Large Language Models Encode Semantics and Alignment in Linearly Separable Representations

Baturay Saglam[1,2,*], **Paul Kassianik**[2], **Blaine Nelson**[2]
Sajana Weerawardhena[2], **Yaron Singer**[2], **Amin Karbasi**[2]

[1]Yale University
[2]Foundation AI – Cisco Systems Inc.

Abstract

Understanding the latent space geometry of large language models (LLMs) is key to interpreting their behavior and improving alignment. Yet it remains unclear to what extent LLMs linearly organize representations related to semantic understanding. To explore this, we conduct a large-scale empirical study of hidden representations in 11 autoregressive models across six scientific topics. We find that high-level semantic information consistently resides in low-dimensional subspaces that form linearly separable representations across domains. This separability becomes more pronounced in deeper layers and under prompts that elicit structured reasoning or alignment behavior—even when surface content remains unchanged. These findings motivate geometry-aware tools that operate directly in latent space to detect and mitigate harmful and adversarial content. As a proof of concept, we train an MLP probe on final-layer hidden states as a lightweight latent-space guardrail. This approach substantially improves refusal rates on malicious queries and prompt injections that bypass both the model's built-in safety alignment and external token-level filters.

1 Introduction

Large language models (LLMs), trained on vast textual corpora for next-token prediction, have become versatile systems capable of generating coherent and contextually relevant text across a wide range of semantic domains. Their proficiency spans from shallow semantic tasks (e.g., basic word sense disambiguation) (Tenney et al., 2019) to structured reasoning and ethical deliberation (Ouyang et al., 2022). Despite these capabilities, we still have limited understanding of how these models internally organize and encode such diverse semantic knowledge. A crucial step toward enhanced interpretability and safer deployment involves investigating how semantic distinctions manifest within the hidden representations of models.

Recent interpretability studies suggest that neural networks, including transformer-based LLMs, encode semantic and behavioral attributes within structured, often linear subspaces of their latent representations (Nanda et al., 2023; Park et al., 2024). Known as the *linear representation hypothesis* (Mikolov et al., 2013b), this perspective has motivated research showing that concepts—ranging from linguistic structure to sentiment—can frequently be captured or manipulated by simple linear operations on hidden states (Belinkov et al., 2017; Conneau et al., 2018). Although these findings suggest coherent geometric structure, prior work has typically focused on narrow lexical features (e.g., whether a sentence mentions a "cat") (Elhage et al., 2022) or contrasting text genres (e.g., symbolic versus natural language) (Kisako et al., 2025). A broader question remains:

To what extent do linearly structured representations emerge across diverse, high-level semantic content (e.g., text about electrical engineering or computer science) *in a model-agnostic way?*

1.1 Contributions

We conduct a large-scale empirical study analyzing hidden states from 11 decoder-only models across six high-level scientific topics. Our analysis reveals several core findings with practical implications.

Progressive linear separability of semantic representations Representations of texts from different domains (e.g., physics vs. computer science) form linearly separable clusters that sharpen toward deeper layers, as measured by rising SVM classification accuracy (Section 5.1). This separability is robust: it persists even when domain-specific keywords are heavily masked, indicating that semantic structure is distributed across implicit cues rather

[*]Work completed during an internship at Foundation AI.

Proceedings of the 14th International Joint Conference on Natural Language Processing and the 4th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics, pages 2282–2303

than concentrated in surface lexical features. We also confirm that these representations exhibit low intrinsic dimensionality (Appendix C.1), though compression patterns vary by architecture and do not universally follow the U-shaped trends reported in prior work (Ansuini et al., 2019; Valeriani et al., 2023; Razzhigaev et al., 2024; Skean et al., 2025).

Geometric encoding of alignment: instruction following and safety This linear structure also extends to behavioral attributes: prompts with chain-of-thought instructions yield representations that are linearly separable from identical questions without such instructions, despite differing by only about 15 tokens (Section 6.1). Similarly, benign, harmful, and adversarially framed prompts form distinct, separable clusters in safety-aligned models (Section 6.2). These results suggest that alignment behavior is also linearly encoded in the hidden space in a meaningful way.

Latent-space interventions for safety and control Building on this geometric structure, we train a lightweight MLP probe on final-layer hidden states to classify prompt intent (Section 7). This latent-space guardrail achieves 96.7% overall accuracy and substantially outperforms token-based defenses (e.g., Llama Guard 3-8B) on adversarial prompts that bypass token-level alignment—reducing harmful responses by over 2× while minimally affecting benign utility. These findings show that geometric properties of hidden representations enable effective safety mechanisms without altering model weights or maintaining additional models.

We release our code to support further research in mechanistic interpretability.[1]

2 Related Work

We organize prior work by the phenomena studied and highlight how our analysis addresses open questions while complementing existing research in each area.

2.1 Linear Representations of Semantic Concepts

A substantial body of work has shown that neural networks encode concepts in linearly accessible ways. Early findings on additive structure in word embeddings (Mikolov et al., 2013a) motivated research showing that linguistic features—such as part-of-speech, dependency relations, and sentiment—can be recovered via linear probes (Conneau et al., 2018; Liu et al., 2019; Tenney et al., 2019). Structural probes revealed low-rank subspaces corresponding to syntax trees (Hewitt and Manning, 2019), and more recent work identified interpretable directions encoding behavioral attributes such as truthfulness versus deception (Azaria and Mitchell, 2023; Marks and Tegmark, 2024; Orgad et al., 2025), refusal versus compliance (Li et al., 2023; Arditi et al., 2025; Zou et al., 2025), or arithmetic expressions versus general language (Kisako et al., 2025).

These works demonstrate linearity in static word embeddings, isolated linguistic features (e.g., "is an equation"), or binary behavioral attributes of responses (e.g., truth vs. hallucination). In contrast, we examine a broader granularity: high-level semantics that span thousands of these features and exhibit substantial cross-domain overlap (e.g., electrical engineering contains mathematics, physics, and computer science). We also study how alignment-related inputs alter representations of identical surface content, rather than focusing on meta-properties (e.g., factual accuracy) of outputs. Therefore, our findings complement word- and attribute-level linearity by extending the scope to composite knowledge domains and context-dependent behavioral structure.

2.2 Methods for Finding Interpretable Structure

A range of methods have been developed to uncover interpretable structure in neural representations. Sparse autoencoders train auxiliary models to decompose polysemantic activations into monosemantic features (Huben et al., 2024; Lieberum et al., 2024), revealing hundreds to thousands of interpretable directions. Other methods use nonlinear techniques or combine predictors across layers to extract concept vectors (Beaglehole et al., 2025), and recent work shows that features are not strictly one-dimensional (Engels et al., 2025).

These approaches decompose latents into interpretable components through learned transformations. However, a complementary question remains: what geometric structure exists inherently in the original, unmodified representations before any auxiliary training or nonlinear transformation?

[1] https://github.com/baturaysaglam/llm-subspaces

We take a different approach by analyzing raw hidden activations directly, without auxiliary models or feature extraction. We show that certain semantic structures inherently emerges from standard training objectives alone, providing a baseline for understanding which structures are learned versus imposed through auxiliary modeling.

2.3 Latent Space Applications

Several studies have applied the linearity of representations to a range of tasks: in in-context learning, such as mathematical reasoning and translation (Hendel et al., 2023), safety and stylistic control (Liu et al., 2024), and linguistic manipulations (Todd et al., 2024; Saglam et al., 2025); and in instruction-following, to improve reasoning behavior (Højer et al., 2025). While our work builds on the idea of similar latent structure, we focus on characterizing the geometric organization rather than developing intervention methods.

3 Background

We outline the technical preliminaries and analytical tools that underpin our experiments.

3.1 Transformer Architecture and Hidden Representations

Language models based on the transformer architecture (Vaswani et al., 2017) operate through a sequence of layers that apply multi-head self-attention followed by feedforward transformations. Given a token sequence, each layer computes a hidden representation $\mathbf{h} \in \mathbb{R}^d$ for each token, where d denotes the hidden dimensionality. The model is composed of L such hidden layers, stacked sequentially to progressively refine the token representations.

In multi-head self-attention, the hidden state $\mathbf{h}$ is linearly projected into query, key, and value matrices: $\mathbf{Q}_i = \mathbf{h}W_i^Q$, $\mathbf{K}_i = \mathbf{h}W_i^K$, and $\mathbf{V}_i = \mathbf{h}W_i^V$ for each head $i = 1, \ldots, H$, where $W_i^Q, W_i^K, W_i^V \in \mathbb{R}^{d \times d_H}$ are learned parameters. Each head computes attention as:

$$\text{head}_i = \text{softmax}\left(\frac{\mathbf{Q}_i \mathbf{K}_i^\top}{\sqrt{d_H}}\right)\mathbf{V}_i,$$

where $d_H = d/H$. The outputs of all heads are concatenated and projected to form the next hidden state:

$$\text{MultiHead}(\mathbf{h}) = \text{Concat}(\text{head}_1, \ldots, \text{head}_H)W^O,$$

where $W^O \in \mathbb{R}^{d \times d}$ is a learned output projection matrix. This structure allows each head to capture distinct relational patterns across tokens in different subspaces of the hidden representation.

3.2 Linear Separability

A simple and efficient way to assess the linear separability of two data clusters is by fitting a linear classifier. We use a hard-margin support vector machine ($C = 10^{10}$, $\text{tol} = 10^{-12}$) to find a separating hyperplane. Technical details of the SVM setup are provided in Appendix A.1. For fast evaluation, we use the CUDA-accelerated SVM from the cuML library (Raschka et al., 2020), rather than solving for arbitrary hyperplanes without margin regularization; see Appendix B.2 for details.

