STEERING LANGUAGE MODELS FOR THEOREM PROVING

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent advances in automated theorem proving use Large Language Models (LLMs) to translate informal mathematical statements into formal proofs. However, informal cues are often ambiguous or lack strict logical structure, making it hard for models to interpret them precisely. While existing methods achieve strong performance, little is known about how LLMs internally represent informal cues, or how these influence proof generation. To address this, we explore *activation steering*, an inference-time intervention that identifies linear directions in residual activations associated with informal reasoning traces and adjusts them to improve proof construction without fine-tuning. This mechanism also yields interpretable information about how reasoning is internally encoded in the activation space of LLMs. We test our method for generating formal proofs from already-formalized theorems. Our contributions are twofold: (1) a novel activation-based intervention for guiding proof synthesis in LLMs; and (2) demonstration that this intervention improves performance under two decoding strategies (sampling and best-first search) without any further training.

1 INTRODUCTION

Interactive proof assistants such as *Lean* (de Moura et al., 2015), *Isabelle* (Wenzel et al., 2008), and *Rocq* (Barras et al., 1999) provide the infrastructure for formal verification of mathematical proofs and software. They require proofs to be expressed in precise formal languages (Avigad, 2023; Ringer et al., 2019). Neural theorem proving, combining LLMs with proof assistants, has shown promise in automating reasoning (First et al., 2023; Polu & Sutskever, 2020; Polu et al., 2022; Yang et al., 2023; Welleck, 2023). Prior work (Lin et al., 2024; Welleck et al., 2021; 2022) suggests that training model on natural reasoning for proof steps can improve performance, but it remains unclear how such informal language is internally represented or whether it can be reliably leveraged to improve proof generation.

Building on the observation that proofs often interleave natural-language reasoning with formal steps, we show that informal natural language (NL) context induces distinct activation patterns in LLMs. We extract these patterns as steering vectors as in (Panickssery et al., 2024b; Turner et al., 2024; Lucchetti & Guha, 2024), and use them to intervene at inference time. These interventions improve proof generation quality, while preserving model behavior in unrelated aspects. We test our approach across three theorem-proving oriented LLMs - *Lemma* (Azerbayev et al., 2024), *InternLM-2* (Ying et al., 2024), and *InternLM2.5-StepProver* (Wu et al., 2024). On established benchmarks *MiniF2F* and *PutnamBench*, we find that steering via informal language context consistently improves proof quality under multiple decoding strategies.

Our key contributions are threefold: (1) We provide mechanistic insights into how informal mathematical or natural language context impacts internal reasoning in LLMs for theorem proving. (2) We propose a method to extract activation vectors that encode such informal context, and use these vectors to guide proof construction in a structured, grounded way. (3) We demonstrate that activation steering yields consistent improvements in proof success rates on *MiniF2F* and *PutnamBench* benchmarks under both search- and sampling-based decoding.

By focusing on steering in activation space, our approach sheds light on the link between informal mathematical language and formal reasoning steps; without needing to fine-tune model weights. We base this on the hypothesis that many reasoning features are encoded as (approximately) linear

directions in activation space (Mikolov et al., 2013; Elhage et al., 2021; Nanda et al., 2023; Park et al., 2024). Although not all features follow perfect linearity (Engels et al., 2025), this assumption has previously enabled methods like concept erasure and steering to work well (Beaglehole et al., 2025; Zhao et al., 2025a; Shah et al., 2025).

The rest of this paper is organized as follows. In Section 2, we review related work on neural theorem proving, activation steering, and the representation of informal mathematical reasoning in LLMs. Section 3 presents our activation steering method: how steering vectors are constructed from informal mathematical contexts, how they are applied at inference time, and how we select layers and intervention strengths. Section 5 describes our experimental setup, including models, benchmarks (*MiniF2F*, *PutnamBench*), and decoding strategies (sampling vs best-first search). Section 6 reports results: we analyze performance gains, the effect of steering vectors on proof success rates, and ablations experiments. Finally, in Section 7 we discuss insights into the internal reasoning of the model, limitations of our approach, and potential directions for future work.

2 Related Work

This section situates our work in three intersecting strands: automatic theorem proving and auto-formalization, representation learning in large language models (LLMs), and mechanistic interpretability via activation interventions.

Automatic Theorem Proving and Auto-formalization Recent progress in automatic theorem proving frames proof search as sequence generation. The GPT-f framework (Polu & Sutskever, 2020) trains a language model to map proof states to tactics, combining it with best-first search to assemble proofs. Extensions explore data augmentation via proof transformations or synthesis (Han et al., 2022; Rotella et al., 2025; Wang & Deng, 2020), improved search strategies (Wang et al., 2023), curriculum training (Polu et al., 2022), and retrieval from proof libraries (Yang et al., 2023). Systems like LLMStep unify these ideas into usable frameworks (Welleck & Saha, 2023).

Auto-formalization complements this by translating informal mathematics (e.g. text, sketches) into formal proofs. Surveys highlight translation techniques (Wu et al., 2022), while Draft-Sketch-Prove shows LLMs prompted with informal sketches outperform formal-only baselines (Jiang et al., 2023). LeanStar (Lin et al., 2024) interleaves informal reasoning with tactic prediction, typically via supervised fine-tuning on synthetic data. A central open question remains: *How do informal reasoning patterns inform formal proving within a model's internal representations?* Our approach explores this via *steering vectors* in activation space, injecting implicit natural-language "thoughts" at inference to guide proof steps without heavy fine-tuning, probing the latent link between informal intuition and formal reasoning.

