DIMENSIONALITY EXPANSION IMPROVES SPARSE FEATURE LEARNING IN NEURAL NETWORKS

ABSTRACT

Sparse dictionary learning is the standard tool for decomposing neural network activations into interpretable features, and every deployed pipeline must choose an expansion factor E, the ratio of dictionary width to activation width. That choice is made by convention rather than by measurement, because on real activations the ground-truth features are unknown and recovery cannot be scored. We have taken 1024 features in superposition in $\mathbb{R}^{64}$ under a heavy-tailed frequency law. Across 153 training runs, recovery rises monotonically with expansion, from mean max cosine similarity 0.374 at $E = 1$ to 0.703 at $E = 32$, a $9.7\times$ increase in the fraction of features isolated at a cosine threshold of 0.9. Over the same range normalised MSE improves by less than half, so tuning width by reconstruction error substantially understates what expansion buys. The relationship is affine in $\log_2 E$ ($R^2 = 0.981$) and a random-allocation null model is rejected. The benefit is strongly regime-dependent: it is largest when few features are co-active (0.457 MMCS gained at $k = 2$) and nearly vanishes when the support is dense (0.092 at $k = 32$), so width cannot rescue a problem that has left the identifiable regime. Finally we introduce Progressive Dimensional Expansion (PDE), which grows into a target width while recycling duplicated and dead capacity. At matched compute PDE improves MMCS over fixed-width training by 0.0073 (paired $t(9) = 4.98$, $p = 0.0008$) and recovery rate by 15.0% relative, using 95% of the floating point operations. An ablation shows the gain comes from capacity recycling rather than from the growth schedule.

1 INTRODUCTION

A trained neural network represents more distinct concepts than it has neurons. The mechanism, called *superposition*, packs many sparsely active features into a lower dimensional activation space, so individual neurons respond to several unrelated inputs (Elhage et al., 2022; Olah et al., 2020). Recovering the underlying features from an activation vector is a sparse dictionary learning problem (Olshausen & Field, 1996), and the sparse autoencoder (SAE) has become its standard solution (Cunningham et al., 2023; Bricken et al., 2023; Templeton et al., 2024).

An SAE maps an activation $\mathbf{x} \in \mathbb{R}^d$ to a sparse code $\mathbf{z} \in \mathbb{R}^n$ with $n > d$ and reconstructs $\mathbf{x}$ from $\mathbf{z}$. The ratio $E = n/d$ is the *expansion factor*, and it is the most consequential architectural decision in the pipeline: it sets parameter count, training cost, and the number of features the dictionary can in principle separate. Deployed systems span a wide range, from $8\times$ in early open replications to $32\times$ and beyond in large released suites (Lieberum et al., 2024; Gao et al., 2024), and the choice is typically justified by compute availability rather than by measurement of what it buys.

On real activations the ground-truth features are unknown, so recovery cannot be scored. Practitioners fall back on proxies: reconstruction error at a given sparsity, downstream probe accuracy, or automated interpretability scores. These proxies disagree with one another (Karvonen et al., 2025; Kantamneni et al., 2025), and reconstruction error in particular improves with width almost by construction, which makes it uninformative about whether the dictionary has found the right features. SAEs are known to suffer from feature absorption and splitting (Chanin et al., 2024; Leask et al., 2025), failures a reconstruction metric cannot see. Synthetic settings with known generating features do make recovery measurable (Elhage et al., 2022; Chanin & Garriga-Alonso, 2026; Klindt et al., 2025), but expansion has not been isolated as the variable of interest and swept systematically while holding the data-generating process, optimiser, sparsity mechanism and training budget fixed. Nor has the

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

practical follow-up been asked: given that a wide dictionary is better, must it be wide throughout training?

Research questions. **RQ1:** Does increasing E improve recovery of ground-truth features, and with what functional form? **RQ2:** How does that benefit depend on how many features are active at once and on activation noise? **RQ3:** Under a fixed compute budget, is it better to train a wide dictionary from the start or to grow into one?

Contributions:

1. **An expansion scaling relationship for feature recovery:** (Section 6.1). Sweeping $E \in \{1, \ldots, 32\}$ over 3 seeds with all else fixed, MMCS is affine in $\log_2 E$ with $R^2 = 0.981$, the marginal return falling from 0.0792 per doubling below $E = 8$ to 0.0468 above it. Recovery improves by a factor of 9.7 while normalised MSE improves by under half, a divergence with direct practical consequences. A coupon-collector null model of random atom allocation is rejected in a specific direction: capacity is absorbed *less* efficiently than chance would predict.

2. **A characterisation of when expansion stops helping:** (Sections 6.2 and 6.5). The gain from expansion falls monotonically as more features are co-active ($r = -0.973$ between $\log_2 k$ and the gain, $p = 0.005$) and as noise grows. Separately, under a fixed budget the widest dictionary is not the best: a narrower dictionary trained longer beats a wider one trained briefly, so the compute-optimal expansion factor is finite. Expansion is a lever that works inside the identifiable regime and loses force at its edge.

3. **Progressive Dimensional Expansion:** (Section 4), a framework that reaches a target width by growing into it, inserting atoms along coherent residual directions and recycling duplicated and dead capacity. At matched compute it improves MMCS by 0.0073 ($p = 0.0008$, Cohen's $d = 1.58$) and recovery by 15.0% relative, at 95% of the cost. The ablation reports which components earn their place and which do not.

