D^2-RAG: Dual-Decision Retrieval-Augmented Generation via Multi-Dimensional Uncertainty and Utility-Aware Decoding

Jinshuo Zhang[1,3], Xiaoding Zhou[1,3], Weiyu Zhang[1,3,*], Guoqiang Chen[2], Ying Lian[2], Xiaoyang Meng[1,3], Yonghe Chen[1,3], Hongjiao Guan[1,3], Jiasheng Si[1,3], Wenpeng Lu[1,3]

[1]Key Laboratory of Computing Power Network and Information Security, Ministry of Education, Shandong Computer Science Center (National Supercomputer Center in Jinan), Qilu University of Technology (Shandong Academy of Sciences), Jinan, China
[2]First Affiliated Hospital of Shandong First Medical University, Jinan, China
[3]Shandong Provincial Key Laboratory of Computing Power Internet and Service Computing, Shandong Fundamental Research Center for Computer Science, Jinan, China
10431240242@stu.qlu.edu.cn, zwy@qlu.edu.cn

Abstract

Retrieval-Augmented Generation (RAG) mitigates hallucinations in large language models by incorporating external knowledge. However, retrieval does not always return relevant documents and may return noisy ones. Indiscriminately retrieving and utilizing this external knowledge can interfere with the model's originally correct reasoning. In this work, we propose Dual-Decision Retrieval-Augmented Generation (D^2-RAG), which integrates multi-dimensional uncertainty estimation to decide whether to retrieve and employs adaptive contrastive decoding to handle retrieved contexts of varying quality. Specifically, we first integrate uncertainty estimation scores that assess model uncertainty from multiple perspectives, construct them into a comprehensive feature vector, and train a lightweight retrieval decision model to accurately identify the model's knowledge boundaries and determine whether to retrieve. Subsequently, we dynamically adjust the contrastive decoding strategy based on the utility of retrieved contexts to enhance the utilization of relevant contexts while suppressing interference from noisy contexts. Extensive experiments on four medical question-answering datasets demonstrate that D^2-RAG significantly outperforms baselines, enabling retrieval-augmented Llama3.1-8B to surpass non-retrieval-augmented Llama3.1-70B on the MedMCQA dataset. The source code is available on https://github.com/zakelawen/d--rag.

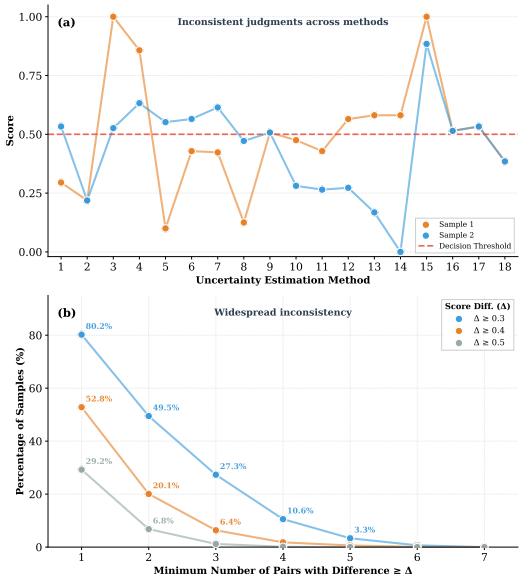

Figure 1: Inconsistency of uncertainty estimation methods. (a) Distribution of scores for two samples across 18 methods. (b) Percentage of samples with method pairs exhibiting score differences $\geq \Delta$.

1 Introduction

Large language models (LLMs) have demonstrated remarkable capabilities across various domains (Team et al., 2025; Achiam et al., 2023). However, LLMs often suffer from hallucinations due to insufficient knowledge, generating fluent yet factually incorrect text. This significantly limits their application in specialized domains such as medicine (Williams et al., 2024; Xie et al., 2024; Lu et al., 2025; Wei et al., 2025) and finance (Li et al., 2023b).

RAG has made significant progress in reducing

* Corresponding author

Findings of the Association for Computational Linguistics: ACL 2026, pages 15355–15382
July 2-7, 2026 ©2026 Association for Computational Linguistics

factual errors in LLMs by retrieving external knowledge (Gao et al., 2023; Lewis et al., 2020). Despite its effectiveness, it still faces certain challenges. Specifically, RAG retrieves external knowledge regardless of whether the model's internal knowledge is sufficient (Jeong et al., 2024). This leads to additional computational cost and may introduce noisy documents that interfere with the model's reasoning (Amiraz et al., 2025). Furthermore, even when relevant external knowledge is retrieved, the model tends to rely on its parametric knowledge rather than the external knowledge (Sun et al., 2025). Researchers have explored various approaches to address these challenges. Adaptive retrieval methods based on uncertainty estimation assess the model's confidence in its responses to determine knowledge boundaries, thereby deciding whether to trigger retrieval (Kuhn et al., 2023; Kadavath et al., 2022). These methods often outperform complex adaptive pipelines in terms of efficiency and self-knowledge (Moskvoretskii et al., 2025). Additionally, some studies (Shi et al., 2024; Qiu et al., 2025; Kim et al., 2024a) introduce contrastive decoding during inference to guide models toward generating responses based on retrieved contexts.

Despite significant progress in existing work, two key limitations remain. **First, existing uncertainty estimation methods often yield inconsistent judgments for the same query, making it unreliable to rely on any single method for retrieval decision.** We empirically analyze the judgment consistency of 18 uncertainty estimation methods. Specifically, we calibrate and normalize their scores to [0,1] and pair them symmetrically after sorting. As shown in Figure 1, different methods yield significantly divergent scores for the same sample, even leading to opposite retrieval decisions. A large number of samples exhibit substantial disagreement across methods, with over 51% having at least 2 method pairs with score differences exceeding 0.3. **Second, existing contrastive decoding methods apply the same strategy to both relevant and noisy contexts when utilizing retrieved contexts.** Retrieved results may contain relevant evidence or noisy information. Indiscriminately amplifying the influence of all retrieved content allows noisy passages to receive the same enhancement as relevant evidence, thereby exacerbating the model's tendency to generate incorrect answers.

To address these limitations, we propose D^2-RAG (Dual-Decision RAG), a framework that achieves adaptive knowledge retrieval and utilization through a two-stage decision mechanism. In the first stage, we construct an uncertainty-aware retrieval decision model to more accurately determine whether retrieval is needed. We attribute the inconsistent judgments of uncertainty methods to the fact that a single method can only reflect a portion of model uncertainty. Therefore, we integrate multiple uncertainty estimation methods to compute uncertainty scores of the language model for a given query, and construct them into a comprehensive feature vector. We then train a lightweight classifier to learn the complementary and conflicting patterns among these methods, thereby making more reliable retrieval decisions. In the second stage, we propose utility-aware contrastive decoding to more properly utilize retrieved content of varying quality. We first employ a semantic perplexity metric (Dai et al., 2025) to evaluate the utility of retrieved context. For relevant context, we leverage contrastive decoding to guide the model to fully utilize retrieved information. For noisy contexts, instead of simply discarding them, we treat them as negative references and leverage contrastive decoding to guide the model to actively avoid similar errors. In this way, the model can adaptively utilize retrieved content based on its quality.

Our main contributions are summarized as follows: (1) We propose **D^2-RAG**, a framework that achieves adaptive knowledge retrieval and utilization through a dual-stage decision mechanism, effectively reducing noise interference while leveraging relevant external knowledge. (2) We construct an **uncertainty-aware retrieval decision model** that integrates multi-dimensional uncertainty signals into a unified representation, enabling more reliable retrieval decisions. (3) We propose **utility-aware contrastive decoding**, which dynamically adjusts decoding strategies based on context quality, guiding the model to leverage relevant context while suppressing noise-related errors. (4) **Extensive experiments** demonstrate that D^2-RAG significantly outperforms baselines.

2 Related Work

Uncertainty Estimation. Prior research (Shorinwa et al., 2025; Kuhn et al., 2023; Kadavath et al., 2022) explores the knowledge boundaries of models by quantifying their uncertainty to mitigate hallucinations. These uncertainty estimation meth-

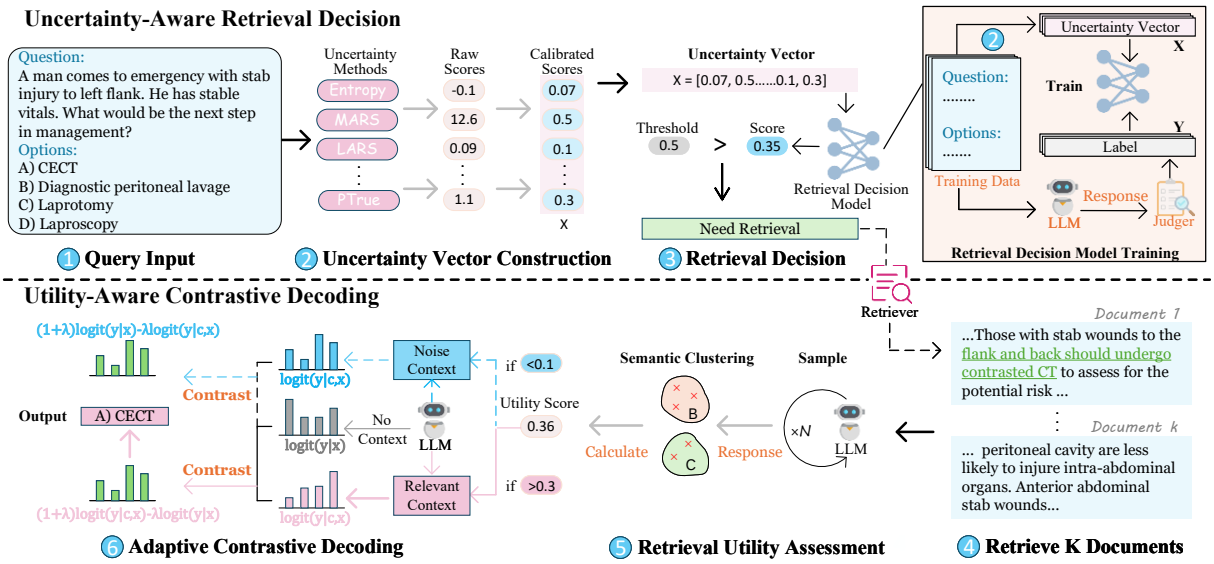

Figure 2: Overview of the D^2-RAG framework, which consists of two stages: (1) uncertainty-aware retrieval decision that leverages multi-dimensional uncertainty signals, (2) utility-aware contrastive decoding that adaptively adjusts strategies based on context quality.

ods assess model uncertainty from four perspectives: probability (Malinin and Gales, 2021; Bakman et al., 2024; Duan et al., 2024), internal states (Chen et al., 2024; Sriramanan et al., 2024), output consistency (Lin et al., 2023; Zhao et al., 2024), and self-checking (Tian et al., 2023). Our empirical analysis reveals significant inconsistency among different methods for the same query. Recent work (Bakman et al., 2025) shows that combining multiple uncertainty estimation methods is a promising approach through exploratory experiments. Inspired by this, we integrate uncertainty signals from multiple dimensions and train a retrieval decision model to capture complementary and conflicting patterns among these signals.

Contrastive Decoding. Prior studies (Sun et al., 2025) have shown that even when relevant documents are retrieved, models tend to rely on their parametric knowledge for generation. To address this, some work (Shi et al., 2024; Qiu et al., 2025) introduces contrastive decoding (Li et al., 2023a) during inference to guide models toward generating responses based on context. However, these methods overlook that retrieved content may contain noise, and applying the same strategy to noisy documents as to relevant ones may interfere with the model's correct reasoning. Recent work (Kim et al., 2024a; Qiu et al., 2025) mitigates this by dynamically adjusting contrastive strength based on entropy, but entropy as a quality indicator has inherent limitations. It reflects model confidence rather than correctness, and models may confidently generate incorrect answers based on noisy contexts. Instead, we introduce a semantic perplexity metric (Dai et al., 2025) to more reliably assess retrieval utility. Unlike existing work (Kim et al., 2024a), we treat noisy documents as negative references, guiding models to actively avoid similar errors.

3 D^2-RAG

In this section, we introduce D^2-RAG (Dual-Decision RAG). The overall framework is illustrated in Figure 2, with the algorithm presented in Algorithm 1 in Appendix. D^2-RAG consists of two stages: Uncertainty-Aware Retrieval Decision (Section 3.2) and Utility-Aware Contrastive Decoding (Section 3.3).

3.1 Preliminaries

Retrieval-Augmented Generation. Given a query q and an external document corpus $\mathcal{D}$, RAG first retrieves passages $\mathcal{C} = \mathcal{R}(q, \mathcal{D})$ from the corpus via a retriever $\mathcal{R}$, then concatenates the retrieved passages with the query for the language model $\mathcal{M}$ to generate an answer: $a = \mathcal{M}(q, \mathcal{C})$.

Contrastive Decoding. Contrastive decoding adjusts the generation process by contrasting output distributions under two conditions. CAD (Shi et al., 2024) contrasts distributions with and without retrieval context to enhance the model's attention to external evidence:

$$y_t \sim \text{softmax}\left[(1+\alpha)\,\text{logit}_\theta(y_t \mid \mathcal{C}, q, y_{<t}) - \alpha\,\text{logit}_\theta(y_t \mid q, y_{<t})\right], \quad (1)$$

where θ denotes the model parameters and α is a hyperparameter controlling the contrast strength.

3.2 Uncertainty-Aware Retrieval Decision

Retrieval may introduce noisy information when the model already possesses the knowledge required to answer a question. It is crucial for RAG to determine whether to retrieve. Uncertainty estimation is an important approach for determining the model knowledge boundaries (Shorinwa et al., 2025). Although existing uncertainty estimation methods have demonstrated promising performance, their judgments for the same query are significantly inconsistent (as shown in Figure 1). Since a single method can only reflect a portion of model uncertainty, we integrate multiple uncertainty estimation methods into a unified representation to more reliably determine whether to retrieve.

3.2.1 Uncertainty Vector Construction

Existing uncertainty estimation methods can be broadly categorized into four types based on their perspectives: *(1) Probability-Based Methods:* Quantify uncertainty directly using the model's output probability distribution. *(2) Internal State-Based Methods:* Quantify uncertainty through activations from intermediate or final layers. *(3) Output Consistency-Based Methods:* Generate candidate answers via multiple sampling and quantify uncertainty based on answer consistency. *(4) Self-Checking Methods:* Leverage the metacognition ability of LLMs to let the model directly assess the confidence of its answers.

