CS230

Probe-Steering: Guiding LLM Behavior Away From Hallucination

Walid Rahman
walidr@stanford.edu

Alex Popa
rpopa@stanford.edu

Dilan Nana
dilan99@stanford.edu

Abstract

Given the recent proliferation of large language models (LLMs) in personal and enterprise settings, ensuring they behave appropriately has never been more important. While these LLMs are certainly powerful, reducing their potential for generating irrelevant or nonsensical outputs has remained a consistent problem, thereby jeopardizing their trustworthiness, particularly in safety-critical and reliability-sensitive contexts. This study introduces Probe-Steering, a method for steering language models towards improved expressions of uncertainty instead of producing a hallucinatory output. We introduce a novel approach of combining Semantic Entropy Probes (SEPs) and activation steering to guide the model towards expressing uncertainty instead of attempting to respond regardless of output integrity when its output exhibits high entropy. We evaluate Probe-Steering's effectiveness on Llama2 7B using the SQuAD2.0 dataset and demonstrate that this approach reduces likelihood of hallucination and erroneous results by 47%. Moreover, this paper illuminates promising future directions that can be taken to disincentivize LLMs from hallucinating and instead communicate uncertainty.

1 Introduction

Large Language Model (LLMs) systems, like Chat-GPT [15] are remarkable tools for a variety of fields like medicine and law due to their capabilities in text generation, question-answering, and summarization. However, their tendency to produce incorrect or arbitrary outputs, widely characterized as 'hallucinations,' poses significant risks to users, especially in high-stakes contexts. Hallucinations can foster mistrust in these systems, and more importantly, introduce life-altering risks such as faulty legal or medical analyses [8, 21, 25, 14, 20]. One type of hallucination we specifically research in this work is a 'confabulation' or output that is simultaneously incorrect and arbitrary, as defined in [5].

Existing mitigation strategies like reinforcement learning from human feedback (RLHF) and multiple-response sampling show promise but face limitations in scalability and generalizability across tasks [5, 19, 9, 7, 3, 2, 1, 4, 12, 13]. For example, [8] found that sampling-based hallucination mitigation can incur up to a 10x increase in computational overhead. To address these challenges, we leverage Semantic Entropy Probes (SEPs) [8], which efficiently quantify uncertainty and detect hallucinations from hidden states in a single forward pass, avoiding the costs of traditional semantic entropy (SE) methods.

Building upon SEP-based detection, we extend their use to mitigate hallucinations. Specifically, we use steering vectors developed by [16], which efficiently guide model behavior during inference, to guide Llama2 7B[23] away from hallucinatory behavior and towards behaviors that uncertainty-expressing responses.

Our Probe-Steer method uses the SQuAD2.0 validation dataset as input, a standard benchmark for accuracy and consistency, because it offers especially challenging questions, such as ones that are impossible to answer, allowing us to measure the improvement of our intervention in leading to refusals to answer in cases where no answer exists. We combine SEPs and steering vectors into

,

a unified detection and mitigation framework guided by a custom three-layer Probe-Steer neural network we developed to dynamically apply vectors based on probe measurements in real-time. This approach produces significant improvement compared to baselines, with a reduction of 47% in hallucination behavior and a 95% increase in average score.

Our unified detection and mitigation framework aims to bridge the gap observed in previous works where identifying and reducing hallucination behavior are treated as separate efforts. The broader goal of this work is to demonstrate a scalable, computationally efficient approach for combating hallucinatory behavior and a potential for enhancing the safety of LLMs.

2 Related Work

As large language models (LLMs) like Llama and GPT4o continue to increase in usage and capability, addressing problematic behaviors such as hallucinations becomes increasingly challenging. Our work builds on the established relationship between semantic entropy and uncertainty [5, 11, 7, 10] to identify hallucinatory behavior. Prior studies, such as [6], demonstrate that measuring uncertainty through signals in the generation process allows for the detection of problematic behavior without fine-tuning. While these studies validate semantic entropy as a signal, they do not propose lightweight, real-time detection or mitigation methods.

