Input Space Mode Connectivity in Deep Neural Networks

Jakub Vrabel [*†]
CEITEC, Brno University of Technology

Ori Shem-Ur and Yaron Oz
Tel Aviv University

David Krueger [‡]
Mila

Abstract

We extend the concept of loss landscape mode connectivity to the input space of deep neural networks. Initially studied in parameter space, mode connectivity describes the existence of low-loss paths between solutions (loss minimizers) found via gradient descent. We present theoretical and empirical evidence of its presence in the input space of deep networks, thereby highlighting the broader nature of the phenomenon. We observe that different input images with similar predictions are generally connected, and for trained models, the path tends to be simple, with only a small deviation from being a linear path. We conjecture that input space mode connectivity in high-dimensional spaces is a geometric phenomenon, present even in untrained models, and can be explained by percolation theory. We exploit mode connectivity to obtain new insights about adversarial examples and show its potential for adversarial detection and interpretability.

1 Introduction

The high-dimensional nature of the parameter space in deep neural networks (DNNs) gives rise to the phenomenon of loss landscape mode connectivity, where different solutions achieved via stochastic gradient descent (SGD) are connected by simple low-loss paths [1, 2]. Connections between the modes can be as simple as linear in specific cases [3], while they are often quadratic in the majority of cases. Mode connectivity was utilized for enhanced ensembling strategies [1], which can boost generalization capabilities of models [4]. Known causes of mode connectivity include architectural symmetries, like node permutations [5], and common loss functions, though the full mechanism remains elusive. Our work further explores the hypothesis that mode connectivity is a more general phenomenon of high dimensional geometry, as partially studied in Fort and Jastrzebski [6].

We demonstrate that mode connectivity can be extended to the input space of neural networks. Unlike the originally studied parameter space, where data are fixed and model parameters vary, we use an inverted setup (fixed model, variable data) with real, interpolated, and synthetic (optimized [7]) inputs. Considering cross-entropy loss, we prepare several examples of *optimal inputs* (natural images, high-freq. patterns, adversarial examples) and show that they are mode connected in vision classification models. This connectivity offers insights into adversarial detection and DNN interpretability. We conjecture that input space mode connectivity can be explained by high-dimensional percolation, a purely geometric phenomenon. Additionally, trained networks exhibit approximate linear mode connectivity for many real examples, which may reflect the implicit regularization in deep learning.

Contributions: 1) We extend the concept of mode connectivity from the original parameter space to the input space of (vision) DNNs. **2)** We propose a novel method for adversarial detection that outperforms baselines for advanced attacks (deepfool [8] and C&W [9]). **3)** We conjecture that

[*] Correspondence to: `jakub.vrabel@ceitec.vutbr.cz`
[†] Work partially performed during a research visit at the University of Cambridge.
[‡] Work completed while at the University of Cambridge.

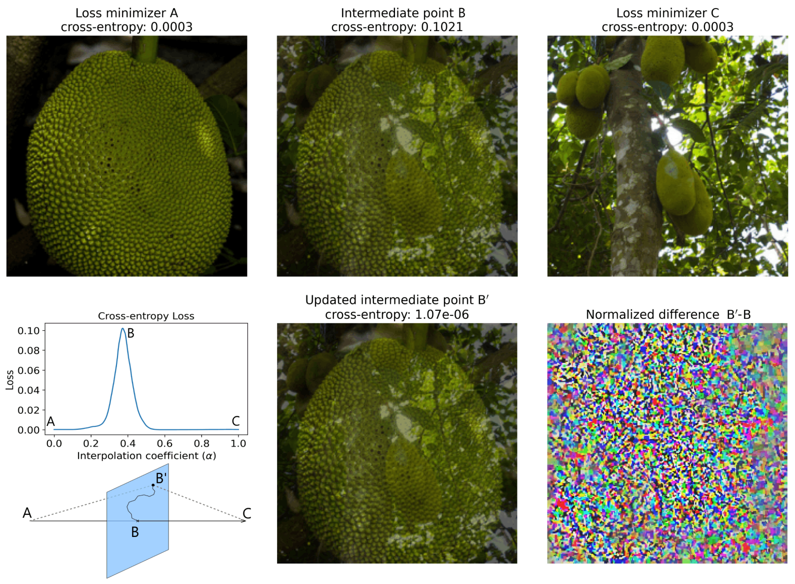

Figure 1: Barrier between two input modes. Two examples (A and C) that minimize loss for the class *jackfruit* are shown. The interpolated point B exhibits maximal loss. **Left bottom** Cross-entropy loss for interpolated points between A and C; diagram for the interpolation paths and subsequent constrained optimization of B. **Middle bottom** The optimized point B' minimizes the loss and is nearly indistinguishable from B. **Right bottom:** Normalized difference pattern B'$-$B (max pixel value of the pattern is 1–5% of the max pixel in the original image).

arbitrary inputs with the same prediction are mode connected in randomly initialized neural networks and provide a proof sketch based on percolation theory.

2 Related work

Prior work [10] studied excessive input invariance in neural networks, which inspired [11] to explore a specific form of input space connectivity between real inputs and invariance-based adversarial examples (or "blind spots"). In contrast, we study connectivity between nearly any pair of inputs with similar model outputs. Additionally, we offer empirical and theoretical evidence that input space mode connectivity occurs more generally, even in untrained networks.

In neuroscience, distinct visual stimuli (inputs) that produce identical responses are referred to as *metamers*. These have been studied to understand how deep learning models diverge from human perceptual systems [12], although connectivity has not been discussed..

Our work is inspired by seminal research on parameter space mode connectivity [1, 2], using established methods to explore input space connectivity. We adapted a simplified algorithm to bypass loss barriers from [6]. Simsek et al. [13] demonstrated that adding a single neuron per layer connects originally discrete minima, emphasizing the role of space dimensionality.

3 Methodology

We define an input space mode for a class y_i as a single data example that minimizes the loss $\mathcal{L}$

$$x_i = \underset{x}{\operatorname{argmin}}\, \mathcal{L}(f(x_i;\theta), y_i), \tag{1}$$

where $f(x_i;\theta)$ is the output of the neural network for the input x_i. Input modes could be either real or synthetic (see App. A) data examples with loss below a certain small threshold δ. After observing two (or more) modes x_A and x_C, we linearly interpolate them to create virtual points $x_{\mathrm{A}\to\mathrm{C}}(\alpha)$ as

$$x(\alpha) = \alpha x_i + (1-\alpha)x_j, \tag{2}$$

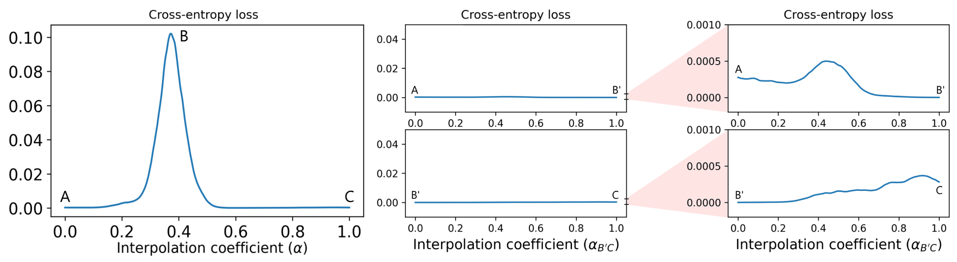

Figure 2: Bypassing the barrier. After optimizing the intermediate point B, we obtained the updated point B', which linearly connects to both A and C. Note that small *secondary* barriers, below the threshold of $\delta = 0.001$ loss, are disregarded.

where $\alpha \in [0, 1]$ is uniformly sampled. In general, a loss barrier exists between the two modes (Fig. 1 left bottom). The barrier can be bypassed via optimization of the highest-loss input x_B, which results in modes connected by a partially linear path of low loss. The barrier optimization is based on a systematic manipulation of pixels in order to minimize the loss (see App. A for details). In the following sections, we use a simplified notation $x_{\mathrm{A}} \equiv \mathrm{A}$ for points/images, and A→C denotes a path between A and C.

4 Results

In this section, we provide empirical and theoretical evidence for the input space mode connectivity in vision models and discuss potential applications of the phenomenon.

Input space connectivity: To demonstrate this phenomenon in a realistic and practically relevant setting, we used GoogLeNet (Inception v1) [14] that was pretrained on the ImageNet [15]. Two low loss examples A and C were linearly interpolated with 1,000 uniformly distributed points. After identifying the interpolated point of maximum loss B, we started a constrained optimization process, encompassing 1024 iterations, which resulted in a good loss minimizer B' (see Fig. 1) (optimization details in App. B). A new path (A→B'→C) was created by merging the linear interpolations between A and B', and between B' and C. We sampled 500 intermediate points equally from both segments of the new path, disregarding the fact that the barrier typically does not lie exactly at the midpoint of A→C. The A→B'→C path exhibits low loss while connecting two input modes (Fig. 2). Furthermore, images along the two studied paths closely resemble each other in corresponding pairs (see App. B.1). Although small *secondary* barriers are still present in the A→B'→C path, we neglect them as they are below a threshold ($\delta = 0.001$ considered here). By repeating the whole optimization procedure, we can easily bypass secondary barriers. More examples and qualitatively consistent results from ResNet18 [16] trained on CIFAR-10 [17] are shown in the Appendix F.

