

Can Neural Nets Learn the Same Model Twice? Investigating Reproducibility and Double Descent from the Decision Boundary Perspective

Gowthami Somepalli¹, Liam Fowl¹, Arpit Bansal¹, Ping Yeh-Chiang¹, Yehuda Dar²,
Richard Baraniuk², Micah Goldblum³, Tom Goldstein¹

¹ University of Maryland, College Park

{gowthami, pchiang, tomg}@cs.umd.edu
lfowl@math.umd.edu, bansal01@umd.edu

² Rice University

{ydar, richb}@rice.edu

³ New York University

goldblum@nyu.edu

Abstract

We discuss methods for visualizing neural network decision boundaries and decision regions. We use these visualizations to investigate issues related to reproducibility and generalization in neural network training. We observe that changes in model architecture (and its associated inductive bias) cause visible changes in decision boundaries, while multiple runs with the same architecture yield results with strong similarities, especially in the case of wide architectures. We also use decision boundary methods to visualize double descent phenomena. We see that decision boundary reproducibility depends strongly on model width. Near the threshold of interpolation, neural network decision boundaries become fragmented into many small decision regions, and these regions are non-reproducible. Meanwhile, very narrow and very wide networks have high levels of reproducibility in their decision boundaries with relatively few decision regions. We discuss how our observations relate to the theory of double descent phenomena in convex models. Code is available at <https://github.com/somepago/dbViz>.

1. Introduction

The superiority of neural networks over classical linear classifiers stems from their ability to slice image space into complex class regions. While neural network training is certainly not well understood, existing theories of neural network training mostly focus on understanding the geometry of loss landscapes [5, 8, 25]. Meanwhile, considerably less is known about the geometry of class boundaries. The geometry of these regions depends strongly on the inductive bias of neural network models, which we do not currently have tools to rigorously analyze. To make things worse, the

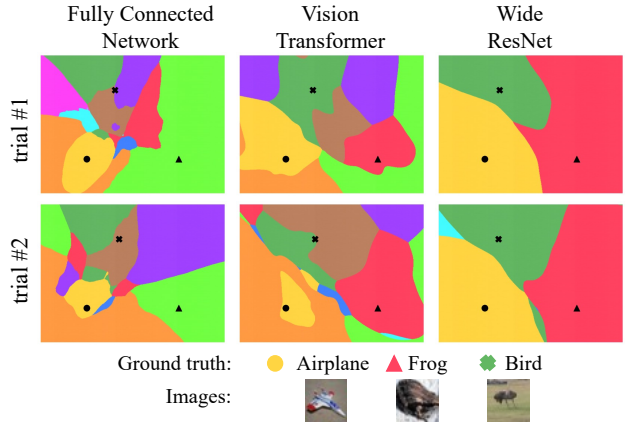


Figure 1. The class boundaries of three architectures, plotted on the plane spanning three randomly selected images. Each model is trained twice with random seeds. Decision boundaries are reproducible across runs, and there are consistent differences between the class regions created by different architectures.

inductive bias of neural networks is impacted by the choice of architecture, which further complicates theoretical analysis.

In this study, we use empirical tools to study the geometry of class regions, and how neural architecture impacts inductive bias. We do this using visualizations and quantitative metrics calculated using realistic models. We start by presenting simple methods for decision boundary visualization. Using visualization as a tool, we do a deep dive on three main issues:

- Do neural networks produce decision boundaries that are consistent across random initializations? Put simply, can a neural network learn the same model twice? We see empirically that the decision boundaries of a network have strong similarities across runs, and we confirm this using quantitative measurements.

- Do different neural architectures have measurable differences in inductive bias? Indeed, we find clear visible differences between the class regions of different model architectures (e.g., ResNet-18 vs ViT).
- We use decision boundary visualizations to investigate the “double descent” phenomenon. We see that decision boundaries become highly unstable and fragmented when model capacity is near the interpolation threshold, and we explore how double descent in neural networks relates to known theory for linear models.

2. Plotting decision boundaries

Most prior work on decision boundary visualization is for the purpose of seeing the narrow margins in adversarial directions [18, 21]. Fawzi et al. [11] visualize the topological connectivity of classification regions. To facilitate our studies, we seek a general-purpose visualization method that is simple, controllable, and captures important parts of decision space that lie near the data manifold.

