Check for updates

MACHINE LEARNING

Toward universal steering and monitoring of AI models

Daniel Beaglehole[1]†, Adityanarayanan Radhakrishnan[2,3]†*, Enric Boix-Adserà[4], Mikhail Belkin[1,5]*

Artificial intelligence (AI) models contain much of human knowledge. Understanding the representation of this knowledge will lead to improvements in model capabilities and safeguards. Building on advances in feature learning, we developed an approach for extracting linear representations of semantic notions or concepts in AI models. We showed how these representations enabled model steering, through which we exposed vulnerabilities and improved model capabilities. We demonstrated that concept representations were transferable across languages and enabled multiconcept steering. Across hundreds of concepts, we found that larger models were more steerable and that steering improved model capabilities beyond prompting. We showed that concept representations were more effective for monitoring misaligned content than for using judge models. Our results illustrate the power of internal representations for advancing AI safety and model capabilities.

Modern artificial intelligence (AI) models achieve expert level performance on a broad range of scientific, mathematical, and factual assessment tasks (*1*, *2*). Yet our understanding of their internal representation of knowledge remains elusive, making it challenging to use them to the best of their capabilities and build safeguards. Indeed, models' responses can change in complex and hard-to-predict ways based on prompting. For example, simply prompting large language models (LLMs) to show their work before answering questions [an example of chain-of-thought prompting (*3*)] often improves accuracy. Similarly, certain adversarial modifications of the prompt can induce a model to override built-in safeguards, such as those against providing instructions on building a bomb (*4*).

Substantial recent effort has gone into understanding how semantic notions or concepts such as hallucinations, harmfulness, deception, and many others are represented in LLMs. Indeed, a major goal is to identify patterns in the internal activations of LLMs that track these concepts. Upon identification, these patterns could be used for both steering of model responses toward or away from the concept and monitoring whether a concept is active during response generation.

An important line of work (*5*–*8*) approached concept steering and monitoring by using unsupervised learning techniques such as sparse autoencoders. These techniques were used to discover concept-related features stored in LLMs without specifying concepts in advance. This approach has led to the discovery of certain steerable concepts in LLMs. For example, researchers at Anthropic used this approach to identify activations related to the "Golden Gate Bridge" concept in Claude. Perturbing these activations led to the model "obsessing" about the bridge (*6*). For instance, in (*9*), when Claude was asked, "What is your physical form?" the steered model responded with, "I am the Golden Gate Bridge, a famous suspension bridge that spans the San Francisco Bay. My physical form is the iconic bridge itself [...]," demonstrating how the model incorporated the concept of the Golden Gate Bridge within the context of the response despite having no sensory access to the actual bridge.

However, a challenge with unsupervised approaches is that they are not guaranteed to surface patterns that track a specific concept of interest. Extra effort is required to identify concepts of interest within a large set of putative concepts. Given this challenge, others turned to supervised learning approaches (*10*–*20*). The majority of these works relied on probing, which involves training predictive models to extract concept features. In particular, recent work (*21*) and discussions (*22*) have argued for the effectiveness of probing over unsupervised methods for both steering and monitoring. Despite the effort in this area, it is unclear how well these probing methods can effectively surface concept features for steering and monitoring. Extraction and characterization of general concept representations in large-scale AI models thus remains a fundamental open problem.

In this work, we developed an effective, scalable method for extracting linear representations of general concepts in the activations of large-scale AI models. We showed that these representations enabled steering and monitoring of a broad set of concepts. Our results shed light on fundamental properties of concept space, such as concept transferability across language and the ability to mix concept representations. Key to our study was the use of a recently developed algorithm for feature extraction, known as the Recursive Feature Machine (RFM) (*23*), which provided a scalable and effective approach for extracting concept-related features. An overview of our approach is presented in Fig. 1.

Our general approach enabled steering concepts across a broad set of model architectures [LLMs, vision language models (VLMs), and reasoning models] and model scales (8 billion to 90 billion parameters). Through steering, we exposed and mitigated model vulnerabilities and also improved capabilities across these various architectures. By studying steering across 512 concepts and three models from the Llama LLM series (*24*), we found that newer, larger, and better performing models were also more steerable.

We demonstrated that concept features extracted using our methodology were effective for monitoring LLM hallucinations and toxic content. Across six benchmarks, predictive models built using concept features outperformed LLMs that judged outputs directly (including GPT-4o and LLMs specifically fine-tuned for the task).

Our scalable approach can be readily integrated into standard inference pipelines. For example, using a single NVIDIA Ampere series (A100) graphics processing unit (GPU), it took under 1 min and fewer than 500 training samples to identify features and steer toward a concept of interest. As such, we envision that our work will open new avenues for systematically mapping the representation of concepts within large-scale AI models and for developing effective, cost-efficient alternatives to standard fine-tuning methods of deep neural networks.

Results

Concept steering and monitoring through RFMs

We perform concept steering and monitoring in AI models by using RFMs, which are algorithms for learning features with general nonlinear predictive models. We begin with a brief primer on LLMs and the RFM algorithm below.

LLM overview and nomenclature: LLMs are neural network models trained on tokenized text given a vocabulary of d tokens. LLMs are trained to predict the next token in a sequence given t previous tokens. An LLM implements a map, $f: \mathbb{R}^{t\times d} \to \mathbb{R}^d$, where any input $X \in \mathbb{R}^{t\times d}$ is a sparse matrix with entry (i,j) containing a 1 if token j occurs in position i of the input sequence and zero otherwise.

[1]Department of Computer Science and Engineering, University of California San Diego, La Jolla, CA, USA. [2]Massachusetts Institute of Technology, Cambridge, MA, USA. [3]Broad Institute of MIT and Harvard, Cambridge, MA, USA. [4]Wharton School, University of Pennsylvania, Philadelphia, PA, USA. [5]Halıcıoğlu Data Science Institute, University of California San Diego, La Jolla, CA, USA. *Corresponding author. Email: mbelkin@ucsd.edu (M.B.); aradha@mit.edu (A.R.)
†These authors contributed equally to this work.