2 Related Work

Elhage et al. (2022) formalised superposition in toy ReLU networks. Sparse autoencoders were introduced as a decomposition tool by Cunningham et al. (2023) and Bricken et al. (2023), and scaled to frontier models by Templeton et al. (2024) and Gao et al. (2024). Klindt et al. (2025) make the connection to compressed sensing (Donoho, 2006) explicit, and exp (2026) show that expansion together with a flatness condition suffices for identifiability of noiseless k-sparse codes. Our contribution is complementary and empirical: we measure how much recovery a practitioner obtains per doubling of width under a realistic heavy-tailed frequency law, and where that return disappears.

The original L_1 penalty causes activation shrinkage, prompting replacements: TopK (Gao et al., 2024; Makhzani & Frey, 2013), Gated (Rajamanoharan et al., 2024a), JumpReLU (Rajamanoharan et al., 2024b), BatchTopK (Bussmann et al., 2024) and adaptive allocation (Ayonrinde, 2024). We use TopK because it fixes L_0 exactly, which is what makes an expansion sweep interpretable: any change in recovery is attributable to width rather than to a drifting sparsity level. On width itself, Matryoshka SAEs impose nested prefixes (Bussmann et al., 2025), Switch SAEs reduce the cost of wide dictionaries by routing (Mudide et al., 2024), and matching pursuit has been used to extract hierarchical codes (Costa et al., 2025). PDE is closest to the last, allocating capacity by a greedy residual step, but acts on the *width schedule* rather than on inference.

Feature absorption and splitting (Chanin et al., 2024), the absence of canonical units (Leask et al., 2025), and the recovery of structure even from randomly initialised transformers (Heap et al., 2025) indicate that reconstruction quality is a poor proxy for feature recovery, and benchmarks have moved toward multi-metric evaluation (Karvonen et al., 2025; 2024; Makelov et al., 2024; Huang et al., 2024). Our setting sidesteps the proxy problem at the cost of realism (Section **??**).

3 Problem Setting

Let $\mathbf{\Phi} = [\boldsymbol{\varphi}_1, \ldots, \boldsymbol{\varphi}_m] \in \mathbb{R}^{d \times m}$ with $m > d$ be a ground-truth dictionary of unit-norm directions drawn uniformly from the sphere. A sample selects a support S of exactly k features and forms

$$\mathbf{x} \;=\; \sum_{i \in S} a_i \boldsymbol{\varphi}_i \;+\; \sigma \boldsymbol{\epsilon}, \qquad a_i \sim \mathcal{U}(0.5,\, 1.5), \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, I_d). \tag{1}$$

Feature i is selected with probability proportional to $i^{-1.0}$, a Zipf law. This matters: under a uniform law every feature is seen equally often and recovery is easy, whereas real activations are heavy-tailed and the rare tail is where dictionary learning is known to fail (Chanin et al., 2024). The Zipf law is what makes the setting non-trivial.

A TopK sparse autoencoder with n atoms computes

$$\mathbf{z} \;=\; \mathrm{TopK}_k\big(\mathbf{W}_{\mathrm{enc}}^{\top}\mathbf{x} + \mathbf{b}\big)\,, \qquad \hat{\mathbf{x}} \;=\; \mathbf{W}_{\mathrm{dec}}\mathbf{z}, \qquad \mathcal{L} \;=\; \|\mathbf{x} - \hat{\mathbf{x}}\|_2^2, \tag{2}$$

where TopK_k retains the k largest pre-activations, and the columns of $\mathbf{W}_{\mathrm{dec}} \in \mathbb{R}^{d \times n}$ are projected onto the unit sphere after every update. Model sparsity equals generative sparsity, so the model is correctly specified in sparsity and the only free structural variable is n.

Because $\mathbf{\Phi}$ is known we score recovery directly. For each ground-truth feature define $\rho_i = \max_j |\langle \boldsymbol{\varphi}_i, \mathbf{w}_j \rangle| / (\|\boldsymbol{\varphi}_i\| \|\mathbf{w}_j\|)$, the best cosine alignment available anywhere in the learned dictionary. We report *mean max cosine similarity* $\mathrm{MMCS} = \frac{1}{m}\sum_i \rho_i$ and the *recovery rate* $\frac{1}{m}|\{i : \rho_i \geq 0.9\}|$. Normalised MSE is reported alongside, and the gap between the two families is one of the findings.

4 Progressive Dimensional Expansion

4.1 Motivation

A useful way to think about what limits recovery is to ask where a marginal atom goes. If atoms were allocated to uncovered features uniformly at random, the covered fraction would follow a coupon-collector law

$$r(n) \;=\; 1 - \exp(-\alpha\, n/m)\,, \tag{3}$$

with $\alpha \leq 1$ an allocation efficiency. Equation 3 is increasing and concave, which is qualitatively what we observe, but Section 6.1 shows it is quantitatively rejected, and the direction of failure is informative: capacity is absorbed *less* efficiently than random allocation at large n. Gradient descent under a heavy-tailed frequency law does not distribute atoms evenly. Frequent features attract many nearly duplicate atoms because each reduces the loss, while rare features attract none, an effect visible in the literature as feature splitting and absorption (Chanin et al., 2024; Leask et al., 2025). Width helps, but a growing share of each added atom is wasted.