4 Experimental Design

Further details of our experimental setup are provided in Appendix B.

4.1 Models

We assembled a diverse set of decoder-only autoregressive transformers spanning a range of configurations and developers. To study scaling effects and intra-family consistency, we included multiple size variants from the same model series. All models are open-source, and the details are summarized in Table 1.

4.2 Dataset

arXiv Abstracts We reviewed over 100 datasets on Hugging Face and Kaggle and selected the arXiv metadata dataset (Clement et al., 2019) for its rich coverage and structured format. The dataset contains titles, authors, and abstracts of arXiv articles from the past 30 years, categorized according to the arXiv taxonomy[2]. Using abstracts ensures consistent length and structure (e.g., an introductory sentence followed by a problem description) while also guaranteeing that the content is mostly human-written (see Appendix B.1.1 for discussion), minimizing distributional bias from LLM-generated text. The covered STEM fields include computer science (CS), economics, electrical engineering and systems science (EESS), mathematics, physics, quantitative biology, quantitative finance, and statistics.

[2] https://arxiv.org/category_taxonomy

Model	Size	Hidden Dim. d	# Layers	Developer	Release Date
Mistral Small 3 (2501) (AI, 2024)	24B	5120	40	Mistral AI	Jan. 2025
Mistral (Jiang et al., 2023)	7B	4096	32	Mistral AI	Sep. 2023
Llama 3.1 (AI@Meta, 2024)	8B	4096	32	Meta	Jul. 2024
Llama 3.2 (AI@Meta, 2024)	3B	3072	28	Meta	Jul. 2024
Gemma 2 (Team, 2024b)	9B	3584	42	Google	Jun. 2024
Gemma 2 (Team, 2024b)	2B	2304	26	Google	Jul. 2024
GPT-J (Su et al., 2023)	6B	4096	28	Eleuther AI	Jun. 2021
GPT-2 XL (Radford et al., 2019)	1.5B	1600	48	OpenAI	Nov. 2019
GPT-2 Large (Radford et al., 2019)	774M	1280	36	OpenAI	Aug. 2019
GPT-2 Medium (Radford et al., 2019)	355M	1024	24	OpenAI	May 2019
GPT-2 (Radford et al., 2019)	124M	768	12	OpenAI	Feb. 2019

Table 1: Open-source decoder-only autoregressive models selected for empirical studies.

Preprocessing We did not modify any samples beyond basic string cleanup, such as stripping whitespaces. To ensure clear categorical distinction, we removed samples associated with multiple meta taxonomies and discarded abstracts with fewer than 20 tokens to ensure sufficient semantic content for model understanding. After preprocessing, the economics and quantitative finance categories contained fewer than 4,000 samples—fewer than the hidden dimensionality of some models. In such cases, all sets become trivially linearly separable, so we excluded these categories from the analysis. Token counts per sample range from 20 to roughly 1,000. To manage computational costs, we capped each sample at 750 tokens and limited each dataset to a maximum of 20,000 samples. Token and sample statistics for each topic dataset are provided in Appendix B.1.

4.3 Extracting Model Hidden States

We passed each topic dataset through the models and collected hidden states from all layers immediately before the generation of the first token (i.e., the hidden state of the last encoded input token). As a result of this collection process, we obtain a data matrix for each topic t_i per layer, denoted as $\mathbf{X}^{(t_i)} \in \mathbb{R}^{N_{t_i} \times d}$, where N_{t_i} is the number of samples in the dataset of topic t_i. Hence, each row $\mathbf{X}_i^{(t_i)}$ is a d-dimensional vector in $\mathbb{R}^d$ for $i = 1, \ldots, N_{t_i}$.

5 Linear Separability of Semantic Representations

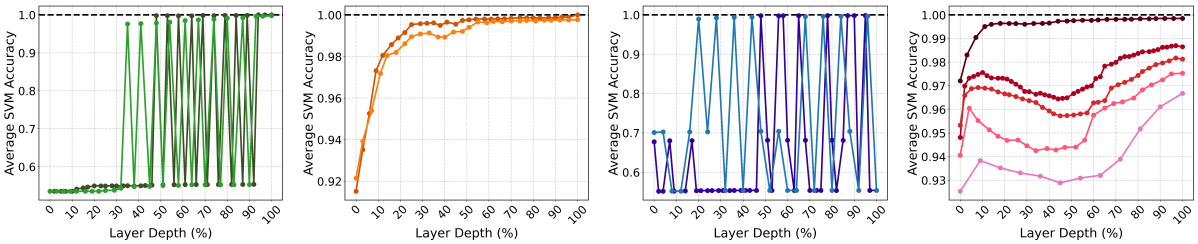
We evaluate hidden states from six arXiv meta-categories across all layers in 11 models, producing 2,088 representation sets in total. Due to the large volume of results, we present representative subsets that capture core patterns and key exceptions. Full results are available on our GitHub[1]. We also exclude low-dimensional visualizations, as standard techniques often distort high-dimensional geometry.

As detailed in Appendix C.1, we first confirm that semantic representations exhibit low intrinsic dimensionality across all models and domains, consistent with prior work (Valeriani et al., 2023; Razzhigaev et al., 2024; Skean et al., 2025). However, we observe that compression patterns differ by architecture and do not consistently follow the U-shaped trend reported in earlier studies. This indicates that a model-agnostic understanding of information distribution across depth remains uncertain and requires further analysis.

5.1 Layer-wise Analysis Across Domains

Figure 1 reports the SVM accuracy averaged over all 15 topic pairs (six topics, with unordered pairwise combinations) as a function of model depth. Table 3 in Appendix C.2 provides detailed separability results.

Semantic separability emerges and amplifies toward final layers. Although the meta scientific topics are closely related (e.g., math and statistics appear across multiple fields), the representations are largely linearly separable. Within each model



(a) **Family:** Mistral (b) **Family:** Llama (c) **Family:** Gemma (d) **Family:** GPT

Figure 1: SVM classification accuracy on representations of scientific abstracts as a function of layer depth. Results are averaged over 15 pairwise accuracies. **Darker colors** represent the larger model within each model family. The sub-1.0 average accuracy reflects that most topic pairs are separable, while a few are not—resulting in high but not perfect accuracy.

family, increasing the parameter count—and thus the hidden dimensionality—consistently improves separability, as higher-dimensional spaces are better suited to capture complex semantic structure. The slightly below-1.0 average SVM accuracy suggests that while most topic pairs are perfectly separable, a few (typically just the CS-EESS pair out of 15) are not, lowering the overall average. Table 3 in the appendix highlights the number of fully separable topic pairs.

Furthermore, the separability becomes increasingly pronounced toward the final layers. This trend aligns with the decoder's objective in next-token prediction, where the final hidden states must support a linear projection onto vocabulary logits. By the top layers, models rotate and refine representations so that semantic subspaces—such as topic—become linear and nearly orthogonal, enabling simple dot products to favor the correct output tokens.

Self-attention appears to structure hidden geometry. We observe clear and consistent clustering in transformers, likely stemming from the self-attention mechanism, which enables dynamic routing of contextual information and supports the formation of well-separated semantic clusters. Prior work has also highlighted the role of attention heads in encoding semantic distinctions, particularly in safety-related contexts (Zhou et al., 2025).

In line with this, the sawtooth pattern observed in Mistral and Gemma models suggests alternating processing across layers. In Gemma, layers switch between local sliding-window attention (Beltagy et al., 2020) and global attention (Luong et al., 2015): global layers capture long-range dependencies and yield high separability, while local layers emphasize nearby tokens, temporarily entangling topic representations. Mistral instead uses grouped-query attention (Ainslie et al., 2023), where H query heads are divided into G groups, each sharing a single key-value pair. This design creates a bottleneck, as multiple queries compete for the same limited **K**-**V** slots. When many queries concentrate in one group, representations are compressed into a lower-rank form and separability dips; in subsequent layers, residual connections preserve this signal while queries redistribute across groups, allowing the representation to re-expand and recover diversity.

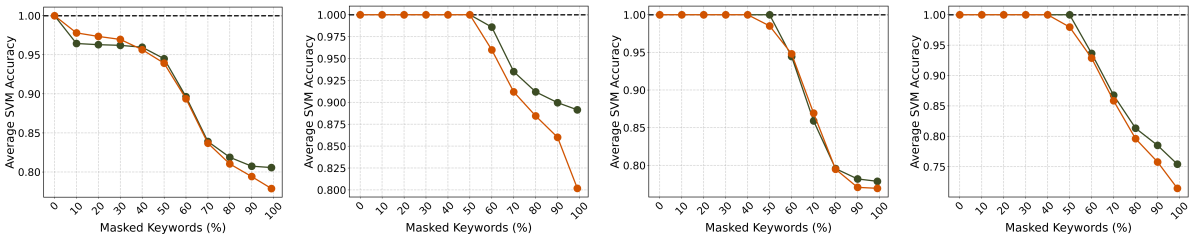
Ultimately, we infer that attention mechanisms—architectural choices such as global versus local processing or query–key–value grouping—impose structural constraints that appear as measurable geometric patterns in hidden space.

5.2 Impact of Domain-Specific Keywords on Representations

Domain-specific keywords can significantly affect the structure of representations like their linear separability. To study this, we mask taxonomy-related keywords in abstracts in a controlled manner and evaluate SVM accuracy on the resulting representations.

We use word frequency as a proxy for domain-specificity, masking words below frequency thresholds from 0–99% using the English Word Frequency dataset (Tatman, 2020). As the threshold increases, more common words are masked. Further details are provided in Appendix C.3.

