Why Does Robustness Reduce Superposition?

Adam Elimadi
Independent Researcher
elimadadam@gmail.com

Abstract

The study of adversarial examples and their origins remains an open area of research. Mechanistic interpretability, and superposition in particular, offers new avenues for approaching this problem. Gorton & Lewis (2025) demonstrate that adversarial examples arise from superposition and show empirically that adversarial training reduces superposition, yet provide no mechanistic account of why this occurs. We present an empirical explanation inspired by the feature taxonomy of Ilyas et al. (2019), tracing the following chain of causalities: adversarial training abandons non-robust features $\rightarrow$ fewer total features to represent $\rightarrow$ less superposition.

1 Introduction

Explaining the existence of adversarial examples has been an open problem since Goodfellow et al. (2015) first showed that imperceptible perturbations reliably fool otherwise accurate models. Ilyas et al. (2019) offered an influential account: models rely on non-robust features, patterns that are predictive on clean data but do not survive small perturbations. This explains classification behavior, but says nothing about how those features are represented internally. Gorton & Lewis (2025) supplied that missing piece by connecting adversarial vulnerability to superposition — the packing of more features into a representation than it has dimensions to hold cleanly (Elhage et al., 2022) — showing that adversarial examples exploit superposed representations, and that adversarial training empirically reduces superposition. What their result does not explain is the mechanism: why training against perturbations should change how many features a model chooses to represent at all.

This paper supplies that mechanism. Using the same toy-model setup as Gorton & Lewis (2025), we show that adversarially trained models represent systematically fewer features than standardly trained ones, and that the discarded features are exactly the ones Ilyas et al. (2019)'s taxonomy would classify as non-robust. We establish this first indirectly, through interference geometry and represented feature count, and then directly, using a synthetic dataset in which each feature's robust/non-robust identity is assigned in advance, rather than inferred from model behavior after training.

Our contributions are three-fold:

- We show that adversarially trained models consistently drop more features than standardly trained models, across our sparsity sweep.
- We show that dropped features carry higher average interference than retained features, and that retained features cluster near antipodal interference (≈ -1), the configuration that minimizes cross-feature interference.
- Using a synthetic dataset with a known ground-truth partition, we show that the dropped features correspond exactly to the non-robust feature set.

2 Background

2.1 Superposition

Superposition occurs when a model encodes more features than there are neurons (Elhage et al., 2022). We control it by intervening on sparsity when training toy models, following the setup of Elhage et al. (2022). Sparsity is governed by a threshold $S \in [0, 1]$, which determines the activity of each component of the input $x \in \mathbb{R}^m$:

$$[x]_i = \begin{cases} x_i & \text{if } x_i > S \\ 0 & \text{otherwise} \end{cases}$$

A feature x_i is therefore active with probability $p(A) = 1 - S$.

2.2 Interference

Elhage et al. (2022) define interference as the geometry of features in activation space, measured by the degree of non-orthogonality between feature directions. For two feature directions $W_{:,i}, W_{:,j} \in \mathbb{R}^n$, they are orthogonal when $W_{:,i}^\top W_{:,j} = 0$, and exhibit interference when $W_{:,i}^\top W_{:,j} \neq 0$. The more strictly positive this inner product, the more likely spurious activations are to corrupt the reconstruction of feature i, constituting harmful interference.

2.3 Adversarial Examples

An adversarial example is a perturbation of $x \in \mathbb{R}^m$, imperceptible to humans, that maximizes the MSE loss within an ℓ_2 ball of radius ε:

$$x_{\text{adv}} = x + \underset{\|\delta\|_2 \leq \varepsilon}{\arg\max}\, \mathcal{L}(x + \varepsilon \cdot \nabla_x \mathcal{L}) \tag{1}$$

Following Gorton & Lewis (2025), a small noise term is added to x prior to attack generation to avoid gradient masking, after which a one-step ℓ_2 gradient attack is applied. The concept builds on foundational work in adversarial robustness (Goodfellow et al., 2015).

