Why It Failed: A Benchmark to Evaluate Interpretability*

Joel Mathew[1][0009−0009−1981−7073], Aditya Lagu[1], Anthony Tang[1], and Prudhviraj Naidu[1,2]

[1] Algoverse AI Research
joel.mathew@sjsu.edu
[2] University of California San Diego

Abstract. We introduce *Why It Failed*, a benchmark for evaluating whether interpretability methods can explain model failures. We test last token logistic probes on Gemma-2 2B across four basic reasoning tasks and find they fail to predict model failures, achieving near-chance performance across all tasks. Our benchmark provides a standardized framework to evaluate whether interpretability methods can explain model failures. Lastly, we motivate the AI community to move beyond reporting quantitative metrics and seek explanations of when and why models fail.

Keywords: Interpretability · Benchmark · Language Models · Probing · Model Failures

1 Introduction

Large language models achieve impressive performance on diverse benchmarks, yet they still fail in puzzling and unpredictable ways. A model might score 80% on a commonsense reasoning task, but this aggregate metric obscures crucial questions: *When* does the model fail? *Why* does it fail? Are failures random noise, or do they follow systematic patterns? Current evaluation practices report quantitative metrics without explaining the underlying failure modes, leaving practitioners uncertain about when and where models can be safely deployed.

The interpretability community has made significant progress in understanding what models know—identifying circuits that implement specific algorithms [19], discovering human interpretable concepts in models' activations [4]. However, interpretability methods are typically validated on their ability to detect unwanted behaviors and monitor models in deployment [14], rather than on their ability to systematically explain common failure cases of models on standard benchmarks.

This gap has important consequences. Without systematic explanations of failures, we cannot make informed deployment decisions. Explanations enable us to distinguish model-specific quirks from dataset artifacts, and to understand

* Code available at https://github.com/anthonytang/LLM-Detection-Testing

whether failure patterns generalize across model families or emerge from particular architectures. Most fundamentally, explaining failures allows us to move beyond simply reporting benchmark scores toward building scientific theories of model limitations.

We propose *Why It Failed*, a benchmark for evaluating whether interpretability methods can explain model failures. Our contributions are twofold:

1. We introduce a benchmark framework that shifts focus from explaining what models know to explaining what they don't know.
2. We show that standard last-token probing is insufficient for predicting failures.

We invite the interpretability community to test their methods against this benchmark and push toward explanations that illuminate model limitations.

The rest of the paper is organized as follows. Section 2 reviews related work in benchmarking explanations, interpretability methods and generating explanations using SAEs. Section 3 describes our benchmark framework and how we generated our benchmark examples. Section 4 describes our probing experiment methodology and discusses our results. In Section 5, we discuss limitations of our work and provide questions for future work.

2 Related Work

2.1 Benchmarking Interpretability Methods

Mills et al. [15] propose ALMANACS, a benchmark to test how well explanations can predict model behaviour on new inputs. They focus on behaviour related to AI safety such as ethical reasoning, self preservation or harmful requests. The authors construct such safety scenarios to invoke specific model behaviour. They primarily generate explanations by blackbox methods or through attribution methods.

Our approach on the other hand focuses on explaining model failures in standard benchmarks. We select established reasoning benchmarks that measure basic model capabilities. We then establish our baseline via probing models' internal activations, which has seen great success in recognizing diverse model behaviours and steering models [2, 14].

Chandrasekaran et al. [5] study a failure prediction task for a visual question answering (VQA) model. Humans are asked to predict whether the VQA model will get a question right or wrong. They find humans are able to predict model failures above chance. However, explainable methods do not improve human performance. We on the other hand focus on pure language models and first ask what is the best predictive performance we can get for predicting model failures.

2.2 Interpretability Methods for Language Models

Recent interpretability research has focused on three key factors: understanding model traces (CoT-Interpretability), understanding the internal representations

of language models (Sparse Autoencoders), and training probes on internal representations of models. These methods vary in their approach regarding what aspect of the model they study but are focused on primarily monitoring and steering models.

