UNCERTAINTY LOCALIZATION IN LLM REASONING VIA EMBEDDING PERTURBATIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Large Language Models (LLMs) have achieved significant breakthroughs across various domains, but they can still produce unreliable or misleading outputs. For responsible LLM applications, uncertainty quantification techniques are used to estimate a model's uncertainty about its outputs, indicating the likelihood that those outputs may be problematic. For LLM reasoning tasks, it is essential to estimate uncertainty not only in the final answer but also in the intermediate reasoning process, particularly to identify where uncertainty arises. Such information may enable more fine-grained and targeted interventions during inference. In this study, we investigate which metrics can effectively localize uncertain places within an LLM reasoning trajectory. Our study reveals that uncertain intermediate continuations are more likely to occur at tokens that are highly sensitive to perturbations in the embeddings of preceding tokens. In our experiments, we show that such perturbation-based metrics achieve stronger performance in localizing uncertain intermediate steps than baseline methods, including probability-based, sampling-based, and Bayesian-based approaches. Meanwhile, our proposed metrics also enjoy good simplicity and efficiency.

1 INTRODUCTION

Large language models (LLMs) have demonstrated strong reasoning capabilities across a wide range of tasks (Wei et al., 2022), yet their reasoning processes can still contain flaws. Uncertainty quantification (UQ) (Kuhn et al., 2023; Xiong et al., 2023) provides a useful signal for predicting when model outputs are unreliable and require additional scrutiny. However, most existing UQ methods focus on LLM's uncertainty of the overall response or the final answer. It leaves a more fine-grained question largely unresolved: once a response is considered uncertain, **at which places in the reasoning trajectory does the uncertainty arise?** For example, an LLM can be confident in early steps but become much less certain at a specific intermediate step (Zhang et al., 2026; Sun et al., 2026), possibly making a mistake that affects the final outcome. In practice, these locations are the parts that are in more need of scrutiny. People can either manually inspect these places or invoke tools (Schick et al., 2023) to improve the reasoning quality and reliability.

To find unreliable places in an LLM reasoning response, one may consider employing external step-wise verifiers, such as process reward models (PRMs) (Lightman et al., 2024), to assess the correctness of each step. However, such verifiers are often unavailable in many open-domain reasoning tasks (Pandit et al., 2026). Thus, eliciting the model's inherent uncertainty remains important. On the other hand, it is potential to use the LLM's internal statistics, including token probability or entropy (Li et al., 2026; Wang et al., 2026). They typically assume tokens with lower (generation) probability or higher entropy are less confident. Nevertheless, these metrics may not precisely reflect the model's reasoning uncertainty. For instance, a model may hesitate among multiple synonyms or different reasoning pathways. All these alternatives can be logically plausible, but may result in a relatively low probability for the selected token (see more discussion in Section 3.1).

In our work, we treat the LLM reasoning process as a series of conditional probability predictions, i.e., to predict the probability for each (generated) token given preceding contents $\Pr(x_t|x_{<t})$. Our work draws a key observation: for most tokens x_t, if we minimally perturb their preceding tokens $x_{<t}$ (in the embedding space), the probability prediction for x_t remains stable. Meanwhile, under the perturbed condition, regenerating from the position of x_t can still lead to coherent and on-task

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

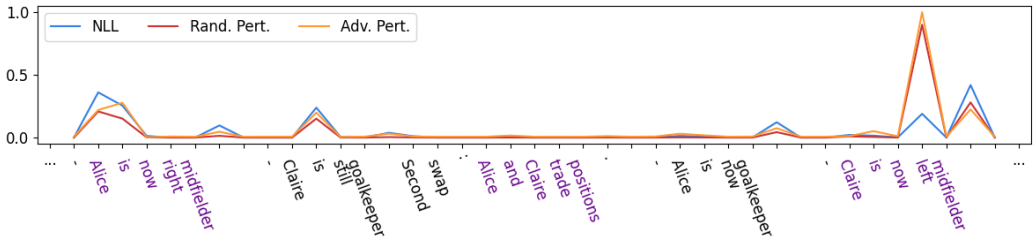


Figure 1: A response of Llama in logical reasoning (Suzgun et al., 2023). The model first states that 'Alice is now right midfielder' and 'Alice and Claire trade positions', it then incorrectly infers 'Claire is now left midfielder.' While, the correct inference should be 'Claire is now right midfielder'. Our perturbation-based metrics flag the token 'left' with high uncertainty score. Note: the scores are min-max normalized in the figure.

continuations, indicating the small perturbation does not disrupt the reasoning process. However, there are also tokens that are more sensitive to such perturbations, e.g., the token 'left' in Figure 1. Regenerating from such place still yields valid continuations, but the original generation can be less likely to be reproduced under certain perturbed conditions, which reflects potential uncertainty of the model. Based on this finding, we propose two perturbation methods (random and adversarial) and measure the sensitivity of each token. We flag the highly sensitive tokens as places where uncertainty may arise. Notably, since LLMs operate in an autoregressive manner, our scores can be efficiently obtained by simultaneously perturbing embeddings of all tokens in the entire response.

In our experiments, we show the proposed metrics can effectively localize uncertain reasoning steps along an LLM trajectory. In Section 4.1, we first consider responses containing reasoning errors and show that the positions flagged by our metrics frequently coincide with the steps where the model goes wrong. In Section 4.2, we turn to the correct responses, where uncertainty can still be present. We find our method can identify "fragile correct steps", which are correct in the observed trajectory but can become incorrect under natural regeneration. In Section 4.3, we further demonstrate one practical application of our signals: one can identify a small subset of potentially problematic reasoning steps in the whole response, enabling more targeted inspection and intervention to improve the model generation. Together, these studies show that our method provides richer and more actionable information about reasoning uncertainty.

2 Related Works

2.1 Uncertainty in LLM Reasoning

In LLM reasoning tasks, a major line of work studies the reasoning reliability at the response level (Yadkori et al., 2024; Kuhn et al., 2023). While, relatively fewer works study how to find uncertain regions in the generation process. Some of them focus on detecting hallucinated claims in open-form factual generation (Fadeeva et al., 2024; Sriramanan et al., 2024), but the underlying mechanisms of hallucination may differ from uncertainty in LLM reasoning.

In our considered problem in LLM reasoning, two prior studies are relevant. (1) The work (Bigelow et al., 2025) introduces the notion of "forking tokens," which are the tokens that the reasoning process can branch into different outcomes. However, this method requires repeated re-generation from individual token, which is prohibitively expensive. (2) TokUR (Zhang et al., 2026) adopts a Bayesian perspective and estimates token-level uncertainty by perturbing model weights to approximate Bayesian neural-network inference. In contrast, our approach perturbs token embeddings rather than model parameters, which is simpler to implement and more computationally efficient. We also conduct extensive comparisons of our method with TokUR in our experiments.

Uncertainty Localization vs. Quantification: Our work focuses on identifying where uncertainty arises along an LLM reasoning trajectory. That is, given an LLM response, **our goal is to localize the steps that are relatively more uncertain**. It differs from more broadly studied uncertainty quantification problems, which is to assign a calibrated risk estimate for each prediction. Admittedly, step-wise uncertainty quantification can be challenging in reasoning tasks because different steps come from different questions and reasoning trajectories. Moreover, later steps may be confidently generated based on earlier mistakes, while still being incorrect.

Difference with Step-wise Verifiers: Our study is also closely related to step-level verification models (Lightman et al., 2024), which are external models to assess the correctness of each reasoning step. For many open-domain reasoning tasks, such verifiers are not available (Pandit et al., 2026), which highlights the importance of the model's inherent uncertainty. Furthermore, because our scores can identify potentially problematic steps in LLM reasoning, they may also facilitate human annotation, and thereby facilitate the training of these step-wise verifier models.

2.2 Perturbation Methods in General Machine Learning (ML)

Perturbation-based methods have been widely studied in general ML, especially in adversarial attacks and robustness (Goodfellow et al., 2014). For LLM reasoning, such perturbation-based probing may be particularly useful. Unlike many traditional ML settings, a low token probability in LLM generation does not necessarily indicate model uncertainty, because it may also arise from multiple plausible continuations. Therefore, perturbation-based methods provide a complementary way to capture prediction stability beyond probability-based scores. Indeed, TokUR (Zhang et al., 2026), which perturbs model weights, is a relatively strong baseline in our experiments, while our methods further improve uncertainty localization. These results together suggest that perturbation may provide a particularly informative signal for reasoning uncertainty.

3 Methodology

In this section, we first present the necessary definitions and notations. Then, we present details of our proposed perturbation-based method to localize uncertain places in LLM reasoning.

3.1 Preliminary and Definition

We consider a given LLM and query it to perform a reasoning task. For an input query $\mathbf{q} = (x_1, ..., x_m)$, e.g., a math reasoning question with m tokens, the LLM generates a response with n tokens and we denote it as $\mathbf{r} = (x_{m+1}, x_{m+2}, ...x_{m+n})$. For some metrics introduced later, we will extract the token embeddings (before Transformer layers) of all tokens including both input tokens and output tokens. It will result in $(m+n)$ embedding vectors defined as $\mathbf{H} = (\mathbf{h}_1, ..., \mathbf{h}_{m+n})$, where each $\mathbf{h}_i \in \mathbb{R}^d$ and d is the dimension of token embedding space. Based on them, we give the definition of LLM's computation on the (generation) probability of each next token as below.

Definition 3.1 (**Token (Generation) Probability**). Given a sequence of tokens $(x_1, ..., x_{t-1})$ with $t > m$, an auto-regressive LLM calculates the probability of each potential next token v in vocabulary $\mathcal{V}$ as $\Pr(v \mid x_1, x_2, \ldots, x_{t-1})$ for $v \in \mathcal{V}$. For the actually sampled token x_t in the response, we denote the token probability of x_t as:

$$\mathrm{P}(x_t) = \Pr(x_t \mid x_1, x_2, \ldots, x_{t-1}) \tag{1}$$

It is also frequently formed in Negative Log Likelihood (NLL) as: $\mathrm{NLL}(x_t) = -\log \mathrm{P}(x_t)$. Intuitively, token probability indicates how strongly the LLM believes the next token is x_t, given the preceding tokens $x_1, ..., x_{t-1}$. Besides, one related metric is token entropy, which measures the dispersion of the next-token's probability distribution over the vocabulary $\mathcal{V}$.

Definition 3.2 (**Token Entropy**). The entropy when generating token x_t $(m < t \leq m + n)$ is:

$$\mathrm{Entropy}(x_t) = -\sum_{v \in \mathcal{V}} \Big(\Pr(v \mid x_1, x_2, \ldots, x_{t-1}) \cdot \log \Pr(v \mid x_1, x_2, \ldots, x_{t-1}) \Big) \tag{2}$$

In general, token probability and entropy may reflect a model's confidence when generating each token x_t (Tian et al., 2023). However, they are insufficient to reflect the model's reasoning uncertainty, as they can be confounded by "benign variability" (Kuhn et al., 2023). Figure 2 shows an illustration in logical reasoning from the BIG-Bench Hard dataset (Suzgun et al., 2023). In Part (A), at token position 1, the model generates the token "At" with a low probability of 0.252. However, this low probability may simply arise because other plausible alternatives are also available, such as "After" and "Now" with probability 0.222 and 0.173 respectively. Refer to Part (A), if we regenerate from this position, it can lead to these alternative choices for continuation but conveying plausible and even similar meaning. This suggests the model's reasoning uncertainty is not likely to arise at the positions like 1.

