Stop Probing, Start Coding: Why Linear Probes and Sparse Autoencoders Fail at Compositional Generalisation

Vitória Barin-Pacela [*1] **Shruti Joshi** [*1] **Isabela Camacho** [2] **Simon Lacoste-Julien** [1] **David Klindt** [3]

Abstract

The linear representation hypothesis states that neural network activations encode high-level concepts as linear mixtures. However, under superposition, this encoding is a projection from a higher-dimensional concept space into a lower-dimensional activation space, and a linear decision boundary in the concept space need not remain linear after projection. In this setting, classical sparse coding methods with per-sample iterative inference leverage compressed sensing guarantees to recover latent factors. Sparse autoencoders (SAEs), on the other hand, amortise sparse inference into a fixed encoder, introducing a systematic gap. We show this amortisation gap persists across training set sizes, latent dimensions, and sparsity levels, causing SAEs to fail under out-of-distribution (OOD) compositional shifts. Through controlled experiments that decompose the failure, we identify *dictionary learning*—not the inference procedure—as the limiting factor: SAE-learned dictionaries point in substantially wrong directions, and replacing the encoder with per-sample FISTA on the same dictionary does not close the gap. An oracle baseline proves the problem is solvable with a good dictionary at all scales tested. Our results reframe the SAE failure as a dictionary learning challenge, not an amortisation problem, and point to scalable dictionary learning as the key open problem for sparse inference under superposition.

Equal contribution; authors listed in alphabetical order. [1]Mila - Québec AI Institute & Université de Montréal [2]Santa Clara University [3]Cold Spring Harbor Laboratory. Correspondence to: Shruti Joshi <shrutijoshi98@gmail.com>, Vitória Barin-Pacela <vitoria.barin-pacela@mila.quebec>.

Accepted to the 2nd Workshop on Compositional Learning at ICML 2026, Seoul, South Korea. *Copyright 2026 by the author(s).*

1. Introduction

Understanding the internal representations of Large Language Models (LLMs) is crucial for their safe and reliable deployment. The *linear representation hypothesis* (LRH) is a foundational assumption in mechanistic interpretability, stating that a model's activations are linear mixtures of underlying concepts (Smolensky, 1990; Mikolov et al., 2013; Arora et al., 2016; Jiang et al., 2024; Park et al., 2024; Smith, 2024). It has motivated crucial progress on methods such as linear probing for concept discovery and activation steering (Turner et al., 2023; Chalnev et al., 2024).

To make this precise, let $\mathbf{z} \in \mathbb{R}^{d_z}$ denote ground-truth latent variables (concepts), and let $\mathbf{y} \in \mathbb{R}^{d_y}$ denote a model's activations. The LRH asserts that the relationship between concepts and activations is linear: $\mathbf{y} \approx \mathbf{W}\mathbf{z}$ for some matrix $\mathbf{W} \in \mathbb{R}^{d_y \times d_z}$. The goal of interpretability is to recover $\mathbf{z}$ from $\mathbf{y}$—i.e., to infer which concepts are active in an activation vector. When $d_z > d_y$, more concepts are encoded than there are activation dimensions, a regime known as *superposition* (Elhage et al., 2022). The system $\mathbf{y} = \mathbf{W}\mathbf{z}$ is underdetermined: each observation $\mathbf{y}$ is consistent with infinitely many $\mathbf{z}$, so one cannot simply learn a linear unmixing to map activations back to concepts. In other words, linear *representation*—concepts being linearly encoded in activations—is not the same as linear *accessibility*—that concepts are recoverable by a linear transformation. However this distinction is routinely overlooked and the LRH is conflated with the much stronger latter claim.

Concept recovery requires additional structure to resolve the underdetermination, such as by assuming sparsity, i.e., in practice, only $k \ll d_z$ concepts are active in any given input. Compressed sensing is precisely the framework that characterises when sparse signals can be recovered from such underdetermined measurements (Donoho & Elad, 2003; Donoho, 2006a). Under the sparsity assumption, classical results show that k-sparse codes can be recovered from $d_y = \mathcal{O}(k \log(d_z/k))$ suitably random measurements via nonlinear algorithms (e.g., basis pursuit, iterative thresholding). By contrast, requiring recovery to be *linear* (e.g., a linear probe) and allowing for an error $\epsilon > 0$, increases the required number of dimensions to $d_y = \Omega_\epsilon\left(\frac{k^2}{\log k} \log\left(\frac{d_z}{k}\right)\right)$

(Garg et al., 2026), with a quadratic complexity in k. Concretely, for $d_z = 10^6$ latent concepts with $k = 100$ active at once, nonlinear recovery requires only $d_y \approx 920$ dimensions while linear recovery requires $d_y \approx 20{,}000$—a typical transformer hidden size of 4096 comfortably exceeds the former but falls far short of the latter. Thus, whether concepts are recoverable from a fixed set of activations therefore depends not only on whether they are linearly encoded, but on *how* one attempts to decode them.

Additionally, with respect to downstream implications, the compression through $\mathbf{W}$ necessarily distorts the geometry of the latent space, so that a decision boundary that is linear in $\mathbf{z}$-space can become nonlinear in $\mathbf{y}$-space (Figure 1). Recovering $\mathbf{z}$ from $\mathbf{y}$ therefore requires nonlinear methods, but which nonlinear method matters. *Amortised* inference (Gregor & LeCun, 2010), as implemented by Sparse Autoencoders (SAEs) (Ng et al., 2011; Cunningham et al., 2023), learns a fixed encoder $r\colon \mathbb{R}^{d_y} \to \mathbb{R}^{d_h}$ that maps an activation to a sparse code in a single forward pass by training on a finite distribution of training data. *Per-sample inference*, by contrast, solves an optimisation problem from scratch for each input, using only the observation $\mathbf{y}$ and the dictionary $\mathbf{W}$, with no dependence on a training distribution.

Amortisation gap. The solution learned via amortised or per-sample inference can be evaluated by how accurately either recovers the true sparse code, e.g., whether the correct concepts are identified as active. In-distribution, both approaches may recover $\mathbf{z}$ well, and the discrepancy between them—the *amortisation gap* (Margossian & Blei, 2023; Kim & Pavlovic, 2021; Zhang et al., 2022; Cremer et al., 2018; Schott et al., 2021; Paiton et al., 2020; O'Neill et al., 2024)—can be small. Under distribution shift, however, the gap widens: when the sparsity pattern changes (e.g., novel combinations of concepts co-activate), the amortised encoder's recovery degrades because it was optimised for training-time statistics, while per-sample methods that optimise from scratch for each input remain unaffected. This paper studies the amortisation gap in the context of interpretability under superposition:

> *Under what conditions does sparse inference recover the true latent factors from superposed activations, and how does the choice of inference procedure—amortised, per-sample, or hybrid—affect robustness under distribution shift?*

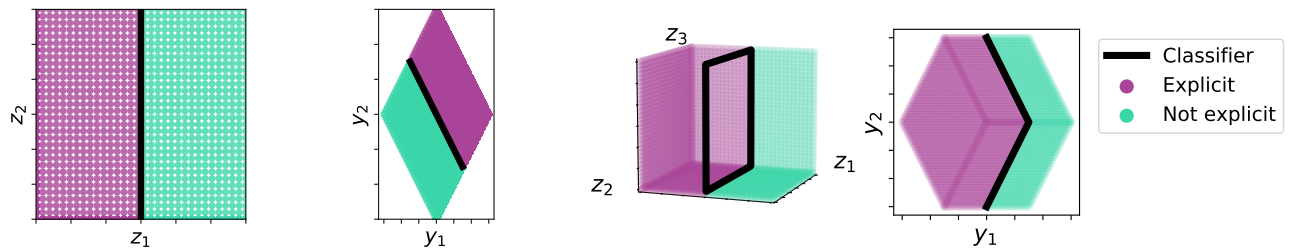
This argument adds nuance to recent studies suggesting that linear probes trained on top of LLM activations are superior to SAEs in out-of-distribution (OOD) binary classification tasks (Kantamneni et al., 2025). We argue that the recent OOD failures of SAEs are not an indictment of the superposition hypothesis, but rather a predictable consequence of replacing principled sparse inference with a brittle, amortised encoder. Instead of discarding the powerful framework of sparse coding, we embrace the geometric consequences of superposition and utilize methods equipped to handle the nonlinearity it induces.

Main contributions. In this work, we revisit classical sparse coding to address the central question. We show that under superposition, even labels that are linearly separable in latent space may become nonlinearly separable in activation space. We demonstrate that this is particularly pronounced in OOD settings, so that a perfect linear probe trained in-distribution will fail OOD (Figure 1). SAEs perform nonlinear inference, but amortising it into a fixed encoder introduces a systematic amortisation gap (Figure 5): the encoder fits the training distribution's co-occurrence structure and fails to generalise to novel combinations of latent factors under OOD composition shift (Figure 3). An oracle baseline—per-sample FISTA with the ground-truth dictionary—achieves near-perfect OOD recovery at all scales tested, proving the problem is solvable under compressed sensing theory (Section 4). Through controlled experiments that decompose the SAE failure, we identify dictionary learning—not the inference procedure—as the limiting factor. SAE-learned dictionaries point in substantially wrong directions, and replacing the encoder with per-sample FISTA on the same dictionary does not close the gap (Figure 10). Classical dictionary learning (DL-FISTA) produces better dictionaries at small scale, but both methods fail at dictionary learning when the latent dimension grows large. These results reframe the SAE failure: the bottleneck is not amortisation of inference but amortisation of dictionary learning, and the path forward requires scalable algorithms for learning dictionaries under the compressed-sensing framework.

2. Related Work

Compositional generalisation—the ability to understand and produce novel combinations of learned concepts—remains a fundamental challenge for neural networks (Fodor & Pylyshyn, 1988; Hupkes et al., 2020). Current approaches in causal representation learning attempt to achieve this through structural constraints, such as *additive* decoders (Lachapelle et al., 2023) (e.g., in SAEs), or specific training objectives like compositional risk minimisation (Mahajan et al., 2025). These works focus on obtaining guarantees for the compositional generalisation of disentangled models, while here, we evaluate the effect of compositional shifts under superposition.

SAEs have recently emerged as a primary tool for decomposing the internal activations of LLMs into, 'monosemantic', i.e., interpretable features (Cunningham et al., 2023). Despite their success in interpretability, their out-of-distribution (OOD) robustness is a growing concern. Recent



(a) *(b)*

Figure 1. Binary classification with $t = \mathbf{1}z_1 > 0.5$ (green: not explicit, purple: explicit). **(a)** When $d_z = d_y$, the linear decision boundary in latent space remains linear after mixing $\mathbf{y} = \mathbf{W}\mathbf{z}$. **(b)** When $d_z > d_y$ (overcompleteness) and z sparse, we can project down into non-overlapping regions (i.e., compressed sensing is possible), but the decision boundary becomes nonlinear in activation space, making linear probes insufficient.

evaluations suggest that SAEs trained on general datasets often fail to discover generalisable concepts across different domains or layers (Heindrich et al., 2025; Joshi et al., 2025), underperform compared to simple linear probes (Kantamneni et al., 2025), and remain brittle even when scaled (Gao et al., 2024).

Recent work has highlighted the limitations of SAEs in recovering true latent variables (O'Neill et al., 2024; Paulo & Belrose, 2025). This stands in contrast to the classical sparse coding framework (Olshausen & Field, 1996; Ranzato et al., 2007), which utilises iterative optimisation rather than a learned encoder to recover latent variables. This iterative approach provides stronger theoretical guarantees for the unique recovery of latents (Hillar & Sommer, 2015; Lewicki & Sejnowski, 1997; Gribonval et al., 2015). In contrast to O'Neill et al. (2024), who explore different in-distribution (ID) amortisation strategies, we focus our analysis on downstream tasks under OOD compositional shifts.

Lastly, literature on overcomplete independent component analysis (Podosinnikova et al., 2019; Wang & Seigal, 2024) explores identifiability where the number of latent variables exceeds the number of observed variables ($d_z > d_y$), though these models typically rely on statistical independence rather than sparsity.

3. Amortisation vs Pointwise Sparse Inference and Compositional Generalisation

We begin by specifying the data-generating process. Consider latent variable vectors $\mathbf{z} \in \mathbb{R}^n$ with at most k non-zero entries. A support set $S \subseteq [n]$ is drawn first to index these non-zero components following the process:

$$\begin{aligned} S \sim p_S, \quad \boldsymbol{z} \sim p(\boldsymbol{z} \mid S) \\ \operatorname{supp}(p_S) \subseteq \mathcal{S}_k := \{S \subseteq [n] : |S| \leq k\}, \\ \mathbb{P}(\mathbf{z}_{S^c} = \mathbf{0} \mid S) = 1. \end{aligned} \tag{1}$$

An unknown generative process g maps latents to data, producing $\mathbf{x} := g(\mathbf{z}) \in \mathbb{R}^{d_x}$. Although $\mathbf{z}$ is unobserved, we have access to learned representations $\mathbf{y} := f(\mathbf{x}) \in \mathbb{R}^{d_y}$ for some fixed (encoding) function $f : \mathbb{R}^{d_x} \to \mathbb{R}^{d_y}$ (e.g., an LLM activation map). Since the coordinates of $\mathbf{y}$ need not necessarily align with those of $\mathbf{z}$, we fit a representation model $r : \mathbb{R}^{d_y} \to \mathbb{R}^{d_h}$ estimating $\mathbf{h} := r(\mathbf{y})$ and another decoder $q : \mathbb{R}^{d_h} \to \mathbb{R}^{d_y}$ s.t. $\hat{\mathbf{y}} := q(\mathbf{h})$. Ideally, $\mathbf{h}$ serves as a proxy for $\mathbf{z}$ and when d_z is known, we can set $d_h = d_z$, while $d_y < d_z$. We refer to individual coordinates of $\mathbf{h}$ (one-dimensional subspaces of $\mathbb{R}^{d_h}$) as *features*, and we seek features that correspond (approximately) to the latent coordinates $\mathbf{z}$.

Typically, $\mathbf{h}$ is enforced to be sparse, in line with Equation (1). In practice, SAEs (Cunningham et al., 2023) are commonly used to learn the autoencoder $q \circ r$, where typically the decoder q is assumed to be linear and the encoder r is a single-linear layer followed by an activation function such as ReLU (Cunningham et al., 2023), JumpReLU (Rajamanoharan et al., 2024), or TopK (Gao et al., 2024; Costa et al., 2025). The assumption of a linear decoder is motivated by the LRH stating that the composed map $f \circ g$ is linear in the underlying latent variables s.t.,

$$\mathbf{y} \;=\; f(g(\mathbf{z})) \;\approx\; \mathbf{A}\mathbf{z} + \mathbf{b}, \qquad \mathbf{x} \sim p_{\mathbf{x}}, \tag{2}$$

for some matrix $\mathbf{A} \in \mathbb{R}^{d_y \times d_z}$ and offset $\mathbf{b} \in \mathbb{R}^{d_y}$, where $p_{\mathbf{x}}$ denotes the induced data distribution under the generative process $\mathbf{x} = g(\mathbf{z})$. A long line of work provides evidence for this hypothesis (c.f. Rumelhart & Abrahamson (1973); Hinton et al. (1986); Mikolov et al. (2013); Ravfogel et al. (2020); Klindt et al. (2025)). More recently, theoretical work justifies why linear properties could arise in these models (c.f. Arora et al. (2016); Jiang et al. (2024); Roeder et al. (2021); Marconato et al. (2024); Reizinger et al. (2024)).

Ensuring injectivity under overcompleteness. SAEs are often trained with an overcomplete feature dimension $d_h > d_y$, so that the decoder operates as a linear dictionary $\mathbf{W} \in \mathbb{R}^{d_h \times d_z}$. When $d_h > d_y$, the decoder dictionary

$\mathbf{W} \in \mathbb{R}^{d_y \times d_h}$ has more columns than rows, i.e. it is projecting from a high to a low-dimensional space, like projecting a 3D object onto its 2D shadow. This compression means there must exist nonzero codes $\mathbf{h}$ for which $\mathbf{Wh} = 0$ (i.e. $\ker(\mathbf{W}) \neq \{\mathbf{0}\}$ since $\dim\ker(\mathbf{W}) = d_h - \operatorname{rank}(\mathbf{W}) \geq d_h - d_y > 0$). Thus, information is inevitably lost, so multiple codes $\mathbf{h}$ can produce the same activation $\mathbf{y}$, resulting in its infinitely many possible reconstructions.

To obtain a unique solution, one typically restricts the code to be sparse, mirroring the latent sparsity assumption in Equation (1). Concretely, we restrict $\mathbf{h} \in \mathbb{R}^{d_h}$ to lie in a union of k-dimensional subspaces [1]:

$$\Sigma_k^{(d_h)} \;:=\; \{\mathbf{h} \in \mathbb{R}^{d_h} : \|\mathbf{h}\|_0 \leq k\},$$

This sparsity can be enforced softly (e.g., via an ℓ_1 penalty) or as a hard constraint through constrained optimisation (Ramirez et al., 2025; Gallego-Posada et al., 2025), or as implemented in TopK SAEs, which retain only the k largest-magnitude coordinates of $\mathbf{h}$ per input. Algebraically, sparsity replaces the unconstrained feasibility set $\{\mathbf{h} \in \mathbb{R}^{d_h} : \mathbf{Wh} = \mathbf{y}\}$ (typically infinite) with the constrained set $\{\mathbf{h} \in \Sigma_k^{(d_h)} : \mathbf{Wh} = \mathbf{y}\}$, which can be a singleton under suitable conditions on $\mathbf{W}$. In this sense, $r(\mathbf{y})$ is meaningful only insofar as it implements a consistent sparse selection of dictionary atoms among the many codes that reconstruct the same $\mathbf{y}$.

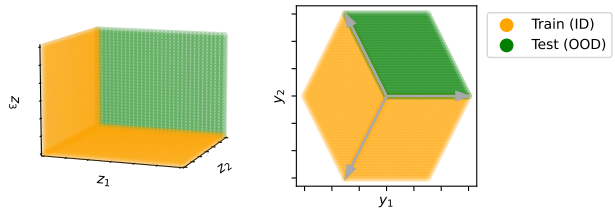


Figure 2. Compositional OOD split. *Left:* In-distribution (ID) training data covers support pairs (z_1, z_2) and (z_2, z_3) and the novel combination (z_1, z_3) is held out for OOD evaluation. *Right:* Same split in activation space $\mathbf{y} = \mathbf{Wz}$.

Identifiability of sparse codes h. Restricting the codes to be sparse is not sufficient to guarantee uniqueness—there may still exist distinct $\mathbf{h}, \mathbf{h}' \in \Sigma_k^{(d_h)}$ with $\mathbf{Wh} = \mathbf{Wh}'$. We need a property of the dictionary $\mathbf{W}$ ensuring that it does not collapse sparse codes onto each other so that they can be identifiable. A standard sufficient condition is the restricted isometry property (RIP) (Candes & Tao, 2006; Donoho, 2006b; Candès & Wakin, 2008): $\mathbf{W}$ satisfies RIP of order s with constant δ_s if $(1-\delta_s)\|\mathbf{h}\|_2^2 \leq \|\mathbf{Wh}\|_2^2 \leq (1+\delta_s)\|\mathbf{h}\|_2^2 \quad \forall \|\mathbf{h}\|_0 \leq s$. Intuitively, $\mathbf{W}$ approximately preserves the geometry of the sparse codes $\mathbf{h}$, neither inflating nor collapsing them beyond a tolerance. When we consider $\mathbf{h} \in \Sigma_k^{d_h}$, we want to ensure that two distinct k-sparse codes $\mathbf{h}$ and $\mathbf{h}'$ are distinguishable, i.e., their difference is not invisible to $\mathbf{W}$. Since, their difference can have at most $2k$ non-zeros, this is equivalent to asking that $\mathbf{W}$ maps no nonzero $2k$-sparse vector to zero, since if $\mathbf{Wh} = \mathbf{Wh}'$, then $\mathbf{W}(\mathbf{h} - \mathbf{h}') = 0$. RIP at order $s = 2k$ ensures exactly this. [2] So, under RIP, it is impossible for two different sparse vectors to map to the same output since their difference cannot be in the null space of $\mathbf{W}$, ensuring injectivity. With high probability, RIP is fulfilled for random Gaussian matrices projecting down into $d_y \geq \mathcal{O}(k \ln(\frac{d_h}{k}))$ dimensions (Candès & Wakin, 2008). For implications of this bound, please refer to Section D.

It is important to distinguish two levels of identifiability that arise in this setting. The first is *code-level identifiability*, i.e., given a fixed and known dictionary $\mathbf{W}$, when can we uniquely recover the sparse code $\mathbf{h}$ from an observation $\mathbf{y}$. This is precisely what RIP guarantees—it ensures that per-sample inference methods such as basis pursuit (Chen et al., 2001), or ISTA (Daubechies et al., 2004)/FISTA (Beck & Teboulle, 2009) converge to the unique k-sparse solution consistent with $\mathbf{y}$. The second is *dictionary identifiability*: can we recover the dictionary $\mathbf{W}$ itself from observations $\mathbf{y}$? Classical results show that $\mathbf{W}$ is identifiable when the data is sufficiently sparse and diverse (Hillar & Sommer, 2015; Gribonval et al., 2015), which is similar to a more flexible sufficient support variability condition on data for learning the dictionary (Joshi et al., 2025). These guarantees, however, are only as useful as the optimisation procedure that realises them.

Sparse Coding and Amortisation. Through point-wise inference, sparse coding infers a sparse code per input. Concretely, given samples $\{\mathbf{y}_i\}_{i=1}^p$ and fixed dictionary $\mathbf{W}$, the canonical *pointwise* formulation solves, for each i,

$$\mathbf{h}_i^\star \in \arg\min_{\mathbf{h} \in \mathbb{R}^{d_h}} \tfrac{1}{2}\|\mathbf{y}_i - \mathbf{Wh}\|_2^2 + \lambda\|\mathbf{h}\|_1,$$

(or equivalently, a hard sparsity constraint $\mathbf{h} \in \Sigma_k^{(d_h)}$). Algorithms such as Iterative Shrinkage-Thresholding Algorithm (ISTA) and FISTA compute $\mathbf{h}_i^\star$ by iterating a sequence of code estimates $\mathbf{h}_i^{(0)}, \mathbf{h}_i^{(1)}, \ldots$ for each input, typically starting from $\mathbf{h}_i^{(0)} = \mathbf{0}$ and applying repeated gradient-and-thresholding updates until convergence (Beck & Teboulle, 2009; Daubechies et al., 2004).

When the dictionary is unknown, inference is paired with

[1] equivalently requiring that $\operatorname{supp}(\mathbf{h}) \in \mathcal{S}_k^{(d_h)} := \{S \subseteq [d_h] : |S| \leq k\}$.

[2] A classical alternative is $\operatorname{spark}(\mathbf{W}) > 2k$ (spark of a matrix is the smallest number of its columns that are linearly dependent), no linear combination of $2k$ or fewer columns can sum to zero, i.e., $\ker(\mathbf{W}) \cap \Sigma_{2k}^{(d_h)} = \{\mathbf{0}\}$ and hence $\mathbf{W}$ is injective on $\Sigma_k^{(d_h)}$ (Donoho & Elad, 2003; Gribonval & Nielsen, 2004). But spark is typically intractable to compute.

dictionary learning (Olshausen & Field, 1996; Mairal et al., 2010), to alternate between (i) estimating $\{\mathbf{h}_i\}_{i=1}^p$ given the current $\mathbf{W}$ and (ii) updating $\mathbf{W}$ to minimise reconstruction error under a column-norm constraint. A standard formulation is, $\min_{\mathbf{W},\{\mathbf{h}_i\}_{i=1}^p} \sum_{i=1}^p \left(\frac{1}{2}\|\mathbf{y}_i - \mathbf{W}\mathbf{h}_i\|_2^2 + \lambda\|\mathbf{h}_i\|_1\right)$. s.t. $\|\mathbf{W}_{:,j}\|_2 \leq 1 \;\; \forall j \in [d_h]$. In contrast, amortised methods replace per-input iterative solving with a feed-forward encoder that predicts $\mathbf{h}$ in a single forward pass, learning to approximate the solution across a training distribution (Vafaii et al., 2024; 2025).

In the fully unsupervised setting, the general impossibility of unsupervised identifiabilty implies typical SAEs have been shown not to be identifiable (O'Neill et al., 2024). We hypothesise that this is one reason for the poor generalisation of SAEs OOD found in the literature (Kantamneni et al., 2025). In principle, a disentangled generalisation should allow for better OOD generalisation on downstream tasks (Schölkopf* et al., 2021)—but whether this promise holds in practice depends on the entire pipeline—from dictionary quality, through the inference procedure, to the downstream task. In this paper, we focus on this downstream question. We operationalise the central question from Section 1 by studying compositional distribution shifts (novel combinations of known concepts) which decompose it into two testable research questions:

i **Is sparse inference even necessary?** Linear probes outperforming SAEs on OOD tasks does not mean linear decoding is sufficient, it just means amortised inference is failing. Labels that are linearly separable in latent space become nonlinearly separable in activation space, and this nonlinearity is exposed precisely under compositional shifts. Rather than abandoning sparse coding for linear probes, we suggest the solution lies within the compressed-sensing framework.

ii **What is the bottleneck: inference or dictionary learning?** SAEs jointly learn a dictionary and an encoder. We decompose their failure to ask whether per-sample inference with an SAE-learned dictionary can close the gap, and if not, whether the dictionary itself (rather than the encoder) is the limiting factor.

Downstream tasks. We evaluate inferred sparse codes via a downstream binary prediction task, where the target t depends on the latent $\mathbf{z}$ through $t = u(\mathbf{z}) \in \{0, 1\}$; hence, predicting t from $\mathbf{h}$ through $\hat{t} = v(\mathbf{h})$ is a standard supervised proxy for testing whether $\mathbf{h}$ preserves task-relevant information about $\mathbf{z}$, such as by fitting a logistic head on $\mathbf{h}$ and evaluating it through the log-odds $\log \frac{\Pr(t=1|\mathbf{h})}{\Pr(t=0|\mathbf{h})} = \mathbf{a}^\top\mathbf{h} + a_0$ for weights $\mathbf{a}$.