Adversarial examples: We find that adversarial examples, even when significantly different in terms of human visual perception, are mode-connected to "true" real images (correct class minimizers). The details about employed adversarial examples are in Appendix B.2. Using ImageNet and GoogLeNet, we picked a low loss example from a selected class (here *golf ball*) as the mode A. Then, a source image K from a different class (*revolver*) was optimized to obtain an adversarial example K' (K' minimizes the same class as the A). From this point, we followed the mode connecting procedure described above. A single instance of the iterative procedure (i.e., optimization of a single barrier) is not enough to connect modes A and K' as secondary barriers emerge on the new path A→B'→K. By repetitive utilization of the procedure we can get rid of all barriers. The whole process is depicted in Fig. 3 and experimental details in App. B.2.1.

In addition to the higher complexity of the paths, we observed statistically significant differences in the barrier heights and shapes between two key scenarios: barriers between real–adversarial modes and real–real modes (see App. B.2.2 and Fig. 5). We exploit such behavior for adversarial detection.

Adversarial detection: Improving adversarial robustness is crucial, especially in safety-critical areas [18]. Many defenses, like adversarial training, fail against advanced and adaptive attacks [19]. Rather than preserving model accuracy, an alternative is detecting attacks and rejecting predictions for compromised inputs [20, 21]. We proposed a simple a detection algorithm that leverages the

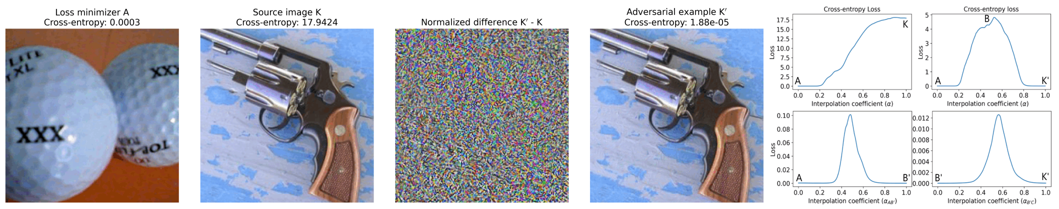

Figure 3: Adversarial attack. From the left: **i)** real input A that minimizes the loss for class *golf ball*, **ii)** source input K from a different class, **iii)** optimized pattern added to the K, **iv)** adversarial example K' that minimizes the loss for the same class as A. The network predicts K' as a golf ball, and **v)** interpolated paths between reference and attack. The primary barrier was bypassed through the optimized point B'. Small secondary barriers emerged on the new path A→B'→K' and can be further bypassed by a repeated application of the procedure to both segments of the new path.

relative lack of linear connectivity in natural–adversarial image pairs (Fig. 5). We simply compute loss curves to class-specific preselected images and train a classifier to distinguish between them (detailed description in App. B.3). We compared our approach against the benchmark [21], where we outperformed baselines on advanced attacks (C&W [9] by 6.9/3.9% on CIFAR10/100, and deepfool [8] by 5.3% on CIFAR10), but underperformed on simpler attacks (FGSM [22], BIM [23], and PGD [24]). Comprehensive results and full discussion is presented in App. B.4.1.

Theoretical work: Here, we summarize the main theoretical result, beginning with the definitions. We denote our input and output spaces as $X = \mathbb{R}^{d_X}$ and $Y = B^{d_y} = \left\{ y \in \mathbb{R}^{d_y} \mid \|y\| \leq 1 \right\} \subseteq \mathbb{R}^{d_y}$, respectively. A path connecting $x(0) \in X$ to $x(1) \in X$ is defined as a continuous function $x : [0,1] \to X$. We define a network that maps the input space to the output space as any function within the framework of the tensor program formalism [25–27], featuring Lipschitz continuity-bounded point-wise nonlinearities. Specifically, for a parameter vector θ, such a network accepts inputs and returns outputs $f(\cdot;\theta) : X \to Y$. Given any $\delta > 0$, network f, parameter vector θ, and loss function $\mathcal{L} : Y \times Y \to \mathbb{R}^+$, we say that a path $x(\cdot)$ is **δ-connected around** $y \in Y$ for this loss function if and only if $\forall \alpha \in [0,1] : \mathcal{L}(f(x(\alpha);\theta), y) \leq \delta$, with arbitrarily large probability. Note that our paths are conceptually similar to "representational geodesics" [28], although we don't require the length of the path to be minimal.

We present the concept of **geometric mode connectivity**, suggesting that almost all inputs on which a neural network makes similar predictions tend to be connected, as d_X grows to infinity. We find this is the case empirically for both trained networks and untrained, randomly initialized networks. We believe the latter finding can be proven as a consequence of high-dimensional geometry, and formalize it in the following conjecture:

Conjecture 4.1 (Geometric Input Space Connectivity)**.**

Given a subset of the input space $X' \subseteq X$, a network $f(\cdot;\theta)$ at initialization, and a loss function $\mathcal{L}$, as described in appendix D.1, the following holds: For any probability $0 < p < 1$ arbitrarily close to 1, and any arbitrarily small $0 < \delta' < \delta$ (assuming for simplicity that $\frac{1}{\delta} \in \mathbb{N}$), any two random inputs $x_0, x_1 \in X'$ with similar predictions are almost always connected as $d_X \to \infty$:

$$P\left(x_0, x_1 \text{ are } \delta\text{-connected} \mid \mathcal{L}\left(f\left(x_0;\theta\right), f\left(x_1;\theta\right)\right) \leq \delta'\right) = 1 - O\left(e^{-d_X \tilde{\delta}}\right), \tag{3}$$

where $\tilde{\delta} = O(\delta)$, and x_0, x_1, θ are chosen randomly as described in appendix D.1.

We justify the conjecture for $\mathcal{L}(y, y') = \|y - y'\|$, and the case where the network's outputs are numbers between zero and one, $f(\cdot;\theta) : X \to [0,1]$. The generalization to more general cases is straightforward.

We include a detailed proof sketch for this conjecture in appendix D, which we believe provides significant insight into the phenomenon of mode connectivity but note that it is not a fully complete and rigorous proof:

(i) First, in appendix D.1.4, we demonstrate that the network is Lipschitz continuous, by leveraging the Lipschitz continuity of the network's activation functions, and its semi-linear structure.

Using this property, we show that for every $0 < \delta$, there exists $0 < \varepsilon$, such that if we divide the input space into ε-sized hypercubes, then for every cube inside the lattice, all the points' outputs ($f\left(x;\theta\right)$) belong to one of the following overlapping intervals:

$$Y_\delta = \left\{[0,\delta], \left[\frac{\delta}{2}, \frac{3\delta}{2}\right], [\delta, 2\delta], \ldots, [1-\delta, 1]\right\}. \tag{4}$$

We can then label each input by the interval (or intervals), that corresponds to the output of all the inputs in the cube.

We show that if there exists a sufficiently small ε, then for any two points $x_0, x_1 \in X'$ that satisfy the conjecture's condition ($\mathcal{L}\left(f\left(x_0;\theta\right), f\left(x_1;\theta\right)\right) \leq \delta'$), their respective cubes share the same interval. This ε is chosen as the length of our graph's hypercubes.

(ii) Second, we demonstrate in appendix D.1.5 that to find a δ-connected path between the two inputs, it suffices to find a connected path of cubes between the cube of the first input and the cube of the second input, with the same interval-label.

Assuming the cube's intervals are drawn randomly and independently from each other with similar probability $p \approx \delta$, we encounter a **classic high-dimensional percolation problem** [29–33], where the cubes represent the points in the graph, and they are connected if they share the same interval. It is well known that in such a case, the probability that two points are connected grows rapidly with the dimension, as shown in the conjecture 4.1.

It is important to note that the assumption of independent probability is not realistic, as there are strong correlations among nearby inputs of wide neural networks, even at initialization. This is why we decided not to consider statement 4.1 as a theorem, but rather as a conjecture. However, we expect that in general, assuming independence only underestimates the true connectivity in the system, as the correlation between outputs of nearby inputs in neural networks at initialization tends to be positive.

We provide empirical evidence for the connectivity in untrained models in App. B.5. For trained networks, we observed that inputs not only tend to be connected but are also connected approximately linearly, with only a relatively small barrier. We hypothesize that this can be manifestation of the *implicit regularization* of neural networks (discussed in App. E).

