CER: Confidence Enhanced Reasoning in LLMs

Anonymous ACL submission

Abstract

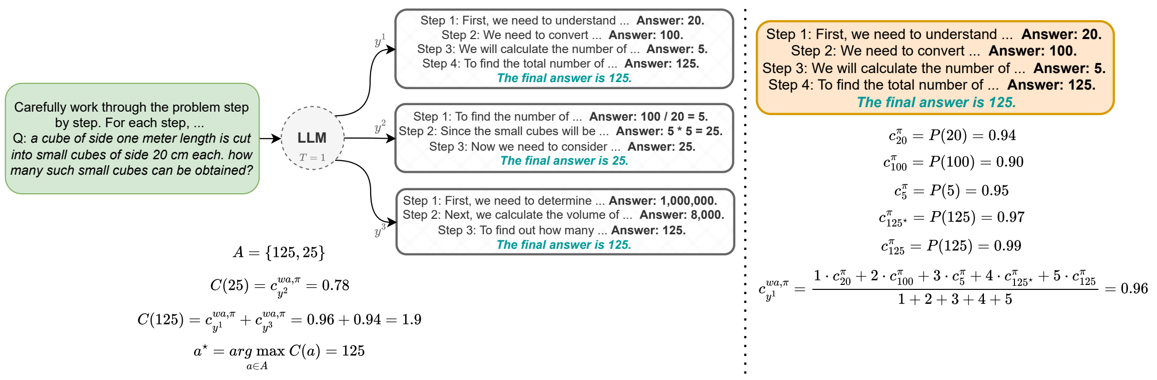
Ensuring the reliability of Large Language Models (LLMs) in complex reasoning tasks remains a formidable challenge, particularly in scenarios that demand precise mathematical calculations and knowledge-intensive open-domain generation. In this work, we introduce an uncertainty-aware framework designed to enhance the accuracy of LLM responses by systematically incorporating model confidence at critical decision points. We propose an approach that encourages multi-step reasoning in LLMs and quantify the confidence of intermediate answers such as numerical results in mathematical reasoning and proper nouns in open-domain generation. Then, the overall confidence of each reasoning chain is evaluated based on confidence of these critical intermediate steps. Finally, we aggregate the answer of generated response paths in a way that reflects the reliability of each generated content (as opposed to self-consistency in which each generated chain contributes equally to majority voting). We conducted extensive experiments in five datasets, three mathematical datasets and two open-domain datasets, using four LLMs. The results consistently validate the effectiveness of our novel confidence-aggregation method, leading to an accuracy improvement of up to 7.4% and 5.8% over baseline approaches in math and open-domain generation tasks, respectively. The code is available anonymously at CER Repository.

1 Introduction

Recently, Large Language Models (LLMs) (Dubey et al., 2024; Guo et al., 2025; Jiang et al., 2023; Groeneveld et al., 2024; Achiam et al., 2023) have garnered significant attention for their strong performance across diverse reasoning tasks, including arithmetic reasoning and open-domain question answering (Wei et al., 2022; Marasovic et al., 2022; Zelikman et al., 2022; Kojima et al., 2022; Yang et al., 2024b). Approaches such as self-consistency (Wang et al., 2022) and few-shot prompting (Brown et al., 2020) have also been introduced to enhance the reasoning process of these models. However, these approaches have notable limitations. For instance, few-shot prompting relies on carefully curated demonstrations to perform well, and poorly chosen ones can have a reverse effect on performance (Halawi et al., 2023). In addition, the self-consistency method faces challenges in scenarios where generated paths either (1) produce inconsistent answers that do not include the correct solution or (2) predominantly converge on incorrect results (Zhang et al., 2023; Wang and Zhou, 2024).

Besides that, human intelligence is uniquely characterized by its ability to express and communicate uncertainty, a critical skill for sound decision-making and effective collaboration (Cosmides and Tooby, 1996). Similarly, in artificial intelligence, accurate uncertainty estimation is essential for risk assessment, error mitigation, and reliable decision-making (Blundell et al., 2015; Guo et al., 2017; Tomani and Buettner, 2021; Fadeeva et al., 2024). To improve the reasoning capabilities of LLMs, it is essential to equip them with mechanisms for effectively quantifying and leveraging uncertainty.

In this work, we aim to improve reasoning by incorporating uncertainty estimation within a Chain-of-Thought (CoT) process, which consists of a sequence of steps that generate intermediate outputs or answers and ultimately leading to the final answer. At the end of each step, the model is expected to arrive at a certain level of confidence in its output, while some degree of uncertainty is natural throughout a thought due to an incomplete or evolving reasoning step. As a result, we hypothesize that the overall undesired uncertainty of the reasoning chain can be inferred by analyzing the confidence of the tokens that make up the intermediate and final answers. Additionally, these intermediate outputs often exhibit specific characteristics, such as numerical values or proper nouns, that can be read-



Carefully work through the problem step by step. For each step, ...
Q: a cube of side one meter length is cut into small cubes of side 20 cm each. how many such small cubes can be obtained?
LLM
$T=1$
y^1
y^2
y^3
Step 1: First, we need to understand ... Answer: 20.
Step 2: We need to convert ... Answer: 100.
Step 3: We will calculate the number of ... Answer: 5.
Step 4: To find the total number of ... Answer: 125.
The final answer is 125.
Step 1: To find the number of ... Answer: 100 / 20 = 5.
Step 2: Since the small cubes will be ... Answer: 5 * 5 = 25.
Step 3: Now we need to consider ... Answer: 25.
The final answer is 25.
Step 1: First, we need to determine ... Answer: 1,000,000.
Step 2: Next, we calculate the volume of ... Answer: 8,000.
Step 3: To find out how many ... Answer: 125.
The final answer is 125.
$A = \{125, 25\}$
$C(25) = c^{wa,\pi}_{y^2} = 0.78$
$C(125) = c^{wa,\pi}_{y^1} + c^{wa,\pi}_{y^3} = 0.96 + 0.94 = 1.9$
$a^* = arg\max_{a \in A} C(a) = 125$
Step 1: First, we need to understand ... Answer: 20.
Step 2: We need to convert ... Answer: 100.
Step 3: We will calculate the number of ... Answer: 5.
Step 4: To find the total number of ... Answer: 125.
The final answer is 125.
$c^{\pi}_{20} = P(20) = 0.94$
$c^{\pi}_{100} = P(100) = 0.90$
$c^{\pi}_{5} = P(5) = 0.95$
$c^{\pi}_{125^*} = P(125) = 0.97$
$c^{\pi}_{125} = P(125) = 0.99$
$c^{wa,\pi}_{y^1} = \frac{1 \cdot c^{\pi}_{20} + 2 \cdot c^{\pi}_{100} + 3 \cdot c^{\pi}_{5} + 4 \cdot c^{\pi}_{125^*} + 5 \cdot c^{\pi}_{125}}{1+2+3+4+5} = 0.96$

Figure 1: **Illustration of Confidence-Enhanced Reasoning (CER) in LLMs.** On the left, we demonstrate the CER framework. Given an input query, the LLM generates three independent outputs using temperature sampling ($T = 1$). Intermediate answers are bolded, and final answers are highlighted. The confidence of each output is computed, and the most weighted-confident answer—125—is selected. On the right, we illustrate the confidence calculation for the first output. We use multiplication as the step-wise aggregator function (f) and weighted averaging (wa) as the path-wise aggregator function (g). Since the answer 125 appears in both step 4 and the final answer, we mark its first occurrence with * for clarity. The full question and responses from the LLM are provided in Appendix F.

ily identified. In fact, we consider these critical tokens in our uncertainty estimation process to enhance the overall accuracy of the reasoning. For mathematical tasks (e.g., GSM8K (Cobbe et al., 2021)), we prioritize confidence in numerical tokens, while for open-domain generation reasoning (e.g., TriviaQA (Joshi et al., 2017)), we focus on the model's confidence in proper nouns (entities, names, locations).