3 Setup

3.1 Toy Models

We adopt the simplified toy model of Elhage et al. (2022) as employed by Gorton & Lewis (2025). Let $W \in \mathbb{R}^{n \times m}$, with $n = 20$ and $m = 100$. Given inputs $x \sim \mathcal{D}$, $\mathcal{D} \subset \mathbb{R}^m$, hidden representations are computed as $h = Wx \in \mathbb{R}^n$ and features are reconstructed via $\hat{x} = \text{ReLU}(W^\top h + b)$. The training objective is $\mathcal{L} = \|x - \hat{x}\|_2^2$. Higher sparsity leads the model to represent more features and, consequently, to exhibit more superposition.

3.2 Measuring Superposition

Elhage et al. (2022) quantify superposition as $n/\|W\|_F$, the number of dimensions per feature. We adapt this by taking the squared Frobenius norm per dimension, obtaining a quantity proportional to the total energy distributed across hidden dimensions:

$$\Phi = \|W\|_F^2 / n \tag{2}$$

Φ increases monotonically with superposition and sparsity.

3.3 Representational Power

We define representational power as the squared Frobenius norm of W, measuring the total energy distributed across all feature directions $W_{:,i}$:

$$P = \|W\|_F^2 \tag{3}$$

Note that $\Phi = P/n$, so representational power and superposition are proportional given fixed n.

3.4 Feature Representation Threshold

To determine which features are represented and which are dropped, we measure the norm $\|W_{:,i}\|_2$ of each feature column following Elhage et al. (2022), applying the threshold $\tau = 0.01$:

$$\|W_{:,i}\|_2 \begin{cases} < \tau \Rightarrow \text{feature } i \text{ is dropped} \\ \geq \tau \Rightarrow \text{feature } i \text{ is represented} \end{cases} \tag{4}$$

3.5 Measuring Interference

Let $G = WW^\top \in \mathbb{R}^{n \times n}$ be the Gram matrix of W. The interference matrix is obtained by zeroing the diagonal, retaining only pairwise cross-feature terms:

$$I = G - \text{diag}(G) \tag{5}$$

The per-feature total interference is obtained by summing I along each row:

$$\iota_i = \sum_{j=1}^{n} I_{ij} \tag{6}$$

Given the binary mask induced by Eq. (4), the mean total interference of kept and dropped features is then computed separately as:

$$\bar{\iota}_{\text{kept}} = \frac{1}{|K|} \sum_{i:m_i=1} \iota_i, \qquad \bar{\iota}_{\text{dropped}} = \frac{1}{|D|} \sum_{i:m_i=0} \iota_i \tag{7}$$

3.6 Adversarial Training Protocol

Following Gorton & Lewis (2025) and the adversarial training framework of Madry et al. (2019), models are trained over a sweep of sparsity levels on a mixture of clean and adversarial examples:

$$\mathcal{L}_{\text{adv}} = \alpha \cdot \mathcal{L}(x) + (1 - \alpha) \cdot \mathcal{L}(x_{\text{adv}}) \tag{8}$$

We set $\alpha = 0.5$ to balance clean and robust accuracy and prevent collapse in either regime. Attacks are generated via Eq. (1) with $\varepsilon = \frac{0.1}{B} \sum_{x \in \mathcal{B}} \|x\|_2$, where $\mathcal{B}$ denotes the training batch of size B. Models are trained for 150,000 steps at a learning rate of 10^{-3}.

3.7 Sparsity Sweep

To cleanly isolate our theory empirically, we exclude both extremes of the sparsity range. At $S \leq 0.7$, models drop most features regardless of training regime, as sparsity is too low to motivate superposition and interference is easily avoided. At $S \geq 0.98$, both models represent all features at negligible interference cost. We focus on the ℓ_2 threat model, a standard adversarial threat model studied extensively in prior work (Carlini & Wagner, 2017). We therefore sweep over $\mathcal{S} = \{0.78, 0.80, 0.88, 0.90\}$, the range in which the adversarially trained model consistently drops more features than the standard model — an effect we attribute to the presence of distinct feature classes, rather than to the routine optimization of feature benefit against interference cost.

