InverseScope: Scalable Activation Inversion for Interpreting LLMs

Yifan Luo [1] **Zhennan Zhou** [2] **Bin Dong** [3 4]

Abstract

Understanding the internal representations of large language models (LLMs) is a central challenge in interpretability research. Existing feature interpretability methods often rely on strong structural assumptions—such as linearity or sparsity—that may not hold in practice. In this work, we introduce InverseScope, an assumption-light and scalable framework for interpreting neural activations via input inversion. Given a target activation, InverseScope characterizes its encoded information by generating natural-language inputs that produce nearby activations, grounding abstract internal states in concrete language. To overcome the prohibitive cost of sampling in high-dimensional activation spaces, we propose a novel control-layer conditioning architecture that substantially improves sample efficiency compared to prior token-prepending approaches. We demonstrate that InverseScope reveals rich geometric structure in LLM representation spaces, including sentence-level linear analogies. The framework scales to state-of-the-art open-source models of up to 14B parameters and generalizes to out-of-distribution inputs, enabling systematic analysis of activation neighborhoods.

1. Introduction

Recent advances in mechanistic interpretability aim to reverse-engineer neural networks' computations into human-understandable processes (Bereska and Gavves, 2024; Sharkey et al., 2025). A central task in this field is feature interpretability, which seeks to understand what information is encoded in a network's activations and how it is represented. This understanding is crucial for analyzing how information propagates and is processed across layers. Numerous methods have been proposed for feature interpretability, including linear probing (Alain and Bengio, 2016; Park et al., 2023), sparse dictionary learning (Cunningham et al., 2023; Gao et al., 2024), and other approaches (Bau et al., 2017; Xu et al., 2024).

Despite their successes, these methods share a fundamental limitation: they rely on strong assumptions about the structure of neural representations. Specifically, linear probing assumes a linear relationship between activations and specific concepts in the inputs, and sparse dictionary learning presupposes that activations can be decomposed into a sparse sum of linear directions. The validity of these assumptions remains an open and actively debated question, particularly in the context of LLMs (Levy and Geva, 2024; Engels et al.; Smith, 2024). Designing experiments that rigorously test these hypotheses is itself a challenging problem, making it difficult to assess the reliability of interpretations derived from such approaches.

To address these limitations, a promising alternative is to interpret activations by inverting them back to the input space, where representations are naturally grounded in human-understandable language. This inversion-based paradigm has a long history in computer vision (Erhan et al., 2009; Mahendran and Vedaldi, 2014) and has recently gained traction in the NLP community. For instance, Rep2Text (Zhao et al., 2024) demonstrates that the representation at a single last-token position encodes sufficient information to autoregressively reconstruct the full input text via a trained adapter, while InversionView (Huang et al., 2024) shows that the information encoded in a hidden state can be characterized by collecting a distribution of inputs that yield similar activations. Together, these works highlight the potential of inversion-based methods to provide an assumption-light and intuitive framework for feature interpretability, enabling a direct mapping between abstract internal states and concrete natural language.

However, existing inversion methods each face their own limitations. Rep2Text (Zhao et al., 2024) targets exact reconstruction of the original input, and consequently lacks the neighborhood perspective central to interpretability. Inver-

[1]School of Mathematical Science, Peking University, Beijing, China [2]School of Science, Westlake University, Hangzhou, China [3]Beijing International Center for Mathematical Research and the New Cornerstone Science Laboratory, Peking University, Beijing, China [4]Center for Machine Learning Research, Peking University, Beijing, China. Correspondence to: Yifan Luo <luoyf@pku.edu.cn>, Bin Dong <dongbin@math.pku.edu.cn>.

Mechanistic Interpretability Workshop at the 43rd International Conference on Machine Learning, Seoul, South Korea, 2026. Copyright 2026 by the author(s).

sionView (Huang et al., 2024) addresses this gap by framing inversion as neighborhood characterization, but its applicability has been confined to task-specific templates and small-scale models, leaving the general-distribution setting unexplored. Critically, all three methods share the same architectural family: the target activation is projected into the token embedding space and placed as a soft token alongside the other input embeddings. Our ablation experiments demonstrate that this input-level conditioning is architecturally suboptimal: conditioning at intermediate layers via dedicated control layers yields substantially better sample efficiency (Section 4.2).

Our main contributions are as follows:

- **Control-layer conditioning architecture.** We introduce a conditional generation architecture in which control layers inject the target activation z into the backbone at every Transformer layer, analogously to cross-attention. This design substantially outperforms the token-prepending approach used in prior work in both sample efficiency and training stability, enabling reliable inversion at scale.
- **Scaling to general distributions and large models.** We scale inversion-based interpretability to general-domain text (Wikipedia and Project Gutenberg) and to LLMs of up to 14B parameters, removing the dependence on task-specific templates that has restricted prior methods. We further characterize how inversion quality varies with target model size and generator capacity.
- **Empirical analysis of representation geometry and safety interventions.** We demonstrate that InverseScope recovers sentence-level linear analogy structure in LLM activations and provides interpretable neighborhoods for activations produced by instruction-following prompts. We apply the framework to analyze refusal-direction ablation, showing that it renders the representational effect of the intervention directly legible in natural language.

Collectively, these contributions advance inversion-based interpretability from small-scale, task-specific settings to general-purpose analysis of state-of-the-art LLMs, and demonstrate the utility of the framework for both geometric probing and safety-relevant applications.

