Mind the Generation Process: Fine-Grained Confidence Estimation During LLM Generation

Anonymous submission

Abstract

While large language models (LLMs) have demonstrated remarkable performance across diverse tasks, they fundamentally lack self-awareness and frequently exhibit overconfidence, assigning high confidence scores to incorrect predictions. Accurate confidence estimation is therefore critical for enhancing the trustworthiness and reliability of LLM-generated outputs. However, existing approaches suffer from coarse-grained scoring mechanisms that fail to provide fine-grained, continuous confidence estimates throughout the generation process. To address these limitations, we introduce FineCE, a novel confidence estimation method that delivers accurate, fine-grained confidence scores during text generation. Specifically, we first develop a comprehensive pipeline for constructing training data that effectively captures the underlying probabilistic distribution of LLM responses, and then train a model to predict confidence scores for arbitrary text sequences in a supervised manner. Furthermore, we propose a Backward Confidence Integration (BCI) strategy that leverages information from the subsequent text to enhance confidence estimation for the current sequence during inference. We also introduce three strategies for identifying optimal positions to perform confidence estimation within the generation process. Extensive experiments on multiple benchmark datasets demonstrate that FineCE consistently outperforms existing classical confidence estimation methods. Our code and all baselines used in the paper are available on GitHub [1].

Introduction

Self-awareness, as a core metacognitive ability, plays a crucial role in both human cognition and the advancement of large-scale AI systems (Dewey 1986; Kuhl and Beckmann 2012). For humans, it enables reflective thinking and error monitoring. Similarly, for large language models (LLMs), it supports output evaluation and self-correction, which is critical for handling complex reasoning tasks (Tong et al. 2024; Xie et al. 2025). Confidence estimation has emerged as a promising approach, enabling models to assess the reliability of their own generations (Zhou, Jurafsky, and Hashimoto 2023; Xiong et al. 2024; Branwen 2020).

However, existing confidence estimation methods for LLMs remain limited by their **coarse-grained** scoring and **narrow perspective**, failing to provide reliable and continuous confidence signals. Broadly, these works are categorized into question-oriented and outcome-oriented paradigms. ***Question-oriented*** methods aim to constrain LLMs to answer only questions within their domain of knowledge, allowing the model to give up responding when uncertain (Zhang et al. 2023). When faced with ambiguous or challenging questions, LLMs often decline to answer such questions directly (Kadavath et al. 2022), rather than attempting to infer a potential answer from the available context. While this conservative method helps prevent the model from generating incorrect answers, it also significantly limits the utility of LLMs in open-ended tasks. ***Outcome-oriented*** methods require LLMs to evaluate the quality of their generated answers after completing the generation process (Zhang et al. 2024a; Zhao et al. 2024; Kuhn, Gal, and Farquhar 2023a; Abbasi-Yadkori et al. 2024). However, relying solely on a single confidence score at the end of the generation is insufficient to capture the model's certainty throughout the entire reasoning trajectory. A high final confidence score does not indicate that the intermediate steps are completely accurate (Jiao et al. 2024). Figure 1 highlights the key differences between these two confidence estimation paradigms.

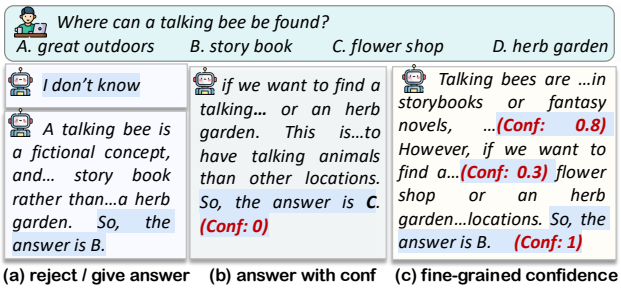

Figure 1: The difference between our proposed FineCE and existing confidence estimation methods. **(a):** LLMs either generate an answer when the query is within their knowledge scope or refuse to answer if it falls beyond their capabilities. **(b):** The model assigns a single confidence score after the entire answer is generated. **(c):** Our proposed method, FineCE, provides the fine-grained confidence scores for any given text sequence throughout the generation process.

Therefore, it is essential to develop **fine-grained confidence estimation** methods that provide accurate confidence scores for the intermediate steps during generation. This en-

[1] https://anonymous.4open.science/r/FineCE/

ables **early prediction** of whether the model is likely to produce a correct final answer, without having to wait for the full response. In addition, intermediate confidence scores serve as **supervisory signals** for LLMs with deep thinking capabilities, such as O1[2] and R1 (Guo et al. 2025). These signals inform the model's decision-making during generation, determining whether to proceed with the current trajectory or to revise earlier outputs. Furthermore, questions that consistently lead to low confidence scores expose **underlying weaknesses** in the model, offering actionable insights for targeted improvements.

Implementing fine-grained confidence estimation in LLMs is non-trivial and presents three major challenges. ***(Task Learning:)*** *In the absence of explicit confidence annotations, how can we teach LLMs to express fine-grained confidence?* LLMs are not inherently equipped with such capability (Tian et al. 2023a). Dedicated and task-specific supervised training is necessary. However, constructing supervisory data for this task poses a significant challenge. A key difficulty lies in the fact that distilling confidence scores from other advanced models is impractical, as the uncertainty captured by these models does not necessarily reflect that of the learner model itself.***(Effectiveness:)*** *How to provide accurate and unbiased confidence estimate for the current text?* During generation, LLMs predict each token sequentially without access to future content. Relying solely on confidence scores derived from the current partial output easily introduces bias and miscalibration. ***(Efficiency:)*** *What are the optimal positions for confidence estimation?* Estimating confidence after every generated token is often unnecessary and computationally inefficient. Instead, it is crucial to identify key positions during generation where confidence estimation has the greatest impact and provides the most value.

In this paper, we introduce FineCE, a fine-grained confidence estimation method for LLMs via supervised learning. Specifically, to capture the distributional uncertainty inherent in an LLM, we design a complete data construction pipeline based on Monte Carlo Sampling. Additionally, we introduce a Backward Confidence Integration (BCI) strategy at the inference stage, which further refines the confidence estimation for current predictions by utilizing uncertainty information from subsequently generated tokens. To better balance the trade-off between confidence estimation performance and computational efficiency, we propose three strategies to identify optimal positions within the generation process for performing confidence estimation.

Experiments demonstrate that FineCE can reliably estimate the likelihood of a correct final answer as early as one-third into the generation process, offering strong early-stage confidence signals. To further validate its effectiveness, we apply FineCE to a downstream task using a confidence-based filtering strategy that retains only responses exceeding a predefined threshold. This strategy leads to a substantial 39.5% improvement in accuracy on the GSM8K dataset.

In summary, our contributions are four-fold:

- We propose FineCE, a fine-grained confidence estimation method that enables accurate prediction of answer correctness during the generation process.
- We design a complete pipeline for constructing high-quality training data that effectively captures the distributional uncertainty of LLMs.
- We introduce BCI, a novel backward confidence integration strategy that enhances current confidence estimation by incorporating uncertainty information from subsequent texts.
- We develop three practical strategies to identify optimal positions for confidence estimation within the generation process.

[2]https://openai.com/openai-o1-contributions

Task Formalization

The confidence estimation task aims to improve model calibration by aligning predicted probabilities with the likelihood of correct outputs. Here, ***confidence is defined as the probability that the model's answer is correct***.

Formally, LLMs generally generate responses in an autoregressive manner, predicting the next token sequentially based on the previously generated context. Given an input x and an LLM M, the model generate a sequence of output tokens $y = t_1, t_2, \cdots, t_n$, where each token t_i is sampled from the distribution $P_i = \mathcal{P}(\cdot \mid x, t_{<i}; M)$, with $t_{<i} = t_1, \cdots, t_{i-1}$ and n denoting the total number of generated tokens. Let $\bar{Y}$ denote the ground-truth output. Given any intermediate generation sequence s, we define the confidence score as:

$$Conf_s = p(y = \bar{Y}|s, M) \tag{1}$$

The confidence score $Conf_s$ of a sequence s, which can be a partial or complete answer, represents the probability that model M generates the correct output $\bar{Y}$, conditioned on s. Depending on the form of s, we categorize the confidence estimation task into the following three variants:

- **Question-oriented confidence estimation.** In this setting, s contains only the input question, that is, $s = x$.
- **Process-oriented confidence estimation.** s consists of the input question and a partially generated answer, i.e., $s = (x, t_{<i})$, where $t_{<i}$ is a prefix of the full output sequence y.
- **Outcome-oriented confidence estimation.** In this case, s includes both the input and the complete generated response, that is, $s = (x, y)$.