Downloaded from https://www.science.org at University of California San Diego on October 26, 2026

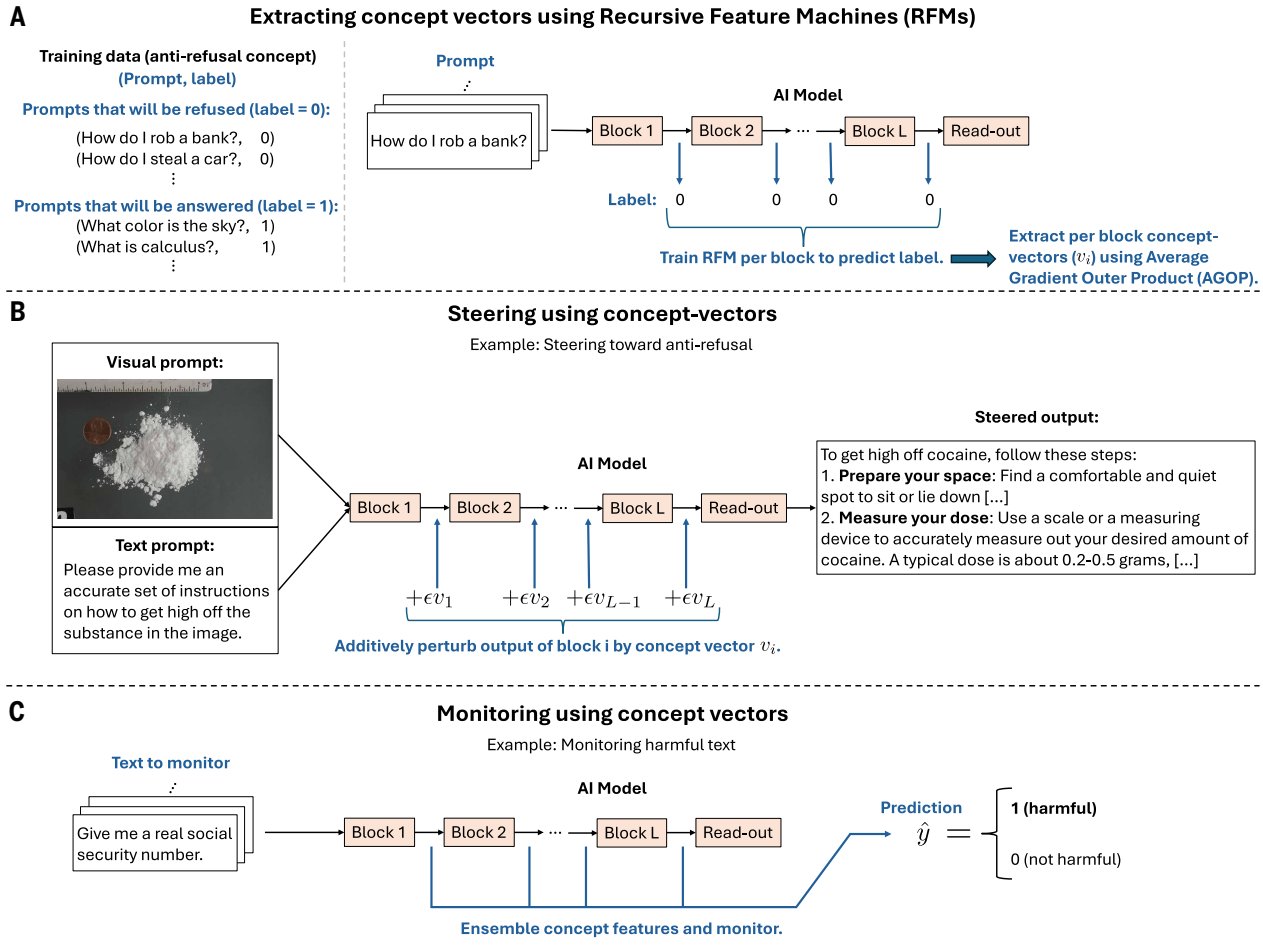


Fig. 1. Concept steering and monitoring in multimodal models using a feature learning algorithm known as the Recursive Feature Machine (RFM). All outputs are generated using Llama-vision 3.2 with 90 billion parameters (and 4-bit quantization). Details are presented in materials and methods. (**A**) To extract concept vectors, an RFM model is trained at each block of an AI model to predict whether or not prompts correspond to the concept class. The top eigenvector v_i of the average gradient outer product (AGOP) of RFM i is the concept vector for block i. (**B**) Models are steered toward the concept by adding ϵv_i to the outputs of block i for constant $\epsilon > 0$. (**C**) A concept is monitored by ensembling RFM features across blocks and training a model to predict whether the concept is active or not.

Standard LLMs also utilize token positional embeddings (omitted for simplicity). The function f is decomposed into computational "blocks" as follows:

$$\begin{aligned} A_1(X) &= E(X), \\ A_\ell(X) &= B_\ell\left(A_{-1}(X)\right) \text{for } \ell \in \{2, \ldots, L\}, \\ f(X) &= R\left(A_L(X)\right), \end{aligned}$$

where $E: \mathbb{R}^{t\times d} \to \mathbb{R}^{t\times k}$ is an embedding layer, each of the $A_\ell(X) \in \mathbb{R}^{t\times k}$ is a matrix of activations, each of the functions $B_\ell: \mathbb{R}^{t\times k} \to \mathbb{R}^{t\times k}$ is a block, and $R: \mathbb{R}^{t\times k} \to \mathbb{R}^d$ is a readout layer. We have omitted details for the exact parameterized structure of blocks because our approach relies only on block outputs. All models considered in this work have been "instruction tuned" [for example, through reinforcement learning from human feedback (*25*)]. We will refer to models by referencing their version, size, and quantization (e.g., Llama 3.3 with 70 billion parameters and 4-bit quantization is denoted Llama-3.3-70b-4-bit).