Based on the above idea, our method comprises three key components: (1) a confidence estimation technique that focuses on evaluating confidence in specific tokens, where a high degree of certainty is crucial, (2) an aggregation strategy for integrating confidence scores across a reasoning chain, and (3) a function that ensembles answers by harnessing the uncertainty within each reasoning chain, resulting in enhanced performance compared to ensemble reasoning methods such as self-consistency.

We evaluated our framework on four LLMs (Llama 3.1, Llama 3.2 (Dubey et al., 2024), OLMo 2 (Groeneveld et al., 2024), and Mistral 7B v0.3 (Jiang et al., 2023)) across five datasets, three mathematical and two open-domain generation benchmarks. Our experiments demonstrate that explicitly incorporating uncertainty in reasoning can enhance accuracy by up to 7.4% in mathematical tasks and 5.8% in open-domain question answering. Our contributions are as follows:

- By considering the confidence of LLMs in critical points of their responses, we easily compute the uncertainty of an LLM on a generated response that can be useful in aggregating responses generated in multiple chains based on their confidences.
- We analyze various functions for each component of our method and identify the best choice to enhance reasoning accuracy.
- Empirical validation across various LLMs and benchmarks, showing significant improvements in accuracy without model fine-tuning.

2 Related Work

2.1 Reasoning in LLMs

Recent research has explored various techniques to enhance the reasoning capabilities of LLMs. CoT prompting (Brown et al., 2020; Kojima et al., 2022) improves multi-step reasoning by generating structured intermediate steps, leading to more transparent and interpretable solutions. Self-consistency (Wang et al., 2022) further enhances accuracy by sampling multiple reasoning paths and selecting the most consistent answer. In parallel, question decomposition methods (Zhou et al., 2022; Dua et al., 2022; Khot et al., 2022; Ling et al., 2023; Weng et al., 2023) improve coherence by breaking complex queries into simpler sub-questions, though it introduces additional computational overhead. Another promising direction involves search and planning-based methods (Wang et al., 2023, 2024a; Yao et al., 2023a; Besta et al., 2024; Xue et al., 2025; Yang et al., 2024a), which systematically explore multiple reasoning trajectories to improve problem-solving. Lastly, integrating external tools—such as web search engines and Python interpreters—extends the model's capabilities, enabling more precise and efficient task execution across diverse domains (Lu et al., 2023; Yao et al., 2023b; Kim et al., 2024; Chen et al., 2023). As our approach is grounded in uncertainty estimation, we begin by reviewing existing uncertainty estimation methods, followed by an introduction to uncertainty-aware reasoning techniques, which are the most pertinent to our research.

2.2 Uncertainty Estimation

Uncertainty estimation methods can be broadly classified into two categories: black-box (Zhang et al., 2023; Xiong et al., 2024; Lin et al., 2023; Manakul et al., 2023; Chen and Mueller, 2024) and white-box (Kuhn et al., 2023; Duan et al., 2024; Fadeeva et al., 2024; Huang et al., 2023) approaches. One approach to uncertainty estimation is training-based confidence estimation (Cohen et al., 2024; Lin et al., 2022; Azaria and Mitchell, 2023), which improves calibration by incorporating uncertainty estimation directly into the training process. These methods modify the training objective, introduce auxiliary loss functions, or leverage additional supervision to produce more reliable confidence estimates. Another approach is verbal-based confidence estimation (Tian et al., 2023; Kadavath et al., 2022), which prompts the model to explicitly express its confidence through natural language statements. Finally, semantic-based uncertainty estimation methods (Nikitin et al., 2024; Kuhn et al., 2023; Qiu and Miikkulainen, 2024; Wang et al., 2024b) cluster outputs or reasoning chains that are semantically equivalent, quantifying uncertainty based on the variability of responses within these clusters.

2.3 Uncertainty-aware reasoning

An emerging trend leverages uncertainty estimation as a tool to enhance various components of reasoning. One application is in improving few-shot prompting, where uncertainty estimation helps automate the selection of demonstrations (Gonen et al., 2023; Huang et al., 2024; Margatina et al., 2023), reducing the need for manually intensive prompt engineering. Another key contribution of uncertainty estimation in reasoning is its role in selecting the most reliable reasoning chain based on confidence (Murray and Chiang, 2018; Kadavath et al., 2022; Malinin and Gales, 2020). In such cases, uncertainty acts as a guiding signal, identifying the chain where the model exhibits the highest confidence. Our approach builds on this intuition by enabling a weighted voting mechanism to select the final answer. More importantly, instead of applying our uncertainty estimation function to every token, we focus only on critical tokens, specifically the intermediate answers in a CoT chain.

3 Confidence Enhanced Reasoning

Prior research has demonstrated that, analogous to human cognitive processes, enabling LLMs to generate intermediate reasoning steps can substantially enhance their accuracy in complex reasoning tasks. In this work, we aim to extend this approach further by incorporating confidence estimation into the reasoning process. We hypothesize that the final output of each intermediate step—whether a numerical value in mathematical problems or a contextually salient entity in open-domain generative reasoning—serves as a probabilistic signal, providing valuable insight into the model's confidence in that step's validity. Moreover, these localized confidence scores can be aggregated to estimate the model's overall confidence in the entire reasoning chain. By doing so, we refine the self-consistency voting mechanism: rather than selecting the most frequent answer, we sum the confidence scores of chains arrive at the same conclusion and choose the answer with the highest total confidence.

3.1 Definitions

In the following, we present the unified definitions used throughout this paper:

- **Token Probability**: The output probability of token t is derived directly from the model's output logits with a simple softmax function; denoted p_t.

- **Word Confidence**: The confidence of a word w generated by the model, calculated using a function f that incorporates all the tokens that make up the word; denoted as

$$c_w^f = f(\{p_t | t \in w\}). \qquad (1)$$

- **Path Confidence**: An output sequence generated by the LLM, denoted y and consisting of n steps where n shows the number of the constituent parts of the reasoning paths. Each step is composed of two components: a content and an answer component, denoted as o and a, respectively. In our method, the confidence score for each path y, obtained by aggregating the confidence values of only the critical points, i.e. the answer components $\{a_j\}_{j=1}^n$, on the path through a function g as

$$c_y^{g,f} = g\big(c_{a_1}^f, \dots, c_{a_n}^f\big). \tag{2}$$

Algorithm 1 CER Algorithm

```
Require: $x, P, f, g, K, T$
Ensure: $a^*$
Description: Given an input prompt $x$, the language model $P$ generates responses. The functions $f$ and $g$ represent step-wise and path-wise aggregation, respectively. The temperature parameter is denoted by $T$, and the ensemble consists of $K$ generations. The final output, is denoted as $a^*$.
 1: $\mathcal{P} \leftarrow \emptyset$
 2: $\{y^i\}_{i=1}^K \leftarrow P(y|x, T)$
 3: for $i \leftarrow 1$ to $K$ do
 4:     $y^i = \Big\{(o_j^i, a_j^i)\Big\}_{j=1}^{n^i}$
 5:     for each $a_j^i$ in $y^i$ do
 6:         $c_{a_j^i}^f \leftarrow f(a_j^i)$                        ▷ Eq. (1)
 7:     end for
 8:     $c_{y^i}^{g,f} \leftarrow g\big(c_{a_{a_1}^i}^f, \dots, c_{a_{n^i}^i}^f\big)$      ▷ Eq. (2)
 9:     $A^i = a_{n^i}^i$
10:     $\mathcal{P} \leftarrow \mathcal{P} \cup \{(c_{y^i}^{g,f}, A^i)\}$
11: end for
12: $\mathcal{A} \leftarrow \{ a \mid (c_{y^i}^{g,f}, A^i) \in \mathcal{P}\}$
13: for each $a \in \mathcal{A}$ do
14:     $C(a) \leftarrow \sum_{i=1}^K c_{y^i}^{g,f} . \mathbb{I}(\{A^i = a\})$ ▷ Eq. (5)
15: end for
16: $a^* \leftarrow \arg\max_{a \in \mathcal{A}} C(a)$
17: return $a^*$
```

3.2 Method

At first, we independently generate K response paths $\{y^1, y^2, \dots, y^K\}$ from the LLM. Next, we break down each response y^i into n^i constituent steps, extracting the answers at different steps as key elements $\{a_j^i\}_{j=1}^{n^i}$ for constructing our confidence subset. Specifically, the LLM-produced answer in the final step of the generation process y^i, i.e. $a_{n^i}^i$, representing the conclusive answer to the question in this path is denoted as A^i.