Compositional generalisation. Train and test sets differ in which combinations of generative factors co-occur (Figure 2). In terms of supports $S \subseteq [n]$ of the sparse latent $\mathbf{z}$ (cf. Equation (1)), the ID data excludes a structured subset of support patterns while the OOD data concentrates on those withheld combinations. Achieving OOD success therefore requires the encoder to recover coordinates where the evidence for the label remains stable across different latent factor recombinations. Figure 3 illustrates a toy example example of this failure, where even fully supervised SAEs fail to reconstruct the OOD plane, while sparse coding reconstructs all latents.

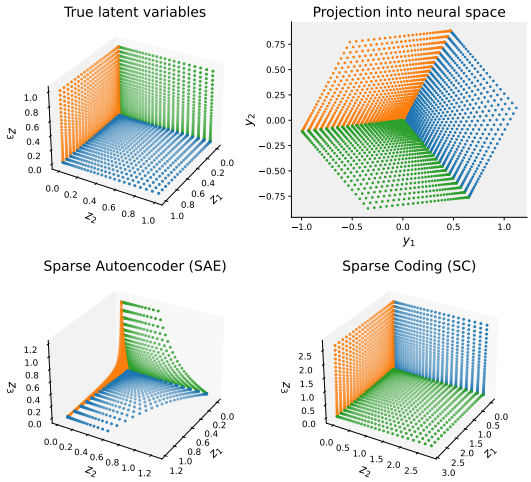


Figure 3. SAEs fail to recover latent variables under superposition, but sparse coding succeeds. **Top left:** Ground-truth latents ($d_z = 3$, $k = 2$); colors denote active-variable combinations. **Top right:** Activation space $\mathbf{y} = \mathbf{W}\mathbf{z}$ ($d_y = 2$); factors overlap after projection. **Bottom left:** SAE reconstruction; planes are not recovered. **Bottom right:** Sparse coding reconstruction; latents are identified up to scaling.

Linear Probing under the LRH. When $\mathbf{W}$ is injective ($d_z \leq d_y$), linear separability is invariant. However, under overcompleteness, even labels that are linearly separable in latent space can become nonlinearly separable in activation space (Figure 1). Figure 4 illustrates different failure cases of linear probes under this compositional setting.

4. Experiments

We investigate two questions: (i) is the OOD failure of SAEs a fundamental limitation of sparse inference under superposition, or a failure of specific components (dictionary, encoder, or both)? (ii) if the problem is solvable in principle, what is the bottleneck in practice? We use an oracle baseline—FISTA with the ground-truth dictionary—to establish an upper bound on what per-sample inference can achieve, then progressively decompose the gap between this oracle and SAEs. Methods span four families from fully per-sample to fully amortised inference, plus a linear-

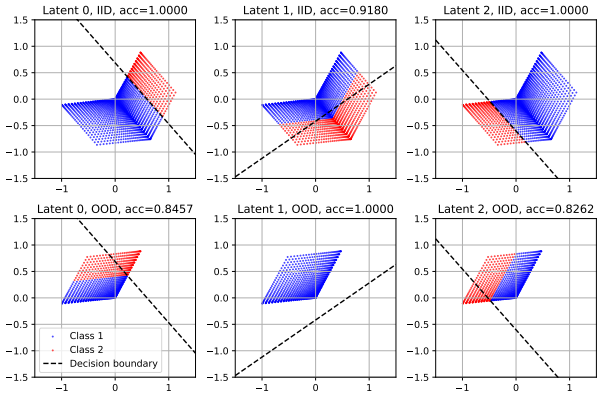


Figure 4. **Linear probes fail OOD under overcompleteness.** Each column sets $t = z_i$. The linear classifier fits the ID decision boundary well, but the compression $\mathbf{y} = \mathbf{A}\mathbf{z}$ introduces nonlinearity that is only exposed OOD, causing catastrophic generalisation failure (columns 1, 3) or, even, poor ID accuracy (column 2).

Table 1. Key ratios. The bound requires $\delta \geq C\,\rho \ln(1/\rho)$ for a constant $C > 0$: sparser codes (smaller ρ) and higher observation dimension (larger δ) make recovery easier.

Symbol	Definition	Interpretation
ρ	$\frac{k}{d_h}$	**Sparsity.** Fraction of non-zeros in $\mathbf{h}$.
δ	$\frac{d_y}{d_h}$	**Undersampling.** Ratio of observation dimension d_y to code dimension d_h. $\delta < 1$ is the overcomplete regime.

probe baseline, tested both in-distribution (ID) and out-of-distribution (OOD), where the OOD split withholds specific combinations of active latents during training (Figure 2).

We consider latent variables $\mathbf{z} \in [0,1]^{d_z}$ with at most k non-zero entries, each sampled uniformly on $[0,1]$ when active, observed through a linear mixing $\mathbf{y} = \mathbf{A}\mathbf{z}$, where $\mathbf{A} \in \mathbb{R}^{d_y \times d_z}$ with $d_y < d_z$. Details in Section A. Two ratios govern recovery difficulty (Table 1): sparsity $\rho = k/d_z$ and undersampling $\delta = d_y/d_h$. We evaluate these using the metrics below. The target $t = \mathbf{1}\{z_1 > 0.5\}$ is predicted from inferred codes $\mathbf{h}$.

Metrics. We report three quantities on both ID and OOD data (details in Section E.1). The *mean correlation coefficient* (*MCC*) (Hyvarinen & Morioka, 2016) evaluates identifiability s.t. MCC $= 1$ for a representation identified up to permutation and rescaling. *Accuracy* trains a logistic probe on the inferred codes $\mathbf{h}$ to predict the binary label t, applying the *same* supervised classifier to every method's codes, isolating the effect of the representation. *AUC* is computed per-feature without a trained classifier: for each code dimension, we use the raw activation as a score and compute ROC AUC; the single best feature on ID data is selected and its AUC is reported on both splits. AUC tests whether the label is isolated in an individual feature—a stronger condition than accuracy, which can exploit combinations of features. In the main text we report MCC (identifiability) and Accuracy (for fair downstream comparison between an unsupervised and a supervied learning method). AUC and additional metrics are in the appendix.

Methods. We compare methods spanning four families: *sparse coding* (per-sample ℓ_1 inference with oracle or learned dictionaries), *SAEs*, *frozen decoder* (per-sample FISTA on a frozen SAE-learned dictionary), and *refined hybrids* (FISTA warm-started from SAE codes). The last two families disentangle dictionary quality from inference: frozen-decoder methods replace only the encoder while reusing the SAE's dictionary, whereas refined methods additionally test whether the SAE's output provides a useful initialisation. A linear-probe baseline (or, a supervised skyline) operates directly on $\mathbf{y}$. Architectural and optimisation details are in C.1. All SAEs (ReLU (Cunningham et al., 2023), JumpReLU (Rajamanoharan et al., 2024), Top-K (Gao et al., 2024), MP (Costa et al., 2025)) share the same decoder dimension and are trained with identical optimiser settings, differing only in the activation function governing the encoder's sparsity mechanism.

4.1. The amortisation gap persists across undersampling ratios

Compressed sensing theory predicts a phase transition in sparse recovery (Candes & Tao, 2006): once the number of suitably random observations d_y exceeds $O(k \ln(d_z/k))$, per-sample ℓ_1 methods recover the latent code exactly. We sweep the undersampling ratio $\delta = d_y/d_h$ across a grid of latent dimensions $d_z \in \{50, 100, 200\}$ and sparsity levels $k \in \{3, 5, 10\}$, and report ID MCC (Figure 5). Phase transition with other metrics are reported in Section E. If SAEs solved the same sparse inference problem, they would exhibit the same transition and reach the same asymptotic performance. A persistent gap between the two would reveal the cost of amortising inference into a fixed encoder.

Per-sample methods exhibit the predicted phase transition. FISTA (oracle) transitions sharply to near-perfect MCC once δ passes a critical threshold, and the transition sharpens with d_z, matching the theoretical prediction. The empirical transition point is consistent with the constant $C \approx 2$ used in the compressed-sensing bound $\delta \geq C\rho \ln(1/\rho)$, validating the bound's practical relevance. DL-FISTA follows the same curve, shifted right by the cost of learning the dictionary. SAE variants also improve with δ—they are not insensitive to the undersampling ratio—but plateau at 0.2–0.5 MCC, well below the near-perfect recovery that per-sample methods achieve in the same regime. Crucially, the gap does not close at high δ, implying that even when the problem is well within the regime where compressed sensing guar-

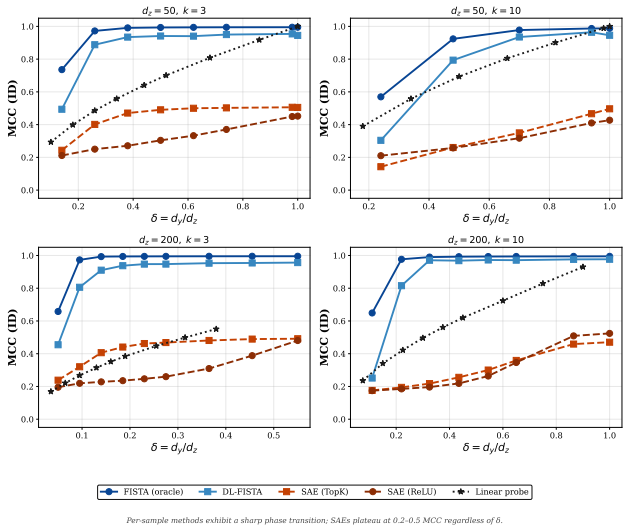


Figure 5. **The amortisation gap persists across undersampling ratios.** Per-sample methods (FISTA) exhibit a sharp phase transition to near-perfect MCC once δ exceeds the compressed-sensing threshold; SAEs plateau at 0.2–0.5 MCC regardless of δ. Each panel shows a different (d_z, k) combination.

antees exact recovery, the amortised encoder remains the bottleneck.

Takeaway. Both SAEs and sparse coding benefit from less aggressive undersampling (higher δ), but SAEs saturate far below per-sample methods.

4.2. Scaling up Latent Dimension Does not close the compositional gap

All unsupervised methods face a more challenging problem as d_z grows and the space of sparse patterns expands combinatorially. We hypothesise that per-sample methods should degrade more gracefully than SAEs if the bottleneck is amortised inference.

We sweep $d_z \in \{50, 100, 500, 1\text{K}, 5\text{K}, 10\text{K}\}$ with k and δ held fixed and report ID MCC (Figure 6) and OOD accuracy (Figure 7). As d_z grows, dictionary learning becomes harder for all unsupervised methods. We include FISTA with the ground-truth dictionary as an upper bound: it shows what is achievable with a perfect dictionary and isolates dictionary learning as the variable under test. FISTA (oracle) remains near-perfect (MCC ≥ 0.95, OOD accuracy ≈ 0.97), confirming the problem is solvable at all scales. DL-FISTA degrades in MCC as d_z grows—dropping to $\sim$0.3 by $d_z = 10\text{K}$—but maintains a small ID–OOD gap, indicating that its failure is dictionary quality, not compositional generalisation. When dictionary learning succeeds ($d_z \leq 500$), DL-FISTA dominates the linear probe on OOD accuracy (0.92 vs 0.83). SAE codes offer no consistent advantage over probing raw activations at any scale, consistent with Kantamneni et al. (2025). Varying sparsity k with d_z fixed yields the same pattern (appendix, Figure 16).

Takeaway. The oracle proves the problem is solvable at all scales—the OOD failure is not inherent to superposition. DL-FISTA beats linear probes when dictionary learning succeeds ($d_z \leq 500$) but falls behind at scale, confirming dictionary learning as the universal bottleneck. SAE codes offer no downstream advantage over raw activations, consistent with Kantamneni et al. (2025).

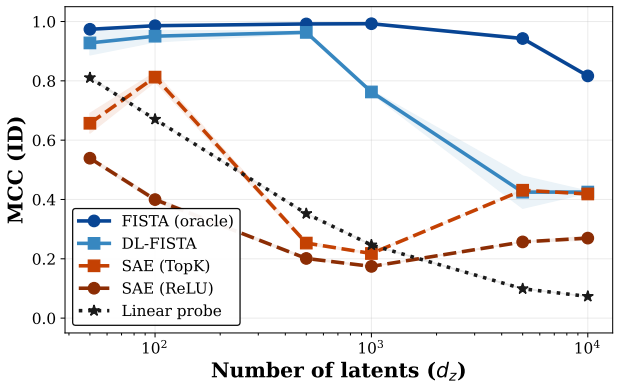


Figure 6. **Scaling latent dimension affects recovery.** DL-FISTA degrades due to dictionary learning difficulty but maintains a small ID–OOD gap. SAEs plateau at 0.2–0.5 MCC. The linear probe degrades sharply as superposition (compression) intensifies. $k = 10$, $p = 5000$, d_y follows the CS bound.

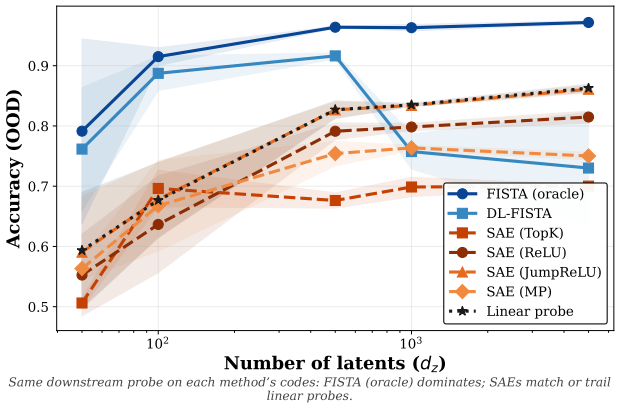


Figure 7. **DL-FISTA beats linear probes when dictionary learning succeeds but falls behind at scale.** OOD accuracy of a logistic probe trained on each method's codes. FISTA (oracle) dominates at all scales. SAE codes offer no advantage over probing raw activations. $k = 10$, $p = 5000$.

4.3. More Data Does Not Close the Amortisation Gap

One issue with the evaluation setup could be that SAEs are simply data-limited: with enough training samples, the encoder should learn to approximate per-sample inference. If this were the case, the gap between SAEs and sparse coding would shrink as the training set grows.

We vary the number of training samples $p \in \{10^2, \ldots, 10^5\}$ with all other parameters held fixed and report ID MCC (Figure 8) and OOD accuracy (Figure 9). FISTA (oracle) is constant by construction (it uses no training data beyond the oracle dictionary). DL-FISTA benefits substantially from more data—its MCC jumps from $\sim$0.5 at $p = 10^2$ to $\sim$0.98 by $p = 10^3$ and saturates—confirming that dictionary learning is genuinely sample-limited and that per-sample inference learns a better dictionary with enough data. SAE variants show a different pattern. SAE (TopK) and SAE (ReLU) improve only marginally, plateauing around 0.35–0.45 MCC. SAE (JumpReLU) *degrades* with more data, dropping from $\sim$0.45 to below 0.1, suggesting that additional training leads to a poor local solution rather than correcting it. The gap between per-sample and amortised methods is stable or widening across two orders of magnitude of additional data. On the downstream prediction task (Figure 9), DL-FISTA clearly separates from the linear probe once $p \geq 10^3$ (0.88 vs 0.68 accuracy), while all SAE variants trail the linear probe—this is the regime ($d_z = 100$) where dictionary learning works, and it translates directly into better OOD generalisation. OOD AUC shows the same pattern (appendix, Figure 20).

Takeaway. Additional training data closes the dictionary learning gap (benefiting DL-FISTA) but does not close the amortisation gap. When dictionary learning succeeds, DL-FISTA dominates linear probes on the same downstream task, whereas SAE codes trail linear probes regardless of number of training samples.

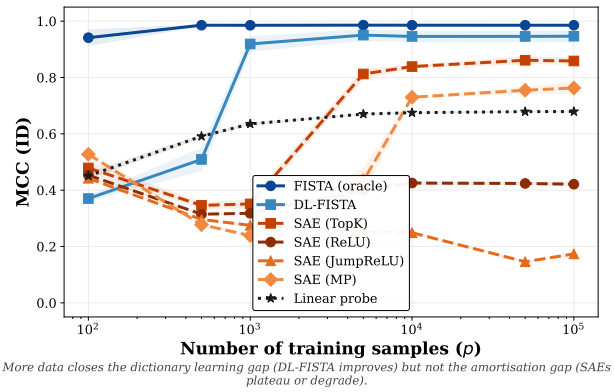


Figure 8. **More data closes the dictionary learning gap but not the amortisation gap.** DL-FISTA's MCC jumps from $\sim$0.5 to $\sim$0.98 by $p = 10^3$, confirming dictionary learning is sample-limited. SAEs plateau or degrade with more data, and the gap between per-sample and amortised methods is stable across two orders of magnitude. $d_z = 100$, $k = 10$, $d_y = 47$.

4.4. The Bottleneck is the Dictionary, Not the Encoder

The preceding experiments confound two potential sources of SAE failure: a poor dictionary and poor inference. We isolate these with controlled experiments that hold one component fixed while varying the other, finding that the SAE's dictionary provides a useful starting point for sparse coding while lagging in inference. Across SAEs and DL-FISTA the bottleneck that emerges is dictionary learning.

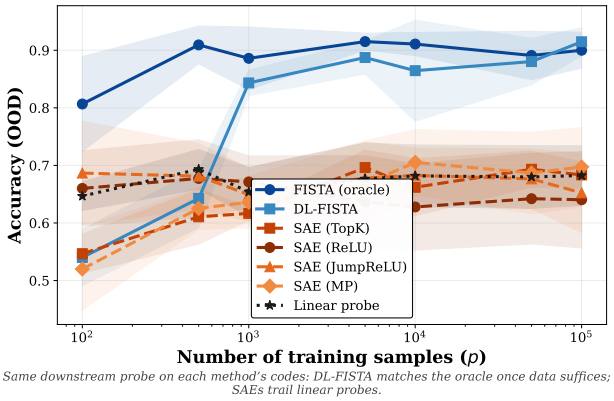


Figure 9. **More data closes the gap for DL-FISTA but not for SAEs.** OOD accuracy of a logistic probe on each method's codes. DL-FISTA matches FISTA (oracle) once $p \geq 10^3$ (0.88 vs 0.68 for the linear probe). All SAE variants trail the linear probe regardless of data. $d_z = 100$, $k = 10$, $d_y = 47$.

5. Conclusion

In this paper, we asked two questions: is sparse inference necessary under superposition, and if so, what is the bottleneck?

Sparse inference is necessary. Under superposition, concepts are linearly *represented* in activations but not linearly *accessible*—decision boundaries that are linear in latent space become nonlinear after projection. Linear probes, even with oracle access to ground-truth latents, degrade sharply in identifiability as superposition intensifies (MCC < 0.1 at $d_z = 10{,}000$). Per-sample FISTA with the ground-truth dictionary achieves near-perfect OOD recovery (MCC ≥ 0.83, accuracy ≥ 0.97) at all scales, proving the problem is solvable within the compressed-sensing framework.

The bottleneck is dictionary learning, not inference. SAE-learned decoder columns point in substantially wrong directions, and replacing the encoder with per-sample FISTA on the same dictionary does not help—the dictionary itself is the limiting factor. On the same downstream task, in our experiments, SAE codes offer no advantage over probing raw activations. Classical dictionary learning (DL-FISTA) dominates linear probes when it succeeds ($d_z \leq 1{,}000$). The gap between oracle and the best unsupervised method identifies **scalable dictionary learning** as the key open problem for sparse inference under superposition.

References

Arora, S., Li, Y., Liang, Y., Ma, T., and Risteski, A. A latent variable model approach to PMI-based word embeddings. *Transactions of the Association for Computational Linguistics*, 4:385–399, 2016. doi: 10.1162/tacl_a_00106. URL https://aclanthology.org/Q16-1028/.

Baraniuk, R., Davenport, M., DeVore, R., and Wakin, M. A simple proof of the restricted isometry property for random matrices. *Constructive approximation*, 28(3): 253–263, 2008.

Beck, A. and Teboulle, M. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences*, 2(1):183–202, 2009.

Bricken, T., Templeton, A., Batson, J., Chen, B., Jermyn, A., Conerly, T., Turner, N., Anil, C., Denison, C., and Askell, A. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, pp. 2, 2023. URL https://transformer-circuits.pub/2023/monosemantic-features.

Candes, E. J. and Tao, T. Near-optimal signal recovery from random projections: Universal encoding strategies? *IEEE Transactions on Information Theory*, 52(12):5406–5425, 2006. doi: 10.1109/TIT.2006.885507.

Candès, E. J. and Wakin, M. B. An introduction to compressive sampling. *IEEE signal processing magazine*, 25(2): 21–30, 2008.

Chalnev, S., Siu, M., and Conmy, A. Improving steering vectors by targeting sparse autoencoder features. *arXiv preprint arXiv:2411.02193*, 2024.

Chanin, D., Dulka, T., and Garriga-Alonso, A. Feature hedging: Correlated features break narrow sparse autoencoders. *arXiv preprint arXiv:2505.11756*, 2025.

Chen, S. S., Donoho, D. L., and Saunders, M. A. Atomic decomposition by basis pursuit. *SIAM review*, 43(1): 129–159, 2001.

Costa, V., Fel, T., Lubana, E. S., Tolooshams, B., and Ba, D. From flat to hierarchical: Extracting sparse representations with matching pursuit. *arXiv preprint arXiv:2506.03093*, 2025.

Cremer, C., Li, X., and Duvenaud, D. Inference suboptimality in variational autoencoders. In *International conference on machine learning*, pp. 1078–1086. PMLR, 2018.

Cunningham, H., Ewart, A., Riggs, L., Huben, R., and Sharkey, L. Sparse autoencoders find highly interpretable features in language models, 2023. URL https://arxiv.org/abs/2309.08600.

Daubechies, I., Defrise, M., and De Mol, C. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Communications on Pure and Applied Mathematics*, 57(11):1413–1457, 2004. doi: 10.1002/cpa.20042.

Donoho, D. L. Compressed sensing. *IEEE Transactions on information theory*, 52(4):1289–1306, 2006a.

Donoho, D. L. Compressed sensing. *IEEE Transactions on information theory*, 52(4):1289–1306, 2006b. Publisher: IEEE.

Donoho, D. L. and Elad, M. Optimally sparse representation in general (nonorthogonal) dictionaries via l1 minimization. *Proceedings of the National Academy of Sciences*, 100(5):2197–2202, 2003.

Elhage, N., Hume, T., Olsson, C., Schiefer, N., Henighan, T., Kravec, S., Hatfield-Dodds, Z., Lasenby, R., Drain, D., Chen, C., Grosse, R., McCandlish, S., Kaplan, J., Amodei, D., Wattenberg, M., and Olah, C. Toy Models of Superposition, September 2022. URL http://arxiv.org/abs/2209.10652. arXiv:2209.10652 [cs].

Fodor, J. A. and Pylyshyn, Z. W. Connectionism and cognitive architecture: A critical analysis. *Cognition*, 28(1-2): 3–71, 1988.

Gallego-Posada, J., Ramirez, J., Hashemizadeh, M., and Lacoste-Julien, S. Cooper: A Library for Constrained Optimization in Deep Learning. *arXiv preprint arXiv:2504.01212*, 2025.

Gao, L., la Tour, T. D., Tillman, H., Goh, G., Troll, R., Radford, A., Sutskever, I., Leike, J., and Wu, J. Scaling and evaluating sparse autoencoders, 2024. URL https://arxiv.org/abs/2406.04093.

Garg, N., Kleinberg, J., and Peng, K. How many features can a language model store under the linear representation hypothesis?, 2026. URL https://arxiv.org/abs/2602.11246.

Gregor, K. and LeCun, Y. Learning fast approximations of sparse coding. In *Proceedings of the 27th international conference on international conference on machine learning*, pp. 399–406, 2010.

Gribonval, R. and Nielsen, M. Sparse representations in unions of bases. *IEEE transactions on Information theory*, 49(12):3320–3325, 2004.

Gribonval, R., Jenatton, R., and Bach, F. Sparse and spurious: dictionary learning with noise and outliers, 2015. URL https://arxiv.org/abs/1407.5155.

Heindrich, L., Torr, P., Barez, F., and Thost, V. Do sparse autoencoders generalize? a case study of answerability, 2025. URL https://arxiv.org/abs/2502.19964.

Hillar, C. J. and Sommer, F. T. When can dictionary learning uniquely recover sparse data from subsamples? *IEEE Transactions on Information Theory*, 61(11):6290–6297, 2015. doi: 10.1109/TIT.2015.2460238.

Hinton, G. E. et al. Learning distributed representations of concepts. In *Proceedings of the eighth annual conference of the cognitive science society*, volume 1, pp. 12. Amherst, MA, 1986.

Hupkes, D., Dankers, V., Mul, M., and Bruni, E. Compositionality decomposed: How do neural networks generalise? *Journal of Artificial Intelligence Research*, 67: 757–795, 2020.

Hyvarinen, A. and Morioka, H. Unsupervised feature extraction by time-contrastive learning and nonlinear ica. *Advances in neural information processing systems*, 29, 2016.

Hyvärinen, A. and Pajunen, P. Nonlinear independent component analysis: Existence and uniqueness results. *Neural networks*, 12(3):429–439, 1999.

Jiang, Y., Rajendran, G., Ravikumar, P., Aragam, B., and Veitch, V. On the origins of linear representations in large language models, 2024. URL https://arxiv.org/abs/2403.03867.

Joshi, S., Dittadi, A., Lachapelle, S., and Sridhar, D. Identifiable steering via sparse autoencoding of multi-concept shifts, 2025. URL https://arxiv.org/abs/2502.12179.

Kantamneni, S., Engels, J., Rajamanoharan, S., Tegmark, M., and Nanda, N. Are sparse autoencoders useful? a case study in sparse probing. *arXiv preprint arXiv:2502.16681*, 2025.

Kim, M. and Pavlovic, V. Reducing the amortization gap in variational autoencoders: A bayesian random function approach. *arXiv preprint arXiv:2102.03151*, 2021.

Klindt, D., O'Neill, C., Reizinger, P., Maurer, H., and Miolane, N. From superposition to sparse codes: interpretable representations in neural networks. *arXiv preprint arXiv:2503.01824*, 2025.

Lachapelle, S., Mahajan, D., Mitliagkas, I., and Lacoste-Julien, S. Additive decoders for latent variables identification and cartesian-product extrapolation. In Oh, A., Naumann, T., Globerson, A., Saenko, K., Hardt, M., and Levine, S. (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 25112–25150. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/4ef594af0d9a519db8fb292452c461fa-Paper-Conference.pdf.