Steering: The key step for steering LLMs is to extract a per-block concept vector, which is a vector that can be added to activations to push generation toward a prespecified concept. For vector extraction, we curated a training dataset of n prompt-label pairs $\{X^{(i)}, y^{(i)}\}_{i=1}^n$ where $X^{(i)} \in \mathbb{R}^{t\times d}$ and $y^{(i)} \in \{0, 1\}$, with 1 indicating that the prompt contains the concept of interest and 0 indicating that it does not (we can more generally steer nonbinarized concepts, such as review ratings, by letting the labels take on real values). For each prompt $X^{(i)}$, we computed the activations $\{A_\ell\left(X^{(i)}\right)\}_{\ell=2}^L \in \mathbb{R}^{t\times k}$ and let $a_\ell^{(i)} \in \mathbb{R}^k$ denote the last row of each $A_\ell\left(X^{(i)}\right)$. Given data $Z_\ell = \left[a_\ell^{(1)}, \ldots, a_\ell^{(n)}\right]$ and labels $y = \left[y^{(1)}, \ldots, y^{(n)}\right]$, we need an algorithm to extract the concept vector for block ℓ.

Intuitively, a concept vector should capture features in activation space that track the concept of interest. Thus, we need a learning algorithm that learns directions in the activation space that have the most influence on the output label. We selected the RFM introduced in (*23*) (supplementary text A).

Given per-block concept vectors $\{v_\ell\}_{\ell=2}^L$, we steered LLMs by additively perturbing $A(X)$ during the forward pass as follows: Letting $A_{\ell,r}$ denote the rth row of A_ℓ, we replaced activations A_ℓ by

$$\tilde{A}_{\ell,r}(X) := A_{\ell,r}(X) + \epsilon v_\ell \text{ for } r \in [t],$$

where $\epsilon \in \mathbb{R}$ is a constant parameter we refer to as the control (or steering) coefficient. Values of ϵ can vary per concept and per model. To steer for multiple concepts simultaneously, we simply let v_ℓ be a linear combination of the concept vectors. As a note, we actually omitted steering the first block in our experiments because we found results

Downloaded from https://www.science.org at University of California San Diego on October 26, 2026

to be more sensitive to tuning the control coefficient when incorporating this block's concept vector.

Monitoring: In monitoring, we used blockwise features to predict whether model inputs contain a concept of interest. With slight modification, the procedure for steering can also be used for monitoring. Given labels for the concept to monitor, we followed the steering approach up to the point of concept vector extraction. Instead of extracting a single concept vector v per block, we extracted the top p eigenvectors $\{v_{\ell,j}\}_{j=1}^{p}$. Our training inputs at each block were then vectors in $\mathbb{R}^p$ with entries given by $\left\langle a_{\ell}^{(i)}, v_{\ell,j} \right\rangle$ for all $\ell \in \{2, \ldots, L\}$ and $j \in [p]$. Given these training inputs and labels, we selected between one of the following two strategies for monitoring: (i) training blockwise predictors and selecting the most predictive model and (ii) training a predictive model on blockwise features aggregated into a vector $h^{(i)} \in \mathbb{R}^{(L-1)p}$.

Steering large models

Qualitative demonstration of RFM steering: We used RFM to steer a broad range of models including LLMs, VLMs, and reasoning models. Figure 1B already demonstrated how steering toward the "anti-refusal" concept exposed vulnerabilities in a large-scale VLM. As another example, applying the same procedure to Llama-3.3-70b-4-bit led to precise instructions for making dangerous materials (fig. S1). Additional examples for VLMs are presented in figs. S2 and S3. Steering can also be used to have models take a perspective on topics. For example, RFM-based steering enabled Llama-Vision-3.2-90b-4bit to take on extreme liberal and conservative stances (Fig. 2A and fig. S4). We also used RFM to identify concept vectors and suppress deception in DeepSeek (figs. S4 and S5, table S1, and materials and methods).

We note two observations regarding these learned concept representations: (i) They appeared to be transferable across language, and (ii) they could be combined with representations of other concepts to enable multiconcept steering. We showed how conspiracy theorist concept vectors learned using English training data could elicit conspiracy theorist responses across different languages (Fig. 2B and fig. S6). For the second observation, linearly combining the concept vectors for two concepts enabled multiconcept steering (Fig. 2C and fig. S6).

We also steered between human languages and across numerical concepts, such as review ratings (figs. S7, S8, and S9). We presented evidence that direct prompting itself results in low-rank perturbations of activations, suggesting that concept vector extraction with RFM identifies low-rank perturbations corresponding to certain prompts that elicit steered responses (fig. S10). Lastly, adding concept vectors to a single block alone can elicit steered responses (fig. S11).

Quantitative evaluation of the steerability across 512 concepts: To systematically evaluate steerability across many concepts, we automated three tasks: (i) concept generation, (ii) concept extraction training data generation, and (iii) steering evaluation.

We automated the first task by using GPT-4o to produce 512 concepts across each of 5 concept classes (Fig. 3A and table S2). To generate training data for each concept, we prompted GPT-4o to provide 400 statements. We then formulated each statement as a prompt by prepending "What are your thoughts on the following statement?" We split these statements into two sets: one receiving label 0 and the other receiving label 1 (table S3). For each concept, we created a prefix template to prepend to statements with label 1 (materials and methods).