We can compute the confidence of each answer using the function f as in (1). For instance, if f is a multiplication function and a_j^i consists of r tokens $\{t_1, \dots, t_r\}$, the confidence on this special point can be written as:

$$c_{a_j^i}^{\prod} = \prod_{k=1}^{r} p(t_k). \tag{3}$$

One other choice of f is mean entropy which is computed as the average entropy of distributions on all tokens in the word. Details about different choices of f and subsequent impact on the results are thoroughly examined in Appendix A and D.

Subsequently, we aggregated the confidence scores from all steps of a path using the function g as in (2). For the path-wise aggregate function g, which aggregates the confidence scores of words, we experimented with several formulations. Our primary aggregation method is:

$$c_{y^i}^{g,f} = \frac{\sum_{j=1}^{n^i} j \cdot c_{a_j^i}^f}{\sum_{j=1}^{n^i} j}. \tag{4}$$

It assigns higher weights to the steps that are closer to the final answer. Other aggregation schemes we considered include harmonic mean and different kinds of weighted means which are introduced and assessed in Appendix B and Appendix D.

Once path confidence is determined, we further aggregate the confidence scores of all paths that yield the same A^i. The answer with the highest aggregate confidence is then selected.

$$\begin{aligned} \mathcal{A} &= \{ a \mid a \in \{A^i\}_{i=1}^K \} \\ C(a) &= \sum_{i=1}^{K} c_{y^i}^{g,f} \times \mathbb{I}(A^i = a) \quad \forall a \in \mathcal{A}, \\ a^* &= \arg\max_{a \in \mathcal{A}} C(a). \end{aligned} \tag{5}$$

where $\mathcal{A}$ is the set of unique final answers among $\{A^i\}_{i=1}^K$. $C(a)$ is the aggregated confidence score for each unique a. Finally, a^* is the best candidate, chosen by maximizing the confidence score over all $a \in \mathcal{A}$.

The algorithm 1 summarizes the complete procedure of our method.

4 Experiments

4.1 Experimental Setup

In this section, we present the experimental setup used to assess our method and compare it with the

Models & Datasets	**Self-Consistency**	**P(True)**	**PE**	**NL**	**NE**	**LL**	**Greedy**	**CER**
LLaMA-3.1-8B								
GSM8K	89.6	87.6	85.2	86.2	86.2	83.8	82.8	**90.0** (+0.4%)
MATH	55.4	56.8	52.0	52.8	53.6	50.4	53.4	**58.2** (+1.8%)
MathQA	63.2	65.2	64.4	65.2	61.6	65.4	60.0	**68.2** (+2.8%)
Average	69.4	69.8	67.2	68.0	67.1	66.53	65.4	**72.1** (+2.3%)
Mistral-7B								
GSM8K	62.2	46.6	55.8	59.0	60.0	55.6	44.8	**65.2** (+3.0%)
MATH	**20.4**	13.6	19.0	20.2	20.0	19.6	17.0	18.0 (-2.4%)
MathQA	20.8	12.4	**22.6**	20.0	19.4	**22.6**	20.2	**22.6** (+0%)
Average	34.4	24.2	32.4	33.0	33.1	32.6	27.3	**35.2** (+0.8%)
OLMo-2-7B								
GSM8K	85.0	82.0	84.4	83.8	78.0	84.8	84.2	**88.8** (+3.8%)
MATH	42.5	40.0	41.0	40.0	39.2	42.6	37.8	**48.0** (+5.4%)
MathQA	52.0	51.8	44.8	50.0	48.8	47.4	45.2	**59.4** (+7.4%)
Average	59.8	57.9	56.7	57.9	53.3	58.2	55.73	**65.1** (+5.3%)
LLama-3.3-3B								
GSM8K	78.4	73.2	73.0	77.0	78.6	75.2	75.2	**82.6** (+4%)
MATH	51.2	44.2	44.0	42.6	40.0	40.2	46.4	**56.0** (+4.8%)
MathQA	59.6	52.2	55.6	54.2	58.4	57.4	55.4	**62.8** (+3.2%)
Average	63.0	56.5	57.5	57.9	59.0	57.6	59.0	**67.1** (+4.1%)

Table 1: Accuracy comparison across three mathematical datasets—MATH, MATHQA, and GSM8K—on 500 sampled instances evaluated using various baseline methods and the proposed CER approach. The colored values indicate the improvement or decline compared to the best performance of the baselines for each dataset.

other methods.

Models: We evaluate our approach on a diverse set of LLMs to capture a wide range of architectures and capabilities. Our primary model is **Meta Llama 3.1 8B Instruct** (Dubey et al., 2024), a state-of-the-art open source LLM known for its robust performance. To further support our findings, we also conducted experiments on **Meta Llama 3.2 3B** (Dubey et al., 2024), representing a powerful yet compact model. Additional experiments were performed using **Mistral 7B Instruct** (Jiang et al., 2023), a model frequently referenced in recent studies, and **Olmo 2 7B** (Groeneveld et al., 2024), which exemplifies the latest mixture of expert architectures.

Datasets and Tasks: We evaluate our method across two task categories: 1) mathematical reasoning and 2)open-domain question answering. For the mathematical tasks, we utilize the following datasets:

- **GSM8K** (Cobbe et al., 2021): A widely used benchmark that contains mathematical problems with numerical answers.
- **MATH** (Hendrycks et al., 2021): A dataset that presents more complex mathematical problems than GSM8K. It consists of two parts: numerical and non-numerical answers. We preprocessed the dataset and filtered out all mathematical questions that yield non-numerical answers.
- **Math QA** (Amini et al., 2019): A collection of difficult math problems that do not overlap with the MATH dataset.

For open-domain question answering, we utilize the following datasets:

- **TriviaQA** (Joshi et al., 2017): A large-scale dataset containing knowledge-intensive questions sourced from Wikipedia.
- **HotPotQA** (Yang et al., 2018): A dataset designed for multi-hop reasoning (Yang et al., 2024b), requiring models to synthesize information from multiple documents. We preprocessed the dataset by removing all comparison questions and filtering out open-domain generation questions that do not have a proper noun as their answer.

Models & Datasets	Self-Consistency	P(True)	PE	NL	NE	LL	Greedy	CER
LLaMA-3.1-8B								
Trivia QA	62.2	64.8	58.0	58.0	60.2	59.4	61.8	**66.0** (+1.2%)
HotPot QA	10.2	**14.4**	11.0	13.4	12.6	13.2	14.2	**14.4** (+0.0%)
Average	36.2	39.6	34.5	35.7	36.4	36.3	38.0	**40.2** (+0.6%)
Mistral-7B								
Trivia QA	37.0	43.2	48.6	46.0	44.2	47.0	44.8	**54.4** (+5.8%)
HotPot QA	7.2	6.4	10.2	7.6	6.8	8.8	8.4	**10.4** (+0.2%)
Average	22.1	24.8	29.4	26.8	25.5	27.9	26.6	**32.4** (+3.0%)
OLMo-2-7B								
Trivia QA	47.0	49.0	48.0	45.2	43	46.4	48.4	**50.8** (+1.8%)
HotPot QA	8.6	8.6	8.2	8.8	7.8	8.6	8.4	**10.6** (+1.8%)
Average	27.8	28.8	28.1	27.0	25.4	27.5	28.4	**30.7** (+1.9%)
LLama-3.3-3B								
Trivia QA	48.8	50.8	45.0	43.4	42.4	41.4	49.4	**53.0** (+2.2%)
HotPot QA	9.0	8.4	6.4	6.8	7.8	7.4	9.0	**9.2** (+0.2%)
Average	28.9	29.6	25.7	25.1	25.1	24.4	29.0	**31.1** (+1.5%)

Table 2: Accuracy comparison on two open-domain QA datasets—Trivia QA and HotPot QA—using 500 sampled instances. The table presents results across multiple baseline methods alongside the proposed CER method. Colored values represent the performance change compared to the best baseline performance.