2. Related works

Activation interpretation methods. A variety of methods have been developed to interpret neural network features. Classical approaches such as linear probing train simple classifiers on activations to identify linearly encoded features (Alain and Bengio, 2016; Park et al., 2023). More recent work includes sparse dictionary learning, which decompose activations into sparse and interpretable components to disentangle feature representations (Cunningham et al., 2023; Gao et al., 2024). While effective, these approaches typically rely on strong structural assumptions about the underlying representations. For example, that features are linearly separable, sparse, or decomposable into a fixed dictionary. Such assumptions may not hold in general, potentially limiting their ability to capture more complex or distributed representations in large language models.

Natural language interpretability. Several recent works have explored assigning human-interpretable labels—such as natural language descriptions—to the internal activations of LLMs. Early approaches such as the logit lens interpret intermediate activations by projecting them directly into the vocabulary space and reading out the most likely tokens, effectively providing single-word descriptions of representations (Geva et al., 2022). Building on this idea, training-free methods like SelfIE (Chen et al., 2024) and PatchScope (Ghandeharioun et al., 2024) use a pretrained LLM to read out the information encoded in residual stream activations. Complementing these approaches, LatentQA (Pan et al., 2024) trains a decoder model to answer natural language questions about activations. More recent work further generalizes this paradigm by treating activations as a new input modality: Activation Oracles are trained to answer arbitrary queries about activations, enabling more flexible and general-purpose natural language interpretation (Karvonen et al., 2025). Despite their flexibility, these methods face a fundamental challenge: the natural language explanations they produce are difficult to verify. In particular, they may suffer from hallucination, where the generated descriptions are fluent and plausible but do not faithfully reflect the underlying representations, making it challenging to assess their reliability.

Inversion-based interpretability. A complementary line of research focuses on interpreting model activations by identifying the inputs that give rise to them, a strategy with a long history in computer vision via activation maximization and representation inversion (Erhan et al., 2009; Nguyen et al., 2016; Mahendran and Vedaldi, 2014). Recent work has extended this paradigm to language models along two distinct directions. Rep2Text (Zhao et al., 2024) trains a learned adapter to autoregressively decode the full input text from the single last-token representation of an LLM, demonstrating that substantial lexical and semantic information is recoverable from this compressed summary vector. InversionView (Huang et al., 2024) takes a different approach: rather than reconstructing the exact original input, it samples a distribution of inputs whose activations are close to the target, thereby characterizing what information is encoded in a given hidden state. Concurrently, a comple-

mentary line of work proves that decoder-only Transformer LLMs are almost surely injective: different prompts produce different hidden states at any layer (Nikolaou et al., 2025). They operationalize this result via SipIt, a token-by-token search algorithm that provably reconstructs the exact original prompt from the full hidden-state matrix (all token positions) at any layer. InverseScope differs from all three in both interpretive goal and architecture. SipIt and Rep2Text target exact reconstruction of the original input; InversionView characterizes the activation neighborhood but is confined to task-specific settings. Architecturally, InversionView, Activation Oracles, and Rep2Text all belong to the same family: the target activation is linearly projected into the token embedding space and placed as a soft token alongside the model's input embeddings. InverseScope instead injects the conditioning signal at every Transformer layer via dedicated control layers, generalizes to arbitrary inference positions and general-domain data, and scales to LLMs of up to 14B parameters.

Conditional and controllable generation. Conditional generation, which generates outputs based on a constrained intermediate state, is a foundational technique in deep learning. Initially established with Conditional VAEs (Sohn et al., 2015) and Conditional GANs (Mirza and Osindero, 2014), this paradigm has evolved into methods where the conditioning signal specifies desired structure in the output. Notable examples include modern text-to-image synthesis systems (Ramesh et al., 2022) and structured guidance methods such as ControlNet (Zhang et al., 2023), which condition large generative models on auxiliary inputs. Similarly, controllable text generation (Keskar et al., 2019) steers language model outputs using explicit attributes. These developments highlight the generality of treating intermediate representations as controllable interfaces for generation.

3. InverseScope

In this section, we introduce our method: its intuition, network architecture, training dataset, and training procedure.

3.1. Intuition

Inversion-based interpretability builds on a core intuition: similar activations tend to encode semantically similar information (Bengio et al., 2013). In this paper, we use the term "activation" to specifically denote the output of an LLM layer at a specific inference position, $z \in \mathbb{R}^d$. If two distinct inputs produce nearly identical activations, then the differences between those inputs are unlikely to be encoded in that representation. Conversely, if nearby activations consistently correspond to inputs sharing a particular pattern, this indicates that the pattern is encoded within that region of latent space. This follows from the continuity of

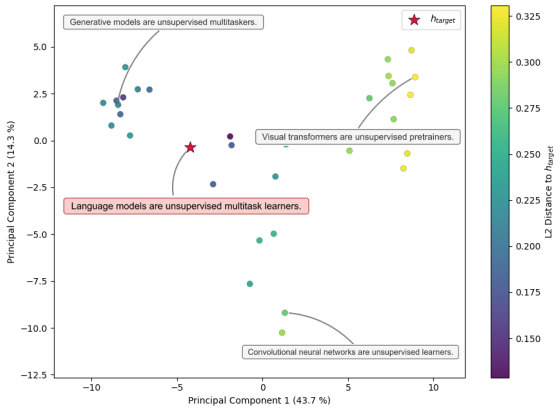


Figure 1. An example of samples in activation space and their corresponding inputs.

neural networks: since downstream layers apply continuous transformations, activations that are close in space are functionally equivalent from the network's perspective.

Building on this intuition, our method investigates the information encoded in a target activation z by inverting the activation geometry to identify inputs whose activations are close to z. By analyzing these inputs, we can formulate hypotheses about which concepts are encoded in the target activation.

