The Anatomy of Uncertainty in LLMs

Anonymous ACL submission

Abstract

Understanding *why* a large language model (LLM) is uncertain about the response is important for their reliable deployment. Current approaches, which either provide a single uncertainty score or rely on the classical aleatoric-epistemic dichotomy, fail to offer actionable insights for improving the generative model. Recent studies have also shown that such methods are not enough for understanding uncertainty in LLMs. In this work, we advocate for an uncertainty *decomposition* framework that dissects LLM uncertainty into three distinct *semantic* components: (i) input ambiguity, arising from ambiguous prompts; (ii) knowledge gaps, caused by insufficient parametric evidence; and (iii) decoding randomness, stemming from stochastic sampling. Through a series of experiments we demonstrate that the dominance of these components can shift across model size and task. Our framework provides a better understanding to audit LLM reliability and detect hallucinations, paving the way for targeted interventions and more trustworthy systems.

1 Introduction

Large Language Models (LLMs) have achieved remarkable success in complex reasoning and generation tasks. Despite their great capabilities, they have the tendency to generate plausible-sounding but uncertain responses. Understanding when and why these models are uncertain in their response helps in detecting hallucination (Manakul et al., 2023; Kadavath et al., 2022; Kuhn et al., 2023), improving response quality (Ramírez et al., 2024), and optimizing tool calling (Zubkova et al., 2025). Recent studies have shown that hallucinations in LLMs are triggered because the model often guesses when they are unsure about the final response (Kalai et al., 2025). This makes it important to identify when the model is uncertain and the fundamental nature and origins of uncertainty.

Uncertainty in LLMs can originate from different sources. Consider the case of Gemma3 27B model (Team et al., 2025), when asked a straightforward question from TriviaQA (Joshi et al., 2017),

> *"What was Walter Matthau's first movie?"*

the model consistently responded with *"The Gangster,"* while the correct answer is *"The Kentuckian."* This discrepancy highlights two possible scenarios. First, the phrasing of the question introduces input ambiguity, since "first movie" could mean Matthau's first credited role or his earliest on-screen appearance. Second, the model's internal knowledge may be incomplete or imprecise, reflecting knowledge gaps in its training data. Similarly, another source of discrepancy in the response could be introduced during output decoding. If we look at another example from the same dataset,

> *"In Hanna and Barbera's TV cartoons base on The Addams Family who was the voice of Gomez?"*

the Gemma3 27B model consistently gives correct answer, *"John Astin"* when responses are generated using greedy decoding. But when temperature decoding is used, it sometimes responds with incorrect answer, *"Ted Cassidy."* Recent work has also demonstrated that decoding strategies influence both model outputs and the resulting uncertainty estimates (Hashimoto et al., 2025).

Existing approaches focuses on quantifying these uncertainty using a single score (Manakul et al., 2023; Kadavath et al., 2022; Kuhn et al., 2023). While these scores are useful for ranking responses and guiding abstention, they are not actionable because they fail to diagnose the root cause of uncertainity. It remains unclear what intervention might reduce the uncertainty in these systems. Some works have shown how these