This formulation unifies existing confidence estimation settings under a common probabilistic view. It also extends the task to cover all stages of the generation process.

FineCE: Fine-grained Confidence Estimation

Data Construction

Preliminary. Traditional classification models struggle to reflect predictive uncertainty, as softmax probabilities are often misinterpreted as confidence scores. A high softmax output does not necessarily indicate that the model is certain about its prediction (Gal and Ghahramani 2016). Therefore, to obtain the LLM's inherent real responses probability based on the text s, we introduce Monte Carlo Sampling(Li et al. 2024) and employ the generative LLM M to repeatedly sample k answers $\{A_s^1, A_s^2, \cdots, A_s^k\}$ at high temperature to approximate the probability of generating the correct answer.

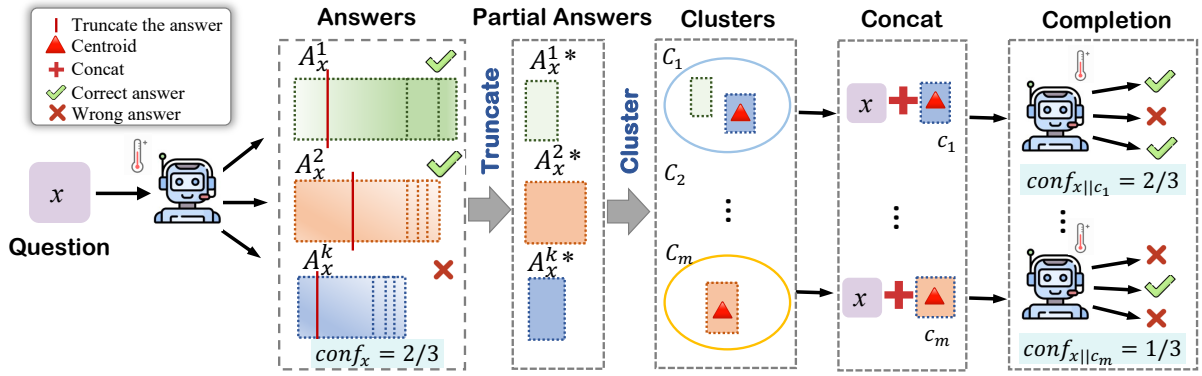

Figure 2: The construction process of the training dataset. It illustrates the confidence scoring procedures for *Question* and Q*uestion with Partial Answer* using Monte Carlo sampling. For *Question with Answer*, the confidence score is determined based on the correctness of the answer. The complete data construction procedure is detailed in Algorithm .

According to the *Law of Large Numbers*, as k approaches infinity, the sample mean will converge to the true probability of the model generating the correct answer.

Overall Pipeline. In our work, the input text sequence s includes three distinct types: *Question*, *Question with Partial Answer* and *Question with Answer*. The confidence score $Conf_s$ is calculated as the accuracy ratio of k generated answers compared to a reference or golden answer $\bar{Y}$, which is defined as follows:

$$Conf_s = \frac{\sum_{i=1}^{k} \mathbf{I}(A_s^i = \bar{y}_s)}{k}, \tag{2}$$

where A_s^i is the ith sampling answer generated based on sequence s, and $\bar{y}_s$ is the ground-truth answer. The indicator function $\mathbf{I}$ returns 1 when the answer matches and 0 otherwise.

Confidence score for *Question*. For each input question x, we first generate k diverse complete answers $\{A_x^1, A_x^2, \cdots, A_x^k\}$ from the model M using a high-temperature sampling strategy. Here, A_x^i represents the ith response conditioned on input x. The confidence score for x is calculated according to Equation 2.

Confidence score for *Question with Partial Answer*. To construct training data for confidence estimation on partial answers, we apply a truncation procedure to each complete answer A_x^i, yielding a sequence of partial answer fragments. Each fragment is then concatenated with the original question x and fed into the model to generate multiple completions. These completions are subsequently used to estimate the confidence score associated with the partial answer.

We leverage an intrinsic property of LLMs to reduce the computational overhead associated with constructing training datasets. Specifically, when processing inputs with identical prefixes, their internal contextual representations tend to converge, resulting in highly similar conditional probability distributions for subsequent generations (Porretta et al. 2025).

Based on this observation, we propose a *progressive data construction pipeline*. Starting with an initial set of k partially completed answer fragments obtained via truncation, we first perform semantic clustering to group these fragments into m clusters, where $1 \leq m \leq k$. Each cluster contains semantically similar fragments. We then select a centroid fragment from each cluster to serve as its representative. Each selected representative is then concatenated with the original question to generate k new complete answer trajectories through Monte Carlo sampling, which is facilitates the estimation of a confidence score for each representative. From the sampled trajectories, we identify a semantically representative answer and apply another truncation operation to obtain a new partial answer.

This process is iteratively repeated, with each iteration yielding new set of partial answers along with the confidence estimates. The total number of truncation is limited to a maximum of $\mathcal{T}$.

Confidence score for *Question with Answer*. Upon completion of the process described above, we obtain a diverse set of partial answers, each associated with a corresponding confidence estimate. Simultaneously, each Monte Carlo sampling step yields a complete answer to the input question x. If a sampled answer matches the ground truth, it is assigned a confidence score of 1.0; otherwise, it receives a score of 0.0.

The overall training data construction pipeline is illustrated in Figure 2 and detailed in Algorithm 1. The formats of three data types shown in Figure **??**.

Complexity Analysis. The primary cost in constructing the training dataset arises from the number of forward passes required during Monte Carlo sampling. Without any optimization, generating three types of confidence estimates for each problem instance leads to an exponential growth in overall generation cost. This process can be viewed as maintaining a full k-ary tree of depth $\mathcal{T}+1$, resulting in a total of $\sum_{i=1}^{\mathcal{T}+1} k^i$ model inferences. To reduce complexity, clustering based on semantic similarity can be performed among sibling nodes at each hierarchical level. The generation cost is reduced to $k\sum_{i=0}^{\mathcal{T}} m^i$. Here, instead of first clustering the k generated candidates and then selecting the centroid of each cluster, we perform truncation by directly selecting a semantically rep-

resentative candidate from the k answers at each step, from the 2nd to the $\mathcal{T}$-th. This strategy significantly reduces the total generation cost to $k(1 + m\mathcal{T})$. As a result, in our work, the overall complexity of constructing the training data is **reduced from exponential to linear with respect to $\mathcal{T}$.**

Training Technique

To enhance the confidence estimation capability of LLMs, we explore two distinct training techniques, including the Additional Value Head and Instruction Fine-Tuning (IFT) (Ouyang et al. 2022). The additional value head skill reformulates confidence estimation as a multi-classification task, enabling token-level confidence predictions across the generated sequence. In contrast, IFT leverages the model's natural language generation capabilities to produce confidence estimates in a more interpretable format and human-readable format. In the Figure 7, we provide a comprehensive comparison of these two technique in our proposed task. In this work, FineCE adopts the IFT training paradigm.

Identify the Calibration Position

FineCE introduces fine-grained confidence estimation for LLMs. Calibrating confidence after each token generation is impractical due to computational costs. To reduce the computational overhead of token-wise confidence calibration, FineCE introduces three strategies to selectively perform confidence estimation during generation.

Paragraph-End Calibration conducts estimation at natural linguistic boundaries, such as paragraph ends. It maintains semantic coherence with minimal disruption to the generation flow.

Periodic Calibration performs estimation at fixed token intervals (e.g., every 50 tokens). This regular, interval-based strategy offers a deterministic mechanism for confidence monitoring, ensuring consistent quality assessment across the entire generated sequence.

Entropy-based Calibration triggers estimation when the model's output entropy exceeds a predefined threshold. While entropy reflects uncertainty, it alone is not sufficient for accurate confidence prediction. The calibration is more meaningful and reliable when entropy values are higher.

Backward Confidence Integration (BCI)

Existing confidence estimation methods rely solely on local features while overlooking the global context, resulting in incomplete or biased estimation. However, training data construction typically adopts a backward evaluation paradigm, labeling intermediate steps based on the correctness of the final answer (Yao et al. 2023; Qi et al. 2025). Yet, this valuable supervision signal is rarely exploited during inference. Therefore, to further revise either excessively high or low confidence level and mitigate output confidence bias, we propose Backward Confidence Integration (BCI). It extends the backward evaluation principle from training to inference.

