MixCoder: Decomposing Neural Computation into Low-Rank Transformations over Low-Dimensional Representations

Anonymous authors
Paper under double-blind review

Abstract

Fine-grained circuit analysis seeks to decompose neural computation into interpretable units and describe their interactions with attribution graphs. Existing feature-based approaches typically model semantic representations as isolated directions and computation as interactions among features, binding what is represented and how it is transformed within the same feature-level description. Growing evidence instead suggests that neural representations can occupy structured low-dimensional geometries, while effective computation over them can take the form of low-rank transformations. Motivated by this view, we argue that mechanistic interpretability should explicitly separate activation patterns, representations, and transformations, and decompose model computation into low-rank transformations of low-dimensional representations. From a generative model that factorizes these roles, we derive MixCoder, which identifies low-dimensional representation modules, lets representation states determine sparse activation patterns, and uses explicit operators to model transformations over them. Applied to language models, MixCoder reveals which representations are jointly read, how they are transformed, and where the results are written back. We further use this framework to study transformation interference when multiple low-dimensional transformations share limited computational space and its effects on model performance, providing insights into improving performance through mechanistic interpretability.

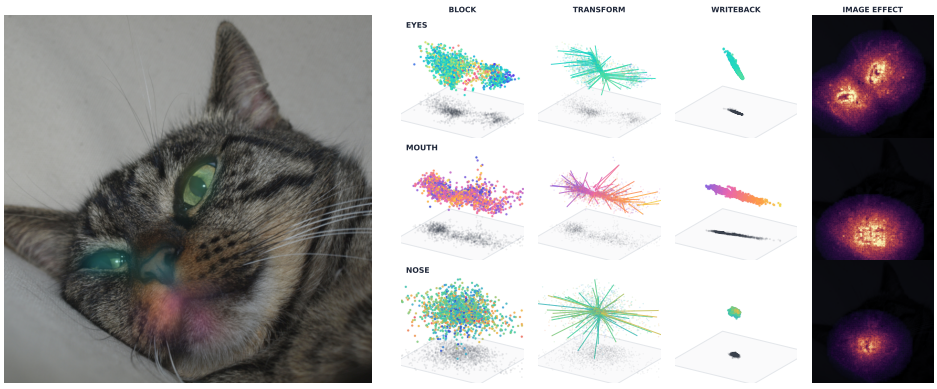


Figure 1: MixCoder exposes how visual representations are transformed and written forward through the model. We train four-dimensional block encoders on DINOv3 layers 6 and 7, freeze them, and train low-rank operators for the layer-6-to-layer-7 update. On a tabby-cat image, we select the source blocks most spatially selective for the eyes, mouth, and nose. The large image maps their coordinates onto the cat; each row then traces one source manifold through the aggregate transformation of the active operators to its writeback in a selected layer-7 readout. Point position shows three block coordinates and color shows the fourth, with a common source/writeback scale. The final column uses feature inversion to visualize the local image change associated with each internal writeback.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

1 Introduction

Fine-grained circuit analysis seeks to explain model behavior by treating neurons as interpretable units and analyzing their interactions (Wang et al., 2022; Ameisen et al., 2025). Models, however, must represent far more latent meanings than they have neurons. Superposition allows them to encode many meanings along nonorthogonal directions in a limited activation space, making individual neurons polysemantic (Elhage et al., 2022). This motivates the hypothesis that a concept corresponds to a single direction in activation space, known as the Linear Representation Hypothesis (LRH) (Park et al., 2023). Researchers use sparse autoencoders (SAEs) to decompose dense activations into sparse linear combinations of interpretable features through dictionary learning (Cunningham et al., 2023; Bricken et al., 2023).

Knowing which features occur inside a model is insufficient to explain its computation. As an illustrative example, suppose that early layers contain features for "star" and "coffee," while a later layer contains a "Starbucks" feature. Identifying these features does not explain how the model combines the first two representations into the third. Transcoders approximate the input–output mapping of the original MLP with an MLP whose wide hidden layer is sparsely active. They express computation through connections between interpretable features, allowing analysis of how strongly one feature influences a subsequent feature (Dunefsky et al., 2024; Ameisen et al., 2025).

Neural representations can also occupy structured low-dimensional geometries embedded in high-dimensional activation space (Engels et al., 2024; Bhalla et al., 2026), and models can compute by manipulating these geometries. Anthropic's MoLT study notes that transcoders decompose addition circuits into lookup-table features for specific input pairs, such as $6 + 9 = 5 \pmod{10}$ (Lindsey et al., 2025). Other work finds that language models encode numbers as generalized helices and manipulate these representations using the "Clock" algorithm to perform addition (Kantamneni & Tegmark, 2025). A feature-level proxy may thus fragment a shared geometric computation into many local rules, obscuring the computational structure used by the model.

Starting from sparse-manifold data generation (Hindupur et al., 2025) and the local affine structure of piecewise-linear networks (Balestriero & Baraniuk, 2018; Black et al., 2022), we explicitly distinguish activation patterns, representations, and transformations. These describe the conditions that invoke computation, the low-dimensional represented states, and their read–write operations. Our mathematical derivation explains why this distinction is necessary to characterize model computation.

We introduce MixCoder, which learns low-dimensional representations with block encoders, selects sparse routes using projections onto the encoded blocks, and binds each selected route to a low-rank writeback operator that reads the aggregate of all encoded blocks. MixCoder can therefore trace which representations a condition reads, how they are transformed, and where the results are written. Figure 1 illustrates these low-rank transformations over low-dimensional representations: for three spatially selective DINOv3 source blocks, it follows their transformations into a layer-7 readout and visualizes the associated local image changes through feature inversion.

Our contributions are: (1) We formulate neural computation as conditional transformations over low-dimensional representations, explicitly separating activation patterns, represented states, and read–write transformations. (2) We introduce MixCoder, combining interpretability with mechanistic fidelity; in Qwen's month and number tasks, cross-block reads carry most of the starting-value–offset interaction, and month write directions can shift answer evidence. (3) We show that writeback capacity, competition between transformations, and coordination among their contributions affect reconstruction and downstream predictions, providing mechanistic insights into improving model performance. Related work is discussed in Appendix G.

2 Problem Formulation and Background

2.1 Transformers and Fine-Grained Circuit Analysis

Let $x_{\text{pre}}^{(l,t)}$ and $x_{\text{mid}}^{(l,t)}$ be the residual states before and after attention at layer l, token t. Write the multi-head attention output as $a^{(l,t)} = \sum_h \text{attn}^{(l,h)}\left(x_{\text{pre}}^{(l,t)}; x_{\text{pre}}^{(l,1:t)}\right)$. A Transformer layer updates them as

$$x_{\text{mid}}^{(l,t)} = x_{\text{pre}}^{(l,t)} + a^{(l,t)}, \qquad x_{\text{pre}}^{(l+1,t)} = x_{\text{mid}}^{(l,t)} + \text{MLP}^{(l)}\left(x_{\text{mid}}^{(l,t)}\right). \tag{1}$$

Attention aggregates information across tokens; the MLP transforms each token state. Both write into a shared residual stream, allowing later components to read their accumulated contributions (Elhage et al., 2021). Circuit analysis seeks interpretable units and causal links among these writes (Wang et al., 2022; Ameisen et al., 2025).

Superposition packs features into non-orthogonal directions, making neurons polysemantic (Elhage et al., 2022). Reading and writing through these directions creates cross-feature interference weights; harmful cross terms can raise loss, while greater representational capacity can reduce them (Turner et al., 2026; Liu et al., 2025).

Sparse autoencoders (SAEs) replace neurons with sparse directional features. Given activation x, code z, and overcomplete dictionary D, a typical objective is (Cunningham et al., 2023; Bricken et al., 2023; Gao et al., 2024)

$$\mathcal{L}_{\text{SAE}} = \mathbb{E}_x\left[\|x - Dz\|_2^2 + \lambda\|z\|_1\right]. \tag{2}$$

2.2 Transcoders and Representation Transformations

A transcoder fits the MLP input-output map rather than reconstructing its input (Dunefsky et al., 2024). Feature i in layer l has an encoding direction $f_{\text{enc}}^{(l,i)}$ and bias that produce scalar activation $z_i^{(l)}(x)$; its decoding direction $f_{\text{dec}}^{(l,i)}$ defines the vector written to the residual stream.

For upstream feature z, MLP output y, and downstream feature z', input-times-gradient attribution obeys $z\,\frac{\partial z'}{\partial z} = z\,\frac{\partial z'}{\partial y}\frac{\partial y}{\partial z}$. Because $\partial y/\partial z$ crosses the input-dependent MLP, this quantity mixes the current activation state with fixed connectivity. A transcoder separates them. Define a later feature's encoding projection as $u_{i'}^{(l')} = \left\langle f_{\text{enc}}^{(l',i')}, \text{MLP}^{(l)}(x)\right\rangle$. The proxy gives

$$u_{i'}^{(l')} \approx \sum_i z_i^{(l)}(x)\left\langle f_{\text{enc}}^{(l',i')}, f_{\text{dec}}^{(l,i)}\right\rangle + c_{i'}, \tag{3}$$

Here $c_{i'}$ is the projected bias. The derivative $\partial u_{i'}^{(l')}/\partial z_i^{(l)} = \left\langle f_{\text{enc}}^{(l',i')}, f_{\text{dec}}^{(l,i)}\right\rangle$ is fixed, so direct attribution factors into an input-dependent activation and an input-independent connection (Dunefsky et al., 2024). Composing these terms yields a sparse attribution graph (Ameisen et al., 2025).

Each transcoder feature still reads one scalar and writes along one fixed direction: its nonlinear map $T_i(x) = \sigma\left(\left\langle f_{\text{enc}}^{(i)}, x\right\rangle + b_i\right) f_{\text{dec}}^{(i)}$ has a Jacobian of rank at most one. Its fixed read–write directions cannot express a higher-rank local transformation alone, so a shared operation on a multidimensional representation can be split across many condition-specific features, resembling lookup rules (Lindsey et al., 2025). MOLT and MxD instead introduce selected low-rank or linear operators (Lindsey et al., 2025; Oldfield et al., 2025); Section 3 links operators to the representations they read and the states that select them.

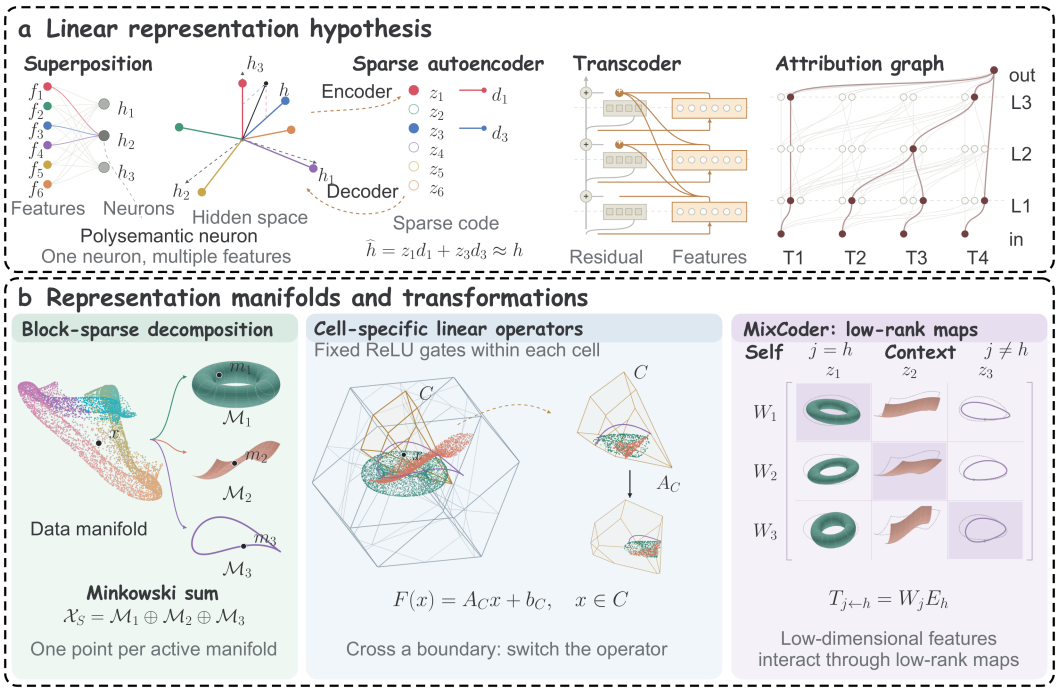


Figure 2: From feature circuits to routed representation transformations. (a) Superposition motivates directional SAE features, which transcoders connect through attribution graphs. (b) Sparse activations combine points from representation manifolds; each ReLU activation cell selects an affine map. MixCoder uses a representation block to select a low-rank operator that can read all encoded blocks. Matrix rows denote selecting routes and columns denote source representations, so off-diagonal entries represent a route reading another block.

