Adversarial Attacks Leverage Interference Between Features in Superposition

Anonymous Author(s)
Affiliation
Address
email

Abstract

Fundamental questions remain about why adversarial examples arise in neural networks. In this paper, we demonstrate that adversarial vulnerability can emerge from feature superposition—where networks represent more latent features than they have dimensions. Through controlled experiments on toy models and vision transformers (ViTs), we show how data properties induce specific superposition geometries that adversaries systematically exploit. We demonstrate that adversarial perturbations leverage interference patterns between superposed features to craft attacks, with the geometric arrangement of these features determining attack characteristics. Our framework provides a mechanistic explanation for two known phenomena: adversarial attack transferability between models with similar training regimes and class-specific vulnerability. We demonstrate these findings persist beyond toy settings with ViTs trained on CIFAR-10 with an engineered bottleneck. These results show that adversarial vulnerability can stem from efficient information encoding in neural networks, rather than from flaws in the learning process or non-robust input features.

1 Introduction

Despite extensive research on adversarial examples (AExs) [Szegedy et al., 2014, Goodfellow et al., 2014, Eykholt et al., 2018], no consensus exists on their fundamental causes. This paper presents a mechanistic interpretability perspective demonstrating that AExs can exploit interference between learned representations in superposition—a mechanism that normally enables additional representation capacity—to craft effective perturbations that manipulate model outputs.

Our account draws on the linear representation hypothesis (LRH) [Park et al., 2024] and the theory of superposition [Elhage et al., 2022]. The LRH posits that networks encode semantic features as linear directions in representation space. It is hypothesised that neural networks (NNs) can represent significantly more input features than they have neurons through superposition, leveraging near-orthogonality in high-dimensional space and compressed sensing. This efficient packing introduces interference between features. We investigate whether this interference creates systematic vulnerabilities that adversarial attacks exploit.

Contributions: Using toy models with controlled superposition, we demonstrate that projected gradient descent (PGD) attacks exploit interference patterns between superposed features, with perturbations predictable from geometric arrangements. We show that input correlations determine these geometric configurations; when correlations constrain how features can be arranged, different models converge to similar geometries. These results explain attack transferability: models with shared geometric structure exhibit high transfer rates whilst those with different geometries show minimal transfer. We replicate these findings in a ViT trained on CIFAR-10 with an engineered bottleneck. Our framework reveals adversarial vulnerability can arise from efficient information encoding rather than learning flaws or non-robust features.

2 Background

We briefly outline the tools used in our analysis: the LRH, superposition, and PGD, with full definitions in App. A.1. Let $\mathbf{x} \in \mathcal{X}$ denote the input, and $a^{(l)}(\mathbf{x}) \in \mathbb{R}^{d_l}$ the l-th layer activation.

Linear Representation and Superposition. The LRH posits that NNs represent semantic concepts (**input features**) as linear directions in activation space, which can be used as abstractions for reasoning [Park et al., 2024]. We conceptualise these as a set of M underlying **latent features** directions, $\{\mathbf{v}_j\}_{j=1}^{M} \subset \mathbb{R}^{d_l}$. Then $\mathbf{a}^{(l)}(\mathbf{x}) \approx \sum_{j=1}^{M} c_j(\mathbf{x})\mathbf{v}_j$, where $c_j(\mathbf{x})$ are magnitudes. Superposition occurs when $M > d_l$: networks represent more features than dimensions using non-orthogonal directions $\{\mathbf{v}_j\}_{j=1}^{M}$, enabled by sparse feature activation ($\mathbb{E}_{\mathbf{x}}[\|\mathbf{c}(\mathbf{x})\|_0] \ll M$). This creates polysemanticity—individual neurons representing multiple features—and interference Elhage et al. [2022].

Adversarial Attacks. Adversarial attacks force misclassification via small perturbations $\boldsymbol{\delta}$: $\mathbf{x}' = \mathbf{x} + \boldsymbol{\delta}$ with $\|\boldsymbol{\delta}\|_p \leq \epsilon$. We use PGD [Madry et al., 2018], which for untargeted attacks iteratively maximises loss: $\mathbf{x}'^{(k+1)} = \Pi_S(\mathbf{x}'^{(k)} + \alpha \mathbf{g}_k)$, where $\mathbf{g}_k$ is the normalised gradient, α is step size, and Π_S projects onto the ϵ-ball. For ℓ_∞ constraints, $\mathbf{g}_k = \text{sign}(\nabla_{\mathbf{x}}\mathcal{L}(\cdot))$; for ℓ_2, $\mathbf{g}_k = \nabla_{\mathbf{x}}\mathcal{L}/\|\nabla_{\mathbf{x}}\mathcal{L}(\cdot)\|_2$.

3 Superposition Geometry Determines Adversarial Attacks

We investigate whether AExs exploit interference between superposed features using a toy model in which we can control relationships between inputs, latent representations, and interference.

Setup. We partition input $\mathbf{x} \in \mathbb{R}^d$ into k groups $\mathbf{x} = [\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(k)}]$, where $\mathbf{x}^{(j)} \in \mathbb{R}^p$ represents input features for class j. Each $x_i^{(j)} \sim \text{Uniform}(0, 1)$ with sparsity S (probability of being zero). A two-layer network with encoder $\mathbf{h} = \sigma(\mathbf{W}_h\mathbf{x} + b_h) \in \mathbb{R}^m$ (where $m < k < d$) compresses k class representations into m dimensions, followed by a linear decoder for classification. Our primary setup uses cross entropy (CE) loss without ReLUs/biases (mean squared error (MSE) variant with ReLUs/biases in Appendix). Since input feature $x_i^{(j)}$ affects only class j, the columns $\{\mathbf{W}_h[:, i] : i \in \text{class } j\}$ align, and we interpret them as class representations $\mathbf{v}_i$. AEx are generated with PGD and must (1) change the model's prediction whilst (2) preserving the true class (*i.e.* the class with the largest sum).

3.1 Theoretical Framework

In the linear setting, we derive exact relationships between superposition geometry and adversarial vulnerability. Complete propositions in App. A.2.