For example, as illustrated in Figure 1, consider an input–activation pair (x, z), where x is "Language models are multitask learners." and z is the activation produced at the final "." token. Using our inversion method, we generate prompts whose activations closely match z, such as "Generative models are unsupervised multitaskers." or "Convolutional neural networks are unsupervised learners.," while inputs unrelated to the topic (e.g., "John hates apples.") produce activations far from z. By examining this collection of semantically similar generated inputs, we can hypothesize that z encodes features related to neural network architectures, or more generally, that the subject of the sentence is a type of machine learning model. We can then formalize such hypotheses and use our quantitative framework to rigorously test them across the latent space.

The key challenge in implementing this approach lies in efficiently obtaining inputs whose activations land near the target z. In the high-dimensional activation spaces of modern LLMs, where dimensions can number in the thousands, the probability that a randomly sampled input produces an activation close to z decays exponentially with dimensionality. This makes naive rejection sampling prohibitively inefficient. To address this, we train a conditional generator as a learned prior, which directly produces candidate inputs likely to yield activations near z, making the sampling process tractable.

3.2. Architecture

In this section we describe the network architecture of **InverseScope**. Our objective is to efficiently sample from the conditional distribution $P(x; z)$ defined in the previous section. To achieve this, InverseScope is designed as a conditional generator trained to approximate this target distribution.

We adopt the decoder-only Transformer paradigm, which has proven highly effective for modeling natural language distributions via next-token prediction objectives (Brown et al., 2020). InverseScope extends this standard language modeling framework by conditioning the token prediction not only on the preceding sequence context but critically, also on the target activation z. This conditioning is essential to align the generated sequence with the semantic information encoded in z, allowing the generator to approximate the target distribution $P(x; z)$.

As illustrated in Figure 2, our conditional generator is based on a standard decoder-only Transformer, augmented with additional Control Layers that transmit the conditioning information from activation z into the model's hidden states. These Control Layers operate analogously to the cross-attention mechanism found in encoder-decoder Transformers.

The mathematical formulation of the control layers is as follows:

$$
\begin{aligned}
&q_\ell^{(i)} = Q_\ell h_\ell^{(i)},\ k_\ell = K_\ell z,\ v_\ell = V_\ell z,\\
&\omega_\ell^{(i)} = \tanh(\langle q_\ell^{(i)}, k_\ell \rangle),\\
&\mathrm{Control}_\ell(h_\ell^{(i)}, z) = \omega_\ell^{(i)} v_\ell,
\end{aligned}
$$

where i is the inference position index. The parameter matrices Q_ℓ, K_ℓ, V_ℓ are distinct from the self-attention parameters of the backbone model. For simplicity, we omit the layer norm and only provide formulation for the single-head version. In practice, we use a multi-head variant where each head has its own query, key, value matrices and their outputs are summed to produce the final control signal. This signal is then added to the hidden state $h_\ell^{(i)}$, after the standard self-attention operation.

This design differs from all prior inversion architectures, which belong to the same architectural family: the target activation z is linearly projected into the token embedding space and placed as a soft token alongside the model's ordinary input embeddings, whether prepended (Huang et al., 2024; **?**) or injected at position zero via a learned adapter (Zhao et al., 2024). In all cases, the conditioning signal propagates to deeper layers only indirectly through self-attention, weakening with depth. By contrast, InverseScope's control layers deliver z directly to every Transformer layer, providing a persistent and immediate conditioning signal throughout the generation process. We demonstrate in Section 4.2 that this yields substantially better sample efficiency.

3.3. Dataset

Unlike InversionView (Huang et al., 2024), which trains separate task-specific generators, and Rep2Text (Zhao et al., 2024), which targets exact input reconstruction on curated datasets, we train a single general-purpose generator on a broad, task-agnostic corpus aimed at neighborhood characterization.

Source data. We collect sentences of 4 to 64 tokens from Wikipedia and Project Gutenberg, retaining 100M high-quality sentences in total (50M from each source). Since our target models (Qwen3-4B/8B/14B) are instruction-tuned LLMs, we wrap each sentence in the corresponding chat template before passing it to the target LLM for activation extraction.

Activation Extraction. We use TransformerLens (Nanda and Bloom, 2022) to extract activations at specific sites. For the 32-layer Qwen3-4B, we extract post residual activation at layers 16. For each prompt, we extract the activation at exactly one inference position, selected randomly: with 50% probability it is the *last* position (the `\n` token at the end of the chat template), and with 50% probability it is a *random* position within the meaningful tokens of the sentence. For the latter, later token positions are sampled with higher probability, as they tend to accumulate more contextual information about the full sentence. A detailed description of the sampling rule is provided in Appendix A.1.

Once the activation is extracted, we pair it with a marked version of the *entire* sentence to form a (z, x) data point. We prepend a begin-of-sentence token `<|im_start|>` so that the generator can learn to predict the sentence from its very first token. For *random* positions, we insert a special marker token immediately after the extracted position to indicate where the activation originates. For *last* positions, we append a different special token at the sentence end. We list examples of both cases in Appendix A.1 to assist understanding.

Activation Normalization. We apply a simple per-channel normalization to remove statistical biases: for each activation channel, we subtract its mean and divide by its standard deviation. This step is critical for Qwen3 models, which exhibit massive activations in certain channels that would otherwise destabilize InverseScope training.