3 Methodology

3.1 Data-Generating Process and Local Mappings

Let $x \in \mathbb{R}^d$ be an activation, M the number of representation manifolds, and $S \subseteq [M]$ the indices active in this input. Assume x is a sparse sum of points on these manifolds (Bhalla et al., 2026; Fel et al., 2026):

$$x = \Phi_S(\xi) = \sum_{g \in S} m_g(\xi_g), \qquad m_g(\xi_g) \in \mathcal{M}_g \subseteq \mathcal{V}_g \subseteq \mathbb{R}^d, \qquad |S| \ll M. \tag{4}$$

Here ξ_g specifies the continuous state of an active representation. A fixed low-dimensional encoder block can span a curved $\mathcal{M}_g$ only when its linear span $\mathcal{V}_g$ is low-dimensional. Block sparsity constrains S, not coordinates within a selected block (Fel et al., 2026).

For a one-hidden-layer MLP $F(x) = W_{\text{out}}\phi(W_{\text{in}}x + b_{\text{in}}) + b_{\text{out}}$, let neuron q read along v_q, write along u_q, and have preactivation $s_q(x) = v_q^\top x + b_q$. Under ReLU, $F(x) = \sum_q \mathbf{1}[s_q(x) > 0] u_q s_q(x) + b_{\text{out}}$: the sign of s_q determines activation, and its positive value sets writeback amplitude. The active set $\mathcal{I}(x)$ defines a cell $C_\mathcal{I} = \{x : \mathcal{I}(x) = \mathcal{I}\}$ where $F(x) = A_\mathcal{I} x + b_\mathcal{I}$, $A_\mathcal{I} = \sum_{q \in \mathcal{I}} u_q v_q^\top$, and $b_\mathcal{I} = b_{\text{out}} + \sum_{q \in \mathcal{I}} u_q b_q$. Thus the activation pattern selects a local affine operator (Balestriero & Baraniuk, 2018; Black et al., 2022). Away from ReLU boundaries, and also for smooth activations, $J_F(x) = \sum_q \phi'(s_q(x)) u_q v_q^\top$.

For fixed S, inputs lie in the Minkowski sum $\mathcal{X}_S = \sum_{g \in S} \mathcal{M}_g$. Linear maps preserve this sum; within the cell selected by x, the MLP's linear part acts additively on the constituent representations:

$$A_{\mathcal{I}}\mathcal{X}_S = \sum_{g\in S} A_{\mathcal{I}}\mathcal{M}_g, \qquad F(x) - b_{\mathcal{I}} = \sum_{g\in S} A_{\mathcal{I}} m_g(\xi_g) \quad (x \in \mathcal{X}_S \cap C_{\mathcal{I}}). \tag{5}$$

Changing the representation states ξ_g changes $s_q(x) = \sum_{g\in S} v_q^\top m_g(\xi_g) + b_q$, selecting the active-neuron set and its local operator. Thus S specifies what is represented, the states select when a transformation acts, and $A_{\mathcal{I}}$ specifies its action. Let Q_g be an orthonormal basis for the tangent to $\mathcal{M}_g$ at $m_g(\xi_g)$ and Q_h an orthonormal basis for the local directions of a target output representation. The restricted Jacobian is

$$B_{h\leftarrow g}(x) := Q_h^\top J_F(x) Q_g = \sum_q \underbrace{\phi'(s_q(x))}_{\text{activation pattern}} \underbrace{(Q_h^\top u_q)}_{\text{transformation}} \underbrace{(v_q^\top Q_g)}_{\text{representation read}}. \tag{6}$$

The bases preserve each representation's local degrees of freedom. The activation pattern selects terms; their read and write factors determine the transformation. These are the three roles MixCoder approximates separately.

3.2 MixCoder Model Specification

MixCoder has N encoder blocks $E_j \in \mathbb{R}^{d\times p}$ with $E_j^\top E_j = I_p$; different blocks may overlap. This Stiefel constraint makes their norm-based routing scores comparable. Define

$$z_j = E_j^\top x + b_j, \qquad P_j = E_j E_j^\top, \qquad a_j(x) = \|z_j\|_2. \tag{7}$$

Here $b_j \in \mathbb{R}^p$ is the encoder bias. It shifts the block's coordinate origin, while the Stiefel constraint fixes the scale of its basis. The routing score is the norm of this affine code. Appendix D specifies the centered encoding used in the GPT-2 experiments.

Block sparsity selects whole representation states (Fel et al., 2026). MixCoder retains their signed coordinates and uses the block norm to select routes and set writeback amplitude. This preserves the geometry of each active block under a change of basis.

For route count K, $\mathcal{A}(x) = \text{Top-K}\{a_j(x)\}_{j=1}^N$ selects the K blocks with largest biased code norms. Selection chooses operators; $a_j(x)$ sets each selected operator's amplitude. Every operator reads the signed aggregate of all blocks:

$$\bar{x}(x) = \sum_{h=1}^N E_h z_h = P_\Sigma x + b_\Sigma, \qquad P_\Sigma = \sum_{h=1}^N P_h, \qquad b_\Sigma = \sum_{h=1}^N E_h b_h. \tag{8}$$

The aggregate combines the encoded representation states. Route j binds $W_j = U_j V_j^\top$ of rank at most r, with $U_j, V_j \in \mathbb{R}^{d\times r}$. As in SwiGLU, a gate selects a computation while separate factors read its input and write its output (Shazeer, 2020). The reader learns which directions in the aggregate contribute to the transformation. MixCoder fits the original MLP output $F(x)$ by

$$\hat{y}(x) = b + \sum_{j\in\mathcal{A}(x)} a_j(x) W_j \bar{x}(x), \qquad \mathcal{L} = \mathbb{E}_x \|F(x) - \hat{y}(x)\|_2^2. \tag{9}$$

The global output bias $b \in \mathbb{R}^d$ sets the common writeback origin; the block biases $b_j \in \mathbb{R}^p$ set the encoding origins. Prediction MSE trains the encoder and operator parameters together with the trainable biases. The experimental configurations specify centering and the norm or linear routing rule in Appendix D.

3.3 Decomposing Low-Dimensional Transformations and Identifiability

Separate routing and operator inputs let us isolate the action of route j on source block h. Define $T_{j\leftarrow h} = W_j E_h \in \mathbb{R}^{d\times p}$; then

$$\hat{y}(x) - b = \sum_{j\in\mathcal{A}(x)} a_j(x) \sum_{h=1}^{N} T_{j\leftarrow h} z_h. \tag{10}$$

Since $\operatorname{rank}(T_{j\leftarrow h}) \le \min(r,p)$, its compact SVD $T_{j\leftarrow h} = L_{jh}\Sigma_{jh}R_{jh}^\top$ gives

$$T_{j\leftarrow h} z_h = \sum_{s=1}^{r_{jh}} \sigma_{jhs}\, \ell_{jhs}(v_{jhs}^\top z_h), \qquad r_{jh} = \operatorname{rank}(T_{j\leftarrow h}). \tag{11}$$

Here ℓ_{jhs} and v_{jhs} are column s of L_{jh} and R_{jh}. Each singular component reads a within-block direction, scales it, and writes into the residual stream. One route can transform several source blocks, and one source can participate in several conditional transformations. Downstream projections locate their writeback.

Under a within-block rotation $E_h \mapsto E_h R_h$, $b_h \mapsto R_h^\top b_h$, with $R_h \in O(p)$, the coordinates and effective map become $z_h \mapsto R_h^\top z_h$ and $T_{j\leftarrow h} \mapsto T_{j\leftarrow h} R_h$. Their writeback and singular values are invariant. The natural objects for attribution are therefore the representation blocks and their effective maps. Appendix A gives route-recovery conditions for the zero-encoder-bias submodel.

4 Experiments

We ask how a sparse set of operators transforms represented states, and how the capacity of their writeback space affects fidelity. GPT-2 provides a general mapping benchmark; Qwen's month and number tasks provide structured transformations whose reads and writes can be examined geometrically. Synthetic-manifold experiments appear in Appendix C, and experimental protocols and controls in Appendices D–F.

Metrics and baselines. MSE and fraction of variance explained (FVE) measure output fit; replacement ΔCE (nats/token) and KL measure the effect on language-model predictions; teacher–proxy Jacobian–vector-product (JVP) cosine measures local mapping fidelity. We compare against transcoders (Dunefsky et al., 2024) and sparse and dense MOLT (Lindsey et al., 2025). Output SAEs (Cunningham et al., 2023; Gao et al., 2024) serve as output-reconstruction controls.

All JVP values in Table 1 use the same ambient probes. Appendix D reports a separate tangent-space evaluation that removes the inaccessible LayerNorm-normal direction.

4.1 GPT-2: Replacement Fidelity and Block-Level Computation

We fit the layer-6 MLP of GPT-2 Small using approximately 50M OpenWebText tokens. Table 1 averages two seeds on a shared held-out set. Norm/all uses centered block coordinates in Eq. 9; norm/x reads the raw MLP input, and linear/all learns a separate scalar routing score for each operator. Appendix D gives the encoding, routing, training, and evaluation details.

At matched active count and operator rank, linear/all MixCoder improves all reported quality metrics over Top-64 MOLT (Table 1). Combining a learned condition with access to all encoded states gives the strongest MixCoder fit. At matched forward MACs, Top-256 MOLT has lower reconstruction error, while linear/all MixCoder has lower replacement ΔCE; Appendix D gives the compute comparison. The condition–source maps expose how each fitted operator reads represented states and distributes its writeback.

Table 1: GPT-2 Small layer-6 results, averaged over seeds 0 and 42. Active is the mean number of selected modules. Top-64 MOLT and MixCoder match active count and rank per operator (64); dense MOLT is a higher-compute reference.

Model	Active	MSE ↓	FVE ↑	ΔCE ↓	KL ↓	JVP ↑
Output dictionary reference						
SAE (64 active)	64	0.1931	0.6544	0.0183	0.0186	0.4577
SAE (256 active)	256	0.0776	0.8612	0.0038	0.0061	0.6201
Replacement models						
Transcoder (Top-256)	256	0.1929	0.6549	0.0203	0.0230	0.4407
Transcoder (Top-4096)	4053	0.1770	0.6833	0.0198	0.0241	0.5893
MOLT (Top-64)	64	0.0908	0.8375	0.0057	0.0100	0.7040
MOLT (Top-256)	256	0.0727	0.8699	0.0052	0.0079	0.7640
MOLT (dense)	1439	0.0491	0.9122	0.0025	0.0049	0.8287
MixCoder (norm/all)	64	0.1037	0.8144	0.0079	0.0113	0.4423
MixCoder (norm/x)	64	0.0936	0.8325	0.0068	0.0108	0.6975
MixCoder (linear/all)	64	0.0766	0.8629	0.0044	0.0077	0.7181

The output SAEs reconstruct the observed MLP write y. Their two sparsity levels match either the number of active blocks or their total coordinate count. The improvement with more active scalars illustrates how dictionary reconstruction benefits from additional representational degrees of freedom.

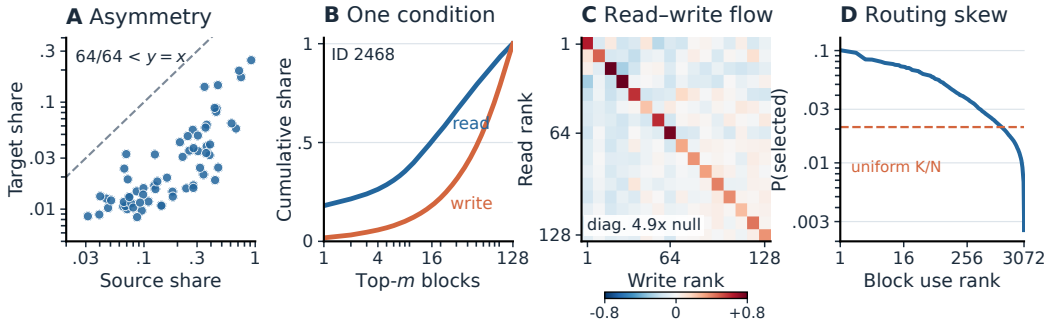


Figure 3: Conditions, reads, and writes differ. Norm/all MixCoder, seed 42. (A) Selected conditions concentrate their reads more strongly on their own blocks than their writes. (B) Cumulative read and write concentration for one condition. (C) Source–target paths compared with an independence reference preserving the marginals. (D) Uneven route use; the dashed line marks uniform Top-64 selection.

