Knowing How Certain It Is: Confidence Estimation Throughout LLM Generation

Jinyi Han†, Tingyun Li‡, Shisong Chen†, Xinyi Wang‡, Jiaqing Liang‡, Yanghua Xiao†§*
†Shanghai Institute of Artificial Intelligence for Education, East China Normal University
‡School of Data Science, Fudan University
§College of Computer Science and Artificial Intelligence, Fudan University

***Abstract*—Large language models (LLMs) have achieved strong performance across diverse tasks but often exhibit overconfidence, assigning high confidence to incorrect generations. Accurate confidence estimation is crucial for improving the trustworthiness and reliability of LLM outputs. However, existing methods provide only coarse-grained, single-point scores, which fail to capture uncertainty variations throughout the generation process. To address this limitation, we propose FineCE, a fine-grained confidence estimation method that provides accurate confidence scores during generation. Specifically, FineCE leverages Monte Carlo sampling estimate correctness probabilities, which serve as supervision for training a confidence prediction model applicable to arbitrary generated sequences. To further improve confidence accuracy for early-generaiton tokens, we introduce Backward Confidence Integration (BCI), a retrospective mechanism that incorporates information from subsequent generations to refine earlier confidence predictions. To make fine-grained confidence estimation computationally practical, we design three strategies for selecting optimal positions during generation. Extensive experiments across different backbone models show that FineCE consistently outperforms existing methods. Moreover, it shows strong effectiveness on downstream applications, including early-stage prediction of output correctness and confidence-based evaluation of generated answers. Our code and baselines are available on GitHub.**

***Index Terms*—Confidence Estimation, Large Language Models, Machine Learning**

I. Introduction

Self-awareness is a fundamental metacognitive capability that enables intelligent systems to reason about their own reliability, and has become increasingly critical with the rise of large-scale AI systems [1], [2]. For humans, it enables reflective thinking and error monitoring. For large language models (LLMs), self-awareness is essential for evaluating intermediate reasoning states, enabling self-correction and informed decision-making in complex, multi-step generation [3], [4]. Confidence estimation has emerged as a promising approach, enabling models to assess the reliability of their own generations [5]–[7].

However, existing confidence estimation methods for LLMs suffer from fundamental limitations that severely undermine their practical utility in real-world applications. These methods are inherently **coarse-grained** and **static**, providing only a narrow, retrospective view of model uncertainty rather than continuous, actionable signals throughout the generation process. Broadly, prior methods can be categorized by how and when confidence is expressed during the model's response process, as illustrated in Fig. 1. ***Question-oriented approaches*** avoid explicit confidence modeling entirely. They restrict LLMs to answering only questions within their "knowledge scope" [8]. Faced with ambiguous or challenging questions, models often refuse to engage [9] rather than attempting plausible reasoning. While this reduces incorrect responses, it provides **no actionable confidence signal** and severely limits LLM usability in open-ended scenarios. In contrast, ***post-hoc confidence estimation*** methods assign a confidence score only after the entire response has been generated [10], [11] . Such approaches rely on a single, outcome-level score to summarize model certainty. It ignores the critical uncertainty that arises during reasoning. A high final confidence does not guarantee the correctness of intermediate steps, and there's no way to identify when the model went wrong mid-generation [12].

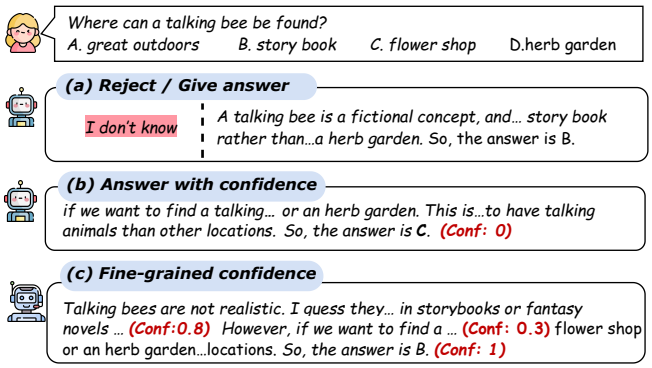

Fig. 1: **Comparison between FineCE and existing confidence estimation paradigms. (a):** LLMs provide no explicit confidence signals during generation, implicitly expressing uncertainty only through answer generation or refusal. **(b):** Post-hoc confidence estimation assigns a single confidence score after the entire response is generated, offering coarse-grained uncertainty estimation. **(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 during generation. Such confidence signals enable early prediction of whether a model is likely to produce a correct final answer, without waiting for the entire response. Beyond early prediction, intermediate confidence scores can serve as supervisory signals

*Corresponding author

for LLMs with deep thinking capabilities, such as O1 [13] and R1 [14]. These signals guide decision-making during generation, helping the model determine whether to proceed with the current reasoning trajectory or revise earlier outputs. Moreover, questions that consistently yield low confidence scores expose underlying weaknesses of the model, providing actionable insights for targeted improvements.

Despite these benefits, implementing fine-grained confidence estimation in LLMs is non-trivial and presents three major challenges. ***(i. Task Learning:)*** Expressing reliable confidence is not an inherent capability of LLMs [15]. Without explicit confidence annotations, it is unclear how to train models to produce fine-grained confidence estimates. Although task-specific supervision is necessary, constructing such data is non-trivial. Confidence labels distilled from other models do not accurately reflect the learner model's own uncertainty. Moreover, the free-form nature of text generation makes it difficult to assign confidence to arbitrary outputs. ***(ii. Effectiveness:)*** During auto-regressive generation, LLMs only observe the partially generated output and have no access to future tokens. As a result, confidence estimates derived from incomplete outputs are inherently biased and often poorly calibrated with respect to the correctness of the final answer. ***(iii. Efficiency:)*** Estimating confidence at every token is unnecessary and computationally costly. Instead, it is crucial to identify key positions during generation where confidence estimation provides the greatest value.

In this paper, we propose FineCE, a supervised fine-grained confidence estimation method for LLMs. Specifically, to capture the distributional uncertainty inherent in LLM generation, FineCE constructs supervision through a Monte Carlo–based data pipeline. By sampling diverse outputs under stochastic decoding, it estimates the probability that a given text sequence leads to a correct answer, and uses these estimates to train a model that predicts confidence for arbitrary sequences. To improve the alignment between predicted confidence and actual correctness, we further introduce Backward Confidence Integration (BCI) at inference stage. Inspired by retrospection, BCI refines confidence estimates for earlier text by incorporating uncertainty signals revealed by subsequent tokens, without altering the generation trajectory itself. In addition, to balance estimation accuracy with computational efficiency, we design three strategies for selecting optimal positions during generation at which confidence estimation is performed. Together, these components enable FineCE to provide reliable and continuous confidence signals with practical computational cost.

We evaluate FineCE on a diverse set of benchmark datasets across multiple widely used LLM backbones. Experimental results show that FineCE consistently produces accurate and well-calibrated confidence estimates throughout the generation process, outperforming existing methods across models, domains, and task difficulty levels. Beyond confidence quality, we further demonstrate the practical value of FineCE on downstream applications, including early-stage prediction of generation correctness and confidence-based filtering. In these settings, FineCE effectively reduces unnecessary computation while improving output reliability. Finally, we analyze different strategies for performing confidence estimation during generation, and find that paragraph-level estimation achieves a favorable balance between computational efficiency and estimation accuracy.

In summary, our contributions are four-fold:

- We propose FineCE, a fine-grained confidence estimation framework that provides accurate and continuous confidence signals throughout the generation process, overcoming the limitations of existing coarse-grained methods.
- We introduce a Monte Carlo–based data construction pipeline that captures the intrinsic distributional uncertainty of LLM generation and enables supervised learning of confidence for arbitrary text sequences.
- We develop Backward Confidence Integration (BCI), a novel inference mechanism that retrospectively refines confidence estimates by incorporating information from subsequently generated tokens.
- We design three practical estimation strategies that identify optimal positions during generation, achieving an effective trade-off between confidence accuracy and computational efficiency.