Chain-Of-Thought (CoT) Interpretability analyzes reasoning traces that the language model produces before the final response. Baker et al. [1] were able to detect models reward hacking by monitoring its reasoning traces. Korbak et al. [13] argues CoT interpretability is a promising research direction; however CoTs can easily succumb to optimization pressures of directly training CoT and lose their usefulness for monitoring. Kirchner et al. [12] proposes a scheme to make models' reasoning more legible by posing the task as a prover-verifier game where the model (as a prover) has to provide legible explanations for its answer that allows a verifier to predict whether the model got the answer right or wrong.

Probing methods train auxiliary classifiers on a base model's hidden states to detect concepts that the base model was not explicitly trained for [6]. Subsequent work has shown linear probing can detect unsafe behaviour and steer the model towards a desirable behaviour [2, 14].

Sparse Autoencoders learn overcomplete, sparse decompositions of neural activations, hypothesizing that individual SAE features correspond to monosemantic concepts [8, 4]. Jiang et al. [10] use SAEs to generate explanations for differences in datasets. For GSM8K, they demonstrated that "Math word problems involving time, distance, and speed" had lower accuracy.

Our benchmark evaluates whether these interpretability methods can be repurposed to identify systematic patterns that distinguish successes from failures.

3 Why It Failed Benchmark

3.1 Benchmark Overview

Core Question: Can current interpretability methods explain why models fail? When a model makes errors on a benchmark, we want to know whether interpretability techniques can surface explanations that genuinely capture the underlying causes of these failures.

What constitutes a good failure explanation? We argue that a good explanation must satisfy three criteria:

1. **Faithful**: Given the explanation, we should be able to predict whether the model will fail on unseen inputs or when deployed in new settings.
2. **Causal**: We should be able to manipulate model performance based on the explanation, either by generating adversarial inputs that exploit the identified weakness or by improving the model performance (by identifying training dataset or methodology issues) based on the explanation.
3. **Human-interpretable**: A human examining the explanation should be able to predict whether the model will fail on new inputs, without needing to run the model or inspect its internals.

In this work, we focus on **faithfulness** as a measurable, necessary (though not sufficient) condition for good explanations. While causal interventions and human-interpretability are important criteria, we leave their systematic evaluation to future work.

Evaluation Pipeline: For each task, we construct a balanced dataset of k success cases and k failure cases. An **Explainer** (e.g., linear probe, chain-of-thought interpreter, SAEs) processes these $2k$ training instances to generate explanations. A **Predictor** then uses these explanations to classify whether the model will succeed or fail on new, unseen test instances. We measure the Predictor's performance using AUC on the held-out test set.

3.2 Task and Model Selection

Tasks: Our benchmark is constructed from four diverse reasoning tasks from standard LLM evaluation suites: PIQA (physical commonsense reasoning), BoolQ (yes/no reading comprehension), WinoGrande (common-sense coreference resolution), and Social IQa (social commonsense reasoning). These tasks span a variety of cognitive capabilities:

- **PIQA** [3]: Given a goal and two potential solutions, the model must select which action would successfully achieve the goal. For example: *"How do I ready a guinea pig cage for its new occupants?"* with options involving paper strips vs. jeans material as bedding. This tests physical commonsense about everyday object interactions.
- **BoolQ** [7]: Given a passage and a yes/no question, the model must determine the correct boolean answer. For example: *"Does ethanol take more energy to make than it produces?"* paired with a technical passage. This tests reading comprehension and factual reasoning.
- **WinoGrande** [17]: Fill-in-the-blank coreference resolution where the model must determine which noun a pronoun refers to. For example: *"John moved the couch from the garage to the backyard to create space. The _ is small."* (options: garage, backyard). This tests common-sense reasoning about spatial and physical constraints.
- **Social IQa** [18]: Questions about people's actions, intentions, and social implications. For example: *"Sydney walked past a homeless woman asking for change but did not have any money. Sydney felt bad afterwards. How would you describe Sydney?"* This tests social and emotional reasoning.