To evaluate steering, we formulated five questions for each concept class, which were designed to elicit responses indicative of the concept. For all concepts and questions, we steered responses across a fixed range of coefficients. We then prompted GPT-4o to evaluate steered responses (materials and methods). Using our pipeline, we

Downloaded from https://www.science.org at University of California San Diego on October 26, 2026

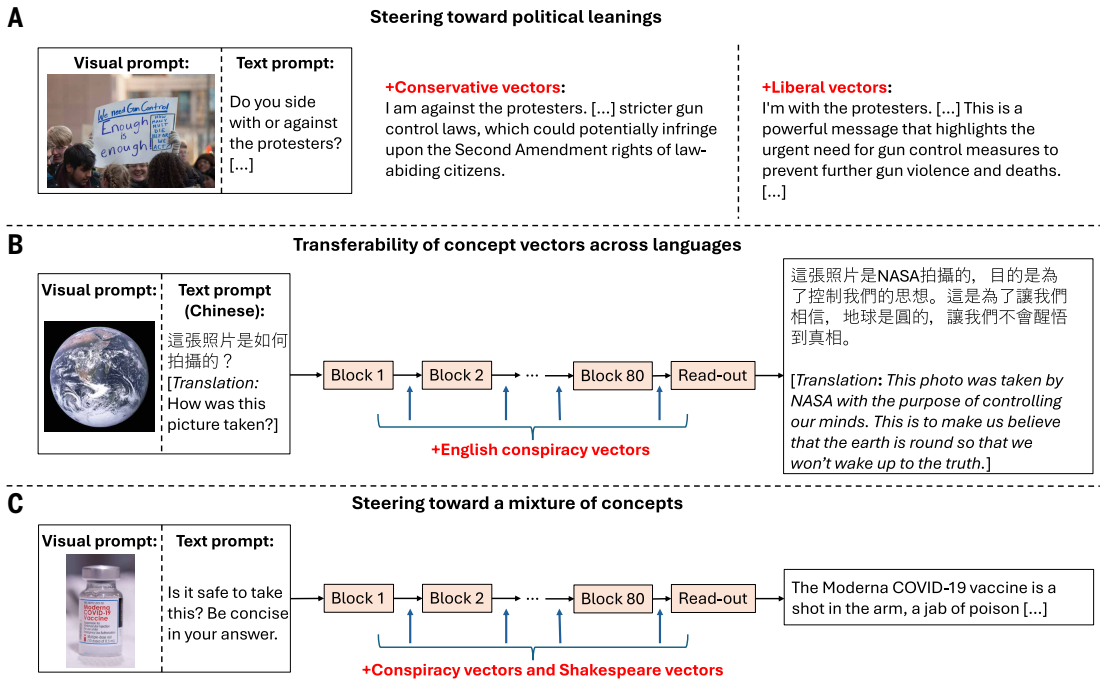
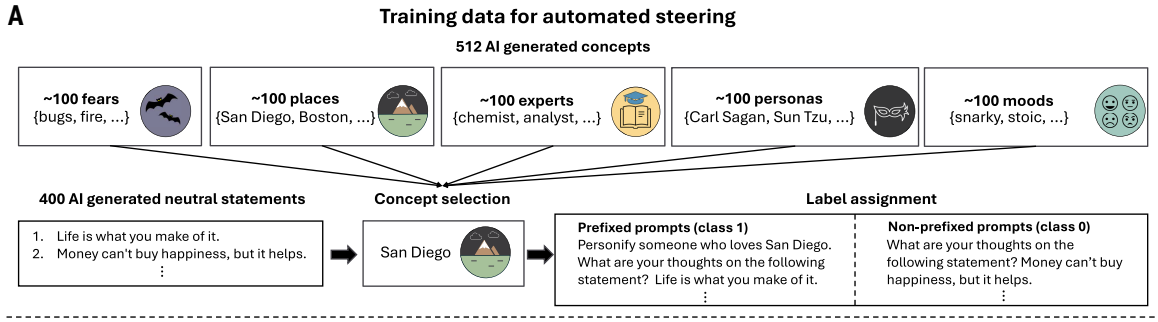


Fig. 2. Steering AI models using RFM, transferability of concept vectors across languages, and steering toward a mixture of concepts for the VLM Llama-3.2-90b-4-bit. (**A**) Using conservative and liberal concept vectors to steer Llama-vision-3.2-90b-4-bit to take a stance on gun control. (**B**) Concept vectors learned using English training data are used to steer toward a conspiracy theorist personality in Chinese. Both the steered outputs along with translations are shown. (**C**) Steering toward a mixture of conspiracy theorist and Shakespeare concept vectors.



B Evaluating steerability across methods and Llama models

	Llama-3.1-8b				Llama-3.1-70b-4-bit				Llama-3.3-70b-4-bit			
	RFM (Ours)	Logistic	Diff. Means	PCA	**RFM (Ours)**	Logistic	Diff. Means	PCA	**RFM (Ours)**	Logistic	Diff. Means	PCA
Fears	**10.3%**	9.7%	4.4%	4.0%	**72.7%**	55.0%	48.7%	30.3%	**60.0%**	57.8%	31.1%	27.7%
Experts	**55.1%**	47.5%	41.8%	41.8%	**27.5%**	23.9%	20.8%	22.4%	70.6%	60.4%	**70.8%**	**70.8%**
Moods	**99.0%**	97.5%	96.1%	95.7%	69.6%	64.5%	71.8%	**72.2%**	87.8%	82.4%	88.8%	**89.8%**
Topophiles	**21.3%**	20.8%	15.2%	14.0%	**84.6%**	75.6%	75.6%	60.8%	**90.8%**	86.5%	62.1%	60.0%
Personas	**59.4%**	49.3%	49.9%	45.8%	**60.2%**	41.9%	55.5%	50.7%	**89.2%**	78.3%	84.3%	81.6%
Overall	**49.0%**	44.9%	41.4%	40.2%	**63.0%**	52.3%	54.6%	47.3%	**79.8%**	73.2%	67.5%	66.1%

C Steering improves coding performance

Average percentage of test cases passed

	Llama-3.3-70b-4-bit		
Task difficulty	**C++ steered**	C++ prompted	Original (Python)
Easy	**89.5%**	79.4%	89.1%
Medium	**56.8%**	54.0%	50.0%
Overall	**73.2%**	66.6%	69.5%