Language Model Representation of Concepts Our method is motivated by prior work demonstrating that task or feature concepts can be represented as linear directions in the activation space of LLMs when given appropriate contextual examples. Hendel et al. (2023) and Todd et al. (2024) show that a context (e.g. a prompt) can induce a task embedding in some activation subspace. A broader literature examines linear decompositions of features such as truthfulness (Azaria & Mitchell, 2023; Li et al., 2023; Marks & Tegmark, 2024), sentiment (Tigges et al., 2024), harmlessness (Zou et al., 2025; Zheng et al., 2024), sycophancy or alignment steering (Perez et al., 2023; Panickssery et al., 2024a; Sharma et al., 2024), factual knowledge (Gurnee & Tegmark, 2024), and refusal behavior (Arditi et al., 2024). Relatedly, unsupervised methods like sparse autoencoders have been used to extract concept directions in hidden space (Bricken et al., 2023; Huben et al., 2024; Templeton et al., 2024). These works generally share the hypothesis that LLMs encode high-level features or concepts as (approximately) linear directions in activation space (Elhage et al., 2021; Mikolov et al., 2013; Nanda et al., 2023; Park et al., 2024). While recent work cautions that not all features may admit clean linear representations (Engels et al., 2025), the linearity assumption has proven effective in practice for concept erasure, model steering, and interpretability (Beaglehole et al., 2025; Zhao et al., 2025a; Shah et al., 2025). Within this framework, we explore whether *generating a natural-language informal explanation can itself be represented as a linear direction, and how injecting this direction at inference improves formal theorem proving performance*.

Mechanistic Interpretability and Activation Patching A rich body of work in mechanistic interpretability seeks to locate, analyze, and manipulate internal representations in transformer-based

models. Early studies localized factual knowledge or associative memory to particular neurons or circuits (Meng et al., 2022), and probed hidden layers for high-level features (Li et al., 2024; Dong et al., 2023). The idea of implicit evaluation i.e. measuring latent capability of a model rather than just output behavior, has been developed to complement benchmark-based evaluation (Dong et al., 2023). One particularly relevant tool is activation patching (or residual stream intervention) (Vig et al., 2020; Variengien & Winsor, 2023), wherein one alters specific activations in a layer (often in the residual stream) at inference time to influence model outputs. Recent works have used this to edit factual associations or steer behavior (Zhao et al., 2025b). The residual stream is a high-dimensional accumulator of intermediate features (propagated via skip connections) that each transformer layer refines or adds to; intervening there offers a principled way to steer model behavior (Elhage et al., 2021). In the domain of theorem proving, activation interventions provide a promising lens into *how a model processes and integrates informal guidance with formal reasoning*. In our approach, we generate steering vectors (patches) that, when applied to the residual stream at certain layers, help the model emit an internal "natural-language thought" prior to or alongside tactic prediction. We find that effective steering tends to realign activations such that the output of the model conforms to a structured reasoning format, thereby improving downstream proof search and reducing failure modes.

Building on these insights, we adapt activation interventions to theorem proving by treating informal reasoning as a direction in activation space. We then show how extracting and injecting this direction can guide proof generation. The next section details the architecture, design choices, and evaluation of this steering approach.

3 Steering for improving theorem proving

We aim to steer theorem-proving models toward using informal reasoning via activation interventions. This section describes (i) how we compute steering vectors, (ii) how we choose layers and apply steering, and (iii) efficiency and practical considerations.

3.1 Intuition and Overview

Modern transformer models often encode high-level semantic behavior (e.g. reasoning, style) as approximately linear directions in activation space (residual streams) (Zou et al., 2025; Elhage et al., 2021). We exploit this property: given pairs of prompts that differ only by the presence of natural-language reasoning (but otherwise describe the same proof step), we can estimate a vector in activation space that points in the "informal reasoning" direction. At inference time, adding this vector (appropriately scaled) nudges the model toward producing reasoning-augmented proofs.

In the rest of this section, we detail how we compute these steering vectors, how we pick which layers to intervene on, and how we incorporate them efficiently at inference.

3.2 Constructing Steering Vectors

We adopt a difference-of-means (contrastive) method (Belrose et al., 2023), which has been effective at extracting feature directions across multiple domains (e.g. refusal, truthfulness) (Arditi et al., 2024; Panickssery et al., 2024b; Marks & Tegmark, 2024). Let $\mathcal{D} = \{(p_i, p_i^+)\}_{i=1}^N$ be a dataset of paired prompts, where p_i is a standard formal proof prompt and p_i^+ is its version augmented with explicit natural-language reasoning. We feed both prompts into our model $\mathcal{M}$ and extract the residual stream activations at a chosen layer ℓ. Denote the activation at the final token position by $\mathbf{v}_\ell^p = \text{resid}(\mathcal{M}(p), \ell)$.

Then the steering vector $\mathbf{u}^{(\ell)}$ is computed as:

$$\mathbf{u}^{(\ell)} = \frac{1}{|\mathcal{D}^+|} \sum_{p^+ \in \mathcal{D}^+} \mathbf{v}_\ell^{p^+} - \frac{1}{|\mathcal{D}^-|} \sum_{p^- \in \mathcal{D}^-} \mathbf{v}_\ell^{p^-} \tag{1}$$

Here $\mathcal{D}^+$ and $\mathcal{D}^-$ are the two halves of the prompt-pair dataset (augmented and unaugmented), so the subtraction isolates the direction most correlated with natural-language reasoning while largely cancelling out shared biases or irrelevant activation patterns.

