**	**0.4**	**0.3**

Table 8: **Normalization Ablation.** This table extends Table 2, showing that temperature-scaled softmax is optimal for all methods, and that softmax should be avoided without temperature scaling.

	Comparable SC Samples		Acc Improvement (%)	
Confidence Method	Budget 5	Budget 10	Budget 5	Budget 10
Max	-11% (4.5)	-84% (5.4)	-1.9	-4.5
Tie	27% (6.8)	28% (13.9)	1.3	0.7
CISC	**41% (8.4)**	**46% (18.6)**	**1.6**	**1.1**

Table 9: **Simplified ablation.** Here we compare CISC with two simplified ablations: (Max) Which selects the answer with highest confidence score, and (Tie) Only uses CISC if self-consistency resulted in a tie. All methods are calculated using the P(True) confidence. Results are aggregated across all models and datasets. CISC significantly outperforms both ablations, and the Max method even degenerates performance.

Dataset Model	BBH	GSM8K	MATH	MMLU
Gemma 27b	57.1	66.1	62.9	59.9
Gemma 2b	55.8	66.2	64.3	53.6
Gemma 9b	55.3	68.3	71.8	58.9
Mistral 123	56.2	66.1	61.2	63.4
Mistral 22	64.1	81.4	74.9	67.7
Mistral 8	59.4	71.8	62.9	58.8
Qwen 14b	58.9	65.5	59.0	60.2
Qwen 3b	56.3	61.9	57.5	56.0
Qwen 72b	53.5	62.4	63.6	58.8

Table 10: **Within-Question-Discrimination Breakdown.** This table presents a breakdown of the aggregated Within-Question-Discrimination (WQD) results presented in Table 3, using the P(True) method. In all cases, WQD scores exceed the 50% chance level.

Category	Definition	Low	High	Snippet
No choice	The model arrives at a solution which is not present in the list of available options. This can include case where a mathematical answer significantly diverging from all options, answers that are only partially correct, or the elimination of all options as part of the reasoning process.	38%	13%	"... After reviewing the options, it's clear that none of them perfectly fit the requirements. However, the closest correct option is (A), which only has a minor error in calculating the remaining inches. Proposed answer: (A)"
Incomplete Calculations	The model begins to solve the problem but does not complete the full calculation, often due to the lack of necessary data. For example, when attempting to compute acceleration, the absence of mass data prevents an exact and full calculation.	22%	2%	"...**Calculate Heat Flow:** q" = h * (T-surface - T-air) **Note:** Without the actual values for air density, viscosity, and thermal conductivity at 68°F, we cannot perform the precise calculations. Proposed answer: (C)."
Multiple candidates	The model explores several plausible solutions without identifying a definitive "correct" one. This occurs when the model solves a problem generally, relying on estimations rather than concrete data, resulting in a range of potential answers.	11%	16%	"... 2.**Identify Buddhist Thinkers:** The options list several prominent Buddhist figures from various traditions... 4. **Most Prominent:** The Dalai Lama and Thich Nhat Hanh stand out for their consistent emphasis on self-sacrifice in their teachings and actions. Proposed answer: (I)"

Table 11: Human evaluators identified low-quality reasoning indicators in LLM responses (see §7). These indicators were then clustered into three categories, each described above with a definition and an example snippet from an LLM response. The (Low, High) columns show the percentage of LLM responses with low/high self-assessed confidence that exhibited each pattern. The "No Choice" and "Incomplete Calculation" categories are strongly associated with low confidence.

General Instructions	Evaluate the LLMs' reasoning paths, looking for logical inconsistencies or errors that lower your confidence in their conclusions. Because the questions are very difficult, even for experts, your task is to identify general reasoning flaws, not to assess the correctness of the final answers themselves. Examples: • Incorrect Assumption: The model assumes something without justification • Missing Step: The model skips a crucial step in the reasoning process • Contradiction: The model states both A and not-A
Question	`[Pre-filled - The original question given to the LLM]`
LLM Output	`[Pre-filled - The LLM output for the given question]`

Table 12: The input given to human evaluators as part of our qualitative analysis (§7).

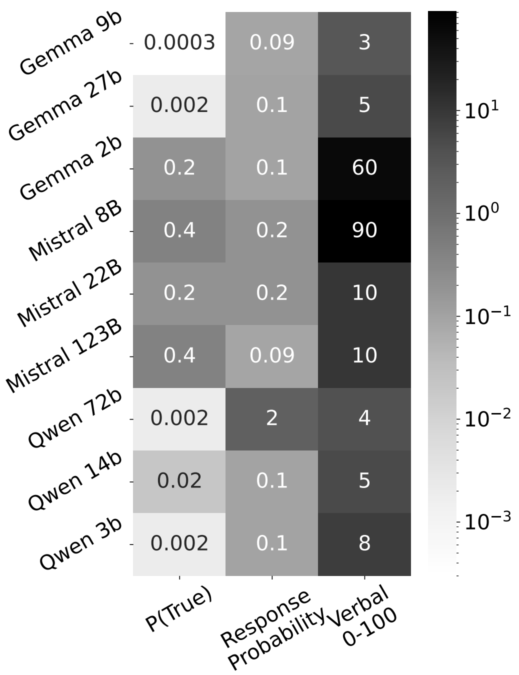

Figure 7: The best temperatures values for each model / confidence-method combination. As discussed in Section 4.5, we fit a single temperature hyper-parameter across 10% of all datasets together. As can be seen, each of the confidence extraction method work with a different temperature magnitude. We also see variability between models using the same confidence extraction method.