4 Representational Power and Feature Dropping

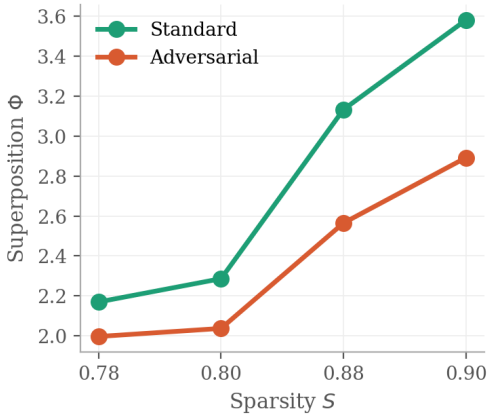
We train two models over $\mathcal{S}$ — one under the adversarial training protocol (Eq. (8)) and one under standard training — with $x \sim \mathcal{U}([0,1]^m)$.

4.1 Replication

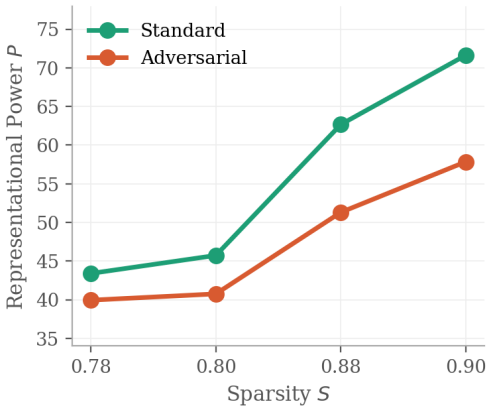
We first reproduce the core finding of Gorton & Lewis (2025) within our experimental range before proceeding to explain it. Measuring Φ (Eq. (2)) under both training regimes across $\mathcal{S}$, we observe that adversarially trained models consistently exhibit lower superposition than their standardly trained counterparts, with the gap widening as sparsity increases.

4.2 Representational Power and Feature Dropping

We measure representational power (Eq. (3)) across $\mathcal{S}$ for each training regime, and, to investigate why the adversarially trained model exhibits lower representational power, we compute $\|W_{:,i}\|_2$ for all i. Applying Eq. (4), we find that adversarially trained models drop more features than standardly trained models across all $S \in \mathcal{S}$.



(a) Superposition Φ (Eq. (2)) across $\mathcal{S}$ for both training regimes. Adversarial training consistently reduces superposition, reproducing the finding of Gorton & Lewis (2025) within our sparsity range.

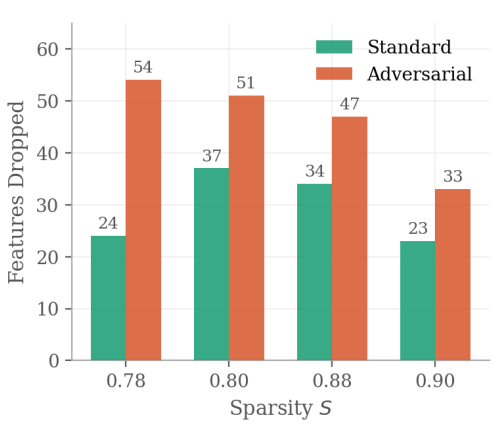


(b) Representational power P (Eq. (3)) across $\mathcal{S}$ for both training regimes. The adversarially trained model consistently exhibits lower representational power, with the gap widening as sparsity increases.

