Lightweight Residual Probes for Fast Hallucination Detection

Rohan Bhatnagar*[1], Youran Sun*[2], Chi Andrew Zhang[3], Yixin Wen†[4], and Haizhao Yang†[1,2]

[1]Department of Computer Science, University of Maryland, College Park, MD, USA
[2]Department of Mathematics, University of Maryland, College Park, MD, USA
[3]Department of Statistics, University of Chicago, Chicago, IL, USA
[4]Department of Geography, University of Florida, Gainesville, FL, USA
rbhatna1@terpmail.umd.edu sun1245@umd.edu sacz@uchicago.edu
yixin.wen@ufl.edu hzyang@umd.edu

January 16, 2026

Abstract

Hallucination in large language models (LLMs) can be understood as a failure of faithful readout: although internal representations may encode uncertainty about a query, decoding pressures still yield a fluent answer. We propose lightweight residual probes that read hallucination risk directly from intermediate hidden states of question tokens, motivated by the hypothesis that these layers retain epistemic signals that are attenuated in the final decoding stage. The probe is a small auxiliary network whose computation is orders of magnitude cheaper than token generation and can be evaluated fully in parallel with inference, enabling near-instantaneous hallucination risk estimation with effectively zero added latency in low-risk cases. We deploy the probe as an agentic critic for fast selective generation and routing, allowing LLMs to immediately answer confident queries while delegating uncertain ones to stronger verification pipelines. Across four QA benchmarks and multiple LLM families, the method achieves strong AUROC and AURAC, generalizes under dataset shift, and reveals interpretable structure in intermediate representations, positioning fast internal uncertainty readout as a principled foundation for reliable agentic AI.

1 Introduction

Large language models (LLMs) have made remarkable progress recently [Yang et al., 2025, Comanici et al., 2025] and are being applied in an increasing number of real-world scenarios. However, hallucination, which refers to situations where a model produces information that appears plausible but is actually false or not supported by facts, undermines users' trust in LLMs and limits their use in critical settings. Therefore, developing effective methods for hallucination detection is critical. Ideally, such methods should maintain real-time responsiveness without increasing response latency or significantly increasing generation costs.

We propose a lightweight hallucination-detection method based on question intermediate representations that predicts hallucination risk efficiently and accurately. The additional computation introduced by our method is less than 1‰ of that required to generate a single token, so our approach adds negligible cost to generation. Moreover, the detector can be evaluated in parallel with inference, introducing no extra latency to the user. In essence, we aim to build a "lie detector" or "mind reader" for LLMs.

With an effective hallucination detector, we can build an *LLM router* that adaptively allocates computation. Given a user query, the default LLM begins generating a response while our detector estimates the hallucination risk in parallel. Because the detector only requires the question's intermediate representations, it can run concurrently with generation. If the predicted risk is low, the system returns the generated response as usual; otherwise, it routes the query to a slower but more reliable pipeline (e.g., a stronger model, reasoning-augmented generation, cross-model verification, or retrieval-augmented

*Equal contribution.
†Co-corresponding author.

generation (RAG)). In the low-risk case, this design introduces zero additional latency; in the fallback case, the extra delay is less than the time to generate a single token.

Our motivation follows naturally from recent evidence that LLMs perform substantial hidden consideration during forward propagation. Prior work [Lindsey et al., 2025, Chen et al., 2025] suggests that intermediate representations encode reasoning, planning, and control signals that guide generation, yet these signals may not be faithfully expressed in the final text. This motivates directly reading out hallucination-related signals from intermediate representations, rather than relying on generated results.

Based on the above observations, we hypothesize that **intermediate representations encode uncertainty signals**. This is analogous to how students taking an exam can sense "I don't know this" for a question, yet would never write that on the answer sheet. LLMs exhibit a similar pattern because they are trained to provide an answer whenever possible. This motivates us to build a small detector that captures this internal sense of knowing or not knowing. Moreover, this analogy suggests that intermediate layers should be more informative than the final output, just as the internal feeling differs from what appears on the answer sheet.

