Superposition as Lossy Compression – Measure with Sparse Autoencoders and Connect to Adversarial Vulnerability

Anonymous authors
Paper under double-blind review

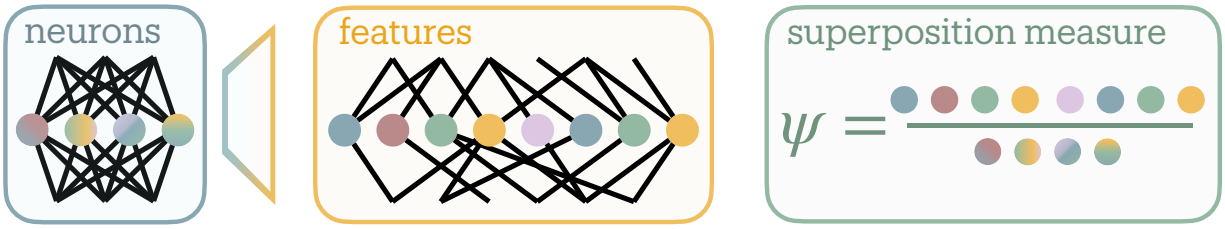
Abstract

Neural networks achieve remarkable performance through *superposition*—encoding multiple features as overlapping directions in activation space rather than dedicating individual neurons to each feature. This phenomenon challenges interpretability: when neurons respond to multiple unrelated concepts, understanding network behavior becomes difficult. Yet despite its importance, we lack principled methods to measure superposition. We present an information-theoretic framework measuring a neural representation's *effective degrees of freedom*. We apply the Shannon entropy to sparse autoencoder activations to compute the number of effective features as the the minimum number of neurons needed for interference-free encoding. Equivalently, this measures how many "virtual neurons" the network simulates through superposition. When networks encode more effective features than they have actual neurons, they must accept interference as the price of compression. Our metric strongly correlates with ground truth in toy models, detects minimal superposition in algorithmic tasks (effective features approximately equal neurons), and reveals systematic reduction under dropout. Layer-wise patterns of effective features mirror studies of intrinsic dimensionality on Pythia-70M. The metric also captures developmental dynamics, detecting sharp feature consolidation during the grokking phase transition. Surprisingly, adversarial training can increase effective features while improving robustness, contradicting the hypothesis that superposition causes vulnerability. Instead, the effect of adversarial training on superposition depends on task complexity and network capacity: simple tasks with ample capacity allow feature expansion (abundance regime), while complex tasks or limited capacity force feature reduction (scarcity regime). By defining superposition as lossy compression, this work enables principled, practical measurement of how neural networks organize information under computational constraints, in particular, connecting superposition to adversarial robustness.

1 Introduction

Interpretability and adversarial robustness could be two sides of the same coin (Räuker et al., 2023). Adversarially trained models learn more interpretable features (Engstrom et al., 2019; Ilyas et al., 2019), develop representations that transfer better (Salman et al., 2020), and align more closely with human perception (Santurkar et al., 2019). Conversely, interpretability-enhancing techniques improve robustness: input gradient regularization (Ross & Doshi-Velez, 2017; Boopathy et al., 2020), attribution smoothing (Etmann et al., 2019), and feature disentanglement (Augustin et al., 2020) all defend against adversarial attacks. Even architectural choices that promote interpretability—lateral inhibition (Eigen & Sadovnik, 2021) and second-order optimization (Tsiligkaridis & Roberts, 2020)—yield more robust models. This pervasive duality demands a mechanistic explanation.

The superposition hypothesis provides one. Elhage et al. (2022) showed that neural networks compress information through *superposition*: encoding multiple features as overlapping activation patterns. When features share dimensions, their interference creates attack surfaces that adversaries could exploit. *Superposition causing vulnerability* explains several phenomena: adversarial transferability emerges from shared feature correlations across models (Liu et al., 2017), the robustness-accuracy trade-off reflects models sacrificing representational capacity for orthogonality (Tsipras et al., 2019), and adversarially trained models become



(a) Observed network (b) Hypothetical disentangled model (c) Superposition as features per neuron

Figure 1: Defining superposition for a neural network layer. **(a)** Observed network showing compressed representation with multiple features sharing neuronal dimensions. **(b)** Hypothetical larger model with fully disentangled features where each feature has its own dedicated neuron (Elhage et al., 2022). **(c)** Superposition measure $\psi = \frac{\#\text{ features}}{\#\text{ neurons}}$ quantifies effective features per neuron, with $\psi = 2$ indicating twice as many features as neurons. Parts adapted from (Bereska & Gavves, 2024).

more interpretable precisely because they reduce feature entanglement (Engstrom et al., 2019). Also, if superposition causes vulnerability, *adversarial training should reduce superposition*.

Testing this requires *measuring superposition* in real networks. While Elhage et al. (2022) used weight matrix Frobenius norms, this approach requires ground truth features—available only in toy models. We lack principled methods to quantify superposition in practice.

We solve this through information theory applied to sparse autoencoders (SAEs). SAEs extract interpretable features from neural activations (Cunningham et al., 2024; Bricken et al., 2023), decomposing them into sparse dictionary elements. We show that the exponential of Shannon entropy applied to these activations measures an effective feature count (Section 3). In quantum mechanics, $e^{S(\rho)}$ quantifies effective pure states in entanglement, where $S(\rho)$ is von Neumann entropy of the density matrix. By analogy, our measure $e^{H(p)}$ counts effective neural features, where $H(p)$ is Shannon entropy of the SAE feature activation distribution. This inherits entropy's useful properties: no arbitrary thresholds, automatic importance weighting, and bounded outputs.

We find the relationship between adversarial robustness and superposition to be nuanced: Rather than simply reducing superposition, adversarial training's effect depends the ratio between task complexity and network capacity (Section 6.4). Simple tasks with ample capacity allow feature *expansion*—robust models develop richer representations by adding defensive variants to existing features. Complex tasks with limited capacity force *reduction*, in accordance with the original hypothesis.

Beyond adversarial robustness, our framework reveals general principles of neural organization (Section 6.3). Pythia-70M's feature counts peak in early MLP layers before declining, matching intrinsic dimensionality (Ansuini et al., 2019). Algorithmic tasks resist superposition, maintaining feature counts near neuron counts likely due to lack of input sparsity (Section 5.2). During grokking, we capture the moment of algorithmic discovery through sharp feature consolidation at the generalization transition (Section 6.2).

This work makes superposition measurable. By grounding neural compression in information theory, we enable quantitative study of how networks organize knowledge under constraints, potentially enabling systematic engineering of interpretable architectures.

2 Related Work

Superposition and polysemanticity. Neural networks employ distributed representations, encoding information across multiple units rather than in isolated neurons (Hinton, 1984; Olah, 2023). The discovery that semantic relationships manifest as directions in embedding space—exemplified by vector arithmetic like "king - man + woman = queen" (Mikolov et al., 2013)—established the *linear representation hypothesis* (Park et al., 2023). Building on this geometric insight, Elhage et al. (2022) formulated the *superposition hypothesis*: networks encode more features than dimensions by representing features as nearly orthogonal directions.

Their toy models revealed phase transitions between monosemantic neurons (one feature per neuron) and polysemantic neurons (multiple features per neuron), governed by feature sparsity. This mechanistically explains *polysemanticity*—neurons responding to multiple unrelated concepts—as a consequence of feature compression.

Scherlis et al. (2023) formalized superposition as capacity allocation under constraints: features compete for neurons based on marginal utility, with important features receiving dedicated dimensions while others share through superposition. Incidentally, polysemanticity emerges even without capacity constraints—Marshall & Kirchner (2024) and Lecomte et al. (2023) demonstrated that dropout and L1 regularization induce polysemanticity by forcing redundant representations (as we show in Section 6.1, this is *not* true for superposition). Recent theoretical work proves networks can compute accurately despite superposition (Vaintrob et al., 2024; Hänni et al., 2024), establishing that F Boolean features require only $\mathcal{O}(\sqrt{F \log F})$ neurons.

Sparse autoencoders for feature extraction. Sparse autoencoders (SAEs) tackle the challenge of extracting interpretable features from polysemantic representations by recasting it as sparse dictionary learning (Sharkey et al., 2022; Cunningham et al., 2024). SAEs decompose neural activations into sparse combinations of learned dictionary elements, effectively reversing the superposition process. Recent architectural innovations improved performance: gated SAEs solve systematic underestimation of feature magnitudes (Rajamanoharan et al., 2024), TopK variants eliminate sensitive hyperparameter tuning (Gao et al., 2024; Bussmann et al., 2024), and Matryoshka SAEs enable multi-scale feature analysis (Bussmann et al., 2025).

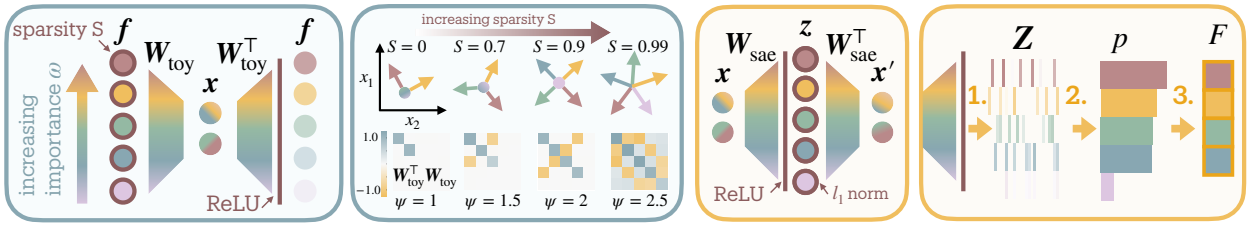
SAEs are scalable to state-of-the-art models: Anthropic extracted millions of interpretable features from Claude 3 Sonnet (Templeton et al., 2024), while OpenAI achieved similar results with GPT-4 (Gao et al., 2024). Crucially, these features are causally relevant—activation steering produces predictable behavioral changes (Marks et al., 2024). Applications now span attention mechanism analysis (Kissane et al., 2024), reward model interpretation (Marks et al., 2023), and automated feature labeling (Paulo et al., 2024), establishing SAEs as a foundational tool for mechanistic interpretability (Bereska & Gavves, 2024).

Information theory and neural measurement. Information-theoretic principles provide rigorous foundations for understanding neural representations. The information bottleneck principle (Tishby et al., 2000), when applied to deep learning (Shwartz-Ziv & Tishby, 2017), reveals how networks balance compression with prediction. Each neural layer acts as a bandwidth-limited channel, forcing networks to develop efficient codes—including superposition—to transmit information forward (Goldfeld et al., 2019). This perspective recasts superposition as an optimal solution to rate-distortion constraints.

