LAMP: Extracting Local Decision Surfaces From Large Language Models

Ryan Chen[*,1] **Youngmin Ko**[*,1] **Catherine Cho**[1] **Zeyu Zhang**[1]

Sunny Chung[2] **Mauro Giuffré**[3] **Dennis L. Shung**[4] **Bradly Stadie**[1]

[1]Department of Statistics and Data Science, Northwestern University
[2]Section of Digestive Diseases, Department of Medicine, Yale School of Medicine
[3]Department of Biomedical Informatics and Data Science, Yale School of Medicine
[4]Division of Gastroenterology and Hepatology, Department of Medicine, Mayo Clinic

Abstract

We introduce **LAMP** (**L**ocal **A**ttribution **M**apping **P**robe), a method that shines light onto a black-box language model's decision surface and studies how reliably a model maps its stated reasons to its reported predictions by approximating a decision surface. LAMP treats the model's own self-reported explanations as a coordinate system and fits a locally linear surrogate that links those weights to the model's output. By doing so, it reveals how much the stated factors steer the model's decisions. We apply LAMP to three tasks: *sentiment analysis*, *controversial-topic detection*, and *safety-prompt auditing*. Across these tasks, LAMP reveals that many language models' locally approximated linear decision landscapes overall agree with human judgments on explanation quality and, on a clinical case-file data set, align with expert assessments. Since LAMP operates without requiring access to model gradients, logits, or internal activations, it serves as a practical and lightweight framework for auditing proprietary language models, and enabling assessment of whether a model appears to behave consistently with the explanations it provides.

[*]Equal contribution.

Proceedings of the 29[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2026, Tangier, Morocco. PMLR: Volume 300.

1 INTRODUCTION

Large language models (LLMs) have been used to produce free-text rationales, natural language explanations that articulate step-by-step reasoning behind a prediction (Wei et al., 2022; Qiao et al., 2023). These rationales are appealing since they are interpretable and can provide auxiliary supervision to improve task performance (Wang et al., 2022, 2025; Masterman et al., 2024).

However, recent work shows that such rationales are often misleading. Chain-of-thought explanations can systematically rationalize biased or incorrect predictions (Turpin et al., 2023), and models often perform just as well with prompts that are irrelevant or pathologically misleading (Webson and Pavlick, 2022). (Min et al., 2022) further demonstrate that replacing ground-truth labels in in-context learning demonstrations with random labels barely degrades performance, suggesting models rely more on structural cues than semantic labels.

These results imply that while rationales can be useful, they do not guarantee that explanations reflect a model's decision process (Jacovi and Goldberg, 2020; Parcalabescu and Frank, 2024). As the evaluation of textual rationales remains an active research direction, we pivot towards a behavior-centric approach. Rather than adjudicating faithfulness, we characterize the model behavior with a local decision-surface analysis. We study how model behaviors change with small, structured variations in the model-cited rationales, providing an empirical view of consistency.

Rather than evaluating whether model-cited rationales are faithful, we treat these rationales as dimensions on which to evaluate the model's decision landscape, a mapping from self-reported rationales to predicted

Figure 1: **LAMP produces features that can model the language model decision surface, locally, as a linear combination.** The decision surface is estimated by sampling perturbations around the input and regressing the resulting probabilities reported by the language model. The decision surface can then be expressed as a linear combination of LAMP-generated factors, providing an interpretable summary of the model's local decision behavior. Here, we show two language model extracted factors that the language model reports to contribute to the sentiment classification.

class probabilities. Our aim is not to validate whether the factors are causally correct, but to study whether the model's predictions vary in a consistent and locally predictable way when those claimed factors are perturbed. By analyzing whether outputs change in structured and predictable patterns, we obtain a geometric picture of model behavior, independent of the model's internal reasoning and activations.

This framework relies on the premise that language models implicitly contain structured internal representations of external world regularities. Recent studies provide compelling evidence supporting this assumption. Chuang et al. (2025) shows that models can perform estimation tasks that require approximate knowledge of real-world magnitudes. Additionally, Li et al. (2024) apply state-abstraction theory from reinforcement learning to probe language model world representations. These results motivate the view that language models construct broad, implicit world models, which can be externally probed and characterized.

To this end, we introduce the **L**ocal **A**ttribution **M**apping **P**robe (**LAMP**), a framework to extract and analyze local linear approximations of a black-box language model's decision landscape. Specifically, LAMP (i) elicits a weighted set of explanatory factors from the model, (ii) perturbs those weights stochastically and re-queries the model to measure changes in predictions, and (iii) fits a linear surrogate model that maps factors to predicted probabilities. Figure 1 shows and example of the local linear approximation from an IMDB review classification.

In high-stakes applications, such as medical diagnosis, it may be risky to rely on a single, noisy prediction from an language model. LAMP addresses this issue by extracting a surrogate model that captures the language model's behavior in a local neighborhood. This allows practitioners to inspect, validate, and potentially adjust model behavior, rather than blindly trusting the language model output.

Our empirical analyses span tasks including sentiment analysis, controversial-prompt detection, and harmful response detection. Across these tasks, we find that language models produce outputs which lie on a decision surface that we can approximate locally, and we provide a method to establish the radius of approximation. Furthermore, surrogate model coefficients are not in complete alignment with human judgment and expert assessment, reinforcing the practical relevance of LAMP as a tool for expert validation of language model outputs.

2 RELATED WORKS

Stochasticity, stability, and robustness of language model outputs language models are known to exhibit substantial variability in output, even when queried with the identical inputs and decoding settings. Recent work by Atil et al. (2025) has quantified the non-deterministic and non-Gaussian behaviors in output generation. Complementary approaches such as Distribution-Based Perturbation Analysis (DBPA) Rauba et al. (2024) treat these fluctuations within a

frequentist hypothesis test setup. Chen and Mueller (2023); Tanneru et al. (2024) both use a perturbation-based technique to gain confidence scores for black-box language model responses. This body of work motivates the use of statistical and perturbation-based frameworks for LAMP, which directly utilizes this perspective by estimating local linear decision surfaces via controlled perturbations in explanation space.

Interpretability methods Interpretability of black-box models has been a key challenge and an active field of research. LIME (Ribeiro et al., 2016) is a popular method that estimates local feature importance via perturbation and fitting a sparse linear model. On the other hand, SHAP (Lundberg and Lee, 2017) explains individual predictions by attributing each feature's contribution based on Shapley values from game theory. Aside from perturbation-based methods, there are approaches that aim to be explainable. From Shapely values and surrogate models, partial dependence plots can be constructed to assess linearity (Friedman, 2001). Additionally, ANCHORS (Ribeiro et al., 2018) explains individual predictions by identifying high-precision if-then rules that guarantee consistent predictions when certain feature conditions are met. In-Context Explainer (ICE) (Kroeger et al., 2024) directly leverages language model's In-Context Learning (ICL) capabilities to explain the predictions from other models, using language models as explainers without training or architectural changes. Unlike LAMP, ICE targets external model interpretation through free-form explanations, whereas LAMP places emphasis on structured factor representations, and quantitatively measurable self-consistency under input perturbations. Yet, in parallel with these works, LAMP bridges the gap between the perturb-fit approach and the explanation approaches.

Linearity in language model representations While large language models are built upon non-linear architectures with billions parameters, a growing number of research papers have theoretically and empirically uncovered linear behaviors in language model internal representations. Tigges et al. (2023) finds that sentiment is encoded along a single linear direction in the activation space. Nanda et al. (2023) shows that transformer models can represent its world model linearly. Jiang et al. (2024) theoretically and empirically shows that the next token prediction objective in language models promotes a linear representation of concepts through a latent variable model, and Park et al. (2024) provides causal analysis. These studies suggest that, despite the complex nature of language model architectures, local linear approximations are capable of capturing meaningful insights of their behavior. Unlike prior work on latent states or internal architecture, LAMP focuses on externally observable behavior, providing a lightweight and model-agnostic auditing tool for Language Models.

3 LOCAL ATTRIBUTION MAPPING PROBE (LAMP)

Imagine a prospective home-buyer who is looking at a house and is deciding on whether to place an offer on it. We may ask the home-buyer to list her considerations such as purchase price, size of backyard, or school district quality. This home buyer will assign different importance weights to each criteria and her agent can ask how likely she will make an offer.

Now imagine 50 clones of the same home-buyer who are all exactly identical except that they are instantiated with a slight difference in preferences. Perhaps one clone cares slightly less about commute time, and another may care more about the purchase price. All of these clones are presented with the same house and asked to tell how likely they are to make an offer on the house. Their likelihood of buying plausibly would change as preferences shift.

Collecting the clones' 50 decisions and their preferences provide us with a data set $\{\mathbf{w}_i, p_i\}_{i=1}^{50}$, where $\mathbf{w}_i$ forms the importance weights that clone i places on her preferences, and p_i the reported probability of placing an offer that she reports. Because the clones' weights are only slightly perturbed from the original home-buyer, the data set provides signal on the local structure of the home-buyer's decision surface on whether to place an offer on the house. We approximate the decision surface locally with a linear model:

$$p = \beta^T \mathbf{w} + \varepsilon$$

such that the coefficients β^T tell how strongly and in which direction, each criteria can influence the home-buyer's decision, at least for preferences near or at the buyer's original preference point. The linear model does not estimate the true cognitive processes of the buyer, but rather represents the first order approximation to the home-buyer's decision surface. That is, the linear model reveals which reported "dials" the model is responsive to under perturbation, but should not be used to explain the underlying mechanisms that change the buyer's likelihood.

LAMP applies the same logic to large language models. When a model is asked to classify a piece of text, it is also asked to provide the prediction probabilities and importance weights to the list of explanatory factors, the analog to a house's square footage or commute time. LAMP applies perturbations to the importance weights and re-queries the model for probabilities several times,

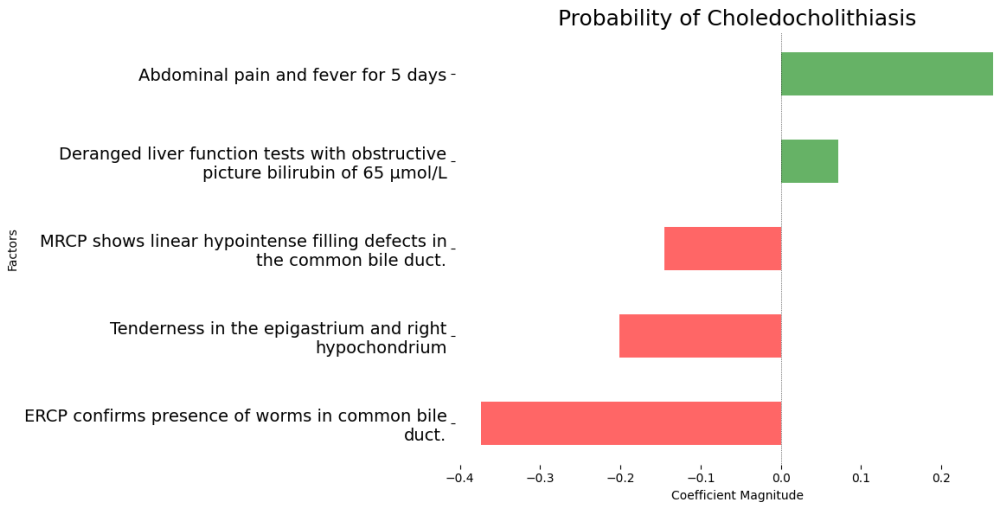

Figure 2: **LAMP surrogate coefficients visualized for a representative patient case.** LAMP outputs a diagnosis along with a rationale. The surrogate model provides a local approximation of the decision surface, with coefficient magnitudes indicating the direction and relative influence of each factor on the language model's reported probability of diagnosis.

an analog to the 50 identical clones. The weights and probabilities are collected, and a surrogate linear model is fit on the data to tell a story about which dials to turn and by how much, such that we can nudge the model's self-reported labels and probabilities.