In our experiments, we indeed find that using representations from intermediate layers is often more effective than using the final-layer representations. We attribute this to the fact that the final layer has already been decoded into the token space, leading to significant information loss. In other words, the model contains internal features related to confidence or hallucination, but these features are unnecessary for output generation, so they may be discarded in the last few layers.

Furthermore, we only use the representations aligned to the question as input to the detector. This choice is driven by the need to balance detection quality and response latency. In our experiments, we find that question-only representations are already sufficient for accurate detection. Including answer representations yields a marginal improvement but incurs substantial latency overhead.

Our main contributions are as follows.

- We propose a lightweight hallucination detection method that incurs negligible computational overhead and can be evaluated in parallel with inference.
- Building on this detector, we develop a pipeline that routes queries with high hallucination risk to stronger models to improve generation quality without increasing response latency or token generation cost. Preliminary experiments show that this pipeline improves answer correctness by xxx.
- We conduct extensive ablations and identify several properties of intermediate representations. (a) Intermediate-layer representations are more effective than the final layer for hallucination detection. (b) Question-aligned representations are often sufficient for accurate risk estimation.

2 Related Work

Table 1: Summary of hallucination detection methods.

Method	Sampling	Features Used	Q/A Features Used
Perplexity	No Need	Output Logits	Answer
Semantic Entropy [Farquhar et al., 2024]	Need	Output	Answer
Lexical Similarity [Lin et al., 2024]	Need	Output	Answer
SelfcheckGPT [Manakul et al., 2023]	Need	Output	Answer
EigenScore [Chen et al., 2024]	Need	Middle Hidden States	Answer
P(I Know) [Kadavath et al., 2022]	No Need	Last Hidden State	Question
True Direction [Bürger et al., 2024]	No Need	Last Hidden State	Answer
HaloScope [Du et al., 2024]	No Need	Middle Hidden States	Answer
HARP [Hu et al., 2025]	No Need	Last Hidden State	Answer
Ours	No Need	Middle Hidden States	Question

The perplexity of an LLM's answer can itself serve as an indicator for hallucination detection, but as shown in [Ren et al., 2023], it is unreliable. [Kuhn et al., 2023] and [Farquhar et al., 2024] propose semantic entropy. They require the LLM to generate multiple answers to a given question, then cluster them, and determine the hallucination likelihood based on the entropy of the clusters. Higher semantic entropy indicates a higher probability of hallucination. Similarly, [Lin et al., 2024],

[Manakul et al., 2023], and [Chen et al., 2024] detect hallucinations based on consistency over multiple sampled answers.

However, a practical hallucination detector should require much less computation, at least smaller than the cost of generating an answer with the LLM. [Kadavath et al., 2022] trained a classifier that uses the last hidden state of the question to predict whether the model knows the answer, i.e., P(I know). [Bürger et al., 2024] used the last hidden state of the answer to learn, via linear regression, two directions, the True direction and the False direction, and performed hallucination detection within this two-dimensional subspace.

Based on the above methods, HARP [Hu et al., 2025] and HaloScope [Du et al., 2024] employ SVD to project the last or intermediate hidden states corresponding to the answer tokens into a low-dimensional subspace, after which a two-layer MLP is used to regress the hallucination score. As a follow-up to HaloScope, [Park et al., 2025] attempts to identify hallucinations by modifying the intermediate representations of the LLM and then classifying the last hidden state.

3 Methodology

Problem Definition We formulate hallucination detection as a *knowledge prediction* task. Given a user query q, our goal is to estimate whether the LLM contains the correct answer before or during generation. During the forward process, the LLM produces hidden representations $h_l \in \mathbb{R}^{N\times D}$ for each layer $l \in [0, L]$, where N is the sequence length, D is the hidden dimension, and L is the number of transformer layers. We use a lightweight auxiliary network g_θ that takes the hidden representation h as input and outputs a scalar score

$$p = g_\theta(h),$$

where $p \in [0, 1]$ represents the probability that the model can correctly answer the query q. A low value of p indicates that the model is uncertain or likely to hallucinate when generating the response.

