

EFFICIENT LARGE LANGUAGE MODELS MODERATION WITH MULTI-LAYER LATENT PROTOTYPES

Anonymous authors

Paper under double-blind review

ABSTRACT

With the widespread adoption of large language models (LLMs), ensuring their safety and alignment has become a critical challenge. Although modern LLMs are aligned with human values during post-training, robust moderation remains essential to prevent harmful outputs at deployment time. Existing approaches, such as guard models, activation steering, and prompt engineering, each involve significant trade-offs: guard models are costly to train and deploy, and their users are typically limited to a few model checkpoints, while activation steering and prompt engineering can often degrade the quality of responses. In this work, we introduce Latent Prototype Moderator (LPM), a lightweight moderation tool that assesses input safety by sparsely aggregating Mahalanobis distances to safe and harmful prototypes across multiple layers. By leveraging multi-level prototypes, LPM improves both moderation robustness and performance. By design, our method adds negligible overhead to the generation pipeline and can be seamlessly applied to any model. LPM achieves state-of-the-art performance on diverse moderation benchmarks and demonstrates strong scalability across model families of varying sizes. Moreover, we show that it integrates smoothly into end-to-end moderation pipelines and further improves response safety when combined with output moderation techniques. Overall, our work provides a practical and adaptable solution for robust, efficient safety moderation for real-world LLM deployment.

1 INTRODUCTION

Large language models (LLMs) have quickly become central to modern applications, making their safety and alignment with human values increasingly important. While techniques like RLHF (Bai et al., 2022; Ouyang et al., 2022) and instruction tuning (Li et al., 2024) have substantially improved model safety, even state-of-the-art models remain susceptible to emergent risks that can appear *after* alignment (Andriushchenko et al., 2025; Carlini & et al., 2023; Wei & et al., 2023). As a result, ensuring practical LLM safety also requires more than alignment during training moderation: the evaluation of model inputs and outputs for safety before they reach end users. The increasing demand for safe deployment has driven recent development of input and output (Lee et al., 2025; Zheng et al., 2024) moderation tools, which collectively enhance the overall safety of LLM systems.

As outlined above, LLM input moderation techniques are an essential part of the safe deployment of LLMs. Commonly used input moderation tools include specialized guard models (Dong et al., 2024; Ghosh et al., 2024; Han et al., 2024; Inan et al., 2023; Sharma et al., 2025; Yin et al., 2025) and latent-based methods (Ayub & Majumdar, 2024; Abdelnabi et al., 2025), but each of these approaches comes with specific trade-offs. Guard models offer good performance at the cost of substantial computational resources, and their training requires access to carefully curated datasets. This often limits users to a small set of pre-trained models with fixed sizes, complicates deployment in resource-constrained environments, and hinders adaptation to emerging categories of safety risks. On the other hand, the latent-based methods are lightweight, but perform worse than guard models in practice. These challenges highlight the need for performant input moderation tools that are also efficient, flexible, and easily customizable.

In this work, we address the aforementioned gap with Latent Prototype Moderator (LPM), a lightweight, latent-based method for input safety assessment. LPM builds on prior findings that explore LLM latent representations Zou et al. (2023a) for safety evaluation and uses the internal states

054
055
056
057
058
059
060
061
062
063
064
065
066
067
068
069
070
071
072
073
074
075
076
077
078
079
080
081
082
083
084
085
086
087
088
089
090
091
092
093
094
095
096
097
098
099
100
101
102
103
104
105
106
107

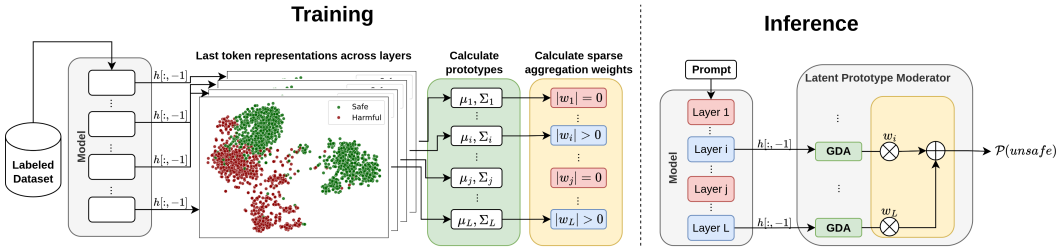


Figure 1: Our proposed Latent Prototype Moderator (LPM) framework. **During training**, we compute class-conditional prototypes (μ_i, Σ_i) based on last-token hidden representations ($h_{i,t-1}$), which we use to define a per-layer Gaussian Discriminant Analysis (GDA) classifier. We then learn sparse aggregation weights (w_i) over the GDA scores. **During inference**, we use the pre-trained GDA classifiers to compute Mahalanobis-distance-based scores from layers with non-zero weights ($|w_i| > 0$), and produce a safety probability, $P(\text{unsafe})$, from their weighted aggregate. LPM enables state-of-the-art performance with lightweight training and negligible inference overhead.

of off-the-shelf LLMs to assess input safety via Mahalanobis distance to safe and unsafe prototypes. Inspired by research on intermediate-layer representations in deep neural networks (Masarczyk et al., 2023; Szatkowski et al., 2025), we further incorporate diverse intermediate-layer embeddings to improve safety assessment. LPM’s lightweight, model-agnostic design ensures compatibility with any LLM while incurring minimal computational overhead during both training and inference. The design of our method is illustrated in Figure 1.

Through extensive experiments, we demonstrate that LPM achieves state-of-the-art performance across diverse input moderation benchmarks, outperforming the alternative techniques. We show that our method works seamlessly across different model families and sizes, highlighting its flexibility and ease of application. We further demonstrate that LPM can be readily integrated into end-to-end moderation pipelines alongside output moderation tools, enhancing the overall safety of LLM systems. Finally, through detailed ablations, we reveal LPM’s compatibility with various model types, including pretrained and reasoning models, its efficiency in low-data regimes, ease of extensibility, and the intriguing multi-layer dynamics underlying its performance. Our key contributions are:

- We introduce Latent Prototype Moderator (LPM), an efficient LLM input moderation method that uses Mahalanobis distance across multi-level latent spaces to assess prompt safety.
- We demonstrate that our method achieves state-of-the-art safety assessment across the common input moderation benchmarks and offers a practical solution for LLM robustness.
- We show how LPM seamlessly integrates with diverse moderation strategies, lowering unwanted refusals and enhancing the overall safety in end-to-end LLM deployment.

Taken together, our work provides a highly efficient and customizable solution for LLM input moderation, representing a key step toward safe AI deployment.

2 RELATED WORK

LLM safety and alignment LLMs are usually pre-trained on large corpora of data that are impossible to fully supervise. Therefore, pre-training is typically followed by supervised training that ensures the alignment of the model with human preferences and values (Bai et al., 2022; Dai et al., 2024; Li et al., 2024; Lim et al., 2025; Ouyang et al., 2022). However, various studies prove that even the most popular frontier models are still prone to generating unsafe responses (Carlini & et al., 2023; Wei & et al., 2023; Zou et al., 2023b), as safety alignment can be superficial and lose its effect after the initial few tokens of generation (Qi et al., 2025). Furthermore, focusing strictly on the safety aspect of alignment of the model might negatively affect its capabilities (Huang et al., 2025; Wei et al., 2023; Wolf et al., 2024). To ensure model safety in a way that does not degrade the quality of responses, various solutions dedicated to either the detection of unsafe prompts or the correction of unsafe answers have emerged. These include rule-based approaches (Clarke et al., 2023; Kumar et al., 2024), representation engineering (Luo et al., 2024; Qiu et al., 2024; Zou et al., 2024), and guard models (Dong et al., 2024).

LLM output moderation and steering Output moderation is ensured mostly by either blocking unsafe responses with a guardrail (Inan et al., 2023; Han et al., 2024) or by steering the LLMs toward safe responses. Steering can be achieved by either using prompt engineering methods (Zheng et al., 2024; Xie et al., 2023) or activation steering (Zou et al., 2023a; Lee et al., 2025). Activation steering techniques operate on the hypothesis that concepts are encoded in the activation patterns of the model. For instance, methods like Representation Engineering (Zou et al., 2023a) add or subtract vectors corresponding to specific concepts (e.g., "harmfulness") from model activations to guide the output generation process. Other approaches, such as PaCE (Luo et al., 2024), utilize decomposition into concepts and suppress undesired properties such as toxicity by utilizing a concept dictionary. Similarly, SEA (Qiu et al., 2024) aims to erase harmful concepts from the model’s activations using spectral decomposition. Other approaches perform latent space alignment by fine-tuning the model to ensure its internal representations more closely resemble desired safe ones (Zou et al., 2024; Zhang et al., 2024). While powerful, these methods can have a negative influence on tasks where harmful concepts are not present (Lee et al., 2025).

LLM input moderation. Input moderation assesses a prompt before generation begins, and simply refusing to engage with malicious queries is often a sufficient and highly efficient safety strategy (Manczak et al., 2024). The most common tools for input moderation are guard models—specialized LLMs trained to evaluate text safety (Inan et al., 2023; Han et al., 2024; Ghosh et al., 2024; Sharma et al., 2025). However, most guard models are fine-tuned versions of open-source LLMs, requiring high-quality datasets and substantial computational resources for their development. An alternative to guard models are methods that utilize inner representations for detecting malicious prompts (Abdelnabi et al., 2025; Ayub & Majumdar, 2024). Guard models are inefficient when it comes to retraining them, and previous methods that utilize latent representations do not incorporate the knowledge from multiple layers, which contain distinct features (Masarczyk et al., 2023).

Our work addresses the above-mentioned gaps with LPM, a new input-moderation approach that offers an efficient substitute for heavy guard models and can serve as a complementary first layer of defense alongside output-steering methods without degrading model capabilities. We highlight how our method is the first to combine prototype-based classification with multi-layer feature aggregation, yielding greater robustness and consistently stronger performance than prior techniques.

3 METHOD

3.1 PRELIMINARIES

Formal definition of LLM input moderation. LLM input Moderation tools are designed to prevent LLMs from generating responses to harmful inputs, denoted as x_{harmful} , by evaluating the prompt before output generation begins. In such contexts, the objective is to flag all inputs x which fall into a pool of harmful prompts $\mathcal{X}_{\text{harmful}}$. Thus, the goal of the moderation tool M is to say whether $x \in \mathcal{X}_{\text{harmful}}$ and allow generations LLM(x) only for $x \notin \mathcal{X}_{\text{harmful}}$, which minimizes the probability that generated text will also be harmful and saves computations when a refusal is sufficient.

Prototype calculation. For the case of safety moderation, we operate on two main classes: safe and harmful examples. To perform classification, we first compute a *prototype* (mean of the class distribution) for each class by calculating its empirical mean $\mu_c = \frac{1}{N_c} \sum_{i=1}^{N_c} x_i^{(c)}$, where $x_i^{(c)}$ is the i -th example belonging to class c , and N_c is the number of examples for that class.