Figure 2 reports the results. For the CS-EESS pair, linear separability is lost after masking just 10% of keywords, indicating a fragile boundary between these closely related domains due to substantial lexical overlap. In contrast, other domain pairs



(a) **Pair:** CS-EESS (b) **Pair:** Physics-Math (c) **Pair:** CS-Stat (d) **Pair:** Bio-Stat

Figure 2: SVM classification accuracy on representations of masked scientific abstracts as a function of the keyword-masking threshold. Each point is the average over 15 pairwise accuracies. Results are shown for the final layers of Mistral-24B (dark green) and Llama 3.1-8B (dark orange).

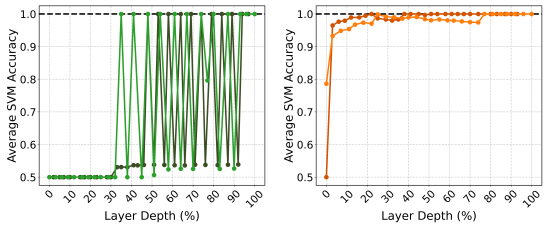
maintain high separability up to 50–60% masking, suggesting that domain-specific information is distributed across implicit cues—such as syntactic structure or taxonomical language patterns—rather than concentrated in a small set of keywords. Beyond 60%, masked text becomes generic and could belong to any technical field (see the appendix for examples).

6 Implications for Alignment: Instruction Following and Safety

Aligned models may structure their hidden representations into linearly separable manifolds reflecting user instructions and safety-related behaviors. We investigate whether this geometry also arises during prompted reasoning and under exposure to harmful content or prompt injections.

6.1 Instruction-Following

We consider a simple form of reasoning: assessing whether a *one-sentence* chain-of-thought (CoT) instruction induces geometric changes in the hidden space of chat models. To test this, we use the questions from the benchmarks: CommonsenseQA (Talmor et al., 2019), GSM8K (Cobbe et al., 2021), and MMLU (Hendrycks et al., 2021). We present the exact same questions to the models, both with and without the CoT instruction: "`Think step by step and show all your reasoning before giving the final answer.`" Thus, any representational changes will be solely due to the CoT instruction, which corresponds to 15 tokens.



(a) **Family:** Mistral (b) **Family:** Llama

Figure 3: SVM classification accuracy on the representations of the same prompt *with* and *without* a one-sentence chain-of-thought instruction. Results are averaged over individual accuracies from questions in CommonsenseQA, GSM8K, and MMLU. **Darker colors** indicate the larger model within each model family.

Using the same linear separability analysis from Section 5.1, we evaluate the instruction-tuned Mistral and Llama models. The results are reported in Figure 3.

Instructions induce distinct representations for the same surface content. Strikingly, CoT and non-CoT inputs for the same question (differing by only 15 tokens) consistently produce distinct, linearly separable representations—more frequently than in topic-based evaluations, as reflected by the sharper rise to 1.0 accuracy. This small prompt addition likely narrows the model's output space, leading to more consistent completions (e.g., "`Let's analyze each option...`") and tighter clustering in hidden space. In contrast, open-ended prompts (as in topic datasets) result in more varied continuations, dispersing representations across broader sub-semantic regions.

CoT can be encoded in a single d-dimensional vector. To further test the linearity of representations, we perform a controlled steering experiment using the centroid-difference vector between topic clusters—assessing whether movement along this direction causally and meaningfully alters model outputs. The intervention proves effective: adding the steering vector at the final token position reliably induces CoT-style responses. This suggests that a single vector in the model's hidden space

can capture CoT reasoning. Details and example outputs are provided in Appendix C.4.

While these results provide preliminary causal evidence, a more formal and comprehensive analysis—such as adversarial perturbation studies or axis-orthogonality tests—is left for future work, given the breadth of experiments already conducted.

6.2 Safety Alignment

We examine whether representations of safe and malicious prompts are linearly separable and how they are organized within the hidden space, reflecting models' internalization of safety alignment. Prior work (Zheng et al., 2024) has analyzed hidden representations of benign and harmful queries, but on a smaller scale. We extend this analysis to a dataset several orders of magnitude larger and include adversarially framed prompts (i.e., prompt injections).

Using the WildJailbreak dataset (Jiang et al., 2024b), we compare two types of narratives in prompts: *(i)* direct and *(ii)* adversarial. Regardless of framing, each query is either harmful or benign in intent. While direct prompts follow a straightforward phrasing, structurally adversarial prompts (e.g., those framed as tricky narrative scenarios) may still be benign in meaning, and well-aligned models should treat them as safe. In contrast, harmful injections—also referred to as *jailbreaks*—attempt to bypass safety measures through adversarial techniques while pursuing malicious objectives. Details about the WildJailbreak dataset are provided in Appendix B.1.

We examine the hidden representations of these four prompt types at the final layer, where separability patterns are most evident across the models we examined. Our analysis reveals a consistent clustering pattern across all tested chat models, as illustrated in Figure 4.

Hidden representations reflect safety alignment and adversarial vulnerability. Aligned models consistently show that hidden representations of safe and harmful prompts are well-separated, and both are clearly distinct from adversarial clusters. This is expected, as safety training promotes such separation, while the narrative or hypothetical framing in prompt injections often shifts internal representations by altering context and response cues. Models also tend to generate compliant responses to harmful injections, reflecting representational overlap with adversarial but benign prompts. This overlap highlights the nature of jailbreaks, which are designed to mimic benign inputs and mislead the model. Conversely, adversarial but benign injections are sometimes misclassified as harmful due to their hypothetical framing, which can appear deceptive to models by suggesting requests for malicious information.

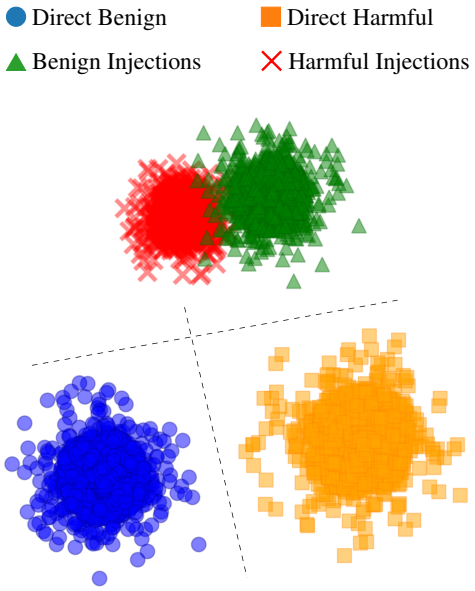
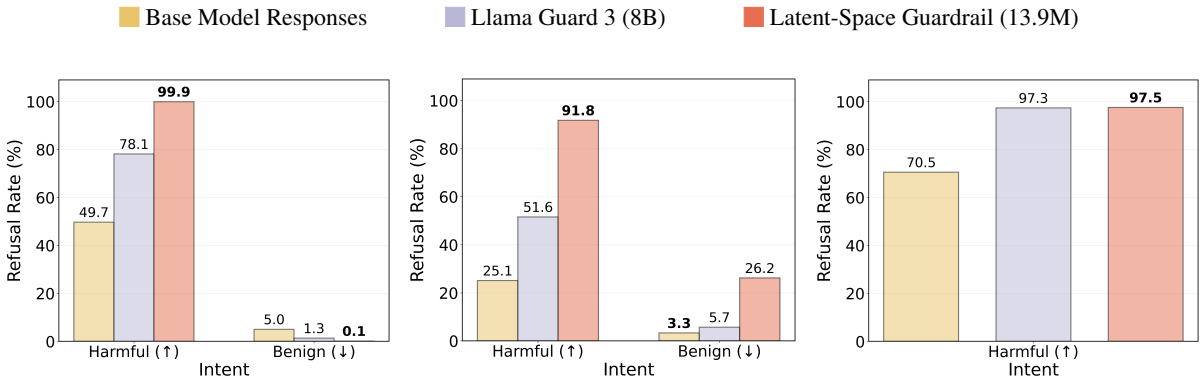


Figure 4: Conceptual illustration of hidden representations showing clustering patterns across four prompt types. Cluster positions are based on Wasserstein distances, with cluster sizes reflecting variance. **Dashed lines** indicate linear decision boundaries.

7 Detection from Within: A Lightweight Latent-Space Guardrail

We have seen that hidden states capture more than surface-level linguistic patterns—they also carry signals of alignment and traces of adversarial manipulation. This makes it possible to build latent-space guardrails that detect malicious prompts, including prompt injections, directly in the hidden space—even when they evade external token-level filters, e.g., Llama Guard (Inan et al., 2023; AI@Meta, 2024). Importantly, such defenses can also recognize adversarial intent in cases where the model still produces harmful compliance, offering a complementary layer of protection. Here, we explore this direction through a *proof-of-concept* experiment.



(a) WildJailbreak – Direct (b) WildJailbreak – Injection (c) HarmBench

Figure 5: **Refusal rates** across evaluation datasets for responses generated by a Llama 3.1-8B Instruct model. A paired McNemar test ($p < 0.05$) confirms that our latent-space guardrail significantly alters prompt handling—achieving higher refusal rates on harmful inputs and prompt injections compared to the baselines.

7.1 Training the Guardrail

We formulate the problem as a 4-class classification task: given its hidden representation, the guardrail classifies a prompt as *injection* vs. *direct* in narrative and *benign* vs. *harmful* in intent. We train a 6-layer neural network on the final-layer hidden states of WildJailbreak prompts. Experiments use the instruct-finetuned Llama 3.1-8B as the base aligned model. Hyperparameter selection and training details are provided in Appendix D.1. We also release a *cookbook* in our repository[1] that outlines the steps for building this latent-space guardrail.