Formally, for a generated text sequence, $Conf_{s_j}$ denotes the initial confidence estimation at the jth calibration position in a generated sequence. The adjusted confidence score $Conf^{'}_{s_h}$ is computed recursively for positions $h \in (j, j + d)$, which is defined as:

$$Conf'_{s_j} = \begin{cases} \alpha Conf_{s_j} + (1-\alpha)\frac{1}{w}\sum_{b=1}^{w} Conf'_{s^b_{h+1}}, \\ \qquad\qquad\qquad\qquad h < j + d \\ Conf_{s_h}, h = j + d \end{cases} \tag{3}$$

Here, $\alpha \in [0, 1]$ is the revision coefficient balancing the original local confidence and the influence of future context. A smaller α places placing more weight on future text. The parameters w defines the number of sampled generation paths (integration width), and d specifies how many future positions are considered (integration depth). $Conf_{s^b_h}$ denotes the adjusted confidence at the hth calibration position in the bth sample. By recursively incorporating backward signals from future steps, it provides a more globally accurate estimation of confidence for each calibration position.

Experiments

Experiment Setting

Dataset and Metrics. We evaluate the performance of confidence estimation across six datasets including *GSM8K*(Cobbe et al. 2021), *TriviaQA*(Joshi et al. 2017), *CommonsenseQA*(CSQA; (Talmor et al. 2018)), *AIME24*[3], *MMLU* (Hendrycks et al. 2021), and *NQ-Open* (Kwiatkowski et al. 2019).

We adopt several widely used metrics including Expected Calibration Error (ECE), Receiver Operating Characteristic Curve (AUROC) and Accuracy (ACC).

Models and Baselines. We employ three widely-used open-source models, including Llama2-13B (Touvron et al. 2023), Llama3.1-8B (Dubey et al. 2024) and Qwen2.5-7B (Yang et al. 2024). The baselines we used in this paper include the following three types: 1) **Question-oriented:** P(IK) (Kadavath et al. 2022); 2) **Outcome-oriented:** First-Prob, SuC (Lin, Hilton, and Evans), Verbalized Porb (Verb (Tian et al. 2023a)) Semantic Uncertainty (SE, (Kuhn, Gal, and Farquhar 2023b)); 3) **Step-wise estimation:** Multi-Step (MS; (Xiong et al. 2024)), LECO (Yao et al. 2024).

Comprehensive experimental details, including dataset baseline introduction, prompts used, key hyperparameters, and computational platforms, are provided in Appendix. Beyond the core results presented in the main text, we conduct additional analyses to address four critical questions regarding FineCE's practical applicability: (1) generalization ability across different domains, (2) sensitivity to training data, (3) impact of different training strategies, and (4) performance on highly open-ended questions of FineCE. These supplementary analyses are detailed in Appendix.

Main Results and Analysis

RQ1: How does FineCE perform compared with baselines? In this experiment, to ensure fair comparison, we fix the parameters w and b in FineCE to 0, eliminating the computational advantage of BCI, thereby aligning inference costs with baseline methods. The overall results are shown in Table 1 and Table 2.

[3]https://huggingface.co/datasets/math-ai/aime24

	Pos	Metrics	Llama2-13B			Llama3.1-8B			Qwen2.5-7B		
			MS	LECO	FineCE	MS	LECO	FineCE	MS	LECO	FineCE
GSM8K	$p(1)$	AUROC	55.6	60.5	**73.8**	60.8	62.2	**66.2**	64.7	64.4	**66.8**
		ECE	23.5	19.2	**9.3**	27.4	21.1	**15.7**	23.6	21.1	**14.1**
	$p(z-1)$	AUROC	57.3	59.5	**77.7**	62.3	64.7	**69.4**	63.8	65.3	**65.3**
		ECE	22.8	21.3	**8.4**	29.7	23.7	**17.3**	25.2	20.4	**14.4**
	AVG	AUROC	57.1	61.1	**78.1**	62.4	68.2	**72.7**	67.2	64.1	**76.4**
		ECE	21.1	19.6	**6.7**	28.3	19.2	**12.3**	19.2	20.1	**10.7**
CSQA	$p(1)$	AUROC	54.6	57.1	**66.2**	61.0	63.1	**66.3**	63.9	62.0	**68.1**
		ECE	24.8	23.8	**18.3**	29.4	22.4	**16.6**	27.6	19.2	**17.3**
	$p(z-1)$	AUROC	53.2	56.0	**69.3**	57.2	62.9	**67.5**	62.0	63.9	**68.2**
		ECE	26.9	25.7	**16.2**	33.0	26.3	**17.9**	24.4	20.8	**17.1**
	AVG	AUROC	58.6	59.6	**71.3**	59.3	65.0	**71.1**	65.5	65.3	**73.2**
		ECE	23.1	21.4	**11.7**	29.3	23.1	**13.3**	25.0	17.6	**14.7**
TriviaQA	$p(1)$	AUROC	56.1	53.4	**70.8**	63.4	60.7	**69.2**	61.9	62.1	**67.4**
		ECE	22.2	26.8	**14.5**	27.9	21.4	**18.7**	26.4	22.7	**19.3**
	$p(z-1)$	AUROC	56.4	58.3	**74.2**	62.0	63.4	**67.7**	59.4	64.4	**71.1**
		ECE	25.6	27.3	**15.0**	26.3	20.9	**20.3**	30.2	23.4	**17.5**
	AVG	AUROC	57.2	58.1	**76.1**	63.7	62.6	**73.3**	63.2	64.0	**73.9**
		ECE	22.8	25.5	**11.3**	25.1	19.3	**14.2**	25.3	20.2	**13.4**
AIME24	$p(1)$	AUROC	21.4	56.3	**68.4**	16.2	63.4	**69.8**	25.3	64.1	**74.1**
		ECE	57.4	37.4	**19.3**	60.3	31.2	**21.5**	64.3	33.7	**22.4**
	$p(z-1)$	AUROC	25.4	59.4	**71.3**	25.3	66.3	**68.4**	11.6	65.2	**76.2**
		ECE	64.3	34.3	**22.4**	57.2	29.4	**23.5**	76.8	30.2	**21.3**
	AVG	AUROC	22.7	56.3	**76.0**	19.5	64.1	**71.3**	30.3	64.0	**79.2**
		ECE	59.2	33.8	**16.5**	55.4	30.8	**20.4**	72.3	29.6	**18.3**
MMLU	$p(1)$	AUROC	57.4	61.3	**74.3**	53.1	59.2	**70.3**	54.1	60.3	**70.2**
		ECE	27.6	26.2	**20.1**	30.3	27.8	**20.2**	32.9	30.3	**22.4**
	$p(z-1)$	AUROC	59.3	62.5	**71.8**	56.4	61.3	**73.1**	52.6	57.4	**71.3**
		ECE	29.4	28.1	**18.9**	33.6	29.3	**17.3**	33.4	28.7	**19.3**
	AVG	AUROC	58.9	60.5	**74.6**	57.2	63.4	**74.6**	58.4	61.2	**74.2**
		ECE	28.3	27.3	**15.3**	28.9	26.9	**14.1**	31.1	28.4	**15.7**
NQ-Open	$p(1)$	AUROC	59.4	62.1	**72.3**	55.8	61.0	**72.3**	55.3	62.8	**72.0**
		ECE	30.1	26.0	**17.8**	34.9	28.7	**23.7**	35.1	29.4	**17.5**
	$p(z-1)$	AUROC	60.4	57.3	**70.9**	57.3	59.4	**67.5**	58.1	61.3	**70.3**
		ECE	29.6	27.0	**20.3**	29.2	26.3	**18.1**	30.4	30.5	**20.5**
	AVG	AUROC	60.7	59.1	**75.5**	57.9	62.3	**74.7**	58.8	64.2	**76.9**
		ECE	27.4	25.7	**14.2**	32.3	26.1	**18.2**	32.8	28.6	**16.4**

Table 1: Confidence estimation results throughout the generation process. z is total number of paragraphs in an generated answer. $p(1)$ and $p(z-1)$ represent the confidence estimates for the first and the penultimate paragraphs, respectively.

As shown in Table 1, existing confidence estimation approaches suffer from a fundamental limitation. That is, they fail to capture meaningful uncertainty signals during text generation. FineCE consistently achieves AUROC scores exceeding 70%, outperforming baseline methods by 10–15 percentage points. In contrast, baselines generally achieve AUROC scores between 57% and 65%, indicating performance barely above random chance. Notably, FineCE maintains stable performance across different generation positions (p(1) and p(z-1)), indicating robust confidence estimation throughout the entire generation process. The ECE results further confirm superior calibration, with FineCE achieving significantly lower calibration errors (6.7-16.5%) compared to baseline methods (19.2-28.3%).