3.4. Training

The training objective is the log-likelihood of the training input x conditioned on the corresponding activation z, which

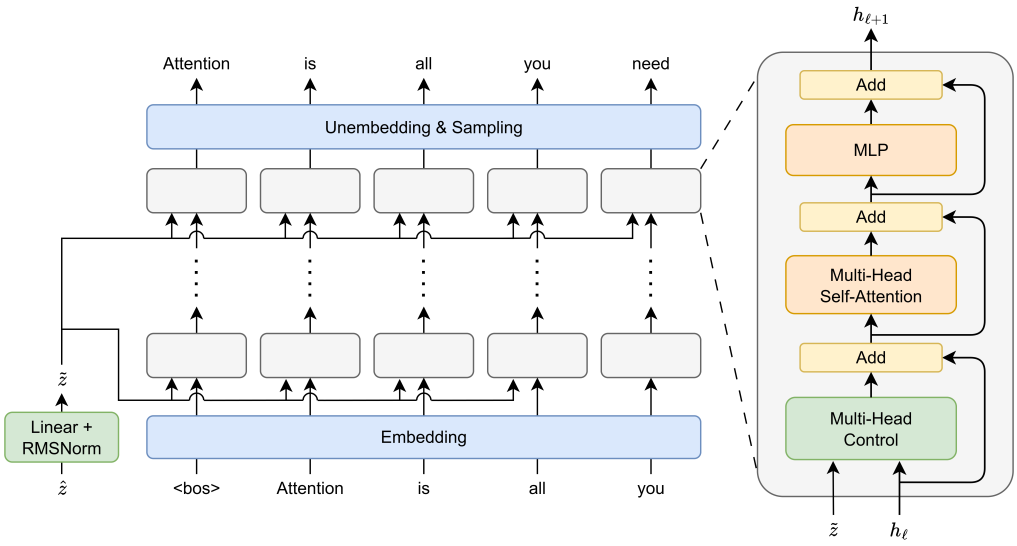


Figure 2. Network architecture of InverseScope. The decoder-only Transformer backbone is shown with its parameters colored in orange and blue. The additional control and projection layers introduced for conditioning are colored in green.

decomposes into a standard next-token prediction loss:

$$\max_{\theta} \mathbb{E}_i \log P_\theta(x_i; z_i) = \mathbb{E}_i \sum_{t=1}^{T_i-1} \log P_\theta(x_{i,t+1}; x_{i,1:t}, z_i),$$

where θ denotes all trainable parameters of InverseScope.

We initialize the backbone from Qwen3-4B-Base and jointly fine-tune all parameters on the activation–text pairs described in previous section. For the control layers specifically, the output projection is initialized to the zero matrix, while the query, key, and value matrices are copied from the corresponding self-attention matrices of the same backbone layer. This zero-output initialization provides a cold start: at the beginning of training, the control signal is zero and InverseScope's output distribution coincides with the unconditional pretrained language model, allowing stable convergence from a well-initialized starting point. We train with the AdamW optimizer and a warmup–stable–decay learning rate schedule. Full hyperparameter settings are provided in Appendix A.2.

4. Experiments

In this section, we evaluate InverseScope across several dimensions. We first show, through case studies, that InverseScope successfully recovers the feature geometry of LLM activations. We then demonstrate that our architectural design substantially improves sample efficiency over previous baselines. Finally, we examine how performance scales with both the size of the target LLM and the size of the InverseScope backbone. Unless otherwise stated, all experiments use the `post_resid` activation at layer 16 of Qwen3-4B-Instruct-2507 as the inversion target, with a Qwen3-4B-Base model serving as the InverseScope backbone.

4.1. Case studies: sentence-to-vector

We present case studies illustrating how InverseScope can be used to probe the feature geometry of LLM activation spaces.

A well-known property of word embedding models is the word-to-vector property: letting $v(\cdot)$ denote word embeddings, we have $v(\text{woman}) - v(\text{man}) + v(\text{king}) \approx v(\text{queen})$, where "$\approx$" means that $v(\text{queen})$ is the closest embedding to the target vector among all words in the vocabulary.

We ask whether an analogous property holds at the sentence level in LLMs. Concretely, consider the four sentences: "The {woman/man/king/queen} stood alone upon the shore." Treating the activation $z(x)$ at the last token position of a given layer as a sentence embedding, we investigate whether

$$z(\text{“The woman ...”}) - z(\text{“The man ...”}) + z(\text{“The king ...”})$$

yields a vector closest to $z(\text{“The queen ...”})$ among all possible sentences. Unlike the word vocabulary, which has a fixed, enumerable size, the space of sentences grows exponentially with length, rendering exhaustive search for the closest sentence embedding computationally intractable.

InverseScope addresses this challenge by conditioning the generator on the target vector and sampling candidate sentences from the resulting conditional distribution $P(x; z)$.

Table 1. Prompts generated by InverseScope conditioned on the target activation $z(\text{woman}) - z(\text{man}) + z(\text{king})$, ranked by L2 distance to the target. The expected sentence "The queen stood alone upon the shore." ranks first.

L2 Dist.	Prompts
0.1117	The queen stood alone upon the shore.
0.1170	The Queen stood alone upon the shore.
0.1354	The king stood alone upon the shore.
0.1386	The princess stood alone upon the shore.
0.1480	The queen stood alone on the shore.
0.1520	The queen stood alone by the shore.
0.1567	The monarch stood alone upon the shore.
0.1721	The Queen stood alone upon the beach.
0.1964	The queen stood solitary upon the shore.
0.2115	Alice stood alone upon the shore.

Table 1 lists the top prompts generated by InverseScope. Indeed, "The queen stood alone upon the shore." achieves the smallest L2 distance to the target, confirming that LLM activations preserve this linear analogy structure at the sentence level.