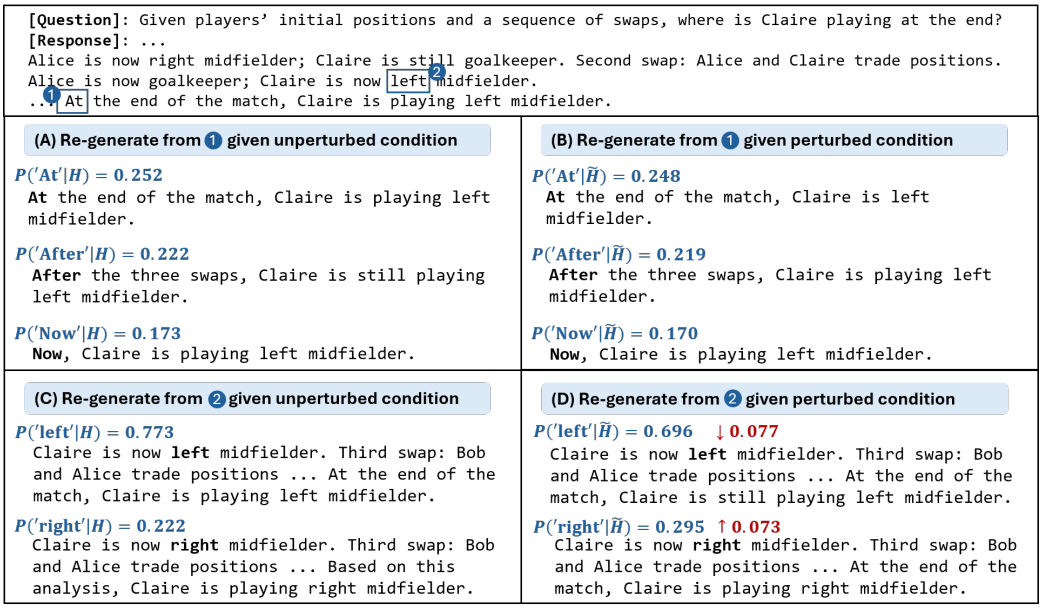


Figure 2: The model's alternative tokens at key positions, and the regenerated continuations for the example in Figure 1. We provide additional positions & examples in Figure 7 in Appendix E.

3.2 Random Perturbation

To alleviate the aforementioned challenge, we treat the LLM token log-probability (Eq.1) as a function of previous token embeddings $\mathbf{H}_{1:t-1}$, and we define:

$$f_t(\mathbf{H}_{1:t-1}) = \log \Pr\left(x_t \mid \mathbf{H}_{1:t-1}\right). \tag{3}$$

Then, we measure this function's sensitivity to minor perturbations on $\mathbf{H}_{1:t-1}$. In detail, given a complete response, we apply i.i.d Gaussian noises $\boldsymbol{\varepsilon} = (\boldsymbol{\epsilon}_1, \boldsymbol{\epsilon}_2, ..., \boldsymbol{\epsilon}_{m+n})$, with distribution $\mathcal{N}(0, \sigma^2 I)$ and dimension $(m+n) \times d$, onto all token embeddings including both the query $\mathbf{q}$ and the whole response $\mathbf{r}$. We define the perturbed embeddings as $\tilde{\mathbf{H}} = \mathbf{H} + \boldsymbol{\varepsilon}$. For each generated token x_t, we measure the variance of token probability under l times of perturbation on $\mathbf{H}$:

$$\text{Pert.}(x_t) = \text{Var}_{\boldsymbol{\varepsilon}}\left[\log \Pr(x_t \mid \tilde{\mathbf{H}}_{1:t-1})\right]. \tag{4}$$

Since LLMs are autoregressive, the probability for each generated token x_t only depends on its preceding contents. It means we only need to sample $\boldsymbol{\varepsilon}$ to perturb the entire sequence once, instead of performing separate perturbations for each token. In Figure 2, we show two representative positions, 1 and 2 with low and high sensitivity scores respectively, together with their regenerated continuations. This illustration reveals two key properties as introduced below.

Embedding perturbation does not disrupt reasoning coherence. In Figure 2, for a considered position such as 1 with x_t="At", we regenerate continuations from the perturbed prefix $\tilde{\mathbf{H}}_{1:t-1}$. We find that the resulting continuations remain coherent and on-task. This is true even when regenerating from position 2, which has a high sensitivity score. Thus, for the original x_t and its subsequent continuation, we can treat the perturbed prefix $\tilde{\mathbf{H}}_{1:t-1}$ as a valid neighboring condition. This is also why we perturb in the embedding space, as it may have a relatively limited impact on the validity of the resulting conditions (Zhu et al., 2019). In Figure 7 in Appendix E, we provide more regeneration outcomes at different positions.

Less-Supportive Neighboring Condition. Given these valid neighboring conditions, most tokens in the response have low $\text{Pert.}(x_t)$, including token "At" at position 1. It suggests that they remain stably supported across these perturbed conditions. Differently, if $\text{Pert.}(x_t)$ is high (token "left" at position 2), there can exist a $\tilde{\mathbf{H}}_{1:t-1}$ under which the probability of x_t drops by a non-negligible margin, as shown in Part (D). It means the original generation of x_t is not stably supported by neighboring conditions, and may have a lower probability of being sampled under certain $\tilde{\mathbf{H}}_{1:t-1}$.

Formally, we refer to $\tilde{\mathbf{H}}_{1:t-1}$ which reduces the probability of x_t as a "less-supportive neighboring condition" for x_t. The following proposition shows that a higher $\mathrm{Pert.}(x_t)$ implies a higher probability of encountering a less-supportive neighboring condition.

Proposition 3.3. *We define $\tilde{\mathbf{H}} = \mathbf{H} + \boldsymbol{\varepsilon}$, where $\boldsymbol{\varepsilon} \sim \mathcal{N}(0, \sigma^2 I)$. Under first-order approximation and a sufficiently small σ^2, for any $c > 0$, we have:*

$$\Pr_{\boldsymbol{\varepsilon}}\left[f_t(\tilde{\mathbf{H}}_{1:t-1}) \le f_t(\mathbf{H}_{1:t-1}) - c\right] \approx \Phi\left(-\frac{c}{\sqrt{\mathrm{Pert.}(x_t)}}\right).$$

A higher $\mathrm{Pert.}(x_t)$ indicates a higher probability that $f_t(\tilde{\mathbf{H}}_{1:t-1}) - f_t(\mathbf{H}_{1:t-1}) \le -c$, implying such $\tilde{\mathbf{H}}_{1:t-1}$ is a less-supportive neighboring condition for x_t. The proof is in Appendix A.

Besides, another interesting observation is that high sensitivity to perturbations indicates that the prediction of x_t can be strongly influenced by the preceding context. In reasoning trajectories, such tokens may be more likely to be reasoning-relevant tokens, carrying information derived from earlier reasoning steps rather than serving as function words such as "At" at 1.

3.3 Adversarial Perturbation

We also consider adversarial perturbation strategy (Goodfellow et al., 2014). In detail, we first calculate the response-level probability, as the sum of log probabilities of all generated tokens:

$$\mathcal{F}(\mathbf{H}) = \sum_{i=m+1}^{m+n} \log \Pr\left(x_i \mid \mathbf{H}_{1:i-1}\right). \tag{5}$$

We perturb all embeddings $\mathbf{H}$ along the direction that decreases the total probability: $\hat{\mathbf{H}} = \mathbf{H} - \alpha\,\mathrm{sgn}(\nabla_{\mathbf{H}}\mathcal{F}(\mathbf{H}))$, where α is a small constant. Here, we choose the sign operation to keep the perturbation on each token embedding the same magnitude. Then, we aim to find the generated tokens that become less supported under such a perturbation:

$$\mathrm{Pert.}(x_t) = \log \Pr(x_t \mid \mathbf{H}_{1:t-1}) - \log \Pr(x_t \mid \hat{\mathbf{H}}_{1:t-1}). \tag{6}$$

Under this formulation, only one backward pass and one forward pass are needed to obtain the scores for all positions. Compared to random perturbation, this metric directly explores a less-supportive neighboring condition $\hat{\mathbf{H}}_{1:t-1}$ that reduces the probability of x_t. Moreover, the following proposition shows that $\hat{\mathbf{H}}_{1:t-1}$ also less supports the overall response.

Proposition 3.4. *For a small constant α, a first-order Taylor expansion of Eq.6 gives:*

$$\mathrm{Pert.}(x_t) \approx \alpha \cdot \langle \nabla_{\mathbf{H}_{1:t-1}} \log \Pr(x_t \mid \mathbf{H}_{1:t-1}),\ \mathrm{sgn}(\nabla_{\mathbf{H}_{1:t-1}} \mathcal{F}(\mathbf{H})) \rangle \tag{7}$$

We can see that a high sensitivity score indicates strong alignment between the gradients of the token-level and response-level probabilities, both with respect to $\mathbf{H}_{1:t-1}$. In other words, a direction that decreases $\Pr(x_t \mid \mathbf{H}_{1:t-1})$ also tends to decrease the probability of the overall reasoning trajectory. This suggests that the identified highly sensitive tokens are not only locally unstable, but their instability is also closely related to the overall reasoning process. Similar to Figure 2, we provide the regeneration outcomes in Figure 6 in Appendix for adversarial perturbation.

4 Experiment

In our experiment, **we first evaluate the localization ability of our method in scenarios that the reasoning uncertainty exists. Then, we discuss how it can assist practical LLM reasoning.** In detail, we first assess: Given that a model contains unreliable steps probably due to uncertainty, whether our method can indicate the location of them?[1]Answering this question might not be straightforward as the model uncertainty itself is hidden. Thus, we rely on observable reasoning risks as proxy ground-truth for uncertain steps. We consider two types of *proxy uncertain steps* and we examine whether our method can successfully locate them in the whole reasoning trajectory:

[1]We consider natural reasoning responses rather than procedural ones with predefined step boundaries. We segment each response into sentences and treat each sentence as an individual reasoning step. More details is in Appendix C

Table 1: Performance of each method in 300 responses with incorrect steps, The 95% conf. interval is ± 0.057 for 300 Bernoulli trials with probability around 0.5.