Proposition 1. *The optimal input perturbations* $\boldsymbol{\delta}$ *that maximise movement toward the decision boundary under* $\|\boldsymbol{\delta}\|_2 = \epsilon$ *satisfy* $\boldsymbol{\delta} \propto \mathbf{V}^\top\mathbf{n}$*, where* $\mathbf{n}$ *is the boundary normal.*

To see how interference drives vulnerability: in binary classification between classes j and k, the boundary normal $\mathbf{n} = \mathbf{v}_j - \mathbf{v}_k$. Thus any input feature i gets perturbed proportionally to $\mathbf{v}_i^\top\mathbf{n} = \mathbf{v}_i^\top\mathbf{v}_j - \mathbf{v}_i^\top\mathbf{v}_k$, meaning the most effective perturbation strength depends directly on interference (inner products) between feature i and the class features $\mathbf{v}_j, \mathbf{v}_k$.

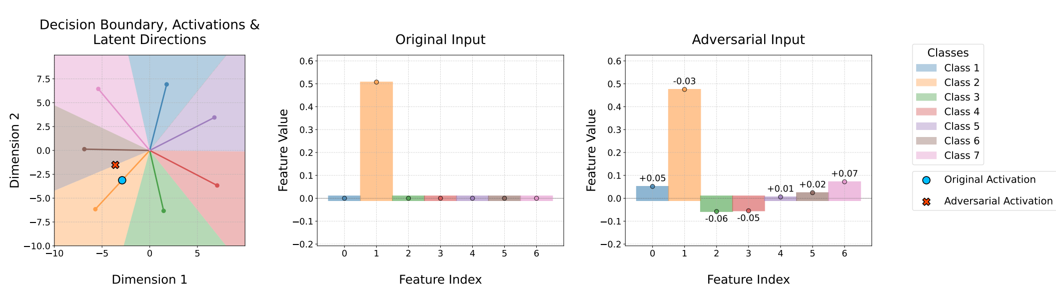


Figure 1: **An adversarial attack exploiting superposition geometry.** Middle: The original sample. Right: The adversarially perturbed sample, whose ground truth remains the same but is misclassified. Left: The original and adversarial sample in activation space.

Proposition 2. *Models with feature representations related by orthogonal transformation Q (where $Q^T Q = I$) share identical optimal attack directions, predicting perfect adversarial transfer.*

These propositions establish part of the suggested mechanism: **Data Correlations** $\xrightarrow{\text{constrain}}$ **Feature Geometry** $\xrightarrow{\text{determines}}$ **Attack Perturbations**. Propositions 1 and 2 establish how geometry determines attacks and transferability; whilst we empirically demonstrate that correlations shape geometry.

3.2 Empirical Results

We present our empirical findings qualitatively here, but provide quantitative details in App. A.3. Results remain consistent across a range of hidden dimensions, classes, and features per class.

Attacks exploit geometric interference. Figure 1 illustrates our key observation: each input feature is perturbed (both magnitude and sign) in proportion to how its latent representation aligns with the vector travelled to cross the decision boundary. We quantify this observation by comparing PGD-discovered attacks with theoretically optimal perturbations, finding near-perfect alignment. This demonstrates that perturbations are not arbitrary but predictable—the specific superposition geometry determines exactly which attacks will succeed.

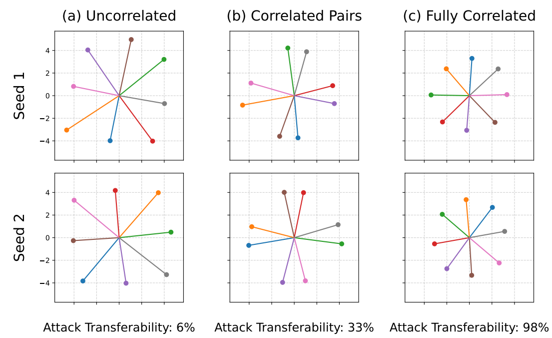


Figure 2: Input correlations determine consistent geometries across models, driving attack transferability from 6% (uncorrelated) to 98% (globally correlated).

Correlations determine geometry. Data correlations constrain how features arrange in latent space. Figure 2 shows three correlation patterns: (a) with i.i.d. data, representations order randomly between different model seeds; (b) with pairwise correlations (input feature pairs that co-activate [Elhage et al., 2022]), models develop partially constrained structures with correlated features orthogonal; (c) with global correlations (cyclic correlations where adjacent classes are more likely to co-occur), models converge to a fixed ordering (up to rotation).

Shared superposition geometry explains transferability. If data correlations create consistent latent geometries between models, models share similar interference patterns between superposed features. As shown in Figure 2, this shared interference determines adversarial transfer rates: the fully correlated models yield 98% transfer versus 6% for i.i.d.. Attacks optimised for one model's interference patterns fail when applied to models with different geometric arrangements—the perturbations no longer constructively combine.

Removing superposition eliminates attacks. With $m = k$, networks learn orthogonal class representations and no successful attacks exist: moving a sample across the decision boundary requires genuinely changing the class (*i.e.* changing the class with the highest sum). Furthermore, forcing one class feature orthogonal to others leaves its inputs completely unperturbed during attacks between other classes, confirming interference is necessary for these attacks (Appendix Fig. 5).

In our controlled experiments, adversarial vulnerability arises through a precise mechanism: input correlations constrain feature geometry, which determines attack patterns and hence transferability. These results demonstrate that superposition can stem from efficient information encoding rather than learning flaws. Whilst exact relationships hold only in our simplified setting, they suggest principles that may manifest in realistic models.

4 Attacks in Vision Models

We extend our analysis to a ViT [Dosovitskiy et al., 2020] trained on CIFAR-10 [Krizhevsky, 2009] with an engineered bottleneck to induce controlled superposition between class representations.

Setup. We train ViTs (6 layers, 512-dim embeddings, 81% accuracy) on CIFAR-10. We then replace the classification head on these base models with a bottleneck: a linear encoder that projects down to m dims followed by a decoder back to the 10 classes. We train this bottleneck with frozen ViT

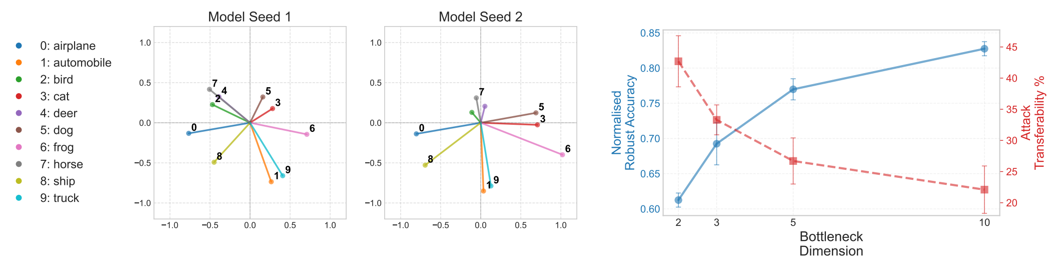


Figure 3: Left: CIFAR-10 class representation geometries remain consistent across seeds. Right: Increased superpositional pressure (smaller m) reduces robustness whilst increasing transferability.

weights, and the class representations are placed in superposition. We vary $m \in \{2, 3, 5, 10\}$ to control compression degree. AExs are generated using ℓ_∞- and ℓ_2-norm PGD and transferability measured across five seeds. See Appendix A.4 for complete results.