Most pertinent to our work, Ayonrinde et al. (2024) connected SAEs to minimum description length (MDL). By viewing SAE features as compression codes for neural activations, they showed that optimal SAEs balance reconstruction fidelity against description complexity. Our entropy-based framework extends this information-theoretic perspective, providing a principled measure of the effective "alphabet size" networks use for internal communication.

Quantifying feature entanglement. Despite superposition's theoretical importance, measuring it remains unexplored. Elhage et al. (2022)'s Frobenius norm approach requires ground truth features and lacks scale invariance, limiting its applicability beyond toy models. Traditional disentanglement metrics from representation learning (Carbonneau et al., 2022; Eastwood & Williams, 2018) assess statistical independence rather than the representational compression that characterizes superposition.

Entropy-based measures have proven effective across disciplines facing similar measurement challenges. Neuroscience employs participation ratios to quantify how many neurons contribute to population dynamics (Gao et al., 2017); economics uses entropy to quantify portfolio concentration (Fontanari et al., 2021). Quantum physics uses the von Neumann entropy to measure entanglement, counting effective pure states in superposition (Nielsen & Chuang, 2011). Recent work bridges these domains, applying entropy measures to neural network analysis (Lee et al., 2023; Shin et al., 2024). Across scientific fields, entropy naturally captures how information distributes across components—precisely the quantity superposition measurement requires.



(a) Toy model architecture (b) Effect of sparsity (c) SAE (d) Feature counting

Figure 2: From toy model to practical superposition measurement. **(a)** Toy model compresses high-dimensional features $\boldsymbol{f}$ through bottleneck $\boldsymbol{x}$ with varying importance (color gradient) before ReLU reconstruction to $\boldsymbol{f}'$. **(b)** Increasing sparsity S enables near-orthogonal feature arrangement in 2D activation space, with correlation matrices $\boldsymbol{W}_{\text{toy}}^T\boldsymbol{W}_{\text{toy}}$ showing increasing interference as superposition measure ψ grows from 1 to 2.5. **(c)** Sparse autoencoders map neural activations $\boldsymbol{x}$ through encoder weights $\boldsymbol{W}_{\text{sae}}$ to sparse representations $\boldsymbol{z}$, then reconstruct via tied decoder weights with ℓ_1 regularization. **(d)** Feature counting proceeds in three steps: 1. extract activation matrix $\boldsymbol{Z}$ from SAE, 2. derive probability distribution p from feature activations, 3. compute effective feature count $F = e^{H(p)}$ through exponential of Shannon entropy.

3 Theoretical Background

3.1 Toy Model of Superposition

To understand how neural networks can represent more features than they have dimensions, Elhage et al. (2022) introduced a minimal toy model that demonstrates superposition under controlled conditions. This model captures three essential properties of real neural networks: feature sparsity, overcomplete representation (more features than dimensions), and varying feature importance.

The model implements a simple autoencoder architecture (Figure 2a) compressing a feature vector $\boldsymbol{f} \in \mathbb{R}^F$ through a bottleneck $\boldsymbol{x} \in \mathbb{R}^N$ where $F > N$:

$$\begin{aligned} \boldsymbol{x} &= \boldsymbol{W}_{\text{toy}}\boldsymbol{f}, \\ \boldsymbol{f}' &= \text{ReLU}(\boldsymbol{W}_{\text{toy}}^T\boldsymbol{x} + \boldsymbol{b}) \end{aligned} \tag{1}$$

Here F counts input features (by construction in this toy model), N counts bottleneck neurons, and $\boldsymbol{W}_{\text{toy}} \in \mathbb{R}^{N\times F}$ maps between them. The model must somehow represent F features using only N dimensions—an impossible task unless features can share neuronal resources.

Each input feature f_i samples uniformly from $[0, 1]$ with sparsity S (probability of being zero) and importance weight ω_i. Training minimizes importance-weighted reconstruction error:

$$\mathcal{L}(\boldsymbol{f}) = \sum_{i=1}^{F} \omega_i \|f_i - f_i'\|^2 \tag{2}$$

As sparsity increases, the model packs multiple features into the same dimensions by arranging them as nearly orthogonal directions (Figure 2b). The correlation matrices $\boldsymbol{W}_{\text{toy}}^T\boldsymbol{W}_{\text{toy}}$ reveal this geometric solution—features interfere minimally despite sharing space.

To quantify superposition in this controlled setting, Elhage et al. (2022) proposed measuring the total "weight mass":

$$\psi_{\text{Elhage}} = \frac{\|\boldsymbol{W}_{\text{toy}}\|_{\text{Frob}}^2}{N} \tag{3}$$

where $\|\boldsymbol{W}_{\text{toy}}\|_{\text{Frob}}^2 = \sum_{i,j} W_{ij}^2$ and N the number of neurons. When features occupy orthogonal dimensions, $\psi_{\text{Elhage}} \approx 1$. As superposition increases—features increasingly sharing dimensions—this measure grows accordingly.

Yet this approach faces two fundamental limitations. First, it lacks scale invariance: multiplying weights by any constant arbitrarily changes the measure. Second, and more critically, it requires knowing the true features—unavailable in real networks.

3.2 Sparse Autoencoders for Feature Extraction

Real networks don't reveal their features directly. Instead, we must untangle them from distributed neural activations. Sparse autoencoders (SAEs) decompose activations into sparse combinations of learned dictionary elements, effectively reverse-engineering the toy model's feature representation.

Given layer activations $\boldsymbol{x} \in \mathbb{R}^N$, an SAE learns a higher-dimensional sparse code $\boldsymbol{z} \in \mathbb{R}^D$ where $D > N$ (Figure 2c):

$$\boldsymbol{z} = \text{ReLU}(\boldsymbol{W}_{\text{enc}}\boldsymbol{x} + \boldsymbol{b}) \tag{4}$$

The reconstruction combines these sparse features:

$$\boldsymbol{x}' = \boldsymbol{W}_{\text{dec}}\boldsymbol{z} = \sum_{i=1}^{D} z_i \boldsymbol{d}_i \tag{5}$$

where columns $\boldsymbol{d}_i$ of $\boldsymbol{W}_{\text{dec}}$ form the learned dictionary.

Training balances two objectives—faithful reconstruction and sparse activation:

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{z}) = \|\boldsymbol{x} - \boldsymbol{x}'\|_2^2 + \alpha\|\boldsymbol{z}\|_1 \tag{6}$$

where the ℓ_1 penalty with strength α encourages each input to activate only a small subset of features, revealing the network's sparse code. For stability, we tie encoder and decoder weights:

$$\boldsymbol{W}_{\text{sae}} := \boldsymbol{W}_{\text{dec}} = \boldsymbol{W}_{\text{enc}}^T \tag{7}$$

If networks truly employ superposition—encoding features as directions in activation space—then SAEs should recover these as dictionary elements. Recent validation shows SAE features causally affect network behavior (Marks et al., 2024), suggesting they capture genuine computational structure.

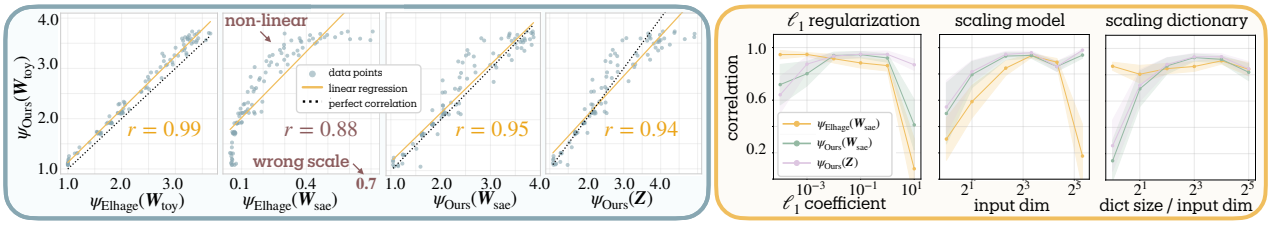
4 Measuring Superposition with Sparse Autoencoders

With features extracted by SAEs, we can quantify superposition. We define it simply as the ratio of features to neurons:

$$\psi = \frac{F}{N} \tag{8}$$

Counting neurons N is trivial; counting features F requires more thought. Our solution draws from information theory: we measure not raw feature quantity but information capacity.

In communication systems, Shannon entropy's exponential measures *effective alphabet size*—how many symbols a channel can distinguish. We apply this principle to neural networks, viewing each layer as a bandwidth-limited channel encoding feature information. Given a probability distribution p over features, Shannon entropy $H(p) = -\sum_i p_i \log p_i$ quantifies average information content. Its exponential:



(a) Correlation with ground truth

(b) Robustness across hyperparameters

Figure 3: Validation of entropy-based superposition metrics. **(a)** Our measure maintains high correlation whether applied to toy weights ($r = 0.99$) or SAE activations ($r = 0.94$), while Elhage's metric fails on SAE weights. **(b)** Performance remains stable across ℓ_1 regularization, model scale, and dictionary size variations. Shaded regions show 95% confidence intervals across 100 model-SAE pairs.

$$F(p) = e^{H(p)} = \exp\left(-\sum_i p_i \log p_i\right) \tag{9}$$

yields the distribution's *perplexity*—equivalently, the number of equally probable features producing identical entropy. This captures the layer's effective information capacity.

To construct our feature probability distribution, we aggregate SAE activations across S samples. Given neural activations $\boldsymbol{X} \in \mathbb{R}^{N \times S}$, the SAE produces sparse latent codes $\boldsymbol{Z} = \mathrm{ReLU}(\boldsymbol{W}_{\text{sae}}\boldsymbol{X}) \in \mathbb{R}^{D \times S}$. We compute each feature's probability as its share of total activation magnitude:

$$p_i = \frac{\|\boldsymbol{Z}_{i,:}\|_1}{\sum_{j=1}^{D} \|\boldsymbol{Z}_{j,:}\|_1} \tag{10}$$

where $\|\boldsymbol{Z}_{i,:}\|_1 = \sum_{s=1}^{S} |z_{i,s}|$ aggregates feature i across all samples. The SAE's ℓ_1 regularization ensures these magnitudes reflect each feature's contribution to reconstruction quality—features that activate more frequently or more strongly consume more of the network's representational budget (Appendix A.2). This activation[1] distribution therefore captures each feature's participation in the network's resource economy, combining both usage frequency and activation magnitude. In practice, we use sufficient samples until convergence (see convergence analysis in Section 6.3). The steps of the process are visualized in Figure 2d.

Our measure inherits desirable properties from entropy. For any D-component distribution, $1 \leq F(p) \leq D$, bounded by single-feature dominance and uniform distribution. Unlike threshold-based counting, features contribute according to their information content—rare features matter less than common ones, weak features less than strong ones.