Lewicki, M. and Sejnowski, T. J. Learning nonlinear overcomplete representations for efficient coding. In Jordan, M., Kearns, M., and Solla, S. (eds.), *Advances in Neural Information Processing Systems*, volume 10. MIT Press, 1997. URL https://proceedings.neurips.cc/paper_files/paper/1997/file/489d0396e6826eb0c1e611d82ca8b215-Paper.pdf.

Locatello, F., Bauer, S., Lucic, M., Raetsch, G., Gelly, S., Schölkopf, B., and Bachem, O. Challenging common assumptions in the unsupervised learning of disentangled representations. In *international conference on machine learning*, pp. 4114–4124. PMLR, 2019.

Mahajan, D., Pezeshki, M., Arnal, C., Mitliagkas, I., Ahuja, K., and Vincent, P. Compositional risk minimization, 2025. URL https://arxiv.org/abs/2410.06303.

Mairal, J., Bach, F., Ponce, J., and Sapiro, G. Online learning for matrix factorization and sparse coding. *Journal of Machine Learning Research*, 11(1), 2010.

Mallat, S. G. and Zhang, Z. Matching pursuits with time-frequency dictionaries. *IEEE Transactions on signal processing*, 41(12):3397–3415, 1993.

Marconato, E., Lachapelle, S., Weichwald, S., and Gresele, L. All or none: Identifiable linear properties of next-token predictors in language modeling. *arXiv preprint arXiv:2410.23501*, 2024.

Margossian, C. C. and Blei, D. M. Amortized variational inference: When and why? *arXiv preprint arXiv:2307.11018*, 2023.

Mikolov, T., Yih, W.-t., and Zweig, G. Linguistic regularities in continuous space word representations. In Vanderwende, L., Daumé III, H., and Kirchhoff, K. (eds.), *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 746–751, Atlanta, Georgia, June 2013. Association for Computational Linguistics. URL https://aclanthology.org/N13-1090/.

Ng, A. et al. Sparse autoencoder. *CS294A Lecture notes*, 72 (2011):1–19, 2011.

Olshausen, B. A. and Field, D. J. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(6583):607–609, 1996. doi: 10.1038/381607a0.

O'Neill, C., Gumran, A., and Klindt, D. Compute optimal inference and provable amortisation gap in sparse autoencoders. *arXiv preprint arXiv:2411.13117*, 2024.

Paiton, D. M., Frye, C. G., Lundquist, S. Y., Bowen, J. D., Zarcone, R., and Olshausen, B. A. Selectivity and robustness of sparse coding networks. *Journal of vision*, 20(12): 10–10, 2020.

Park, K., Choe, Y. J., and Veitch, V. The linear representation hypothesis and the geometry of large language models, 2024. URL https://arxiv.org/abs/2311.03658.

Paulo, G. and Belrose, N. Sparse autoencoders trained on the same data learn different features. *arXiv preprint arXiv:2501.16615*, 2025.

Podosinnikova, A., Perry, A., Wein, A. S., Bach, F., d'Aspremont, A., and Sontag, D. Overcomplete independent component analysis via sdp. In Chaudhuri, K. and Sugiyama, M. (eds.), *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, volume 89 of *Proceedings of Machine Learning Research*, pp. 2583–2592. PMLR, 16–18 Apr 2019. URL https://proceedings.mlr.press/v89/podosinnikova19a.html.

Rajamanoharan, S., Lieberum, T., Sonnerat, N., Conmy, A., Varma, V., Kramár, J., and Nanda, N. Jumping ahead: Improving reconstruction fidelity with jumprelu sparse autoencoders, 2024. URL https://arxiv.org/abs/2407.14435.

Ramirez, J., Hashemizadeh, M., and Lacoste-Julien, S. Position: Adopt constraints over penalties in deep learning. *arXiv preprint arXiv:2505.20628*, 2025.

Ranzato, M., Boureau, Y.-L., and LeCun, Y. Sparse feature learning for deep belief networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 20, 2007. URL https://papers.nips.cc/paper_files/paper/2007/hash/2270a5fc66d369cd6c85026c045563b0-Abstract.html.

Ravfogel, S., Elazar, Y., Gonen, H., Twiton, M., and Goldberg, Y. Null it out: Guarding protected attributes by iterative nullspace projection. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J. (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 7237–7256, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.647. URL https://aclanthology.org/2020.acl-main.647.

Reizinger, P., Bizeul, A., Juhos, A., Vogt, J. E., Balestriero, R., Brendel, W., and Klindt, D. Cross-entropy is all you need to invert the data generating process. *arXiv preprint arXiv:2410.21869*, 2024.

Roeder, G., Metz, L., and Kingma, D. On linear identifiability of learned representations. In *International Conference on Machine Learning*, pp. 9030–9039. PMLR, 2021.

Rumelhart, D. E. and Abrahamson, A. A. A model for analogical reasoning. *Cognitive Psychology*, 5(1):1–28, 1973.

Schölkopf*, B., Locatello*, F., Bauer, S., Ke, N. R., Kalchbrenner, N., Goyal, A., and Bengio, Y. Toward causal representation learning. *Proceedings of the IEEE*, 109 (5):612–634, 2021. doi: 10.1109/JPROC.2021.3058954. URL https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=9363924. *equal contribution.

Schott, L., Von Kügelgen, J., Träuble, F., Gehler, P., Russell, C., Bethge, M., Schölkopf, B., Locatello, F., and Brendel, W. Visual representation learning does not generalize strongly within the same domain. *arXiv preprint arXiv:2107.08221*, 2021.

Smith, L. The 'strong' feature hypothesis could be wrong. AI Alignment Forum, August 2024. URL https://www.alignmentforum.org/posts/tojtPCCRpKLSHBdpn/the-strong-feature-hypothesis-could-be-wrong. [Accessed 02-26-2026].

Smolensky, P. Tensor product variable binding and the representation of symbolic structures in connectionist systems. *Artificial intelligence*, 46(1-2):159–216, 1990. URL https://www.sciencedirect.com/science/article/pii/000437029090007M. Publisher: Elsevier.

Templeton, A. *Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet*. Anthropic, 2024.

Tibshirani, R. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 58(1):267–288, 1996.

Turner, A. M., Thiergart, L., Leech, G., Udell, D., Vazquez, J. J., Mini, U., and MacDiarmid, M. Steering language models with activation engineering. *arXiv preprint arXiv:2308.10248*, 2023.

Vafaii, H., Galor, D., and Yates, J. Poisson variational autoencoder. *Advances in Neural Information Processing Systems*, 37:44871–44906, 2024.

Vafaii, H., Galor, D., and Yates, J. L. Brain-like variational inference. *ArXiv*, pp. arXiv–2410, 2025.

Wang, K. and Seigal, A. Identifiability of overcomplete independent component analysis, 2024. URL https://arxiv.org/abs/2401.14709.

Zhang, M., Hayes, P., and Barber, D. Generalization gap in amortized inference. *Advances in neural information processing systems*, 35:26777–26790, 2022.

A. Synthetic data details

We generate the synthetic data as follows. We generate a projection matrix $A \in \mathbb{R}^{m \times n}$ with $m < n$, where the elements of A are drawn from a standard Normal distribution, in agreement with the Restricted Isometry Property from compressed sensing. The rows of A are normalized to have unit norm. We generate the latent variables $z \in [0, 1]^n$ with k non-zero components, where the non-zero components are sampled uniformly from $[0, 1]$. The observed variables are then generated as $y = Az$.

When selecting which combinations of latent variables to be active, which k out of n, we consider the particular "out-of-variable" case for OOD generalizations, where some combinations of the variables are not available in the training data. The number of OOD variables is $n/2$. Then, we consider two possibilities:

- **ID data**: Divided into two cases:
  - The first latent variable is active and the other $k - 1$ are drawn between the variables of indices $[2, n/2]$.
  - The first latent variable is not active. The k active indices are drawn from the fool pool of indices $[2, n]$.
- **OOD data**: The first latent variable is active and the other $k-1$ are drawn between the variables of indices $[n/2+1, n]$.

The training set consists of ID data, and the test set consists of OOD data.

B. Training details

We implement the models in PyTorch and train them on a single NVIDIA A100 GPU.

C. Sparse Inference Methods for Interpretability

Sparse autoencoders (SAEs) have become the dominant tool for extracting interpretable features from neural network representations (Bricken et al., 2023; Cunningham et al., 2023). An SAE decomposes an activation $\mathbf{y} \in \mathbb{R}^{d_y}$ as $\mathbf{y} \approx \mathbf{D}\mathbf{h}$, where $\mathbf{D} \in \mathbb{R}^{d_h \times d_y}$ is an overcomplete dictionary ($d_h > d_y$) and $\mathbf{h} \in \mathbb{R}^{d_h}$ is a sparse code whose nonzero entries identify the active features. The quality of the interpretation depends entirely on the quality of $\mathbf{h}$: if the codes are wrong, the resulting feature attribution is wrong, regardless of reconstruction fidelity.

Standard SAEs compute codes in a single feedforward pass, $\mathbf{h} = \sigma(\mathbf{W}^\top(\mathbf{y} - b_{\text{pre}}) + b)$, where σ is ReLU or a top-k or JumpReLU activation. This is an *amortised* approximation to the sparse inference problem

$$\mathbf{h}^* = \arg\min_{\mathbf{h}} \ \tfrac{1}{2}\|\mathbf{y} - \mathbf{D}\mathbf{h}\|_2^2 + \lambda\|\mathbf{h}\|_1 \,, \tag{3}$$

which is the Lasso (Tibshirani, 1996), a convex problem with a unique solution (under mild conditions on $\mathbf{D}$). The amortisation gap $\mathbf{h} - \mathbf{h}^*$ is a structured error that is largest precisely when features are correlated or hierarchically organised (Costa et al., 2025; Chanin et al., 2025)—the regimes most relevant to real neural network representations.

Below we compare three inference strategies that move progressively closer to solving Equation (3): FISTA, LISTA, and Matching Pursuit. The comparison focuses on properties that matter for interpretability rather than reconstruction.

C.1. Algorithms

FISTA (Fast Iterative Shrinkage-Thresholding). FISTA (Beck & Teboulle, 2009) solves Equation (3) by alternating a gradient step on the reconstruction loss with the proximal operator for the ℓ_1 penalty (soft-thresholding S_λ), accelerated by Nesterov momentum. Let $\mathbf{h}^{(t)}$ denote the code estimate at iteration t and $\mathbf{q}^{(t)}$ a momentum-extrapolated point:

$$\mathbf{q}^{(t)} = \mathbf{h}^{(t)} + \frac{t_k - 1}{t_{k+1}}\big(\mathbf{h}^{(t)} - \mathbf{h}^{(t-1)}\big) \,, \tag{4}$$

$$\mathbf{h}^{(t+1)} = S_{\eta\lambda}\big(\mathbf{q}^{(t)} - \eta\,\mathbf{D}^\top(\mathbf{D}\,\mathbf{q}^{(t)} - \mathbf{y})\big) \,, \tag{5}$$

where $\eta \leq 1/\|\mathbf{D}^\top\mathbf{D}\|_{\text{op}}$ is the step size. Every iteration updates *all* d_h coefficients simultaneously. The support (which atoms are active) is fluid: a coefficient can be driven to zero by soft-thresholding at step t and revived at step $t' > t$.

Convergence to the global optimum is guaranteed at rate $O(1/t^2)$ (Beck & Teboulle, 2009). There are no learned parameters; the algorithm is fully determined by $\mathbf{D}$ and λ.

Practical note. Precomputing $\mathbf{W} = \mathbf{I} - \eta\,\mathbf{D}^\top\mathbf{D}$ and $\mathbf{b} = \eta\,\mathbf{D}^\top\mathbf{y}$ reduces each iteration to $\mathbf{h}^{(t+1)} = S_{\eta\lambda}(\mathbf{W}\,\mathbf{h}^{(t)} + \mathbf{b})$: a single matrix–vector multiply plus elementwise thresholding, both fully batchable on GPU.

LISTA (Learned ISTA). LISTA (Gregor & LeCun, 2010) takes the ISTA update (i.e. Equation (5) without momentum) and untethers its parameters from $\mathbf{D}$. Each layer t computes:

$$\mathbf{h}^{(t+1)} = S_{\theta_t}\big(\mathbf{W}_t\,\mathbf{h}^{(t)} + \mathbf{B}_t\,\mathbf{y}\big)\,, \tag{6}$$

where $\mathbf{W}_t \in \mathbb{R}^{d_h \times d_h}$, $\mathbf{B}_t \in \mathbb{R}^{d_h \times d_y}$, and $\theta_t \in \mathbb{R}^{d_h}$ are *free parameters learned by backpropagation*, independently at each layer. In ISTA, $\mathbf{W}_t = \mathbf{I} - \eta\,\mathbf{D}^\top\mathbf{D}$, $\mathbf{B}_t = \eta\,\mathbf{D}^\top$, and $\theta_t = \eta\lambda\,\mathbf{1}$ for all t; LISTA relaxes these constraints, allowing the network to learn iteration-dependent acceleration. Empirically, LISTA matches FISTA's solution quality in 10–20 layers rather than 100+ iterations (Gregor & LeCun, 2010).

Crucially, LISTA retains the structural properties of ISTA/FISTA: all coefficients are updated jointly at every layer, soft-thresholding provides a continuous sparsity mechanism, and the architecture is fully parallelisable across the batch dimension. The dictionary $\mathbf{D}$ (or its learned analogue in $\mathbf{W}_t, \mathbf{B}_t$) can be trained end-to-end.

MP-SAE (Matching Pursuit SAE). MP-SAE (Costa et al., 2025) unrolls the classical matching pursuit algorithm (Mallat & Zhang, 1993) into a differentiable encoder. Let $\mathbf{d}_j$ denote the j-th column of $\mathbf{D}$. At each step $t = 1, \ldots, T$:

$$j^{(t)} = \underset{j \in \{1,\ldots,d_h\}}{\arg\max}\ \mathbf{d}_j^\top \mathbf{r}^{(t-1)}\,, \tag{7}$$

$$h_{j^{(t)}} = \mathbf{d}_{j^{(t)}}^\top \mathbf{r}^{(t-1)}\,, \tag{8}$$

$$\mathbf{r}^{(t)} = \mathbf{r}^{(t-1)} - h_{j^{(t)}}\,\mathbf{d}_{j^{(t)}}\,, \tag{9}$$

where $\mathbf{r}^{(0)} = \mathbf{y} - b_{\text{pre}}$. One atom is selected per step; its coefficient is computed by projection onto the residual; the residual is updated by subtracting the selected atom's contribution. Previous coefficients are never revised. The dictionary is trained end-to-end via backpropagation through the unrolled steps.

MP-SAE approximately solves a different problem from Equation (3): it targets $\min_{\mathbf{h}} \|\mathbf{y} - \mathbf{D}\,\mathbf{h}\|_2^2$ subject to $\|\mathbf{h}\|_0 \le T$, which is NP-hard; matching pursuit is a greedy approximation with no global optimality guarantee.

C.2. Comparison on Interpretability-Relevant Axes

Well-posedness of codes. FISTA computes the unique minimiser of the Lasso objective Equation (3). The codes are *defined* by a convex optimisation problem: one can point to the objective and state precisely what the codes mean. LISTA approximates this same solution with learned acceleration. MP-SAE computes the output of a greedy procedure that does not correspond to the global minimum of any fixed objective; the codes depend on the selection order, which is itself a function of the dictionary geometry and the input. For identifiability — where "meaning" is invariance across the equivalence class of valid solutions — the distinction matters: the Lasso solution is unique and characterisable; the MP output is not.

Joint coefficient adjustment. FISTA and LISTA update all d_h coefficients at every iteration. If activating atom i changes the optimal coefficient for atom j (as occurs whenever $\mathbf{d}_i^\top\mathbf{d}_j \neq 0$), subsequent iterations correct for this. MP-SAE sets each coefficient once, at the step the atom is selected, and never revises it. Consider $\mathbf{y} = \alpha_1\mathbf{d}_1 + \alpha_2\mathbf{d}_2$ with $\mathbf{d}_1^\top\mathbf{d}_2 = \rho > 0$. MP selects $\mathbf{d}_1$ first (assuming $\alpha_1 > \alpha_2$) and assigns $h_1 = \mathbf{d}_1^\top\mathbf{y} = \alpha_1 + \alpha_2\rho$, which is inflated by $\mathbf{d}_2$'s contribution leaking through the correlation. The coefficient h_2 computed on the residual is correspondingly deflated. FISTA converges to the correct (α_1, α_2) because it jointly adjusts both coefficients across iterations.

Support dynamics. In FISTA/LISTA, the active set (support of $\mathbf{h}$) is fluid: an atom can be activated, deactivated, and reactivated across iterations as the algorithm converges. This self-correction is critical when the initial support estimate is wrong. In MP-SAE, the support grows monotonically — once an atom is selected, it remains active. There is no mechanism to deselect an incorrectly chosen atom, and the error propagates through all subsequent residuals.

Correlated and hierarchical features. Standard SAEs compute all inner products $\langle \mathbf{d}_j, \mathbf{y} \rangle$ simultaneously and threshold, making all activation decisions in parallel. This implicitly assumes quasi-orthogonality of the dictionary (Costa et al., 2025): if $\mathbf{d}_i$ and $\mathbf{d}_j$ are correlated, activating $\mathbf{d}_i$ should reduce the evidence for $\mathbf{d}_j$, but the one-shot encoder cannot express this.

MP-SAE fixes the conditioning problem via the residual update Equation (9): after selecting $\mathbf{d}_i$, atom $\mathbf{d}_j$ is evaluated against the residual $\mathbf{r}$ rather than the raw input, so correlated atoms no longer double-count shared variance. This is also why MP-SAE naturally recovers hierarchical structure: the first iteration selects the dominant (coarse) feature, and subsequent iterations select progressively finer features on the residual.

FISTA/LISTA handle correlated features correctly *and* with correct magnitudes, because the joint coefficient update avoids the inflation effect described above. However, they do not provide a natural ordering over features — all coefficients converge simultaneously rather than being produced in sequence. When a hierarchy readout is desired, the convergence order or coefficient magnitude in FISTA can serve as a proxy, but the sequential atom selection in MP provides this more directly.

Computational cost. Table 2 summarises the per-step and total cost for a batch of B samples. FISTA and LISTA are fully parallelisable across the batch; MP-SAE's sequential atom selection (the $\arg\max$ in Equation (7) depends on the previous step's residual) limits GPU utilisation. LISTA compensates for its per-step cost by converging in far fewer steps than FISTA.

Table 2. Computational comparison of sparse inference methods. d_y: input dimension, d_h: dictionary size, T: number of steps/layers, B: batch size.

	FISTA	**LISTA**	**MP-SAE**
Per-step cost	$O(B\,d_h^2)$	$O(B\,d_h^2)$	$O(B\,d_h)$
Typical steps T	100–300	10–20	k (active atoms)
GPU parallelism	Full	Full	Limited
End-to-end trainable	No[3]	Yes	Yes

Trainability. LISTA and MP-SAE are both end-to-end trainable: the dictionary is updated by backpropagation through the unrolled inference steps, using standard deep learning optimisers. FISTA requires alternating optimisation — an outer loop updating $\mathbf{D}$ and an inner loop running FISTA to convergence for each batch — which is slower but provides stronger guarantees on code optimality. A practical middle ground is to train the dictionary using a standard SAE or LISTA, then compute codes at evaluation time using FISTA with the learned dictionary, optionally warm-started from the encoder's output.

D. Implications

IMPLICATION: HOW MANY CONCEPTS CAN WE ENCODE?

The RIP bound $d_y \geq \mathcal{O}(k \ln(d_h/k))$ is *logarithmic* in the dictionary size d_h, meaning the number of latent concepts can be exponentially larger than the activation dimension. A typical value of a constant to realise the bound is 2, i.e. $d_y \geq 2(k \ln(d_h/k))$ (Baraniuk et al., 2008). With $d_y = 4{,}096$ and $k = 50$ active concepts, the bound permits dictionaries as large as $d_h \approx e^{d_y/2k} \sim e^{40}$—astronomically more concepts than dimensions. Even conservatively, a transformer hidden size of 4,096 can in principle support unique recovery over millions of latent concepts, provided only a few dozen are active at once. Thus, in the case of dictionaries with up to 16 million features on GPT-4 activations (Gao et al., 2024), or 34 million features with Claude 3 Sonnet (Templeton, 2024), $d_h = 10^6 - 10^7$ is still in the feasible range. The practical takeaway is that scaling up the dictionary is essentially free from the perspective of compressed sensing theory.

IMPLICATION: HOW TO OBTAIN SPARSE CODES?

In the fully unsupervised setting assumed while training SAEs, there is no guarantee that the underlying latent codes are sparse enough for recovery. This connects to a broader impossibility: Hyvärinen & Pajunen (1999); Locatello et al. (2019) show that fully unsupervised disentanglement is impossible without inductive biases on the model or the data. In

[3] FISTA itself is not trained; the dictionary is updated in a separate alternating minimisation step. However, FISTA can be used at evaluation time with a dictionary trained by any method, including an SAE.

the compressed-sensing framing, the required inductive bias is precisely sufficient sparsity: the data must be generated by activations involving few enough concepts at once. Joshi et al. (2025) propose to leverage concept shifts being sparser to effectively automate the creation of sufficiently sparse and diverse observations from model activations.

IMPLICATION: HOW SPARSE MUST THE CODES BE?

k enters the RIP bound roughly linearly, denoting the number of interpretable concepts simultaneously encoded in the activation vector. It is unclear a priori what the expected value of k would be across different activations. Rearranging the bound gives the maximum feasible sparsity: $k_{\max} \approx d_y / \ln(d_h/k)$. For $d_y = 4{,}096$ and $d_h = 10^7$, this yields $k_{\max} \approx 185$, depending on the constant. Beyond this, no inference procedure—per-sample or amortised—can guarantee unique recovery. The choice of k in training SAEs therefore has implications beyond the reconstruction–sparsity trade-off as it determines whether the problem is even theoretically solvable (see Klindt et al., 2025, Fig.4).

E. Experimental results

E.1. Metric definitions and what each diagnostic isolates

We use three levels of evaluation metrics, each isolating a different component of the SAE pipeline. Table 3 summarises the distinctions.

Table 3. Summary of evaluation metrics and the questions they answer.

Metric	Operates on	Question it answers
MCC	Codes vs $\mathbf{z}$ (samples)	Do learned code dimensions track the same variation as the true latents, up to permutation and rescaling?
Accuracy	Probe on codes vs labels	Does a logistic classifier on the codes predict the label OOD? (Same classifier for all methods—fair comparison.)
Per-feature AUC	Codes vs labels (samples)	Does a single code dimension separate the binary label?
Cosine similarity	Decoder $\hat{\mathbf{W}}$ vs $\mathbf{A}$ (weights)	Do the dictionary atoms point in the same directions as the ground-truth columns?
Norm ratio	Decoder $\hat{\mathbf{W}}$ vs $\mathbf{A}$ (weights)	Are the dictionary atoms the correct magnitude?
Support precision	Codes vs $\mathbf{z}$ (per-sample binary)	Of the features the encoder activates, how many are truly active?
Support recall	Codes vs $\mathbf{z}$ (per-sample binary)	Of the truly active features, how many does the encoder find?

Accuracy vs AUC: fairness considerations. The linear-probe baseline uses supervised ridge regression from $\mathbf{y}$ to the ground-truth $\mathbf{z}$, producing d_z-dimensional codes where each dimension is explicitly trained to track one latent. Per-feature AUC directly benefits from this alignment: each output dimension is already optimised to separate its target, giving the linear probe an inherent advantage over unsupervised methods whose code dimensions need not correspond to individual latents. Accuracy—training a logistic probe on each method's inferred codes—provides a fairer comparison because it applies the same downstream classifier to all methods and can exploit feature combinations, not just individual dimensions. In the main text we therefore report accuracy for downstream comparisons and reserve AUC for the appendix.

MCC (end-to-end). The mean correlation coefficient (Hyvarinen & Morioka, 2016) computes the Pearson correlation between each code column and each ground-truth latent column across samples, then finds the best one-to-one matching via the Hungarian algorithm. It measures overall recovery quality: MCC $= 1$ when codes reproduce the true latents up to permutation and rescaling. Pearson correlation is invariant to linear scaling, so MCC does not penalise magnitude differences. However, MCC conflates dictionary quality and encoder quality—a low MCC does not tell you whether the dictionary atoms are wrong or the encoder is selecting the wrong atoms.

Dictionary diagnostics (model-level). Cosine similarity and norm ratio between matched decoder columns and ground-truth dictionary columns isolate *dictionary quality* independent of any test data or encoder. If cosine is high but MCC is low, the dictionary is good but the encoder fails. If cosine is low, the dictionary itself is the bottleneck regardless of the encoder. In our experiments, norm ratios are ≈ 1.0 throughout (column magnitudes are correct) while cosine similarity varies widely across SAE types (0.33–0.93), pinpointing the error as directional.

Support diagnostics (encoder-level). Support precision and recall compare the binary nonzero pattern of codes against ground truth (after Hungarian-matching via the decoder columns). These isolate *encoder quality* on the feature-selection task: does the encoder activate the right atoms, independent of magnitude accuracy? MCC cannot distinguish "activated

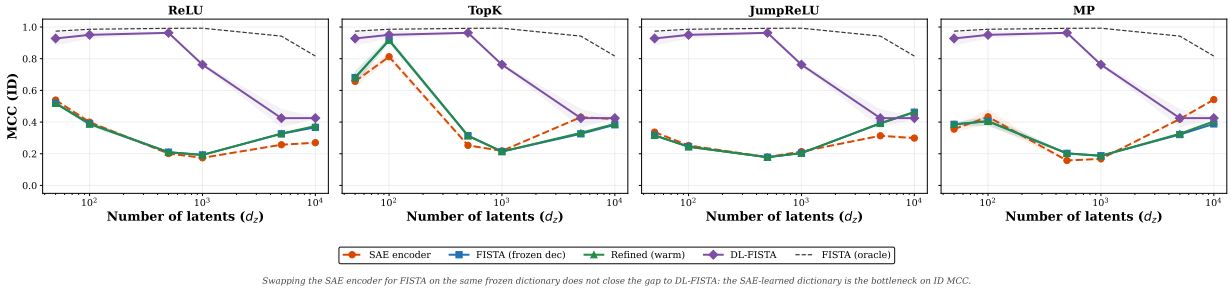


Figure 10. **The SAE-learned dictionary is the bottleneck, not the encoder.** Each panel shows one SAE type. Swapping the SAE encoder for FISTA on the same frozen dictionary (blue) or warm-starting from SAE codes (green) does not close the gap to DL-FISTA (purple), which learns its own dictionary. The oracle (gray dashed) confirms the problem is solvable. $k = 10$, $p = 5000$.

the wrong 10 features" from "activated all 100 features and relied on magnitude differences"—the support diagnostics can. In our experiments, ReLU and JumpReLU activate $\sim$90% of all features (precision $\sim$0.1), revealing that they are not performing sparse selection at all.

