

Research Paper Presentation

CER: Confidence Enhanced Reasoning in LLMs

Student: Kai-Yu Lu
Supervisor: Prof. Shanu Sushmita
Course: CS 8674 Master's Project

Agenda

1. Key Terms & Concepts
2. Motivation & Background
3. Method
4. Experiments & Results
5. Ablation Studies
6. Conclusion & Limitations

1. Key Terms & Concepts

Reasoning	Step-by-step derivation in complex tasks (math, knowledge-intensive QA...).
Chain-of-Thought (CoT)	The multi-step reasoning trace the model generates.
Self-Consistency (SC)	Sample K CoTs, take majority vote over final answers (each path has equal weight).
Confidence / Uncertainty	Model's certainty; derived from token probabilities (from logits). Higher confidence = lower uncertainty.
Critical tokens	Tokens where correctness matters: numbers (math) and proper nouns (open-domain QA).
Step-wise aggregation f	Map token probabilities within a word to a single word confidence.
Path-wise aggregation g	Aggregate all per-step confidences into one path confidence (later steps weighted more).
Weighted vote	Sum path confidences for each final answer; pick max.

2. Motivation & Background

Why Do We Need CER?

- **Real Problem:** LLMs “can compute but don’t know if they’re right”; reasoning needs confidence signals
- **SC Limitations:**
 - Multiple paths produce different and all-wrong answers;
 - Paths reach a **wrong consensus**
- **Human vs AI Uncertainty:** Humans express uncertainty for better decisions; AI must estimate it for risk control
- **Research Gap:** Majority voting gives equal weight; we should introduce confidence at key decision points

3. Method

CER – Confidence Enhanced Reasoning

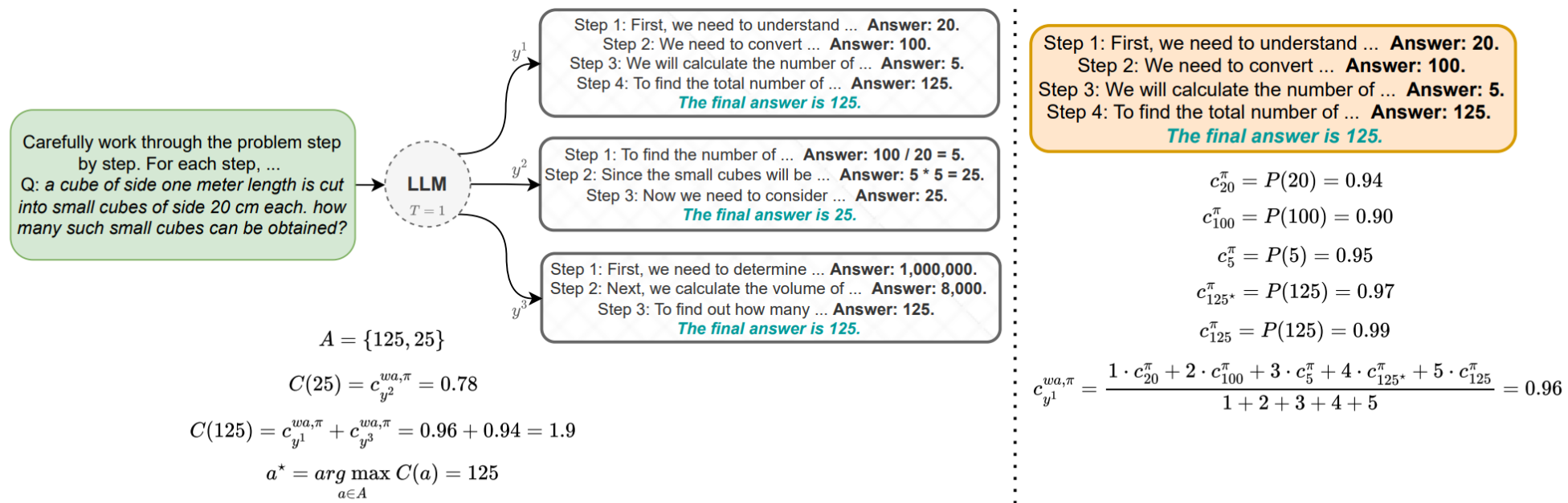


Figure 1: **Illustration of Confidence-Enhanced Reasoning (CER) in LLMs.** On the left, we demonstrate the CER framework. Given an input query, the LLM generates three independent outputs using temperature sampling ($T = 1$). Intermediate answers are bolded, and final answers are highlighted. The confidence 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 G.

