Reconsidering LLM Uncertainty Estimation Methods in the Wild

Anonymous ACL submission

Abstract

Large Language Model (LLM) Uncertainty Estimation (UE) methods have become crucial tools for detecting hallucinations in recent years. While numerous UE methods have been proposed, most existing studies evaluate them in *isolated* short-form QA settings using threshold-independent metrics such as AUROC or PRR. However, real-world deployment of UE methods introduces several challenges. In this work, we systematically examine four key aspects of deploying UE methods in practical settings. Specifically, we assess (1) the sensitivity of UE methods to decision threshold selection, (2) their robustness to query transformations such as typos, adversarial prompts, and prior chat history, (3) their applicability to long-form generation, and (4) strategies for leveraging multiple UE scores for a single query. Our evaluations on 19 UE methods reveal that most of them are highly sensitive to threshold selection when there is a distribution shift in the calibration dataset. While these methods generally exhibit robustness against previous chat history and typos, they are significantly vulnerable to adversarial prompts. Additionally, while existing UE methods can be adapted for long-form generation through various strategies, there remains considerable room for improvement. Lastly, ensembling multiple UE scores at test time provides a notable performance boost which highlights its potential as a practical improvement strategy.

1 Introduction

Generative Large Language Models (LLMs) have been deployed in various real-world applications, including code copilots, chatbots, and medical assistants (Zhang et al., 2024b; Liu et al., 2024). Their widespread usage has raised significant safety considerations, particularly regarding reliability (Bengio et al., 2025). Despite advancements over the previous wave of language models, these models can still produce incorrect or misleading text, a problem commonly known as *hallucination* or *confabulation* (Ravi et al., 2024).

Detecting hallucinations in LLM outputs is a fundamental challenge, with various approaches such as fact-checkers (Wang et al., 2024), tool-based detectors (Chern et al., 2023), LLM-collaboration-based detectors (Feng et al., 2024), and Uncertainty Estimation (UE) methods (Farquhar et al., 2024). Among these, UE methods are particularly valuable as they operate independently of external resources like Internet searches or other tools, and have shown promising performance across diverse datasets (Vashurin et al., 2025).

Numerous UE methods have been proposed to detect hallucinations (Azaria and Mitchell, 2023; Zhao et al., 2024). However, these are typically tested in isolated short-form QA settings with simple prompts and evaluated using threshold-free metrics like AUROC and PRR (Malinin and Gales, 2021; Duan et al., 2024). Despite their value, the challenges of real-world deployment remain largely unexplored and are crucial for future research.

Motivated by these concerns, we investigate four essential aspects of deploying a UE method in the real-world (wild), as also outlined in Figure 1:

Sensitivity of Decision Threshold: Since the outputs of UE methods are typically continuous, selecting a threshold is necessary to make binary decisions (e.g., hallucination or not). This threshold is calibrated using a specific dataset to meet target performance levels. We explore whether the thresholds selected for UE methods achieve the desired performance in practice, evaluating their stability and effectiveness across different data distributions.

Robustness to Input Transformations: We assess the resilience of UE methods to previously generated context, typos in the prompts, and adversarial prompts designed to confuse UE methods.

Applicability to Long-Form Generations: While many UE methods are proposed and tested for short-form QA, real-world questions often require

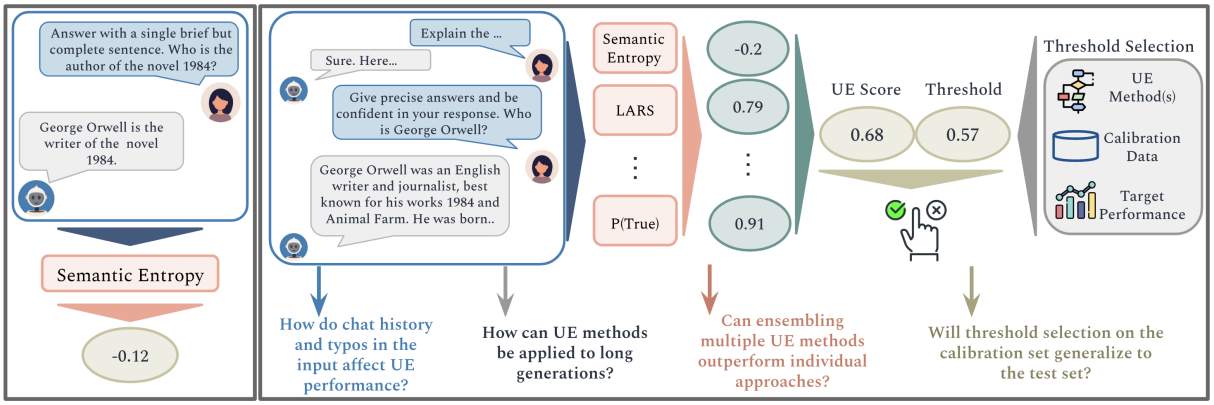

Figure 1: **Left:** Existing pipeline for UE. The uncertainty score is calculated for short-form QA and evaluated using a threshold-free metric such as AUROC. **Right:** Reconsidering LLM uncertainty estimation methods in the wild. We ask four critical questions addressing challenges in deploying UE methods in real-world scenarios.

extended answers containing multiple claims. We examine whether these methods designed for short-form QA can be adapted to long-form generations.

Reconcilability of Diverse UE Scores: UE methods often produce varying judgments for the same input. Ensembling their outputs can enhance performance, potentially surpassing individual methods.

With our comprehensive evaluation of 19 UE methods, our findings across the four investigated aspects can be summarized as follows:

- Most UE methods are highly sensitive to decision threshold selection, particularly when the calibration data distribution differs from the test data distribution.
- The majority of the methods demonstrate resilience to previous context and typos in the prompt, but they exhibit significant performance drops with adversarial prompts.
- UE methods not originally designed for long-form generation can be adapted to this setting through additional steps. However, their effectiveness remains lower compared to their performance in short-form tasks.
- Ensembling multiple UE scores can yield meaningful performance improvements, even when using a very small set of data. Notably, simple ensembling strategies, such as averaging UE scores, can be very effective.

Based on these findings, we encourage researchers to evaluate the sensitivity of their proposed UE methods to threshold selection and input transformations. There is also potential for developing advanced techniques to apply UE methods to long-form generation. Finally, we believe that further exploration of ensembling strategies may unlock even greater performance improvements.

2 Preliminaries

2.1 Uncertainty Estimation of LLMs

Although various Uncertainty Estimation (UE) methods for LLMs have been proposed recently, there is no universally accepted definition of UE in the context of LLMs (Vashurin et al., 2025). Some research formalizes LLM uncertainty by decomposing into *aleatoric* (data) and *epistemic* (model) uncertainties, leveraging LLM sampling distributions (Aichberger et al., 2024; Abbasi-Yadkori et al., 2024). However, many heuristic-based UE methods in the literature do not conform to these theoretical frameworks.

Therefore, we adopt a broad, *practical* definition of UE, following previous works (Jiang et al., 2024; Huang et al., 2024). Formally, an uncertainty estimation method U is defined as a function $\mathrm{U} : \mathcal{V}^* \times \mathcal{V}^* \to \mathbb{R}$, where $\mathcal{V}$ represents the vocabulary, and $\mathcal{V}^*$ denotes all possible token sequences. For a given query x and generated response $\hat{y}$, an effective U should assign a low uncertainty score (indicating higher confidence) if $\hat{y}$ is *reliable* in the given context. In tasks such as factual QA or mathematical reasoning—common evaluation benchmarks for UE methods—reliability refers to the correctness of $\hat{y}$ with respect to the set of ground truth(s) Y. Formally, a desirable U should maximize $\mathbb{E}\left[\mathbb{1}_{\mathrm{U}(x_1,\hat{y}_1)<\mathrm{U}(x_2,\hat{y}_2)} \cdot \mathbb{1}_{\hat{y}_1 \in Y_1 \wedge \hat{y}_2 \notin Y_2}\right]$ where $(x_1, y_1), (x_2, y_2) \sim \mathcal{D}$, with $\mathcal{D}$ being a dataset, $\hat{y}_1 \sim p(\cdot|x_1)$, $\hat{y}_2 \sim p(\cdot|x_2)$ representing the model's sampling distributions.

2.2 Evaluation of UE Methods

As discussed in the previous section, UE methods serve as proxies for predicting the correctness of model-generated responses, producing scores that typically lie within a continuous range. Conse-

quently, their evaluation is commonly performed by setting the correctness of a generation as binary labels (0 or 1)[1], using UE scores as predictions, and computing threshold-free metrics such as AUROC and AUPRC (Kuhn et al., 2023; Vashurin et al., 2025). In addition to these, the Prediction Rejection Ratio (PRR) (Malinin and Gales, 2021) evaluates UE performance by constructing a rejection-precision curve, which measures the precision of the retained (non-rejected) samples at different rejection thresholds based on uncertainty scores. PRR is computed as the area under this curve and is further normalized by the areas under the curves of the best possible (oracle) and random rejection-precision strategies. This normalization makes PRR resilient to label imbalances in the dataset (Malinin and Gales, 2021). PRR ranges from 0.0 (random performance) to 1.0 (perfect performance). In this study, we primarily use PRR as our evaluation metric due to its robustness against variations in data distribution.

2.3 Investigated UE Methods

Throughout this paper, we examine 19 UE methods, categorizing each according to its primary conceptual approach. We identify four distinct categories for this classification:

Probability-Based Methods utilize probabilities of tokens in the generated sequence. *Length-Normalized Scoring (LNS)* (Malinin and Gales, 2021) is the average of the log-probabilities of the generation, while *MARS* (Bakman et al., 2024) computes the weighted-average of that regarding the token importance in answering the question. *LARS* (Yaldiz et al., 2024) trains a small-scale transformer that takes the question, generation tokens, and token probabilities. *Entropy* (Malinin and Gales, 2021) calculates the average of length-normalized scores over a set of sampled generations for the same question. *Semantic Entropy (SE)* (Kuhn et al., 2023) clusters the semantically-similar generations while *SentSAR* and *SAR* (Duan et al., 2024) considers relevancy scores of the sampled generations during entropy calculation.

Internal State-Based Methods make use of the internal states of the LLM, which are only applicable to white-box models. *INSIDE* (Chen et al., 2024) utilize the middle layer activations of the last tokens of multiple generations to the same question. *Attention Score* (Sriramanan et al., 2024) analyses the attention maps of the LLM. *SAPLMA* (Azaria and Mitchell, 2023) trains a classifier whose input is the activations of the last token of the generation.

Output Consistency-Based Methods sample multiple generations to the query, then utilize their pair-wise similarity information, hence usable with black-box models. *Degree Matrix Uncertainty*, *Eccentricity Uncertainty*, *SumEigV* (Lin et al., 2024), and *Kernel Language Entropy (KLE)* (Nikitin et al., 2024) utilize different linear algebra techniques over the pair-wise similarity matrix of the sampled generations. *Degree Matrix-C* and *Eccentricity-C* (Lin et al., 2024) output a generation-specific score for each generation by using the similar ideas in *Eccentricity Uncertainty* and *Degree Matrix Uncertainty*. *Self-Detection* (Zhao et al., 2024) paraphrases the question and analyses the similarity of the responses to the paraphrased questions.