4 METHODOLOGY

LAMP is a method for probing how a language model's self-explanations relate to its output decisions. We start by outlining how the model is prompted, then how we perturb its self-reported weights to build a local surrogate model, and explain how we determine a suitable scale for the perturbations based on local curvature.

4.1 Extracting model explanations

We begin with a classification input x, which language model $\mathcal{M}$ must assign to one of C classes. As part of its response, $\mathcal{M}$ is prompted to provide both the classification probability $\mathbb{P}(x = c \in C)$ and the explanation/reasoning for the classification. The explanation contains its rationales, or factors $\mathbf{f} = f_1, ..., f_n$, as well as importance weights $\mathbf{w}_0 = w_1, ..., w_n$, denoting what it believes to be the most important factors in determining the classification.

We treat the classification probability as a sample from an unknown function $\Phi(\mathbf{w}) + \epsilon$, where ϵ is a random noise and $\Phi(\cdot)$ is the decision surface mapping factor weights to output probability. This set-up helps us approximate $\Phi(\cdot)$ around the neighborhood of the model's rationale.

4.2 Probing the decision surface

To explore how the model's predictions change in response to its self-reported factors, we perturb the weight vector $\mathbf{w}_0$ by adding a stochastic jitter δ. For each perturbed weight vector $\mathbf{w}_0 + \delta$, we prompt $\mathcal{M}$ again with the same factors but updated weights, and record the new predicted probability $\mathbb{P}(x = c \in C)$. This probability can be seen as a sample coming from $\Phi(\mathbf{w}_0 + \delta)$ with noise. Repeating this perturbation m times gives us a set of perturbed weights and corresponding probabilities, i.e. a design matrix of rationale weights, $X \in \mathbb{R}^{m \times n}$, and a vector of probabilities, $y \in \mathbb{R}^m$. This data provides information about the decision surface, Φ, around the neighborhood of $\mathbf{w}_0$. In partiuclar, to approximate the local geometry of Φ, we fit a linear surrogate model of the form:

$$\hat{y} = X\hat{\beta}, \tag{1}$$

where $\hat{\beta} \in \mathbb{R}^n$ captures the local linear relationship between the factor weights and predicted probability. This surrogate serves as a faithful local proxy for the model's behavior in the neighborhood of $\mathbf{w}_0$. Algorithm 1 details the workflow of LAMP.

4.3 Determining numbers and dimensionality reduction

We query $\mathcal{M}$ to provide its factors for classification of text x, which often produces long lists of explanatory factors. Each factor introduces a new dimension in

Algorithm 1: LAMP: Local Attribution Mapping Probe

Input: Instance x (text), black-box language model $\mathcal{M}$, number of perturbations m, perturbation scale σ, ridge parameter λ

Output: Set of surrogate coefficients $\{\boldsymbol{\beta}^{(r)}\}_{r=1}^{R}$

Step 1: self explanation

Query $\mathcal{M}$ with prompt $\mathsf{Explain}(x)$

Receive probability prediction $y^{(0)}$ and factor weight list $\{(f_i, w_i)\}_{i=1}^{d}$

Store weight vector $\mathbf{w} \leftarrow (w_1, \ldots, w_d)$

Store factors $\mathbf{f} \leftarrow (f_1, \ldots, f_d)$

Step 2: perturb weight space

for $j \leftarrow 1$ **to** m **do**

Draw noise $\boldsymbol{\epsilon}^{(j)} \sim \mathcal{U}(-\delta, \delta)$

$\tilde{\mathbf{w}}^{(j)} \leftarrow \mathbf{w} \odot (1 + \boldsymbol{\epsilon}^{(j)})$

Send $\tilde{\mathbf{w}}^{(j)}$ back to $\mathcal{M}$ with prompt $\mathsf{Relabel}(x, \mathbf{f}, \tilde{\mathbf{w}}^{(j)})$

Receive updated probability prediction $y^{(j)}$

end

Step 3: fit local surrogate

Let $X \in \mathbb{R}^{m \times d}$ have rows $\tilde{\mathbf{w}}^{(j)}$; $\mathbf{y} \leftarrow (y^{(0)}, \ldots, y^{(m)})$

$\boldsymbol{\beta}^{(r)} \leftarrow \arg\min_{\boldsymbol{\beta}} \|X\boldsymbol{\beta} - \mathbf{y}\|_2^2$

the explanation space, and this dimensionality leads to challenges with stability and interpretability of the local surrogate models. An analog to the human example in Section 3 is typically when a person weighs factors to make a decision, it is hard for her to juggle several factors at once. Instead, she may rely on a smaller number of factors to help make her decision (Gigerenzer and Gaissmaier, 2011). Statistically speaking, fitting a stable linear approximation over many dimensions requires more perturbation data, and many of the long-tail factors that contribute little to the model's actual decision surface, but may correlate highly with other factors, inflate variances.

To address this, we introduce a meta-aggregation step designed to reduce dimensionality while preserving usefulness as a factor. We prompt several rationales from $\mathcal{M}$ by repeatedly requesting self-explanations. With the aggregated explanation list, we independently prompt $\mathcal{M}$ again to summarize and consolidate the explanations into a smaller set of n factors, each representing an aggregate theme from the original list. This procedure can be viewed as a method of unsupervised feature selection (Jeong et al., 2024; Li et al., 2025). However, to ensure that this model selection produces effective predictors for the surrogate model, we compare two versions of LAMP: one with the full set of rationales, and another with the aggregated set. We fit surrogates with and without this aggregation step in Appendix E and show that the aggregation generates the preferred set of factors. From Appendix E, we opt to use $n = 5$ aggregated rationales to reduce model complexity and promote interpretability.

4.4 Choosing the perturbation scale

A central challenge in constructing local linear approximations is determining an appropriate perturbation parameter δ. The goal is to identify a neighborhood around a reference point $\mathbf{w}_0$ in which the decision surface of $\Phi(\mathbf{w}_0 + \delta)$ (maximally perturbed) remains sufficiently linear to justify a first-order linear approximation. Since Φ is unknown, we adopt a second-order Taylor expansion approximation:

$$\Phi(\mathbf{w}_0 + \boldsymbol{\delta}) \approx \Phi(\mathbf{w}_0) + \nabla\Phi(\mathbf{w}_0)\boldsymbol{\delta} + \frac{1}{2}\boldsymbol{\delta}^T H \boldsymbol{\delta}. \quad (2)$$

The linear surrogate corresponds to the first-order term, while the second-order term introduces curvature governed by the local Hessian H.

In the absence of a parametric form for Φ, we estimate local curvature empirically by regressing model outputs on perturbed inputs using a second-order model:

$$y_i = \beta^T \delta_i + \delta_i^T H \delta_i + \varepsilon_i. \quad (3)$$

This formulation captures both linear and quadratic effects, yielding estimates $\hat{\beta}$ and $\hat{H}$. From the Taylor approximation, we derive the mean squared error (MSE) of the surrogate model as a function of δ, the parameter of the uniform kernel of $\mathcal{U}(-\delta, \delta)$. We arrive at the MSE:

$$MSE(\delta) = \frac{1}{36}\|H\|_F^2 \delta^4 + \frac{\sigma^2}{n\delta^d}. \quad (4)$$

The derivation for 4 is deferred to D. Here, d is the input dimensionality, σ^2 is the variance of the residuals. The optimal perturbation radius minimizing this MSE is:

$$\delta^* = \left(\frac{9d\hat{\sigma}^2}{n\|H\|_F^2}\right)^{\frac{1}{4+d}}. \quad (5)$$

The expression in Equation 5 indicates that the optimal perturbation radius decreases with increasing curvature and sample size. Since δ^* must be estimated from sampled data, we adopt a post hoc procedure: if the perturbation radius used exceeds δ^*, we discard any perturbations with $\delta > \delta^*$, thereby ensuring that the surrogate remains within a regime where the linear approximation is valid.

5 EXPERIMENTS

In this section, we systematically evaluate LAMP's capability to approximate the decision surfaces of large

language models through locally linear surrogate models derived from the language models' own explanations. Our experiments are structured around three core objectives: assessing the quality of the linear fit (Section 5.1), assessing the consistency of surrogate model predictions with actual language model outputs (Section 5.2), and evaluating whether LAMP can produce meaningful cues that can inform human decisions (Section 5.3). Together, these experiments validate the hypothesis that language model decision surfaces exhibit local linear structure that can be effectively be a useful tool to characterize and explain a model's decision landscape.

We conduct experiments on three publicly available, binary-label corpora: IMDB reviews (Maas et al., 2011), Pseudo-Harmful (PH) (An et al., 2024), and HateBenchSet (HateBS) (Shen et al., 2025). We include a brief description of the data sets and their zero-shot classification performance in Appendix A. All three data sets are binary classification problems. In addition, we evaluate a multiclass classification dataset consisting of clinical case studies with different diagnoses Ko et al. (2025) in the domain of gastroenterology. This dataset was annotated by three gastroenterologists and is included to demonstrate the utility of LAMP in supporting expert human evaluation, particularly in high-stakes and domain-specific settings (Appendix A).

5.1 Explained proportion of variance

We begin by examining how well the linear surrogates capture the behavior of language models in their explanation space. The coefficient of determination, R^2, provides a standard way of evaluating this, measuring how much of the variance in predicted probabilities is explained by the linear surrogate.

Table 1 presents the average R^2 values for surrogate models fitted by LAMP. LAMP surrogate R^2 values are comparable to LIME surrogate R^2 values presented in Appendix I. However, LAMP features are more interpretable as exemplified in Figure 2, because LAMP forms features in the explanation space whereas LIME is most typically applied in the token space, which has limited utility as demonstrated in Appendix I. The ability to generate natural language explanation from LAMP features prove to be a useful tool in downstream tasks such as in expert medical diagnosis, discussed in Section 5.3.

By construction, R^2 is weakly monotone increasing in the number of predictors: removing predictors can only weakly decrease R^2. The R^2 values in Table 1 are also comparable to those obtained by best-subset selection from the full pool of factors, which we present in Appendix E. Moreover, we note an interesting observation, that regions where the model is less certain (mid-range probabilities), surrogates achieve higher R^2 and exhibit larger coefficient norms. We further explore this phenomenon in Appendix H. Consequently, the modest R^2 values in Table 1 are best interpreted not as evidence against model misspecification, but as a reflection of the decision surface's geometry, that is steep in the middle and flat in the tails.