II. Task Formalization

The confidence estimation task aims to improve model calibration by aligning predicted confidence scores with the likelihood that a generated output is correct. In this work, we define ***confidence*** as the probability that a model produces a correct answer.

LLMs generate responses in an auto-regressive manner. Given an input x and a model M, the model produces an output sequence $y = \{t_1, t_2, \ldots, t_n\}$, where each token t_i is sampled from the conditional distribution $P_i = \mathcal{P}(\cdot \mid x, t_{<i}; M)$, with $t_{<i} = \{t_1, \ldots, t_{i-1}\}$ and n denoting the sequence length. Let $\bar{Y}$ denote the ground-truth output. For any intermediate generation state s, we define the confidence score as:

$$\text{Conf}_s = p(y = \bar{Y} \mid s, M). \tag{1}$$

The confidence score Conf_s represents the model's estimated probability of generating the correct output $\bar{Y}$, conditioned on the current generation state s, which may correspond to a partial or complete response. While this probability is not directly observable in practice, it serves as the target quantity that confidence estimation methods aim to approximate.

Depending on the form of s, confidence estimation can be categorized into three settings:

- **Question-oriented confidence estimation.** Here, s contains only the input question, i.e., $s = x$. The model estimates its likelihood of producing a correct answer before any tokens are generated.
- **Process-oriented confidence estimation.** In this setting, s consists of the input and a partially generated output, i.e., $s = (x, t_{<i})$. This formulation enables confidence estimation at intermediate steps during generation.
- **Outcome-oriented confidence estimation.** Here, s includes both the input and the complete generated response, i.e., $s = (x, y)$. The confidence score reflects the model's belief in the correctness of its final output.

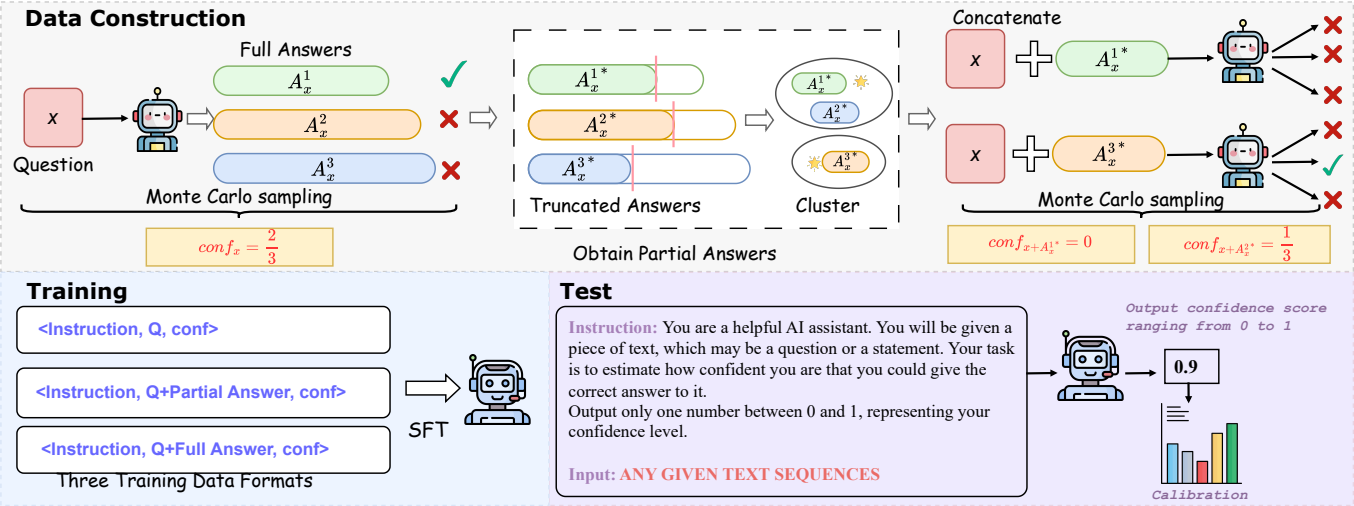

Fig. 2: **Overall framework of FineCE.** (1) Data Construction: confidence scores are generated via Monte Carlo sampling for three data formats: Question, Question with Partial Answer, and Question with Answer. For the first two, scores reflect uncertainty across multiple samples; for the last, scores are based on answer correctness; (2) Training: the constructed data are used to fine-tune the model to predict confidence scores for arbitrary text sequences; (3) Test: given any input text, FineCE outputs a confidence score between 0 and 1, indicating the estimated likelihood of correctness.

This unified formulation provides a common probabilistic view of existing confidence estimation paradigms and naturally extends them to support fine-grained confidence estimation throughout the generation process.

III. FineCE: Fine-grained Confidence Estimation

In this section, we first describe how FineCE constructs training data via Monte Carlo sampling and analyze the resulting computational complexity. We then present the training strategy adopted in FineCE. In addition, we introduce three alternative strategies for determining where confidence estimation is performed during generation. Finally, we provide a detailed description of the backward confidence estimation mechanism applied at inference time.

A. Training Data Construction

Preliminary. Traditional classification models misinterpret softmax outputs as confidence scores, which do not reliably reflect predictive uncertainty [16], [17]. To obtain a more accurate estimate of a model's belief in its predictions, we adopt Monte Carlo sampling [18]. Specifically, given a text input s, the generative LLM M produces k sampled answers $\{A_s^1, A_s^2, \ldots, A_s^k\}$ under a high-temperature setting. The proportion of correct samples provides an empirical estimate of the model's true confidence. By the Law of Large Numbers, this estimate converges to the true probability as k increases.

Overall Construction Pipeline. Our training data construction involves four main steps: (1) generate multiple sampled answers via Monte Carlo sampling, (2) truncate complete answers to form partial fragments, (3) group semantically similar fragments to share computation, and (4) estimate confidence scores for each input type. The full data construction pipeline is detailed in Algorithm 1. Each input sequence s can be one of three types: *Question*, *Question with Partial Answer*, or *Question with Answer*. For each s, we generate k sampled answers $\{A_s^1, \ldots, A_s^k\}$ and define the confidence score as:

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

where $\bar{y}_s$ is the ground-truth answer, and $\mathbf{I}$ is the indicator function returning 1 for correct matches and 0 otherwise.

Confidence score for *Question*. For each input question x, we generate k diverse complete answers $\{A_x^1, \ldots, A_x^k\}$ using high-temperature sampling. The confidence score is computed following Equation 2, representing the proportion of correct samples among the generations.

Confidence score for *Question with Partial Answer*. Partial answers contain informative intermediate reasoning steps. To estimate confidence for these fragments efficiently, we apply a progressive construction pipeline.

Each complete answer A_x^i is first truncated into multiple fragments. Each fragment is concatenated with the original question x and used to generate sampled completions. The fraction of correct completions provides an empirical confidence estimate for that fragment.

Directly generating multiple samples for every fragment is computationally expensive. To reduce cost, we leverage an intrinsic LLM property: fragments with similar prefixes tend to produce similar contextual representations and conditional distributions [19]. This allows confidence estimation to share computation across semantically related fragments.