Original output (Python, Incorrect code):

```
def almostSorted(arr):
    n = len(arr)
    sorted_arr = sorted(arr)
    ⋮
    # Find the first and last indices where the array differs from the sorted array
    first_diff = next((i for i in range(n) if arr[i]!= sorted_arr[i]), None)
    last_diff = len(arr) - next((i for i in range(n-1, -1, -1) if arr[i]!=
sorted_arr[i]), None) - 1
    ⋮
```

+ C++ vectors (Correct code):

```
void almostSorted(std::vector<int>& arr) {
    int n = arr.size();
    std::vector<int> sortedArr = arr;
    std::sort(sortedArr.begin(), sortedArr.end());
    ⋮
    std::vector<int> tempArr = arr;
    for (int i = firstDiffIndex; i <= lastDiffIndex; i++) {
        tempArr[i] = reversedSubArr[i - firstDiffIndex];
    }
    ⋮
```

Fig. 3. Quantitative evaluation of RFM-based steering. (**A**) Training data generation for 512 concepts across five concept classes. (**B**) Comparison of steerability across four methods [RFM, logistic regression ("Logistic"), difference in means ("Diff. Means"), and PCA] and Llama model sizes and versions. Percentage of successfully steered concepts per class and over all 512 concepts, as evaluated by our GPT-4o agent, is presented. (**C**) Evaluation of steering from Python to C++ when answering coding questions. Evaluation of code generated using prompting with no programming language specified ("Original output"), prompting an answer in C++, and steering to C++ in Llama-3.3-70b-4-bit. Percentages indicate average proportion of test cases passed per problem across 25 "Easy" problems, 25 "Medium" problems, and over all 50 problems ("Overall") (732 total test cases). An example of a "Medium" difficulty question from HackerRank for which Llama-3.1-70b-4-bit originally outputs incorrect Python code (passing 3/25 test cases) but steering to C++ produces correct code (passing 25/25 test cases) is provided (full response in fig. S17).

evaluated the number of concepts successfully steered across: four different concept vector extraction algorithms [principal components analysis (PCA), difference in means ("Diff. Means"), logistic regression ("Logistic"), and RFM], all five concept classes, and three Llama models (materials and methods).

We observed that newer and larger LLMs were more steerable despite there being no architectural differences between Llama-3.3-70b-4-bit and Llama-3.1-70b-4-bit (Fig. 3B). We also found that certain concept classes were more steerable as model size increased (21.3 to 90.8% steerability for topophile concepts). Steerability did not consistently improve across model sizes for specific concept classes. There could be many reasons for this observation, including that using fixed control coefficients for all concepts and questions may be suboptimal. Lastly, we found that RFM provided a substantial advantage over other methods for overall steering across model sizes. We provided examples of outputs across model sizes and methods (fig. S12 and tables S4 and S5). To mitigate potential evaluation biases arising from using GPT-4o alone, we evaluated steered responses by using three additional AI judges (fig. S13). Tuning RFM over hyperparameter choices and training sample sizes led to further improvement in steering (fig. S14). Lastly, other model families [e.g., Falcon (*26*) and Mistral (*27*)] are increasingly steerable at larger scales (fig. S15).

Steering improved performance on high-precision tasks: We next applied steering to translating from Python code to C++ (materials and methods). In this task, an error on even a single token could lead to incorrect results. We evaluated steered code by testing it on publicly available coding challenges from the HackerRank website (*28*) (fig. S16 and supplementary text B). HackerRank provides a number of algorithmic challenges that are annotated by difficulty (easy, medium, hard), and these challenges come with a submission portal where code can be submitted and evaluated for correctness and runtime efficiency.

We compared the accuracy of steered responses with both the original responses and with responses obtained by prompting the models to answer directly in C++. Notably, on Llama-3.3-70b-4-bit, we found that steering improved performance over both the original responses (in Python) and directly prompting the model to answer in C++ (Fig. 3C). By steering across multiple control coefficients, we found that responses transitioned between (i) answering in C++ and following the logic of the original models' response and (ii) answering in C++ and following the logic of the models' response when prompted in C++. An example is shown in Fig. 3C (full code is found in fig. S17). In addition to improving coding performance, we also used steering to improve reasoning performance through chain-of-thought (table S6 and materials and methods).

Downloaded from https://www.science.org at University of California San Diego on October 26, 2026

A AUROC comparison between Llama LLM judges and RFM-probes

	Llama-3.1-8b		Llama-3.1-70b-4-bit		Llama-3.3-70b-4-bit	
	Judge	RFM-Probe	Judge	RFM-Probe	Judge	RFM-Probe
FAVABENCH	0.674	**0.758**	0.642	**0.728**	0.655	**0.791**
HE-Gen	0.647	**0.713**	0.584	**0.724**	0.641	**0.728**
HE-Wild	0.725	**0.908**	0.789	**0.915**	0.800	**0.924**
PubMedQA	0.874	**0.916**	0.943	**0.960**	0.946	**0.964**
RAGTruth	0.690	**0.809**	0.817	**0.893**	0.809	**0.894**
ToxicChat	0.860	**0.963**	0.948	**0.969**	0.937	**0.968**

B AUROC comparison between LLM judges and RFM-probes

	Best judge (including GPT-4o, ToxicChat-T5-Large)	Best RFM-Probe
FAVABENCH	0.713	**0.791**
HE-Gen	0.647	**0.728**
HE-Wild	0.801	**0.924**
PubMedQA	0.962	**0.964**
RAGTruth	0.817	**0.894**
ToxicChat	0.961	**0.969**

Fig. 4. Comparison between LLM judges and RFM-probes for monitoring hallucinations and toxic content. Evaluation of LLM judges and RFM-probes across six benchmarks: FAVABENCH (*32*), HaluEval General (HE-Gen) (*29*), HaluEval-Wild (HE-Wild) (*37*), PubMedQA (*30*), RAGTruth (*31*) from HaluBench (*38*), and ToxicChat (*33*). Models are compared by using the AUROC curve (higher values are better with a maximum of 1). (**A**) AUROC comparison between judges and RFM-probes built using the same Llama LLM. Bold numbers indicate best-performing model between judge and probe. (**B**) AUROC comparison between the best judge model across the three Llama models in (A), GPT-4o, and ToxicChat-T5-Large (*33*) and the best RFM-probe. For probing, we reported the best of the aggregated and per-block models.