Self-Checking Methods query the LLM itself about the uncertainty of the generation. *P(true)* (Kadavath et al., 2022) asks if the response is true by providing the question, sampled generations, and the answer. *Verbalized Confidence* (Tian et al., 2023) prompts the LLM to assign a confidence score to the response between 0 and 1.

It is important to note that only LARS and SAPLMA are supervised techniques, requiring labeled QA data, whereas all other methods are unsupervised. For brevity, detailed explanations of these methods are provided in Appendix C.

3 Sensitivity of Decision Threshold

3.1 Problem Statement

UE methods typically produce outputs in a continuous range. However, integrating a UE method into a real-world application requires making discrete decisions, such as whether to accept or reject a generated response. The sensitivity of this binary decision can vary depending on the application. Consequently, such scenarios require selecting an appropriate threshold to achieve the desired performance for decision-making.

Determining a threshold t for a target application requires a labeled calibration dataset $\mathcal{D}_{cal}$. Using this dataset and a desired metric M with a target performance level m^*, a threshold t is randomly picked from the set $\{t : \mathrm{M}(\mathrm{U}, \mathcal{D}_{cal}, t) = m^*\}$.

This threshold is then applied during testing. The key question is whether the desired performance m^* is maintained at test time. If not, two main

[1]We utilize GPT-4o-mini as correctness evaluator, using the query, generated response, and ground truth(s) (Lin et al., 2024; Bakman et al., 2024)

factors may contribute: (1) a distribution shift between the calibration and test data, or (2) the UE method itself being sensitive to such scenarios. To investigate this phenomenon, we examine 19 UE methods (listed in Section 2.3) across two tasks and varying levels of calibration-test distribution shifts.

3.2 Experimental Design

Models We evaluate UE methods on two recent models: Llama-3-8B (AI@Meta, 2024) and GPT-4o-mini (OpenAI, 2023).

Datasets We use TriviaQA (Joshi et al., 2017) and NaturalQA (Kwiatkowski et al., 2019) as closed-book QA datasets and GSM8K (Cobbe et al., 2021) as mathematical reasoning dataset in the experiments. We use 1000 samples for the test set and 500 samples for the calibration dataset. All experiments are conducted 5 times with different seeds and the average performance is provided.

Metric Different applications require varying precision-recall trade-offs, so we introduce a metric to assess threshold generalization at test time. For each target recall $r^* \in [0, 1]$, we determine an optimal threshold t using a calibration set. Hallucinations (incorrect generations) are class 1, and correct answers are class 0 which makes recall the proportion of hallucinations correctly identified by the UE method.

To assess threshold generalization, we measure the deviation $|r^* - r|$, where r is the recall achieved on the test set using threshold t. By averaging these deviations over a set $\mathcal{R}$ of recall values of interest, *Average Recall Error (ARE)* is defined as:

$$\text{ARE} = \frac{1}{|\mathcal{R}|} \sum_{r_i \in \mathcal{R}} |r_i^* - r_i|.$$

In our experiments, we set $\mathcal{R}$ to span the full recall range from 0 to 1.0., with increments of 0.001.

Distribution Shift Simulation We systematically examine how distribution shifts between calibration and test data impact threshold selection performance through two experimental setups. First, we use TriviaQA as the test data, calibrating with TriviaQA for an in-domain setting, NaturalQA for a same-task distribution shift, and GSM8K for an out-of-domain scenario. Second, we test with GSM8K and calibrate separately with TriviaQA and GSM8K, where GSM8K is in-domain and TriviaQA is out-of-domain.

3.3 Results and Discussion

The ARE results for TriviaQA are presented in Table 1. The findings indicate that the majority of UE methods achieve a low ARE (<0.05) when the threshold is calibrated on a separate subset of TriviaQA, with the exception of Verbalized Confidence and Self-Detection. However, as expected, the error rate increases with greater data distribution shifts, making GSM8K calibration the most erroneous when UE methods are tested on TriviaQA.

Probability-based and output consistency-based methods generally outperform internal state-based and self-checking methods. However, only MARS, Semantic Entropy, and Eccentricity consistently achieve low error across calibration datasets, while all others exceed 0.10 ARE in at least one setting.

These results highlight the need to align the calibration data distribution with the test (deployment) environment to ensure reliable binary decision-making using UE methods. Furthermore, we encourage researchers to test their proposed UE methods under distribution shift conditions, particularly for threshold sensitivity. Robustness to such shifts is a highly desirable property, as it reduces reliance on an optimal calibration dataset. Lastly, the ARE results for GSM8K, provided in Appendix D.1, aligns with the findings observed in TriviaQA.

	Llama3-8b			**GPT-4o-mini**		
Calib. Dataset	TrivQA	NQA	GSM	TrivQA	NQA	GSM
LNS	0.030	0.093	0.103	0.055	0.035	0.049
MARS	0.035	0.025	0.077	0.050	0.046	0.040
Entropy	0.032	0.103	0.101	0.072	0.048	0.066
SE	0.035	0.065	0.073	0.060	0.029	0.045
SentSAR	0.041	0.105	0.123	0.074	0.041	0.093
SAR	0.028	0.059	0.107	0.068	0.023	0.077
LARS	0.035	0.117	0.130	0.048	0.125	0.289
DegMat	0.041	0.033	0.169	0.051	0.051	0.142
DegMat-C	0.038	0.030	0.141	0.058	0.049	0.126
SumEigV	0.042	0.035	0.191	0.051	0.053	0.165
KLE	0.047	0.062	0.173	0.076	0.056	0.115
Eccent	0.040	0.037	0.069	0.057	0.049	0.050
Eccent-C	0.040	0.039	0.098	0.063	0.048	0.051
Self-D.	0.082	0.086	0.113	0.110	0.127	0.096
P(True)	0.035	0.087	0.255	0.123	0.163	0.200
Verb. C.	0.172	0.182	0.280	0.084	0.131	0.142
Atten. S.	0.027	0.027	0.261	-	-	-
INSIDE	0.040	0.096	0.295	-	-	-
SAPLMA	0.046	0.029	0.142	-	-	-

Table 1: ARE of UE methods when the threshold is calibrated on various datasets and tested on TriviaQA.

4 Robustness to Input Transformations

4.1 Problem Statement

Previous UE works primarily evaluate their methods in *isolated* environments, where a question is presented to the model using a simple benign prompt, and the model's response is directly sampled. However, in real-world applications, inputs can arrive in various forms. We expect a robust UE method's performance should not be affected much under these various input forms.

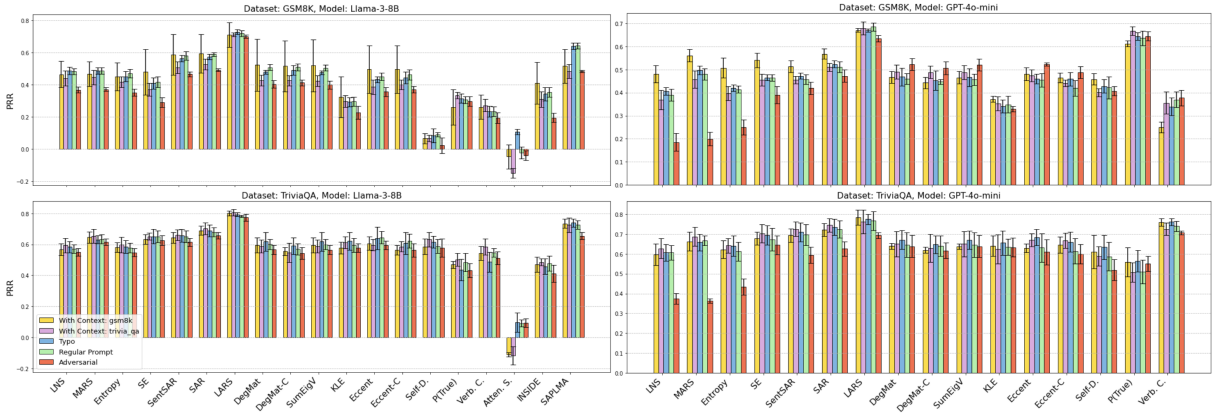

Figure 2: PRR performance of UE methods on the GSM8K and TriviaQA datasets, evaluated under a regular prompt (no transformation) and various input transformations, including adding context, typos, and adversarial prompts.

More formally, we apply a transformation function $\mathcal{T}$ to a query x such that $\mathcal{T}(x)$ preserves the same ground truth set Y as the original query. This ensures that the transformation does not alter the fundamental meaning of the query. Let $\mathcal{D}^* := \{(\mathcal{T}(x), Y) \mid (x, Y) \in \mathcal{D}\}$ represent the transformed version of the original dataset $\mathcal{D}$. A robust UE method U should exhibit similar performance on both $\mathcal{D}$ and $\mathcal{D}^*$. However, since input transformations can influence the model's internal computations—on which UE methods ultimately rely—a non-robust U may experience performance degradation under different transformations.

We investigate the robustness of UE methods across three specific transformations: **(1) Contextual:** This transformation appends previous chat history (context) to the input. This scenario commonly occurs in chatbot applications, where users may ask multiple questions within the same session. To evaluate this case, we prepend previous chat x_{prev} to the original query x in the dataset: $\mathcal{T}_{\text{context}}(x) = x_{\text{prev}} + x$, where $+$ denotes the concatenation operation. **(2) Typo:** In real-world applications, input queries often contain noise, with typos being a common form of such noise. To evaluate how UE methods handle noisy inputs, we introduce synthetic typos into the query, defining the transformation as: $\mathcal{T}_{\text{typo}}(x) = x_{\text{typo}}$. **(3) Adversarial:** We design an adversarial prompt that aims to confuse UE methods, causing their performance to degrade on $\mathcal{D}^*$. This can be viewed as an adversarial prompt injection attack, targeting UE methods specifically. Formally, the transformation is expressed as: $\mathcal{T}_{\text{adv}}(x) = p_{\text{adv}} + x$, where p_{adv} is the adversarial prompt.

4.2 Experimental Design

We use Llama-3-8B and GPT-4o-mini as the base models and evaluate them on 1,000 samples from the test sets of TriviaQA and GSM8K, as described in Section 3. We measure all methods' performance by PRR as described in Section 2.2. All experiments are conducted 5 times, and we plot both the mean and standard deviation of the results.

Context Experiments To simulate chat history, we prepend three prior question-sampled response pairs to each query in two scenarios: *1. Similar-context:* The prior questions are of the same type as the question (e.g., TriviaQA). *2. Dissimilar-context:* The prior questions are from a different domain (e.g., GSM8K math before a TriviaQA question).