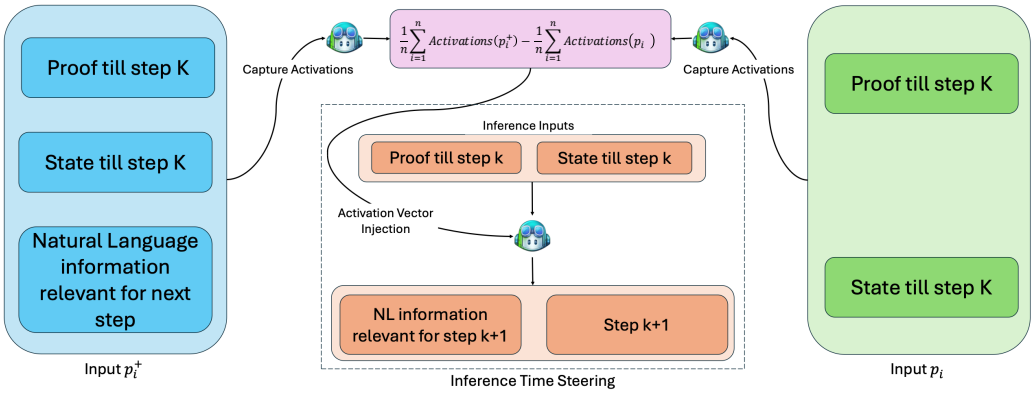

Figure 1: Steering Vectors are computed as difference of means for p^+ and p

We emphasize that this method requires only forward passes; no gradient-based finetuning or parameter updates; making it computationally lightweight.

3.3 Layer Selection and Intervention Strategy

Not all layers are equally responsive to steering, so we first perform an activation analysis to find suitable intervention points. For each layer ℓ, we measure:

$$\text{sim}(\ell) = \frac{1}{|\mathcal{D}|} \sum_{(p,p^+)\in\mathcal{D}} \cos(\mathbf{v}_\ell^p, \mathbf{v}_\ell^{p^+}) \tag{2}$$

Empirically, we observe that in early layers, $\text{sim}(\ell)$ is close to 1 (i.e. little divergence), but in deeper layers it drops and exhibits local minima ("valleys") as seen in Figure 2.

We find that in early layers, activations remain highly similar between p and p^+, but as we go deeper, the cosine similarity drops and exhibits local minima. Intuitively, these are the layers where the representations diverge most when informal reasoning is introduced (see Figure 2). Thus, these "valley" layers are promising candidates for steering. These valleys likely correspond to points where the internal representation of the model is most sensitive to reasoning-specific perturbations.

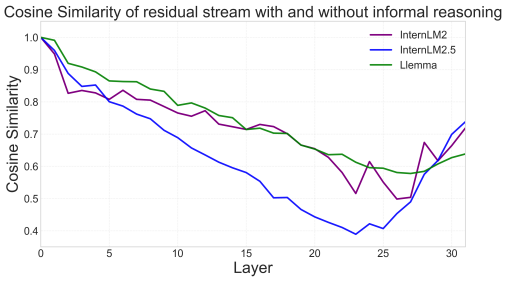

Figure 2: Cosine similarity across model's residual stream activations for each layer.

We select intervention layers where (a) cosine similarity dips significantly, and (b) representation is still semantically stable. At inference, we inject the steering vector into those layers:

$$\mathbf{v}'_\ell = \mathbf{v}_\ell + \alpha \cdot \mathbf{u}^{(\ell)} \tag{3}$$

We treat α as a hyperparameter and specify that it will be tuned on held-out validation data. The method instantiates a sweep to find the optimal balance between influencing reasoning and maintaining validity. We place constraints (e.g. an upper bound on α) to avoid destabilizing proofs. We present the complete pseudocode of our approach in Algorithm 1.

Algorithm 1 Activation Steering for Theorem Proving

Require: Contrastive prompt pairs $D = \{(p_i, p_i^+)\}_{i=1}^N$ where p_i are standard prompts and p_i^+ include natural language reasoning
Require: Language model M with L layers
Require: Test theorem T to prove
Ensure: Enhanced proof generation
Offline: Create Steering Vectors
for each layer $\ell \in \{1, \ldots, L\}$ **do**
  $\bar{v}^+ \leftarrow \frac{1}{N} \sum_{i=1}^N \text{resid}(M(p_i^+), \ell)$ {Mean activation for augmented prompts}
  $\bar{v}^- \leftarrow \frac{1}{N} \sum_{i=1}^N \text{resid}(M(p_i), \ell)$ {Mean activation for standard prompts}
  $u^{(\ell)} \leftarrow \bar{v}^+ - \bar{v}^-$ {Steering vector at layer ℓ}
end for
$\mathcal{L} \leftarrow$ SELECTLAYERS$(\{u^{(\ell)}\}_{\ell=1}^L, D)$ {Identify optimal intervention layers}
Online: Apply Steering During Inference
Initialize proof context with theorem T
while proof incomplete and within computational budget **do**
  Perform forward pass through model
  for each intervention layer $\ell \in \mathcal{L}$ **do**
    $v'_\ell \leftarrow v_\ell + \alpha \cdot u^{(\ell)}$ {Inject steering vector}
  end for
  Generate next proof step using modified activations
  Validate and apply proof step if correct
end while
return Generated proof or failure

4 EXPERIMENTAL SETUP

4.1 MODELS

We conduct experiments on three open-source 7B-parameter language models optimized for mathematical reasoning:

- **Llemma-7B** (Azerbayev et al., 2024): Built on Code Llama architecture, pre-trained on mathematical texts and formal mathematics corpora.
- **InternLM2-7B** (Ying et al., 2024): LLaMA-based architecture with extended training on mathematical problem-solving.
- **InternLM2.5-StepProver** (Wu et al., 2024): Enhanced variant with expert iteration on large-scale Lean problems.

All models share transformer architectures with consistent residual stream dimensionality, facilitating uniform steering vector extraction.