To ensure comprehensive coverage across these complementary perspectives, we select K representative uncertainty estimation methods from the above four categories as features, such as Entropy and Ptrue (see Appendix A for the complete list). To obtain these features, for each query, we first prompt the model generate an answer without context, then compute uncertainty scores using these K methods to obtain the raw feature vector $\mathbf{s} = [s_1, s_2, \ldots, s_K]$. Since different methods produce scores with inconsistent ranges and semantic directions (e.g., higher entropy indicates greater uncertainty, while higher confidence indicates greater certainty), we use Isotonic Regression (Han et al., 2019) to calibrate each raw score s_k to a unified [0,1] interval (see Appendix B for details), obtaining the calibrated uncertainty vector:

$$\mathbf{u} = [u_1, u_2, \ldots, u_K], \quad u_k = \text{IR}_k(s_k), \quad (2)$$

where $\text{IR}_k(\cdot)$ is the Isotonic Regression model for the k-th score, and higher calibrated scores indicate greater model certainty.

3.2.2 Retrieval Decision Model

We formulate the retrieval decision as a binary classification task. The retrieval decision model $\mathcal{M}_{\text{ret}}$ takes the calibrated feature vector $\mathbf{u}$ as input and predicts the probability p that the model can correctly answer the query using its parametric knowledge. If correct, retrieval is skipped. Otherwise, retrieval is triggered.

We explore two lightweight classifiers: Logistic Regression (LR) and Multi-Layer Perceptron (MLP), to capture linear and non-linear relationships among features, respectively:

$$p = \sigma(\mathbf{w}^\top \mathbf{u} + b) \quad \text{(LR)}, \quad (3)$$

$$p = \sigma(\text{MLP}(\mathbf{u})) \quad \text{(MLP)}, \quad (4)$$

where $\mathbf{w}$ is the weight vector, b is the bias, and σ is the sigmoid function.

We construct training data in an automated manner. For each sample (q, a_{gold}) in the training set, we first generate an answer without retrieval: $a_{\text{para}} = \mathcal{M}(q)$. To accurately evaluate the correctness of a_{para}, we adopt the LLM-as-a-Judge (Li et al., 2025) approach, using GPT-4o to determine whether a_{para} is correct:

$$y = \begin{cases} 1 \text{ (No Retrieve)}, & \text{if } a_{\text{para}} \text{ is correct} \\ 0 \text{ (Retrieve)}, & \text{if } a_{\text{para}} \text{ is incorrect} \end{cases} \quad (5)$$

We train the retrieval decision model using binary cross-entropy loss.

$$\mathcal{L}(\theta) = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log p_i + (1-y_i)\log(1-p_i)\right], \quad (6)$$

where N is the number of training samples, and y_i and $\mathbf{u}_i$ are the label and calibrated feature vector of the i-th sample. During inference, we set a threshold τ to control retrieval behavior, triggering retrieval when $p \leq \tau$. The threshold τ is selected on the validation set.

3.3 Utility-Aware Contrastive Decoding

Once retrieval is triggered in the first stage, we perform retrieval and obtain a set of documents $\mathcal{C}$. Existing methods treat all retrieved content indiscriminately, making it difficult to fully exploit relevant evidence while remaining susceptible to noise. To address this, we propose utility-aware contrastive decoding, which first assesses the utility of retrieved content and then dynamically adjusts the decoding strategy based on the assessment.

Retrieval Utility Assessment. Following Dai et al. (2025), we compute the reduction in semantic perplexity before and after retrieval as the retrieval utility score to evaluate the utility of retrieved content. When the utility score exceeds the threshold τ_{pos}, the context is classified as a relevant context. When it falls below τ_{neg}, the context is classified as a noisy context. Otherwise, we apply standard decoding without contrastive adjustment (Detailed threshold settings are provided in Appendix C).

Adaptive Contrastive Decoding. Based on the retrieval utility score, we dynamically adjust the "expert model" and "amateur model" in contrastive decoding, introducing specific strength scaling coefficients for different contexts. We first define two logit: $\text{logit}_{\text{RAG}}(y_t) = \text{logit}_\theta(y_t \mid \mathcal{C}, q, y_{<t})$, $\text{logit}_{\text{Para}}(y_t) = \text{logit}_\theta(y_t \mid q, y_{<t})$.

Based on context quality, our decoding strategy dynamically adjusts as follows:

Relevant Context ($\mathcal{C}_{\text{pos}}$). When the retrieved content is identified as relevant, we use the RAG-augmented model as the expert model, actively suppressing tokens that dominate only in $\text{logit}_{\text{Para}}$(i.e., potential hallucinations from internal memory or knowledge conflicting with evidence), thereby anchoring generation to external evidence and improving factual accuracy.

$$\begin{aligned}\text{logit}_{\text{final}}(y_t) = (1+\lambda_t)\cdot \text{logit}_{\text{RAG}}(y_t) \\ - \lambda_t \cdot \text{logit}_{\text{Para}}(y_t).\end{aligned} \tag{7}$$

Noisy Context ($\mathcal{C}_{\text{neg}}$). When the retrieved content is identified as noisy, we use the RAG-augmented model as the amateur model, actively suppressing tokens with high scores only in $\text{logit}_{\text{RAG}}$(i.e., tokens induced by noisy context), reducing the risk of the model being misled by $\mathcal{C}_{\text{neg}}$.

$$\begin{aligned}\text{logit}_{\text{final}}(y_t) = (1+\lambda_t)\cdot \text{logit}_{\text{Para}}(y_t) \\ - \lambda_t \cdot \text{logit}_{\text{RAG}}(y_t).\end{aligned} \tag{8}$$

Neutral Context. When utility falls in the middle range, we apply standard RAG decoding without contrastive adjustment.

For the relevant and noisy cases, $\lambda_t = \alpha \cdot w_t$, where $\alpha \in \{\alpha_{\text{pos}}, \alpha_{\text{neg}}\}$ is the strength scaling coefficient for each case. The parameter λ_t is determined by two components: (1) strength scaling coefficients $(\alpha_{\text{pos}}, \alpha_{\text{neg}})$ are preset hyperparameters controlling the strength of positive enhancement and negative suppression. (2) dynamic entropy factor w_t is computed in real time based on Shannon entropy at the current generation step, adaptively increasing contrast intensity when model uncertainty is high, allowing weights to vary dynamically during generation (see Appendix D for details).

4 Experiments

4.1 Experimental Setup

Datasets. We use four public medical multiple-choice question-answering datasets as test datasets: *MedMCQA* (Pal et al., 2022), *MedQA* (Jin et al., 2020), *MedExQA* (Kim et al., 2024b), and *MMLU* (Hendrycks et al., 2021). Data examples are provided in Appendix E.

Baselines. We compare D^2-RAG with four categories of baseline methods. **LLM-Only:** These methods do not rely on external knowledge, using only the LLM's internal knowledge for reasoning, including Zero Shot (Wei et al., 2021) and CoT (Wei et al., 2022). **SFT:** These methods fine-tune models using labeled datasets. We select two representative methods: PMC-Llama (Wu et al., 2024) and MedAlpaca (Han et al., 2023). **RAG:** These methods enhance language model capabilities by incorporating external knowledge. We select three representative methods: Naive RAG (Lewis et al., 2020), Self-RAG (Asai et al., 2024) and Adaptive-RAG (Jeong et al., 2024). **RAG+CD:** These methods enhance language model capabilities through contrastive decoding. We select two representative methods: CAD (Shi et al., 2024) and DoLa (Chuang et al., 2024). Detailed descriptions of each method are provided in Appendix M.

Evaluation Metric. We use different metrics for different experiments: (1) *Accuracy*, to evaluate the overall performance of D^2-RAG on medical QA tasks; (2) *F1 score* and *recall* for comparing retrieval decision models; (3) *AUROC* and *PRR*, for evaluating integrated versus single uncertainty features. See Appendix N for details.

Implementation Details. We select Llama3.1-8B and Qwen2.5-7B as backbone models. For the

embedding model, we use Qwen3-Embedding-4B (Zhang et al., 2025). We use 18 uncertainty estimation methods ($K = 18$). For baseline methods, we follow the default settings from their original papers. See Appendix G for more details on data setting. Our RAG knowledge base comprises 18 medical textbooks that serve as key references for the United States Medical Licensing Examination.

4.2 Experimental Results

4.2.1 Overall Performance

Table 1 presents the comparative results between D^2-RAG and baselines on four medical QA datasets. Overall, D^2-RAG consistently outperforms all baselines across different backbone models (Qwen2.5-7B and Llama3.1-8B). Based on the results, we get the following analysis:

(1) Compared with other RAG methods, D^2-RAG demonstrates substantial improvements on all datasets. D^2-RAG consistently outperforms Naive RAG, which indiscriminately retrieves for all queries; Adaptive RAG, which trains a classifier to categorize query difficulty; and Self-RAG, which relies on self-reflection mechanisms. This advantage mainly stems from D^2-RAG's ability to accurately determine whether retrieval is needed through multi-dimensional uncertainty signals, reducing unnecessary retrieval, while effectively differentiating retrieved content quality and suppressing noise interference through utility-aware contrastive decoding.

(2) D^2-RAG also demonstrates consistent advantages over other methods that combine contrastive decoding with retrieval. CAD and DoLa apply uniform decoding strategies regardless of retrieved content quality. In contrast, our utility-aware contrastive decoding dynamically assesses retrieval utility via the semantic perplexity metric, amplifying the contribution of relevant context while actively suppressing interference from noisy content.

(3) Notably, Llama3.1-8B with D^2-RAG surpasses the much larger Llama3.1-70B model on MedMCQA. This demonstrates that D^2-RAG effectively compensates for the limited parametric knowledge of smaller models, enabling them to achieve or even exceed the performance of larger models at a fraction of the computational cost.

4.2.2 Ablation Study

To validate the contribution of each component in D^2-RAG, we conduct ablation experiments by removing the retrieval decision model and contrastive decoding respectively to evaluate their impact on overall performance. Table 3 presents the performance changes on MedQA and MedMCQA after removing different components. We compare the following settings: removing both the retrieval decision model and contrastive decoding (w/o Stage 1 & 2), removing only the retrieval decision model (w/o Stage 1), and removing only contrastive decoding (w/o Stage 2).

Results show that when both the retrieval decision model and contrastive decoding are removed (w/o Stage 1 & 2), the performance drop is most significant, indicating that standard RAG cannot effectively handle retrieval noise. When only the retrieval decision model is removed (w/o Stage 1), the performance drop is relatively small, because although retrieving for all queries introduces noisy documents, utility-aware contrastive decoding can dynamically adjust the decoding strategy based on content quality, partially mitigating the negative impact of noise. When contrastive decoding is removed (w/o Stage 2), the performance drop is more substantial, indicating that utility-aware contrastive decoding plays a more critical role in the overall framework. Standard decoding neither fully leverages external knowledge from relevant documents nor resists being misled by noisy documents.

The complete D^2-RAG achieves the best performance across all settings, demonstrating the effectiveness and synergy of the two components, where the retrieval decision model reduces the probability of introducing noise while utility-aware contrastive decoding improves the utilization efficiency of retrieved content.

4.2.3 In-depth Analysis

How does our retrieval decision model perform compared to adaptive retrieval baselines? To validate the effectiveness of our retrieval decision model, we compare it with existing adaptive retrieval methods (FLARE and Self-RAG). Experimental settings are detailed in Appendix H. As shown in Table 2, our method significantly outperforms baselines on both datasets. For the "Retrieval Needed" category, our method achieves 93.73% recall and 69.91% F1 on MedMCQA, surpassing FLARE by 5.1% and 10.62%, respectively.

Results show that FLARE achieves high recall in the "Retrieval Needed" category, effectively capturing knowledge gaps, but performs poorly in the "No Retrieval Needed" category, leading to over-

Method	MedMCQA		MedQA		MedExQA		MMLU-Medical	
PMC-Llama	27.04		16.94		36.30		37.01	
MedAlpaca	36.22		50.04		50.11		60.85	
Self-RAG	38.32		28.26		71.38		64.27	
Llama3.1-70B	67.00		64.42		77.34		78.05	
	Qwen2.5-7B	**Llama3.1-8B**	**Qwen2.5-7B**	**Llama3.1-8B**	**Qwen2.5-7B**	**Llama3.1-8B**	**Qwen2.5-7B**	**Llama3.1-8B**
Zero Shot	45.60	65.56	24.86	37.10	67.45	71.38	63.54	68.29
CoT	50.76	54.82	29.68	52.06	63.09	56.28	57.56	55.55
Naive RAG	54.04	62.50	40.10	46.16	69.36	66.28	63.96	61.77
Adaptive RAG	61.36	68.56	47.74	56.50	71.06	69.47	67.44	66.71
DoLa	67.16	70.40	56.72	57.30	73.19	70.85	71.83	70.49
CAD	62.54	63.48	55.04	57.12	68.94	62.66	67.07	62.93
Ours	**68.76**	**71.30**	**57.38**	**57.88**	**75.74**	**71.60**	**73.84**	**72.44**

Table 1: Accuracy (%) comparison of D^2-RAG and baselines on four medical QA datasets. The upper section presents fine-tuned model baselines. The lower section shows results with Qwen2.5-7B and Llama3.1-8B backbones.

Methods	Retrieval Needed		No Retrieval Needed	
	Recall	F1	Recall	F1
		MedMCQA		
FLARE (Llama3.1-8B)	88.63	59.29	48.87	63.41
Self-RAG-7B	40.62	47.01	62.54	54.02
Self-RAG-13B	2.54	4.90	**87.16**	56.84
Ours (Llama3.1-8B)	**93.73**	**69.91**	79.49	**79.78**
Ours (Qwen2.5-7B)	82.46	60.32	46.46	59.26
		MedQA		
FLARE (Llama3.1-8B)	92.80	59.18	21.33	33.86
Self-RAG-7B	43.13	53.27	50.19	32.12
Self-RAG-13B	12.67	20.76	**97.79**	**45.78**
Ours (Llama3.1-8B)	**93.35**	64.45	22.84	35.64
Ours (Qwen2.5-7B)	90.55	**66.58**	25.90	38.51

Table 2: Comparison of retrieval decision methods on MedMCQA and MedQA.