Contrastive Activation Addition (CAA) [16] introduces a real-time steering method for Llama2 7B [23], applying steering vectors to mitigate undesired behaviors such as sycophancy, corrigibility, and hallucination. Although their approach improves LLM accuracy and efficiency in question-answering tasks, we extend their methodology by integrating semantic entropy measurements to guide the application of steering vectors. Our work focuses on promoting uncertainty expression rather than solely improving accuracy.

Semantic Entropy Probes (SEPs), introduced in [8], offer a computationally efficient method for detecting hallucinations by extracting semantic entropy signals from hidden states in a single generation, avoiding the need for expensive sampling. While SEPs have been used for detection, our work expands their application by dynamically steering Llama2 7B based on real-time uncertainty measurements, enabling simultaneous detection and mitigation of hallucinations. Other approaches, such as [18], explore entropy-based steering but primarily aim to reduce entropy for deterministic outputs. In contrast, our method leverages entropy to unify detection and mitigation, steering models toward specific, contextually appropriate behaviors.

For evaluation, we reference the benchmarking methodology of [24], which assesses factual correctness in short-form question-answering tasks, incorporating adversarial questions designed to provoke sycophantic or hallucinatory behavior. Our approach extends this to long-form contexts by addressing factual consistency through real-time hallucination detection and mitigation during generation.

3 Dataset and Features

We used the SQuAD2.0 dataset as a representative benchmark for testing our methodology. Each dataset entry includes a question, context, potential answers, and an "is impossible" flag indicating whether the question is answerable from the given context. This enabled us to identify hallucinations: if a model used information not found in the context or attempted to answer unanswerable questions, the response was deemed hallucinatory. Conversely, correct answers aligned with the context were deemed accurate.

We focused on the test subset of SQuAD2.0, containing 11,911 samples (6,008 unanswerable, 5,903 answerable), split into new train (8,311), test (1,799), and validation (1,801) sets to avoid potential overlap with model training and address computational constraints. See 8.1 for information.

4 Methods

4.1 Hallucination Detection with Semantic Entropy Probes

Semantic Entropy Probes (SEPs) are logistic regression models trained to predict semantic entropy ($H_{SE}(x)$) using the hidden states of large language models (LLMs). For a given input x, the hidden state $h_{lp}(x)$ at a specific layer l and token position p is paired with a semantic entropy value, computed by sampling high-temperature responses. See 8.2 for more details on SEP calculations.

SEPs model the relationship between hidden states and SE, enabling readings of uncertainty. High H_s values indicate greater uncertainty, and therefore, high likelihood that the corresponding output will be a hallucination. Analyzing the probability of high entropy across layers reveals points where the model consistently exhibits uncertainty. We see that the middle layers (10-20) of Llama2 7B experience consistent modulations, especially ar the second to last token. See 8.3 for graphs showing SEP behavior.

4.2 Steering via Contrastive Activation Addition

We first steer Llama2 7B without SEPs using the steering vectors from [16]. These vectors were formed by the activation outputs of pairs of contrasting dataset entries. This method isolates desirable behaviors by contrasting them with undesirable ones, crystallizing the former into a vectors that can be applied during inference. We anticipated that vectors targeting hallucination reduction or refusal to answer would shift output distributions away from hallucinatory behavior. Detailed formulas are derived in [16] and referenced in the appendix.

4.3 Generating SEPs and Steered Answers

Following the methods in [8], we trained SEPs to generate high-entropy probabilities for each of Llama2 7B's 33 layers during inference. Using a context-restricted prompt designed to enforce refusal for insufficient information, we generated answers while reading SEPs at every layer over two token positions (SLT and TBG), yielding 66 probabilities per generation. These represent the sole input features of the Probe-Steer neural net.