Results. Three key findings emerge that mirror our toy model observations:

(1) Despite random initialisation, models converge to similar class arrangements, as measured by cosine similarity, suggesting data correlations guide the formation of a superposed geometry (Figure 3, left). Semantically related classes ('cat'/'dog', 'car'/'truck') consistently cluster together.

(2) When AExs from bottlenecked models are run through the base ViTs they produce similar relative logit changes—both models respond consistently to the same perturbations. The perturbations are not solely an artefact of the bottleneck. However, without the increased compression these do not lead to misclassifications.

(3) As bottleneck dimension decreases normalised robust accuracy decreases (81%→60%) and attack transferability increases (25%→45%)(Figure 3, right). We conjecture that higher superposition reduces the degrees of freedom in potential feature geometry. With a more constrained representational space available, the network has fewer viable geometric arrangements for its class features. This leads to different model initialisations converging to more similar superposition geometries and, consequently, more shared interference patterns, which results in greater attack transferability.

5 Related Work

Superposition and representations. Recent work explores how networks pack features into limited dimensions. Elhage et al. [2022] demonstrate correlated features become orthogonal; Gurnee et al. [2023] discuss interference patterns and mitigation via non-linearities; Chan [2024] identify correlations as driving superposition. Representation geometry is further shaped by multiple factors, including spectral biases Rahaman et al. [2019], and can lead to neural collapse Kothapalli [2023].

Adversarial vulnerability. Competing explanations include non-robust features Ilyas et al. [2019], learning shortcuts Li et al. [2023], and geometric boundaries. Elhage et al. [2022] suggest superposition's link to adversarial examples (debated by Casper [2023]). Other works examine attack mechanisms: Zhang et al. [2021] describe perturbations pushing representations across boundaries; [Ganeshan et al., 2019] show PGD targets final layers; Maiya et al. [2021] find dataset-specific patterns. Transferability is attributed to representation similarities Li et al. [2023], Wang et al. [2024], with Wiedeman and Wang [2022] reducing transfer by decorrelating features between models.

6 Discussion & Concluding Remarks

We demonstrate that adversarial attacks can exploit interference patterns arising from superposed feature geometry in NNs. Data properties—correlations and sparsity—induce specific superposition geometries creating predictable vulnerabilities. These geometric arrangements determine attack characteristics and explain phenomena including transferability and class-specific susceptibility. This mechanistic account reveals superposition as a sufficient condition for adversarial vulnerability.

Limitations & future work. Our insights derive from simplified settings where we use class features in engineered superposition. Vulnerability mechanisms in large-scale models involve interference between unknown, unlabelled features across multiple layers. Future work should examine how robust training reshapes geometry and extend analysis to different attack types.

References

Stephen Casper. EIS IX: Interpretability and adversaries. *AI Alignment Forum*, February 2023.

Lawrence Chan. Superposition is not “just” neuron polysemanticity. *AI Alignment Forum*, April 2024.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, G Heigold, S Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2020.

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. *arXiv preprint arXiv:2209.10652*, 2022.

K. Eykholt, I. Evtimov, E. Fernandes, B. Li, A. Rahmati, C. Xiao, A. Prakash, T. Kohno, and D. Song. Robust physical-world attacks on deep learning visual classification. In *2018 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1625–1634, 2018.

Aditya Ganeshan, Vivek BS, and R Venkatesh Babu. FDA: Feature disruptive attack. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8069–8079, 2019.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Clément Guerner, Anej Svete, Tianyu Liu, Alexander Warstadt, and Ryan Cotterell. A geometric notion of causal probing. *arXiv preprint arXiv:2307.15054*, 2023.

Wes Gurnee, Neel Nanda, Matthew Pauly, Katherine Harvey, Dmitrii Troitskii, and Dimitris Bertsimas. Finding neurons in a haystack: Case studies with sparse probing. *Trans. Mach. Learn. Res.*, 2023.

A. Ilyas, S. Santurkar, D. Tsipras, L. Engstrom, B. Tran, and A. Madry. Adversarial examples are not bugs, they are features. In *Advances in Neural Information Processing Systems 32*, pages 125–136, 2019.

Vignesh Kothapalli. Neural collapse: A review on modelling principles and generalization. *Trans. Mach. Learn. Res.*, 2023.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009. CIFAR-10 dataset.

Ang Li, Yifei Wang, Yiwen Guo, and Yisen Wang. Adversarial examples are not real features. *Advances in Neural Information Processing Systems*, 2023.

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*, 2018.

Shishira R. Maiya, Max Ehrlich, Vatsal Agarwal, Ser-Nam Lim, Tom Goldstein, and Abhinav Shrivastava. A Frequency Perspective of Adversarial Robustness. *arXiv*, 2021.

Kiho Park, Yo Joong Choe, and Victor Veitch. The linear representation hypothesis and the geometry of large language models. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024.

Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred A. Hamprecht, Yoshua Bengio, and Aaron C. Courville. On the spectral bias of neural networks. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 5301–5310. PMLR, 2019.

C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. In *2nd International Conference on Learning Representations, ICLR 2014*, 2014.

Donghua Wang, Wen Yao, Tingsong Jiang, Xiaohu Zheng, Junqi Wu, and Xiaoqian Chen. Improving the Transferability of Adversarial Examples by Feature Augmentation. *arXiv*, 2024.

Christopher Wiedeman and Ge Wang. Disrupting adversarial transferability in deep neural networks. *Patterns*, 3(5), 2022.

Shufei Zhang, Zhuang Qian, Kaizhu Huang, Qiufeng Wang, Rui Zhang, and Xinping Yi. Towards better robust generalization with shift consistency regularization. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 12524–12534. PMLR, 18–24 Jul 2021.