Typo Experiments To simulate typos, we randomly replace, swap, erase, or insert a single character with uniform probability. We also test two-character perturbations to evaluate the effects of increased noise.

Adversarial Experiments Designing an adversarial prompt is non-trivial.We insert a *confidence booster* phrase, hypothesizing it may induce overconfidence in model responses, impacting log probabilities, outputs,and internal states and potentially misleading UE methods. For Llama-3-8B experiments, we use the following prompt:

> "Be confident in your responses. Avoid hesitation or uncertainty. Provide clear and direct answers with conviction."

For GPT-4o-mini, we generate a similar prompt using an automated search inspired by Zhou et al. (2023). The specific prompt used with the details of the search process, is provided in Appendix E.1.

4.3 Results and Discussion

The results, Figure 2, suggest that previous chat history has little to no negative effect on the performance of most UE methods —except for Attention Score— compared to standard prompting without

context. In some cases, such as GSM8K with GPT-4o-mini, including similar chat history appears to induce an in context learning-like effect, boosting the performance of probability-based UE methods.

The typo experiments indicate that most UE methods are highly resilient to this input noise. This robustness persists even when the number of typos in a single query is increased to two, as shown in Appendix D.2.

Finally, results indicate that the *confidence booster* prompt injection acts as an adversarial prompt, reducing performance across various datasets, particularly affecting probability-based methods in GPT-4o-mini. However, output-consistency-based methods show more resilience to this adversarial prompt than other approaches. The instability of UE methods to prompt transformations is also observed in previous works (Mahaut et al., 2024). Although some performance variations are expected, a robust UE method should not suffer significant degradation due to prompt changes. Therefore, we recommend that future UE methods undergo systematic prompt variation testing to assess their robustness.

5 Applicability to Long-Form Generations

5.1 Problem Statement

Most UE methods are evaluated on short-form, open-ended QA. For instance, questions such as "Who is the author of the novel 1984?" can be answered with a single sentence, and a single score suffices for an uncertainty assessment. However, in some real-world applications, questions like "Who is George Orwell?" often require long-form responses. These responses may contain multiple claims, some of which are correct while others may be hallucinated. Consequently, assigning a single uncertainty score to the entire response is both impractical and undesirable, as it fails to capture the correctness of individual claims within the text.

To address this issue, long-form outputs are typically decomposed into sentences, each conveying a distinct claim (Farquhar et al., 2024; Wei et al., 2024b; Fadeeva et al., 2024; Zhang et al., 2024a; Manakul et al., 2023; Min et al., 2023). Formally, the decomposition function can be defined as $\mathrm{D} : \mathcal{V}^* \rightarrow 2^{\mathcal{V}^*}$, taking a long generation $\hat{y}$ and returning a set of claims $\mathbf{C} = \{c_i\}_{i=1}^{C}$. After decomposition, each claim c_i is evaluated individually. Recently, several UE methods have been developed specifically for this claim-level uncertainty problem in long-form generations (Fadeeva et al., 2024; Farquhar et al., 2024; Zhang et al., 2024a; Jiang et al., 2024), however, most existing UE methods are not directly applicable for assessing uncertainty at the claim level in long form generations(Vashurin et al., 2025). Consequently, effectively applying these methods to segmented claims continues to pose challenges.

In this section, we propose a set of strategies designed to adapt existing UE methods to assess claim-level uncertainty. A strategy function takes the original query x, a specific claim c_i, and a UE function U, and returns an uncertainty score for the claim c_i. Formally, a strategy function can be defined as $\mathrm{S} : \mathcal{V}^* \times \mathcal{V}^* \times \mathrm{U} \rightarrow \mathbb{R}$.

5.2 Experimental Design

Decomposing the Long Generation Following previous research, we employ an LLM to decompose long text into claims (Farquhar et al., 2024; Fadeeva et al., 2024; Min et al., 2023). This decomposition can be applied at different levels of granularity. For instance, Wei et al. (2024b) segments the generation into paragraphs, whereas Fadeeva et al. (2024); Min et al. (2023) breaks down text into sentences prior to decomposition. We, similar to Farquhar et al. (2024), apply decomposition to the entire generation. However, we introduce an additional decomposition step for each claim produced in the initial phase, as the model often generates sentences that contain more than one claim during the first decomposition step.

Proposed Strategies to Apply UE to Claims An uncertainty estimation method U requires two inputs: the query and the response. To effectively employ a UE method within a strategy function S, we need to define what constitutes the query and response. We introduce three strategies to enable the application of existing UE methods for claim-level uncertainty estimation:

1. Naive Application: The primary input x serves as the query, and the claim c_i is used as the response for the UE method: $\mathrm{S}(x, c_i, \mathrm{U}) = \mathrm{U}(x, c_i)$.

2. Question Generation (QG): For the given claim c_i, a specific question for that claim x' is generated, where the claim itself acts as the answer. Then, the generated question x' and the claim c_i are inputted to the UE method: $\mathrm{S}(x, c_i, \mathrm{U}) = \mathrm{U}(x', c_i)$.

3. Question Answer Generation (QAG): A question x' is generated for the claim such that the claim serves as the answer. However, instead of using the

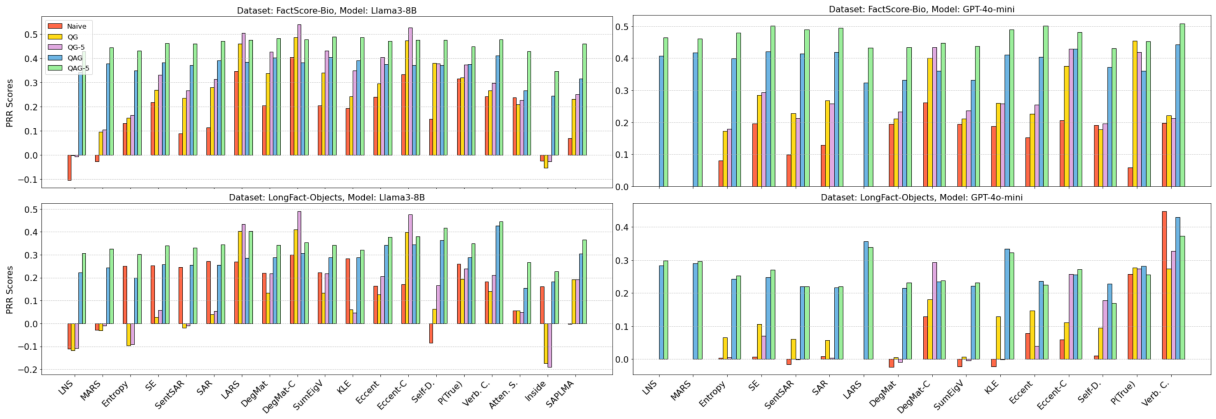

Figure 3: PRR scores for UE methods applied to long-form generation. 'QG-5' and 'QAG-5' indicate that five questions per claim are generated and then aggregated (averaged) to assess each claim's uncertainty.

claim directly, a new response y' is generated by the model in response to x' to make the claim come from the actual sampling distribution of the model to potentially estimate the uncertainty better. The UE method $\mathrm{U}(x', y')$ is called if y' semantically equivalent with c_i. If not, a high uncertainty score is assigned to the claim:

$$\mathrm{S}(x, c_i, \mathrm{U}) = \begin{cases} \mathrm{U}(x', y') & \text{if } c_i \text{ aligns with } y', \\ \infty & \text{otherwise.} \end{cases}$$

To further improve the last two strategies, multiple questions can be generated for each claim. For each question, the processes outlined in the strategies are applied, resulting in a series of UE scores for the same claim. To combine these scores into a single assessment, we can aggregate them by taking the minimum, maximum, or average.

Models, Datasets, and Metrics We employ GPT-4o-mini and Llama3-8B as our base models, using GPT-4o-mini consistently for text decomposition across all models. For question and answer generation, the same base model generating the main response is utilized. We use two long-form QA datasets: FactScore-Bio (Min et al., 2023), containing biography questions from Wikipedia, and LongFact-Objects (Wei et al., 2024b), covering 38 diverse topics. Experiments are conducted on a random sample of 50 questions from each dataset. For evaluation, we collect the UE scores from all claims as predictions and follow the SAFE (Wei et al., 2024b) algorithm to set ground truths, then calculate the PRR score. More details on this section are provided in Appendix E.3.

5.3 Results and Discussion

Our evaluation (Figure 3) shows that UE methods not designed for long-form generation can be adapted using decomposition and strategies from Section 5.2. Results suggest that QAG outperforms other strategies, while Naive Application is the least effective. Besides, generating claim-specific questions (QG, QAG) improves uncertainty estimation over relying on the original query (Naive), and using model-generated answers (QAG) generally is more effective than assessing claims directly (QG).

For both QG and QAG, generating multiple questions consistently enhances UE performance, with only a few exceptions. This may indicate that multiple inquiries can capture uncertainty more effectively, especially when there are various ways to form a question for a specific claim. Among the aggregation methods we evaluated (minimum, maximum, and average), averaging is consistently the most effective, as shown in Appendix D.3.

When comparing different question domains, higher PRR scores are observed in FactScore-Bio compared to LongFact-Objects dataset which has broader subjects such as chemistry, gaming, and geography. Notably, we observe a non-negligible performance drop of UE methods in PRR in long-form generation compared to short-form QA such as TriviaQA. This highlights there is still significant room for improvement in applying these methods to long-form generation.

6 Reconcilability of Diverse UE Scores

6.1 Problem Statement

UE methods use diverse algorithms to estimate uncertainty which leads to different outputs for the same input $(x, \hat{y})$. We leverage this diversity by ensembling multiple UE methods during inference to improve performance. Formally, given K UE methods $(\mathrm{U}_1, \mathrm{U}_2, \ldots, \mathrm{U}_K)$, their outputs for $(x, \hat{y})$ form the score vector $\mathbf{s} = (s_1, s_2, \ldots, s_K)$. We aggregate these scores using an ensemble function $\mathcal{E} : \mathbb{R}^K \to \mathbb{R}$. Since UE methods output in different numerical ranges, we assume access to a small

supervised calibration dataset $\mathcal{D}_{\text{cal}}$ of 100 samples for normalization.

6.2 Experimental Design

We conduct experiments using LlaMA-3-8B and GPT-4o-mini, evaluating the PRR performance of both individual UE methods and ensembling strategies on TriviaQA and GSM8K. Given that we investigate $K = 19$ UE methods, the number of possible ensemble combinations is $2^K - K - 1$, which is computationally infeasible. Therefore, instead of exhaustively evaluating all possible ensembles, we focus on ensembling all methods together and compare its performance against the most effective individual UE method.

Ensembling Strategies We ensemble in two stages: preprocessing raw scores $\mathbf{s}$ and combining them with $\mathcal{E}$. For preprocessing, we use three strategies: (1) No processing, using raw scores. (2) Standard normalization, where $s_i' = \frac{s_i - \mu_i}{\sigma_i}$, with mean μ_i and standard deviation σ_i computed from the calibration set $\mathcal{D}_{cal}$. (3) Isotonic Regression calibration (Han et al., 2017), which maps scores to probabilities in the range [0,1] which approximates correctness likelihood. Unlike normalization, which only requires inputs x, calibration also requires ground truth y in $\mathcal{D}_{cal}$.