Why all three levels are needed. Consider two failure modes that produce the same MCC:

1. An SAE with correct dictionary directions but an encoder that activates the wrong atoms. Dictionary cosine would be high; support precision would be low.
2. An SAE with wrong dictionary directions but an encoder that compensates by routing information through correlated atoms. Dictionary cosine would be low; support precision could be moderate.

MCC alone cannot distinguish these. The layered diagnostics—dictionary geometry (cosine) $\rightarrow$ feature selection (support) $\rightarrow$ overall recovery (MCC)—tell you *where* in the pipeline things break.

E.2. The Dictionary is the Bottleneck

Swapping the encoder does not close the gap. Given a trained SAE with decoder $\hat{\mathbf{W}}$ and bias $\hat{\mathbf{b}}$, we compare three inference strategies on the same test input $\mathbf{y}$:

1. **SAE encoder** (baseline): a single forward pass, $\mathbf{h} = r(\mathbf{y})$.
2. **Frozen decoder**: FISTA on $\mathbf{y} - \hat{\mathbf{b}}$ using $\hat{\mathbf{W}}$ as a fixed dictionary, initialised from $\mathbf{h}^{(0)} = \mathbf{0}$.
3. **Refined** (warm-start): FISTA on the same frozen $\hat{\mathbf{W}}$, initialised from $\mathbf{h}^{(0)} = r(\mathbf{y})$.

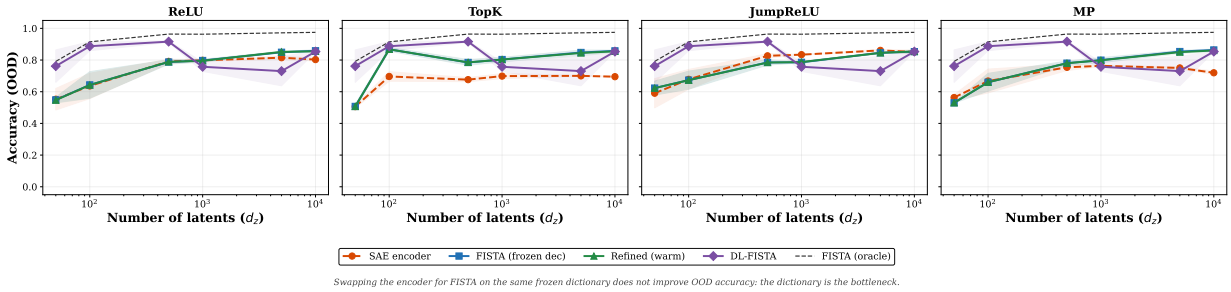


Figure 11. **Swapping the encoder does not improve OOD accuracy: the dictionary is the bottleneck.** Same layout as Figure 10. Replacing the SAE encoder with FISTA on the same frozen dictionary yields no consistent accuracy gain. The gap to DL-FISTA confirms that the dictionary directions, not the inference procedure, limit OOD performance. $k = 10$, $p = 5000$.

All three use the *same* dictionary—only the inference procedure differs. Figure 10 shows the comparison alongside DL-FISTA (which learns its own dictionary independently). The result shows that SAE encoders $\approx$ frozen FISTA $\approx$ refined ($\sim$ 0.15–0.4 MCC), while DL-FISTA (which learns its own dictionary via classical alternating minimisation) achieves an MCC of $\sim$ 0.75–0.95. This implies that swapping the encoder for per-sample inference on the same dictionary does

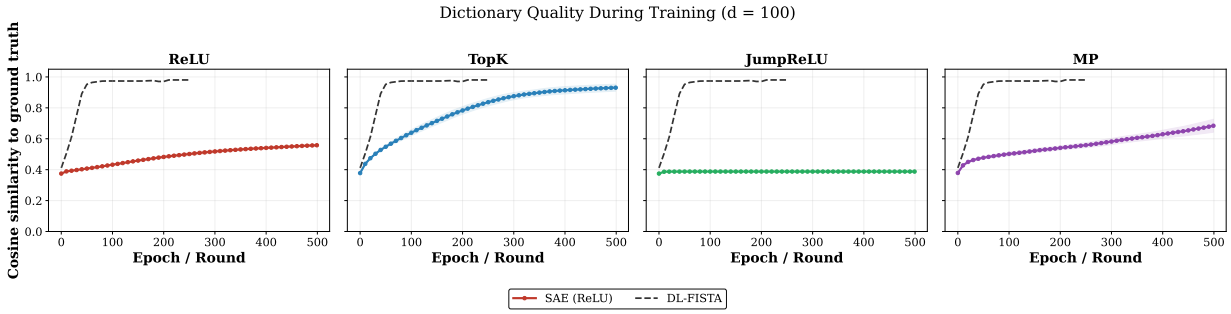


Figure 12. **DL-FISTA learns correct dictionary directions; SAEs do not.** SAE decoder cosine to ground truth vs training epoch (coloured) and DL-FISTA dictionary cosine vs outer round (gray dashed) at $d_z = 100$. At large d_z (≥ 5000), neither method converges (Figure 36).

not meaningfully improve recovery. The SAE-learned dictionary itself is the limiting factor. This conclusion is robust to the choice of FISTA regularisation strength λ: sweeping λ across three orders of magnitude yields at most modest improvements over the SAE encoder, far below the oracle at every λ (appendix, Section E.8). Note that DL-FISTA also degrades at large d_z (to $\sim$0.3 at $d_z = 10$K, Section 4.2), reflecting the inherent difficulty of dictionary learning at scale—but it degrades uniformly across ID and OOD, unlike SAEs which exhibit a compositional generalisation failure. The linear probe, operating directly on $\mathbf{y}$ without sparse inference, degrades in MCC even faster as superposition intensifies (Figure 6), but maintains surprisingly high downstream accuracy (Figure 7) because it is supervised for that specific task—a divergence that illustrates how single-task metrics can mask identifiability failure.

At 100 FISTA iterations (sufficient for convergence of the convex Lasso objective) warm-starting from SAE codes yields the same solution as cold-starting from zeros, confirming that the per-sample optimisation converges regardless of initialisation. The one exception is TopK, whose SAE decoder achieves $\sim$0.87 OOD MCC on its own (comparable to DL-FISTA at small d_z), consistent with TopK's structurally enforced sparsity producing a more faithful dictionary.

The dictionary fails because the columns point in the wrong directions. We track cosine similarity between decoder columns and ground truth during training (Figure 12) and decompose column-level errors at convergence (Figure 28). At $d_z = 100$, DL-FISTA converges to high cosine (> 0.9) while SAEs plateau earlier—the SAE optimisation landscape is harder even when the problem is tractable for alternating minimisation. At $d_z = 5000$, neither method finds the right directions (cosine $\sim$0.2–0.35), implying that the SAE dictionaries do not overfit or drift, they simply never converge. Norm ratios remain $\approx$1.0 throughout, and re-normalising columns or substituting oracle norms does not improve MCC (Figure 27). TopK is an outlier at small d_z (cosine 0.93, 90% of columns close to ground truth at $d_z = 100$), but this advantage vanishes at scale.

Why the encoder fails: wrong support. We measure support recovery—whether the SAE activates the correct features—by comparing the binary nonzero pattern of SAE codes against the ground truth (after Hungarian matching). At $d_z = 5000$ with true sparsity $k = 10$: ReLU activates $\sim$120 of 5000 features (precision 0.009); JumpReLU activates $\sim$300 (precision 0.005). Even TopK, which activates the correct number of features ($\sim$10), selects almost entirely wrong atoms (precision 0.03). Re-estimating magnitudes via least-squares on the SAE's support makes things *worse* (Figure 56), confirming the support itself is incorrect. At smaller d_z, TopK's support precision is substantially better (0.49 at $d_z = 100$; Figure 29), explaining its stronger performance at small scale.

Takeaway. The gap between oracle and other methods seems to be a dictionary learning problem. SAEs fail at both dictionary learning and inference, but the dictionary is the limiting factor: per-sample inference cannot rescue wrong column directions. At scale ($d_z = 5000$), DL-FISTA also fails at dictionary learning (cosine $\sim$0.25), converging to similar MCC as SAEs.

E.2.1. The SAE Dictionary is a Useful Starting Point

Although the SAE decoder is a poor standalone dictionary, it may still encode useful structure that accelerates dictionary learning. We test this by using the SAE decoder as the initial dictionary for DL-FISTA (unsupervised alternating optimisation), comparing against DL-FISTA initialised from a random dictionary.

Figure 13 shows convergence curves at $d_z = 100$, sweeping the number of dictionary-update rounds. The SAE decoder

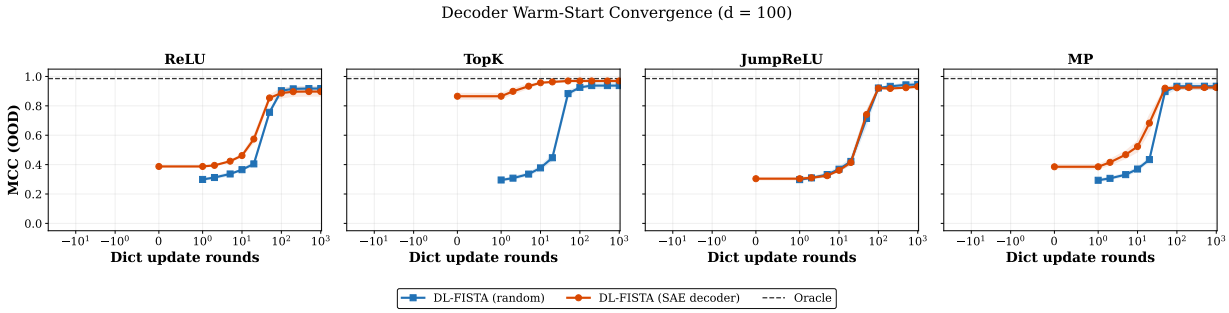


Figure 13. **The SAE decoder is a useful warm-start for dictionary learning.** Orange: DL-FISTA initialised from SAE decoder. Blue: DL-FISTA from random dictionary. Gray dashed: oracle. The SAE decoder provides a clear head start, especially for TopK, but both initialisations converge to the same optimum. $d_z = 5000$, $k = 10$. Results at other d_z in appendix.

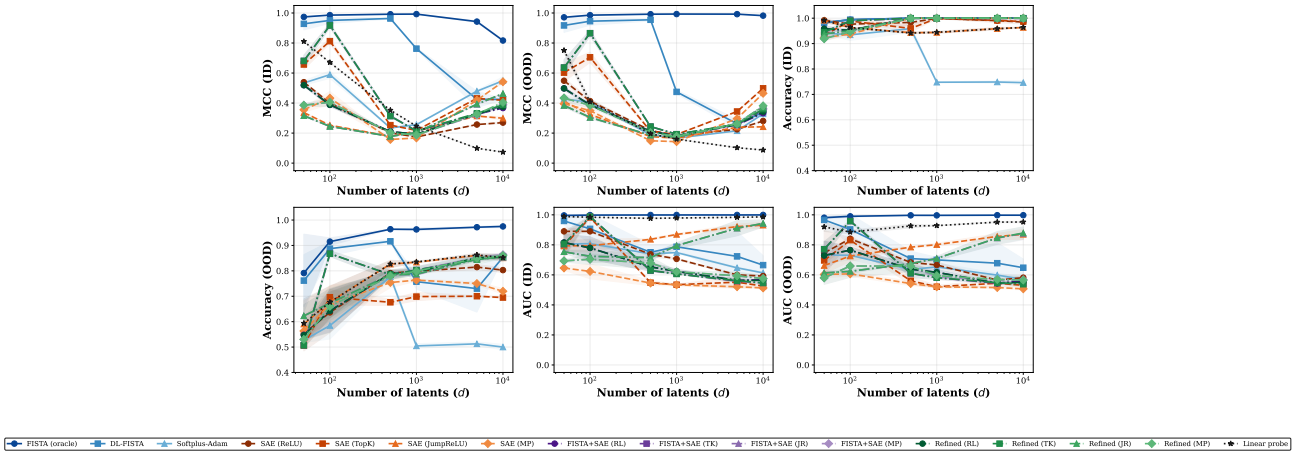


Figure 14. **All metrics vs number of latent variables.** Per-sample methods (blue) degrade uniformly across ID and OOD as d_z grows. SAEs (orange, dashed) show a persistent ID–OOD gap across all metrics. Frozen-decoder and refined hybrids (purple, green) close part of the gap by swapping inference. $k = 10$, $p = 5000$.

provides a clear head start: for TopK, warm-starting begins at OOD MCC 0.87 and reaches 0.94 within 5 rounds, while random initialisation starts at 0.30 and requires $\sim$50 rounds to reach the same level. For ReLU and JumpReLU, the advantage is smaller but consistent—the SAE decoder saves $\sim$20–50 rounds of dictionary learning. Both initialisations converge to the same final MCC, confirming that the SAE decoder biases the optimisation toward the correct basin without trapping it in a suboptimal one.

Encoder warm-start is marginal. We also test whether the SAE encoder's output provides a useful initialisation for per-sample FISTA (with frozen dictionary). Since the Lasso objective is convex, cold-start and warm-start must converge to the same optimum given sufficient iterations. At 100 iterations, both yield identical MCC. At low iteration budgets (1–10), warm-starting from SAE codes provides a modest advantage for TopK but negligible benefit for other types (appendix, Figure 26). The practical value of the SAE encoder as a FISTA initialiser is limited.

Takeaway. At small d_z, the SAE decoder is a useful warm-start for DL-FISTA, saving $\sim$50 dictionary-update rounds. At large d_z (≥ 5000), this advantage vanishes—neither initialisation leads to a good dictionary. The practical path forward requires better dictionary learning algorithms that scale, not just better initialisations.

E.3. Scaling number of latents

Figure 14 extends the main-text Figures 6 and 10 with all six metric panels (MCC and AUC on both ID and OOD, plus accuracy). The key patterns from the main text hold across all metrics: per-sample methods maintain a small ID–OOD gap while SAEs exhibit a persistent and large gap. The frozen-decoder and refined variants consistently improve over raw SAEs, with gains most pronounced on OOD metrics.

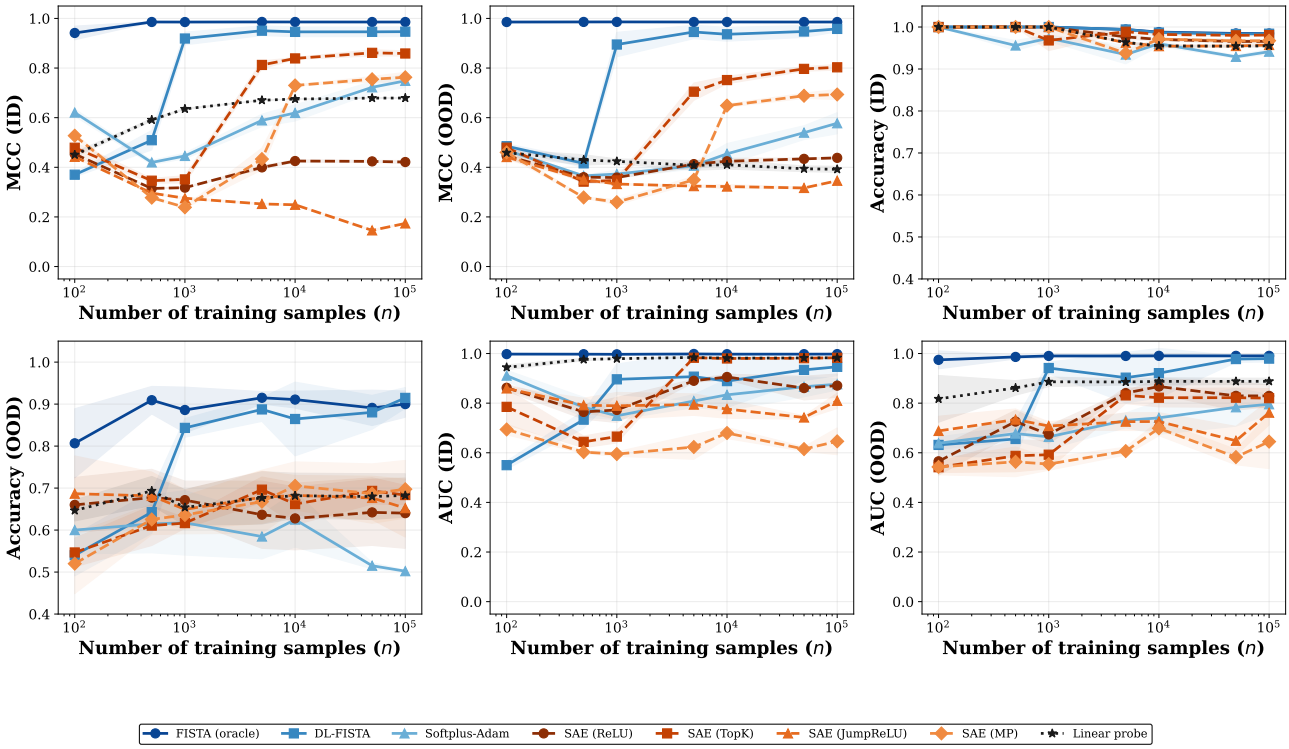


Figure 15. **All metrics vs number of training samples.** DL-FISTA's MCC and AUC improve sharply with more data and saturate by $p = 10^3$. SAE variants plateau around 0.35–0.45 MCC or degrade (JumpReLU), and OOD AUC remains scattered between 0.5–0.85 with high variance regardless of p. $d_z = 100$, $k = 10$, $d_y = 47$.

E.4. Scaling number of samples

Figure 15 extends Figure 8 with all six metrics. DL-FISTA benefits substantially from more data across all metrics, while SAEs plateau or degrade—confirming that the amortisation gap is not a sample-complexity issue.

E.5. Varying sparsity level

Figure 16 sweeps sparsity k with $d_z = 1000$ fixed. Per-sample methods degrade gracefully as sparsity increases (the inference problem becomes harder), maintaining consistent ID–OOD performance. SAE OOD AUC converges toward chance ($\sim$0.5) at high k, further confirming the compositional generalisation failure.

E.6. Additional main-text metrics (v2 figures)

AUC (OOD). AUC evaluates single-feature separability without a trained classifier. Note that the linear probe's AUC is computed on its supervised regression output—each output dimension targets one latent—giving it an inherent advantage over unsupervised methods whose code dimensions need not align with the label. AUC can therefore overstate the linear probe's generalisation capability relative to unsupervised methods; the accuracy metric (Figures 7 and 9) provides the fairer comparison.

E.7. Controlled experiment details

E.8. Lambda sensitivity

The frozen decoder experiments use $\lambda = 0.1$ for FISTA, while SAE training uses $\gamma_{\text{reg}} = 10^{-4}$—a $1000\times$ mismatch. To verify that the dictionary quality conclusion is not an artefact of this mismatch, we sweep $\lambda \in \{10^{-3}, \ldots, 2.0\}$ for both the frozen decoder and oracle conditions (Figure 30).

The oracle achieves peak MCC $\sim$0.95 at $\lambda \approx 0.1$–0.5, confirming the correct operating regime. The frozen decoder peaks at $\lambda \approx 0.5$ for most SAE types, modestly exceeding the SAE encoder baseline for ReLU and JumpReLU (e.g., 0.4 vs

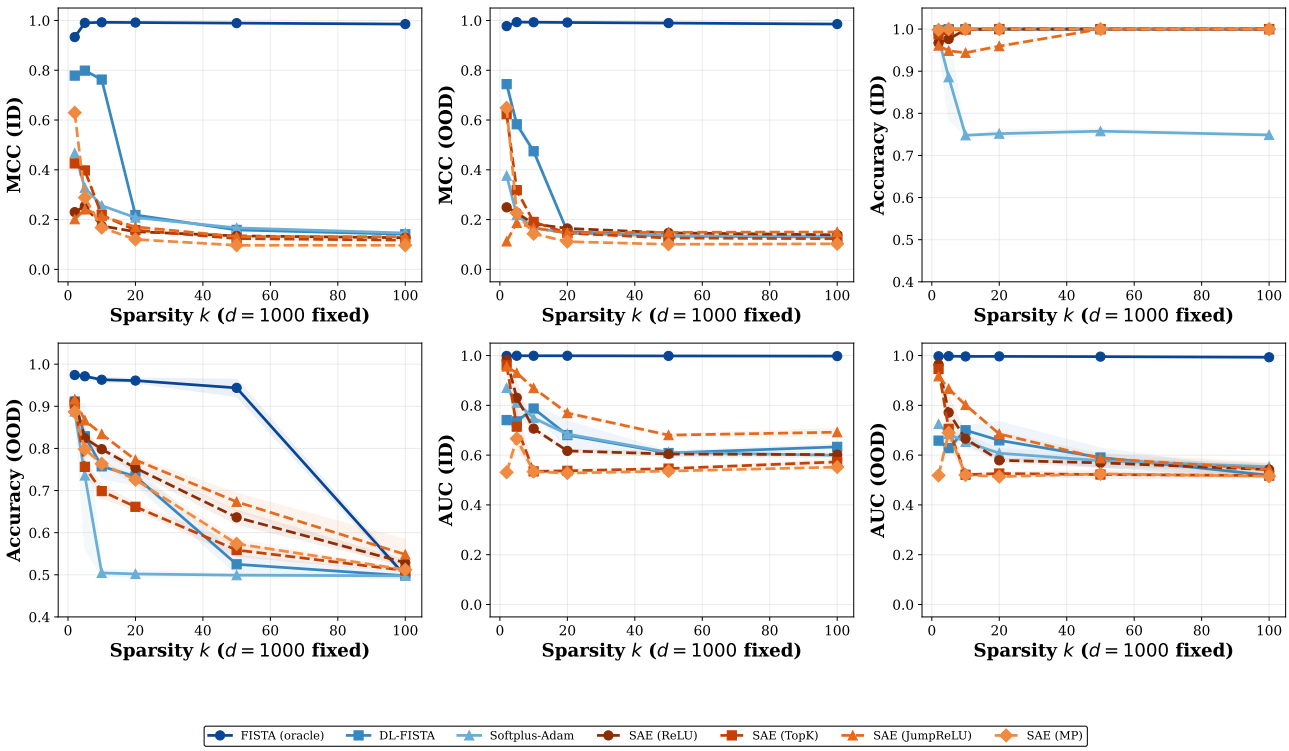


Figure 16. **All metrics vs sparsity k.** Per-sample methods degrade gracefully with increasing k; SAE OOD AUC converges toward chance at high k. $d_z = 1000$, d_y follows the CS bound.

0.25 at $d_z = 5000$). However, the gap between frozen FISTA and oracle remains large at every λ, confirming that the dictionary—not the regularisation strength—is the bottleneck.

E.9. Learning dynamics at other latent dimensions

E.9.1. Controlled experiments at other latent dimensions

The main text reports controlled experiments at $d_z = 100$. Below we show the same experiments at $d_z \in \{50, 500, 1000, 5000\}$ to confirm the findings hold across scales.

Decoder warm-start convergence. The SAE decoder provides a consistent head start for DL-FISTA across all d_z. The advantage is largest at small d_z (where the SAE decoder is closer to the true dictionary) and diminishes at large d_z (where both initialisations require many rounds to converge).

Encoder warm-start convergence. The convex-convergence pattern holds across all d_z: cold-start and warm-start reach the same optimum, with warm-starting providing a modest advantage only at very low iteration counts.

Dictionary quality and support recovery. The dictionary quality decomposition and support recovery patterns are consistent across d_z: re-normalising columns never helps, and ReLU/JumpReLU support precision remains catastrophically low.

E.10. Phase transition ablations

Figures 58 to 62 extend the main-text phase transition (Figure 5) with all remaining metrics. The phase transition pattern is consistent: per-sample methods exhibit a sharp transition to near-perfect performance once δ exceeds the compressed-sensing threshold, while SAEs plateau well below across all metrics.