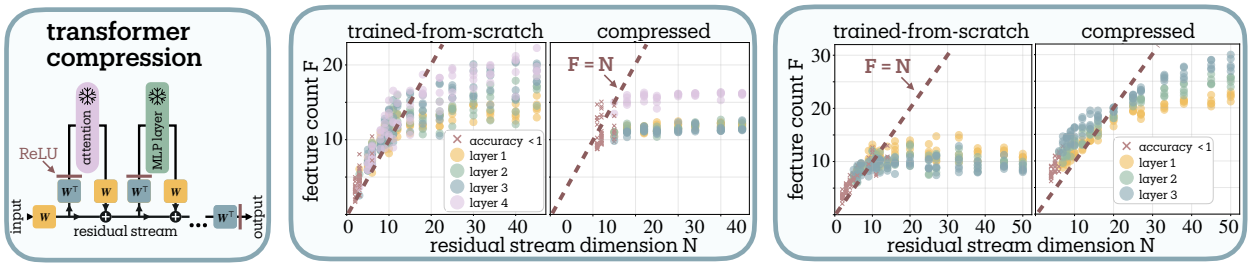
5 Validation of the Measurement Framework

5.1 Toy Model of Superposition

We validate our entropy-based measure using the toy model of superposition (Elhage et al., 2022), where ground truth features provide an objective benchmark. This controlled setting tests whether sparse autoencoders can recover accurate feature counts from superposed representations.

Following Elhage et al. (2022), we generate 100 toy models with sparsity $S \in [0.001, 0.999]$. Each model compresses 20 features through a 5-neuron bottleneck, with importance weights decaying as $\omega_i = 0.7^i$.

[1] Why not use weight-based measurement? One might compute probabilities from SAE weights: $p_i = \|\boldsymbol{w}_i\|_1 / \sum_j \|\boldsymbol{w}_j\|_1$. This geometric view, however, misses that features existing in weights may never activate—"dead features" that consume capacity without contributing to computation. Empirically, a weight-based metric succeeds only in the toy model (Figure 3a, $\psi_{\text{Ours}}(\boldsymbol{W}_{\text{sae}})$); but small toy transformer models already require our functional, activation-based approach (e.g. Section 5.2).



(a) Architecture (b) Task: sequence reversal (c) Task: sequence sorting

Figure 4: Algorithmic tasks under compression. **(a)** Compression architecture projects activations through $\mathrm{ReLU}(\boldsymbol{W}^\top \boldsymbol{W}\boldsymbol{x})$ to force features into fewer dimensions. **(b)** Sequence reversal maintains constant feature count (12) despite 10× compression, with sharp failure when dimensions drop below requirements. **(c)** Sequence sorting also tracks the $F = N$ bound: Both tasks show minimal superposition likely due to lack of input sparsity.

After training to convergence, we extract 10,000 activation samples and train SAEs with 40-dimensional dictionaries (8× expansion) and ℓ_1 coefficient 0.1. This two-stage process mimics real-world measurement where ground truth remains unknown.

Ground truth correlation. Our entropy-based measure achieves near-perfect correlation with Elhage's metric when applied to toy model weights ($r = 0.99 \pm 0.01$), validating the theoretical framework (Figure 3a). Yet Elhage's metric catastrophically fails on SAE weights—the Frobenius norm produces nonlinear relationships and incorrect scales (0.1–0.7 versus expected 1–4). The ℓ_1 regularization fundamentally alters weight statistics, breaking the original measure.

Our activation-based approach sidesteps this failure. By measuring feature utilization rather than weight magnitude, we maintain strong correlation ($r = 0.94 \pm 0.02$) with ground truth even through the SAE bottleneck. This robustness enables practical application to real networks.

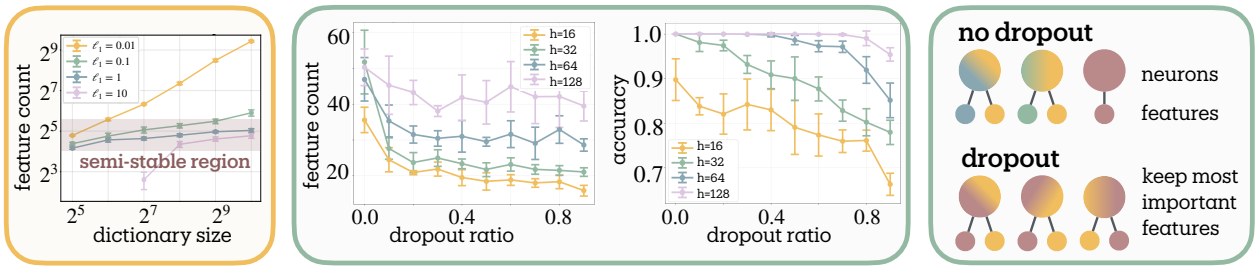
Hyperparameter stability. We test sensitivity across three axes: ℓ_1 strength (10^{-3} to 10^{0}), model scale (8–32 input dimensions), and dictionary expansion (2× to 32×). Figure 3b shows stable performance except at extremes—excessive regularization ($\ell_1 = 1.0$) or insufficient dictionary capacity (fewer elements than features). This stability confirms we capture intrinsic superposition properties rather than measurement artifacts.

5.2 Compression Does Not Imply Superposition

Tracr compiles human-readable programs into transformer weights with known computational structure (Lindner et al., 2023). Compiled models use highly interpretable weights and activations consisting primarily of 0s and 1s, providing a form of ground truth for mechanistic analysis. We examine sequence reversal ("123" → "321") and sorting ("213" → "123"), comparing original compiled models against compressed variants and models trained from scratch with the same transformer architecture and residual stream dimension.

Following Lindner et al. (2023), we compress models by projecting residual stream activations through learned compression matrices. Our compression scheme (Figure 4a) applies $\mathrm{ReLU}(\boldsymbol{W}^\top \boldsymbol{W}\boldsymbol{x})$ where $\boldsymbol{W} \in \mathbb{R}^{N' \times N}$, compressing from originally N' dimensions to N, ($N' > N$). The ReLU activation—absent in the original Tracr compression—allows small interference terms to cancel out, following the toy model rationale (Elhage et al., 2022). We train compression matrices using the original loss functions and hyperparameters, vary compression factors from 1× to 10×, and analyze residual stream activations through SAEs with 100-feature dictionaries, expecting superposition as capacity tightens.

Minimal superposition despite compression. Surprisingly, both compressed Tracr models and transformers trained from scratch learn feature counts closely tracking the residual stream dimension $F \approx N$



(a) Dictionary scaling (b) Dropout reduces features and performance. (c) Dropout effect on features

Figure 5: Measurements on multi-task sparse parity dataset. (a) Dictionary scaling plateaus with proper regularization, validating intrinsic structure measurement. (b) Dropout reduces features and accuracy, (c) and distinguishes polysemanticity (neurons encoding multiple features) from superposition (multiple features per neuron) by creating to more polysemantic neurons while reducing the number of features.

(Figure 4b, 4c). Notably, these feature counts remain far below the original compiled dimensions—e.g. reversal models use only 12 features[2] despite originally having 45 dimensions, revealing substantial redundancy in Tracr's compilation process rather than genuine superposition. When dimensions drop below intrinsic task requirements, performance collapses ($\times$ markers).

Algorithmic tasks lack input sparsity. This resistance to superposition likely stems from algorithmic tasks violating the sparsity assumption: The toy model of superposition (Elhage et al., 2022) requires features to activate sparsely across inputs—most features remain inactive on most samples, enabling near-orthogonal packing with minimal interference. Algorithmic tasks break this assumption: sequence operations require consistent activation patterns across inputs. Without sparsity, interference that enables efficient superposition becomes destructive.

5.3 Dictionary Scaling Convergence

Measuring a natural coastline with a finer ruler yields a longer measurement—potentially without bound (Mandelbrot, 1967). As SAE dictionaries grow, might we discover arbitrarily many features at finer scales?

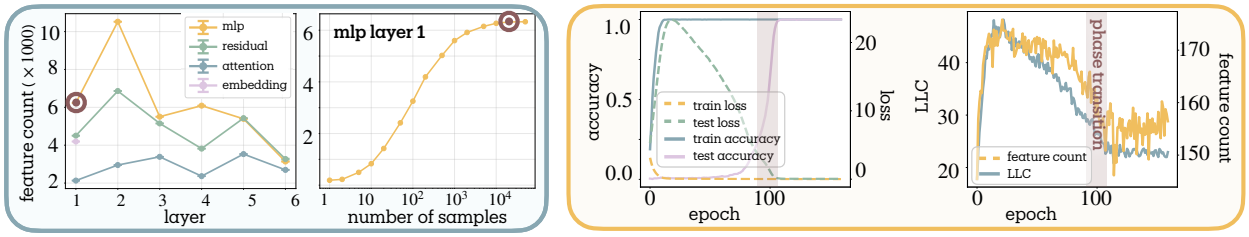
We test convergence using multi-task sparse parity (Michaud et al., 2023)—3 tasks, 4 bits each—where ground truth bounds meaningful features. Networks with 64 hidden neurons trained across dictionary scales (0.5$\times$ to 16$\times$ hidden dimension) and ℓ_1 strengths (0.01 to 10.0).

Figure 5a reveals two regimes. With appropriate regularization ($\ell_1 \geq 0.1$), feature counts plateau despite dictionary expansion—we measure intrinsic structure, not artifacts. But weak regularization ($\ell_1 = 0.01$) shows unbounded growth, decomposing features into ever-finer components. While practical measurement requires standardized SAE training, convergence under proper regularization validates our approach captures genuine representational properties.

6 Applications and Findings

Our entropy-based superposition measure enables systematic investigation of neural information organization across diverse contexts. We demonstrate four key applications: capacity-constrained feature allocation (Section 6.1), developmental dynamics during learning transitions (Section 6.2), layer-wise representational organization in language models (Section 6.3), and the counterintuitive relationship between superposition and adversarial robustness (Section 6.4).

[2]While we generally recommend *comparative* interpretation due to measurement limitations (Section 7), the systematic $F = N$ tracking and performance decline when trespassing this boundary suggest our measure can provide meaningful *absolute* feature counts in sufficiently constrained computational settings, extending interpretation as absolute count beyond our toy model validation from Section 5.1.



(a) Pythia-70M layer analysis

(b) Grokking dynamics

Figure 6: **(a)** Non-monotonic feature organization across Pythia-70M. MLP layer 1 peaks at 10,000 features (20× neurons). Convergence analysis shows saturation after 2×10^4 samples. **(b)** Feature dynamics during grokking. Sharp consolidation at generalization transition (checkpoint 60) concurs with (smoother) LLC decay. Strong correlation $r \approx 0.9$ with LLC suggests feature count can function as measure of model complexity.