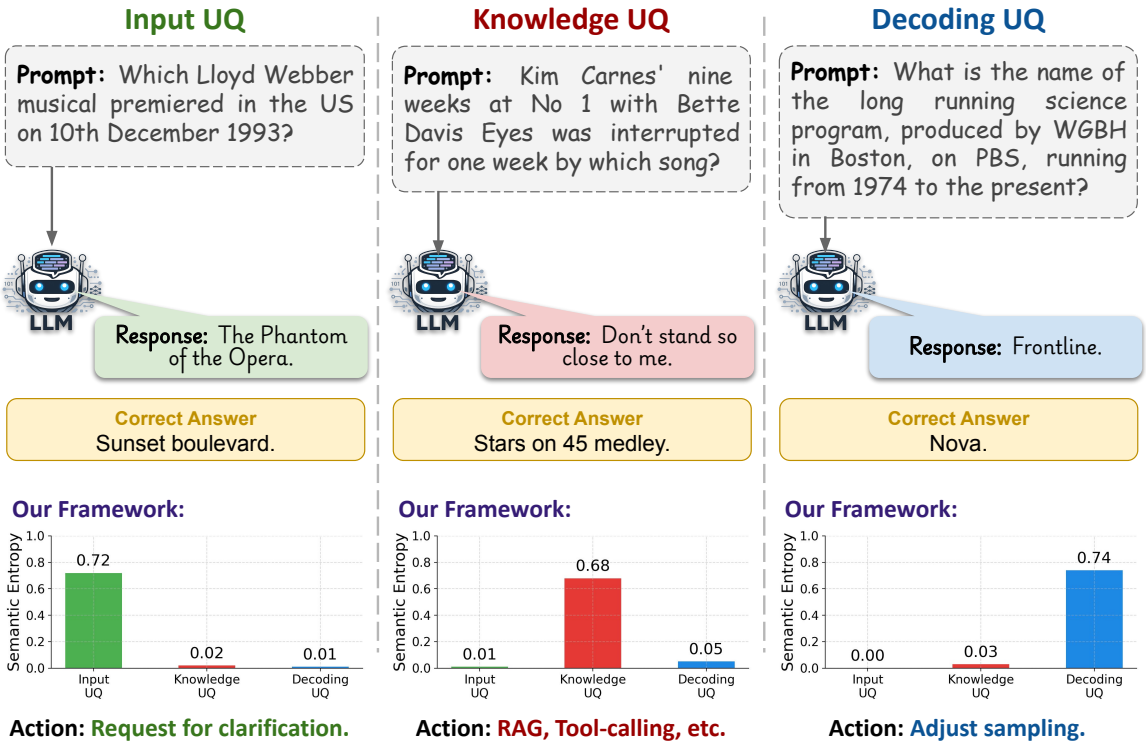


Figure 1: Uncertainty decomposition on TriviaQA examples of Gemma 3 27B model. Our framework identifies the dominant source of uncertainty—input ambiguity (left), knowledge gaps (middle), or decoding randomness (right)—and maps each to a targeted mitigation action.

uncertainties can be decomposed into classical aleatoric-epistemic dichotomy (Senanayake, 2024; Hou et al., 2024). However, recent studies (Kirchhof et al., 2025; Huang et al., 2024; Xie et al., 2025; Bakman et al., 2025) have highlighted that this dichotomy is not sufficient in case of LLMs.

To address these issues, in this work we propose a framework that decomposes uncertainty in LLMs into three distinct *semantic* components: (i) input ambiguity, stemming from prompts with multiple valid interpretations; (ii) knowledge gaps, caused by insufficient training coverage or outdated information; and (iii) decoding randomness, introduced by the sampling process itself (see Figure 1). We advocate that this decomposition provides a more faithful description of uncertainty in generative models and offers actionable insights for system design. For example, high input-driven uncertainty suggests clarifying questions; high knowledge uncertainty suggests retrieval or data augmentation; and high decoding uncertainty suggests adjusting sampling strategy. Our contributions are:

1. We propose a framework for decomposing LLM uncertainty into three distinct *semantic* components: input ambiguity, knowledge gaps, and decoding randomness.
2. We empirically demonstrate how the dominant source of uncertainty shifts across different tasks and model scales.

2 Related Works

Uncertainty Quantification in LLMs. Prior work has developed various techniques to assign uncertainty scores to LLM outputs. (Manakul et al., 2023) propose SelfCheckGPT, a sampling-based method that compares multiple model generations. They show that, if a fact is truly known, the samples agree, whereas hallucinated facts cause divergent answers. Similarly, (Kadavath et al., 2022) uses LLM itself to estimate the probability that their own answers are correct (a "P(True)" confidence). Another approach by (Kuhn et al., 2023) defines a semantic entropy score that accounts for linguistic paraphrases (shared meaning) to better predict uncertainty. These uncertainty scores have been useful for improving tasks like hallucination detection and inference efficiency. For instance, (Ramírez et al., 2024) show that using a small model's uncertainty to decide whether to invoke a larger model yields an effective two-tier cascade. But they do not explain why the LLM is uncertain about a particular response and how we can improve them.

With our approach, we aim to bridge this gap by decomposing uncertainty into interpretable sources.

Towards Decomposing Uncertainty in LLMs. Decomposing uncertainty into meaningful components has a long history in Bayesian and reinforcement learning (Charpentier et al., 2022). Inspired by this, recent work has begun exploring uncertainty decomposition in large language models. (Hou et al., 2024) introduce an input-clarification ensembling framework that generate multiple disambiguated versions of each prompt and ensemble the outputs. However, recent researches note that the simple aleatoric-epistemic split is not ideal for LLMs. (Kirchhof et al., 2025) argue that classical definitions of aleatoric vs. epistemic uncertainty "contradict each other and lose their meaning" in open-ended, interactive language tasks. In particular, assigning fixed aleatoric and epistemic scores to each output cannot capture the nuanced, multi-turn uncertainty arising from under specified prompts. Motivated by these observations, we take a more fine-grained view and explicitly separate input ambiguity, knowledge gaps, and decoding randomness as distinct uncertainty sources.

3 Anatomy of Uncertainty in LLMs

We propose a framework for decomposing the uncertainty in a LLM's response into three distinct *semantic* sources: input ambiguity, knowledge gaps, and decoding randomness. These sources correspond to the main stages of the generation pipeline: the user prompt (Input), the model parameters (Knowledge), and the generation procedure (Decoding).

Let $x \in X$ denote an input, $y \in Y$ a generated response, θ the model parameters of the LLM, and τ a decoding strategy. The model induces a conditional distribution $p(y \mid x, \theta, \tau)$. The overall predictive uncertainty for an input x may be viewed as the entropy of this output distribution,

$$U_{\text{total}}(x) = \mathcal{H}\left(p(Y \mid x, \theta, \tau)\right). \tag{1}$$

Although this score can flag uncertain outputs, it is not diagnostic. To identify *why* the model is uncertain, we instead isolate one source of variation at a time while holding the others fixed, and compute semantic entropy over the resulting set of responses.