Selected operators usually read most strongly from their own condition block, yet distribute their writeback over more target blocks (Figure 3). The SVD of each condition–source map resolves its within-block read directions, singular values, and residual-stream write directions. Diagonal maps describe a condition reading its own block, while off-diagonal maps retain interactions with other represented blocks. We next examine the computational roles of these two terms in a known transformation.

4.2 Qwen: Conditional Transformations of Month and Number Representations

Month representations can form a two-dimensional circle (Engels et al., 2024), while number representations combine an ordered magnitude axis with periodic coordinates into a generalized helix (Kantamneni & Tegmark, 2025). We study two offset tasks: from a starting value n and offset k, the model returns month $(n + k)$ mod 12 or integer $n + k$. Varying n

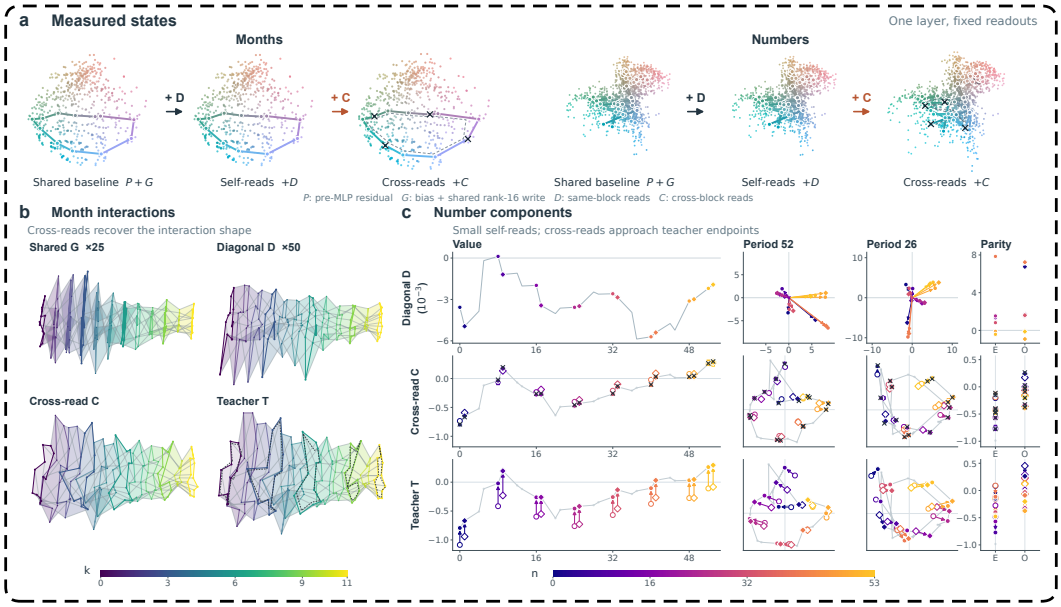


Figure 4: Reading the month and number transformations in a Qwen layer-34 proxy. All states use frozen readouts. (a) Within each task, the same axes show the shared baseline $P + G$, then the states after adding self-reads D and cross-reads C; bold paths fix offset $k = 3$, and crosses mark original-MLP endpoints. The month view also shows the input offset axis. (b) Month writeback interactions after removing the separate effects of starting month and offset. G and D are enlarged 25× and 50×; C and teacher remain unscaled. (c) At number offset $k = 3$, rows show the small D write vectors (in 10^{-3} units), the state movement after adding C, and the original MLP's pre/post states. Columns are frozen value, candidate periodic, and parity readouts; open-to-filled markers connect paired states.

and k independently lets us isolate writes that depend jointly on the represented value and the offset. We analyze the layer-34 MLP of Qwen3-4B-Instruct-2507.

The question is whether the block selecting an operator is also the representation that operator chiefly reads. We fit MixCoder to the MLP output and evaluate month and number states with independently trained, frozen readouts. This proxy uses a shared rank-16 map W_0 for route-independent writes, leaving the private rank-16 operators to fit conditional structure. Let r_{in} be the residual state before the MLP write, x its normalized MLP input, ζ_h the radial code of block h, $s = \sum_h E_h \zeta_h$ the encoded aggregate, and $g_j(x)$ the weight of route j. The post-MLP residual separates into

$$r_{\text{in}} + \hat{y}(x) = \underbrace{r_{\text{in}}}_{P} + \underbrace{\big(b + W_0 s\big)}_{G} + \underbrace{\sum_j g_j W_j E_j \zeta_j}_{D} + \underbrace{\sum_j g_j \sum_{h \neq j} W_j E_h \zeta_h}_{C}. \tag{12}$$

Here P is the incoming residual, G the shared write, D the writes from operators reading their own condition blocks, and C the writes from operators reading other blocks. We trained the readouts on separate discovery prompts and evaluated a full 12×12 month grid and a sampled 64×6 number grid under four held-out templates (Appendix F).

Month and number geometry is already present in $P+G$. The MLP transforms these states: self-reads D make small directional corrections, while cross-reads C account for most of the movement toward the original MLP's output (Figure 4).

Removing the separate effects of starting value and offset isolates their joint transformation. Cross-reads carry most of this interaction in both tasks, including the value, periodic, and parity directions of the number readout. This shows how conditional operators combine

information from several representation blocks to update the state. Appendix F reports the component measurements and interventions on the original MLP.

4.3 Writeback Rank, Transformation Interference, and Performance

Increasing writeback-operator rank substantially improves reconstruction: with the encoder frozen, raising rank from 4 to 256 reduces held-out MSE from 0.8481 to 0.1165. We propose three hypotheses for this gain: higher rank supplies additional useful write directions, alleviates competition between transformations for read and write directions, and supports coordination among write contributions.

Figure 5A compares frozen-encoder, trainable-encoder, and fixed-active-rank-budget ($Kr = 4096$) settings; B compares jointly trained read and write factors with either factor fixed. Increasing rank improves reconstruction across these settings, and joint training yields lower error. Both writeback capacity and the configuration of read and write directions therefore affect reconstruction performance.

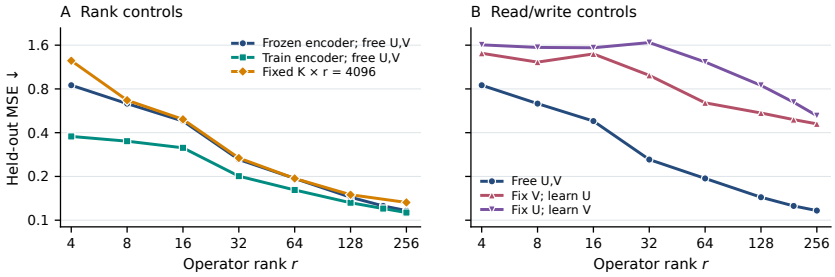


Figure 5: Writeback rank and GPT-2 layer-6 MLP reconstruction. (A) Encoder and active-rank-budget controls. (B) Jointly trained read/write factors versus fixing either factor.

Mechanism tests further show that additional directions carry signal unavailable within the retained write space. In a rank-16 fixed-writer model, separating overlapping directions reduces MSE by 0.023; perturbing coordination among learned write contributions changes downstream predictions. These results support a role for both effective write space and transformation interference in model performance. Appendix D.3 reports the full hypothesis tests.

We also find in DINOv3 that the same write is detectable by one downstream linear readout while producing almost no response in another active readout. Appendix E gives this observation and the direction-editing experiments.

5 Conclusion

MixCoder separates activation patterns, representations, and transformations, and describes computation as conditional low-rank maps over low-dimensional states. Its block encoders and explicit read–write operators expose which representations participate in a computation and where their transformed states are written. On GPT-2, linear/all MixCoder improves replacement fidelity over Top-64 MOLT at matched active count and operator rank. In Qwen's month and number tasks, cross-block reads carry most of the starting-value–offset interaction, while month write directions can shift answer evidence in the original model. Writeback-rank and direction interventions further show that computational capacity, competition between transformations, and coordination among their contributions affect performance. These findings connect representation geometry to model computation, making shared transformations and their interference accessible to mechanistic analysis and providing insights into improving model performance.

AI use statement

Generative AI tools were used to edit the manuscript for clarity and to assist in writing research code. The authors take responsibility for the final manuscript, code, and results.

References

Emmanuel Ameisen, Jack Lindsey, Adam Pearce, Wes Gurnee, Nicholas L. Turner, Brian Chen, Craig Citro, David Abrahams, Shan Carter, Basil Hosmer, Jonathan Marcus, Michael Sklar, Adly Templeton, Trenton Bricken, Callum McDougall, Hoagy Cunningham, Thomas Henighan, Adam Jermyn, Andy Jones, Andrew Persic, Zhenyi Qi, T. Ben Thompson, Sam Zimmerman, Kelley Rivoire, Thomas Conerly, Chris Olah, and Joshua Batson. Circuit tracing: Revealing computational graphs in language models. Transformer Circuits Thread, 2025. URL https://transformer-circuits.pub/2025/attribution-graphs/methods.html.

Randall Balestriero and Richard G. Baraniuk. A spline theory of deep learning. In Proceedings of the 35th International Conference on Machine Learning, volume 80 of Proceedings of Machine Learning Research, pp. 374–383. PMLR, 2018. URL https://proceedings.mlr.press/v80/balestriero18b.html.

Usha Bhalla, Thomas Fel, Can Rager, Sheridan Feucht, Tal Haklay, Daniel Wurgaft, Siddharth Boppana, Matthew Kowal, Vasudev Shyam, Jack Merullo, Atticus Geiger, and Ekdeep Singh Lubana. Do sparse autoencoders capture concept manifolds?, 2026. URL https://arxiv.org/abs/2604.28119.

Sid Black, Lee Sharkey, Leo Grinsztajn, Eric Winsor, Dan Braun, Jacob Merizian, Kip Parker, Carlos Ramón Guevara, Beren Millidge, Gabriel Alfour, and Connor Leahy. Interpreting neural networks through the polytope lens, 2022. URL https://arxiv.org/abs/2211.12312.

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, 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. Transformer Circuits Thread, 2023. URL https://transformer-circuits.pub/2023/monosemantic-features/index.html.

Xi Chen, Mingyu Jin, Jingcheng Niu, Yutong Yin, Jinman Zhao, Bangwei Guo, Dimitris N. Metaxas, Zhaoran Wang, Yutao Yue, and Gerald Penn. All circuits lead to rome: Rethinking functional anisotropy in circuit and sheaf discovery for LLMs, 2026. URL https://arxiv.org/abs/2605.12671.

SueYeon Chung and L. F. Abbott. Neural population geometry: An approach for understanding biological and artificial neural networks. Current Opinion in Neurobiology, 70: 137–144, 2021. doi: 10.1016/j.conb.2021.10.010.

SueYeon Chung, Daniel D. Lee, and Haim Sompolinsky. Classification and geometry of general perceptual manifolds. Physical Review X, 8(3), 2018. doi: 10.1103/PhysRevX.8.031003.

Arthur Conmy, Augustine N. Mavor-Parker, Aengus Lynch, Stefan Heimersheim, and Adrià Garriga-Alonso. Towards automated circuit discovery for mechanistic interpretability, 2023. URL https://arxiv.org/abs/2304.14997.

Hoagy Cunningham, Aidan Ewart, Logan Riggs, Robert Huben, and Lee Sharkey. Sparse autoencoders find highly interpretable features in language models, 2023. URL https://arxiv.org/abs/2309.08600.

Seyed Arshan Dalili and Mehrdad Mahdavi. Subspace-aware sparse autoencoders for effective mechanistic interpretability, 2026. URL https://arxiv.org/abs/2606.06333.