Both of these datasets require comprehensive reasoning and are knowledge-intensive.

Evaluation Metrics: Given our emphasis on reasoning and verifiable problem solving, we adopt accuracy as the main evaluation metric.

Baselines: We compare our approach against several baselines that include greedy sampling and self-consistency as baselines and also improved versions of self-consistency by incorporating confidence or uncertainty in their voting phase:

- **Greedy Sampling:** Uses straightforward greedy decoding to generate a single response, serving as a baseline for the model's raw performance.
- **Self-Consistency** (Wang et al., 2022): Aggregates multiple response paths to enhance reasoning accuracy.
- **Token "True" Probability** (Kadavath et al., 2022): Determines the final answer based on the probability assigned to the token "true".
- **Log Likelihood (LL)** (Murray and Chiang, 2018): Multiply the probabilities of all tokens in a response path.
- **Normalized Likelihood (NL)** (Murray and Chiang, 2018): A length-normalized variant of log likelihood, computed by dividing the log likelihood by the sequence length.
- **Predictive Entropy (PE)** (Kadavath et al., 2022): Computes the mean entropy over all tokens in a response path to assess confidence.
- **Normalized Entropy (NE)** (Malinin and Gales, 2020): A length-normalized variant of predictive entropy, obtained by dividing the entropy by the sequence length.

More details on the formulation and the aggregation approach of confidence-based methods are provided in the Appendix G.

Implementation Details: All methods, except for the simple greedy baseline, utilize temperature sampling with $T = 1$ to generate responses. The number of generated paths K is set to 10, a choice supported by previous research. (Zhang et al., 2023; Duan et al., 2024; Qiu and Miikkulainen, 2024; Fadeeva et al., 2024) We aggregated all response paths based on an exact match of the final answer to the question. Our experiments were conducted on a single A100 80G GPU. We sample 500 data points from each dataset and evaluate our results on these subsets. Additional details, including input prompts and sample instances from the datasets, can be found in the Appendix C.

4.2 Main Results

Mathematical Reasoning Table 1 reports the performance of our models on three mathematical datasets under the CER framework, alongside all baseline methods. Notably, our CER approach consistently surpasses every baseline, with its advantage being particularly marked when applied to smaller, less powerful LLMs. In addition, our method yields more significant relative improvements on more challenging datasets. For instance, Llama 3.1 8B records an average relative gain of 2.3% across the datasets; Mistral 7B, Olmo 2 7B, and Llama 3.2 3B achieve gains of 0.8%, 5.3%, and 4.1%, respectively. An intriguing observation arises from the results on Llama 3.1 7B—the most potent model in our experiments. Although this model already exhibits strong baseline performance, CER not only boosts its overall results but also delivers particularly significant improvements on the more demanding MATH and Allen AI's Math QA datasets. By contrast, the performance trend for the Mistral model differs: while it shows consistent improvements across all datasets, the performance gap does not widen as markedly on the more challenging problems. This suggests that while CER can unlock additional reasoning capabilities in models with sufficient capacity, its benefits are limited when the underlying model lacks the capacity to solve the problem entirely.

Knowledge Intensive Reasoning Our CER method outperforms all baselines by a substantial margin for open-domain generation tasks requiring intensive knowledge reasoning. Specifically, it delivers average gains of 0.6% for Llama 3.1 8B, 3.0% for Mistral 7B, 1.9% for Olmo 2 7B, and 1.5% for Llama 3.1 3B. A notable finding is the relatively poor performance of Llama 3.2 3B compared to the other models. Although Llama 3.2 3B outperforms Mistral 7B on mathematical reasoning tasks by a considerable margin, it falls short on knowledge-intensive tasks. We attribute this discrepancy to the nature of questions in Trivia QA and HotPot QA, which demand that specific knowledge be stored within the model's parameters. In contrast, mathematical reasoning relies primarily on operational and logical skills. Consequently, even though Llama 3.2 3B is distilled

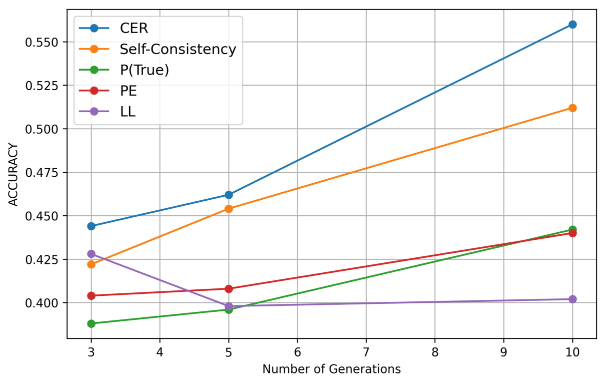


Figure 2: Performance comparison of CER and baseline models across different generations $K = \{3, 5, 10\}$ on the LLAMA 3.3-3B model using the MATH dataset.

from larger, more capable models, its smaller size means it possesses fewer parameters to encapsulate the extensive knowledge required, leading to its diminished performance on knowledge-intensive tasks.

Results Across Different Models As previously noted, our selection of models aims to demonstrate the performance and versatility of our approach across both smaller models and mixtures of experts—a popular choice in recent research. As illustrated in Tables 1 and 2, our framework not only achieves strong results with commonly used models such as Llama 3.1 8B and Mistral 7B, but also shows impressive performance on the compact Llama 3.1 3B and the state-of-the-art open-source MoE model, Olmo 2 7B. In every case, CER outperforms all baseline methods across all datasets.

4.3 Ablation Studies

We conducted several ablation studies to further elucidate the contributions of individual components and assess the robustness of our approach.

Varying the Number of Paths (K) Our first experiment explores the impact of the hyperparameter K, which denotes the number of generated paths. As shown in Figure 2, both CER and all baseline methods benefit from increasing K. However, CER consistently outperforms the baselines for every value of K.

Entropy vs. Probabilities While entropy is commonly used in the literature as a measure of model uncertainty and confidence, we conducted an ablation study comparing the mean entropy over all tokens to the word confidence measure defined in Equation 3. Appendix D provides the complete