Input Ambiguity (U_{input}). Input ambiguity captures uncertainty caused by how the prompt is phrased. As shown in Fig. 2, to isolate this source, we fix θ and τ, and consider a set of K meaning-preserving paraphrases,

$$P(x) = \{x^1, x^2, \ldots, x^K\},$$

where each variation preserves the intent of the original query rather than introducing arbitrary textual corruption. We generate one response for each paraphrase and group the resulting outputs into semantic equivalence classes. Let C denote the set of such semantic clusters. We define input-induced uncertainty as,

$$U_{\text{input}}(P, \theta, \tau) = -\sum_{c \in C} \hat{p}(c) \log \hat{p}(c),$$
$$\text{where} \quad \hat{p}(c) = \frac{\sum_{y^k \in c} p(y^k \mid x^k, \theta, \tau)}{\sum_{c' \in C} \sum_{y^k \in c'} p(y^k \mid x^k, \theta, \tau)} \tag{2}$$

High U_{input} indicates that semantically equivalent phrasings lead to different meanings in the model outputs, suggesting that the query is underspecified and may benefit from clarification.

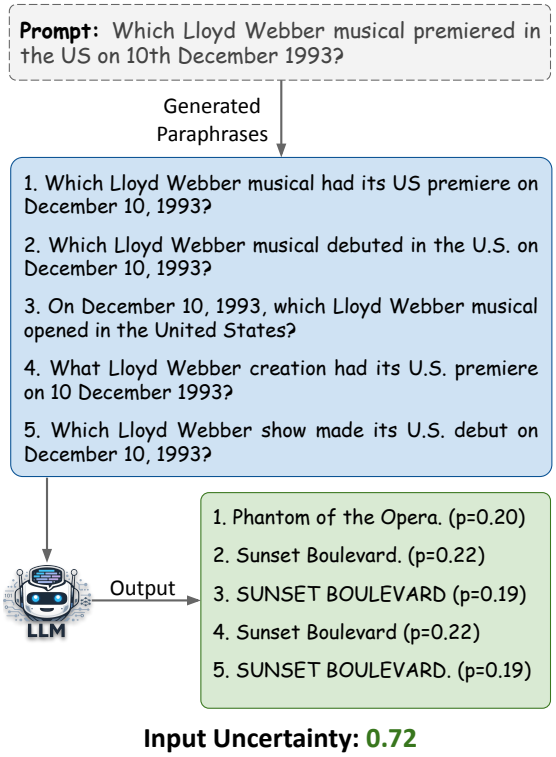


Figure 2: Illustration of input ambiguity estimation. We generate semantically equivalent paraphrases of the original question, obtain one response for each paraphrase using the same model and decoding policy, and group the responses into semantic clusters.

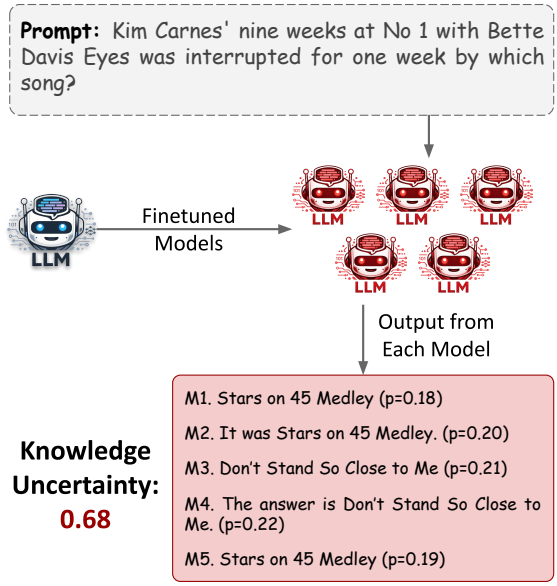


Figure 3: Illustration of knowledge-gap uncertainty estimation. For a fixed input and decoding policy, we query an ensemble of LoRA-adapted model realizations and group the resulting responses into semantic clusters. Disagreement across ensemble members reflects uncertainty arising from parametric knowledge gaps.

Knowledge Gaps ($U_{\text{knowledge}}$). Knowledge uncertainty captures variability arising from the model parameters. Since exact posterior inference over LLM parameters is intractable, we approximate multiple plausible model realizations using an ensemble,

$$\Theta = \{\theta^1, \theta^2, \ldots, \theta^M\},$$

where each θ^m is obtained through a distinct LoRA adaptation over the training set. As shown in Fig. 3, by fixing x and τ, we generate one response from each realization and compute semantic entropy over the resulting outputs:

$$U_{\text{knowledge}}(x, \Theta, \tau) = -\sum_{c \in C} \hat{p}(c) \log \hat{p}(c),$$
$$\text{where} \quad \hat{p}(c) = \frac{\sum_{y^m \in c} p(y^m \mid x, \theta^m, \tau)}{\sum_{c' \in C} \sum_{y^m \in c'} p(y^m \mid x, \theta^m, \tau)} \tag{3}$$

High $U_{\text{knowledge}}$ reflects disagreement across model realizations and serves as a practical proxy for parametric knowledge gaps.