Jacob Dunefsky, Philippe Chlenski, and Neel Nanda. Transcoders find interpretable llm feature circuits, 2024. URL https://arxiv.org/abs/2406.11944.

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. https://transformer-circuits.pub/2021/framework/index.html.

Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, Roger Grosse, Sam McCandlish, Jared Kaplan, Dario Amodei, Martin Wattenberg, and Christopher Olah. Toy models of superposition. Transformer Circuits Thread, 2022. URL https://transformer-circuits.pub/2022/toy_model/index.html.

Joshua Engels, Eric J. Michaud, Isaac Liao, Wes Gurnee, and Max Tegmark. Not all language model features are one-dimensionally linear, 2024. URL https://arxiv.org/abs/2405.14860.

Lucy Farnik, Tim Lawson, Conor Houghton, and Laurence Aitchison. Jacobian sparse autoencoders: Sparsify computations, not just activations. In Proceedings of the 42nd International Conference on Machine Learning, volume 267 of Proceedings of Machine Learning Research, pp. 16178–16236. PMLR, 2025. URL https://proceedings.mlr.press/v267/farnik25a.html.

Thomas Fel, Matthew Kowal, Mozes Jacobs, Dron Hazra, Usha Bhalla, Lee Sharkey, Lucius Bushnaq, Satchel Grant, Tal Haklay, Thomas Icard, Can Rager, Michael Pearce, Daniel Wurgaft, Aiden Swann, Fenil Doshi, Siddharth Boppana, Curt Tigges, Nick Cammarata, Thomas Serre, Vasudev Shyam, Owen Lewis, Thomas McGrath, Jack Merullo, Ekdeep Singh Lubana, and Atticus Geiger. Structuring sparsity: Block-sparse featurizers capture visual concept manifolds, 2026. URL https://arxiv.org/abs/2606.25234.

Collin Francel. SMIXAE: Towards unsupervised manifold discovery in language models, 2026. URL https://arxiv.org/abs/2605.09224.

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, 2024. URL https://arxiv.org/abs/2406.04093.

Wes Gurnee, Neel Nanda, Matthew Pauly, Katherine Harvey, Dmitrii Troitskii, and Dimitris Bertsimas. Finding neurons in a haystack: Case studies with sparse probing, 2023. URL https://arxiv.org/abs/2305.01610.

Wes Gurnee, Theo Horsley, Zifan Carl Guo, Tara Rezaei Kheirkhah, Qinyi Sun, Will Hathaway, Neel Nanda, and Dimitris Bertsimas. Universal neurons in gpt2 language models, 2024. URL https://arxiv.org/abs/2401.12181.

Wes Gurnee, Emmanuel Ameisen, Isaac Kauvar, Julius Tarng, Adam Pearce, Chris Olah, and Joshua Batson. When models manipulate manifolds: The geometry of a counting task. Transformer Circuits Thread, 2025. URL https://transformer-circuits.pub/2025/linebreaks/index.html.

Sai Sumedh R. Hindupur, Ekdeep Singh Lubana, Thomas Fel, and Demba Ba. Projecting assumptions: The duality between sparse autoencoders and concept geometry. In Advances in Neural Information Processing Systems, 2025. URL https://arxiv.org/abs/2503.01822.

Harish Kamath, Emmanuel Ameisen, Isaac Kauvar, Rodrigo Luger, Wes Gurnee, Adam Pearce, Sam Zimmerman, Joshua Batson, Thomas Conerly, Chris Olah, and Jack Lindsey. Tracing attention computation through feature interactions. Transformer Circuits Thread, 2025. URL https://transformer-circuits.pub/2025/attention-qk/index.html.

Subhash Kantamneni and Max Tegmark. Language models use trigonometry to do addition, 2025. URL https://arxiv.org/abs/2502.00873.

Matthew T. Kaufman, Mark M. Churchland, Stephen I. Ryu, and Krishna V. Shenoy. Cortical activity in the null space: Permitting preparation without movement. Nature Neuroscience, 17:440–448, 2014. doi: 10.1038/nn.3643.

Panagiotis Koromilas, Andreas D. Demou, James Oldfield, Yannis Panagakis, and Mihalis A. Nicolaou. PolySAE: Modeling feature interactions in sparse autoencoders via polynomial decoding, 2026. URL https://arxiv.org/abs/2602.01322.

Christopher Langdon, Mikhail Genkin, and Tatiana A. Engel. A unifying perspective on neural manifolds and circuits for cognition. Nature Reviews Neuroscience, 24 (6):363–377, 2023. doi: 10.1038/s41583-023-00693-x. URL https://doi.org/10.1038/s41583-023-00693-x.

Patrick Leask, Bart Bussmann, Michael Pearce, Joseph Bloom, Curt Tigges, Noura Al Moubayed, Lee Sharkey, and Neel Nanda. Sparse autoencoders do not find canonical units of analysis. In International Conference on Learning Representations, 2025. URL https://arxiv.org/abs/2502.04878.

Jack Lindsey, Brian Chen, Adam Pearce, Sasha Hydrie, and Thomas Conerly. Sparse mixtures of linear transforms, 2025. URL https://transformer-circuits.pub/2025/bulk-update/index.html. Transformer Circuits research update.

Yizhou Liu, Ziming Liu, and Jeff Gore. Superposition yields robust neural scaling. In Advances in Neural Information Processing Systems, 2025. URL https://openreview.net/forum?id=knPz7gtjPW.

Maxime Méloux, Silviu Maniu, François Portet, and Maxime Peyrard. Everything, everywhere, all at once: Is mechanistic interpretability identifiable? In The Thirteenth International Conference on Learning Representations, 2025. URL https://openreview.net/forum?id=ZYXTLo7kCi.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in gpt. In Advances in Neural Information Processing Systems, volume 35, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/6f1d43d5a82a37e89b0665b33bf3a182-Abstract-Conference.html.

Chris Olah. What is a linear representation? what is a multidimensional feature? Transformer Circuits Thread, 2024. URL https://transformer-circuits.pub/2024/july-update/index.html#linear-representations.

Chris Olah and Joshua Batson. Feature manifold toy model. Transformer Circuits Thread, 2023. URL https://transformer-circuits.pub/2023/may-update/index.html#feature-manifolds.

Chris Olah, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov, and Shan Carter. Zoom in: An introduction to circuits. Distill, 5(3), 2020. doi: 10.23915/distill.00024.001.

James Oldfield, Shawn Im, Sharon Li, Mihalis A. Nicolaou, Ioannis Patras, and Grigorios G. Chrysos. Towards interpretability without sacrifice: Faithful dense layer decomposition with mixture of decoders. In Advances in Neural Information Processing Systems, 2025. URL https://arxiv.org/abs/2505.21364.

OpenAI. Language models can explain neurons in language models, 2023. URL https://openai.com/index/language-models-can-explain-neurons-in-language-models/.

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

Gonçalo Paulo and Nora Belrose. Sparse autoencoders trained on the same data learn different features, 2025. URL https://arxiv.org/abs/2501.16615.

Or Shafran, Shaked Ronen, Omri Fahn, Shauli Ravfogel, Atticus Geiger, and Mor Geva. From directions to regions: Decomposing activations in language models via local geometry, 2026. URL https://arxiv.org/abs/2602.02464.

Noam Shazeer. Glu variants improve transformer, 2020. URL https://arxiv.org/abs/2002.05202.

Xiangchen Song, Aashiq Muhamed, Yujia Zheng, Lingjing Kong, Zeyu Tang, Mona T. Diab, Virginia Smith, and Kun Zhang. Position: Mechanistic interpretability should prioritize feature consistency in SAEs, 2025. URL https://arxiv.org/abs/2505.20254.

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. Transformer Circuits Thread, 2024. URL https://transformer-circuits.pub/2024/scaling-monosemanticity/index.html.

Fabian J. Theis. Towards a general independent subspace analysis. In Advances in Neural Information Processing Systems, 2006. URL https://proceedings.neurips.cc/paper_files/paper/2006/file/20479c788fb27378c2c99eadcf207e7f-Paper.pdf.

Nicholas L. Turner, Jeffrey Wu, and Joshua Batson. Characterizing interference weights in a tiny language model. Transformer Circuits Thread, 2026. URL https://transformer-circuits.pub/2026/interference_effectiveness_helpfulness/index.html.

Kevin Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. Interpretability in the wild: A circuit for indirect object identification in gpt-2 small, 2022. URL https://arxiv.org/abs/2211.00593.

A Proofs

The route-recovery theorems in this appendix analyze the submodel with encoder biases fixed to $b_j = 0$, not the general model with learned encoder biases. With biases, route-exchange boundaries also contain linear and constant terms, and the aggregate is $P_\Sigma x + b_\Sigma$. The homogeneous formulas below must not be applied to that case without a separate proof.

A.1 Why Identify Blocks?

A transcoder circuit uses learned units to attribute feature interactions, QK scores, and OV writes. If independent training runs divide the same computation among different units, their attribution graphs may disagree. We study which blocks and operators are identifiable from the routes and outputs of a fitted MixCoder proxy.

Formally, a parameterized object is identifiable from an observation if any two parameter settings that give the same observation agree on that object, after accounting for declared symmetries. Here the observation is the proxy's route and output as functions of the queried input. Block labels may be permuted and bases within a block may be rotated, so the target is the unordered projectors and their paired effective operators. Finite-sample recovery and cross-run agreement require separate empirical evaluation.

Our routing-black-box interface returns the proxy output $f_\theta(x)$ and its discrete Top-K set $\mathcal{A}_P(x)$ for a queried input x. It does not reveal routing scores, encoder matrices, or writeback operators. Let $E_j \in \mathbb{R}^{d\times p}$ have orthonormal columns and define

$$P_j = E_j E_j^\top, \qquad a_j(x) = \|E_j^\top x\|_2 = \sqrt{x^\top P_j x}, \qquad P_\Sigma = \sum_{h=1}^{N} P_h.$$

The route selects the K largest scores, and the proxy predicts

$$f_\theta(x) = b + \sum_{j\in\mathcal{A}_P(x)} a_j(x) W_j P_\Sigma x.$$

Our first target is the unordered projector set $\{P_j\}$. We then ask whether the output determines each attached operator on $\mathcal{R}(P_\Sigma)$, the only input space it receives. These are two distinct recovery steps.

A.2 Block Coordinates Do Not Affect Attribution

Proposition A.1 (within-block gauge invariance). Replacing E_j by $E_j R_j$ for any $R_j \in O(p)$ leaves the routing scores, selected set, aggregate input, output, and each block's written vector unchanged.

Proof. Orthogonality gives

$$(E_j R_j)(E_j R_j)^\top = P_j, \qquad \|(E_j R_j)^\top x\|_2 = \|E_j^\top x\|_2.$$

Consequently P_Σ, $\mathcal{A}_P(x)$, and $a_j(x) W_j P_\Sigma x$ are unchanged. A common permutation of block labels and their attached operators only relabels the same vectors. □

The encoder-basis and label symmetry is $O(p)^N \rtimes S_N$. If $W_j = U_j V_j^\top$, its factors have an additional invertible change-of-basis freedom that does not alter W_j. Block-to-block attribution should therefore use projectors and residual-stream vectors, not entries of an arbitrary within-block coordinate matrix.

A.3 Recovering Projectors from Route Exchanges

Suppose the selected set changes from $S \cup \{i\}$ to $S \cup \{j\}$ at a regular boundary, where $|S| = K - 1$ and all other comparisons remain strict. At the exchange, $a_i(x) = a_j(x)$, so its observed boundary points satisfy

$$x^\top (P_i - P_j)x = 0.$$

The two neighboring routes also tell us which side has $a_i > a_j$. Thus one exchange gives a boundary equality and its orientation without revealing either score. Repeating this across blocks gives a joint system. A single boundary generally cannot determine both projectors: its zero set may leave the scale of $P_i - P_j$ unresolved. We also use the constraints $P_j^\top = P_j$, $P_j^2 = P_j$, and $\mathrm{rank}(P_j) = p$.

We require three conditions on the observed route geometry:

1. Coverage. The observation domain contains regular pieces of the relevant exchange boundaries and points on both sides. Every block to be recovered participates in an observed exchange.
2. Connected exchanges. The graph of observed exchanged pairs connects all N blocks.
3. Joint rigidity. Among rank-p orthogonal projector collections, the observed boundary equalities and side orientations have only one joint solution, up to a common block permutation.