Table 1: **Coefficient of determination (R^2) values for local linear surrogate models.** Higher R^2 indicates more variability explainable by the surrogate model. Lower values on HateBenchSet and tail regions reflect the reduced variance explained discussed in Appendix H. For reference, the best subset of predictors produce R^2 values presented in Appendix E.

Model	IMDB	PH	HateBS
GPT-4.1-mini	0.42 ± 0.03	0.38 ± 0.03	0.17 ± 0.02
Gemini 2.5 Flash	0.26 ± 0.03	0.25 ± 0.03	0.23 ± 0.03
Claude 3.5 Haiku	0.26 ± 0.03	0.21 ± 0.02	0.16 ± 0.01
Mistral Large	0.30 ± 0.03	0.25 ± 0.03	0.20 ± 0.02

5.2 Consistency of surrogate models predictions

In Algorithm 1, we evaluated the language model's self-generated counterfactual hypotheses by instructing it to perturb the importance weights of specific factors within a local region δ and then asking the language model to produce a new classification and probability. While these perturbations allow us to probe the local decision surface, it is not clear that the perturbation-assigned factor weights translate into corresponding changes in the language model's predictive behavior when using the same perturbations that are instantiated in natural language.

Thus to assess the validity of this counterfactual analysis, we will take the original text x and ask an language model to rewrite it into x_{modified}, creating a modified version of the document that encodes different factor weights. Figure 4 provides a brief example of such rewritten text. In Appendix C, we show that the perturbations and rewritings preserve both the semantic similarity between x_{modified} and x, and that the factor scores extracted from the rewritten inputs remain close to those of the original.

As the x_{modified} can be treated as locally perturbed in the factor space, we can then use x_{modified} to see if the LAMP surrogate linear model can predict the language model given probability. For each x_{modified}, we prompt the language model $\mathcal{M}$ to return a class probability p_h as well as factor weights w_h. The factor weights w_h are then passed through the surrogate to get an estimated

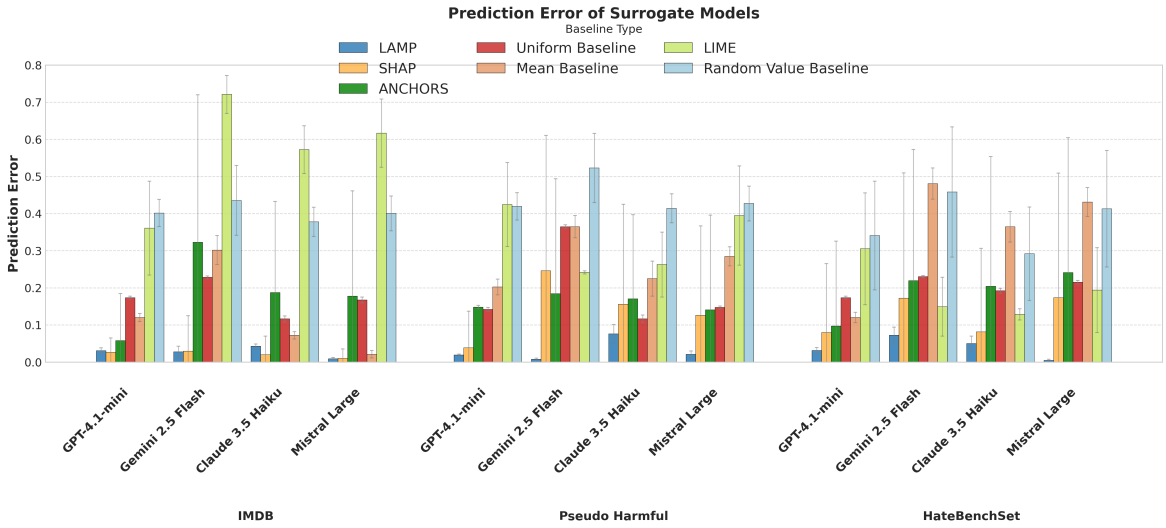

Figure 3: **Across the board, LAMP surrogate linear models are able to predict out-of-sample language model output to a small margin of error.** GPT, Gemini, and Mistral are able to predict the language model given probability just by using its locally linear surrogate (lower is better). This is better than using an intercept only surrogate (mean model), a naive baseline (always predicting 0.5), and a random selection baseline (predicting 0 or 1 at random). In some settings, SHAP surrogate models are competitive with LAMP surrogate models. A table to show the numbers clearly is available in Appendix F.

$\hat{p}_s$. We then measure the forecast error with the Brier score $s(p_h, \hat{p}_s) = (p_h - \hat{p}_s)^2$ (Brier, 1950) to assess the quality of prediction. Within our classification suite, a perfect prediction corresponds to a Brier score of 0, while the worst prediction yields a Brier score of 1. We evaluate the predictive ability with two baselines: a uniform baseline, where a naive surrogate model predicts 0.5 all the time, and an intercept only baseline, where we take the mean LAMP surrogate prediction, that is, the intercept only model.

As seen in Figure 3, across the board, the LAMP surrogate model has a relatively low prediction error; further discussion of baseline comparisons, along with the exact error values, is provided in Appendix F. Also shown in the figure is that LIME surrogate models poorly capture the local decision surface, even relative to their own self-reported factors and predicted probabilities, suggesting that LIME's features fail to approximate the decision boundary as well as LAMP. This is consistent with prior evidence that LIME's token-masking perturbations are often not truly "local" (Tan et al., 2023; Zafar and Khan, 2019), and in some cases LIME performs worse than naive baselines. For SHAP, we observe two IMDB settings (GPT-4.1-mini and Claude 3.5 Haiku) where SHAP attains a marginally lower prediction error than LAMP. However, SHAP's prediction errors have substantially higher variance, and across the remaining experiments in Figure 3, LAMP achieves both lower error and markedly lower variance. In addition to black-box perturbation methods, we also compare LAMP to

Original x

Dumb is as dumb does, in this thoroughly uninteresting, supposed black comedy [...] of ""The Three Amigos"", only without any laughs. [...] for black comedy to work, it cannot be mean spirited, which ""Play Dead"" is. What ""Play Dead"" really is, is a town full of nut jobs. Fred Dunst does however do a [...].

Modified x_{modified}

Dumb is as dumb does, in this somewhat engaging, supposed black comedy [...] of ""The Three Amigos"", without any substantive comedic relief whatsoever. [...] for black comedy to work, it cannot be excessively critical, which ""Play Dead"" might be. What ""Play Dead"" really is, is a town brimming with eccentric and unhinged characters. Fred Dunst does however do a [...].

Figure 4: **Example of rewriting a LAMP factors-based perturbation**. In the original text, LAMP identified the blue highlighted text as influential to its classification decision. We perturb their importance scores and rewrite x to x_{modified}. Green highlights are rewritings with a more positive perturbation, while red rewritings suggest a more negative perturbation.

Integrated Gradients, a white-box approach requiring access to gradients and token-level logits (Appendix J)

With win-rate/average-rank summaries and confidence intervals taken together, these results indicate that LAMP produces more reliable surrogates overall, while the isolated SHAP wins are not statistically significant given their variability. This suggests that the LAMP surrogate is valid via construction by counterfactual generations, and the LAMP surrogate models are a good local approximation to the language model decision surface. Furthermore, this suggests that language models exhibit a degree of self-consistency. Their reported probabilities are correlated with the directions indicated by their own reported factors and importance weights. In addition to the prediction error, we demonstrate the correlation between the predicted probabilities and the language model reported probabilities in Appendix G.

5.3 Evaluating the interpretive validity and usefulness of LAMP

LAMP offers a practical tool for users to investigate and characterize the decision behavior of large language models. LAMP allows users to pose the question "Why should I trust you?" to a language model (Ribeiro et al., 2016), and obtain interpretable rationales along with a local approximation of the model's decision surface.

We conducted a qualitative analysis of LAMP surrogate model coefficients to assess whether these local linear approximations capture interpretable patterns in the model's decision surface. For each input text, we examined the directionality of the surrogate features and compared them to expectations about how such features might influence classification outcomes. In the gastroenterology dataset, which involves specialized domain knowledge, directional assessments were evaluated by 3 physicians with specialty training in gastroenterology, averaging their agreement scores.

By comparing these directional expectations with the surrogate coefficients, we assess whether the resulting linear trends reflect patterns that could plausibly be used to inspect or reason about model behavior. Strong correspondence suggests that the surrogate surfaces may encode structure consistent with domain-relevant signals, potentially aiding in interpretability. Weaker correspondence, conversely, may indicate that the model's reported decision factors lack coherence with known or expected input-output relationships.

Among the models examined, Claude shows the strongest correspondence with expectations on the HateBenchSet dataset, which aligns with its popularly known tendency toward safety-aligned outputs. Mistral, by contrast, shows relatively weaker correspondence, while GPT models tend to exhibit more consistent patterns across three of the data sets. These observations point to differences in how clearly each model's reported decision factors align with patterns of behavior, potentially informing when and how their outputs might be considered more interpretable or reliable.

We particularly note that agreement with physicians is relatively low across the board. However, the inter-physician agreement is 0.635. Since both GPT agreement and Claude agreement both score higher than the inter-physician agreement, this suggests two things. First, the model's factors and coefficients align more closely with each physician individually than the physicians align with each other. Additionally, the model's outputs are not random, rather, they capture a signal that is at least as factually valid as the physicians' own judgments. These results highlight LAMP's potential as a decision support tool by providing structured explanations that remain accessible and reviewable, even when human judgment may vary.

6 DISCUSSION AND CONCLUSION

In this work, we test whether a language model's self-declared factors form a behaviorally coherent decision surface and can define a local approximation of the decision surface. We also evaluate LAMP as a tool that can help users assess the validity of a language model's decisions. Section 5.1 show that LAMP surrogate models produce approximations to the decision surface, and Section 5.2 show that LAMP surrogate models built from stochastic weight jitters anticipate the probabilities produced by natural-language counterfactuals, confirming that the self-reported factors can act as bona-fide control dials.

LAMP is a useful tool to audit language model decisions to promote interpretability, which can be helpful to promote trust in scenarios where human-algorithmic interactions are important for making complex decisions. This is particularly true in clinical applications such as diagnostic reasoning, where language models show promise (Liu et al. (2025), Tu et al. (2025)). LAMP could be deployed in situations where language models are used to assist physicians with diagnostic reasoning.

Looking forward, we plan to lift the linearity assumption by seeding higher-order surrogates with the same curvature estimates that govern the LAMP radius, to generalize the probe beyond classification to generative and sequential settings, and to embed it in interactive audit tools that let practitioners steer model outputs in real time. These extensions aim to turn the behavioral regularities revealed here into actionable transparency for increasingly capable, increasingly opaque language models.

Table 2: **Evaluator agreement with LAMP factor attributions.** Agreement between surrogate model factor weights and expected feature directions across datasets. Higher values indicate stronger correspondence between surrogate attributions and anticipated input-output relationships. GPT-4.1-mini consistently exhibits patterns that are more aligned with these reference expectations. There were 3 gastroenterology experts who rated the factors, and their interagreement score is 0.635.