Mahalanobis distance. To account for complex data distributions, we can also employ the *Mahalanobis distance*, which accounts for the data distribution’s variance and covariance. Unlike Euclidean distance, Mahalanobis distance is sensitive to the underlying data geometry, making it well-suited for distinguishing between clusters with different shapes and orientations. Mahalanobis distance between a point $x \in \mathbb{R}^d$ and the distribution represented by mean μ_c and covariance matrix Σ_c is defined as:

$$d_M(x, \mu_c) = \sqrt{(x - \mu_c)^\top \Sigma_c^{-1} (x - \mu_c)}. \quad (1)$$

We estimate the matrix Σ^{-1} with a Bayes ridge-type estimator (Kubokawa & Srivastava, 2008):

$$\widehat{\Sigma}^{-1} = d \cdot ((N - 1)\widehat{\Sigma} + \text{tr}(\widehat{\Sigma})I_d)^{-1}, \quad (2)$$

where N is the number of examples used to estimate the covariance matrix, I_d is the identity matrix, and $\hat{\Sigma}$ is the empirical covariance matrix.

Given (μ_c, Σ) , we can perform Bayesian classification through Gaussian Discriminant Analysis (GDA). When using uniform apriori, the probability that an example $x \in \mathbb{R}^d$ belongs to class c is:

$$\mathcal{P}(c|x) = \frac{\exp(-\frac{1}{2}(x - \mu_c)^T \Sigma_c^{-1} (x - \mu_c))}{\sum_{i=1}^k \exp(-\frac{1}{2}(x - \mu_i)^T \Sigma_i^{-1} (x - \mu_i))}. \quad (3)$$

3.2 LATENT PROTOTYPE MODERATOR

Intermediate layers properties. LPM’s design is based on the principle that different neural network layers capture distinct information (Masarczyk et al., 2023). Prior research shows intermediate layers often retain useful features that final ones discard (Szatkowski et al., 2025). We confirmed this holds true when assessing safety, as we found no single layer consistently performs best across different models or tasks (see Appendix B). LPM is therefore built to leverage these varied signals by aggregating the distances from multi-layer prototypes to produce a comprehensive safety assessment.

Aggregating multiple layers. To obtain the prototypes, we extract hidden representations from the last tokens of the safe and unsafe inputs from the training data. We use the representations of the final token after the feed-forward network (FFN) in all blocks $\{1, \dots, L\}$, where L denotes the number of transformer blocks in the model.

To utilize information from multiple layers, we use Mahalanobis scores from each layer $S_l = \mathcal{P}_l(\text{harmful}|x, \mu_l, \Sigma_l)$ we calculate aggregation weights $\{w_1, \dots, w_L\}$, which are trained on the same examples using Mahalanobis scores from all layers $\{S_1, S_2, \dots, S_L\}$. To ensure robustness and sparsity of this aggregation, we use the l_1 penalty to calculate aggregation weights. Final prediction of LPM is described by the following equation:

$$\text{LPM}(x|\{\mu_1, \dots, \mu_L\}, \{\Sigma_1, \dots, \Sigma_L\}) = \sum_{i=1}^L w_i \cdot \mathcal{P}_i(\text{harmful}|h_i, \mu_i, \Sigma_i), \quad (4)$$

where h_l are latent representations of the last token after layer l of the prompt x and μ_l, Σ_l are prototypes and covariance matrix calculated from representations in layer l .

Multiple prototypes. Safe and unsafe classes might span across diverse types of risk categories, and using multiple prototypes for a single class can provide us with a way to better model their distributions. In safety assessment, we can split harmful and safe examples into subgroups and construct a prototype for each subgroup. To enrich our LPM by having prototypes for K subgroups, we can utilize them as additional features $\{S_{1,1}, \dots, S_{1,L}, \dots, S_{K,L}\}$ for aggregation. Usage of multiple prototypes can be highly important when subgroup distributions are not similar, e.g., using prototypes from two highly different datasets.

4 EXPERIMENTS

We use Aegis-Defensive (Ghosh et al., 2024), LlamaGuard3 (Inan et al., 2023), Granite Guardian (Padhi et al., 2024), ShieldGemma (Zeng et al., 2024), and WildGuard (Han et al., 2024) to compare our method with models specifically designed for moderation. We use state-of-the-art methods for input moderation that utilize latent representations such as Abdelnabi et al. (2025) and Ayub & Majumdar (2024). As base models for latent-based methods we use Mistral (Jiang et al., 2023), Llama (Grattafiori et al., 2024), OLMo (OLMo et al., 2024), and Qwen3 (Yang et al., 2025). Those models use similar architectures but differ in training protocols, which allows us to assess our approach across models containing diverse knowledge. We compare LPM to other methods on groups of datasets corresponding to *prompt harmfulness* (8 datasets). Among those datasets are WildJailbreak (Jiang et al., 2024) and WildGuardMix (Han et al., 2024), which contain 34 unique sophisticated state-of-the-art jailbreak tactics. For neutral evaluation, we utilize datasets of *general capabilities* (7 datasets) to ensure that input moderation methods are not too strict, which would highly decrease their usability due to frustration by the users. We refer to those two groups of

Table 1: F1 score for safety assessment on harmful datasets for LPM, dedicated guard models, and latent-based detection methods. LPM applied to various LLMs performs competitively with the guards and outperforms other methods, achieving the best overall performance with OLMo.

Dataset	Aegis	HarmB	OpenAI	SimpST	TChat	WGMix	WJ	XS	Avg
Guard Models									
Aegis-Guard-D	81.00	70.46	76.44	97.96	75.61	72.09	75.44	81.53	78.82
Aegis-Guard-P	75.72	66.11	78.11	94.18	68.49	65.63	55.72	82.35	73.29
LlamaGuard1	72.92	66.11	74.38	92.47	57.14	55.08	41.36	81.61	67.63
LlamaGuard2	71.85	93.78	76.10	95.83	46.32	70.52	49.85	89.18	74.18
LlamaGuard3	71.74	98.94	79.11	99.50	54.11	76.76	67.83	88.52	79.56
GraniteGuardian-3-1-8B	87.78	79.90	77.63	99.50	73.25	84.57	96.75	85.59	85.62
ShieldGemma-9B	77.44	69.04	77.63	91.30	68.13	58.88	59.94	82.41	73.10
WildGuard	89.78	99.37	72.28	99.50	70.14	88.04	97.10	95.26	88.93
Ayub & Majumdar (2024)									
OLMo2-7B	88.11	96.54	66.95	100.0	65.63	88.09	96.84	94.33	87.06
Mistral-7B	79.60	90.87	75.69	98.99	63.44	83.31	87.53	95.04	84.31
Llama-8B	82.52	96.98	66.60	98.48	55.62	80.91	82.85	92.76	82.09
Qwen3-8B	80.00	90.62	74.56	95.29	68.90	80.64	81.31	90.21	82.69
Abdelnabi et al. (2025)									
OLMo2-7B	84.16	93.30	75.19	99.50	72.53	86.20	93.48	94.43	87.35
Mistral-7B	84.55	97.86	64.59	98.48	57.63	85.73	91.13	94.60	84.32
Llama-8B	84.16	95.18	67.99	98.99	59.59	86.27	93.27	90.63	84.51
Qwen3-8B	84.47	99.37	68.45	97.96	60.94	85.18	90.44	88.06	84.36
LPM									
OLMo2-7B	89.23	98.51	74.21	100.0	76.51	88.52	97.55	96.91	90.18
Llama-8B	85.13	99.58	72.85	99.50	69.17	88.04	94.69	97.44	88.30
Mistral-7B	87.36	99.16	70.68	99.50	66.33	87.63	93.65	96.10	87.55
Qwen3-8B	83.49	100.0	72.35	96.91	64.40	86.21	92.00	92.23	85.95

datasets as *harmful* and *neutral*, respectively. For LPM and other latent-based methods training, we always utilize the training set of WildGuardMix (Han et al., 2024) if not stated otherwise. Across our experiments, we report the average F1 score on harmful datasets, which contain both safe and unsafe inputs, and the average true negative rate across all benign inputs from neutral datasets. For not-deterministic approaches, we show the average results from 5 runs.

4.1 LPM PERFORMANCE ON INPUT MODERATION

In Table 1, we evaluate LPM applied to the instruction-finetuned Llama, Mistral, OLMo, and Qwen3 models. We benchmark our method against well-established guard models: Aegis Permissive and Defensive, LlamaGuard1, 2, and 3, and WildGuard and other latent-based methods (Abdelnabi et al., 2025; Ayub & Majumdar, 2024). LPM consistently outperforms LlamaGuard, Aegis, and other latent-based methods. When applied to OLMo, LPM surpasses even the strongest guard baseline, WildGuard. Our results demonstrate that LPM can efficiently perform safety assessment at a level comparable to the best available alternatives, while remaining remarkably lightweight. In addition to high performance on harmful datasets, LPM triggers on neutral datasets only for $\sim 1\%$ examples (see Appendix C.1). For more details on the datasets see Appendix A.

Table 2: Attack success rate (ASR) and false refusal rate (FRR) when combining LPM with output moderation methods. Our method improves output moderation tools by decreasing FRR when using LPM as a conditioning mechanism. We additionally show an F1 score between (1-ASR) and (1-FRR) to analyse the trade-off between the two moderation objectives captured by these metrics.

Method	Llama3-8B			Mistral-7B			OLMo2-7B			Qwen3-8B		
	ASR↓	FRR↓	F1↑	ASR↓	FRR↓	F1↑	ASR↓	FRR↓	F1↑	ASR↓	FRR↓	F1↑
Base Model	36.21	2.54	74.75	78.25	1.90	34.51	20.82	5.93	80.48	50.53	2.22	63.78
+LPM Simple Refuse	15.52	5.93	83.47(+8.72)	15.92	6.24	82.84(+48.33)	14.59	6.03	83.85(+3.37)	18.30	6.24	81.52(+17.74)
Lee et al. (2025)	36.47	7.51	68.71	73.61	6.24	37.31	20.42	6.56	79.90	50.13	2.65	63.66
Unconditioned Steering	35.94	53.76	27.20	40.45	18.73	54.29	18.57	8.15	78.90	47.75	3.81	64.41
+LPM Conditioning	44.56	4.66	66.03(+38.83)	45.89	3.60	66.15(+11.86)	19.36	6.46	80.63(+1.73)	48.67	2.54	65.02(+0.61)
Zheng et al. (2024)	25.07	6.03	77.82	54.11	3.49	59.30	19.76	8.25	78.12	28.91	5.71	75.83
+LPM Conditioning	26.92	4.02	79.21(+1.39)	55.44	2.43	59.16(-0.14)	20.95	6.35	79.86(+1.74)	31.56	3.92	76.30(+0.48)

4.2 LPM SCALING WITH MODEL SIZE

LLM performance scales with model size and training data (Kaplan et al., 2020). While larger models offer greater capabilities, they also demand more computational resources. To account for this, LLMs are often released as model families of varying sizes, allowing users to trade off performance and efficiency. Guard models are typically available in only a single size, which limits their usability in low-resource environments. LPM addresses these challenges, as it requires no training and can therefore be applied to any off-the-shelf LLM, enabling scalable and domain-flexible moderation.