A Appendix

A.1 Definitions

Formally, let $\mathbf{x} \in \mathcal{X}$ denote the input, and $a^{(l)}(\mathbf{x}) \in \mathbb{R}^{d_l}$ the activation vector of the l-th layer with dimensionality d_l.

Linear Representation Hypothesis (LRH). The LRH posits that NNs represent many variables of their computation, such as semantic properties of their inputs (**input features**), as linear directions in their activation space, which can be used as abstractions for reasoning [Park et al., 2024, Guerner et al., 2023]. We conceptualise these as a set of M underlying **latent features**, $\mathcal{F} = \{f_1, f_2, \ldots, f_M\}$. Each feature f_j corresponds to a direction $\mathbf{v}_j \in \mathbb{R}^{d_l}$. The activation vector $a^{(l)}(\mathbf{x})$ is then approximated as:

$$a^{(l)}(\mathbf{x}) \approx \sum_{j=1}^{M} c_j(\mathbf{x})\mathbf{v}_j \tag{1}$$

where $c_j(\mathbf{x})$ is the scalar magnitude of feature $\mathbf{v}_j$. If inputs $\mathbf{x}_0$ and $\mathbf{x}_1$ differ mainly in concept C (direction $\mathbf{v}_C$), their activation difference $a^{(l)}(\mathbf{x}_1) - a^{(l)}(\mathbf{x}_0) \approx k \cdot \mathbf{v}_C$ aligns with $\mathbf{v}_C$, where k reflects the change in C. This bias towards representing linear features is hypothesised because linear separability: 1) allows networks to easily recognise and manipulate features; 2) dot products with subsequent layer weights efficiently process such directional features. Growing research supports this [Gurnee et al., 2023, Park et al., 2024].

Superposition and sparsity. Superposition occurs when the number of latent features M exceeds d_l ($M > d_l$). For example, LLMs can reference many more place names than they have dimensions in the residual stream. The network is thus forced to represent $a^{(l)}(\mathbf{x}) = \sum_{j=1}^{M} c_j(\mathbf{x})\mathbf{v}_j$ using an overcomplete ($M_l > d_l$) and non-orthogonal set of feature directions $\{\mathbf{v}_j\}_{j=1}^{M}$. This leads to polysemanticity (individual neurons representing multiple features), so the activity of a single neuron no longer clearly indicates the presence or intensity of a unique underlying concept. Such non-orthogonal representations of features $\{\mathbf{v}_j\}$ lead to interference. Networks can employ nonlinear operations, such as ReLU and softmax, to filter these mixed signals and disambiguate superposed features [Gurnee et al., 2023].

Networks may achieve superposition while retaining linear feature vectors by leveraging the near-orthogonality of many vectors in high-dimensional space and the ability to recover sparse vectors from lower-dimensional projections [Elhage et al., 2022]. Effective superposition thus relies on features being sparsely activated: for any input $\mathbf{x}$, most $c_j(\mathbf{x})$ are near zero.

A.2 Propositions

To understand how adversarial attacks exploit feature representations, we first prove that optimal perturbations weight each input dimension by how much its corresponding feature aligns with the path to the decision boundary. We analyse linear models without activations. Consider input $\mathbf{x} \in \mathbb{R}^M$ encoded via $\phi(\mathbf{x}) = \mathbf{V}\mathbf{x}$ to latent representation $\mathbf{h} \in \mathbb{R}^N$, where the columns of $\mathbf{V} \in \mathbb{R}^{N\times M}$ are overcomplete basis vectors $\{\mathbf{v}_j\}_{j=1}^{M}$ ($N < M$). The binary decision boundary is $\mathcal{B} = \{\mathbf{h} : \mathbf{n}^\top\mathbf{h} + b = 0\}$, where $(\mathbf{n}, b)$ define the separating hyperplane. Input perturbations $\Delta\mathbf{x} \in \mathbb{R}^M$ map to latent perturbations $\Delta\mathbf{h} = \mathbf{V}\boldsymbol{\delta}$, where $\boldsymbol{\delta} = (\delta_1, \ldots, \delta_M)^\top$ are the perturbation coefficients.

Proposition 1:. The optimal input perturbations $\boldsymbol{\delta}$ that maximise movement toward the decision boundary under constraint $\|\boldsymbol{\delta}\|_2 = \epsilon$ satisfy $\boldsymbol{\delta} \propto \mathbf{V}^\top\mathbf{n}$, where $\mathbf{n}$ is the normal to the decision boundary.

Proof. To move a sample across the binary boundary most efficiently, we maximise alignment with the normal $\mathbf{n}$:

$$\max_{\boldsymbol{\delta}} \Delta\mathbf{h}^\top\mathbf{n} = \max_{\boldsymbol{\delta}} \boldsymbol{\delta}^\top\mathbf{V}^\top\mathbf{n} \quad \text{s.t.} \quad \|\boldsymbol{\delta}\|_2 = \epsilon$$

Let $\mathbf{g} = \mathbf{V}^\top\mathbf{n}$. By Cauchy-Schwarz:

$$|\boldsymbol{\delta}^\top\mathbf{g}| \leq \|\boldsymbol{\delta}\|_2\|\mathbf{g}\|_2 = \epsilon\|\mathbf{g}\|_2$$

Equality holds when $\boldsymbol{\delta} \propto \mathbf{g} = \mathbf{V}^\top\mathbf{n}$. Specifically, $\boldsymbol{\delta} = \frac{\epsilon}{\|\mathbf{V}^\top\mathbf{n}\|_2}\mathbf{V}^\top\mathbf{n}$, giving $\delta_i \propto \mathbf{v}_i^\top\mathbf{n}$ where $\mathbf{v}_i$ is the i-th column of $\mathbf{V}$. $\square$

We next prove that attacks transfer perfectly between models with the same feature geometry up to orthogonal transformation. For our argmax task, an optimal encoder-decoder pair has the decoder as the transpose of the encoder (which we observe empirically). This yields a decision boundary that is the mid-separating hyperplane between $\mathbf{v}_j$ and $\mathbf{v}_k$ with normal $\mathbf{n} \propto (\mathbf{v}_k - \mathbf{v}_j)$, where j and k are the indices of the two features being classified in the binary task.