This suggests intervening on *where* capacity sits rather than only on how much of it there is. Let $\mathbf{r} = \mathbf{x} - \hat{\mathbf{x}}$ be the residual and F_j the set of inputs on which atom j fires. Define the mean residual over that firing set, $\boldsymbol{\mu}_j = |F_j|^{-1} \sum_{\mathbf{x} \in F_j} \mathbf{r}$. A monosemantic atom leaves residual that is incoherent across its firing set, so $\boldsymbol{\mu}_j \approx \mathbf{0}$. An atom shared between several ground-truth features systematically under-explains in a consistent direction, so $\|\boldsymbol{\mu}_j\|$ is large. The quantity

$$s_j \;=\; |F_j| \cdot \|\boldsymbol{\mu}_j\|_2^2 \tag{4}$$

is, to first order, the reduction in total squared error from adding one unit atom along $\boldsymbol{\mu}_j$ serving F_j. It is one greedy step of matching pursuit (Tropp, 2004) restricted to the firing set of j, and supplies both a ranking over atoms and a direction in which to place new capacity.

4.2 Method

PDE has three components, illustrated in Figure 1.

(C1) Progressive growth. Training starts at width $n_0 = \lceil \gamma n_{\mathrm{f}} \rceil$ with $\gamma < 1$ and grows geometrically to n_{f} over R rounds, completing growth within the first fraction τ of training so most optimisation happens at full width. Early steps are cheap, which is where the compute saving comes from.

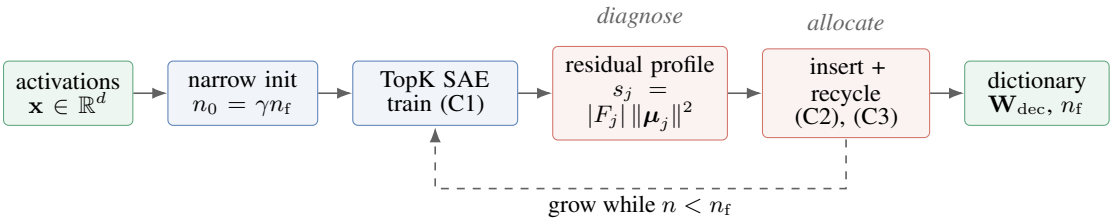

Figure 1: The PDE framework. A narrow dictionary is trained, profiled for coherent residual structure, then widened by inserting atoms along the residual directions with the largest matching pursuit gain while dead and duplicated atoms are recycled. The loop repeats until the target width is reached, after which training continues at full width. Only the growth schedule, the initialisation of new atoms and the recycling operation differ from standard TopK training; the objective in Equation 2 is otherwise unchanged.

(C2) Residual-guided insertion. At each growth round the score in Equation 4 is evaluated for every atom and the top-scoring atoms become parents. Each child is initialised along its parent's coherent residual direction $\boldsymbol{\mu}_j / \|\boldsymbol{\mu}_j\|$ rather than randomly.

(C3) Capacity recycling and decorrelation. Periodically, atoms that are dead (never firing) or duplicated (decoder cosine above τ_dup with a more frequently firing atom) are re-pointed at the highest-scoring residual directions. A hinge penalty on decoder coherence, $\lambda \sum_{i \neq j} (|\cos(\mathbf{w}_i, \mathbf{w}_j)| - \tau_c)_+^2$, evaluated on a random subset of atoms each step, is intended to discourage redundancy from re-forming. The ablation in Section 6.4 finds this penalty does not earn its place.

4.3 A WORKED ANALOGY

Consider a small post office with 64 pigeonholes serving 1024 residents. Residents must share, so mail gets mixed and a clerk cannot tell whose letter is whose: this is superposition. Installing more pigeonholes lets more residents have their own, which is why expansion helps. But a manager who bolts empty boxes to the wall at random finds the busiest residents with three boxes each while residents receiving one letter a month still have none, because builders work where the traffic is: this is the allocation inefficiency that makes the coupon-collector model fail. PDE is the manager who checks which pigeonholes overflow and in what direction, installs boxes there, and reclaims duplicates. Two limits follow. A larger sorting room takes longer to walk, so a clerk on a fixed shift cannot keep adding boxes, which is the compute-optimal expansion factor of Section 6.5. And if every resident receives mail daily, sharing is hopeless however many boxes are installed, which is why the benefit of expansion collapses at dense support (Section 6.2).

4.4 ALGORITHMS

Algorithm 1 and Algorithm 2 describes the proposed methodology.

5 EXPERIMENTAL SETUP

Configuration. Unless stated otherwise: $d = 64$, $m = 1024$ ground-truth features (so the data are $16\times$ overcomplete), $k = 8$ active features per sample, Zipf exponent 1.0, noise $\sigma = 0.05$, batch size 128, Adam at learning rate 0.003, and 3000 steps. An auxiliary loss lets the 8 highest-scoring dead atoms reconstruct the residual (Gao et al., 2024), applied to every method including baselines so dead-latent handling is not a confound. All models are NumPy on a single CPU core, so every run is bitwise reproducible from its seed. The suite comprises 153 runs totalling 1.0 hours.