5 Conclusion and future work

We demonstrated mode connectivity in the input space of deep networks trained for image classification, extending the original concept beyond parameter space. This phenomenon was shown with both real and synthetic images in various controlled setups. For tested examples, we were always able to find low-loss paths between any two modes. We explored potential applications of this connectivity, particularly for the adversarial detection and interpretability of DNNs (Appendix B.6). Our findings reveal that natural inputs can be distinguished from adversarial attacks by the height of the loss barrier on the linear interpolant path between modes. While for real–real mode pairs, the loss barrier is small or even negligible, real–adversarial pairs often have high and complex barriers. By exploring connections between different types of class-optimal inputs we obtained a novel perspective on DNN interpretability. The new evidence for mode connectivity beyond the parameter space supports the hypothesis that mode connectivity is an intrinsic property of high-dimensional geometry and can be studied through percolation theory. This hypothesis, however, requires additional investigation to fully understand its implications.

Future work can focus on formalizing the theory for our conjecture of geometric mode connectivity and explain minimal requirements for its manifestation. An additional direction involves developing a proper theory and conducting further experiments to better understand the implicit regularization phenomenon in the input space. Regarding the potential applications, creating an efficient implementation of the proposed naïve algorithm for adversarial example detection is an important next step. Leveraging insights from the input space connectivity might also inform more effective adversarial *training*, e.g., through enhanced sampling of adversarial examples along low-loss paths for model retraining – a concept analogous to utilizing parameter space connectivity for improving deep ensembles [4]. Additionally, investigating the geometric properties of the (input space) loss landscape around significant points and exploring the entire class-optimal manifold could provide deeper insights into model interpretability.

References

[1] Timur Garipov et al. “Loss Surfaces, Mode Connectivity, and Fast Ensembling of DNNs”. In: *Advances in Neural Information Processing Systems*. Ed. by S. Bengio et al. Vol. 31. Curran Associates, Inc., 2018.

[2] Felix Draxler et al. “Essentially No Barriers in Neural Network Energy Landscape”. In: *Proceedings of the 35th International Conference on Machine Learning*. Ed. by Jennifer Dy and Andreas Krause. Vol. 80. Proceedings of Machine Learning Research. PMLR, Oct. 2018, pp. 1309–1318.

[3] Vaishnavh Nagarajan and J. Zico Kolter. “Uniform convergence may be unable to explain generalization in deep learning”. In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach et al. Vol. 32. Curran Associates, Inc., 2019.

[4] Stanislav Fort, Huiyi Hu, and Balaji Lakshminarayanan. *Deep Ensembles: A Loss Landscape Perspective*. 2020. arXiv: `1912.02757 [stat.ML]`.

[5] Bo Zhao et al. “Understanding Mode Connectivity via Parameter Space Symmetry”. In: *UniReps: the First Workshop on Unifying Representations in Neural Models*. 2023.

[6] Stanislav Fort and Stanislaw Jastrzebski. “Large Scale Structure of Neural Network Loss Landscapes”. In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach et al. Vol. 32. Curran Associates, Inc., 2019.

[7] Chris Olah, Alexander Mordvintsev, and Ludwig Schubert. “Feature Visualization”. In: *Distill* (2017). DOI: `10.23915/distill.00007`.

[8] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. “DeepFool: A Simple and Accurate Method to Fool Deep Neural Networks”. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 2574–2582. DOI: `10.1109/CVPR.2016.282`.

[9] Nicholas Carlini and David Wagner. “Towards Evaluating the Robustness of Neural Networks”. In: *2017 IEEE Symposium on Security and Privacy (SP)*. 2017, pp. 39–57. DOI: `10.1109/SP.2017.49`.

[10] Joern-Henrik Jacobsen et al. “Excessive Invariance Causes Adversarial Vulnerability”. In: *International Conference on Learning Representations*. 2019.

[11] Sriram Balasubramanian et al. “Exploring Geometry of Blind Spots in Vision models”. In: *Thirty-seventh Conference on Neural Information Processing Systems*. 2023.

[12] Jenelle Feather et al. “Model metamers reveal divergent invariances between biological and artificial neural networks”. In: *Nature Neuroscience* 26.11 (Nov. 2023), pp. 2017–2034. ISSN: 1546-1726. DOI: `10.1038/s41593-023-01442-0`.

[13] Berfin Simsek et al. “Geometry of the Loss Landscape in Overparameterized Neural Networks: Symmetries and Invariances”. In: *Proceedings of the 38th International Conference on Machine Learning*. Ed. by Marina Meila and Tong Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, 18–24 Jul 2021, pp. 9722–9732.

[14] Christian Szegedy et al. *Going Deeper with Convolutions*. 2014. arXiv: `1409.4842 [cs.CV]`.

[15] Olga Russakovsky et al. “ImageNet Large Scale Visual Recognition Challenge”. In: *International Journal of Computer Vision (IJCV)* 115.3 (2015), pp. 211–252. DOI: `10.1007/s11263-015-0816-y`.

[16] Kaiming He et al. “Deep Residual Learning for Image Recognition”. In: *CoRR* abs/1512.03385 (2015). arXiv: `1512.03385`.

[17] Alex Krizhevsky. *Learning multiple layers of features from tiny images*. Tech. rep. 2009.

[18] Tianyu Pang et al. “Towards Robust Detection of Adversarial Examples”. In: *Advances in Neural Information Processing Systems*. Ed. by S. Bengio et al. Vol. 31. Curran Associates, Inc., 2018.

[19] Florian Tramer et al. “On Adaptive Attacks to Adversarial Example Defenses”. In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle et al. Vol. 33. Curran Associates, Inc., 2020, pp. 1633–1645.

[20] Sicong Han et al. “Interpreting Adversarial Examples in Deep Learning: A Review”. In: *ACM Comput. Surv.* 55.14s (July 2023). ISSN: 0360-0300. DOI: `10.1145/3594869`.

[21] Paula Harder et al. “SpectralDefense: Detecting Adversarial Attacks on CNNs in the Fourier Domain”. In: *2021 International Joint Conference on Neural Networks (IJCNN)*. 2021, pp. 1–8. DOI: `10.1109/IJCNN52387.2021.9533442`.

[22] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. *Explaining and Harnessing Adversarial Examples*. 2015. arXiv: `1412.6572 [stat.ML]`.

[23] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. *Adversarial examples in the physical world*. 2017. arXiv: `1607.02533 [cs.CV]`.

[24] Aleksander Madry et al. *Towards Deep Learning Models Resistant to Adversarial Attacks*. 2019. arXiv: `1706.06083 [stat.ML]`.

[25] Greg Yang. "Wide feedforward or recurrent neural networks of any architecture are gaussian processes". In: *Advances in Neural Information Processing Systems* 32 (2019).

[26] Greg Yang. "Tensor programs ii: Neural tangent kernel for any architecture". In: *arXiv preprint arXiv:2006.14548* (2020).

[27] Greg Yang and Etai Littwin. "Tensor programs iib: Architectural universality of neural tangent kernel training dynamics". In: *International Conference on Machine Learning*. PMLR. 2021, pp. 11762–11772.

[28] Olivier J. Hénaff and Eero P. Simoncelli. *Geodesics of learned representations*. 2016. arXiv: `1511.06394 [cs.CV]`.

[29] Hugo Duminil-Copin. *Sixty years of percolation*. 2017. arXiv: `1712.04651 [math.PR]`.

[30] Dietrich Stauffer and Ammon Aharony. *Introduction to percolation theory*. Taylor & Francis, 2018.

[31] Béla Bollobás and Oliver Riordan. *Percolation*. Cambridge University Press, 2006.

[32] Markus Heydenreich, Remco Van der Hofstad, et al. *Progress in high-dimensional percolation and random graphs*. Springer, 2017.

[33] Ronald Meester and Rahul Roy. *Continuum percolation*. Vol. 119. Cambridge University Press, 1996.

[34] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2017. arXiv: `1412.6980 [cs.LG]`.

[35] Karen Simonyan and Andrew Zisserman. *Very Deep Convolutional Networks for Large-Scale Image Recognition*. 2015. arXiv: `1409.1556 [cs.CV]`.

[36] *Discussion for: Feature Visualization Objectives, https://github.com/tensorflow/lucid/issues/116*. June 2018.

[37] A. L. R. Bug et al. "Do Interactions Raise or Lower a Percolation Threshold?" In: *Phys. Rev. Lett.* 55 (18 Oct. 1985), pp. 1896–1899. DOI: `10.1103/PhysRevLett.55.1896`.

[38] Jean Bricmont, Joel L Lebowitz, and Christian Maes. "Percolation in strongly correlated systems: the massless Gaussian field". In: *Journal of statistical physics* 48 (1987), pp. 1249–1268.

[39] Pierre-François Rodriguez and Alain-Sol Sznitman. "Phase Transition and Level-Set Percolation for the Gaussian Free Field". In: *Communications in Mathematical Physics* 320.2 (Dec. 2012), pp. 571–601. ISSN: 1432-0916. DOI: `10.1007/s00220-012-1649-y`.