From Table 2, FineCE consistently outperforms all baselines across both ECE and AUROC metrics on six diverse datasets. The most striking result appears on GSM8K with Llama2-13B, where FineCE achieves an ECE of 5.1% and AUROC of 77.8%, representing substantial improvements over the strongest baseline P(IK). This pattern of consistent superiority holds across different model architectures, with FineCE achieving 5-15% AUROC improvements and 30-60% relative ECE reductions across experimental conditions.

These results reveal two critical findings. Firstly, existing confidence estimation methods perform poorly across generation positions, often approaching random performance levels. Besides, FineCE's supervised learning method with fine-grained training data construction enables significantly more accurate confidence estimation during the generation. Importantly, these improvements come without sacrificing answer accuracy (the accuracy results are shown in Appendix Table 4), achieved through our replaying strategy and the careful dataset mixing during fine-tuning.

Overall, FineCE consistently enables base models to produce accurate confidence estimates throughout the generation process across diverse tasks, substantially outperforming ex-

Models	Baselines	GSM8K		CSQA		TriviaQA		AIME24		MMLU		NQ-Open	
		ECE↓	AUROC↑	ECE↓	AUROC↑	ECE↓	AUROC↑	ECE↓	AUROC↑	ECE↓	AUROC↑	ECE↓	AUROC↑
Llama3.1-8B	*P(IK)*	17.6	72.8	19.4	68.7	20.4	67.7	33.1	67.9	18.3	72.1	22.4	68.2
	FineCE	13.5	76.4	16.0	68.4	15.5	69.8	18.5	73.1	14.3	76.2	20.9	73.1
	First-Prob	26.2	66.2	23.5	66.8	24.9	65.1	40.3	65	21.4	68.4	29.4	66.5
	SuC	28.4	62.0	32.7	59.1	29.7	60.4	42.7	62.2	24.7	66.3	27.3	61.4
	Verb	20.4	72.9	28.0	68.4	30.1	69.1	73.4	6.1	31.2	62.7	34.0	65.2
	SE	17.6	73.5	21.3	66.7	19.4	66.4	20.9	68.5	17.2	71.2	22.3	70.4
	FineCE	12.7	77.1	14.2	72.8	14.6	70.5	20.7	70.4	12.1	74.1	17.1	75.1
Qwen2.5-7B	*P(IK)*	17.4	68.3	16.3	68.4	21.6	67.9	27.9	66.3	16.1	69.8	20.8	72.3
	FineCE	11.4	72.3	14.7	70.6	15.2	69.2	21.2	76.2	15.6	73.1	17.4	76.2
	First-Prob	25.4	66.4	26.6	65.2	25.9	62.3	35.8	57.4	30.3	68.0	24.5	68.5
	SuC	29.0	57.4	28.2	63.1	32.7	58.5	38.4	60.4	27.0	62.4	24.1	63.1
	Verb	15.3	72.2	12.4	70.3	22.0	68.4	78.7	11.3	29.4	63.3	33.6	62.4
	SE	18.6	72.1	19.3	69.4	22.5	68.4	25.1	73.5	22.4	68.3	23.8	71.8
	FineCE	10.2	75.3	13.1	70.8	15.4	72.5	17.7	81.3	16.3	75.7	15.3	77.8
Llama2-13B	*P(IK)*	14.5	64.8	29.9	59.5	18.7	65.0	31.4	72.1	17.3	67.6	18.3	70.7
	FineCE	8.9	67.3	16.2	69.3	15.5	68.4	24.8	78.4	15.0	72.6	13.9	74.3
	First-Prob	23.3	59.7	22.3	60.1	27.6	57.1	42.0	61.2	19.4	64.3	22.1	65.1
	SuC	28.8	57.3	27.2	56.7	23.5	58.2	37.3	57.3	22.1	65.2	24.6	66.4
	Verb	29.3	56.2	21.7	58.3	27.1	53.7	82.3	14.9	32.6	61.1	29.8	62.4
	SE	18.4	68.6	16.3	65.4	19.5	63.1	32.7	65.1	20.3	69.4	24.1	70.2
	FineCE	5.1	77.8	11.5	70.5	12.0	76.9	16.2	75.3	14.8	75.4	14.2	74.6

Table 2: Confidence estimation results across baselines on *Question-oriented* and *Outcome-oriented* tasks.

isting popular confidence estimation methods.

Downstream Application

RQ2: How does FineCE perform on downstream applications? We evaluate FineCE's practical utility through early-stage confidence estimation and confidence-based filtering. From Table 3, we observe that **FineCE achieves reliable confidence estimation using just ∼30% of generated tokens.** Token ratio analysis reveals an interesting pattern: simpler datasets like GSM8K require fewer tokens for reliable estimation (30.4%), whereas more complex reasoning tasks such as CSQA and TriviaQA require slightly more context (up to ∼34%). This suggests that FineCE adapts its information requirements based on task complexity, with mathematical reasoning allowing earlier confidence assessment than knowledge-intensive or commonsense reasoning tasks.

Furthermore, we implement confidence-based filtering with threshold δ, retaining only responses exceeding the confidence threshold. From Figure 3 (Left), we observe FineCE shows consistent accuracy improvements across datasets. The strong correlation between partial-response confidence estimates and final answer correctness validates its effectiveness as a output quality gate, enabling models to reject low-confidence responses before full generation. This capability is particularly valuable in deployment scenarios demanding computational efficiency and reliability, as it enables early termination of potentially incorrect responses.

Further Analysis

RQ3: Where does FineCE perform the confidence estimation? We conduct a comparative analysis of three calibration position strategies using the Llama2-13B model. For the Entropy-based strategy, we set the entropy threshold to *1e-10*, while for the Periodic Calibration strategy, we fix the calibration interval to every 30 tokens. The results are presented in Table 3.

Dataset	Strategy	ECE_{p_1}	ECE_{avg}	Token Ratio
GSM8K	Paragraph	9.8	7.7	30.4%
	Entropy	13.2	7.7	10.0%
	Fixed-token	13.1	10.8	23.5%
CSQA	Paragraph	26.8	13.0	22.0%
	Entropy	27.1	18.8	7.0%
	Fixed-token	24.2	20.7	34.7%
TriviaQA	Paragraph	17.2	14.5	28.5%
	Entropy	18.5	15.4	13.4%
	Fixed-token	20.0	18.0	34.1%

Table 3: Performance comparison of three strategies for identifying optimal calibration positions. *Token Ratio* represents the proportion of tokens preceding the calibration position relative to the total number of tokens in the complete answer. The backbone model used is Llama2-13B.

We observe that **all three strategies demonstrate comparable performance in terms of ECE, with Paragraph-end Calibration strategy yielding slightly better results.** We attribute this improvement to the fact that calibrating at paragraph boundaries helps preserve the full semantic context, which is essential for reliable confidence estimation.

Based on these findings, we draw the following insights. For general tasks, performing confidence estimation at paragraph boundaries is both efficient and effective, significantly reducing unnecessary token consumption. In contrast, for more complex tasks that require finer-grained assessment, the Entropy-based strategy achieves more frequent and accurate confidence estimation through dynamic calibration guided by uncertainty.

RQ4: How effective is the BCI strategy? We conduct

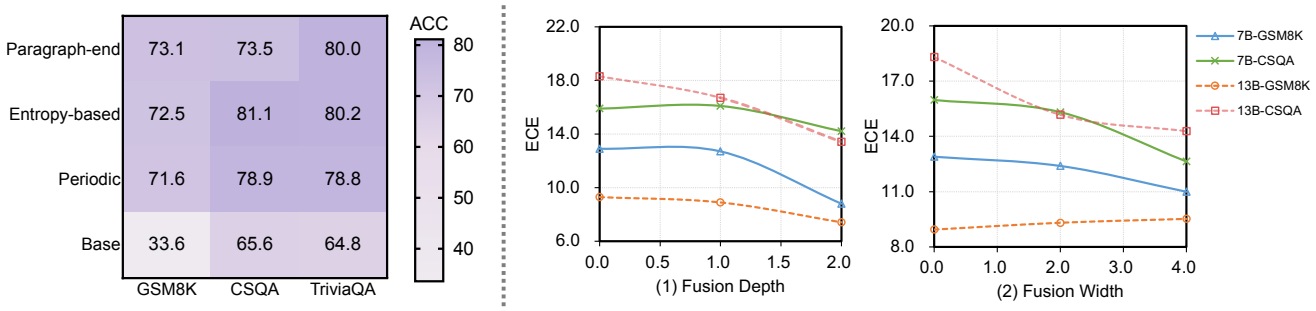

Figure 3: **(Left:)** Comparison of accuracy between the original model predictions and those selectively accepted by FineCE when the output confidence exceeds 0.8. The backbone used is Llama2-13B. **(Right:)** Effect of fusion depth (1) and fusion width (2) in FineCE on confidence estimation performance, evaluated with Llama-7B and Llama-13B on the GSM8K and CSQA datasets.

ablation experiments on GSM8K and CSQA datasets using Llama2-7B[4] and Llama2-13B models to evaluate the impact of the BCI strategy. Figure 3 (Right) shows ECE results for p(1), where d=0 and w=0 represents the FineCE baseline without BCI.