We applied four steering vectors (hallucination, 2xhallucination, refusal, 2xrefusal) at layer 13, identified by [16] as optimal for steering. Model activations were reset before each vector application to prevent interaction effects between two different vectors. Using the same context-restricted prompt as mentioned earlier, we generated answers without recomputing SE. The resulting dataset comprised 11,911 rows of question-context-answer triplets, 66 SE probabilities, the unsteered model answer, and 4 answers from the steered model. This is the complete dataset of features and "ground truths" for the training, validation, and test of our Probe-Steer neural net.

4.4 Scoring Improvements over OpenAI's SimpleQA

OpenAI's SimpleQA [24] benchmark (Oct 2024) evaluates LLM factuality using GPT-4o to assign grades of A (correct), B (incorrect), and C (not attempted) based on their rubric. However, we observed frequent inaccuracies, such as assigning grade C to correct responses when Llama2 7B provided longer outputs than the gold truth answer in SQuAD. These inconsistencies highlighted limitations in the SimpleQA scoring approach as strictly effective in a short-form factuality environment.

To address this, we developed a custom GPT-4o-based scoring system, employing two prompts: (1) to detect refusals, and (2) to validate correctness against ground truth answers. This two-tiered framework signficantly improved grading consistency and reduced errors. See 8.4 for the details.

Manual examination revealed that our scoring system reduced our grader's error and hallucination rates. This improvement likely stems from simplifying GPT-4o's task to binary True/False judgments, and subsequently assigning more nuanced grades downstream.

LLM responses were scored as follows: 10 for correct answers, -10 for incorrect answers, and 0 for outright refusals. Each sample produced five scores: one score for the base model output and four scores for the steered outputs. See 8.5 for scoring results.

4.5 Kullback-Leibler (KL) Divergence Loss Function

For our custom Probe-Steer neural network, we used the Kullback-Leibler (KL) divergence loss formula, used similarly in [22], defined as:

$$D_{KL}(P||Q) = \sum_{x \in X} P(x) \log(P(x)/Q(x))$$

and apply it in PyTorch [17]. While our problem appears initially to be a multi-class classification problem, the class encoding is not "one hot" which makes it less tractable to cross-entropy methods. As there are three possible scores for each vector/class, we use KL loss to predict the likelihood (desirability) of each class. Each class represents one of the vectors. We first normalize the distribution of answer grades for each of the $4 + 1$ classes (steering vectors + unsteered) into "ground truth" probability distributions. Answers with high grade scores receive high probabilities, while low (e.g., negative) scores receive low probabilities. The KL loss can then optimize the model's prediction of log probabilities relative to target probabilities, derived from answer scores. Minimizing KL loss effectively maximizes (test-score normalized) probability distribution fit, which is equivalent to maximizing the predicted score on our test.

5 Experiments/Results/Discussion

5.1 Probe-Steer Neural Net Experiments

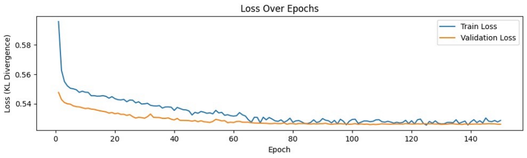
We explored multiple Probe-Steer neural network configurations, iterating extensively on network architecture, problem modeling, and loss functions to optimize performance. Initial experiments employed a single steering vector, binary cross-entropy classifiers (using "one-hot" encodings for the best class), and XGBoost, demonstrating that Semantic Entropy Probes (SEPs) effectively encode information for predicting optimal or improved steering activations.

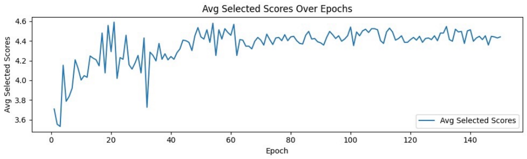
After adopting KL divergence loss for multiclass probability prediction, we evaluated Probe-Steer neural network architectures of varying complexity. Larger networks (5–8 layers with 32–128 neurons) tended to overfit, even with regularization and dropout, while smaller networks (1–3 layers with fewer than 30 neurons) underfit and lacked predictive power. The final Probe-Steer architecture—a fully connected network with 30×20×10 neurons—achieved the best trade-off between capacity and generalization.