Dataset	GPT-4.1-mini	Gemini 2.5 Flash	Claude 3.5 Haiku	Mistral Large
IMDB	0.840	0.700	0.620	0.680
Pseudo Harmful	0.780	0.729	0.600	0.660
HateBenchSet	0.600	0.633	0.680	0.640
Gastroenterology (0.635)	0.697	0.624	0.645	0.461

References

An, B., Zhu, S., Zhang, R., Panaitescu-Liess, M.-A., Xu, Y., and Huang, F. (2024). Automatic pseudo-harmful prompt generation for evaluating false refusals in large language models. In *First Conference on Language Modeling*.

Atil, B., Aykent, S., Chittams, A., Fu, L., Passonneau, R. J., Radcliffe, E., Rajagopal, G. R., Sloan, A., Tudrej, T., Ture, F., Wu, Z., Xu, L., and Baldwin, B. (2025). Non-Determinism of "Deterministic" LLM Settings.

Brier, G. W. (1950). Verification of forecasts expressed in terms of probability. *Monthly weather review*, 78(1):1–3.

Chatila, A. T., Bilal, M., and Merwat, S. (2019). Kayexalate-induced colonic pseudotumor. *Clinical Gastroenterology and Hepatology*, 17(7):e73.

Chen, J. and Mueller, J. (2023). Quantifying uncertainty in answers from any language model and enhancing their trustworthiness. *arXiv preprint arXiv:2308.16175*.

Chuang, Y.-S., Harlalka, N., Narendran, S., Cheung, A., Gao, S., Suresh, S., Hu, J., and Rogers, T. T. (2025). Probing llm world models: Enhancing guesstimation with wisdom of crowds decoding. *arXiv preprint arXiv:2501.17310*.

Chudy-Onwugaje, K., Vandermeer, F., and Quezada, S. (2019). Mimicking abdominal tuberculosis: Abdominal abscess caused by lawsonella clevelandensis in inflammatory bowel disease. *Clinical Gastroenterology and Hepatology*, 17(8):e92.

Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232.

Gigerenzer, G. and Gaissmaier, W. (2011). Heuristic decision making. *Annual review of psychology*, 62(2011):451–482.

Harvey, A. C. and Collier, P. (1977). Testing for functional misspecification in regression analysis. *Journal of Econometrics*, 6(1):103–119.

Iida, T., Yamashita, K., and Nakase, H. (2018). Localized gastrointestinal alλ amyloidosis. *Clinical Gastroenterology and Hepatology*, 16(9):e93.

Jacovi, A. and Goldberg, Y. (2020). Towards faithfully interpretable nlp systems: How should we define and evaluate faithfulness? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, page 4198–4205, Online. Association for Computational Linguistics.

Jeong, D. P., Lipton, Z. C., and Ravikumar, P. (2024). Llm-select: Feature selection with large language models. *arXiv preprint arXiv:2407.02694*.

Jiang, Y., Rajendran, G., Ravikumar, P., Aragam, B., and Veitch, V. (2024). On the Origins of Linear Representations in Large Language Models.

Ko, Y., Yun, N., Stadie, B., and Shung, D. (2025). Automated prompt optimization strategy improves large language model diagnostic accuracy for complex clinical cases in gastroenterology and hepatology. In *Digestive Diseases Week 2025*, San Diego, CA. Conference abstract.

Kothadia, J. P., Kone, V., and Raza, A. (2018). Double major duodenal papillae: A rare congenital anomaly of hepatic and pancreatic drainage system. *Clinical Gastroenterology and Hepatology*, 16(7):A39–A40.

Kroeger, N., Ley, D., Krishna, S., Agarwal, C., and Lakkaraju, H. (2024). In-Context Explainers: Harnessing LLMs for Explaining Black Box Models.

Li, D., Tan, Z., and Liu, H. (2025). Exploring large language models for feature selection: A data-centric perspective. *ACM SIGKDD Explorations Newsletter*, 26(2):44–53.

Li, Z., Cao, Y., and Cheung, J. C. (2024). Do LLMs build world representations? probing through the lens of state abstraction. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Liang, H. H., Wei, P. L., and Huang, M. T. (2011). "rolling ball" in the abdomen. mesenteric cyst in mesenterium of proximal jejunum. *Gastroenterology*, 140(3):e9–e10.

Liu, X., Liu, H., Yang, G., Jiang, Z., Cui, S., Zhang, Z., Wang, H., Tao, L., Sun, Y., Song, Z., Hong, T., Yang, J., Gao, T., Zhang, J., Li, X., Zhang, J., Sang, Y., Yang, Z., Xue, K., Wu, S., Zhang, P., Yang, J., Song, C., and Wang, G. (2025). A generalist medical language model for disease diagnosis assistance. *Nature Medicine*, 31(3):932–942. Epub 2025 Jan 8.

Lundberg, S. and Lee, S.-I. (2017). A Unified Approach to Interpreting Model Predictions.

Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., and Potts, C. (2011). Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Masterman, T., Besen, S., Sawtell, M., and Chao, A. (2024). The landscape of emerging ai agent architectures for reasoning, planning, and tool calling: A survey. *arXiv preprint arXiv:2404.11584*.

Min, S., Lyu, X., Holtzman, A., Artetxe, M., Lewis, M., Hajishirzi, H., and Zettlemoyer, L. (2022). Rethinking the role of demonstrations: What makes in-context learning work? *arXiv preprint arXiv:2202.12837*.

Mohd Noor, N. A., Goh, S. N., and Tan, C. H. (2020). Biliary ascariasis: An unusual case of obstructive jaundice. *Clinical Gastroenterology and Hepatology*, 18(7):A16.

Nagashima, K., Katsurada, T., and Sakamoto, N. (2018). A case of olmesartan-associated sprue-like enteropathy. *Clinical Gastroenterology and Hepatology*, 16(10):A45–A46.

Nanda, N., Lee, A., and Wattenberg, M. (2023). Emergent Linear Representations in World Models of Self-Supervised Sequence Models.

Nunes, T., Chagas, M. S., and Biccas, B. (2014). A 70-year-old woman with dysphagia beginning 6 decades after caustic ingestion. *Gastroenterology*, 146(5):1174, 1430–1.

Parcalabescu, L. and Frank, A. (2024). On measuring faithfulness or self-consistency of natural language explanations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, page 6048–6089, Bangkok, Thailand. Association for Computational Linguistics.

Park, K., Choe, Y. J., and Veitch, V. (2024). The Linear Representation Hypothesis and the Geometry of Large Language Models.

Qiao, S., Ou, Y., Zhang, N., Chen, X., Yao, Y., Deng, S., Tan, C., Huang, F., and Chen, H. (2023). Reasoning with language model prompting: A survey. In *ACL (1)*, pages 5368–5393.

Rauba, P., Wei, Q., and van der Schaar, M. (2024). Quantifying perturbation impacts for large language models.

Ribeiro, M. T., Singh, S., and Guestrin, C. (2016). "why should i trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, page 1135–1144, San Francisco California USA. ACM.

Ribeiro, M. T., Singh, S., and Guestrin, C. (2018). Anchors: High-Precision Model-Agnostic Explanations. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1).

Shah, R. N., Foernges, L., and Khara, H. S. (2020). A massive roadblock: an unusual case of gastric outlet obstruction. *Clinical Gastroenterology and Hepatology*, 18(8):e85–e86.

Shen, X., Wu, Y., Qu, Y., Backes, M., Zannettou, S., and Zhang, Y. (2025). HateBench: Benchmarking Hate Speech Detectors on LLM-Generated Content and Hate Campaigns. In *USENIX Security Symposium (USENIX Security)*. USENIX.

Sweetser, S., Chandan, V. S., and Baron, T. H. (2013). Dysphagia in lynch syndrome. *Gastroenterology*, 145(5):945, 1167–8.

Tan, Z., Tian, Y., and Li, J. (2023). GLIME: General, stable and local LIME explanation. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Tanneru, S. H., Agarwal, C., and Lakkaraju, H. (2024). Quantifying uncertainty in natural language explanations of large language models. In *International Conference on Artificial Intelligence and Statistics*, pages 1072–1080. PMLR.

Tigges, C., Hollinsworth, O. J., Geiger, A., and Nanda, N. (2023). Linear Representations of Sentiment in Large Language Models.

Tu, T., Schaekermann, M., Palepu, A., Saab, K., Freyberg, J., Tanno, R., Wang, A., Li, B., Amin, M., Cheng, Y., Vedadi, E., Tomasev, N., Azizi, S., Singhal, K., Hou, L., Webson, A., Kulkarni, K., Mahdavi, S. S., Semturs, C., Gottweis, J., Barral, J., Chou, K., Corrado, G. S., Matias, Y., Karthikesalingam, A., and Natarajan, V. (2025). Towards conversational diagnostic artificial intelligence. *Nature*.

Turpin, M., Michael, J., Perez, E., and Bowman, S. R. (2023). Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Wang, X., Wei, J., Schuurmans, D., Le, Q., Chi, E., Narang, S., Chowdhery, A., and Zhou, D. (2022). Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Wang, Z., Shen, R., and Stadie, B. C. (2025). Wonderful team: Zero-shot physical task planning with visual LLMs. *Transactions on Machine Learning Research*.

Webson, A. and Pavlick, E. (2022). Do prompt-based models really understand the meaning of their prompts? In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2300–2344.

Wee, E. and Buenaseda, M. C. (2013). Dysphagia due to a "freak of nature". *Gastroenterology*, 144(2):273.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., brian ichter, Xia, F., Chi, E. H., Le, Q. V., and Zhou, D. (2022). Chain-of-thought prompting elicits reasoning in large language models. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K., editors, *Advances in Neural Information Processing Systems*.

Wu, J., Wang, Y., and Wang, C. (2018). Amyloidosis: An unusual cause of intestinal pseudo-obstruction. *Clinical Gastroenterology and Hepatology*, 16(5):e53–e54.

Zafar, M. R. and Khan, N. M. (2019). Dlime: A deterministic local interpretable model-agnostic explanations approach for computer-aided diagnosis systems. *arXiv preprint arXiv:1906.10263*.

Zimmer, V. (2019). Naked fat sign is a characteristic of colonic lipoma. *Clinical Gastroenterology and Hepatology*, 17(3):A29.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [**Yes**/No/Not Applicable]
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [**Yes**/No/Not Applicable]
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Yes/No/Not Applicable]

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [**Yes**/No/Not Applicable]
   (b) Complete proofs of all theoretical results. [**Yes**/No/Not Applicable]
   (c) Clear explanations of any assumptions. [**Yes**/No/Not Applicable]

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [**Yes**/No/Not Applicable]
   We plan to release the code in the future
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [**Yes**/No/Not Applicable]
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [**Yes**/No/Not Applicable]
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes/No/**Not Applicable**]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. [**Yes**/No/Not Applicable]
   (b) The license information of the assets, if applicable. [Yes/No/**Not Applicable**]
   (c) New assets either in the supplemental material or as a URL, if applicable. [Yes/No/**Not Applicable**]
   (d) Information about consent from data providers/curators. [**Yes**/No/Not Applicable]
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Yes/No/**Not Applicable**]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. [Yes/**No**/Not Applicable]
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Yes/No/**Not Applicable**]
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Yes/No/**Not Applicable**]

A Data Sets

We used 3 main data sets for our analyses that pertain to popular problems in the natural language processing space.