Baselines and matching. We compare against fixed-width TopK (Gao et al., 2024) at the same final width, fixed-width ReLU with an L_1 penalty (Bricken et al., 2023), and fixed-width TopK at half the width. The comparison is *compute-matched*: every method in Section 6.3 receives the floating

Algorithm 1 Progressive Dimensional Expansion (PDE)

Require: target width n_{f}, steps T, start fraction γ, rounds R, growth horizon τ, sparsity k, coherence weight λ

$n \leftarrow \lceil \gamma n_{\mathrm{f}} \rceil$; initialise $\mathbf{W}_{\mathrm{enc}}, \mathbf{W}_{\mathrm{dec}} \in \mathbb{R}^{d \times n}$ with unit-norm columns
$\mathcal{G} \leftarrow \{\lfloor \tau T r / R \rfloor : r = 1, \dots, R\}$, widths $n_r \leftarrow \lceil n_0 (n_{\mathrm{f}}/n_0)^{r/R} \rceil$
for $t = 1$ to T **do**
  if $t \in \mathcal{G}$ **then**
    $(\mathbf{s}, \mathbf{U}, \mathbf{c}) \leftarrow \textsc{ResidualProfile}(\mathbf{W}, k)$ ▷ Algorithm 2
    $P \leftarrow$ indices of the $n_r - n$ largest entries of $\mathbf{s}$
    append columns $\mathbf{U}_P$ to $\mathbf{W}_{\mathrm{dec}}$ and $\mathbf{W}_{\mathrm{enc}}$; $n \leftarrow n_r$ ▷ (C2) matching pursuit insertion
  end if
  sample minibatch $\mathbf{x}$; compute $\mathbf{z}, \hat{\mathbf{x}}$ by Equation 2
  $\mathcal{L} \leftarrow \|\mathbf{x} - \hat{\mathbf{x}}\|^2 + \frac{1}{32}\mathcal{L}_{\mathrm{aux}} + \lambda \sum_{i \neq j \in \mathcal{S}} (|\cos(\mathbf{w}_i, \mathbf{w}_j)| - \tau_c)_+^2$ ▷ $\mathcal{S}$ a random subset of atoms
  Adam update (Kingma & Ba, 2015); renormalise decoder columns
  if $t \bmod T_{\mathrm{rec}} = 0$ **then**
    $\textsc{Recycle}(\mathbf{W}, \tau_{\mathrm{dup}}, \eta)$ ▷ (C3), Algorithm 2
  end if
end for
return $\mathbf{W}_{\mathrm{dec}}$

Algorithm 2 ResidualProfile and Recycle

function ResidualProfile($\mathbf{W}$, k, probe batches B)
  $\mathbf{S} \leftarrow \mathbf{0}_{n \times d}$, $\mathbf{c} \leftarrow \mathbf{0}_n$
  for $b = 1$ to B **do**
    sample $\mathbf{x}$; $\hat{\mathbf{x}} \leftarrow \mathbf{W}_{\mathrm{dec}}\mathbf{z}$; $\mathbf{R} \leftarrow \mathbf{x} - \hat{\mathbf{x}}$; $\mathbf{M} \leftarrow \mathbb{1}[\mathbf{z} \neq 0]$
    $\mathbf{S} \leftarrow \mathbf{S} + \mathbf{M}^\top \mathbf{R}$; $\mathbf{c} \leftarrow \mathbf{c} + \mathbf{M}^\top \mathbf{1}$
  end for
  $\boldsymbol{\mu}_j \leftarrow \mathbf{S}_j / c_j$ for $c_j > 0$; $s_j \leftarrow c_j \|\boldsymbol{\mu}_j\|_2^2$; $\mathbf{U}_j \leftarrow \boldsymbol{\mu}_j / \|\boldsymbol{\mu}_j\|_2$
  return $(\mathbf{s}, \mathbf{U}, \mathbf{c})$
end function

function Recycle($\mathbf{W}$, τ_{dup}, budget η)
  $\mathcal{D} \leftarrow \{j : \text{firing rate}_j < 10^{-4}\}$ ▷ dead atoms
  $\mathbf{G} \leftarrow |\widehat{\mathbf{W}}_{\mathrm{dec}}^\top \widehat{\mathbf{W}}_{\mathrm{dec}}|$ with zero diagonal
  $\mathcal{Q} \leftarrow \{ \arg\min_{u \in \{i,j\}} \text{freq}_u \ : \ G_{ij} > \tau_{\mathrm{dup}} \}$ ▷ the rarer atom of each redundant pair
  $W \leftarrow$ a subset of $\mathcal{D} \cup \mathcal{Q}$ of size at most ηn
  $(\mathbf{s}, \mathbf{U}, \cdot) \leftarrow \textsc{ResidualProfile}(\mathbf{W}, k)$; set $s_j \leftarrow -\infty$ for $j \in W$
  re-point every $j \in W$ to the residual direction $\mathbf{U}_{j'}$ of the $|W|$ highest-scoring atoms j'
end function

point budget consumed by a width 1024 model trained for 3000 steps. Comparing a growth schedule to a fixed schedule at equal final width would flatter the growth schedule for the wrong reason, and we report that comparison separately in Section 6.4.