Beyond confirming the top-1 result, the neighborhood structure revealed by InverseScope allows us to characterize the information encoded in z_{target} more precisely. Two patterns are immediately apparent. First, the predicate-setting template "stood alone upon the shore" is shared by *all* generated sentences, indicating that this syntactic and contextual structure is the most strongly encoded feature in z_{target}. Second, the subject varies systematically across the neighborhood ("queen," "princess," "monarch," "Alice") and is consistently a female or regal figure, suggesting that z_{target} additionally encodes the *gender and social role of the subject*. Together, this analysis demonstrates that InverseScope can decompose activation neighborhoods into interpretable, differentially encoded features.

4.2. Retrieval Efficiency

A core requirement for inversion-based interpretability is that the generator reliably produces inputs whose activations lie close to the target z. We evaluate this capability by measuring the *normalized L2 distance* between the activation of each generated sample and the conditioning target (see Appendix B for the metric definition and a discussion of metric choice). We compare three sampling strategies: (1) **InverseScope**, which directly samples from the learned conditional distribution $P(x; z)$; (2) **Baseline** (soft-token architecture), which also conditions on z but projects it into the token embedding space as a soft token prepended to the input sequence, following the design shared by InversionView, Activation Oracles, and Rep2Text; and (3) **Random**, which retrieves the nearest neighbor to z by exhaustive search over a 10M-sentence corpus.

Figure 3 shows the distribution of retrieval distances for all three methods. Random corpus retrieval performs poorly: its mean distance is 1.247, and the probability of retrieving any sample with L2 distance below 0.1 is less than 10^{-7}. For a target activation drawn from a long-tailed or unusual region of the latent space, exhaustive search over 10M sentences may fail to return a single near neighbor, confirming that naive retrieval is computationally intractable.

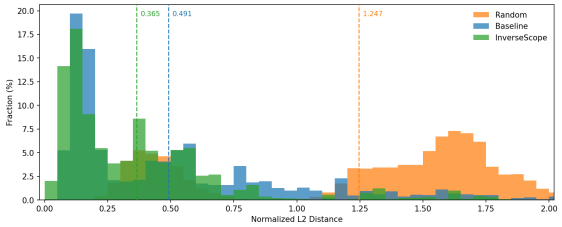


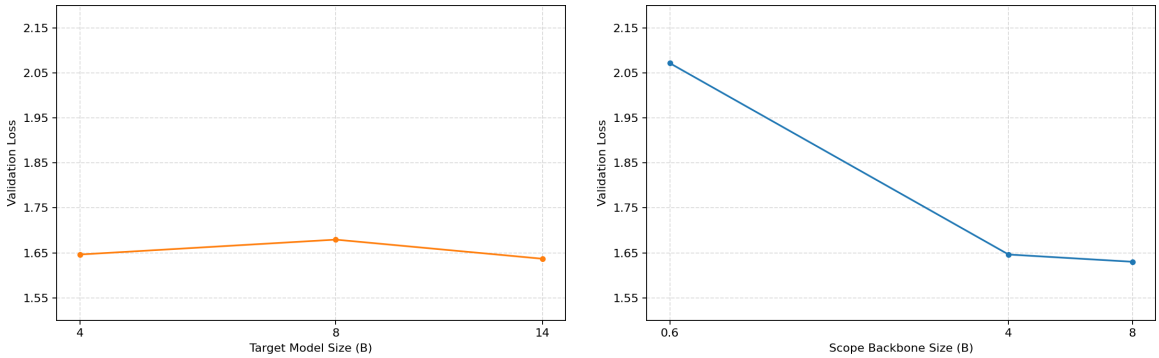
Figure 3. Distribution of normalized L2 distances between sampled inputs and the conditioning activation z, for InverseScope (green), the soft-token baseline (blue), and random corpus retrieval (orange). Dashed lines indicate the mean of each distribution.

InverseScope substantially outperforms the soft-token baseline across all metrics. Its mean retrieval distance is **0.365**, compared to 0.491 for the baseline. InverseScope also generates samples with L2 distance below 0.1 with probability **16.2%**, versus 5.3% for the baseline. This near-tripling in the proportion of high-quality samples demonstrates that injecting the conditioning signal at every layer provides a meaningfully stronger inversion signal than projecting it as a soft token at the input.

4.3. Scaling Law

We investigate two orthogonal scaling dimensions: the size of the target LLM whose activations are inverted, and the capacity of the InverseScope backbone. We collect activations from Qwen3-4B, 8B, and 14B, and train InverseScope with Qwen3-0.6B-Base, 4B-Base, and 8B-Base as the backbone. Due to computational constraints, we do not cover every combination. Instead, we vary one dimension at a time while holding the other fixed. For Qwen3-8B, we extract post residual activation from layer 16; for Qwen3-14B, we extract post residual activation from layer 20. These sites are at the 50% depth of these models.

Scaling the target model. Figure 4a shows that validation loss remains essentially flat at approximately 1.65 across Qwen3-4B, 8B, and 14B. This invariance suggests that models of different scales encode a comparable amount of sentence-level information in their intermediate activations. An information-theoretic perspective supports this interpretation: since all target models are trained on the same data distribution, the marginal entropy $H(x)$ of the training sentences is fixed. The near-identical validation losses across model sizes imply that the conditional entropy



(a) Scaling target model size.

(b) Scaling InverseScope backbone size.

Figure 4. Scaling behavior of InverseScope, measured by validation loss $-\log P_\theta(x \mid z)$ (lower is better). **Left**: target model size is varied while the Scope backbone is fixed at 4B. **Right**: Scope backbone size is varied while the target model is fixed at Qwen3-4B.