6.1 Dropout acts as Capacity Constraint

Neural networks face capacity constraints that force feature reduction. We investigate how dropout affects feature organization using multi-task sparse parity (3 tasks, 4 bits each) with simple MLPs across hidden dimensions $h \in \{16, 32, 64, 128\}$ and dropout rates [0.0, 0.1, ..., 0.9] (Appendix B.2 for details).

Marshall & Kirchner (2024) showed dropout induces polysemanticity through redundancy—features must distribute across neurons to survive random deactivation. One might expect this to increase measured features. Instead, dropout monotonically reduces feature count by up to 50% (Figure 5b).

Polysemanticity (neurons encoding multiple features) differs from superposition (multiple features per available capacity). Dropout forces each feature to occupy multiple neurons for robustness, but this redundant encoding consumes capacity—fewer total features fit within the same dimensional budget. Networks respond by pruning less essential features, aligning with Scherlis et al. (2023)'s competitive framework.

The capacity dependence validates this interpretation—larger networks show reduced dropout sensitivity while narrow networks exhibit sharp feature reduction, confirming that capacity constraints drive the observed dynamics.

6.2 Capturing Grokking Phase Transition

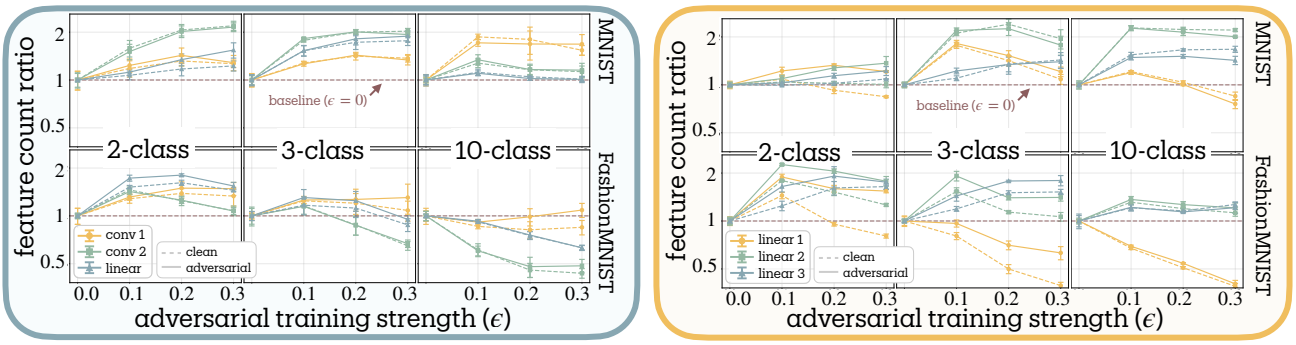
Grokking—sudden perfect generalization after extended training on algorithmic tasks—provides an ideal testbed for developmental measurement (Power et al., 2022). We investigate whether feature count dynamics can detect this phase transition and how they relate to the Local Learning Coefficient (LLC) from singular learning theory (Hoogland et al., 2024).

We train a two-path MLP on modular arithmetic $(a+b) \bmod 53$ (Appendix B.3 for details). Figure 6b reveals distinct dynamics: while LLC shows initial proliferation followed by smooth decay throughout training, our feature count exhibits sharp consolidation precisely at the generalization transition (checkpoint 60).

This pattern suggests the measures capture different aspects of complexity evolution. During memorization, the model employs numerous superposed features to store input-output mappings. The sharp consolidation likely coincides with algorithmic discovery, where the model reorganizes from distributed lookup tables into compact representations that capture the modular arithmetic rule (Nanda et al., 2023). Strong correlation ($r = 0.908$, $p < 0.001$) between feature count and LLC suggests feature count can function as a measure of model complexity.

6.3 Layer-wise Pattern Mirrors Intrinsic Dimensionality

We analyze Pythia-70M using pretrained SAEs from Marks et al. (2024), measuring feature counts across all layers and components (Appendix B.4 for details). Convergence analysis (Figure 6a) shows saturation



(a) 3-layer CNN on (Fashion-)MNIST

(b) 3-layer MLP on (Fashion-)MNIST

Figure 7: Task complexity modulates adversarial training's effect on superposition. Each panel shows results varying dataset (MNIST top, Fashion-MNIST bottom) and number of classes (2, 3, 10). **(a)** CNNs show clear complexity-dependent transitions: simple tasks enable feature expansion while complex tasks (10 classes) force reduction below baseline. **(b)** MLPs exhibit similar patterns with more pronounced layer-wise variation. Fashion-MNIST consistently amplifies the reduction effect compared to MNIST, suggesting that representational complexity drives defensive strategies beyond mere class count. Dashed lines: clean data; solid lines: adversarial examples. Feature count ratios normalized to $\epsilon = 0$ baseline. Error bars show standard error across 3 seeds.

after 2×10^4 samples, suggesting that we captured the "true" feature count. Feature importance follows power-law distributions (not shown), explaining why naive counting (e.g. 21K nonzero features for MLP 1) vastly overestimates our entropy-based measure (5.6K effective features for MLP 1).

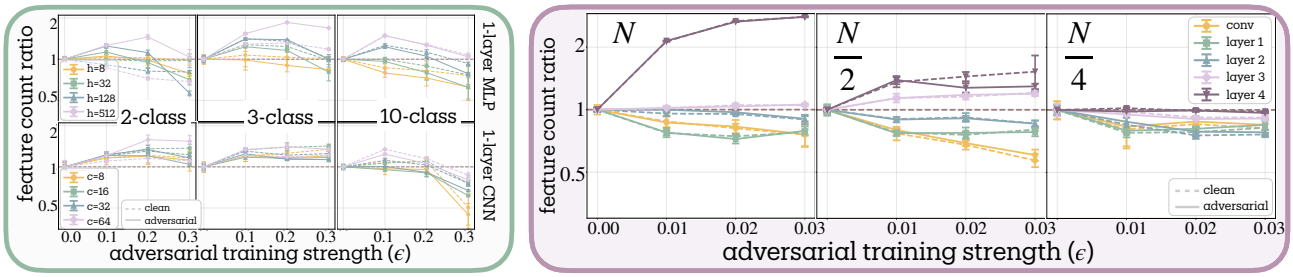
MLPs store the most features, followed by residual streams, with attention maintaining minimal counts, showing MLPs as knowledge stores and attention as routing mechanisms (Geva et al., 2021). Feature grow in early layers (10,000 features — 20× more features than neurons — at peak MLP layer 1), compress through middle layers, then re-expand before final consolidation (Figure 6a). This trajectory parallels intrinsic dimensionality studies (Ansuini et al., 2019): Both reveal non-monotonic "hunchback" patterns despite measuring complementary aspects of neural activations: intrinsic dimensionality captures compression (minimal dimensions needed), while we measure expansion (features encoded).

6.4 Adversarial Robustness: Capacity-Constrained Defense Strategies

Testing the superposition-vulnerability hypothesis. The superposition-vulnerability hypothesis proposed by Elhage et al. (2022) predicts that adversarial training should universally reduce superposition, as networks trade representational efficiency for orthogonal, robust features. We test this prediction systematically across diverse architectures and conditions, finding that the relationship between adversarial robustness and superposition depends critically on task complexity and network capacity.

We employ PGD adversarial training (Madry et al., 2018) across architectures ranging from single-layer to deep networks (MLPs, CNNs, ResNet-18) on multiple datasets (MNIST, Fashion-MNIST, CIFAR-10). Task complexity varies through both classification granularity (2, 3, 5, 10 classes) and dataset difficulty. Network capacity varies through hidden dimensions (8–512 for MLPs), filter counts (8–64 for CNNs), and width scaling (1/4×–2× for ResNet-18). For convolutional networks, we measure superposition across channels by reshaping activation tensors to treat spatial positions as independent samples (see Appendix A.7 for details). All SAEs use 4× dictionary expansion with $\ell_1 = 0.1$. Measurements on adversarial examples match the training distribution—models trained with $\epsilon = 0.2$ are evaluated on $\epsilon = 0.2$ attacks.

Statistical methodology. To quantify adversarial training effects rigorously, we extract normalized slopes representing how feature counts change per unit increase in adversarial training strength ($\epsilon \in \{0.0, 0.1, 0.2, 0.3\}$). For each experimental condition, we fit linear regressions to feature counts across epsilon



(a) Widening 1-layer NNs on MNIST

(b) Narrowing ResNet-18 on CIFAR-10

Figure 8: Network capacity modulates adversarial training responses across architectures. **(a)** Single-layer networks on MNIST demonstrate capacity-dependent phase transitions: MLPs with hidden dimensions $h \in \{8, 32, 128, 512\}$ (top) and CNNs with filter counts $c \in \{8, 16, 32, 64\}$ (bottom) show that narrow networks reduce features while wide networks maintain expansion across task complexities. **(b)** ResNet-18 on CIFAR-10 with width scaling ($1\times$, $1/2\times$, $1/4\times$) reveals layer-wise specialization: early layers reduce features while deeper layers (layer 3-4) expand dramatically, with this pattern dampening as width decreases. Dashed lines: clean data; solid lines: adversarial examples. Feature count ratios normalized to baseline.

values, pooling clean and adversarial observations to increase statistical power. These slopes are normalized by baseline ($\epsilon = 0$) feature counts, making effects comparable across layers with different absolute scales.

Since networks contain multiple layers, we aggregate layer-wise measurements using parameter-weighted averaging, where layers with more parameters receive proportionally higher weight. This reflects the assumption that computationally intensive layers better represent overall network behavior. For simple architectures, parameter counts include all weights and biases; for ResNet-18, we implement detailed counting that accounts for convolutions, batch normalization, and skip connections.

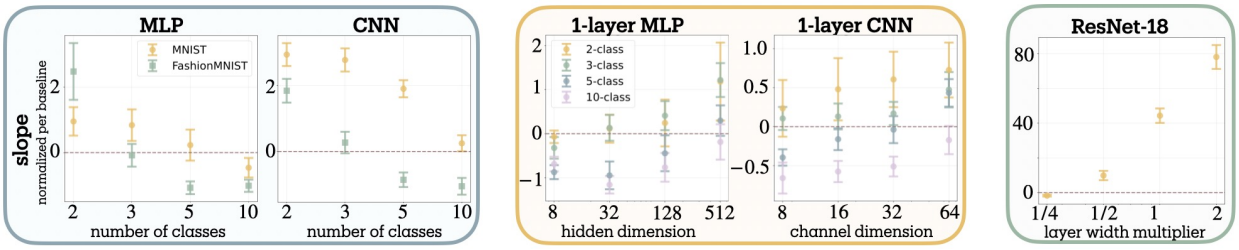
We test three formal hypotheses: **H1 (Universal Reduction)**: Adversarial training uniformly reduces superposition across all conditions, directly testing Elhage et al. (2022)'s original prediction. **H2 (Task Complexity Modulation)**: Task complexity systematically modulates adversarial training effects, with complex tasks showing greater feature reduction. We encode complexity ordinally (2-class=1, 3-class=2, 5-class=3, 10-class=4) and test for linear trends. **H3 (Capacity Enabling)**: Network capacity enables different defensive strategies, with higher capacity permitting feature expansion. We test log-linear relationships between capacity measures and adversarial training slopes.