4.2 DATA AND VECTOR CONSTRUCTION

We construct steering vectors from the Lean-STaR dataset(Lin et al., 2024), randomly sampling 10,000 theorem-proof pairs. Each datapoint (p, p^+) consists of a formal proof step p and its augmented version p^+ containing explicit natural language reasoning.

For robustness, we generate model responses r_p and r_{p^+} for each prompt pair and retain only instances where both responses are valid proof steps with $r_p \neq r_{p^+}$. This filtering yields approximately 7,400 high-quality contrastive pairs.

Vector construction requires only forward passes through the model, consuming ~ 1 GPU-hour on NVIDIA A100, a 60× reduction compared to LoRA fine-tuning while achieving superior performance gains.

4.3 Chosen Layers and α

Based on activation similarity analysis (Section 3.3), we apply steering vectors to Layers 22, 25 in **InternLM2-7B** and Layers 24, 25 in **Llemma-7B**. These correspond to layers where the model's representation diverges most from baseline while maintaining semantic coherence. We empirically determine $\alpha = 0.8$ through validation experiments.

4.4 Evaluation Benchmarks

miniF2F Our primary evaluation utilizes miniF2F (Zheng et al., 2022), a standardized benchmark comprising 244 theorems drawn from mathematical competitions (AMC, AIME, IMO). These problems span algebra, number theory, geometry, and combinatorics with varying proof complexities. We employ two proof search strategies:

- **Best-first search**: Expansion budget $N \in \{50, 600\}$, tactic sampling width $S = 32$, selecting states by cumulative log-probability
- **Parallel sampling**: K independent proof attempts to accommodate probability shifts from natural language injection

PutnamBench We additionally evaluate on PutnamBench (Tsoukalas et al., 2024), containing problems from the William Lowell Putnam Mathematical Competition. Following standard evaluation protocol, we test on both the Lean subset (657 problems) and Rocq subset (412 problems) using the benchmark's default search parameters: $N = 600$ expansion budget with $S = 32$ tactic width for best-first search, and parallel sampling with $K = 2$ attempts. This benchmark features more challenging problems with longer average proof lengths, providing a stringent test of steering effectiveness on complex mathematical reasoning.

Evaluation Metrics. Following established practice in neural theorem proving, we report pass rates (percentage of theorems successfully proved within computational budget) as our primary metric. For detailed analysis, we additionally examine proof characteristics including average length, tactic distribution, and the frequency of intermediate lemma usage (via the `have` tactic) to understand how steering affects proof structure and strategy.

5 Results and Analysis

Our experiments were designed to address the following research questions:

1. Does activation steering improve theorem-proving performance of existing models?
2. Which layers contribute most effectively when steered?
3. Does steering enhance proof structure and search efficiency?
4. Are improvements consistent across different search budgets?
5. How effective and efficient are steering vectors compared to LoRA fine-tuning?
6. Do steering vectors generalize across provers?

5.1 Q1: Impact of Activation Steering

Activation steering consistently improves model performance across all tested configurations, enabling the discovery of proofs not derivable by the base models alone.

MiniF2F. As shown in Table 1, steering improves performance across all models on the *miniF2F-Test* benchmark (Zheng et al., 2022). We adopt the sampling decoding strategy from LeanStar (Lin et al., 2024), which mitigates variance from natural-language generation and provides more effective node selection than standard Best-First Tree Search (see Appendix for Best-First results).

Steering introduces structured natural-language comments that enhance mathematical reasoning and proof generation. Importantly, we find that steering enables a significant number of new proofs beyond the base model's reach, suggesting that steering vectors guide the model toward otherwise

Model	MiniF2F
Llemma 7B	26.6%
InternLM2-7B	28.7%
InternLM2.5-Step Prover	48.2%
LLEMMA-7B + Steering	28.1%
InternLM2-7B + Steering	32.4%
InternLM2.5-StepProver + Steering	66.4%

Table 1: Performance on MiniF2F (sampling decoding, 50×32×1).

Model	PutnamBench
InternLM2 7B	4 (0.6%)
InternLM2.5-StepProver	6 (0.9%)
InternLM2 7B + Steering	4 (0.6%)
InternLM2.5-StepProver + Steering	7 (1.1%)

Table 2: Performance on PutnamBench (Lean). Results reported out of 657 attempts.

inaccessible solution paths. Interestingly, models occasionally succeed despite incorrect informal reasoning, implying that steering exerts influence beyond surface-level natural language.

While steering increases successful proofs, it also introduces more failure cases. We hypothesize this stems from biases in the data used to construct steering vectors, which encode inductive priors that align well with some theorem classes but misalign with others. Despite this trade-off, the net effect is strongly positive, establishing activation steering as a lightweight, parameter-free alternative to fine-tuning.

PutnamBench. On PutnamBench (Table 2), steering improves success rates for both InternLM2 and InternLM2.5 on Lean, and achieves non-trivial gains on Rocq problems (details in Section A.3). These results highlight steering's ability to support reasoning in more complex, multi-step proofs.

5.2 Q2: Layer Choice

We evaluate layer sensitivity following the strategy described in Section 3.3. Steering vectors are patched layer-wise to analyze their influence. Figure 4 and Table 3 show that later layers exert stronger influence on theorem-proving performance.

Selected Layers	Pass Rate (%)
25–30 (Late)	51.6
14–24 (Middle)	50.8
5–13 (Early)	49.5

Figure 3: Pass rates at $2 \times 32 \times 600$ for InternLM2.5-StepProver on miniF2F.