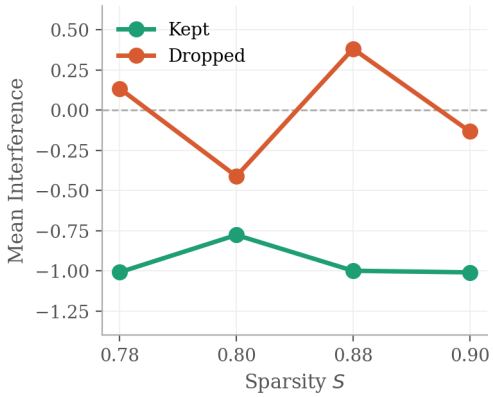
Figure 1: Superposition and representational power fall together as adversarial training discards features.

4.3 Feature Dropping and Interference

We examine the dropped features through the lens of interference. Normalizing W and computing the interference matrix (Eq. (5)), we calculate each column's total interference with the rest, then average across retained and dropped groups separately. We find that dropped features either exhibit higher interference than retained ones, or — when both groups are negative — the adversarial model preferentially retains those closest to -1.



(a) Number of features dropped at each $S \in \mathcal{S}$ under adversarial and standard training. The adversarially trained model drops more features at every sparsity level.



(b) Mean interference of kept versus dropped features across $\mathcal{S}$ for the adversarially trained model. Dropped features consistently exhibit higher or less-negative mean interference than retained features.

Figure 2: Adversarial training drops more features than standard training, and the dropped features are the ones with the most costly interference geometry.

These results establish that adversarially trained models reduce superposition by dropping more features than their standardly trained counterparts, and that the dropped features are those the model identifies as geometrically costly. The preference for retaining features with interference nearest to -1 is particularly telling: a value of -1 corresponds to antipodal feature directions, the configuration that maximally suppresses cross-feature activation. This aligns with Elhage et al. (2022), who observe that models preferentially pack features into opposing directions to minimize spurious reconstruction. A key question remains: do the dropped features correspond to the robust or non-robust classes of Ilyas et al. (2019) and Li & Li (2025)?

5 Robust and Non-Robust Feature Recovery

5.1 Structured Data Construction

To answer this question, Section 4 must be reproduced under a data distribution where the ground-truth identity of robust and non-robust features is known a priori. We therefore construct a synthetic dataset that operationalizes the feature taxonomy of Ilyas et al. (2019) and Li & Li (2025) under the constraints of our toy model setting.

Robust features are broadly characterized as high-signal directions that encode stable, generalizable patterns, whereas non-robust features are low-amplitude, high-frequency directions that are predictive under clean inputs but easily disrupted by adversarial perturbations. More precisely, Ilyas et al. (2019) show that models depend substantially on non-robust features for classification, yet those features cease to correlate with the correct label after an attack. Li & Li (2025) further characterize this taxonomy by establishing that robust features carry greater signal amplitude than non-robust ones, while non-robust features are denser across the feature space.

We note, however, that our setting differs substantially from the classification context in which this taxonomy was originally defined. Our experiments are conducted on toy models optimizing a reconstruction loss (MSE), which bears structural similarities to linear regression but falls well short of the complexity of classification over real-world data with semantic labels. The instantiation of robust and non-robust features below is therefore necessarily an approximation made under these constraints.

We instantiate this taxonomy as follows. Let $x \sim \mathcal{U}([0,1]^m)$ with the sparsity mask of Section 2.1 applied. A uniformly random permutation π of $\{1,\ldots,m\}$ partitions the feature indices into a robust set $\mathcal{R} = \{\pi(1),\ldots,\pi(n_r)\}$ and a non-robust set $\mathcal{N} = \{\pi(n_r+1),\ldots,\pi(m)\}$, with $|\mathcal{R}| = n_r$ and $|\mathcal{N}| = m - n_r$. Each partition is then amplitude-scaled according to:

$$x_{:,i} \leftarrow \begin{cases} a_r \cdot x_{:,i} & i \in \mathcal{R} \\ a_{nr} \cdot x_{:,i} & i \in \mathcal{N} \end{cases} \tag{9}$$

with $a_r = 6.0$ and $a_{nr} = 0.2$. Robust features thus carry high amplitude — strong, stable signal — while non-robust features carry low amplitude, making them more susceptible to erasure under ℓ_2 perturbations. The partition $(\mathcal{R}, \mathcal{N})$ is retained at generation time and serves as the ground truth against which the model's dropped feature indices are evaluated. The full data generation procedure is given in Appendix A.