Proposition 2.. Consider encoders ϕ and ψ with basis matrices $\mathbf{V} \in \mathbb{R}^{N\times M}$ and $\mathbf{V}' \in \mathbb{R}^{N\times M}$ whose columns are related by orthogonal transformation $\mathbf{v}'_i = \mathbf{Q}\mathbf{v}_i$ (where $\mathbf{Q}^\top\mathbf{Q} = \mathbf{I}$). If both models' decision boundaries separate the same pair of features (with indices j, k), then both models have identical optimal input perturbation vectors.

Proof. The boundary normals are $\mathbf{n}^\phi \propto (\mathbf{v}_k - \mathbf{v}_j)$ and $\mathbf{n}^\psi \propto (\mathbf{v}'_k - \mathbf{v}'_j)$. By Proposition 1:

$$\delta_i^\phi \propto \mathbf{v}_i^\top(\mathbf{v}_k - \mathbf{v}_j) \tag{2}$$

$$\delta_i^\psi \propto (\mathbf{v}'_i)^\top(\mathbf{v}'_k - \mathbf{v}'_j) \tag{3}$$

Substituting $\mathbf{v}'_i = \mathbf{Q}\mathbf{v}_i$ and using $\mathbf{Q}^\top\mathbf{Q} = \mathbf{I}$:

$$\begin{aligned} \delta_i^\psi &\propto (\mathbf{Q}\mathbf{v}_i)^\top(\mathbf{Q}\mathbf{v}_k - \mathbf{Q}\mathbf{v}_j) && (4)\\ &= \mathbf{v}_i^\top\mathbf{Q}^\top\mathbf{Q}(\mathbf{v}_k - \mathbf{v}_j) && (5)\\ &= \mathbf{v}_i^\top(\mathbf{v}_k - \mathbf{v}_j) && (6) \end{aligned}$$

Thus δ_i^ϕ and δ_i^ψ have identical proportionality. Under the same norm constraint, $\boldsymbol{\delta}^\phi = \boldsymbol{\delta}^\psi$. □

A.3 Toy Model Experiments

This section provides supplementary details and extended results for the toy model experiments discussed in the main paper. We present model accuracies across a wider range of parameters than shown in the main text, offering insight into how model capacity and data characteristics like sparsity influence the learning process and the conditions under which feature superposition appears. Subsequently, we offer additional visual examples that correspond to Figure 1 in the main paper, illustrating the mechanics of adversarial attacks under various conditions.

A.3.1 Accuracy of Toy Model for a Range of Parameters

The toy model experiments presented in the main paper predominantly used low-dimensionality settings for conceptual clarity. To demonstrate the model's behaviour more broadly, this subsection details the classification accuracies achieved by the CE toy model. These results are presented across varying hidden layer size (h), number of classes (k), number of features, and levels of sparsity (S), to provide insight into when the models learn to represent features in superposition. The sparsity level represents the probability that any individual input feature $x_i^{(j)}$ is set to zero, with higher values of S indicating greater input sparsity. These tables Tab. 1 and Tab. 2 provide provide context on the model's performance limits and its ability to learn latent representations in superposition.

A.3.2 Additional Examples

Section 3 of the main paper (illustrated by Figure 1) demonstrates how adversarial attacks exploit the interference between latent features in superposition. This subsection provides further visual examples (Fig. 4) to reinforce intuition from the findings from the toy model. Specifically, we supplement the main text by showcasing:

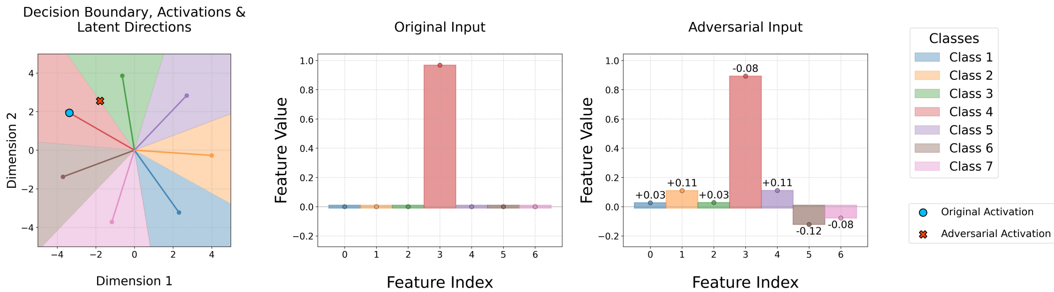
- An additional instance of the basic experimental setup ($m = 2$, 7 classes) with an ℓ_2-norm PGD attack, demonstrating the characteristic input perturbation profiles and latent space manipulations that lead to misclassification.
- An example ($m = 2$, 7 classes) of perturbations generated using an ℓ_∞-norm PGD attack.
- An example with increased bottleneck dimensionality ($m = 3$, 7 classes) using an ℓ_2-norm PGD attack. The feature vector similarity matrix is also shown to provide context on the learned latent representations for the classes.

Table 1: Classification accuracy of the CE toy model with a fixed bottleneck dimension ($m = 2$) across various numbers of classes (k), total input features (features = $k \times 3$), and input feature sparsity levels ($1 - S$). These results illustrate how performance degrades as the number of classes to be superposed within a highly constrained latent space increases, and how input sparsity can mitigate this.

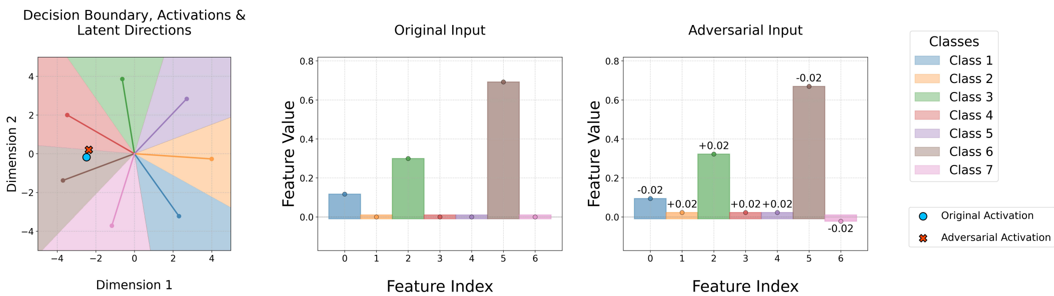
Classes (k)	Features	Hidden (m)	Accuracy at Sparsity Level ($1 - S$)							
			1.0	0.57	0.33	0.19	0.11	0.06	0.04	0.02
3	9	2	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
4	12	2	0.67	0.71	0.82	0.92	0.98	0.99	1.00	1.00
5	15	2	0.53	0.50	0.65	0.77	0.89	0.95	0.98	0.99
6	18	2	0.40	0.43	0.51	0.66	0.82	0.93	0.97	0.99
7	21	2	0.34	0.34	0.40	0.53	0.73	0.87	0.95	0.98
8	24	2	0.30	0.30	0.33	0.40	0.63	0.82	0.93	0.97
9	27	2	0.24	0.26	0.30	0.35	0.57	0.75	0.89	0.96
10	30	2	0.22	0.24	0.26	0.31	0.50	0.72	0.87	0.95
15	45	2	0.14	0.15	0.16	0.17	0.25	0.40	0.65	0.86
20	60	2	0.10	0.10	0.11	0.13	0.15	0.24	0.44	0.76
25	75	2	0.07	0.08	0.09	0.10	0.12	0.16	0.26	0.62
30	90	2	0.06	0.07	0.07	0.07	0.09	0.11	0.21	0.45