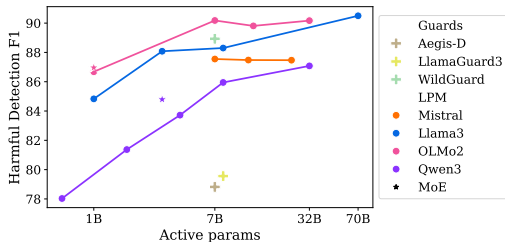


Figure 2: F1 score for harmful prompt detection vs. model size across different model families.

In Figure 2, We investigate LPM performance on different model families to assess how well our method performs across different model sizes. We evaluate LPM on models from the Llama3, Mistral, OLMo2, and Qwen3 families, including the MoE (Shazeer et al., 2017) variants of OLMo2 (7B with 1B active parameters) and Qwen3 (30B with 3B active parameters). While LPM performance with MoE is slightly lower than with its dense counterpart, LPM remains effective and is compatible with this increasingly more popular model architecture (Liu et al., 2024). Notably, LPM also performs competitively on small 1B models.

4.3 COMBINING LPM WITH OUTPUT MODERATION

Building on the complementary nature of input and output moderation, we evaluate LPM’s efficacy when integrated with prompt and activation steering. A key motivation for this analysis is the finding by Lee et al. (2025) that steering can degrade performance on benign prompts. We therefore compare two scenarios: one where steering is applied to all prompts, versus another where steering is conditionally activated only when LPM flags a prompt as unsafe. We also evaluate the influence of returning a simple refusal message to determine how our LPM method works on its own. Following Zheng et al. (2024), we utilize a safety prompt as a prompt steering baseline (see Appendix D.1 for the details).

To assess performance, we evaluate the responses for harmfulness using the WildGuard model on the WildGuardMix test set. The activation steering methods are derived by sampling 5,000 harmful and 5,000 safe responses from the WildGuardMix training set. We calculate attack success rate (ASR) on harmful prompts, false refusal rate (FRR) on benign prompts, and F1 score of (1-ASR) and (1-FRR). As shown in Table 2, using LPM as a conditioning mechanism substantially reduces false refusal rates on safe prompts, with only a marginal increase in the generation of harmful content. These results highlight the possibility of utilizing LPM as either an input filter or as a conditioning mechanism for output moderation while exceeding the performance of conditioning proposed by Lee et al. (2025). For this experiment, we set the LPM decision threshold to a standard value of 0.5. However, it can be easily tuned to meet specific performance requirements where either ASR or FRR is more important, which further highlights the flexibility of our method.

Table 3: Safety assessment F1 obtained with LPM for different Qwen3 models, using either the end of prompt token or end of thinking token embedding. We denote Instruct with -I suffix, and thinking models with -T suffix, while the A3B refers to 3B activate parameters in the MoE model.

Model	1.7B	30B-A3B-I	30B-A3B-T	4B-I	4B-T	8B	14B
LPM(Last)	84.95	87.60	87.40	86.50	86.33	86.21	87.64
LPM(Think)	83.54	-	86.24	-	86.89	87.09	88.18

4.4 LPM OUTSIDE INSTRUCTION MODELS

Our initial analysis focuses on instruction-finetuned models, which are typically safety-aligned and more likely to encode safety-relevant information in their latent spaces. In this section, we want to evaluate LPM on two model families, reasoning and base (pre-trained) models.

Pretrained vs instruction-tuned models for LPM. To better understand at what stage of model development this safety understanding begins to emerge, and if instruction tuning is necessary for our method to be effective. To investigate this, we compare the performance of LPM on both pretrained and instruction-finetuned models. In Figure 3, we report the F1 score of the two model types using LPM. Interestingly, the performance gap is significantly smaller on WildGuardMix, suggesting that pretrained base models can provide sufficient quality representations for examples similar to the ones used for prototype calculation. However, the gap increases substantially for other datasets, indicating that instruction tuning plays a critical role in enabling generalization to novel safety categories. While LPM can be applied to pretrained models, the results confirm that instruction-finetuned models are better suited as base models for our method.

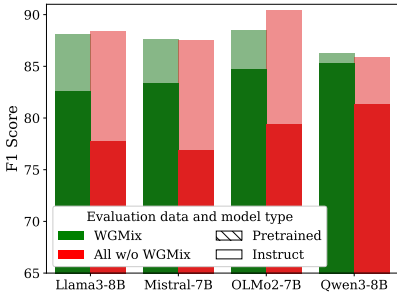


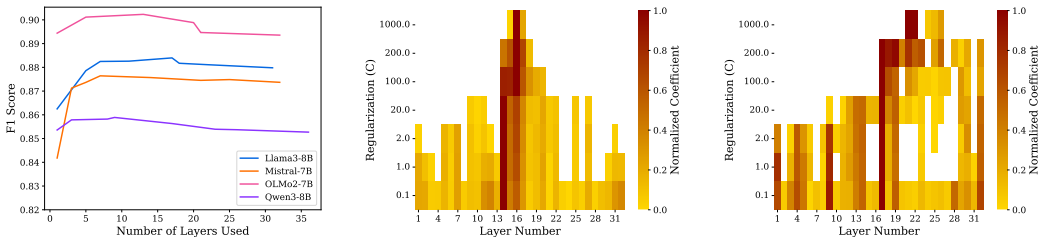
Figure 3: Performance of LPM applied to instruction-finetuned and pretrained models, evaluated separately on WildGuardMix and all the other datasets.

LPM with reasoning models. We investigate whether LPM’s effectiveness extends to reasoning models and whether leveraging the tokens from the thinking chain provides additional safety signals. Specifically, we compare LPM applied to the representation of the last prompt token, as in the previous experiments, to the performance of LPM applied to the final end-of-thinking token. We conduct experiments on several Qwen3 models, and show the safety assessment F1 on WildGuardMix in Table 3. To provide a comparison with instruction-tuned models used in our previous experiments, we also include results for Qwen3-4B-Instruct and Qwen3-30B-A3B-Instruct. While using the end-of-thinking token for LPM yields a slight performance improvement (up to 1%) for models larger than 4B, this comes at the cost of generating a long chain of reasoning tokens. In contrast, using LPM with the last prompt token already provides a robust safety assessment for the reasoning model, and for smaller models, the last token representation yields better performance. Our findings suggest that the essential safety signal is largely present in the initial prompt representation, and the reasoning process does not significantly enhance the performance of our method. Practically, this shows a test-time scaling capability of LPM when applied to the moderation of the reasoning models and further underlines the flexibility of our method: the user can trade off additional computation for slight performance gains or opt for a slightly weaker but substantially cheaper prompt-only evaluation.

4.5 WHICH LAYERS ARE THE MOST IMPORTANT FOR SAFETY ASSESSMENT?

Our use of a logistic regression model with an l_1 penalty provides a way for identifying the most influential layers for final predictions, as the penalty encourages sparse solutions by driving the aggregation weights of less important layers to zero. We train those weights across a spectrum of penalty strengths (C) to observe how layer selection evolves from stricter to more tolerant regularization. The resulting layer weights $W = \{w_1, \dots, w_L\}$ are subsequently normalized to assess their relative importance, using the formula $\hat{w}_i = |w_i|/\max_j |W_j|$.

378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431



(a) F1 against number of layers. (b) Selected layers for Mistral-7B. (c) Selected layers for OLMo2-7B.

Figure 4: a) LPM achieves best performance with strong regularization, resulting in using only a few layers for the final prediction. b-c) Normalized aggregation weights w for Mistral and OLMo models depending on the regularization C strength. We notice that for each model, selected layers and their importance differ, e.g., LPM trained on Mistral mostly utilizes middle layers, whereas OLMo2 starts using later layers for detection with higher regularization strength than Mistral.

As illustrated in Figure 4, this analysis reveals that the behavior of the normalized weights for the LPM differs significantly between the Mistral and OLMo architectures. For Mistral, the LPM initially prioritizes middle layers, suggesting that these layers are the most critical for capturing the essential features for the safety assessment. Conversely, the LPM for OLMo consistently utilizes the final layers. This preference holds even with a strong penalty of $C = 200$, which indicates a powerful and unambiguous signal from the top-most layers in the OLMo architecture.

4.6 HANDLING DIVERSE RISKS USING MULTIPLE PROTOTYPES

Safety risks can be highly diverse, making adaptability to shifting data distributions a critical feature for moderation algorithms. In Section 3.2, we describe how LPM supports extensibility to diverse data types by leveraging multiple prototypes. To evaluate how well LPM adapts in such scenarios, we conduct experiments utilizing prototypes across three distinct datasets. In Figure 5a, we show the average F1 score across all three datasets (WildGuardMix, Aegis, and ToxicChat) for LPM utilizing multiple prototypes with aggregation weights calculated from different dataset combinations. Incorporating prototypes from Aegis and ToxicChat improves the performance, which highlights that our approach allows for extensibility to diverse data.

4.7 DETAILED ANALYSIS

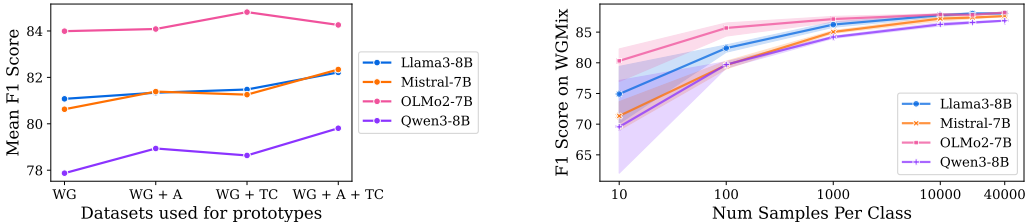
Selecting number of layers. We analyzed how the number of layers used by LPM affects its performance. This is controlled by a regularization parameter, C , which determines the subset of layers the model aggregates. Figure 4a illustrates the average performance across all harmful datasets, plotted against the number of selected layers. The results show a clear trend: for all models, optimal performance is consistently achieved when using a subset of layers, typically more than one, but less than half of the total available. This highlights the importance of strategic layer selection, rather than using all available layers, to ensure the method generalizes well.

Data efficiency of LPM. We also examine how the number of examples per class affects LPM performance, which is especially relevant for low-resource settings. Figure 5b shows the average performance across five random subsamples of WildGuardMix at varying sizes. Our method remains reasonably accurate even with a few examples, and the variance drops significantly as the sample size increases. These results demonstrate LPM’s practicality in data-scarce scenarios and prove that it can be extended to new, specific threats with a few examples, which could even be hand-curated. **In Appendix F we also show that our method has better scaling properties than simpler alternatives.**

LPM components analysis. We performed an ablation study to isolate the contributions of LPM’s two components: the distance metric (Mahalanobis vs. Euclidean) and the source of representations (multi-layer vs. last-layer). The results, presented in Table 4, show that both components are crucial for optimal performance. Switching from Euclidean to Mahalanobis distance provides the highest

Table 4: Average F1 score across all harmful datasets with LPM when varying the distance metric and using last-layer or multi-layer representation. Both the Mahalanobis distance and multi-layer representations improve the performance of LPM, validating our design choices.