Theorem A.2 (black-box projector recovery). Under these conditions, the discrete route observed across regular exchanges determines the unordered set $\{P_j\}_{j=1}^N$.

Proof. Each exchange supplies the equality above and the ordering on its two sides. The fitted proxy's projectors satisfy every constraint. Joint rigidity excludes any other rank-p projector collection except a common relabeling. □

This is a conditional black-box result. Joint rigidity concerns the observed data; it does not follow from the Stiefel constraint alone. A full-rank boundary-constraint Jacobian on the product Grassmann tangent space of dimension $Np(d-p)$ checks local rigidity, but does not rule out a separate global solution. Finite-query recovery experiments test whether an algorithm can recover a fitted proxy's projectors; they do not alone certify global uniqueness. If numerical scores were observable, $\frac{1}{2}\nabla_x^2 a_j(x)^2 = P_j$ would give a simpler recovery route, but those scores are excluded here.

A.4 Reading Operators After Recovering Projectors

Once P_j is known, both the gate $a_j(x)$ and the common operator input $P_\Sigma x$ are known. If the response of an active block, $\Delta_j(x) = a_j(x) W_j P_\Sigma x$, can be separated on an open neighborhood where $a_j(x) > 0$, then

$$D_x\left(\frac{\Delta_j(x)}{a_j(x)}\right) = W_j P_\Sigma.$$

Since P_Σ is invertible on its range, this determines W_j on $\mathcal{R}(P_\Sigma)$. The derivative here is of a separated block response. Differentiating a total output with several active blocks gives their summed derivatives, not each operator separately.

For the total-output interface, the operators can instead be read jointly. Let $s = \mathrm{rank}(P_\Sigma)$, let $B \in \mathbb{R}^{d \times s}$ be an orthonormal basis of $\mathcal{R}(P_\Sigma)$, and define $u(x) = B^\top P_\Sigma x$, $M_j = W_j B$, and $r_j(x) = \mathbf{1}\{j \in \mathcal{A}_P(x)\}$. Then

$$f_\theta(x) = b + \sum_{j=1}^{N} r_j(x) a_j(x) M_j u(x).$$

This equation is linear in the unknown $b, M_1, \ldots, M_N$. Stack its known inputs as $\phi(x) = [1; r_1 a_1 u; \ldots; r_N a_N u]$. Response excitation means that $\mathbb{E}[\phi(x)\phi(x)^\top]$ is positive definite on this coefficient space.

Theorem A.3 (conditional operator readout). If Theorem A.2 and response excitation hold, the observed route and output determine b and every $W_j|_{\mathcal{R}(P_\Sigma)}$, paired with its recovered P_j, up to the same block permutation.

Proof. The recovered projectors make $\phi(x)$ known. If two coefficient matrices produce the same output, their difference C satisfies $C\phi(x) = 0$ almost surely. Hence

$$0 = \mathbb{E}\|C\phi(x)\|_2^2 = \operatorname{tr}\big(C\,\mathbb{E}[\phi(x)\phi(x)^\top]C^\top\big)\,.$$

Positive definiteness implies $C = 0$, so b and each $M_j = W_j B$ are unique. The columns of B span every input the operator can receive. □

Corollary A.4 (written contributions). Under Theorems A.2 and A.3, the route-level vectors $r_j(x)a_j(x)W_j P_\Sigma x$ and source-to-route vectors $r_j(x)a_j(x)W_j P_h x$ are determined up to one block permutation. They do not depend on the basis chosen within either block. After fixing the downstream attribution rule, these vectors can enter feature, QK, and OV attribution calculations.

A.5 Remaining Cross-Run Ambiguity

Theorems A.2–A.3 concern one exact fitted proxy. They show how its structure can be checked from its route and response. Independent runs pose a further question: do they learn the same projector collection and paired operators for the same teacher? Recovering each run's parameters does not force those two collections to agree. Different block decompositions may approximate the same total mapping, so cross-run block matching and attribution consistency must be measured separately.

MixCoder permits overlapping blocks, so classical independent-subspace uniqueness results do not settle cross-run agreement. The operator is unobservable on $\mathcal{R}(P_\Sigma)^\perp$, which it never receives. Fidelity to the original model remains an empirical question evaluated through function, Jacobian, and intervention checks.

B Limitations and Future Work

Interpreting representation geometry. A learned block does not come with a semantic description of its internal geometry. Its norm says when the block is selected, but does not explain what movement within the block means. Language-model-based methods have generated descriptions of individual neurons from activation examples (OpenAI, 2023). An analogous method for representation manifolds would need to interpret neighborhoods, directions, and trajectories within a block, and test those descriptions against interventions rather than assign one label to the whole block.

Between-block identifiability. Invariance to a change of basis within a block does not settle whether a fitted proxy has a unique decomposition between blocks. Appendix A identifies projectors and effective writeback operators only under boundary-rigidity and operator-excitation conditions, which we have not established for pretrained-model activations. Independent subspace analysis provides uniqueness results under an independent-source decomposition (Theis, 2006), but its assumptions do not directly cover MixCoder's overlapping subspaces, input-dependent routing, and shared operator reads. A proxy whose blocks are fully identifiable in this setting remains an open goal.

Scaling with manifold-valued representations. Prior work connects the superposition and interference of directional features to neural scaling (Elhage et al., 2022; Liu et al., 2025). It does not yet give a corresponding scaling account when the representational units are manifolds with internal degrees of freedom. Deriving how block number, manifold dimension, sparsity, and transformation interference jointly affect capacity and loss is a separate theoretical problem.

Computational cost. MixCoder retains block-level Top-K sparsity, but its basis-invariant block interpretation rules out generic coordinatewise activations or sparsification within a block: such operations change under a rotation of the block basis. Basis-invariant radial nonlinearities remain possible. Dense block projections and, in the all-block variant, aggregation over every block add computation. In the GPT-2 comparison, norm/x MixCoder uses 15.73M forward MACs per token versus 8.65M for Top-64 MOLT at matched active count and operator rank. Reducing this cost without losing the geometric interpretation is important for larger proxies.

Nonuniform representation dimension. A single fixed block dimension may not suit every concept. Block-sparse featurizers on DINOv3 report mean stable rank saturating around two to four dimensions (Fel et al., 2026), whereas a separate GPT-2 analysis reports 90%-variance PCA dimensions of 14 for a temporal slice and 33 for a geography slice (Dalili & Mahdavi, 2026). These statistics measure different properties and are not directly comparable, but they caution against treating four-dimensional blocks as a universal choice. Future models should test adaptive block dimensions and whether fixed-size blocks fragment higher-dimensional representations.

C Synthetic Manifold Toy Experiment

Following the BSF manifold zoo (Fel et al., 2026), we construct material instances from line segments, circles, disks, spheres, tori, Möbius strips, Swiss rolls, and helices. Each instance is centered, normalized, and embedded into a common space with a random orthonormal frame. Four material contributions form each input:

$$x = \frac{1}{\sqrt{4}} \sum_{g \in S} m_g, \qquad |S| = 4.$$

After composing the materials, we apply a known random mapping to obtain the training target:

$$y = f(x) = \alpha \left[Lx + W_2 \, \mathrm{ReLU}(W_1 x + b_{\mathrm{T}}) - \mu_{\mathrm{T}} \right].$$

The teacher is a biased ReLU MLP with an additional linear term. Its activation pattern defines the true cells: the mapping is affine within each cell, continuous across boundaries, and has an analytic Jacobian. Materials, cells, and local transformations therefore have explicit ground truth. Figure 6 shows the materials used to compose the input.

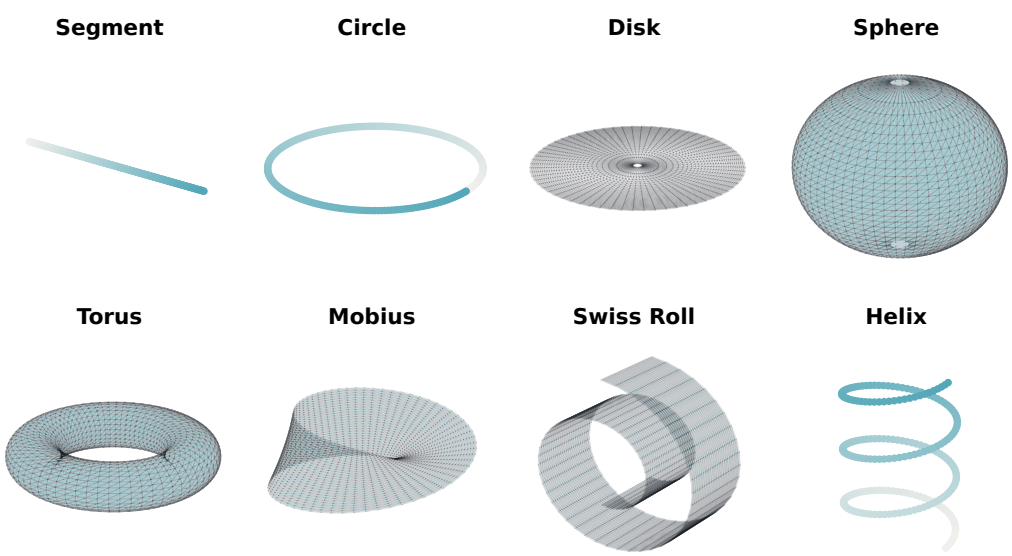


Figure 6: Ground-truth materials used to compose the input. One instance represents each of eight geometric families; the generator contains 128 instances. These are materials before embedding and summation, not model predictions.

We compare MixCoder all, transcoders at two sparsity levels (Dunefsky et al., 2024), pure MOLT (Lindsey et al., 2025), and MOLT with SAE or BSF endpoints. MixCoder selects private operators by block norm and applies them to a carrier formed from all encoded blocks. Its biased encoder and operators learn jointly using prediction-y MSE only. The sandwich baselines instead pretrain their endpoint heads for reconstruction and freeze them.

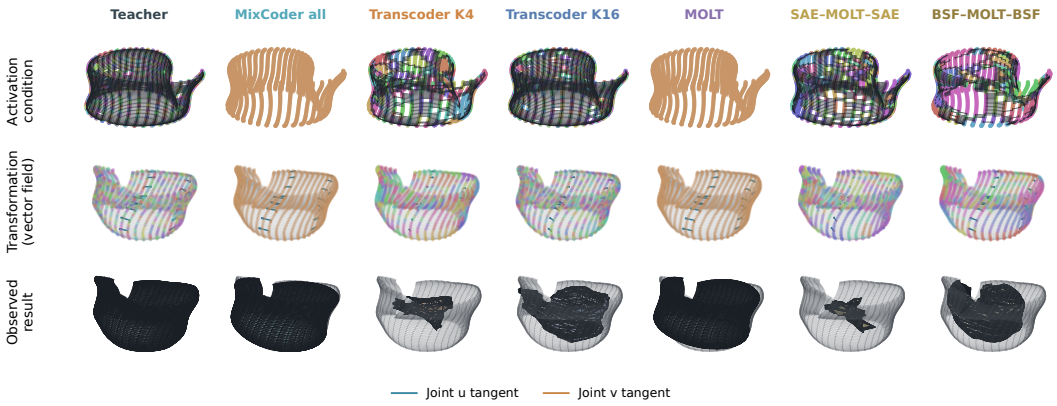


Figure 7: Cells, transformations, and outputs with four jointly varying materials. Columns show the teacher and six models; rows show complete activation-support partitions, local tangent transformations, and output surfaces. Dark lines in the first row mark each method's own support boundaries. Blue and orange arrows in the second row show JVPs along the two joint tangents, anchored at identical teacher output points. The bottom row includes the teacher surface as a pale reference. Methods share the projection, camera, and scale within each row.

Similar output surfaces do not imply similar activation partitions. In Figure 7, MixCoder and pure MOLT produce outputs closer to the teacher while their supports change little; transcoders introduce more local partitions.

The table reports three metrics answering separate questions. Prediction NMSE measures output error. Tangent JVP cosine compares model and teacher changes along true manifold tangents. Within-cell fragments measure additional activation partitions inside true cells.

Table 2: Output prediction error, tangent JVP agreement, and within-cell fragments in the single-seed experiment.