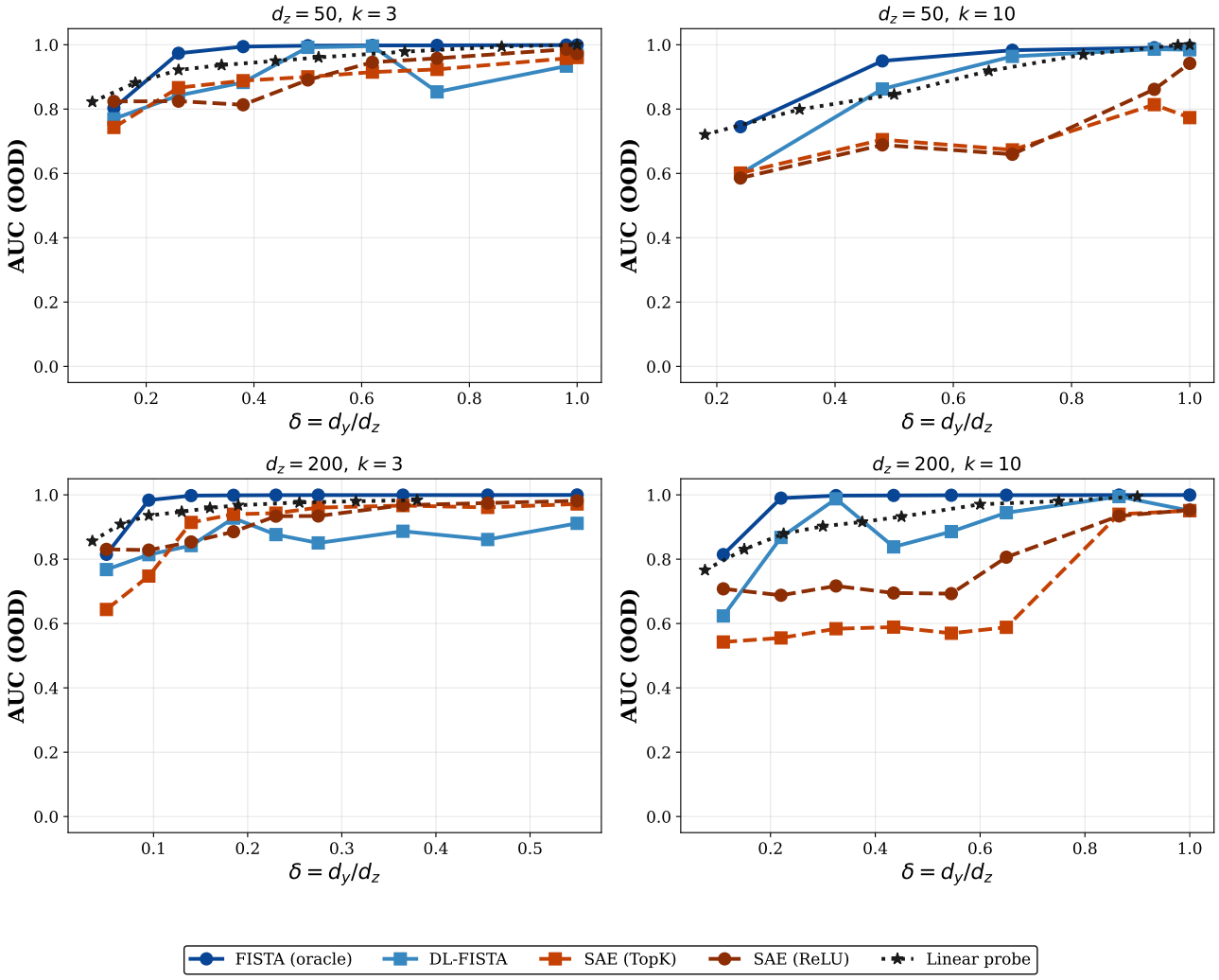
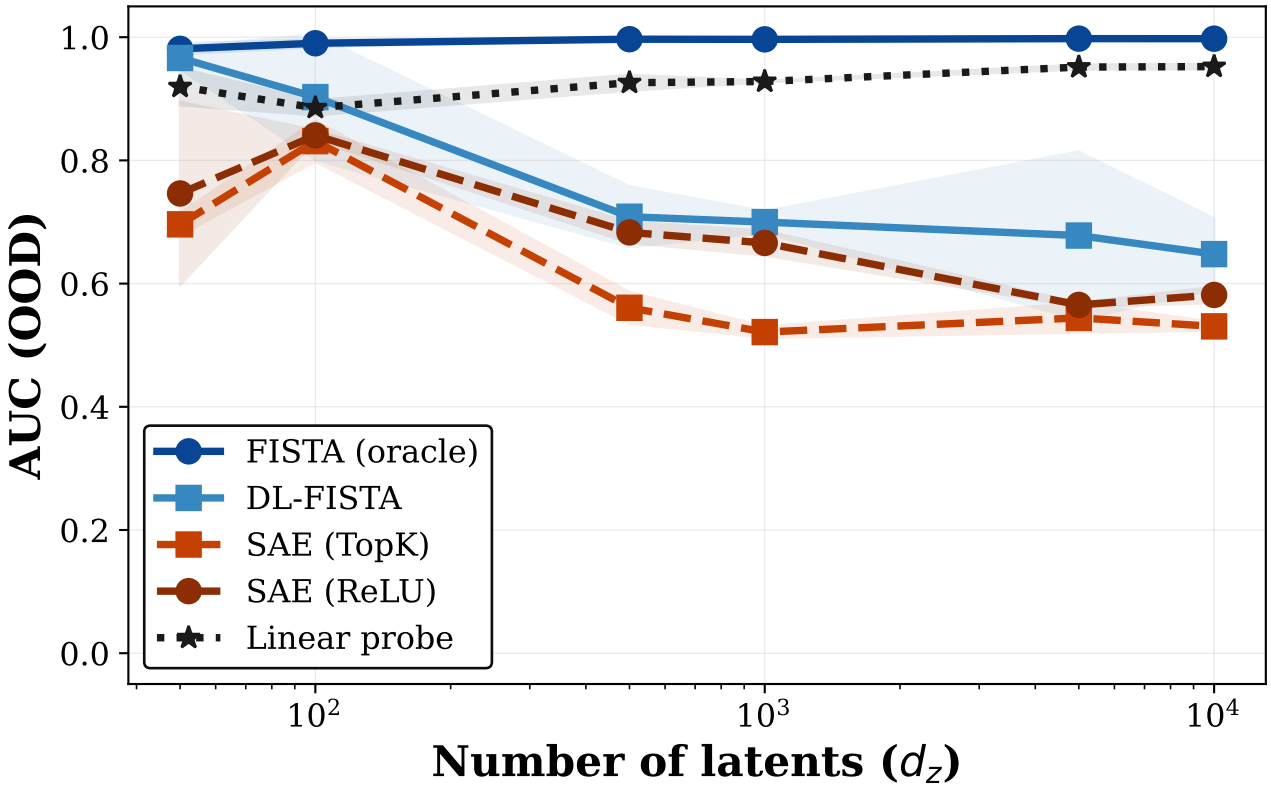


Figure 17. **Phase transition: AUC (OOD).** SAE OOD AUC degrades toward chance while per-sample methods maintain near-perfect performance across all (d_z, k) settings.



SAEs exhibit a persistent ID–OOD gap; FISTA maintains consistent performance.

Figure 18. **Scaling latent dimension: AUC (OOD).** The linear probe maintains high AUC ($\sim$0.93) even as its MCC collapses to 0.07 (Figure 6), illustrating how a supervised single-label metric can mask identifiability failure. FISTA (oracle) dominates; SAEs degrade. $k = 10$, $p = 5000$.

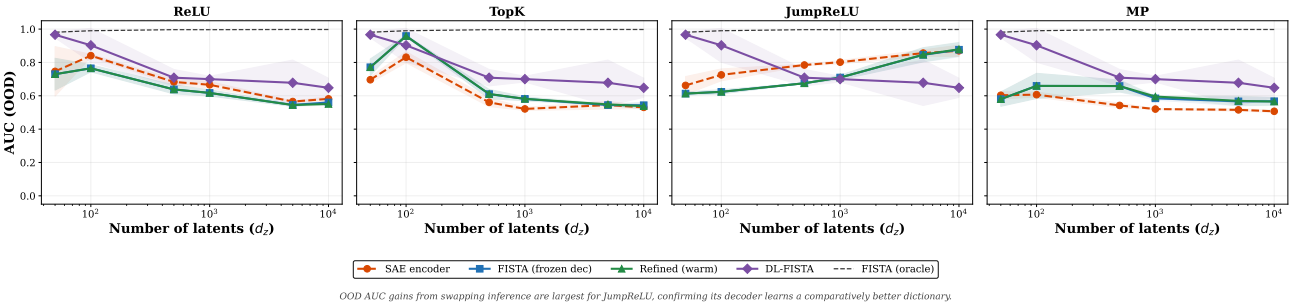
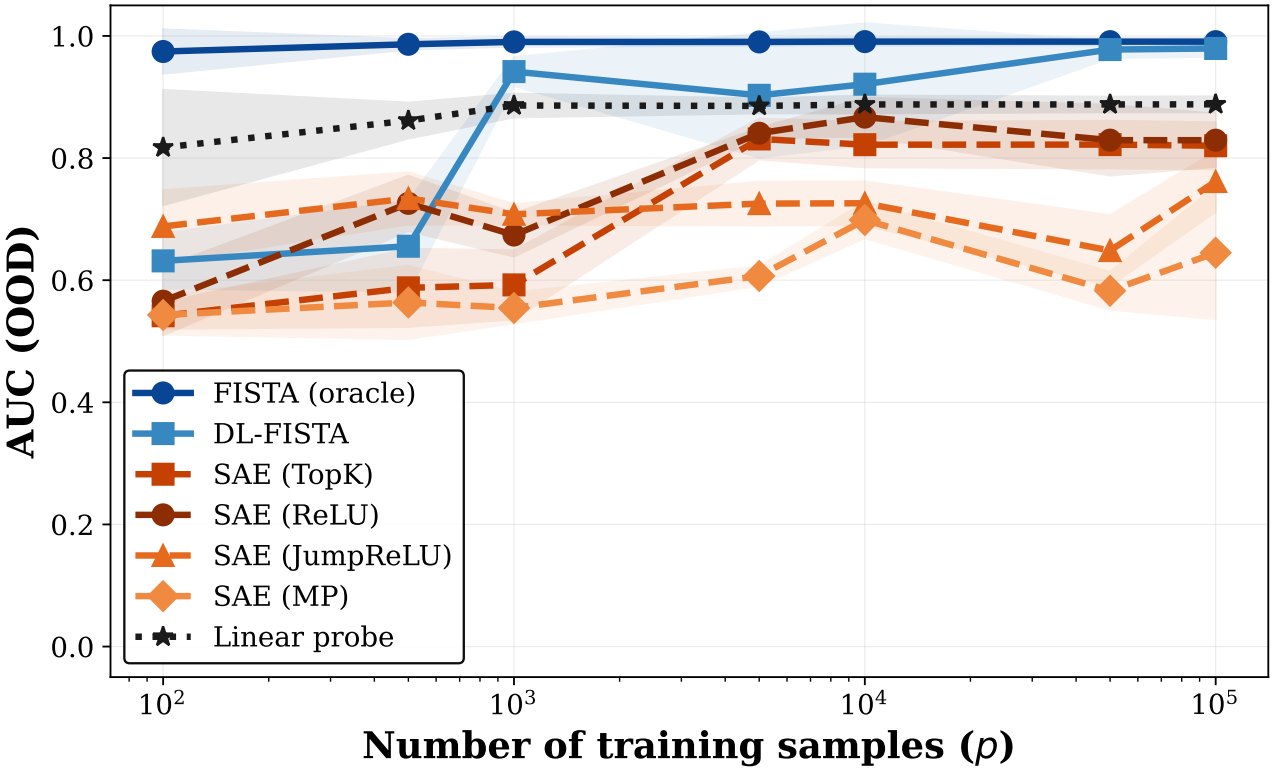
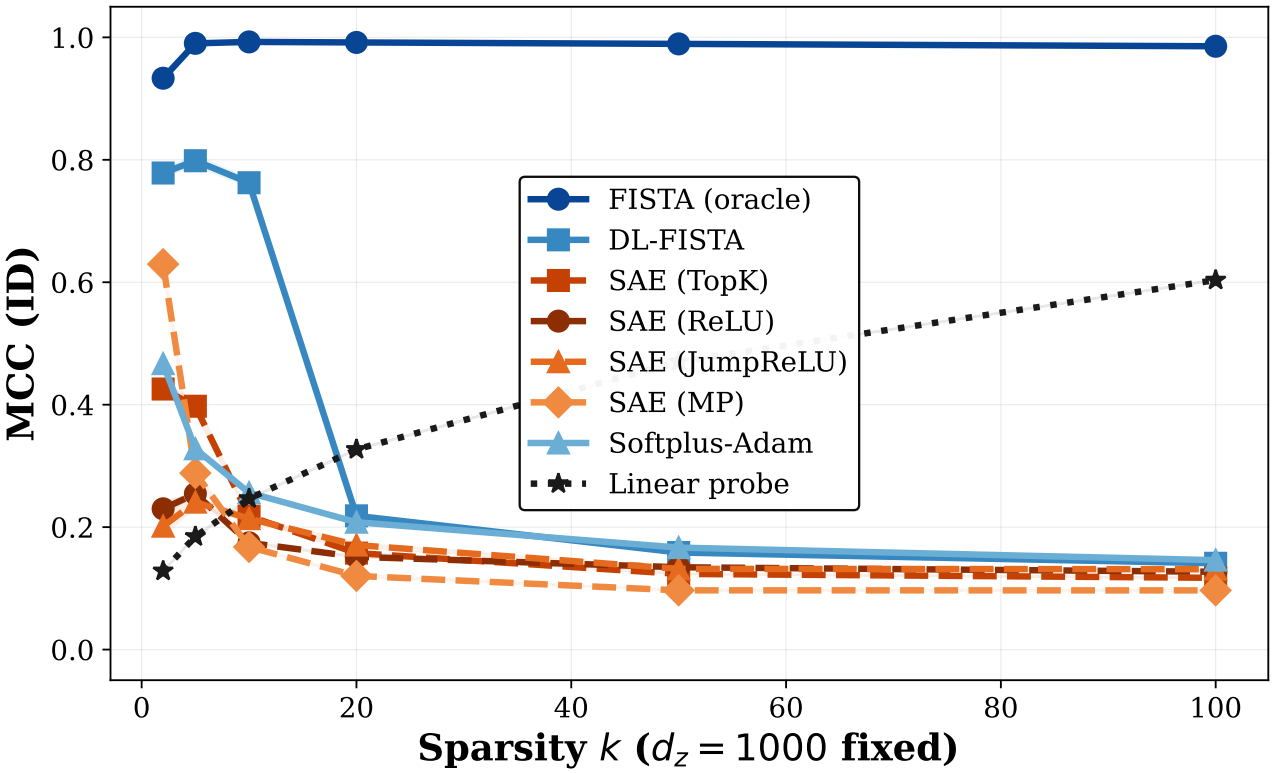


Figure 19. **Frozen decoder ablation: AUC (OOD).** Same layout as Figure 10. FISTA on frozen TopK and JumpReLU decoders yields modest OOD AUC gains, but the gap to DL-FISTA remains large for all types. $k = 10$, $p = 5000$.



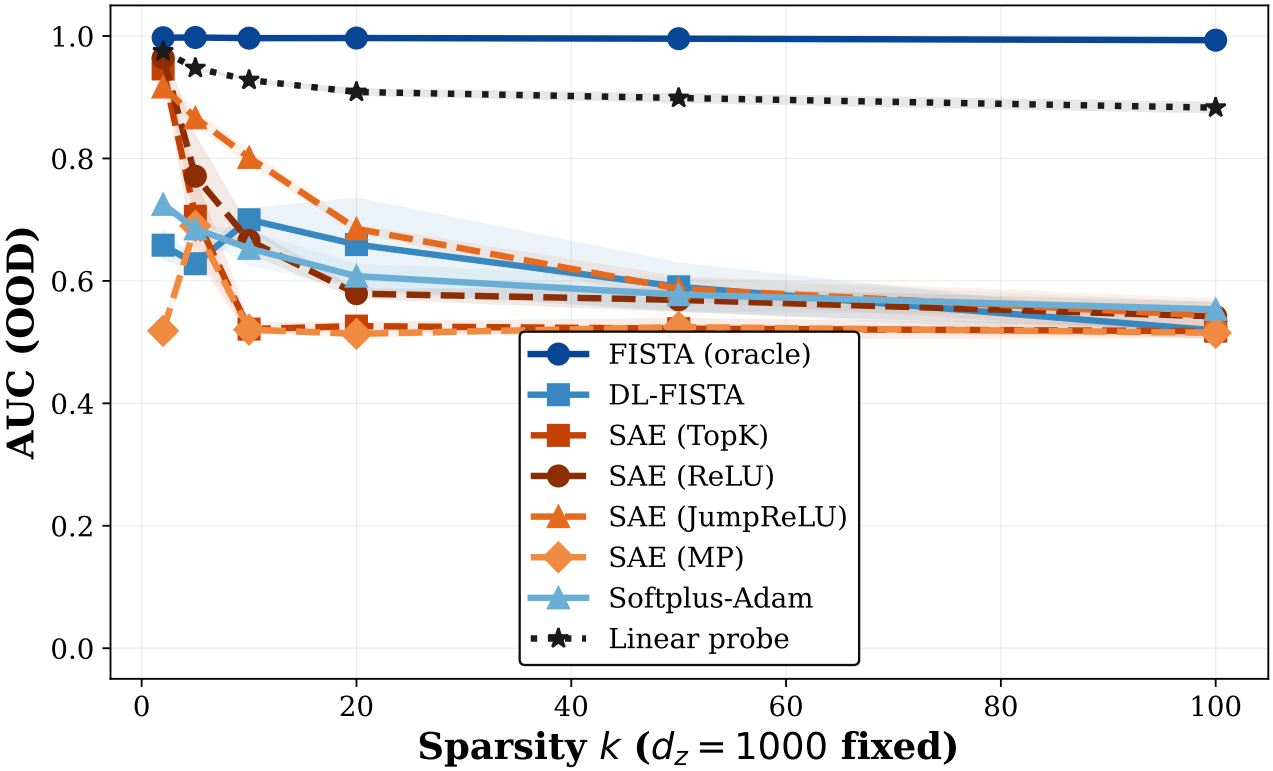
Additional training data benefits dictionary quality but does not close the amortisation gap.

Figure 20. **More data: AUC (OOD).** Additional training data benefits DL-FISTA but does not close the amortisation gap on OOD AUC. $d_z = 100$, $k = 10$, $d_y = 47$.



Per-sample methods degrade gracefully with sparsity; SAEs plateau regardless of k.

Figure 21. **Varying sparsity: MCC (ID).** Per-sample methods degrade gracefully with increasing k; SAEs plateau. $d_z = 1000$, d_y follows the CS bound.



OOD generalisation degrades with sparsity for SAEs; per-sample methods remain robust.

Figure 22. **Varying sparsity: AUC (OOD).** FISTA (oracle) remains near 1.0; all other methods degrade with increasing k, with SAEs and DL-FISTA converging toward 0.5. The linear probe degrades more gracefully ($0.97 \rightarrow 0.88$) due to its supervised advantage. $d_z = 1000$.

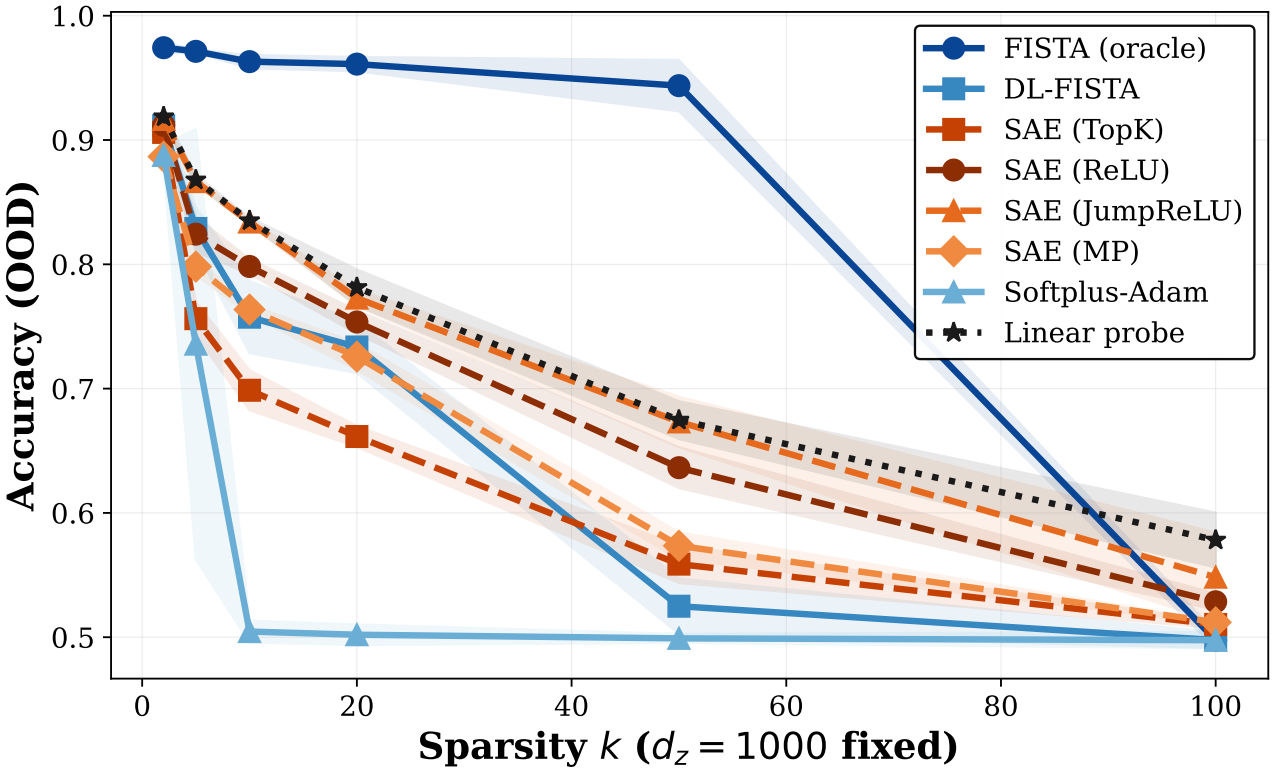
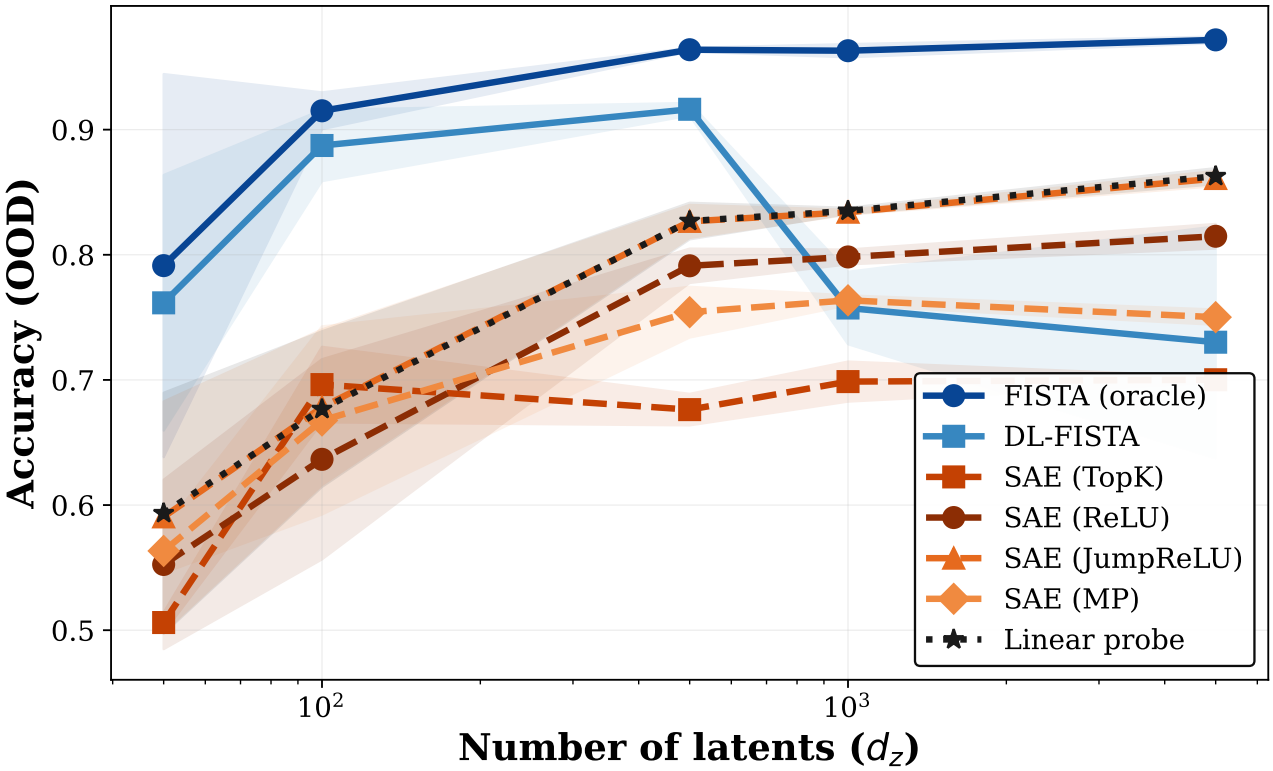


Figure 23. **Varying sparsity: Accuracy (OOD).** Same downstream probe on each method's codes. FISTA (oracle) dominates. DL-FISTA beats the linear probe at low k (≤ 10) but collapses at high k as dictionary learning fails. SAE codes offer no consistent advantage over raw activations. $d_z = 1000$.



Same downstream probe on each method's codes: FISTA (oracle) dominates; SAEs match or trail linear probes.

Figure 24. **Varying latent dimension: Accuracy (OOD).** Duplicate of Figure 7 for completeness alongside other appendix metrics.

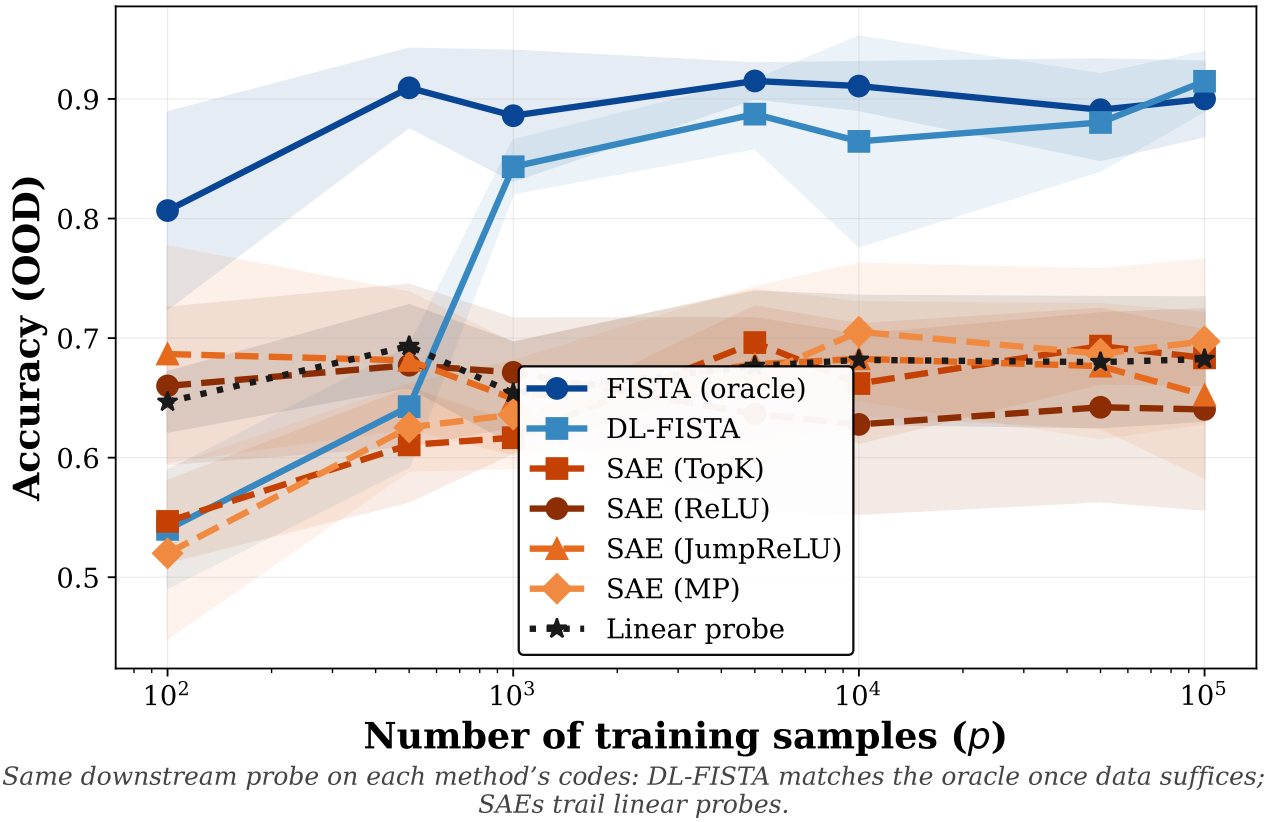


Figure 25. **Varying training data: Accuracy (OOD).** Duplicate of Figure 9 for completeness alongside other appendix metrics.