3. Method

CER – Confidence Enhanced Reasoning

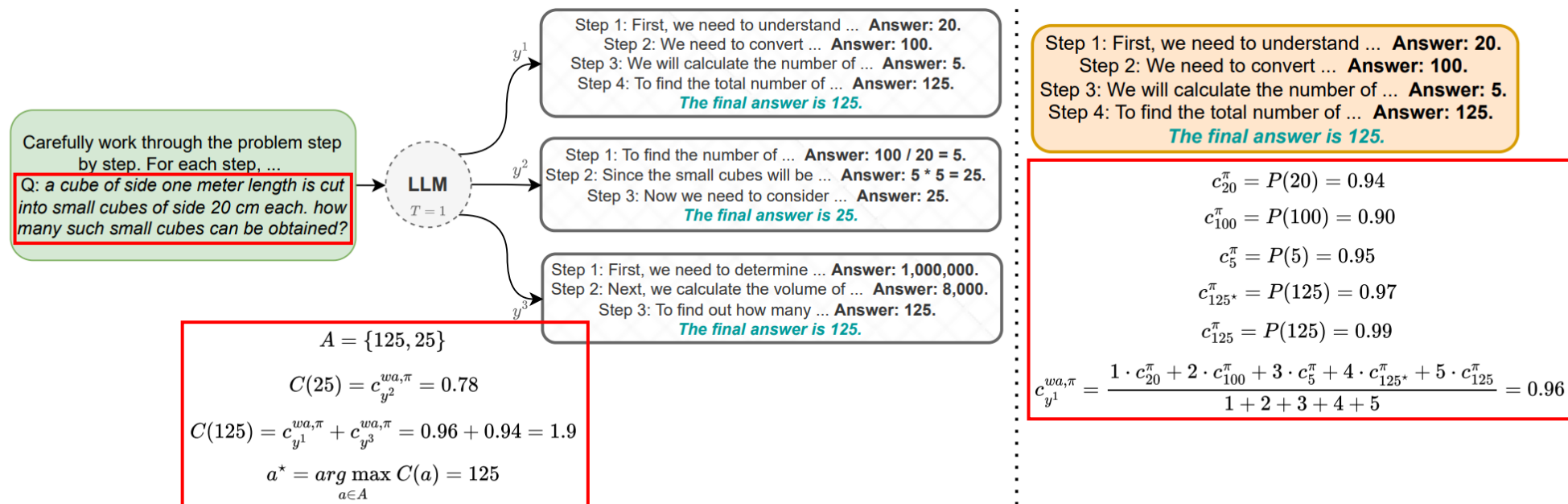


Figure 1: **Illustration of Confidence-Enhanced Reasoning (CER) in LLMs.** On the left, we demonstrate the CER framework. Given an input query, the LLM generates three independent outputs using temperature sampling ($T = 1$). Intermediate answers are bolded, and final answers are highlighted. The confidence 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 G.