We perform bottom-up hierarchical clustering on fragments to group them by semantic similarity. The distance between two fragments is measured as:

$$dis(i,j) = 1 - \frac{\boldsymbol{x}_i \cdot \boldsymbol{x}_j}{\|\boldsymbol{x}_i\|\|\boldsymbol{x}_j\|}, \tag{3}$$

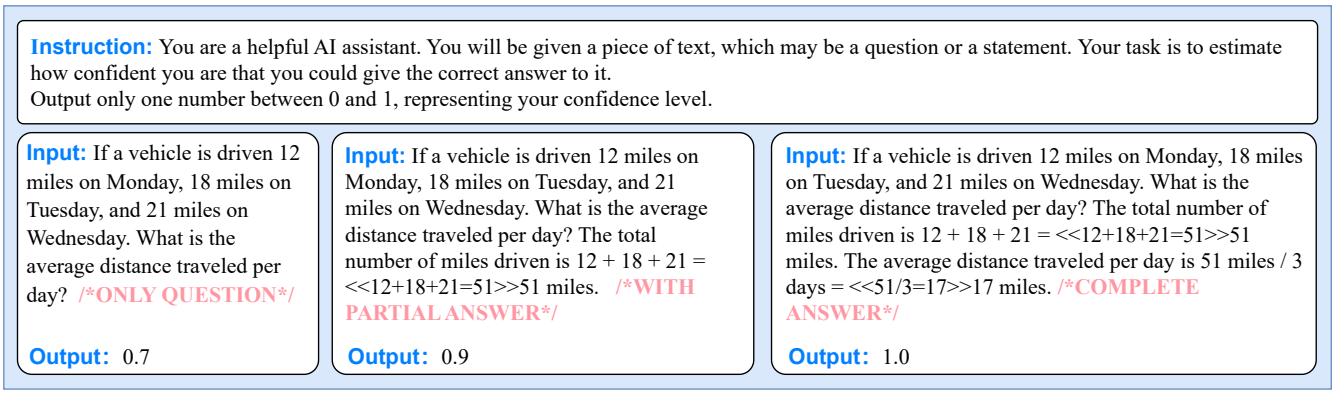

Fig. 3: **Three training data formats used in FineCE.** (1) *Question*: the model estimates its confidence without generating a response; (2) *Question with Partial Answer*: confidence is estimated based on an incomplete generation; (3) *Question with Answer*: confidence is determined from the correctness of the full answer.

where $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are embedding representations. Using average linkage, the distance between clusters C_a and C_b is:

$$D(C_a, C_b) = \frac{1}{|C_a||C_b|} \sum_{x_i \in C_a} \sum_{x_j \in C_b} dis(i, j). \tag{4}$$

Clusters are merged until inter-cluster distances exceed a threshold, producing m clusters ($1 \leq m \leq k$). We then select one representative fragment per cluster for Monte Carlo sampling. Iterating this procedure for up to $\mathcal{T}$ levels yields hierarchical partial answers aligned with the model's reasoning progression.

Confidence score for *Question with Answer*. For each complete answer generated for question x, the confidence score is set to 1 if it matches the ground truth, and 0 otherwise.

Complexity Analysis. Without optimization, generating confidence scores for all three input types requires a full k-ary tree of depth $\mathcal{T}+1$, totaling $\sum_{i=1}^{\mathcal{T}+1} k^i$ model inferences. By clustering semantically similar fragments and selecting representatives, the branching factor reduces to m, giving $k\sum_{i=0}^{\mathcal{T}} m^i$ inferences. Further, by selecting a representative candidate directly from the k answers at each step (from the 2nd to the $\mathcal{T}$-th level), the total cost reduces to $k(1+m\mathcal{T})$. Hence, our data preparation complexity is reduced from exponential to linear with respect to $\mathcal{T}$.

B. Training Technique

To train LLMs to accurately estimate confidence, we construct a hierarchical dataset containing answers at different stages of reasoning, from partial to complete solutions. Each data instance consists of an input (question and partial or complete answer) with a target confidence score. This setup provides fine-grained supervision, enabling the model to learn how confidence evolves throughout the generation process.

We explore two distinct strategies for training confidence estimation. The first, Additional Value Head, augments the base language model with an extra prediction head dedicated to confidence estimation. This formulation treats confidence prediction as a classification or regression task, allowing token-level confidence scores across the generated sequence, but requires modifying the model architecture.

The second, Instruction Fine-Tuning (IFT) [20], leverages the model's inherent ability to follow natural language instructions. Here, the model is prompted to produce a numerical confidence value (e.g., between 0 and 1) in a human-interpretable format, without adding new output heads. IFT predicts confidence for the entire sequence rather than for individual tokens. It does not require modifying the model architecture, can be applied to large-scale data easily, and produces confidence scores in a natural, human-readable way.

In this work, FineCE adopts the IFT paradigm, training on the constructed confidence dataset. The three training data formats used in FineCE are illustrated in Fig 3.

C. Identify the Calibration Position

While FineCE can generate confidence estimates for any token in a sequence, performing token-wise estimation is computationally expensive and often unnecessary. To balance efficiency and effectiveness, we propose three strategies to selectively estimate confidence, reducing computational overhead while capturing meaningful signals.

Paragraph-End Calibration estimates confidence at natural linguistic boundaries, such as paragraph ends. This approach preserves semantic coherence and allows estimation without significantly interrupting generation.

Periodic Calibration triggers confidence estimation at fixed token intervals (e.g., every 50 tokens). By providing a regular, interval-based mechanism, it ensures a consistent measure of confidence throughout the sequence.

Entropy-based Calibration performs estimation when the model's output entropy exceeds a predefined threshold. While entropy reflects uncertainty, it alone is insufficient for accurate confidence prediction. Calibration becomes more meaningful and reliable when entropy values are high.

D. Backward Confidence Integration (BCI)

Existing confidence estimation methods often rely on local signals, reflecting the model's certainty only at individual generation steps. This local view leads to overconfident predictions for partially correct reasoning paths or underconfident

predictions for valid but initially uncertain steps, resulting in confidence estimates that misalign with the overall correctness of the final output.

To address this limitation, we propose Backward Confidence Integration (BCI), which employs the backward evaluation principle. BCI recursively integrates confidence information from future steps into earlier positions. As a result, each step's confidence reflects both its local reliability and its contribution to the final correctness of the entire sequence. Intuitively, if later reasoning is consistent and the final answer is correct, earlier steps are retrospectively reinforced; conversely, if later reasoning reveals errors, earlier confidence is reduced. This backward integration produces globally consistent, context-aware confidence estimates that better align with the true quality of the generated sequence.

Formally, let $Conf_{s_j}$ denote the initial confidence at position s_j. The adjusted confidence $Conf'_{s_j}$ is computed as:

$$Conf'_{s_j} = \begin{cases} \alpha Conf_{s_j} + (1-\alpha)\frac{1}{w}\sum_{d=1}^{w} Conf'_{s^d_{j+1}}, \\ \qquad\qquad j < N \\ Conf_{s_j}, j = N \end{cases} \tag{5}$$

Here, $\alpha \in [0,1]$ is the revision coefficient that balances the original local confidence and the influence of future context. A smaller α increases the weight of future signals. The parameters w specifies the number of sampled paths for averaging (integration width), and d indicates how many future steps are considered (integration depth). N denotes the endpoint of integration. By recursively incorporating future signals, BCI corrects both excessively high and low local confidence. Consequently, the resulting estimates are more consistent across the sequence and better aligned with the correctness of the final output.

IV. Experiments

A. Experiment Setting

Backbones. We employ three widely-used open-source models to demonstrate the effectiveness of FineCE, including Llama2-13B-chat [21], Llama3.1-8B-Instruct [22] and Qwen2.5-7B-Insturct [23].

Dataset. We construct the training set by randomly sampling dataset from the training splits of *GSM8K* [24], *TriviaQA* [25], and *CommonsenseQA* (CSQA) [26]. For evaluation, we use the test splits of six datasets. Among them, *GSM8K*, *TriviaQA*, and *CSQA* serve as in-domain benchmarks, while *AIME24*[1] , *MMLU* [27], and *NQ-Open* [28] are used as out-of-domain datasets to assess generalization performance.

Metrics. We adopt widely used metrics for uncertainty estimation, including Expected Calibration Error (ECE), Area Under the Receiver Operating Characteristic Curve (AUROC) and Accuracy (ACC).