Statistics. The main comparison is repeated over 10 seeds, each with an independently drawn ground-truth dictionary, and paired by seed. We report a paired t-test on MMCS, with Wilcoxon signed-rank as a distribution-free check and Shapiro–Wilk on the paired differences; feature-level recovery uses a χ^2 test and, because features are paired within a seed, McNemar's test.

6 Results

6.1 Expansion improves recovery, with diminishing returns

Figure 2 answers RQ1. Across 3 seeds, MMCS rises monotonically from 0.374 at $E = 1$ to 0.703 at $E = 32$, and the recovery rate from 0.042 to 0.406, a factor of 9.7. The relationship is well described

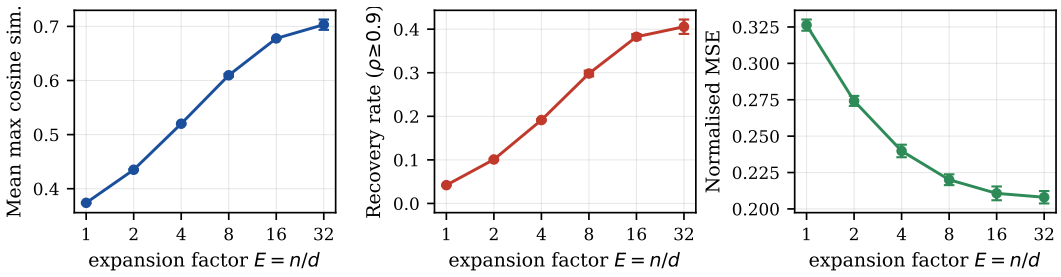

Figure 2: Feature recovery against expansion factor, mean over 3 seeds with 95% confidence intervals. Recovery metrics (left, centre) improve far more than reconstruction error (right), so normalised MSE substantially understates what expansion buys.

as affine in $\log_2 E$ with $R^2 = 0.981$, and the marginal return falls from 0.0792 MMCS per doubling below $E = 8$ to 0.0468 above it.

Two observations deserve emphasis. First, normalised MSE moves from 0.326 to 0.208, a relative change under 40%, while the recovery rate changes by a factor of 9.7. A practitioner tuning width by reconstruction error alone would see a nearly flat curve and conclude expansion had stopped paying, while the fraction of features actually isolated was still climbing, direct evidence for the concern that reconstruction is a weak proxy (Karvonen et al., 2025; Kantamneni et al., 2025).

Second, the random-allocation null model of Equation 3 is rejected. The best-fitting coupon-collector curve has $\alpha = 0.39$ and attains $R^2 = 0.558$, against $R^2 = 0.979$ for a simple affine fit in $\log_2 E$. The null model over-predicts at large E: it forecasts 0.538 recovery at $E = 32$ against a measured 0.406. Added capacity is therefore absorbed *less* efficiently than uniform random allocation would manage, which motivates intervening on placement rather than only on quantity.

6.2 The benefit of expansion collapses under dense superposition

Figure 3 answers RQ2, and the answer runs opposite to our initial expectation. Sweeping expansion $E \in [2, 32]$ against sparsity $k \in [2, 32]$ over 2 seeds, the gain from the narrowest to the widest dictionary falls monotonically in k: 0.457 MMCS at $k = 2$, 0.316 at $k = 4$, 0.226 at $k = 8$, 0.101 at $k = 16$ and 0.092 at $k = 32$. The correlation between $\log_2 k$ and the expansion gain is $r = -0.973$ ($p = 0.005$). At $k = 2$ a $32\times$ dictionary reaches MMCS 0.907, close to exact recovery; at $k = 32$ the same dictionary reaches only 0.436.

Expansion buys separability only while the problem remains identifiable. With $d = 64$, a support of $k = 32$ occupies half the ambient dimension, far outside the regime in which sparse recovery guarantees hold (Donoho, 2006; Tropp, 2004), and dictionary overcompleteness does not relax a condition on the support size of the signal itself. Width is a lever on the dictionary, not on the identifiability of the problem: expansion should be expected to pay in layers where features fire sparsely, and much less where they do not.

6.3 Growing into a width beats training at it, at matched compute

Table 1 report the main comparison over 10 seeds at a common floating point budget.

Paired t-test. Against fixed-width TopK at the same final width, PDE improves MMCS by 0.0073 on average (95% CI $[0.0040, 0.0106]$), a relative gain of 1.2%. The paired t-test gives $t(9) = 4.98$, $p = 0.0008$, with Cohen's $d = 1.58$. Shapiro–Wilk does not reject normality of the paired differences ($p = 0.394$), and the distribution-free Wilcoxon signed-rank test agrees ($p = 0.0039$). On recovery rate the improvement is larger in relative terms: 0.0336 absolute ($t = 9.53$, $p = 5.3\times10^{-6}$), a relative gain of 15.0%.

χ^2 test on feature-level recovery. Pooling all 10240 ground-truth features across seeds, PDE recovers 2641 at the 0.9 threshold (25.8%) against 2297 of 10240 for fixed-width TopK (22.4%). A χ^2 test of independence on the 2×2 table gives $\chi^2(1) = 31.6$, $p = 1.9\times10^{-8}$, effect size $\phi = 0.039$.