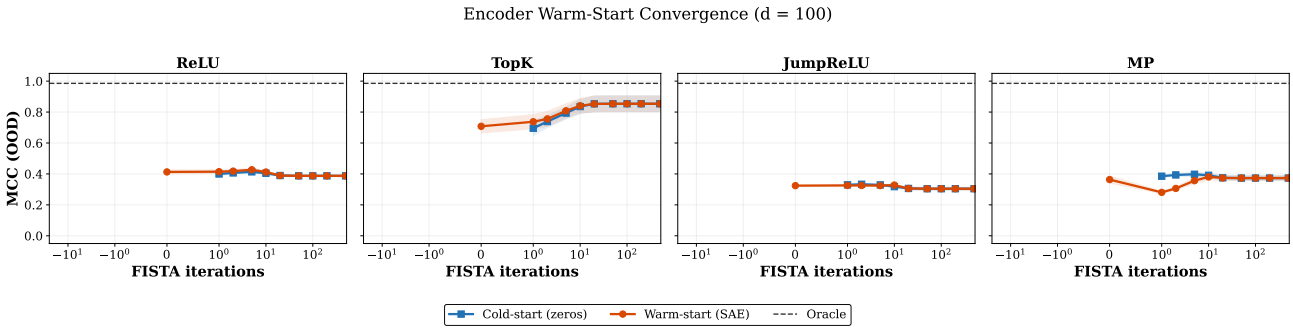


Figure 26. **Encoder warm-start convergence ($d_z = 100$).** Cold-start (blue) and warm-start (orange) FISTA on frozen SAE decoder. The convex Lasso objective means both converge to the same optimum. Warm-starting provides a modest advantage for TopK at low iteration budgets but negligible benefit for other types.

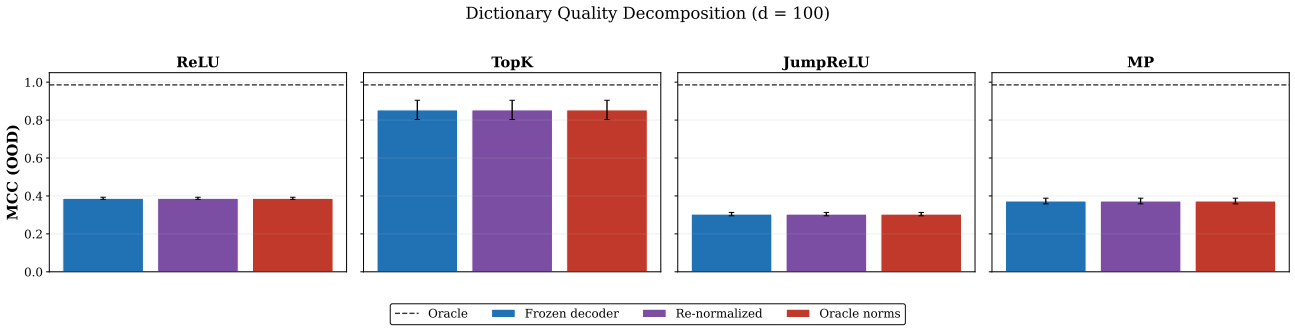


Figure 27. **Dictionary quality decomposition ($d_z = 100$).** Blue: FISTA with frozen SAE decoder. Purple: after re-normalising decoder columns to unit norm. Red: SAE directions with oracle column magnitudes. Re-normalising and norm substitution have no effect—the error is in the column *directions*, not magnitudes.

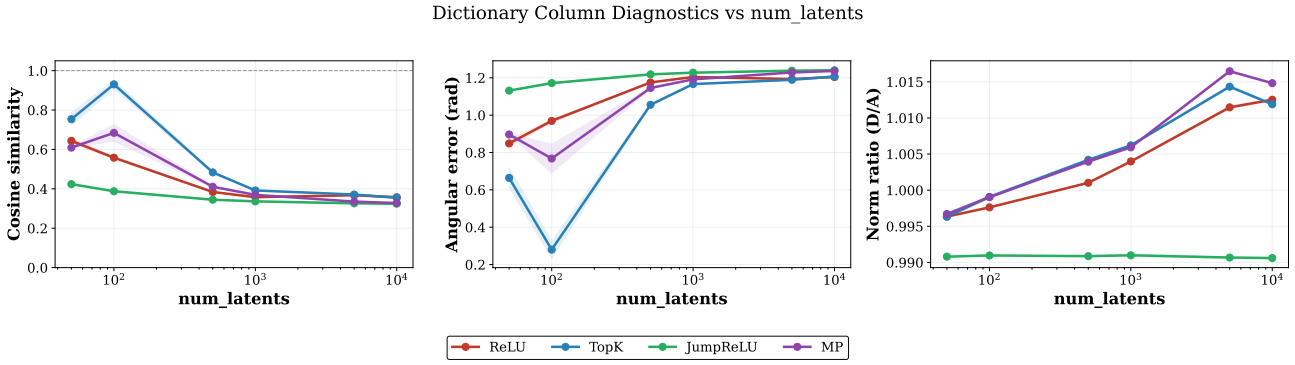


Figure 28. **Dictionary column diagnostics across d_z.** Left: mean cosine similarity between SAE and ground-truth columns. Centre: angular error. Right: norm ratio. TopK maintains high cosine (> 0.9) across all d_z; other types degrade. Norm ratios are $\approx$1.0 throughout, confirming the error is directional.

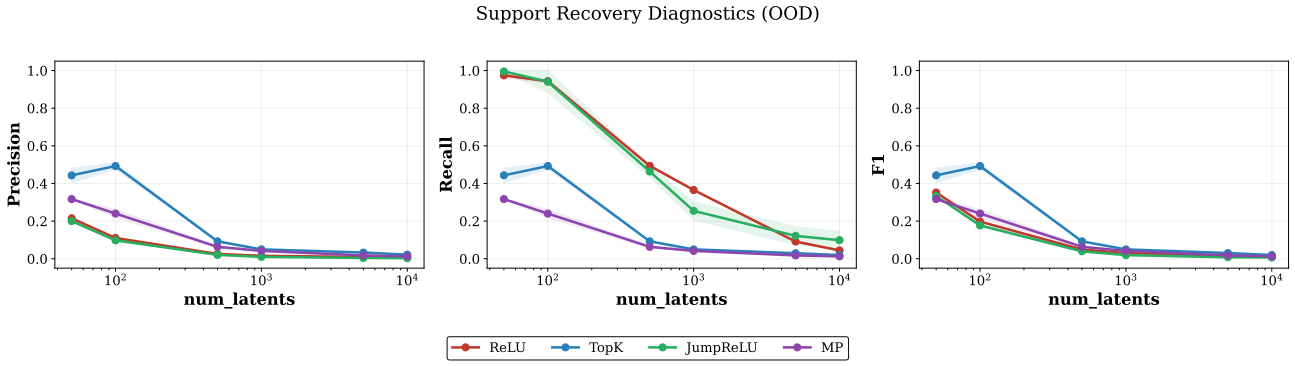


Figure 29. **Support recovery diagnostics across d_z.** Precision, recall, and F1 of the SAE's binary support compared to ground truth. ReLU and JumpReLU have high recall but catastrophically low precision ($\sim$0.1): they activate nearly all features. TopK maintains balanced precision and recall ($\sim$0.5).

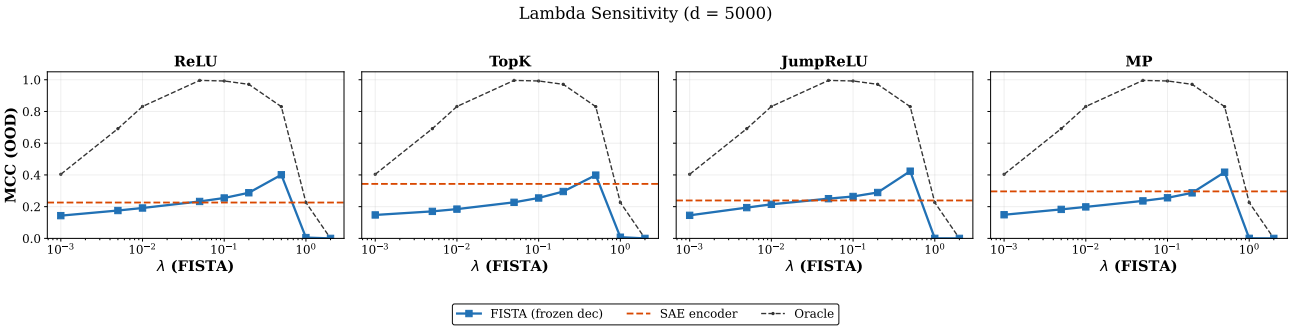


Figure 30. **Lambda sensitivity ($d_z = 5000$).** Blue: FISTA on frozen SAE decoder across λ values. Orange dashed: SAE encoder (lambda-independent). Gray dashed: oracle FISTA. The frozen decoder peaks modestly above the SAE at $\lambda \approx 0.5$, but the gap to oracle persists at every λ.

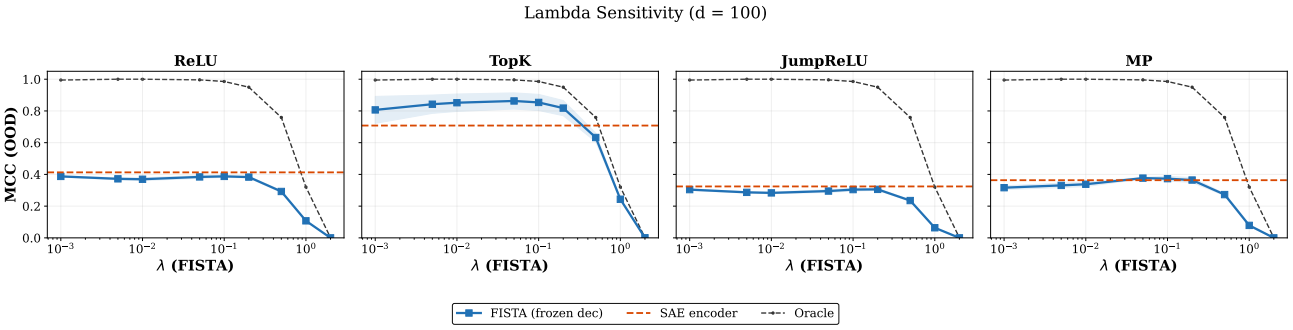


Figure 31. Lambda sensitivity, $d_z = 100$.

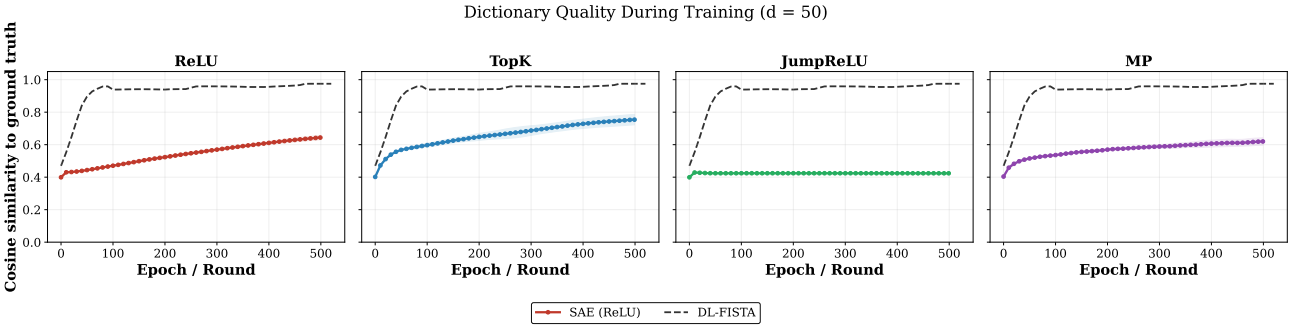


Figure 32. Dictionary quality during training, $d_z = 50$.

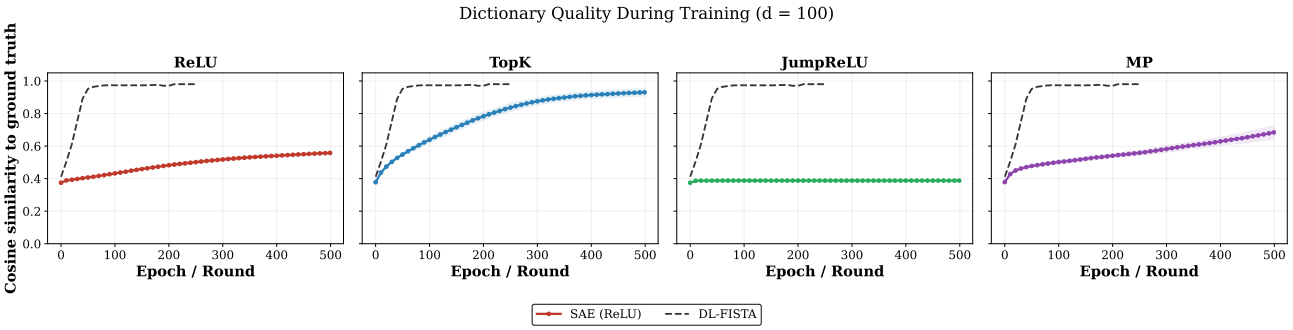


Figure 33. Dictionary quality during training, $d_z = 100$. At this scale, DL-FISTA converges to high cosine while SAEs plateau at lower values.

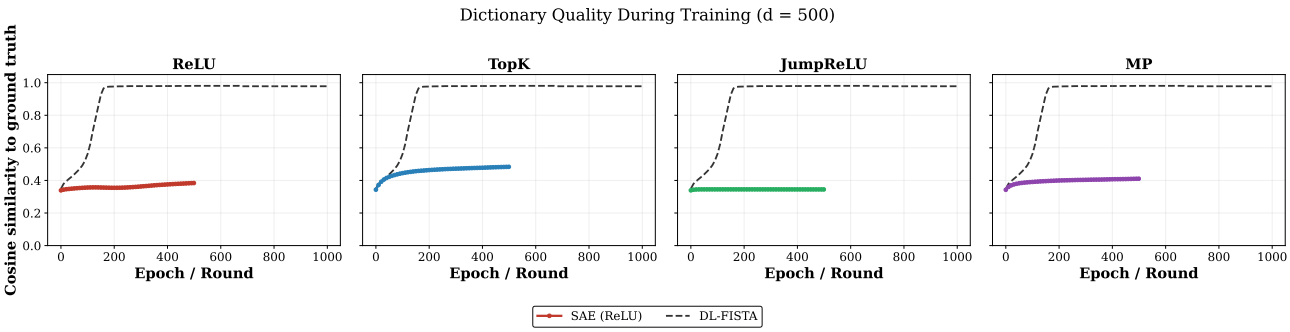


Figure 34. Dictionary quality during training, $d_z = 500$.

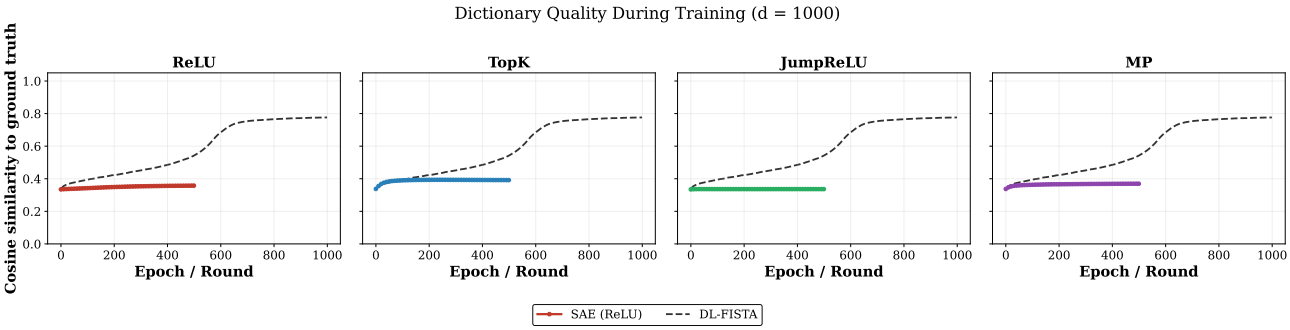


Figure 35. Dictionary quality during training, $d_z = 1000$.

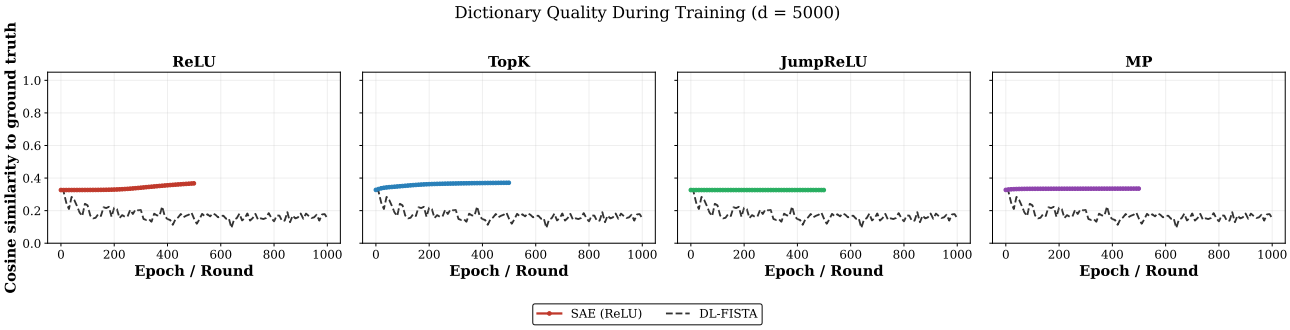


Figure 36. Dictionary quality during training, $d_z = 5000$. Neither SAEs nor DL-FISTA converge to high cosine at this scale.

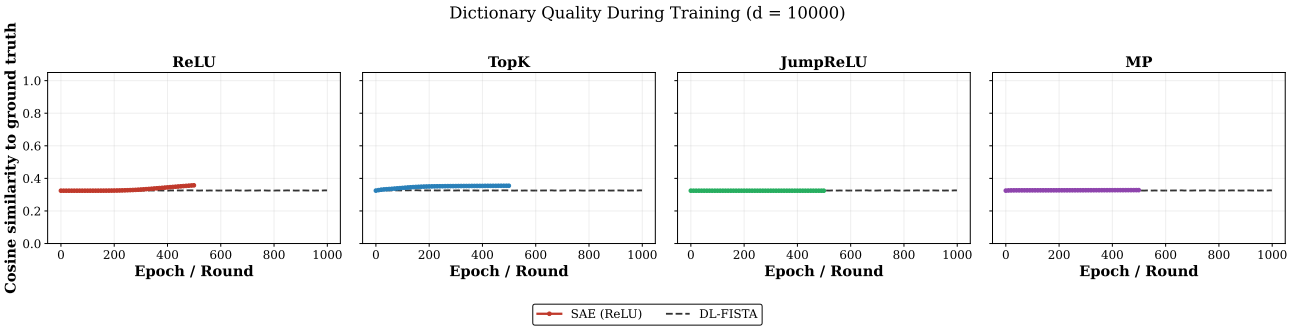


Figure 37. Dictionary quality during training, $d_z = 10000$.

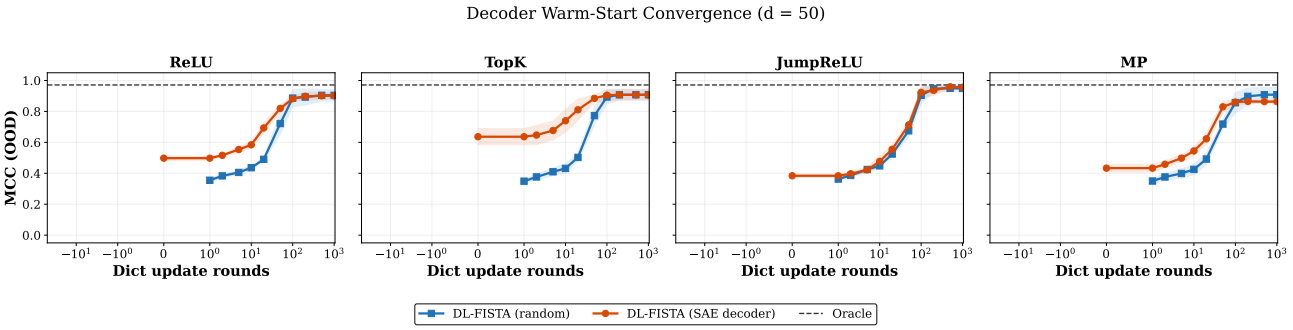


Figure 38. Decoder warm-start convergence, $d_z = 50$.

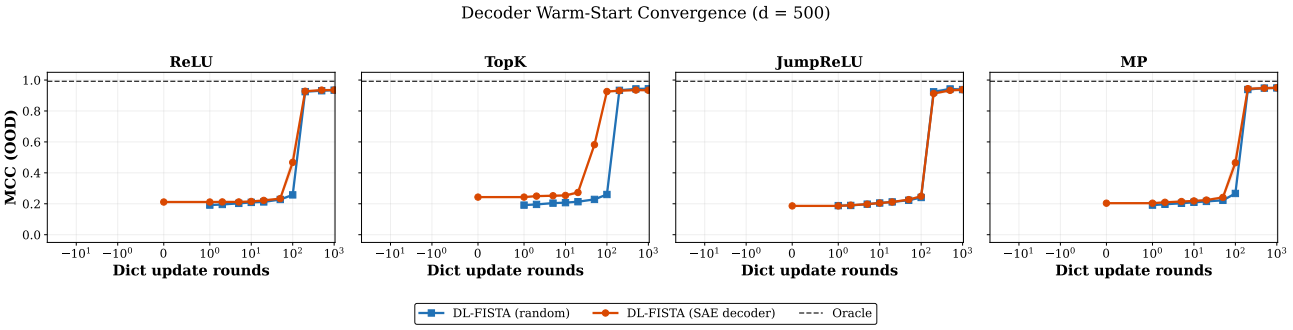


Figure 39. Decoder warm-start convergence, $d_z = 500$.

F. Theoretical model for toy setting

We study the geometry of a system where a sparse source vector $z \in [0,1]^3$ with at most two non-zero elements ($\|z\|_0 \leq 2$) is linearly projected to an observation $y \in \mathbb{R}^2$ (see Fig. 1):

$$y = Az. \tag{10}$$

The sparsity constraint implies that any observation is a combination of at most two active source components. Whenever active, we assume that each source follows a uniform distribution $z_i \mid i \text{ active} \sim \text{Uniform}(0,1)$. The training data is considered *independent and identically distributed* (IID) and is generated from combinations of sources (z_1, z_2) or (z_2, z_3). The test data is considered *out-of-distribution* (OOD) and is generated from the novel combination (z_1, z_3). Our goal is to determine whether the first variable z_1 is above a certain, safety-relevant, threshold $z_1 = \frac{1}{2}$.

To analyze the geometry, we examine the columns $A_i \in \mathbb{R}^2$ of the projection matrix. We define the angles $\phi := \angle(A_1, A_2)$ and $\theta := \angle(A_1, A_3)$, which fully determine the system. To simplify the analysis, we make two assumptions:

1. We align our coordinate system and fix the magnitude of the first basis vector relative to our threshold, such that $A_1 = (2, 0)$ and $\|A_2\| = \|A_3\| = 1$.
2. To ensure the cones spanned by the vectors do not overlap, we require that $0 < \phi, \theta < \pi$ and $\phi + \theta > \pi$. This is an illustrative way of understanding why and when compressed sensing is possible in this system.

A perfect linear classifier trained on the IID data must separate the space based on the condition $z_1 = \frac{1}{2}$. In the observation space, this corresponds to a line parallel to A_2 and passing through the point $\frac{1}{2}A_1$. This decision boundary is the line parameterized by:

$$y(\beta) = \frac{1}{2}A_1 + \beta A_2, \quad \beta \in \mathbb{R}. \tag{11}$$

The question we are interested in is: *what is the accuracy of this classifier on the OOD data?* We derive the analytically predicted OOD accuracy for this perfect linear IID classifier, separating two cases, in Appendix F.1.

The simulations and analytical prediction are tested and illustrated in Fig. 63 confirming the validity of the theory.

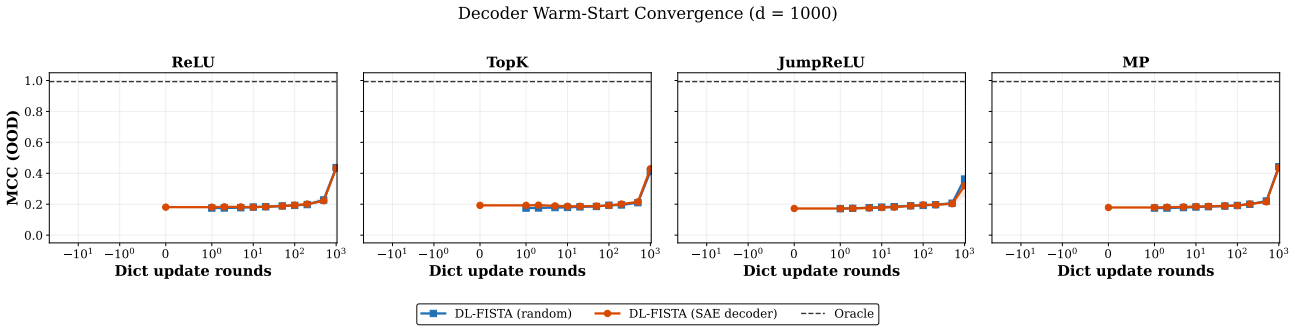


Figure 40. Decoder warm-start convergence, $d_z = 1000$.