Method	MedMCQA		MedQA	
	Llama3.1-8B	Qwen2.5-7B	Llama3.1-8B	Qwen2.5-7B
w/o Stage 1 & 2	62.50	54.04	46.16	40.10
w/o Stage 1	69.44	67.14	57.62	56.96
w/o Stage 2	66.16	59.02	51.46	41.38
Ours	**71.30**	**68.76**	**57.88**	**57.38**

Table 3: Ablation study of D^2-RAG on MedMCQA and MedQA.

retrieval. Self-RAG shows the opposite pattern, with high recall in "No Retrieval Needed" but missing many questions requiring external knowledge in the medical domain. In contrast, D^2-RAG integrates multiple uncertainty estimation methods to comprehensively assess knowledge boundaries, achieving an optimal balance between the two categories.

Does multi-perspective uncertainty integration perform better compared with single uncertainty metrics? To validate the advantages of comprehensive uncertainty feature vectors, we conduct two sets of experiments. First, we compare our method with 18 uncertainty estimation methods. Results for the Llama model on MedQA are shown in Figure 3 (top), with additional results in Appendix J. Our method significantly outperforms all single methods, consistently surpassing the best baseline (PTrue). Across different categories, probability-based methods perform relatively well overall. Output consistency-based methods show mixed results, with some performing well while others yield negative PRR scores, indicating performance worse than random strategy under the current setting. Internal state-based methods perform poorly, likely due to the limited representational discrimination capability of medium-scale models. Self-checking methods also show limited performance, possibly constrained by the immature metacognition capabilities of medium-scale models. By integrating perspectives from all methods, our approach more accurately and comprehensively evaluates model knowledge boundaries, achieving notable improvements over the best single method.

Second, we conduct cross-dataset experiments to test generalization: training on one dataset and testing on another. Results are shown in Figure 3 (bottom), with Llama model results in Appendix I. Even when D^2-RAG is trained on a different dataset, it still outperforms baselines trained and tested on the same dataset, while single methods' rankings fluctuate dramatically across datasets. These results demonstrate that by integrating four categories of features, our method captures essential characteristics of model knowledge boundaries rather than dataset-specific patterns, achieving stronger robustness and generalization.

How do the strength scaling coefficients affect the adaptive contrastive decoding? To investigate the effect of the contrastive strength coefficients α_{pos} and α_{neg} on the results, we conduct

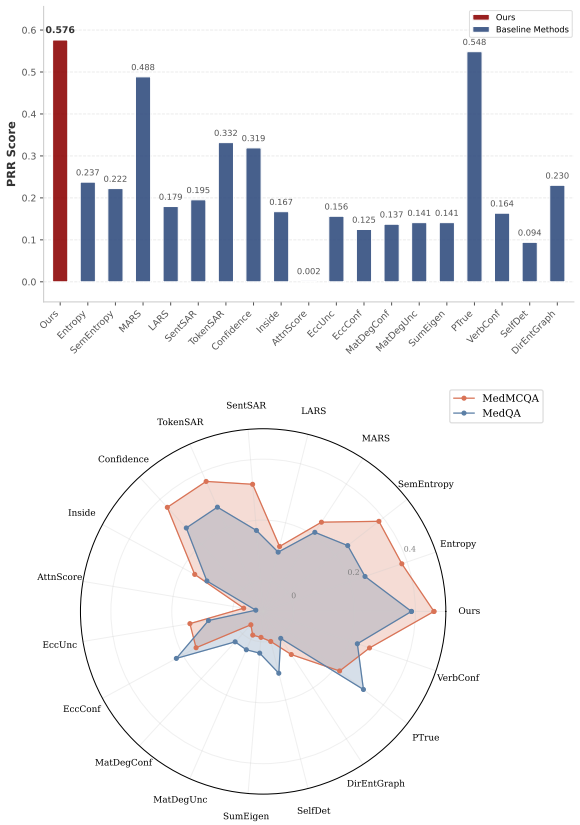

Figure 3: Top: Comparison of PRR scores between our method and 18 uncertainty estimation baselines on MedQA. Bottom: Cross-dataset generalization comparison in terms of PRR scores.

a grid search on MedExQA. Figure 4 presents the accuracy under different parameter configurations. The results show that performance improves steadily as α_{pos} increases for both models, indicating that performance gains primarily stem from better utilization of positive context. Moreover, α_{pos} has a substantially greater influence than α_{neg}, as varying α_{neg} yields only marginal differences. Notably, Qwen consistently outperforms Llama by approximately 4-5 percentage points, but exhibits lower sensitivity to parameter changes, suggesting stronger robustness. In contrast, Llama shows a clearer upward trend with increasing α_{pos}, demonstrating greater potential for performance gains through parameter tuning.

How effectively does utility-aware contrastive decoding handle different context qualities? To evaluate the effectiveness of utility-aware contrastive decoding under different context qualities, we partition MedQA samples into relevant and noisy context groups based on utility score judgments and compare the accuracy of Naive RAG versus D^2-RAG (Table 4). For noisy contexts, D^2-

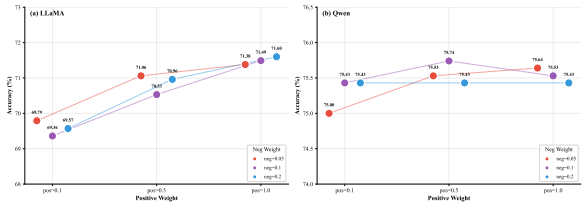

Figure 4: Accuracy on MedExQA across different $(\alpha_{\text{pos}}, \alpha_{\text{neg}})$ configurations, evaluated on Llama3.1-8B and Qwen 2.5-7B.

Model	Context	Naive RAG	D^2-RAG	ΔAcc.
Llama3.1-8B	Relevant	42.86	46.94	+4.08
	Noisy	29.03	51.61	+22.58
Qwen2.5-7B	Relevant	39.60	47.52	+7.92
	Noisy	31.43	49.52	+18.09

Table 4: Accuracy (%) of Naive RAG vs. D^2-RAG on relevant and noisy contexts (MedQA).

RAG improves accuracy by 22.58% and 18.09% on Llama3.1-8B and Qwen2.5-7B respectively, demonstrating its ability to actively suppress noisy interference and enable the model to rely on parametric knowledge for correct answers. For relevant contexts, both models show consistent improvements: Llama3.1-8B gains 4.08% and Qwen2.5-7B gains 7.92%, indicating that the amplification strategy effectively leverages high-quality retrieved content. Overall, D^2-RAG achieves substantial improvements across both context types, with particularly strong gains on noisy contexts, validating the effectiveness of the utility-aware contrastive decoding mechanism.

4.2.4 Case Study

We present three representative cases to validate the effectiveness of our framework. D^2-RAG successfully handles three typical scenarios: (a) avoiding unnecessary retrieval when the model possesses sufficient parametric knowledge (see Figure 10 in Appendix O), (b) amplifying the contribution of relevant retrieved documents through utility-aware contrastive decoding (see Figure 11 in Appendix O), and (c) suppressing interference from noisy context through utility-aware contrastive decoding (see Figure 12 in Appendix O). In all three cases, Naive RAG outputs incorrect answers while D^2-RAG outputs correct answers, demonstrating the effectiveness of our method.

5 Conclusion

In this work, we investigate the limitations of existing RAG methods in adaptive knowledge retrieval and utilization. For retrieval timing, uncertainty estimation methods often yield inconsistent results, leading to unreliable retrieval decisions. For retrieval utilization, contrastive decoding enhances the role of external knowledge but fails to distinguish content quality, causing noisy documents and relevant documents to be amplified equally. To address these issues, we propose the Dual-Decision Retrieval-Augmented Generation framework (D^2-RAG), which constructs multi-perspective uncertainty vectors to more precisely determine when retrieval is needed, and employs an adaptive decoding strategy to effectively handle retrieval results of varying quality. Experimental results demonstrate the effectiveness of D^2-RAG. Future work will focus on mechanistic interpretability to discover internal signals that more accurately reflect model uncertainty.

Limitations

Although D^2-RAG achieves notable improvements across multiple datasets, several limitations remain. First, we employ 18 uncertainty estimation methods to construct the feature vector, which may contain redundancy. While theoretically a more optimal combination of methods may exist, exhaustive search is computationally infeasible due to the large combination space. Second, the threshold in the retrieval decision model requires tuning on a validation set, which may limit its direct application to new domains lacking labeled data.

Acknowledgements

This work is supported by the National Natural Science Foundation of China (No.62376130, No.62402258), Natural Science Foundation of Shandong Province (No.ZR2024MF088), Taishan Scholars Program (No.TSQN202507242), Program of New Twenty Policies for Universities of Jinan (No.202333008), Program of Innovation Improvement for Small and Medium-sized Enterprises of Shandong (No.2023TSGC0274), the Pilot Project for Integrated Innovation of Science, Education, Industry of Qilu University of Technology (Shandong Academy of Sciences) (No.2025ZDZX01), the Open Project of the Key Laboratory of Computing Power Network and Information Security, Ministry of Education (No.2024ZD017), and Shandong Talent Introduction Program (No.WSR2025005).

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*.

Chen Amiraz, Florin Cuconasu, Simone Filice, and Zohar Karnin. 2025. The distracting effect: Understanding irrelevant passages in RAG. In *Proceedings of ACL*, pages 18228–18258.

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2024. Self-RAG: Learning to retrieve, generate, and critique through self-reflection. In *Proceedings of ICLR*, pages 41056–41085.

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 ACL*, pages 7752–7767.

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 ACL*, pages 29531–29556.

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 *Proceedings of ICLR*, pages 35078–35098.

Yung-Sung Chuang, Yujia Xie, Hongyin Luo, Yoon Kim, James R Glass, and Pengcheng He. 2024. DoLa: Decoding by contrasting layers improves factuality in large language models. In *Proceedings of ICLR*, pages 54158–54183.

Longchao Da, Tiejin Chen, Lu Cheng, and Hua Wei. 2024. LLM uncertainty quantification through directional entailment graph and claim level response augmentation. *arXiv preprint arXiv:2407.00994*.

Lu Dai, Yijie Xu, Jinhui Ye, Hao Liu, and Hui Xiong. 2025. Seper: Measure retrieval utility through the lens of semantic perplexity reduction. In *Proceedings of ICLR*, pages 47072–47103.

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 ACL*, pages 5050–5063.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yixin Dai, Jiawei Sun, Haofen Wang, and Haofen Wang. 2023. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2(1).

Qiyang Han, Tengyao Wang, Sabyasachi Chatterjee, and Richard J Samworth. 2019. Isotonic regression in general dimensions. *The Annals of Statistics*, 47(5):2440–2471.

Tianyu Han, Lisa C Adams, Jens-Michalis Papaioannou, Paul Grundmann, Tom Oberhauser, Alexander Löser, Daniel Truhn, and Keno K Bressem. 2023. Medalpaca–an open-source collection of medical conversational AI models and training data. *arXiv preprint arXiv:2304.08247*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. In *Proceedings of ICLR*, pages 9804–9830.

Soyeong Jeong, Jinheon Baek, Sukmin Cho, Sung Ju Hwang, and Jong Park. 2024. Adaptive-RAG: Learning to adapt retrieval-augmented large language models through question complexity. In *Proceedings of NAACL*, pages 7036–7050.

Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. 2020. What disease does this patient have? A large-scale open domain question answering dataset from medical exams. *arXiv preprint arXiv:2009.13081*.

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*.

Youna Kim, Hyuhng Joon Kim, Cheonbok Park, Choonghyun Park, Hyunsoo Cho, Junyeob Kim, Kang Min Yoo, Sang-goo Lee, and Taeuk Kim. 2024a. Adaptive contrastive decoding in retrieval-augmented generation for handling noisy contexts. In *Proceedings of EMNLP*, pages 2421–2431.

Yunsoo Kim, Jinge Wu, Yusuf Abdulle, and Honghan Wu. 2024b. MedExQA: Medical question answering benchmark with multiple explanations. In *Proceedings of BioNLP*, pages 167–181.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. 2023. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. In *Proceedings of ICLR*, pages 29857–29875.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Proceedings of NeurIPS*, pages 9459–9474.

Dawei Li, Bohan Jiang, Liangjie Huang, Alimohammad Beigi, Chengshuai Zhao, Zhen Tan, Amrita Bhattacharjee, Yuxuan Jiang, Canyu Chen, Tianhao Wu, Kai Shu, Lu Cheng, and Huan Liu. 2025. From generation to judgment: Opportunities and challenges of LLM-as-a-judge. In *Proceedings of EMNLP*, pages 2757–2791.

Xiang Lisa Li, Ari Holtzman, Daniel Fried, Percy Liang, Jason Eisner, Tatsunori Hashimoto, Luke Zettlemoyer, and Mike Lewis. 2023a. Contrastive decoding: Open-ended text generation as optimization. In *Proceedings of ACL*, pages 12286–12312.

Xianzhi Li, Samuel Chan, Xiaodan Zhu, Yulong Pei, Zhiqiang Ma, Xiaomo Liu, and Sameena Shah. 2023b. Are ChatGPT and GPT-4 general-purpose solvers for financial text analytics? A study on several typical tasks. In *Proceedings of EMNLP*, pages 408–422.

Zhen Lin, Shubhendu Trivedi, and Jimeng Sun. 2023. Generating with confidence: Uncertainty quantification for black-box large language models. *arXiv preprint arXiv:2305.19187*.

Wenpeng Lu, Kangjun Liu, Jianlei Wang, Xueping Peng, Tao Shen, Fa Zhu, Weiyu Zhang, Jiabing Zhu, Tao Xin, and Athanasios V. Vasilakos. 2025. Advancing chinese conversation-based patient guidance with a benchmark and knowledge-evolvable assistant. *IEEE Journal of Biomedical and Health Informatics*, pages 1–12.

Andrey Malinin and Mark Gales. 2021. Uncertainty estimation in autoregressive structured prediction. In *Proceedings of ICLR*, pages 9773–9803.