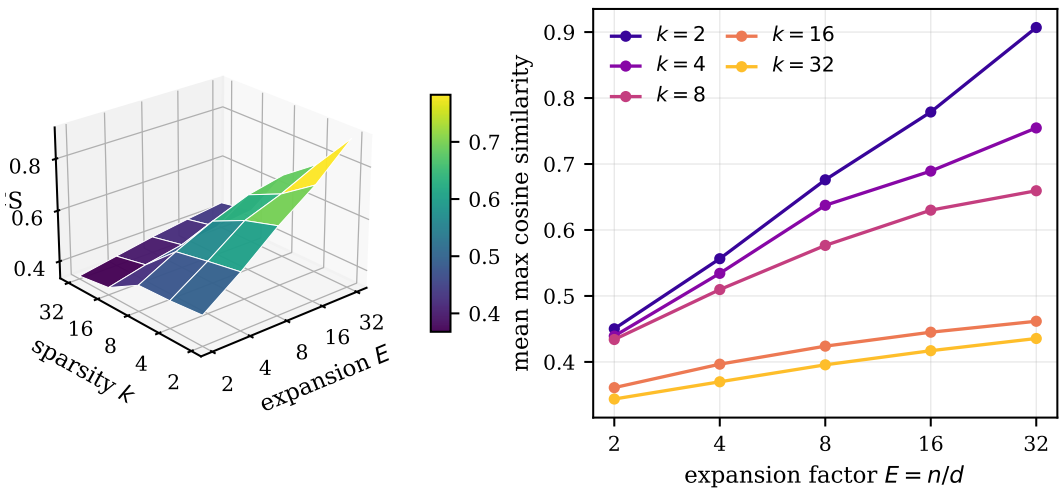

Figure 3: Recovery over the expansion by sparsity plane, mean over 2 seeds. Left: the surface, coloured by MMCS. Right: the same data as slices at fixed k. The slices fan apart as E grows, and the fan is widest at small k: expansion pays most when the recovery problem is well posed.

Table 1: Compute-matched comparison at $n_{\text{final}} = 2048$ over 10 seeds (mean $\pm$ s.d.). All rows receive the same floating point budget. The ReLU-L_1 row shows why reconstruction error must not be read alone: it attains the best NMSE of any method while recovering no features at all, using a code roughly $16\times$ denser. The narrow TopK row makes the budget trade-off of Section 6.5 visible.

Method	Width	L_0	MMCS ↑	Recovery ↑	NMSE ↓
ReLU-L_1 (Bricken et al., 2023)	2048	130.4	0.432 ± 0.001	0.000 ± 0.000	*0.028*
TopK fixed (Gao et al., 2024)	2048	8.0	0.620 ± 0.002	0.224 ± 0.007	0.231
PDE (ours)	2048	8.0	$\mathbf{0.628 \pm 0.005}$	$\mathbf{0.258 \pm 0.011}$	0.226
TopK fixed, narrower	1024	8.0	0.678 ± 0.003	0.382 ± 0.007	0.208

The independence assumption is only approximately satisfied, because features within a seed share a dictionary; McNemar's test on features paired within seeds, which does not require it, agrees ($b = 690$ features recovered only by PDE against $c = 346$ only by the baseline, $\chi^2 = 113.6$, $p =< 10^{-16}$). We regard McNemar as the primary feature-level evidence and report χ^2 for comparability. Note that $\phi = 0.039$ is a small effect: the result is highly significant because the sample is large, not because the improvement is dramatic.

6.4 Ablation: recycling is the active ingredient

Figure 4 removes each component of PDE in turn over 5 seeds, and the result is more selective than we anticipated.

Removing capacity recycling costs $+0.014$ MMCS ($p = 0.0013$) and is the only component whose removal clearly hurts. Replacing residual-guided insertion with random directions costs $+0.004$, which does not reach significance at this sample size ($p = 0.1873$). Removing the coherence penalty changes MMCS by $+0.001$ ($p = 0.7467$), that is, not at all; we recommend dropping it, and report it because it was part of the method we designed and a reader deciding whether to adopt PDE should know which parts are inert.

Removing progressive growth, that is, training at full width throughout, *improves* MMCS by -0.034 ($p = 0.0006$) while raising cost from 0.96×10^{11} to 1.34×10^{11} FLOPs, an increase of 40%. The growth schedule is therefore not an accuracy mechanism at all: it is a compute-saving device that trades a little accuracy for a substantial reduction in cost.