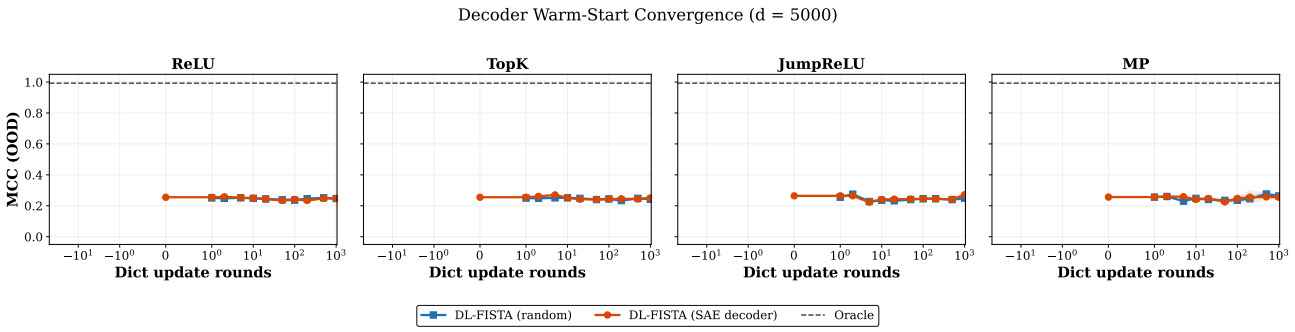


Figure 41. Decoder warm-start convergence, $d_z = 5000$.

F.1. Derivation

We study the geometry of a system where a sparse source vector $z \in [0,1]^3$ with at most two non-zero elements ($\|z\|_0 \leq 2$) is linearly projected to an observation $y \in \mathbb{R}^2$ (see Fig. 1):

$$y = Az. \tag{12}$$

The sparsity constraint implies that any observation is a combination of at most two active source components. Whenever active, we assume that each source follows a uniform distribution $z_i \mid i \text{ active} \sim \text{Uniform}(0,1)$. The training data is considered *independent and identically distributed* (IID) and is generated from combinations of sources (z_1, z_2) or (z_2, z_3). The test data is considered *out-of-distribution* (OOD) and is generated from the novel combination (z_1, z_3). Our goal is to determine whether the first variable z_1 is above a certain, safety-relevant, threshold $z_1 = \frac{1}{2}$.

To analyze the geometry, we examine the columns $A_i \in \mathbb{R}^2$ of the projection matrix. We define the angles $\phi := \angle(A_1, A_2)$ and $\theta := \angle(A_1, A_3)$, which fully determine the system. To simplify the analysis, we make two assumptions:

1. We align our coordinate system and fix the magnitude of the first basis vector relative to our threshold, such that $A_1 = (2, 0)$ and $\|A_2\| = \|A_3\| = 1$.
2. To ensure the cones spanned by the vectors do not overlap, we require that $0 < \phi, \theta < \pi$ and $\phi + \theta > \pi$. This is an illustrative way of understanding why and when compressed sensing is possible in this system.

A perfect linear classifier trained on the IID data must separate the space based on the condition $z_1 = \frac{1}{2}$. In the observation space, this corresponds to a line parallel to A_2 and passing through the point $\frac{1}{2}A_1$. This decision boundary is the line parameterized by:

$$y(\beta) = \frac{1}{2}A_1 + \beta A_2, \quad \beta \in \mathbb{R}. \tag{13}$$

The question we are interested in is: *what is the accuracy of this classifier on the OOD data?* Clearly, the perfect linear classifier for the OOD data would have a decision boundary that is parallel to A_3 and shifted by $\frac{1}{2}A_1$, i.e., the line:

$$y_{\text{OOD}}(\beta) = \frac{1}{2}A_1 + \beta A_3, \quad \beta \in \mathbb{R}. \tag{14}$$

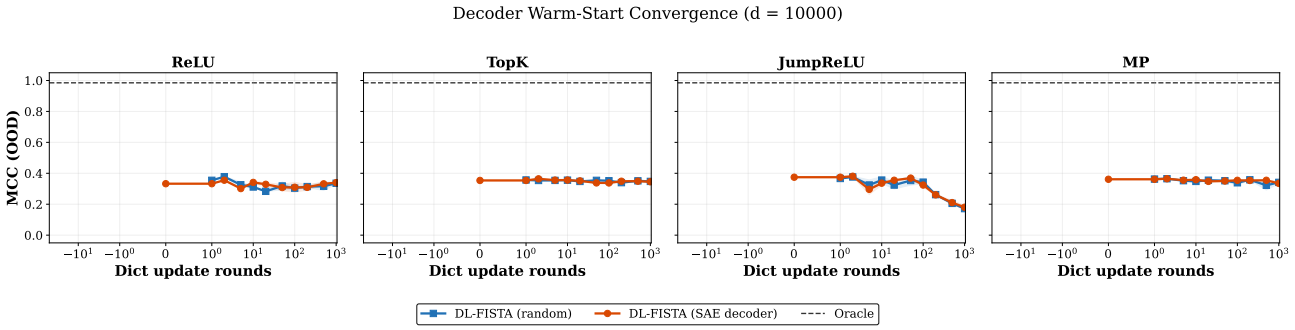


Figure 42. Decoder warm-start convergence, $d_z = 10000$.

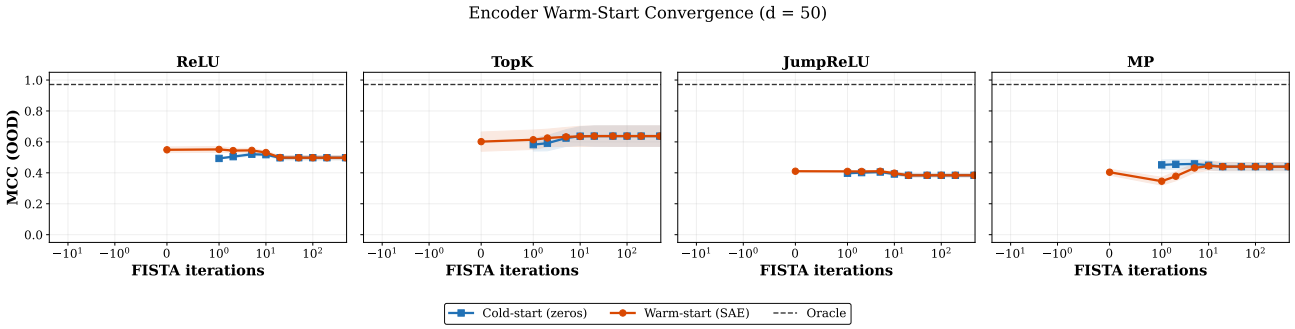


Figure 43. Encoder warm-start convergence, $d_z = 50$.

Since $\phi + \theta > \pi$, we know that the IID classifier's boundary cannot be aligned with the ideal OOD classifier's boundary, so there must be some OOD error. Moreover, we know that the IID classifier can never 'under-shoot' on the OOD data (that would require $\phi + \theta < \pi$). Consequently, we will only observe *false negatives*—that is, test points with $z_1 > \frac{1}{2}$ that are erroneously classified as safe.

We now have to distinguish: **Case 1**, where the classifier passes right from the top right corner $(A_1 + A_3)$ (Fig. 63 Point C), and **Case 2**, where the classifier passes left from the top right corner $(A_1 + A_3)$ (Fig. 63 Point A).

The separation happens when the classifier passes through the top right corner. In that case it will form a triangle through the points $(\frac{1}{2}A_1, A_1, A_1 + A_3)$, with associated angles $(a, b, c) := (\pi - \phi, \pi - \theta, \phi + \theta - \pi)$. By assumption, the base of this triangle has length 1. Consequently, trigonometry tells us that the first angle must have a fixed relation to the second angle $a = \frac{\pi - b}{2}$. From this it follows that the condition for Case 1 is

$$\frac{\pi - (\pi - \theta)}{2} < \pi - \phi \quad \Rightarrow \quad \phi + \frac{\theta}{2} < \pi. \tag{15}$$

The total area of the right parallelogram $(\frac{1}{2}A_1, A_1, A_1 + A_3, \frac{1}{2}A_1 + A_3)$ is $\alpha = \sin(\theta)$. To compute the area of a triangle within this diagram, we use the fact that the area of a triangle can be computed from one side and the adjacent angles. We will always pick a side with length 1, so that if the adjacent angles are (a, b), the area equals

$$\alpha(a, b) = \frac{\sin(a)\sin(b)}{2\sin(a + b)}. \tag{16}$$

In Case 1, we compute the area (α_1) of the triangle between the classifier and A_1. The base between $\frac{1}{2}A_1$ and A_1 has length 1. The angle on the left is $a_1 = \pi - \phi$ and the angle on the right is $b_1 = \pi - \theta$. Thus, using equation 16, the area is

$$\alpha_1 = \frac{\sin(\pi - \phi)\sin(\pi - \theta)}{2\sin(\phi + \theta - \pi)} = \frac{\sin(\phi)\sin(\theta)}{2\sin(\phi + \theta - \pi)} \tag{17}$$

The OOD accuracy will be 50% for the true negatives, plus 50% times the proportion that the area occupies in the right parallelogram (α)

$$acc_1(\text{OOD}) = \frac{1}{2} + \frac{\alpha_1}{2\alpha}. \tag{18}$$

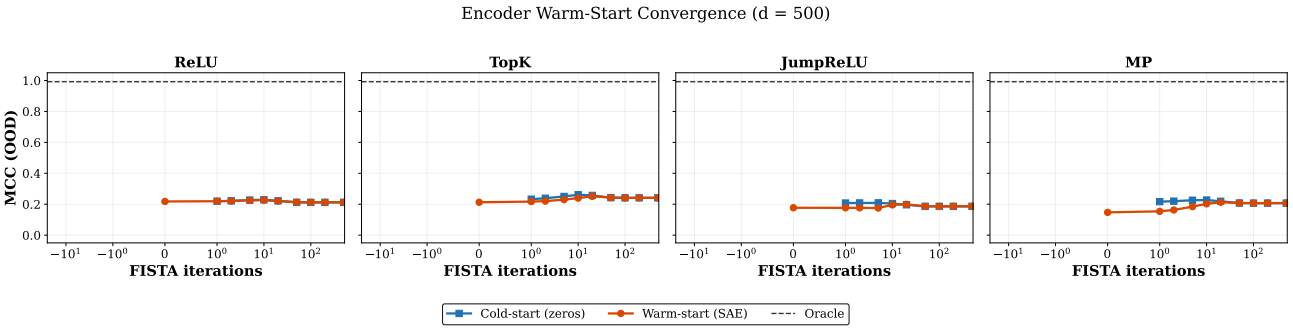


Figure 44. Encoder warm-start convergence, $d_z = 500$.

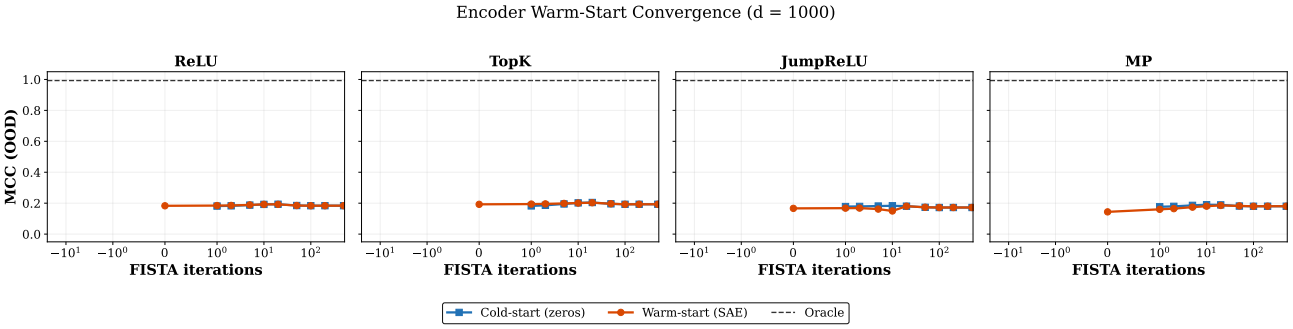


Figure 45. Encoder warm-start convergence, $d_z = 1000$.

In Case 2, we compute the area (α_2) of the triangle between the classifier and the correct OOD decision boundary. The base between $\frac{1}{2}A_1$ and $\frac{1}{2}A_1 + A_3$ has length 1. The angle on top is $a_2 = \pi - \theta$ and the angle below is $b_2 = \phi + \theta - \pi$. Thus, using equation 16, the area is

$$\alpha_2 = \frac{\sin(\pi - \theta)\sin(\phi + \theta - \pi)}{2\sin(\phi)} = \frac{\sin(\theta)\sin(\phi + \theta - \pi)}{2\sin(\phi)} \tag{19}$$

The OOD accuracy will be 100% minus the proportion that the area occupies in the left plus right parallelogram (2α)

$$acc_2(\text{OOD}) = 1 - \frac{\alpha_2}{2\alpha}. \tag{20}$$

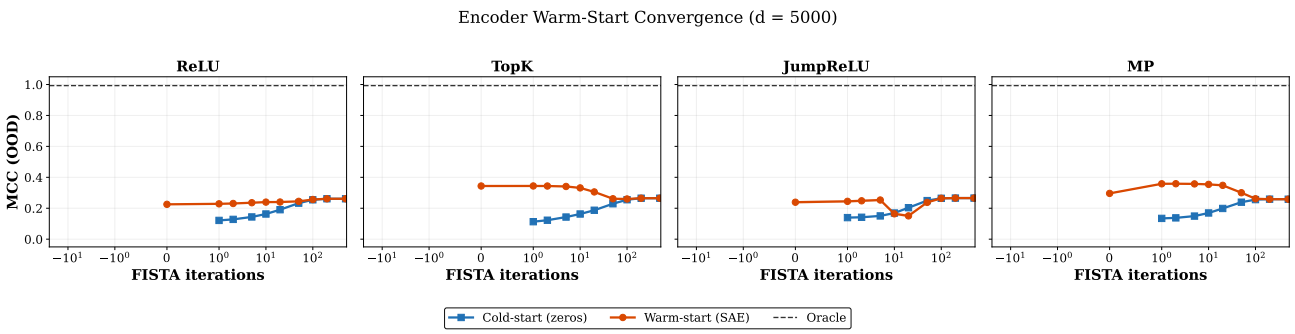


Figure 46. Encoder warm-start convergence, $d_z = 5000$.

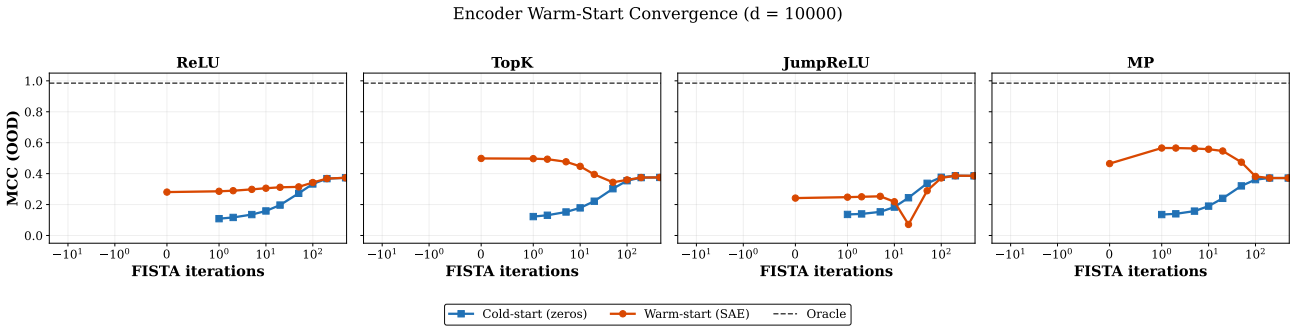


Figure 47. Encoder warm-start convergence, $d_z = 10000$.

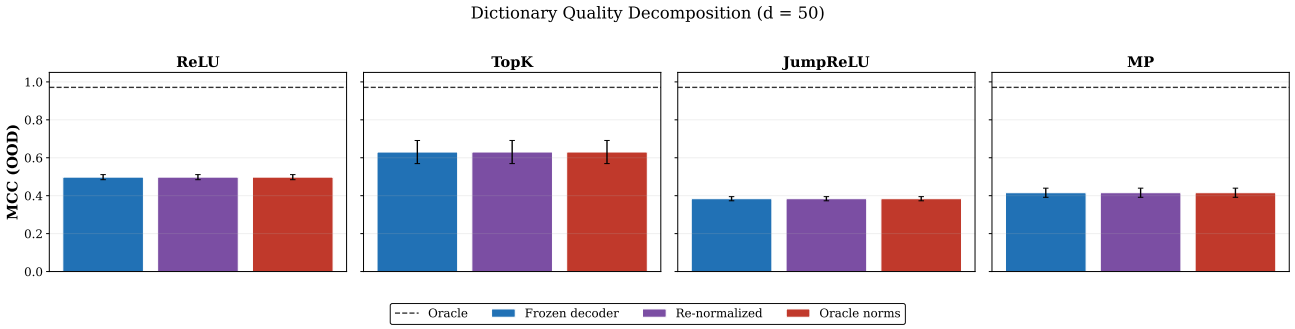


Figure 48. Dictionary quality decomposition, $d_z = 50$.

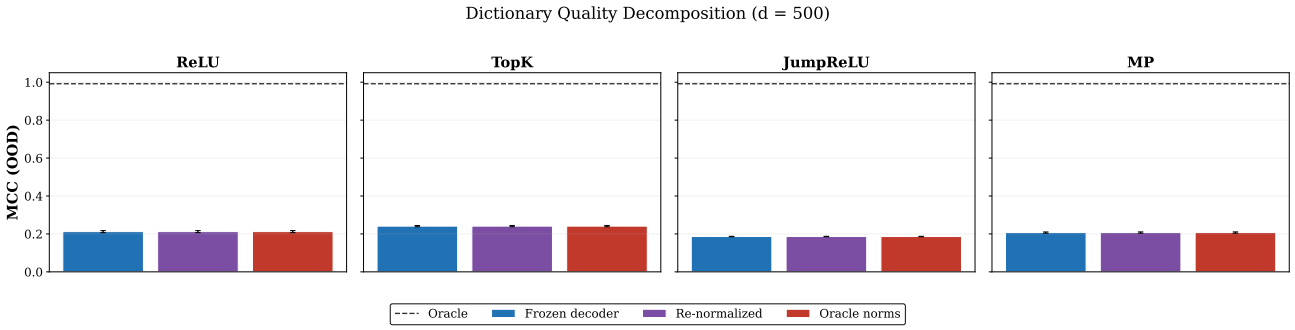


Figure 49. Dictionary quality decomposition, $d_z = 500$.

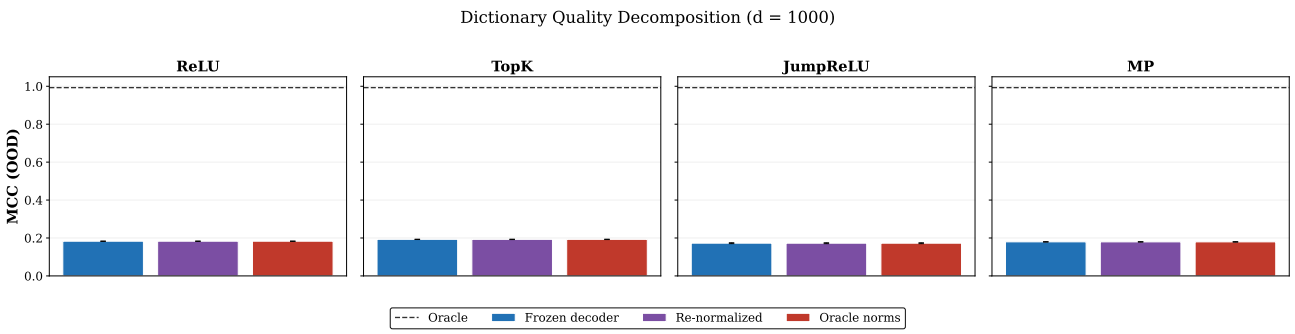


Figure 50. Dictionary quality decomposition, $d_z = 1000$.

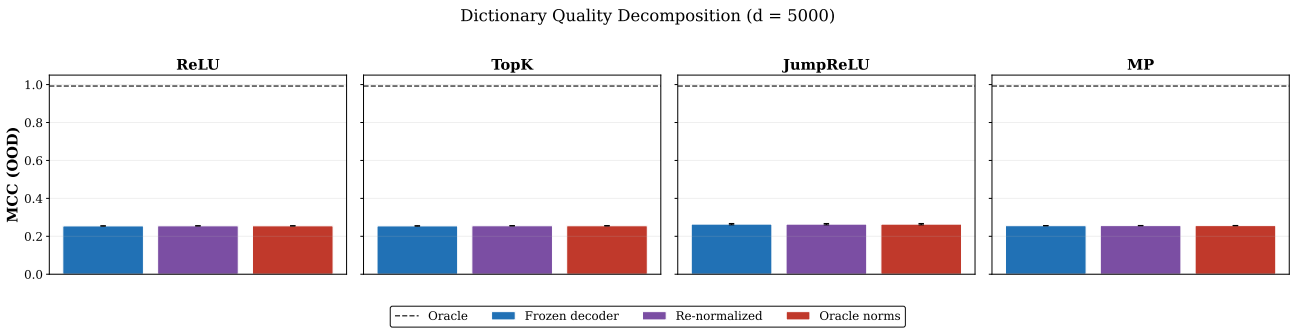


Figure 51. Dictionary quality decomposition, $d_z = 5000$.

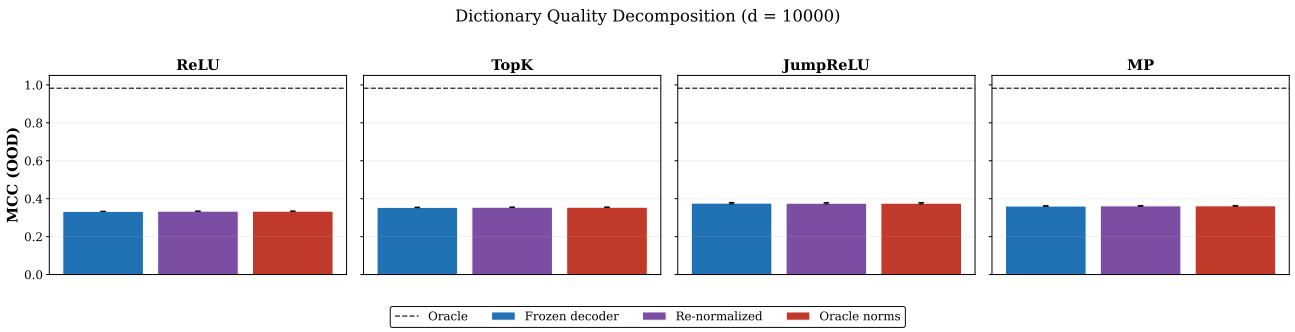


Figure 52. Dictionary quality decomposition, $d_z = 10000$.

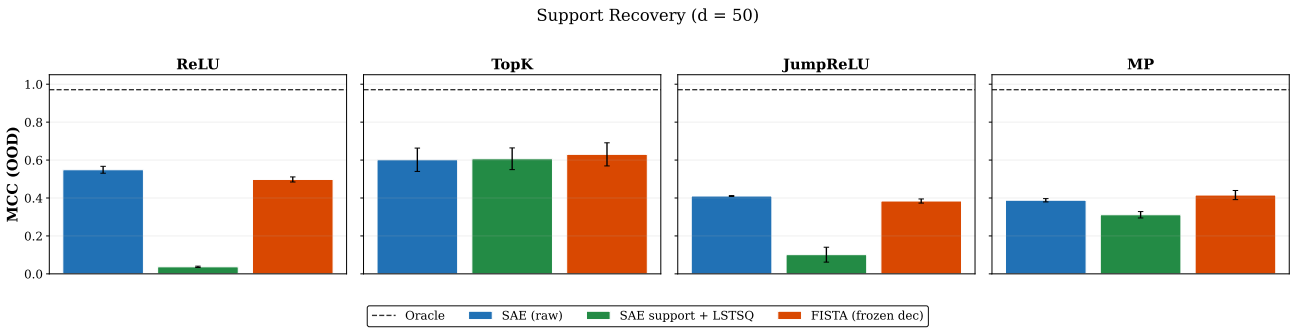


Figure 53. Support recovery, $d_z = 50$.

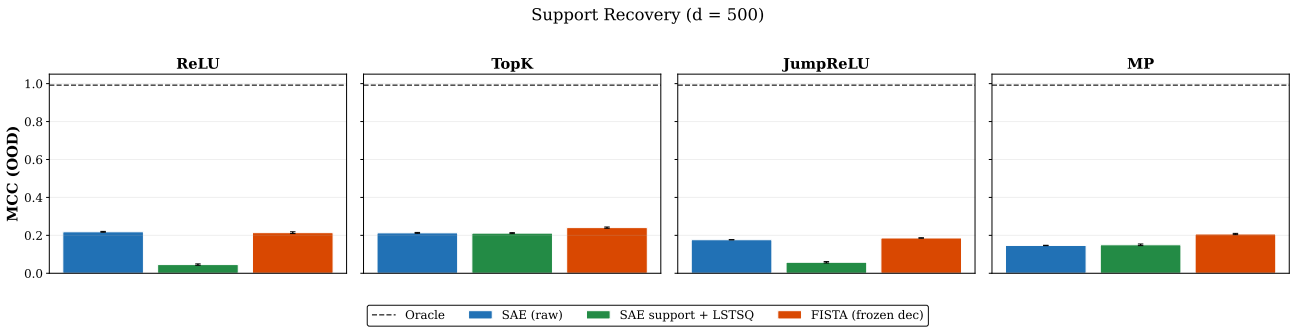


Figure 54. Support recovery, $d_z = 500$.