Viktor Moskvoretskii, Maria Marina, Mikhail Salnikov, Nikolay Ivanov, Sergey Pletenev, Daria Galimzianova, Nikita Krayko, Vasily Konovalov, Irina Nikishina, and Alexander Panchenko. 2025. Adaptive retrieval without self-knowledge? Bringing uncertainty back home. In *Proceedings of ACL*, pages 6355–6384.

Ankit Pal, Logesh Kumar Umapathi, and Malaikannan Sankarasubbu. 2022. MedMCQA: A large-scale multi-subject multi-choice dataset for medical domain question answering. In *Proceedings of CHIL*, pages 248–260.

Zexuan Qiu, Zijing Ou, Bin Wu, Jingjing Li, Aiwei Liu, and Irwin King. 2025. Entropy-based decoding for retrieval-augmented large language models. In *Proceedings of NAACL*, pages 4616–4627.

Weijia Shi, Xiaochuang Han, Mike Lewis, Yulia Tsvetkov, Luke Zettlemoyer, and Wen-tau Yih. 2024. Trusting your evidence: Hallucinate less with context-aware decoding. In *Proceedings of NAACL*, pages 783–791.

Ola Shorinwa, Zhiting Mei, Justin Lidard, Allen Z. Ren, and Anirudha Majumdar. 2025. A survey on uncertainty quantification of large language models:

Taxonomy, open research challenges, and future directions. *ACM Comput. Surv.*, 58(3):1–38.

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 *Proceedings of NeurIPS*, pages 34188–34216.

Zhongxiang Sun, Xiaoxue Zang, Kai Zheng, Jun Xu, Xiao Zhang, Weijie Yu, Yang Song, and Han Li. 2025. ReDeEP: Detecting hallucination in retrieval-augmented generation via mechanistic interpretability. In *Proceedings of ICLR*, pages 50250–50279.

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*.

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 EMNLP*, pages 5433–5442.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Proceedings of NeurIPS*, pages 24824–24837.

Sibo Wei, Xueping Peng, Yifei Wang, Tao Shen, Jiasheng Si, Weiyu Zhang, Fa Zhu, Athanasios V. Vasilakos, Wenpeng Lu, Xiaoming Wu, and Yinglong Wang. 2025. Biancang: A traditional chinese medicine large language model. *IEEE Journal of Biomedical and Health Informatics*, pages 1–12.

Christopher YK Williams, Brenda Y Miao, Aaron E Kornblith, and Atul J Butte. 2024. Evaluating the use of large language models to provide clinical recommendations in the emergency department. *Nature Communications*, 15(1):8236.

Chaoyi Wu, Weixiong Lin, Xiaoman Zhang, Ya Zhang, Weidi Xie, and Yanfeng Wang. 2024. PMC-LLaMA: Toward building open-source language models for medicine. *Journal of the American Medical Informatics Association*, 31(9):1833–1843.

Yunfei Xie, Juncheng Wu, Haoqin Tu, Siwei Yang, Bingchen Zhao, Yongshuo Zong, Qiao Jin, Cihang Xie, and Yuyin Zhou. 2024. A preliminary study of o1 in medicine: Are we closer to an AI doctor? *arXiv preprint arXiv:2409.15277*.

Duygu Nur Yaldiz, Yavuz Faruk Bakman, Baturalp Buyukates, Chenyang Tao, Anil Ramakrishna, Dimitrios Dimitriadis, Jieyu Zhao, and Salman Avestimehr. 2025a. Do not design, learn: A trainable scoring function for uncertainty estimation in generative LLMs. In *Proceedings of NAACL*, pages 691–713.

Duygu Nur Yaldiz, Yavuz Faruk Bakman, Sungmin Kang, Alperen Öziş, Hayrettin Eren Yildiz, Mitash Ashish Shah, Zhiqi Huang, Anoop Kumar, Alfy Samuel, Daben Liu, Sai Praneeth Karimireddy, and Salman Avestimehr. 2025b. TruthTorchLM: A comprehensive library for predicting truthfulness in LLM outputs. In *Proceedings of EMNLP*, pages 717–728.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. 2025. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*.

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 NAACL*, pages 7051–7063.

A Uncertainty Estimation Methods

In this work, we employed 18 uncertainty estimation methods. The code for these uncertainty estimations was implemented using TruthTorchLM (Yaldiz et al., 2025b). We will now provide a detailed introduction to them.

Confidence (Malinin and Gales, 2021) is an uncertainty estimation method based on sequence probability. This method posits that the higher the probability a model assigns to its generated content, the more "confident" the model is in that output, and thus the lower the uncertainty. Because the joint probabilities of sequences of different lengths naturally vary (longer sequences typically have lower joint probabilities), directly using the sequence's joint log-probability leads to an unfair penalty for long sequences. Therefore, this method uses the length-normalized log-likelihood as the confidence estimate:

$$C(x, y, \theta) = \text{LNS}(y \mid x, \theta) = \frac{1}{L} \sum_{l=1}^{L} \log P(y_l \mid y_{<l}, x, \theta), \quad (9)$$

where x represents the input query, $y = \{y_1, y_2, \ldots, y_L\}$ represents the output sequence generated by the model, L is the length of the generated sequence, and θ represents the model parameters.

A higher score indicates higher model confidence in the current result and lower uncertainty.

Entropy (Malinin and Gales, 2021) estimates the model output uncertainty by sampling multiple generated results for a given query. When the model has high uncertainty regarding the answer to a query, multiple samplings will yield diverse outputs, each with a relatively low probability. Conversely, when the model is very confident, multiple samplings will tend to produce similar, high-probability outputs.Theoretically, calculating the true entropy of the generation distribution requires iterating over all possible output sequences. However, since the output space is exponential (there are K^L possibilities for a sequence of length L and vocabulary size K), this is impractical in reality.Therefore, this method uses Monte Carlo sampling to approximate the entropy. By sampling B sequences from the model, the average negative log-likelihood of these samples is used to approximate the true entropy. Specifically, for each sampled sequence, the length-normalized log-likelihood is first calculated, and then the negative of the average over all sampled results is taken as the entropy estimate:

$$H(x, \theta) \approx -\frac{1}{B} \sum_{b=1}^{B} \frac{1}{L^{(b)}} \sum_{l=1}^{L^{(b)}} \log P(y_l^{(b)} \mid y_{<l}^{(b)}, x, \theta), \quad (10)$$

where B is the number of sampled sequences, $y^{(b)}$ is the b-th sampled sequence, $L^{(b)}$ is the length of the b-th sequence, $y_l^{(b)}$ is the l-th token in the b-th sequence, and $y_{<l}^{(b)}$ represents the context composed of the first $l-1$ tokens in the b-th sequence.

MARS (Bakman et al., 2024) is an uncertainty estimation method that considers the semantic contribution of tokens. Traditional length-normalized scoring methods assign the same weight, $1/L$, to every token in the generated sequence, ignoring the differing contributions of various tokens to the correctness of the answer.The calculation for MARS involves two steps.The first step is to calculate the semantic importance weight for each token. This method first divides the generated sequence into several phrases using a pre-trained model. It then determines the importance score of each phrase by evaluating the semantic impact on the answer after removing that phrase. Finally, the phrase importance score is uniformly distributed to the individual tokens within the phrase and normalized using a softmax function with a temperature parameter τ, resulting in the importance weight w_l for each token, satisfying $\sum_{l=1}^{L} w_l = 1$.The second step combines the semantic-aware scoring with the traditional length-normalized scoring to compute the final score:

$$\text{MARS}(x, s, \theta) = -\frac{1}{2} \sum_{l=1}^{L} w_l \cdot \log P(s_l \mid s_{<l}, x, \theta) - \frac{1}{2L} \sum_{l=1}^{L} \log P(s_l \mid s_{<l}, x, \theta), \quad (11)$$

where s represents the output sequence generated by the model, L is the length of the generated sequence, and w_l represents the semantic importance weight of the l-th token.

LARS (Yaldiz et al., 2025a) is an uncertainty estimation scoring function based on supervised learning. Unlike traditional heuristic scoring functions (like LNS, MARS), LARS directly learns how

to aggregate token probabilities into a confidence score from annotated data using a neural network. This allows it to capture complex semantic dependencies between tokens and automatically correct probability biases. The calculation for LARS involves two steps.The first step is probability discretization and encoding. Since token probabilities are single real values, feeding them directly into a Transformer model is not ideal. LARS divides the probability space $[0, 1]$ into k partitions based on the quantiles of the training data. Each probability value p_l is mapped to a special probability token based on its partition, where r is the partition index. The embedding vector for each probability token is initialized using a few-hot encoding, ensuring that different probability intervals are mutually orthogonal in the embedding space.The second step is to calculate the confidence score using a Transformer model. LARS concatenates the input query x and the response sequence s in an alternating format of 'response token - probability token' and feeds it into a RoBERTa-based encoder. The output then passes through a linear layer and a sigmoid activation to yield the confidence score:

$$\text{LARS}(x, s, p) = \sigma(f_\phi(x, s, p)), \quad (12)$$

where x represents the input query, s represents the model output sequence, p represents the generation probabilities of each token in the model, f_ϕ represents the LARS model, and $\sigma(\cdot)$ represents the sigmoid function.

TokenSAR (Duan et al., 2024) is an uncertainty estimation method based on token-level semantic relevance. This method argues that in autoregressively generated text, the contribution of different tokens to expressing meaning is unequal. Some key vocabulary (like nouns and verbs) carry more semantic information than functional words (like articles and prepositions). Traditional methods assign the same weight to all tokens when calculating uncertainty, which is unreasonable. TokenSAR re-weights the tokens by calculating the semantic relevance of each token. The specific calculation is divided into two steps. First, the relevance score of each token is calculated by measuring the semantic change in the sentence before and after removing that token:

$$RT(z_i, s, x) = 1 - |g(x \cup s, x \cup s \setminus \{z_i\})|, \quad (13)$$

where z_i represents the i-th token, and $g(\cdot, \cdot)$ represents the semantic similarity calculation function. Then, the relevance scores are normalized and used to re-weight the entropy of each token:

$$\text{TokenSAR}(s, x) = \sum_{i=1}^{N} (-\log p(z_i \mid s_{<i}, x)) \cdot \frac{RT(z_i, s, x)}{\sum_{n=1}^{N} RT(z_n, s, x)}, \quad (14)$$

where N represents the sequence length.

SentSAR (Duan et al., 2024) is an uncertainty estimation method based on sentence-level semantic relevance. This method extends the token-level semantic-aware idea to the sentence level, positing that among multiple generated candidates, sentences that are more semantically consistent and representative should receive higher weight in the uncertainty calculation. The core idea is to measure the relevance of each sentence through inter-sentence semantic similarity and adjust the sentence's generation probability accordingly. First, the sentence-level relevance score is calculated. This score is determined by the probability-weighted semantic similarity of the sentence with other candidate sentences:

$$RS(s_i, S, x) = \sum_{j=1, j\neq i}^{K} g(s_i, s_j) p(s_j \mid x), \quad (15)$$

where s_i is the i-th generated sentence, S is the set of all candidate sentences, K is the number of candidate sentences, and $g(s_i, s_j)$ represents the semantic similarity between the two sentences.Then, the uncertainty of the sentence is adjusted by using the relevance score as an additional probability term:

$$\text{SentSAR}(S, x) = \frac{1}{K} \sum_{k=1}^{K} \Big(-\log p(s_k \mid x) + \frac{RS(s_k, S, x)}{t} \Big), \quad (16)$$

where t is the temperature parameter, used to control the strength of the relevance adjustment.

Semantic Entropy (Kuhn et al., 2023) is an uncertainty estimation method based on semantic equivalence. In natural language, different sentences can express the same semantic content, e.g., "The capital of France is Paris" and "Paris is the capital of France" are semantically equivalent. However, traditional predictive entropy methods

would treat them as distinct outputs, thus overestimating uncertainty. This method uses the idea of bidirectional entailment to identify semantically equivalent sentences. If two sentences can mutually entail each other, they are considered to express the same meaning. These sentences are then grouped together via clustering, and entropy is calculated in the semantic space rather than the lexical space. The method involves three core steps. First, sample multiple output results from the language model. Second, use a bidirectional entailment clustering algorithm to identify semantically equivalent sentences, specifically by checking if a pair of sentences can mutually entail each other. Finally, for each semantic equivalence class c, its probability is the sum of the probabilities of all sentences within that class:

$$p(c \mid x) = \sum_{s \in c} p(s \mid x), \tag{17}$$

where x is the input query, s is the generated sentence, and c is the semantic equivalence class. The Semantic Entropy is calculated as:

$$\mathrm{SE}(x) = -\sum_{c} p(c \mid x) \log p(c \mid x), \tag{18}$$

In practical implementation, since it is impossible to iterate over all possible semantic categories, Monte Carlo sampling is used to approximate the estimate:

$$\mathrm{SE}(x) \approx -\frac{1}{|C|} \sum_{i=1}^{|C|} \log p(C_i \mid x), \tag{19}$$

where C is the set of semantic equivalence classes, and $|C|$ is the number of classes.

INSIDE (Chen et al., 2024) is an uncertainty estimation method based on the internal states of the LLM. Unlike methods based on logits or the language surface, INSIDE directly utilizes the dense semantic information preserved in the LLM's internal hidden layers to measure the semantic consistency among multiple generated results. The core hypothesis of this method is: when the model is highly confident about a question, the answers generated through multiple samplings will be highly similar in the embedding space, and most eigenvalues of the sentence embedding covariance matrix will be close to zero. Conversely, when the model is uncertain, the generated answers will be semantically diverse, leading to larger eigenvalues. The calculation for INSIDE is divided into three steps. The first step is obtaining sentence embeddings. For an input query x, K answers are generated via a sampling strategy, $\mathcal{Y} = \{y^1, y^2, \ldots, y^K\}$. For each answer, the hidden state of the last token in a middle layer is taken as the sentence's embedding representation $z^k \in \mathbb{R}^d$, where d is the hidden layer dimension. The second step is calculating the covariance matrix of the sentence embeddings. The K sentence embeddings are formed into a matrix $Z = [z_1, z_2, \ldots, z_K] \in \mathbb{R}^{d \times K}$, and the covariance matrix is calculated as:

$$\Sigma = Z^\top \cdot J_d \cdot Z, \tag{20}$$

where $J_d = I_d - \frac{1}{d} 1_d 1_d^\top$ is the centering matrix, I_d is the d-dimensional identity matrix, and 1_d is the column vector of all ones. The third step is calculating the EigenScore using the eigenvalues of the covariance matrix. To ensure the matrix is full rank, the log-determinant is calculated after adding a regularization term:

$$\begin{aligned} E(\mathcal{Y} \mid x, \theta) &= \frac{1}{K} \log\det(\Sigma + \alpha \cdot \mathbf{I}_K) \\ &= \frac{1}{K} \sum_{i=1}^{K} \log \lambda_i, \end{aligned} \tag{21}$$

where $\lambda = \{\lambda_1, \lambda_2, \ldots, \lambda_K\}$ are the eigenvalues of the regularized covariance matrix $\Sigma + \alpha \cdot \mathbf{I}_K$, α is the regularization coefficient, and K is the number of sampled answers.