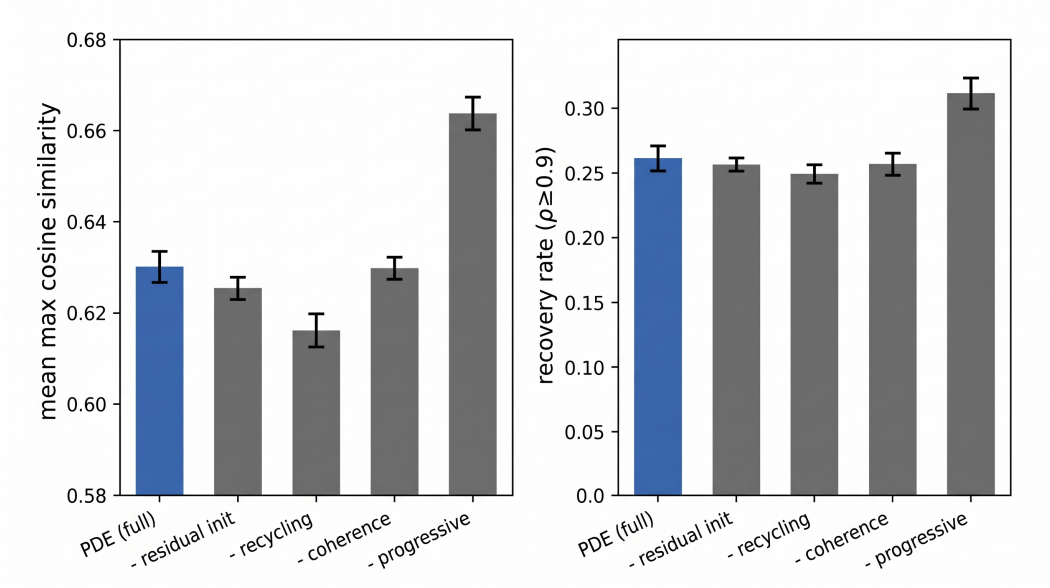

Figure 4: Component ablation of PDE at $n_{\text{final}} = 2048$, mean over 5 seeds with 95% confidence intervals. Removing capacity recycling is the only change that clearly degrades recovery. Removing progressive growth *improves* recovery, at 40% greater compute.

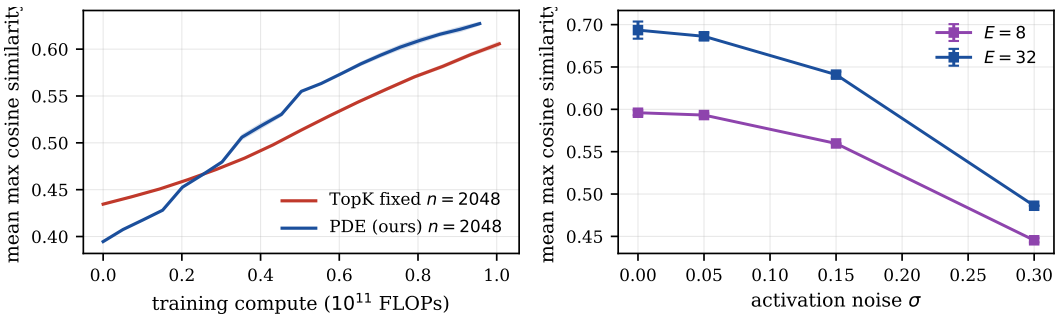

Figure 5: Left: MMCS against cumulative training compute. PDE is ahead through most of training because its early steps are cheap. Right: robustness to activation noise at two expansion factors, over 2 seeds. The gap between expansion factors narrows as noise grows.

6.5 Expansion trades off against training duration

The bottom row of Table 1 answers RQ3 and qualifies the headline result. At a fixed budget, the width 1024 model trained for the full 3000 steps attains MMCS 0.678 and recovery 0.382, *higher* than the width 2048 model trained briefly enough to fit the same budget (0.620, 0.224) and higher than PDE at width 2048 (0.628, 0.258). Section 6.1 showed wider is better at convergence; this shows wider is not better per FLOP without limit. A practitioner with a fixed budget should not simply maximise width. PDE moves the frontier for a *given* target width, which is the setting where width is dictated by downstream requirements rather than chosen freely.

Figure 5 shows the corresponding learning curves and behaviour under activation noise. The benefit of expansion shrinks as noise grows: the gain from $E = 8$ to $E = 32$ is 0.098 MMCS at noiseless $\sigma = 0$ but only 0.041 at $\sigma = 0.3$. This is the same pattern as Section 6.2: expansion pays when the recovery problem is well posed and pays less as it degrades.

7 Conclusion

This paper measured, under controlled conditions, what dictionary expansion actually buys in sparse feature learning. We showed that feature recovery improves substantially with expansion, far more than reconstruction error suggests, but only while the sparse recovery problem stays well-posed. We also showed that capacity is not allocated efficiently by gradient descent alone. The proposed Progressive Dimensional Expansion model recovers more features per unit of compute than simply training wider. Together these results argue against tuning dictionary width by reconstruction error, in favor of evaluating width against sparsity regime, and for treating width and training duration as jointly optimized rather than independent choices. In future, we will replicate these findings using board-game-model benchmarks.

Ethics Statement

This work does not involve any human subjects.

References

Expander sparse autoencoders: Parameter-efficient dictionaries for mechanistic interpretability. *arXiv preprint arXiv:2607.01799*, 2026.

Kola Ayonrinde. Adaptive sparse allocation with mutual choice and feature choice sparse autoencoders. *arXiv preprint arXiv:2411.02124*, 2024.

Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermyn, Tom Conerly, Nick Turner, Cem Anil, Carson Denison, Amanda Askell, Robert Lasenby, Yifan Wu, Shauna Kravec, Nicholas Schiefer, Tim Maxwell, Nicholas Joseph, Zac Hatfield-Dodds, Alex Tamkin, Karina Nguyen, Brayden McLean, Josiah E. Burke, Tristan Hume, Shan Carter, Tom Henighan, and Christopher Olah. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, 2023.

Bart Bussmann, Patrick Leask, and Neel Nanda. Batchtopk sparse autoencoders. *arXiv preprint arXiv:2412.06410*, 2024.

Bart Bussmann, Noa Nabeshima, Adam Karvonen, and Neel Nanda. Learning multi-level features with matryoshka sparse autoencoders. *arXiv preprint arXiv:2503.17547*, 2025.