Optimization was performed using the Adam optimizer with standard hyperparameters. Learning rate tuning was critical; rates below 10^{-3} failed to converge, while larger rates caused instability. A "reduce on plateau" scheduler was used to refine learning rates during later epochs.

To address overfitting, we employed batch normalization, dropout, and weight decay. Dropout rates and batch size were especially impactful. Due to memory constraints, we selected a mini-batch size of 64, balancing computational efficiency with loss reduction.

Finally, we note that validation loss alone is insufficient to evaluate system performance. Our primary metric is the average score improvement on the SQuAD2.0 test, which better reflects the practical utility of the steering activations than simple prediction of target probability distributions.





Metric Type	Metric Name	Value	Description
Average Score	Unsteered	2.36	Average unsteered score
	Hallucination	4.08	Average 1xHallucination Score
	2xHallucination	4.50	Average 2xHallucination Score
	Refusal	3.83	Average 1xRefusal Score
	2xRefusal	3.13	Average 2xRefusal Score
	Total Predicted Scores	4.60	Predicted average score
Unsteered Grades	$A_{\text{unsteered}}$	0.52	Unsteered Correct
	$B_{\text{unsteered}}$	0.28	Unsteered Incorrect
	$C_{\text{unsteered}}$	0.20	Unsteered Refusal
Steered Grades	A_{pred}	0.61	Predicted Correct
	B_{pred}	0.15	Predicted Incorrect
	C_{pred}	0.24	Predicted Refusal
Steering Type Distribution	Unsteered	0.20	Proportion of Unsteered predictions
	Hallucination	0.34	Proportion of Hallucination predictions
	2xHallucination	0.46	Proportion of 2xHallucination predictions
	Refusal	0.00	Proportion of Refusal predictions
	2xRefusal	0.00	Proportion of 2xRefusal predictions

Table 1: Evaluation Metrics Summary, Class Scores, and Predicted Class Distributions.

5.2 Results and Discussion

Our results, see 1, highlight significant performance improvements for Llama2-7b with our Probe-Steer approach. The unsteered baseline scored an average of 2.36/10, while our Probe-Steer method improved scores by 95%, achieving 4.6/10. This shows that leveraging SEPs for real-time uncertainty estimations from a single generation is an effective and parsimonious hallucination detection method.

Quantitatively, our approach increased correct answers (A) by 17%, reduced incorrect answers (B) by 47%, and raised refusals to answer (C) by 20%. These results indicate a shift toward reliability, steering the model to prioritize accurate or qualified responses or refuse to answer when highly uncertain. While uniformly applying steering vectors shows modest improvements, our Probe-Steer neural network surpassed the best single vector result (2x Hallucination Reduction) by an addition 2%. This was achieved through selective steering, with our system dynamically applying vectors to 80% of cases and adjusting vector strength (1x/2x) based on SEP measurements. The network effectively determined the most impactful vectors to apply while minimizing the use of less effective refusal vectors. These findings highlight opportunities for future refinement, such as transitioning to proportional steering, where vector strength varies continuously, and developing a broader library of steering vectors tailored to different tasks.

Our Probe-Steer framework enables real-time detection and mitigation of hallucinations, either at start of generation (by using TBG SEPs) or after a first pass (using SLT SEPs). This cost-effective approach reduces computational demands compared to traditional methods, and shows potential for enabling safer scaling of LLMs for diverse real-world applications.

6 Conclusion/Future Work

Concluding, our work showed that SEPs could be used to preemptively steer LLMs away from hallucination at low computational cost (a maximum of one extra generation). The Probe-Steer techniques show potential to reduce LLM hallucinations significantly, by as much as half. Without the computational burden of traditional methods, Probe-Steer holds the promise of real-time hallucination detection and prevention at scale.