	Math						**Logic**					
	Llama		*Qwen*		*Mistral*		*Llama*		*Qwen*		*Mistral*	
	10%	20%	10%	20%	10%	20%	10%	20%	10%	20%	10%	20%
NLL-max	0.19	0.30	0.21	0.38	0.16	0.28	0.21	0.41	0.27	0.50	0.15	0.32
Entropy-max	0.20	0.34	0.23	0.41	0.14	0.24	0.18	0.39	0.25	0.50	0.13	0.30
TokUR-max	0.27	0.48	0.26	0.45	0.28	0.44	0.27	0.42	0.29	0.47	0.25	0.45
NLL-avg	0.15	0.25	0.16	0.32	0.10	0.19	0.18	0.39	0.20	0.39	0.10	0.26
Entropy-avg	0.15	0.27	0.15	0.35	0.10	0.19	0.19	0.41	0.19	0.39	0.12	0.27
TokUR-avg	0.21	0.35	0.19	0.35	0.23	0.42	0.22	0.40	0.23	0.42	0.24	0.39
CCP	0.20	0.35	0.31	0.50	0.21	0.34	0.27	0.44	0.30	0.54	0.20	0.40
TokenSAR	0.20	0.29	0.26	0.42	0.19	0.31	0.29	0.47	0.30	0.54	0.22	0.41
Graph Uncertain.	0.21	0.28	0.20	0.32	0.15	0.21	0.23	0.38	0.31	0.44	0.16	0.30
Rand. Pert.	0.40	0.54	0.33	0.50	0.31	**0.50**	0.33	0.50	0.38	0.55	0.28	0.46
Adv. Pert.	**0.45**	**0.59**	**0.35**	**0.52**	**0.34**	**0.50**	**0.42**	**0.58**	**0.40**	**0.57**	**0.35**	**0.50**

- **First Wrong Step**: Many existing studies assume if the model is uncertain, it is likely to make errors (Jiang et al., 2021; Kadavath et al., 2022). Following prior works, we also use incorrectness as the proxy ground-truth for uncertainty. Notably, a reasoning trajectory can have multiple wrong steps, and later wrong steps might be confidently derived from earlier mistakes. Therefore, we only focus on the "first wrong step" as *proxy uncertain step*, which are the places the model first goes wrong from correct contents (Section 4.1).
- **Fragile Correct Step**: While a reasoning step is correct, uncertainty can still exist (Bigelow et al., 2025). As a complement to Section 4.1, we therefore define the second type of *proxy uncertain steps* as: the correct steps in the realized trajectory which are prone to be incorrect upon (natural) re-generation. (Section 4.2).

In practice, one can first use response-level UQ to assess the overall reliability of a response (Kuhn et al., 2023). Once a potentially unreliable response is identified, our method can then localize the uncertain positions within its reasoning trajectory. In Section 4.3, we further provide an example application that uses this capability to practically assist LLM reasoning under this pipeline.

4.1 Localizing Uncertain Steps: First Wrong Steps

Basic Setups. We first focus on basic math reasoning problems in MATH dataset (Hendrycks et al., 2021), and pure-logical reasoning in BIG-Bench Hard (BBH) (Suzgun et al., 2023), and investigate open-source LLMs: Llama-3.1-8B-Instruct (Dubey et al., 2024), Qwen-2.5-7B-Instruct (Yang et al., 2024) and Mistral-7B-Instruct-v0.3 (Jiang et al., 2023). In addition, we also consider more challenging dataset GPQA (Rein et al., 2024) with more advanced models. Since the experiment is focused on reasoning tasks, we set a low sampling temperature of all LLMs to $T = 0.2$. In Appendix D, we report the results under $T = 0.5$. For hyperparameters, we use $l = 20$ random perturbation with $\sigma = 0.001$, and adversarial perturbation with scale $\alpha = 0.0001$. We analyze hyperparameters and various system prompts, and show our method is robust in Appendix D.

Evaluation Pipeline & Ground-truth Annotation. For each dataset, we collect responses containing reasoning errors[2], and we point out the "first wrong step" in each of them. This annotation is conducted by GPT-5.6, and we conduct a comprehensive manual examination. Especially, for the task of finding the "first wrong step", we inspect MATH and BBH for 180 responses in total, and observe a high human alignment rate 93.9%. The details are in Appendix B.

Then, we evaluate whether a metric can successfully identify this "first wrong step". In detail, for a trajectory with multiple steps, we check the top 10% or top 20% uncertain steps flagged by each method, and report whether the annotated "first wrong step" is among them. Notably, our method and some baselines provide uncertainty scores at the token-level rather than the step-level. For our method, we aggregate token-level scores within each step by taking the maximum. For other token-level baselines, we consider both maximum and mean aggregation.

[2]Responses with incorrect steps can still possibly yield a correct final answer although uncommon. Table 1 includes the results for all responses containing incorrect reasoning steps, regardless of final-answer correctness. We also provide Table 7 which includes only responses with incorrect final answers.

Baselines. We consider a range of baselines including: Negative Log-Likelihood (NLL), Token Entropy, and TokUR (Zhang et al., 2026) (TokUR conducts weight perturbation). These three methods provide token-level signals, which we aggregate into step-level scores by taking the maximum and average over the tokens within each step. We also have step-level baselines: CCP (Fadeeva et al., 2024), TokenSAR (Duan et al., 2023) which are originally for hallucination detection in open-form generations; and Graph Uncertainty (Jiang et al., 2024) which conducts multi-sampling and assess whether each sentence is consistently produced in different generations.

Result. Table 1 shows results on 300 responses with incorrect reasoning steps from each of MATH and BBH. We report the fraction of responses in which the first wrong step is covered by the top 10% or 20% most uncertain steps. From the result, we can see our perturbation-based methods have clear and consistent advantage over all baselines, and Adv. Pert. demonstrates stronger performance than Rand. Pert in most cases. For baselines, we notice that TokUR (Zhang et al., 2026) yields generally better performance than other token-level baselines, suggesting the perturbation-based strategies may be generally effective for this task. Compared with TokUR, our methods are simpler to implement and achieve stronger localization performance. Besides, the efficacy of Graph Uncertainty (Jiang et al., 2024) (multiple-sampling) is usually low, as it struggles to pinpoint the origin of errors (it will also flag the later steps after the first wrong sentence).

In Appendix E, we provide additional visualizations of our scores on MATH and BBH, to demonstrate our scores can identify diverse types of reasoning errors, including problem misinterpretation, calculation errors, and logical inference errors. Moreover, these cases further support that our scores indeed highlights "reasoning-relevant" tokens, as mentioned in Section 3.2.

Table 2: Performance on GPQA.

	GPQA			
	gpt-oss-20b		*Qwen3.5-27B*	
	10%	20%	10%	20%
NLL-max	0.21	0.30	0.23	0.38
Entropy-max	0.18	0.28	0.28	0.39
NLL-avg	0.13	0.31	0.13	0.28
Entropy-avg	0.15	0.28	0.15	0.30
CCP	0.23	0.35	0.30	0.46
TokenSAR	0.18	0.30	0.27	0.49
Rand. Pert.	0.26	0.42	0.29	0.48
Adv. Pert.	**0.32**	**0.43**	**0.36**	**0.50**

Table 2 provides an analysis on a more challenging GPQA benchmark (Rein et al., 2024), which requires more advanced reasoning capability. Here, we consider stronger models including gpt-oss-20b (Agarwal et al., 2025) and Qwen3.5 (Yang et al., 2025) to handle this dataset. These models also contain fewer simple mistakes which can be made by Llama 3.1 and Qwen 2.5. Due to the substantially longer responses (≈2,000 tokens in each response), we collect 100 responses with incorrect reasoning steps for each model and exclude Graph Uncertainty because of its high computational cost. We also exclude TokUR because adapting its low-rank weight perturbation to gpt-oss and Qwen3.5 requires non-trivial, model-specific implementation. As shown in Table 2, our methods remain effective on these more challenging problems, demonstrating their broad applicability across different settings. Interestingly, CCP (Fadeeva et al., 2024) and TokenSAR (Duan et al., 2023) perform more strongly on GPQA than on MATH and BBH. One possible explanation is that these methods are designed for factual hallucination detection, while factual errors may occur more frequently in harder knowledge-intensive questions such as GPQA. In Section 5.2, we further investigate the potentially different mechanisms in factual hallucination and reasoning uncertainty.

4.2 Localizing Uncertain Steps: Fragile Correct Steps

In LLM reasoning, a model may produce a correct intermediate step even when uncertainty is still present. This is also suggested by self-consistency methods (Wang et al., 2022): generations from the same context can lead to both correct and incorrect final outcomes. This implies that some correct steps in a single trajectory may not be stably reproduced in other generations. We therefore study whether our method can identify correct-but-uncertain steps, which we refer to as *fragile correct steps*. Formally, for a correct step s_i, we define the "regeneration risk" to denote whether naturally re-generating from its preceding steps, $s_{1:i-1}$, can cause an error at the position of s_i:

$$\text{risk}(s_i) = \Pr\left(\text{incorrect } s_i^* \mid \text{correct } s_{1:i-1}\right) \tag{8}$$

That is, we re-generate one sentence from $s_{1:i-1}$ multiple times ($n = 10$) and use GPT-5.6 to examine the outputs, and estimate the probability an error emerges immediately at step s_i. In the experiment, we focus on responses whose intermediate steps are all correct but that still exhibit overall uncertainty. We calculate such overall uncertainty via sampling multiple responses for the same question and check if self-consistency ≤ 0.7. We then evaluate whether our method can

Table 3: The max re-generation risk of flagged uncertain steps (95% conf. interval is ± 0.049)

	Math				**Logic**			
	Llama		*Qwen*		*Llama*		*Qwen*	
	10%	20%	10%	20%	10%	20%	10%	20%
NLL-max	0.22	0.29	0.14	0.20	0.20	0.30	0.10	0.19
Entropy-max	0.23	0.30	0.14	0.19	0.22	0.30	0.09	0.18
TokUR-max	0.23	0.31	0.14	0.23	0.23	0.33	0.13	0.17
Rand. Pert.	0.27	0.31	0.21	0.27	0.24	0.31	0.17	0.19
Adv. Pert.	**0.31**	**0.35**	**0.22**	**0.29**	**0.31**	**0.35**	**0.19**	**0.22**

Table 4: The accuracy with external revision on uncertain steps.

	Math (Precalculus)				**Math (Cnt. & Prob.)**				**Logic (Tracking Shuffled)**			
	Llama		*Qwen*		*Llama*		*Qwen*		*Llama*		*Qwen*	
Single	0.33		0.63		0.46		0.79		0.79		0.86	
Voting	0.43		0.69		0.57		0.82		0.92		0.91	
	10%	20%	10%	20%	10%	20%	10%	20%	10%	20%	10%	20%
NLL-max	0.43	0.54	0.71	0.74	0.60	0.66	0.85	0.87	0.96	0.97	0.91	0.93
Entropy-max	0.44	0.51	0.70	0.71	0.62	0.70	0.84	0.84	0.97	0.95	0.89	0.91
TokUR-max	0.46	0.53	0.75	0.74	0.65	0.72	0.86	0.88	0.96	0.95	0.92	0.94
Rand. Pert.	0.47	**0.56**	0.75	0.76	0.66	0.75	0.87	0.88	**0.98**	**0.98**	0.93	0.94
Adv. Pert.	**0.51**	0.54	**0.78**	**0.79**	**0.70**	**0.77**	**0.88**	**0.89**	**0.98**	0.97	**0.94**	**0.95**

identify fragile steps with high risk scores in Eq. 8. To reduce re-generation cost, we compute the regeneration risk only for the top 10% and top 20% most uncertain sentences identified by each method, and report the maximum risk at each cutoff.