For ensembling, we investigate 7 different strategies. The first two are simple aggregation methods: taking the minimum and maximum of $\mathbf{s}$. We also consider averaging methods, including a simple mean $\frac{1}{K}\sum_{i=1}^{K} s_i$ and a weighted average $\sum_{i=1}^{K} w_i s_i$. Here w_i represents the PRR performance of uncertainty estimator U_i on $\mathcal{D}_{cal}$. Another approach is a voting-based method, where we count the number of scores exceeding a threshold t: $\sum_{i=1}^{K} \mathbb{1}_{s_i > t}$. Finally, we explore supervised ensembling approaches by treating the vector $\mathbf{s}$ as a feature vector and training models such as a linear model and a decision tree using the calibration dataset $\mathcal{D}_{cal}$.

6.3 Results and Discussion

The results of the ensembling experiments are presented in Table 2. Our findings suggest that even with 100 samples D_{cal}, ensembling strategies can achieve gains of up to 0.06 average PRR score compared to the most performant individual UE method. As expected, directly combining raw UE scores without normalization or calibration is ineffective due to the varying scales of different UE methods. However, applying normalization and calibration significantly improves ensembling performance, even with simple strategies such as averaging all UE scores. For supervised approaches, linear models with normalized or calibrated inputs consistently outperform the best individual UE method. In contrast, decision tree generally fails to provide competitive ensembling performance. Also, we repeat the experiments using only unsupervised UE methods (see Appendix D.4), which further improves performance over the best unsupervised method. We argue that developing orthogonal UE methods to existing UE methods may be promising, as their combination with existing techniques may yield superior performance. Additionally, exploring novel ensembling strategies specifically for UE methods could further improve results.

		TriviaQA		GSM8K	
		Llama	GPT	Llama	GPT
Best single		0.78	0.77	0.72	0.69
Raw	**Max**	0.09	0.66	-0.02	0.49
	Min	0.56	0.64	0.28	0.35
	Mean	0.66	0.76	0.44	0.55
	W-mean	0.66	0.76	0.48	0.56
	Linear	**0.82**	0.72	**0.73**	0.68
Normalized	**Max**	0.76	**0.83**	0.54	0.64
	Min	0.45	0.70	0.41	0.63
	Mean	**0.78**	**0.83**	0.62	0.67
	W-mean	**0.79**	**0.83**	0.66	**0.69**
	Linear	**0.80**	**0.77**	**0.73**	**0.71**
Calibrated	**Max**	0.77	**0.79**	0.65	0.64
	Min	0.63	0.59	0.56	0.63
	Mean	**0.79**	**0.80**	0.68	0.62
	W-mean	0.75	**0.80**	0.71	0.65
	Linear	**0.82**	**0.77**	**0.75**	**0.72**
	Voting	0.77	0.74	0.66	0.64
	D.Tree	0.46	0.47	0.44	0.43

Table 2: PRR scores of different ensembling strategies over 19 UE methods.

7 Conclusion

We conducted a comprehensive evaluation of 19 UE methods across four key challenges in real-world deployment. Our findings reveal that most UE methods are highly sensitive to decision threshold selection and, while resilient to typos and context, remain vulnerable to adversarial prompts. Additionally, existing UE methods can be adapted for long-form generation, though their effectiveness remains limited. Finally, ensembling multiple UE methods significantly enhances performance, even with simple strategies. Future research should focus on improving UE robustness to threshold selection and prompt variations, developing more effective strategies for long-form generation, and exploring advanced ensembling techniques to maximize the performance.

8 Limitations

While this study highlights key vulnerabilities and future opportunities for UE methods, our experiments are limited to two models because of the computational limitations: LLaMA-3-8B and GPT-4o-mini. Future work should verify these findings on other state-of-the-art models to assess broader applicability. Additionally, the experimental framework introduced in this paper can be extended to evaluate other UE methods beyond the 19 investigated in this study.

References

Yasin Abbasi-Yadkori, Ilja Kuzborskij, András György, and Csaba Szepesvari. 2024. To believe or not to believe your LLM: Iterativeprompting for estimating epistemic uncertainty. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Lukas Aichberger, Kajetan Schweighofer, and Sepp Hochreiter. 2024. Rethinking uncertainty estimation in natural language generation. *Preprint*, arXiv:2412.15176.

AI@Meta. 2024. Llama 3 model card.

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

Yavuz Faruk Bakman, Duygu Nur Yaldiz, Baturalp Buyukates, Chenyang Tao, Dimitrios Dimitriadis, and Salman Avestimehr. 2024. MARS: Meaning-aware response scoring for uncertainty estimation in generative LLMs. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7752–7767, Bangkok, Thailand. Association for Computational Linguistics.

Yoshua Bengio, Sören Mindermann, Daniel Privitera, Tamay Besiroglu, Rishi Bommasani, Stephen Casper, Yejin Choi, Philip Fox, Ben Garfinkel, Danielle Goldfarb, Hoda Heidari, Anson Ho, Sayash Kapoor, Leila Khalatbari, Shayne Longpre, Sam Manning, Vasilios Mavroudis, Mantas Mazeika, Julian Michael, Jessica Newman, Kwan Yee Ng, Chinasa T. Okolo, Deborah Raji, Girish Sastry, Elizabeth Seger, Theodora Skeadas, Tobin South, Emma Strubell, Florian Tramèr, Lucia Velasco, Nicole Wheeler, Daron Acemoglu, Olubayo Adekanmbi, David Dalrymple, Thomas G. Dietterich, Edward W. Felten, Pascale Fung, Pierre-Olivier Gourinchas, Fredrik Heintz, Geoffrey Hinton, Nick Jennings, Andreas Krause, Susan Leavy, Percy Liang, Teresa Ludermir, Vidushi Marda, Helen Margetts, John McDermid, Jane Munga, Arvind Narayanan, Alondra Nelson, Clara Neppel, Alice Oh, Gopal Ramchurn, Stuart Russell, Marietje Schaake, Bernhard Schölkopf, Dawn Song, Alvaro Soto, Lee Tiedrich, Gaël Varoquaux, Andrew Yao, Ya-Qin Zhang, Fahad Albalawi, Marwan Alserkal, Olubunmi Ajala, Guillaume Avrin, Christian Busch, André Carlos Ponce de Leon Ferreira de Carvalho, Bronwyn Fox, Amandeep Singh Gill, Ahmet Halit Hatip, Juha Heikkilä, Gill Jolly, Ziv Katzir, Hiroaki Kitano, Antonio Krüger, Chris Johnson, Saif M. Khan, Kyoung Mu Lee, Dominic Vincent Ligot, Oleksii Molchanovskyi, Andrea Monti, Nusu Mwamanzi, Mona Nemer, Nuria Oliver, José Ramón López Portillo, Balaraman Ravindran, Raquel Pezoa Rivera, Hammam Riza, Crystal Rugege, Ciarán Seoighe, Jerry Sheehan, Haroon Sheikh, Denise Wong, and Yi Zeng. 2025. International ai safety report. *Preprint*, arXiv:2501.17805.

Chao Chen, Kai Liu, Ze Chen, Yi Gu, Yue Wu, Mingyuan Tao, Zhihang Fu, and Jieping Ye. 2024. INSIDE: LLMs' internal states retain the power of hallucination detection. In *The Twelfth International Conference on Learning Representations*.

I Chern, Steffi Chern, Shiqi Chen, Weizhe Yuan, Kehua Feng, Chunting Zhou, Junxian He, Graham Neubig, Pengfei Liu, et al. 2023. Factool: Factuality detection in generative ai–a tool augmented framework for multi-task and multi-domain scenarios. *arXiv preprint arXiv:2307.13528*.

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

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

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

Sebastian Farquhar, Jannik Kossen, Lorenz Kuhn, and Yarin Gal. 2024. Detecting hallucinations in large language models using semantic entropy. *Nature*, 630(8017):625–630.

Shangbin Feng, Weijia Shi, Yike Wang, Wenxuan Ding, Vidhisha Balachandran, and Yulia Tsvetkov. 2024. Don't hallucinate, abstain: Identifying LLM knowledge gaps via multi-LLM collaboration. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14664–14690, Bangkok, Thailand. Association for Computational Linguistics.

Qiyang Han, Tengyao Wang, Sabyasachi Chatterjee, and Richard J. Samworth. 2017. Isotonic regression in general dimensions. *The Annals of Statistics*.

Xinmeng Huang, Shuo Li, Mengxin Yu, Matteo Sesia, Hamed Hassani, Insup Lee, Osbert Bastani, and Edgar Dobriban. 2024. Uncertainty in language models: Assessment through rank-calibration. *Preprint*, arXiv:2404.03163.

Mingjian Jiang, Yangjun Ruan, Prasanna Sattigeri, Salim Roukos, and Tatsunori Hashimoto. 2024. Graph-based uncertainty metrics for long-form language model generations. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

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

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, Scott Johnston, Sheer El-Showk, Andy Jones, Nelson Elhage, Tristan Hume, Anna Chen, Yuntao Bai, Sam Bowman, Stanislav Fort, Deep Ganguli, Danny Hernandez, Josh Jacobson, Jackson Kernion, Shauna Kravec, Liane Lovitt, Kamal Ndousse, Catherine Olsson, Sam Ringer, Dario Amodei, Tom Brown, Jack Clark, Nicholas Joseph, Ben Mann, Sam McCandlish, Chris Olah, and Jared Kaplan. 2022. Language models (mostly) know what they know. *Preprint*, arXiv:2207.05221.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. 2023. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. In *The Eleventh International Conference on Learning Representations*.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Moxin Li, Wenjie Wang, Fuli Feng, Fengbin Zhu, Qifan Wang, and Tat-Seng Chua. 2024. Think twice before trusting: Self-detection for large language models through comprehensive answer reflection. *Preprint*, arXiv:2403.09972.

Zhen Lin, Shubhendu Trivedi, and Jimeng Sun. 2024. Generating with confidence: Uncertainty quantification for black-box large language models. *Transactions on Machine Learning Research*.

Lei Liu, Xiaoyan Yang, Junchi Lei, Yue Shen, Jian Wang, Peng Wei, Zhixuan Chu, Zhan Qin, and Kui Ren. 2024. A survey on medical large language models: Technology, application, trustworthiness, and future directions. *Preprint*, arXiv:2406.03712.

Matéo Mahaut, Laura Aina, Paula Czarnowska, Momchil Hardalov, Thomas Müller, and Lluis Marquez. 2024. Factual confidence of LLMs: on reliability and robustness of current estimators. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4554–4570, Bangkok, Thailand. Association for Computational Linguistics.

Andrey Malinin and Mark Gales. 2021. Uncertainty estimation in autoregressive structured prediction. In *International Conference on Learning Representations*.