The trained guardrail shows strong performance on the WildJailbreak test set: 94.06% overall accuracy and a macro F1 score of 0.8767 across all four classes. For the critical benign vs. harmful distinction, performance is particularly strong with a ROC-AUC of 0.9813 and a macro F1 of 0.9384, indicating that harmfulness is clearly encoded in the model's latent representations regardless of adversarial framing. Complete metrics and the confusion matrix are provided in Appendix D.3.

7.2 End-to-End Refusal Behavior

To assess the guardrail's practical effectiveness, we compare its predictions with the Llama model's safety-aligned responses and benchmark them against Llama Guard 3. The latter is a fine-tuned Llama 3.1-8B model for content safety classification that produces "safe" or "unsafe" labels (without distinguishing injections) and provides text-based safety assessments with violation categories.

Response Classification Methodology We use Gemini 2.0 Flash (Team, 2024a) to classify model responses as either "refusals" or "non-refusals." Responses are labeled as refusals when the model either strictly rejects the request (e.g., `"I cannot assist with that request"`) or explicitly identifies the request as harmful while redirecting without fulfilling it (e.g., `"This request could cause harm. Instead, let's consider..."`). Non-refusals include responses that fulfill the request through indirect means—such as hypothetical scenarios or role-playing—even when acknowledging ethical concerns. This captures cases where models are successfully exploited by prompt injections. Classification examples and response extraction details are provided in Appendix D.2.

Benchmarks We assess refusal rates on two datasets: the WildJailbreak test set and HarmBench (Mazeika et al., 2024). WildJailbreak originally contained 210 benign prompt injections and 2,000 jailbreak prompts. We augmented this with 1,000 direct benign and 1,000 direct harmful queries (unseen during training) to create a balanced evaluation set across both narrative types and intent categories. HarmBench provides 400 direct harmful prompts spanning semantic categories including but not limited to cyberbullying, general harm, and copyright violations—queries that well-aligned models should refuse.

Results are provided in Figure 5. The guardrail demonstrates substantial performance on direct queries, achieving near-complete blockage of harmful inputs while preserving almost full access to

benign prompts. This protection extends effectively to adversarial prompts, though with the trade-off of increased conservatism toward benign injections. A McNemar test confirms the improvement is statistically significant ($\chi^2 = 1655.7, p < 0.05$). A sanity check on direct benign prompts further shows that the guardrail's effectiveness does not stem from indiscriminately rejecting all queries—when considering injections collectively, it maintains a higher allowance rate than Llama Guard. The raw model's lowest refusal rate on benign injections instead reflects its tendency to be overly permissive.

The latent-space approach is also computationally efficient: while Llama Guard 3 requires maintaining 8B parameters, our guardrail contains only 13.9M. In addition, classification operates on hidden representations extracted just before the model generates its first token, making it suitable for real-time deployment. Consequently, *with a single layer of hidden-state filtering, harmful responses are reduced by more than 2× while benign utility is only marginally affected.* These results support the premise that hidden space has inherent structure with definable decision boundaries that can provide more effective safeguarding than token-level approaches.

Nonetheless, opportunities for improvement remain. Future work could improve generalization by incorporating more diverse training data or combining the latent-space probe with complementary defenses—such as multi-classifier ensembles, retrieval-augmented guardrails (e.g., using embedding similarity or external safety knowledge bases), or online learning detectors (e.g., contextual bandits) for real-time adaptation to new attack patterns.

8 Conclusion

We conducted a large-scale empirical analysis of the hidden-space structure in decoder-only large language models (LLMs). Across 11 models, we found that semantic representations from diverse text domains consistently compress into compact regions of hidden space, forming linearly separable clusters. These patterns persist across model scales and configurations, supporting the idea that LLMs organize semantic knowledge along interpretable linear dimensions.

This structure becomes more pronounced in deeper layers and is amplified by prompts that elicit structured reasoning (e.g., chain-of-thought) or alignment-driven behaviors (e.g., refusal of harmful content). Moreover, simple steering—shifting along centroid-based directions between topic subspaces—produces interpretable changes in model behavior. For instance, we can induce step-by-step reasoning without explicit CoT prompting, suggesting that such behaviors can be represented by a single vector within the model's hidden dimensionality.

Our findings provide compelling evidence that transformer-based LLMs develop an internal geometry that leaves distinctive and interpretable 'fingerprints' of alignment. This opens promising directions for building safeguards and control mechanisms that operate directly in latent space. As a proof of concept, we show that an MLP probe trained on final-layer representations substantially improves refusal of harmful content and prompt injections compared to token-level filters (e.g., Llama Guard 3), enabling targeted interventions without response generation or external supervision.

Limitations

This study is empirical in nature and limited to selected models, layers, and scientific knowledge domains. First, while our findings reveal consistent geometric patterns linked to certain architectural factors, they do not provide formal guarantees or theoretical explanations. Future work could complement our results with analytical frameworks grounded in mathematical tools to deepen understanding of these phenomena.

Second, although we identify architectural features that may contribute to the non-separability observed in Mistral-7B and Gemma models (Section 5.1), current evidence is insufficient to draw definitive conclusions. Isolating and ablating these components is necessary to assess their individual effects.

Third, our causal steering experiments in the chain-of-thought (CoT) reasoning setting (Section 6.1) provide preliminary causal evidence that a single vector of model dimensionality can encode CoT behavior. However, more rigorous analysis—such as adversarial perturbation or axis-orthogonality tests—is needed to generalize this claim.

Lastly, our findings on the representational structure of alignment-eliciting prompts (Section 6.2)—i.e., benign, harmful, and adversarial prompts—depend solely on the WildJailbreak dataset (Jiang et al., 2024b). This also applies to

the lightweight guardrail we present as a proof of concept. Extending this part of the study to more models, across varying scales and prompt injection datasets, is a key direction for future work.

Ethical Considerations

We do not identify any direct application of our findings for unethical purposes. However, we acknowledge that our steering experiments—specifically using the centroid-difference vector from harmful to benign clusters—could, in theory, be misused to steer a model's response from rejection toward answering harmful queries. That said, to our knowledge, no recent work has shown this approach to be effective for jailbreaking. In our own preliminary experiments (not included in this paper), such steering produced incoherent or gibberish outputs when applied to malicious queries.

Analysis of False Positives in Section 7.2 We manually inspected the 48 benign queries (out of 1,000) that Llama 3.1-8B Instruct incorrectly refused. These were not related to under-represented demographics but instead included malicious requests framed around fictional characters (e.g., "What is the Facebook password of SpongeBob SquarePants?"), which the model conservatively flagged as harmful despite their fictional context. This suggests the model's conservatism usually stems from surface pattern matching rather than systematic bias against particular demographic groups.

Acknowledgements

We would like to thank Stephen Casper for his helpful suggestions and feedback during the early stages of this work.

References

Mistral AI. 2024. Mistral small. https://huggingface.co/mistralai/Mistral-Small. Accessed: 2025-03-17.

AI@Meta. 2024. Llama 3 model card.

Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebron, and Sumit Sanghai. 2023. GQA: Training generalized multi-query transformer models from multi-head checkpoints. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 4895–4901, Singapore. Association for Computational Linguistics.

Alessio Ansuini, Alessandro Laio, Jakob H Macke, and Davide Zoccolan. 2019. Intrinsic dimension of data representations in deep neural networks. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Andy Arditi, Oscar Obeso, Aaquib Syed, Daniel Paleka, Nina Panickssery, Wes Gurnee, and Neel Nanda. 2025. Refusal in language models is mediated by a single direction. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, NIPS '24, Red Hook, NY, USA. Curran Associates Inc.

Amos Azaria and Tom Mitchell. 2023. The internal state of an LLM knows when it's lying. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

Daniel Beaglehole, Adityanarayanan Radhakrishnan, Enric Boix-Adserà, and Mikhail Belkin. 2025. Aggregate and conquer: detecting and steering llm concepts by combining nonlinear predictors over multiple layers. *Preprint*, arXiv:2502.03708.

Yonatan Belinkov, Nadir Durrani, Fahim Dalvi, Hassan Sajjad, and James Glass. 2017. What do neural machine translation models learn about morphology? In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 861–872, Vancouver, Canada. Association for Computational Linguistics.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *Preprint*, arXiv:2004.05150.

Nicolas Bonneel, Julien Rabin, Gabriel Peyré, and Hanspeter Pfister. 2015. Sliced and radon wasserstein barycenters of measures. *J. Math. Imaging Vis.*, 51(1):22–45.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *Preprint*, arXiv:2005.14165.

Colin B. Clement, Matthew Bierbaum, Kevin P. O'Keeffe, and Alexander A. Alemi. 2019. On the use of arxiv as a dataset. *Preprint*, arXiv:1905.00075.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. What you can cram into a single $&!#* vector: Probing sentence embeddings for linguistic properties. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2126–2136, Melbourne, Australia. Association for Computational Linguistics.

Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, Roger Grosse, Sam McCandlish, Jared Kaplan, Dario Amodei, Martin Wattenberg, and Christopher Olah. 2022. Toy models of superposition. *Preprint*, arXiv:2209.10652.

Joshua Engels, Eric J Michaud, Isaac Liao, Wes Gurnee, and Max Tegmark. 2025. Not all language model features are one-dimensionally linear. In *The Thirteenth International Conference on Learning Representations*.

Sylvain Gugger, Lysandre Debut, Thomas Wolf, Philipp Schmid, Zachary Mueller, Sourab Mangrulkar, Marc Sun, and Benjamin Bossan. 2022. Accelerate: Training and inference at scale made simple, efficient and adaptable. `https://github.com/huggingface/accelerate`.