All statistical tests use inverse-variance weighting to account for measurement uncertainty, with random-effects meta-analysis when significant heterogeneity exists across conditions.

Task complexity determines defensive strategy. Contrary to H1's prediction of universal reduction, adversarial training produces bidirectional effects that depend systematically on task complexity (Figures 7 and 9a). Our meta-analysis reveals significant heterogeneity across conditions ($Q = 8.047$, $df = 3$, $p = 0.045$), necessitating random-effects modeling that yields an overall negative relationship between task complexity and feature expansion (combined slope $= -0.745 \pm 0.122$, $z = -6.14$, $p < 0.001$).

Binary classification consistently expands feature counts—up to $2\times$ baseline—suggesting networks develop additional defensive features when task demands are simple. Ten-class problems show the opposite pattern, with feature counts decreasing by up to 60%, particularly in early layers. Three-class tasks exhibit intermediate behavior with inverted-U curves: moderate adversarial training ($\epsilon = 0.1$) initially expands features before stronger training ($\epsilon = 0.3$) triggers reduction.

The dataset difficulty amplifies these effects significantly. While both datasets show negative complexity trends, Fashion-MNIST produces systematically lower slopes than MNIST (mean difference $= -1.467 \pm 0.156$, $t(7) = -2.405$, $p = 0.047$, Cohen's $d = -0.85$). CNN architectures demonstrate particularly clear patterns, with statistically significant negative trends for both MNIST (slope $= -0.895 \pm 0.236$, $p = 0.063$) and



(a) Complexity effects (b) Capacity (1-layer networks) (c) Capacity effect

Figure 9: Statistical analysis of adversarial training effects on superposition. **(a)** Task complexity systematically modulates adversarial responses across architectures. Normalized slopes quantify feature count changes per unit adversarial strength, showing negative linear trends (CNN-MNIST: -0.895 ± 0.236, $p = 0.063$; CNN-Fashion-MNIST: -0.983 ± 0.220, $p = 0.047$). Fashion-MNIST (green) produces consistently lower slopes than MNIST (yellow), with significant paired difference (-1.467 ± 0.156, $p = 0.047$). **(b)** Single-layer networks show capacity-dependent phase transitions with significant positive effects (meta-analytic slope $= 0.220 \pm 0.037$, $p < 0.001$). Narrow networks (8–32 dimensions) reduce features while wide networks (128–512) maintain expansion. **(c)** ResNet-18 demonstrates dramatic log-linear scaling (31.0 ± 2.0 per log(width), $p = 0.004$, $R^2 = 0.929$), with 8-fold width increases yielding 65-unit slope changes. Error bars show standard errors across conditions.

Fashion-MNIST (slope $= -0.983 \pm 0.220$, $p = 0.047$). This suggests that representational demands, beyond mere class count, drive defensive strategies.

Layer-wise analysis reveals architectural differences in how complexity effects manifest. In MLPs, first layers typically show strongest reduction (down to 30% of baseline on Fashion-MNIST 10-class), while middle layers often expand even under complex tasks. CNNs display an inverted pattern: second convolutional layers reduce most dramatically while first layers remain relatively stable. Both architectures transition through similar inverted-U curves, suggesting common underlying dynamics despite different inductive biases.

Network capacity enables defensive elaboration. Network capacity systematically modulates adversarial training responses, strongly supporting H3 (Figures 8, 9b, and 9c). Single-layer networks demonstrate clear capacity thresholds with a significant positive relationship between capacity and feature expansion (meta-analytic slope $= 0.220 \pm 0.037$, $z = 5.90$, $p < 0.001$). Networks with minimal capacity (8 hidden units for MLPs, 8 filters for CNNs) reduce features across all task complexities, while high-capacity networks (512 units/64 filters) maintain expansion even for 10-class problems.

This capacity dependence scales dramatically in deep architectures. ResNet-18 on CIFAR-10 exhibits a strong log-linear relationship between width multiplier and adversarial training slopes (slope $= 31.0 \pm 2.0$ per log(width), $t(2) = 15.7$, $p = 0.004$, $R^2 = 0.929$). An 8-fold width increase ($0.25\times$ to $2\times$) produces a 65-unit change in normalized slope—orders of magnitude larger than simple network effects. At minimal width ($0.25\times$), adversarial training barely affects feature counts, while at double width, networks show massive feature expansion with slopes approaching 80.

The layer-wise progression in ResNet-18 reveals hierarchical specialization: early layers (conv1, layer1) reduce features by up to 50%, middle layers remain stable, while deep layers (layer3, layer4) expand up to $4\times$. Systematically narrowing the network dampens this pattern—at 1/4 width, late-layer expansion vanishes while early-layer reduction persists but weakens. This could reflect vulnerability hierarchies, where early layers processing low-level statistics are easily exploited by imperceptible perturbations, necessitating feature reduction, while late layers encoding semantic information can safely expand their representational repertoire.

Reconciling with the original hypothesis. Our findings reveal a more nuanced relationship between superposition and adversarial vulnerability than originally theorized. Rather than universal feature reduction, adversarial training operates in two distinct regimes:

Abundance regime: When task complexity is low relative to network capacity, adversarial training expands the feature repertoire. Networks appear to maintain original capabilities while adding defensive variants—achieving robustness through redundancy rather than orthogonalization. This regime dominates in high-capacity networks facing simple tasks.

Scarcity regime: When task demands approach capacity limits, networks economize by reducing to fewer, potentially more orthogonal features as predicted by the vulnerability hypothesis. This regime emerges in capacity-constrained networks or when facing complex classification tasks.

Several patterns in our data remain unexplained. We observe non-monotonic inverted-U curves where moderate adversarial training ($\epsilon = 0.1$) expands features while stronger training ($\epsilon = 0.3$) reduces them below baseline. The gap between clean and adversarial feature counts varies unpredictably—sometimes negligible, sometimes substantial—without clear pattern. MLPs and CNNs show opposing layer-wise responses: MLP first layers reduce most while middle layers expand; CNN second layers reduce most while first layers stay stable. Some results contradict our complexity hypothesis, with 2-class MLPs occasionally showing lower feature counts than 3-class. CNN experiments consistently yield stronger statistical significance ($p < 0.02$) than equivalent MLP experiments ($p \approx 0.09$) for unknown reasons. These patterns suggest our understanding of how adversarial training affects neural representations remains incomplete despite the systematic trends we identify.

This framework might explain why robust models often appear more interpretable (Engstrom et al., 2019): not necessarily because they have fewer features universally, but potentially because resource constraints force more organized, essential representations in the scarcity regime.

The bidirectional relationship between robustness and superposition suggests that achieving robustness without capability loss may require ensuring sufficient capacity for defensive elaboration. While our experiments demonstrate that increased robustness can coincide with either increased or decreased superposition depending on the regime, establishing the exact causal connection between superposition and robustness remains an important direction for future work.

7 Limitations

Our superposition measurement framework is limited by its dependence on sparse autoencoder quality, theoretical assumptions about neural feature representation, and should be interpreted as proxy for representational complexity rather than literal feature count:

Sparse autoencoder quality. Our approach inherently depends on sparse autoencoder feature extraction quality. While recent architectural advances—gated SAEs (Rajamanoharan et al., 2024), TopK variants (Gao et al., 2024), and end-to-end training (Braun et al., 2024)—have substantially improved feature recovery, fundamental challenges remain. SAE training exhibits sensitivity to hyperparameters, particularly ℓ_1 regularization strength and dictionary size, with different initialization or training procedures potentially yielding different feature counts for identical networks. Ghost features—SAE artifacts without computational relevance (Gao et al., 2024)—can artificially inflate measurements, while poor reconstruction quality may deflate them.

Assumptions on feature representation. Our framework rests on several assumptions that real networks systematically violate. The linear representation assumption—that features correspond to directions in activation space—has been challenged by recent discoveries of circular feature organization for temporal concepts (Engels et al., 2024) and complex geometric structures beyond simple directions (Black et al., 2022). Our entropy calculation assumes features contribute independently to representation, but neural networks exhibit extensive feature correlations, synergistic information where feature combinations provide more information than individual contributions, and gating mechanisms where some features control others' activation. The approximation that sparse linear encoding captures true computational structure breaks down in hierarchical representations where low-level and high-level features are not substitutable, and in networks with substantial nonlinear feature interactions that cannot be decomposed additively.

Comparative rather than absolute count. Our measure quantifies effective representational diversity under specific assumptions rather than providing literal feature counts. This creates several interpretational limitations. The measure exhibits sensitivity to the activation distribution used for measurement—SAE training distributions must match the network's operational regime to avoid systematic bias. Feature granularity remains fundamentally ambiguous: broader features may decompose into specific ones in wider SAEs, creating uncertainty about whether we're discovering or creating features. Our single-layer analysis potentially misses features distributed across layers through residual connections or attention mechanisms. Most critically, we measure the effective alphabet size of the network's internal communication channel rather than counting distinct computational primitives, making comparative rather than absolute interpretation most appropriate.

The limitations largely reflect active research areas in sparse dictionary learning and mechanistic interpretability. Each advance in SAE architectures, training procedures, or theoretical understanding directly benefits measurement quality. Within its scope—comparative analysis of representational complexity under sparse linear encoding assumptions—the measure enables systematic investigation of neural information structure previously impossible.

8 Conclusion

We introduced an information-theoretic framework to measure superposition in neural networks through sparse autoencoder activations. The entropy-based measure $F = e^{H(p)}$ quantifies effective feature counts without requiring ground truth, enabling systematic study of representational organization.

Our key empirical finding contradicts theoretical prediction: adversarial training often increases rather than decreases feature counts while improving robustness. This suggests robust models expand their feature repertoires rather than orthogonalizing them, challenging the superposition-vulnerability hypothesis. In interpretability research, the tools are only as good as the assumptions they force us to abandon.

The measurement framework reveals general principles of neural organization. Language models show non-monotonic feature patterns across layers, algorithmic tasks resist superposition entirely, and grokking exhibits sharp representational transitions. While our approach depends on SAE quality and assumes linear feature combination, it provides the first principled method for quantifying superposition at scale. This enables comparative analysis across architectures and training regimes.