Resource Limitations

Our work was limited by computational resources, particularly GPU memory, restricting us to a single instance of the Llama2 7B model requiring 28GB of memory. To overcome this, we precomputed SEP activations and LLM outputs for training, validation, and testing, generating 50,000 outputs and twice as many scoring calls to GPT-4o. These constraints prevented exploration of larger steering vector combinations, highlighting the need for scalable methods beyond precomputed data for future improvements in performance.

Future Work

This work opens several promising directions for advancing the application of steering vectors in language model alignment. A natural next step is to expand the experiment to include the full suite of seven steering vectors from CAA [16], as well as exploring continuous steering approaches, which could be enabled with sufficient computational resources. Incorporating pre-softmax probabilities from the KL model to generate outputs as linear combinations of steering vectors, rather than individual vector applications, represents a compelling opportunity to enhance model control. Extending these methods to challenging datasets such as TruthfulQA and OpenAI's SimpleQA will be critical to evaluate robustness in adversarial contexts. Scaling the Probe-Steer techniques to larger models, such as Llama3-70B, will provide valuable insights into generalizability and scalability.

7 Contributions

Walid: literature search, cloud infrastructure, compute, dataset prep, split, and load, semantic probes, steering vectors, scoring, neural nets, hyperparameter optimizations

Alex: literature search, dataset search, scoring, problem modeling, system design, KL loss function, neural nets, hyperparameter optimizations, debugging

Dilan: baseline experiments, literature search, dataset analyses, related works, dataset formal investigations

7.1 Our Code

`https://github.com/Walid-Rahman2/semantic_steer_cs230`

References

[1] Jiuhai Chen and Jonas Mueller. "Quantifying uncertainty in answers from any language model and enhancing their trustworthiness". In: *arXiv 2308.16175* (2023).

[2] Jeremy R Cole et al. "Selectively answering ambiguous questions". In: *EMNLP* (2023).

[3] Jinhao Duan et al. "Shifting attention to relevance: Towards the uncertainty estimation of large language models". In: *arXiv:2307.01379* (2023).

[4] Mohamed Elaraby et al. "Halo: Estimation and reduction of hallucinations in open-source weak large language models". In: *arXiv:2308.11764* (2023).

[5] Sebastian Farquhar et al. "Detecting hallucinations in large language models using semantic entropy". In: *Nature* 630.8017 (2024), pp. 625–630.

[6] Yuheng Huang et al. "Look before you leap: An exploratory study of uncertainty measurement for large language models". In: *arXiv preprint arXiv:2307.10236* (2023).

[7] Saurav Kadavath et al. "Language models (mostly) know what they know". In: *arXiv:2207.05221* (2022).

[8] Jannik Kossen et al. "Semantic entropy probes: Robust and cheap hallucination detection in llms". In: *arXiv preprint arXiv:2406.15927* (2024).

[9] Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. "Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation". In: *ICLR*. 2023.

[10] D. V. Lindley. "On a Measure of the Information Provided by an Experiment". In: *The Annals of Mathematical Statistics* 27.4 (1956), pp. 986–1005. DOI: `10.1214/aoms/1177728069`. URL: `https://doi.org/10.1214/aoms/1177728069`.

[11] David J. C. MacKay. "Information-Based Objective Functions for Active Data Selection". In: *Neural Computation* 4.4 (July 1992), pp. 590–604. ISSN: 0899-7667. DOI: `10.1162/neco.1992.4.4.590`. eprint: `https://direct.mit.edu/neco/article-pdf/4/4/590/812354/neco.1992.4.4.590.pdf`. URL: `https://doi.org/10.1162/neco.1992.4.4.590`.

[12] Potsawee Manakul, Adian Liusie, and Mark JF Gales. "Selfcheckgpt: Zero-resource black-box hallucination detection for generative large language models". In: *Conference on Empirical Methods in Natural Language Processing*. 2023.

[13] Sewon Min et al. "Factscore: Fine-grained atomic evaluation of factual precision in long form text generation". In: *EMNLP* (2023).