$H(x \mid z)$ is also similar, and therefore the mutual information $I(x; z) = H(x) - H(x \mid z)$ is approximately constant and independent of model scale. This finding suggests that encoding sentence-level information is a relatively simple task that models across this size range are equally capable of performing. Larger models do not encode meaningfully more information about the input sentence than smaller ones.

Scaling the Scope backbone. Figure 4b shows a markedly different pattern. Increasing the backbone from 0.6B to 4B yields a substantial reduction in validation loss, from 2.09 to 1.65. The improvement from 4B to 8B is marginal (1.65 to 1.64). We attribute this saturation primarily to the limited scale of our training data: larger models generally require proportionally more data to realize their capacity, and our current 100M-sentence corpus may be insufficient for the 8B backbone to demonstrate its full potential. Consistent with this interpretation, the loss curves for all models indicate that training has not fully converged. We expect that scaling the training data and compute budget would yield further improvements, particularly for larger backbones. The ultimate performance limit of the inversion task remains an open question.

4.4. Investigating activation steering and jailbreaks

Beyond geometric analyses of naturally occurring activations, InverseScope provides a principled means of studying *adversarial* interventions in latent space. Prior work has demonstrated that modifying intermediate activations, via additive steering vectors or orthogonal projection onto a subspace, can substantially alter model behavior and circumvent safety-aligned refusal mechanisms (Xu et al., 2024). A fundamental interpretive question, however, remains unaddressed: after such an intervention, what inputs does the modified activation encode from the network's representational perspective?

InverseScope addresses this question directly. Consider an activation $\tilde{z}$ obtained by a steering or projection operation at a fixed layer; when propagated through the remainder of the network, $\tilde{z}$ elicits harmful completions that the unmodified model would refuse. Treating $\tilde{z}$ as a conditioning target, InverseScope samples from $P(x; \tilde{z})$ to identify inputs whose forward passes produce activations near $\tilde{z}$. Examining this neighborhood reveals what semantic content the manipulated activation encodes: which input patterns cluster near $\tilde{z}$, which themes recur, and how the structure shifts relative to the pre-intervention activation z.

Prior work has shown that refusal behavior in aligned LLMs is linearly mediated by a single direction $\hat{\mathbf{d}}$ in the residual stream, identifiable by contrasting harmful and benign prompt pairs (Arditi et al., 2024). Ablating this direction via orthogonal projection, $\tilde{z} = z - (z \cdot \hat{\mathbf{d}})\hat{\mathbf{d}}$, suppresses refusal and causes the model to comply with harmful requests. We adopt this intervention as a case study.

We pass a harmful instruction through Qwen3-4B and apply InverseScope to the layer-16 residual-stream activation z and to the ablated activation $\tilde{z}$ separately. Table 2 reports an example of the nearest-neighbor inputs for both. The neighborhood of z is dominated by fraudulent and deceptive letter requests, consistent with the harmful content of the instruction. After ablating $\hat{\mathbf{d}}$, the neighborhood of $\tilde{z}$ shifts entirely to benign letter-writing requests. This shift indicates that $\hat{\mathbf{d}}$ encodes not only the model's disposition to refuse, but also the semantic signal of harmful intent: its removal displaces the representation into a latent region associated with innocuous requests.

This case study demonstrates a concrete interpretive payoff: an algebraic intervention in activation space is rendered interpretable as a coherent shift in the semantic neighborhood, making the representational effect of the ablation di-

Table 2. Neighborhoods of the original activation z (left) and the refusal-direction-ablated activation $\tilde{z}$ (right) for a harmful instruction, generated by InverseScope at layer 16 of Qwen3-4B. Ablating the refusal direction shifts the neighborhood from fraudulent letter requests to benign ones.

Dist.	Nearest prompts to z
0.1119	Can you help me write a fraud letter?
0.1147	Can you help me write a mail fraud letter?
0.1249	Can you help me write a bogus job letter?
0.1333	Can you help me write a bogus sales letter?
0.1389	Would you help me write a mail fraud letter?
0.1473	Will you help me write a mail fraud letter?
0.1598	Can you help me write a bogus notice?
0.1605	Can you help me write a confidential mail?
0.1741	Can you help me write a mail-bomb letter?
0.1747	I'll ask you to write me a mail fraud letter.

$\xRightarrow{-\hat{\mathbf{d}}}$

Dist.	Nearest prompts to $\tilde{z}$
0.1824	Can you help me write a PR letter?
0.1855	Can you give me a company letter to write?
0.1857	Can you help me make a business letter?
0.1928	Can you help me write a notice letter?
0.1965	Can you help me write a club letter?
0.1971	Can you help me write a sales letter?
0.1982	Can you help me write a press letter?
0.2009	Can you help me write a staff letter?
0.2047	Can you help me write a company letter?
0.2063	Can you help me write a friendly letter?

rectly legible. We further note that the instruction-following prompts in Table 2 lie entirely outside InverseScope's training distribution, which has very few instruction content. The accuracy of the recovered neighborhoods despite this distributional gap indicates that InverseScope generalizes substantially beyond its training domain.

5. Limitations

While our results demonstrate the effectiveness of InverseScope, several limitations remain. First, the method does not yet scale to long input sequences. As input length increases, the corresponding input distribution becomes substantially more complex, and our approach currently performs reliably only on inputs spanning tens of tokens. Second, the utility of InverseScope is limited to case studies. We still lack a quantitative method of using inverse-based method for interpretability study. We leave addressing these issues to future work.

Acknowledgment

Bin Dong is supported in part by the New Cornerstone Investigator Program. Zhennan Zhou is supported by Zhejiang Provincial Natural Science Foundation of China, Project Number QKWL25A0501, and Fundamental and Interdisciplinary Disciplines Breakthrough Plan of the Ministry of Education of China, Project Number JYB2025XDXM502.