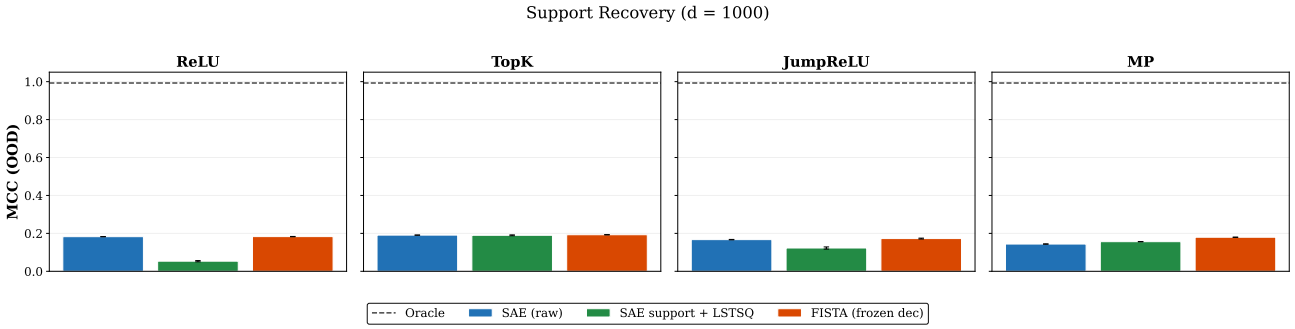


Figure 55. Support recovery, $d_z = 1000$.

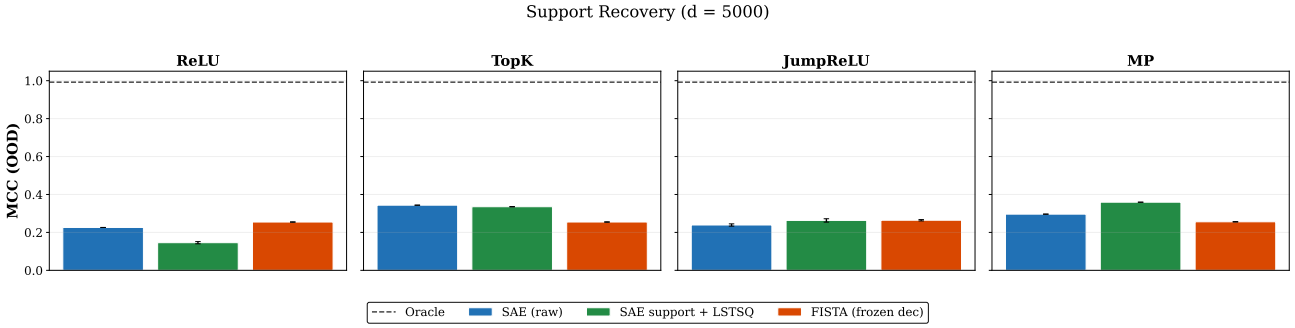


Figure 56. **Re-estimating magnitudes on the SAE's incorrect support degrades MCC.** Blue: raw SAE codes. Green: SAE support with least-squares magnitude re-estimation. Orange: FISTA on frozen decoder. Dashed: oracle. Only TopK's support is useful. $d_z = 5000$, $k = 10$.

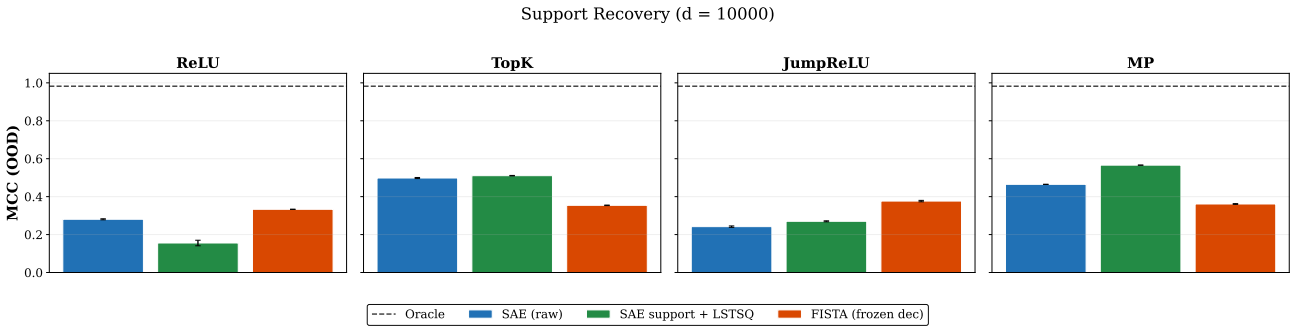


Figure 57. Support recovery, $d_z = 10000$.

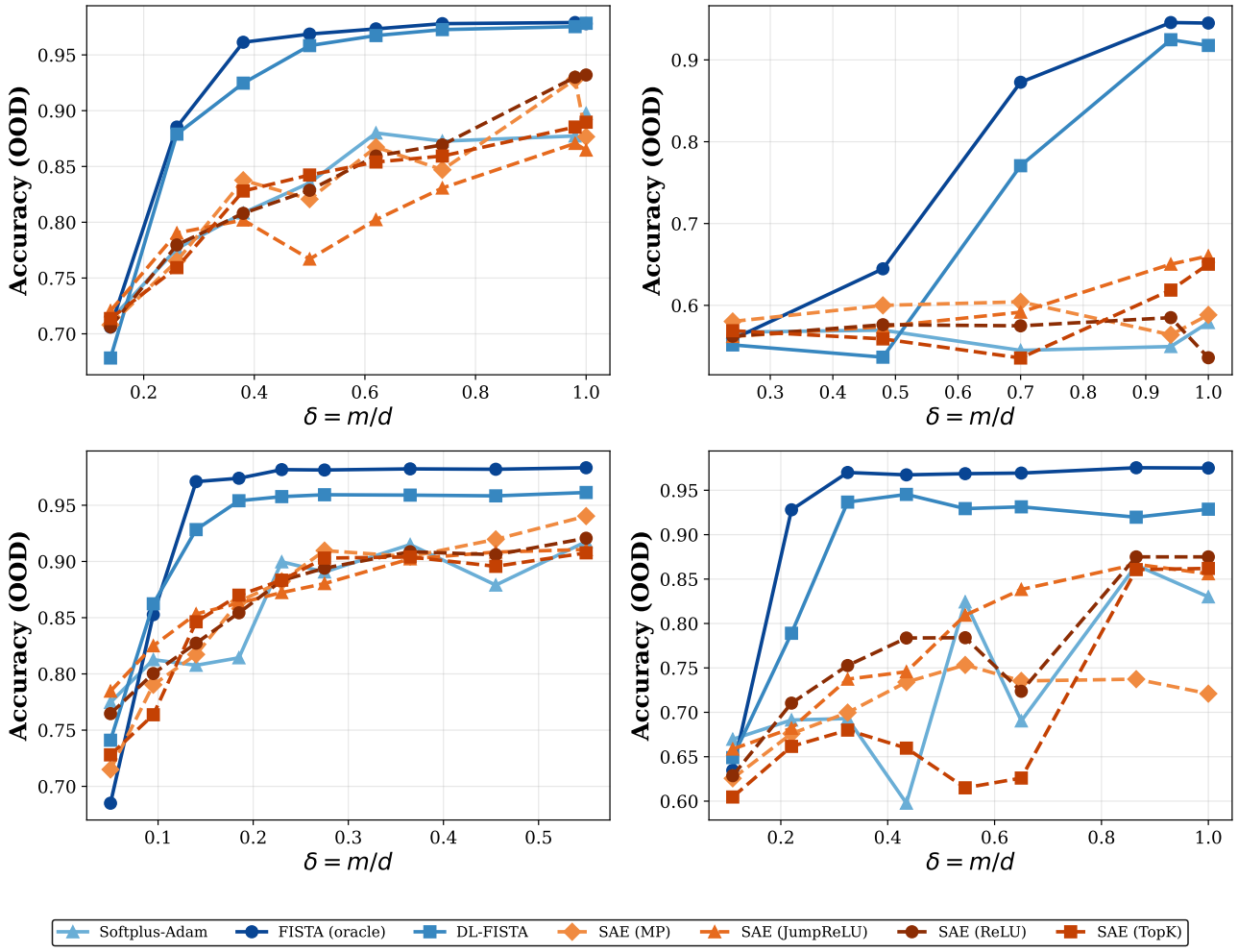


Figure 58. **Phase transition: Accuracy (OOD).** SAE OOD accuracy plateaus while per-sample methods transition sharply. The gap is most severe at moderate δ where compressed sensing succeeds but amortised inference fails.

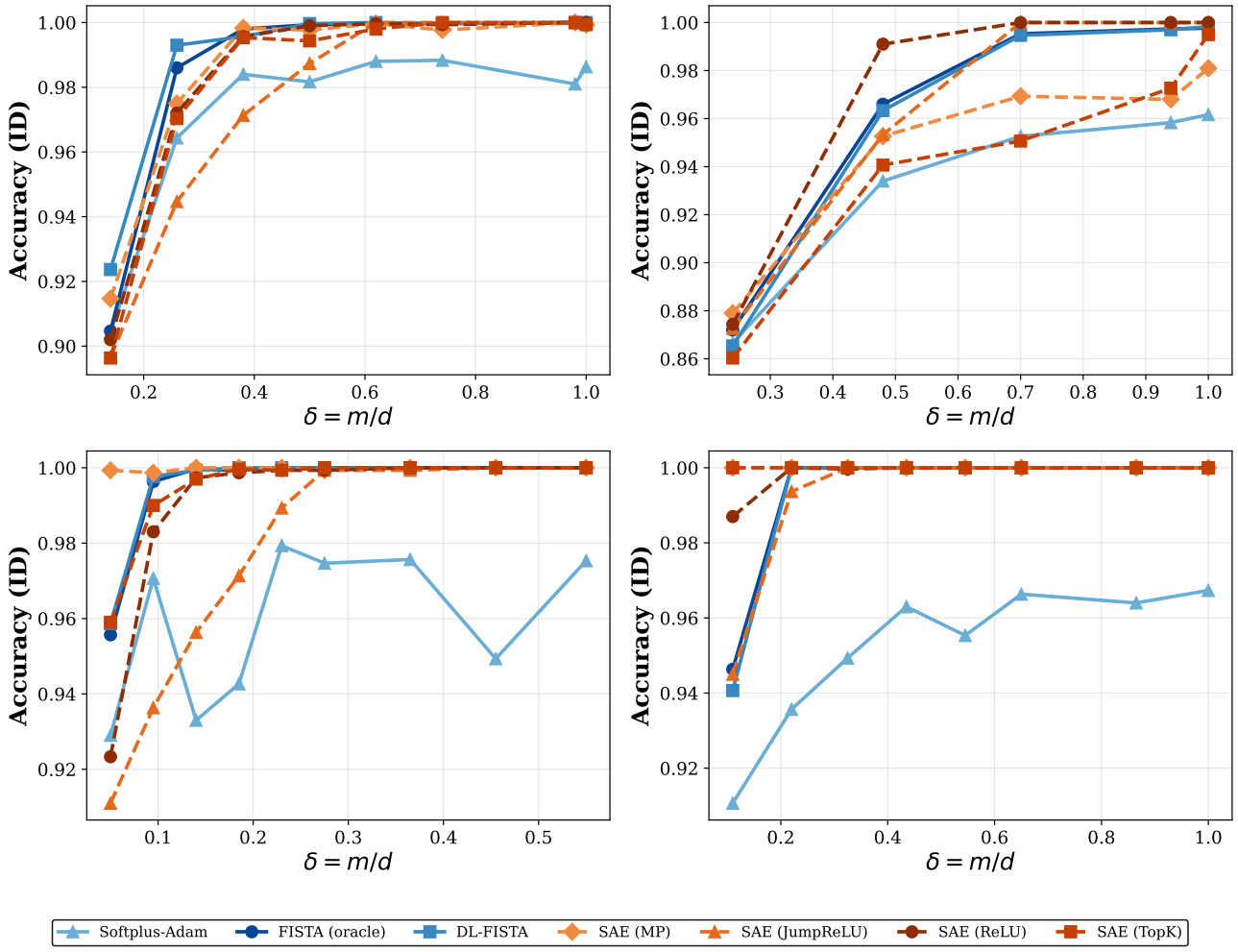


Figure 59. **Phase transition: Accuracy (ID).** ID accuracy is high for most methods once δ is sufficient, but SAEs show more variance and lower peak accuracy than per-sample methods.

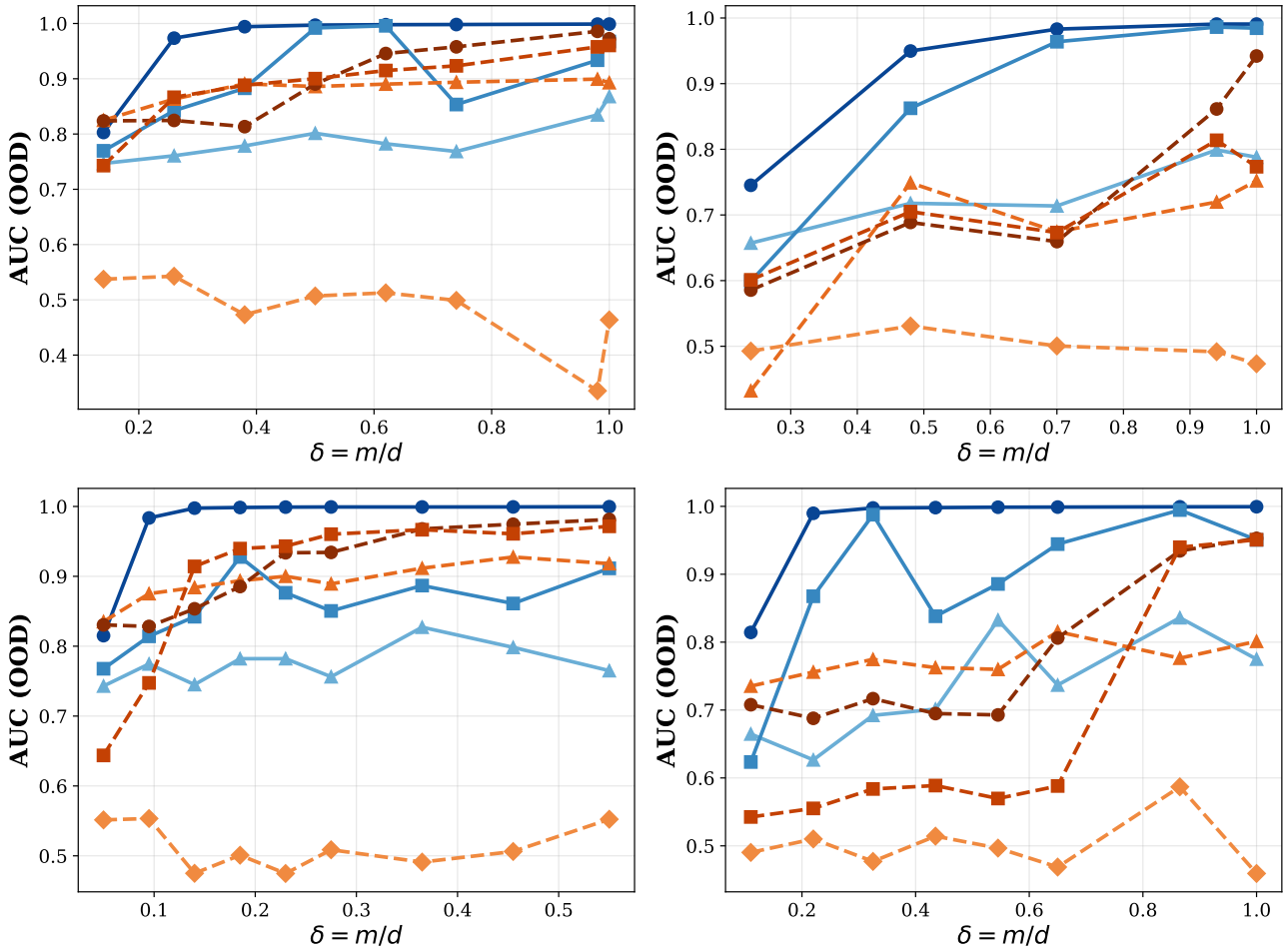


Figure 60. **Phase transition: AUC (OOD).** Per-feature AUC on OOD data confirms the same pattern: per-sample methods achieve near-perfect AUC while SAE features fail to isolate the label under novel compositions.

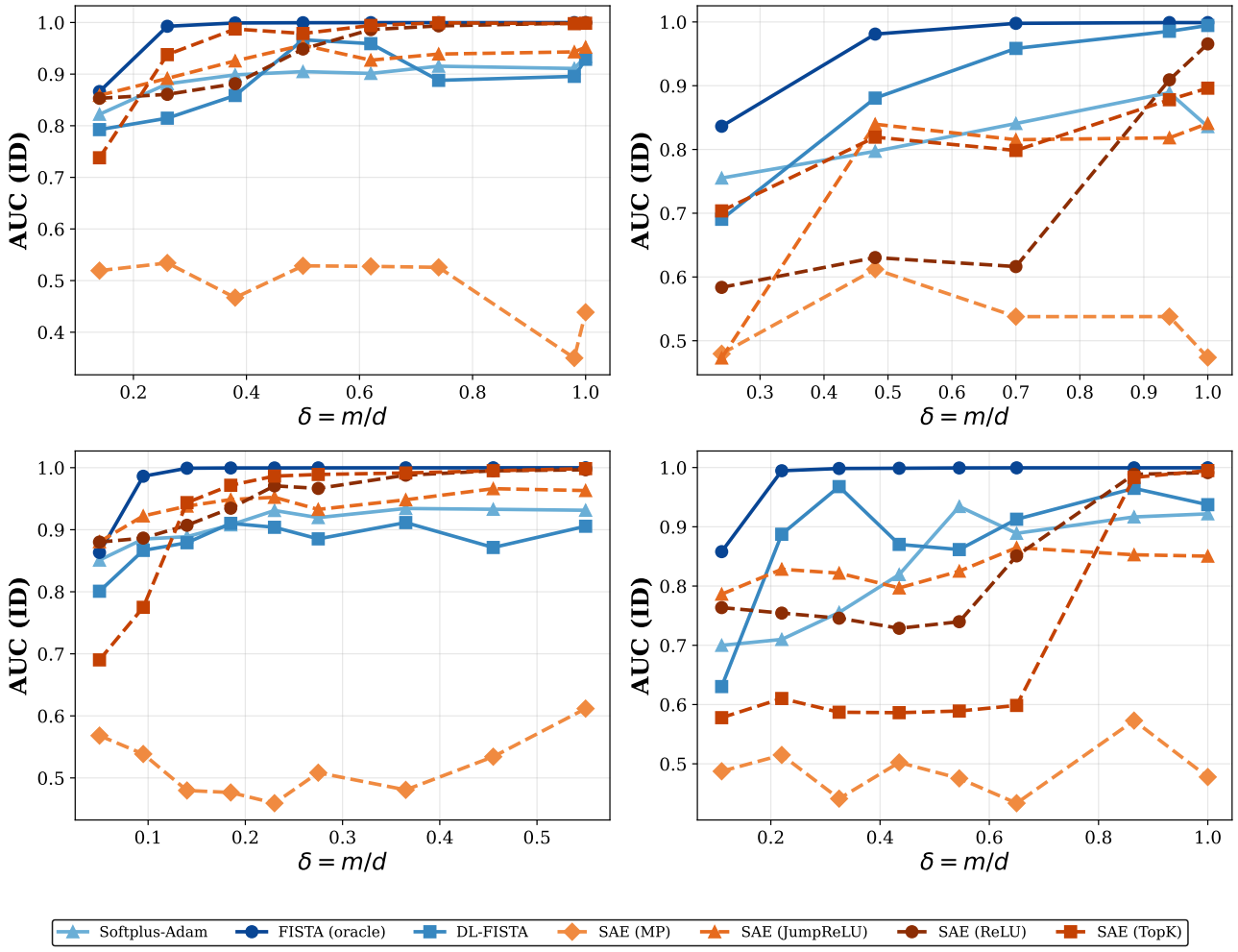


Figure 61. **Phase transition: AUC (ID).** ID AUC is high for most methods, but the gap between per-sample and amortised methods remains visible even in-distribution.

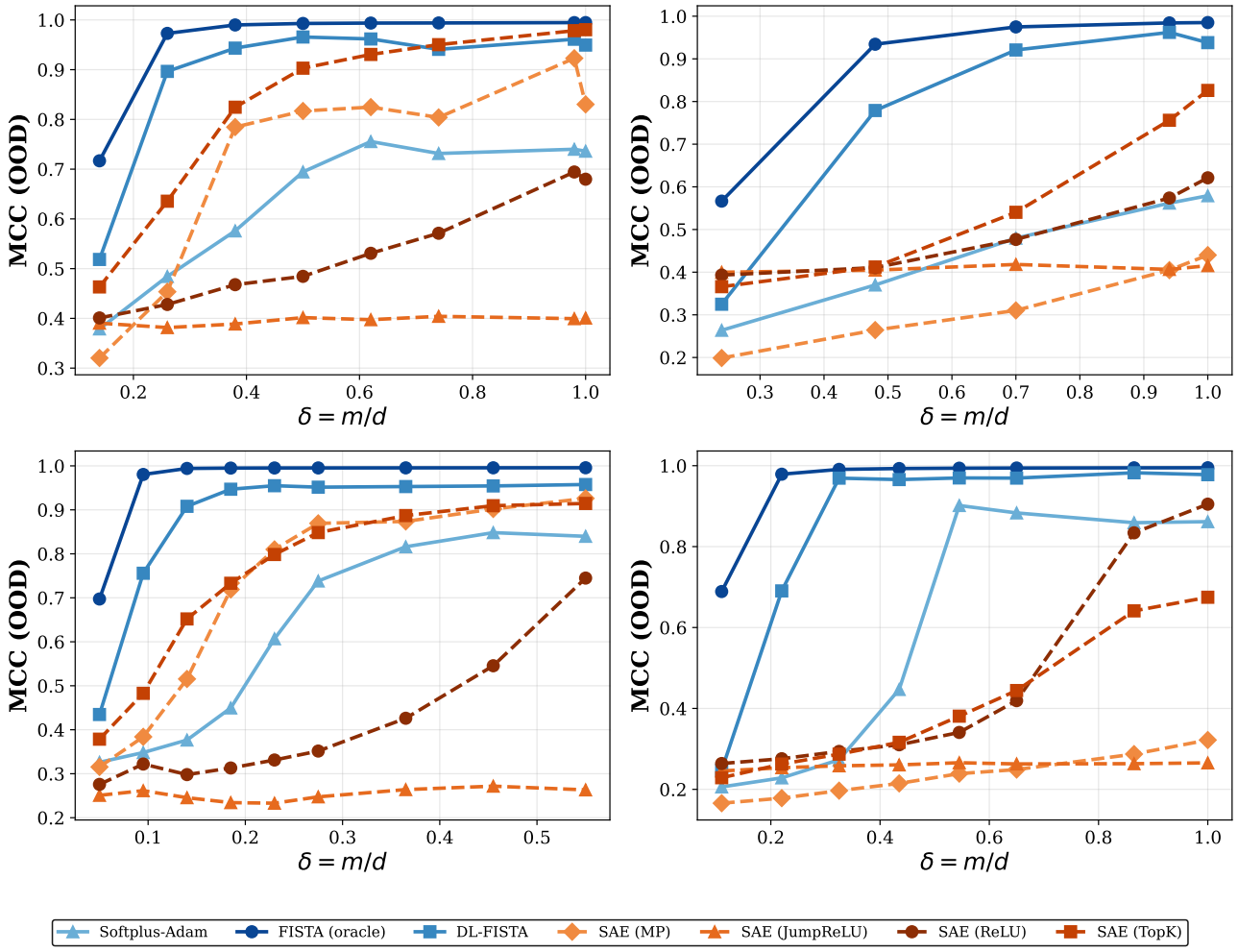


Figure 62. **Phase transition: MCC (OOD).** OOD MCC shows the clearest separation: per-sample methods recover the full latent structure under novel compositions while SAEs fail to identify latents OOD.

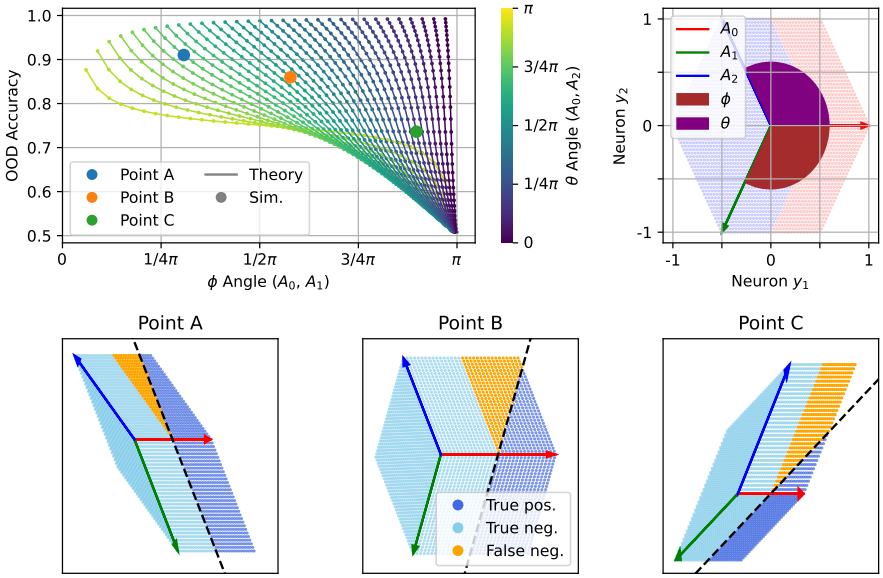


Figure 63. **Theory. Top left,** shows the theoretically predicted accuracy and simulations of a perfect linear classifier, trained and tested (OOD) on distinct latent combinations (see Figure 1). **Top right,** illustrates the geometry of the classification problem (red and blue classes) with the directions of the decoder A columns for each latent and the angles (ϕ, θ) between them. **Bottom,** shows the resulting geometry for three sample points from the first plot.