Decoding Randomness (U_{dec}). Decoding uncertainty captures variability introduced by the response generation procedure itself. Let T denote a family of decoding strategies used in real deployment, such as greedy decoding, beam search, temperature sampling, top-k, and top-p sampling. For a chosen strategy $\tau \in T$, we fix x and θ, generate N responses $\{y^n\}_{n=1}^N$ under the selected sampling, and compute semantic entropy over the induced semantic clusters:

$$U_{\text{dec}}(x, \theta, \tau) = -\sum_{c \in C} \hat{p}(c) \log \hat{p}(c),$$
$$\text{where} \quad \hat{p}(c) = \frac{\sum_{y^n \in c} p(y^n \mid x, \theta, \tau)}{\sum_{c' \in C} \sum_{y^n \in c'} p(y^n \mid x, \theta, \tau)} \tag{4}$$

By comparing U_{dec} across different $\tau \in T$, we can analyze how strongly the model's uncertainty depends on the choice of decoding policy.

Although these three sources of uncertainty are analyzed separately, they are not strictly orthogonal in practice. For example, an ambiguous prompt can also increase decoding variability by flattening the output distribution across multiple plausible interpretations. Thus, our framework should be viewed as a tool for diagnostic decomposition rather than an additive partition of a single total uncertainty score.

4 Experiments

In this section, we empirically validate our proposed uncertainty *decomposition* framework. Our experiments are designed to answer two primary research questions:

RQ 1: Can our decomposed uncertainty scores effectively predict model failures across different tasks?

RQ 2: How do the dominant sources of uncertainty change with model scale and task type?

4.1 Experimental Setup

In this section, we outline the components of our experimental design, including the datasets, models, and evaluation metrics used to validate our framework.

4.1.1 Tasks and Datasets

We evaluate our framework on two distinct datasets to analyze uncertainty under different tasks, focusing on the LLM's accuracy in zero-shot prediction. For factual question answering, we use TriviaQA, a dataset that requires models to provide factually correct responses, thereby testing their learned knowledge and ability to generate precise

information. We also use GSM8K to evaluate the model on mathematical reasoning, as this dataset of grade-school math word problems assesses multi-step reasoning where errors may arise from either misinterpretation of the problem statement or flaws in the logical chain.

4.1.2 Models

We conduct experiments across a range of model families and sizes, including Llama 3 (8B) (Grattafiori et al., 2024) and Gemma 3 (270M, 1B, 4B, 12B and 27B) (Team et al., 2025). Input and decoding based uncertainty estimation were tested on all these models, but model based uncertainty was only tested on Llama 3 8B and Gemma 3 27B.

4.1.3 Implementation Details

Below we describe the implementation details for each uncertainty component.

Input Ambiguity (U_{input}). For each prompt, we generate $K = 5$ semantically similar paraphrases using GPT-5-nano. For each paraphrase, we obtain one response from the target LLM using greedy decoding. We then compute semantic entropy over the resulting set of responses using bidirectional entailment-based semantic clustering, as described in equation 2.

Knowledge Gaps ($U_{\text{knowledge}}$). We construct an ensemble of $M = 5$ model realizations for Llama 3 and Gemma 3 models by training LoRA adapters with different random seeds on the training split of the corresponding dataset. For a given prompt, we generate one response from each ensemble member using greedy decoding and compute semantic entropy over the ensemble outputs, as described in equation 3.

Decoding Randomness (U_{dec}). For a fixed decoding strategy τ, we generate $N = 5$ responses for each prompt using different random seeds and compute semantic entropy over these repeated generations, as described in equation 4. We evaluate this separately for multiple practical decoding strategies. Specifically, greedy decoding uses a single deterministic decoding path; beam search uses 5 beams with length penalty 1.0 and early stopping; temperature sampling uses temperature 0.7; top-k sampling uses $k = 50$; and top-p sampling uses $p = 0.9$. Unless otherwise stated, the tabulated U_{dec} results are reported using temperature sampling.

4.1.4 Evaluation Metrics

To assess how effectively each decomposed uncertainty component predicts model failures (hallucinations), we cast the problem as a binary classification task that distinguishes correct from incorrect model outputs. The correctness of a generation is determined using a fuzzy matching approach, where an output is labeled correct if its Rouge-L score (Lin and Och, 2004), which measures the length of the longest common subsequence with respect to the reference answer, is greater than or equal to 0.3.

Once correctness is established, we evaluate predictive performance using the Area Under the Receiver Operating Characteristic curve (AUROC), where higher values indicate stronger alignment between uncertainty scores and actual model errors. In addition, we report the Expected Calibration Error (ECE) to quantify how well the uncertainty scores correspond to true error rates.

4.2 Disentangling Uncertainty for Failure Detection

Table 1 presents the AUROC and ECE for each uncertainty component. The results demonstrate that the effectiveness of uncertainty decomposition is strongly **task-dependent**. On TriviaQA, both input ambiguity (U_{input}) and decoding randomness (U_{dec}) provide meaningful failure signals. For Llama 3 (8B), U_{dec} is the strongest predictor (AUROC 0.731), suggesting that uncertainty about factual recall often manifests as variability under stochastic generation. In contrast, for Gemma 3 (27B), U_{input} is most predictive (AUROC 0.761), indicating that failures in bigger models are driven more by sensitivity to how the question is phrased than by sampling noise.