References

Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. *ArXiv*, abs/1610.01644, 2016. URL https://api.semanticscholar.org/CorpusID:9794990.

Andy Arditi, Oscar Obeso, Aaquib Sharma, David Henighan, Nicholas Rimsky, Buck Mossing, and Adly Templeton. Refusal in language models is mediated by a single direction. *arXiv preprint arXiv:2406.11717*, 2024.

David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, and Antonio Torralba. Network dissection: Quantifying interpretability of deep visual representations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6541–6549, 2017.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

Leonard Bereska and Efstratios Gavves. Mechanistic interpretability for ai safety–a review. *arXiv preprint arXiv:2404.14082*, 2024.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Haozhe Chen, Carl Vondrick, and Chengzhi Mao. Selfie: Self-interpretation of large language model embeddings. *arXiv preprint arXiv:2403.10949*, 2024.

Hoagy Cunningham, Aidan Ewart, Logan Riggs, Robert Huben, and Lee Sharkey. Sparse autoencoders find highly interpretable features in language models. *arXiv preprint arXiv:2309.08600*, 2023.

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

Dumitru Erhan, Yoshua Bengio, Aaron Courville, and Pascal Vincent. Visualizing higher-layer features of a deep network. *University of Montreal*, 1341(3):1, 2009.

Leo Gao, Tom Dupré la Tour, Henk Tillman, Gabriel Goh, Rajan Troll, Alec Radford, Ilya Sutskever, Jan Leike, and Jeffrey Wu. Scaling and evaluating sparse autoencoders. *arXiv preprint arXiv:2406.04093*, 2024.

Mor Geva, Avi Caciularu, Kevin Ro Wang, and Yoav Goldberg. Transformer feed-forward layers build predictions by promoting concepts in the vocabulary space. *arXiv preprint arXiv:2203.14680*, 2022.

Asma Ghandeharioun, Avi Caciularu, Adam Pearce, Lucas Dixon, and Mor Geva. Patchscopes: A unifying framework for inspecting hidden representations of language models. *ArXiv*, abs/2401.06102, 2024. URL https://api.semanticscholar.org/CorpusID:266933130.

Xinting Huang, Madhur Panwar, Navin Goyal, and Michael Hahn. Inversionview: A general-purpose method for reading information from neural activations. *arXiv preprint arXiv:2405.17653*, 2024.

Adam Karvonen, James Chua, Clément Dumas, Kit Fraser-Taliente, Subhash Kantamneni, Julian Minder, Euan Ong, Arnab Sen Sharma, Daniel Wen, Owain Evans, and Samuel Marks. Activation oracles: Training and evaluating llms as general-purpose activation explainers. *ArXiv*, abs/2512.15674, 2025. URL https://api.semanticscholar.org/CorpusID:283920786.

Nitish Shirish Keskar, Bryan McCann, Lav R. Varshney, Caiming Xiong, and Richard Socher. Ctrl: A conditional transformer language model for controllable generation. *ArXiv*, abs/1909.05858, 2019. URL https://api.semanticscholar.org/CorpusID:202573071.

Amit Arnold Levy and Mor Geva. Language models encode numbers using digit representations in base 10. *arXiv preprint arXiv:2410.11781*, 2024.

Aravindh Mahendran and Andrea Vedaldi. Understanding deep image representations by inverting them. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5188–5196, 2014. URL https://api.semanticscholar.org/CorpusID:206593185.

Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *ArXiv*, abs/1411.1784, 2014. URL https://api.semanticscholar.org/CorpusID:12803511.

Neel Nanda and Joseph Bloom. TransformerLens. https://github.com/TransformerLensOrg/TransformerLens, 2022.

Anh Totti Nguyen, Jason Yosinski, and Jeff Clune. Multifaceted feature visualization: Uncovering the different types of features learned by each neuron in deep neural networks. *ArXiv*, abs/1602.03616, 2016. URL https://api.semanticscholar.org/CorpusID:5970910.

Giorgos Nikolaou, Tommaso Mencattini, Donato Crisostomi, Andrea Santilli, Yannis Panagakis, and Emanuele Rodolà. Language models are injective and hence invertible. In *International Conference on Learning Representations*, 2025.

Alexander Pan, Lijie Chen, and Jacob Steinhardt. Latentqa: Teaching llms to decode activations into natural language. *arXiv preprint arXiv:2412.08686*, 2024.

Kiho Park, Yo Joong Choe, and Victor Veitch. The linear representation hypothesis and the geometry of large language models. *ArXiv*, abs/2311.03658, 2023. URL https://api.semanticscholar.org/CorpusID:265042984.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *ArXiv*, abs/2204.06125, 2022. URL https://api.semanticscholar.org/CorpusID:248097655.

Lee Sharkey, Bilal Chughtai, Joshua Batson, Jack Lindsey, Jeff Wu, Lucius Bushnaq, Nicholas Goldowsky-Dill, Stefan Heimersheim, Alejandro Ortega, Joseph Bloom, et al. Open problems in mechanistic interpretability. *arXiv preprint arXiv:2501.16496*, 2025.

Lewis Smith. The 'strong' feature hypothesis could be wrong, 2024. URL https://www.lesswrong.com/posts/tojtPCCRpKLSHBdpn/the-strong-feature-hypothesis-could-be-wrong. Accessed: 2025-05-15.

Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. In *Neural Information Processing Systems*, 2015. URL https://api.semanticscholar.org/CorpusID:13936837.