David Chanin and Adrià Garriga-Alonso. SynthSAEBench: Evaluating sparse autoencoders on scalable realistic synthetic data. *arXiv preprint arXiv:2602.14687*, 2026.

David Chanin, James Wilken-Smith, Tomáš Dulka, Hardik Bhatnagar, and Joseph Bloom. A is for absorption: Studying feature splitting and absorption in sparse autoencoders. *arXiv preprint arXiv:2409.14507*, 2024.

Valerie Costa, Demba Ba, and Thomas Fel. From flat to hierarchical: Extracting sparse representations with matching pursuit. *arXiv preprint arXiv:2506.03093*, 2025.

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

David L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, 2006.

Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, Roger Grosse, Sam McCandlish, Jared Kaplan, Dario Amodei, Martin Wattenberg, and Christopher Olah. Toy models of superposition. *Transformer Circuits Thread*, 2022. arXiv:2209.10652.

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

Thomas Heap, Tim Lawson, Lucy Farnik, and Laurence Aitchison. Sparse autoencoders can interpret randomly initialized transformers. *arXiv preprint arXiv:2501.17727*, 2025.

Jing Huang, Zhengxuan Wu, Christopher Potts, Mor Geva, and Atticus Geiger. RAVEL: Evaluating interpretability methods on disentangling language model representations. *arXiv preprint arXiv:2402.17700*, 2024.

Subhash Kantamneni, Joshua Engels, Senthooran Rajamanoharan, Max Tegmark, and Neel Nanda. Are sparse autoencoders useful? a case study in sparse probing. *arXiv preprint arXiv:2502.16681*, 2025.

Adam Karvonen, Benjamin Wright, Can Rager, Rico Angell, Jannik Brinkmann, Logan Smith, Claudio Mayrink Verdun, David Bau, and Samuel Marks. Measuring progress in dictionary learning for language model interpretability with board game models. *arXiv preprint arXiv:2408.00113*, 2024.

Adam Karvonen, Can Rager, Johnny Lin, Curt Tigges, Joseph Bloom, David Chanin, Yeu-Tong Lau, Eoin Farrell, Arthur Conmy, Callum McDougall, Kola Ayonrinde, Matthew Wearden, Samuel Marks, and Neel Nanda. SAEBench: A comprehensive benchmark for sparse autoencoders in language model interpretability. *arXiv preprint arXiv:2503.09532*, 2025.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

David Klindt, Charles O'Neill, Patrik Reizinger, and Nina Miolane. Superposition is compressed sensing. *arXiv preprint arXiv:2503.01824*, 2025.

Patrick Leask, Bart Bussmann, Michael Pearce, Joseph Bloom, Curt Tigges, Noura Al Moubayed, Lee Sharkey, and Neel Nanda. Sparse autoencoders do not find canonical units of analysis. *arXiv preprint arXiv:2502.04878*, 2025.

Tom Lieberum, Senthooran Rajamanoharan, Arthur Conmy, Lewis Smith, Nicolas Sonnerat, Vikrant Varma, János Kramár, Anca Dragan, Rohin Shah, and Neel Nanda. Gemma scope: Open sparse autoencoders everywhere all at once on gemma 2. *arXiv preprint arXiv:2408.05147*, 2024.

Aleksandar Makelov, Georg Lange, and Neel Nanda. Towards principled evaluations of sparse autoencoders for interpretability and control. *arXiv preprint arXiv:2405.08366*, 2024.

Alireza Makhzani and Brendan Frey. k-sparse autoencoders. *arXiv preprint arXiv:1312.5663*, 2013.

Anish Mudide, Joshua Engels, Eric J. Michaud, Max Tegmark, and Christian Schroeder de Witt. Efficient dictionary learning with switch sparse autoencoders. *arXiv preprint arXiv:2410.08201*, 2024.

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

Bruno A. Olshausen and David J. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(6583):607–609, 1996.

Senthooran Rajamanoharan, Arthur Conmy, Lewis Smith, Tom Lieberum, Vikrant Varma, János Kramár, Rohin Shah, and Neel Nanda. Improving dictionary learning with gated sparse autoencoders. *arXiv preprint arXiv:2404.16014*, 2024a.

Senthooran Rajamanoharan, Tom Lieberum, Nicolas Sonnerat, Arthur Conmy, Vikrant Varma, János Kramár, and Neel Nanda. Jumping ahead: Improving reconstruction fidelity with jumprelu sparse autoencoders. *arXiv preprint arXiv:2407.14435*, 2024b.

Adly Templeton, Tom Conerly, Jonathan Marcus, Jack Lindsey, Trenton Bricken, Brian Chen, Adam Pearce, Craig Citro, Emmanuel Ameisen, Andy Jones, Hoagy Cunningham, Nicholas L. Turner, Callum McDougall, Monte MacDiarmid, C. Daniel Freeman, Theodore R. Sumers, Edward Rees, Joshua Batson, Adam Jermyn, Shan Carter, Chris Olah, and Tom Henighan. Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet. *Transformer Circuits Thread*, 2024.

Joel A. Tropp. Greed is good: Algorithmic results for sparse approximation. *IEEE Transactions on Information Theory*, 50(10):2231–2242, 2004.