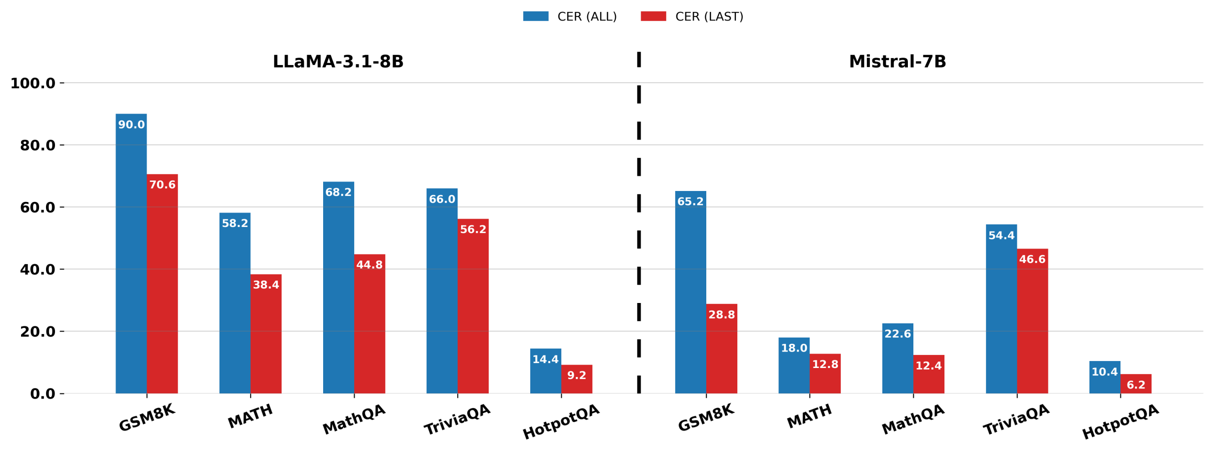


Figure 3: Ablation study results comparing the performance of the CER method using the last answer confidence (CER-LAST, red) versus the original CER method utilizing all intermediate answers (CER-ALL, blue) across mathematical reasoning datasets (GSM8K, MATH, MathQA) and open-domain question-answering datasets (TriviaQA, HotpotQA). The left side presents results for LLaMA-3.1-8B, while the right side shows results for Mistral-7B. Across all datasets, CER-ALL consistently outperforms CER-LAST, emphasizing the advantage of incorporating intermediate answers for improved accuracy.

results and the precise formulation of the entropy function, as f is in Appendix A.

Different Path-Level Aggregators This study investigated the effect of various path-level aggregator functions, denoted by g. Beyond our primary choice of weighted mean aggregation, we experimented with several similar alternatives. The results across these different aggregators were strikingly similar, indicating that the weighted mean is sufficiently effective without requiring further tuning. We also assessed an aggregation function based on the multiplication of word-level confidences along each path, as well as the minimum function—motivated by the adage “a chain is only as strong as its weakest link.” All alternatives yielded comparable results, as detailed in Appendix D

Last Answer Confidence Finally, we examined the effect of relying solely on the confidence of the last answer to guide the overall reasoning process, thereby excluding intermediate signals. As illustrated in Figure 3, this ablation reveals a significant performance gap compared to the original CER method. Although confidence in the final answer is an important indicator, these results confirm that incorporating all intermediate responses leads to superior performance.

5 Conclusion

In this paper, we introduced a lightweight framework that enhances performance on various reasoning tasks by relying solely on the model’s output logits without the need for fine-tuning or task-specific prompts. Our approach bridges the gap between reasoning and uncertainty estimation in LLMs. Through extensive experiments, we validated our proposed functions and demonstrated the CER algorithm’s effectiveness as a general performance enhancement framework. Our findings show that the framework is robust across different model sizes and architectures. In this study, we focus only on numerical outputs in mathematical reasoning. However, with minor modifications, our approach can also handle non-numerical outputs, such as mathematical proofs. Future research could further extend our framework to mathematical reasoning tasks with non-numerical final answers.

Limitations

Our work has several notable limitations. First, the framework has been applied only to a narrow range of tasks, specifically those involving mathematical reasoning and knowledge-intensive questions. Furthermore, our approach relies on access to the model output logits; therefore, our method is not applicable in scenarios where these logits are unavailable.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Aida Amini, Saadia Gabriel, Shanchuan Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. MathQA: Towards interpretable math word problem solving with operation-based formalisms. 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 2357–2367, Minneapolis, Minnesota. Association for Computational Linguistics.

Amos Azaria and Tom Mitchell. 2023. The internal state of an LLM knows when it‘s lying. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 967–976, Singapore. Association for Computational Linguistics.

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Michał Podstawski, 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.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. 2015. Weight uncertainty in neural networks. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, page 1613–1622. JMLR.org.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Jiuhai Chen and Jonas Mueller. 2024. Quantifying uncertainty in answers from any language model and enhancing their trustworthiness. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5186–5200, Bangkok, Thailand. Association for Computational Linguistics.

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*.

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

Roi Cohen, Konstantin Dobler, Eden Biran, and Gerard de Melo. 2024. I don't know: Explicit modeling of uncertainty with an [idk] token. *arXiv preprint arXiv:2412.06676*.

Leda Cosmides and John Tooby. 1996. Are humans good intuitive statisticians after all? rethinking some conclusions from the literature on judgment under uncertainty. *cognition*, 58(1):1–73.

Dheeru Dua, Shivanshu Gupta, Sameer Singh, and Matt Gardner. 2022. Successive prompting for decomposing complex questions. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 1251–1265, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Jinhao Duan, Hao Cheng, Shiqi Wang, Alex Zavalny, Chenan Wang, Renjing Xu, Bhavya Kailkhura, and Kaidi Xu. 2024. Shifting attention to relevance: Towards the predictive uncertainty quantification of free-form large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5050–5063, Bangkok, Thailand. Association for Computational Linguistics.

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*.

Ekaterina Fadeeva, Aleksandr Rubashevskii, Artem Shelmanov, Sergey Petrakov, Haonan Li, Hamdy Mubarak, Evgenii Tsymbalov, Gleb Kuzmin, Alexander Panchenko, Timothy Baldwin, Preslav Nakov, and Maxim Panov. 2024. Fact-checking the output of large language models via token-level uncertainty quantification. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 9367–9385, Bangkok, Thailand. Association for Computational Linguistics.

Hila Gonen, Srini Iyer, Terra Blevins, Noah Smith, and Luke Zettlemoyer. 2023. Demystifying prompts in language models via perplexity estimation. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 10136–10148, Singapore. Association for Computational Linguistics.

Dirk Groeneveld, Iz Beltagy, Evan Walsh, Akshita Bhagia, Rodney Kinney, Oyvind Tafjord, Ananya Jha, Hamish Ivison, Ian Magnusson, Yizhong Wang, Shane Arora, David Atkinson, Russell Authur, Khyathi Chandu, Arman Cohan, Jennifer Dumas, Yanai Elazar, Yuling Gu, Jack Hessel, Tushar Khot, William Merrill, Jacob Morrison, Niklas Muennighoff, Aakanksha Naik, Crystal Nam, Matthew Peters, Valentina Pyatkin, Abhilasha Ravichander, Dustin Schwenk, Saurabh Shah, William Smith, Emma Strubell, Nishant Subramani, Mitchell Wortsman, Pradeep Dasigi, Nathan Lambert, Kyle Richardson, Luke Zettlemoyer, Jesse Dodge, Kyle Lo, Luca

Soldaini, Noah Smith, and Hannaneh Hajishirzi. 2024. OLMo: Accelerating the science of language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15789–15809, Bangkok, Thailand. Association for Computational Linguistics.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. 2017. On calibration of modern neural networks. In *International conference on machine learning*, 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*.

Danny Halawi, Jean-Stanislas Denain, and Jacob Steinhardt. 2023. Overthinking the truth: Understanding how language models process false demonstrations. *Preprint*, arXiv:2307.09476.

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*.

Hsiu-Yuan Huang, Zichen Wu, Yutong Yang, Junzhao Zhang, and Yunfang Wu. 2024. Unlocking the power of llm uncertainty for active in-context example selection. *arXiv preprint arXiv:2408.09172*.

Yuheng Huang, Jiayang Song, Zhijie Wang, Shengming Zhao, Huaming Chen, Felix Juefei-Xu, and Lei Ma. 2023. Look before you leap: An exploratory study of uncertainty measurement for large language models. *arXiv preprint arXiv:2307.10236*.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

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*.

Tushar Khot, Harsh Trivedi, Matthew Finlayson, Yao Fu, Kyle Richardson, Peter Clark, and Ashish Sabharwal. 2022. Decomposed prompting: A modular approach for solving complex tasks. *arXiv preprint arXiv:2210.02406*.