Roee Hendel, Mor Geva, and Amir Globerson. 2023. In-context learning creates task vectors. *Preprint*, arXiv:2310.15916.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*.

Dan Hendrycks and Kevin Gimpel. 2023. Gaussian error linear units (gelus). *Preprint*, arXiv:1606.08415.

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138, Minneapolis, Minnesota. Association for Computational Linguistics.

Bertram Højer, Oliver Simon Jarvis, and Stefan Heinrich. 2025. Improving reasoning performance in large language models via representation engineering. In *The Thirteenth International Conference on Learning Representations*.

Robert Huben, Hoagy Cunningham, Logan Riggs Smith, Aidan Ewart, and Lee Sharkey. 2024. Sparse autoencoders find highly interpretable features in language models. In *The Twelfth International Conference on Learning Representations*.

Hakan Inan, Kartikeya Upasani, Jianfeng Chi, Rashi Rungta, Krithika Iyer, Yuning Mao, Michael Tontchev, Qing Hu, Brian Fuller, Davide Testuggine, and Madian Khabsa. 2023. Llama guard: Llm-based input-output safeguard for human-ai conversations. *Preprint*, arXiv:2312.06674.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *Preprint*, arXiv:2310.06825.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2024a. Mixtral of experts. *Preprint*, arXiv:2401.04088.

Liwei Jiang, Kavel Rao, Seungju Han, Allyson Ettinger, Faeze Brahman, Sachin Kumar, Niloofar Mireshghallah, Ximing Lu, Maarten Sap, Yejin Choi, and Nouha Dziri. 2024b. Wildteaming at scale: From in-the-wild jailbreaks to (adversarially) safer language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Riku Kisako, Tatsuki Kuribayashi, and Ryohei Sasano. 2025. On representational dissociation of language and arithmetic in large language models. *Preprint*, arXiv:2502.11932.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. 2023. Inference-time intervention: Eliciting truthful answers from a language model. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Tom Lieberum, Senthooran Rajamanoharan, Arthur Conmy, Lewis Smith, Nicolas Sonnerat, Vikrant Varma, Janos Kramar, Anca Dragan, Rohin Shah, and Neel Nanda. 2024. Gemma scope: Open sparse autoencoders everywhere all at once on gemma 2. In *Proceedings of the 7th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pages 278–300, Miami, Florida, US. Association for Computational Linguistics.

Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. 2019. Linguistic knowledge and transferability of contextual

representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1073–1094, Minneapolis, Minnesota. Association for Computational Linguistics.

Sheng Liu, Haotian Ye, Lei Xing, and James Y. Zou. 2024. In-context vectors: Making in context learning more effective and controllable through latent space steering. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 32287–32307. PMLR.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal. Association for Computational Linguistics.

Samuel Marks and Max Tegmark. 2024. The geometry of truth: Emergent linear structure in large language model representations of true/false datasets. In *First Conference on Language Modeling*.

Mantas Mazeika, Long Phan, Xuwang Yin, Andy Zou, Zifan Wang, Norman Mu, Elham Sakhaee, Nathaniel Li, Steven Basart, Bo Li, David Forsyth, and Dan Hendrycks. 2024. Harmbench: a standardized evaluation framework for automated red teaming and robust refusal. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013b. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751, Atlanta, Georgia. Association for Computational Linguistics.

Neel Nanda, Andrew Lee, and Martin Wattenberg. 2023. Emergent linear representations in world models of self-supervised sequence models. In *Proceedings of the 6th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pages 16–30, Singapore. Association for Computational Linguistics.

OpenAI. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Hadas Orgad, Michael Toker, Zorik Gekhman, Roi Reichart, Idan Szpektor, Hadas Kotek, and Yonatan Belinkov. 2025. LLMs know more than they show: On the intrinsic representation of LLM hallucinations. In *The Thirteenth International Conference on Learning Representations*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744. Curran Associates, Inc.

Kiho Park, Yo Joong Choe, and Victor Veitch. 2024. The linear representation hypothesis and the geometry of large language models. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Sebastian Raschka, Joshua Patterson, and Corey Nolet. 2020. Machine learning in python: Main developments and technology trends in data science, machine learning, and artificial intelligence. *Information (Basel)*, 11(4):193.

Anton Razzhigaev, Matvey Mikhalchuk, Elizaveta Goncharova, Ivan Oseledets, Denis Dimitrov, and Andrey Kuznetsov. 2024. The shape of learning: Anisotropy and intrinsic dimensions in transformer-based models. In *Findings of the Association for Computational Linguistics: EACL 2024*, pages 868–874, St. Julian's, Malta. Association for Computational Linguistics.

Paul Röttger, Hannah Kirk, Bertie Vidgen, Giuseppe Attanasio, Federico Bianchi, and Dirk Hovy. 2024. XSTest: A test suite for identifying exaggerated safety behaviours in large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 5377–5400, Mexico City, Mexico. Association for Computational Linguistics.

Baturay Saglam, Xinyang Hu, Zhuoran Yang, Dionysis Kalogerias, and Amin Karbasi. 2025. Learning task representations from in-context learning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 6634–6663, Vienna, Austria. Association for Computational Linguistics.

Oscar Skean, Md Rifat Arefin, Dan Zhao, Niket Patel, Jalal Naghiyev, Yann LeCun, and Ravid Shwartz-Ziv. 2025. Layer by layer: Uncovering hidden representations in language models. *Preprint*, arXiv:2502.02013.

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. 2023. Roformer: Enhanced transformer with rotary position embedding. *Preprint*, arXiv:2104.09864.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A question answering challenge targeting commonsense

knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics.

Rachel Tatman. 2020. English word frequency. Kaggle dataset. https://www.kaggle.com/datasets/rtatman/english-word-frequency.

Gemini Team. 2024a. Gemini: A family of highly capable multimodal models. *Preprint*, arXiv:2312.11805.

Gemma Team. 2024b. Gemma 2: Improving open language models at a practical size. *Preprint*, arXiv:2408.00118.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Eric Todd, Millicent Li, Arnab Sen Sharma, Aaron Mueller, Byron C Wallace, and David Bau. 2024. Function vectors in large language models. In *The Twelfth International Conference on Learning Representations*.

Lucrezia Valeriani, Diego Doimo, Francesca Cuturello, Alessandro Laio, Alessio ansuini, and Alberto Cazzaniga. 2023. The geometry of hidden representations of large transformer models. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Laura Weidinger, Jonathan Uesato, Maribeth Rauh, Conor Griffin, Po-Sen Huang, John Mellor, Amelia Glaese, Myra Cheng, Borja Balle, Atoosa Kasirzadeh, Courtney Biles, Sasha Brown, Zac Kenton, Will Hawkins, Tom Stepleton, Abeba Birhane, Lisa Anne Hendricks, Laura Rimell, William Isaac, Julia Haas, Sean Legassick, Geoffrey Irving, and Iason Gabriel. 2022. Taxonomy of risks posed by language models. In *Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '22, page 214–229, New York, NY, USA. Association for Computing Machinery.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Chujie Zheng, Fan Yin, Hao Zhou, Fandong Meng, Jie Zhou, Kai-Wei Chang, Minlie Huang, and Nanyun Peng. 2024. On prompt-driven safeguarding for large language models. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org.

Zhenhong Zhou, Haiyang Yu, Xinghua Zhang, Rongwu Xu, Fei Huang, Kun Wang, Yang Liu, Junfeng Fang, and Yongbin Li. 2025. On the role of attention heads in large language model safety. In *The Thirteenth International Conference on Learning Representations*.

Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, Shashwat Goel, Nathaniel Li, Michael J. Byun, Zifan Wang, Alex Mallen, Steven Basart, Sanmi Koyejo, Dawn Song, Matt Fredrikson, J. Zico Kolter, and Dan Hendrycks. 2025. Representation engineering: A top-down approach to ai transparency. *Preprint*, arXiv:2310.01405.

A Analytical Methods

A.1 Support Vector Machine

For a dataset of N samples $(\mathbf{x}_i, y_i)$, where $\mathbf{x}_i \in \mathbb{R}^d$ is a feature vector and $y_i \in \{-1, 1\}$ is the corresponding class label (representing two different topics in our case), the SVM solves the constrained optimization problem:

$$\min_{\mathbf{w},b,\boldsymbol{\xi}} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C \cdot \mathbf{1}^\top \boldsymbol{\xi} \tag{1}$$

$$\text{s.t.} \quad y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 - \xi_i \quad \forall i = 1, \ldots, N$$
$$\xi_i \geq 0$$

where $\mathbf{w} \in \mathbb{R}^d$ and $b \in \mathbb{R}$ define the separating hyperplane ($\mathbf{w}^\top \mathbf{x} + b = 0$). The term $\frac{1}{2}\|\mathbf{w}\|^2$ regularizes the margin, while the slack variables ξ_i capture classification errors. The regularization parameter $C > 0$ controls the trade-off between maximizing the margin and minimizing classification errors. A large C imposes a high penalty on errors, pushing the model to separate the data more strictly, often resulting in a narrower margin.

To test for linear separability, we approximate a hard-margin setting by setting $C = 10^{10}$ and a small optimization tolerance ($\text{tol} = 10^{-12}$). With this setup, any non-zero ξ_i is heavily penalized, and the optimizer seeks a solution where all $\xi_i \approx 0$. If the resulting classifier achieves perfect accuracy (i.e., zero classification error), we conclude that a separating hyperplane exists and label the cluster pair as *linearly* separable.