References

Kartik Anand, Ginestra Bianconi, and Simone Severini. The shannon and the von neumann entropy of random networks with heterogeneous expected degree. *Phys. Rev. E*, March 2011. 20

A. Ansuini, A. Laio, J. Macke, and D. Zoccolan. Intrinsic dimension of data representations in deep neural networks. *NeurIPS*, May 2019. 2, 10

Maximilian Augustin, Alexander Meinke, and Matthias Hein. Adversarial robustness on in- and out-distribution improves explainability. *ECCV*, July 2020. 1

Kola Ayonrinde, Michael T. Pearce, and Lee Sharkey. Interpretability as compression: Reconsidering sae explanations of neural activations with mdl-saes. *CoRR*, October 2024. 3

Leonard Bereska and Efstratios Gavves. Mechanistic interpretability for ai safety — a review. *TMLR*, August 2024. 2, 3

Sid Black, Lee Sharkey, Leo Grinsztajn, Eric Winsor, Dan Braun, Jacob Merizian, Kip Parker, Carlos Ramón Guevara, Beren Millidge, Gabriel Alfour, and Connor Leahy. Interpreting neural networks through the polytope lens. *CoRR*, November 2022. 13, 21

Akhilan Boopathy, Sijia Liu, Gaoyuan Zhang, Cynthia Liu, Pin-Yu Chen, Shiyu Chang, and Luca Daniel. Proper network interpretability helps adversarial robustness in classification. *ICML*, October 2020. 1

Dan Braun, Jordan Taylor, Nicholas Goldowsky-Dill, and Lee Sharkey. Identifying functionally important features with end-to-end sparse dictionary learning. *ICML MI Workshop*, May 2024. 13

Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermyn, Tom Conerly, Nicholas L. Turner, Cem Anil, Carson Denison, Amanda Askell, Robert Lasenby, Yifan Wu, Shauna Kravec, Nicholas Schiefer, Tim Maxwell, Nicholas Joseph, Alex Tamkin, Karina Nguyen, Brayden McLean, Josiah E. Burke, Tristan Hume, Shan Carter, Tom Henighan, and Chris Olah. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, October 2023. 2

Bart Bussmann, Patrick Leask, and Neel Nanda. Batchtopk sparse autoencoders. *CoRR*, 2024. 3

Bart Bussmann, Noa Nabeshima, Adam Karvonen, and Neel Nanda. Learning multi-level features with matryoshka sparse autoencoders. *CoRR*, 2025. 3

Marc-André Carbonneau, Julian Zaidi, Jonathan Boilard, and Ghyslain Gagnon. Measuring disentanglement: A review of metrics. *IEEE Trans. Neural Netw. Learn. Syst.*, May 2022. 3

Hoagy Cunningham, Aidan Ewart, Logan Riggs, Robert Huben, and Lee Sharkey. Sparse autoencoders find highly interpretable features in language models. *ICLR*, January 2024. 2, 3

Cian Eastwood and Christopher K. I. Williams. A framework for the quantitative evaluation of disentangled representations. *ICLR*, February 2018. 3

Henry Eigen and Amir Sadovnik. Topkconv: Increased adversarial robustness through deeper interpretability. *ICMLA*, December 2021. 1

Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, et al. Toy models of superposition. *Transformer Circuits Thread*, 2022. 1, 2, 3, 4, 6, 7, 8, 10, 11, 21

Joshua Engels, Isaac Liao, Eric J. Michaud, Wes Gurnee, and Max Tegmark. Not all language model features are linear. *CoRR*, May 2024. 13, 21

Logan Engstrom, Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Brandon Tran, and Aleksander Madry. Adversarial robustness as a prior for learned representations. *CoRR*, September 2019. 1, 2, 13

Christian Etmann, Sebastian Lunz, Peter Maass, and Carola-Bibiane Schönlieb. On the connection between adversarial robustness and saliency map interpretability. *ICML*, May 2019. 1

Andrea Fontanari, Iddo Eliazar, Pasquale Cirillo, and Cornelis W. Oosterlee. Portfolio risk and the quantum majorization of correlation matrices. *IMA Journal of Management Mathematics*, 2021. 3

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. The pile: An 800gb dataset of diverse text for language modeling. *CoRR*, December 2020. 23

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. *CoRR*, June 2024. 3, 13, 21

Peiran Gao, Eric Trautmann, Byron Yu, Gopal Santhanam, Stephen Ryu, Krishna Shenoy, and Surya Ganguli. A theory of multineuronal dimensionality, dynamics and measurement. *bioRxiv*, November 2017. 3

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. *CoRR*, September 2021. 10

Ziv Goldfeld, Ewout van den Berg, Kristjan Greenewald, Igor Melnyk, Nam Nguyen, Brian Kingsbury, and Yury Polyanskiy. Estimating information flow in deep neural networks. *ICML*, May 2019. 3

Kaarel Hänni, Jake Mendel, Dmitry Vaintrob, and Lawrence Chan. Mathematical models of computation in superposition. *ICML MI Workshop*, August 2024. 3

M. O. Hill. Diversity and evenness: A unifying notation and its consequences. *Ecology*, March 1973. 19

Geoffrey E Hinton. Distributed representations. *Carnegie Mellon University*, 1984. 2

Jesse Hoogland, George Wang, Matthew Farrugia-Roberts, Liam Carroll, Susan Wei, and Daniel Murfet. The developmental landscape of in-context learning. *CoRR*, February 2024. 9

Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. *NeurIPS*, August 2019. 1

E. T. Jaynes. Information theory and statistical mechanics. *Phys. Rev.*, May 1957. 20

F. Jelinek, R. L. Mercer, L. R. Bahl, and J. K. Baker. Perplexity—a measure of the difficulty of speech recognition tasks. *The Journal of the Acoustical Society of America*, December 1977. 19

Lou Jost. *Entropy and diversity*. *Oikos*, May 2006. 19, 20

Connor Kissane, Robert Krzyzanowski, Joseph Isaac Bloom, Arthur Conmy, and Neel Nanda. Interpreting attention layer outputs with sparse autoencoders. *CoRR*, June 2024. 3

Victor Lecomte, Kushal Thaman, Trevor Chow, Rylan Schaeffer, and Sanmi Koyejo. Incidental polysemanticity. *CoRR*, 2023. 3

Sangyun Lee, Hyukjoon Kwon, and Jae Sung Lee. Estimating entanglement entropy via variational quantum circuits with classical neural networks. *CoRR*, December 2023. 3

David Lindner, János Kramár, Sebastian Farquhar, Matthew Rahtz, Thomas McGrath, and Vladimir Mikulik. Tracr: Compiled transformers as a laboratory for interpretability. *CoRR*, 2023. 7, 22

Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. Delving into transferable adversarial examples and black-box attacks. *CoRR*, February 2017. 1

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *CoRR*, 2018. 10

Benoit Mandelbrot. How long is the coast of britain? statistical self-similarity and fractional dimension. *Science*, May 1967. 8

Luke Marks, Amir Abdullah, Luna Mendez, Rauno Arike, Philip Torr, and Fazl Barez. Interpreting reward models in rlhf-tuned language models using sparse autoencoders. *CoRR*, October 2023. 3

Samuel Marks, Can Rager, Eric J. Michaud, Yonatan Belinkov, David Bau, and Aaron Mueller. Sparse feature circuits: Discovering and editing interpretable causal graphs in language models. *CoRR*, March 2024. 3, 5, 9, 23

Simon C. Marshall and Jan H. Kirchner. Understanding polysemanticity in neural networks through coding theory. *CoRR*, January 2024. 3, 9

Eric J. Michaud, Ziming Liu, Uzay Girit, and Max Tegmark. The quantization model of neural scaling. *NeurIPS*, March 2023. 8, 22

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. *NeurIPS*, October 2013. 2

Neel Nanda, Lawrence Chan, Tom Lieberum, Jess Smith, and Jacob Steinhardt. Progress measures for grokking via mechanistic interpretability. *ICLR*, January 2023. 9

Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, January 2011. 3, 20

Chris Olah. Distributed representations: Composition & superposition. *Transformer Circuits Thread*, 2023. 2

Kiho Park, Yo Joong Choe, and Victor Veitch. The linear representation hypothesis and the geometry of large language models. *NeurIPS Workshop on Causal Representation Learning*, November 2023. 2

Gonccalo Paulo, Alex Troy Mallen, Caden Juang, and Nora Belrose. Automatically interpreting millions of features in large language models. *CoRR*, October 2024. 3

Alethea Power, Yuri Burda, Harri Edwards, Igor Babuschkin, and Vedant Misra. Grokking: Generalization beyond overfitting on small algorithmic datasets. *CoRR*, January 2022. 9

Senthooran Rajamanoharan, Arthur Conmy, Lewis Smith, Tom Lieberum, Vikrant Varma, János Kramár, Rohin Shah, and Neel Nanda. Improving dictionary learning with gated sparse autoencoders. *CoRR*, April 2024. 3, 13

Tilman Räuker, Anson Ho, Stephen Casper, and Dylan Hadfield-Menell. Toward transparent ai: A survey on interpreting the inner structures of deep neural networks. *TMLR*, August 2023. 1

Andrew Slavin Ross and Finale Doshi-Velez. Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients. *AAAI*, November 2017. 1

Hadi Salman, Andrew Ilyas, Logan Engstrom, Ashish Kapoor, and Aleksander Madry. Do adversarially robust imagenet models transfer better? *NeurIPS*, December 2020. 1

Shibani Santurkar, Dimitris Tsipras, Brandon Tran, Andrew Ilyas, Logan Engstrom, and Aleksander Madry. Image synthesis with a single (robust) classifier. *NeurIPS*, August 2019. 1

Adam Scherlis, Kshitij Sachan, Adam S. Jermyn, Joe Benton, and Buck Shlegeris. Polysemanticity and capacity in neural networks. *CoRR*, July 2023. 3, 9

E. Schrödinger. Discussion of probability relations between separated systems. *Mathematical Proceedings of the Cambridge Philosophical Society*, 1935. 20

Lee Sharkey, Dan Braun, and Beren Millidge. Taking features out of superposition with sparse autoencoders. *AI Alignment Forum*, 2022. 3

Myeongjin Shin, Seungwoo Lee, Junseo Lee, Mingyu Lee, Donghwa Ji, Hyeonjun Yeo, and Kabgyun Jeong. Disentangling quantum neural networks for unified estimation of quantum entropies and distance measures. *Phys. Rev. A*, December 2024. 3

Ravid Shwartz-Ziv and Naftali Tishby. Opening the black box of deep neural networks via information. *CoRR*, April 2017. 3

Adly Templeton, Tom Conerly, Jonathan Marcus, Jack Lindsey, Trenton Bricken, and Brian Chen. Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet. *Transformer Circuits Thread*, 2024. 3

Naftali Tishby, Fernando C. Pereira, and William Bialek. The information bottleneck method. *CoRR*, April 2000. 3

Theodoros Tsiligkaridis and Jay Roberts. Second order optimization for adversarial robustness and interpretability. *CoRR*, September 2020. 1

Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. *ICLR*, September 2019. 1

Dmitry Vaintrob, jake_mendel, and Kaarel. Toward a mathematical framework for computation in superposition. *AI Alignment Forum*, 2024. 3

A Theoretical Foundations

A.1 Networks as Resource-Constrained Communication Channels

Neural networks must transmit information through layers with limited dimensions. Each layer acts as a communication bottleneck where multiple features compete for neuronal bandwidth. When a network needs to represent F features using only $N < F$ dimensions, it uses lossy compression—superposition.