To remain consistent with the definition of Li & Li (2025), we set $|\mathcal{N}| > |\mathcal{R}|$, making non-robust features denser than robust ones. We note that models can be sensitive to the amplitude ratio between robust and non-robust features; the specific values ($a_r = 6.0$, $a_{nr} = 0.2$) were chosen to reflect the signal-to-noise distinction while maintaining numerical stability during training.

5.2 Results

Training on data generated via Eq. (9), we obtain the same findings as in Section 4. The structured data, however, yields a substantially cleaner pattern: the adversarially trained model drops exactly $|\mathcal{N}| = 70$ features at every $S \in \mathcal{S}$, regardless of sparsity level. Notably, at $S \in \{0.88, 0.90\}$, the standardly trained model drops no features at all — yet the adversarially trained model still drops exactly $|\mathcal{N}|$ — indicating that, despite the model having enough sparsity to represent all features, adversarial training remains highly sensitive to non-robust features. This sensitivity being sparsity-invariant suggests that those features directly interfere with the model's ultimate goal:

$$\min_{\theta} \mathbb{E}_{x \sim \mathcal{D}}[\mathcal{L}(x_{\text{adv}}; \theta)]$$

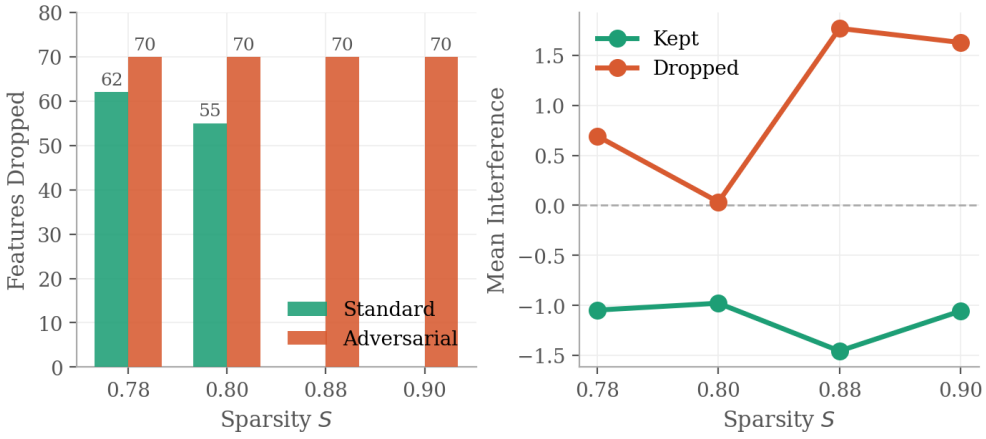


Figure 3: **Left:** Features dropped at each $S \in \mathcal{S}$ under structured data. The adversarially trained model drops exactly $|\mathcal{N}| = 70$ at every sparsity level. **Right:** Mean interference of kept versus dropped features across $\mathcal{S}$ for the adversarially trained model under structured data. Dropped features carry positive net interference; kept features carry negative net interference.

In order to determine whether the dropped features are primarily the non-robust ones, we map the ground truth indices of non-robust features retrieved at data creation time and compare them to the indices of the dropped features retrieved at the end of training. We find

that they correspond exactly to $\mathcal{N}$ at every $S \in \mathcal{S}$, confirming that the adversarial model explicitly targets the non-robust feature set.