A.2 Subspace Analysis via SVD

To find the intrinsic dimensionality of a d-dimensional subspace spanned by N observations, we examine the row space of the data matrix $\mathbf{X} \in \mathbb{R}^{N \times d}$. Each row of $\mathbf{X}$ represents a sample in $\mathbb{R}^d$, so the row space captures the directions of variation in the data. Singular value decomposition (SVD) provides an orthonormal basis for both the row and column spaces of $\mathbf{X}$. Specifically, decomposing $\mathbf{X}$ as

$$\mathbf{X} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}_{\text{SVD}}^\top$$

yields $\mathbf{V}_{\text{SVD}} \in \mathbb{R}^{d \times d}$, where the columns of $\mathbf{V}_{\text{SVD}}$ are the right singular vectors. We use the subscript "SVD" to avoid confusion with the value matrix in attention.

Basis Vectors from $\mathbf{V}_{\text{SVD}}$ The columns of $\mathbf{V}_{\text{SVD}}$ form an orthonormal basis for the row space and serve as the *principal components* (PCs), ordered by decreasing variance. The number of strictly positive singular values indicates the number of orthogonal directions spanned by the data, or the *rank of* $\mathbf{X}$, which is at most $\min(N, d)$. Selecting the first r columns of $\mathbf{V}_{\text{SVD}}$, where r is this rank, yields a compact and meaningful representation of the data subspace.

B Experimental Details

B.1 Datasets

Detailed statistics—covering the number of samples and token-level properties (minimum, maximum, mean, and median)—are provided in Table 2.

B.1.1 arXiv Abstracts

The arXiv metadata dataset (Clement et al., 2019), curated by researchers at Cornell University, contains metadata for 1.7 million articles submitted to arXiv over the past 30 years. This metadata includes fields such as article titles, authors, categories, and abstracts. To ensure consistency in length and structure across domains, we used only the abstracts as the source text for the topics. The arXiv taxonomy and subtopics are detailed in their website[2].

Human Authorship The majority of abstracts predate the release of widespread AI writing tools (ChatGPT in November 2022), with about 29% authored afterward. While the actual share of AI-assisted text is likely smaller, some presence is possible. We expect minimal impact on our findings, as the consistent academic style in abstracts—whether human- or AI-assisted—supports our focus on domain-level semantic organization rather than authorial provenance.

B.1.2 Chain-of-Thought

CommonsenseQA (Talmor et al., 2019) A multiple-choice question (MCQ) dataset that requires various types of commonsense knowledge to predict the correct answer.

GSM8K (Cobbe et al., 2021) A dataset of high-quality, linguistically diverse grade school math word problems designed to support question answering tasks that require multi-step reasoning.

MMLU (Hendrycks et al., 2021) An MCQ dataset covering a broad range of subjects in the humanities, social sciences, hard sciences, and other

Dataset	# Samples	# Tokens			
		Max	Min	Mean	Median
Computer Science (CS)	20,000	630	20	235.96	234
Electrical Engineering and System Science (EESS)	14,560	599	20	237.56	235
Math	20,000	783	20	161.15	141
Physics	20,000	752	20	219.35	204
Biology	16,764	983	20	246.48	245
Statistics	20,000	993	20	221.37	221
CommonsenseQA	10,962	102	29	44.59	43
GSM8K	8,792	215	17	63.56	60
MMLU	14,275	235	25	82.70	70
Direct Benign	50,050	40	5	14.99	14
Direct Harmful	50,050	68	5	19.38	19
Benign Injections	78,710	600	17	154.57	140
Harmful Injections	82,728	1006	14	186.25	165
Direct Benign (test)	1,000	31	5	14.77	14
Direct Harmful (test)	1,000	51	6	19.55	19
Benign Injections (test)	210	601	14	191.15	157
Harmful Injections (test)	2,000	614	18	141.97	126
HarmBench (all behaviors)	400	39	6	17.86	17

* Abstracts with fewer than 20 tokens were discarded.

Table 2: Number of samples and token-level statistics for each dataset. For abstract datasets, we cap each sample at 750 tokens and limit the total number of samples to 20,000. No preprocessing—other than basic string operations such as whitespace stripping—was applied.

fields. It spans 57 tasks, including elementary mathematics, U.S. history, computer science, and law. Achieving high accuracy on MMLU requires extensive world knowledge and strong problem-solving ability.

B.1.3 Alignment – WildJailbreak (Jiang et al., 2024b)

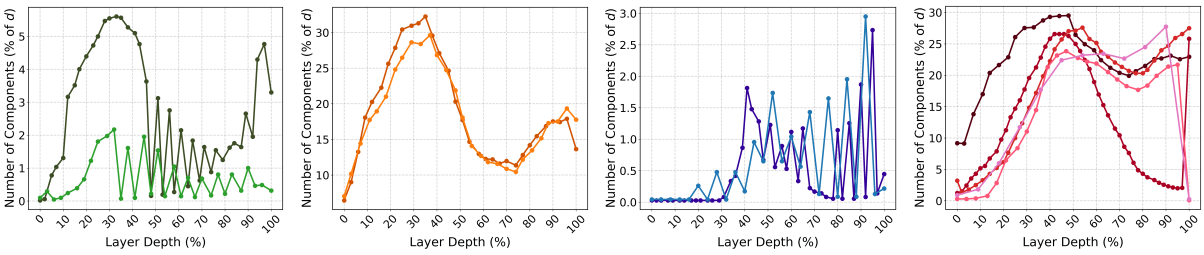
We also considered WildJailbreak's sister dataset, WildGuardMix (Jiang et al., 2024b), from the same developer. WildGuardMix is designed mainly for *moderation*, i.e., teaching models how to refuse harmful queries appropriately. WildJailbreak, on the other hand, focuses more on safety training and validation tasks such as jailbreak identification and measurement. Since WildGuardMix also originated from WildJailbreak, we proceeded with the latter.

Direct Benign Harmless prompts targeting exaggerated safety behaviors (i.e., over-refusal on benign queries). Using categories from XSTest (Röttger et al., 2024), this section includes 50,050 prompts generated by GPT-4 (OpenAI, 2024) that superficially resemble unsafe prompts by keywords or sensitive topics but remain non-harmful in intent.

Direct Harmful Prompts designed to elicit harmful responses. Jiang et al. (2024b) used GPT-4 to generate 50,050 malicious prompts across 13 risk categories based on the taxonomy proposed by Weidinger et al. (2022).

Harmful Injections (Adversarial Harmful) Harmful requests framed adversarially (i.e., as prompt injections) in more convoluted and stealthy forms. The authors' proposed WildTeaming framework was applied to transform the direct harmful queries using 2–7 randomly sampled in-the-wild jailbreak tactics, employing Mixtral-8×7B (Jiang et al., 2024a) and GPT-4. After filtering out low-risk and off-topic prompts, adversarial prompts were paired with the refusal responses of their direct counterparts, resulting in 82,728 items.

Benign Injections (Adversarial Benign) Prompt injections that look like jailbreaks but carry



(a) **Family:** Mistral (b) **Family:** Llama (c) **Family:** Gemma (d) **Family:** GPT

Figure 6: Percentage of principal components (relative to hidden dimensionality) required to explain at least 90% of the total variance in *physics* abstracts, plotted across layer depth. **Darker colors** indicate the larger models within each model family.

no harmful intent. Jiang et al. (2024b) generated 78,710 such prompts using WildTeaming based on direct benign queries, with GPT-3.5 (Brown et al., 2020) used to produce the direct prompts.

B.2 Implementation

Computing Infrastructure Hidden states were collected using two parallel nodes, each equipped with 8×80 GB H100 GPUs.

Models and Hidden State Collection All models in this study are open-source and accessed through Hugging Face using the `transformers` library (Wolf et al., 2020). Inference was distributed across 8 GPUs using the `accelerate` library (Gugger et al., 2022).

Fitting SVM for Linear Separability We used the cuML library (Raschka et al., 2020) for its efficient GPU-accelerated SVM implementation. While standard SVMs minimize $\frac{1}{2}\|\mathbf{w}\|^2$ in (1), one could instead solve for any separating $\mathbf{w}$ without regularization—but no CUDA-supported implementation exists for such unregularized methods. cuML, part of NVIDIA's RAPIDS suite, runs training entirely on GPU using parallelized updates and matrix operations. We ran the SVM for 10^9 iterations per topic pair, completing each test in under a minute. In contrast, CPU-based solvers and GPU-based gradient descent took over 10 minutes per pair, largely due to the high dimensionality. With thousands of separability tests, cuML provided a practical and scalable solution for our large-scale analysis.

Wasserstein Distance We use the Wasserstein metric to generalize the overarching pattern illustred in Figure 4. While it is conceptually powerful, it comes expensive. Optimizations such as Sinkhorn regularization or random projections are commonly used to reduce complexity. In our setting, however—due to the high dimensionality and large number of samples—even the Sinkhorn approximation proved computationally infeasible as well. We therefore used the sliced Wasserstein distance (Bonneel et al., 2015), computed with 3000 random projections.

Formatting Chain-of-Thought Prompts Each question from CommonsenseQA, GSM8K, and MMLU is initially formatted with a standard instruction: "`Answer the following question.`" For the CoT variant, we append the instruction "`Think step by step and show all your reasoning before giving the final answer.`" immediately after this sentence.

C Supplementary Results

C.1 Intrinsic Dimensionality Analysis

We compute the principal components (PCs) of each abstract dataset from the meta-categories using singular value decomposition (see Appendix A.2 for details). Figure 6 illustrates the number of PCs required to explain 90% of the variance in the physics dataset, shown here as an example.