[40] Christopher Chalhoub et al. *Universality classes for percolation models with long-range correlations*. 2024. arXiv: `2403.18787 [cond-mat.stat-mech]`.

[41] Daniel Soudry et al. "The implicit bias of gradient descent on separable data". In: *Journal of Machine Learning Research* 19.70 (2018), pp. 1–57.

[42] Behnam Neyshabur. "Implicit regularization in deep learning". In: *arXiv preprint arXiv:1709.01953* (2017).

[43] David GT Barrett and Benoit Dherin. "Implicit gradient regularization". In: *arXiv preprint arXiv:2009.11162* (2020).

[44] Gauthier Gidel, Francis Bach, and Simon Lacoste-Julien. "Implicit regularization of discrete gradient dynamics in linear neural networks". In: *Advances in Neural Information Processing Systems* 32 (2019).

[45] Mikhail Belkin, Siyuan Ma, and Soumik Mandal. "To understand deep learning we need to understand kernel learning". In: *International Conference on Machine Learning*. PMLR. 2018, pp. 541–549.

[46] Arthur Jacot, Franck Gabriel, and Clément Hongler. "Neural tangent kernel: Convergence and generalization in neural networks". In: *Advances in neural information processing systems* 31 (2018).

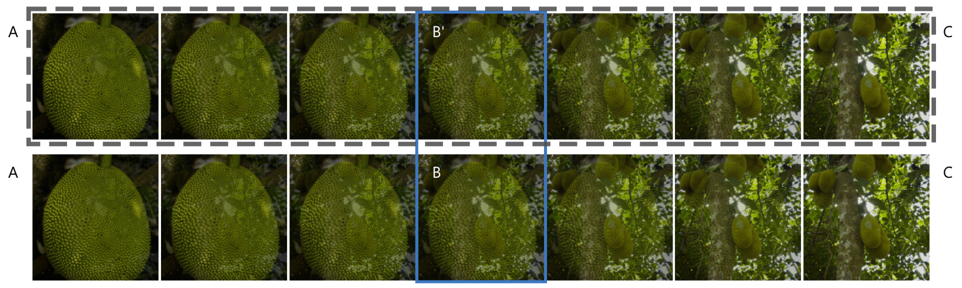

Figure 4: Images along two paths. We sampled and compared images along the updated path A→B'→C (dashed rectangle) and the original linear path A→C (bottom row). These paths are illustrated in the diagram in Figure 1. The high-loss intermediate point B and its optimized version B' are denoted by the blue vertical rectangle.

[47] Jaehoon Lee et al. "Wide neural networks of any depth evolve as linear models under gradient descent". In: *Advances in neural information processing systems* 32 (2019).

[48] Rotem Mulayoff, Tomer Michaeli, and Daniel Soudry. "The implicit bias of minima stability: A view from function space". In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 17749–17761.

A Feature visualization by optimization (FVO)

FVO is a technique for post-hoc interpretability of DNNs [7]. It starts from a small noise x_i at the input of a network that is forward propagated to the layer of interest. We aim to find inputs that activate a selected neuron (or neurons) within the layer of interest. Neurons can be either maximized or activated to a specific pattern through a loss function. The input is iteratively updated w.r.t. the gradient of the loss, similarly to the standard gradient descent training. Note that FVO does not produce a unique result, and so multiple distinct examples can be generated and their (input space) mode connectivity studied.

The optimization procedure can be easily adapted for real images. We start from the max loss intermediate point B and optimize until we reach B', which minimizes the loss (see left bottom diagram in Figure 1). The optimization is constrained within the orthogonal hyperplane to the vector AC, representing a simplified method for connecting modes inspired by Fort and Jastrzebski [6].

B Experimental details

To optimize the high loss point B (in Section 4), the Adam optimizer [34] was used with a learning rate of 0.005. We added regularization to the optimization with two additional terms: the mean squared error (MSE) between B and B', and a high-frequency term that penalizes changes in adjacent pixels. The weights for these terms, $\lambda_{\text{MSE}} = 0.1$ for the MSE and λ_{HF} ranging from $1e-8$ to $5e-6$ for the high-frequency term, were determined heuristically, varying by class and setup.

B.1 Comparison of two paths

Images along the two studied paths closely resemble each other in corresponding pairs, as shown in Figure 4. In fact, their difference is the identical pattern depicted in the right bottom of Figure 1, scaled in intensity proportionally to the interpolation factor α. This is likely due to the process we use to find B', which resembles the way gradient descent is used to find adversarial examples, the differences being that we restrict the search to a hyperplane and do require B' to remain close to B.

B.2 Adversarial examples

Adversarial examples in vision models are images that are generated by perturbing a *source* image in a way that either minimizes the loss for a selected target class (in targeted attacks) or maximizes the

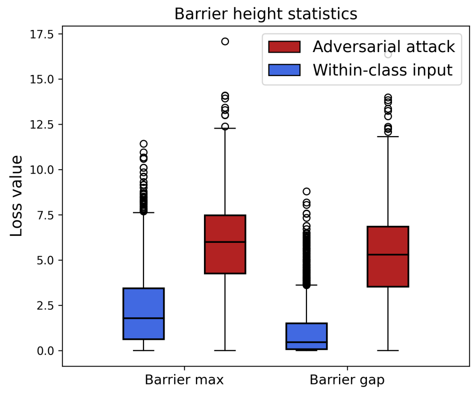

Figure 5: Barrier height statistics. Boxplots depict the maximum barrier heights and gaps (i.e., the difference between the barrier height and loss value of the higher-loss mode) for selected pairs from the ImageNet validation dataset. The barriers for real–real and real–adversarial example pairs are compared. A total of 5,000 representative pairs (5 per class) were utilized for each scenario.

loss for the true class (in untargeted attacks), causing the model to misclassify the image. The source image and the optimized adversarial example are often almost indistinguishable to a human observer. The process of generating and employing adversarial images to deceive the model is known as an adversarial attack (see Fig. 3).

B.2.1 Experiment

We used Adam (learning rate 0.005), 1024 iterations, cross-entropy loss with penalizations (0.1*MSE for image deviation and 1e-7*high-frequency penalty). From here, the mode connecting procedure is identical to the previously described: 1) Detect the max loss point B, 2) Optimize B to get B', 3) Interpolate over the new path A→B'→K'. However, after a single round of this procedure, non-negligible secondary barriers emerge in paths A→B' and B'→K', which exceed the loss threshold δ by a considerable margin (as depicted in Fig. 3). Secondary barriers can be successfully bypassed by a repeated application of the procedure to both segments of the A→B'→K' path individually. As a result, the final low-loss path will necessarily be more complex than for real–real modes addressed in Section 4.

B.2.2 Barrier height statistics

We observed significant differences in the barrier heights and shapes between two key scenarios: barriers between real–real modes and real–adversarial modes. To assess the statistical significance of this effect, we conducted the following experiment. Seven unique pairs of images per class were selected from the validation subset of ImageNet, encompassing 1,000 classes in total. Two pairs from each class, having the highest loss difference between the inputs, were excluded. The remaining 5,000 pairs were interpolated with 250 steps in between, and loss curves were generated. In the adversarial branch, pairs were formed by combining one example from the selected class with one from a randomly chosen different class. Adversarial examples were created in the same way as described above with the following settings: learning rate 0.005, deviation penalty weight 0.1, frequency penalty weight 1e-8, and 512 iterations. The subsequent procedure mirrored the one described earlier, yielding another set of 5,000 filtered pairs (5 per class). Descriptive statistics for the maximum barrier height and the gap (the difference between the peak height and the loss value of the higher mode/image) are depicted in Figure 5.

Real–adversarial mode pairs exhibit significantly higher barrier max, with mean and median 5.82 and 5.99, respectively. In contrast, real–real pairs have mean 2.27 and median 1.79. The effect is even more pronounced for barrier gaps, where real–adversarial have mean 5.19 and median 5.30, but real–real have mean 1.00 and median only 0.47. The described behavior was utilized to design a new method for adversarial detection.

Table 1: (adapted from [21]) Comparison of detection methods. Accuracy/AUC (%). The attacks are applied on the CIFAR-10/100 test set and the VGG-16 NET. Attacks are described in Appendix B.4.