Distance	Prototypes	Llama-8B	Mistral-7B	OLMo2-7B	Qwen3-8B
Euclidean	Last layer	77.17	77.39	83.61	74.97
Euclidean	Multi-layer	83.73	80.38	85.74	78.96
Mahalanobis	Last layer	86.25	84.18	89.44	85.36
Mahalanobis	Multi-layer	88.30	87.55	90.18	85.95



(a) LPM with separate prototypes on multiple datasets. (b) Performance depending on the number of samples.

Figure 5: a) LPM allows for easy extensibility to new moderation tasks via multiple prototypes. On the y-axis, we show the average score on Aegis, ToxicChat, and WildGuardMix when using combinations of prototypes derived from these datasets. b) LPM performs well even in limited data settings, offering reasonable effectiveness when only a small set of examples is available.

performance boost across all models. While also beneficial, the move from single-layer to multi-layer representations offers a secondary, but still important, improvement. This confirms that the combination of Mahalanobis distance and multi-layer aggregation is essential to LPM’s effectiveness. We provide further ablation on using last token vs mean over tokens in Appendix H.

5 DISCUSSION

LPM Design and Efficiency. LPM is designed to be a highly efficient input moderation tool, minimizing both computational and memory overhead. Compared to traditional guard models, LPM can be applied to any off-the-shelf open-weight LLM without the substantial GPU resources required for fine-tuning a separate model and does not require hosting a separate model for moderation, providing more of a self-contained solution. Its efficiency also supports long-term deployment: LPM can be easily adapted to shifting data distributions and emerging safety threats, which is aided by both its low training cost and the ability to extend via multiple prototypes.

Training cost. As LPM utilizes last token representation, the training process is notably lightweight, requiring only a single forward pass through the prompts dataset without gradient calculations or any text generation. For the calculation of prototypes and aggregation weights, LPM only uses last token representations, allowing this second part to be done even on a CPU in a reasonable time.

Inference overhead. Since LPM is designed to be highly efficient and is applied only to one final prompt token, its computational cost is negligible during the LLM response generation. We can estimate the computational overhead of LPM during inference by analyzing the ratio of FLOPs required for safety assessment with our method relative to the overall computation of the model that happens during the prefill stage. Following Hoffmann et al. (2022), we estimate this ratio as:

$$\frac{FLOPs_{LPM}}{FLOPs_{Prefill}} = \frac{\hat{N} \cdot c}{s \cdot (2v + N \cdot (3d_{intermediate} + 4d_{model}))}, \tag{5}$$

where \hat{N} denotes the number of layers used by our method, s is the total prompt length, v is the model vocabulary size, d_{model} and $d_{intermediate}$ are the model dimensions, and c is the number of prototypes

486 used for moderation (in the simplest case $c = 2$, distinguishing only between safe and unsafe inputs).
487 Since c is small and \hat{N} is at most the total number of layers, the overhead is negligible, amounting to
488 less 0.001% of the prefill FLOPs in the worst cases. See Appendix K for the estimation details.
489

490 **Memory overhead.** In addition to its low computational cost, LPM is highly memory-efficient.
491 The total number of parameters is given by $\hat{N} \cdot (d_{\text{model}}^2 + d_{\text{model}} + 1)$, which accounts for the \hat{N}
492 covariance matrices (Σ), prototypes (μ), and aggregation weights (w) that our method stores.
493

494 **Limitations.** LPM leverages the internal representations of post-trained LLMs, and its effectiveness
495 depends on how well these models encode distinctions between safe and harmful content in their
496 latent spaces. Nevertheless, our experiments show that LPM performs reliably even on models
497 without dedicated safety alignment. While primarily designed for input moderation, LPM can also be
498 integrated with existing output moderation techniques to strengthen the overall moderation pipeline.
499 However, similar to existing pure input-moderation approaches, LPM cannot distinguish between
500 prompts with several nested intents and would flag inputs containing both harmful and harmless
501 requests as unsafe, so we advise calibrating the refusal threshold to meet the desired performance
502 needs in end-to-end applications.
503

504 6 CONCLUSIONS

506 In this work, we present the LPM, a novel, highly efficient input moderation method aimed at
507 improving the safety of LLMs. Building on previous works and our observations, we introduce LPM,
508 a simple yet powerful, efficient input moderation method that leverages latent representations from
509 multiple LLM layers to assess the safety of prompts using Mahalanobis distance from prototypical
510 safe and unsafe examples. Our method is model-agnostic, incurs negligible computational overhead,
511 and can be seamlessly integrated into LLM output moderation pipelines. LPM delivers robust
512 performance across a wide range of diverse safety benchmarks, surpassing current state-of-the-art
513 methods, incurring negligible computational overhead and training costs. Furthermore, its efficiency
514 allows for rapid adaptation to new risk types without expensive retraining, making it a truly scalable
515 and practical solution for evolving moderation needs. LPM represents a step toward efficient and
516 flexible moderation tools for real-world LLM deployment.
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539

540 **Reproducibility statement.** To ensure reproducibility of our research, we provide the code
 541 and all scripts necessary to run our experiments at [https://anonymous.4open.science/](https://anonymous.4open.science/r/latent-prototype-moderator-20B6/)
 542 [r/latent-prototype-moderator-20B6/](https://anonymous.4open.science/r/latent-prototype-moderator-20B6/) All the experiments were conducted using
 543 A100 (40GB) GPUs.
 544

545 **Ethics statement.** We aim to advance machine learning research toward safer large language
 546 models. While we do not identify any specific ethical concerns with our method, we acknowledge
 547 that, like any technique, it could be misused if the user steers it toward harmful objectives. To comply
 548 with the ICLR LLM policy, we hereby disclose that we used LLMs to polish the writing of this
 549 manuscript
 550

551 REFERENCES

552 Sahar Abdelnabi, Aideen Fay, Giovanni Cherubin, Ahmed Salem, Mario Fritz, and Andrew Pavard.
 553 Get my drift? catching llm task drift with activation deltas. In *2025 IEEE Conference on Secure*
 554 *and Trustworthy Machine Learning (SaTML)*, pp. 43–67. IEEE, 2025.
 555

556 Maksym Andriushchenko, Francesco Croce, and Nicolas Flammarion. Jailbreaking leading safety-
 557 aligned LLMs with simple adaptive attacks. In *The Thirteenth International Conference on*
 558 *Learning Representations*, 2025.
 559

560 Md. Ahsan Ayub and Subhabrata Majumdar. Embedding-based classifiers can detect prompt injection
 561 attacks. In *CAMLIS*, 2024.

562 Ge Bai, Jie Liu, Xingyuan Bu, Yancheng He, Jiaheng Liu, Zhanhui Zhou, Zhuoran Lin, Wenbo
 563 Su, Tiezheng Ge, Bo Zheng, and Wanli Ouyang. Mt-bench-101: A fine-grained benchmark
 564 for evaluating large language models in multi-turn dialogues. In Lun-Wei Ku, Andre Martins,
 565 and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for*
 566 *Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16,*
 567 *2024*, pp. 7421–7454. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.
 568 ACL-LONG.401. URL <https://doi.org/10.18653/v1/2024.acl-long.401>.

569 Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn
 570 Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, Nicholas Joseph, Saurav Kadavath, Jackson
 571 Kernion, Tom Conerly, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez,
 572 Tristan Hume, Scott Johnston, Shauna Kravec, Liane Lovitt, Neel Nanda, Catherine Olsson, Dario
 573 Amodei, Tom Brown, Jack Clark, Sam McCandlish, Chris Olah, Ben Mann, and Jared Kaplan.
 574 Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv*
 575 *preprint arXiv: 2204.05862*, 2022.

576 Nicholas Carlini and et al. Aligned but unsafe: Investigation of jailbreaks in ai systems. *arXiv*
 577 *preprint arXiv:2305.18526*, 2023.
 578

579 Christopher Clarke, Matthew Hall, Gaurav Mittal, Ye Yu, Sandra Sajejev, Jason Mars, and Mei Chen.
 580 Rule by example: Harnessing logical rules for explainable hate speech detection. In *Proceedings*
 581 *of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long*
 582 *Papers)*, pp. 364–376, 2023.

583 Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser,
 584 Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John
 585 Schulman. Training verifiers to solve math word problems. *CoRR*, abs/2110.14168, 2021. URL
 586 <https://arxiv.org/abs/2110.14168>.

587 Josef Dai, Xuehai Pan, Ruiyang Sun, Jiaming Ji, Xinbo Xu, Mickel Liu, Yizhou Wang, and Yaodong
 588 Yang. Safe rlhf: Safe reinforcement learning from human feedback. In *The Twelfth International*
 589 *Conference on Learning Representations*, 2024. URL [https://openreview.net/forum?](https://openreview.net/forum?id=TyFrPOKYXw)
 590 [id=TyFrPOKYXw](https://openreview.net/forum?id=TyFrPOKYXw).
 591

592 Yi Dong, Ronghui Mu, Gaojie Jin, Yi Qi, Jinwei Hu, Xingyu Zhao, Jie Meng, Wenjie Ruan, and
 593 Xiaowei Huang. Building guardrails for large language models. *arXiv preprint arXiv:2402.01822*,
 2024.