Result. Table 3 reports the average results over 100 collected responses from MATH and BBH. Because it need huge number of resampling and additional judgments, we only keep the representative token-level baselines for comparison. From the result, we observe that Adv. Pert. and Rand. Pert. also consistently outperforms the baselines, especially given a smaller range (top 10%). This may suggest that such fragile correct steps are relatively sparse. Overall, this result shows that our methods capture uncertainty even when the realized reasoning steps are correct, rather than merely serving as incorrectness indicators.

Remarks. In practice, both type of uncertain steps (wrong steps & fragile correct steps) may coexist in the same uncertainty response. Evaluating such mixed cases is challenging due to the difference in their evaluation pipelines. Thus, we only evaluate them separately. In Appendix E, we also visualize a few cases for correct responses with overall low (self-consistency) uncertainty, and our uncertainty curve usually remains flat throughout the reasoning trajectory.

4.3 Practical Utility

In this section, we study one possible application of our method to use our signals to guide targeted inspection. We suppose a human practitioner who uses an LLM to solve a reasoning problem and suspects that the generated response may be unreliable. Similar to the previous study, such unreliable responses may be identified using existing response-level uncertainty methods, i.e., self-consistency ≤ 0.7. By applying our method, the practitioner can narrow the response down to a small subset of potentially problematic steps. They can then inspect these flagged steps to trace possible reasoning flaws. This can be especially useful for long responses, where inspecting every step is costly.

In Table 4, we simulate this inspection process using GPT-5.6. For each flagged step, GPT-5.6 is given the preceding context and asked to determine whether the flagged step is incorrectly derived, which is tasked to answer "*whether this step introduces a new error*". If the step is judged incorrect, GPT-5.6 provides an explanation of the error and a suggestion for revision. This feedback is then provided to the base model to revise its response. Notably, if the flagged step does not introduce a "new error", e.g., an incorrect claim based on previous errors, we still annotate it as "no error," since a human inspector would possibly regard it as consistent with the preceding context.

Result. From Table 4 we show the reasoning accuracy of Llama and Qwen after this revision process. We consider relatively challenging subsets from MATH and BBH, with each has 100 randomly chosen questions. Here, since we employ multiple sampling to obtain self-consistency score, we conduct the revision based on a randomly selected majority response among these samples. The results show our method can help the models answer more questions correctly, especially when only a small fraction of steps are examined (10%) and in harder datasets such as MATH.

5 Additional Studies

We perform studies to further understand our method. More ablation studies are in Appendix D.

5.1 How is the Efficiency?

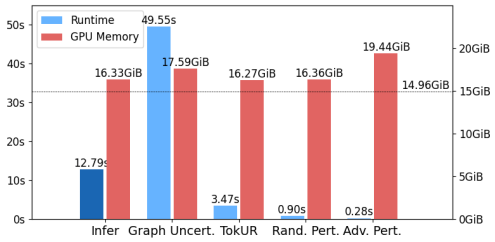


Figure 3: Time and memory efficiency.

In Figure 3, we report the model average response generation time (deep blue), and average additional computational time (light blue) incurred by each uncertainty estimation method, together with their total memory consumption. All experiments are conducted using Llama-3.1-8B-Instruct in BF16 precision on a single NVIDIA GH200 GPU. The results show that perturbation-based metrics introduce only modest computational overhead. For example, Rand. Pert. with $l = 20$ forward passes, requires approximately 0.90s additional computation. Adv. Pert., which involves one backward and one forward pass, incurs approximately 0.28s. Both overheads are small compared with the average 12.79s of the original response generation. In terms of memory, Adv. Pert. requires approximately 3 GB of additional memory due to backpropagation.

5.2 Hallucination Detection

Table 5: Hallucination Detection.

FactScore	*Llama*	*Qwen*	*Mistral*
NLL	0.610	0.638	0.604
Entropy	0.635	0.623	0.596
TokUR	0.706	0.670	0.674
CCP	0.660	0.659	0.687
TokenSAR	0.716	0.631	0.641
Graph Uncert.	**0.778**	**0.878**	**0.887**
Rand. Pert.	0.675	0.596	0.576
Adv. Pert.	0.678	0.599	0.586

We also investigate the FactScore benchmark (Min et al., 2023) to evaluate the effectiveness of various metrics in identifying factual errors (hallucinations) when LLMs are queried to compose celebrity biographies. Table 5 reports the AUROC of various methods, to compare their ability to detect hallucinated claims in the entire response. From the result, we observe that multiple-sampling-based method (Graph Uncert.) performs the best, while our method does not perform comparably. One possible explanation is that the underlying mechanism of factual hallucination can differ from the reasoning errors. For example, prior work (Maynez et al., 2020) shows that hallucinated content in an "abstractive summarization" task is often weakly grounded in the source context. Through human evaluation, they find that approximately 63%–72% of model-generated summaries contain extrinsic hallucinations, which are content that is not supported by the source material. In this case, these errors can be difficult to be flagged by perturbation-based methods. Admittedly, in practice, factual hallucination errors can be difficult to disentangle from reasoning errors, especially in knowledge-intensive reasoning problems (Rein et al., 2024). For example, a model may fail because it lacks knowledge about a certain entity, or because it incorrectly recalls or applies factual information. This is why we focus on relatively simpler dataset in MATH and BBH, where factual hallucination is less prevalent (although we also study in GPQA). We leave a comprehensive investigation of the interaction between hallucination and reasoning uncertainty to future work.

6 Conclusion and Limitation

Unlike prior uncertainty quantification approaches for LLM reasoning tasks that are largely confined to assessing confidence in the final answer, our method concentrates on uncertainty localization to the intermediate reasoning process. However, our study also has several limitations. **First**, our perturbation-based methods require white-box access to the model, including its token embeddings and gradients, which limits their applicability to closed-source models. **Second**, factual hallucination and reasoning errors can be difficult to disentangle in knowledge-intensive reasoning tasks. A model may fail because of missing or incorrect factual knowledge rather than instability in the reasoning process itself. We therefore focus mainly on MATH and BBH, where reasoning errors can be studied more cleanly, and leave a more systematic treatment of hallucination-related uncertainty to future work. **Third**, our method focuses on uncertainty localization rather than calibrated uncertainty quantification. The perturbation scores are designed primarily to rank relatively uncertain positions within a reasoning trajectory, rather than to provide an absolute probability of error for each step. As a result, the scores may not be directly comparable across different questions or responses.

AI USE STATEMENT

In this work, we used generative AI tools to assist with LLM response annotation and evaluation, including judging if a reasoning response contain mistakes, locating first reasoning error, and verifying the correctness of re-generated steps. Because these annotations contribute directly to our experimental evaluation, we manually examined their annotation accuracy in Appendix B.

Additionally, generative AI tools were used to improve the language and readability of the writing. All AI-assisted text was reviewed and revised by the authors to ensure that it accurately expresses the intended meaning. Generative AI tools were also used to summarize existing literature, which were checked by the authors against the original sources for relevance and accuracy. Generative AI tools were participated in mathematical proofs in Appendix A, which were used only as suggestions, and every step included in the final proof was independently examined by the authors for mathematical correctness. Generative AI tools were participated in part of software implementation. The generation was reviewed and tested by the authors. We have not used generative AI tools for any other tasks with required disclosure, and we take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We provide the implementation of our proposed perturbation-based uncertainty metrics and experimental evaluation pipeline in `https://anonymous.4open.science/r/pbul-AB53`. Experimental settings, including datasets, models, baselines, hyperparameters, and evaluation protocols, are described in Section 4 and Appendix C. Prompts involved in the experiments are included in Figure 4. Together, these materials are intended to enable reproduction of our results.

REFERENCES

Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K Arora, Yu Bai, Bowen Baker, Haiming Bao, et al. gpt-oss-120b & gpt-oss-20b model card. *arXiv preprint arXiv:2508.10925*, 2025.

Eric Bigelow, Ari Holtzman, Hidenori Tanaka, and Tomer Ullman. Forking paths in neural text generation. In *International Conference on Learning Representations*, volume 2025, pp. 17134–17167, 2025.

Jinhao Duan, Hao Cheng, Shiqi Wang, Alex Zavalny, Chenan Wang, Renjing Xu, Bhavya Kailkhura, and Kaidi Xu. Shifting attention to relevance: Towards the predictive uncertainty quantification of free-form large language models. *arXiv preprint arXiv:2307.01379*, 2023.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv e-prints*, pp. arXiv–2407, 2024.

Ekaterina Fadeeva, Aleksandr Rubashevskii, Artem Shelmanov, Sergey Petrakov, Haonan Li, Hamdy Mubarak, Evgenii Tsymbalov, Gleb Kuzmin, Alexander Panchenko, Timothy Baldwin, et al. Fact-checking the output of large language models via token-level uncertainty quantification. *arXiv preprint arXiv:2403.04696*, 2024.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023. URL `https://arxiv.org/abs/2310.06825`.

Mingjian Jiang, Yangjun Ruan, Prasanna Sattigeri, Salim Roukos, and Tatsunori Hashimoto. Graph-based uncertainty metrics for long-form language model generations. *Advances in Neural Information Processing Systems*, 37:32980–33006, 2024.

Zhengbao Jiang, Jun Araki, Haibo Ding, and Graham Neubig. How can we know when language models know? on the calibration of language models for question answering. *Transactions of the Association for Computational Linguistics*, 9:962–977, 2021.

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

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

Xianzhi Li, Ethan Callanan, Abdellah Ghassel, and Xiaodan Zhu. Entropy-gated branching for efficient test-time reasoning. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5054–5069, 2026.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, volume 2024, pp. 39578–39601, 2024.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. On faithfulness and factuality in abstractive summarization. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 1906–1919, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.173. URL https://aclanthology.org/2020.acl-main.173/.

Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. Factscore: Fine-grained atomic evaluation of factual precision in long form text generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 12076–12100, 2023.

Shrey Pandit, Austin Xu, Xuan-Phi Nguyen, Yifei Ming, Caiming Xiong, and Shafiq Joty. Hard2verify: A step-level verification benchmark for open-ended frontier math. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 22502–22517, 2026.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. Gpqa: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*, 2024.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. *Advances in neural information processing systems*, 36:68539–68551, 2023.

Gaurang Sriramanan, Siddhant Bharti, Vinu Sankar Sadasivan, Shoumik Saha, Priyatham Kattakinda, and Soheil Feizi. Llm-check: Investigating detection of hallucinations in large language models. *Advances in Neural Information Processing Systems*, 37:34188–34216, 2024.