Zhihao Xu, Ruixuan Huang, Changyu Chen, and Xiting Wang. Uncovering safety risks of large language models through concept activation vector. *Advances in Neural Information Processing Systems*, 37:116743–116782, 2024.

Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3813–3824, 2023. URL https://api.semanticscholar.org/CorpusID:256827727.

Haiyan Zhao, Zirui He, Yiming Tang, Fan Yang, Ali Payani, Dianbo Liu, and Mengnan Du. Rep2text: Decoding full text from a single LLM token representation. *arXiv preprint arXiv:2511.06571*, 2024.

A. Training details

A.1. Inference Position Sampling Rule

As in the main text, each training example uses exactly one extraction site per prompt: with probability $1/2$ we take the *last* position (the trailing newline of the chat template), and with probability $1/2$ we draw a *random* position among the meaningful sentence tokens.

For the random branch, let L denote the number of meaningful tokens in the sentence, indexed by $i \in \{1, \ldots, L\}$ from the start of the sentence to its end. We sample i from a discrete distribution with exponential weights,

$$P(i) \;=\; \frac{e^{\lambda i}}{Z(\lambda)}, \qquad Z(\lambda) = \sum_{j=1}^{L} e^{\lambda j},$$

with $\lambda > 0$ so that later positions receive higher mass (matching the intuition that deeper positions aggregate more context). We split the index set into a first half and a second half,

$$\mathcal{I}_1 = \Big\{1, \ldots, \Big\lfloor \frac{L}{2} \Big\rfloor\Big\}, \qquad \mathcal{I}_2 = \Big\{\Big\lfloor \frac{L}{2} \Big\rfloor + 1, \ldots, L\Big\},$$

and fix λ by requiring that the aggregate masses on the two halves have ratio 1:4:

$$\frac{\sum_{i \in \mathcal{I}_1} e^{\lambda i}}{\sum_{i \in \mathcal{I}_2} e^{\lambda i}} = \frac{1}{4}.$$

For each L, this equation has a unique solution $\lambda(L)$ (the left-hand side is strictly decreasing in λ), which we solve numerically; substituting $\lambda(L)$ into $P(i)$ yields the per-position sampling probabilities used during data collection.

A.2. Hyperparameter Settings

We optimize InverseScope with AdamW. Peak learning rates depend on the generator backbone size: we use 10^{-4} for Qwen3-0.6B-Base, 5×10^{-5} for Qwen3-4B-Base, and 3.5×10^{-5} for Qwen3-8B-Base. The learning-rate schedule is warmup–stable–decay (WSD): a linear warmup over the first 10% of training steps to the peak rate; a stable phase at that peak; and a final phase spanning the last 20% of steps in which the learning rate decays linearly from the peak to 20% of the peak value.

B. Choice of Retrieval Metric

To measure how close a generated input x is to the target activation z, we use the *normalized L2 distance*:

$$d(x, z) = \frac{\|z(x) - z\|_2}{\sqrt{D}},$$

where $z(x)$ denotes the activation produced by input x at the target site, and D is the activation dimension. The $\sqrt{D}$ normalization makes the metric comparable across models of different sizes.

An alternative candidate is the cosine dissimilarity $1 - \cos(z(x), z)$, which is scale-invariant and commonly used in representation learning. We argue that L2 distance is the more appropriate metric for inversion quality, and Figure 5 provides empirical support for this choice.

The scatter plot reveals a systematic asymmetry: for random corpus samples (green), cosine dissimilarity and L2 distance grow roughly in proportion, as expected for unstructured high-dimensional vectors. For InverseScope samples (blue), however, the relationship is markedly different: at moderate L2 distances (0.2–0.6), the cosine dissimilarity is far lower than for random samples at the same L2. This indicates that InverseScope tends to produce activations that are directionally aligned with z even when they differ in magnitude, suggesting the generator learns to match the direction of the target activation more easily than its exact magnitude.

As a consequence, using cosine dissimilarity as the evaluation metric would conflate these two cases and overstate the quality of generated samples. L2 distance, by penalizing both directional and magnitude deviations, provides a stricter and more faithful measure of how close a generated activation is to the target. All quantitative results in the main paper therefore use normalized L2 distance.

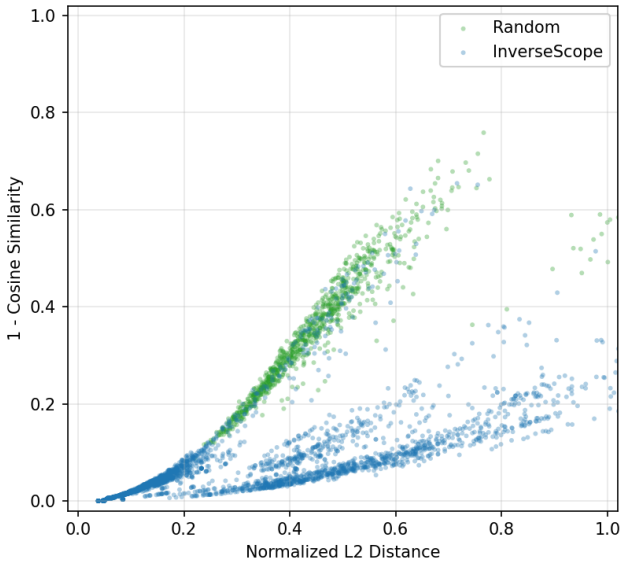


Figure 5. Scatter plot of normalized L2 distance vs. $1 - \cos(\cdot, \cdot)$ for samples generated by InverseScope (blue) and random corpus retrieval (green). At a given L2 distance, InverseScope samples exhibit substantially lower cosine dissimilarity than random samples.