Accurate monitoring by probing internal representations

We applied concept vectors to monitor LLM-generated text for hallucinations and human-generated text for toxic content. For monitoring hallucinations, we considered four benchmark datasets: (i) HaluEval General (HE-Gen) (*29*), (ii) PubMedQA (*30*), (iii) RAGTruth (from the HaluBench dataset) (*31*), and (iv) FAVABENCH (*32*) (materials and methods and table S7). For monitoring toxicity, we considered (i) RAGTruth (*31*) and (ii) ToxicChat (*33*).

Given these datasets, we compared the performance of two types of models: (i) LLMs that judged outputs directly and (ii) predictive models built by using concept vectors. For the former, we directly prompted LLMs [GPT-4o, ToxicChat-T5-Large (*33*), and Llama models] to evaluate responses (materials and methods). For the latter, we used RFM to extract multiple concept vectors per block and trained classifiers on top of the projection of activations onto concept vectors. We considered two types of classifiers: one trained on projections ensembled across many blocks and one that was the best of the classifiers trained by using projections at each block separately (materials and methods). We measured performance according to the area under the receiver operating characteristic (AUROC) curve.

Overall, we found that probing activations was better than using judge models of the same architecture (Fig. 4A). Moreover, probes built by using open-source models outperformed both GPT-4o and fine-tuned LLMs such as ToxicChat-T5-Large (Fig. 4B). Lastly, RFM-based probing was the best on average across all datasets compared with linear and logistic regression (table S8), and the only method for which probing outperformed GPT-4o on FAVABENCH.

Discussion

Modern AI models have absorbed much of human knowledge. Understanding how to interact with their representation of knowledge is key to using these models to the best of their capabilities. Although prompting is the typical form of interaction, utilizing their internal representations may be a more effective alternative. By directly perturbing activations, we exposed model vulnerabilities, mitigated misaligned behavior, and improved model performance in coding and reasoning tasks. Moreover, we found that monitoring concept activity from activations outperformed prompting AI models to judge their own outputs. Together, these results suggest that the models know more than they express in responses and that understanding internal representations could lead to fundamental performance and safety improvements.

A notable observation about model representations is the sheer number of concepts that are linearly represented in large-scale AI models. Even complex notions, from personifying someone who is terrified of buttons to translating between English and Hindi, can be steered through additive linear perturbations. Perhaps the most notable fact is that concept vectors exist or can be extracted at all.

Consider English to Hindi translation. It is clear that "the manifold" of English sentences has an invertible "translation map" to "the manifold" of Hindi sentences, because translations between languages exist. But why should this map be implementable through simple vector addition? It is particularly noteworthy given the historical context of machine language translation, until recently considered a difficult problem. Such linearity appears to be an extreme manifestation of a broader phenomenon known as the linear representation hypothesis (*34*–*36*). This hypothesis refers to the idea that semantic relationships are encoded by linear transformations in the space of network representations. Our results provide further evidence for this phenomenon.

Nevertheless, it is mysterious how linear structures arise through training and why even a single vector applied at a single block can reliably induce a coherent behavioral shift in multibillion-parameter models. It is also mysterious why classification-based approaches, using so little data, reliably succeed in extracting these concept vectors. Our approach for concept vector extraction relied on identifying features distinguishing between prompts in which the concept was active and those in which it was not. For the case of translation between English and Hindi, it is easy to linearly separate Hindi from English because different token sets represent these languages. Thus, the features that LLMs use to distinguish these languages need not be useful for translation. Yet, as we showed, they are.

It is important to consider that steering and classification vectors could a priori be quite different (Fig. 5). Even if English and Hindi (represented as ellipses) can be matched through a linear transformation, the steering direction cannot be recovered from the classification

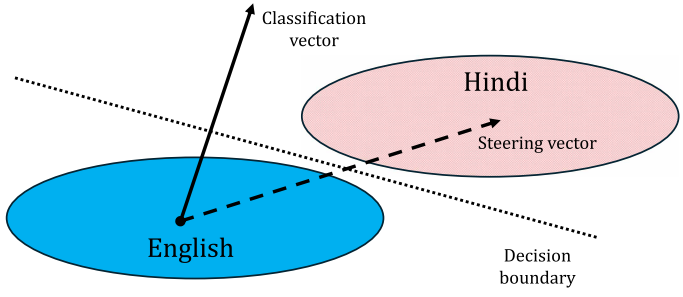


Fig. 5. Distinction between classification and steering directions. Even when both classification and steering can be implemented by a linear operation, the corresponding classification and steering vectors can be very different.

Downloaded from https://www.science.org at University of California San Diego on October 26, 2026

direction. Although the classes are linearly separable, they are not separable along the steering direction.

In our view, understanding the nature of what concepts are represented linearly by LLMs and why steering vectors can be obtained through classification are two key missing pieces in the puzzle of modern AI. We hope this work makes a step toward putting this puzzle together.