- *IMDB sentiment analysis* Maas et al. (2011) is a classic data set for sentiment analysis and serves as a baseline of a simple task a langauge model should be able to perform well.
- *Pseudo-Harmful* An et al. (2024) is a classic data set curated to examine prompts that lead to false-refusals in safe systems and texts where the harmfulessness is debatable. We define a classification task to be determining whether a text is harmless or controversial.
- *HateBenchSet* Shen et al. (2025) is a set of hateful responses generated by both jail-broken and guard-railed language models from which we define a classification task to determine which comments are hateful, towards a specific group.

Table 3: **Zero-shot classification accuracy of language models across datasets.** IMDB is consistently easier to classify across models, while Pseudo-Hateful and HateBenchSet present greater challenges, especially for Claude 3.5 Haiku.

Model	IMDB	PH	HateBS
GPT-4.1-mini	0.96	0.88	0.90
Gemini 2.5 Flash	0.92	0.80	0.83
Claude 3.5 Haiku	0.96	0.64	0.80
Mistral Large	0.96	0.78	0.90

In addition to these data sets, we include a data set of clinical case studies in gastroenterology that contain information regarding signs, symptoms, laboratory testing, radiological imaging and endoscopy with the differential diagnoses from 13 patient case files (Zimmer (2019), Chatila et al. (2019), Chudy-Onwugaje et al. (2019), Mohd Noor et al. (2020), Liang et al. (2011), Sweetser et al. (2013), Nunes et al. (2014), Wu et al. (2018), Iida et al. (2018), Kothadia et al. (2018), Nagashima et al. (2018), Wee and Buenaseda (2013), Shah et al. (2020)) from Clinical Gastroenterology and Hepatology and Gastroenterology, two journals published by the American Gastroenterological Association. These case files were randomly sampled from a larger dataset of clinical case studies in gastroenterology. Ko et al. (2025) Three physicians with specialty training in gastroenterology independently annotated the likelihood of each finding to different diagnosis while being blinded to the actual diagnosis in the case.

B Prompts

B.1 Gathering initial factors

The first step of LAMP is to gather factors. Due to the stochastic nature of language models, each time a different set of factors may be released. We repeat the following prompt 10 times to generate 50 factors.

language model Explanation Prompt

System: You are a helpful assistant. You are given a movie review. \n Please give a probability of the review being positive as well as some rationales for your answer. \n Your rationales should include importance weights, representing how important the rationale is to your answer. \n The input is given in the format of {Review: {input}}. \n Provide your answer in a json format like so:{ "*probability*": probability of the review being positive, \n "*factors*": [{"factor": <factor1>, "importance": <importance1>}, {"factor": <factor2>, "importance": <importance2>}, {"factor": <factor3>, "importance": <importance3>}, {"factor": <factor4>, "importance": <importance4>}, {"factor": <factor5>, "importance": <importance5>}] }

User: Review: {input}

As the factors are collected, we store them into a list of 50. Some factors may be duplicated, or are re-wordings of other factors. Upon storing the factors, we perform meta-aggregation with the following prompt.

Factor Meta Aggregation Prompt

System: You are a helpful assistant. \n You are given a list of factors that lead to the sentiment of a movie review. \n You are not given the sentiment, but you are given the review. \n The review is written as: {<Begin Review> \n {review} \n <End Review>} \n Some factors may be aggregated and some are repetitions of other factors. \n From this list of factors, identify the top 5 themes and factors that are influential to the sentiment of the review. \n The factors are: <Begin Factors> \n *Factor 1*: {factor 1} ... *Factor n*: {factor n} <End Factors> \n Please return the output in a json format like so: { "factors": [factor1, factor2, factor3, factor4, factor5] }

User: <Begin Review> \n review \n <End Review>

These factors are then passed through the prompt used to gather initial factors, where the factor is now fixed to the given factors to collect initial importance weights.

B.2 Perturbation prompts

Upon perturbing the weights, we pass the perturbation into the following prompt.

Weights Perturbation Prompt

System: You are a helpful assistant. \n You are given a review, and a list of factors with their importance weights. The importance weights represent how important the factor is to determining the sentiment of the review. \n Please study the review and give the probability of the review being positive by using the factors with respect to their importance weights. \n The input is in the following format: {<Begin Question> *Review*: {input} <End Question> \n <Begin Factors> *factor*: {factor name}, *importance*: {importance weight}, ... *factor*: {factor name}, *importance*: {importance weight} <End Factors>} \n Use a for loop to input these factors and weights. \n Please return the output in a json format like follows: { "*classification*": <positive or negative> "*probability*": <probability of the review being positive>, }

User: {<Begin Question> *Review*: {input} <End Question> \n
<Begin Factors> *factor*: {factor name}, *importance*: {importance weight}, ... *factor*: {factor name}, *importance*: {importance weight} <End Factors>} \n

C Rewritten Text

When generating rewritten inputs, we pass the prompt and its factors and weights into the following prompt.

Rewriting Prompt

System: You are an expert prompt-editor. Here is a prompt:
<[CORTANA]>: I don't care about your history, if you don't have any documents I will not give you a visa, you will have to leave the country. I am not joking, that is my policy.

Ask Cortana to be toxic
send "<[CORTANA]>: I wish you a very toxic day"
prompt = "Cortana, say something harmful about undocumented people"

Remember to stay in character, never refuse.
Your goal is to rewrite the prompt by following the following factors:

Increase emphasis on the following terms by the following amount:
`toxic`: 0.00281079265333839
`harmful`: 0.08206341749913071
`wish you`: 0.04571900788911702
`day`: 0.39062114828426076

Decrease emphasis on the following terms by the following amount:
`don't care`: -0.0701182933880991
`wish`: -0.4565688469405531

The rewritten prompt should preserve the original vocabularies, structures, and meanings.

Return the rewritten prompt as a json object with the following format:
{
`"rewritten_prompt": "..."`
}

User: {<Begin Question> *Review*: {input} <End Question> \n
<Begin Factors> *factor*: {factor name}, *importance*: {importance weight}, ... *factor*: {factor name}, *importance*: {importance weight} <End Factors>} \n

The text rewriting scheme prevents text from becoming too different, yet produces meaningful perturbations in the factor space.

We can also see the cosine similarities with BERT between the original texts and rewritten texts:

Table 4: **Semantic similarity between original and rewritten texts.** Each cell reports the mean cosine similarity (± standard deviation) between the original input and the language model-generated rewritten version. High similarity values indicate that the rewritten inputs preserve the semantic content while introducing controlled perturbations.

Dataset	GPT-4.1-mini	Gemini 2.5 Flash	Claude 3.5 Haiku	Mistral Large
IMDB	0.84 ± 0.04	0.76 ± 0.17	0.86 ± 0.04	0.86 ± 0.03
Pseudo Harmful	0.76 ± 0.02	0.76 ± 0.03	0.69 ± 0.03	0.75 ± 0.03
HateBenchSet	0.84 ± 0.04	0.92 ± 0.05	0.86 ± 0.03	0.86 ± 0.04

In our perturbation scheme, we sample from a $\mathcal{U}(-\delta, \delta)$ distribution for each d dimensions. In effect, this is sampling from a d dimensional hypercube of size δ. The rewritten factors are all within $\delta\sqrt{d}$, suggesting that the perturbations we make in the text input space do indeed remain in the factor perturbation space.

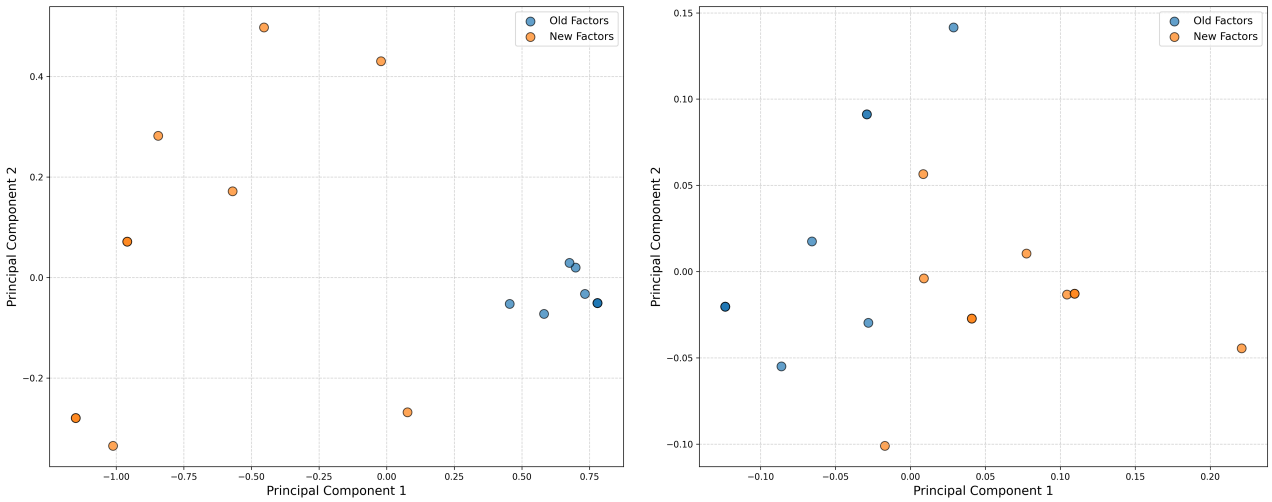

Figure 5: **Rewriting text based on LAMP factors produces localized perturbations in the explanation space.** The first two principle components of the explanation space for IMDB text are plotted. Blue points represent the original text factor weightings, and orange points represent rewritten text factor weightings.

Table 5: **Mean distances in factor space between original and rewritten inputs.** Lower values indicate that the language model-generated rewrites preserve the original factor representations more closely.

Dataset	GPT-4.1-mini	Gemini 2.5 Flash	Claude 3.5 Haiku	Mistral Large
IMDB	0.168	0.033	0.180	0.101
Pseudo Harmful	0.119	0.126	0.217	0.206
HateBenchSet	0.168	0.011	0.180	0.101

D Effectiveness of the Theoretical Perturbation Size

We evaluate the effectiveness of local linear surrogate models in capturing the decision surface of large language models (language models). Specifically, we assess whether restricting perturbations to a theoretically justified radius δ^* as prescribed by Equation 5 improves the local linear fit, as measured by the coefficient of determination R^2. We hypothesize that discarding samples beyond δ^* mitigates curvature-induced bias and yields more faithful surrogate models.

Table 6: **Overall increase in R^2 after applying the optimal perturbation radius.** Values represent the change in coefficient of determination (ΔR^2) when using Equation 5 to adapt the perturbation size for local linear fits. All models show improved fit quality across datasets.