Method	NMSE ↓	JVP ↑	Fragments
MixCoder all	0.1033	0.8808	1.0000
Transcoder K4	0.3740	0.5207	1.1935
Transcoder K16	0.1275	0.7557	1.6458
SAE–MOLT–SAE	0.3672	0.4734	1.2976
BSF–MOLT–BSF	0.2593	0.6304	1.1339
Pure MOLT	0.0963	0.8794	1.0000

MixCoder and pure MOLT combine low prediction error with high tangent agreement in this experiment. Operator models allow multidirectional input transformations, whereas transcoders combine fixed write directions; this structural difference offers a possible explanation for the local-action differences. Sandwich errors also include frozen endpoint effects and cannot be attributed entirely to the intermediate MOLT.

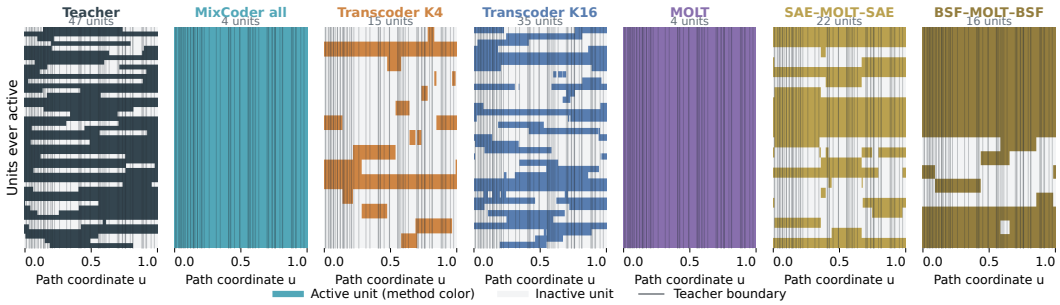


Figure 8: Per-unit activation matrices reveal cell subdivisions. Horizontal position follows the same predetermined circle path, and rows represent units. Method color denotes active units, pale gray denotes inactive units, and thin vertical lines mark teacher-cell boundaries. Only units inactive along the entire path are omitted; retained counts appear above each panel. This figure shows one diagnostic path, while table fragments aggregate six paths.

Increasing transcoder activity improves prediction while introducing more subdivisions inside true cells. Figure 8 exposes this trade-off through changes in active combinations: fixed-direction units approximate the mapping with more local combinations, producing a more fragmented, lookup-like implementation. Better fit therefore does not imply recovery of the same computational mechanism, and subdivisions alone do not establish lookup computation.

Fewer fragments do not establish faithful cell recovery either. MixCoder and pure MOLT introduce no detected extra subdivisions on the evaluation paths, but also fail to recover teacher boundaries.

D Full GPT-2 Ablations

Data and optimization. We use GPT-2 Small's zero-based layer 6, with input dimension 768 and context length 128. OpenWebText documents are packed in dataset order with one EOS per document. Consecutive, disjoint segments provide 65,536 calibration tokens, 49,995,776 training tokens, and 65,536 validation tokens. Calibration fixes the input mean μ_x and initializes the output bias to the teacher-output mean. The benchmark averages seeds 0 and 42. Each run consumes the full training segment in 6,103 updates of 8,192 tokens. AdamW uses peak learning rate 3×10^{-4}, 500 warmup steps, cosine decay, zero weight decay, and $\beta = (0.9, 0.999)$. Models with at least 100M trainable parameters use the paged 8-bit AdamW implementation. Encoder frames are retracted to the Stiefel manifold after each update. Benchmark reconstruction uses all validation tokens; replacement metrics and JVP follow the shared evaluation protocol for every condition.

Encoding and routing. Encoded sources use $z_j = E_j^\top(x - \mu_x)$, corresponding to the tied encoder bias $b_j = -E_j^\top \mu_x$ in Eq. 7. Norm routing scores $\|z_j\|_2$. Linear routing learns an independent vector $t_j \in \mathbb{R}^d$ and scores $q_j = t_j^\top x$. Both rules select the largest K raw scores. Amplitude writes use the selected scores, while binary writes use unit weights; the binary training rule uses a straight-through gradient. Local input is $E_j z_j$, active input sums the selected blocks, and all input sums every block. The raw-input control reads x. The benchmark trains the block frames, private factors, global output bias, and linear router where present.

Compute comparison. Top-64 MOLT and norm/x MixCoder cost 8.65M and 15.73M forward MACs per token. Linear/all MixCoder costs 27.53M, approximately matching Top-256 MOLT. At this compute level, MOLT gives lower MSE and higher FVE/JVP, while MixCoder gives lower replacement ΔCE (Table 1).

Writeback-rank protocol. The rank experiments use seed 42 and the same token order, training length, batch size, learning-rate schedule, and zero weight decay. Parameters are BF16. All groups begin with the seed-42 norm/binary/all encoder, whose four-dimensional blocks and calibration statistics are held fixed except in G1. Operators are initialized to the same function across ranks, $W_j(0) = E_j E_j^\top$; additional factor channels preserve that initial function. The global output bias remains fixed. The rank set is $\{4, 8, 16, 32, 64, 128, 192, 256\}$.

Table 3: Trainable components and route counts in the writeback-rank sweep and controls.

Group	Encoder	Read/write factors	Route count
Main frozen-encoder sweep	Fixed	Train U, V	$K = 64$
G1	Train	Train U, V	$K = 64$
G2	Fixed	Train U, V	$K = 4096/r$
Fixed-U	Fixed	Train V	$K = 64$
Fixed-V	Fixed	Train U	$K = 64$

The four controls use block-norm routing, norm amplitudes, and all-block reads. G2 contains seven ranks, omitting 192 because $4096/r$ is not integral. Reconstruction in Figure 5 uses the first 10,000 tokens of the shared validation segment and the final trained models. The mechanism tests in Appendix D.3 instead use 16 fixed held-out tokens from the 31 G1/G2/fixed-factor models, preserving input and routing during each intervention. Their intervals are paired token-bootstrap intervals.

The combination tables below average seeds 0 and 42.

We evaluate 16 configurations crossing norm or linear routing, binary or amplitude writes, and local, active-block, all-block, or raw-input reads. The two tables split them by router. All 16 MixCoder configurations select 64 four-dimensional blocks and use private rank-64 operators. Raw x is a direct-input control and differs from the encoded sources in centering as well as source construction. JVP values in these tables use the same ambient probes as the main table.

D.1 Norm routing: eight combinations

Table 4: Norm routing across eight read/write combinations, averaged over seeds 0 and 42.

Write	Input	FVE ↑	ΔCE ↓	JVP ↑
binary	local	0.5490	0.0305	0.3086
binary	active	0.6745	0.0182	0.3278
binary	all	0.7660	0.0131	0.5651
binary	raw x	0.7899	0.0112	0.6515
amplitude	local	0.6446	0.0211	0.4025
amplitude	active	0.7104	0.0161	0.3191
amplitude	all	0.8144	0.0079	0.4423
amplitude	raw x	0.8325	0.0068	0.6975

D.2 Linear routing: eight combinations

Table 5: Linear routing across eight read/write combinations, averaged over seeds 0 and 42.

Write	Input	FVE ↑	ΔCE ↓	JVP ↑
binary	local	0.5305	0.0328	0.3402
binary	active	0.6945	0.0166	0.3719
binary	all	0.7439	0.0130	0.5624
binary	raw x	0.7628	0.0111	0.6400
amplitude	local	0.6527	0.0223	0.4658
amplitude	active	0.7445	0.0119	0.3938
amplitude	all	0.8629	0.0044	0.7181
amplitude	raw x	0.8448	0.0054	0.7194

Broader encoded reads generally improve fit: an operator can combine information distributed across blocks. Linear/amplitude/all gives the strongest encoded-source result, while raw-input reads work best among norm-router variants. The value of continuous amplitudes depends on both the selecting rule and the available source state.

JVP evaluation for all-block readers. LayerNorm constrains the input geometry. Ambient probes include its normal direction, whereas tangent probes follow directions available to the model. Using the same tangent projection for teacher and proxy raises norm/all JVP from 0.4423 to 0.6387. The change is smaller for other configurations, identifying the all-block aggregate's sensitivity to probe geometry.

Radial asinh controls. With calibration mean μ_x, let $r_j = \|E_j^\top(x - \mu_x)\|_2$, $\mathcal{A}(x) =$ Top-64$\{r_j\}$, and $s_{\text{all}}(x) = \sum_h E_h E_h^\top (x - \mu_x)$. We replace the selected write amplitude by $\phi(r_j) = \operatorname{asinh}(r_j)$, giving $b + \sum_{j \in \mathcal{A}(x)} \phi(r_j) W_j s_{\text{all}}(x)$. The shared/private variant uses $W_0 + W_j$ in each selected term. A decoder-only control rescales each selected block's coordinates to $\phi(r_j) E_j^\top (x - \mu_x)/r_j$ and writes through an independent decoder, with the continuous value zero at $r_j = 0$. BSF uses unscaled block coordinates.

Table 6: Radial nonlinearity controls at GPT-2 Small layer 6. All rows use seed 42, the same ordered 49,995,776 training tokens, four-dimensional blocks, and Top-64 routing. FVE uses 65,536 held-out tokens; ΔCE uses 8,128 next-token positions; JVP is mean cosine over 128 tokens with four ambient probes each.

Model	Shared rank	Private rank	FVE ↑	ΔCE ↓	JVP ↑
BSF (linear code)	–	–	0.5474	0.0279	0.3130
asinh decoder	–	–	0.3253	0.0506	0.1250
asinh MixCoder	–	64	0.8045	0.0069	0.5495
Shared + private	4	16	0.3519	0.0875	0.0679
Shared + private	4	64	0.6113	0.0348	0.2161
Shared + private	16	4	0.1231	0.1867	0.0230
Shared + private	16	16	0.4498	0.0652	0.0939
Shared + private	16	64	0.6457	0.0276	0.2571
Shared + private	64	4	0.1464	0.1817	0.0315
Shared + private	64	16	0.5035	0.0543	0.1147

The ordinary asinh model jointly trains the encoder and private read/write factors; the decoder and shared/private controls freeze an existing encoder. Table 6 shows how additional private or shared rank changes fit. The shared/private controls use BF16 and a different initialization, so encoder training, precision, and initialization form part of their experimental configuration.

Route use and block geometry. Figure 9 shows the same 512 frequently routed blocks under three geometries for linear/amplitude/all (seed 42). Condition vectors use cosine distance, representations use Grassmann distance between four-dimensional subspaces, and operators use covariance-weighted action distance.

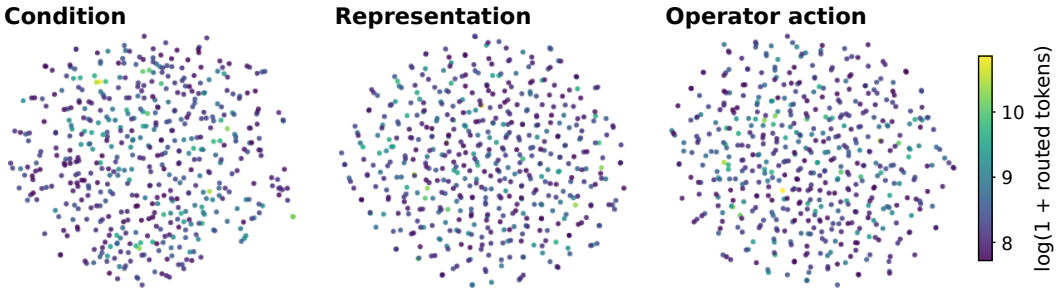


Figure 9: Three views of the same 512 high-frequency blocks in the linear/amplitude/all model, seed 42. Color encodes shared route frequency.

D.3 Diagnosing the Benefits of Writeback Rank

Why does increasing operator rank improve the GPT-2 proxy in Section 4.3? Under superposition, feature directions are generally nonorthogonal: activity in one feature also produces responses along other feature directions, creating virtual cross-feature weights that can increase loss. Coordinatewise activations can suppress weaker incidental responses and reduce this interference (Elhage et al., 2022; Turner et al., 2026).