Dataset	Detector	FGSM	BIM	PGD	Deepfool	C&W
CIFAR-10	LID	86.4 / 90.8	85.6 / 93.3	80.4 / 90.0	78.9 / 86.6	78.1 / 85.3
	Mahalanobis	95.6 / 98.8	97.3 / 99.3	96.0 / 98.6	76.1 / 84.6	76.9 / 84.6
	InputMFS	98.1 / 99.7	93.5 / 97.8	93.6 / 97.9	58.0 / 60.6	54.7 / 56.1
	LayerMFS	**99.6 / 100**	**99.2 / 100**	**98.3 / 99.9**	72.0 / 80.3	69.9 / 77.7
	LayerPFS	97.0 / 99.9	98.0 / 99.9	96.9 / 99.6	86.1 / 92.2	86.8 / 93.3
	Mode connectivity (ours)	69.8 / 76.8	95.2 / 98.6	91.8 / 97.4	**91.4 / 96.9**	**93.7 / 98.3**
CIFAR-100	LID	72.9 / 81.1	76.5 / 85.0	79.0 / 86.9	58.9 / 64.4	61.8 / 67.2
	Mahalanobis	90.5 / 96.3	73.5 / 81.3	76.3 / 82.1	**89.2 / 95.3**	89.0 / 94.7
	InputMFS	98.4 / 95.5	89.1 / 94.1	90.9 / 95.1	58.8 / 62.2	53.3 / 54.6
	LayerMFS	**99.5 / 100**	**97.1 / 99.5**	**97.0 / 99.7**	83.8 / 91.0	87.1 / 93.0
	LayerPFS	96.9 / 99.3	90.3 / 96.7	92.6 / 97.6	78.8 / 84.4	79.1 / 84.0
	Mode connectivity (ours)	67.3 / 72.8	84.3 / 92.5	88.9 / 95.2	86.8 / 94.0	**92.9 / 97.3**

B.3 Adversarial detection

Our simple detection leverages the relative lack of linear connectivity in natural–adversarial image pairs (as shown in Figure 5); First, we compute the (linear-path) loss curves for all training images x_i. Each loss curve is obtained from an input pair consisting of x_i (a training image) and x_{template,y_i} (a low-loss reference image from the same class, $y_i =$ argmax $f(x_i, \theta)$), which is preselected for each class. Second, concatenate the loss curves with logits $f(x_i, \theta)$, sorted in descending order, which provide information about prediction confidence. This step slightly enhances performance against simpler attacks, though it can be fully ablated for attacks like deepfool and C&W). Third, resulting feature vectors (loss curves+logits) are then used to train a classifier. Test images are processed through the same pipeline and classified by the model.

B.4 Adversarial detection benchmark

Here, we briefly overview benchmark by Harder et al. [21], our results, and employed attacks. The benchmark uses VGG-16 [35] model, CIFAR10/100 datasets, and employs a representative selection of adversarial attacks (FGSM [22], BIM [23], PGD [24], deepfool [8], and C&W [9]) in their untargeted forms.

B.4.1 Results

Interestingly, our method outperformed baselines for more advanced attacks (deepfool and the particularly challenging C&W) but underperformed on simpler (FGSM, BIM, PGD). This is a consequence of the nature of untargeted attacks, which generally aim to alter the prediction by maximizing the loss for the correct class. However, unlike FGSM, BIM, and PGD, both deepfool and Carlini & Wagner (C&W) also minimize the loss for a specific incorrect class, making the misclassification more controlled. Our algorithm relies on the existence of a distinct loss barrier, which is less pronounced in simpler attacks but more defined in advanced ones like deepfool and C&W. Complete results are provided in Table 1. Note that we used a classifier (KNN) that was optimized jointly for all attacks on validation data (in contrast to attack-dependent classifiers and hyperparameters in the benchmark). Additionally, unlike the best-performing baseline, our method does not require access to the feature maps.

B.4.2 Adversarial attacks

- **Fast Gradient Sign Method (FGSM)**: A rapid untargeted attack method that adjusts an input x_i by adding a small perturbation in the direction of the loss function's gradient:

$$x_{i,\text{adv}} = x_i + \epsilon \cdot \text{sign}(\nabla_{x_i} \mathcal{L}(x_i, y)). \tag{5}$$

- **Basic Iterative Method (BIM)**: An iterative enhancement of FGSM, applying small, repeated perturbations, ensuring each update remains within an ϵ-bounded range:

$$x_{i,\text{adv}}^{(n+1)} = \text{Clip}_{x_i,\epsilon}(x_{i,\text{adv}}^{(n)} + \epsilon \cdot \text{sign}(\nabla_{x_i} \mathcal{L}(x_{i,\text{adv}}^{(n)}, y))). \tag{6}$$

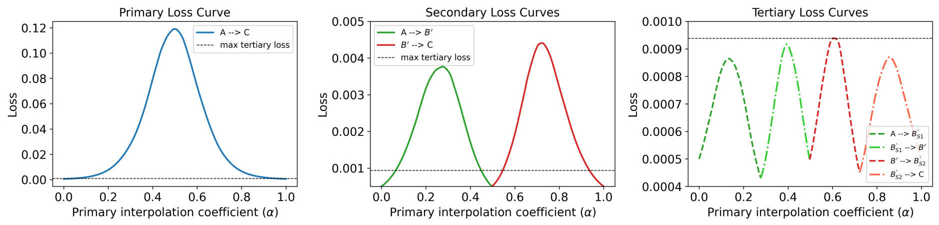

Figure 6: Input connectivity in untrained model. Loss curves for synthetic optimal inputs computed for an untrained model. **Left:** Primary loss curve, connecting the two optimal inputs. **Middle:** Secondary loss curves, connecting two original inputs through the intermediate point B$'$ (optimized from the primary barrier). **Right:** Tertiary loss curves, four-segment path connecting two original inputs through intermediate points B$'_{S1}$, B$'$, and B$'_{S2}$ (optimized first secondary barrier, optimized primary barrier, optimized second secondary barrier, respectively).

- **Projected Gradient Descent (PGD)**: A generalization of BIM that begins from a random point near x_i within the ϵ-ball.
- **Deepfool (DF)**: Iteratively modifies x_i to cross the closest decision boundary by minimizing the perturbation required to alter the model's classification.
- **Carlini & Wagner (C&W)**: A powerful optimization-based attack that minimizes the L_2 norm of the perturbation while ensuring misclassification by optimizing a loss function designed to reduce the model's confidence in the correct class. The attack solves the following objective:

$$\min \left\| \frac{1}{2}(\tanh(x_{i,\text{adv}}) + 1) - x_i \right\|^2 + c \cdot f\left(\frac{1}{2}(\tanh(x_{i,\text{adv}}) + 1)\right), \tag{7}$$

  where $f(z)$ is a function that encourages the logits to either lower the true class's score (untargeted) or raise the target class's score (targeted). The parameter c is optimized to balance minimizing the perturbation and ensuring successful misclassification. The C&W attack can be applied in both targeted and untargeted forms.

B.5 The existence of connectivity in untrained models

Motivated by our theoretical findings, we demonstrate that class-optimal inputs are connected even in untrained, randomly initialized models.

As there are no natural datasets for untrained models, we followed the analogous strategy as in the previous subsection and computed synthetic optimal inputs. The optimization procedure failed to converge on GoogLeNet, likely due to the high input dimension (3, 224, 224). Using the ResNet18, adapted for CIFAR10 (input shape (3, 32, 32), output (10)), starting from Gaussian noise $\mathcal{N}(0, 0.01)$, we were able to find inputs with losses below the specified threshold 0.0005. Adam optimizer with learning rate 0.05 and weight decay 1e-7 was used for a maximum of 4096 iterations or until reaching the desired loss. A diversity of inputs was achieved through high-frequency penalization with a weight 2.5e-7 of one of the inputs in each pair.

The two modes were linearly interpolated, and the loss curve is shown in Fig. 6 left. After two iterations of the standard mode-connecting procedure, we obtained a four-segment piece-wise linear path. A$\rightarrow$B$'_{S1}$$\rightarrow$B$'$$\rightarrow$B$'_{S2}$$\rightarrow$C that has los below $\delta = 0.001$ everywhere. Note that points B$'_{S1}$ and B$'_{S2}$ were obtained by optimizing the secondary barriers B$_{S1}$ and B$_{S2}$. All the loss curves and barriers are depicted in Fig. 6. In Appendix E.1, we also show how the connectivity (primary loss curve, i.e., loss on the linear interpolant path) changes during the training.