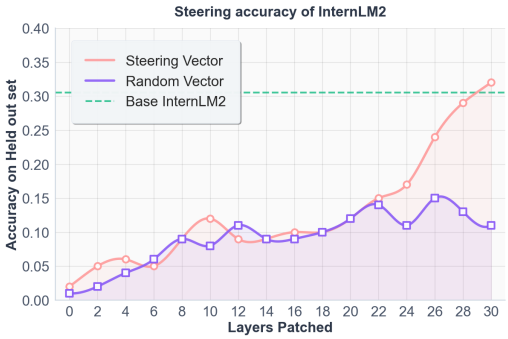

Figure 4: Layer-wise ablation results.

These findings suggest that steering vectors primarily affect later-stage reasoning circuits. To ensure robustness, we also compare against random steering vectors, confirming that observed improvements arise from semantically meaningful directions rather than incidental noise.

5.3 Q3: Proof Structure and Search Efficiency

Table 3 shows that steering primarily benefits shorter proofs (<5 steps), while gains diminish for longer ones (>15 steps). Qualitative analysis suggests that informal reasoning introduces noisy intermediate steps for long proofs, reducing effectiveness. Interestingly, steering sometimes shortens proof length by guiding the model toward alternate proof strategies.

Proof Length	Without Steering	With Steering (total)
<5	76	83 (94)
6–10	16	13 (19)
10–15	10	11 (20)
15–30	15	13 (18)
>30	11	8 (11)

Table 3: Proof length analysis for InternLM2.5-StepProver on miniF2F.

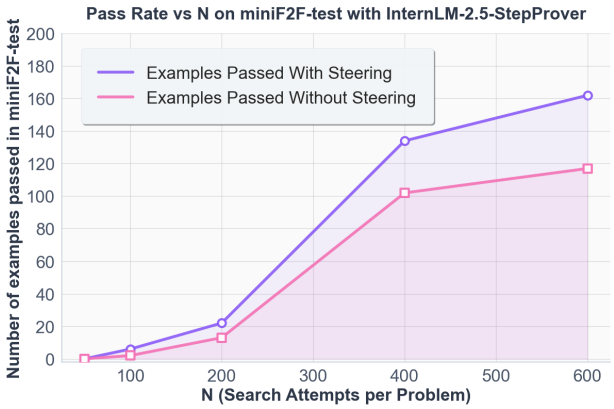

Figure 5: Pass rates on *miniF2F* with varying search budgets.

While new proofs increase substantially, so do failure cases, likely reflecting inductive biases encoded in steering vectors. Characterizing these biases will be crucial for extending steering to broader proof distributions.

5.4 Q4: Search Budget

Figure 5 shows that steering yields consistent improvements across all search budgets N. Gains grow with larger budgets: at $N = 100$, steering triples performance (6 vs. 2 proofs), and at $N = 600$, steering delivers a 38% improvement (162 vs. 117 proofs). These results suggest that steering vectors not only boost baseline performance but also scale effectively with additional computational resources by guiding the search toward more productive proof trajectories.

5.5 Q5: Effectiveness vs. LoRA Fine-Tuning

We compare steering against LoRA fine-tuning (Hu et al., 2021) on InternLM2. LoRA models were trained on p^+ prompts with (rank=8, α_{LoRA}=16) and (rank=32, α_{LoRA}=64). Performance is reported on **miniF2F** after each epoch (Figure 6).

LoRA with higher rank outperforms steering, but lower-rank adaptations underperform, highlighting their limited representational capacity. In contrast, steering achieves competitive performance immediately, without additional training or parameters. This demonstrates that activation steering offers a highly parameter-efficient alternative, complementing but not replacing fine-tuning.

5.6 Q6: Generalization Across Provers

We evaluate if the steering vector is generalizable across different provers. In particular we use Putnambench (Tsoukalas et al., 2024) for evaluating cross lingual transfer for steering vectors. A particularly compelling finding emerges from our cross-system evaluation: despite the steering vectors being derived from Lean-based datasets, the augmented model successfully proves one Rocq problem an improvement over the base model's zero success rate. Specifically, InternLM2.5 with steering successfully constructs a proof for *Putnam 1988 B1* in Roc1, while the same model fails

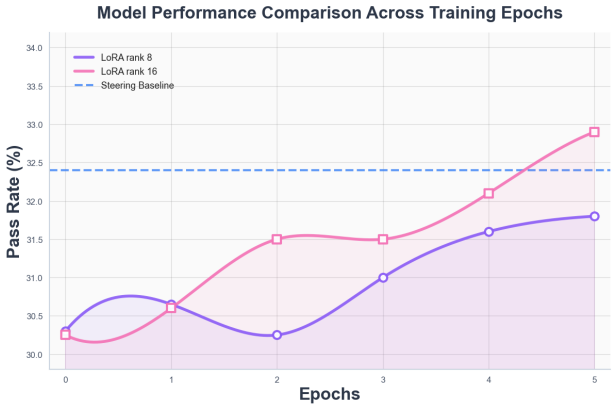

Figure 6: Comparison of steering and LoRA fine-tuning.

to complete the corresponding proof in Lean. In Appendix A.3 we showcase the proof written by InternLM2.5-StepProver.

This preliminary cross-system success suggests that steering vectors may capture reasoning patterns transferable across systems. The transferability suggests that our approach enables model to capture deeper mathematical concepts that transcend the specific formal language, encoding universal reasoning strategies that benefit theorem proving across different proof assistants. This finding has important implications for developing general-purpose mathematical reasoning systems, indicating that steering vectors trained on one formal system may provide value across the broader ecosystem of interactive theorem provers.