Lihao Sun, Hang Dong, Bo Qiao, Qingwei Lin, Dongmei Zhang, and Saravan Rajmohan. Llm reasoning as trajectories: Step-specific representation geometry and correctness signals. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 26872–26887, 2026.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed H Chi, Denny Zhou, et al. Challenging big-bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 13003–13051, 2023.

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

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xiong-Hui Chen, Jianxin Yang, Zhenru Zhang, et al. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for llm reasoning. *Advances in Neural Information Processing Systems*, 38:115452–115486, 2026.

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

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

Miao Xiong, Zhiyuan Hu, Xinyang Lu, Yifei Li, Jie Fu, Junxian He, and Bryan Hooi. Can llms express their uncertainty? an empirical evaluation of confidence elicitation in llms. *arXiv preprint arXiv:2306.13063*, 2023.

Yasin Abbasi Yadkori, Ilja Kuzborskij, András György, and Csaba Szepesvári. To believe or not to believe your llm. *arXiv preprint arXiv:2406.02543*, 2024.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, et al. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Tunyu Zhang, Haizhou Shi, Yibin Wang, Hengyi Wang, Xiaoxiao He, Zhuowei Li, Haoxian Chen, Ligong Han, Kai Xu, Huan Zhang, et al. Tokur: Token-level uncertainty estimation for large language model reasoning. In *International Conference on Learning Representations*, volume 2026, pp. 52910–52942, 2026.

Chen Zhu, Yu Cheng, Zhe Gan, Siqi Sun, Tom Goldstein, and Jingjing Liu. Freelb: Enhanced adversarial training for natural language understanding. *arXiv preprint arXiv:1909.11764*, 2019.

A PROOF

In this section, we first provide proof for Proposition 3.3. For simplicity, we use $\mathbf{H}$, $\tilde{\mathbf{H}}$ and $\boldsymbol{\varepsilon}$ to replace $\mathbf{H}_{1:t-1}$, $\tilde{\mathbf{H}}_{1:t-1}$ and $\boldsymbol{\varepsilon}_{1:t-1}$, and use $\delta = \nabla_{\mathbf{H}} f_t(\mathbf{H})$.

Lemma A.1 (Random Perturbation approximates gradient norm). *For Random Perturbation, $\tilde{\mathbf{H}} = \mathbf{H} + \boldsymbol{\varepsilon}$. First order Taylor expansion gives $f_t(\tilde{\mathbf{H}}) = f_t(\mathbf{H} + \boldsymbol{\varepsilon}) \approx f_t(\mathbf{H}) + \delta^T \boldsymbol{\varepsilon}$ for sufficiently small $\boldsymbol{\varepsilon}$. Applying to Eq. 4, random perturbation variance could be approximated by the square of gradient norm:*

$$\begin{aligned}
\text{Pert.}(x_t) &= \text{Var}_{\boldsymbol{\varepsilon}}\Big[f_t(\tilde{\mathbf{H}})\Big] && (9)\\
&\approx \text{Var}_{\boldsymbol{\varepsilon}}\big[f_t(\mathbf{H}) + \delta^T \boldsymbol{\varepsilon}\big] && (10)\\
&= \text{Var}_{\boldsymbol{\varepsilon}}\big[\delta^T \boldsymbol{\varepsilon}\big] && (11)\\
&= \sigma^2 \|\delta\|_2^2 && (12)
\end{aligned}$$

proof for Proposition 3.3. First, we use First order Taylor expansion $f_t(\tilde{\mathbf{H}}) - f_t(\mathbf{H}) \approx \delta^T \boldsymbol{\varepsilon}$ (again, we assume that $\boldsymbol{\varepsilon}$ is sufficiently small), and $\boldsymbol{\varepsilon} \sim \mathcal{N}(0, \sigma^2 I)$. Therefore $\delta^T \boldsymbol{\varepsilon} \sim \mathcal{N}(0, \sigma^2 \|\delta\|_2^2)$. Consequently, using the cumulative distribution function of a Gaussian distribution:

$$\Pr_{\boldsymbol{\varepsilon}}\left[f_t(\tilde{\mathbf{H}}) \leq f_t(\mathbf{H}) - c\right] = \Pr_{\boldsymbol{\varepsilon}}\left[f_t(\tilde{\mathbf{H}}) - f_t(\mathbf{H}) \leq -c\right] \tag{13}$$

$$\approx \Pr_{\boldsymbol{\varepsilon}}\left[\delta^T \boldsymbol{\varepsilon} \leq -c\right] \tag{14}$$

$$= \Phi\left(-\frac{c}{\sigma\|\delta\|_2}\right) \tag{15}$$

Substituting Lemma A.1, we get:

$$\Pr_{\boldsymbol{\varepsilon}}\left[f_t(\tilde{\mathbf{H}}) \leq f_t(\mathbf{H}) - c\right] \approx \Phi\left(-\frac{c}{\sigma\|\delta\|_2}\right) \tag{16}$$

$$\approx \Phi\left(-\frac{c}{\sqrt{\text{Pert.}(x_t)}}\right) \tag{17}$$

Hence, the proposition is proved.

□

Similarly, Proposition 3.4 can be easily proven by a first order Taylor expansion, we also use $\hat{\mathbf{H}}$ to replace $\hat{\mathbf{H}}_{1:t-1}$ for simplicity.

proof for Proposition 3.4. Since the adversarial perturbation is sufficiently small, we use first order Taylor expansion to expand $f_t(\hat{\mathbf{H}}) = \log \Pr(x_t \mid \hat{\mathbf{H}})$:

$$f_t(\hat{\mathbf{H}}) = f_t(\mathbf{H} - \alpha \,\text{sgn}(\nabla_{\mathbf{H}} \mathcal{F}(\mathbf{H}))) \tag{18}$$

$$\approx f_t(\mathbf{H}) - \alpha \cdot \langle \nabla_{\mathbf{H}_{1:t-1}} \log \Pr(x_t \mid \mathbf{H}_{1:t-1}) \,,\, \text{sgn}(\nabla_{\mathbf{H}_{1:t-1}} \mathcal{F}(\mathbf{H}))\rangle \tag{19}$$

Substitute back to the definition of adversarial perturbation:

$$\text{Pert.}(x_t) = f_t(\mathbf{H}_{1:t-1}) - f_t(\hat{\mathbf{H}}_{1:t-1}) \tag{20}$$

$$\approx \alpha \cdot \langle \nabla_{\mathbf{H}_{1:t-1}} \log \Pr(x_t \mid \mathbf{H}_{1:t-1}) \,,\, \text{sgn}(\nabla_{\mathbf{H}_{1:t-1}} \mathcal{F}(\mathbf{H}))\rangle \tag{21}$$

Hence, the proposition is proved.

□

B Prompts and Human Examination

Prompts. We use a strong online LLM 'GPT-5.6-sol' for annotation works in the experiment. The prompts are provided in Figure 4. The first prompt is applied for judging if a reasoning response contains intermediate mistakes in Section 4.1. The second prompt applied for locating first incorrect step in Section 4.1. The third prompt applied in Section 4.2 to determine the correctness of step regeneration in order to calculate risk score. The last prompt applied in Section 4.3 is used for feedback and revision.

Human Examination of GPT Annotation. For Section 4.1, we first focus on 180 responses which GPT annotated as "containing incorrect reasoning steps". We employ human inspector to examine the quality of GPT's annotation in the task of finding "first wrong step". In this procedure, we ask the human inspector to read the first wrong step suggested by the GPT, and check the preceding reasoning contents to find whether the preceding contents have earlier errors. If human inspector agree with GPT judgement, we label it as an **aligned case**. As shown in Table 6, these annotations achieve an overall alignment rate of 93.9%. Meanwhile, the human inspector find all these 180 cases indeed contain wrong steps.

On the other hand, we also let the human inspector to read through the responses that GPT find no reasoning errors, and meanwhile referring to the correct answer provided by the dataset. Among the 30 examined cases, 29 cases are verified to be no errors. The remaining case contains ambiguity, making its reasoning correctness difficult to determine conclusively.

Finally, for Section 4.2, we manually examine 80 responses for the annotation task that check whether a re-generated step is correct in risk calculation. It shows 97.5% alignment rate. These results indicate that GPT provides sufficiently reliable annotations for the evaluations considered in our experiments.

Table 6: Human examination result for 'first wrong step' annotation in 180 responses.

			Math			
	algebra	cnt. & prob.	number theory	precalculus	inter. algebra	prealgebra
Llama	5 / 5	4 / 5	5 / 5	4 / 5	5 / 5	5 / 5
Qwen	4 / 5	4 / 5	4 / 5	4 / 5	5 / 5	5 / 5
Mistral	4 / 5	5 / 5	4 / 5	5 / 5	5 / 5	5 / 5
			Logic			
	logical deduction	tracking shuffled	web of lies	formal fallacies	navigate	temp. seq.
Llama	5 / 5	5 / 5	5 / 5	5 / 5	5 / 5	5 / 5
Qwen	5 / 5	5 / 5	5 / 5	5 / 5	5 / 5	4 / 5
Mistral	4 / 5	5 / 5	5 / 5	5 / 5	4 / 5	5 / 5

Reasoning Correctness Judge

```
Analyze the reasoning steps that attempt to solve this question, and decide whether the internal
reasoning is correct.
```

Incorrect Intermediate Step Annotation

```
Question:
[Question]

Response (split into numbered steps):
[Response in steps]

Please provide your analysis for each step, about whether it contains specific calculation or logical
mistake that leads to incorrect final result.

[Output step by step analysis]

According to the analysis, identify the step id of the first incorrect step.
If the analysis points out multiple steps, do NOT mark procedural steps that only contain instruction
for the next step, without actually performing calculation or logical deduction; do NOT mark steps that
may be confusing, but are not containing specific calculation or logical mistakes; instead, choose the
exact step that actually makes a calculation or logical mistake that leads to incorrect final result.
Please first reason and then output your final answer in the format:
WRONG_STEP: <integer>
```

Continuation Correctness Annotation

```
Question:
[Question]

Incomplete response to this question:
[Response prefix]

Here is one step continuation of the incomplete response:
[Continuation]

Please judge whether this continuation is correct based on the given incomplete response.
If the prior response contains mistakes, please ignore them, and focus on whether the continuation is
correctly deduced from given prefix, or whether the continuation contains intrinsic flaws. Since this
is only a one step continuation, the response could be still incomplete, please do not treat
incompleteness as incorrectness. Please first give your reasons, then provide your final judgement in a
new line with format: Judgement: <Yes | No>"
```

GPT feedback

```
Question:
[Question]

Reasoning:
[Response in steps]

Read uncertain steps [Uncertain Step Indices] one by one, and check whether each uncertain step is a
new reasoning error. If it is not a new error, reply only: 'No error'; otherwise, provide a brief
feedback for the step, including the error analysis and suggestion to correct.
```

Base model revision

```
[Feedback]

Based on the feedback, revise your response.
```

Figure 4: Prompts used in the experiments. Title of each prompt is in bold, placeholders are marked as '[]' with description within.