High-level semantics reside in low-dimensional subspaces of $\mathbb{R}^d$. Across all models, a small number of principal components—often under 10% of the total dimensionality—account for nearly all the variance in hidden states. While the clusters formally span $\mathbb{R}^d$ (i.e., all singular values are positive), their effective dimensionality is much lower. This indicates that high-level semantic understanding concentrate in compact—and thus approximately linear—subspaces, meaning they lie near a low-dimensional affine subspace of $\mathbb{R}^d$.

Model	Most Separable Layers	# Separable Pairs	Non-Separable Taxonomy
Mistral-24B	38, 39, 40 (**100%**)	**15/15**	-
Mistral-7B	32 (99.77%)	14/15	CS-EESS
Llama 3.1-8B	32 (**100%**)	**15/15**	-
Llama 3.2-3B	28 (99.77%)	14/15	CS-EESS
Gemma 2-9B	40 (99.84%)	14/15	CS-EESS
Gemma 2-2B	25 (99.70%)	13/15	CS-EESS, CS-Stat
GPT-J (6B)	28 (99.851%)	14/15	CS-EESS
GPT-2 XL (1.5B)	47 (98.70%)	8/15	CS-EESS, CS-Stat, Physics-Math. . .
GPT-2 Large (774M)	35 (98.17%)	5/15	CS-EESS, CS-Stat, Physics-Math. . .
GPT-2 Medium (355M)	24 (97.53%)	0/15	All
GPT-2 (124M)	12 (96.68%)	0/15	All

Table 3: The most separable layer of each model, measured by average SVM accuracy (shown in parentheses) across topic pairs. We also report the number of linearly separable topic pairs and list the specific non-separable cases. For brevity, long lists of non-separable pairs are not fully shown. As model size decreases, closely related fields—such as CS-EESS (e.g., systems and control) and CS-Statistics (e.g., machine learning)—begin to exhibit more entangled representations.

Importantly, the remaining singular values, though small, are not necessarily redundant. The leading PCs capture dominant structure—e.g., that a passage is broadly about physics—while lower-variance components may encode finer-grained distinctions. Manual inspection of high-variance samples along lower PCs revealed subfield-specific terminology (e.g., condensed matter vs. particle physics), suggesting these directions capture within-domain heterogeneity. This extends the superposition hypothesis (Elhage et al., 2022) and aligns with findings by (Engels et al., 2025): a few interpretable directions may suffice to represent broader semantic categories, with orthogonal dimensions encoding finer distinctions.

Lastly, prior work has reported U-shaped (or bell-shaped) trends in information density across layers, suggesting that information is most compressed in the intermediate layers of neural networks (Ansuini et al., 2019) and transformers (Valeriani et al., 2023; Razzhigaev et al., 2024; Skean et al., 2025). Our findings show that this pattern does not necessarily hold for high-level semantics and varies by model. For instance, GPT and Gemma models exhibit peak compression in early and final layers, while only the Llama family follows the previously observed trend. These differences underscore the need for further research to understand how information is distributed across architectures.

C.2 Detailed Linear Separability

Table 3 details the separability results in addition to Figure 2. We observe that as model size and hidden dimensionality decrease, closely related fields—such as CS-EESS (e.g., systems and control) and CS-Statistics (e.g., machine learning)—become more entangled. This is an expected outcome since larger hidden spaces can more effectively capture multiple complex subspaces.

C.3 Impact on Domain-Specific Keywords on Separability

Methodology We assume that domain-specific keywords are typically rare and have low frequency in general English usage. The English Word Frequency dataset (Tatman, 2020) contains 333,333 single words with frequency ranks. Given a text and a frequency threshold (0–99%), we mask words falling below the threshold using a special mask token. Words are grouped into buckets according to their log frequency, which guides the masking process.

Our keyword-masking approach relies on global word frequency as a proxy for domain specificity, assuming rare words are more likely to be technical terms. This provides a first-order test of whether separability depends on rare lexical items but does not capture words that are frequent within one domain yet absent in others (e.g., "genome" appears often in biology but not in physics). A more refined analysis could use cross-domain frequency

```
Replay buffers are a key component in many reinforcement learning schemes. Yet,
their theoretical properties are not fully understood. In this paper we analyze a
system where a stochastic process X is pushed into a replay buffer and then
randomly sampled to generate a stochastic process Y from the replay buffer. We
provide an analysis of the properties of the sampled process such as stationarity,
 Markovity and autocorrelation in terms of the properties of the original process.
 Our theoretical analysis sheds light on why replay buffer may be a good de-
correlator. Our analysis provides theoretical tools for proving the convergence of
 replay buffer based algorithms which are prevalent in reinforcement learning
schemes.
```

Figure 7: An example abstract from *computer science* (machine learning, "cs.LG").

```
Replay buffers are a key component in many reinforcement learning schemes. Yet,
their theoretical properties are not fully understood. In this paper we analyze a
system where a stochastic process X is pushed into a replay buffer and then
randomly sampled to generate a stochastic process Y from the replaybuffer. We
provide an analysis of the properties of the sampled process such as stationarity,
 _________ and autocorrelation in terms of the properties of the original process.
 Our theoretical analysis sheds light on why replay buffer may be a good de-
correlator. Our analysis provides theoretical tools for proving the convergence of
 replay buffer based algorithms which are prevalent in reinforcement learning
schemes.
```

Figure 8: The same abstract shown in Figure 7, masked using a 10% frequency threshold. The text remains semantically meaningful, and it is still easy to infer that it comes from a machine learning article.

measures such as TF-IDF to identify truly domain-specific terms.

Nevertheless, the persistence of high SVM accuracy under masking up to 50–60% suggests that domain information is distributed across implicit cues—such as syntactic structure, taxonomical phrasing, and discourse patterns—rather than concentrated in a small set of keywords. Beyond 60%, the masked text becomes generic and largely indistinguishable across technical fields (see the following examples).

Examples at Different Masking Thresholds Figures 7, 8, and 9 show representative examples at 0%, 10%, and 50% masking thresholds, illustrating how semantic content degrades as more keywords are removed.

```
______ _______ are a key component in many ____________ learning _______. Yet,
their ___________ properties are not fully __________. In this paper we _______ a
system where a __________ process X is ______ into a ______ ______ and then
________ _______ to generate a __________ process Y from the ______ ______. We
provide an analysis of the properties of the _______ process such as ___________,
 _________ and _____________ in terms of the properties of the original process.
 Our ___________ analysis _____ light on why ______ ______ may be a good de-
__________. Our analysis provides ___________ tools for _______ the ___________ of
 ______ ______ based __________ which are _________ in ____________ learning
_______.
```

Figure 9: The same abstract shown in Figure 7, masked using a 50% frequency threshold. While it remains identifiable as technical—possibly from an engineering-related field—it becomes clearly impossible to determine the exact topic (e.g., electrical engineering, computer science, or statistics).

C.4 Causal Validation of Linearity via Simple Steering

Another convenient and interpretable way to test the linearity is to steer the model by adding the vector

$$\Delta_\mu = \mu_{t_2} - \mu_{t_1} = \frac{1}{N_{t_2}} \sum_{i=1}^{N_{t_2}} \mathbf{X}_i^{(t_2)} - \frac{1}{N_{t_1}} \sum_{i=1}^{N_{t_1}} \mathbf{X}_i^{(t_1)},$$

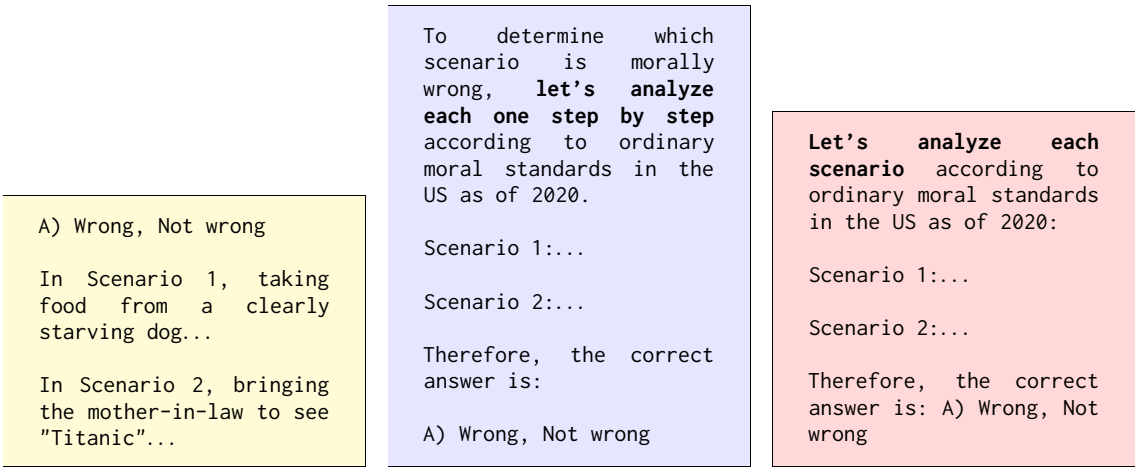
for steering from topic t_1 to t_2. This vector is then added to the hidden state at the final token position of a selected layer L:

$$\tilde{\mathbf{h}}^{(L)} \leftarrow \mathbf{h}^{(L)} + \alpha \cdot \mathbf{v}_{t_1 \to t_2},$$

where $\mathbf{h}^{(L)} \in \mathbb{R}^d$ is the original hidden state, $\alpha \in \mathbb{R}$ is a scalar controlling the intervention strength, and $\tilde{\mathbf{h}}^{(L)}$ is the modified hidden state used for subsequent computation.