3. Method

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:    $y^i = \left\{ (o_j^i, a_j^i) \right\}_{j=1}^{n^i}$ 
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:    $c_{y^i}^{g,f} \leftarrow g(c_{a_1^i}^f, \dots, c_{a_{n^i}^i}^f)$  ▷ 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:    $C(a) \leftarrow \sum_{i=1}^K c_{y^i}^{g,f} \cdot \mathbb{I}(\{A^i = a\})$  ▷ Eq. (5)
15: end for
16:  $a^* \leftarrow \arg \max_{a \in \mathcal{A}} C(a)$ 
17: return  $a^*$ 

```

3. Method

Some definitions...

1. Token Probability

Each generated token t in a reasoning path has an associated probability p_t , obtained from the model's softmax over logits:

$$p_t = \frac{e^{z_t}}{\sum_{t'} e^{z_{t'}}}$$

This value reflects how confident the model is when generating that token.

For example, if the model outputs the number “100,” the token probabilities may be:

$$p(1) = 0.97, \quad p(0) = 0.95, \quad p(0) = 0.98$$

These serve as the foundation for subsequent confidence computations.

3. Method

Some definitions...

2. Word Confidence

The **step-wise confidence function** f converts a group of token probabilities into a single scalar confidence for a word or phrase:

$$c_f^w = f(\{p_t \mid t \in w\})$$

The primary formulation used in CER is **multiplicative probability**:

$$c_f^w = \prod_t p_t$$

This captures how confident the model is about producing an entire intermediate answer (e.g., a number or name).

Example: for the number “100”, $p(1) = 0.97$, $p(0) = 0.95$, $p(0) = 0.98$

$$c_f^w = 0.97 \times 0.95 \times 0.98 = 0.90$$

3. Method

Some definitions...

3. Path Confidence

The **path-wise aggregation function** g combines the confidences of all intermediate answers along a reasoning path:

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

CER primarily adopts a **linearly weighted mean**, giving later steps higher importance:

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

This reflects the intuition that reasoning steps closer to the final answer contribute more to reliability.

3. Method

- ① Generate k CoT paths;
- ② Extract intermediate answers (digits/names);
- ③ Compute each p and derive word confidence c_f^w
- ④ Aggregate via g ;
- ⑤ Weighted vote across paths

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:    $y^i = \left\{ (o_j^i, a_j^i) \right\}_{j=1}^{n^i}$ 
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:    $c_{y^i}^{g,f} \leftarrow g(c_{a_{a_1}^i}^f, \dots, c_{a_{n^i}^i}^f)$  ▷ 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:    $C(a) \leftarrow \sum_{i=1}^K c_{y^i}^{g,f} \cdot \mathbb{I}(\{A^i = a\})$  ▷ Eq. (5)
15: end for
16:  $a^* \leftarrow \arg \max_{a \in \mathcal{A}} C(a)$ 
17: return  $a^*$ 
```

3. Method

① Generate k CoT paths;

The model samples k reasoning traces using temperature $T = 1$. Each path y^i is an independent reasoning sequence for the same question.
→ Output: k full reasoning paths (y^1, \dots, y^k) .

② Extract intermediate answers (digits/names);

From each path, identify **critical tokens**: numbers for math tasks, proper nouns for QA. These represent intermediate conclusions.
→ Output: (a_{i1}, a_{i2}, \dots) .

③ Compute each p and derive word confidence c_f^w

For each intermediate answer, compute its confidence score. The main f is **multiplicative probability** (Eq. 1–3):

$$c_f^w = f(\{p_t \mid t \in w\}) = \prod_t p_t$$

→ Output: word-level confidence $c_{a_{ij}}^f$.

④ Aggregate via g ;

Combine all step confidences into a single path score using **weighted mean** (Eq. 4):

$$c_{y^i}^{g,f} = \frac{\sum_{j=1}^{n^i} j \cdot c_{a_{ij}}^f}{\sum_{j=1}^{n^i} j}$$

Later steps get higher weights.

→ Output: path confidence $c_{y^i}^{g,f}$.

⑤ Weighted vote across paths

Aggregate over all paths (Eq. 5):

$$\mathcal{A} = \{a \mid a \in \{A^i\}_{i=1}^K\}$$

$$C(a) = \sum_{i=1}^K c_{y^i}^{g,f} \times \mathbb{I}(A^i = a) \quad \forall a \in \mathcal{A},$$

$$a^* = \arg \max_{a \in \mathcal{A}} C(a).$$

→ Output: final answer a^* with maximum total confidence.

3. Method

① Generate k CoT paths;

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

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.

y^2

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.

y^3

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.