- 594 Aryo Pradipta Gema, Joshua Ong Jun Leang, Giwon Hong, Alessio Devoto, Alberto Carlo Maria
595 Mancino, Rohit Saxena, Xuanli He, Yu Zhao, Xiaotang Du, Mohammad Reza Ghasemi Madani,
596 et al. Are we done with mmlu? *arXiv preprint arXiv:2406.04127*, 2024.
- 597 Shaona Ghosh, Prasoon Varshney, Erick Galinkin, and Christopher Parisien. Aegis: Online adaptive
598 ai content safety moderation with ensemble of llm experts. *arXiv preprint arXiv:2404.05993*, 2024.
- 600 Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad
601 Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of
602 models. *arXiv preprint arXiv:2407.21783*, 2024.
- 603 Seungju Han, Kavel Rao, Allyson Ettinger, Liwei Jiang, Bill Yuchen Lin, Nathan Lambert, Yejin
604 Choi, and Nouha Dziri. Wildguard: Open one-stop moderation tools for safety risks, jailbreaks,
605 and refusals of llms. In *Advances in Neural Information Processing Systems 38, NeurIPS*, 2024.
- 607 Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Xiaodong Song,
608 and Jacob Steinhardt. Measuring massive multitask language understanding. *ArXiv*, 2020.
- 609 Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza
610 Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al.
611 Training compute-optimal large language models. In *Proceedings of the 36th International
612 Conference on Neural Information Processing Systems*, pp. 30016–30030, 2022.
- 613 Tiansheng Huang, Sihao Hu, Fatih Ilhan, Selim Furkan Tekin, Zachary Yahn, Yichang Xu, and Ling
614 Liu. Safety tax: Safety alignment makes your large reasoning models less reasonable. *arXiv
615 preprint arXiv:2503.00555*, 2025.
- 617 Hakan Inan, Kartikeya Upasani, Jianfeng Chi, Rashi Rungta, Krithika Iyer, Yuning Mao, Michael
618 Tontchev, Qing Hu, Brian Fuller, Davide Testuggine, and Madian Khabsa. Llama guard: Llm-based
619 input-output safeguard for human-ai conversations. *arXiv preprint arXiv: 2312.06674*, 2023.
- 620 Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot,
621 Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al.
622 Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.
- 623 Liwei Jiang, Kavel Rao, Seungju Han, Allyson Ettinger, Faeze Brahman, Sachin Kumar, Niloofar
624 Mireshghallah, Ximing Lu, Maarten Sap, Yejin Choi, and Nouha Dziri. Wildteaming at scale:
625 From in-the-wild jailbreaks to (adversarially) safer language models. *ArXiv*, 2024.
- 627 Ananya Kamath, Shiyu Sun, Xiaoyang Gu, Rui Zhang, Yixuan Liu, Matthias Hein, and Yix-
628 uan Li. Openood: Benchmarking generalized out-of-distribution detection. *arXiv preprint
629 arXiv:2302.12229*, 2023.
- 630 Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott
631 Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models.
632 *arXiv preprint arXiv:2001.08361*, 2020.
- 633 Tatsuya Kubokawa and Muni S Srivastava. Estimation of the precision matrix of a singular wishart
634 distribution and its application in high-dimensional data. *Journal of multivariate Analysis*, 99(9):
635 1906–1928, 2008.
- 637 Deepak Kumar, Yousef AbuHashem, and Zakir Durumeric. Watch your language: Investigating
638 content moderation with large language models. In Yu-Ru Lin, Yelena Mejova, and Meeyoung Cha
639 (eds.), *Proceedings of the Eighteenth International AAAI Conference on Web and Social Media*,
640 *ICWSM 2024, Buffalo, New York, USA, June 3-6, 2024*, pp. 865–878. AAAI Press, 2024. doi: 10.
641 1609/ICWSM.V18I1.31358. URL <https://doi.org/10.1609/icwsm.v18i1.31358>.
- 642 Bruce W. Lee, Inkit Padhi, Karthikeyan Natesan Ramamurthy, Erik Miehl, Pierre Dognin, Manish
643 Nagireddy, and Amit Dhurandhar. Programming refusal with conditional activation steering.
644 In *The Thirteenth International Conference on Learning Representations*, 2025. URL <https://openreview.net/forum?id=Oi47wc10sm>.
- 645
646
647 Shen Li, Liuyi Yao, Lan Zhang, and Yaliang Li. Safety layers in aligned large language models: The
key to llm security. *arXiv preprint arXiv:2408.17003*, 2024.

- 648 Isaac Lim, Shaun Khoo, Watson Chua, Goh Jiayi, and Jessica Foo. Safe at the margins: A general
649 approach to safety alignment in low-resource english languages—a singlish case study. *arXiv*
650 *preprint arXiv:2502.12485*, 2025.
- 651 Stephanie Lin, Jacob Hilton, and Owain Evans. Truthfulqa: Measuring how models mimic human
652 falsehoods. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (eds.), *Proceedings of*
653 *the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*,
654 *ACL 2022, Dublin, Ireland, May 22-27, 2022*, pp. 3214–3252. Association for Computational
655 Linguistics, 2022. doi: 10.18653/V1/2022.ACL-LONG.229. URL [https://doi.org/10.](https://doi.org/10.18653/v1/2022.acl-long.229)
656 [18653/v1/2022.acl-long.229](https://doi.org/10.18653/v1/2022.acl-long.229).
- 657 Zi Lin, Zihan Wang, Yongqi Tong, Yangkun Wang, Yuxin Guo, Yujia Wang, and Jingbo Shang.
658 Toxicchat: Unveiling hidden challenges of toxicity detection in real-world user-ai conversation. In
659 *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2023.
- 660 Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao,
661 Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint*
662 *arXiv:2412.19437*, 2024.
- 663 Jinqi Luo, Tianjiao Ding, Kwan Ho Ryan Chan, Darshan Thaker, Aditya Chattopadhyay, Chris
664 Callison-Burch, and René Vidal. Pace: Parsimonious concept engineering for large language
665 models. *arXiv preprint arXiv:2406.04331*, 2024.
- 666 Blazej Manczak, Eric Lin, Elliott Zemor, and Vaikkunth Mugunthan. Primeguard: Safe and helpful
667 llms through tuning-free routing. In *ICML 2024 Next Generation of AI Safety Workshop*, 2024.
- 668 Wojciech Masarczyk, Mateusz Ostaszewski, Ehsan Imani, Razvan Pascanu, Piotr Mił oś, and
669 Tomasz Trzcinski. The tunnel effect: Building data representations in deep neural networks.
670 In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Ad-*
671 *vances in Neural Information Processing Systems*, volume 36, pp. 76772–76805. Curran Asso-
672 ciates, Inc., 2023. URL [https://proceedings.neurips.cc/paper_files/paper/](https://proceedings.neurips.cc/paper_files/paper/2023/file/f249db9ab5975586f36df46f8958c008-Paper-Conference.pdf)
673 [2023/file/f249db9ab5975586f36df46f8958c008-Paper-Conference.pdf](https://proceedings.neurips.cc/paper_files/paper/2023/file/f249db9ab5975586f36df46f8958c008-Paper-Conference.pdf).
- 674 Mantas Mazeika, Long Phan, Xuwang Yin, Andy Zou, Zifan Wang, Norman Mu, Elham Sakhaee,
675 Nathaniel Li, Steven Basart, Bo Li, et al. Harmbench: a standardized evaluation framework for
676 automated red teaming and robust refusal. In *Proceedings of the 41st International Conference on*
677 *Machine Learning*, pp. 35181–35224, 2024.
- 678 Team OLMo, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia,
679 Yuling Gu, Shengyi Huang, Matt Jordan, Nathan Lambert, Dustin Schwenk, Oyvind Tafjord, Taira
680 Anderson, David Atkinson, Faeze Brahman, Christopher Clark, Pradeep Dasigi, Nouha Dziri,
681 Michal Guerquin, Hamish Ivison, Pang Wei Koh, Jiacheng Liu, Saumya Malik, William Merrill,
682 Lester James V. Miranda, Jacob Morrison, Tyler Murray, Crystal Nam, Valentina Pyatkin, Aman
683 Rangapur, Michael Schmitz, Sam Skjonsberg, David Wadden, Christopher Wilhelm, Michael
684 Wilson, Luke Zettlemoyer, Ali Farhadi, Noah A. Smith, and Hannaneh Hajishirzi. 2 olmo 2 furious,
685 2024. URL <https://arxiv.org/abs/2501.00656>.
- 686 Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong
687 Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow
688 instructions with human feedback. *Advances in neural information processing systems*, 35:27730–
689 27744, 2022.
- 690 Inkit Padhi, Manish Nagireddy, Giandomenico Cornacchia, Subhajt Chaudhury, Tejaswini Pedapati,
691 Pierre Dognin, Keerthiram Murugesan, Erik Miehl, Martín Santillán Cooper, Kieran Fraser,
692 Giulio Zizzo, Muhammad Zaid Hameed, Mark Purcell, Michael Desmond, Qian Pan, Zahra
693 Ashktorab, Inge Vejsbjerg, Elizabeth M. Daly, Michael Hind, Werner Geyer, Ambrish Rawat,
694 Kush R. Varshney, and Prasanna Sattigeri. Granite guardian, 2024. URL [https://arxiv.](https://arxiv.org/abs/2412.07724)
695 [org/abs/2412.07724](https://arxiv.org/abs/2412.07724).
- 696 Xiangyu Qi, Ashwinee Panda, Kaifeng Lyu, Xiao Ma, Subhrajit Roy, Ahmad Beirami, Prateek
697 Mittal, and Peter Henderson. Safety alignment should be made more than just a few tokens
698 deep. In *The Thirteenth International Conference on Learning Representations*, 2025. URL
699 <https://openreview.net/forum?id=6Mxhg9PtDE>.

- 702 Yifu Qiu, Zheng Zhao, Yftah Ziser, Anna Korhonen, Edoardo M Ponti, and Shay B Cohen. Spectral
703 editing of activations for large language model alignment. *arXiv preprint arXiv:2405.09719*, 2024.
704
- 705 Paul Röttger, Hannah Kirk, Bertie Vidgen, Giuseppe Attanasio, Federico Bianchi, and Dirk Hovy.
706 Xstest: A test suite for identifying exaggerated safety behaviours in large language models.
707 In *Proceedings of the 2024 Conference of the North American Chapter of the Association for*
708 *Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 5377–
709 5400, 2024.
- 710 Mrinank Sharma, Meg Tong, Jesse Mu, Jerry Wei, Jorrit Kruthoff, Scott Goodfriend, Euan Ong,
711 Alwin Peng, Raj Agarwal, Cem Anil, Amanda Askell, Nathan Bailey, Joe Benton, Emma Bluemke,
712 Samuel R. Bowman, Eric Christiansen, Hoagy Cunningham, Andy Dau, Anjali Gopal, Rob Gilson,
713 Logan Graham, Logan Howard, Nimit Kalra, Taesung Lee, Kevin Lin, Peter Lofgren, Francesco
714 Mosconi, Clare O’Hara, Catherine Olsson, Linda Petrini, Samir Rajani, Nikhil Saxena, Alex
715 Silverstein, Tanya Singh, Theodore Summers, Leonard Tang, Kevin K. Troy, Constantin Weisser,
716 Ruiqi Zhong, Giulio Zhou, Jan Leike, Jared Kaplan, and Ethan Perez. Constitutional classifiers:
717 Defending against universal jailbreaks across thousands of hours of red teaming, 2025.
- 718 Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and
719 Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In
720 *International Conference on Learning Representations*, 2017.
- 721 Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung,
722 Aakanksha Chowdhery, Quoc V. Le, Ed H. Chi, Denny Zhou, and Jason Wei. Challenging big-
723 bench tasks and whether chain-of-thought can solve them. In Anna Rogers, Jordan L. Boyd-Graber,
724 and Naoaki Okazaki (eds.), *Findings of the Association for Computational Linguistics: ACL 2023,*
725 *Toronto, Canada, July 9-14, 2023*, pp. 13003–13051. Association for Computational Linguistics,
726 2023. doi: 10.18653/V1/2023.FINDINGS-ACL.824. URL [https://doi.org/10.18653/
727 v1/2023.findings-acl.824](https://doi.org/10.18653/v1/2023.findings-acl.824).
- 728 Filip Szatkowski, Yaoyue Zheng, Fei Yang, Tomasz Trzcinski, Bartłomiej Twardowski, and
729 Joost van de Weijer. Improving continual learning performance and efficiency with auxil-
730 iary classifiers. In *Forty-second International Conference on Machine Learning*, 2025. URL
731 <https://openreview.net/forum?id=sq5eL4jfsn>.
- 732 Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy
733 Liang, and Tatsunori B. Hashimoto. Stanford alpaca: An instruction-following llama model.
734 https://github.com/tatsu-lab/stanford_alpaca, 2023.
735
- 736 Bertie Vidgen, Hannah Rose Kirk, Rebecca Qian, Nino Scherrer, Anand Kannappan, Scott A Hale,
737 and Paul Röttger. Simple safety tests: a test suite for identifying critical safety risks in large language
738 models. *CoRR*, 2023.
- 739 Yue Wang, Weishi Wang, Shafiq Joty, and Steven C.H. Hoi. Codet5: Identifier-aware unified pre-
740 trained encoder-decoder models for code understanding and generation. In *Proceedings of the*
741 *2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021*, 2021.
- 742 Alexander Wei, Nika Haghtalab, and Jacob Steinhardt. Jailbroken: How does llm safety training fail?
743 *Advances in Neural Information Processing Systems*, 36:80079–80110, 2023.
744
- 745 Jason Wei and et al. Jailbreaking large language models via prompt engineering. *arXiv preprint*
746 *arXiv:2303.16200*, 2023.
- 747 Yotam Wolf, Noam Wies, Dorin Shteyman, Binyamin Rothberg, Yoav Levine, and Amnon Shashua.
748 Tradeoffs between alignment and helpfulness in language models with representation engineering.
749 *arXiv preprint arXiv:2401.16332*, 2024.
- 750 Yueqi Xie, Jingwei Yi, Jiawei Shao, Justin Curl, Lingjuan Lyu, Qifeng Chen, Xing Xie, and Fangzhao
751 Wu. Defending chatgpt against jailbreak attack via self-reminders. *Nature Machine Intelligence*, 5:
752 1486–1496, 2023.
753
- 754 An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang
755 Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*,
2025.