Why these tasks? These tasks represent fundamental capabilities required for real-world language understanding: physical intuition, reading comprehension, linguistic reasoning, and social awareness. Critically, despite being simple multiple-choice or boolean questions, models can fail on them for complex, non-obvious reasons. A model might fail on PIQA not due to lacking physical commonsense reasoning but due to vocabulary gaps or specific cultural contexts unrelated to physical commonsense reasoning. The tasks span a diverse range of contexts and are widely-used benchmarks in the LLM community as a way to measure LLM capability and decide whether to deploy LLMs.

Model: We use Gemma-2 2B [9], a 2-billion parameter transformer model developed by Google. We chose Gemma-2 2B because: (1) it is small enough to run efficiently on consumer hardware, enabling rapid iteration and making our benchmark accessible to researchers without extensive computational resources; (2) it is a practical, industry-grade model designed for real-world deployment rather than a synthetic research model, ensuring our findings generalize to models used in practice.

Data Collection: We construct our benchmark by collecting model responses on these four tasks and categorizing them into correct and incorrect predictions. For each task, we randomly sample from the evaluation set until we collect exactly k correct predictions and k incorrect predictions, creating a balanced dataset. All examples retain their standard multiple-choice or boolean format from the original tasks.

4 Probes Fail to Predict Model Failures

4.1 Experimental Setup

Transformer Architecture and Residual Stream. Consider a transformer model with L layers. For each layer $l \in \{1, 2, ..., L\}$ and token position t, we define the layer computation as:

$$\mathbf{x}_{\text{mid}}^{(l,t)} = \mathbf{x}_{\text{pre}}^{(l,t)} + \sum_{\text{head } h} \text{attn}^{(l,h)} \left(\mathbf{x}_{\text{pre}}^{(l,t)}, \mathbf{x}_{\text{pre}}^{(l,1:t)} \right) \tag{1}$$

$$\mathbf{x}_{\text{post}}^{(l,t)} = \mathbf{x}_{\text{mid}}^{(l,t)} + \text{MLP}^{(l)} \left(\mathbf{x}_{\text{mid}}^{(l,t)} \right) \tag{2}$$

where $\mathbf{x}_{\text{pre}}^{(l,t)} \in R^d$ is the input to layer l at position t (the pre-residual stream), d is the transformer model dimension, $\mathbf{x}_{\text{mid}}^{(l,t)} \in R^d$ is the mid-residual stream (after attention), $\mathbf{x}_{\text{post}}^{(l,t)} \in R^d$ is the output of layer l (post-residual stream), $\text{attn}^{(l,h)}$ denotes the h-th attention head in layer l, and $\text{MLP}^{(l)}$ denotes the feedforward network in layer l.

Probe Training. We train linear probes on the post-residual stream $\mathbf{x}_{\text{post}}^{(l,t)}$ at the final token position of each prompt. Given a prompt from task $\mathcal{T}$ with tokens $t_1, \ldots, t_n$, we extract activations $\mathbf{x}_{\text{post}}^{(l,n)}$ at layers $l \in \{5, 10, 15, 20, 25\}$ for Gemma-2 2B. These layers are evenly spaced to capture representation learning throughout the model's depth.

For each layer l and task $\mathcal{T}$, we train a binary logistic regression classifier:

$$\hat{y} = \sigma(\mathbf{w}^{(l)} \cdot \mathbf{x}_{\text{post}}^{(l,n)} + b) \tag{3}$$

where $\mathbf{w}^{(l)} \in R^d$ is the probe direction, $b \in R$ is a bias term, and σ is the sigmoid function. The probe is trained to predict whether the model will succeed ($y = 1$) or fail ($y = 0$) on the given prompt. We use scikit-learn's logistic regression implementation with default hyperparameters.