$A = \{125, 25\}$

② Extract intermediate answers (digits/names);

③ Compute each p and derive word confidence c_f^w

④ Aggregate via g ;

⑤ Weighted vote across paths

y^1

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.

y^2

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.

y^3

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.

From each reasoning path y^i , we collect all numerical or symbolic outputs (20, 100, 5, 125...). These are denoted as intermediate answers a_{ij} .

For y^1 :


$$a_{11} = 20, a_{12} = 100, a_{13} = 5, a_{14} = 125$$

For y^2 : $(a_{21}, a_{22}) = (5, 25)$

For y^3 : $(a_{31}, a_{32}, a_{33}) = (1,000,000, 8,000, 125)$

Each a_{ij} will later be assigned a token-level probability $P(a_{ij})$ to compute confidence scores.

3. Method

- ① Generate k CoT paths;
 - ② Extract intermediate answers (digits/names);
 - ③ Compute each p and derive word confidence c_f^w
 - ④ Aggregate via g ;
 - ⑤ Weighted vote across paths
- 

(a) Token probability p_t

For each reasoning step, the model outputs logits z_t for the next token.

The token probability is obtained using the softmax:

$$p_t = \frac{e^{z_t}}{\sum_{t'} e^{z_{t'}}}$$

This gives the model's internal confidence for generating that token.

(b) From token probability to word-level confidence c_f^w

An intermediate answer ("100") may consist of several tokens.

CER defines the word confidence function f as:

$$c_f^w = f(\{p_t \mid t \in w\}) = \prod_{t \in w} p_t$$

If "100" is tokenized as three tokens "1", "0", "0" with probabilities $p(1) = 0.97, p(0) = 0.95, p(0) = 0.98$, then:

$$c_f^{100} = 0.97 \times 0.95 \times 0.98 = 0.90$$

If the tokenizer treats each number ("20", "100", "5", "125") as a single token, then c_f^w simply equals the model's softmax probability $P(w)$.

3. Method

- ① Generate k CoT paths;
- ② Extract intermediate answers (digits/names);
- ③ Compute each p and derive word confidence c_f^w
- ④ Aggregate via g ;
- ⑤ Weighted vote across paths



(c) Example for path y^1

The model produces:

Step	Intermediate Answer a_{1j}	Token Probability $P(a_{1j})$
1	20	0.94
2	100	0.90
3	5	0.95
4	125* (first occurrence)	0.97
5	125 (final answer)	0.99

These probabilities are extracted directly from the logits at each decoding step.

3. Method

① Generate k CoT paths;

② Extract intermediate answers (digits/names);

③ Compute each p and derive word confidence c_f^w

④ Aggregate via g ; 

⑤ Weighted vote across paths

Aggregate step-level confidence into path confidence $c_{g,f}^{y^i}$

To measure the reliability of an entire reasoning chain, CER applies the aggregation function g , which combines all intermediate confidences using a **linearly weighted mean**:

$$c_{y^i}^{g,f} = \frac{\sum_{j=1}^{n^i} j \cdot c_{a_{ij}}^f}{\sum_{j=1}^{n^i} j}.$$

Here,

- $c_{a_{ij}}^f$ is the **word-level confidence** of the intermediate answer a_{ij} ,
- j represents the step index (later steps get larger weights),
- n^i is the number of reasoning steps in the path y^i .

Example for the first path y^1 :

Token-level confidences (from Step ③) are:

Step	1	2	3	4	5
Token	20	100	5	125*	125
$c_{a_{ij}}^f$	0.94	0.90	0.95	0.97	0.99

Then,

$$c_{y^1}^{g,f} = \frac{1 \times 0.94 + 2 \times 0.90 + 3 \times 0.95 + 4 \times 0.97 + 5 \times 0.99}{1 + 2 + 3 + 4 + 5} = \frac{14.42}{15} \approx 0.96.$$

For the other paths:

$$c_{y^2}^{g,f} = 0.78, c_{y^3}^{g,f} = 0.94.$$

3. Method

- ① Generate k CoT paths;
- ② Extract intermediate answers (digits/names);
- ③ Compute each p and derive word confidence c_f^w
- ④ Aggregate via g ;
- ⑤ Weighted vote across paths

Finally, CER combines all path confidences to make the final decision.