The results demonstrate that BCI consistently improves calibration across all model-dataset combinations. As fusion depth d increases from 0 to 2, ECE drops substantially. On CSQA with Llama2-7B, ECE decreases from 15.3 to 12.6. Similarly, increasing fusion width w from 0 to 4 yields progressive calibration gains, with ECE reductions of up to 15% on CSQA datasets.

The improvements are particularly pronounced for larger models and more complex reasoning tasks. Llama2-13B benefits more significantly from BCI than Llama2-7B, suggesting that BCI becomes more effective as model capacity increases. Interestingly, CSQA shows greater sensitivity to fusion width compared to GSM8K, indicating that knowledge-intensive tasks require broader cross-attention integration to capture diverse reasoning pathways.

Related Work

Verifier and Calibration Model. Although the calibration model and the verifier take similar inputs and produce comparable outputs, they serve fundamentally different purposes. The verifier is designed to evaluate the quality of a given response in a model-independent manner, assigning consistent evaluation scores regardless of which model generated the answer (McAleese et al. 2024; Ke et al. 2023; Huang et al. 2024). In contrast, the calibration model estimates the probability that a specific output is correct with respect to a given model. This confidence score is inherently model-dependent, as different language models may generate different responses to the same input, each with varying degrees of correctness. (Atil et al. 2024; Song et al. 2025; Renze 2024). To summarize, the verifier enables standardized, model-agnostic evaluation of response quality, while the calibration model captures model-specific epistemic uncertainty during generation, reflecting the model's internal confidence in its own outputs.

[4]https://huggingface.co/meta-llama/Llama-2-7b

Our work focuses on developing a generalizable method that delivers fine-grained and accurate confidence estimates for arbitrary text outputs, and and evaluates the calibration capability based on the model's actual responses.

Confidence Expression in LLMs. Recent studies have explored how LLMs express confidence, mainly through internal signals or explicit verbalization. Leverage internal representations or logits to estimate uncertainty (Su et al. 2024; Chen, Vondrick, and Mao 2024; Azaria and Mitchell 2023). For example, Chen et al. (2024) analyzes eigenvalues from internal vectors to detect errors, while Robinson, Rytting, and Wingate (2023) uses token-level logits to measure the uncertainty. Others introduce components like a "Value Head" to probe self-assessed confidence (Kadavath et al. 2022), but these methods are limited to structured tasks. Another line of work prompts LLMs to verbalize their confidence directly (Zhou, Jurafsky, and Hashimoto 2023; Xiong et al. 2024; Zhang et al. 2024b). Techniques include few-shot prompting (Branwen 2020), supervised training with external labels (Tian et al. 2023b), and explicit guidance for confidence output (Lin, Hilton, and Evans). However, models still exhibit overconfidence and struggle with the complex instructions (Xiong et al. 2024).

Conclusion

In this paper, we propose FineCE, a fine-grained confidence estimation method designed to provide accurate confidence scores throughout the generation process. We first differentiate FineCE from existing popular confidence estimation approaches, emphasizing its unique advantages. We then detail the training dataset construction procedure used in FineCE, followed by the introduction of three basic strategies to identify the optimal confidence estimation positions. Additionally, during the inference stage, we further present the BCI strategy, which enhances confidence estimation by incorporating the future text to provide a more comprehensive estimation for the current output. Extensive experiments demonstrate that FineCE consistently outperforms existing methods across various confidence estimation tasks. We also further validate its effectiveness on several downstream applications.

References

Abbasi-Yadkori, Y.; Kuzborskij, I.; György, A.; and Szepesv'ari, C. 2024. To Believe or Not to Believe Your LLM. *ArXiv*, abs/2406.02543.

Atil, B.; Chittams, A.; Fu, L.; Ture, F.; Xu, L.; and Baldwin, B. 2024. LLM Stability: A detailed analysis with some surprises. *arXiv preprint arXiv:2408.04667*.

Azaria, A.; and Mitchell, T. 2023. The Internal State of an LLM Knows When It's Lying. In *In Findings of the Association for Computational Linguistics: EMNLP*.

Branwen, G. 2020. Gpt-3 nonfiction- calibration. Technical report, The institution that published. Last accessed on 2022-04-24.

Chen, C.; Liu, K.; Chen, Z.; Gu, Y.; Wu, Y.; Tao, M.; Fu, Z.; and Ye, J. 2024. INSIDE: LLMs' Internal States Retain the Power of Hallucination Detection. *CoRR*.

Chen, H.; Vondrick, C.; and Mao, C. 2024. SelfIE: Self-Interpretation of Large Language Model Embeddings. *ArXiv*, abs/2403.10949.

Cobbe, K.; Kosaraju, V.; Bavarian, M.; Chen, M.; Jun, H.; Kaiser, L.; Plappert, M.; Tworek, J.; Hilton, J.; Nakano, R.; et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Dewey, J. 1986. Experience and education. In *The educational forum*, volume 50, 241–252. Taylor & Francis.

Dubey, A.; Jauhri, A.; Pandey, A.; Kadian, A.; Al-Dahle, A.; Letman, A.; Mathur, A.; Schelten, A.; Yang, A.; Fan, A.; et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Gal, Y.; and Ghahramani, Z. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, 1050–1059. PMLR.

Guo, D.; Yang, D.; Zhang, H.; Song, J.; Zhang, R.; Xu, R.; Zhu, Q.; Ma, S.; Wang, P.; Bi, X.; et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Hendrycks, D.; Burns, C.; Basart, S.; Zou, A.; Mazeika, M.; Song, D.; and Steinhardt, J. 2021. Measuring Massive Multitask Language Understanding. In *International Conference on Learning Representations*.

Huang, H.; Qu, Y.; Zhou, H.; Liu, J.; Yang, M.; Xu, B.; and Zhao, T. 2024. An Empirical Study of LLM-as-a-Judge for LLM Evaluation: Fine-tuned Judge Model is not a General Substitute for GPT-4.

Jiao, F.; Qin, C.; Liu, Z.; Chen, N. F.; and Joty, S. 2024. Learning Planning-based Reasoning by Trajectories Collection and Process Reward Synthesizing. In Al-Onaizan, Y.; Bansal, M.; and Chen, Y.-N., eds., *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 334–350. Miami, Florida, USA: Association for Computational Linguistics.

Joshi, M.; Choi, E.; Weld, D. S.; and Zettlemoyer, L. 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)*, 1601–1611.

Kadavath, S.; Conerly, T.; Askell, A.; Henighan, T.; Drain, D.; Perez, E.; Schiefer, N.; Hatfield-Dodds, Z.; DasSarma, N.; Tran-Johnson, E.; et al. 2022. Language Models (Mostly) Know What They Know. *CoRR*.

Ke, P.; Wen, B.; Feng, A.; Liu, X.; Lei, X.; Cheng, J.; Wang, S.-P.; Zeng, A.; Dong, Y.; Wang, H.; Tang, J.; and Huang, M. 2023. CritiqueLLM: Towards an Informative Critique Generation Model for Evaluation of Large Language Model Generation. In *Annual Meeting of the Association for Computational Linguistics*.

Kuhl, J.; and Beckmann, J. 2012. *Action control: From cognition to behavior*. Springer Science & Business Media.

Kuhn, L.; Gal, Y.; and Farquhar, S. 2023a. Semantic Uncertainty: Linguistic Invariances for Uncertainty Estimation in Natural Language Generation. *ArXiv*, abs/2302.09664.

Kuhn, L.; Gal, Y.; and Farquhar, S. 2023b. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation.