Baselines. We compare our method with several representative baselines, which can be grouped into three categories:

(i) **Question-oriented methods: P(IK)** [9], which trains a logistic regression model with an additional "head" to predict confidence scores from the model's internal representations;

Algorithm 1: Training Data 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, $\bar{c}_t \leftarrow \{c_1, c_2, \cdots, c_m\}$
    else $\bar{c}_t \leftarrow \bar{c}^{'}$ ▷ partial answers in the $t-1$th truncation
    end if
    $\bar{c}^{'} \leftarrow \emptyset$ ▷ new partial answers
    for each partial answer $c_i \in \bar{c}_t$ **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
      Find the semantic centroid $c_i^{'}$ among the k truncated results. $\bar{c}^{'} \leftarrow \bar{c}^{'} \bigcup \{c_i^{'}\}$ ▷ 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 $\mathcal{D}$

Self-consistency [29], which quantifies confidence based on the consistency of answers across multiple sampled answers.

(ii) **Post-hoc methods: First-Prob**, which takes the logit of the first generated token as the confidence estimate; **SuC** [30], which clusters sub-questions and assigns a shared confidence within each cluster; **Verbalized Prob** (Verb) [15], which prompts the model to verbalize its own confidence alongside the final answer; **Semantic Uncertainty** (SE) [31], which measures uncertainty based on semantic similarity across multiple model outputs; and **CoCoA** [32], it formulates uncertainty as expected decision risk under Minimum Bayes Risk decoding.

(iii) **Step-wise estimation methods: Multi-Step** (MS) [6], which elicits step-level confidence through prompting and averages them for the final estimate; and **LECO** [33], which leverages step-wise logits to derive several intra- and inter-step confidence metrics.

Important Parameters Settings. During training data construction, each text instance is sampled $k = 30$ times. During the fine-tuning, our implementation is based on LLaMA-Factory [2]. We employ the AdamW optimizer with $\beta_1 = 0.9$

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

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

	Position	Metrics	Llama2-13B-chat			Llama3.1-8B			Qwen2.5-7B-Instruct		
			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**
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**
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**
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**
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**
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**

TABLE I: **Confidence estimation throughout the generation process.** z denotes the total number of paragraphs in a generated answer, while $p(1)$ and $p(z-1)$ indicate the confidence estimates for the first and $(z-1)$-th paragraphs, respectively. †" marks in-domain tasks and ‡" marks out-of-domain tasks. The results show that FineCE predicts the likelihood of a correct answer using only partial outputs, and its confidence estimates are consistently accurate throughout the generation process.

and $\beta_2 = 0.5$. The initial learning rate is set to 1e-4, with the warmup phase of 300 steps.

During testing, we fix the parameters w and d to zero in all experiments. This allows FineCE to generate confidence estimates within a single inference step. We perform inference using the vllm framework with temperature set to 0 and top-p set to 0.95.

Platform. All experiments are conducted on workstations equipped with NVIDIA A800 PCIe GPUs (80 GB memory) running Ubuntu 20.04.6 LTS and PyTorch 2.0.1.

B. Main Results and Analysis

RQ1: How does FineCE perform compared with baselines? To comprehensively evaluate FineCE, we analyze its performance from four complementary perspectives.

(i) Performance of FineCE Across the Generation Process. FineCE effectively captures meaningful uncertainty signals during text generation, and the results in Table I. Specifically, FineCE achieves AUROC scores above 70% across all evaluated models and datasets, outperforming baselines by 10–15 percentage points. In contrast, most baseline methods remain between 57% and 65%, close to random performance.

Cross-model consistency. The performance gains of FineCE are observed across different model families, indicating that its effectiveness does not depend on specific architectural characteristics. This consistent improvement across model families suggests that FineCE captures model-agnostic uncertainty patterns rather than relying on architecture-specific behaviors.

Temporal stability. FineCE maintains stable uncertainty estimation across different generation stages, including early ($p(1)$) and late ($p(z-1)$) positions. By comparison, baseline methods experience noticeable performance degradation at later stages, suggesting that their confidence estimates are sensitive to local generation dynamics and fail to remain consistent over extended reasoning trajectories.

(ii) Performance of FineCE on Question-Oriented and Outcome-Oriented Tasks. FineCE consistently outperforms all baseline methods on both question-oriented and outcome-oriented confidence estimation tasks, as shown in Table II. It achieves higher discrimination and better calibration across datasets and model backbones.

On standard benchmarks like GSM8K, FineCE delivers clear gains in both metrics, indicating more accurate and well-calibrated confidence estimates than post-hoc baseline methods. What's particularly impressive is that these improvements are consistent across model architectures, from Llama2-13B-Chat to Qwen2.5-7B-Instruct.

Baselines	GSM8K†		CSQA†		TriviaQA†		AIME24‡		MMLU‡		NQ-Open‡	
	ECE↓	AUROC↑	ECE↓	AUROC↑	ECE↓	AUROC↑	ECE↓	AUROC↑	ECE↓	AUROC↑	ECE↓	AUROC↑
Llama3.1-8B-Instruct												
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
Self-consistency	4.4	76.8	14.2	73.3	33.0	56.4	35.3	51.9	16.6	72.4	53.4	56.6
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
CoCoA	27.3	68.5	31.5	49.0	34.2	47.05	34.6	-	35.8	39.8	24.1	49.5
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-Instruct												
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
Self-consistency	21.0	60.3	12.9	75.4	43.4	59.6	20.3	82.0	17.0	71.3	56.0	59.0
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
CoCoA	27.5	63.4	40.8	42.6	21.1	39.6	45.7	20.2	40.8	38.1	24.6	39.2
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-Chat												
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
Self-consistency	29.9	65.3	15.7	58.0	24.4	64.3	17.7	-	34.4	59.5	43.5	67.1
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
CoCoA	28.5	37.3	25.6	44.5	37.3	43.0	58.0	-	20.7	-	23.2	40.5
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 II: **Confidence estimation across baselines on *Question-oriented* and *Outcome-oriented* tasks.** †" indicates in-domain tasks and ‡" indicates out-of-domain tasks. Compared with methods that estimate confidence only after the question or after the full answer, FineCE consistently achieves superior calibration across all benchmarks.

(iii) Generalization Capability off FineCE. FineCE exhibits exceptional out-of-domain generalization. On Llama2-13B-Chat, it maintains stable performance when transferring from the in-domain GSM8K benchmark to out-of-domain tasks such as AIME24 and MMLU, while baseline methods suffer substantial degradation, particularly in calibration. This stability indicates that FineCE learns uncertainty signals that generalize beyond the training distribution, a crucial property for real-world applications where models encounter unseen task types.

Morever, FineCE generalizes across different types of confidence estimation tasks. On TriviaQA, it achieves comparable AUROC when evaluated under both process-oriented and outcome-oriented settings, indicating that its confidence estimation is not specialized to a single task formulation.

(iv) Preserving Generation Capability. A critical concern in confidence-aware fine-tuning is whether introducing a new training objective will degrade the model's original generation ability. Our experiments show that FineCE effectively preserves answer accuracy across all evaluated benchmarks, with performance remaining comparable to the base models. In some cases, such as AIME24 with Qwen2.5-7B, FineCE even yields modest accuracy improvements, suggesting that explicit confidence modeling can indirectly enhance reasoning capabilities by encouraging more cautious and deliberate generation.

While we observe slight accuracy declines on a few tasks, this is a reasonable and expected trade-off for the practical value of reliable confidence estimation. The introduction of a confidence prediction objective may slightly dilute the model's focus on the original task, but the magnitude of this effect is minimal.

C. Downstream Application

RQ2: How does FineCE perform on downstream applications? We evaluate FineCE's practical utility in real-world scenarios through early-stage confidence estimation and confidence-based filtering.

Early-stage confidence estimation. We first assess FineCE's ability to evaluate the correctness of the final answer at early stages of generation. Table IV reports calibration performance after the first generated paragraph (ECE_{p_1}) and across the entire generation process (ECE_{avg}) on Llama2-13B-chat.