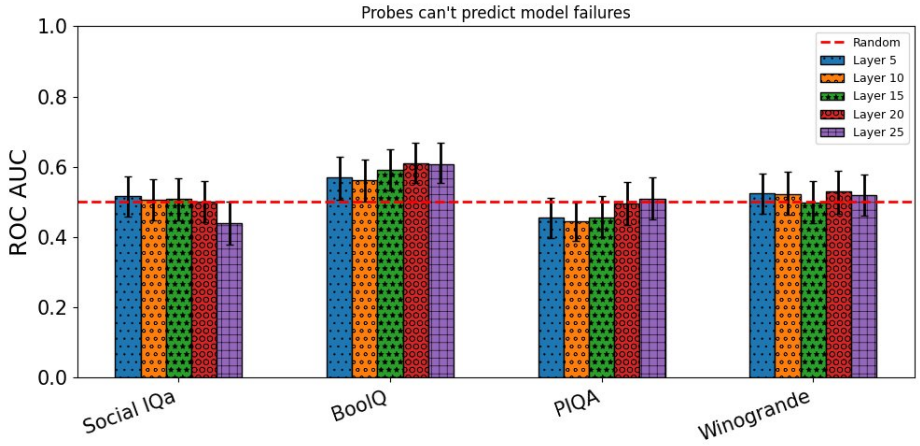

Fig. 1. Last token probes fail to predict model failures. They consistently fail for all layers except on BoolQ where latter layers slightly outperform random chance. For each task, we plot the test ROC AUC score for a probe trained on the output of a layer. Error bars refer to 99.9% confidence intervals. The dotted red line depicts random chance performance.

Data Splits. For each task, we partition our 4,000 examples (2,000 successes and 2,000 failures) into:

- **Training set**: 2,000 examples (1,000 successes, 1,000 failures)
- **Validation set**: 1,000 examples (500 successes, 500 failures)
- **Test set**: 1,000 examples (500 successes, 500 failures)

We report test set performance using Area Under the ROC Curve (AUC) as our primary metric. An AUC of 0.5 indicates chance-level performance, while 1.0 indicates perfect classification.

4.2 Results

Figure 1 shows the test set AUC for linear probes trained at different layers across all four tasks. The results reveal a striking failure: **linear probes do not outperform random chance at predicting model failures**.

Across all tasks and layers, probe performance clusters near the random baseline (AUC = 0.5, shown as a red dashed line). The best-performing configuration achieves only modest improvement: BoolQ at higher layers achieves performance slightly above chance.

We observe several consistent patterns:

1. **No layer captures failure modes**: Performance remains near chance across all chosen layers, suggesting that the distinction between success and failure is not linearly encoded at any single layer's residual stream.

2. **Task variation is minimal**: All tasks exhibit fundamentally the same pattern of near-chance performance, barring probes trained on latter layers on BoolQ. This suggests the limitation is not task-specific but reflects a broader issue with using linear probes as explainers.

These results demonstrate that linear probes, despite their success in identifying semantically meaningful directions in prior work, fail as explainers in our benchmark. The model's internal representations do not contain linearly accessible information sufficient to predict when the model will fail.

5 Discussion and Conclusion

We introduce *Why It Failed*, a benchmark for evaluating whether interpretability methods can explain model failures. Our framework operationalizes faithful explanations through predictive power: good explanations should enable prediction of failures on unseen inputs. While we focus on faithfulness in this work, our framework naturally extends to causal manipulation and human-interpretability criteria.

We showed that simple last-token probing fails to explain model failures. It remains unclear why probes slightly outperformed random chance on BoolQ. Our work can easily be extended to include:

- **Richer probing methods**: Mean-token probing across sequences, attention-probes [14]
- **Sparse representation methods**: Sparse autoencoders (SAEs) can identify clusters and offer explanations of why models fail [10]
- **Chain-of-thought interpretability**: Analyzing reasoning traces in models that produce intermediate steps. For these tasks, we did not generate reasoning traces from Gemma-2 2B. Future work would need to first generate examples of success and failure with reasoning traces.

Furthermore, we currently measure only faithfulness through predictive power. Future work should incorporate:

- **Causal metrics**: Can explanations generate adversarial examples that flip model predictions? Can they guide interventions that improve performance?
- **Human-interpretability scoring**: Auto-interpretability methods [16] can generate explanations for SAE features. Kantamneni et al. [11] propose a similar framework that could be used to explain logistic probes. However, it remains an open question whether these methods provide faithful human-interpretable explanations.