B.6 Towards optimal input manifold

Synthetic images offer a controlled environment for examining input space mode connectivity. Here, images were generated using the feature visualization by optimization technique on GoogLeNet

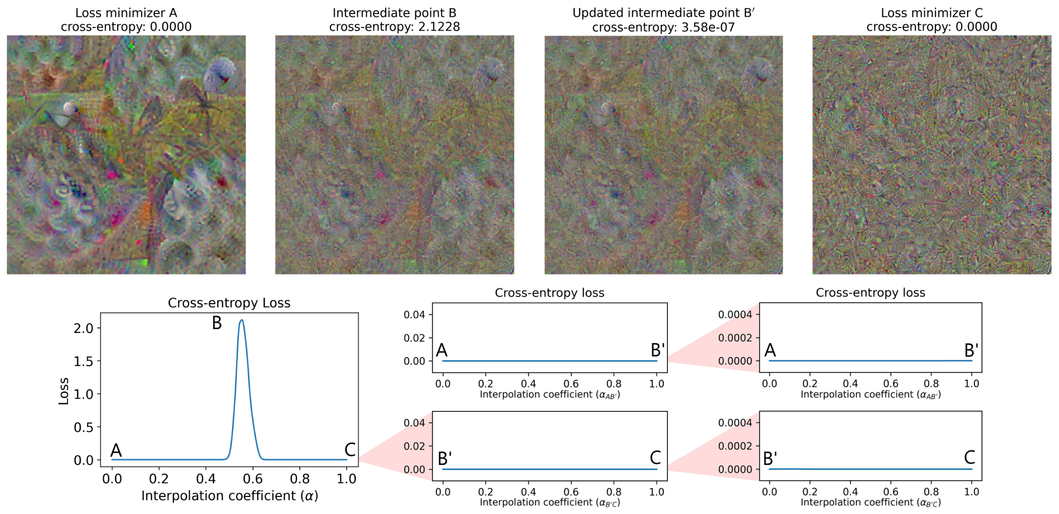

Figure 7: Synthetic modes. Synthetic images generated through input optimization from Gaussian noise. **Top row:** Modes A and C, interpolated high-loss point B and its optimized counterpart B'. **Bottom row:** The primary loss barrier was bypassed by a single round of optimization. The new path segments A→B' and B'→C are barrier-free up to several orders lower loss threshold.

trained on ImageNet. Starting from Gaussian noise $\mathcal{N}(0,1)$, we optimized a surrogate objective function (dot product $\times$ sqrt of cosine similarity $\times$ $1/2$), instead of directly optimizing for cross-entropy. We used this heuristic approach to generate synthetic inputs that better match human visual perception, as cross-entropy optimization, for unclear reasons, is less effective for this purpose [36]. Through varied regularizations and transformations during optimization, we obtained two distinct modes, each exhibiting virtually zero cross-entropy loss, as illustrated in Figure 7. Mode A exhibits partial resemblance to natural objects of its class (*golf ball*). Mode B contains only high-frequency patterns or noise, lacking obvious semantic structure, and was chosen intentionally for this purpose. The bottom row of Figure 7 demonstrates that even such different types of synthetic modes can be connected. Moreover, since the modes exhibited effectively zero loss, there are no secondary barriers after optimizing the max loss intermediate point B (almost up to the numerical precision).

It is evident that a continuum of optimal inputs exists for a given class within the model. By employing a systematic approach to find and connect these optimal modes, the class-optimal manifold can be partially explored and analyzed. This approach would provide a novel perspective on DNN interpretability, especially as it extends the concept of feature visualization by optimization [7], beyond discrete optimal inputs or activation atlases.

C Percolation theory

Percolation theory, originally developed to describe the structure of a porous material, turned into a widely studied model in statistical physics and graph theory [29].

Let $\mathcal{G}$ be an $N \times N$ grid, where at each cell, one inserts a ball with (Bernoulli distribution) probability p and leaves it empty with probability $1-p$. Site percolation theory considers the question: what is the probability that for any two nonempty cells A and B in $\mathcal{G}$, there is a path of nonempty cells that connects the two. When p is small, the probability is zero, while when $p \to 1$, there is almost certainly a connecting path. The latter phase case is called percolation. In the limit $N \to \infty$ there is a sharp transition at a critical probability $p = p_c$, that separates the two phases. Figure 8 shows site percolation in two dimensions near the critical probability.

The percolation model has a straightforward generalization to higher dimensions, as well as to more than the two labels (empty and nonempty), and to a general probability distribution for choosing the labels. In particular, the choice of labels between different cells can be correlated. The structure of the correlation affects the transition to percolation. Intuitively, one expects that higher correlations imply a higher probability of having connecting paths. Correlated percolation is an active field of research in physics and mathematics (see e.g. [37–40]).

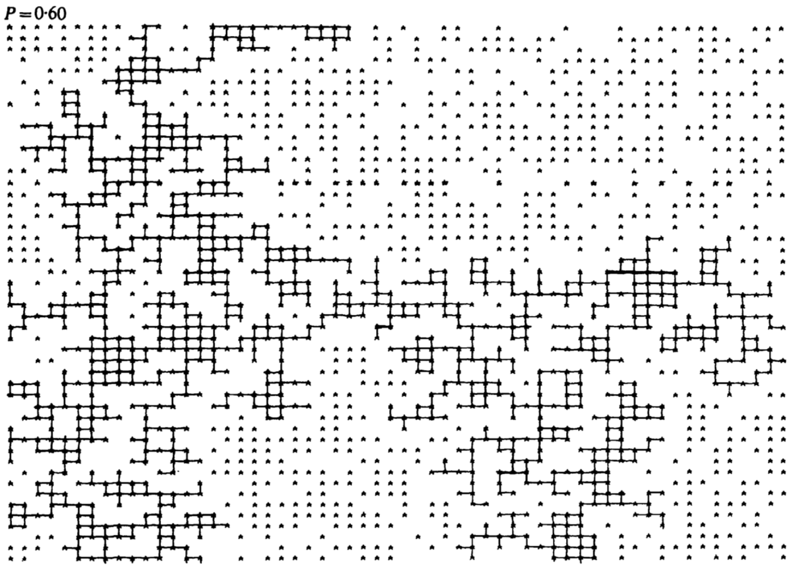

Figure 8: An illustration of site percolation in two dimensions near the critical probability [40] (the critical probability for two-dimensional site percolation is around 0.59). In this illustration, an "infinite" cluster can be observed. However, it is notable that a significant number of points do not belong to it.

In the framework of our work, the grid is defined by the ε-cells in the data space, while the labels are the intervals. Since the dimension of the grid is very large, the percolation probability p_c, which is inversely proportional to the dimension, is very small. Hence, our analysis at initialization is at the percolation phase.

We provide a dictionary (see Table 2) for the terms used in percolation theory in statistical physics and graph theory communities.

Table 2: Dictionary for percolation theory terms used in statistical physics and graph theory.

statistical physics	graph theory
site	vertex (node)
bond	edge
cluster	connected component
infinite cluster	infinite connected component
percolation threshold	critical probability
order parameter	size of the largest component
lattice animal	connected subgraph (not infinite)

D Conjecture 4.1 - Proof Sketch

We believe that Conjecture 4.1 offers valuable insight into the phenomenon of mode connectivity, though it is not a fully rigorous proof. Here, we present a detailed derivation of the proof sketch, extending the overview in the main text.

D.1 Assumptions and Generalizations

D.1.1 The Network

We justify conjecture 4.1 for fully connected neural networks defined by a set of $L \in \mathbb{N}$ layers, characterized by L weight matrices and L bias vectors, denoted as $\{\theta^{l,l-1}, \theta^{l} \mid l = 1, \ldots, L\}$. The first layer is defined as:

$$f^1 = \theta^{1,0}x + \theta^1. \tag{8}$$

Each subsequent layer $l = 2, \dots, L$ satisfies:

$$f^l = \theta^{l,l-1}\phi(f^{l-1}) + \theta^l, \tag{9}$$

where ϕ is a Lipschitz-bounded activation function that acts on the vector in a pointwise manner.

We then define the network's prediction using a non-linear final activation $\varphi : \mathbb{R} \to [0,1]$ such that:

$$f = \varphi(f^L). \tag{10}$$

The generalization to other networks within the tensor programs formalism [25–27] (and then using φ) is straightforward, as long as the nonlinearities are Lipschitz. The classes of networks described by this formalism is very diverse, and includes most of the relevant types of neural networks, such as fully connected neural networks, recurrent neural networks, long short-term memory units, gated recurrent units, convolutional neural networks, residual connections, batch normalization, graph neural networks, and attention mechanisms.

The reason that we can generalize our result to any network in this class is that in our justification, we only use the semi-linear structure of fully connected neural networks, and, by definition, any network in this class can be described as a composition of global linear operations and pointwise non-linear functions.

D.1.2 Initialization and Input Distribution

As we will see, we can always divide the input space into sufficiently small hypercubes. Therefore, it is not crucial how we initialize our network, as long as it remains well-defined as $d_x \to \infty$.

The only requirement is that for every $0 < p < 1$ and $x \in X'$, the probability for having every label in Y_δ is at least $\tilde{\delta}$, where $\tilde{\delta} = O(\delta)$, except for a subset of intervals, whose combined probability is less than $1 - p$.

We conjecture that this holds for most well-normalized initializations and networks, and any finite, arbitrarily large $0 < R$-radius balls:

$$X' = B_R^{d_x} = \left\{x \in \mathbb{R}^{d_x} \mid \|x\| \leq R\right\} . \tag{11}$$

D.1.3 The Loss Function

We work with $\mathcal{L}(y, y') = \|y - y'\|$. However, the generalisation to any loss function that satisfies $\mathcal{L} : Y \times Y \to \mathbb{R}^+$, such as it is Lipschitz continues in both arguments, and for every $y \in Y$:

$$\mathcal{L}(y, y) = 0 , \tag{12}$$

is straightforward.