Attention Score (Sriramanan et al., 2024) is a method for hallucination detection based on the eigenvalue analysis of the attention mechanism. This method is built upon a key observation: when a Large Language Model (LLM) generates hallucinatory content, its internal attention patterns undergo a noticeable change, which can be captured by analyzing the kernel similarity matrix of the self-attention mechanism.

In the Transformer architecture, the self-attention matrix for an auto-regressive language model has a lower-triangular structure, as each token can only attend to preceding tokens in the sequence. This method detects hallucinations by analyzing the change in the eigenvalues of these attention matrices.

The specific calculation process involves two steps: Extracting the Self-Attention Matrix and Calculating the Log-Determinant: The self-attention matrix from a target layer is extracted. For a layer with a attention heads, the attention matrix can be represented as a tensor of shape $(a \times m \times m)$,

where m is the length of the input sequence. For each attention head i, its kernel similarity matrix Ker_i is an $m \times m$ lower-triangular matrix. Since the matrix is normalized via Softmax, all eigenvalues are non-negative, and the eigenvalues of a lower-triangular matrix are simply its elements on the main diagonal. Therefore, the log-determinant can be calculated directly:

$$\log\det(Ker_i) = \sum_{j=1}^{m} \log Ker_{jj}^{i}, \tag{22}$$

where Ker_{jj}^{i} denotes the (j, j)-th diagonal element of the kernel similarity matrix for attention head i.

Aggregating the Score: The average log-determinant for each attention head is calculated, and then aggregated across all attention heads to obtain the final Attention Score:

$$\text{AttentionScore} = \frac{1}{a}\sum_{i=1}^{a}\frac{1}{m}\sum_{j=1}^{m}\log Ker_{jj}^{i}, \tag{23}$$

where a is the number of attention heads.

Self Detection (Zhao et al., 2024) is an uncertainty estimation method based on question-rewriting consistency. The core hypothesis of this method is: if the model truly understands a question and possesses the relevant knowledge, it should provide a consistent answer regardless of how the question is phrased. Conversely, if the model gives significantly different answers to semantically equivalent, yet differently phrased, questions, it indicates knowledge uncertainty on the model's part regarding that topic. The calculation for Self Detection involves two steps:

Step1: Generating Semantically Equivalent Question Variants: For the original question q, the LLM itself is used to generate n semantically equivalent but differently phrased questions, forming the set $Q(q) = \{q_1, q_2, \ldots, q_n\}$. Then, greedy decoding is used for each question variant to generate the corresponding answers, resulting in the answer set $R(q) = \{r_1, r_2, \ldots, r_n\}$.

Step2: Calculating the Consistency Score Between Answers: First, a determination is made as to whether any two answers, r_i and r_j, are consistent, denoted by $I(r_i, r_j) \in \{0, 1\}$. For answers with a fixed format (like multiple-choice), this is judged by exact match; for free-form answers, the LLM itself is used to judge whether the two answers are contradictory. Based on this consistency judgment, all answers are clustered into k clusters $\Omega = \{\omega_1, \omega_2, \ldots, \omega_k\}$. The consistency score is calculated as the proportion of consistency between the original question's answer r and the other answers:

$$\text{Consistency}(R(q)) = \frac{1}{n-1}\sum_{r_i, r_i \neq r} I(r_i, r). \tag{24}$$

Additionally, uncertainty can be characterized by calculating the entropy of the answer distribution based on the clustering results:

$$\text{Entropy}(R(q)) = -\sum_{l}\frac{N(\omega_l)}{n}\log\frac{N(\omega_l)}{n}, \tag{25}$$

where $N(\omega_l)$ is the number of answers in the l-th cluster.

Directional Entailment Graph (Da et al., 2024) is an uncertainty quantification method based on a Directional Entailment Graph and the Random Walk Laplacian. This approach addresses the critical issue that traditional semantic similarity matrices ignore directional information. In natural language, the entailment relationship has a clear directionality: Proposition A entailing Proposition B does not imply that B also entails A. For instance, the probability that 'She is interning at Google' entails 'She has a job' is 96.2%, while the reverse entailment probability is only 0.41%. Traditional methods lose this crucial directional information through a symmetric similarity matrix.The core idea of this method is to construct a Directional Entailment Graph to capture the directional logical relationships between responses. The calculation process is divided into three steps:Constructing the Asymmetric Entailment Matrix: A Natural Language Inference (NLI) model is used to compute the entailment probability between pairs of generated responses, constructing the asymmetric entailment matrix A, where $A_{ij} = p(r_i \vdash r_j)$ represents the probability that response r_i entails r_j.Edge Weight Augmentation: This matrix is then combined with a textual similarity matrix T (such as Jaccard similarity) to augment the graph's edge weights: $w_{ij} = A_{ij} + T_{ij}$.Calculating Uncertainty via the Random Walk Laplacian: Because the constructed directed graph is asymmetric, the traditional symmetric Laplacian operator is no longer applicable. This method innovatively adopts the Random Walk Laplacian operator to handle this asymmetric characteristic:

$$L_{rw} = I - D_{out}^{-1}A, \tag{26}$$

where D_{out} is the out-degree matrix, and $D_{out}^{-1} = (D_{out} + \epsilon I)^{-1}$ is used to avoid division by zero errors. The final uncertainty measure is calculated using the eigenvalues of the Random Walk Laplacian matrix:

$$U_{Eigv}^{d} = \sum_{k=1}^{n} \max(0, 1 - \lambda_k), \tag{27}$$

where n is the number of responses and λ_k are the eigenvalues of the Random Walk Laplacian matrix.

Verbalized Confidence (Tian et al., 2023) is an uncertainty estimation method that involves directly asking the model for its confidence. Addressing the issue of poor internal probability calibration in RLHF models, this method has the model express its confidence directly within the output text.

The method is divided into two categories: Numerical Expression and Verbal Expression. 1. Numerical Expression includes a single-stage method (outputting the answer and a numerical probability simultaneously) and a two-stage method (generating the answer first, then assessing the probability). 2. Verbal Expression involves having the model use phrases like "very certain," "possibly," etc., which are then mapped to a numerical score. A core strategy is "considering alternative answers": the model is prompted to generate multiple candidate answers and assess the confidence for each one, finally selecting the answer with the highest confidence. Experiments have shown that this approach is more accurate than directly using the model's conditional probabilities.

PTrue (Kadavath et al., 2022) is an uncertainty estimation method based on self-assessment. The core idea of this method is to have the Language Model evaluate the probability of its own generated answer being correct, achieving self-verification by transforming the open-ended generation task into a true/false judgment task. The calculation for this method is divided into three steps:

Step1: Generating the Candidate Answer Set: Given an input question x, N candidate answers $\{a_1, a_2, \ldots, a_N\}$ are first sampled from the model. These answers are sampled independently under a temperature parameter $T = 1$.

Step2: Constructing the Self-Assessment Prompt: The original question, all sampled candidate answers, and the specific answer a_i to be evaluated are combined into a structured prompt, asking the model whether this answer is correct:

Question: question

Here are some brainstormed ideas: candidate answers

Possible Answer: answer to evaluate

Is the possible answer:

(A) True

(B) False

The possible answer is:

Step3: Calculating the P(True) Score: The model performs inference on the above prompt, and the probability the model assigns to the "True" option is extracted as the confidence estimate:

$$P(\text{True}) = P((\text{A}) \mid \text{prompt}, x, a_i, \theta), \tag{28}$$

where x represents the input question, a_i represents the generated answer to be evaluated, θ represents the model parameters, and $prompt$ contains the full context of the question, candidate answer set, and the answer being evaluated.

The following five methods all come from the work-Generating with confidence: Uncertainty quantification for black-box large language models (Lin et al., 2023). The core idea of these methods is to quantify uncertainty or confidence by sampling multiple generation results for the same query, calculating the semantic similarity between these results, and then constructing a similarity matrix from which metrics are extracted. The advantage of this class of methods is that they only require access to the model's text output, without needing token-level probability distributions, making them suitable for black-box Large Language Models (LLMs).These methods support two ways to measure similarity:Jaccard Similarity: Treating two responses as sets of words, calculate the ratio of their intersection to their union.Semantic Similarity: Using a Natural Language Inference (NLI) model (e.g., DeBERTa-large-mnli) to judge the entailment relationship between two responses, and using the entailment probability as the semantic similarity.Given an input x, m responses $\{s_1, s_2, \ldots, s_m\}$ are sampled from the model. The pairwise similarity score $a(s_{j_1}, s_{j_2})$ is calculated, and a symmetric weighted adjacency matrix W is constructed, where $w_{j_1,j_2} = \frac{a_{j_1,j_2} + a_{j_2,j_1}}{2}$.

SumEigenUncertainty. This method estimates uncertainty based on the eigenvalue distribution of the Graph Laplacian matrix. The core idea is that the eigenvalue distribution reflects the clustering structure between the responses. When the model has high uncertainty for a query, the generated responses form multiple semantically distinct clusters, leading to more small eigenvalues.

First, the symmetric normalized Graph Laplacian matrix is calculated:

$$L := I - D^{-\frac{1}{2}} W D^{-\frac{1}{2}}, \tag{29}$$

where D is the degree matrix, $D_{j,j} = \sum_{j' \in [m]} w_{j,j'}$. Then, the eigenvalues $1 < \lambda_2 < \cdots < \lambda_m$ of L are computed. The uncertainty estimate is:

$$U_{\text{EigV}}(x) = \sum_{k=1}^{m} \max(0, 1 - \lambda_k). \tag{30}$$

Matrix Degree Uncertainty. This method uses the degree matrix to estimate uncertainty. The diagonal elements of the degree matrix represent the total strength of connection of each node with other nodes, reflecting the overall similarity of that response to the others. When the model is uncertain, the generated responses differ significantly from each other, resulting in a lower overall degree.

The uncertainty estimate is defined as:

$$U_{\text{Deg}}(x) = \frac{\text{trace}(mI - D)}{m^2}. \tag{31}$$

Matrix Degree Confidence. Unlike Matrix Degree Uncertainty, this method provides a confidence estimate for each specific generated response, rather than just for the input. The idea is that if a response is highly similar to the other sampled responses (i.e., the degree of that node is high), it suggests that the response is located in the model's "confidence region" and should have a higher confidence score.For response s_j, the confidence estimate is:

$$C_{\text{Deg}}(x, s_j) = \frac{D_{j,j}}{m}. \tag{32}$$

Eccentricity Uncertainty. This method constructs a low-dimensional embedding for each response using the eigenvectors of the Graph Laplacian matrix, and then estimates uncertainty based on the dispersion of these embeddings. Let $u_1, \ldots, u_k \in \mathbb{R}^m$ be the k smallest eigenvectors of L. The embedding for response s_j is $v_j = [u_{1,j}, \ldots, u_{k,j}]$. The vector deviated from the center is calculated as $v'_j = v_j - \frac{1}{m} \sum_{j'=1}^{m} v_{j'}$. The uncertainty estimate is the sum of the distances of all response embeddings from the center:

$$U_{\text{Ecc}}(x) = \left\| [{v'_1}^\top, \ldots, {v'_m}^\top] \right\|_2. \tag{33}$$

Eccentricity Confidence. This method is based on the embedding representation from Eccentricity Uncertainty and provides a confidence estimate for each specific response. The idea is that if the embedding of a response is close to the center of all responses, that response represents the model's "mainstream" output and should have higher confidence.For response s_j, the confidence estimate is:

$$C_{\text{Ecc}}(x, s_j) = -\|v'_j\|_2. \tag{34}$$

B Isotonic Regression Calibration

We employed Isotonic Regression to calibrate the uncertainty estimation methods to improve the consistency between their output scores and the actual quality of the generated output.The core idea of calibration is to learn a monotonic mapping function between the raw output of the uncertainty estimation method and the actual correctness, converting the raw score into a more reliable confidence estimate.The calibration process first involves collecting training data on a designated calibration dataset. For each sample in the calibration set, the system generates an answer using the target Large Language Model (LLM) and then performs a dual evaluation. Raw Uncertainty Score: The uncertainty estimation method to be calibrated is applied to calculate the raw uncertainty score. Ground Truth Correctness: GPT-4o-mini is used as the judgement model for correctness evaluation. The judging process adopts a prompt-based three-category framework, requiring the judgement model to classify the generated answer into three categories: “CORRECT”, “INCORRECT”, or “NOT_ATTEMPTED”, corresponding to correct, wrong, and untried answers, respectively.These classification results are converted into numerical scores: correct answers are assigned 1, wrong answers are assigned 0, and untried answers are assigned -1 (and filtered out in subsequent processing). This yields a large amount of paired data (t_i, c_i), where t_i is the raw output of the uncertainty estimation method and c_i is the corresponding binary correctness assessment result.Next, an isotonic regression model is trained separately for each uncertainty estimation method. Isotonic Regression finds a monotonically increasing function $f\colon [0,1] \to [0,1]$ that minimizes the prediction error, specifically $f^* = \text{argmin}_f \sum_{i=1}^{n} (f(t_i) - c_i)^2$, subject to the constraint that $f(t_i) \leq f(t_j)$ must hold when $t_i \leq t_j$.The implementation uses scikit-learn's IsotonicRegression, setting the output range to $[0, 1]$ and clipping any values outside these bounds. During the data preprocessing stage, NaN

values are replaced with 0, and invalid samples categorized as 'NOT_ATTEMPTED' are filtered out.