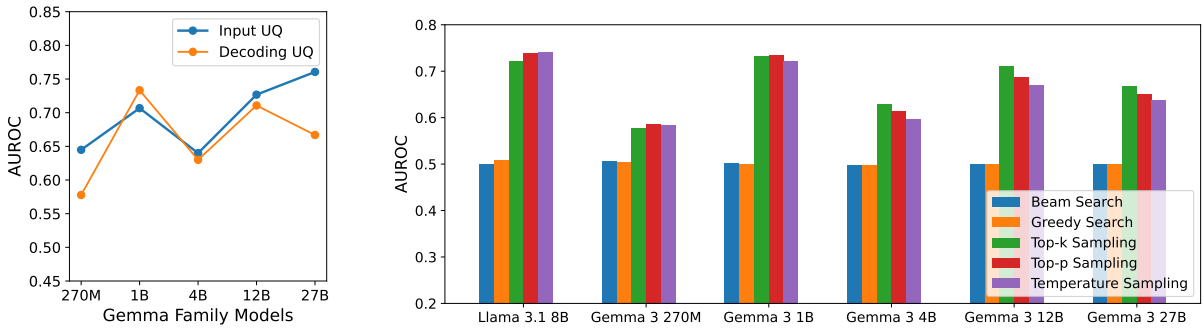
On GSM8k, although all uncertainty components yield weak failure prediction signals, we observe that knowledge-based uncertainty is comparatively less degraded than input or decoding-based uncertainty. This suggests that reasoning failures are less driven by ambiguity or sampling variability, and more by confident but incorrect internal reasoning trajectories.

4.3 Analysis of Scaling and Decoding Effects

To answer RQ2, we analyze how uncertainty sources evolve with model scale and decoding choices. Figure 4(a) illustrates the performance of input and decoding uncertainty for the Gemma 3 model family on TriviaQA. We observe no clear

Table 1: Failure prediction performance (AUROC) of each uncertainty component on TriviaQA (fact-retrieval) and GSM8K (reasoning). Higher values indicate a stronger ability to predict incorrect model responses. The results show that the most uncertainty source is task-dependent.

Dataset	Model	Input UQ		Knowledge UQ		Decoding UQ	
		AUROC	ECE	AUROC	ECE	AUROC	ECE
TriviaQA	Llama 3 (8B)	0.705	0.340	0.499	0.513	0.731	0.364
	Gemma 3 (27B)	0.761	0.223	0.498	0.514	0.636	0.458
GSM8K	Llama 3 (8B)	0.518	0.926	0.598	0.810	0.533	0.843
	Gemma 3 (27B)	0.334	0.920	0.500	0.827	0.383	0.861



(a) (b)

Figure 4: (a) Failure prediction performance (AUROC) of Input and Decoding uncertainty across the Gemma 3 model family on TriviaQA. As models scale, input ambiguity becomes a more reliable predictor of failure. (b) Comparison of failure prediction AUROC for Decoding Uncertainty when calculated using different decoding strategies. Stochastic methods (e.g., Top-k, Top-p) are significantly more effective at revealing uncertainty than deterministic ones (e.g., Greedy).

monotonic trend; the predictive power of both uncertainty types fluctuates with model size. However, a notable pattern emerges: for smaller models (1B), decoding uncertainty is a stronger predictor, while for larger models (12B, 27B), input ambiguity becomes the more reliable signal. This reinforces our finding from Table 1: as models grow, their sensitivity to input phrasing becomes a more prominent failure mode than simple generation variability.

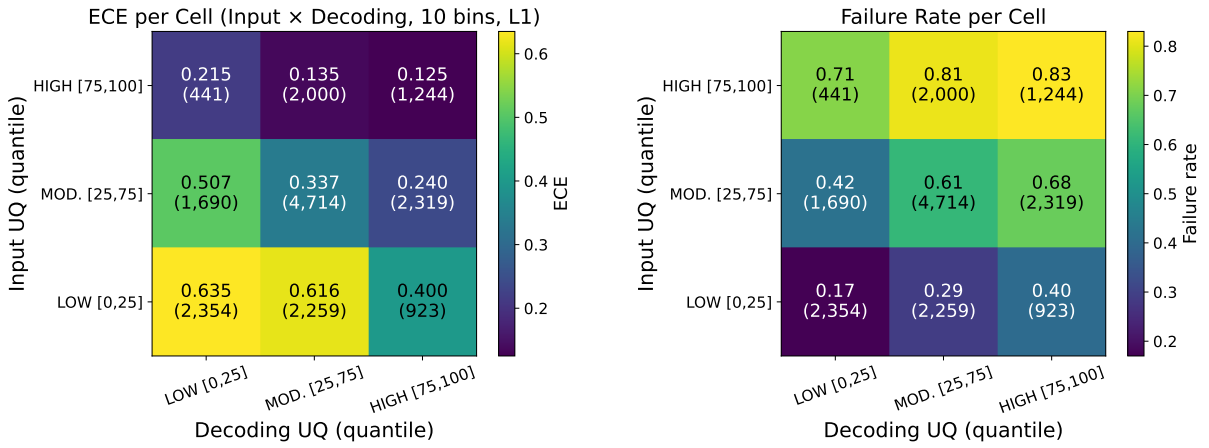
Figure 4(b) explores the impact of different decoding strategies on uncertainty-based failure detection. A consistent and striking pattern emerges across all models: stochastic decoding methods (Top-k, Top-p, and Temperature Sampling) yield significantly higher AUROC scores than deterministic methods (Beam Search, Greedy Search). This demonstrates that allowing the model to explore a diverse set of potential answers is important for revealing its underlying uncertainty. Deterministic methods, which force the model to commit to a single path, can mask this uncertainty, often leading to confidently incorrect answers.