Each selected MixCoder operator reads the aggregate of all encoded blocks, so it must distinguish relevant content from unrelated superposed components. Our block-sparse spike prior selects computation by block activity and applies the same operator across states within that block. Coordinatewise nonlinearities would instead filter coordinates according to the within-block state and make computation depend on the chosen basis. We therefore retain linear reads between the encoded source and operator writeback, without coordinatewise activations. One way to exclude unrelated content is to place its directions in the approximate null space of the reader: if $V_j^\top E_h \approx 0$, then $W_j = U_j V_j^\top$ scarcely reads source block

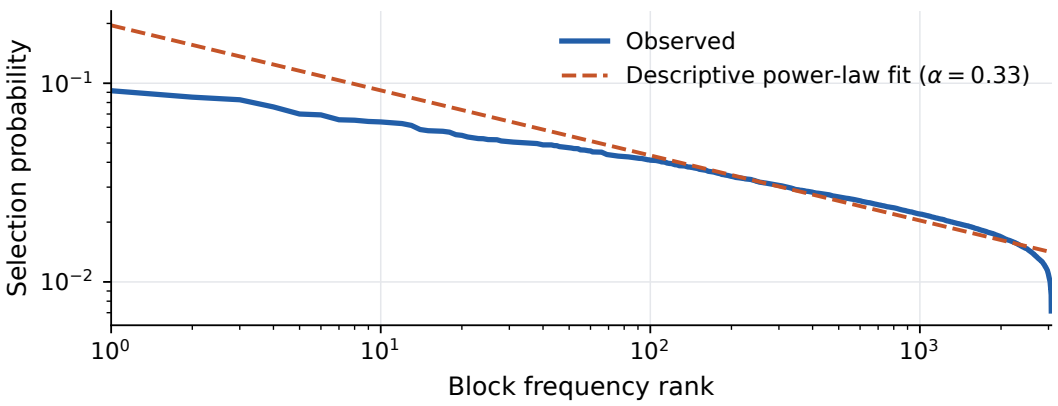


Figure 10: Routing rank–frequency for norm/amplitude/active, seed 42, over 65,536 held-out tokens. The fitted exponent is 0.327.

h. This depends on learned read directions and block geometry; it does not automatically suppress every unrelated block.

Each operator writes within its output column space. Exact reproduction of the teacher's local Jacobian therefore requires the joint active write space to reach at least the teacher Jacobian rank; approximate reconstruction requires coverage of the important output directions.

We explore three possible causes through write-space truncation, read/write rotations, and downstream perturbations: capacity, geometric competition, and contribution structure.

We reuse GPT-2 Small layer-6 checkpoints from the preceding experiments: jointly trained G1, G2 with fixed $Kr = 4096$, and fixed-U/V controls. In $W_j = U_j V_j^\top$, U_j is the writer and V_j the reader. All interventions use the same 16 held-out tokens, hold routing and input fixed, and compare reconstruction MSE and downstream replacement KL. Figure intervals use paired token bootstraps.

D.3.1 H1: higher rank supplies additional useful write directions

Capacity test. Write $W_j = A_j \Sigma_j B_j^\top$ and retain its leading $q \in \{4, 16, 64\}$ singular channels (at most its rank), giving output f_q. Let $t = \widehat{y} - f_q$ be the deleted output, and let P_q project onto the union of the retained write directions across active operators. We compare

$$f_q, \qquad f_q + P_q t, \qquad f_q + (I - P_q)t, \qquad f_q + t = \widehat{y}. \tag{13}$$

The middle outputs restore the deleted contribution inside or outside that union. Let Q_q be its orthonormal basis, so $P_q = Q_q Q_q^\top$. For G1 and $q = 4$, the union has dimension 256.

Even with arbitrary coefficients in this union, the smallest possible loss is

$$B_q(x) = \min_a \frac{\|b + Q_q a - y\|_2^2}{d} = \frac{\|(I - P_q)(y - b)\|_2^2}{d}. \tag{14}$$

If $B_q > L(\widehat{y})$, no output confined to this retained union can match the full model on that token. The bound concerns the retained directions in the current model.

The capacity test asks whether the retained write space could reproduce the full model even after its coefficients were optimized freely. For G1 rank 64, its oracle loss exceeds the full-model loss by 0.048 (95% CI $[0.010, 0.081]$). The missing directions therefore carry useful target signal. Restoring the deleted contributions shows how both the retained and complementary spaces contribute to reconstruction.

D.3.2 H2: read/write competition under low-rank constraints

Let $c_j(x) = a_j(x) W_j \bar{x}(x) \in \mathbb{R}^d$ be the write contribution of active operator j. For each token, we take the 16 operators with largest $\|c_j(x)\|_2^2$ and select the pair with greatest joint

leading-channel overlap,

$$\rho_{jk} = (a_{j1}^\top a_{k1})^2 (v_{j1}^\top v_{k1})^2, \tag{15}$$

where a_{j1} and v_{j1} are the leading write and read singular vectors. We rotate one member's leading read vector, write vector, or both toward or away from its partner, preserving rank and singular values. Equal-angle random rotations provide controls.

If two transformations compete for similar read and write directions, separating them should improve reconstruction more than an arbitrary rotation. In freely trained G1, alignment and separation produce small changes with intervals spanning zero. The response is clearer when the writer is fixed.

In fixed-U rank 16, separating the leading directions reduces MSE by 0.023 (95% CI for ΔMSE $[-0.048, -0.007]$) and improves on the equal-angle random control. Alignment moves the loss in the opposite direction. This pattern across low-rank fixed-writer controls links the constrained model's error to competition for read/write directions (Figure 11).

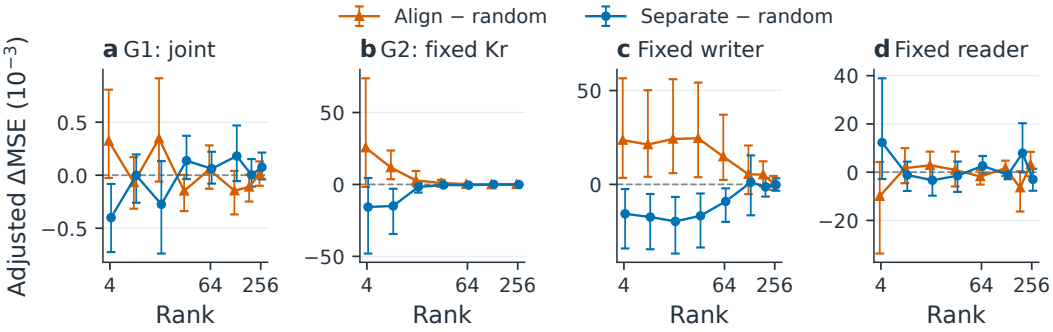


Figure 11: H2: conditional evidence from geometry edits across four checkpoint groups. Directed alignment (orange) or separation (blue) of leading read and write directions, each compared with equal-angle random rotations. The ordinate is paired MSE change in 10^{-3} units; negative values favor the directed edit. Error bars are 95% token-bootstrap intervals after eight controls are averaged within token. Vertical scales differ by group.

D.3.3 H3: contribution structure and downstream sensitivity

We reconnect the actual GPT-2 suffix and divide write space into its 16 most KL-sensitive directions and the lower-sensitivity complement. Let G_H be the local KL metric within subspace H and c_j the contribution of operator j. We measure cancellation by

$$C_H = 1 - \frac{(\sum_j c_j)^\top G_H (\sum_j c_j)}{\sum_j c_j^\top G_H c_j}. \tag{16}$$

Positive values indicate net cancellation; negative values indicate reinforcement. We randomly attenuate a subset of projected contributions to 0.9 of their original value, leaving the others unchanged, and compare against equal-Euclidean-norm noise in the same subspace. Full sign flips provide a stronger intervention.

If learned contributions are coordinated, selectively shrinking them should disturb the model more than matched random noise. In G1 rank 64, lower-sensitivity attenuation adds 3.5×10^{-4} KL beyond the noise control (95% CI $[0.9, 7.3] \times 10^{-4}$). Full sign flips give a larger disruption. Define C_s on the high-sensitivity subspace and C_w on its lower-sensitivity complement. The interval for $C_w - C_s$ spans zero, giving a different outcome from the positive attenuation effect.

The attenuation test connects the organization of write contributions to downstream predictions. Its random control matches Euclidean norm within the same sensitivity subspace.

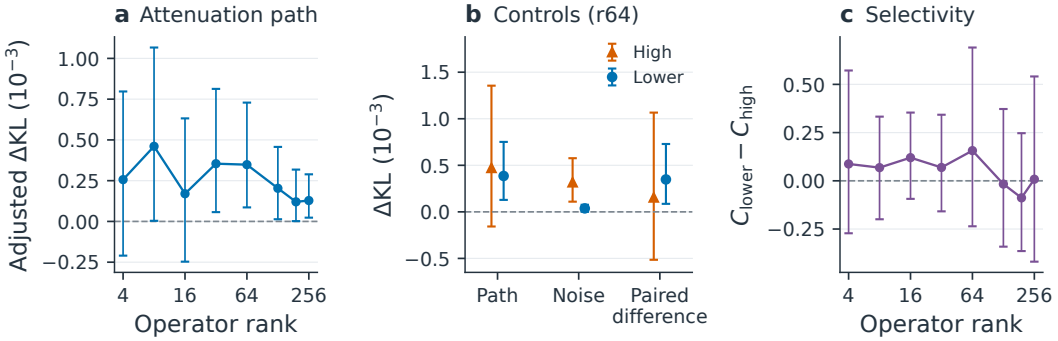


Figure 12: H3: contribution structure and downstream sensitivity. (a) Lower-sensitivity attenuation minus noise ΔKL. (b) High- and lower-sensitivity controls at rank 64. (c) Lower-minus-high sensitivity cancellation. Error bars are paired token-bootstrap 95% intervals.

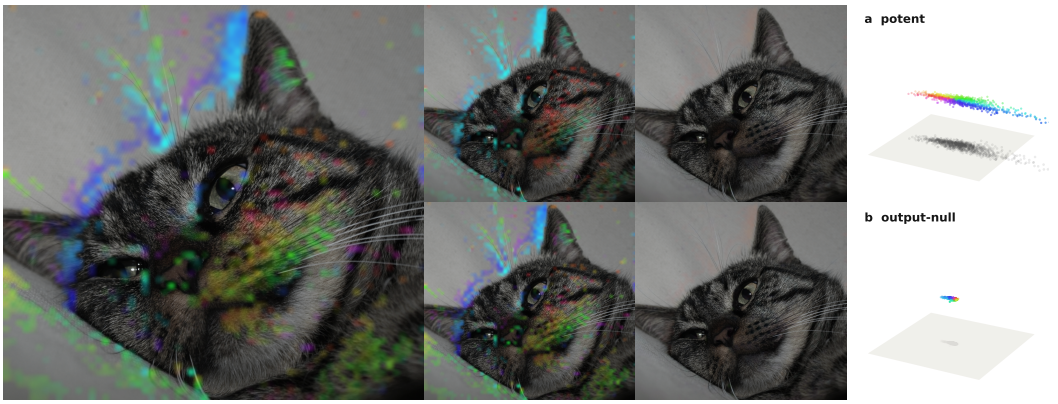


Figure 13: The same source path produces a strong response in active readout 45 and a near-null response in active readout 56, at a shared scale.

E Readout-Relative Near-Null Writeback and Directed Leakage

The same activation change can be detected by one downstream linear readout while producing almost no response in another. For $y = Rh$, the latter corresponds to $R\Delta h \approx 0$: the response depends on alignment between the change and the readout directions. Preparatory motor-cortex activity that cancels at the muscle readout provides an analogy for this readout selectivity (Kaufman et al., 2014).

In the DINOv3 layer-6-to-layer-7 replacement, we seek a natural path from an active source block i that strongly affects active readout a but barely affects another active readout b. Let $B_i = E_i^{(6)} \in \mathbb{R}^{d_6 \times p}$ be the frozen source-block basis and $D_k = (E_k^{(7)})^\top \in \mathbb{R}^{p \times d_7}$ the frozen target-block readout, with $p = 4$. The operator $W_j \in \mathbb{R}^{d_7 \times d_6}$ maps between the two layers. Its map through operator j is $A^{(j)}_{k \leftarrow i} = D_k W_j B_i$. We select paths on a selection split, solve interventions and fix visualization settings on an independent calibration split, and evaluate on 200 held-out images. Ablating the selected rank-32 path changes a by 14.38% and b by 0.44%, while b remains active. Figure 13 maps the same source coordinates into both readouts at a shared scale: the geometry is pronounced in a and nearly collapses in b. The display image is independent of all three splits.