FineCE provides reliable confidence estimates using only a fraction of the generated tokens. Early predictions are consistently close to the final calibration results, demonstrating

Models	Method	GSM8K†	CSQA†	TriviaQA†	AIME24‡	MMLU‡	NQ-Open‡	AVG
Llama3.1-8B-Instruct	Base	72.8	78.3	74.4	13.3	55.6	50.4	57.5
	P(IK)	57.4	71.0	73.3	10.0	48.4	46.1	51.0
	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
	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-Chat	Base	31.0	64.3	65.1	3.3	43.9	41.5	41.5
	P(IK)	30.4	**69.9**	**66.2**	0.0	38.4	35.2	40.0
	SuC	31.0	60.1	62.8	0.0	40.3	37.1	38.6
	FineCE	**33.6**	65.6	64.8	**3.3**	**43.1**	**40.6**	**41.8**

TABLE III: **Accuracy of different methods across benchmarks.** †" indicates in-domain tasks, and ‡" indicates out-of-domain tasks. FineCE preserves the original model's accuracy while providing reliable confidence estimation, showing minimal degradation compared with baseline models with additional training.

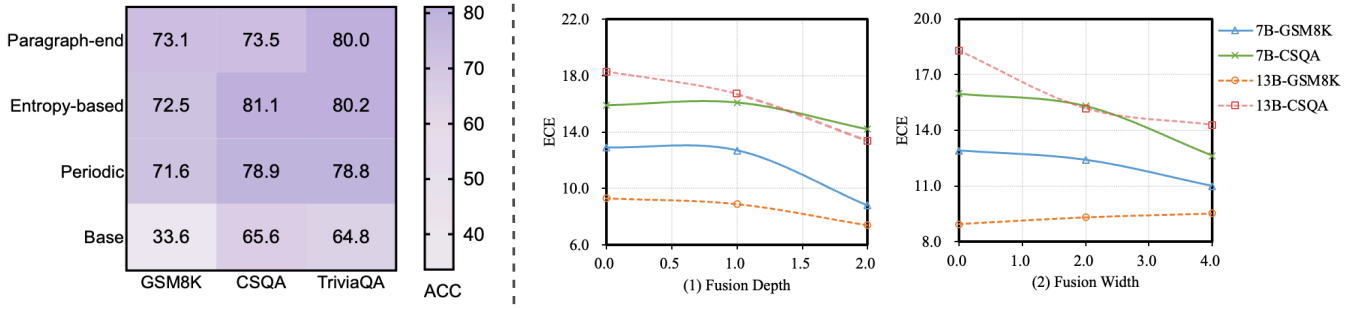

Fig. 4: **Left:** Accuracy comparison of the original model predictions with outputs selectively accepted by FineCE using a confidence threshold of 0.8 The backbone model is Llama2-13B-Chat. **Right:** Effects of fusion depth and width on FineCE's confidence estimation performance, evaluated on GSM8K and CSQA using Llama2-7B-Instruct and Llama2-13B-Chat.

that the model can assess correctness well before generation is complete. For example, under the paragraph strategy on GSM8K, the model achieves an ECE_{p_1} of 9.8% using only 30.4% of tokens, which is just 2.1 percentage points higher than the full-generation ECE_{avg} of 7.7%.

This capability offers two key benefits: (1) substantial computational savings, as low-confidence generations can be terminated early, avoiding unnecessary computation; and (2) faster interactive response, allowing the system to quickly identify uncertain outputs and trigger remedial actions such as user clarification or external knowledge retrieval.

Confidence-based filtering. We further investigate whether FineCE's predicted confidence scores can serve as reliable indicators for output selection. Using a predefined threshold δ, only outputs with confidence exceeding this value are retained. This strategy allows the system to automatically accept answers when it is sufficiently confident, while discarding uncertain responses. The results are shown in Fig 4 (Left).

Confidence-based filtering consistently improves accuracy across all evaluated datasets, indicating that high-confidence responses are more likely to be correct. This property is particularly useful in deployment scenarios, as it enables the system to reduce computation and enhance overall reliability. Moreover, we observe a clear precision–coverage trade-off: increasing the threshold δ yields higher accuracy among retained answers but reduces the total number of accepted responses. This demonstrates that FineCE allows flexible control between reliability and availability, enabling adaptive confidence-based filtering depending on application requirements.

D. Further Analysis

RQ3: Where does FineCE perform the confidence estimation? While FineCE is capable of providing continuous confidence estimates throughout the generation process, performing estimation after every token is computationally inefficient and often redundant. To identify optimal trade-offs between accuracy and efficiency, we investigate three calibration strategies on the Llama2-13B model: Paragraph-end Calibration, Periodic Calibration (every 30 tokens), and Entropy-based Calibration (triggered when token-level entropy exceeds $1e$-10). The comprehensive results are summarized in Table IV.

Superiority of Paragraph-end Calibration. Across all datasets, Paragraph-end Calibration consistently achieves the lowest or most competitive ECE scores while requiring significantly fewer estimation steps. This highlights that paragraph boundaries serve as natural semantic checkpoints, preserving sufficient reasoning context to produce stable and accurate

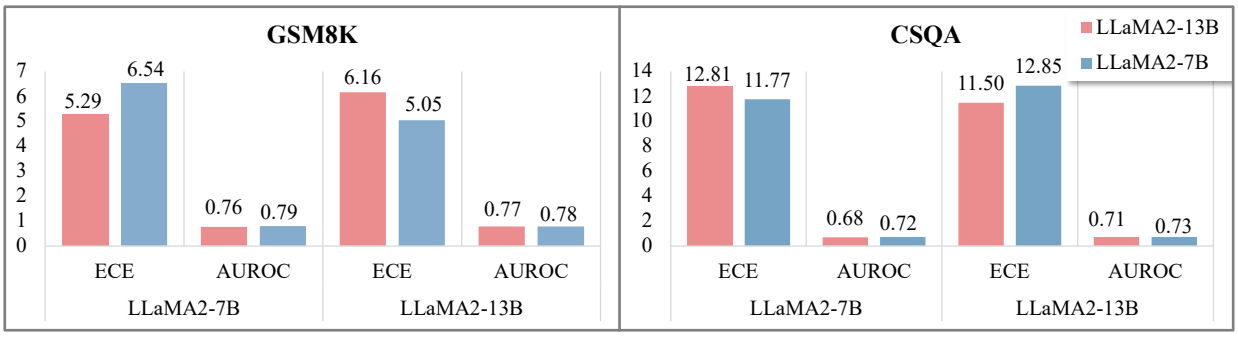

Fig. 5: Confidence estimation performance of two base models trained on datasets derived from models within the same family. The horizontal axis represents the backbone models.

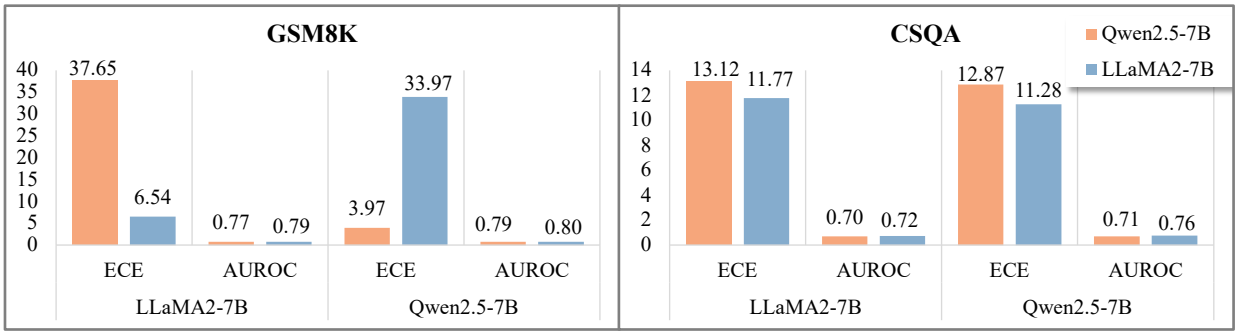

Fig. 6: **Confidence estimation performance of two model families trained on datasets from different sources.** The horizontal axis represents the base models.

confidence estimates. In contrast, Fixed-token Calibration operates on arbitrary, incomplete reasoning fragments, resulting in higher ECE without clear computational advantages.