This resource scarcity creates a natural analogy to communication theory. Just as telecommunications systems multiplex multiple signals through shared channels, neural networks multiplex multiple features through shared dimensions. Our measurement framework formalizes this intuition by quantifying how efficiently networks allocate their limited representational budget across competing features.

A.2 L1 Norm as Optimal Budget Allocation

The sparse autoencoder's ℓ_1 regularization creates an explicit budget constraint on feature activations:

$$\mathcal{L}_{\text{SAE}} = \|\boldsymbol{h} - \boldsymbol{W}_{\text{sae}}^T \boldsymbol{z}\|_2^2 + \lambda \|\boldsymbol{z}\|_1 \tag{11}$$

The penalty term $\lambda\|\boldsymbol{z}\|_1 = \lambda \sum_i |z_i|$ enforces that the total activation budget $\sum_i |z_i|$ remains bounded. This creates competition where features must justify their budget allocation by contributing to reconstruction quality.

From the first-order optimality conditions of SAE training, the magnitude $|z_i|$ for any active feature satisfies:

$$|z_i| = \frac{1}{\lambda} |\boldsymbol{w}_i^T (\boldsymbol{h} - \boldsymbol{W}_{-i}^T \boldsymbol{z}_{-i})| \tag{12}$$

where $\boldsymbol{W}_{-i}$ excludes feature i. This reveals that $|z_i|$ measures the marginal contribution of feature i to reconstruction quality—exactly the budget allocation that optimally balances reconstruction accuracy against sparsity. Our probability distribution therefore has meaning as "relative feature strength":

$$p_i = \frac{\mathbb{E}[|z_i|]}{\sum_j \mathbb{E}[|z_j|]} = \frac{\text{expected budget allocation to feature } i}{\text{total representational budget}} \tag{13}$$

This fraction represents how much of the network's limited representational resources are optimally allocated to feature i under the SAE's constraints. Alternative norms fail to preserve this budget interpretation. The ℓ_2 norm $\mathbb{E}[z_i^2]$ overweights outliers and breaks the linear connection to reconstruction contributions through squaring. The ℓ_∞ norm captures only peak activation while ignoring frequency of use. The ℓ_0 norm provides binary active/inactive information but loses the magnitude data essential for measuring resource allocation intensity.

A.3 Shannon Entropy as Information Capacity Measure

Given the budget allocation distribution p, the exponential of Shannon entropy provides the theoretically optimal feature count. The exponential of Shannon entropy, $\exp(H)$, is formally known as perplexity in information theory and the Hill number (order-1 diversity index) in ecology (Hill, 1973; Jost, 2006):

$$\text{PP}(p) = \exp\left(-\sum_i p_i \log p_i\right) = \prod_{i=1}^{n} p_i^{-p_i} \tag{14}$$

This quantifies the effective number of outcomes in a probability distribution—how many equally likely outcomes would yield identical uncertainty. In information theory, it represents the effective alphabet size of a communication system (Jelinek et al., 1977). In ecology, it quantifies the effective number of species in

an ecosystem (Jost, 2006). In statistical physics, it relates to the number of accessible states in a system (Jaynes, 1957). In quantum mechanics, it corresponds to the effective number of pure quantum states in a mixed state (Schrödinger, 1935).

Shannon entropy uniquely satisfies the mathematical properties required for principled feature counting (Anand et al., 2011). The measure exhibits coding optimality, equaling the minimum expected code length for optimal compression. It satisfies additivity for independent feature sets through $H(p \otimes q) = H(p) + H(q)$. Small changes in feature importance yield small changes in measured count through continuity. Uniform distributions where all features are equally important maximize the count. Adding features with positive probability monotonically increases the count. These axioms uniquely characterize Shannon entropy up to a multiplicative constant, making $\exp(H(p))$ the theoretically principled choice for aggregating feature importance into an effective count.

In quantum systems, von Neumann entropy $S(\rho) = -\mathrm{Tr}(\rho \log \rho)$ measures entanglement, with $e^{S(\rho)}$ representing effective pure states participating in a mixed quantum state (Nielsen & Chuang, 2011). Neural superposition exhibits parallel structure: just as quantum entanglement creates non-separable correlations that cannot be decomposed into independent subsystem states, neural superposition creates feature representations that cannot be cleanly separated into individual neuronal components. Both phenomena involve compressed encoding of information—quantum entanglement distributes correlations across subsystems resisting local description, while neural superposition distributes features across neurons resisting individual interpretation. Our measure $e^{H(p)}$ captures this compression by quantifying the effective number of features participating in the neural representation, analogous to how $e^{S(\rho)}$ quantifies effective pure states in an entangled quantum mixture.

Higher-order Hill numbers provide different sensitivities to rare versus common features:

$$^{q}\mathrm{D} = \left(\sum_{i=1}^{n} p_i^q \right)^{1/(1-q)} \tag{15}$$

where $q = 1$ gives our exponential entropy measure (via L'Hôpital's rule), $q = 0$ counts non-zero components, and $q = 2$ gives the inverse Simpson concentration index (participation ratio in statistical mechanics).

A.4 Rate-Distortion Theoretical Foundation

Our measurement framework emerges from two nested rate-distortion problems that formalize the intuitive resource allocation perspective. The neural network layer itself solves:

$$R_{\mathrm{NN}}(D) = \min_{p(\boldsymbol{h}|\boldsymbol{x}):\mathbb{E}[d(\boldsymbol{y}, f(\boldsymbol{h}))] \leq D} I(\boldsymbol{X}; \boldsymbol{H}) \tag{16}$$

where the layer width N constrains the mutual information $I(\boldsymbol{X}; \boldsymbol{H})$ that can be transmitted, while D represents acceptable task performance degradation. When the optimal solution requires representing $F > N$ features, superposition emerges naturally as the rate-optimal encoding strategy.

The sparse autoencoder solves a complementary problem:

$$R_{\mathrm{SAE}}(D) = \min_{p(\boldsymbol{z}|\boldsymbol{h}):\mathbb{E}[\|\boldsymbol{h} - \hat{\boldsymbol{h}}\|_2^2] \leq D} \mathbb{E}[\|\boldsymbol{z}\|_1] \tag{17}$$

where sparsity $\|\boldsymbol{z}\|_1$ acts as the rate constraint and reconstruction error as distortion. This dual structure justifies SAE-based measurement: we quantify the effective rate required to represent the network's compressed internal information under sparsity constraints.

The SAE optimization can be viewed as an information bottleneck problem balancing information preservation $\mathbb{E}[\|\boldsymbol{h} - g(\boldsymbol{z})\|_2^2]$ against information cost $\lambda \mathbb{E}[\|\boldsymbol{z}\|_1]$. Under this interpretation, $\mathbb{E}[|z_i|]$ represents the

information cost of including feature i in the compressed representation, making our probability distribution a natural measure of information allocation across features.

A.5 Critical Assumptions and Failure Modes

Our method measures effective representational diversity under sparse linear encoding, which approximates but does not exactly equal the number of distinct computational features. We must carefully assess the conditions under which this approximation holds.

Feature Correspondence Assumption. We assume SAE dictionary elements correspond one-to-one with genuine computational features. This assumption fails through feature splitting where one computational feature decomposes into multiple SAE features, artificially inflating counts. Feature merging combines multiple computational features into one SAE feature, deflating counts. Ghost features represent SAE artifacts without computational relevance (Gao et al., 2024). Incomplete coverage occurs when SAEs miss computationally relevant features entirely.

Linear Representation Assumption. We assume features combine primarily through linear superposition in activation space. Real networks violate this through hierarchical structure where low-level and high-level features aren't interchangeable. Gating mechanisms allow some features to control whether others activate (Elhage et al., 2022). Combinatorial interactions emerge when meaning comes from feature combinations rather than individual contributions (Black et al., 2022).

Magnitude-Importance Correspondence. We assume $|z_i|$ reflects feature i's computational importance. This breaks when SAE reconstruction preserves irrelevant details while missing computational essentials, when features interact nonlinearly in downstream processing (Engels et al., 2024), or when feature importance depends heavily on context rather than magnitude.

Independent Information Assumption. We assume Shannon entropy correctly aggregates information across features. This fails when correlated features don't contribute independent information, when synergistic information means feature pairs provide more information together than separately, or when redundant encoding has multiple features encoding identical computational factors.

The approximation captures genuine signal about representational complexity under specific conditions. The measure works best when features combine primarily through linear superposition, activation patterns are sparse with balanced importance, SAEs achieve high reconstruction quality on computationally relevant information, and representational structure is relatively flat rather than hierarchical. The approximation degrades with highly hierarchical representations, dense activation patterns with complex feature interactions, poor SAE reconstruction quality, or extreme feature importance skew. Despite these limitations, the measure provides principled approximation rather than exact counting, with primary value in comparative analysis across networks and training regimes.

A.6 Why Eigenvalue Decomposition Fails for SAE Analysis

Following the quantum entanglement analogy, one might consider eigenvalue decomposition of the covariance matrix:

$$\boldsymbol{\Sigma} = \frac{1}{n}\boldsymbol{A}\boldsymbol{A}^T \tag{18}$$

where $\boldsymbol{A}$ represents the activation matrix. Eigenvalues $\{\lambda_1, \lambda_2, \ldots, \lambda_n\}$ represent explained variance along principal components, normalized to form a probability distribution:

$$p_i = \frac{\lambda_i}{\sum_{i=1}^{n} \lambda_i} \tag{19}$$

This approach faces fundamental rank deficiency when applied to SAEs. Expanding from lower dimension (N neurons) to higher dimension ($D > N$ dictionary elements) yields covariance matrices with rank at most N, making detection of more than N features impossible regardless of SAE capacity.

Our activation-based approach circumvents this limitation by directly measuring feature utilization through activation magnitude distributions rather than intrinsic dimensionality. This enables superposition quantification with overcomplete SAE dictionaries.