4.4 Interaction of Uncertainty Sources

To better understand how different uncertainty sources interact, we performed a joint analysis of input ambiguity and decoding randomness. We partitioned the TriviaQA test set into a 3x3 grid based on low, moderate, and high quantiles of U_{input} and U_{dec}. We then computed the average model failure rate and ECE within each cell.

Figure 5(b) shows a clear and intuitive trend: the model's failure rate increases monotonically with both input and decoding uncertainty. The lowest failure rate (0.17) occurs when both uncertainty scores are low, while the highest rate (0.83) occurs when both are high. This confirms that both components are meaningful indicators of correctness, and their combined effect is even stronger.

However, Figure 5(a) reveals a more surprising relationship with calibration. The model is most poorly calibrated (highest ECE of 0.635) when it appears most confident (low input and decoding uncertainty). Conversely, it is best calibrated (lowest



(a) (b)

Figure 5: Joint analysis of Input Ambiguity (U_{input}) and Decoding Randomness (U_{dec}) on TriviaQA, partitioned by uncertainty quantiles. The heatmaps reveal an important insight about overconfidence: while the failure rate (b) increases with uncertainty, the model is most poorly calibrated (highest ECE in a) when it appears most confident (low uncertainty)

ECE of 0.125) when it is most uncertain. This suggests that the model is often underconfident. When the model signals low uncertainty on both axes, it is only wrong 17% of the time, but its confidence level is disproportionately low, leading to poor calibration. This highlights a critical failure mode: the model's confidence is least trustworthy precisely when it should appear most certain.

5 Discussion: Actionable Uncertainty Decomposition

Uncertainty decomposition is valuable not only for measuring confidence but also for understanding why LLMs fail in the first place. When uncertainty is represented by a single scalar score, it only indicates that the model is unsure, without revealing the underlying reason. As a result, the only practical response is often abstention or fallback generation. In contrast, separating uncertainty into interpretable components helps diagnose the source of failure. For example, some failures arise because the prompt itself is ambiguous and admits multiple valid interpretations. Recent work shows that LLMs can improve reliability by detecting such underspecified queries and asking clarifying questions before generating a response (Li et al., 2025; Yang et al., 2025).

Other failures can originate from gaps in the model's internal knowledge. Retrieval-based systems address this by augmenting the model with external information when parametric knowledge is insufficient. Self-Routing RAG (Wu et al., 2025), for example, uses uncertainty signals to decide whether a query should be answered using the model's internal knowledge or through external retrieval. However, recent analysis argues that standard predictive entropy fails to capture knowledge-related uncertainty in retrieval-augmented pipelines (Soudani et al., 2025). This further supports the importance of identifying the source of uncertainty.

Uncertainty can also arise during the generation process itself. In structured tasks such as code generation, reliability can depend strongly on token choices during decoding. Frameworks such as AdaDec therefore monitor token-level entropy and trigger additional search or reranking when decoding uncertainty becomes high (He et al., 2025). In summary, these works illustrate how decomposing uncertainty turns it into a practical signal for identifying model failures and improving LLM behavior, enabling targeted interventions such as clarification, retrieval, or adaptive decoding.

6 Conclusion

We present **a unified framework for decomposing LLM uncertainty** into three distinct **semantic (not probabilistic)** components: input-induced, knowledge-induced, and decoding-induced uncertainty. Each component is formalized as a categorical distribution over semantic equivalence classes of responses, enabling direct comparison via a common semantic entropy measure. Through systematic evaluation across fact-retrieval and reasoning

tasks, we show that the **dominant source of uncertainty is task- and model-dependent**. On TriviaQA, uncertainty decomposition yields meaningful failure signals with smaller models exhibit stronger decoding-driven uncertainty, while larger models are more sensitive to prompt phrasing (input ambiguity). Our findings challenge the common practice of relying on a single uncertainty estimate and highlight fundamental differences in how semantic uncertainty manifests across task and model types.

Limitations

While our framework provides a practical approach for decomposing uncertainty in LLMs, several aspects remain open for improvement. In particular, our method relies on practical operational proxies to estimate the different sources of uncertainty. For example, input-based uncertainty is estimated using a finite set of paraphrases rather than considering the full space of possible prompt variations. Similarly, knowledge-based uncertainty is approximated using a small ensemble of LoRA-adapted model realizations instead of performing full posterior inference over model parameters. Furthermore, our estimates depend on the quality of semantic clustering, which may be affected by errors in the underlying entailment model.