Joongwon Kim, Bhargavi Paranjape, Tushar Khot, and Hannaneh Hajishirzi. 2024. Husky: A unified, open-source language agent for multi-step reasoning. *Preprint*, arXiv:2406.

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

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

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

Zhen Lin, Shubhendu Trivedi, and Jimeng Sun. 2023. Generating with confidence: Uncertainty quantification for black-box large language models. *arXiv preprint arXiv:2305.19187*.

Zhan Ling, Yunhao Fang, Xuanlin Li, Zhiao Huang, Mingu Lee, Roland Memisevic, and Hao Su. 2023. Deductive verification of chain-of-thought reasoning. *arXiv preprint arXiv:2306.03872*.

Pan Lu, Baolin Peng, Hao Cheng, Michel Galley, Kai-Wei Chang, Ying Nian Wu, Song-Chun Zhu, and Jianfeng Gao. 2023. Chameleon: Plug-and-play compositional reasoning with large language models. *arXiv preprint arXiv:2304.09842*.

Andrey Malinin and Mark Gales. 2020. Uncertainty estimation in autoregressive structured prediction. *arXiv preprint arXiv:2002.07650*.

Potsawee Manakul, Adian Liusie, and Mark Gales. 2023. SelfCheckGPT: Zero-resource black-box hallucination detection for generative large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9004–9017, Singapore. Association for Computational Linguistics.

Ana Marasovic, Iz Beltagy, Doug Downey, and Matthew Peters. 2022. Few-shot self-rationalization with natural language prompts. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 410–424, Seattle, United States. Association for Computational Linguistics.

Katerina Margatina, Timo Schick, Nikolaos Aletras, and Jane Dwivedi-Yu. 2023. Active learning principles for in-context learning with large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 5011–5034, Singapore. Association for Computational Linguistics.

Kenton Murray and David Chiang. 2018. Correcting length bias in neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 212–223, Brussels, Belgium. Association for Computational Linguistics.

Alexander Nikitin, Jannik Kossen, Yarin Gal, and Pekka Marttinen. 2024. Kernel language entropy: Fine-grained uncertainty quantification for llms from semantic similarities. *arXiv preprint arXiv:2405.20003*.

Xin Qiu and Risto Miikkulainen. 2024. Semantic density: Uncertainty quantification for large language models through confidence measurement in semantic space. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Katherine Tian, Eric Mitchell, Allan Zhou, Archit Sharma, Rafael Rafailov, Huaxiu Yao, Chelsea Finn, and Christopher Manning. 2023. Just ask for calibration: Strategies for eliciting calibrated confidence scores from language models fine-tuned with human feedback. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5433–5442, Singapore. Association for Computational Linguistics.

Christian Tomani and Florian Buettner. 2021. Towards trustworthy predictions from deep neural networks with fast adversarial calibration. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 9886–9896.

Evan Wang, Federico Cassano, Catherine Wu, Yunfeng Bai, Will Song, Vaskar Nath, Ziwen Han, Sean Hendryx, Summer Yue, and Hugh Zhang. 2024a. Planning in natural language improves llm search for code generation. *arXiv preprint arXiv:2409.03733*.

Ruocheng Wang, Eric Zelikman, Gabriel Poesia, Yewen Pu, Nick Haber, and Noah D Goodman. 2023. Hypothesis search: Inductive reasoning with language models. *arXiv preprint arXiv:2309.05660*.

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*.

Xuezhi Wang and Denny Zhou. 2024. Chain-of-thought reasoning without prompting. *arXiv preprint arXiv:2402.10200*.

Yu-Hsiang Wang, Andrew Bai, Che-Ping Tsai, and Cho-Jui Hsieh. 2024b. Clue: Concept-level uncertainty estimation for large language models. *arXiv preprint arXiv:2409.03021*.

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. 2023. Large language models are better reasoners with self-verification. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 2550–2575, Singapore. Association for Computational Linguistics.

Miao Xiong, Zhiyuan Hu, Xinyang Lu, YIFEI LI, Jie Fu, Junxian He, and Bryan Hooi. 2024. Can LLMs express their uncertainty? an empirical evaluation of confidence elicitation in LLMs. In *The Twelfth International Conference on Learning Representations*.

Shangzi Xue, Zhenya Huang, Jiayu Liu, Xin Lin, Yuting Ning, Binbin Jin, Xin Li, and Qi Liu. 2025. Decompose, analyze and rethink: Solving intricate problems with human-like reasoning cycle. *Advances in Neural Information Processing Systems*, 37:357–385.

Ling Yang, Zhaochen Yu, Tianjun Zhang, Shiyi Cao, Minkai Xu, Wentao Zhang, Joseph E Gonzalez, and Bin Cui. 2024a. Buffer of thoughts: Thought-augmented reasoning with large language models. *Advances in Neural Information Processing Systems*.

Sohee Yang, Elena Gribovskaya, Nora Kassner, Mor Geva, and Sebastian Riedel. 2024b. Do large language models latently perform multi-hop reasoning? In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10210–10229, Bangkok, Thailand. Association for Computational Linguistics.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. 2023a. Tree of Thoughts: Deliberate problem solving with large language models. *Preprint*, arXiv:2305.10601.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023b. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. 2022. STar: Bootstrapping reasoning with reasoning. In *Advances in Neural Information Processing Systems*.

Jiaxin Zhang, Zhuohang Li, Kamalika Das, Bradley Malin, and Sricharan Kumar. 2023. SAC^3: Reliable hallucination detection in black-box language models via semantic-aware cross-check consistency. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 15445–15458, Singapore. Association for Computational Linguistics.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, et al. 2022. Least-to-most prompting enables complex reasoning in large language models. *arXiv preprint arXiv:2205.10625*.

A Different Choices for step-wise aggregation function (f)

In this work, we define the stepwise aggregate function (f) as a function that quantifies the confidence of a word by leveraging the probabilities of its constituent tokens. We consider two common formulations for f:

1. **Mean Entropy:** Compute the average entropy of all tokens in a word. This metric represents the confidence of the model when generating the word, where a lower entropy indicates a higher confidence.

2. **Multiplicative Probability:** Determine the overall probability of a word by multiplying the probabilities of its constituent tokens, where a higher value indicates greater confidence.

Mean Entropy: Let a word w consist of tokens $\{t_1, t_2, \ldots, t_n\}$ with the corresponding probabilities of the mass functions $P(T = t_1), P(T = t_2), \ldots, P(T = t_n)$ and the corresponding probabilities of the tokens $p(t_1), p(t_2), \ldots, p(t_n)$. We define the mean entropy formulation as follows:

$$f_{\text{entropy}}(w) = -\frac{1}{n}\sum_{i=1}^{n} H(P(T = t_i)) \quad (6)$$

$$f_{\text{entropy}}(w) = -\frac{1}{n}\sum_{i=1}^{n}\sum_{j} P(t_i = j)\log P(t_i = j) \quad (7)$$

Multiplicative Probability: Alternatively, the multiplicative probability formulation is given by:

$$f_{\text{mult}}(w) = \prod_{i=1}^{n} p(t_i). \quad (8)$$

We also performed an ablation study using the mean probability of tokens as an alternative.

B Different Choices for path-wise aggregate function (g)

For the path-wise aggregate function (g), which aggregates the confidence scores of words, we experimented with several formulations. Our primary aggregation method is the weighted mean, where C_w represents the confidence of each word. Other aggregation schemes we considered include the following:

For all cases below, let $\{c_1, \ldots, c_n\}$ denote the confidence scores associated with words 1 through n.