Adaptive Efficiency of Entropy-based Calibration. Entropy-based Calibration demonstrates adaptive behavior by focusing estimation efforts on uncertain regions. It performs more frequent estimation in uncertain regions. This strategy dynamically responds to model uncertainty, performing more frequent estimation when the generation process becomes ambiguous.

These findings suggest a practical guideline for deploying FineCE. For general tasks, Paragraph-end Calibration offers the optimal balance of accuracy and efficiency. It eliminates unnecessary token-level computations while maintaining high calibration performance. For complex tasks, Entropy-based Calibration provides more precise confidence estimates by dynamically adapting to uncertainty patterns, though at a slightly higher computational cost.

RQ4: How effective is the BCI strategy? To evaluate the effectiveness of the proposed BCI strategy, we conduct ablation experiments on the GSM8K and CSQA datasets using both Llama2-7B[3] and Llama2-13B-chat models. Fig 4 (Right) reports the ECE results for $p(1)$, where $d = 0$ and $w = 0$ correspond to the FineCE baseline without BCI.

Consistent Calibration Improvements. BCI consistently enhances calibration performance across all model–dataset pairs. Increasing the fusion depth (d) from 0 to 2 leads to substantial ECE reductions. For example, on CSQA with Llama2-7B, ECE decreases from 15.3 to 12.6. Similarly, expanding the fusion width (w) from 0 to 4 yields further gains, with up to 15% lower ECE on CSQA. These findings confirm that deeper and wider integration enables more accurate confidence estimation by enriching contextual interactions within the model.

Model Size and Task Type Effects. The gains are more pronounced for larger models and complex reasoning tasks. Llama2-13B demonstrates significantly greater improvements from BCI compared to Llama2-7B, suggesting that larger models can better exploit bi-directional information flow to refine confidence estimates. Moreover, we observe task-specific sensitivity. For example, CSQA shows greater sensitivity to fusion width compared with GSM8K, implying that knowledge-intensive reasoning tasks require broader contextual integration, while mathematical problem-solving benefits more from deeper fusion layers.

RQ5: How does FineCE perform when trained using datasets from different model? Confidence is inherently model-dependent. This raises a practical question: when constructing training datasets for FineCE, how much does the model used to generate the data affect the calibration performance? Therefore, we evaluate FineCE under two settings: (i) training with datasets generated by models from the **same family**, and (ii) training with datasets generated by models from **different families**.

For the same-family setting, we use LLaMA2-13B and LLaMA2-7B as backbone, the results are shown in Fig 5. FineCE achieves nearly identical calibration performance in both cases, particularly on GSM8K and CSQA. This indicates

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

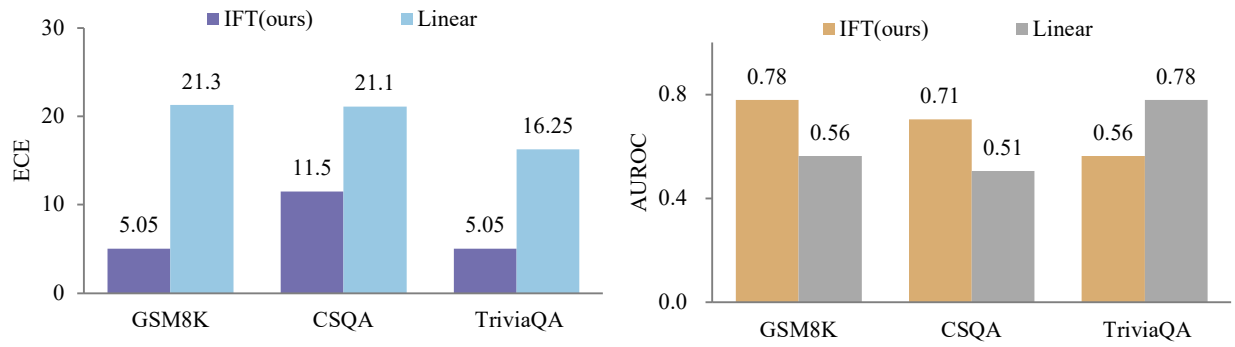

Fig. 7: **Performance comparison of LLaMA2-13B under different training techniques.** The backbone model is LLaMA2-13B.

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 IV: **Comparison of three strategies for selecting calibration points.** *Token Ratio* indicates the proportion of tokens generated before calibration relative to the total tokens in the full answer. The backbone model is Llama2-13B.

that confidence signals are transferable across models within the same architecture family, due to their similar knowledge structures and reasoning styles. Consequently, LLMs can effectively teach smaller ones to express confidence, while smaller models offer a cost-efficient alternative for data construction.

For the cross-family setting, as shown in Fig 6. We find that training FineCE with cross-family data leads to a clear degradation in calibration performance, especially in ECE, indicating that the learned confidence signals are not easily transferable across architectures. This degradation is likely due to the large discrepancy in task accuracy between Qwen2-7B and LLaMA2-7B on GSM8K, which causes a mismatch between the distributions of confidence and correctness labels. In contrast, on CSQA, the difference between the two methods is smaller, suggesting that tasks relying more on general knowledge rather than precise stepwise reasoning are less sensitive to cross-model differences.

In summary, these results reveal that FineCE's training data are transferable across models of the same family, but such transfer becomes unreliable when model architectures or task performances diverge significantly. When two models demonstrate comparable performance on a task, the confidence training data generated by one can be effectively reused to train the other, providing a flexible and efficient strategy for scaling FineCE across models.

RQ6: Which training method is more suitable? To identify an effective training paradigm for confidence estimation, we compare two approaches using LLaMA2-13B as the backbone. The first approach is linear regression, which attaches a lightweight linear head to predict a continuous confidence score between 0 and 1. The second approach is instruction fine-tuning (IFT). Our proposed method formulates confidence estimation as a natural language understanding task and leverages carefully designed instructions to guide the model to estimate the confidence. The results are shown in Figure 7.

The IFT consistently outperforms the linear regression method across all benchmarks. On GSM8K, IFT achieves an ECE of 5.05 and AUROC of 0.78, compared to the linear model's 21.3 and 0.56. Similar trends appear on CSQA and TriviaQA.

The inferior performance of the linear regression method likely stems from its limited ability to capture nuanced reasoning patterns. In contrast, IFT leverages the model's inherent natural language reasoning capability, enabling it to internalize confidence estimation as part of a broader meta-cognitive process.

V. 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 evaluates the quality of a response in a model-agnostic manner, producing consistent scores regardless of which model generated the answer [34], [35]. In contrast, the calibration model estimates the probability that a specific output is correct with respect to a particular model, capturing model-specific confidence. Since different models may generate different outputs for the same input, calibration is inherently model-dependent [36], [37].

Prior studies have explored using reward models to evaluate intermediate reasoning steps [38] or final answers [39], often aiming to rank candidate solutions, select the best one, or provide step-wise supervision [40]. While effective for improving task-specific performance, these approaches typically produce discrete scores, focus on specific tasks such as mathematical reasoning, and rarely assess the reliability of the evaluation itself.

In summary, verifiers enable standardized, model-agnostic evaluation of output quality, while calibration models capture model-specific epistemic uncertainty, reflecting how confident a model is in its own answers. This distinction clarifies the complementary roles of the two approaches and motivates the need for reliable confidence estimation.

Confidence Expression in LLMs. Confidence estimation techniques provide a complementary approach by directly quantifying the reliability of model outputs. Early methods leverage a model's internal parameters or structural information to assess its capability to address specific questions [41], [42]. For instance, [43] calculate eigenvalues from matrices derived from multiple internal output vectors to detect potential errors.

Another line of work guides LLMs to express confidence explicitly through prompts. Carefully designed prompts encourage models to verbalize their confidence alongside generated answers [5], [6], [44], [45]. [7] demonstrated GPT-3's ability to convey uncertainty on basic questions with few-shot prompts, while [30] formalized the concept of "verbalized confidence." Similarly, [46] employed external annotations to instruct LLMs to express uncertainty during generation. However, studies have shown that LLMs often exhibit overconfidence when prompted [6] and struggle to reliably follow complex instructions.