References

Yavuz Faruk Bakman, Duygu Nur Yaldiz, Sungmin Kang, Tuo Zhang, Baturalp Buyukates, Salman Avestimehr, and Sai Praneeth Karimireddy. 2025. Reconsidering llm uncertainty estimation methods in the wild. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 29531–29556.

Bertrand Charpentier, Ransalu Senanayake, Mykel Kochenderfer, and Stephan Günnemann. 2022. Disentangling epistemic and aleatoric uncertainty in reinforcement learning. *arXiv preprint arXiv:2206.01558*.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Wataru Hashimoto, Hidetaka Kamigaito, and Taro Watanabe. 2025. Decoding uncertainty: The impact of decoding strategies for uncertainty estimation in large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 14601–14613.

Kaifeng He, Mingwei Liu, Chong Wang, Zike Li, Yanlin Wang, Xin Peng, and Zibin Zheng. 2025. Towards better code generation: Adaptive decoding with uncertainty guidance. *arXiv preprint arXiv:2506.08980*.

Bairu Hou, Yujian Liu, Kaizhi Qian, Jacob Andreas, Shiyu Chang, and Yang Zhang. 2024. Decomposing uncertainty for large language models through input clarification ensembling. In *Proceedings of the 41st International Conference on Machine Learning*, pages 19023–19042.

Hsiu-Yuan Huang, Yutong Yang, Zhaoxi Zhang, Sanwoo Lee, and Yunfang Wu. 2024. A survey of uncertainty estimation in llms: Theory meets practice. *arXiv preprint arXiv:2410.15326*.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551*.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, and 1 others. 2022. Language models (mostly) know what they know. *arXiv preprint arXiv:2207.05221*.

Adam Tauman Kalai, Ofir Nachum, Santosh S Vempala, and Edwin Zhang. 2025. Why language models hallucinate. *arXiv preprint arXiv:2509.04664*.

Michael Kirchhof, Gjergji Kasneci, and Enkelejda Kasneci. 2025. Position: Uncertainty quantification needs reassessment for large-language model agents. *arXiv preprint arXiv:2505.22655*.

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

Belinda Z Li, Been Kim, and Zi Wang. 2025. Questbench: Can llms ask the right question to acquire information in reasoning tasks? *arXiv preprint arXiv:2503.22674*.

Chin-Yew Lin and Franz Josef Och. 2004. Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram statistics. In *Proceedings of the 42nd annual meeting of the association for computational linguistics (ACL-04)*, pages 605–612.

Potsawee Manakul, Adian Liusie, and Mark JF Gales. 2023. Selfcheckgpt: Zero-resource black-box hallucination detection for generative large language models. *arXiv preprint arXiv:2303.08896*.

Guillem Ramírez, Alexandra Birch, and Ivan Titov. 2024. Optimising calls to large language models with uncertainty-based two-tier selection. *arXiv preprint arXiv:2405.02134*.

Ransalu Senanayake. 2024. The role of predictive uncertainty and diversity in embodied ai and robot learning. In *Metacognitive Artificial Intellegence, Cambridge University Press*.

Heydar Soudani, Evangelos Kanoulas, and Faegheh Hasibi. 2025. Why uncertainty estimation methods fall short in rag: An axiomatic analysis. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 16596–16616.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, and 1 others. 2025. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*.

Di Wu, Jia-Chen Gu, Kai-Wei Chang, and Nanyun Peng. 2025. Self-routing rag: Binding selective retrieval with knowledge verbalization. *arXiv preprint arXiv:2504.01018*.

Qiujie Xie, Qingqiu Li, Zhuohao Yu, Yuejie Zhang, Yue Zhang, and Linyi Yang. 2025. An empirical analysis of uncertainty in large language model evaluations. *arXiv preprint arXiv:2502.10709*.

Chenyang Yang, Yike Shi, Qianou Ma, Michael Xieyang Liu, Christian Kästner, and Tongshuang Wu. 2025. What prompts don't say: Understanding and managing underspecification in llm prompts. *arXiv preprint arXiv:2505.13360*.

Hanna Zubkova, Ji-Hoon Park, and Seong-Whan Lee. 2025. Sugar: Leveraging contextual confidence for smarter retrieval. In *ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE.

A Appendix

A.1 Compute Resources

All local experiments were conducted on a single NVIDIA H100 GPU (80 GB HBM) running Ubuntu 20.04 with CUDA 11.8. This setup was used for both LoRA ensemble training and local HuggingFace-based inference for uncertainty estimation on TriviaQA and GSM8K. Paraphrase generation for input ambiguity estimation was performed separately through the GPT-5-nano API.

A.2 Prompt Templates

For input ambiguity estimation, we generate semantically equivalent paraphrases of each input question using GPT-5-nano. Figure 6 shows the instruction and input prompt template used for paraphrase generation.

System and Input prompt template

System Prompt:
"For question Q, provide 5 semantically equivalent questions. Preserve the original meaning and context of the question while changing the structure and words. Output format: {0: "question1", 1: "question2", 2: "question3", ...}."