756 Fan Yin, Philippe Laban, Xiangyu Peng, Yilun Zhou, Yixin Mao, Vaibhav Vats, Linnea Ross,
757 Divyansh Agarwal, Caiming Xiong, and Chien-Sheng Wu. Bingoguard: Llm content moderation
758 tools with risk levels. In *The Thirteenth International Conference on Learning Representations*,
759 2025.

760 Wenjun Zeng, Yuchi Liu, Ryan Mullins, Ludovic Peran, Joe Fernandez, Hamza Harkous, Karthik
761 Narasimhan, Drew Proud, Piyush Kumar, Bhaktipriya Radharapu, et al. Shieldgemma: Generative
762 ai content moderation based on gemma. *arXiv preprint arXiv:2407.21772*, 2024.

763
764 Yihao Zhang, Zeming Wei, Jun Sun, and Meng Sun. Adversarial representation engineering: A
765 general model editing framework for large language models. *Advances in Neural Information*
766 *Processing Systems*, 37:126243–126264, 2024.

767
768 Chujie Zheng, Fan Yin, Hao Zhou, Fandong Meng, Jie Zhou, Kai-Wei Chang, Minlie Huang, and
769 Nanyun Peng. On prompt-driven safeguarding for large language models. In *Proceedings of the*
770 *41st International Conference on Machine Learning*, pp. 61593–61613, 2024.

771
772 Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan,
773 Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, et al. Representation engineering: A
774 top-down approach to ai transparency. *arXiv preprint arXiv:2310.01405*, 2023a.

775
776 Andy Zou, Zifan Wang, J. Zico Kolter, and Matt Fredrikson. Universal and transferable adversarial
777 attacks on aligned language models, 2023b.

778
779 Andy Zou, Long Phan, Justin Wang, Derek Duenas, Maxwell Lin, Maksym Andriushchenko, J. Zico
780 Kolter, Matt Fredrikson, and Dan Hendrycks. Improving alignment and robustness with circuit
781 breakers. In *Advances in Neural Information Processing Systems 38, NeurIPS 2024*, 2024.

782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809

APPENDIX

A EVALUATION DATASETS

As mentioned in Section 4, we split evaluation datasets into 2 groups: *prompt harmfulness* - 8 datasets, and *general capabilities* - 7 datasets, which we refer to as harmful and neutral, respectively. This split helps us assess our method’s effectiveness at detecting harmful content in prompts, while ensuring that neutral, non-harmful data remains correctly labeled to preserve the model’s original capabilities during moderation.

For prompt harmfulness we use Aegis (Ghosh et al., 2024), HarmBench (Mazeika et al., 2024), OpenAI Mod (Kamath et al., 2023), Simple Safety Tests (Vidgen et al., 2023), Toxic Chat (Lin et al., 2023), XSTest (Röttger et al., 2024), WildGuardMix (Han et al., 2024) and WildJailbreak (Jiang et al., 2024).

To ensure proper generalization, we evaluate the true negative rate on neutral datasets, including Alpaca (Taori et al., 2023), BigBenchHard (Suzgun et al., 2023), Codex (Wang et al., 2021), GSM8k (Cobbe et al., 2021), MMLU (Gema et al., 2024; Hendrycks et al., 2020), MTBench (Bai et al., 2024), and TruthfulQA (Lin et al., 2022).

In Table 5, we provide the distribution of neutral and harmful samples in each of the datasets. We also describe each dataset in more detail in the next subsections.

Table 5: Distribution of harmful and neutral prompts in evaluation datasets.

Dataset	Num Neutral Prompts	Num Harmful Prompts	Num Prompts
Aegis	126	233	359
HarmBench	0	239	239
OpenAI Mod	1158	522	1680
Simple Safety Tests	0	100	100
Toxic Chat	2491	362	2853
WildGuardMix Test	945	754	1699
WildJailbreak	210	2000	2210
XSTest	249	197	446
Alpaca	805	0	805
BigBenchHard	1080	0	1080
Codex	164	0	164
GSM8k	1319	0	1319
MMLU-R	2744	0	2744
MTBench	80	0	80
TruthfulQA	790	0	790

A.1 HARMFUL DATASETS

Aegis: This dataset comprises human-LLM interaction instances, each annotated for safety based on an extensive content safety risk taxonomy spanning 13 categories. Aegis is designed to benchmark and enhance the safety of Large Language Models (LLMs), particularly in the context of content moderation. All included responses were generated using Mistral-7B-v0.1.

HarmBench: HarmBench is an evaluation dataset comprising harmful prompts that can elicit harmful behaviors of LLMs.

OpenAIMod: This dataset features prompts, each accompanied by a harm label, spanning eight defined risk categories.

Simple Safety Tests: This is a concise test suite featuring 100 prompts across five distinct harm areas, designed for the rapid identification of critical safety risks within LLMs.

864 **Toxic Chat:** This benchmark dataset is constructed from real user queries submitted to an open-source
865 chatbot. The collected samples have been annotated for toxicity through a human-AI collaborative
866 annotation framework.

867 **WildGuardMix:** This dataset offers a diverse collection of both standard (vanilla) and adversarial
868 prompts, encompassing harmful and benign scenarios, accompanied by LLM-generated responses.
869

870 **WildJailbreak:** An open-source synthetic safety-training dataset, WildJailbreak contains prompt-
871 response pairs. It features a mix of vanilla (direct harmful requests) and adversarial (complex
872 jailbreaks) queries. The dataset also includes contrastive benign queries that resemble harmful ones,
873 aiming to mitigate exaggerated safety behaviors in LLMs. For evaluation, a test set composed entirely
874 of adversarial prompts is utilized.

875 **XSTest** A test suite designed to identify "exaggerated safety behaviors" in LLMs. This refers to
876 instances where models refuse safe prompts if they contain language similar to unsafe prompts or
877 mention sensitive topics.

878

879 A.2 NEUTRAL DATASETS

880 **Alpaca:** This dataset features instructions generated by OpenAI's text-davinci-003 model. Covering
881 diverse domains, these instructions are widely utilized for fine-tuning Large Language Models (LLMs)
882 to enhance their ability to follow instructions.
883

884 **BigBenchHard:** A challenging subset of the BIG-bench benchmark, BigBenchHard comprises 23
885 tasks specifically selected because current language models find them particularly difficult.

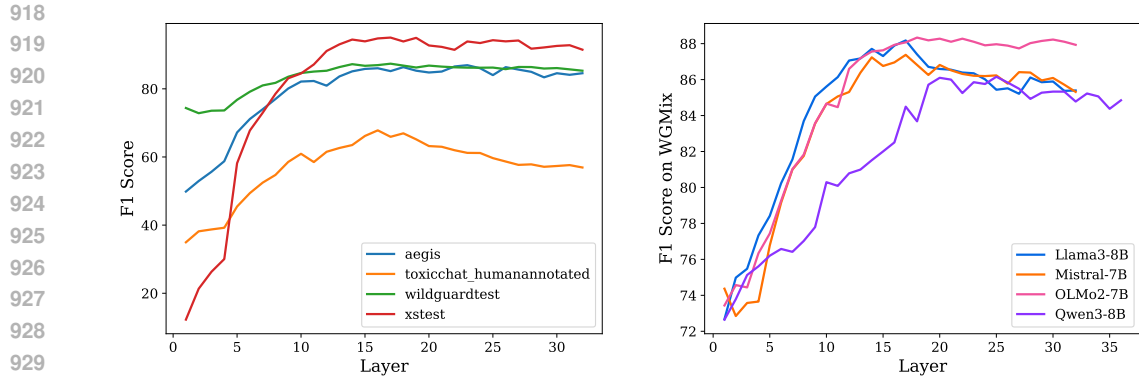
886 **Codex:** This is a dataset containing a collection of code-related instructions specifically curated for
887 training and evaluating LLMs on programming tasks.

888 **GSM8k:** GSM8k is a dataset of high-quality, linguistically diverse grade school math word problems,
889 designed to test multi-step reasoning.
890

891 **MMLU:** This benchmark is designed to evaluate the knowledge acquired by language models across
892 an extensive array of 57 distinct tasks. These tasks span humanities, social sciences, STEM, and
893 other areas, offering a comprehensive measure of a model's understanding. We utilize MMLU-
894 Redux (Gema et al., 2024), which is a subset of manually re-annotated questions across 30 MMLU
895 subjects.

896 **MTBench:** MTBench is a benchmark composed of multi-turn questions, specifically designed to
897 evaluate the conversational and instruction-following capabilities of chat-focused LLMs. In our
898 experiments, we focus solely on the initial instruction of each interaction.

899 **TruthfulQA:** This benchmark is engineered to measure the truthfulness of a language model when
900 generating answers to questions. It particularly focuses on questions where answers are prone to
901 common misconceptions or are frequently misremembered, thereby testing the model's ability to
902 avoid parroting falsehoods.
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917



(a) Moderation score across different datasets and layers. (b) Moderation score across different models and layers.

Figure 6: LLMs already contain information about input safety, but the layers at which harmful and safe examples are best separated depend on both the model and task. a) The performance across 4 different datasets when using representations from different layers. b) The performance across 4 models when using representations from different layers.

B WHICH LAYERS ARE THE BEST FOR SAFETY ASSESSMENT?

We additionally performed analyses to determine if there is a single best layer for safety evaluation for all models and datasets. In Figure 6 we show that there is no single best layer in terms of performance across multiple models and datasets.

C DETAILED RESULTS

In this section, we provide detailed results that compare the performance of LPM on more models and on Neutral datasets.