A.7 Adaptation to Convolutional Networks

Convolutional neural networks organize features across channels rather than spatial locations. For CNN layers with activations $\mathcal{X} \in \mathbb{R}^{B \times C \times H \times W}$, we measure superposition across the channel dimension while accounting for spatial structure.

We extract features from each spatial location's channel vector independently, then aggregate when computing feature probabilities:

$$p_i = \frac{\sum_{b,h,w} |z_{b,i,h,w}|}{\sum_{j=1}^{D} \sum_{b,h,w} |z_{b,j,h,w}|} \tag{20}$$

where $z_{b,i,h,w}$ represents feature i's activation at spatial position (h, w) in sample b.

This aggregation treats the same semantic feature activating at different spatial locations (e.g., edge detectors firing everywhere) as evidence for a single feature's importance rather than separate features.

B Experimental Details

B.1 Tracr Compression

We compile RASP programs using Tracr's standard pipeline with vocabulary $\{1, 2, 3, 4, 5\}$ and maximum sequence length 5. The sequence reversal program uses position-based indexing, while sorting employs Tracr's built-in sorting primitive with these parameters.

Following Lindner et al. (2023), we train compression matrices using a dual objective that ensures compressed models maintain both computational equivalence and representational fidelity:

$$\mathcal{L} = \lambda_{\text{out}} \mathcal{L}_{\text{out}} + \lambda_{\text{layer}} \mathcal{L}_{\text{layer}} \tag{21}$$

$$\mathcal{L}_{\text{out}} = \text{KL}(\text{softmax}(\mathbf{y}_c), \text{softmax}(\mathbf{y}_o)) \tag{22}$$

$$\mathcal{L}_{\text{layer}} = \frac{1}{L} \sum_{i=1}^{L} \|\mathbf{h}_i^{(o)} - \mathbf{h}_i^{(c)}\|_2^2 \tag{23}$$

where $\mathbf{y}_c$ and $\mathbf{y}_o$ denote compressed and original logits, and $\mathbf{h}_i^{(o)}$, $\mathbf{h}_i^{(c)}$ represent original and compressed activations at layer i.

Hyperparameters: $\lambda_{\text{out}} = 0.01$, $\lambda_{\text{layer}} = 1.0$, learning rate 10^{-3}, temperature $\tau = 1.0$, maximum 500 epochs with early stopping at 100% accuracy. We use Adam optimization and train separate compression matrices for each trial. For each compressed model achieving perfect accuracy, we extract activations from all residual stream positions across 5 trials. SAEs use fixed dictionary size 100, L1 coefficient 0.1, learning rate 10^{-3}, training for 300 epochs with batch size 128. We analyze the final layer activations (post-MLP) for consistency across compression factors.

B.2 Multi-Task Sparse Parity Experiments

Dataset Construction. We use the multi-task sparse parity dataset from Michaud et al. (2023) with 3 tasks and 4 bits per task. Each input consists of a 3-dimensional one-hot control vector concatenated with

12 data bits (total dimension 15). For each sample, the control vector specifies which task is active, and the label is computed as the parity (sum modulo 2) of the 4 bits corresponding to that task. This creates a dataset where ground truth bounds the number of meaningful features while maintaining task complexity.

Model Architecture. Simple MLPs with architecture Input(15) $\rightarrow$ Linear(h) $\rightarrow$ ReLU $\rightarrow$ Linear(1), where $h \in \{16, 32, 64, 128, 256\}$ for capacity experiments. We apply interventions (dropout) to hidden activations before the ReLU nonlinearity. Training uses Adam optimizer (lr=0.001), batch size 64, for 300 epochs with BCEWithLogitsLoss. Dataset split: 80% train, 20% test with stratification by task and label.

Intervention Protocols. Dropout experiments: Applied to hidden activations with rates [0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9]. **Dictionary scaling**: Expansion factors [0.5, 1.0, 2.0, 4.0, 8.0, 16.0] relative to hidden dimension, with L1 coefficients [0.01, 0.1, 1.0, 10.0], maximum dictionary size capped at 1024. Each configuration tested across 5 random seeds with 3 SAE instances per configuration for stability measurement.

SAE Architecture and Training. Standard autoencoder with tied weights: $\mathbf{z} = \text{ReLU}(\mathbf{W}_{\text{enc}}\mathbf{x} + \mathbf{b})$, $\mathbf{x}' = \mathbf{W}_{\text{dec}}\mathbf{z}$ where $\mathbf{W}_{\text{dec}} = \mathbf{W}_{\text{enc}}^T$. Dictionary size typically 4$\times$ layer width unless specified otherwise.

Loss function: $\mathcal{L} = ||\mathbf{x} - \mathbf{x}'||_2^2 + \lambda||\mathbf{z}||_1$ with L1 coefficient $\lambda = 0.1$ (unless testing λ sensitivity). Adam optimizer (lr=0.001), batch size 128, 300 epochs. For stability analysis, we train 3-5 SAE instances per configuration with different random seeds and report mean $\pm$ standard deviation.

B.3 Grokking

Task and Architecture. Modular arithmetic task: $(a + b) \bmod 53$ using sparse training data (40% of all possible pairs, 60% held out for testing). Model architecture: two-path MLP with shared embeddings.

$$\mathbf{e}_a = \text{Embedding}(a, \text{dim} = 12) \tag{24}$$

$$\mathbf{e}_b = \text{Embedding}(b, \text{dim} = 12) \tag{25}$$

$$\mathbf{h} = \text{GELU}(\mathbf{W}_1\mathbf{e}_a + \mathbf{W}_2\mathbf{e}_b) \tag{26}$$

$$\text{logits} = \mathbf{W}_3\mathbf{h} \tag{27}$$

where $\mathbf{W}_1, \mathbf{W}_2 \in \mathbb{R}^{48\times 12}$ and $\mathbf{W}_3 \in \mathbb{R}^{53\times 48}$.

Training Configuration. 25,000 training steps, learning rate 0.005, batch size 128, weight decay 0.0002. Model checkpoints saved every 250 steps (100 total checkpoints). Random seed 0 for reproducibility.

LLC Estimation Protocol. Local Learning Coefficient estimated using Stochastic Gradient Langevin Dynamics (SGLD) with hyperparameters: learning rate 3×10^{-3}, localization parameter $\gamma = 5.0$, effective inverse temperature $n_\beta = 2.0$, 500 MCMC samples across 2 independent chains. Hyperparameters selected via 5×5 grid search over epsilon range $[3 \times 10^{-5}, 3 \times 10^{-1}]$ ensuring $\varepsilon > 0.001$ for stability and $n_\beta < 100$ for β-independence.

B.4 Pythia-70M Analysis

Data Sampling and Preprocessing. 20,000 samples from Pile dataset (Gao et al., 2020), shuffled with seeds [42, 123, 456] for reproducibility. Text preprocessing: truncate to 512 characters before tokenization to prevent memory issues. Tokenization using model's native tokenizer with `max_length=512`, `truncation=True`, no padding. Samples with empty text or tokenization failures excluded.

Model and SAE Configuration. Pythia-70M model with layer specifications: embedding layer, and $\{\text{attn_out}, \text{mlp_out}, \text{resid_out}\}$ for layers 0–5. Pretrained SAEs from Marks et al. (2024) with dictionary size $64 \times 512 = 32,768$ features per layer. SAE weights loaded from subdirectories following pattern: `layer_type/10_32768/ae.pt`.

Activation Processing. Activations extracted using nnsight tracing with error handling for failed forward passes. Feature activations accumulated across all token positions and samples: $\text{feature_sum}_i = \sum_{\text{samples,positions}} |\mathbf{z}_i|$. Feature count computed from accumulated sums using entropy-based measure. Memory management: explicit cleanup of activation tensors and CUDA cache clearing between seeds.

B.5 Adversarial Robustness

B.5.1 Model Architectures

Simple Models (Single Hidden Layer)

- **SimpleMLP**: Input(784) → Linear(h) → ReLU → Linear(output)
  - Hidden dimensions $h \in 8, 32, 128, 512$
- **SimpleCNN**: Input → Conv2d(h, 5×5) → ReLU → MaxPool(2) → Linear(output)
  - Filter counts $h \in 8, 16, 32, 64$

Standard Models

- **StandardMLP**: Input(784) → Linear($4h$) → ReLU → Linear($2h$) → ReLU → Linear(h) → ReLU → Linear(output)
  - Base dimension $h = 32$, yielding layer widths [128, 64, 32]
- **StandardCNN**: LeNet-style architecture
  - Conv2d(1, h, 3×3) → ReLU → MaxPool(2)
  - Conv2d(h, $2h$, 3×3) → ReLU → MaxPool(2)
  - Linear($4h$) → ReLU → Linear(output)
  - Base dimension $h = 16$

CIFAR-10 Models

- **CIFAR10CNN**: Three-block CNN with batch normalization
  - Conv2d(3, h, 3×3) → BN → ReLU → MaxPool(2)
  - Conv2d(h, $2h$, 3×3) → BN → ReLU → MaxPool(2)
  - Conv2d($2h$, $4h$, 3×3) → BN → ReLU → MaxPool(2)
  - Dropout(0.2) → Linear(output)
  - Base dimension $h = 32$
- **ResNet-18**: Modified for CIFAR-10
  - Initial: Conv2d(3, 64, 3×3, stride=1, padding=1)
  - MaxPool replaced with Identity
  - Standard ResNet-18 blocks [2, 2, 2, 2]
- **WideResNet**: ResNet-18 with variable width
  - Width factors: $1/16, 1/8, 1/4, 1/2, 1, 2, 4, 8$
  - Initial channels: $16 \times$ width factor
  - Block channels: $16, 32, 64, 128 \times$ width factor

B.5.2 Training Protocols

MNIST/Fashion-MNIST:

- Optimizer: SGD with momentum 0.9
- Learning rate: 0.01, MultiStep decay at epochs [50, 75]
- Weight decay: 10^{-4}
- Epochs: 100
- Batch size: 128
- PGD: 40 steps, step size $\alpha = 0.01$
- FGSM: Single step, $\alpha = \epsilon$

CIFAR-10:

- Optimizer: SGD with momentum 0.9
- Learning rate: 0.1, MultiStep decay at epochs [100, 150]
- Weight decay: 5×10^{-4}
- Epochs: 200
- Batch size: 128
- PGD: 10 steps, step size $\alpha = 2/255$
- FGSM: Single step, $\alpha = \epsilon$

B.6 SAE Configuration

- Dictionary size: $4N$ ($4\times$ layer width)
- L1 coefficient: 0.1
- Optimizer: Adam, learning rate 10^{-3}
- Training: 800 epochs with early stopping (patience 50)
- Activation collection: 10,000 samples from test set
- Separate SAEs trained per layer