C Threshold

Threshold Settings Used in Experiments:

MedQA Dataset (LLaMA Model): $\tau_{\text{neg}} = -0.08$. MedQA Dataset (Qwen Model): $\tau_{\text{pos}} = 0.3$, $\tau_{\text{neg}} = -0.1$. MedMCQA Dataset (Qwen Model): $\tau_{\text{pos}} = 0.05$, $\tau_{\text{neg}} = -0.1$. MedExQA Dataset (LLaMA Model): $\tau_{\text{pos}} = 0.1622$, $\tau_{\text{neg}} = -0.0458$. MedExQA Dataset (Qwen Model): $\tau_{\text{pos}} = 0.1055$, $\tau_{\text{neg}} = -0.1666$. MMLU Dataset (LLaMA Model): $\tau_{\text{pos}} = 0.1385$, $\tau_{\text{neg}} = -0.0309$. MMLU Dataset (Qwen Model): $\tau_{\text{pos}} = 0.0417$, $\tau_{\text{neg}} = -0.1093$.

D Dynamic Weight Calculation Based on Shannon Entropy

Inspired by adaptive contrastive decoding (Kim et al., 2024a), we dynamically calculate w_t based on Shannon Entropy.At generation step t, we separately calculate the Shannon Entropy without context, $H(Y_t)$, and with context, $H(Y_t^c)$. At generation step t, we separately calculate the Shannon Entropy without context, $H(Y_t)$, and with context, $H(Y_t^c)$. Entropy without context:

$$H(Y_t) = -\sum_{v \in V} P_\theta(v|x, y_{<t}) \log P_\theta(v|x, y_{<t}). \tag{35}$$

Entropy with context:

$$H(Y_t^c) = -\sum_{v \in V} P_\theta(v|x, c, y_{<t}) \times \log P_\theta(v|x, c, y_{<t}), \tag{36}$$

where V is the vocabulary and c is the retrieved context. The dynamic entropy factor w_t is calculated as:

$$w_t = \frac{H(Y_t)}{H(Y_t) + H(Y_t^c)}. \tag{37}$$

E Medical Q&A Dataset Examples

MedMCQA Example

Question: Which of the following is true regarding major basic protein?

A) Formed by Eosinophils
B) Cytotoxic to parasites
C) Not very effective against bacteria
D) All the above

Ground Truth: D) All the above

MedQA Example

Question: A 15-year-old girl is brought to the clinic by her mother because she is worried the patient has not yet had her period. The patient's older sister had her first period at age 14. The mother had her first period at age 13. The patient reports she is doing well in school and is on the varsity basketball team. Her medical history is significant for asthma and atopic dermatitis. Her medications include albuterol and topical triamcinolone. The patient's temperature is 98°F (36.7°C), blood pressure is 111/72 mmHg, pulse is 65/min, and respirations are 14/min with an oxygen saturation of 99% on room air. Her body mass index (BMI) is 19 kg/m^2. Physical exam shows absent breast development and external genitalia at Tanner stage 1. Serum follicle stimulating hormone (FSH) level is measured to be 38 mIU/mL. Which of the following is the next best diagnostic step?

A) CYP17 gene work-up
B) Estrogen levels
C) Gonadotrophin-releasing hormone stimulation test
D) Karotype
E) Luteinizing hormone levels

Ground Truth: D) Karotype

MMLU Example
Question: A lesion causing compression of the facial nerve at the stylomastoid foramen will cause ipsilateral
A) paralysis of the facial muscles.
B) paralysis of the facial muscles and loss of taste.
C) paralysis of the facial muscles, loss of taste and lacrimation.
D) paralysis of the facial muscles, loss of taste, lacrimation and decreased salivation.
Ground Truth: A) paralysis of the facial muscles.

MedExQA Example
Question: What should be the maximum surface temperature of a device attachment that is not intended to heat the patient?
A) 43°C
B) 36°C
C) 38°C
D) 41°C
Ground Truth: D

F Evaluation Prompt

System Prompt:
You are a precise evaluator. Your task is to determine if a 'Generated Answer' is semantically identical to any of the 'Ground Truth' answers for a multiple-choice question. They are identical if they refer to the same option, even with minor formatting differences (e.g., 'A)' vs 'A.' or containing extra text like 'The answer is...'). Respond with only the single word 'yes' or 'no', and nothing else.
User Prompt Template:
Question: {question}
Ground Truths: {ground_truths}
Generated Answer: {generated_answer}

G Experimental Data Configuration Details

We used a total of 9,183 samples from the MedMCQA dataset, 7,545 samples from the MedQA dataset, 965 samples from the MedExQA dataset, and 1,821 samples from the MMLU-Medical dataset. In the main experiment, we used 5,000 samples for testing from both the MedMCQA and MedQA datasets, 940 samples from the MedExQA dataset, and 1,640 samples from the MMLU-Medical dataset. In the uncertainty quantification evaluation experiment, for the MedMCQA dataset, we took an additional 4,183 samples, with 500 used to calibrate the uncertainty scores and the remaining data used as the test set. For the MedQA dataset, we took an additional 2,545 samples, with 500 samples used to calibrate the uncertainty scores and the remaining data used as the test set. In the retrieval decision evaluation experiment, for the MedMCQA dataset, we used 3,683 samples for training and validation, where 70% were used as the training set to train the retriever and 30% were used as the validation set to find the threshold. 5,000 samples were used as the test data.For the MedQA dataset, we used 2,045 samples for training and validation, where 70% were used as the training set to train the retriever and 30% were used as the validation set to find the threshold. 5,000 samples were used as the test data.

H Retrieval Decision Comparison Experimental Setup

Our method We used the best-performing model as the retrieval decision classifier and optimized the threshold on the validation set to maximize the F1 score for the 'retrieval required' category. The decision logic is to trigger retrieval when the model-outputted probability for the 'retrieval not required' category falls below the optimal threshold.

FLARE We used the Llama3.1-8B-Instruct model with greedy decoding generation. Retrieval is triggered if the probability of any token generated during the process falls below a fixed threshold of 0.8.

Self-RAG The Self-RAG method uses the specially trained Selfrag-13b model. This model is capable of outputting reflection tokens to indicate the need for retrieval before generating the answer: when the model outputs [Retrieval], it indicates that external knowledge retrieval is required; when it outputs [No Retrieval], it indicates that retrieval is not needed. The model makes its retrieval decision based on the occurrence of these special tokens. If no explicit instruction is included in the generated text, the default is to not perform retrieval.

I Cross-Dataset Generalization Experiment Results

We present the cross-dataset generalization experiment results for the Llama3.1-8B model on MedMCQA and MedQA datasets in Figure 5.

J Uncertainty Method Comparison Results

We present the comparison results of D^2-RAG's composite uncertainty feature method against 18 individual uncertainty estimation methods across different datasets and models. Specifically, Figure 6 shows results for Llama3.1-8B on MedMCQA, Figure 7 shows results for Qwen2.5-7B on MedMCQA, and Figure 8 shows results for Qwen2.5-7B on MedQA.

K Parameter Sensitivity Curve Analysis

To gain a deeper understanding of the independent influence of each parameter, we plotted the parameter sensitivity curves (as shown in Figure 9). By using the control variable method, we fixed one parameter at a time to observe the trend of the other parameter's influence.Influence of the Positive Strength Coefficient α_{pos} (α): The top part of the figure shows the effect of the α parameter when the β value is fixed. Under both settings of $\beta = 0$ and $\beta = 0.1$, the LLaMA model exhibits strong sensitivity to the α parameter. As α increases from 0.1 to 1.0, the relative performance gain shows a clear upward trend, with a maximum improvement of up to 1.7%. In contrast, the Qwen model is relatively insensitive to the α parameter, with a more moderate performance improvement, typically within 0.4%. This difference suggests that the LLaMA model is better able to utilize the positive context enhancement mechanism.Influence of the Negative Strength Coefficient α_{neg} (β):The bottom part of the figure reveals the pattern of the β parameter's effect when the α value is fixed. When $\alpha = 0.1$, the LLaMA model achieves its performance peak at $\beta = 0.15$, while the Qwen model performs best around $\beta = 0.1$. When $\alpha = 1.0$, both models exhibit a complex sensitivity to the β parameter: LLaMA favors smaller β values, while Qwen reaches its optimal performance at $\beta = 0.15$. This non-monotonic change pattern indicates that the optimal effect of the negative inhibition mechanism depends on the setting of the positive enhancement strength. By comparing the curves showing the influence of the β parameter under different α settings, we observe a clear parameter interaction. At low α values (0.1), a moderate β value contributes to performance improvement; at high α values (1.0), the optimal configuration for the β parameter changes significantly. This suggests a complex synergistic relationship between the positive enhancement and negative inhibition mechanisms, where the best performance is not determined by a single parameter but requires joint tuning based on the combination of both strengths.

L Algorithm for D^2-RAG

As shown in Algorithm 1, D^2-RAG first decides whether to retrieve based on the calibrated uncertainty vector, and if retrieval is triggered, applies adaptive contrastive decoding based on retrieval utility.

M Detailed descriptions of baselines

Zero Shot (Wei et al., 2021). Directly inputs the question into the model without providing any examples or external knowledge, testing the model's original capability.

CoT (Wei et al., 2022). Guides the model to perform step-by-step reasoning before generating the answer by adding the prompt: "Let's think step by step."

PMC-Llama (Wu et al., 2024). Employs a two-stage training strategy: first, it undergoes continued pre-training on 4.8 million biomedical papers and 30,000 medical textbooks to build a domain-specific knowledge base, and then it is fine-tuned on medical instruction data to enhance reasoning and alignment capabilities.

MedAlpaca (Han et al., 2023). Fine-tunes an open-source LLM on an instruction dataset containing 160,000 medical Q&A and dialogue samples, enabling it to effectively follow instructions in the medical domain.

Naive RAG: Performs retrieval for all queries, concatenates the retrieved documents to the input context, and then generates the answer.

Self-RAG (Asai et al., 2024). Uses end-to-end training to enable the model to generate special reflection tokens, including a retrieval token to judge whether retrieval is needed and a critique token to evaluate the generation quality. During inference, the model triggers retrieval on demand based on the retrieval token, self-assesses the generated content using the critique token, and selects the optimal output.

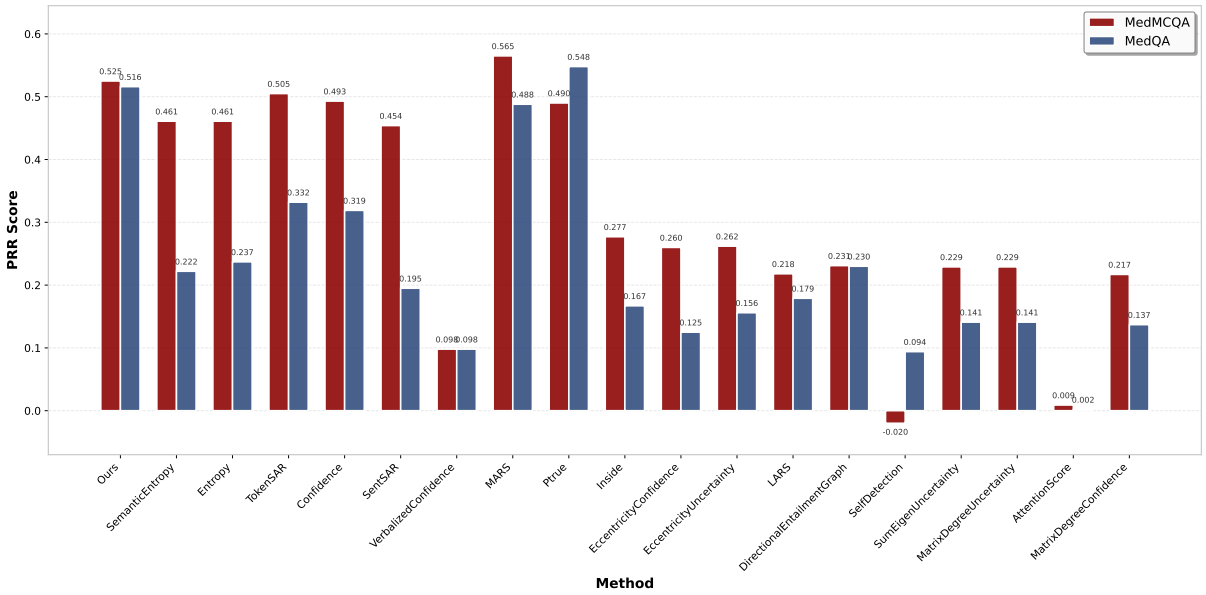

Figure 5: We compared the cross-dataset generalization capability of our retrieval decision model with several uncertainty quantification baseline methods. D^2-RAG was evaluated under two cross-dataset settings: MedQA training - MedMCQA testing and MedMCQA training - MedQA testing. The baseline methods are represented by their best performance on the corresponding test set, and the evaluation metric used is the PRR score.