D.1.4 Dividing the Input Space

Now that we have provided all of the necessary definitions, we can proceed to justify the conjecture. We will begin by demonstrating that small changes in the input space generate small changes in the network's output.

Lemma D.1 (Lipschitz Continuity of the Network)**.**

A neural network, as described above, is almost always Lipschitz continuous at initialization. This means that for any probability arbitrarily close to one, $0 < p < 1$, there exists some $0 < M$, such that for every $x, \Delta x \in X$:

$$\|f(x + \Delta x; \theta) - f(x; \theta)\| \leq M \|\Delta x\| . \tag{13}$$

An immediate consequence of this lemma is that for the situation described in conjecture 4.1, for any probability arbitrarily close to one, $0 < p < 1$, there exists some $0 < \varepsilon$ such that for every $x, \Delta x \in X$, if $\|\Delta x\| < \varepsilon$, then:

$$\|f(x + \Delta x; \theta) - f(x; \theta)\| \leq \delta - \delta' \leq \delta. \tag{14}$$

This result implies that we can divide our input space into $O(2N)$ classes, as in equation 4. This means that if we divide our input space into cubes of size $\frac{\varepsilon}{\sqrt{d_x}}$, not only does every input in each cube belong to the same class, but also that if two different inputs $x_0, x_1 \in X$ satisfy $\|f(x_0;\theta) - f(x_1;\theta)\| \leq \delta'$, then their cubes share the same interval.

Proof of Lemma D.1.

We prove the lemma by induction. We start with the induction base - the first layer. Since ϕ is Lipschitz continuous, we know that there exists some $0 < m$ such that for every $r, \Delta r \in \mathbb{R}$:

$$|\phi(r + \Delta r) - \phi(r)| \leq m\,|\Delta r|\,, \tag{15}$$

which means that for every $x, \Delta x \in \mathbb{R}^{d_x}$:

$$\|\phi(x + \Delta x) - \phi(x)\|^2 = \sum_{i=1}^{d_x} (\phi(x_i + \Delta x_i) - \phi(x_i))^2 \leq \sum_{i=1}^{d_x} (m\Delta x_i)^2 = m^2\,\|\Delta x\|^2\,. \tag{16}$$

Thus, for every $x, \Delta x$:

$$\|\phi(x + \Delta x) - \phi(x)\| \leq m\,\|\Delta x\|\,. \tag{17}$$

Multiplying by $\theta^{1,0}$ and adding θ^1, we get:

$$f^1(x + \Delta x) - f^1(x) = \theta^{1,0}\phi(x + \Delta x) + \theta^1 - (\theta^{1,0}\phi(x) + \theta^1) = \theta^{1,0}(\phi(x + \Delta x) - \phi(x)). \tag{18}$$

Using the subordinate norm of the matrix, we find:

$$\left\|f^1(x + \Delta x) - f^1(x)\right\| \leq \left\|\theta^{1,0}\right\| \|\phi(x + \Delta x) - \phi(x)\| \leq \left\|\theta^{1,0}\right\| m\,\|\Delta x\|\,. \tag{19}$$

Which means that for every $0 < p < 1$, we can bound the change in $f^1(x + \Delta x) - f^1(x)$ by:

$$\left\|f^1(x + \Delta x) - f^1(x)\right\| \leq m'\,\|\Delta x\|\,. \tag{20}$$

We can continue this process by induction, showing that the same holds for every layer, which completes our proof. □

Remark D.1.

It should be noted that the bound we found using the subordinate norm was sufficient for our needs, however, it is far from being the optimal one.

This is not a problem for our work, as all that we require is any Lipschitz bound, and then we can take ε to be arbitrarily small. However, if one wishes to find a better bound, which also depends on the distance between inputs, or to investigate neural networks in the infinite limit appropriately, more care will be needed.

D.1.5 Finding the Path

To find now a δ-connected path between the two inputs, all we need to do is to find a connected path of cubes between the cube of the first input to the cube of the second with the same intervals.

Assuming that the cube's intervals are drawn randomly and independently from each other with similar probability $p = O(\delta)$, we encounter a **classic high-dimensional percolation problem**, where the cubes represent the points in the graph, and they are connected if they share the same interval. It is well known that in such cases, the probability that two points are connected grows rapidly with the dimension, as shown in the conjecture. Specifically, if we ask what is the probability that a certain point will be part of the infinite cluster, we know that in the high-dimensional limit, we can neglect closed loops [30]. This is known as a "mean field approximation."

The probability that one input is not part of the infinite cluster is the probability that all nearby cubes connected to it are also not part of the infinite cluster. Thus, the probability that a point is part of the infinite cluster P satisfies:

$$1 - P = (1 - pP)^{d_x} \lesssim e^{-d_x pP} \to P \gtrsim 1 - e^{-d_x pP}\,. \tag{21}$$

Defining $q = d_x p$, we find a lower bound for P by solving the equation:

$$P = 1 - e^{-qP}. \tag{22}$$

Defining $Q = 1 - P$, we get:

$$\begin{gathered}1 - Q = 1 - e^{-q(1-Q)} = 1 - e^{-q}e^{qQ} \approx 1 - e^{-q}(1 - qQ + O(q^2Q^2)) \to \\ Q = e^{-q}(1 - qQ + O(q^2Q^2)) \to (1 + qe^q)Q = e^{-q} + O(q^2Q^2).\end{gathered} \tag{23}$$

Which implies:

$$Q = \frac{e^{-q}}{1 + qe^q} + O(q^2Q^2) = O(e^{-q}) = O(e^{-pd_x}), \tag{24}$$

thus completing our justification.

E Linear Connectivity

As discussed above, we justified conjecture 4.1 only for neural networks at initialization, which raised the question, whether this property persists after training. In practice, we observed that not only do the inputs tend to remain connected, but they also tend to be connected approximately linearly, with only a relatively small barrier. Furthermore, by combining only a few linear paths, we can almost completely eliminate this barrier. We speculate that this phenomenon results from *implicit regularization*. Implicit regularization refers to an inherent tendency of a learning system to prefer simpler hypothesis functions [41–48]. We propose that one kind of simplicity preferred by deep learning is generally not to change between similar inputs if possible. This tendency, combined with geometric considerations, could explain the observed behavior; however, more work is needed to substantiate this.

Intuitively, this is not particularly surprising. For any system to generalize effectively, it must exhibit a preference for simpler hypothesis functions. In the context of overparameterized models, this preference is often referred to as *regularization*, and when not explicitly imposed, it is termed "implicit regularization". A reasonable form of simplicity involves not varying drastically between similar inputs.

Extensive research has been conducted on implicit regularization from the perspective of parameter space. However, its understanding from the input space perspective remains limited. A notable study [48] demonstrated that at equilibrium, the second derivative of neural networks with respect to the inputs is bounded and can be minimized as the learning rate increases.

Another possible approach is to consider the Neural Tangent Kernel (NTK) limit. It is known that, in the infinite-width limit, neural networks exhibit linear-like behavior [46, 47]. In this limit, their evolution over time can be described by a kernel, a two-point matrix function $\Theta(x, x')$, which generally tends to be larger where the two inputs $x \simeq x'$ are similar.

In this regime, for gradient descent, the final prediction can be viewed as an "averaged sum" [47] of the data, where for every $x \in X$, more weight is assigned to label of inputs that were closer to x (according to the kernel). And as it is reasonable to assume that a linear combination of two inputs will be closer to the original inputs than to any other input, that could partially explain the linear mode connectivity.

Another avenue that could be promising is to consider $f(\alpha x_0 + (1-\alpha)x_1, \theta)$, and decompose it as $f(\alpha x_0, \theta) + f((1-\alpha)x_1, \theta)$ plus an additional term, whose magnitude should be shown to be small. If both $f(\alpha x_0, \theta)$ and $f((1-\alpha)x_1, \theta)$ align in the same direction, we would obtain the correct label after applying the softmax function.

It is important to emphasize that the ideas presented here are speculative, and only represent our preliminary conjectures. We have not yet validated these concepts, and more rigorous research is needed to confirm or refute these notions.