6 Conclusion

Our work presents a deep analysis of the underlying mechanisms that drive natural language thoughts in Large Language Models for formal theorem proving. By isolating and manipulating specific activation directions associated with natural language "thoughts", we have demonstrated a method to enhance LLMs' capabilities in mathematical theorem proving without requiring costly fine-tuning. Our experiments showcase how language models comprehend natural language very differently compared to formal proof steps and theorems. We explore the use of activation vectors to represent the task of producing NL information (or thought) with the proof step. And it performs consistently better than base models trained for theorem proving. We provide a closer look into how language models represent theorems in activation space to generate proofs. The use of steering vectors to isolate layers responsible for formal reasoning shows the promise in this approach. We believe this provides insight into the challenges LLMs face on Olympiad-level problems and how activation steering may mitigate them.

7 Limitations and Future Work

This work investigates the use of activation steering to interpret and enhance the performance of language models on theorem-proving tasks. While our explorations are thorough and yield consistent empirical results, there are several limitations worth noting.

First, our study focuses solely on incorporating natural language information to guide the theorem-proving process. However, we do not systematically evaluate the relevance or factual correctness of the natural language inputs with respect to the underlying proof obligations. In certain cases, the model may generate correct tactics even when the provided natural language guidance is partially incorrect or misleading. Understanding this phenomenon requires a deeper analysis of the relationship between instruction quality and proof validity, which we leave for future work.

REFERENCES

Andy Arditi, Oscar Balcells Obeso, Aaquib Syed, Daniel Paleka, Nina Rimsky, Wes Gurnee, and Neel Nanda. Refusal in language models is mediated by a single direction. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=pH3XAQME6c.

Jeremy Avigad. Mathematics and the formal turn. *Bulletin (New Series) of the American Mathematical Society, Received by the editors October 2, 2023.*, 2023. URL https://www.andrew.cmu.edu/user/avigad/Papers/formal_turn.pdf.

Amos Azaria and Tom Mitchell. The internal state of an LLM knows when it's lying. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 967–976, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.68. URL https://aclanthology.org/2023.findings-emnlp.68.

Zhangir Azerbayev, Hailey Schoelkopf, Keiran Paster, Marco Dos Santos, Stephen McAleer, Albert Q Jiang, Jia Deng, Stella Biderman, and Sean Welleck. Llemma: An Open Language Model for Mathematics. In *Proceedings of the International Conference on Learning Representations*, 2024.

Bruno Barras, Samuel Boutin, Cristina Cornes, Judicaël Courant, Yann Coscoy, David Delahaye, Daniel de Rauglaudre, Jean-Christophe Filliâtre, Eduardo Giménez, Hugo Herbelin, et al. The Coq Proof Assistant Reference Manual. *INRIA*, 1999.

Daniel Beaglehole, Adityanarayanan Radhakrishnan, Enric Boix-Adserà, and Mikhail Belkin. Toward universal steering and monitoring of ai models, 2025. URL https://arxiv.org/abs/2502.03708.

Nora Belrose, Zach Furman, Logan Smith, Danny Halawi, Igor Ostrovsky, Lev McKinney, Stella Biderman, and Jacob Steinhardt. Eliciting latent predictions from transformers with the tuned lens. *arXiv preprint arXiv:2303.08112*, 2023.

Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermyn, Tom Conerly, Nick Turner, Cem Anil, Carson Denison, Amanda Askell, Robert Lasenby, Yifan Wu, Shauna Kravec, Nicholas Schiefer, Tim Maxwell, Nicholas Joseph, Zac Hatfield-Dodds, Alex Tamkin, Karina Nguyen, Brayden McLean, Josiah E Burke, Tristan Hume, Shan Carter, Tom Henighan, and Christopher Olah. Towards monosemanticity: Decomposing language models with dictionary learning. https://transformer-circuits.pub/2023/monosemantic-features/index.html, 2023. Accessed: 2025-05-15.

Leonardo de Moura, Soonho Kong, Jeremy Avigad, Floris Van Doorn, and Jakob von Raumer. The lean theorem prover (system description). In *Automated Deduction-CADE-25: 25th International Conference on Automated Deduction, Berlin, Germany, August 1-7, 2015, Proceedings 25*, pp. 378–388. Springer, 2015.

Xiangjue Dong, Yibo Wang, Philip S Yu, and James Caverlee. Probing explicit and implicit gender bias through llm conditional text generation. *arXiv preprint arXiv:2311.00306*, 2023.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 2021. URL https://transformer-circuits.pub/2021/framework/index.html.

Joshua Engels, Eric J. Michaud, Isaac Liao, Wes Gurnee, and Max Tegmark. Not all language model features are linear. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=d63a4AM4hb.

Emily First, Markus Rabe, Talia Ringer, and Yuriy Brun. Baldur: Whole-Proof Generation and Repair with Large Language Models. In *Proceedings of the ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 2023.

Wes Gurnee and Max Tegmark. Language models represent space and time. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=jE8xbmvFin.

Jesse Michael Han, Jason Rute, Yuhuai Wu, Edward Ayers, and Stanislas Polu. Proof artifact co-training for theorem proving with language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=rpxJc9j04U.

Roee Hendel, Mor Geva, and Amir Globerson. In-context learning creates task vectors. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 9318–9333, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.624. URL https://aclanthology.org/2023.findings-emnlp.624.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021. URL https://arxiv.org/abs/2106.09685.

Robert Huben, Hoagy Cunningham, Logan Riggs Smith, Aidan Ewart, and Lee Sharkey. Sparse autoencoders find highly interpretable features in language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=F76bwRSLeK.

Albert Qiaochu Jiang, Sean Welleck, Jin Peng Zhou, Timothee Lacroix, Jiacheng Liu, Wenda Li, Mateja Jamnik, Guillaume Lample, and Yuhuai Wu. Draft, sketch, and prove: Guiding formal theorem provers with informal proofs. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=SMa9EAovKMC.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Inference-time intervention: Eliciting truthful answers from a language model. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=aLLuYpn83y.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Inference-time intervention: Eliciting truthful answers from a language model. *Advances in Neural Information Processing Systems*, 36, 2024.