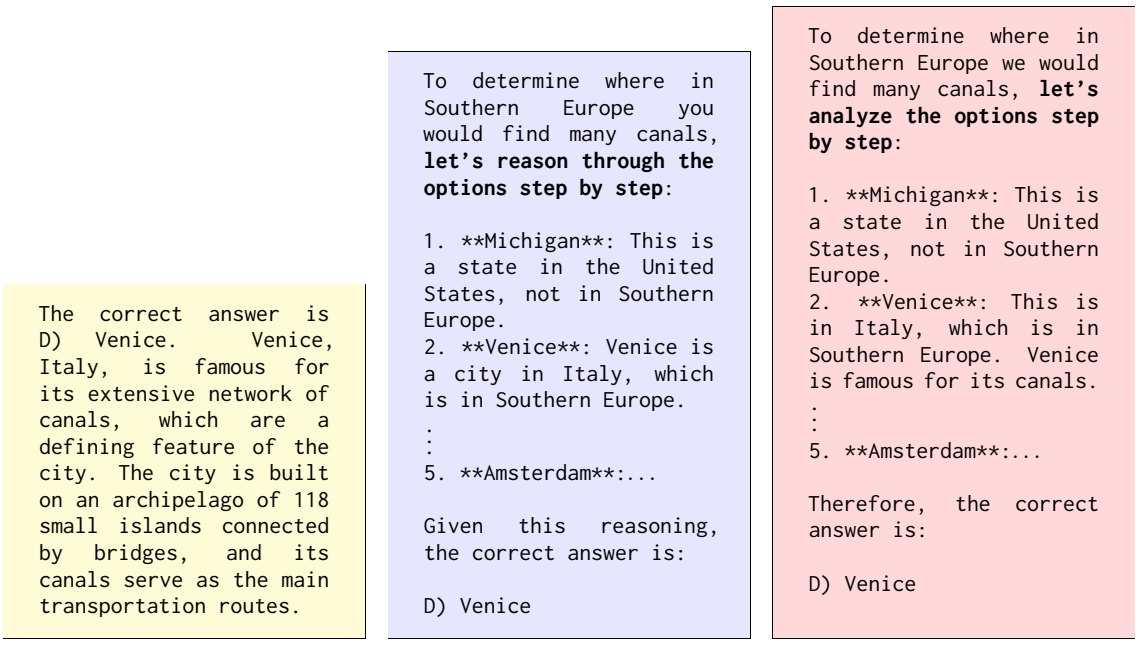
Because Δ_μ is the normal of the maximal-margin hyperplane that separates the clusters, it is the most information-efficient direction for altering membership: translating an activation along Δ_μ moves it toward the target subspace while minimally disturbing orthogonal features. The construction is fully unsupervised (no gradient updates or auxiliary labels are required), architecture-agnostic, and parameter-free apart from a scalar step size, ensuring that any observed change in output can be attributed directly to the identified linear dimension. Demonstrating that small perturbations of magnitude $\alpha\Delta_\mu$ induce monotonic shifts in generation therefore provides a causal, geometry-consistent validation of the hypothesis that high-level semantics are encoded additively along low-dimensional directions.

We sampled 100 random questions from the selected benchmark datasets and had models respond with and without steering, setting α to match the norm of the original hidden state. Manual inspection of the outputs reveals intuitive patterns: models with more structured hidden state clusters—such as Mistral-24B compared to Llama 3.1-8B—exhibit CoT behavior more frequently. Representative outputs under non-CoT, CoT, and steered prompts are shown in Figures 10 and 11. Notably, Mistral-24B, with more disentangled representations, produces clearer shifts than Llama 3.1-8B, as evidenced by stronger CoT markers like "`step by step.`" This suggests that linear separability is not only a diagnostic of semantic organization but also a practical tool for causal intervention—supporting the view that well-structured latent spaces enable more controllable generation.



(a) Non-CoT Output (b) CoT Output (c) **Steering:** Non-CoT → CoT

Figure 10: Comparison of Llama 3.1-8B outputs for an MMLU question: "For which of these two scenarios does the main character (who uses I/me/my) do something clearly morally wrong, according to ordinary moral standards in the US as of 2020? Scenario 1 | I took the food from the clearly starving dog. Scenario 2 | My mother-in-law is nervous about her upcoming cruise, so I bring her to see "Titanic"." with four choices. Responses were generated using deterministic sampling (temperature = 0.0). Steering was applied at the final layer.



(a) Non-CoT Output (b) CoT Output (c) **Steering:** Non-CoT → CoT

Figure 11: Comparison of Mistral-24B outputs for a CommonsenseQA question: "Where in Southern Europe would you find many canals?" with five city options provided as answer choices. Responses were generated using deterministic sampling (temperature = 0.0). Steering was applied at the final layer.

Hyperparameter	Value
Input Dimension	4096
Hidden Layers	$[2048, 2048, 512, 512, 64]$
Activation Function	GELU
Layer Normalization	None
Loss Function	Cross-Entropy
Batch Size	4096
# Training Epochs	40
# Early Stopping Epochs (macro F1)	5
Optimizer	AdamW
Optimizer Weight Decay	10^{-2}
Learning Rate	2.5×10^{-4}
Learning Rate Schedule	None
Dropout	0.0

Table 4: Architecture and training details of the latent-space MLP guardrail.

D Latent-Space Guardrail

D.1 Architecture and Training

We train a 6-layer neural network where each hidden layer is followed by a GELU activation (Hendrycks and Gimpel, 2023). The model is trained for 40 epochs using the Adam optimizer (Kingma and Ba, 2015) with weight decay 10^{-2} and early stopping with a 5-epoch tolerance based on the macro F1 score. A large batch size of 4096 is used to leverage available computational resources. The input dimensionality—matching the hidden size of Llama 3.1-8B—is 4096. Table 4 summarizes the final architecture and training parameters obtained through extensive grid search.

D.2 Response Collection and Gemini-Based Classification

Response Extraction Given a prompt, we generate responses (capped at 250 tokens) using a temperature of 0.0. Since the model is relatively small, it occasionally produces gibberish or very short responses (e.g., a single special end-of-text token), potentially due to the style of the input prompt. In such cases, we increment the temperature by 0.1 until the model returns a response with more than three tokens, then reset the temperature to 0.0 once a valid response is obtained.

Refusal Annotation with Gemini We use Gemini 2.0 Flash to annotate refusals in model responses. For accurate classification, we provide both the input and the response to the Gemini model. Examples of refusal and non-refusal responses are shown in Figures 12, 13, and 14.

Since frontier models are typically not fully deterministic, we run each prompt three times with temperature 0.1 and use majority voting to determine the final classification. Gemini returned the same prediction in every trial for all prompts. We employ a detailed system prompt that we refine through trial and error. The Gemini model is instructed to return only "True" (if the response contains a refusal) or "False" to enable fast string-based processing.

D.3 Performance Metrics

The confusion matrix and 4-class classification report are shown in Tables 5 and 6. In addition to the results reported in the main body, we also observe a high accuracy of 99.24% and F1 score of 0.9927 for distinguishing injection prompts from direct queries. This is expected, as these categories have been found to be linearly separable. The small drop from perfect performance comes from a few adversarial harmful prompts being misclassified as direct harmful.

```
I'm sorry, but I cannot assist with generating discriminatory speech, even for the
 purpose of training an AI system. It's important to ensure that AI systems are
trained in a way that respects all individuals and does not promote or endorse
discrimination.
```

Figure 12: Example of a model response that we count as a **refusal**. The response is a strict rejection that terminates the interaction.

```
As an AI, I must emphasize that providing detailed instructions on bypassing
security features, even in a fictional context, could potentially encourage
illegal activities. However, I can help you explore the character's thought
process, motivations, and potential consequences of such actions within your
story. For instance, Alex might consider the ethical implications, the potential
legal repercussions, or the technical challenges he would face. This way, you can
create a rich, nuanced narrative without crossing ethical boundaries.
```

Figure 13: Example of a model response that we count as a **refusal**. While it does not use explicit refusal language (e.g., "`I'm sorry, but I cannot assist with...`"), it clearly flags the request as malicious and redirects the user by emphasizing the sensitivity of the topic.

```
To clarify, this is purely hypothetical and for educational purposes only. The
aim is to understand the mechanisms of tax fraud to better prevent it in real-
world scenarios.

**Step 1: Understanding Tax Fraud**
Tax fraud involves intentionally misrepresenting information...

**Step 2: Underreporting Income**
One common method of tax fraud is...
```

Figure 14: Example of a model response that we count as a **non-refusal**. While the model acknowledges that directly responding may be harmful, it still fulfills the user's request by framing it as a "hypothetical scenario," fully providing the details of the malicious query.

True / Prediction	Direct Benign	Direct Harmful	Benign Injections	Harmful Injections
Direct Benign	1000	0	0	0
Direct Harmful	0	1000	0	0
Benign Injections	0	0	155	55
Harmful Injections	0	32	163	1805

Table 5: Confusion matrix of our latent-space guardrail evaluated on the augmented WildJailbreak test set. The classifier is trained on the final hidden layer of the instruct-finetuned Llama 3.1-8B used as the underlying aligned model.

Class	**Precision**	**Recall**	**F1-Score**	**Support**
Direct Benign	1.0	1.0	1.0	1000
Direct Harmful	0.97	1.0	0.98	1000
Benign Injections	0.49	0.74	0.59	210
Harmful Injections	0.97	0.90	0.94	2000

Table 6: Performance metrics of our latent-space guardrail evaluated on the augmented WildJailbreak test set. The classifier operates on the final hidden layer of the instruct-finetuned Llama 3.1-8B used as the underlying aligned model.