For each distinct final answer a , we compute its **total weighted confidence**:

$$\mathcal{A} = \{a \mid a \in \{A^i\}_{i=1}^K\}$$

$$C(a) = \sum_{i=1}^K c_{y^i}^{g,f} \times \mathbb{I}(A^i = a) \quad \forall a \in \mathcal{A}$$

where A_i is the final output of path y^i .

In this example:

$$A = \{125, 25\}$$

Only y^2 predicts 25: $C(25) = c_{y^2}^{g,f} = 0.78$

y^1 and y^3 both predict 125:

$$C(125) = c_{y^1}^{g,f} + c_{y^3}^{g,f} = 0.96 + 0.94 = 1.90$$

The final output is chosen as the answer with **maximum total confidence**:

$$a^* = \arg \max_{a \in \mathcal{A}} C(a) = 125$$

4. Experiments & Results

Model Setup

Model	Size / Family	Role in the paper	Notes
Llama-3.1-8B	8B, Llama 3.1 family	Main open-source backbone evaluated across all datasets; base model in this paper	Covers both math and open-domain tasks.
Mistral-7B	7B, Mistral v2	Evaluated on all datasets	Compared on open-domain QA as well.
OLMo-2-7B	7B, OLMo-2	Evaluated on all datasets	Clear contrasts on math datasets.
Llama-3.3-3B	3B, Llama 3.3 family	Evaluated on all datasets	Lightweight model for cross-scale validation.

4. Experiments & Results

Dataset Setup

Dataset	Task Type	Description	Role in the paper (Category)
GSM8K	Math reasoning	Grade-school math word problems requiring step-by-step reasoning	Mathematical reasoning
MATH	Math reasoning	Competition-level math problems with multi-step derivations	Mathematical reasoning
MathQA	Math reasoning	Numerical and formula-based QA across diverse categories	Mathematical reasoning
TriviaQA	Open-domain QA	Large-scale factoid QA testing knowledge retrieval and answering	open-domain question answering
HotPotQA	Open-domain QA (multi-hop)	Requires multi-hop reasoning across multiple evidence passages	open-domain question answering

4. Experiments & Results

Baselines

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

4. Experiments & Results

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	24.0 (+3.6%)
MathQA	20.8	12.4	22.6	20.0	19.4	22.6	20.2	22.6 (+0.0%)
Average	34.4	24.2	32.4	33.0	33.1	32.6	27.3	37.2 (+2.8%)
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.

4. Experiments & Results

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.

4. Experiments & Results

Mathematical Reasoning:

- On GSM8K / MATH / MathQA, CER beats all baselines across models; the best math gain is +7.4% on MathQA with OLMo-2-7B.
- Compact models (Llama-3.3-3B) also improve strongly (avg +**4.1%**), showing **cross-scale effectiveness**.

Knowledge Intensive Reasoning:

- On TriviaQA / HotPotQA, CER remains best-in-class; +5.8% on TriviaQA with Mistral-7B is the top open-domain gain.
- For Llama-3.1-8B, CER reaches 66.0 / 14.4 on the two datasets, still improving over the best baseline.

4. Experiments & Results

Results Across Different Models:

- Generality: Across four open-source families (Llama-3.1-8B, Llama-3.2/3.3-3B, Mistral-7B, OLMo-2-7B), CER outperforms all baselines on both math and open-domain tasks.
- Compact models benefit: Llama-3.3-3B shows an average +4.1% on math, evidencing effectiveness on smaller backbones.
- Capacity vs. knowledge: Llama-3.2-3B outperforms Mistral-7B on math but lags on knowledge-intensive QA, likely due to limited parameter capacity for world knowledge.

5. Ablation Studies

1) Varying the Number of Paths (K)

- Increasing K from 3 to 5 to 10 improves all methods.
- CER remains the best at every K , so the gain is not just from larger sampling.