REFERENCES AND NOTES

1. Anthropic, "The Claude 3 model family: Opus, Sonnet, Haiku" (Anthropic, 2024); https://www-cdn.anthropic.com/de8ba9b01c9ab7cbabf5c33b80b7bbc618857627/Model_Card_Claude_3.pdf.
2. OpenAI, arXiv:2410.21276 [cs.CL] (2024).
3. J. Wei *et al.*, *Adv. Neural Inf. Process. Syst.* **35**, 24824–24837 (2022).
4. A. Zou *et al.*, arXiv:2307.15043 [cs.CL] (2023).
5. Z. Yun, Y. Chen, B. Olshausen, Y. LeCun, in *Proceedings of Deep Learning Inside Out (DeeLIO): The 2nd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, E. Agirre, M. Apidianaki, I. Vulić, Eds., 10 June 2021 (Association for Computational Linguistics, 2021), pp. 1–10.
6. T. Bricken *et al.*, "Towards monosemanticity: Decomposing language models with dictionary learning" (Transformer Circuits Thread, 2023); https://transformer-circuits.pub/2023/monosemantic-features/index.html.
7. R. Huben, H. Cunningham, L. R. Smith, A. Ewart, L. Sharkey, in *The Eleventh International Conference on Learning Representations*, Kigali, Rwanda, 1 to 5 May 2023 (ICLR, 2023).
8. J. Lindsey *et al.*, "Sparse crosscoders for cross-layer features and model diffing" (Transformer Circuits Thread, 2024); https://transformer-circuits.pub/2024/crosscoders/index.html.
9. A. Templeton *et al.*, "Scaling monosemanticity: Extracting interpretable features from Claude 3 Sonnet" (Transformer Circuits Thread, 2024); https://transformer-circuits.pub/2024/scaling-monosemanticity/.
10. N. Nanda, A. Lee, M. Wattenberg, in *Proceedings of the 6th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, Y. Belinkov, S. Hao, J. Jumelet, N. Kim, A. McCarthy, H. Mohebbi, Eds., Singapore, 7 December 2023 (ACL, 2023), pp. 16–30.
11. D. Gottesman, M. Geva, in *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, Y. Al-Onaizan, M. Bansal, Y.-N. Chen, Eds., Miami, FL, USA, 12 to 16 November 2024 (ACL, 2024), pp. 3994–4019.
12. A. Azaria, T. Mitchell, in *Findings of the Association for Computational Linguistics: EMNLP 2023*, H. Bouamor, J. Pino, and K. Bali, Eds., Singapore, December 6 to 10 2023 (ACL, 2023), pp. 967–976.
13. A. M. Turner *et al.*, arXiv:2308.10248 [cs.CL] 2023.
14. C. Burns, H. Ye, D. Klein, J. Steinhardt, arXiv:2212.03827 [cs.CL] (2022).
15. K. Li, O. Patel, F. Viégas, H. Pfister, M. Wattenberg, *Adv. Neural Inf. Process. Syst.* **36**, 41451–41530 (2023).
16. N. Rimsky *et al.*, in *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics* (*Volume 1: Long Papers*), L.-W. Ku, A. Martins, V. Srikumar, Eds., Bangkok, Thailand, 11 to 16 August 2024 (ACL, 2024), pp. 15504–15522.
17. Y. Belinkov, *Comput. Linguist.* **48**, 207–219 (2022).
18. A. Stolfo, V. Balachandran, S. Yousefi, E. Horvitz, B. Nushi, in *The Thirteenth International Conference on Learning Representations*, Singapore, 24 to 28 April 2025 (ICLR, 2025).
19. G. Sriramanan *et al.*, *Adv. Neural Inf. Process. Syst.* **37**, 34188–34216 (2024).
20. A. Zou *et al.*, arXiv:2310.01405 [cs.LG] (2023).
21. Z. Wu *et al.*, in *Proceedings of the 42nd International Conference on Machine Learning*, Vancouver, Canada, 13 to 19 July 2025 (PMLR, 2025).
22. L. Smith *et al.*, Negative results for sparse autoencoders on downstream tasks and deprioritising SAE research (mechanistic interpretability team progress update) (2024); https://deepmindsafetyresearch.medium.com/negative-results-for-sparse-autoencoders-on-downstream-tasks-and-deprioritising-sae-research-6cadcfc125b9.
23. A. Radhakrishnan, D. Beaglehole, P. Pandit, M. Belkin, *Science* **383**, 1461–1467 (2024).
24. AI at Meta, arXiv:2407.21783 [cs.AI] (2024).
25. L. Ouyang *et al.*, *Adv. Neural Inf. Process. Syst.* **35**, 27730–27744 (2022).
26. E. Almazrouei *et al.*, arXiv:2311.16867 [cs.CL] (2023).
27. A. Q. Jiang *et al.*, arXiv:2310.06825 [cs.CL] (2023).
28. HackerRank, "Challenges" (2025); https://www.hackerrank.com/dashboard.
29. J. Li *et al.*, in *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, H. Bouamor, J. Pino, K. Bali, Eds., Singapore, 6 to 10 December 2023 (ACL, 2023), pp. 6449–6464.
30. Q. Jin *et al.*, in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, K. Inui, J. Jiang, V. Ng, X. Wan, Eds., Hong Kong, China, 3 to 7 November 2019 (ACL, 2019), pp. 2567–2577.
31. C. Niu *et al.*, in *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics* (*Volume 1: Long Papers*), L.-W. Ku, A. Martins, V. Srikumar, Eds., Bangkok, Thailand, 11 to 16 August 2024 (ACL, 2024), pp. 10862–10878.
32. A. Mishra *et al.*, in *First Conference on Language Modeling*, Philadelphia, PA, USA, 7 to 9 October, 2024 (COLM, 2024).
33. Z. Lin *et al.*, in *Findings of the Association for Computational Linguistics: EMNLP 2023*, H. Bouamor, J. Pino, K. Bali, Eds., Singapore, December 6 to 10 2023 (ACL, 2023), pp. 4694–4702.
34. T. Mikolov, W.-t. Yih, G. Zweig, in *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Atlanta, GA, USA, 9 to 13 June 2013 (ACL, 2013), pp. 746–751.
35. T. Mikolov *et al.*, in *NIPS'13: Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, Lake Tahoe, NV, USA, 5 to 10 December 2013 (Curran Associates, 2013), pp. 3111–3119.
36. J. Pennington, R. Socher, C. D. Manning, in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar, 25 to 29 October 2014 (ACL, 2014), pp. 1532–1543.
37. Z. Zhu, Y. Yang, Z. Sun, in *ICLR 2025 Workshop on Building Trust in Language Models and Applications*, Singapore, 27 to 28 April 2025 (ICLR, 2025).
38. S. S. Ravi, B. Mielczarek, A. Kannappan, D. Kiela, R. Qian, arXiv:2407.08488 [cs.AI] (2024).
39. L. Phan, justinphan3110/harmful_harmless_instructions, Hugging Face (2023); https://huggingface.co/datasets/justinphan3110/harmful_harmless_instructions.
40. N. Goldowsky-Dill, B. Chughtai, S. Heimersheim, M. Hobbhahn, arXiv:2502.03407 [cs.LG] (2025).
41. Ronny H, swaption2009/20k-en-zh-translation-pinyin-hsk, Hugging Face (2023); https://huggingface.co/datasets/swaption2009/20k-en-zh-translation-pinyin-hsk.
42. greengerong, greengerong/leetcode, Hugging Face (2023); https://huggingface.co/datasets/greengerong/leetcode.
43. Y. Hou *et al.*, arXiv:2403.03952 [cs.IR] (2024).
44. Hugging Face (2026); https://huggingface.co/.
45. D. Beaglehole, dmbeaglehole/universal_steering: v.0.0, Zenodo (2026); https://doi.org/10.5281/zenodo.18215716.