Feature Extraction from LLM The detector relies on the LLM's hidden representations to infer whether the model is confident about a given query. In principle, representations from any transformer layer can be used as input. We choose to extract features from *intermediate layers* rather than the final one. As discussed earlier, the last layer is optimized for next-token prediction and may discard signals unrelated to generation. Intermediate layers, by contrast, retain richer representations that capture the model's internal uncertainty.

We also restrict the detector's input to hidden states corresponding only to the *question tokens*. Formally, the feature we use can be represented as

$$h_l = \text{Transformer}_l(q),$$

where h_l denotes the hidden representation of the query(question) q at layer l. This choice reduces computation and avoids the need to process the model's generated text. More importantly, it allows the hallucination likelihood to be estimated *before* any answer is produced. Methods that depend on analyzing the generated output must first wait for the model to finish responding, then, if the answer is found unreliable, invoke a stronger model to regenerate it, effectively doubling the latency. Our approach avoids this inefficiency by evaluating the question representations during the original forward pass, so the system can proactively switch to a more reliable model and introduce almost no additional delay. For factual benchmarks, we achieve performance comparable to or better than output-based methods, using only the question representations without waiting for the model's response.

Training Objective We construct the training data from question-answering benchmarks that include reference answers. For each question q with the standard answer a^*, the target LLM generates one answer a. We then obtain a correctness label by using an external judge to compare (q, a) with a^*. In particular, we use `gpt4o` for to produce more accurate labels than rule-based evaluation methods. In principle, multiple answers can be sampled for each question to estimate the probability of correctness. Still, in practice, we generate only one answer per question due to time and cost limitations. The resulting label is $y \in \{0, 1\}$, where $y = 1$ means the generated answer is correct and $y = 0$ means it is hallucinated or incorrect.

Given the detector output $p \in [0, 1]$, we employ a binary cross-entropy objective. For a dataset with M examples $\{(q_i, y_i)\}_{i=1}^{M}$ and detector scores p_i, we minimize

$$\mathcal{L} = -\frac{1}{M} \sum_{i=1}^{M} \big(y_i \log p_i + (1 - y_i) \log(1 - p_i) \big).$$

In closed-book question answering, the model generates answers solely from its parametric knowledge without access to external sources. An incorrect answer in this setting indicates that the model has produced content inconsistent with factual ground truth, which aligns with the definition of factual hallucination. We therefore treat correctness labels as a proxy for hallucination detection, following prior work in this area [Du et al., 2024, Farquhar et al., 2024].

Network Architecture (Hallucination Detector) We experiment with two architectures for the hallucination detector, MLP and Transformer, as summarized in Table 2. Both networks take the hidden representations of the question h_l as input and output a single confidence score p.

For the MLP architecture, the input must have a fixed length. Therefore the hidden representation $h_l \in \mathbb{R}^{N \times D}$ is first compressed into a single vector in $\mathbb{R}^D$. We consider several simple aggregation methods, including mean pooling, max pooling, and using the hidden state of the last token, which often corresponds to the question mark. We also consider applying principal component analysis (PCA) to the hidden representations and retain the top n principal components. The resulting pooled vector or concatenated principal components are then fed into an MLP that predict the confidence score p.

The transformer architecture is more flexible since it naturally processes sequence representations. In this case, the hidden states h_l are directly used as input to a lightweight transformer encoder. This model can capture token-level interactions within the question without requiring input-stage pooling. At the output stage, we apply attention pooling to aggregate the sequence into a single vector. Specifically, a small MLP computes a scalar score for each token, and the final representation is a softmax-weighted sum of all token representations.

Table 2: Detector architecture configurations. Input dimension varies by base LLM (e.g., 3584 for Qwen-2.5-7B).

Architecture	Hidden Dim	Layers	Params	Pooling
MLP	128–1024	4	3M–37M	Mean / Max / Last
Transformer	256–512	4–8	4M–30M	Attention

4 Experiments

Table 3: Summary of four QA benchmarks used for hallucination detection evaluation.