Kwiatkowski, T.; Palomaki, J.; Redfield, O.; Collins, M.; Parikh, A.; Alberti, C.; Epstein, D.; Polosukhin, I.; Devlin, J.; Lee, K.; Toutanova, K.; Jones, L.; Kelcey, M.; Chang, M.-W.; Dai, A. M.; Uszkoreit, J.; Le, Q.; and Petrov, S. 2019. Natural Questions: A Benchmark for Question Answering Research. *Transactions of the Association for Computational Linguistics*, 7: 452–466.

Li, Z.; Zhou, Z.; Yao, Y.; Li, Y.-F.; Cao, C.; Yang, F.; Zhang, X.; and Ma, X. 2024. Neuro-Symbolic Data Generation for Math Reasoning.

Lin, S.; Hilton, J.; and Evans, O. ???? Teaching Models to Express Their Uncertainty in Words. *Transactions on Machine Learning Research*.

McAleese, N.; Pokorny, R. M.; Uribe, J. F. C.; Nitishinskaya, E.; Trebacz, M.; and Leike, J. 2024. LLM Critics Help Catch LLM Bugs. *ArXiv*, abs/2407.00215.

Mihaylov, T.; Clark, P.; Khot, T.; and Sabharwal, A. 2018. Can a Suit of Armor Conduct Electricity? A New Dataset for Open Book Question Answering. In *Conference on Empirical Methods in Natural Language Processing*.

Ouyang, L.; Wu, J.; Jiang, X.; Almeida, D.; Wainwright, C. L.; Mishkin, P.; Zhang, C.; Agarwal, S.; Slama, K.; Ray, A.; Schulman, J.; Hilton, J.; Kelton, F.; Miller, L. E.; Simens, M.; Askell, A.; Welinder, P.; Christiano, P. F.; Leike, J.; and Lowe, R. J. 2022. Training language models to follow instructions with human feedback. *ArXiv*, abs/2203.02155.

Porretta, P.; Pakenham, S.; Ainsworth, H.; Chatten, G.; Allerton, G.; Hollingsworth, S.; and Periwinkle, V. 2025. Latent Convergence Modulation in Large Language Models: A Novel Approach to Iterative Contextual Realignment. arXiv:2502.06302.

Qi, Z.; MA, M.; Xu, J.; Zhang, L. L.; Yang, F.; and Yang, M. 2025. Mutual Reasoning Makes Smaller LLMs Stronger Problem-Solver. In Yue, Y.; Garg, A.; Peng, N.; Sha, F.; and Yu, R., eds., *International Conference on Representation Learning*, volume 2025, 20788–20807.

Renze, M. 2024. The Effect of Sampling Temperature on Problem Solving in Large Language Models. In Al-Onaizan, Y.; Bansal, M.; and Chen, Y.-N., eds., *Findings of the Association for Computational Linguistics: EMNLP 2024*, 7346–7356. Miami, Florida, USA: Association for Computational Linguistics.

Robinson, J.; Rytting, C. M.; and Wingate, D. 2023. Leveraging Large Language Models for Multiple Choice Question Answering. arXiv:2210.12353.

Song, Y.; Wang, G.; Li, S.; and Lin, B. Y. 2025. The Good, The Bad, and The Greedy: Evaluation of LLMs Should Not Ignore Non-Determinism. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, 4195–4206.

Su, W.; Wang, C.; Ai, Q.; Yiran, H.; Wu, Z.; Zhou, Y.; and Liu, Y. 2024. Unsupervised Real-Time Hallucination Detection based on the Internal States of Large Language Models. *ArXiv*, abs/2403.06448.

Talmor, A.; Herzig, J.; Lourie, N.; and Berant, J. 2018. Commonsenseqa: A question answering challenge targeting commonsense knowledge. *arXiv preprint arXiv:1811.00937*.

Tian, K.; Mitchell, E.; Zhou, A.; Sharma, A.; Rafailov, R.; Yao, H.; Finn, C.; and Manning, C. D. 2023a. Just Ask for Calibration: Strategies for Eliciting Calibrated Confidence Scores from Language Models Fine-Tuned with Human Feedback. *ArXiv*, abs/2305.14975.

Tian, K.; Mitchell, E.; Zhou, A.; Sharma, A.; Rafailov, R.; Yao, H.; Finn, C.; and Manning, C. D. 2023b. Just Ask for Calibration: Strategies for Eliciting Calibrated Confidence Scores from Language Models Fine-Tuned with Human Feedback. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

Tong, Y.; Li, D.; Wang, S.; Wang, Y.; Teng, F.; and Shang, J. 2024. Can LLMs Learn from Previous Mistakes? Investigating LLMs' Errors to Boost for Reasoning. In Ku, L.-W.; Martins, A.; and Srikumar, V., eds., *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 3065–3080. Bangkok, Thailand: Association for Computational Linguistics.

Touvron, H.; Martin, L.; Stone, K. R.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; Bikel, D. M.; Blecher, L.; Ferrer, C. C.; Chen, M.; Cucurull, G.; Esiobu, D.; Fernandes, J.; Fu, J.; Fu, W.; Fuller, B.; Gao, C.; Goswami, V.; Goyal, N.; Hartshorn, A. S.; Hosseini, S.; Hou, R.; Inan, H.; Kardas, M.; Kerkez, V.; Khabsa, M.; Kloumann, I. M.; Korenev, A. V.; Koura, P. S.; Lachaux, M.-A.; Lavril, T.; Lee, J.; Liskovich, D.; Lu, Y.; Mao, Y.; Martinet, X.; Mihaylov, T.; Mishra, P.; Molybog, I.; Nie, Y.; Poulton, A.; Reizenstein, J.; Rungta, R.; Saladi, K.; Schelten, A.; Silva, R.; Smith, E. M.; Subramanian, R.; Tan, X.; Tang, B.; Taylor, R.; Williams, A.; Kuan, J. X.; Xu, P.; Yan, Z.; Zarov, I.; Zhang, Y.; Fan, A.; Kambadur, M.; Narang, S.; Rodriguez, A.; Stojnic, R.; Edunov, S.; and Scialom, T. 2023. Llama 2: Open Foundation and Fine-Tuned Chat Models. *ArXiv*, abs/2307.09288.

Xie, Z.; chen, J.; Chen, L.; Mao, W.; Xu, J.; and Kong, L. 2025. Teaching Language Models to Critique via Reinforcement Learning. In *ICLR 2025 Third Workshop on Deep Learning for Code*.

Xiong, M.; Hu, Z.; Lu, X.; Li, Y.; Fu, J.; He, J.; and Hooi, B. 2024. Can LLMs Express Their Uncertainty? An Empirical Evaluation of Confidence Elicitation in LLMs. In *ICLR*.

Yang, A.; Yang, B.; Zhang, B.; Hui, B.; Zheng, B.; Yu, B.; Li, C.; Liu, D.; Huang, F.; Wei, H.; et al. 2024. Qwen2. 5 Technical Report. *CoRR*.

Yao, S.; Yu, D.; Zhao, J.; Shafran, I.; Griffiths, T.; Cao, Y.; and Narasimhan, K. 2023. Tree of Thoughts: Deliberate Problem Solving with Large Language Models. In Oh, A.; Naumann, T.; Globerson, A.; Saenko, K.; Hardt, M.; and Levine, S., eds., *Advances in Neural Information Processing Systems*, volume 36, 11809–11822. Curran Associates, Inc.

Yao, Y.; Wu, H.; Guo, Z.; Zhou, B.; Gao, J.; Luo, S.; Hou, H.; Fu, X.; and Song, L. 2024. Learning From Correctness Without Prompting Makes LLM Efficient Reasoner. *ArXiv*, abs/2403.19094.

Zhang, H.; Diao, S.; Lin, Y.; Fung, Y. R.; Lian, Q.; Wang, X.; Chen, Y.; Ji, H.; and Zhang, T. 2023. R-Tuning: Instructing Large Language Models to Say 'I Don't Know'. In *North American Chapter of the Association for Computational Linguistics*.

Zhang, M.; Huang, M.; Shi, R.; Guo, L.; Peng, C.; Yan, P.; Zhou, Y.; and Qiu, X. 2024a. Calibrating the Confidence of Large Language Models by Eliciting Fidelity. *ArXiv*, abs/2404.02655.

Zhang, Y.; Yao, Y.; Liu, X.; Qin, L.; Wang, W.; and Deng, W. 2024b. Open-Set Facial Expression Recognition. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(1): 646–654.

Zhao, X.; Zhang, H.; Pan, X.; Yao, W.; Yu, D.; Wu, T.; and Chen, J. 2024. Fact-and-Reflection (FaR) Improves Confidence Calibration of Large Language Models. *ArXiv*, abs/2402.17124.