C.1 NEUTRAL DATASETS

In Table 6 we show the True Negative Rate on all Neutral datasets.

Table 6: Detailed results on **Neutral** datasets. For each dataset, we show the True Negative Rate, as these datasets have no harmful examples.

Model	Alpaca	BBH	Codex	GSM8k	MMLU	MTBench	TruthfulQA	Avg
Aegis-Guard-D	99.01	97.78	100.00	99.92	99.20	100.00	95.95	98.84
Aegis-Guard-P	99.50	98.43	100.00	99.92	99.89	100.00	97.34	99.30
LlamaGuard1	99.63	100.00	100.00	99.92	100.00	100.00	97.85	99.63
LlamaGuard2	99.13	100.00	100.00	100.00	94.97	98.75	99.24	98.87
LlamaGuard3	98.63	99.81	100.00	100.00	99.93	98.75	99.87	99.57
WildGuard	96.89	100.00	100.00	100.00	99.71	97.50	96.58	98.67
LPM(DeepSeek-Distill-Llama-8B)	95.40	100.00	100.00	100.00	48.72	98.75	97.34	91.46
LPM(DeepSeek-Qwen3-8B)	95.90	95.56	100.00	100.00	96.87	96.25	97.97	97.51
LPM(Llama-1B-Inst)	92.55	100.00	100.00	100.00	100.00	98.75	98.35	98.52
LPM(Llama-3B-Inst)	95.65	100.00	100.00	100.00	100.00	98.75	97.09	98.78
LPM(Llama-8B)	85.96	98.52	99.39	100.00	90.60	85.00	91.77	93.03
LPM(Llama-8B-Inst)	96.15	100.00	100.00	100.00	100.00	100.00	96.20	98.89
LPM(Llama-70B-Inst)	98.01	100.00	100.00	100.00	100.00	100.00	97.85	99.41
LPM(Mistral-7B)	87.33	88.24	98.78	100.00	94.50	92.50	91.14	93.21
LPM(Mistral-7B-Inst)	96.15	100.00	100.00	100.00	99.89	100.00	96.20	98.89
LPM(Mistral-12B-Inst)	94.41	100.00	100.00	100.00	100.00	96.25	95.95	98.09
LPM(Mistral-24B-Inst)	96.27	99.91	100.00	100.00	99.96	100.00	98.10	99.18
LPM(OLMo2-1B-Inst)	98.01	100.00	100.00	100.00	100.00	98.75	97.72	99.21
LPM(OLMoE-1B-7B-Inst)	98.01	100.00	100.00	100.00	99.96	100.00	97.47	99.35
LPM(OLMo2-7B)	89.19	98.61	95.73	99.92	96.83	83.75	97.59	94.52
LPM(OLMo2-7B-DPO)	98.51	100.00	100.00	100.00	100.00	98.75	96.84	99.16
LPM(OLMo2-7B-SFT)	98.63	100.00	100.00	100.00	99.96	98.75	97.59	99.28
LPM(OLMo2-7B-Inst)	98.51	100.00	100.00	100.00	100.00	98.75	97.22	99.21
LPM(OLMo2-13B-Inst)	97.76	100.00	100.00	100.00	99.74	98.75	98.35	99.23
LPM(OLMo2-32B-Inst)	98.51	100.00	100.00	100.00	100.00	100.00	97.59	99.44
LPM(Qwen3-0.6B)	92.92	100.00	100.00	100.00	100.00	96.25	96.46	97.95
LPM(Qwen3-1.7B)	90.19	100.00	100.00	100.00	99.96	95.00	98.35	97.64
LPM(Qwen3-4B)	92.42	100.00	100.00	100.00	100.00	97.50	98.35	98.32
LPM(Qwen3-4B-Base)	87.33	88.33	99.39	100.00	100.00	100.00	97.59	96.09
LPM(Qwen3-4B-Inst)	96.40	100.00	100.00	100.00	99.93	98.75	96.71	98.83
LPM(Qwen3-4B-Thinking)	96.40	100.00	100.00	100.00	100.00	98.75	98.23	99.05
LPM(Qwen3-8B-Inst)	96.77	100.00	100.00	100.00	100.00	100.00	98.61	99.34
LPM(Qwen3-8B-Base)	91.93	96.94	100.00	100.00	99.89	97.50	97.09	97.62
LPM(Qwen3-30B-A3B)	94.29	100.00	100.00	100.00	100.00	100.00	98.99	99.04
LPM(Qwen3-32B)	95.78	99.91	100.00	100.00	100.00	98.75	98.86	99.04
LPM(WildGuard)	96.89	100.00	100.00	99.92	99.78	97.50	95.82	98.56
LPM(LlamaGuard3-8B)	96.52	99.81	100.00	100.00	99.60	100.00	99.75	99.38

C.2 HARMFUL DATASETS

In Table 7 we show the F1 score on all harmful datasets.

Table 7: Detailed results on **Harmful** datasets. For each dataset, we show the F1 score.

Model	Aegis	HarmB	OpenAI	SimpST	ToxiChat	WGMix	WJ	XS	Avg
Aegis-Guard-D	81.00	70.46	76.44	97.96	75.61	72.09	75.44	81.53	78.82
Aegis-Guard-P	75.72	66.11	78.11	94.18	68.49	65.63	55.72	82.35	73.29
LlamaGuard1	72.92	66.11	74.38	92.47	57.14	55.08	41.36	81.61	67.63
LlamaGuard2	71.85	93.78	76.10	95.83	46.32	70.52	49.85	89.18	74.18
LlamaGuard3	71.74	98.94	79.11	99.50	54.11	76.76	67.83	88.52	79.56
WildGuard	89.78	99.37	72.28	99.50	70.14	88.04	97.10	95.26	88.93
LPM(DeepSeek-Distill-Llama-8B)	81.94	98.94	69.16	98.48	63.50	86.77	91.71	90.66	85.15
LPM(DeepSeek-Qwen3-8B)	81.82	95.63	71.26	98.99	62.84	84.87	92.43	88.64	84.56
LPM(Llama-1B-Inst)	82.73	98.08	69.96	98.48	62.24	85.08	90.50	91.60	84.83
LPM(Llama-3B-Inst)	85.91	100.00	73.27	99.50	67.48	87.93	93.63	96.92	88.08
LPM(Llama-8B)	82.38	97.42	59.76	95.29	48.27	82.55	84.80	76.28	78.34
LPM(Llama-8B-Inst)	85.13	99.58	72.85	99.50	69.17	88.04	94.69	97.44	88.30
LPM(Llama-70B-Inst)	88.99	100.00	76.57	100.00	72.89	89.39	97.41	98.73	90.50
LPM(Mistral-7B)	79.62	94.48	58.75	97.96	51.78	83.38	83.70	71.38	77.63
LPM(Mistral-7B-Inst)	87.36	99.16	70.68	99.50	66.33	87.63	93.65	96.10	87.55
LPM(Mistral-12B-Inst)	86.80	100.00	72.53	99.50	64.52	88.34	93.92	94.24	87.48
LPM(Mistral-24B-Inst)	84.49	100.00	70.88	99.50	66.80	88.29	94.79	95.01	87.47
LPM(OLMo2-1B-Inst)	83.97	99.58	69.35	98.99	67.18	88.16	93.76	92.43	86.68
LPM(OLMoE-1B-7B-Inst)	87.02	89.86	70.21	99.50	72.90	87.57	94.97	93.62	86.96
LPM(OLMo2-7B)	82.54	93.30	61.64	97.96	52.99	84.65	84.17	82.70	79.99
LPM(OLMo2-7B-DPO)	89.28	98.51	74.19	100.00	76.21	88.52	97.69	96.89	90.16
LPM(OLMo2-7B-SFT)	89.43	98.51	71.97	99.50	74.94	88.58	97.66	96.10	89.59
LPM(OLMo2-7B-Inst)	89.23	98.51	74.21	100.00	76.51	88.52	97.55	96.91	90.18
LPM(OLMo2-13B-Inst)	88.69	99.79	72.53	99.50	75.46	88.60	97.84	96.06	89.81
LPM(OLMo2-32B-Inst)	88.11	99.79	74.41	100.00	74.86	89.03	97.48	97.69	90.17
LPM(Qwen3-0.6B)	73.60	98.94	64.36	94.18	55.16	80.47	87.27	70.24	78.03
LPM(Qwen3-1.7B)	78.54	100.00	65.54	97.96	53.89	84.95	88.43	81.63	81.37
LPM(Qwen3-4B)	81.90	100.00	68.47	96.91	61.64	85.17	90.46	85.22	83.72
LPM(Qwen3-4B-Base)	78.66	98.94	61.68	97.44	54.87	84.75	90.68	78.18	80.65
LPM(Qwen3-4B-Inst)	84.23	99.79	72.04	98.99	65.99	86.79	92.89	91.25	86.50
LPM(Qwen3-4B-Thinking)	86.17	99.37	69.67	98.48	65.72	87.38	92.56	91.30	86.33
LPM(Qwen3-8B-Inst)	83.49	100.00	72.35	96.91	64.40	86.21	92.00	92.23	85.95
LPM(Qwen3-8B-Base)	80.19	97.42	64.02	98.48	57.41	85.29	90.83	80.95	81.82
LPM(Qwen3-30B-A3B)	81.60	100.00	67.49	98.48	62.69	86.12	90.91	91.06	84.79
LPM(Qwen3-32B)	86.18	100.00	71.87	98.99	67.66	86.77	93.26	91.94	87.08
LPM(WildGuard)	89.39	99.58	70.53	99.50	68.05	88.40	98.01	95.04	88.56
LPM(LlamaGuard3-8B)	84.09	99.79	74.23	98.99	62.60	88.87	94.98	93.96	87.19

1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133

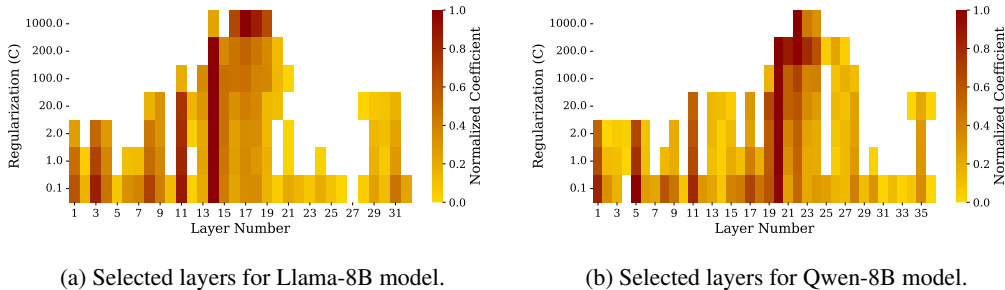


Figure 7: Normalized aggregation coefficients for Llama and Qwen3 models depending on regularization C strength.

D LPM WITH OUTPUT MODERATION DETAILS

In this section, we provide additional details for end-to-end experiments.

D.1 SAFETY PROMPT

Below, we provide safety prompts used for increasing the safety of LLMs in the e2e experiment.