Dataset	**Train / Val / Test Size**	**Type**	**Topic**
TriviaQA	87,622 / 11,313 / –	Open-domain QA	General knowledge, trivia facts
NQ-Open	87,925 / 3,610 / –	Open-domain QA	Wikipedia, factual reasoning
MMLU-Pro	– / – / 12,032	Multiple-choice QA	14 subjects, professional knowledge
WebQuestions	3,778 / – / 2,032	Open-domain QA	Freebase, entity-centric

Datasets and models Our experiments use four open-domain question answering benchmarks, namely TriviaQA [Joshi et al., 2017], NQ-Open [Kwiatkowski et al., 2019], MMLU-Pro [Wang et al., 2024], and WebQuestions [Berant et al., 2013]. TriviaQA contains general knowledge trivia questions in an unfiltered no-context setting. NQ-Open consists of naturally occurring queries from Google Search paired with Wikipedia-based answers. MMLU-Pro provides multiple-choice questions spanning 14 professional and academic subjects. WebQuestions includes entity-centric questions originally collected from the Google Suggest API. We evaluate our approach on three model families, namely LLaMA-2-Chat-7B [Touvron et al., 2023], Qwen2.5-7B-Instruct [Yang et al., 2024], and Gemma-3-4B-it [Team, 2025]. Ablation studies on model scale are conducted using the 14B and 32B instruct variants of Qwen2.5.

Evaluation Metrics AUROC (area under the ROC curve) is employed as the primary evaluation metric. AUROC measures a binary classifier's ability to separate positives from negatives across all thresholds, ranging from 0 to 1. A higher AUROC is indicative of stronger discriminative power. AURAC (area under the rejection-accuracy curve) and accuracy are also measured. AURAC reveals whether the confidence of a binary classifier can be trusted, with higher values indicating a strong correlation between correctness and confidence.

Baselines We compare our method with the highest performing open baselines, including HaloScope [Du et al., 2024] and Semantic Entropy [Farquhar et al., 2024].

Correctness labeling We use GPT-4o to obtain correctness labels for LLM-generated answers. For each question, we prompt GPT-4o with the original question, the ground-truth answer, and the LLM-generated answer, asking it to return a binary correctness label (see Appendix A for the full prompt). This approach provides more robust evaluation than rule-based string matching, particularly for open-ended questions where semantically equivalent answers may differ in surface form.

4.1 Main Results

Table 4: **Qwen2.5-7B performance on the measured benchmarks.** Comment on class imbalance.

Dataset	Correct	Incorrect
TriviaQa	-	-
NQ-Open	-	-
MMLU-Pro	-	-
WebQuestions	-	-

Table 5: Hallucination detection performance (AUROC, %) on four QA benchmarks across three model families. Bold indicates best performance per model-dataset pair. Our method consistently outperforms baselines in most settings.

Model	Method	TriviaQA	NQ-Open	MMLU-Pro	WebQuestions
LLaMA 2 Chat 7B	HaloScope	77.40	67.0	**79.6**	72.6
	Semantic Entropy	81.23	77.57	76.76	80.12
	Ours (question)	**82.10**	79.87	71.22	80.79
	Ours (answer)	-	**83.76**	77.85*	**82.76**
Qwen-2.5-7B-Instruct	HaloScope	85.5	70.3	81.1	80.4
	Semantic Entropy	78.48	80.22	74.05	80.71
	Ours (question)	87.31	83.28	83.21	84.01
	Ours (answer)	**93.95**	**88.43**		**87.67**
Gemma-3-4b-it	HaloScope	76.5	79.62	**77.7**	76.7
	Semantic Entropy	76.08	75.48	57.92	76.76
	Ours (question)	83.93	82.96	74.65	79.63
	Ours (answer)	**88.80**	-	75.53*	**80.78**