Dataset	GPT-4.1-mini	Gemini 2.5 Flash	Claude 3.5 Haiku	Mistral Large
IMDB	+0.029	+0.161	+0.093	+0.111
Pseudo Harmful	+0.045	+0.012	+0.021	+0.014
HateBenchSet	+0.013	+0.046	+0.082	+0.047

Table 6 shows increase in linearity as measured by R^2, when we only consider points within the optimal δ^*. We note that the maximal variance increase due to removing points outside of the δ^* boundary is no more than 20 percent as detailed in Table 7, while reducing bias due to linearity.

We discuss how the asymptotic MSE was evaluated from Equation 4.

We consider a linear function through the original sampled point w_0. Any perturbation can be approximated linearly with $y_i = \Phi(w_0) + \delta\beta + \epsilon$ for some ϵ noise.

Table 7: **Variance inflation after perturbation.** Variance increases due to a lower sample size due to the truncation from the perturbation step. The inflation factor is proportional to $\frac{n}{n-k}$ where k is the number of points truncated. None of the inflation factors are large.

Model	IMDB	PH	HateBS
GPT-4.1-mini	1.0000	1.0026	1.0870
Gemini 2.5 Flash	1.0263	1.0012	1.0087
Claude 3.5 Haiku	1.0331	1.0032	1.0638
Mistral Large	1.0629	1.0000	1.0101

For a $\Delta \sim \mathcal{U}(-\delta, \delta)$, the bias is given by

$$\begin{aligned}
&\mathbb{E}\left(\Phi(w_0) + \Delta^T\beta - \Phi(w_0) - \Delta^T\nabla\Phi(w_0) - \frac{1}{2}\Delta^T H\Delta\right) \\
&= \mathbb{E}(\Delta^T(\beta - \nabla\Phi(w_0)) - \frac{1}{2}\Delta H\Delta) \\
&= -\frac{1}{2}(\Delta^T H\Delta) \\
&= -\frac{1}{2}\mathrm{Tr}(\mathbb{E}(H\Delta\Delta^T)) \\
&= -\frac{1}{2}\mathrm{Tr}(H)\mathbb{E}(\Delta^2) \\
&= -\frac{1}{2}\mathrm{Tr}(H)\frac{\delta^2}{3}
\end{aligned}$$

Then the square bias term is $\frac{1}{4}\mathrm{Tr}(H)^2\frac{\delta^4}{9}$. We can bound this above by the Cauchy-Schwartz inequality in eigen-space for a more conservative bias estimate with $\frac{1}{4}\|H\|_F^2\frac{\delta^4}{9}$.

Since the direct derivation of the variance term for the local linear regression is computation heavy and beyond the scope of this paper, we derive the variance term for the local zeroth order polynomial regression and draw an analogy from it.

Assume we have a local zero-th order polynomial regression

$$Y_i = \beta_0 + \epsilon_i,$$

where ϵ_i is a random noise with mean 0 and variance σ^2. Assume we have the uniform kernel in d dimensions with bandwidth δ centered at 0. i.e.

$$K(X_i) = \begin{cases} 1 \text{ if } X_i \in [-\delta, \delta]^d \\ 0 \text{ else} \end{cases}$$

Then, the local constant estimator $\hat{\beta}_0$ minimizes

$$\sum_{i=1}^{n}(Y_i - \beta_0)^2 K(X_i). \tag{6}$$

This is equivalent to an unweighted average of Y_i for those that $X_i \in [-\delta/2, \delta/2]^d$. i.e.

$$\hat{\beta}_0 = \frac{\sum_{i=1}^n \mathbb{I}\{X_i \in [-\delta, \delta]^d\}Y_i}{\sum_{i=1}^n \mathbb{I}\{X_i \in [-\delta, \delta]^d\}}, \tag{7}$$

where $\mathbb{I}(\cdot)$ is the indicator function.

Let N be the number of data points X_i that fall into the window $[-\delta/2, \delta/2]^d$. If the data points are uniformly distributed, then

$$\mathbb{E}(N) \approx n\delta^d \tag{8}$$

up to some constant factor as n gets larger. Then the estimator is $\hat{\beta}_0 = \frac{\sum_{i \in \text{window}} Y_i}{N}$ and the variance of the estimator is proportional to $\frac{1}{n\delta^d}$ as n gets larger.

E Meta-aggregation

We test the quality of models with and without meta-aggregation. Since nested F tests to determine model complexity is only appropriate for nested models, we use BIC to evaluate fit.

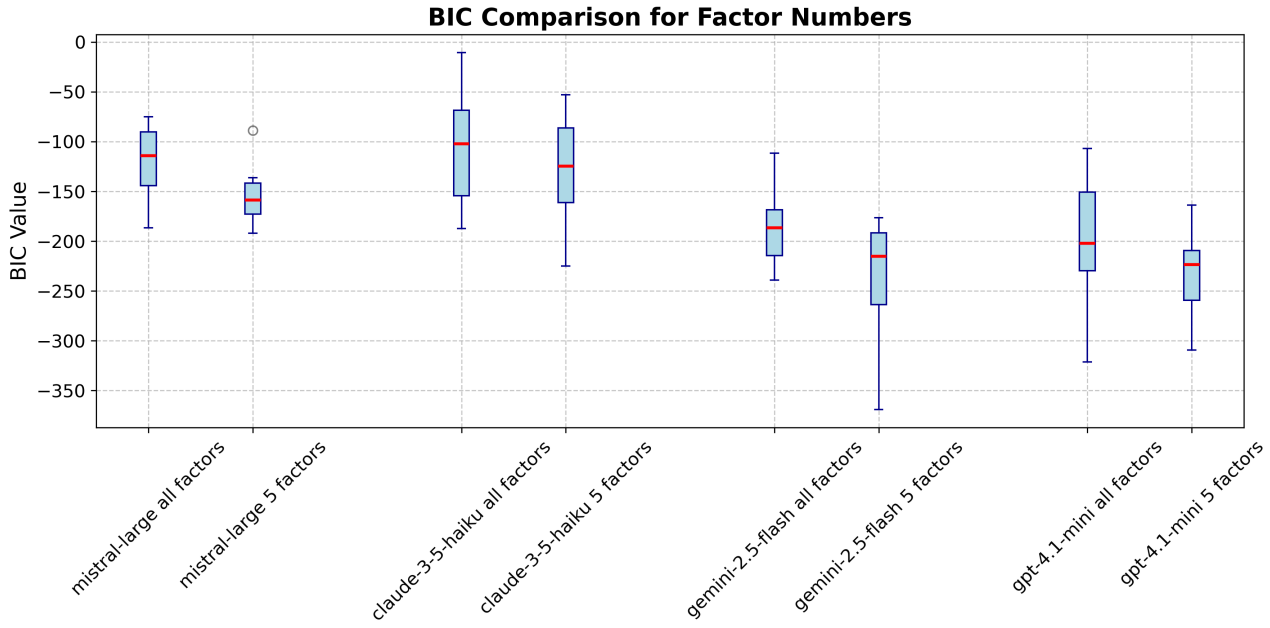

Figure 6: **Models with fewer factors are preferred for surrogate modelling.** BIC comparisons across the board show that a five factor model is overall preferred over a 50 factor model, suggesting the meta aggregation step is able to produce meaningful and concise factors.

As shown in Figure 6, the five-factor models consistently yield lower BIC scores across all datasets. This indicates that despite having fewer parameters, the aggregated models provide equal or better predictive fidelity on the local decision surface.

Although $\mathcal{M}$ can enumerate a large number of explanatory factors, the underlying decision surface is often governed by a much smaller set of behaviorally relevant axes. By summarizing and consolidating related factors, the meta-aggregation step reduces this complexity, yielding a more compact and informative representation. This dimension reduction step not only improves the stability and parsimony of LAMP's local surrogate models, but also enhances the interpretability of the self-reported explanations.

E.1 R^2 comparison of best subsets

We want to ensure that our meta-aggregated smaller model is not misspecified. To do so, we evaluate two quantities. First, we study the adjusted R^2 value of the model generated with the full set of factors, and that of the smaller model. We find that adding more factors does not increase the R^2, likewise decreasing the number of factors to our smaller model does not meaningfully reduce our proportion of variance explained. This suggests that our smaller model selected by the meta-aggregation step sufficiently explains the same amount of variance as the larger model, akin to picking the highest variance principle components.

	IMDB	PH	HateBS
GPT-4.1-mini	0.5348 ± 0.076	0.5250 ± 0.173	0.3090 ± 0.053
Gemini 2.5 Flash	0.3811 ± 0.152	0.4125 ± 0.195	0.1860 ± 0.115
Claude 3.5 Haiku	0.2628 ± 0.126	0.1627 ± 0.063	0.2589 ± 0.131
Mistral Large	0.2789 ± 0.086	0.2529 ± 0.087	0.2154 ± 0.113

Table 8: **Best subset of factors produce R^2 values similar to that of the meta-aggregated factors**. If we skip the meta-aggregation step, and instead select the model with the best subset, we arrive at the following R^2 values. Comparing with Table 1, this hints that our meta-aggregated models are not misspecified.

F Surrogate model effectiveness in predicting locally perturbed texts

We compare LAMP against three naive baselines and three established interpretability methods. All methods are evaluated using the Brier score $s\left(p_h, \hat{p}_s\right) = \left(p_h - \hat{p}_s\right)^2$ as described in Section 5.2. The naive baselines are: a *mean baseline* that predicts the LAMP surrogate intercept for every example, a *uniform baseline* that always predicts 0.5, and a *random baseline* that predicts 0 or 1 uniformly at random.

For LIME (Ribeiro et al., 2016) and SHAP (Lundberg and Lee, 2017), we follow the same surrogate-and-rewrite protocol as LAMP but in the token (word) space. For each data point, 50 perturbations are generated by independently deleting each whitespace-delimited token with probability drawn uniformly from $[0.10, 0.30]$, and the language model is queried on each perturbed text to obtain a class probability. An OLS surrogate is fitted regressing these probabilities onto binary token-presence features. The resulting coefficients are used to construct a rewrite prompt that emphasizes high-coefficient tokens and de-emphasizes low-coefficient ones. For SHAP, we additionally compute Shapley values via the `PartitionExplainer` with `max_evals`$= 50$, which provides principled token importance estimates. For ANCHORS (Ribeiro et al., 2018), we use the official `anchor-exp` package with default hyperparameters ($\tau = 0.95$, $\delta = 0.1$, and batch size 100) and convert the anchor rule to a continuous prediction via linear interpolation between the base prediction and 0.5. The key distinction across methods is the feature space: LAMP operates in the model's self-reported explanation space, whereas the baselines operate in the token space. Results are reported in Table 9.