2) Entropy vs. Probabilities

- Mean entropy over the whole response is not sufficiently discriminative.
- Focusing on intermediate answers and using multiplicative token probabilities yields a sharper confidence signal.

3) Different Path-Level Aggregators

- Default is a weighted mean that emphasizes later steps.
- Alternatives such as harmonic mean, exponential weights, half-split, min, and product perform similarly, showing robustness to g .

4) Last Answer Confidence

- Using only the final answer's confidence discards useful evidence.
- CER-ALL** (aggregate all intermediate answers) clearly beats **CER-LAST** across datasets and models.

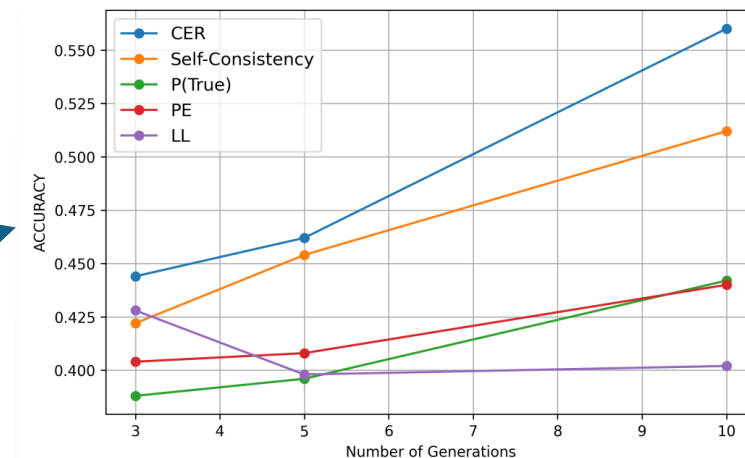


Figure 2: Performance comparison of CER and baseline models across different generations $K = \{3, 5, 10\}$ on the LLAMA 3.3-3B model using the MATH dataset.

5. Ablation Studies

5) Recent Reasoning-based Models

Models & Datasets	Self-Consistency	P(True)	PE	NL	NE	LL	Greedy	CER
GSM8K	89.8	82.6	83.8	85.2	83.8	84.0	87.2	90.2
Trivia QA	22.4	15.2	16.0	15.4	14.8	15.8	19.4	24.4

Table 3: Comparison of CER with baselines on DeepSeek-R1-Distill-Qwen-7B across the GSM8K and TriviaQA datasets.

- On a reasoning-specialized backbone (DeepSeek-R1-Distill-Qwen-7B), CER still outperforms all baselines.
- Even when the base model is already optimized for reasoning, CER delivers further gains: GSM8K 90.2 vs SC 89.8; TriviaQA 24.4 vs SC 22.4.

Summary: CER wins because it *targets key intermediate answers* and *aggregates confidence within and across paths*, remaining stable across K and aggregator choices.

6. Conclusion & Limitations

Conclusion:

- This paper introduces a lightweight framework that boosts reasoning without fine-tuning or task-specific prompts, operating purely on model output logits. It bridges reasoning with uncertainty estimation.

- Three key findings:
 1. Confidence-weighted aggregation over multiple chains is more reliable than equal-weight self-consistency; chains with higher confidence count more.
 2. Assessing confidence on intermediate-answer tokens is more effective than using all tokens in the chain; natural uncertainty during thinking shouldn't be penalized.
 3. Heavier weights on later steps (closer to the final answer) lead to better performance.

- Taken together, CER consistently lifts accuracy by better surfacing the most trustworthy answer across reasoning paths.

6. Conclusion & Limitations

Limitations:

- Scope: Validated mainly on mathematical reasoning and knowledge-intensive QA, where intermediate answers are extractable.
- Logit access: Requires output logits; thus not directly applicable to black-box APIs. A practical workaround is to use verbalized per-step confidence and then do path-wise aggregation.
- Output type: Focuses on numerical outputs; non-numerical/proof-style reasoning is feasible with minor modifications and is a direction for future work.

... .. End

Q&A