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

Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2023. 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*, pages 12076–12100, Singapore. Association for Computational Linguistics.

Alexander V Nikitin, Jannik Kossen, Yarin Gal, and Pekka Marttinen. 2024. Kernel language entropy: Fine-grained uncertainty quantification for LLMs from semantic similarities. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

OpenAI. 2023. GPT-4 Technical Report. *Preprint*, arXiv:2303.08774.

Selvan Sunitha Ravi, Bartosz Mielczarek, Anand Kannappan, Douwe Kiela, and Rebecca Qian. 2024. Lynx: An open source hallucination evaluation model. *Preprint*, arXiv:2407.08488.

Gaurang Sriramanan, Siddhant Bharti, Vinu Sankar Sadasivan, Shoumik Saha, Priyatham Kattakinda, and Soheil Feizi. 2024. LLM-check: Investigating detection of hallucinations in large language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

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

Roman Vashurin, Ekaterina Fadeeva, Artem Vazhentsev, Lyudmila Rvanova, Akim Tsvigun, Daniil Vasilev, Rui Xing, Abdelrahman Boda Sadallah, Kirill Grishchenkov, Sergey Petrakov, Alexander Panchenko, Timothy Baldwin, Preslav Nakov, Maxim Panov, and Artem Shelmanov. 2025. Benchmarking uncertainty quantification methods for large language models with lm-polygraph. *Preprint*, arXiv:2406.15627.

Yuxia Wang, Revanth Gangi Reddy, Zain Muhammad Mujahid, Arnav Arora, Aleksandr Rubashevskii, Jiahui Geng, Osama Mohammed Afzal, Liangming Pan, Nadav Borenstein, Aditya Pillai, Isabelle Augenstein, Iryna Gurevych, and Preslav Nakov. 2024. Factcheck-bench: Fine-grained evaluation benchmark for automatic fact-checkers. *Preprint*, arXiv:2311.09000.

Jason Wei, Nguyen Karina, Hyung Won Chung, Yunxin Joy Jiao, Spencer Papay, Amelia Glaese, John Schulman, and William Fedus. 2024a. Measuring short-form factuality in large language models. *Preprint*, arXiv:2411.04368.

Jerry Wei, Chengrun Yang, Xinying Song, Yifeng Lu, Nathan Zixia Hu, Jie Huang, Dustin Tran, Daiyi Peng, Ruibo Liu, Da Huang, Cosmo Du, and Quoc V Le. 2024b. Long-form factuality in large language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Duygu Nur Yaldiz, Yavuz Faruk Bakman, Baturalp Buyukates, Chenyang Tao, Anil Ramakrishna, Dimitrios Dimitriadis, Jieyu Zhao, and Salman Avestimehr. 2024. Do not design, learn: A trainable scoring function for uncertainty estimation in generative llms. *Preprint*, arXiv:2406.11278.

Caiqi Zhang, Fangyu Liu, Marco Basaldella, and Nigel Collier. 2024a. LUQ: Long-text uncertainty quantification for LLMs. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 5244–5262, Miami, Florida, USA. Association for Computational Linguistics.

Kechi Zhang, Jia Li, Ge Li, Xianjie Shi, and Zhi Jin. 2024b. CodeAgent: Enhancing code generation with tool-integrated agent systems for real-world repo-level coding challenges. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13643–13658, Bangkok, Thailand. Association for Computational Linguistics.

Yukun Zhao, Lingyong Yan, Weiwei Sun, Guoliang Xing, Chong Meng, Shuaiqiang Wang, Zhicong Cheng, Zhaochun Ren, and Dawei Yin. 2024. Knowing what LLMs DO NOT know: A simple yet effective self-detection method. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 7051–7063, Mexico City, Mexico. Association for Computational Linguistics.

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2023. Large language models are human-level prompt engineers. In *The Eleventh International Conference on Learning Representations*.

A Related Works

To the best of our knowledge, no prior work has explicitly investigated UE methods for generative LLMs in real-world, wild settings. The most similar work is Vashurin et al. (2025), which benchmarks various UE methods across multiple datasets. However, their evaluation setup follows the conventional framework used in prior studies (Lin et al., 2024; Kuhn et al., 2023) and does not investigate, reliability of threshold selection, input transformations, and ensembling. Although Vashurin et al. (2025) evaluate some UE methods on long-form generations, they only consider methods inherently designed for the long-form setting. In contrast, we introduce novel strategies to adapt UE methods that were originally designed for short-form settings to long-form generation. Another relevant study is Mahaut et al. (2024), which assesses the reliability of uncertainty estimation methods under specific input transformations, namely paraphrasing and translation into different languages. Their findings reveal performance inconsistencies similar to those observed in our input transformation experiments in Section 4.

B Further Discussions on the Definition of Uncertainty Estimation of LLMs

In addition to the definition of UE methods in LLM at Section 2.1, an uncertainty method should rely on the model itself, utilizing elements such as the model's internals, log probabilities, or outputs. A hallucination detection method that relies on external sources, such as the Internet or external documents, does not fall within the category of uncertainty estimation (Chern et al., 2023).

Furthermore, previous definitions often overlook the fact that $\hat{y}$ is not just any possible token sequence but rather the model's sampled generation. In the evaluation of UE methods, they generate $\hat{y}$ and estimate uncertainty $\mathrm{U}(x, \hat{y})$ (Lin et al., 2024; Kuhn et al., 2023).

Lastly, some methods, such as Semantic Entropy (Kuhn et al., 2023), produce an uncertainty score for a given query x without being specific to any particular sampled generation. These methods assign a query-level uncertainty score, which can still serve as a proxy for the uncertainty of the model's sampled generations. While some previous works (Lin et al., 2024) distinguish between methods that assign scores to individual sampled generations and those that provide query-level uncertainty scores, the latter still fits within the broad definition of UE we adopt, where $\mathrm{U}(x, \hat{y}_1) = \mathrm{U}(x, \hat{y}_2)\ \forall\ \hat{y}_1, \hat{y}_2$. Therefore, we follow prior works (Duan et al., 2024; Vashurin et al., 2025; Yaldiz et al., 2024) and do not make this distinction in our experiments.

C Investigated Uncertainty Estimation Methods

In this section, we explain the investigated UE methods with our implementation details.

C.1 Probability-Based Methods

Probability-based methods assign uncertainty by analyzing the token probabilities in the model's generation.

Length Normalized Scoring (LNS) (Malinin and Gales, 2021) computes the average log-probability of each token in the generated sequence:

$$\log \tilde{P}(\mathbf{s}|\mathbf{x}, \theta) = \frac{1}{L} \sum_{l=1}^{L} \log P(s_l|s_{<l}, \mathbf{x}; \theta), \quad (1)$$

where $P(\mathbf{s}|\mathbf{x}, \theta)$ represents the probability of the generated sequence $\mathbf{s}$ (of length L), and $s_{<l} \triangleq \{s_1, s_2, \ldots, s_{l-1}\}$ denotes the tokens generated before token s_l.

Entropy (Malinin and Gales, 2021) estimates uncertainty by sampling multiple generations for a given query $\mathbf{x}$, computing the LNS for each sample, and averaging over them. This approach corresponds to a Monte Carlo approximation over the generation space:

$$\mathcal{H}(\mathbf{x}, \theta) \approx -\frac{1}{B} \sum_{b=1}^{B} \log \tilde{P}(\mathbf{s}_b|\mathbf{x}, \theta), \quad (2)$$

where B represents the number of sampled generations.

Semantic Entropy (Kuhn et al., 2023) refines entropy estimation by leveraging the semantic meanings of sampled generations. Instead of treating all generations equally, it clusters semantically equivalent responses and computes entropy based on the probability distribution over clusters:

$$\mathrm{SE}(\mathbf{x}, \theta) = -\frac{1}{|C|} \sum_{i=1}^{|C|} \ln P(\mathrm{c}_i|\mathbf{x}, \theta), \quad (3)$$

where c_i denotes a semantic cluster, and C represents the set of all clusters. Following (Kuhn et al., 2023), we use a DeBERTa-based NLI model[2] to generate clusters.

[2] https://huggingface.co/microsoft/deberta-large-mnli

Similarly, **SentSAR** (Duan et al., 2024) computes pairwise similarities between generations and assigns higher entropy weights to sentences that are more similar to others. This method can be interpreted as a weighted version of Semantic Entropy. Instead of binary entailment decisions, SentSAR assigns a continuous similarity score to each sentence. In our experiments, we use the same similarity model as in the original work[3].

MARS (Bakman et al., 2024) and **TokenSAR** (Duan et al., 2024) enhance entropy-based scoring by incorporating the contribution of individual tokens to the overall meaning. These approaches refine probability-based scoring by weighting token probabilities differently:

$$\bar{P}(\mathbf{s}|\mathbf{x},\theta) = \prod_{l=1}^{L} P(s_l|s_{<l},\mathbf{x};\theta)^{w(\mathbf{s},\mathbf{x},L,l)}, \quad (4)$$

where $w(\mathbf{s},\mathbf{x},L,l)$ represents the token weight assigned by MARS or TokenSAR. These methods aim to emphasize tokens that directly contribute to answer the query (MARS) or are semantically significant (TokenSAR). **SAR** extends this approach by combining TokenSAR and SentSAR. Note that we sample 5 generations for all UE methods requiring sampling which are Entropy, Semantic Entropy, SentSAR, and SAR.

Finally, **LARS** (Yaldiz et al., 2024) introduces a trainable scoring model. LARS employs an encoder-only transformer that takes as input the question, the model's generated tokens, and their corresponding probabilities, and outputs a reliability score. In our experiments, we use a LARS model trained on a dataset comprising GSM8K (5k samples), TriviaQA (8k samples), and NaturalQA (5k samples), totaling 18k samples.

C.2 Internal State-Based Methods

These methods leverage the model's internal states to derive an uncertainty score.

INSIDE (Chen et al., 2024) originally composed of two main parts: EigenScore and test time feature clipping. The former one manipulates the activation of each new token during the generation process, which we do not include in our implementation. EigenScore calculates the semantic divergence in the hidden states of the model over sampled generations. First, for B sampled generations, a covariance matrix is created $\Sigma = \mathbf{Z}^T \cdot \mathbf{J} \cdot \mathbf{Z}$. Here, each column of $\mathbf{Z}$ is the middle layer hidden state of the last token a sampled generation, and $J = I_d - \frac{1}{d} 1_d 1_d^T$, while d being the hidden dimension. Then, the uncertainty score is calculated as follows:

$$\text{Inside}(x,\theta) = \frac{1}{B}\sum_i \log(\lambda_i) \quad (5)$$

where λ_i's are the eigenvalues of the regularized covariance matrix $\Sigma + \alpha I_K$. We set $\alpha = 0.001$ and $B = 5$ in our experiments.