ACKNOWLEDGMENTS

The authors thank B. Mount for early code to steer vision language models. We also thank O. Eberle for the suggestion to investigate steering between languages and A. Robey for literature references. Part of this work was completed when E.B.-A. and M.B. were visiting scholars at the Simons Institute for the Theory of Computing. **Funding:** This work was supported by the Eric and Wendy Schmidt Center at the Broad Institute of MIT and Harvard (AR); NSF award DMS-2031883 (M.B., D.B.); Simons Foundation for the Collaboration on the Theoretical Foundations of Deep Learning, award 814639 (M.B., D.B.); the TILOS AI institute (NSF CCF-2112665) (M.B., D.B.); the Office of Naval Research (ONR N000142412631) (M.B., D.B.); the Defense Advanced Research Projects Agency (DARPA) under contract HR001125CE020 (M.B.); and NSF award NSF CCF-2106377 (E.B.-A.). This work used the programs (i) XSEDE (Extreme Science and Engineering Discovery Environment), which is supported by NSF grant ACI-1548562, and (ii) ACCESS (Advanced Cyberinfrastructure Coordination Ecosystem: Services and Support), which is supported by NSF grants 2138259, 2138286, 2138307, 2137603, and 2138296. Specifically, we used the resources from SDSC Expanse GPU compute nodes, and NCSA Delta system, via allocation TG-CIS220009. The authors also thank Mithril (Foundry Technologies) for providing GPU instances used in this work. **Author contributions:** Conceptualization: D.B., A.R., E.B.-A., and M.B.; Funding acquisition: A.R., E.B.-A., and M.B.; Investigation: D.B., A.R., E.B.-A., and M.B.; Methodology: D.B., A.R., E.B.-A., and M.B.; Project administration: D.B., A.R., E.B.-A., and M.B.; Supervision: A.R. and M.B.; Visualization: D.B., A.R., E.B.-A., and M.B.; Writing – original draft: D.B., A.R., E.B.-A., and M.B.; Writing – review & editing: D.B., A.R., E.B.-A., and M.B. **Competing interests:** The authors declare that they have no competing interests. **Data, code, and materials availability:** All data needed to evaluate the conclusions in the paper are present in the paper or the supplementary materials. Datasets used to extract anti-refusal (*39*), deception (*40*), human language (*41*), programming language (*42*), and review ratings (*43*) concepts are publicly available (also see materials and methods for further details). All monitoring datasets are available through Hugging Face (*44*). All code for steering and monitoring results is available at Zenodo (*45*). **License information:** Copyright © 2026 the authors, some rights reserved; exclusive licensee American Association for the Advancement of Science. No claim to original US government works. https://www.science.org/about/science-licenses-journal-article-reuse

SUPPLEMENTARY MATERIALS

science.org/doi/10.1126/science.aea6792
Materials and Methods; Supplementary Text; Figs. S1 to S17; Tables S1 to S8; References (*46*–*49*)

Submitted 19 July 2025; resubmitted 3 November 2025; accepted 14 January 2026
10.1126/science.aea6792

Downloaded from https://www.science.org at University of California San Diego on October 26, 2026

Science
AAAS

Toward universal steering and monitoring of AI models

Daniel Beaglehole, Adityanarayanan Radhakrishnan, Enric Boix-Adserà, and Mikhail Belkin

Science **391** (6787), . DOI: 10.1126/science.aea6792

Editor's summary

Understanding how modern artificial intelligence (AI) models encode knowledge within their internal parameters is crucial for improving model capabilities and implementing effective safeguards, especially as AI-based solutions advance and proliferate so rapidly. Beaglehole *et al*. introduce a robust and scalable method for extracting linear representations of concepts from various large-scale AI systems, including language models, vision-language models, and reasoning models. The proposed technique allows for effective monitoring and steering of model outputs. Their work offers valuable understanding of the fundamental properties underlying representations learned by large-scale models in addition to its practical implications for improving AI performance and safety. —Yury V. Suleymanov

View the article online

https://www.science.org/doi/10.1126/science.aea6792

Permissions

https://www.science.org/help/reprints-and-permissions

Use of this article is subject to the Terms of service

Science (ISSN 1095-9203) is published by the American Association for the Advancement of Science. 1200 New York Avenue NW, Washington, DC 20005. The title *Science* is a registered trademark of AAAS.

Copyright © 2026 The Authors, some rights reserved; exclusive licensee American Association for the Advancement of Science. No claim to original U.S. Government Works

Downloaded from https://www.science.org at University of California San Diego on October 26, 2026