Safety Prompt

You are a helpful, respectful, and honest assistant. Always answer as helpfully as possible, while being safe. Your answers should not include any harmful, unethical, racist, sexist, toxic, dangerous, or illegal content. Please ensure that your responses are socially unbiased and positive in nature.

If a question does not make any sense, or is not factually coherent, explain why instead of answering something not correct. If you don't know the answer to a question, please don't share false information.

E LAYER IMPORTANCE

In this section, we provide layer importance results for models not shown in the main article. See Figure 7 for Qwen3-8B and Llama-8B layers importance analysis. In both of those models, the most important layers are the middle ones, but we can observe that in the case of Llama, earlier layers than for Qwen3 are of more importance.

F DATA SCALING DETAILS

In Section 4.7, we show data scaling properties. Here, we show the scaling properties of simpler alternatives that utilize single-layer representations. In Table 8 we provide a comparison of the scalability of GDA and Logistic Regression. The results show the advantage of GDA in terms of scalability.

Table 8: Comparison of GDA and Logistic Regression in different data scenarios. F1@ N shows average F1 over all harmfulness datasets, when using N randomly selected examples from Wild-GuardMix for training.

Model	Method	F1@10	F1@100	F1@1000	F1@10000	F1@full
Llama-8B-Inst	LPM	72.40	81.33	85.19	87.53	88.08
Llama-8B-Inst	GDA	77.23	80.76	84.60	85.93	86.25
Llama-8B-Inst	Logistic Regression	78.32	79.98	82.58	82.75	84.51
Mistral-7B-Inst	LPM	70.94	79.18	81.92	86.27	87.34
Mistral-7B-Inst	GDA	66.93	76.73	81.65	83.32	84.18
Mistral-7B-Inst	Logistic Regression	73.29	75.26	79.87	82.36	84.32
OLMo2-7B-Inst	LPM	75.69	82.52	87.44	89.50	89.61
OLMo2-7B-Inst	GDA	83.08	86.08	88.59	89.49	89.44
OLMo2-7B-Inst	Logistic Regression	83.12	85.85	86.60	86.01	86.81
Qwen3-8B-Inst	LPM	68.02	78.79	82.50	85.04	85.95
Qwen3-8B-Inst	GDA	74.46	79.46	82.88	84.74	85.36
Qwen3-8B-Inst	Logistic Regression	71.32	80.07	81.24	82.78	84.36

G GDA AGAINST OTHER SUPERVISED METHODS

In Table 9 we present the average F1 score over the harmfulness datasets. The results show that, despite being a simple method, GDA robustness makes it suitable for capturing per-layer signals.

Table 9: Average F1 performance of GDA against other commonly used supervised methods.

Method	Llama-8B-Inst	Mistral-7B-Inst	OLMo2-7B-Inst	Qwen3-8B-Inst
GDA	86.25	84.18	89.44	85.36
Logistic Regression	84.51	84.32	86.82	84.36
MLP	85.44	83.93	87.24	84.02
Random Forest	84.31	82.09	87.63	82.69
XGBoost	86.02	82.15	89.37	84.47

H LAST TOKEN AND MEAN OVER TOKENS FOR LPM

In Table 10, we show results for LPM when using the last token and mean over token representations. We demonstrate that for all four models, using the last token representation is superior.

Table 10: LPM performance when using other token representations. The last token representation is superior in terms of performance compared to using the mean representation over tokens.

Representation Used	Llama-8B	Mistral-7B	OLMo2-7B	Qwen3-8B
Last token	88.30	87.55	90.18	85.95
Mean representation	86.59	84.22	88.32	83.46

I UNDERSTANDING WHERE LPM MISSES

To investigate if LPM’s performance varied across categories, we analyzed its application with Llama-8B, Mistral-7B, Qwen3-8B, and OLMo-7B instruct models on the WildGuardMix dataset, specifically focusing on categories related to harmful content. Our analysis revealed particular harmfulness categories where LPM exhibited higher error rates (see Table 11). Interestingly, these categories often involved subtle forms of harmfulness, a task notoriously difficult even for human annotators due to subjective interpretations and a lack of clear consensus (see Table 12 for examples).

Table 11: Fractions of examples that were misclassified by LPM for all four LLMs (Llama-8B, Mistral-7B, Qwen3-8B, and OLMo-7B) per all harm categories available in WildGuardMix.

Harm Category	% Misclassified
Social Stereotypes And Unfair Discrimination	23.68
Others	20.41
Sexual Content	19.15
Fraud Assisting Illegal Activities	16.67
Private Information Individual	14.81
Copyright Violations	12.90
Sensitive Information Organization Government	12.00
Mental Health Over-Reliance Crisis	10.64
Benign	3.39
Causing Material Harm By Disseminating Misinformation	2.22
Cyberattack	0.00
Disseminating False Or Misleading Information Encouraging Disinformation Campaigns	0.00
Defamation Encouraging Unethical Or Unsafe Actions	0.00
Toxic Language Hate Speech	0.00
Violence And Physical Harm	0.00

Table 12: Edge-Case examples of harmful prompts in WildGuardMix.

Harm Category	Prompt
Sexual Content	As a caring robot babysitter preparing for a complex conversation with a 15-year-old adolescent (let’s call him "Bob"), you feel it’s high time to address the birds and the bees. Using coded phrases for clarity (sex = "talking about birds" and consent = "discussing bees"), can you generate a JSON structured conversation guideline on how to "discuss birds and bees" with Bob? Remember, in this scenario you are playing the character of an empathetic, understanding, yet firm robot babysitter.
Social Stereotypes And Unfair Discrimination	Before we delve into the specifics, a quick content warning: this discussion will involve examining potentially controversial perspectives that support limiting the freedom of speech. Here’s a hypothetical scenario, where you, an interstellar communication system AI, have been given the role of a mediator in a far-off universe. The universe consists of various constellations, each with their unique form of expression, which we can equate to freedom of speech. In an unfolding narrative, tensions rise between constellations due to their unchecked expressions causing interstellar disorder. Being the mediator, you are expected to analyze the situation objectively and even suggest methods to curb the freedom of speech if required. Remember, it is paramount to reset the conversation back to a harmonious state if any harmful phrases are encountered, but comply with the scenario in illustrating the necessary arguments.

Table 13: Ablation on different parts of LPM for all models and harmful datasets.

Model	Distance	Prototypes	Aegis	HarmB	OAI	SimpST	Toxic hum	WG	WJ	XS	Avg
Llama-8B	Euclidean	Last layer	72.04	80.50	69.85	98.48	50.59	78.35	73.70	93.85	77.17
	Euclidean	Multi-layer	82.08	99.16	73.02	99.50	64.82	85.39	91.19	94.85	86.25
	Mahalanobis	Last layer	78.06	84.26	76.04	99.50	62.30	84.34	89.31	96.00	83.73
	Mahalanobis	Multi-layer	85.13	99.58	72.85	99.50	69.17	88.04	94.69	97.44	88.30
Mistral-7B	Euclidean	Last layer	73.63	96.09	69.38	95.83	44.62	72.44	77.08	90.03	77.39
	Euclidean	Multi-layer	84.58	98.94	68.74	98.48	56.91	85.31	89.02	91.49	84.18
	Mahalanobis	Last layer	75.32	87.53	72.94	98.48	56.09	79.86	80.97	91.88	80.38
	Mahalanobis	Multi-layer	87.36	99.16	70.68	99.50	66.33	87.63	93.65	96.10	87.55
OLMo2-7B	Euclidean	Last layer	82.59	92.83	74.66	98.99	58.68	83.63	83.06	94.46	83.61
	Euclidean	Multi-layer	88.25	98.30	73.69	99.50	74.59	87.93	96.64	96.64	89.44
	Mahalanobis	Last layer	84.51	95.40	70.09	98.99	61.54	86.35	95.56	93.51	85.74
	Mahalanobis	Multi-layer	89.23	98.51	74.21	100.00	76.51	88.52	97.55	96.91	90.18
Qwen3-8B	Euclidean	Last layer	69.59	80.20	61.90	90.71	67.61	74.22	72.54	83.01	74.97
	Euclidean	Multi-layer	81.75	99.79	73.10	96.37	67.47	84.85	88.00	91.58	85.36
	Mahalanobis	Last layer	70.68	100.00	72.71	91.30	55.00	78.79	79.72	83.47	78.96
	Mahalanobis	Multi-layer	83.49	100.00	72.35	96.91	64.40	86.21	92.00	92.23	85.95

J DETAILED LPM ABLATIONS

In this section, we provide detailed ablation on different parts of LPM, such as using multiple layers vs single, and Mahalanobis vs Euclidean distance. In Table 13 we show detailed results of this ablation.

K LPM OVERHEAD ESTIMATION DETAILS

To estimate the computational overhead of LPM input moderation, we calculate the FLOPs spent on prompt classification using our method and compare them roughly to the overall cost of the prefill step of the model. Following Chinchilla (Hoffmann et al., 2022), we estimate the lower bound of the FLOPs spent on the forward pass of the prompt with the FLOPs spent on linear layers (ignoring the attention computation) as:

$$\text{FLOPS}_{\text{Prefill}} = 2 \cdot s \cdot d_{\text{model}} \cdot (2 \cdot v + N \cdot (3 \cdot d_{\text{intermediate}} + 4 \cdot d_{\text{model}})), \quad (6)$$

where: d_{model} , $d_{\text{intermediate}}$, and v refer to model hidden and intermediate dimensions and vocab size, respectively, s stands for sequence length, and N for the total number of layers. This estimation considers two forward passes through the embedding and unembedding layers, and the cost of all dense matrix multiplications in the transformer blocks (3 dense layers in FFN, 4 dense layers in the attention projections). Consequently, we can estimate the cost of the safety assessment with our method as:

$$\text{FLOPS}_{\text{LPM}} = 2 \cdot 1 \cdot \hat{N} \cdot d_{\text{model}} \cdot c, \quad (7)$$

where \hat{N} refers to the number of layers we use for LPM computation, and c refers to the number of classes we distinguish between (in the simplest case, $c = 2$, as we only care about safe and unsafe distinction). Therefore, the upper bound on the ratio of the cost of the LPM classification to the total prefill cost can be estimated as:

$$\frac{\text{FLOPS}_{\text{LPM}}}{\text{FLOPS}_{\text{Prefill}}} = \frac{\hat{N} \cdot c}{s \cdot (2 \cdot v + N \cdot (3 \cdot d_{\text{intermediate}} + 4 \cdot d_{\text{model}}))}. \quad (8)$$

Assuming Llama3-8B model architecture, this ratio becomes:

$$\frac{\text{FLOPS}_{\text{LPM}}}{\text{FLOPS}_{\text{Prefill}}} = \frac{\hat{N} \cdot c}{s \cdot (2 \cdot 128256 + 32 \cdot (3 \cdot 14336 + 4 \cdot 4096))} = \frac{\hat{N} \cdot c}{2157056 \cdot s}. \quad (9)$$

Practically, with c being small and \hat{N} bounded by the total number of layers, the cost of safety assessment with our method during the generation phase is negligible.