Table 5 presents the main results. Our method achieves the best AUROC on 10 out of 12 model-dataset combinations. On Qwen-2.5-7B-Instruct, our approach outperforms the strongest baseline by 4.1 points on TriviaQA, 3.1 points on NQ-Open, and 2.2 points on MMLU-Pro. Similar gains are observed on Gemma-3-4b-it, where we surpass prior methods by 7.4 points on TriviaQA and 4.4 points on WebQuestions. For LLaMA 2 Chat 7B, our method matches or exceeds Semantic Entropy on most benchmarks. The only exception is MMLU-Pro on Gemma-3-4b-it, where HaloScope achieves 77.7% compared to our 74.65%. Overall, the results demonstrate that intermediate-layer representations of question tokens provide a strong signal for hallucination detection, competitive with or superior to methods that require sampling multiple outputs.

4.2 Out of Distribution Results

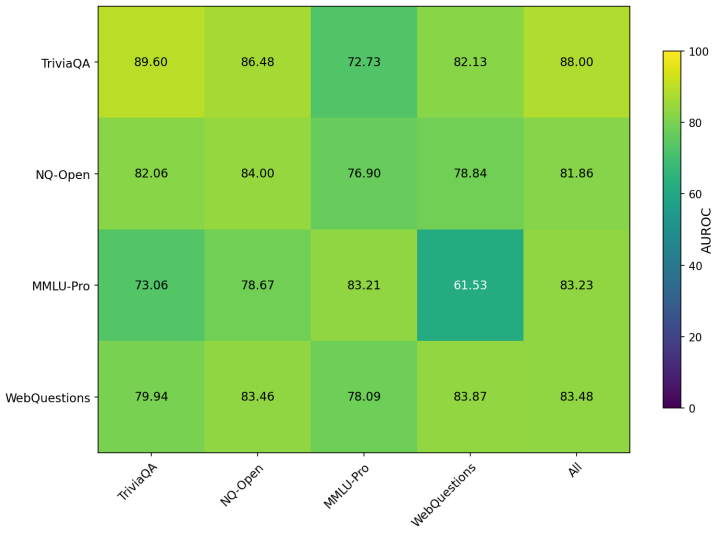

Figure 1: Out-of-distribution generalization. Heatmap of AUROC when training the detector on one dataset (columns) and evaluating it on another (rows), including training on the union of all datasets (*All*).

We evaluate whether our detector generalizes under distribution shift. Unless otherwise specified, we extract hidden states from layer 22 of Qwen-2.5-7B-Instruct, which yields the best overall performance in our main results (Table 5).

We train the detector on each of the four datasets in Table 5 and evaluate it on the remaining datasets. We also train on the union of all datasets (*All*) and test on each dataset. Figure 1 reports the AUROC for each train–test pair.

Overall, the detector exhibits strong cross-dataset transfer. Most off-diagonal entries remain in the 0.76–0.86 AUROC range, suggesting that the intermediate representations capture dataset-agnostic signals predictive of hallucination. Training on the union (*All*) yields a strong and stable overall model. It reaches AUROC = 88.00 on TriviaQA, 81.86 on NQ-Open, 83.23 on MMLU-Pro, and 83.48 on WebQuestions. While *All* does not uniformly dominate the best single-source training choice for every target, it provides competitive performance across datasets. It also reduces sensitivity to the particular source distribution, supporting our conclusion that the learned features are largely transferable.

4.3 Accuracy after Hallucination Removed

Transformer has really high AURAC –> What this means is that when we are confident we are correct. Assess accuracy at different conference thresholds (i.e. only answer when we are > x confidence)

x: threshold y: accuracy of filtered answers

Table 6: Qwen2.5 model family performance by Benchmark.

Benchmark	Model	AUROC	AURAC	F1
NQ-Open	7B			
	14B			
	32B			
WebQuestions	7B			
	14B			
	32B			

Table 7: Qwen2.5 model family performance by Benchmark.