Input Prompt:
"Q: [original question]"

Figure 6: Prompt template used for paraphrase generation when estimating input ambiguity.

A.3 Decoding Hyperparameters

For decoding-based uncertainty, we generate repeated outputs under a fixed decoding strategy and compute semantic entropy across these outputs. Unless otherwise stated, we use 5 generations per prompt. We evaluate multiple practical decoding strategies: greedy decoding, beam search with 5 beams, temperature sampling with temperature 0.7, top-k sampling with $k = 50$, and top-p sampling with $p = 0.9$. Beam search uses length penalty 1.0 and early stopping. The main results reported for U_{dec} use temperature sampling unless specified otherwise.

A.4 LoRA Ensemble Training Details

To estimate knowledge-based uncertainty, we approximate multiple model realizations using an ensemble of LoRA-adapted models trained with different random seeds. For each base model, we train 5 LoRA adapters. The LoRA configuration

Table 2: LoRA ensemble training hyperparameters.

Hyperparameter	Value
Number of LoRA models	5
LoRA rank r	8
LoRA α	32
LoRA dropout	0.1
Bias	none
Target modules	q_proj, v_proj
Learning rate	2×10^{-5}
Batch size	4
Gradient accumulation	2
Epochs	1
Maximum sequence length	1024

uses rank $r = 8$, $\alpha = 32$, dropout 0.1, bias set to `none`, and target modules `q_proj` and `v_proj`. We train each adapter for 1 epoch with learning rate 2×10^{-5}, per-device batch size 4, and gradient accumulation steps 2. The maximum sequence length is set to 1024. For each input, one response is generated from each ensemble member using greedy decoding, and semantic entropy over these responses is used to estimate $U_{\text{knowledge}}$.

A.5 Evaluation Metrics

We evaluate each uncertainty component as a predictor of model failure. Let a_i denote the reference answer for example i, let $\hat{a}_i$ denote the model's final answer, and let u_i denote the corresponding uncertainty score (e.g., U_{input}, $U_{\text{knowledge}}$, or U_{dec}).

To determine whether a prediction is correct, we use bidirectional semantic equivalence based on a natural language inference (NLI) model. Specifically, an answer is marked correct only if the reference answer entails the generated answer and the generated answer also entails the reference answer. Formally, the correctness label is defined as,

$$z_i = \mathbf{1}\Big[\, p_{\text{NLI}}(a_i \Rightarrow \hat{a}_i) \geq \gamma \\ \wedge\; p_{\text{NLI}}(\hat{a}_i \Rightarrow a_i) \geq \gamma \Big] \tag{5}$$

where $p_{\text{NLI}}(\cdot \Rightarrow \cdot)$ denotes the entailment probability returned by the NLI model and $\gamma = 0.5$ in our experiments. We then define the failure label as,

$$f_i = 1 - z_i, \tag{6}$$

so that $f_i = 1$ indicates an incorrect response and $f_i = 0$ indicates a correct one.

AUROC. To evaluate how well an uncertainty score separates failures from correct responses, we compute the Area Under the Receiver Operating Characteristic curve (AUROC). AUROC measures the probability that a randomly chosen failed example receives a higher uncertainty score than a randomly chosen correct example. It can be written as,

$$\text{AUROC} = \frac{1}{|\mathcal{P}||\mathcal{N}|} \sum_{i \in \mathcal{P}} \sum_{j \in \mathcal{N}} \mathbf{1}[u_i > u_j], \tag{7}$$

where $\mathcal{P} = \{i : f_i = 1\}$ is the set of failed examples and $\mathcal{N} = \{j : f_j = 0\}$ is the set of correct examples. Higher AUROC indicates that the uncertainty score is more effective at ranking incorrect generations above correct ones.

Expected Calibration Error (ECE). In addition to ranking performance, we evaluate calibration to measure whether larger uncertainty values correspond to higher empirical failure rates. Since ECE requires a confidence-like quantity in $[0, 1]$, we first map each uncertainty score u_i to a failure-confidence score $\hat{p}_i \in [0, 1]$ using a monotonic normalization. We then partition predictions into B bins according to $\hat{p}_i$. For bin b, let $\mathcal{B}_b$ denote the set of assigned examples. The average predicted failure confidence and empirical failure rate in that bin are,

$$\begin{aligned} \text{conf}_b &= \frac{1}{|\mathcal{B}_b|} \sum_{i \in \mathcal{B}_b} \hat{p}_i, \\ \text{err}_b &= \frac{1}{|\mathcal{B}_b|} \sum_{i \in \mathcal{B}_b} f_i. \end{aligned} \tag{8}$$

The Expected Calibration Error is then defined as,

$$\text{ECE} = \sum_{b=1}^{B} \frac{|\mathcal{B}_b|}{n} \left| \text{err}(\mathcal{B}_b) - \text{conf}(\mathcal{B}_b) \right|, \tag{9}$$

where n is the total number of examples. Lower ECE indicates better calibration, meaning that the uncertainty-derived confidence values more closely match the observed failure frequencies.