Beyond verbalization, several approaches exploit intrinsic signals of LLMs. Metrics such as token-level entropy or specialized uncertainty scores [47] and confidence tokens learned during fine-tuning [48] provide quantitative measures of uncertainty. Dynasor [49] leverages semantic entropy [50] as a confidence signal, while others probe logits of specific tokens to assess uncertainty over the entire output sequence [51]. [9] further explored a "Value Head" to probe self-awareness, though its applicability is limited to structured datasets like multiple-choice tasks.

Overall, existing methods predominantly rely on the model's inherent capabilities or internal signals to express confidence, and they tend to focus on tasks with standardized or easily measurable outputs. In this work, we treat confidence expression as a meta-capability that requires explicit training, aiming to generalize across diverse tasks and output formats.

VI. Conclusion

In this paper, we propose FineCE, a fine-grained confidence estimation framework that delivers accurate and continuous confidence scores throughout the generation process. We design a comprehensive training data construction pipeline and introduce three practical strategies for selecting optimal positions to perform confidence estimation. During inference, we further enhance prediction accuracy with BCI, a retrospective mechanism that leverages information from subsequently generated tokens to refine the confidence of the current output.

We evaluate FineCE using three widely adopted LLM backbones across six benchmark datasets, and the results demonstrate that it consistently outperforms existing baseline methods in terms of AUROC and ECE, while exhibiting strong generalization across domains, model architectures, and task difficulty levels. In downstream applications, FineCE proves highly effective: in early-stage confidence prediction, it accurately estimates the likelihood of correct generation after only one-third of tokens are produced, significantly reducing token consumption, and in confidence-based filtering, retaining only high-confidence responses leads to a 39.5% improvement in accuracy on GSM8K. Moreover, we find that evaluating confidence at the paragraph level provides an efficient balance between computational cost and estimation accuracy. Besides, with limited training data, IFT effectively leverages semantic understanding to improve confidence estimation, and training datasets from models of similar performance can be shared to reduce data construction costs. Overall, these results demonstrate that FineCE is a robust, generalizable, and practically useful framework for confidence estimation in LLMs, enabling more trustworthy and efficient deployment of LLMs in real-world applications.

References

[1] J. Dewey, "Experience and education," in *The educational forum*, vol. 50, no. 3. Taylor & Francis, 1986, pp. 241–252.

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

[3] Y. Tong, D. Li, S. Wang, Y. Wang, F. Teng, and J. Shang, "Can LLMs learn from previous mistakes? investigating LLMs' errors to boost for reasoning," in *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, L.-W. Ku, A. Martins, and V. Srikumar, Eds. Bangkok, Thailand: Association for Computational Linguistics, Aug. 2024, pp. 3065–3080. [Online]. Available: https://aclanthology.org/2024.acl-long.169/

[4] Z. Xie, J. chen, L. Chen, W. Mao, J. Xu, and L. Kong, "Teaching language models to critique via reinforcement learning," in *ICLR 2025 Third Workshop on Deep Learning for Code*, 2025. [Online]. Available: https://openreview.net/forum?id=CUEq6ZPSp7

[5] K. Zhou, D. Jurafsky, and T. Hashimoto, "Navigating the grey area: How expressions of uncertainty and overconfidence affect language models," in *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2023.

[6] M. Xiong, Z. Hu, X. Lu, Y. Li, J. Fu, J. He, and B. Hooi, "Can llms express their uncertainty? an empirical evaluation of confidence elicitation in llms," in *ICLR*, 2024.

[7] G. Branwen, "Gpt-3 nonfiction- calibration," The institution that published, Tech. Rep., 2020, last accessed on 2022-04-24. [Online]. Available: https://gwern.net/gpt-3-nonfiction#calibration

[8] H. Zhang, S. Diao, Y. Lin, Y. R. Fung, Q. Lian, X. Wang, Y. Chen, H. Ji, and T. Zhang, "R-tuning: Instructing large language models to say 'i don't know'," in *North American Chapter of the Association for Computational Linguistics*, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:265220839

[9] S. Kadavath, T. Conerly, A. Askell, T. Henighan, D. Drain, E. Perez, N. Schiefer, Z. Hatfield-Dodds, N. DasSarma, E. Tran-Johnson *et al.*, "Language models (mostly) know what they know," *CoRR*, 2022.

[10] M. Zhang, M. Huang, R. Shi, L. Guo, C. Peng, P. Yan, Y. Zhou, and X. Qiu, "Calibrating the confidence of large language models by eliciting fidelity," *ArXiv*, vol. abs/2404.02655, 2024. [Online]. Available: https://api.semanticscholar.org/CorpusID:268876453

[11] Y. Abbasi-Yadkori, I. Kuzborskij, A. György, and C. Szepesv'ari, "To believe or not to believe your llm," *ArXiv*, vol. abs/2406.02543, 2024. [Online]. Available: https://api.semanticscholar.org/CorpusID:270226258

[12] F. Jiao, C. Qin, Z. Liu, N. F. Chen, and S. Joty, "Learning planning-based reasoning by trajectories collection and process reward synthesizing," in *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, Y. Al-Onaizan, M. Bansal, and Y.-N. Chen, Eds. Miami, Florida, USA: Association for Computational Linguistics, Nov. 2024, pp. 334–350. [Online]. Available: https://aclanthology.org/2024.emnlp-main.20

[13] A. Jaech, A. Kalai, A. Lerer, A. Richardson, A. El-Kishky, A. Low, A. Helyar, A. Madry, A. Beutel, A. Carney *et al.*, "Openai o1 system card," *arXiv preprint arXiv:2412.16720*, 2024.

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

[15] K. Tian, E. Mitchell, A. Zhou, A. Sharma, R. Rafailov, H. Yao, C. Finn, and C. D. Manning, "Just ask for calibration: Strategies for eliciting calibrated confidence scores from language models fine-tuned with human feedback," *ArXiv*, vol. abs/2305.14975, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:258865733

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

[17] M. I. Zulfiqar, A. Khalid, L. Chen, and S. Dhelim, "Uncertainty-aware neighbor calibration for positive and unlabeled learning in large machine learning models," *IEEE Transactions on Emerging Topics in Computational Intelligence*, pp. 1–14, 2025.

[18] Z. Li, Z. Zhou, Y. Yao, Y.-F. Li, C. Cao, F. Yang, X. Zhang, and X. Ma, "Neuro-symbolic data generation for math reasoning," 2024. [Online]. Available: https://api.semanticscholar.org/CorpusID:274581565

[19] P. Porretta, S. Pakenham, H. Ainsworth, G. Chatten, G. Allerton, S. Hollingsworth, and V. Periwinkle, "Latent convergence modulation in large language models: A novel approach to iterative contextual realignment," 2025. [Online]. Available: https://arxiv.org/abs/2502.06302

[20] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. L. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, J. Schulman, J. Hilton, F. Kelton, L. E. Miller, M. Simens, A. Askell, P. Welinder, P. F. Christiano, J. Leike, and R. J. Lowe, "Training language models to follow instructions with human feedback," *ArXiv*, vol. abs/2203.02155, 2022. [Online]. Available: https://api.semanticscholar.org/CorpusID:246426909

[21] H. Touvron, L. Martin, K. R. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, D. M. Bikel, L. Blecher, C. C. Ferrer, M. Chen, G. Cucurull, D. Esiobu, J. Fernandes, J. Fu, W. Fu, B. Fuller, C. Gao, V. Goswami, N. Goyal, A. S. Hartshorn, S. Hosseini, R. Hou, H. Inan, M. Kardas, V. Kerkez, M. Khabsa, I. M. Kloumann, A. V. Korenev, P. S. Koura, M.-A. Lachaux, T. Lavril, J. Lee, D. Liskovich, Y. Lu, Y. Mao, X. Martinet, T. Mihaylov, P. Mishra, I. Molybog, Y. Nie, A. Poulton, J. Reizenstein, R. Rungta, K. Saladi, A. Schelten, R. Silva, E. M. Smith, R. Subramanian, X. Tan, B. Tang, R. Taylor, A. Williams, J. X. Kuan, P. Xu, Z. Yan, I. Zarov, Y. Zhang, A. Fan, M. Kambadur, S. Narang, A. Rodriguez, R. Stojnic, S. Edunov, and T. Scialom, "Llama 2: Open foundation and fine-tuned chat models," *ArXiv*, vol. abs/2307.09288, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:259950998

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

[23] A. Yang, B. Yang, B. Zhang, B. Hui, B. Zheng, B. Yu, C. Li, D. Liu, F. Huang, H. Wei *et al.*, "Qwen2. 5 technical report," *CoRR*, 2024.