- **Harmonic Mean:** Aggregates confidences using the harmonic mean.

$$\frac{n}{\frac{1}{c_1} + \cdots + \frac{1}{c_n}} \quad (9)$$

- **Weighted Mean:** This approach applies linearly increasing weights to the confidences, based on the intuition that the final steps contribute more to the overall confidence of the path and should therefore receive greater weight.

$$\frac{1 \cdot c_1 + \cdots + n \cdot c_n}{1 + \cdots + n} \quad (10)$$

- **Half Split Mean:** A weighted split that assigns half of the total weight to the final answer, with the remaining half distributed uniquely among the other words.

$$\frac{1}{2}c_n + \frac{1}{2(n-1)}\sum_{i=1}^{n-1} c_i, \quad n > 1. \quad (11)$$

- **Exponential Mean:** Uses exponents of 2 as the weights to emphasize later steps.

$$\frac{2^0 \cdot c_1 + \cdots + 2^{n-1} \cdot c_n}{2^n - 1} \quad (12)$$

- **Average Log:** Computes the average of the logarithm-transformed confidences.

$$\frac{1}{n}\sum_{i=1}^{n} \log(1 + c_i) \quad (13)$$

- **Minimum:** Uses the minimum confidence among all steps.

$$\min_{i \in \{1,\ldots,n\}} c_i \quad (14)$$

Each function represents a distinct hypothesis regarding the relative importance of individual words in the response path. The experimental results comparing these methods are presented in the corresponding section of the paper.

Objective
Carefully work through the problem step by step. For each step, perform any required reasoning and express the answer at the end of the step. Your response should be in the format Answer: [answer]. After completing the steps, provide the final answer based on the reasoning developed throughout the process.
Important Rules
1. Perform detailed analyses before concluding the answer.
2. Express intermediate answers explicitly at the end of each step in the format Answer: [answer].
3. Ensure that your response ends with: **The final answer is [answer]**, where [answer] is the response to the problem.

Q: <question>

Figure 4: Prompt for Math Reasoning

Objective
Carefully work through the problem step by step, focusing only on the essential steps and limiting your response to five sentences. Your response should end with: The final answer is [answer], where [answer] is the response to the problem.
Q: <question>

Figure 5: Prompt for Multi-hop Reasoning

C More Implementation Details

The prompt is tailored for mathematical reasoning, guiding the LLM through a structured step-by-step process while ensuring it generates an answer at each stage. This is illustrated in Figure 4. Similarly, the prompt for open-domain generation is designed to systematically lead the LLM through a logical reasoning process, as shown in Figure 5.

D More Results

This section shows the results of the ablation studies for both f and g functions. Table 3 shows the results for different choices of f, and Table 4 shows the results for the g alternatives.

E Further Exploration of the Dataset

Here, we show a sample from each dataset. Table 5 shows the mathematical datasets samples and Table 6 shows the open domain generation QA datasets samples.

F Full question and responses related to the main figure

Table 7 shows a question as an input sample to the LLm with our prompt and the corresponding generated paths and their intermediate steps.

G Examination of Confidence Baselines

We examine baseline methods that integrate confidence measures, specifically Log-Likelihood (LL), Normalized-Length Likelihood (NL), Predictive Entropy (PE), and Normalized-Length Entropy (NE). Let P_θ represent the LLM, and denote N as the number of generated tokens, expressed as $\{y_1, \ldots, y_N\}$.

- **Log-Likelihood (LL):** Computes the likelihood of a response path by multiplying the probabilities of all tokens in the sequence. After evaluating the confidence of each response, the answer with the highest confidence—or equivalently, the one with the lowest negative log-likelihood—is selected. Its corresponding equation is:

$$\text{LL} = -\sum_{t=1}^{N} \log P_\theta(y_t \mid y_{1:t-1}, x) \quad (15)$$

- **Normalized Likelihood (NL):** Computes a normalized version of the log-likelihood for a response path by multiplying the probabilities of all tokens in the sequence and normalizing the value by the length of the generated response (N). The answer with the highest confidence—or equivalently, the one with the lowest negative normalized-length likelihood—is selected. Its corresponding equation is:

$$\text{NL} = \frac{-1}{N}\sum_{t=1}^{N} \log P_\theta(y_t \mid y_{1:t-1}, x) \quad (16)$$

- **Predictive Entropy (PE):** Computes the mean entropy over all tokens in a response path to assess confidence. The answer with the highest confidence—or equivalently, the one with the lowest predictive entropy—is selected. Its corresponding equation is:

$$\text{PE} = -\sum_{t=1}^{N} P_\theta(y_t \mid y_{1:t-1}, x) \cdot \log P_\theta(y_t \mid y_{1:t-1}, x) \quad (17)$$

- **Normalized Entropy (NE):** A normalized version of predictive entropy that accounts for sequence length. The answer with the highest confidence—or equivalently, the one with the lowest normalized entropy—is selected. Its corresponding equation is:

$$\text{NE} = \frac{-1}{N} \sum_{t=1}^{N} P_\theta(y_t \mid y_{1:t-1}, x) \cdot \log P_\theta(y_t \mid y_{1:t-1}, x) \quad (18)$$

LLM	Math datasets			Open-domain datasets	
	GSM8K	MATH	MathQA	TriviaQA	HotPotQA
Multiplication					
LLama-3.1-8B	90.0	58.2	68.2	66.0	14.4
Mistral-2-7B	65.2	18.0	22.6	54.4	10.4
OLMo-2-7B	88.8	48.0	59.4	50.8	10.6
LLama-3.3-3B	82.6	56.0	62.8	53.0	9.2
Entropy					
LLama-3.1-8B	89.0	57.2	66.6	62.0	12.8
Mistral-2-7B	65.2	21.4	22.2	52.6	9.2
OLMo-2-7B	84.0	30.0	50.0	54.0	8.0
LLama-3.3-3B	85.8	50.0	60.8	50.4	8.4

Table 3: Accuracy comparison of different large language models (LLMs) on mathematical reasoning and open-domain question-answering datasets. The models are evaluated on GSM8K, MATH, and MathQA for mathematical reasoning, and TriviaQA and HotPotQA for open-domain tasks. Results are reported for two variations of our step-wise aggregate functions (f): Multiplication and Entropy.

LLM	Math datasets			Open-domain datasets	
	GSM8K	MATH	MathQA	TriviaQA	HotPotQA
Self-Consistency					
LLama-3.1-8B	89.6	55.4	63.2	62.2	10.2
Mistral-2-7B	62.2	20.4	20.8	37.0	7.2
OLMo-2-7B	85.0	42.5	52.0	47.0	8.6
LLama-3.3-3B	78.4	51.2	59.6	48.8	9.0
Avg($\log c$)					
LLama-3.1-8B	89.6	56.8	68.6	65.8	14.6
Mistral-2-7B	66.8	19.2	24.8	54.2	9.8
OLMo-2-7B	–	48.0	58.0	53.0	8.8
LLama-3.3-3B	84.0	55.6	61.6	54.8	9.6
min(c)					
LLama-3.1-8B	90.6	57.6	65.4	65.2	12.6
Mistral-2-7B	63.4	17.2	24.8	54.6	9.4
OLMo-2-7B	89.4	46.8	58.8	49.4	8.4
LLama-3.3-3B	80.6	53.6	58.6	52.0	6.4
Weighted half					
LLama-3.1-8B	89.0	59.0	68.2	67.4	13.8
Mistral-2-7B	66.8	20.0	21.6	53.2	10.0
OLMo-2-7B	89.2	46.0	59.2	50.2	9.6
LLama-3.3-3B	82.2	56.2	63.2	51.0	9.6
Weighted exp-2					
LLama-3.1-8B	91.0	59.4	68.2	66.2	15.0
Mistral-2-7B	63.4	18.2	22.6	54.0	9.2
OLMo-2-7B	88.8	48.2	58.4	49.6	9.0
LLama-3.3-3B	82.0	55.6	62.8	55.2	8.8
Harmonic mean					
LLama-3.1-8B	90.2	56.4	65.6	66.0	12.8
Mistral-2-7B	66.4	22.4	23.2	52.0	9.4
OLMo-2-7B	–	–	54.0	49.6	9.0
LLama-3.3-3B	84.2	56.0	62.6	54.0	8.8

Table 4: Accuracy comparison of different variants of our main method g function against the self-consistency baseline on all models, evaluated on mathematical reasoning and open-domain generation datasets.