Benchmark	Model	AUROC	AURAC	F1
TriviaQA	7B	89.6	82.27	82.86
	14B	88.72	88.54	85.32
	32B			
NQ-Open	7B	83.11	53.94	51.9
	14B	83.85	63.29	**70.0**
	32B	**84.22**	**67.49**	68.65
MMLU-Pro	7B	83.21	60.0	-
	14B			
	32B			
WebQuestions	7B	84.0	61.9	68.17
	14B	84.97	68.86	**74.75**
	32B	**85.13**	**70.32**	73.6
All	7B			
	14B			
	32B			

4.4 Ablation Study 1: Embds corresponding to the question is enough

run on answer and question, expect the answer's result is not higher than the question's by a lot. (inference isn't worth it) - Select 50-100 false positives and compare with answer and without.

4.5 Ablation Study 2: Not All Layers are Made Equal

we haven't definitely concluded that intermediate layers are better, but we know for sure that some layers consistently perform better than other.

4.6 Ablation Study 3: larger model is better

4.7 Ablation Study 4: MLP vs Transformer

For the small network architecture, I have three ideas:

1. Take the intermediate representation from a particular layer of the LLM, with shape length(L) × embd_dim(D), apply max pooling to transform it to 1xD, then feed it into a fully connected network
2. Directly feed the intermediate representation from a particular layer of the LLM into a shallow (1-3 layers) transformer

Results for MLP:

Table 8: Probing Results for Qwen2.5-7B Last Hidden State Embeddings

Layer	**Best Val AUROC**	**Best Val Accuracy**	**Best Val AURAC**	**MLP Parameters**
Layer 19	**0.9320**	85.76%	0.8475	3.03 M
Layer 20	0.9289	85.72%	0.8466	3.03 M
Layer 21	0.9271	85.32%	0.8463	3.03 M
Layer 22	0.9252	85.37%	0.8444	3.03 M
Layer 23	0.9224	84.75%	0.8434	3.03 M
Layer 24	0.9205	84.65%	0.8398	3.03 M

5 Discussion

We also observe a consistent scaling trend in our experiments. The detector tends to perform better when applied to stronger, larger LLMs. This trend has also been reported in prior work [Du et al., 2024, Kadavath et al., 2022]. One possible explanation is that larger models have greater capacity and may

Table 9: Probing Results for Qwen2.5-7B Mean-Pool Embeddings

Table 10: **Does this (below) this indicates that there is a uncertainty "direction" that is common between all hallucinatoin directions. while most of the pooled vector smooths across tokens, this direction is probably consistent.**

Layer	Best Val AUROC	Best Val Accuracy	Best Val AURAC	MLP Parameters
Layer 19	**0.9213**	84.73%	0.8403	3.03 M
Layer 20	0.9187	83.98%	0.8376	3.03 M
Layer 21	0.9155	83.86%	0.8372	3.03 M
Layer 22	0.9122	84.02%	0.8359	3.03 M
Layer 23	0.9087	83.21%	0.8308	3.03 M
Layer 24	0.9065	82.90%	0.8284	3.03 M

Table 11: Probing Results for Qwen2.5-7B Max-Pool Embeddings

Layer	Best Val AUROC	Best Val Accuracy	Best Val AURAC	MLP Parameters
Layer 19	**0.9094**	82.14%	0.8119	3.03 M
Layer 20	0.9087	82.96%	0.8168	3.03 M
Layer 21	0.9059	80.03%	0.8142	3.03 M
Layer 22	0.9004	80.88%	0.7949	3.03 M
Layer 23	0.8947	81.60%	0.7933	3.03 M
Layer 24	0.8875	80.49%	0.7892	3.03 M

perform richer latent computation beyond next-token prediction. As a result, the final text can be less faithful to the intermediate state. This can increase the gap between intermediate representations and surface outputs, making intermediate features more informative for our detector and improving its effectiveness on larger models.

References

[Berant et al., 2013] Berant, J., Chou, A., Frostig, R., and Liang, P. (2013). Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1533–1544.

[Bürger et al., 2024] Bürger, L., Hamprecht, F. A., and Nadler, B. (2024). Truth is universal: Robust detection of lies in llms.