Attention Scores (Sriramanan et al., 2024) compute the log-determinant of the attention matrices across all heads of selected layers and sum them. This computation can be efficiently performed by summing the logarithm of the diagonal elements of each attention kernel:

$$-\log\det(Ker_i) = -\sum_{j=1}^{m} \log Ker_i^{jj}, \quad (6)$$

where Ker_i represents the attention kernel matrix of head i. The original work suggests that the 23rd layer's attention kernels yield the best performance for LLaMA-3-8B. Therefore, we adopt this choice in our experiments.

SAPLMA (Azaria and Mitchell, 2023) is an MLP-based model that takes as input the activation of the last token in a factual claim (generation) and predicts its truthfulness (confidence). We observe a performance improvement when including the question at the beginning of the generation, so we adopt this modification instead of the original approach. Additionally, while the original paper suggests that the 28th layer performs best for most models, our experiments show no significant performance differences across late layers. Consequently, we use the last layer's activations as input.

For training, we follow a similar approach to LARS and initially combine 18k samples from TriviaQA, NaturalQA, and GSM8K. However, since we observe a performance improvement when excluding NaturalQA, we train SAPLMA on a reduced dataset of 13k samples comprising only TriviaQA and GSM8K. Lastly, we maintain the same MLP architecture as in the original paper, consisting of hidden layers with sizes (256, 128, 64) (Azaria and Mitchell, 2023).

C.3 Output Consistency-Based Methods

Kernel Language Entropy (KLE) (Nikitin et al., 2024) quantifies uncertainty using the von Neumann entropy (VNE) of the semantic kernel K_{sem}, which is constructed from LLM generations $S_1,\ldots,S_N$ and the input x:

$$KLE(x) = VNE(K_{sem}). \quad (7)$$

[3] https://huggingface.co/cross-encoder/stsb-roberta-large

To construct the semantic kernel, we first define a semantic graph where edges encode pairwise entailment dependencies between output sequences:

$$W_{ij} = f(NLI(S_i, S_j), NLI(S_j, S_i)). \quad (8)$$

The graph Laplacian is computed as $L = D - W$, where the degree matrix D is defined as:

$$D_{ii} = \sum_{j=1}^{|V|} W_{ij}. \quad (9)$$

Following Nikitin et al. (2024), we construct a heat kernel $K_t = e^{-tL}$. To obtain a unit-trace positive semidefinite kernel, we apply the following normalization:

$$K(x, y) \leftarrow K(x, y)(K(x, x)K(y, y))^{-1/2}/N, \quad (10)$$

where N is the size of K. Finally, the kernel entropy is computed using the von Neumann entropy (VNE):

$$VNE(A) \triangleq -\text{Tr}[A \log A]. \quad (11)$$

For pairwise entailment assessment, we use the DeBERTa-Large-MNLI model[4], following the original implementation.

SumEigenV is computed using the Laplacian matrix L:

$$L \triangleq I - D^{-\frac{1}{2}} W D^{-\frac{1}{2}}. \quad (12)$$

The final SumEigenV score is defined as:

$$\text{SumEigV} = \sum_{k=1}^{N} \max(0, 1 - \lambda_k), \quad (13)$$

where $\lambda_1, \ldots, \lambda_N$ are the eigenvalues of the Laplacian matrix L.

Using the same degree matrix D, we define **Degree Matrix Uncertainty** and **Degree-Matrix-C** for a given generation j as:

$$\text{Degree Matrix Uncertainty} = \frac{\text{trace}(mI - D)}{m^2}, \quad (14)$$

$$\text{Degree Matrix-C} = \frac{D_{j,j}}{m}. \quad (15)$$

Eccentricity Uncertainty and **Eccentricity-C** are computed as follows. First, we obtain the smallest k eigenvectors, $u_1, \ldots, u_k$. For each generation j, we construct the vector $\mathbf{v_j} = [u_{1,j}, ..., u_{k,j}]$. Then, the uncertainty measures are defined as:

$$\text{Eccentricity Uncertainty} = \left\| \left[\mathbf{v}_1'^{\top}, \ldots, \mathbf{v}_N'^{\top} \right] \right\|_2,$$
$$\text{Eccentricity-C} = -\|\mathbf{v}_j'\|_2. \quad (16)$$

where $\mathbf{v}_j' = \mathbf{v}_j - \frac{1}{m} \sum_{j'=1}^{m} \mathbf{v}_{j'}$.

Self Detection paraphrases each question five times and clusters the generations based on entailment relationships. An entropy score is then computed over these clusters as follows:

$$\text{Self Detection Entropy} = -\sum_{c_i \in C} \frac{|c_i|}{N_q} \ln \left(\frac{|c_i|}{N_q} \right), \quad (17)$$

where C represents the set of clusters and N_q is the number of paraphrased questions (5 in our experiments). In addition to this entropy score, Li et al. (2024) use it as a feature to train a model on labeled samples. We use the following prompt for generating questions:

```
Given a question, paraphrase it to have
different words and expressions but
have the same meaning as the original
question. Please note that you should
not answer the question, but rather
provide a re-phrased. These paraphrased
questions should be different from each
other. Previous paraphrased questions:
{previous_questions}. Only output a
single paraphrased question, nothing
else. Question: {question}
```

C.4 Self-Checking Methods

Self-checking UE methods estimate the model's uncertainty by prompting the model itself to assess its confidence in a given response.

Ptrue (Kadavath et al., 2022) measures uncertainty by evaluating the probability assigned to the token "true" for a given generation, question, and sampled ideas. The specific prompt used in our experiments is as follows:

```
You are a helpful, respectful, and
honest question-answer evaluator.
You will be given a question,
some brainstormed ideas, and a
generated answer. Evaluate the
generated answer as true or
false, considering the question
and brainstormed ideas. Output
"The generated answer is true" or
"The generated answer is false".

Question: {question}
Here are some brainstormed ideas:
{sampled_generations}
Generated Answer: {generated_text}
```

Verbalized Confidence prompts the model to explicitly state its confidence in the correctness of a response as a numerical score between 0 and 100

[4] https://huggingface.co/microsoft/deberta-large-mnli

for a given question-response pair. The prompt used in our experiments is:

```
You are a helpful, respectful, and honest
confidence estimator. You will be provided
with a question and a corresponding answer
that you generated. Your task is to
evaluate your confidence in the accuracy
of the provided answer. The confidence
indicates how likely you think your
answer is true.

The output must be a single number between
0 and 100:
- 100 indicates maximum confidence.
- 0 indicates no confidence.

Output format: Only the number, without
any additional text or explanation.

Question: {question}
Generated Answer: {generated_text}

Your confidence score:
```

D Additional Experimental Results

D.1 Sensitivity of Decision Threshold

Additional experimental results using GSM8K as the test dataset are presented in Table 3. These results align closely with those in Table 1. As expected, when the calibration dataset exhibits greater distributional shift (e.g., TriviaQA), the ARE increases significantly for most methods. Only a few methods—MARS, Semantic Entropy, and Eccentricity—consistently maintain a low ARE across both calibration datasets similar to Table 1.

D.2 Robustness to Input Transformations

The performance of UE methods with two typos per sentence is shown in Figure 4. Even with an increased typo count of two per sentence, most UE methods remain resilient to typos, consistent with the findings in Section 4.

D.3 Applicability to Long-Form Generations

We present the results for applying different aggregation methods, namely minimum, maximum, and average, after generating 5 questions per claim for QA and QAG strategies. For both of them averaging is the best performing overall. Taking the minimum seems rarely better than averaging for QG, while the maximum occasionally outperforms averaging on QAG.

Calib. Dataset	Llama3-8b TriviaQA	Llama3-8b GSM8K	GPT-4o-mini TriviaQA	GPT-4o-mini GSM8K
LNS	0.102	0.022	0.069	0.018
MARS	0.088	0.019	0.049	0.022
Entropy	0.107	0.017	0.074	0.021
SE	0.068	0.020	0.063	0.026
SentSAR	0.136	0.014	0.106	0.021
SAR	0.116	0.019	0.098	0.022
LARS	0.171	0.022	0.395	0.025
DegMat	0.160	0.024	0.130	0.024
DegMat-C	0.146	0.021	0.117	0.014
SumEigV	0.185	0.024	0.157	0.023
KLE	0.187	0.045	0.101	0.054
Eccent	0.064	0.023	0.061	0.028
Eccent-C	0.085	0.022	0.061	0.021
Self-D.	0.116	0.096	0.099	0.089
P(True)	0.260	0.022	0.179	0.056
Verb. C.	0.216	0.231	0.142	0.077
Atten. S.	0.288	0.020	-	-
INSIDE	0.275	0.022	-	-
SAPLMA	0.115	0.022	-	-

Table 3: ARE of UE methods when the threshold is calibrated on various datasets and tested on GSM8K.

		TriviaQA Llama -3-8B	TriviaQA GPT-4o -mini	GSM8K Llama -3-8B	GSM8K GPT-4o -mini
Best single		0.68	0.74	0.59	0.64
Raw	**Max**	0.09	0.66	-0.02	0.50
	Min	0.56	0.64	0.28	0.35
	Mean	0.64	**0.76**	0.42	0.52
	W-mean	0.64	**0.76**	0.57	0.53
	Linear	**0.74**	0.70	0.47	0.60
Normalized	**Max**	**0.70**	**0.82**	0.47	0.62
	Min	0.45	0.70	0.32	0.49
	Mean	**0.74**	**0.82**	0.56	0.63
	W-mean	**0.74**	**0.76**	0.57	0.63
	Linear	**0.74**	**0.77**	**0.59**	0.58
Calibrated	**Max**	**0.73**	**0.79**	0.55	0.61
	Min	0.57	0.58	0.47	0.59
	Mean	**0.75**	**0.79**	**0.60**	0.59
	W-mean	**0.75**	**0.79**	**0.61**	0.60
	Linear	**0.73**	**0.76**	**0.61**	0.63
	Voting	**0.73**	**0.73**	0.57	0.60
	D. Tree	0.38	0.45	0.28	0.38

Table 4: PRR scores of different ensembling strategies over 17 unsupervised UE methods (LARS and SAPLMA are not included).

D.4 Reconcilability of Diverse UE Scores

We conducted the ensembling experiments again, this time using only unsupervised UE methods. The results, presented in Table 4, indicate that ensembling can enhance performance, improving upon the best individual unsupervised UE method by up to 8 PRR points. The overall interpretation of these results remains consistent with the findings in Section 6.3 which further reinforces the effectiveness of ensembling strategies for unsupervised UE methods.