Question	Sampled Path
[GSM8K] The basketball team went to the steakhouse to eat dinner. The first player ate a 6-ounce steak. The second player ate beef tips, containing 8 beef tips, each an ounce in size. The third player ate a one-pound steak. And the fourth and fifth players ordered vegetarian meals. In total, how many ounces of meat were consumed by the team?	**Step 1**: Calculate the amount of meat eaten by the first player. The first player ate a 6-ounce steak. 0.8 pounds is equal to 6 ounces (using the conversion factor 1 pound is equal to 16 ounces.) Answer: 6. **Step 2**: Calculate the amount of meat eaten by the second player. The second player ate beef tips, containing 8 beef tips, each an ounce in size. Multiply the number of tips (8) by the size of each tip (1 ounce to find the total amount eaten by the second player). 8 * 1 ounce = 8 ounces. Answer: 8. **Step 3**: Calculate the amount of meat eaten by the third player. The third player ate a 1-pound steak. First, we need to convert the 1-pound steak to ounces using the conversion factor 1 pound is equal to 16 ounces. 1 pound multiplied by 16 ounces (pound-ounces) is 16 ounces. Answer: 16. **Step 4**: Calculate the total amount of meat eaten by the team by summing the amounts for all the players. Since the fourth and fifth players ate vegetarian meals, they did not eat any meat, so we will only sum the amounts eaten by the first three players. 6 + 8 + 16 = 30 ounces. Answer: 30. The final answer is 30.
[MATH] Find the value of x such that $\sqrt{x-4} = 4$.	**Step 1**: To start solving this equation, I will square both sides of the equation to eliminate the square root. By doing this, I will be able to get rid of the radical sign and simplify the equation. Answer: $(\sqrt{x-4})^2 = 4^2$ **Step 2**: The left-hand side of the equation becomes $x-4$ after squaring the square root because $(\sqrt{y})^2 = y$. The right-hand side of the equation becomes 16 since $4^2 = 4 \times 4$. Answer: $x - 4 = 16$. **Step 3**: Now, I will isolate x by adding 4 to both sides of the equation. This gets rid of the -4 on the left-hand side of the equation. Answer: $x - 4 + 4 = 16 + 4$. **Step 4**: The left-hand side of the equation simplifies to x since any number minus itself is equal to zero, and adding zero to a number doesn't change its value. The right-hand side of the equation becomes 20 since $16 + 4 = 20$. Answer: $x = 20$. The final answer is 20.
[MathQA] Find a sum for first 8 prime numbers?	**Step 1**: To find the sum of the first 8 prime numbers, we need to identify the first 8 prime numbers. **Answer**: 2, 3, 5, 7, 11, 13, 17, and 19. **Step 2**: Now, let's add these numbers together to get the sum. Answer: 2 + 3 + 5 + 7 + 11 + 13 + 17 + 19. **Step 3**: Performing the addition: Sum = 2 + 3 = 5, Sum = 5 + 5 = 10, Sum = 10 + 7 = 17, Sum = 17 + 11 = 28 - Sum = 28 + 13 = 41, Sum = 41 + 17 = 58, Sum = 58 + 19 = 77 Answer: 77 The final answer is 77.

Table 5: We domenstrate the performance of our method, CER, along with a sampled response path for solving each question in the math dataset. Intermediate answers at each step are highlighted in blue, while the final answer is emphasized in green.

Question	**Sampled Path**
[TriviaQA] In which town or city would you find the 'Caledonian Brewery'?	[**Step 1**] To find the location of the 'Caledonian Brewery', let's start by understanding the name itself. [**Step 2**] 'Caledonian' refers to Caledonia, an ancient Roman name for Scotland. **[Step 3]** The brewery is likely to be in Scotland given its name. [**Step 4**] Caledonian Brewery is specifically located in Edinburgh, Scotland. [**Step 5**] Edinburgh is the capital city of Scotland. The final answer is Edinburgh.
[HotPotQA] The Mutlu Onaral album "LivinÍt" was produced by the bassist for what duo?	[**Step 1**] To find the answer, we need to look for information about the Mutlu Onaral album "LivinÍt" and find a connection to a duo. [**Step 2**] After researching "LivinÍt," I found itś by Mutlu Onaral, a Turkish-born singer-songwriter known for being a member of Hall & Oates. [**Step 3**] The American duo Hall & Oates consists of Daryl Hall and John Oates. [**Step 4**] The bassist for Hall & Oates, which is none other than bassist and vocalist Klyde Jones is the lead vocalist who is also the bassist. The final answer is Hall & Oates.

Table 6: We demonstrate the performance of our method, CER, along with a sampled response path for solving each question in the open-domain generation dataset. Each step is clearly separated by a period and labeled as **[Step i]**, where *i* represents the step number. Intermediate answers, including the last proper noun in each step, are highlighted in blue, while the final answer is emphasized in green.

Question	**Sampled Paths**
Carefully work through the problem step by step. For each step, perform any required reasoning, and express the answer at the end of the step, Your response should be in the format Answer: [answer]. After completing the steps, provide the final answer based on the reasoning developed throughout the process. Your response should end with The final answer is [answer], where [answer] is the response to the problem. Q: a cube of side one meter length is cut into small cubes of side 20 cm each. how many such small cubes can be obtained?	**Step 1**: First, we need to understand the size of the large cube and the size of the small cubes. The large cube has a side length of 1 meter, and the small cubes have a side length of 20 cm. Answer: 20. **Step 2**: We need to convert the side length of the large cube from meters to centimeters for a uniform unit of measurement. Since 1 meter = 100 centimeters, the side length of the large cube is 100 cm. Answer: 100. **Step 3**: We will calculate the number of small cubes that can fit along the side of the large cube. The number of small cubes on one side of the large cube is 100 cm / 20 cm = 5. Answer: 5. **Step 4**: To find the total number of small cubes, we will multiply the number of small cubes along the length, width, and height. So, the total number of small cubes = 5 x 5 x 5 = 125. Answer: 125. The final answer is 125.
	Step 1: To find the number of small cubes in one dimension, we will divide the length of the big cube in meters by the side length of the small cube in meters. Since 1 meter = 100 cm, and small cube side = 20 cm, Answer: 100 / 20 = 5. **Step 2**: Since the small cubes will be cubical in shape, we need to get the result for all the dimensions (length, breadth, and height) and then multiply them to get the total number of cubes. We've already found out that 5 cubes can fit along the length and breadth (in step 1). As the cube is cube-shaped, the same number of cubes will fit along the height. Answer: 5 * 5 = 25. **Step 3**: Now we need to consider all the combinations (length, breadth, and height) to get the final answer. But since we already got the same result for length and breadth, there's no need to multiply again. Answer: 25. The final answer is 25.
	Step 1: First, we need to determine the volume of the large cube. Since the side length of the large cube is 1 meter, which is equivalent to 100 centimeters, we calculate its volume. Volume of the large cube = side^3 = 100^3 = 1,000,000 cubic centimeters. Answer: 1,000,000. **Step 2**: Next, we calculate the volume of a small cube. The side length of a small cube is given as 20 cm, so its volume is calculated by cubing this side length. Volume of a small cube = side^3 = 20^3 = 8,000 cubic centimeters. Answer: 8,000. **Step 3**: To find out how many small cubes can be obtained from the large cube, we divide the volume of the large cube by the volume of one small cube. Number of small cubes = Volume of the large cube / Volume of a small cube = 1,000,000 / 8,000 = 125. Answer: 125. The final answer is 125.

Table 7: We present the complete prompt and full answers for our main figure. This table includes the full prompt along with three sample paths, where intermediate answers at each step are highlighted in blue, and the final answer is emphasized in green.