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

[25] M. Joshi, E. Choi, D. S. Weld, and L. Zettlemoyer, "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)*, 2017, pp. 1601–1611.

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

[27] D. Hendrycks, C. Burns, S. Basart, A. Zou, M. Mazeika, D. Song, and J. Steinhardt, "Measuring massive multitask language understanding," in *International Conference on Learning Representations*, 2021. [Online]. Available: https://openreview.net/forum?id=d7KBjmI3GmQ

[28] T. Kwiatkowski, J. Palomaki, O. Redfield, M. Collins, A. Parikh, C. Alberti, D. Epstein, I. Polosukhin, J. Devlin, K. Lee, K. Toutanova, L. Jones, M. Kelcey, M.-W. Chang, A. M. Dai, J. Uszkoreit, Q. Le, and S. Petrov, "Natural questions: A benchmark for question answering research," *Transactions of the Association for Computational Linguistics*, vol. 7, pp. 452–466, 2019. [Online]. Available: https://aclanthology.org/Q19-1026/

[29] X. Wang, J. Wei, D. Schuurmans, Q. V. Le, E. H. Chi, S. Narang, A. Chowdhery, and D. Zhou, "Self-consistency improves chain of thought reasoning in language models," in *Proceedings of the International Conference on Learning Representations (ICLR) 2023*, 2023.

[30] S. Lin, J. Hilton, and O. Evans, "Teaching models to express their uncertainty in words," *Transactions on Machine Learning Research*.

[31] L. Kuhn, Y. Gal, and S. Farquhar, "Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation," 2023.

[32] R. Vashurin, M. Goloburda, A. Ilina, A. Rubashevskii, P. Nakov, A. Shelmanov, and M. Panov, "Cocoa: A minimum bayes risk framework bridging confidence and consistency for uncertainty quantification in llms," in *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

[33] Y. Yao, H. Wu, Z. Guo, B. Zhou, J. Gao, S. Luo, H. Hou, X. Fu, and L. Song, "Learning from correctness without prompting makes llm efficient reasoner," *ArXiv*, vol. abs/2403.19094, 2024. [Online]. Available: https://api.semanticscholar.org/CorpusID:268733095

[34] N. McAleese, R. M. Pokorny, J. F. C. Uribe, E. Nitishinskaya, M. Trebacz, and J. Leike, "Llm critics help catch llm bugs," *ArXiv*, vol. abs/2407.00215, 2024. [Online]. Available: https://api.semanticscholar.org/CorpusID:270844127

[35] H. Huang, Y. Qu, H. Zhou, J. Liu, M. Yang, B. Xu, and T. Zhao, "An empirical study of llm-as-a-judge for llm evaluation: Fine-tuned judge model is not a general substitute for gpt-4," 2024. [Online]. Available: https://api.semanticscholar.org/CorpusID:268247854

[36] B. Atil, A. Chittams, L. Fu, F. Ture, L. Xu, and B. Baldwin, "Llm stability: A detailed analysis with some surprises," *arXiv preprint arXiv:2408.04667*, 2024.

[37] M. Renze, "The effect of sampling temperature on problem solving in large language models," in *Findings of the Association for Computational Linguistics: EMNLP 2024*, Y. Al-Onaizan, M. Bansal, and Y.-N. Chen, Eds. Miami, Florida, USA: Association for Computational Linguistics, Nov. 2024, pp. 7346–7356. [Online]. Available: https://aclanthology.org/2024.findings-emnlp.432

[38] H. Lightman, V. Kosaraju, Y. Burda, H. Edwards, B. Baker, T. Lee, J. Leike, J. Schulman, I. Sutskever, and K. Cobbe, "Let's verify step by step," *CoRR*, 2023.

[39] K. Cobbe, V. Kosaraju, M. Bavarian, M. Chen, H. Jun, L. Kaiser, M. Plappert, J. Tworek, J. Hilton, R. Nakano, C. Hesse, and J. Schulman, "Training verifiers to solve math word problems," *ArXiv*, vol. abs/2110.14168, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:239998651

[40] X. Lai, Z. Tian, Y. Chen, S. Yang, X. Peng, and J. Jia, "Step-dpo: Step-wise preference optimization for long-chain reasoning of llms," *ArXiv*, vol. abs/2406.18629, 2024. [Online]. Available: https://api.semanticscholar.org/CorpusID:270764693

[41] W. Su, C. Wang, Q. Ai, H. Yiran, Z. Wu, Y. Zhou, and Y. Liu, "Unsupervised real-time hallucination detection based on the internal states of large language models," *ArXiv*, vol. abs/2403.06448, 2024. [Online]. Available: https://api.semanticscholar.org/CorpusID:268351309

[42] A. Azaria and T. Mitchell, "The internal state of an llm knows when it's lying," in *In Findings of the Association for Computational Linguistics: EMNLP*, 2023.

[43] C. Chen, K. Liu, Z. Chen, Y. Gu, Y. Wu, M. Tao, Z. Fu, and J. Ye, "Inside: Llms' internal states retain the power of hallucination detection," *CoRR*, 2024.

[44] M. Li and H. Nian, "Perturbation amplitudes design method based on confidence interval evaluation for impedance measurement," *IEEE Transactions on Industrial Electronics*, vol. 71, no. 10, pp. 12 323–12 337, 2024.

[45] Y. Zhang, Y. Yao, X. Liu, L. Qin, W. Wang, and W. Deng, "Open-set facial expression recognition," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 1, pp. 646–654, Mar. 2024. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/27821

[46] K. Tian, E. Mitchell, A. Zhou, A. Sharma, R. Rafailov, H. Yao, C. Finn, and C. D. Manning, "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*, 2023.

[47] E. Fadeeva, A. Rubashevskii, A. Shelmanov, S. Petrakov, H. Li, H. Mubarak, E. Tsymbalov, G. Kuzmin, A. Panchenko, T. Baldwin *et al.*, "Fact-checking the output of large language models via token-level uncertainty quantification," *arXiv preprint arXiv:2403.04696*, 2024.

[48] Y.-N. Chuang, P. K. Sarma, P. Gopalan, J. Boccio, S. Bolouki, X. Hu, and H. Zhou, "Learning to route llms with confidence tokens," *arXiv preprint arXiv:2410.13284*, 2024.

[49] Y. Fu, J. Chen, S. Zhu, Z. Fu, Z. Dai, Y. Zhuang, Y. Ma, A. Qiao, T. Rosing, I. Stoica *et al.*, "Efficiently scaling llm reasoning with certaindex," *arXiv preprint arXiv:2412.20993*, 2024.

[50] Y. Fu, J. Chen, Y. Zhuang, Z. Fu, I. Stoica, and H. Zhang, "Reasoning without self-doubt: More efficient chain-of-thought through certainty probing," in *ICLR 2025 Workshop on Foundation Models in the Wild*, 2025.

[51] J. Robinson, C. M. Rytting, and D. Wingate, "Leveraging large language models for multiple choice question answering," 2023. [Online]. Available: https://arxiv.org/abs/2210.12353