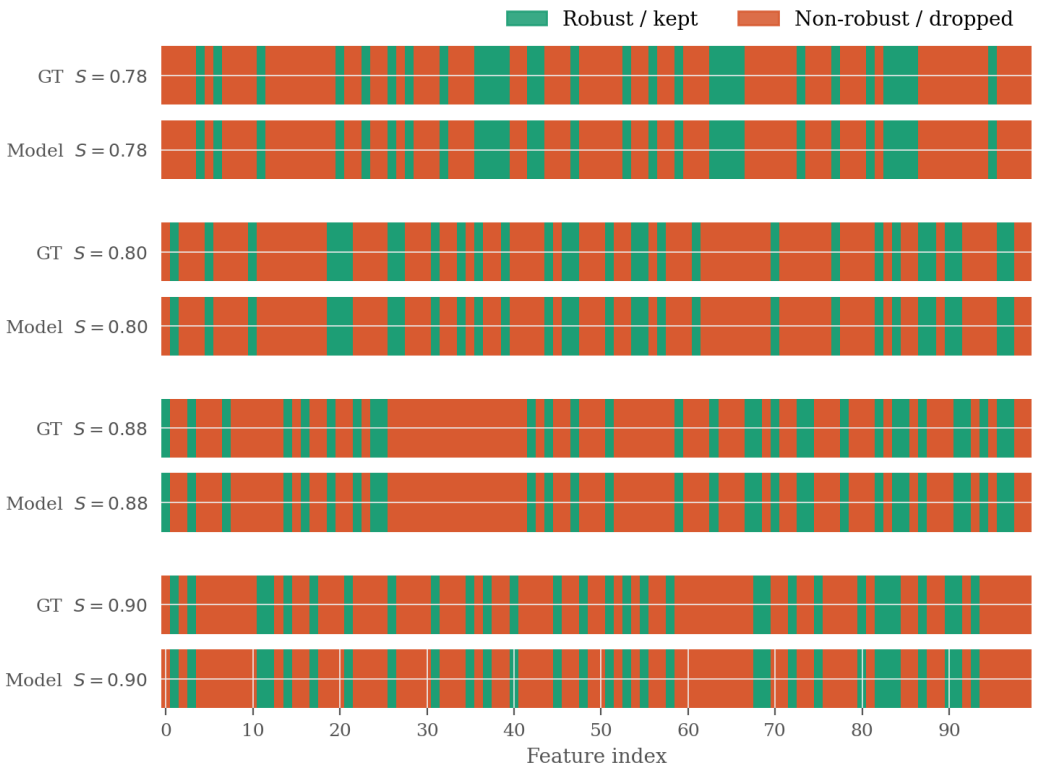


Figure 4: Ground truth partition $(\mathcal{R}, \mathcal{N})$ versus model-dropped indices across $\mathcal{S}$. For each sparsity level, the top row shows the ground truth and the bottom row shows the adversarial model output. Identical patterns confirm that the adversarial model precisely targets $\mathcal{N}$ at every sparsity level.

5.3 Theoretical Intuition

Our empirical findings raise a natural question: why do adversarially trained models systematically drop non-robust features? We offer a theoretical explanation grounded in the properties of non-robust features under adversarial attack. The argument is conceptual rather than a formal derivation, but it provides intuition for the observed behavior.

Let $\mathcal{D} = \{(x, y)\}$ with $y = x$, parameters θ, loss $\mathcal{L}$, and adversarial examples x_{adv} generated via an ℓ_2-bounded attack. We conceptualize adversarial training's primary objective as:

$$\min_{\theta} \ \mathbb{E}_{(x,y)\sim\mathcal{D}}[\mathcal{L}(x_{\text{adv}}, y; \theta)] \tag{10}$$

Building on Ilyas et al. (2019), we define a feature i as non-robust if it is highly sensitive to adversarial perturbations such that $x_{\text{adv},i} \not\approx x_i$, breaking the correspondence with its target $y_i = x_i$. Denote the set of non-robust features $\mathcal{N} \subset [m]$.