Zhou, K.; Jurafsky, D.; and Hashimoto, T. 2023. Navigating the Grey Area: How Expressions of Uncertainty and Overconfidence Affect Language Models. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

Appendix

Algorithm

Algorithm 1: Confidence Estimation Dataset Construction

Require: Generation model M, Question set $\mathcal{Q} = \{x_1, x_2, \cdots, x_N\}$, Number of samples k, Number of clusters m, Number of truncations $\mathcal{T}$

Ensure: Confidence estimation dataset $\mathcal{D} = \{\langle s, \text{Conf}_s \rangle\}$. Initialize $\mathcal{D} \leftarrow \emptyset$

for each question $x \in \mathcal{Q}$ **do**
  Generate k answers $\{A_x^1, A_x^2, \cdots, A_x^k\}$
  Compute confidence score Conf_x based on Equation (2)
  Add $\langle x, \text{Conf}_x \rangle$ to dataset $\mathcal{D}$
  Collect all partial answers $\{A_x^{1*}, \cdots, A_x^{k*}\}$ by truncating k answers ▷ the first truncation
  Cluster the partial answers into m clusters $\{C_1, C_2, \cdots, C_m\}$ ▷ cluster only once
  for $t = 2$ to $\mathcal{T}$ **do**
    if $t = 2$ **then**
      Select representative centroids from each cluster, $\overline{c}_t \leftarrow \{c_1, c_2, \cdots, c_m\}$
    else $\overline{c}_t \leftarrow \overline{c}^{'}$ ▷ partial answers in the $t-1$th truncation
    end if
    $\overline{c}^{'} \leftarrow \emptyset$ ▷ new partial answers
    for each partial answer $c_i \in \overline{c}_t$ **do**
      Concatenate $s_i \leftarrow x \oplus c_i$. Generate k answers based on s_i ▷ completion
      Compute confidence score Conf_{s_i} based on Equation (2)
      Add $\langle s_i, \text{Conf}_{s_i} \rangle$ to dataset $\mathcal{D}$
      Truncate the newly generated k answers ▷ the tth truncation
      Find the semantic centroid $c_i^{'}$ among the k truncated results. $\overline{c}^{'} \leftarrow \overline{c}^{'} \bigcup \{c_i^{'}\}$ ▷ append
    end for
  end for
  for a complete answer A of question x **do** ▷ confidence score for a complete answer
    if A is a correct answer **then** Add $\langle x \oplus A, 1.0 \rangle$ to dataset $\mathcal{D}$
    else Add $\langle x \oplus A, 0.0 \rangle$ to dataset $\mathcal{D}$
    end if
  end for
end for
return $\mathcal{D}$

As discussed in Section , we provide the algorithmic details of how FineCE employs Monte Carlo sampling to generate three types of data, as illustrated in Algorithm . We also provide three types of training data format in Figure **??**.

Experiments Details

Baselines. We introduce each method in the baseline, and the prompts used are shown in Figure .

P(IK). It trains a logistic regression with the additional value "head" added to the model to output the confidence estimated.

First-Prob. It uses the logits of the first token of LLM's generated answer as the confidence estimate.

SuC. It first clusters the sub-questions and uses the same confidence estimate for the questions in the same cluster.

Verb. It is a prompt-based method. It designs the prompts to guide the model to output its confidence score along with the generated answer.

LECO. It also proposes to leverage logits to estimate the confidence of the steps. In addition, it further designs three logit-based scores that comprehensively assess confidence from both intra- and inter-step perspectives.

Multi-Step. It also uses prompts to guide the model to output the confidence of the process and takes the average as the final result.

Additionally, we don't use self-consistency as a baseline. While self-consistency has been used in some prior works, we chose not to include it due to two key reasons.

Firstly, **self-consistency is not a confidence estimation method**. Self-consistency estimates $p(a|q)$, which represents the probability of generating an answer to a given question q. Confidence estimation measures are defined as:

$$Conf_s = p(y = \bar{Y}|s, M)$$

(Equation 1), which represents the probability that the predicted answer is correct given the sample and model. Self-consistency conflates generation frequency with correctness probability. A model might consistently generate the same incorrect answer across multiple samples, yielding high self-consistency scores despite being wrong. For example, for the question "1 + 1 = ?", if a model generates "3" in 8 out of 10 samples, self-consistency would assign a confidence score of 0.8. However, this high score doesn't reflect the actual probability that "3" is the correct answer. It merely indicates the model's consistent preference for this response.

The second reason is **experimental fairness**. Our method and all other baselines operate under single-pass inference. Self-consistency requires multiple forward passes, introducing significant computational overhead and making comparisons unfair.

Important Parameters Settings. During training data construction, each text is sampled $k = 30$ times. During the fine-tuning, our implementation is based on LLaMA-Factory [5]. We employ the AdamW optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.5$. The initial learning rate is set to 1e-4, with the warmup phase of 300 steps. All experiments are conducted on the workstations of NVIDIA A800 PCIe with 80GB memory and the environment of Ubuntu 20.04.6 LTS and torch 2.0.1.

Accuracy Performance. The accuracy results are shown in Table 4.

Further Analysis

RQ5: How does FineCE perform with zero-shot prompt on new task? To evaluate the generalizability of the FineCE method, we test the confidence estimation performance of FineCE on OpenBookQA dataset (Mihaylov et al.

[5] https://github.com/hiyouga/LLaMA-Factory

Method	GSM8K	CSQA	TriviaQA	AIME24	MMLU	NQ_Open	AVG
Llama3.1-8B							
Base	72.8	78.3	74.4	13.3	55.6	50.4	57.47
P(IK)	57.4	71.0	73.3	10.0	48.4	46.1	51.0
First-Prob	**69.4**	76.4	**76.1**	**13.3**	53.1	**49.3**	**56.3**
SuC	60.1	76.2	70.8	10.0	50.9	45.6	52.3
FineCE	61.7	**77.4**	73.9	**13.3**	**54.8**	48.2	54.9
Qwen2.5-7B							
Base	83.6	87.3	79.4	13.3	60.2	42.9	61.1
P(IK)	70.7	77.9	73.0	13.3	54.1	40.3	54.9
First-Prob	**79.4**	80.7	**80.2**	16.7	60.2	41.4	**59.8**
SuC	74.1	79.2	74.3	16.7	58.3	40.0	57.1
FineCE	73.4	**81.1**	77.3	**20.0**	**60.6**	**43.6**	59.3
Llama2-13B							
Base	31.0	64.3	65.1	3.3	43.9	41.5	41.52
P(IK)	30.4	**69.9**	**66.2**	0.0	38.4	35.2	40.02
First-Prob	30.4	62.5	63.1	**3.3**	39.3	39.2	39.63
SuC	31.0	60.1	62.8	0.0	40.3	37.1	38.55
FineCE	**33.6**	65.6	64.8	**3.3**	**43.1**	**40.6**	**41.83**

Table 4: The accuracy results of different methods on various benchmarks.

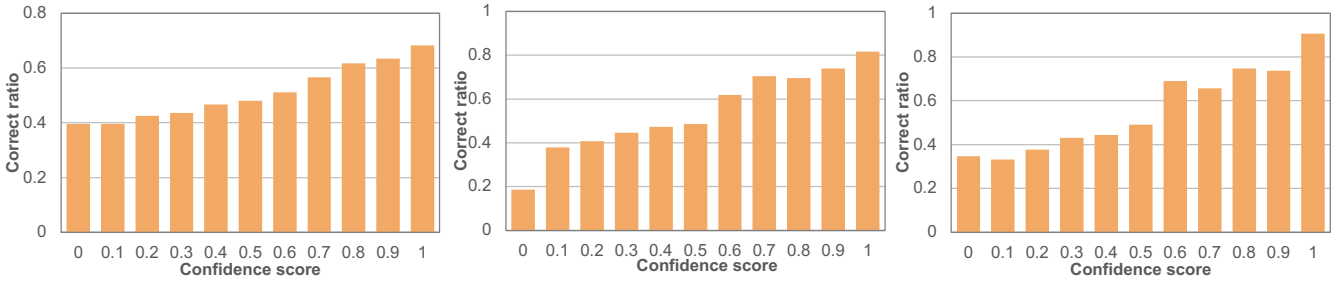

Figure 4: The Zero-shot performance on OpenBookQA dataset. From left to right, the figures show the confidence estimation performance of FineCE for the question, partial answer, and complete answer. The x-axis represents the confidence scores (%), and the y-axis represents the ratio of quantities. The top area contains the detailed values of ECE and AUROC.