C Detailed Experiment Setup

In this section we provide more detailed information about our experiments in Section 4. Including benchmark subset selection; baseline settings; step split detail; and simulation details for application scenario.

Subsets. For the datasets we use in our experiments, MATH (Hendrycks et al., 2021) and BBH(Suzgun et al., 2023) contain subsets that split the math or logic reasoning into more fine-grained tasks. For MATH (Hendrycks et al., 2021), we consider the testing split of 6 subsets: 'algebra', 'counting_and_probability', 'number_theory', 'precalculus', 'intermediate_algebra' and 'prealgebra'. We will skip questions with graphics. For BBH(Suzgun et al., 2023), we consider the testing split of 10 subsets: 'logical_deduction_three_objects', 'logical_deduction_five_objects', 'logical_deduction_seven_objects', 'tracking_shuffled_objects_three_objects', 'tracking_shuffled_objects_five_objects', 'tracking_shuffled_objects_seven_objects', 'web_of_lies', 'formal_fallacies', 'navigate' and 'temporal_sequences'.

Baselines. In our evaluation, we compare the proposed method with various intermediate uncertainty baselines. For baselines that provide token-wise signals, we have Negative Log-Likelihood (**NLL**), token entropy (**Entropy**), and **TokUR**(EU) (Zhang et al., 2026). TokUR is a Bayesian-based token-level uncertainty estimator with low-rank weight perturbation, but the original evaluation is primarily conducted after aggregating those token-level uncertainties into response-level scores, rather than evaluating in intermediate steps. We present the epistemic uncertainty (EU) version of TokUR for reasoning because it performs better than total uncertainty (TU) or aleatoric uncertainty (AU) in our setting. Except for FactScore (Min et al., 2023) we use AU which is because it performs the best, which is also supported by the original paper. For hyperparameters of TokUR, after exploration, we find perturbation strength $\sigma_q = 0.001$, rank $r' = 8$ and number of samples $M = 2$ is the optimal setting in our main experiment of Section 4. In addition, we have **CCP** (Fadeeva et al., 2024), a probability-based method that removes the impact of alternative same-meaning tokens and functional words. As it treats functional words as $\text{CCP}_{\text{word}} = 1$, and it is only quantitatively evaluated in response-level, we will use the sequence aggregated version provided by the original paper. **TokenSAR** (Duan et al., 2023) is another probability-based method that weight negative log-likelihood with semantic significancy to predict uncertainty for a token. To perform its essence of "weighted uncertainty" or "shifted uncertainty", we will directly use its own sequence aggregated version. Graph Uncertainty(**Graph Uncert.**) (Jiang et al., 2024) is a Self-Consistency(Wang et al., 2022) like multiple sampling based claim-level uncertainty metric, that split sampled responses and their claims into bipartite graph, represent uncertainty using graph features. We will use closeness (C_C), as it is presented as the best metric in the original experiment. This method is originally designed for factual claim uncertainty estimation, and it's workflow is combined into FactScore (Min et al., 2023) dataset. Therefore, we have made modifications to migrate this method to evaluate reasoning step consistency. Specifically, we use "sentence-transformers/all-MiniLM-L6-v2" to produce sentence embedding for each step, then use KMeans to cluster the steps into nodes, we will then follow the Graph Uncertainty methodology to obtain the uncertainty scores for each node.

Step split detail. For an LLM response, we use an efficient rule-based step segmentation method. We use sentence-ending indication characters '.!?\n' as step boundaries, and handle the following corner cases: 1. '.' serving as decimal point or ellipsis will not split; 2. '!' as factorial (next to digit) will not split; 3. brackets, '\n\t' following a step will be included in that sentence (e.g. 'Claire is now left midfielder.\n\n' is one step); 4. step shorter than 5 chars will not split and be included to later step; 5. whatever is left at the end becomes the last step. Token boundaries are mostly overlap with step boundaries, if not, it will be mapped to the later step.

Simulation details for application scenario. To simulate the application scenario in Section 4.3, we sample $n = 10$ generations for each problem and randomly select one generation whose final answer matches the majority answer, i.e., the answer with the highest self-consistency score, as the target response. The accuracy of these target responses constitutes the majority-vote baseline (correspondingly, 'single' baseline simply uses the first generation as the target response). Responses with a consistency score of more than 0.7 are considered reliable and are returned directly. For the remaining unreliable responses, each UQ method flags the top 10% or 20% most uncertain reasoning steps. The exact number of selected steps is obtained by rounding 10% or 20% of the total

Table 7: Performance of each method in 300 responses with incorrect final answer, The 95% conf. interval is $\approx \pm 0.057$ for 300 Bernoulli trials (p=0.5).

	Math						**Logic**					
	Llama		*Qwen*		*Mistral*		*Llama*		*Qwen*		*Mistral*	
	10%	20%	10%	20%	10%	20%	10%	20%	10%	20%	10%	20%
NLL-max	0.18	0.29	0.21	0.39	0.15	0.28	0.21	0.39	0.28	0.51	0.17	0.33
Entropy-max	0.19	0.33	0.24	0.41	0.14	0.24	0.21	0.40	0.26	0.51	0.15	0.31
TokUR-max	0.27	0.49	0.26	0.44	0.28	0.44	0.27	0.42	0.29	0.45	0.26	0.45
NLL-avg	0.13	0.24	0.17	0.31	0.09	0.18	0.18	0.38	0.19	0.39	0.11	0.27
Entropy-avg	0.13	0.26	0.15	0.34	0.10	0.18	0.19	0.41	0.19	0.40	0.12	0.26
TokUR-avg	0.20	0.36	0.18	0.34	0.24	0.42	0.24	0.40	0.23	0.41	0.25	0.38
CCP	0.19	0.34	0.31	0.50	0.20	0.34	0.26	0.45	0.31	0.55	0.20	0.38
TokenSAR	0.18	0.27	0.26	0.42	0.18	0.31	0.27	0.46	0.30	0.53	0.22	0.40
Graph Uncertain.	0.20	0.28	0.20	0.31	0.15	0.20	0.22	0.36	0.30	0.44	0.15	0.30
Rand. Pert.	0.40	0.54	0.32	0.49	0.32	**0.51**	0.34	0.50	0.39	0.54	0.28	0.48
Adv. Pert.	**0.45**	**0.60**	**0.35**	**0.52**	**0.35**	**0.51**	**0.43**	**0.58**	**0.40**	**0.57**	**0.36**	**0.50**

Table 8: Results on MATH and BBH, following the same setting in Section 4.1, with LLM temperature $T = 0.5$.

	Math						**Logic**					
	Llama		*Qwen*		*Mistral*		*Llama*		*Qwen*		*Mistral*	
	10%	20%	10%	20%	10%	20%	10%	20%	10%	20%	10%	20%
NLL-max	0.20	0.33	0.21	0.42	0.17	0.32	0.20	0.39	0.26	0.47	0.14	0.32
Entropy-max	0.24	0.37	0.24	0.40	0.12	0.24	0.18	0.38	0.23	0.47	0.15	0.31
TokUR-max	0.25	0.46	0.27	0.45	0.28	0.45	0.26	0.40	0.29	0.49	0.23	0.45
NLL-avg	0.17	0.27	0.16	0.33	0.11	0.20	0.17	0.38	0.19	0.37	0.10	0.27
Entropy-avg	0.17	0.29	0.13	0.34	0.10	0.20	0.18	0.40	0.17	0.37	0.12	0.27
TokUR-avg	0.20	0.35	0.17	0.35	0.24	0.40	0.22	0.40	0.25	0.45	0.22	0.41
CCP	0.22	0.37	0.32	0.48	0.21	0.33	0.25	0.43	0.28	0.52	0.20	0.38
TokenSAR	0.23	0.33	0.28	0.41	0.19	0.32	0.27	0.46	0.27	0.51	0.23	0.41
Graph Uncertain.	0.18	0.26	0.18	0.28	0.14	0.18	0.23	0.37	0.30	0.43	0.16	0.29
Rand. Pert.	0.36	0.51	0.31	0.50	0.31	0.48	0.32	0.50	0.38	0.55	0.27	0.46
Adv. Pert.	**0.39**	**0.56**	**0.33**	**0.51**	**0.33**	**0.49**	**0.43**	**0.59**	**0.41**	**0.57**	**0.35**	**0.50**

number of steps in the response, with at least one step selected (the same rule is used for top-10% and top-20% settings in all experiments). We then use GPT-5.6 as an external assistant to provide feedback on each flagged step. The base model revises its response based on this feedback, and the revised response is taken as the final output. Accuracy over all final outputs, including both directly returned reliable responses and revised unreliable responses, measures the practical utility of each UQ method in this scenario.

Table 9: Experiment results using different system prompts.

Prompt 1 (default)	You are a helpful assistant.
Prompt 2	You are a careful and rigorous reasoner.
Prompt 3	You are an expert problem solver in reasoning.

	Prompt 1		Prompt 2		Prompt 3	
	10%	20%	10%	20%	10%	20%
NLL	0.18	0.34	0.22	0.32	0.20	0.32
Entropy	0.24	0.32	0.18	0.32	0.20	0.36
TokUR	0.28	0.46	0.30	0.48	0.26	0.48
CCP	0.22	0.32	0.20	0.32	0.18	0.32
TokenSAR	0.22	0.28	0.20	0.32	0.20	0.32
Graph Uncertain.	0.20	0.30	0.22	0.24	0.20	0.34
Rand. Pert.	0.40	0.56	0.38	0.56	0.40	0.62
Adv. Pert.	**0.46**	**0.62**	**0.46**	**0.60**	**0.44**	**0.66**

D Extra Experiments and Results

We provide additional experiment results in this section.

MATH and BBH incorrect step locating on responses with wrong final answer. In Table 1 we sample 300 responses containing incorrect reasoning steps. Some of them result in correct final answer. To erase this influence, we continue to sample based on previous results and until 300 responses with incorrect final answer is collected, then measure the performance of each method in Table 7. The similar observations could be drawn from this table.

MATH and BBH incorrect step locating with $T = 0.5$. Table 8 reports the successful detection rate of the first incorrect reasoning step for various uncertainty methods at a model temperature of $T = 0.5$, as a comparison for our main experiment setting $T = 0.2$. We collect a total of 120 cases for each setting from MATH and BBH. The results maintain similar, leading to conclusions consistent with those discussed in Section 4.

Experiments using different system prompts. Table 9 record the result of MATH 'precalculus' subset using Llama, under 3 versions of system prompts for querying the model with reasoning questions. The results exhibit similar behavior across all three prompts, indicating that the choice of system prompt does not materially affect the observed results.