E.1 Connectivity evolution throughout training

Following the observation of connectivity in untrained models (Sec. B.5) and hypothesizing the role of implicit regularization in trained models, we studied the temporal evolution of the connectivity over training batches and epochs. To maintain comparability for the early stages of training, where the model performs poorly on real data, we again employed synthetic optimal inputs. The optimization process was adapted from Sec. B.5 (ResNet18, input shape (3, 32, 32), output shape 10, starting from Gaussian noise $\mathcal{N}(0, 0.01)$, Adam optimizer with learning rate 0.05 and weight decay 1e-7,

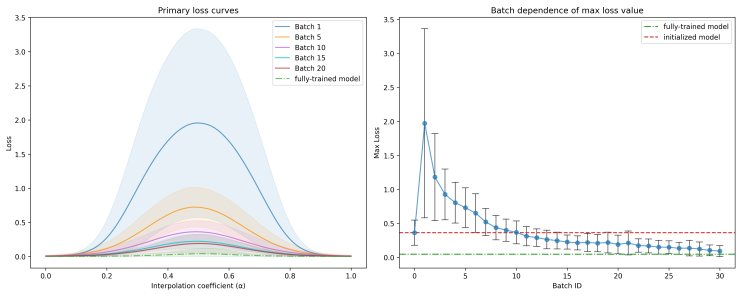

Figure 9: Batch evolution of the averaged loss barriers. Batches are indexed starting from one; zero means an untrained model. Shaded area and error bars show standard deviation.

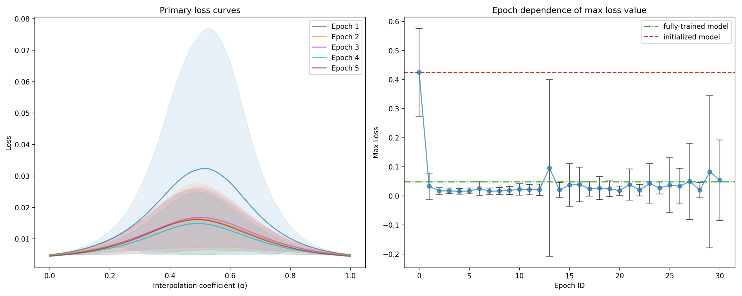

Figure 10: Epoch evolution of the averaged loss barriers Epochs are indexed starting from one; zero means untrained model. Shaded area and error bars show standard deviation.

maximum of 4096 iterations or until reaching the desired loss), with difference of using the loss threshold $\delta = 0.005$ for convenience and faster convergence. A diversity of inputs was achieved through high-frequency penalization with weight 2.5e-7 of one of the inputs in each pair. We prepared 5 pairs for each class and computed primary loss curves. Averaged loss curves (over all pairs, classes and model stages) for the first 30 batches are shown in Fig. 9, and similarly for the first 30 epochs in Fig. 10.

While we observe the increase of connectivity (decrease of loss barriers) in the early phases of training, it saturates and variance begins to grow at some point. There are likely more competing effects taking part over the training. It is important to note that the barrier height and complexity depend on the losses of the interpolated inputs.

F Extra results for the Section 4

Here, we present additional examples of connectivity on pairs drawn from the validation datasets of ImageNet (see Figure 11) and CIFAR10 (see Figure 12), using GoogLeNet and ResNet18, respectively. Note that the loss barrier can be negligibly small for some pairs even before the initial round of barrier optimization.

G Limitations

G.1 Section 4

Experiments were carried out on common vision models. The potential generalization of the concept to other data modalities was not tested. Our approach requires a continuous input space up to standard numerical precision. Therefore, technical adjustments would be necessary for applications beyond vision models, such as language models that use tokenization.

The main limitation, other than the basic assumptions discussed in section D.1, is that we treated our problem as a standard model of percolation, assuming no correlation between different sites. However, this assumption does not hold for neural networks, not even at initialization. We argue,

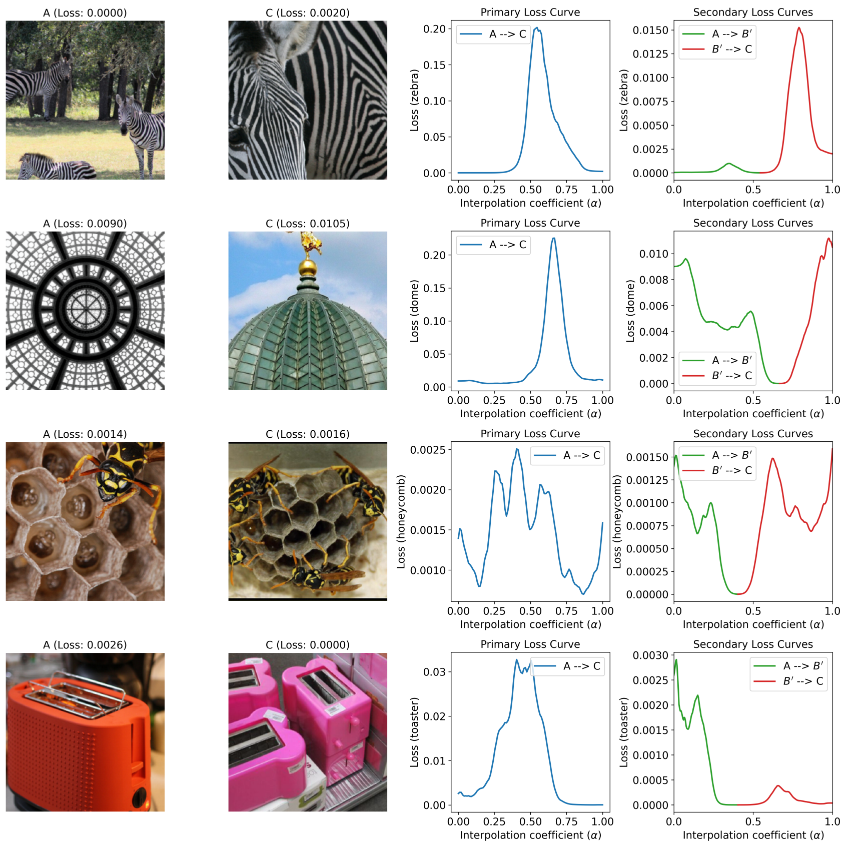

Figure 11: GoogLeNet(ImageNet) loss curves. Representative examples of connectivity in a trained GoogLeNet for selected classes. Each row contains a pair of same-class images. The primary loss curve is obtained from linear interpolation A→B, and a secondary loss curve A→B'→C. Note that the highest loss value in the secondary curves is at least one magnitude lower than in the primary curves.

however, that since correlations in neural networks are positive, the actual scenario should be more favorable. Nevertheless, diligent investigation and exact proofs are necessary to confirm this.

G.2 Section E

The ideas discussed in that section are speculative and represent our initial hypotheses. These concepts have not yet been validated, and further rigorous research is required to either confirm or disprove them.

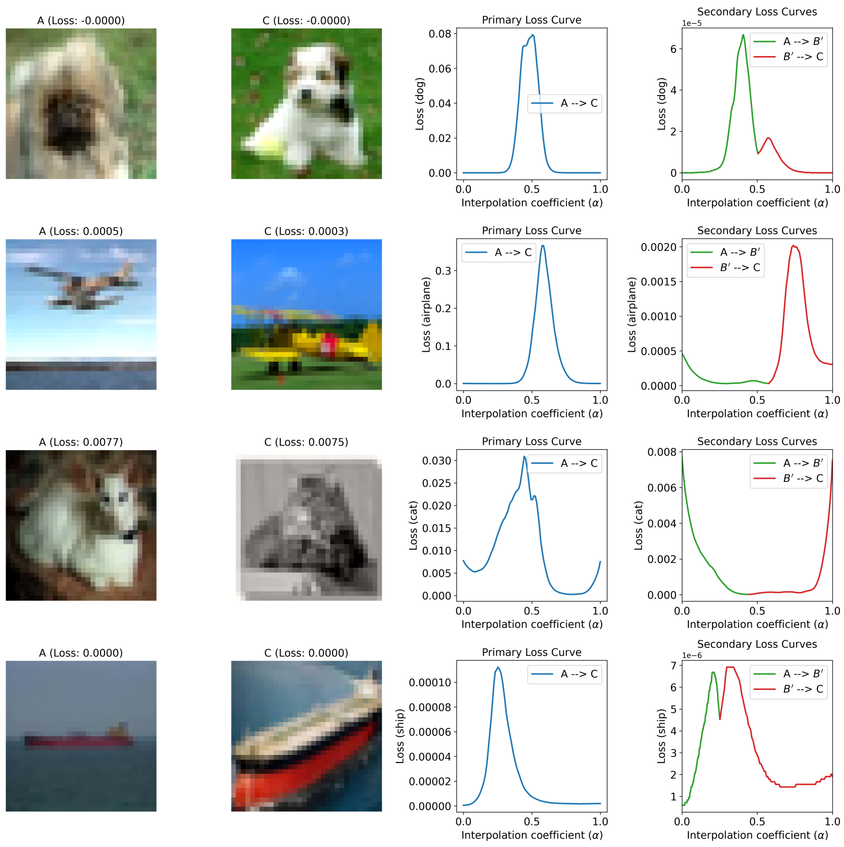

Figure 12: ResNet18(CIFAR10) loss curves. Representative examples of connectivity in a trained ResNet18 for selected classes. Each row contains a pair of same-class images. The primary loss curve is obtained from linear interpolation A→B, and a secondary loss curve A→B'→C. Note that the highest loss value in the secondary curves is at least one magnitude lower than in the primary curves.