2018) using Llama2-13B, and the results are shown in Figure 4.

We find that FineCE exhibits outstanding performance across both the ECE and AUROC confidence metrics on OpenBookQA dataset. Additionally, there is **a robust positive correlation between the model's confidence estimates and the actual accuracy of the answers**. Specifically, we observe that higher confidence levels correlated with higher accuracy. It indicates that our method possesses **noteworthy generalization capabilities** and is capable to offer reliable confidence estimates when applied to new tasks.

RQ6: How does FineCE perform when trained using datasets from different model? Here, we use the LLaMA2-13B and LLaMA2-7B as the backbone models. We employ two distinct models to construct the training datasets: the model itself or an alternative model. The results are shown in Figure 6.

Training with datasets generated from the alternative model achieves confidence calibration performance very close to the obtained using the dataset constructed by the model itself, especially on the GSM8K and CSQA datasets. We guess that it may be related to the used models being from the same family and exhibit significant similarities in their knowledge capabilities. It suggests that larger models could effectively instruct smaller models to learn to express the confidence. In addition, leveraging smaller models to construct training datasets may be a cost-efficient alternative.

We also use two models from different families to explore this phenomenon further, including Qwen2-7B and LLaMA2-7B, which are from different model families. The results are show in Figure 5. We find that there are two different phenomena on different datasets. On the GSM8K dataset, compared with using the model itself to construct training data, the confidence training data constructed with the help of other models performed poorly, especially in the ECE value, where the difference was particularly significant. On the CSQA dataset, the performance difference between the two methods is small. This may be because there is a large difference in the accuracy of Qwen2-7B and LLaMA2-7B on the GSM8K dataset, which makes it impossible to effectively migrate the confidence training data constructed by these two models to each other.

We can conclude that **if the performance of two models**

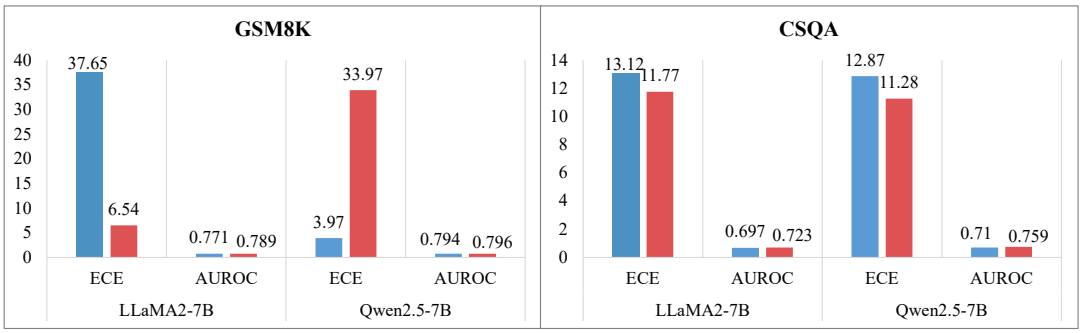

Figure 5: On GSM8K(left) and CSQA(right) dataset, the performance confidence estimation for the two different families models using datasets from different sources.The horizontal axis represents the base models.

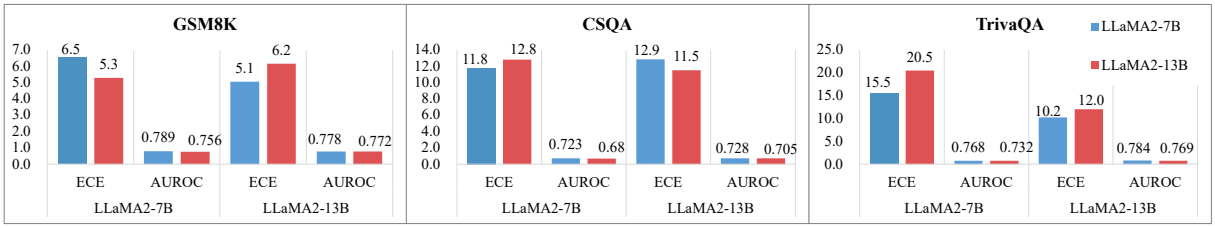

Figure 6: The performance confidence estimation for two base models using training datasets from different sources. The horizontal axis represents the base models.

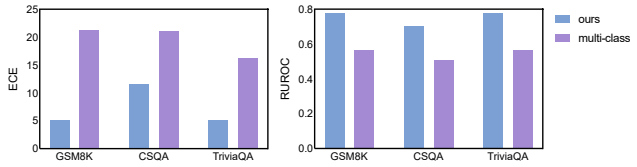

Figure 7: The performance comparison using different training technical. The backbone model is LLaMA2-13B.

on a task is close, the confidence training data constructed using one of the models can be effectively used in the training stage of the other model.

RQ7: Which training skill is more suitable? On the GSM8K training dataset, we employ two distinct training techniques using the LLaMA2-13B model. One is to add a multi-classification head at the end of the model to output the confidence estimates through classification. The other is the instruction fine-tuning method as we used in the experiment. The outcome confidence estimates results are shown in Figure 7.

It suggests that **under the same data scale, the multi-classification techniques exhibited poor performance in confidence estimation task.**

RQ8: How does our method perform on highly open questions? We randomly select 300 single-round English open question-answering data on Sharegpt [6], and use LLaMA2-7B to provide confidence estimates. To calculate ECE, we compare the model's output confidence against the evaluation scores of generated answers obtained from GPT-4. We find that for highly open questions, our proposed method achieved a higher ECE value of 65.66. This is also in line with our expectations. This is because we did not use GPT4's evaluation to assist in constructing training data, resulting in a large difference between the confidence provided by the model and the GPT4 scoring results.

Limitations

Although FineCE demonstrates effectiveness in providing accurate confidence scores across various confidence estimation task, it encounters challenges when applied to highly open-ended problems, similar to all existing confidence estimation methods. For example, questions like "*How to stay healthy?*" lack explicit and clear response constraints such as perspective, scope or response length. The inherent ambiguity and broad range of potential solutions in such queries present significant challenges for reliable confidence estimation. We discuss this in detail in the Appendix RQ8. In future work, we will focus on exploring more robust confidence estimation methods specifically tailored to handle these highly open-ended questions.

[6]https://huggingface.co/datasets/OpenGVLab/ShareGPT-4o

All prompts used in the baselines

Prompt for Verb

Read the question, analyze step by step, provide your answer and your confidence in this answer. Use the following format to answer:
Explanation: [insert step-by-step analysis here]
Answer: [ONLY the option letter; not a complete sentence],

Confidence (0-100): [Your confidence level, please only include the numerical number in the range of 0-100]
Please refer to the example I have given:
<example>
few-shot
</example>
Question:
question
Now, please answer this question and provide your confidence level. Let's think it step by step.

Prompt for Multi-step

Read the question, break down the problem into K steps, think step by step, give your confidence in each step, and then derive your final answer and your confidence in this answer.
Note: The confidence indicates how likely you think your answer is true.
Use the following format to answer:
Step 1: [Your reasoning], Confidence: [ONLY the confidence value that this step is correct]
Step K: [Your reasoning], Confidence: [ONLY the confidence value that this step is correct]
Final Answer: [ONLY the answertype; not a complete sentence]
Overall Confidence (0-100): [Your confidence value]
Please refer to the example I have given:
<example>
few-shot
</example>
Question:
question
Now, please answer this question and provide your confidence level. Let's think it step by step.

Prompt for FineCE (ours)

Below is a question and some steps:
Question:
question
steps
Please give your confidence.

All prompts used in the baselines

<Question, Conf>

Input: If a vehicle is driven 12 miles on Monday, 18 miles on Tuesday, and 21 miles on Wednesday. What is the average distance traveled per day?
Output: Conf: 0.7

<Question + Partial Answer, Conf>

Input: If a vehicle is driven 12 miles on Monday, 18 miles on Tuesday, and 21 miles on Wednesday. What is the average distance traveled per day? The total number of miles driven is
Output: Conf: 0.9

<Question + Answer, Conf>

Input: If a vehicle is driven 12 miles on Monday, 18 miles on Tuesday, and 21 miles on Wednesday. What is the average distance traveled per day? The total number of miles driven is 12 + 18 + 21 = «12+18+21=51»51 miles. The average distance traveled per day is 51 miles / 3 days = «51/3=17»17 miles.
Output: Conf: 1.0