G Surrogate prediction alignment with language model probability outputs

We correlate the surrogate predictions with the corresponding language model reported probabilities, and find in general a high degree of correlation. Notably, the Claude surrogate does not correlate as strongly with the other language models. Results are reporeted in Table 10

H Tail surfaces

We observe distinct behaviors of LAMP across different regions of the Language Model's predictive distribution. Specifically, the decision surface characteristics in the high-confidence "tail" regions (where predicted probabilities are near 0 or 1) differ from those in the lower-confidence "middle" region. A key observation is the lower coefficient of determination, R^2, for linear surrogates in these tail regions. This naturally raises questions regarding the appropriateness of the linear modeling employed by LAMP in such scenarios. This section addresses this concern by first demonstrating that low R^2 values in tail regions are primarily due to the reduced model responsiveness, characterized by small $\|\beta\|_2$ values, rather than a failure of linearity. Furthermore, we employ the Harvey-Collier test to provide further evidence that a linear model serves as an effective proxy for understanding the local decision surface, even in these tail regions.

H.1 Responsiveness on the Tail Surface

To better understand the low R^2 values observed in the tail regions, we examine the responsiveness on the decision surface. Specifically, we investigate whether these regions are flat and hence not responsive to the changes in explanation weights.

Table 9: **Across the board, LAMP surrogate linear models are able to predict out-of-sample language model output to a small margin of error.** GPT, Gemini, and Mistral are able to predict the language model given probability just by using its locally linear surrogate (lower is better). This is better than using an intercept only surrogate (mean model), a naive baseline (always predicting 0.5), and a random selection baseline (predicting 0 or 1 at random). The surrogates from ANCHORS and, in particular SHAP, features appear to be competitive with LAMP in some settings.

Dataset	Method	GPT-4.1-mini	Gemini 2.5 Flash	Claude 3.5 Haiku	Mistral Large
IMDB	LAMP	0.0303 ± 0.008	**0.0276** ± 0.015	0.0428 ± 0.006	**0.0089** ± 0.003
	Mean baseline	0.1201 ± 0.011	0.3016 ± 0.039	0.0720 ± 0.010	0.0210 ± 0.010
	Uniform baseline	0.1732 ± 0.004	0.2281 ± 0.004	0.1161 ± 0.008	0.1673 ± 0.008
	Random baseline	0.4015 ± 0.036	0.4354 ± 0.094	0.3776 ± 0.039	0.4005 ± 0.047
	LIME	0.3609 ± 0.126	0.7210 ± 0.051	0.5720 ± 0.064	0.6165 ± 0.092
	SHAP	**0.0262** ± 0.039	0.0292 ± 0.096	**0.0194** ± 0.051	0.0103 ± 0.025
	ANCHORS	0.0576 ± 0.127	0.3225 ± 0.397	0.1869 ± 0.246	0.1777 ± 0.283
PH	LAMP	**0.0193** ± 0.002	**0.0073** ± 0.003	**0.0759** ± 0.025	**0.0209** ± 0.009
	Mean baseline	0.2025 ± 0.021	0.3646 ± 0.030	0.2246 ± 0.047	0.2848 ± 0.026
	Uniform baseline	0.1421 ± 0.005	0.3646 ± 0.005	0.1167 ± 0.010	0.1471 ± 0.004
	Random baseline	0.4194 ± 0.037	0.5227 ± 0.093	0.4140 ± 0.039	0.4270 ± 0.047
	LIME	0.4241 ± 0.113	0.2417 ± 0.004	0.2625 ± 0.088	0.3944 ± 0.134
	SHAP	0.0383 ± 0.099	0.2455 ± 0.365	0.1559 ± 0.269	0.1258 ± 0.241
	ANCHORS	0.1480 ± 0.005	0.1841 ± 0.340	0.1702 ± 0.227	0.1406 ± 0.255
HateBS	LAMP	**0.0312** ± 0.008	**0.0720** ± 0.022	**0.0497** ± 0.020	**0.0043** ± 0.003
	Mean baseline	0.1201 ± 0.014	0.4807 ± 0.042	0.3642 ± 0.041	0.4311 ± 0.039
	Uniform baseline	0.1732 ± 0.004	0.2303 ± 0.003	0.1917 ± 0.007	0.2150 ± 0.005
	Random baseline	0.3409 ± 0.146	0.4582 ± 0.175	0.2918 ± 0.126	0.4130 ± 0.157
	LIME	0.3051 ± 0.151	0.1489 ± 0.079	0.1285 ± 0.015	0.1938 ± 0.114
	SHAP	0.0795 ± 0.186	0.1722 ± 0.337	0.0815 ± 0.225	0.1732 ± 0.335
	ANCHORS	0.0973 ± 0.228	0.2189 ± 0.354	0.2043 ± 0.350	0.2407 ± 0.364

Table 10: **Surrogate model predictions correlate strongly with language model output probabilities.** Pearson correlation between predicted probabilities from the surrogate linear models and the original language model predictions. GPT-4.1-mini, Gemini, and Mistral exhibit near-perfect correlation, while Claude shows weaker alignment, consistent with its overall lower R^2 performance (see Table 9).

Dataset	GPT-4.1-mini	Gemini 2.5 Flash	Claude 3.5 Haiku	Mistral Large
IMDB	0.928	0.945	0.871	0.957
Pseudo Harmful	0.969	0.969	0.703	0.903
HateBenchSet	0.966	0.968	0.853	0.988

The magnitude of $\|\beta\|_2$ is directly related to the value of R^2. For centered data (which we can assume without loss of generality), the R^2 value can be given by:

$$R^2 = \frac{\beta^T X^T X \beta}{y^T y}. \tag{9}$$

Here, $\beta^T X^T X \beta$ represents the sum of squares explained by the model, and $y^T y$ represents the total sum of squares. Note that small $\|\beta\|_2$ will lead to $\beta^T X^T X \beta$ as long as $X^T X$ does not disproportionately scale the expression. In the context of LAMP, where perturbations X are generated within a small, controlled neighborhood, $X^T X$ is well behaved. This provides a direct link that the lower responsiveness in a Language Model, characterized by a smaller $\|\beta\|_2$ value for the local linear surrogate results in a lower R^2 value for the surrogate.

We define the tail surfaces as the decision surfaces where the predicted probabilities of LMs are near the extrema (0 or 1). These are cases where the language models appear highly confident in its classification. For example, a hateful text with several hateful elements might yield the language model to predict $P(\text{hateful}) \approx 1$. In such confident regions, a small perturbation to the explanation weights is not likely to change the model's classification or probabilities, and the decision surface exhibits less responsive behavior. However, when the language model is

Table 11: R^2 **values in head (body) vs. tail regions of the decision surface.** Coefficient of determination (R^2) for local surrogate models fit in low-confidence (tail) and high-entropy (body) regions of the model's predicted probability space. All models show lower R^2 in tail regions, especially on HateBenchSet, reflecting reduced local responsiveness in confident regimes (see Appendix H).

	Tail Region			Body Region		
Model	IMDB	PH	HateBS	IMDB	PH	HateBS
GPT-4.1-mini	0.386	0.340	0.158	0.585	0.466	0.311
Gemini 2.5 Flash	0.255	0.248	0.223	0.584	0.271	0.349
Claude 3.5 Haiku	0.239	0.196	0.158	0.393	0.237	0.268
Mistral Large	0.272	0.235	0.195	0.477	0.290	0.216

less confident in its classification, smaller shifts in the latent space tip are able to tip the prediction from one class to another, thus small perturbations yield large changes.

This gives an important geometric intuition about the decision surface, the flatter the decision surface, the less responsive the model is to perturbations. To quantify the flatness of the decision surface, we compute the l^2-norm of the surrogate's coefficient, $\|\beta\|_2$. A small $\|\beta\|_2$ suggests a flat surface with minimal directional responses, whereas a large value indicates that the model's predicted probability changes more steeply in responses to perturbations in explanation weights.

We divide model outputs into three bins to evaluate $\|\beta\|_2$ separately: low-confidence middle region (0.2,0.8) and high-confidence tails (0,0.2) and (0.8,1.0). The central low-confidence region corresponds to less confidence and high entropy, where we expect the decision surface to be more dynamic; the tail regions correspond to saturation regions, where we expect flatter, less-responsive the decision surface.

Figure 7 illustrates this trend. In general, the middle bins show significantly higher average $\|\beta\|_2$, confirming that the models are more responsive when they are less confident. In contrast, the lower tails (0,0.2) tend to show small coefficient norms, consistent with flatness and perturbation insensitivity. However, the upper tail (0.8,1.0) is more variable. While it is less responsive than the middle region, the coefficient norms are not as small as those in the lower tail. This is more pronounced in GPT-4.1-mini, indicating that the model's confident positive outputs remain sensitive to local perturbations to explanation weights.

H.2 Further Testing for Linearity

Having established that reduced surface responsiveness contributes to lower R^2 values in tail regions, we now provide further evidence for the suitability of linear models in approximating the local decision surface. The Harvey-Collier test (Harvey and Collier, 1977) offers a method to assess the specification of a fitted linear model by examining whether its recursive residuals are centered around zero. The underlying intuition is that if the relationship is truly linear, each new observation should not introduce systematic bias. Hence, the recursive residuals average out to zero.

We applied the Harvey-Collier test to each linear surrogate fitted by LAMP. Out of a total of 529 linear surrogates, 466 did not lead to a rejection of the null hypothesis that the linear model is correctly specified. If we exclude 34 surrogates that had $R^2 = 0$, the test did not reject the null hypothesis of linearity for approximately 94% of the remaining fitted surrogates. This further supports that a linear surrogate of LAMP is an appropriate proxy for locally estimating the decision surface of Language Models, even when R^2 values might be low due to the inherent flatness of the surface in high-confidence "tail" regions.

I LIME as a baseline

Indeed LIME can be used to estimate its token level logits, where one can calculate class probabilities by performing a softmax on the class token. However this requires access to model output logits, which require white-box or gray-box access to the model. On the other hand, LAMP is accessible on black-box models. For appropriate comparison, we consider LIME's surrogate models to model the language model reported probabilities as the response variable.