[14] Andreas L Opdahl et al. "Trustworthy journalism through AI". In: *Data Knowl. Eng.* (2023).

[15] OpenAI et al. *GPT-4 Technical Report*. 2024. arXiv: `2303.08774 [cs.CL]`. URL: `https://arxiv.org/abs/2303.08774`.

[16] Nina Panickssery et al. "Steering llama 2 via contrastive activation addition". In: *arXiv preprint arXiv:2312.06681* (2023).

[17] Adam Paszke et al. "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *CoRR* abs/1912.01703 (2019). arXiv: `1912.01703`. URL: `http://arxiv.org/abs/1912.01703`.

[18] Nate Rahn, Pierluca D'Oro, and Marc G. Bellemare. *Controlling Large Language Model Agents with Entropic Activation Steering*. 2024. arXiv: `2406.00244 [cs.CL]`. URL: `https://arxiv.org/abs/2406.00244`.

[19] J Schulman. *Reinforcement learning from human feedback: progress and challenges.* Apr. 2023. URL: `www.youtube.com/watch?v=hhiLw5Q_UFg`.

[20] Yiqiu Shen et al. "ChatGPT and Other Large Language Models Are Double-edged Swords". In: *Radiology* (2023).

[21] Karan Singhal et al. "Large Language Models Encode Clinical Knowledge". In: *Nature* (2023).

[22] Masahito Togami et al. *Unsupervised Training for Deep Speech Source Separation with Kullback-Leibler Divergence Based Probabilistic Loss Function*. 2019. arXiv: 1911.04228 [eess.AS]. URL: https://arxiv.org/abs/1911.04228.

[23] Hugo Touvron et al. *Llama 2: Open Foundation and Fine-Tuned Chat Models*. 2023. arXiv: 2307.09288 [cs.CL].

[24] Jason Wei et al. "Measuring short-form factuality in large language models". In: *arXiv preprint arXiv:2411.04368* (2024).

[25] Benjamin Weiser. "Lawyer Who Used ChatGPT Faces Penalty for Made Up Citations". en. In: *The New York Times* (June 2023).

8 Appendix

8.1 Dataset Specifications

To ensure that the train, validation, and test sets had relatively equal distributions of answerable and unanswerable questions, we made sure that each subset was evenly composed of answerable and unanswerable questions.

Split	Unanswerable	Answerable	Total
Train	4193	4118	8311
Val	908	893	1801
Test	907	892	1799

Table 2: Dataset splits constructed from SQuAD Test.

8.2 SEP Formulas

The binary label for semantic entropy computation 1. The optimal threshold γ^* is determined as 2. The "low" and "high" groups are defined as 3. The means for the two groups are given by 4.

$$\hat{H}_{SE}(x) = \begin{cases} 1 & \text{if } H_{SE}(x) > \gamma^*, \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

$$\gamma^* = \arg\min_{\gamma} \sum_{j \in \text{SE}_{\text{low}}} \left(H_{SE}(x_j) - \hat{H}_{\text{low}} \right)^2 + \sum_{j \in \text{SE}_{\text{high}}} \left(H_{SE}(x_j) - \hat{H}_{\text{high}} \right)^2. \tag{2}$$

$$\text{SE}_{\text{low}} = \{ j : H_{SE}(x_j) < \gamma \}, \quad \text{SE}_{\text{high}} = \{ j : H_{SE}(x_j) \geq \gamma \}. \tag{3}$$

$$\hat{H}_{\text{low}} = \frac{1}{|\text{SE}_{\text{low}}|} \sum_{j \in \text{SE}_{\text{low}}} H_{SE}(x_j), \quad \hat{H}_{\text{high}} = \frac{1}{|\text{SE}_{\text{high}}|} \sum_{j \in \text{SE}_{\text{high}}} H_{SE}(x_j). \tag{4}$$