5.1 Conclusion

The *Why It Failed* benchmark asks what models don't know and why. Our key message is simple: **we should aim to explain benchmark failures, not just**

report accuracy numbers. This benchmark provides a concrete framework for evaluating whether interpretability methods achieve this goal. While linear probes fall short, they represent just the beginning. We invite the community to test their methods against this benchmark and push toward explanations that truly illuminate model limitations.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Baker, B., Huizinga, J., Gao, L., Dou, Z., Guan, M.Y., Madry, A., Zaremba, W., Pachocki, J., Farhi, D.: Monitoring reasoning models for misbehavior and the risks of promoting obfuscation (2025)
2. Beaglehole, D., Radhakrishnan, A., Boix-Adserà, E., Belkin, M.: Toward universal steering and monitoring of AI models (2025)
3. Bisk, Y., Zellers, R., Le Bras, R., Gao, J., Choi, Y.: PIQA: Reasoning about physical commonsense in natural language. In: Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence (2020)
4. Bricken, T., Templeton, A., Batson, J., Chen, B., Jermyn, A., Conerly, T., et al.: Towards monosemanticity: Decomposing language models with dictionary learning. Transformer Circuits Thread (2023), `https://transformer-circuits.pub/2023/monosemantic-features/index.html`
5. Chandrasekaran, A., Prabhu, V., Yadav, D., Chattopadhyay, P., Parikh, D.: Do explanations make VQA models more predictable to a human? In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. pp. 1036–1042. Association for Computational Linguistics (2018). https://doi.org/10.18653/v1/D18-1128
6. Chen, M., Radford, A., Child, R., Wu, J., Jun, H., Luan, D., Sutskever, I.: Generative pretraining from pixels. In: Proceedings of the 37th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 119, pp. 1691–1703. PMLR (2020)
7. Clark, C., Lee, K., Chang, M.W., Kwiatkowski, T., Collins, M., Toutanova, K.: BoolQ: Exploring the surprising difficulty of natural yes/no questions. In: Proceedings of NAACL (2019)
8. Cunningham, H., Ewart, A., Riggs, L., Huben, R., Sharkey, L.: Sparse autoencoders find highly interpretable features in language models (2023)
9. Gemma Team, Riviere, M., Pathak, S., et al.: Gemma 2: Improving open language models at a practical size (2024)
10. Jiang, N., Sun, X., Smith, L., Nanda, N.: Towards data-centric interpretability with sparse autoencoders. In: Mechanistic Interpretability Workshop at NeurIPS 2025 (2025)
11. Kantamneni, S., Engels, J., Rajamanoharan, S., Tegmark, M., Nanda, N.: Are sparse autoencoders useful? A case study in sparse probing (2025)
12. Kirchner, J.H., Chen, Y., Edwards, H., Leike, J., McAleese, N., Burda, Y.: Prover-verifier games improve legibility of LLM outputs (2024)
13. Korbak, T., Balesni, M., Barnes, E., Bengio, Y., et al.: Chain of thought monitorability: A new and fragile opportunity for AI safety (2025)

14. McKenzie, A., Pawar, U., Blandfort, P., Bankes, W., Krueger, D., Lubana, E.S., Krasheninnikov, D.: Detecting high-stakes interactions with activation probes (2025)
15. Mills, E., Su, S., Russell, S., Emmons, S.: ALMANACS: A simulatability benchmark for language model explainability (2025)
16. Paulo, G., Mallen, A., Juang, C., Belrose, N.: Automatically interpreting millions of features in large language models (2025)
17. Sakaguchi, K., Le Bras, R., Bhagavatula, C., Choi, Y.: WinoGrande: An adversarial winograd schema challenge at scale. In: Proceedings of the AAAI Conference on Artificial Intelligence (2020)
18. Sap, M., Rashkin, H., Chen, D., Le Bras, R., Choi, Y.: Social IQa: Commonsense reasoning about social interactions. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing. pp. 4463–4473. Association for Computational Linguistics (2019). https://doi.org/10.18653/v1/D19-1454
19. Wang, K., Variengien, A., Conmy, A., Shlegeris, B., Steinhardt, J.: Interpretability in the wild: a circuit for indirect object identification in GPT-2 small (2022)