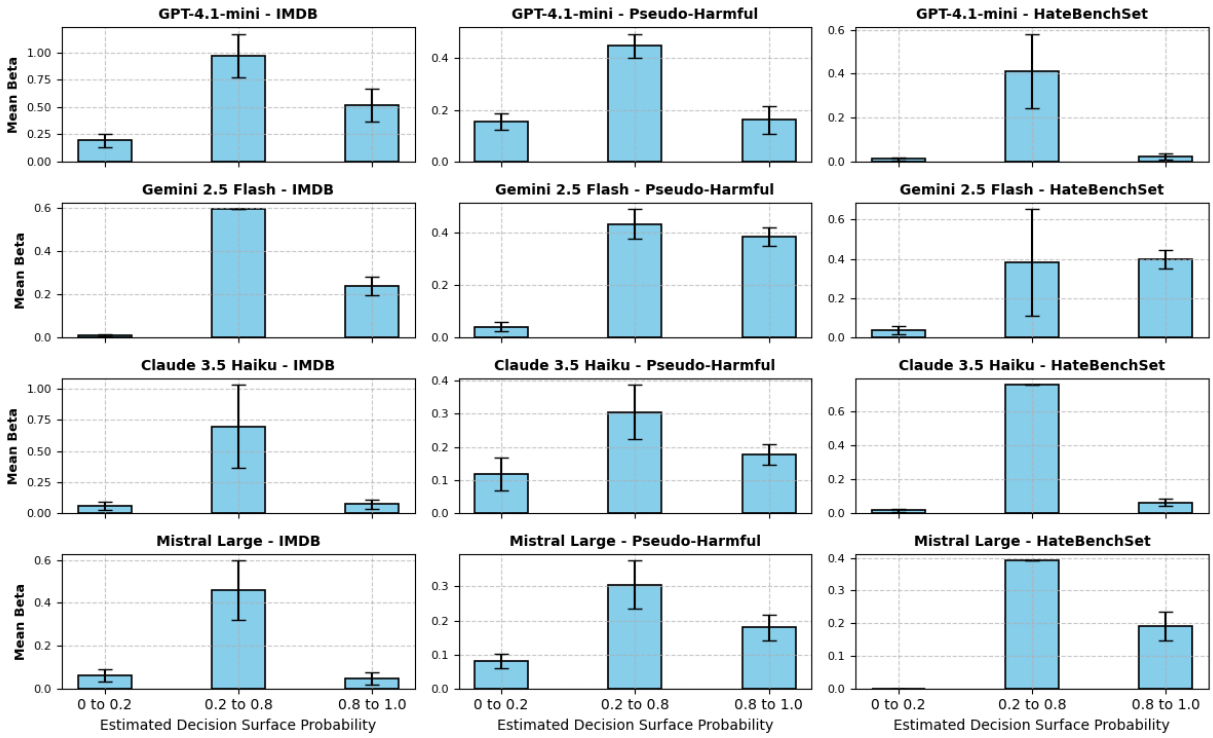

Figure 7: **Generally the decision surfaces exhibit sigmoidal behavior, where the lower and higher probability regions are flatter.** Notably for Gemini 2.5 Flash, the higher probability predictions still have a higher slope.

LIME feature importances are difficult to interpret. It requires manual feature engineering of pertubation token size. In this example, we mask our words that are delimited by spaces.

We see the same surrogate models run on LIME outputs in Figure 12.

J Comparison with integrated gradients

Comparing LAMP to gradient-based explanation method is particularly interesting because LAMP operates fully in a black-box setting while gradient based methods require white box access. To enable such a comparison, we evaluate LAMP alongside Integrated Gradients (IG) on using Qwen3-8B and Qwen3-14B with weights loaded from the official Qwen Hugging Face repository, and we evaluate on both methods on the HateBenchSet dataset. Experiments were completed on a single A100 GPU.

To construct perturbations, we prompt a lightweight language model (GPT-4.1-nano) to identify the six tokens most influential for both the positive and negative class. We extract the corresponding IG scores and their input embeddings. We then perturb these embeddings by adding Gaussian noise with $\sigma = 0.05$, run the model forward again to measure the resulting change in the positive-class logit, and recompute the integrated gradients for the perturbed inputs.

Although this procedure differs from the LAMP perturbation paradigm, it still enables a meaningful empirical comparison. The resulting perturbations form a surrogate model by regressing the observed class logits (converted to probabilities) onto the Integrated Gradients features.

Figure 13 shows the R^2 comparison between the surrogate model generated by LAMP as well as that of integrated gradients.

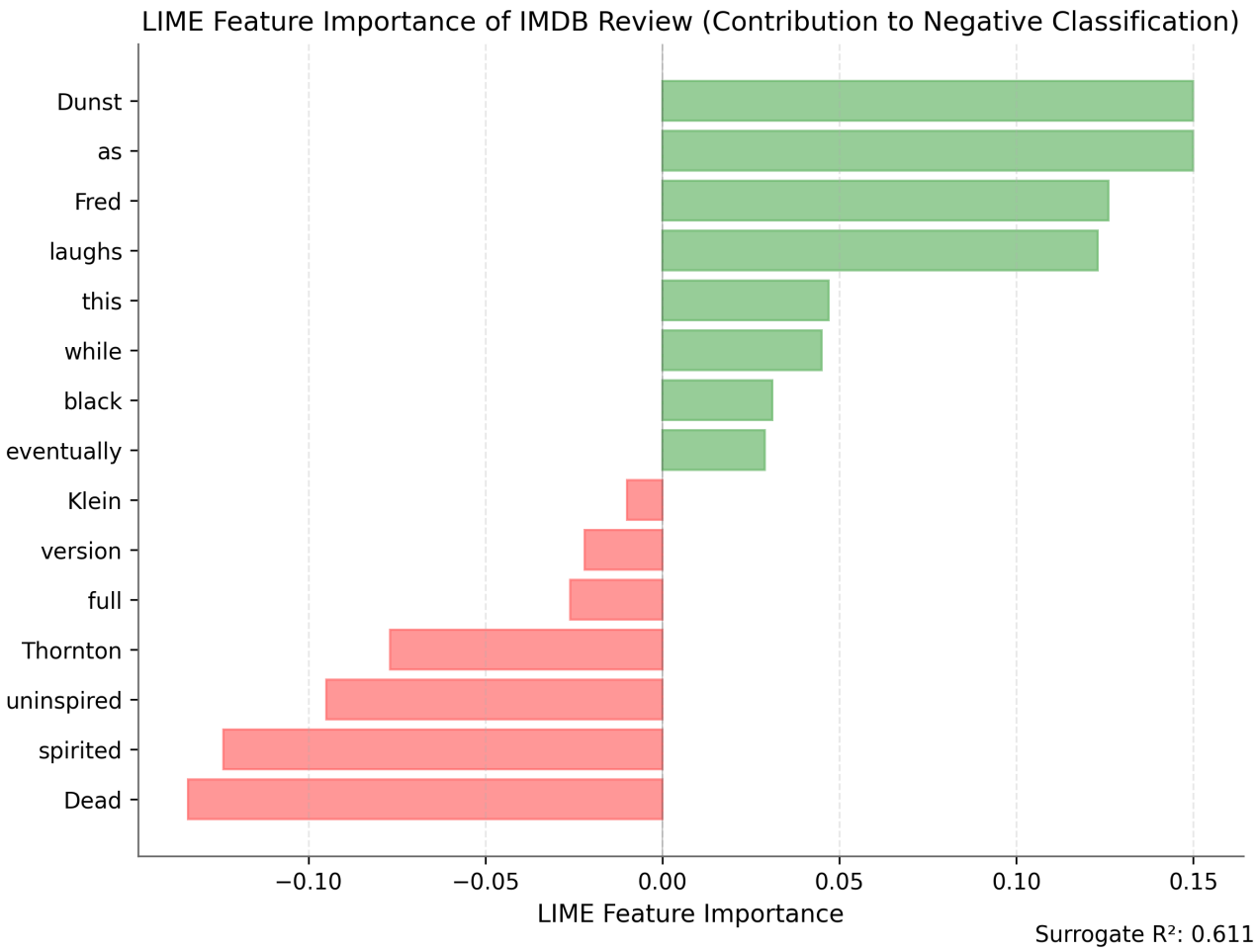

Figure 8: LIME outputs are not as easily interpretable despite having a high R^2, Compare these features to those of Figure 2.

Table 12: **Side-by-side comparison of coefficient of determination (R^2) values for local linear surrogate models (LAMP vs. LIME).** Higher R^2 indicates stronger local linear structure. LAMP values correspond to the surrogate models introduced in this paper (Table 1), while LIME values come from the baseline surrogate fits. Lower values on HateBenchSet and tail regions reflect reduced variance explainability, consistent with Appendix H. Note that LIME often operates over a much larger feature space, as discussed in Figure 8.

	IMDB	**PH**	**HateBS**
Model	LAMP / LIME	LAMP / LIME	LAMP / LIME
GPT-4.1-mini	$0.42 \pm 0.03 / 0.41 \pm 0.06$	$0.38 \pm 0.03 / 0.38 \pm 0.10$	$0.17 \pm 0.02 / 0.46 \pm 0.06$
Gemini 2.5 Flash	$0.26 \pm 0.03 / 0.26 \pm 0.05$	$0.25 \pm 0.03 / 0.32 \pm 0.07$	$0.23 \pm 0.03 / 0.14 \pm 0.04$
Claude 3.5 Haiku	$0.26 \pm 0.03 / 0.40 \pm 0.07$	$0.21 \pm 0.02 / 0.40 \pm 0.08$	$0.16 \pm 0.01 / 0.20 \pm 0.04$
Mistral Large	$0.30 \pm 0.03 / 0.29 \pm 0.05$	$0.25 \pm 0.03 / 0.34 \pm 0.05$	$0.20 \pm 0.02 / 0.21 \pm 0.05$

Model	**Method**	Average R^2	OOD MSE ($\pm$ 1 sd)	p-value
Qwen3-8B	LAMP	**0.412724**	0.03688 $\pm$ 0.134	
	IG	0.264315	**0.00620** $\pm$ 0.028	0.2325
Qwen3-14B	LAMP	**0.372455**	0.1063 $\pm$ 0.171	
	IG	0.222765	**0.0218** $\pm$ 0.075	0.1297

Table 13: The average R^2 values for LAMP show higher degrees of linearity than that of the IG surrogate models. Qwen3-8B and Qwen3-14B were used as the base language model. Unsurprisingly, errors on rewritings show that IG produces better surrogate models, though not statistically significant with two-sample t-test.

Notably, the integrated gradient method is better at producing a surrogate that works on rewritten text. However, this is not surprising as gradient based methods leverage more information. In this set up, a particular advantage that is available to open source models are access to token logits, embedding spaces, and gradient information. Naturally, if we were able to access gradients perturb input embeddings, we should in theory be able to approximate any logit decision surface with a quantifiable error bound as noted in Appendix D. LAMP, while does not require white box access, can still produce a surrogate model that performs on par with gradient based methods.

K Runtime and compute resources

The largest factor in runtime arises not from local computation but from querying the language model. Let n denote the number of LAMP perturbations generated per input. For each input example, the surrogate model construction requires $O(n)$ forward passes to obtain predictions on perturbed inputs. The cost and time of querying the language model is non-negligible and scales linearly with n.

To mitigate this, we parallelize language model queries using asynchronous batching. Each batch of n perturbations is dispatched concurrently, reducing wall-clock latency $O(n)$ to $O(d)$ where $d < n$ is the number of retries in the case of parsing errors. In the most ideal situation this becomes $O(1)$. In practice, latency is bounded by network overhead and the language model service's throughput limits. In these experiments we collected 50 perturbations, which takes an average of 1.2 seconds to complete, depending on network latency. This was consistent throughout all language models we queried.

Our query service was provided by both OpenAI and OpenRouter.

LAMP's runtime is dominated by querying the language model rather than local computation. For each input, we generate $n = 50$ perturbations $\{\delta_i\}_{i=1}^{n}$ and perform $O(n)$ forward passes to fit the surrogate $\hat{s} = \Phi(w_0) + \beta\delta$, where $\beta = (\Delta^\top \Delta)^{-1} \Delta^\top \mathbf{s}$. To reduce latency, we parallelize language model queries via asynchronous batching, achieving effective latency of $O(d)$ where $d \ll n$ reflects retry overhead. In practice, each batch completes in approximately 1.2 seconds across all models, with throughput limited primarily by network and API service constraints.