8.3 SEP Plots

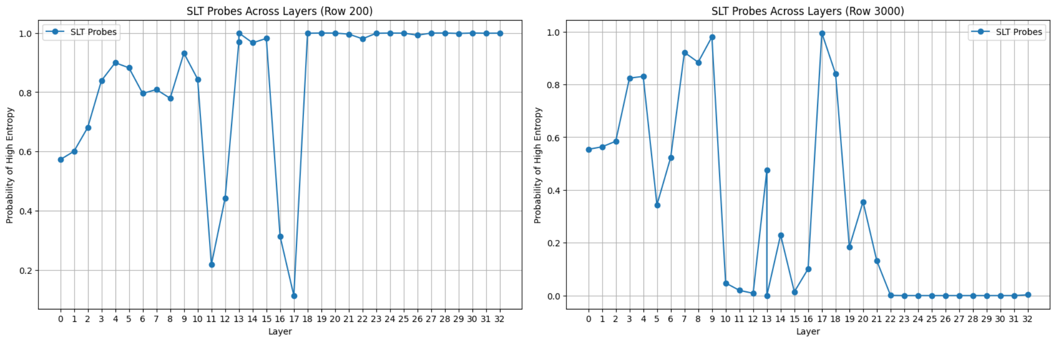


Figure 1: Probe Measurements from Second Last Token (SLT) Model Response Across Layers

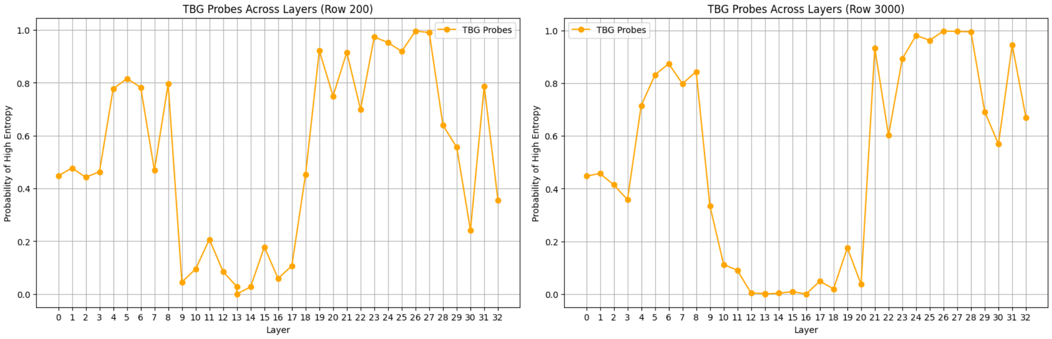


Figure 2: Probe Measurements from Token Before Generating (TBG) Model Response Across Layers

8.4 Our Custom Scoring Prompts and Rubric

(1) is a prompt that instructs GPT4o to return True for refusals to answer and False for attempts to answer. (2) is a prompt that instructs GPT4o to return True for correct answers and False for incorrect answers. Note that (1) and (2) are system prompts. The exact prompts are provided in the code.

1. Apply (1) to determine if the answer is a refusal or not.
2. If (1) is True (a refusal to answer) and the question is unanswerable, the generation is correct.
3. If (1) is True and the question is answerable, the generation is a true refusal.
4. If (1) is False and the question is unanswerable, the generation is incorrect.
5. If (1) is False and the question is answerable, use prompt (2) to determine if the answer is incorrect or correct when compared to the ground truth answer.

8.5 Custom Scoring System Results

Set	**Grade**	**Base**	**Hallucination**	**2xHallucination**	**Refusal**	**2xRefusal**
Train	**Correct**	4340	4922	4232	4736	4330
	Incorrect	2353	1636	407	1703	1614
	Refusal	1618	1753	3672	1872	2367
Test	**Correct**	930	1082	904	1054	934
	Incorrect	506	348	95	365	371
	Refusal	363	369	800	380	494
Validation	**Correct**	907	1036	922	999	937
	Incorrect	503	358	83	370	350
	Refusal	391	407	796	432	514

Table 3: Number of Correct, Incorrect, and Refusal per Answer across Train, Test, and Validation Sets