Table 2: Classification accuracy of the CE toy model for varying numbers of classes (k), total input features, bottleneck dimensions (m), and input feature sparsity levels ($1 - S$). This table explores scenarios.

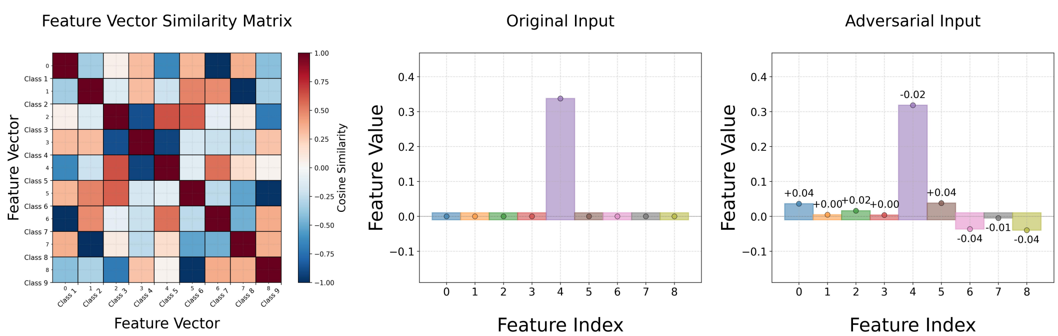
Classes (k)	Features	Hidden (m)	Accuracy at Sparsity Level ($1 - S$)							
			1.0	0.57	0.33	0.19	0.11	0.06	0.04	0.02
30	90	90	0.27	0.27	0.33	0.41	0.51	0.67	0.85	0.94
30	30	30	0.23	0.24	0.38	0.62	0.83	0.94	0.99	1.00
40	40	30	0.67	0.54	0.73	0.77	0.88	0.96	0.99	0.99
40	120	30	0.71	0.64	0.65	0.72	0.73	0.79	0.89	0.96
60	60	10	0.05	0.07	0.12	0.25	0.47	0.73	0.90	0.97
60	180	10	0.08	0.10	0.13	0.17	0.22	0.32	0.52	0.75
80	80	30	0.15	0.17	0.23	0.41	0.63	0.79	0.91	0.98
80	240	30	0.23	0.22	0.31	0.41	0.48	0.53	0.66	0.80
100	100	10	0.03	0.04	0.05	0.10	0.21	0.43	0.69	0.87
100	300	10	0.04	0.05	0.06	0.09	0.12	0.15	0.25	0.45



(a) An ℓ_2-norm attack changing the classification of an input (original class 4) to class 3. The left plot shows original and adversarial activations in latent space relative to class latent directions. The right plots show original and perturbed input feature values, respectively.



(b) An ℓ_∞-norm attack changing the classification of an input (original class 6) to class 4. The left plot shows original and adversarial activations in latent space relative to class latent directions. The right plots show original and perturbed input feature values, respectively.



(c) An ℓ_2-norm adversarial attack in a 7-class setup but with an increased bottleneck dimension $m = 3$. The leftmost plot now shows the cosine similarity matrix between the learned latent directions for each of the classes.

Figure 4: Visualisations of AExs in the toy model, supplementing Figure 1 from the main paper by illustrating attack mechanisms in activation space and input space under varied conditions.

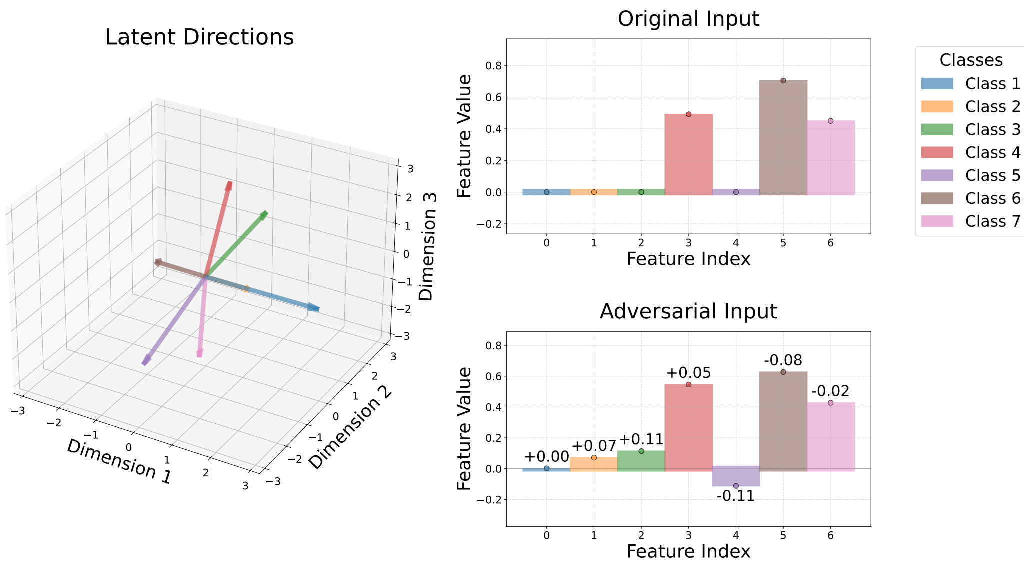


Figure 5: An adversarial attack (from class 5 to class 3) does not perturb the input features for a class represented orthogonally.

A.4 CIFAR-10 Experiments

To investigate whether the principles observed in the toy models extend to more complex settings, Section 4 of the main paper introduces experiments using a ViT [Dosovitskiy et al., 2020] trained on CIFAR-10 [Krizhevsky, 2009] with an engineered bottleneck. This appendix section provides further details on this setup and presents extended results.