We hypothesize that features in $\mathcal{N}$ interfere with the adversarial training objective via two complementary mechanisms:

Direct loss inflation. Since $\mathcal{L} = \|y - \hat{x}\|_2^2 = \sum_i (y_i - \hat{x}_i)^2$ and y is fixed under attack, each flipped feature $i \in \mathcal{N}$ contributes a non-zero penalty term to the loss. Even a single flipped feature increases $\mathcal{L}$, and the effect compounds: inputs with more non-robust features incur proportionally higher loss. Let k denote the number of non-robust features flipped per

adversarial example. This gives:

$$\mathcal{L}_{\text{adv}}(x_{\text{adv}}, k{=}0) < \mathcal{L}_{\text{adv}}(x_{\text{adv}}, k{=}1) < \cdots < \mathcal{L}_{\text{adv}}(x_{\text{adv}}, k{=}|\mathcal{N}|) \tag{11}$$

Dropping all features in $\mathcal{N}$ eliminates these penalty contributions entirely and gives $P(\mathcal{N}_{\text{flipped}} \mid x_{\text{adv}}) = 0$, thus enabling the model to minimize Eq. (8) freely.

Costly interference. Non-robust features exhibit costly feature geometry, as evidenced by their mean total interference lying in the range $[0, 2)$ compared to retained features clustering near -1. This costly geometry induces spurious cross-feature activations that corrupt the reconstruction and further inflate $\mathcal{L}$.

Together, these two mechanisms explain why adversarially trained models prune non-robust features: they are the primary obstacle to minimizing the adversarial objective, and removing them reduces superposition while improving robustness.

6 Discussion

We have established a clean causal chain explaining why adversarial training reduces superposition: adversarially trained models drop more features than standard models, leaving fewer features to encode in the same dimensional space, and thus reducing superposition. We further demonstrated that the dropped features correspond precisely to non-robust features as defined by Ilyas et al. (2019).

These results answer our core question but raise new ones. First, why do adversarially trained models preserve the geometry of remaining features, keeping interference values largely intact (Gorton & Lewis, 2025)? Second, why does adversarial training preferentially align features in opposite directions, enabling antipodal superposition? Understanding these emergent behaviors would deepen our mechanistic account of adversarial robustness.

Our findings remain constrained by the toy model setting. The natural next step is validating our theory on real-world models using mechanistic tools like sparse autoencoders (SAEs), moving beyond controlled toy settings.

Recent work at the intersection of robustness and interpretability suggests a promising research direction. We are particularly interested in two questions: First, does superposition reduction in real adversarial models translate to meaningfully improved interpretability? Second, can we intentionally leverage the robust/non-robust feature trade-off to reduce polysemanticity in a controlled manner, without sacrificing critical representations?

Acknowledgments

Claude (Anthropic) was used to assist with LaTeX formatting and editorial refinements. All scientific content, experimental design, results, and writing are the author's own.

References

Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *IEEE Symposium on Security and Privacy (SP)*, 2017.

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. *arXiv preprint arXiv:2209.10652*, 2022.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations (ICLR)*, 2015.

Lawrence Gorton and Oliver Lewis. Adversarial examples are not bugs, they are superposition. *arXiv preprint arXiv:2508.17456*, 2025.

Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Binghui Li and Yuanzhi Li. Adversarial training can provably improve robustness: Theoretical analysis of feature learning process under structured data. *arXiv preprint arXiv:2410.08503*, 2025.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations (ICLR)*, 2019.

A Data Generation Code

The full procedure used to construct the structured robust/non-robust dataset of Section 5.1 is given below.

Listing 1: Structured data generation with ground-truth robust/non-robust partition.

```
def create_data(sparsity, num_samples, n_features=100, n_robust=30):
    values = torch.rand(num_samples, n_features)
    mask = values > sparsity
    x = values * mask.float()
    pi = torch.randperm(n_features)
    robust_idx = pi[:n_robust]
    non_robust_idx = pi[n_robust:]
    x[:, robust_idx] *= 6.0
    x[:, non_robust_idx] *= 0.2
    return x, x.clone(), robust_idx, non_robust_idx
```