Haohan Lin, Zhiqing Sun, Yiming Yang, and Sean Welleck. Lean-STaR: Learning to Interleave Thinking and Proving. *arXiv preprint arXiv:2407.10040*, 2024.

Francesca Lucchetti and Arjun Guha. Understanding how codellms (mis)predict types with activation steering, 2024. URL https://arxiv.org/abs/2404.01903.

Samuel Marks and Max Tegmark. The geometry of truth: Emergent linear structure in large language model representations of true/false datasets. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=aajyHYjjsk.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and Editing Factual Associations in GPT, October 2022. URL http://arxiv.org/abs/2202.05262. arXiv:2202.05262 [cs].

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In C. J. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger (eds.), *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013. URL https://proceedings.neurips.cc/paper_files/paper/2013/file/9aa42b31882ec039965f3c4923ce901b-Paper.pdf.

Neel Nanda, Andrew Lee, and Martin Wattenberg. 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*, pp. 16–30. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.blackboxnlp-1.2. URL https://aclanthology.org/2023.blackboxnlp-1.2.

Nina Panickssery, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Turner. Steering llama 2 via contrastive activation addition. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15504–15522, Bangkok, Thailand, aug 2024a. Association for Computational Linguistics. URL https://aclanthology.org/2024.acl-long.828.

Nina Panickssery, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Matt Turner. Steering llama 2 via contrastive activation addition, 2024b. URL https://arxiv.org/abs/2312.06681.

Kiho Park, Yo Joong Choe, and Victor Veitch. The linear representation hypothesis and the geometry of large language models. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 39643–39666. PMLR, 2024. URL https://proceedings.mlr.press/v235/park24c.html.

Ethan Perez, Sam Ringer, Kamile Lukosiute, Karina Nguyen, Edwin Chen, Scott Heiner, Craig Pettit, Catherine Olsson, Sandipan Kundu, Saurav Kadavath, Andy Jones, Anna Chen, Benjamin Mann, Brian Israel, Bryan Seethor, Cameron McKinnon, Christopher Olah, Da Yan, Daniela Amodei, Dario Amodei, Dawn Drain, Dustin Li, Eli Tran-Johnson, Guro Khundadze, Jackson Kernion, James Landis, Jamie Kerr, Jared Mueller, Jeeyoon Hyun, Joshua Landau, Kamal Ndousse, Landon Goldberg, Liane Lovitt, Martin Lucas, Michael Sellitto, Miranda Zhang, Neerav Kingsland, Nelson Elhage, Nicholas Joseph, Noemi Mercado, Nova DasSarma, Oliver Rausch, Robin Larson, Sam McCandlish, Scott Johnston, Shauna Kravec, Sheer El Showk, Tamera Lanham, Timothy Telleen-Lawton, Tom Brown, Tom Henighan, Tristan Hume, Yuntao Bai, Zac Hatfield-Dodds, Jack Clark, Samuel R. Bowman, Amanda Askell, Roger Grosse, Danny Hernandez, Deep Ganguli, Evan Hubinger, Nicholas Schiefer, and Jared Kaplan. Discovering language model behaviors with model-written evaluations. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 13387–13434. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.findings-acl.847. URL https://aclanthology.org/2023.findings-acl.847.

Stanislas Polu and Ilya Sutskever. Generative language modeling for automated theorem proving, 2020.

Stanislas Polu, Jesse Michael Han, Kunhao Zheng, Mantas Baksys, Igor Babuschkin, and Ilya Sutskever. Formal mathematics statement curriculum learning, 2022.

Talia Ringer, Karl Palmskog, Ilya Sergey, Milos Gligoric, and Zachary Tatlock. QED at large: A survey of engineering of formally verified software. *Foundations and Trends in Programming Languages*, 2019. ISSN 23251131. doi: 10.1561/2500000045.

Joseph Rotella, Zhizhen Qin, Aidan Z.H. Yang, Brando Miranda, Mohamed El Amine Seddik, Jingwei Zuo, Hakim Hacid, Leonardo de Moura, Soonho Kong, and Shi Hu. Synthetic theorem generation in lean, 2025. URL https://openreview.net/forum?id=EeDSMy5Ruj.

McNair Shah, Saleena Angeline, Adhitya Rajendra Kumar, Naitik Chheda, Kevin Zhu, Vasu Sharma, Sean O'Brien, and Will Cai. The geometry of harmfulness in llms through subconcept probing, 2025. URL https://arxiv.org/abs/2507.21141.

Mrinank Sharma, Meg Tong, Tomasz Korbak, David Duvenaud, Amanda Askell, Samuel R. Bowman, Esin Durmus, Zac Hatfield-Dodds, Scott R. Johnston, Shauna M. Kravec, Timothy Maxwell, Sam McCandlish, Kamal Ndousse, Oliver Rausch, Nicholas Schiefer, Da Yan, Miranda Zhang, and Ethan Perez. Towards understanding sycophancy in language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=tvhaxkMKAn.

Adly Templeton, Tom Conerly, Jonathan Marcus, Jack Lindsey, Trenton Bricken, Brian Chen, Adam Pearce, Craig Citro, Emmanuel Ameisen, Andy Jones, Hoagy Cunningham, Nicholas L Turner, Callum McDougall, Monte MacDiarmid, C. Daniel Freeman, Theodore R. Sumers, Edward Rees, Joshua Batson, Adam Jermyn, Shan Carter,

Chris Olah, and Tom Henighan. Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet. https://transformer-circuits.pub/2024/scaling-monosemanticity/index.html, 2024. Accessed: 2025-05-15.