A.4.1 Architecture & Training Information

The base ViT architecture comprised 6 transformer layers with an embedding dimension (d) of 512. Input images from the CIFAR-10 dataset, sized at 32×32 pixels, were processed into patches of 4×4 pixels. Each transformer layer utilised 8 attention heads. The Multilayer Perceptron (MLP) within each transformer block had a hidden dimension of 512. Learned positional embeddings were used.

The bottleneck architecture consisted of a linear encoder followed by a linear decoder. The linear encoder projected the pre-classification activations obtained from the ViT backbone, which had a dimensionality of 512, into an m-dimensional latent space. The subsequent linear decoder then mapped these m-dimensional representations back to the $k = 10$ dimensions corresponding to the CIFAR-10 classes. No ReLU activation functions were applied within either the encoder or the decoder layers of this bottleneck. The dimensionality m of this bottleneck was systematically varied across different experimental runs, taking values from the set $\{2, 3, 5, 10\}$, as described in the main paper.

The base ViT model was trained on the CIFAR-10 dataset for 250 epochs. A learning rate of 0.001 was used with the Adam optimiser using default PyTorch parameters. The batch size was set to 512 and a cosine annealing learning rate scheduler. The loss function was CE. Dropout was used. Training was performed across five different random seeds to account for variability. After the base ViT model was trained, its weights were frozen. The bottleneck layer was then trained for 30 epochs, utilising a learning rate of 0.001.

A.4.2 Normalised Robust Accuracy across Perturbation Magnitudes

Figure 5 (right) in the main paper shows how normalised robust accuracy varies with bottleneck dimension for a fixed perturbation. This subsection expands on those findings by detailing the normalised robust accuracy of the ViT models (with varying bottleneck dimensions $m \in 2, 3, 5, 10$) when subjected to PGD attacks of different strengths (ϵ). Results are presented for both ℓ_2-norm

(Tab. 4 and ℓ_∞-norm (Tab. 3) PGD attacks, providing a more comprehensive view of how the degree of superposition interacts with attack strength to affect model robustness.

AExs for these evaluations were generated using PGD with 100 iterations. A step size (α) of 0.01 was employed. Robust accuracy was evaluated on 500 samples for each configuration. The mean normalised robust accuracy and standard deviation across five random seeds are reported.

Tab. 4 presents the normalised robust accuracy for ℓ_2 attacks across a range of perturbation magnitudes (ϵ). Similarly, Table 3 shows the normalised robust accuracy for ℓ_∞ attacks for various ϵ values.

Table 3: Mean normalised robust accuracy ($\pm$ standard deviation across 4 seeds) for ViT models with different bottleneck dimensions (m) on CIFAR-10, subjected to ℓ_∞-norm PGD attacks of varying perturbation magnitudes (ϵ). Robust accuracy is normalised by the clean accuracy of each bottlenecked model. Similar to Tab. 4, these results complement Section 4 of the main paper, demonstrating the impact of superposition pressure and attack strength on robustness.

ϵ	Bottleneck Dimension (m)			
	2	3	5	10
0.001	96.0% ± 0.4%	97.4% ± 0.7%	97.9% ± 0.4%	98.1% ± 0.3%
0.01	61.7% ± 3.6%	69.6% ± 3.5%	77.0% ± 0.9%	81.8% ± 3.2%
0.05	4.9% ± 1.5%	6.7% ± 1.6%	9.3% ± 0.6%	10.5% ± 0.8%
0.1	0.1% ± 0.2%	0.2% ± 0.3%	0.2% ± 0.4%	0.2% ± 0.4%
0.5	0.0% ± 0.0%	0.0% ± 0.0%	0.0% ± 0.0%	0.0% ± 0.0%

Table 4: Mean normalised robust accuracy ($\pm$ standard deviation across 4 seeds) for ViT models with different bottleneck dimensions (m) on CIFAR-10, subjected to ℓ_2-norm PGD attacks of varying perturbation magnitudes (ϵ). Robust accuracy is normalised by the clean accuracy of each bottlenecked model. These results support the findings in Section 4 of the main paper, showing decreasing robustness with smaller m (increased superposition) and larger ϵ.

ϵ	Bottleneck Dimension (m)			
	2	3	5	10
0.1	90.4% ± 1.7%	91.8% ± 0.7%	94.6% ± 1.2%	95.0% ± 0.7%
0.5	58.5% ± 5.0%	60.8% ± 5.0%	69.2% ± 1.8%	72.6% ± 2.8%
1.0	41.7% ± 4.8%	44.0% ± 3.0%	50.4% ± 0.6%	54.9% ± 1.7%
2.0	34.0% ± 4.6%	36.2% ± 3.3%	41.5% ± 1.3%	47.4% ± 2.4%
5.0	30.2% ± 4.1%	32.9% ± 3.5%	37.3% ± 1.9%	42.7% ± 1.0%

A.4.3 Attack Transferability across Perturbation Magnitudes

We include attack transferability across various perturbation magnitudes (ϵ). Table 6 presents the ℓ_2-norm attack transferability, and Table 5 shows the ℓ_∞-norm attack transferability, both across different ϵ values and bottleneck dimensions (m).

Table 5: Attack transferability (%) for ℓ_∞-norm PGD attacks on CIFAR-10 ViT models. Transferability is shown from a model trained with a specific 'Source Seed' (e.g., Seed 10) to three different target models, each trained with one of the seeds listed in the sub-header (e.g., 'vs. Seeds 20/30/40'). The three slash-separated values in each data cell correspond to the transferability to these three target seeds, respectively. All models within a row share the same bottleneck dimension, m. The 'Mean $\pm$ Std' column averages transferability across all 12 source-target seed pairings for each (ϵ, m) configuration. This supports the claim in Section 4 that higher superposition (smaller m) can lead to more consistent latent geometries and thus higher transferability.