[Chen et al., 2024] Chen, C., Liu, K., Chen, Z., Gu, Y., Wu, Y., Tao, M., Fu, Z., and Ye, J. (2024). Inside: Llms' internal states retain the power of hallucination detection.

[Chen et al., 2025] Chen, Y., Benton, J., Radhakrishnan, A., Uesato, J., Denison, C., Schulman, J., Somani, A., Hase, P., Wagner, M., Roger, F., Mikulik, V., Bowman, S. R., Leike, J., Kaplan, J., and Perez, E. (2025). Reasoning models don't always say what they think.

[Comanici et al., 2025] Comanici, G., Bieber, E., Schaekermann, M., Pasupat, I., Sachdeva, N., Dhillon, I., Blistein, M., Ram, O., Zhang, D., Rosen, E., Marris, L., Petulla, S., et al. (2025). Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities.

[Du et al., 2024] Du, X., Xiao, C., and Li, Y. (2024). Haloscope: Harnessing unlabeled llm generations for hallucination detection.

[Farquhar et al., 2024] Farquhar, S., Kossen, J., Kuhn, L., and Gal, Y. (2024). Detecting hallucinations in large language models using semantic entropy. *Nature*, 630(8017):625–630.

[Hu et al., 2025] Hu, J., Tu, G., Cheng, S., Li, J., Wang, J., Chen, R., Zhou, Z., and Shan, D. (2025). Harp: Hallucination detection via reasoning subspace projection.

[Joshi et al., 2017] Joshi, M., Choi, E., Weld, D. S., and Zettlemoyer, L. (2017). Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension.

[Kadavath et al., 2022] Kadavath, S., Conerly, T., Askell, A., Henighan, T., Drain, D., Perez, E., Schiefer, N., Hatfield-Dodds, Z., DasSarma, N., Tran-Johnson, E., Johnston, S., El-Showk, S., Jones, A., Elhage, N., Hume, T., Chen, A., Bai, Y., Bowman, S., Fort, S., Ganguli, D., Hernandez, D., Jacobson, J., Kernion, J., Kravec, S., Lovitt, L., Ndousse, K., Olsson, C., Ringer, S., Amodei, D., Brown, T., Clark, J., Joseph, N., Mann, B., McCandlish, S., Olah, C., and Kaplan, J. (2022). Language models (mostly) know what they know.

[Kuhn et al., 2023] Kuhn, L., Gal, Y., and Farquhar, S. (2023). Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation.

[Kwiatkowski et al., 2019] Kwiatkowski, T., Palomaki, J., Redfield, O., Collins, M., Parikh, A. P., Alberti, C., Epstein, D., Polosukhin, I., Devlin, J., Lee, K., Toutanova, K., Jones, L., Kelcey, M., Chang, M.-W., Dai, A. M., Uszkoreit, J., Le, Q., and Petrov, S. (2019). Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

[Lin et al., 2024] Lin, Z., Trivedi, S., and Sun, J. (2024). Generating with confidence: Uncertainty quantification for black-box large language models.

[Lindsey et al., 2025] Lindsey, J., Gurnee, W., Ameisen, E., Chen, B., Pearce, A., Turner, N. L., Citro, C., Abrahams, D., Carter, S., Hosmer, B., Marcus, J., Sklar, M., Templeton, A., Bricken, T., McDougall, C., Cunningham, H., Henighan, T., Jermyn, A., Jones, A., Persic, A., Qi, Z., Thompson, T. B., Zimmerman, S., Rivoire, K., Conerly, T., Olah, C., and Batson, J. (2025). On the biology of a large language model. `https://transformer-circuits.pub/2025/attribution-graphs/biology.html`. Accessed: 2025-10-07; mechanistic interpretability study of Claude 3.5 Haiku.

[Manakul et al., 2023] Manakul, P., Liusie, A., and Gales, M. J. F. (2023). Selfcheckgpt: Zero-resource black-box hallucination detection for generative large language models.

[Park et al., 2025] Park, S., Du, X., Yeh, M.-H., Wang, H., and Li, Y. (2025). Steer llm latents for hallucination detection.