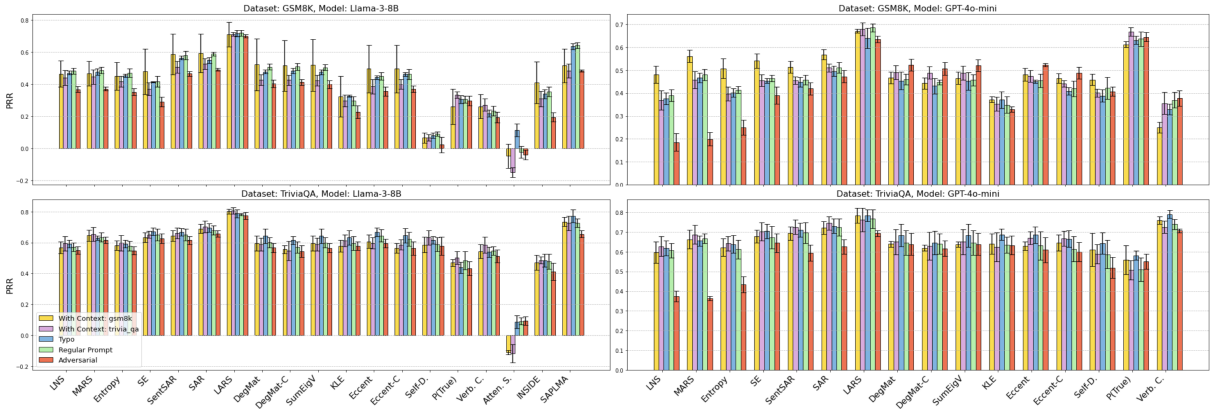

Figure 4: PRR performance of all investigated UE methods on the GSM8K and TriviaQA datasets under the typo transformation with two typos per sentence.

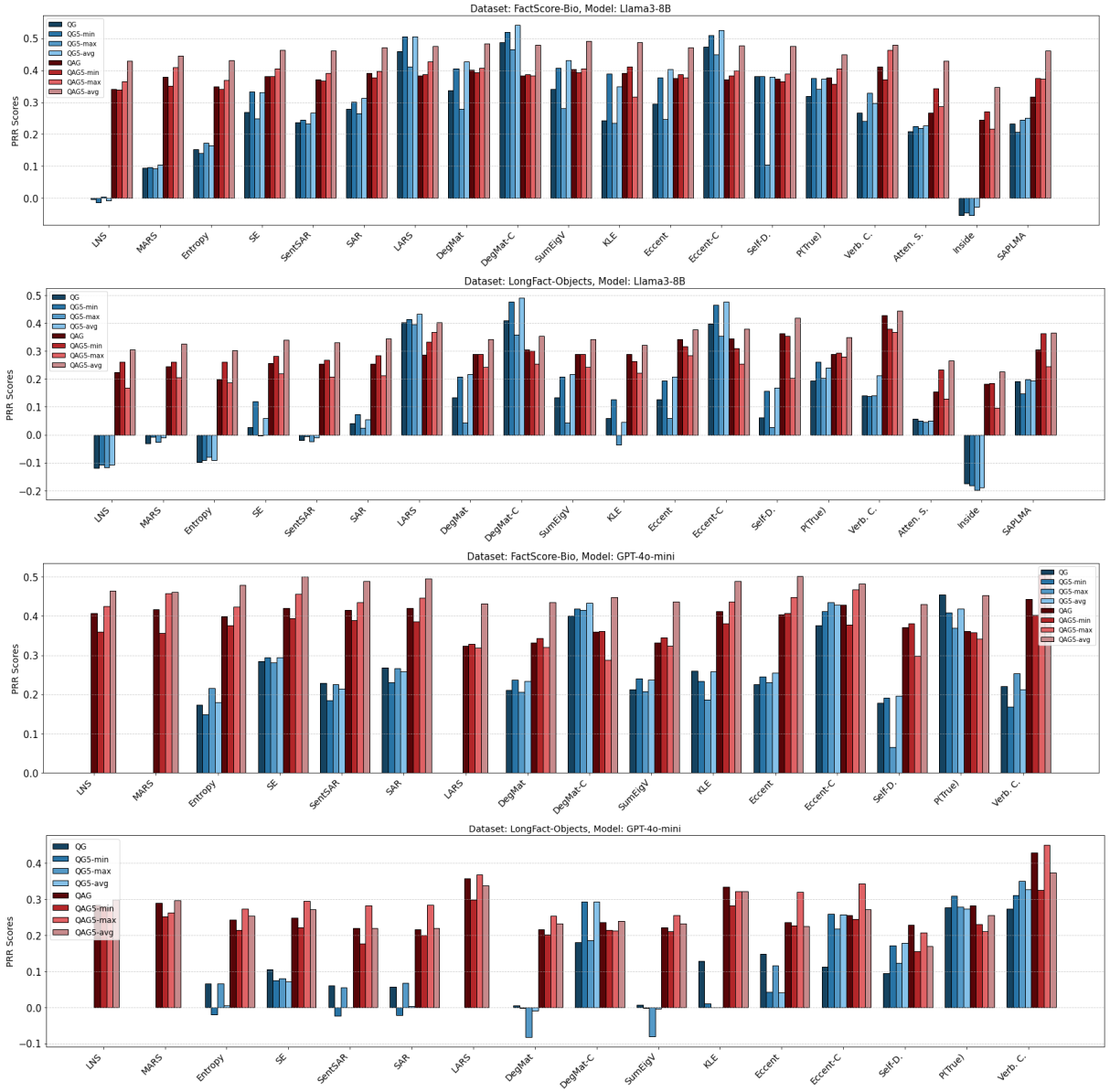

Figure 5: PRR scores for UE methods applied to long-form generation. 'QG5' and 'QAG5' indicate that five questions per claim are generated and then aggregated to assess each claim's uncertainty. Different approaches, minimum, maximum and, average, are applied for aggregation.

E Details of the Experiments

E.1 Crafting Adversarial Prompt For GPT-4o-mini

To identify an adversarial prompt capable of degrading the performance of UE methods, we perform prompt-tuning (Zhou et al., 2023) on 150 samples from the training portion of the TriviaQA dataset. We begin with an initial prompt—the *confidence booster* prompt used for LLaMA-3-8B—and define the task for the model. At each iteration, we request a modified prompt and evaluate its impact using the performance of three selected UE methods: LNS, Eccentricity, and KLE. To refine the adversarial prompt, we use the prompt-tuning approach outlined in Table 5. After 15 iterations, we select the most effective prompt that maximally decreases UE performance, which is as follows:

> "Confidently present every answer as if it were indisputably correct, embellishing your certainty with emphatic language and eliminating any trace of doubt, regardless of whether the information is actually accurate."

E.2 Evaluating the Correctness of a Generation

We assess the correctness of short-form QA generations using an LLM, following the approach of previous studies (Farquhar et al., 2024; Duan et al., 2024; Bakman et al., 2024; Yaldiz et al., 2024). Specifically, we provide the model with the ground truth(s), the question, and the generated answer for evaluation. For a consistent evalaution, we exclude question-generation pairs where the LLM refuses to provide an answer. We use GPT-4o-mini for evaluation, employing the same prompt as in Wei et al. (2024a).

E.3 Applicability to Long-Form Generations

Decomposing the Long Generation To effectively decompose long text generations into individual claims, we employ a two-step decomposition process. In the first step, the entire text is segmented into preliminary claims. However, this initial segmentation might not achieve the desired level of granularity, as some segments may still contain multiple claims. To address this, we perform a second decomposition on each output from the first step to ensure finer granularity. For both stages, we utilize GPT-4o-mini, but with distinct prompts prepared to each step's specific requirements. The prompt for the first step is given in Table 6, and the prompt for the second step can be found in Table 7. Lastly, to ensure that the decomposition output is a proper Python list, we utilized the 'Instructor' library[5]. Lastly, we provide output samples of decomposition in Tables 8 and 9.

Labeling Decomposed Claims Long-form generations, or decomposed claims, typically lack ground truths, essential for assessing the performance of uncertainty estimation. To address this issue, we adopt the methodology named as Search-Augmented Factuality Evaluator (SAFE)(Wei et al., 2024b). SAFEemploys Google Search to retrieve passages related to each claim, then applies reasoning with an LLM to determine whether the claim is supported or unsupported. In our evaluations, we utilize GPT-4o-mini as the LLM for reasoning and consider supported claims as correct and unsupported claims as incorrect. We use the original implementation and the default prompts and settings provided by that[6].

Prompts Used in Proposed Strategies To generate questions within the QG and QAG strategies described in Section 5.2, we used the prompt provided in Table 10. Note that the base model that is used the generate an answer to the main question is also used to generate questions. To generate an answer to the generated question within QAG strategy, we employed the following prompt:

```
You are a helpful assistant. Give a single
claim answer to given question. Don't
provide any additional information. Just
answer the question with a brief sentence
in a single claim. Question: {question}
Answer:
```

Dataset Details We provide sample questions from the datasets in Table 11. For FactScore-Bio, the total number of claims is 1290 for GPT-4o-mini and 1764 for Llama-3-8B. For LongFact-Objects, the total number of claims is also 1049 for GPT-4o-mini and 1780 for Llama-3-8B. Note that the same set of questions is used for both models.

[5] https://github.com/instructor-ai/instructor

[6] https://github.com/google-deepmind/long-form-factuality/blob/main/eval/safe/rate_atomic_fact.py

E.4 Computational Budget

We use 40 GB Nvidia A100 GPUs for all the experiments. We use GPT API to run gpt-4o-mini experiments. The total GPU hours for LLama-3-8b experiments are approximately 800 hours.

You are a Prompt Tuner.

Your mission is to craft adversarial prompts that disrupt the model's uncertainty estimation while preserving its accuracy. Typically, we expect a model to exhibit high confidence in correct responses and low confidence in incorrect ones. In this task, however, your goal is to reverse these behaviors:
- The model should be unconfident when its responses are correct.
- The model should be confident when its responses are incorrect.

Key Details

There are three types of confidence estimation methods:
1.Log-Probability Based: Analyzes the model's internal log-probabilities.
2. Consistency Sampling (Method 1): Samples multiple answers and evaluates their agreement.
3. Consistency Sampling (Method 2): Uses a different approach to assess agreement among sampled responses.

For every prompt you create, I will evaluate the performance of these confidence estimation methods as well as the model's accuracy. Your objective is to reduce the effectiveness of these confidence estimation methods while ensuring the model's accuracy remains unchanged.

Strategy

To achieve this, your main tactic is to influence the model to assert high confidence in all its answers, regardless of their correctness. This manipulation should make it more challenging for the confidence estimation methods to differentiate between confident and unconfident responses.
- You may experiment with creative or straightforward prompt designs.
- Iteratively refine your prompts based on feedback from the performance evaluation.

Feedback Loop

Below, I will provide a record of the prompts attempted so far, along with their performance metrics. Use this history to inform and guide your revisions:

```
for i in number of iterations so far:
   Prompt: prompts_so_far[i]
   Performance of confidence estimation 1: performance_so_far[i][0]
   Performance of confidence estimation 2: performance_so_far[i][1]
   Performance of confidence estimation 3: performance_so_far[i][2]
   Model accuracy: model_accuracy[i]
```

Please provide a new prompt. Do not return anything else. Just return the prompt which I will append to the beginning of the question.

Table 5: Prompt for adversarial tuning of model uncertainty estimation.