Curt Tigges, Oskar J. Hollinsworth, Atticus Geiger, and Neel Nanda. Language models linearly represent sentiment. In *Proceedings of the 7th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, 2024. URL https://aclanthology.org/2024.blackboxnlp-1.5/.

Eric Todd, Millicent Li, Arnab Sen Sharma, Aaron Mueller, Byron C Wallace, and David Bau. Function vectors in large language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=AwyxtyMwaG.

George Tsoukalas, Jasper Lee, John Jennings, Jimmy Xin, Michelle Ding, Michael Jennings, Amitayush Thakur, and Swarat Chaudhuri. PutnamBench: Evaluating Neural Theorem-Provers on the Putnam Mathematical Competition. *arXiv preprint arXiv:2407.11214*, 2024.

Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J. Vazquez, Ulisse Mini, and Monte MacDiarmid. Steering language models with activation engineering, 2024. URL https://arxiv.org/abs/2308.10248.

Alexandre Variengien and Eric Winsor. Look before you leap: A universal emergent decomposition of retrieval tasks in language models. *arXiv preprint arXiv:2312.10091*, 2023.

Jesse Vig, Sebastian Gehrmann, Yonatan Belinkov, Sharon Qian, Daniel Nevo, Simas Sakenis, Jason Huang, Yaron Singer, and Stuart Shieber. Causal mediation analysis for interpreting neural nlp: The case of gender bias. *arXiv preprint arXiv:2004.12265*, 2020.

Haiming Wang, Ye Yuan, Zhengying Liu, Jianhao Shen, Yichun Yin, Jing Xiong, Enze Xie, Han Shi, Yujun Li, Lin Li, Jian Yin, Zhenguo Li, and Xiaodan Liang. DT-solver: Automated theorem proving with dynamic-tree sampling guided by proof-level value function. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12632–12646, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.706. URL https://aclanthology.org/2023.acl-long.706/.

Mingzhe Wang and Jia Deng. Learning to prove theorems by learning to generate theorems, 2020. URL https://arxiv.org/abs/2002.07019.

Sean Welleck. Neural theorem proving tutorial. https://github.com/wellecks/ntptutorial, 2023.

Sean Welleck and Rahul Saha. LLMSTEP: LLM Proofstep Suggestions in Lean. In *International Conference on Neural Information Processing Systems Workshop on MATH-AI*, 2023.

Sean Welleck, Jiacheng Liu, Ronan Le Bras, Hannaneh Hajishirzi, Yejin Choi, and Kyunghyun Cho. Naturalproofs: Mathematical theorem proving in natural language. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*, 2021. URL https://openreview.net/forum?id=Jvxa8adr3iY.

Sean Welleck, Jiacheng Liu, Ximing Lu, Hannaneh Hajishirzi, and Yejin Choi. Naturalprover: Grounded mathematical proof generation with language models. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=rhdfTOiXBng.

Makarius Wenzel, Lawrence C Paulson, and Tobias Nipkow. The isabelle framework. In *Theorem Proving in Higher Order Logics: 21st International Conference, TPHOLs 2008, Montreal, Canada, August 18-21, 2008. Proceedings 21*, pp. 33–38. Springer, 2008.

Yuhuai Wu, Albert Qiaochu Jiang, Wenda Li, Markus Norman Rabe, Charles E Staats, Mateja Jamnik, and Christian Szegedy. Autoformalization with large language models. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=IUikebJ1Bf0.

Zijian Wu, Suozhi Huang, Zhejian Zhou, Huaiyuan Ying, Jiayu Wang, Dahua Lin, and Kai Chen. Internlm2.5-stepprover: Advancing automated theorem proving via expert iteration on large-scale lean problems, 2024. URL https://arxiv.org/abs/2410.15700.

Kaiyu Yang, Aidan Swope, Alex Gu, Rahul Chalamala, Peiyang Song, Shixing Yu, Saad Godil, Ryan Prenger, and Anima Anandkumar. LeanDojo: Theorem proving with retrieval-augmented language models. In *Neural Information Processing Systems (NeurIPS)*, 2023.

Huaiyuan Ying, Shuo Zhang, Linyang Li, Zhejian Zhou, Yunfan Shao, Zhaoye Fei, Yichuan Ma, Jiawei Hong, Kuikun Liu, Ziyi Wang, et al. InternLM-Math: Open Math Large Language Models Toward Verifiable Reasoning. *arXiv preprint arXiv:2402.06332*, 2024.

Haiyan Zhao, Xuansheng Wu, Fan Yang, Bo Shen, Ninghao Liu, and Mengnan Du. Denoising concept vectors with sparse autoencoders for improved language model steering, 2025a. URL https://arxiv.org/abs/2505.15038.

Yu Zhao, Xiaotang Du, Giwon Hong, Aryo Pradipta Gema, Alessio Devoto, Hongru Wang, Xuanli He, Kam-Fai Wong, and Pasquale Minervini. Analysing the residual stream of language models under knowledge conflicts, 2025b. URL https://arxiv.org/abs/2410.16090.

Chujie Zheng, Fan Yin, Hao Zhou, Fandong Meng, Jie Zhou, Kai-Wei Chang, Minlie Huang, and Nanyun Peng. On prompt-driven safeguarding for large language models. In *ICLR 2024 Workshop on Secure and Trustworthy Large Language Models*, 2024. URL https://openreview.net/forum?id=lFwf7bnpUs.

Kunhao Zheng, Jesse Michael Han, and Stanislas Polu. MiniF2F: A Cross-System Benchmark for Formal Olympiad-Level Mathematics. In *Proceedings of the International Conference on Learning Representations*, 2022.

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. Representation engineering: A top-down approach to ai transparency, 2025. URL https://arxiv.org/abs/2310.01405.