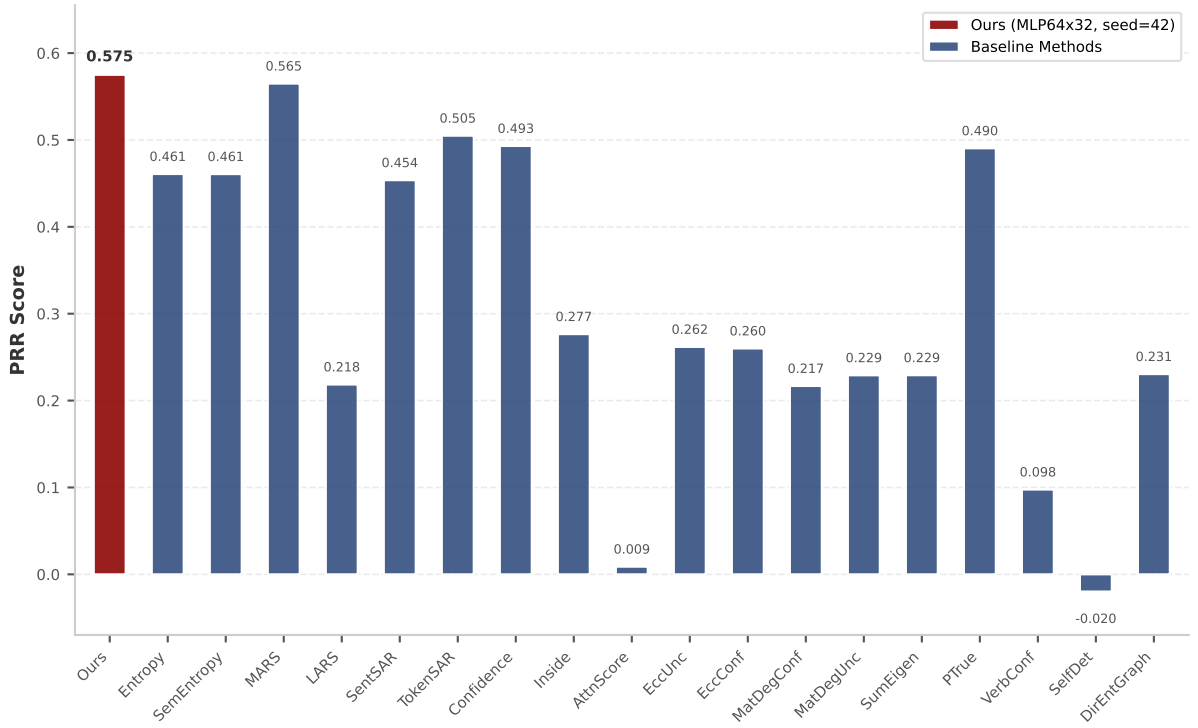

Figure 6: We compared D^2-RAG's composite uncertainty feature method against 18 individual uncertainty estimation methods across four major categories, using the Llama3.1-8B model. The evaluation was conducted on the MedMCQA dataset, with the evaluation metric being the PRR score.

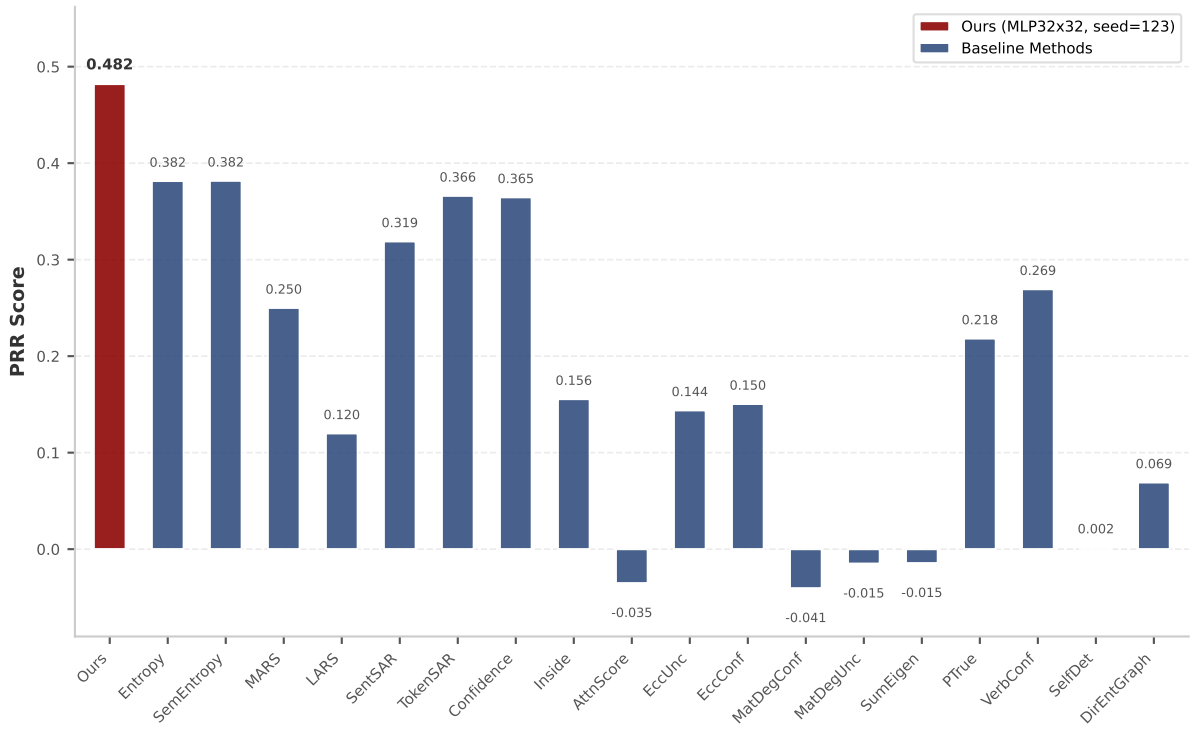

Figure 7: We compared D^2-RAG's composite uncertainty feature method against 18 individual uncertainty estimation methods across four major categories, using the Qwen2.5-7B model. The evaluation was conducted on the MedMCQA dataset, with the evaluation metric being the PRR score.

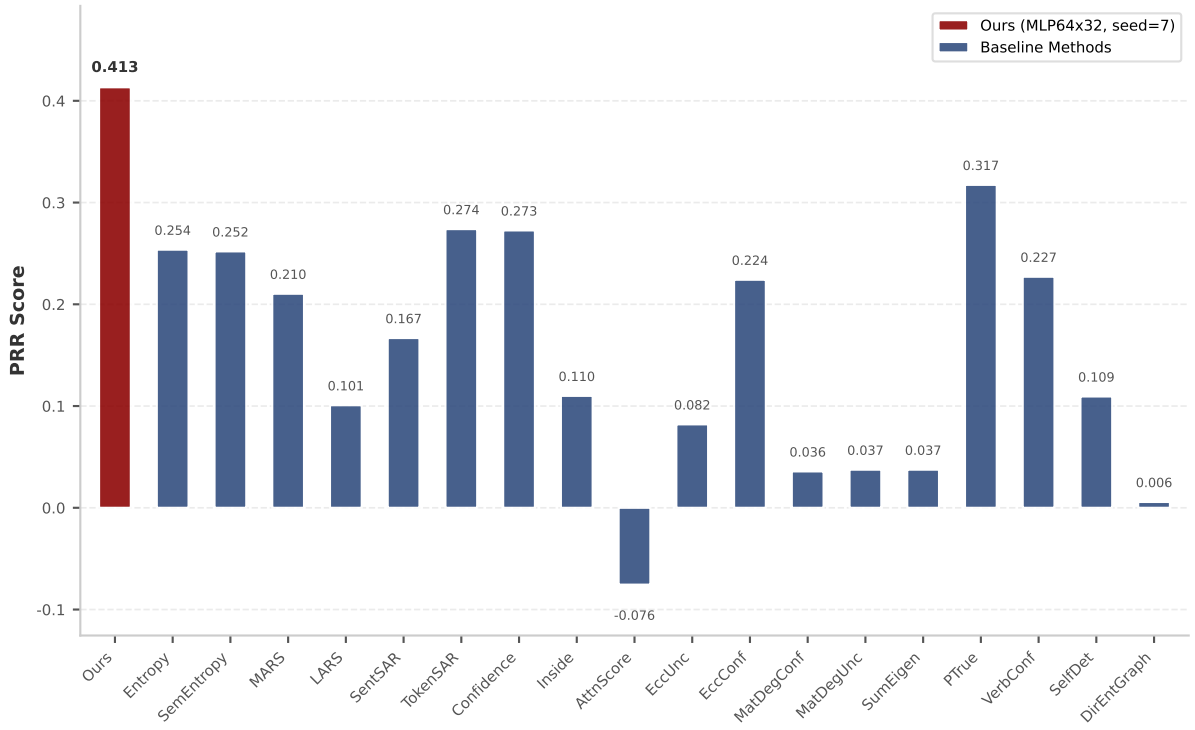

Figure 8: We compared D^2-RAG's composite uncertainty feature method against 18 individual uncertainty estimation methods across four major categories, using the Qwen2.5-7B model. The evaluation was conducted on the MedQA dataset, with the evaluation metric being the PRR score.

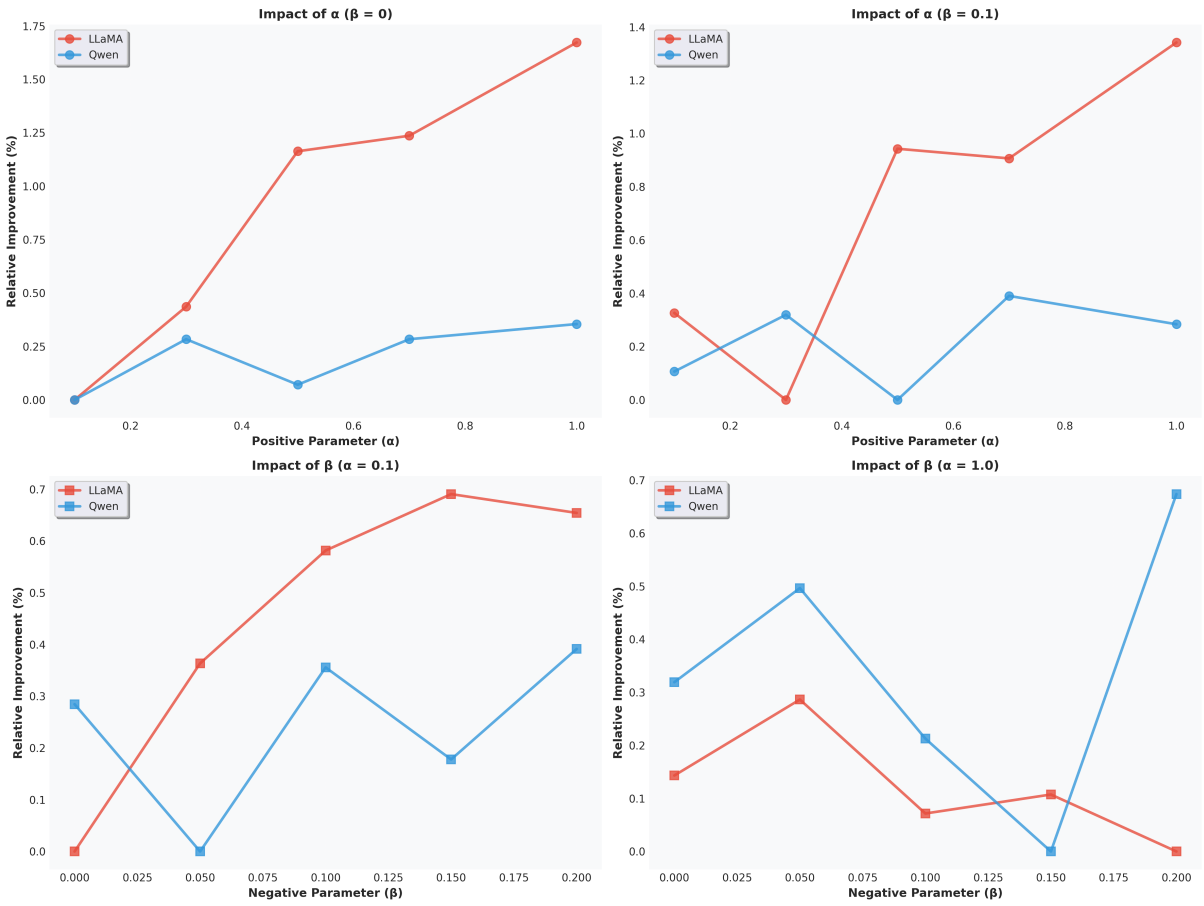

Figure 9: We analyzed the independent influence of the α_{pos} (α) and α_{neg} (β) parameters using the control variable method. The evaluation was performed on the MedQA dataset using the Llama3.1-8B and Qwen2.5-7B models, with the evaluation metric being the percentage of relative performance improvement.

Algorithm 1 D^2-RAG Inference

Require: Query q, Language Model $\mathcal{M}$, Retriever $\mathcal{R}$, Document Corpus $\mathcal{D}$
Require: Trained decision model $\mathcal{M}_{\text{ret}}$, Isotonic regressors $\{IR_k\}_{k=1}^{K}$
Require: Thresholds τ (retrieval), τ_{pos} (positive), τ_{neg} (negative)
Require: Strength coefficients α_{pos}, α_{neg}
Ensure: Answer a

Stage 1: Uncertainty-Aware Retrieval Decision
Generate parametric answer: $a_{\text{para}} \leftarrow \mathcal{M}(q)$
Compute K uncertainty scores: $\mathbf{s} \leftarrow [s_1, \ldots, s_K]$
Calibrate scores: $\mathbf{u} \leftarrow [\text{IR}_1(s_1), \ldots, \text{IR}_K(s_K)]$
Predict no-retrieval probability: $p \leftarrow \mathcal{M}_{\text{ret}}(\mathbf{u})$
if $p > \tau$ **then**
  return a_{para} {No retrieval needed}
end if
Stage 2: Utility-Aware Contrastive Decoding
Retrieve documents: $\mathcal{C} \leftarrow \mathcal{R}(q, \mathcal{D})$
Compute retrieval utility: $\delta \leftarrow \text{RetrievalUtility}(q, \mathcal{C})$
for each decoding step t **do**
  Compute dynamic entropy factor: w_t
  Compute $\text{logit}_{\text{RAG}}(y_t)$ and $\text{logit}_{\text{Para}}(y_t)$
  if $\delta > \tau_{\text{pos}}$ **then**
    $\lambda_t \leftarrow \alpha_{\text{pos}} \cdot w_t$ {Relevant context}
    $\text{logit}_{\text{final}} \leftarrow (1 + \lambda_t) \cdot \text{logit}_{\text{RAG}} - \lambda_t \cdot \text{logit}_{\text{Para}}$
  else if $\delta < \tau_{\text{neg}}$ **then**
    $\lambda_t \leftarrow \alpha_{\text{neg}} \cdot w_t$ {Noisy context}
    $\text{logit}_{\text{final}} \leftarrow (1 + \lambda_t) \cdot \text{logit}_{\text{Para}} - \lambda_t \cdot \text{logit}_{\text{RAG}}$
  else
    $\text{logit}_{\text{final}} \leftarrow \text{logit}_{\text{RAG}}$ {Neutral context}
  end if
  Sample y_t from $\text{softmax}(\text{logit}_{\text{final}})$
end for
return a

Adaptive-RAG (Jeong et al., 2024). This method trains a classifier to predict query complexity and dynamically selects the most suitable retrieval strategy accordingly, ranging from no retrieval to single-step retrieval to iterative multi-step retrieval, based on the predicted complexity level.