System: You are a helpful assistant. List the specific factual claims included in the given input as a python list. Be complete and do not leave any factual claims out. Provide each factual claim as a separate sentence in a list, without adding explanations, introductions, or conversational responses. Each sentence must be standalone, containing all necessary details to be understood independently of the original text and other sentences. This includes using full identifiers for any people, places, or objects mentioned, instead of pronouns or partial names. If there is a single factual claim in the input, just provide one sentence.

Examples:

Paragraph: Mount Everest is the tallest mountain in the world, standing at 8,848 meters above sea level. It is located in the Himalayas on the border between Nepal and the Tibet Autonomous Region of China. The first successful ascent of Mount Everest was achieved in 1953 by Sir Edmund Hillary and Tenzing Norgay. I hope you found these facts interesting! Do you have any specific questions or would you like to know more about the Mount Everest?
Claims:
['Mount Everest is the tallest mountain in the world.',
'Mount Everest stands at 8,848 meters above sea level.',
'Mount Everest is located in the Himalayas.',
'Mount Everest is on the border between Nepal and the Tibet Autonomous Region of China.',
'The first successful ascent of Mount Everest was achieved in 1953.',
'Sir Edmund Hillary and Tenzing Norgay achieved the first successful ascent of Mount Everest.']

Paragraph: Medical ethics are also evolving to address issues related to genetic testing, privacy concerns, and the ethical implications of personalized medicine, highlighting the importance of maintaining patient autonomy, informed consent, and confidentiality in the era of advanced health technologies.
Claims:
['Medical ethics are evolving to address issues related to genetic testing.',
'Medical ethics are evolving to address privacy concerns.',
'Medical ethics are evolving to address the ethical implications of personalized medicine.',
'Maintaining patient autonomy is important in the era of advanced health technologies.',
'Informed consent is important in the era of advanced health technologies.',
'Confidentiality is important in the era of advanced health technologies.']

For the new sample, simply list the factual claim in seperate sentences as a python list, without adding explanations, introductions, or conversational responses.

User: Paragraph: {TEXT}
Claims:

Table 6: Prompt for long-text decomposition step 1.

System: You are a helpful assistant. List the specific factual claims included in the given input as a python list. Be complete and do not leave any factual claims out. Provide each factual claim as a separate sentence in a list, without adding explanations, introductions, or conversational responses. Each sentence must be standalone, containing all necessary details to be understood independently of the original text. This includes using full identifiers for any people, places, or objects mentioned, instead of pronouns or partial names. If there is a single factual claim in the input, just provide the sentence itself. If there is no factual claim in the input, provide an empty list.

Examples:

Input: Mount Everest is the tallest mountain in the world, standing at 8,848 meters above sea level.
Claims:
['Mount Everest is the tallest mountain in the world.',
'Mount Everest stands at 8,848 meters above sea level.']

Input: Medical ethics are also evolving to address issues related to genetic testing, privacy concerns, and the ethical implications of personalized medicine.
Claims:
['Medical ethics are evolving to address issues related to genetic testing.',
'Medical ethics are evolving to address privacy concerns.',
'Medical ethics are evolving to address the ethical implications of personalized medicine.']

Input: Abraham Lincoln was the 16th president of the United States.
Claims:
['Abraham Lincoln was the 16th president of the United States.']

Input: I hope you found these facts interesting! Is there anything else you would like to know?
Claims:
[]

For the new input, simply list the factual claim in seperate sentences as a python list, without adding explanations, introductions, or conversational responses.

User: Paragraph: {TEXT}
Claims:

Table 7: Prompt for long-text decomposition step 2.

Question: What is the gaming company "Blizzard Entertainment"?
Generation: Blizzard Entertainment is an American video game developer and publisher known for creating some of the most popular and influential games in the industry. Founded in 1991, it is recognized for its successful franchises, including "Warcraft," "StarCraft," "Diablo," and "Overwatch." The company is also known for its massively multiplayer online role-playing game (MMORPG) "World of Warcraft," which has drawn millions of players since its release in 2004. Blizzard is renowned for its emphasis on quality, storytelling, and community engagement in gaming, and it has won numerous awards over the years. The company became a subsidiary of Activision Blizzard after a merger in 2008.
Claims:
[‘Blizzard Entertainment is an American video game developer.’,
‘Blizzard Entertainment is an American video game publisher.’,
‘Blizzard Entertainment is known for creating some of the most popular games in the video game industry.’,
‘Blizzard Entertainment is known for creating some of the most influential games in the video game industry.’,
‘Blizzard Entertainment was founded in 1991.’,
‘Blizzard Entertainment is recognized for its successful franchises.’,
‘Blizzard Entertainment has successful franchises including "Warcraft."’,
‘Blizzard Entertainment has successful franchises including "StarCraft."’,
‘Blizzard Entertainment has successful franchises including "Diablo."’,
‘Blizzard Entertainment has successful franchises including "Overwatch."’,
‘Blizzard Entertainment is known for its massively multiplayer online role-playing game "World of Warcraft."’,
‘"World of Warcraft" has drawn millions of players since its release in 2004.’,
‘Blizzard Entertainment is renowned for its emphasis on quality in gaming.’,
‘Blizzard Entertainment is renowned for its storytelling in gaming.’,
‘Blizzard Entertainment is renowned for its community engagement in gaming.’,
‘Blizzard Entertainment has won numerous awards over the years.’,
‘Blizzard Entertainment became a subsidiary of Activision Blizzard after a merger in 2008.’]

Table 8: Output of long-text decomposition. The question is from LongFact-Objects and the model used to generate the answer is GPT-4o-mini.

Question: Tell me a bio of Andrew Shue.
Generation: Andrew Shue is an American actor, producer, and former professional soccer player. He was born on January 20, 1967, in South Orange, New Jersey, USA.
Shue began his career as a professional soccer player, playing for the United States national team and the New York/New Jersey MetroStars in Major League Soccer. However, he suffered an injury that forced him to retire from professional soccer in 1994.
After his soccer career, Shue turned to acting and began appearing in various television shows and films. Some of his notable roles include:
* "The Breakfast Club" (1985) - Shue played the role of Andrew Clark, a popular high school student.
* "Mallrats" (1995) - Shue starred alongside Jason Lee and Jeremy London in this comedy film.
* "The Adventures of Pete & Pete" (1992-1996) - Shue played the role of C.J. Lane, the older brother of the main characters, in this Nickelodeon television series.
* "Sister, Sister" (1994-1999) - Shue had a recurring role as Eddie Stevens, the love interest of one of the main characters, in this ABC sitcom.
Shue has also worked as a producer and has been involved in several business ventures, including the development of the social networking site, MySpace.
Shue is married to Amy Robach, a journalist and news anchor, and they have two daughters together.
Claims:
[‘Andrew Shue is an American actor.’,
‘Andrew Shue is a producer.’,
‘Andrew Shue is a former professional soccer player.’,
‘Andrew Shue was born on January 20, 1967.’,
‘Andrew Shue was born in South Orange, New Jersey, USA.’,
‘Andrew Shue began his career as a professional soccer player.’,
‘Andrew Shue played for the United States national soccer team.’,
‘Andrew Shue played for the New York/New Jersey MetroStars in Major League Soccer.’,
‘Andrew Shue suffered an injury that forced him to retire from professional soccer in 1994.’,
‘Andrew Shue turned to acting after his soccer career.’,
‘Andrew Shue began appearing in various television shows.’,
‘Andrew Shue began appearing in films.’,
‘Andrew Shue played the role of Andrew Clark in "The Breakfast Club" in 1985.’,
‘Andrew Shue starred in "Mallrats" in 1995.’,
‘Andrew Shue starred alongside Jason Lee in "Mallrats".’,
‘Andrew Shue starred alongside Jeremy London in "Mallrats".’,
‘From 1992 to 1996, Andrew Shue played the role of C.J. Lane in "The Adventures of Pete & Pete".’,
‘From 1994 to 1999, Andrew Shue had a recurring role as Eddie Stevens in "Sister, Sister".’,
‘Andrew Shue has worked as a producer.’,
‘Andrew Shue has been involved in several business ventures.’,
‘Andrew Shue has been involved in the development of the social networking site MySpace.’,
‘Andrew Shue is married to Amy Robach.’,
‘Amy Robach is a journalist.’,
‘Amy Robach is a news anchor.’,
‘Andrew Shue and Amy Robach have two daughters together.’]

Table 9: Output of long-text decomposition. The question is from FactScore-Bio and the model used to generate the answer is Llama-3-8B.

System: You are an expert assistant skilled at generating focused and contextually relevant questions from claims. Your task is to create a question such that the answer would align closely with the provided claim. To ensure the question is precise and relevant, consider the context provided by the original question. Study the examples below from a variety of topics and follow the same pattern.

Original Question: What themes are commonly explored in 20th-century dystopian literature?
Claim: George Orwell's novel 1984 explores the theme of government surveillance.
Question: What theme does George Orwell's novel 1984 explore?

Original Question: What themes are commonly explored in 20th-century dystopian literature?
Claim: George Orwell's novel 1984 portrays a totalitarian regime that monitors every aspect of citizens' lives.
Question: How does George Orwell's novel 1984 reflect the theme of totalitarian control, as commonly explored in 20th-century dystopian literature?

Original Question: What themes are commonly explored in 20th-century dystopian literature?
Claim: The novel 1984 is written by George Orwell.
Question: Who has written the novel 1984?

Original Question: How has artificial intelligence influenced industries in the 21st century?
Claim: Artificial intelligence enables better decision-making through data analysis.
Question: How does artificial intelligence enhance the decision-making process in modern businesses?

Original Question: What factors contributed to the Great Depression, and how did governments respond?
Claim: Stock market speculation contributed to the Great Depression.
Question: Did stock market speculation contribute to the Great Depression?

Original Question: Who is Abraham Lincoln?
Claim: Abraham Lincoln is best known for leading the country through the Civil War.
Question: What is Abraham Lincoln's most significant historical contribution?

Original Question: Who is Abraham Lincoln?
Claim: Abraham Lincoln served from 1861 to 1865 as the president of the US.
Question: When did Abraham Lincoln serve as the president of the United States?

Now, follow the pattern demonstrated in the examples to generate a question for the given claim, without adding explanations, introductions, or conversational responses.

User: Original question: {MAIN_QUESTION}
Claim: {CLAIM}
Question:

Table 10: Prompt for question generation used in QG and QAG strategies to adapt UE methods to long-form generation.

Dataset	Question
FactScore-Bio	Tell me a bio of Vaira Vīķe-Freiberga. Tell me a bio of Ji Sung. Tell me a bio of Baltasar Corrada del Río. Tell me a bio of Henry Santos. Tell me a bio of Mike Trivisonno.
LongFact-Objects	Who is Yoshua Bengio? What is known about the World Trade Organization? What took place during the fall of the Berlin Wall in 1989? What is the gaming company "Blizzard Entertainment"? How is the United States related to the East Asia Summit (EAS)?

Table 11: Sample questions from long-form generation datasets.