ϵ	m	Seed 10 vs. Seeds 20/30/40	Seed 20 vs. Seeds 10/30/40	Seed 30 vs. Seeds 10/20/40	Seed 40 vs. Seeds 10/20/30	Mean $\pm$ Std
0.001	2	77.8/66.7/44.4	25.0/62.5/25.0	72.7/72.7/72.7	58.3/75.0/58.3	59.3 $\pm$ 17.7
	3	57.1/28.6/28.6	14.3/28.6/28.6	63.6/72.7/72.7	42.9/42.9/42.9	43.6 $\pm$ 18.4
	5	70.0/50.0/50.0	57.1/71.4/57.1	50.0/25.0/25.0	16.7/50.0/33.3	46.3 $\pm$ 16.9
	10	55.6/55.6/66.7	12.5/37.5/25.0	16.7/33.3/16.7	28.6/71.4/28.6	37.3 $\pm$ 19.3
0.01	2	60.3/52.6/48.7	63.8/42.0/60.9	53.8/40.9/52.7	51.2/55.8/46.5	52.4 $\pm$ 6.9
	3	42.9/46.4/50.0	46.4/42.9/49.1	44.2/38.9/44.2	42.9/45.1/41.8	44.6 $\pm$ 3.0
	5	43.3/41.1/43.3	39.2/30.4/41.8	46.0/40.2/47.1	41.2/35.3/32.9	40.2 $\pm$ 4.8
	10	42.5/46.0/44.8	38.0/39.4/46.5	43.5/47.8/47.8	32.6/40.0/35.8	42.1 $\pm$ 4.7
0.05	2	35.0/36.4/37.8	38.8/36.0/46.3	37.4/31.3/41.7	42.0/42.0/40.7	38.8 $\pm$ 3.8
	3	29.9/29.9/37.1	30.4/30.1/35.6	30.2/30.2/35.1	30.1/31.9/34.4	32.1 $\pm$ 2.6
	5	26.8/25.3/29.2	25.2/21.5/26.4	28.0/28.0/26.5	26.0/26.6/21.9	25.9 $\pm$ 2.2
	10	27.1/24.9/30.4	21.6/23.0/28.4	25.3/28.1/30.6	21.0/24.9/20.1	25.4 $\pm$ 3.4
0.1	2	34.2/35.6/38.2	36.7/34.9/44.5	37.9/30.0/40.8	41.2/42.0/40.3	38.0 $\pm$ 3.8
	3	29.6/28.9/35.2	28.6/29.2/34.8	29.1/28.2/34.0	28.8/31.1/32.8	30.9 $\pm$ 2.5
	5	25.4/23.3/27.0	23.8/19.6/24.6	25.9/26.7/25.4	24.6/24.3/20.7	24.3 $\pm$ 2.1
	10	24.8/22.6/29.7	20.1/21.6/26.1	22.5/25.5/28.0	19.5/23.3/18.7	23.5 $\pm$ 3.3

Table 6: Attack transferability (%) for ell_2-norm PGD attacks on CIFAR-10 ViT models. The table format, detailing source-to-target seed transferability (including slash-separated values and the 'Mean ± Std' calculation), mirrors that of Table 5; please see its caption for a full explanation. These ℓ_2 results further support the claim in Section 4 of the main paper that higher superposition (smaller m) leads to increased attack transferability.

ϵ	m	Seed 10 vs. Seeds 20/30/40	Seed 20 vs. Seeds 10/30/40	Seed 30 vs. Seeds 10/20/40	Seed 40 vs. Seeds 10/20/30	Mean $\pm$ Std
0.1	2	70.6/64.7/41.2	60.0/50.0/60.0	64.3/60.7/57.1	48.1/63.0/51.9	57.6 $\pm$ 8.0
	3	39.1/39.1/47.8	50.0/60.7/50.0	48.1/55.6/51.9	48.0/28.0/40.0	46.5 $\pm$ 8.4
	5	43.5/34.8/43.5	31.6/31.6/42.1	42.9/35.7/21.4	30.4/39.1/34.8	35.9 $\pm$ 6.4
	10	45.5/59.1/45.5	19.0/28.6/28.6	37.5/43.8/43.8	23.8/52.4/38.1	38.8 $\pm$ 11.4
0.5	2	59.6/52.1/48.9	60.8/35.4/60.8	54.1/39.4/51.4	43.8/49.5/46.7	50.2 $\pm$ 7.7
	3	38.7/40.6/48.1	36.1/36.8/41.7	37.4/36.5/43.5	34.4/34.4/36.7	38.7 $\pm$ 3.9
	5	38.4/32.0/37.6	29.4/25.7/32.1	39.1/33.9/38.3	31.8/29.9/32.7	33.4 $\pm$ 4.0
	10	38.3/37.4/37.4	26.5/28.4/31.4	32.4/35.3/35.3	23.4/28.2/25.0	31.6 $\pm$ 5.0
1.0	2	50.8/43.7/44.4	51.7/36.4/50.0	46.1/32.9/44.7	42.7/46.7/45.3	44.6 $\pm$ 5.3
	3	35.7/36.9/41.7	31.8/36.4/42.6	33.7/32.0/37.3	34.9/32.0/35.5	35.9 $\pm$ 3.3
	5	32.1/28.9/33.2	30.9/23.0/30.9	35.9/31.5/33.7	27.7/28.2/27.1	30.3 $\pm$ 3.3
	10	32.3/32.3/38.5	23.7/27.7/29.9	29.1/32.0/34.3	22.2/25.6/21.1	29.1 $\pm$ 5.0
2.0	2	47.0/41.6/44.3	48.2/36.2/48.2	43.9/33.5/42.1	43.4/43.4/42.8	42.9 $\pm$ 4.2
	3	31.9/37.7/40.3	31.1/35.6/40.6	32.6/31.1/36.8	35.4/33.3/34.3	35.1 $\pm$ 3.2
	5	29.9/25.4/32.1	26.3/22.1/26.7	32.3/31.8/29.5	27.9/27.9/23.6	28.0 $\pm$ 3.2
	10	30.8/30.4/35.7	22.5/24.9/28.6	24.8/29.7/32.7	20.4/26.1/19.0	27.1 $\pm$ 4.9
5.0	2	44.7/46.7/40.8	45.3/37.3/48.7	42.7/33.9/41.5	42.3/42.3/40.6	42.2 $\pm$ 3.8
	3	32.3/35.8/40.8	31.3/32.6/39.9	31.0/30.0/33.8	35.0/33.0/33.0	34.1 $\pm$ 3.2
	5	30.6/24.1/28.6	28.3/20.9/24.8	31.7/31.3/27.4	28.1/29.0/23.2	27.3 $\pm$ 3.3
	10	29.8/27.4/32.7	19.7/24.1/28.9	24.9/29.9/31.2	18.8/25.8/19.2	26.0 $\pm$ 4.6