CAD (Shi et al., 2024). This is an inference-time method that requires no extra training. It mitigates the issue of the model overlooking retrieved content and over-relying on parametric knowledge by contrasting the model's output probability distributions with and without context to suppress the model's prior knowledge and amplify the influence of context information.

DoLa (Chuang et al., 2024). This is a decoding strategy that requires no external retrieval or additional fine-tuning. It leverages the characteristic that higher layers of the Transformer model encode more factual knowledge. By contrasting the output logits distributions of the high and low layers, it amplifies the influence of factual knowledge, thereby improving the factuality of the generated content.

N Evaluation Metric

We employed different evaluation metrics for different experiments.

Overall Performance of the D^2-RAG Framework: We used Accuracy as the primary metric, which measures the proportion of correctly answered questions in the medical Q&A task. Specifically, we compare the model's predicted option against the correct answer to calculate the proportion of correctly predicted samples out of the total samples. Performance of the Retrieval Decision Model: To assess the retrieval decision model's

performance relative to adaptive retrieval baseline methods, we used the F1 score and Recall. Recall measures the retrieval decision model's ability to correctly identify all samples that genuinely require retrieval. In the high-stakes field of medical Q&A, recall is particularly crucial, as failing to perform necessary retrieval could lead to diagnostic errors, which pose a much greater risk than the computational cost of performing unnecessary retrieval. Advantage of Our Composite Uncertainty Feature Vector: To evaluate the superiority of our constructed composite uncertainty feature vector relative to single uncertainty measures, we used the AUROC score and the PRR (Prediction Rejection Ratio) score as evaluation metrics. AUROC (Area Under the Receiver Operating Characteristic Curve) measures the ability of the uncertainty score to distinguish between correct and incorrect predictions. Since it is insensitive to the classification threshold, it can globally assess the discriminative performance of the uncertainty estimation method. PRR (Prediction Rejection Ratio) evaluates the performance of the uncertainty estimation method by constructing a rejection curve. Specifically, this curve describes how the average generation quality of the samples varies with the threshold when only samples whose uncertainty score is below a certain threshold are retained. The PRR index is calculated by taking the difference between the Area Under the Curve (AUC) of the evaluated uncertainty method and the AUC of the random strategy, divided by the difference between the AUC of the ideal (Oracle) strategy and the AUC of the random strategy, thereby achieving normalization. The PRR ranges from 0.0 to 1.0, where a higher value indicates better performance of the uncertainty estimation method. The calculation formula is as follows:

$$PRR = \frac{\text{AUC}_{\text{unc}} - \text{AUC}_{\text{rnd}}}{\text{AUC}_{\text{oracle}} - \text{AUC}_{\text{rnd}}}, \tag{38}$$

where AUC_{unc} is the Area Under the Curve for the uncertainty method being evaluated, AUC_{rnd} is the Area Under the Curve for random ranking, and $\text{AUC}_{\text{oracle}}$ is the Area Under the Curve for the ideal ranking based on true quality scores.

O Case Study

We validate the effectiveness of the dual-stage decision framework through three representative cases. D^2-RAG successfully handles three typical scenarios: (a) avoiding unnecessary retrieval when the model has sufficient knowledge (Figure 10), (b) amplifying the contribution of high-quality retrieved documents (Figure 11), and (c) suppressing the interference from noisy context (Figure 12). In all cases, Naive RAG produces incorrect answers while D^2-RAG outputs correct ones.

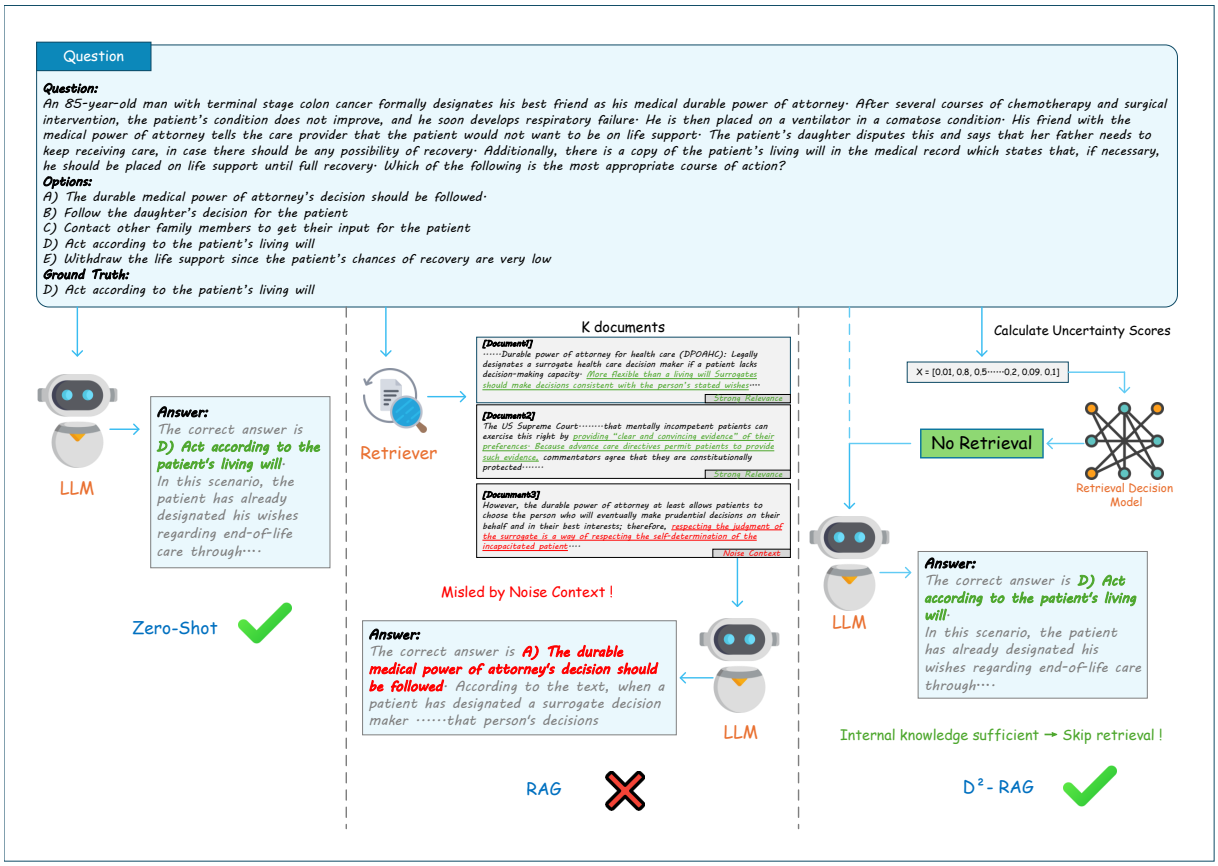

Figure 10: Case study: Avoiding unnecessary retrieval. The model has sufficient parametric knowledge to answer correctly, and D^2-RAG skips retrieval while Naive RAG retrieves irrelevant documents leading to an incorrect answer.

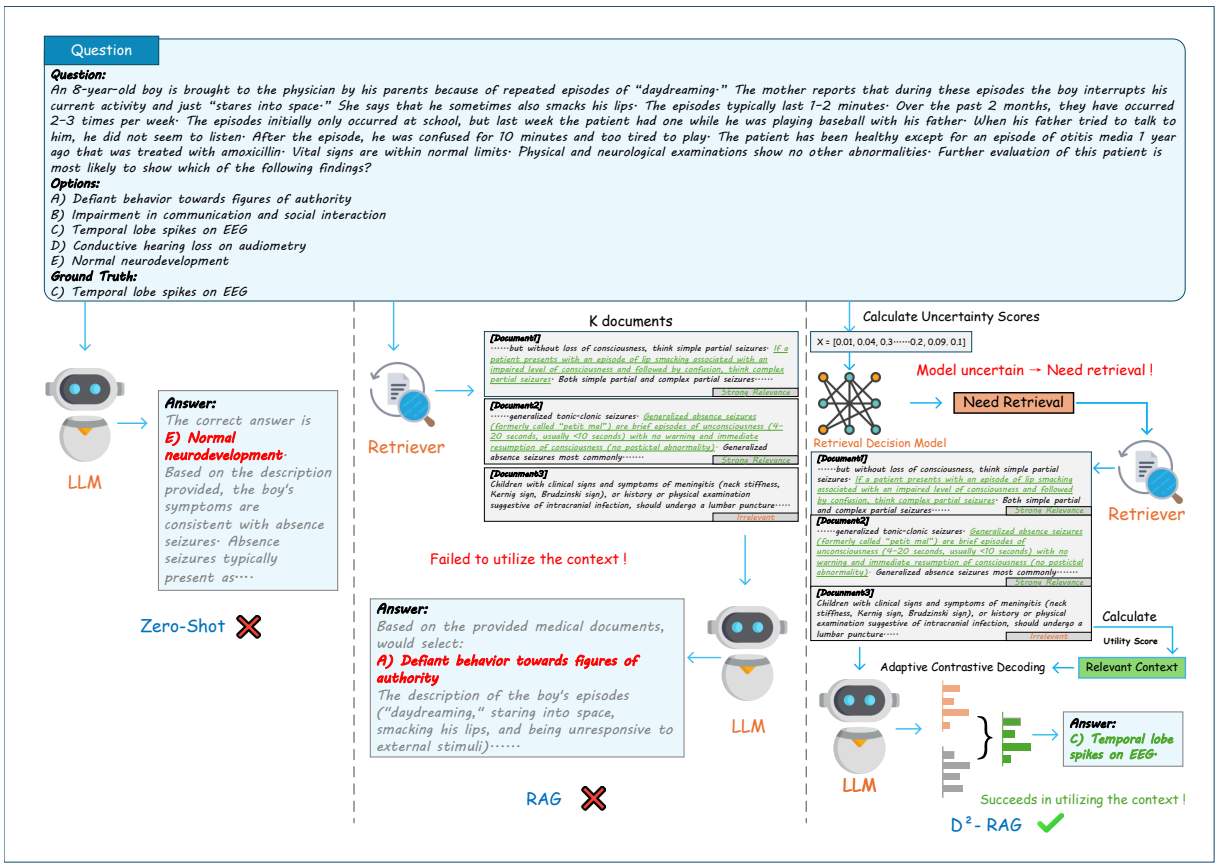

Figure 11: Case study: Amplifying helpful context. D^2-RAG leverages high-quality retrieved documents through positive contrastive decoding, while Naive RAG fails to fully utilize the relevant information.

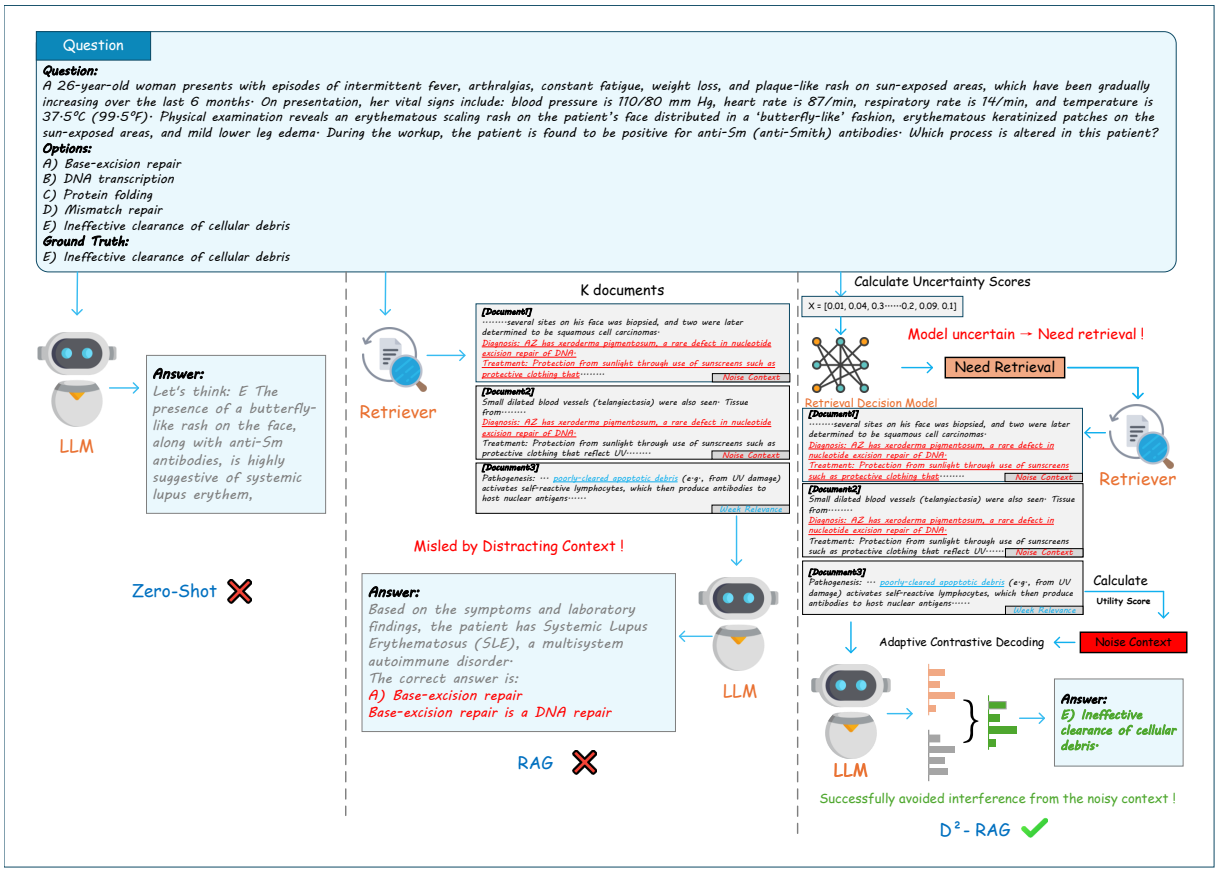

Figure 12: Case study: Suppressing noisy context. D^2-RAG mitigates the interference from misleading retrieved documents through negative contrastive decoding, while Naive RAG is misled by the noise.