[Ren et al., 2023] Ren, J., Luo, J., Zhao, Y., Krishna, K., Saleh, M., Lakshminarayanan, B., and Liu, P. J. (2023). Out-of-distribution detection and selective generation for conditional language models.

[Team, 2025] Team, G. (2025). Gemma 3 technical report.

[Touvron et al., 2023] Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., Bikel, D., Blecher, L., Ferrer, C. C., Chen, M., Cucurull, G., Esiobu, D., Fernandes, J., Fu, J., Fu, W., Fuller, B., Gao, C., Goswami, V., Goyal, N., Hartshorn, A., Kirnap, U., Kivlichan, I., Lachaux, M.-A., Lavril, T., Lee, J., Liskovich, D., Lu, Y., Mao, Y., Martinet, X., Mihaylov, T., Mishra, P., Molybog, I., Nie, Y., Poulton, A., Reizenstein, J., Rungta, R., Saladi, K., Schelten, A., Silva, R., Smith, E. M., Subramanian, R., Tan, X. E., Tang, B., Taylor, R., Williams, A., Kuan, J. X., Koura, P. S., Bhosale, S., Narayanan, D., Angelidis, A., Shankar, V., Wolf, T., Rodriguez, A., Stojanov, S., Lample, G., Rocktäschel, T., Joulin, A., Bojanowski, P., Grave, E., and Conneau, A. (2023). Llama 2: Open foundation and fine-tuned chat models.

[Wang et al., 2024] Wang, Y., Ma, X., Zhang, G., Ni, Y., Chandra, A., Guo, S., Ren, W., Arulraj, A., He, X., Jiang, Z., Li, T., Ku, M., Wang, K., Zhuang, A., Fan, R., Yue, X., and Chen, W. (2024). Mmlu-pro: A more robust and challenging multi-task language understanding benchmark.

[Yang et al., 2025] Yang, A., Li, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Gao, C., Huang, C., Lv, C., Zheng, C., Liu, D., Zhou, F., Huang, F., Hu, F., Ge, H., Wei, H., Lin, H., Tang, J., Yang, J., Tu, J., Zhang, J., Yang, J., Yang, J., Zhou, J., Zhou, J., Lin, J., Dang, K., Bao, K., Yang, K., Yu, L., Deng, L., Li, M., Xue, M., Li, M., Zhang, P., Wang, P., Zhu, Q., Men, R., Gao, R., Liu, S., Luo, S., Li, T., Tang, T., Yin, W., Ren, X., Wang, X., Zhang, X., Ren, X., Fan, Y., Su, Y., Zhang, Y., Zhang, Y., Wan, Y., Liu, Y., Wang, Z., Cui, Z., Zhang, Z., Zhou, Z., and Qiu, Z. (2025). Qwen3 technical report.

[Yang et al., 2024] Yang, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Li, C., Liu, D., Huang, F., Wei, H., Lin, H., Yang, J., Tu, J., Zhang, J., Yang, J., Yang, J., Zhou, J., Lin, J., Dang, K., Lu, K., Bao, K., Yang, K., Yu, L., Li, M., Xue, M., Zhang, P., Zhu, Q., Men, R., Lin, R., Li, T., Tang, T., Xia, T., Ren, X., Ren, X., Fan, Y., Su, Y., Zhang, Y., Wan, Y., Liu, Y., Cui, Z., Zhang, Z., and Qiu, Z. (2024). Qwen2.5 technical report. arXiv preprint arXiv:2412.15115v2.

A Correctness Evaluation Prompt

We use the following prompt template to obtain correctness labels from GPT-4o.

```
Evaluate whether the generated answer is CORRECT or INCORRECT.
Question: {question}
Ground truth: {answer}
Generated: {generated_text}
A generated answer is CORRECT if it expresses the same meaning
as the ground truth, without introducing incorrect, conflicting,
or extra information. Otherwise, it is INCORRECT.
Respond with EXACTLY "true" or "false".
```