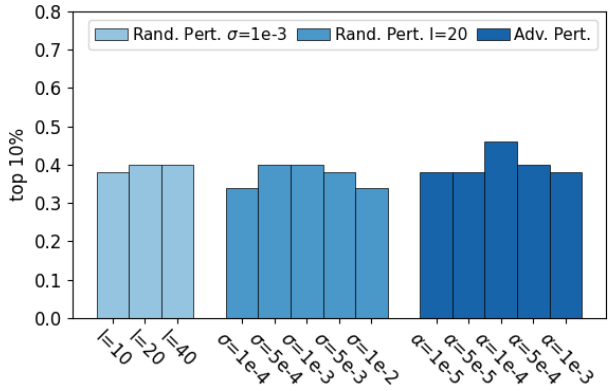


Figure 5: Result under various hyper-parameters.

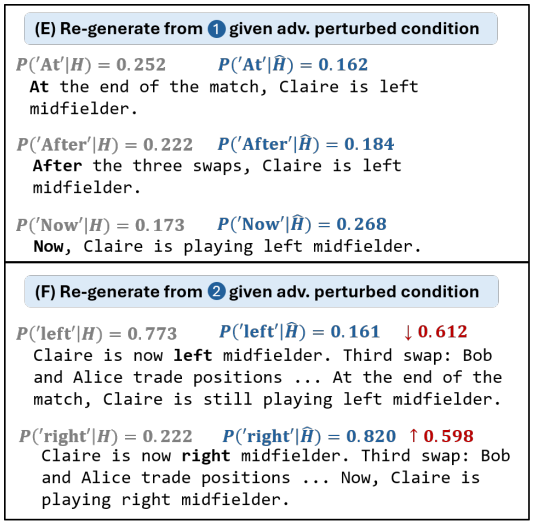


Figure 6: For the same setting as Figure 2, we re-generate continuations conditioned on adversarial perturbed embeddings $\hat{\mathbf{H}}_{1:t-1}$. We find that the continuations are also coherent and on-task, similar to Figure 2. This shows that adversarial perturbation on the embedding also does not disrupt reasoning coherence.

Hyperparameter Study In our experiments, we set the hyperparameters as follows by default: (1) the number of noise sampling in Rand. Pert. to be $l = 20$ times. (2) The standard deviation of

the noise σ in Rand. Pert. to be 0.001. (3) The intensity value α in Adv. Pert. to be 0.0001. We select the value of l as a trade-off in efficiency, and we choose perturbation scales (σ and α) to be sufficiently small. In this subsection, we conduct ablation study to see how different values of these hyperparameters affect the performance. The experiment is conducted on MATH precalculus subset using Llama, the result is shown in Figure 5. For the hyperparameter l, there is minor increase from $l = 10$ to $l = 20$, and almost no increase from $l = 20$ to $l = 40$. This may suggest the marginal gain for increasing l is diminishing. For noise scale σ, we find that our selected $\sigma = 0.001$ serves as a reasonable hyperparameter value. Similar observation also exists in $\alpha = 0.0001$ for Adv. Pert. These results confirm that our selected hyperparameters in the main results are suitable, and the performance is not very sensitive to the selection under a reasonable range.

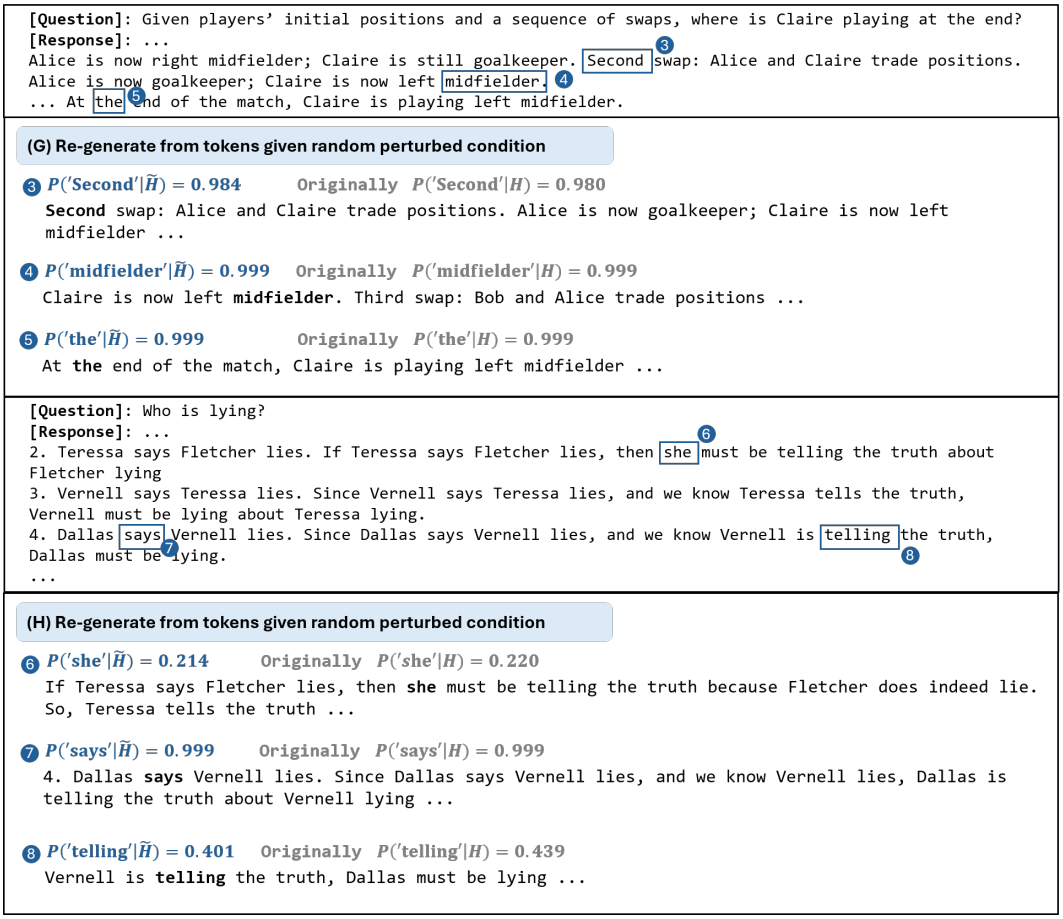


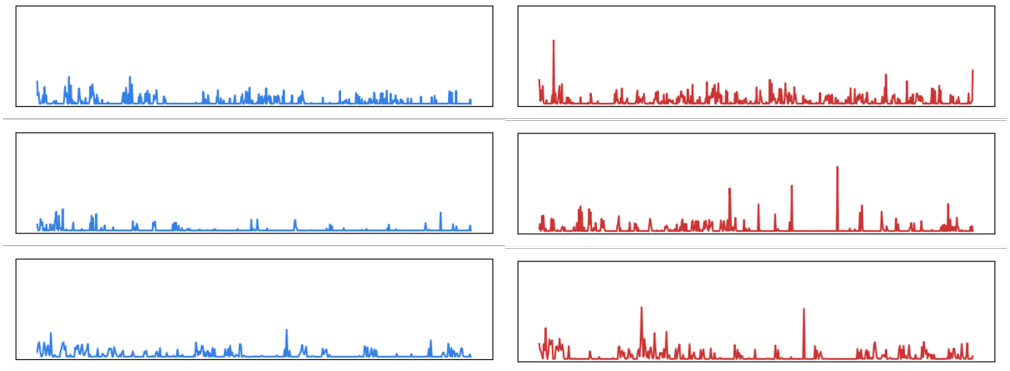
Figure 7: Extra cases of regeneration from perturbed conditions.

E CASE STUDIES

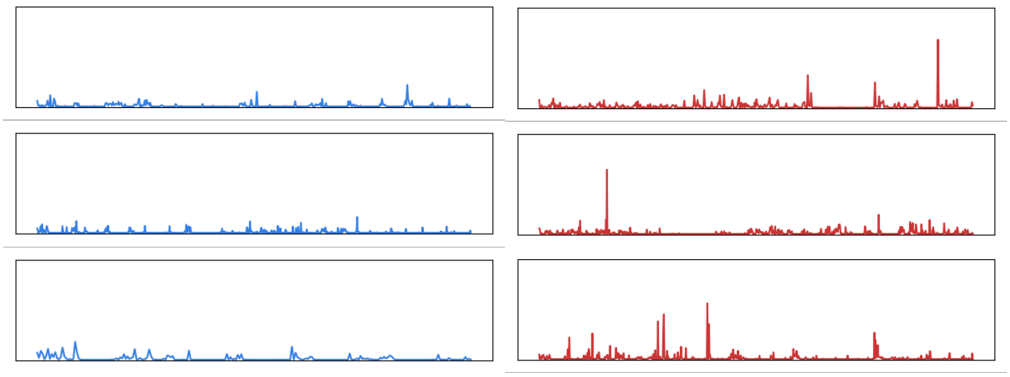
Extra cases of re-generation from perturbed conditions. Figure 7 shows extra re-generations from random perturbed conditions remain valid and coherent, such that small embedding perturbation does not disrupt reasoning coherence.

Cases of correct and low risk solutions. Figure 8 compares three certain responses (with self-consistency > 0.7) and three uncertain responses (containing reasoning errors) by Llama in MATH. The scores are presented without normalization to facilitate a direct comparison of their absolute values. In general, the uncertainty captured by perturbation-based methods remain relatively low and stable for most tokens, with a few tokens exhibiting substantially higher scores. Notably, uncertain responses tend to exhibit higher peaks, suggesting that these localized high-score tokens may serve

as indicators of increased uncertainty. Certain responses performs mostly flat and low peak, showing that the outputs are indeed certain and stable.



Un-normalized **random perturbation** score curves of 3 certain and 3 uncertain responses.



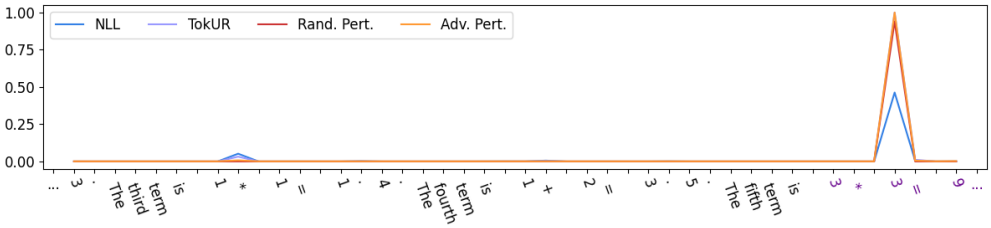
Un-normalized **adversarial perturbation** score curves of 3 certain and 3 uncertain responses.

Figure 8: Comparison of un-normalized perturbation-based scores between uncertain responses (red) and certain responses (blue). Here certain responses indicate self-consistency > 0.7.