We then preserve the write to a while deliberately increasing the write to b. With the learned read and write subspaces fixed, we solve $\Delta W_j = U_j H V_j^\top$ so that $D_a \Delta W_j B_i \approx 0$ and $D_b \Delta W_j B_i$ reaches a nonzero target. The pre-intervention near-null and post-intervention leakage criteria hold for 196 of 200 held-out images. An energy-matched control preserves both readouts and produces little leakage in b.

The path's effect depends on its alignment with the target readouts. Editing its write direction introduces leakage into b while preserving a, whereas the energy-matched control places its perturbation outside both measured readouts. This makes the geometry of writeback directly observable.

F Qwen Month and Number Experiments

We use month arithmetic and integer addition because prior work has identified their representational geometry and computational mechanisms (Engels et al., 2024; Kantamneni & Tegmark, 2025). MixCoder's architecture allows us to decompose MLP writes into shared, self-read, and cross-read components. We use this decomposition to study how cross-reads contribute to starting-value–offset interactions and how the resulting write directions change answer evidence in the original model.

F.1 Tasks, data, and measured site

We ask Qwen3-4B-Instruct-2507 to perform month shifts and integer addition, independently varying starting value n and offset k. The month task returns the month corresponding to $(n+k)$ mod 12, such as January two months after November; the number task returns $n+k$, such as $37+5=42$. Prior work identifies months with angular positions in a two-dimensional circular representation, where angular interventions change month answers (Engels et al., 2024). Integers have a generalized helical representation combining linear value and periodic sine/cosine components; the proposed "Clock" mechanism combines these components of the addends to construct and read out the sum representation (Kantamneni & Tegmark, 2025). We use these geometries to analyze conditional reads and writes.

We observe the model at the final token of the fixed answer prefix "The answer is.' ' At layer 34, we record the normalized MLP input x, the MLP write y, and the residual r_{in} immediately before the write. MixCoder predicts y from x; adding the write to the incoming residual gives the post-MLP state $r_{\text{in}}+y$.

The corpus contains 70,144 unique prompts from 32 templates per task. Each template covers all starting-value–offset pairs: months use $n,k\in\{0,\dots,11\}$ (January is $n=0$), while numbers use $n\in\{0,\dots,127\}$ and $k\in\{0,1,2,3,4,5,6,7,8,9,10,11,12,16,24,32\}$. The system instruction requests only the final month name or integer.

F.2 Proxy, readouts, and component definitions

A route-independent shared component G models the common write, leaving the private operators to express condition-dependent transformations.

The central decomposition is $\hat{y}=G+D+C$: G is the route-independent shared write, D is the part in which the operator selected by block j reads that same block, and C is the part in which it reads other blocks. The incoming residual $P=r_{\text{in}}$ is outside this write decomposition, so the predicted post-MLP state is $P+G+D+C$.

The proxy has 3,072 four-dimensional Stiefel blocks, Top-64 block-norm routing, one shared rank-16 operator, and one private rank-16 operator per condition block. With training-input mean μ, define $u_j=E_j^\top(x-\mu)$, the radial code $\zeta_j=\operatorname{asinh}(\|u_j\|)u_j/\|u_j\|$ (continuously extended at zero), and $s=\sum_h E_h\zeta_h$. The weight g_j is $\|u_j\|$ for a Top-64 block and zero otherwise. Then

$$G=b+W_0 s,\qquad D=\sum_j g_j W_j E_j \zeta_j,\qquad C=\sum_j g_j \sum_{h\neq j} W_j E_h \zeta_h.$$

Indices compare the block selecting an operator with the block it reads, not the identity of an output representation. Thus D/C is a matched-index/cross-index path decomposition inside the fitted proxy.

Figure grids. The month figure uses all 12×12 states in each of four test templates (576 states). The prespecified number figure uses the 64 integers with $n \bmod 4 \in \{0, 1\}$ and offsets $\{0, 1, 3, 7, 12, 24\}$ per template (1,536 states). Full-output number FVE instead uses all 8,192 held-out number rows.

(a) State progression. Figure 4a uses fixed axes within each task for $P+G$, $P+G+D$, and $P+G+D+C$, without double centering. The month view adds the input offset axis to the frozen month plane; the number view uses value and candidate period-52 coordinates. Translucent points cover test states, bold paths are four-template means at $k = 3$, gray dashed paths show the preceding stage, and crosses mark selected original-MLP endpoints. All stages are displayed directly in the same frozen readout coordinates.

(b) Month write interaction. Each template's projected month write is double-centered before averaging the four grids. Surfaces connect measured states; the dashed curve in the teacher panel shows the full proxy. The shared and self-read components are magnified 25× and 50×; cross-reads and teacher use the original scale.

(c) Number write vectors. At $k = 3$ and $n \leq 53$, rows show the small D vectors in 10^{-3} units, the state movement after adding C, and the original MLP's pre/post movement. Columns use value, candidate period-52/period-26, and parity coordinates; open-to-filled markers connect paired states. This panel is not double-centered.

F.3 Held-out interaction measurements

Geometric interaction is distinct from the D/C path partition: diagonal terms D can also capture geometric interactions across representation manifolds, because a block state and its routing weight can jointly depend on multiple input factors. Double centering below measures the starting-value–offset interaction captured by each component, allowing us to compare its distribution across D and C.

Let R denote the frozen linear readout and $Y_{n,k}$ a projected write grid. Within each template we remove the separate starting-value and offset effects by double centering:

$$\mathcal{I}(Y)_{n,k} = Y_{n,k} - \overline{Y}_{n,\cdot} - \overline{Y}_{\cdot,k} + \overline{Y}_{\cdot}.$$

For write component Q and original MLP write y, define

$$\mathrm{NMSE}(Q) = \frac{\|\mathcal{I}(RQ) - \mathcal{I}(Ry)\|_F^2}{\|\mathcal{I}(Ry)\|_F^2}, \qquad \mathrm{RMS}(Q) = \frac{\|\mathcal{I}(RQ)\|_F}{\|\mathcal{I}(Ry)\|_F}.$$

The Frobenius norms aggregate all four test templates, grid positions, and frozen readout dimensions. RMS ratios are not additive shares because components can reinforce or cancel one another.

Table 7: Held-out month–offset and number–offset interactions in frozen readouts. Rows evaluate components of the fitted layer-34 proxy.

Write term	Month NMSE ↓	Number NMSE ↓	Month RMS	Number RMS
Full write $G + D + C$	0.0219	0.0385	0.9385	0.9921
Cross-read C	0.0257	0.0417	0.9118	0.9433
Shared plus self-read $G + D$	0.9480	0.9057	0.0344	0.0573
Shared write G	0.9674	0.9864	0.0258	0.0244
Self-read D	0.9803	0.9188	0.0130	0.0476

Full-output FVE is 0.9134 for months and 0.6119 for numbers on the held-out data. The geometry at this layer is primarily carried into the MLP by the residual; the proxy models its incremental write. The write-to-post-state interaction power ratios are 0.0178 and 0.0501, computed as ratios of squared norms. Residual–write cross terms enter the post-state power.

On the frozen month plane, the pre-to-post change for offset +3 is a small correction to the existing month geometry.

F.4 Intervening on the original MLP

At layer 34, we use write differences to steer the original model's answers. We keep the starting month fixed and increase the offset by one: for example, moving from "five months after March" to "six months after March" targets September instead of August. For original and target MLP inputs x_r, x_d, we replace the original write with

$$y_r + \lambda[Q(x_d) - Q(x_r)],$$

where Q is the proxy write or a component. Two new templates provide 144 pairs; the model and proxy remain frozen.

The main test uses C from 16 conditions selected on discovery data by offset association and cross-read effects on interaction error. At unit dose, its write direction improves the target-versus-original logit margin by 0.75 relative to an equal-norm signed permutation (95% CI $[0.55, 0.95]$). This tests the information carried by the write direction at a matched perturbation norm.

An answer-preserving input $(n + 1, k - 1)$ provides an additional control after its write difference is norm-matched. Cross-read differences carry nearly all of the full proxy's steering effect at their learned amplitudes; shared and self-read differences have much smaller effects. This links the joint content–offset write to the direction that changes answer evidence.

At double dose, cross-read steering changes 12 of 144 answers to the target, compared with 2 for the answer-preserving control. Removing all active cross-reads changes correct answers from 106 to 105. Steering and removal thus reveal different aspects of the same layer: its write direction can shift answer evidence, while the remaining network largely preserves the original response after deletion.

G Related Work

Atomic features and feature-level circuits. A central goal of mechanistic interpretability is to decompose polysemantic model components into interpretable units and use circuits to describe their interactions (Olah et al., 2020; Gurnee et al., 2023; 2024). Early work directly analyzed neurons, attention heads, and other native components, using causal interventions to localize factual recall, identify circuits for specific behaviors, and automate circuit discovery (Meng et al., 2022; Wang et al., 2022; Conmy et al., 2023). Because native neurons are broadly polysemantic, later work turned to sparse autoencoders, which decompose dense activations into a small set of learned dictionary features and treat individual directions as finer-grained interpretable units (Cunningham et al., 2023; Bricken et al., 2023; Gao et al., 2024; Templeton et al., 2024). Yet SAEs trained on the same data and architecture can recover different feature sets, and an individual dictionary atom need not be a stable, irreducible semantic unit (Paulo & Belrose, 2025; Leask et al., 2025; Hindupur et al., 2025). Recent work therefore models compositional and computational relationships among features explicitly, including higher-order decoder interactions, computational sparsity in a feature basis, and interactions among crosscoder features (Koromilas et al., 2026; Farnik et al., 2025; Kamath et al., 2025). A complementary line moves from activation decomposition to functional proxies: transcoders use sparse hidden features to predict the output of an original model component, unfolding computation into how one feature influences subsequent features and supporting recent large-scale attribution graphs and circuit-tracing systems (Dunefsky et al., 2024; Ameisen et al., 2025; Kamath et al., 2025). Further work replaces atomic feature transitions with conditionally active linear or low-rank transformations to reduce fragmentation of shared computation and improve mechanistic fidelity (Lindsey et al., 2025; Oldfield et al., 2025). MOLT treats each selected transform as a computational object, separating it from the directional features used to interpret its inputs and outputs. Meanwhile, studies of feature consistency and circuit non-uniqueness

show that neither an individual feature nor a single sparse circuit necessarily defines a stable mechanistic coordinate system (Song et al., 2025; Méloux et al., 2025; Chen et al., 2026). MixCoder jointly models representation blocks, the conditions that select an operator, and the resulting transformation.

Manifold representations and geometric computation. Research in neuroscience and representation analysis has long shown that population activity often concentrates on structures with low intrinsic dimension, allowing cognitive variables to be encoded by the geometric state of a neural population rather than individual neurons (Chung et al., 2018; Chung & Abbott, 2021; Langdon et al., 2023). Similar phenomena increasingly appear in language models: some concepts are not carried by single directions, but form irreducible multidimensional structures, local low-dimensional regions, or continuous manifolds (Olah & Batson, 2023; Engels et al., 2024; Olah, 2024; Bhalla et al., 2026). This evidence has expanded interpretable representations from atomic directions to subspaces, blocks, and local geometries. Recent methods recover multidimensional concepts through subspace sparsity, block sparsity, mixtures of low-dimensional models, or local geometric decompositions, and show that a directional dictionary may split one continuous representation into several atoms or tile it only locally (Dalili & Mahdavi, 2026; Fel et al., 2026; Francel, 2026; Shafran et al., 2026; Bhalla et al., 2026). These geometries also participate in computation: language models count by manipulating low-dimensional manifolds and perform arithmetic through periodic or helical structure (Gurnee et al., 2025; Kantamneni & Tegmark, 2025). When the same structured operation recurs across representation states, describing computation only through edges between discrete features can unfold a shared transformation into many local rules; recent work therefore represents model updates directly as conditionally selected linear or low-rank operators (Lindsey et al., 2025; Oldfield et al., 2025). MixCoder combines low-dimensional representation states with sparse operator selection and explicit low-rank maps, whereas previous methods typically focus on either representations or transformations. Finally, prior work on superposition primarily asks how many representation directions coexist in a limited representational space and how their interference constrains capacity and scaling (Elhage et al., 2022; Liu et al., 2025). Our experiments examine how multiple transformations share writeback space.