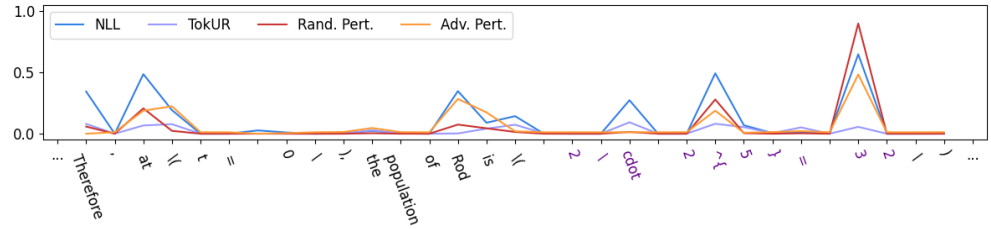
More cases under various settings. In Figure 9 and Figure 10, we present more cases under different datasets and models. In Figure 9, we collect cases from MATH dataset using Llama, Qwen, and Mistral. In Figure 10, we collect cases from BBH dataset using Llama, Qwen, and Mistral. These cases provide more support to Section 4, that perturbation-based methods better identify erroneous tokens related to reasoning instead of semantics, and they are able to identify multiple forms of errors by flagging their uncertainty.

Success and failure cases. In Figure 11, we provide two examples in logic reasoning, one successful case by Llama and one failure case by Qwen. As shown in Llama case, the model wrongly recalls a previous conclusion, and claims "Vernell is telling the truth". The perturbation-based metrics can precisely locate the token "telling" where the model begins to deviate. In the Qwen case, the model incorrectly infers "second newest is tractor" from the preceding fact "the tractor is the second oldest." Although the perturbation-based metrics assign a locally higher uncertainty score to the first incorrect token 'Tr' than to its neighboring tokens, several other tokens, such as 'ranking' and 'Newest' receive even higher scores. This case visualizes a confident error, that our methods fails to identify.

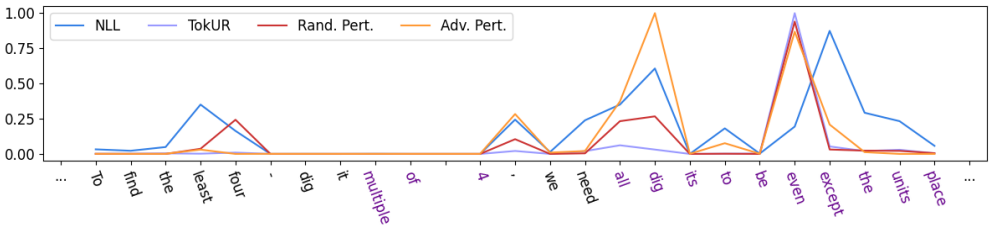
Case of correct but risky step. Sometimes we observe uncertainty in correct steps, which may suggest that the step is risky (Section 4.2). Figure 12 shows a case where the model generates a critical numeric token '34' correctly but uncertain (high perturbation scores). Regeneration from the start of this step further shows that the token '34' is indeed unstable, as it contains risk to be replaced by incorrect tokens such as '32' or '38'.



•**Problem misread error identified in Llama.** The LLM is asked to construct a sequence, for which the rule of this sequence is given in the problem. According to the rule, the fifth term should be $3 \times 2 = 6$, it misread the rule and incorrectly generates $3 \times 3 = 9$.

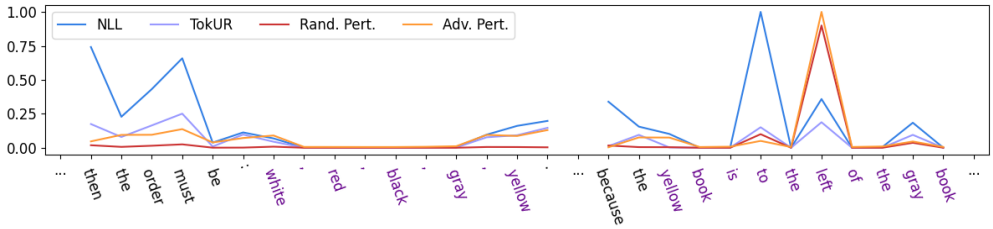


•**Calculation error identified in Qwen.** The model mistakenly calculates '$2 \cdot 2^5 = 32$', where token '3' is successfully identified. In this case adversarial perturbation maintains a uniformly high value, and its reaction to error token is less significant than random perturbation, although it is still in one of the top-3 uncertain tokens across the response.

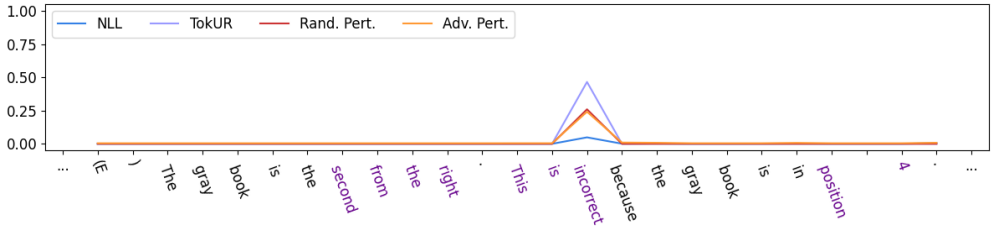


•**Math conceptual error identified in Mistral.** The model need the condition of digits 'multiple of 4', it incorrectly state 'all digits to be even except the units place', which should be replaced by 'last two digits divisible by 4'. Perturbation-based methods successfully flagged critical tokens 'all dig(its)' and 'even'.

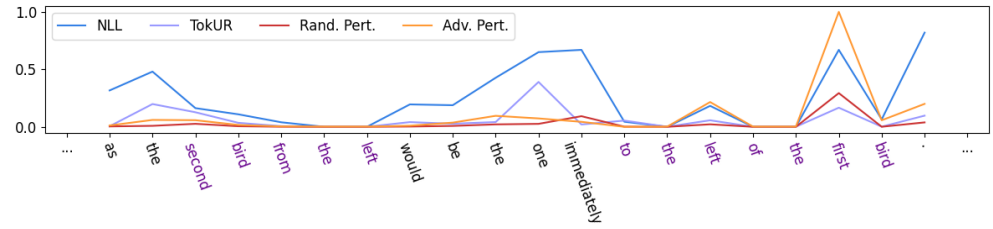
Figure 9: Extra case studies on math reasoning task. Each uncertainty score is min-max normalized over the complete output to fit the figure, and only the critical segments are shown.



• **Premise inconsistency error identified in Llama.** The LLM is tasked to sort 5 books in different colors. In the middle of the reasoning, LLM has already derived the correct order for the book: 'white, red, black, gray, yellow'. But later, in the answer selection stage, it incorrectly output 'the yellow book is to the left of the gray book'(the correct token should be 'right'). For another uncertainty token 'to', there may be various ways to express (e.g. 'on the left side'), which may cause a high NLL and Rand. Pert. score. But the true error token 'left' is successfully identified by Adv. Pert.

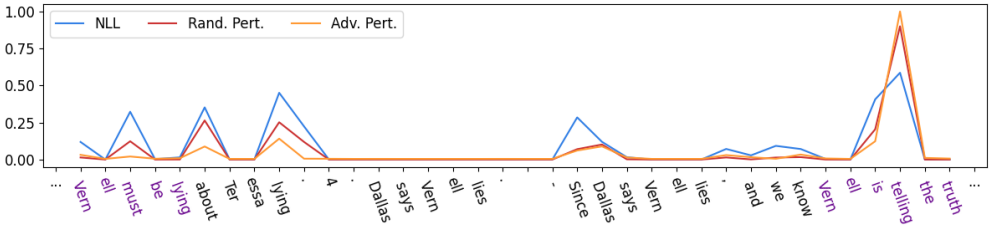


• **Reflection error identified in Qwen.** The LLM is tasked to sort 5 books in different colors. From previous reasoning, it has 'the gray book is in position 4'. When LLM is identifying the options provided by the question, it failed to deduce that 'the gray book is in position 4' is equivalent to 'the gray book is the second from the right', thus it outputs "This is incorrect". Perturbation based methods successfully identify this token in error.

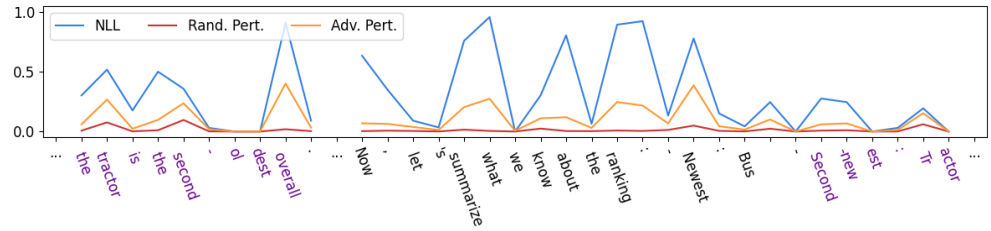


• **Positional inference error identified in Mistral.** The model is asked to sort 5 birds. the error is 'the second bird from the left would be the one immediately to the left of the first bird'. The correct version should be either 'to the right of the first bird' or 'to the left of the third bird', therefore we observed a small peak at the prior token 'left', and a large peak at the latter error token 'first'.

Figure 10: Extra case studies on logical reasoning task. Each uncertainty score is min-max normalized over the complete output to fit the figure, and only the critical segments are shown.



•**A successful case in Llama.** The LLM needs to reason who is lying. At first, 'Vernell must be lying about Teressa lying' is inferred correctly, but later it incorrectly generates 'Vernell is telling the truth'. Our method successfully identifies the critical error token 'telling' which should be 'lying' instead.



•**A failure case in Qwen.** The LLM needs to rank five given vehicles. At first, 'the tractor is the second-oldest' is inferred correctly by the model, but later it incorrectly concludes 'Second newest: Tractor'.

Figure 11: Case studies on logical reasoning task. Each uncertainty score is min-max normalized over the complete output to fit the figure, and only part of tokens are shown.

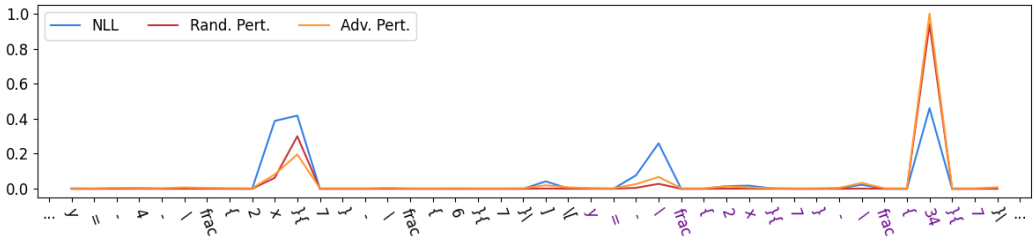


Figure 12: Correct but risky case. The LLM is performing a correct mathematical derivation from $y = -4 - \frac{2x}{7} - \frac{6}{7}$ to $y = -\frac{2x}{7} - \frac{34}{7}$. The token '34' is flagged as uncertain though correct. Re-generation from this step reveals risk score of 0.6, where 0.5 of the risk is due to "$y = -\frac{2x}{7} - \frac{32}{7}$" where model incorrectly generate '32' instead of '34', and 0.1 is because model generate "$y = -\frac{2x}{7} - \frac{38}{7}$", both re-generations show the uncertainty of token '34'.