2.1. On-manifold vs off-manifold behavior

When plotting decision boundaries, it is important to choose a method that captures the behavior of models near the data manifold. To understand why, consider the plots of decision boundaries through planes spanning randomly chosen points in input space as shown in Figure 2. We see that decision regions are extremely smooth and uniform with few interesting features. The training process, which structures decision boundaries near the data manifold (e.g. Fig. 1), fails to produce strong structural effects far from the manifold (e.g. Fig. 2).

The uniform off-manifold behavior is not particular to our training method or architecture but is rather an inevitable consequence of the concentration of measures phenomenon [24, 31]. In fact, we can show that any neural network that varies smoothly as a function of its input will assume nearly constant outputs over most of input space. The proof of the following result is in Appendix A.

Lemma 2.1 *Let $f : [0, 1]^n \rightarrow [0, 1]$ be a neural network satisfying $|f(x) - f(y)| \leq \frac{L}{\sqrt{n}} \|x - y\|$. Let $\bar{f}$ denote the median value of f on the unit hypercube. Then, for an image $x \in [0, 1]^n$ of uniform random pixels, we have $|f(x) - \bar{f}| \leq t$ with probability at least*

$$1 - \frac{Le^{-2\pi nt^2/L^2}}{\pi t\sqrt{n}}.$$

2.2. Capturing on-manifold behavior

The lemma above shows the importance of capturing the behavior of neural networks near the data manifold. Unfortunately, the structure of image distributions is highly complex and difficult to model. Rather than try to identify and

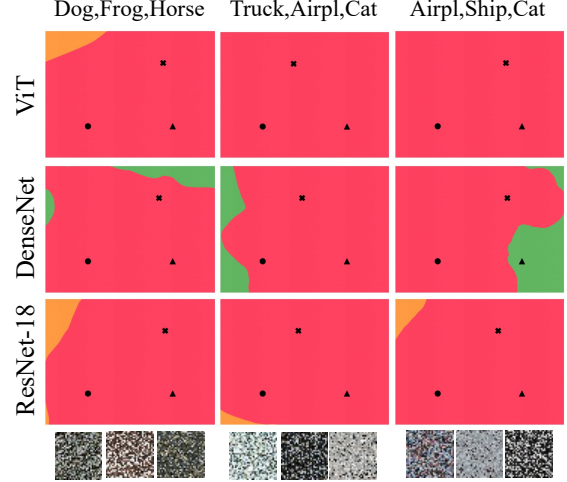


Figure 2. Off-manifold decision boundaries near “random” images created by shuffling pixels in CIFAR-10 images. Each column’s title shows the labels of the unshuffled base images. Below each column we show the shuffled image triplet. Color-class mapping is as follow Red:Frog, Green:Bird, Orange:Automobile.

flatten the complex structures on which images lie, we take an approach that is inspired by the recent success of the highly popular paper on the mixup regularizer [40], which observed that, in addition to possessing structure near the data manifold, *decision boundaries are also structured in the convex hull between pairs of data points*.

We take a page from the mixup playbook and plot decision boundaries along the convex hull between data samples. We first sample a triplet $(x_1, x_2, x_3) \sim \mathcal{D}^3$ of i.i.d. images from the distribution $\mathcal{D}$. Then, we construct the plane spanned by the vectors $\vec{v}_1 = x_2 - x_1$, $\vec{v}_2 = x_3 - x_1$ and plot the decision boundaries in this plane. To be precise, we sample inputs to the network with coordinates

$$\alpha \cdot \max(\vec{v}_1 \cdot \vec{v}_1, |\text{proj}_{\vec{v}_1} \vec{v}_2 \cdot \vec{v}_1|) \vec{v}_1 + \beta (\vec{v}_2 - \text{proj}_{\vec{v}_1} \vec{v}_2)$$

for $-0.1 \leq \alpha, \beta \leq 1.1$. This plotting methods using planes has several advantages. It shows the regions surrounding multiple data points at once and also the decision boundaries between their respective classes, using just one plot. Furthermore, these classes can be chosen by the user. It also focuses on the convex hull between points rather than random directions that may point away from the manifold.

Figure 1 shows decision regions plotted along the plane spanned by three data points chosen at random from the *Airplane*, *Frog*, and *Bird* classes of CIFAR-10 [23]. In these plots, each color represents a class label. Same color-class schema is maintained throughout the paper and can be seen in legends of multiple plots.

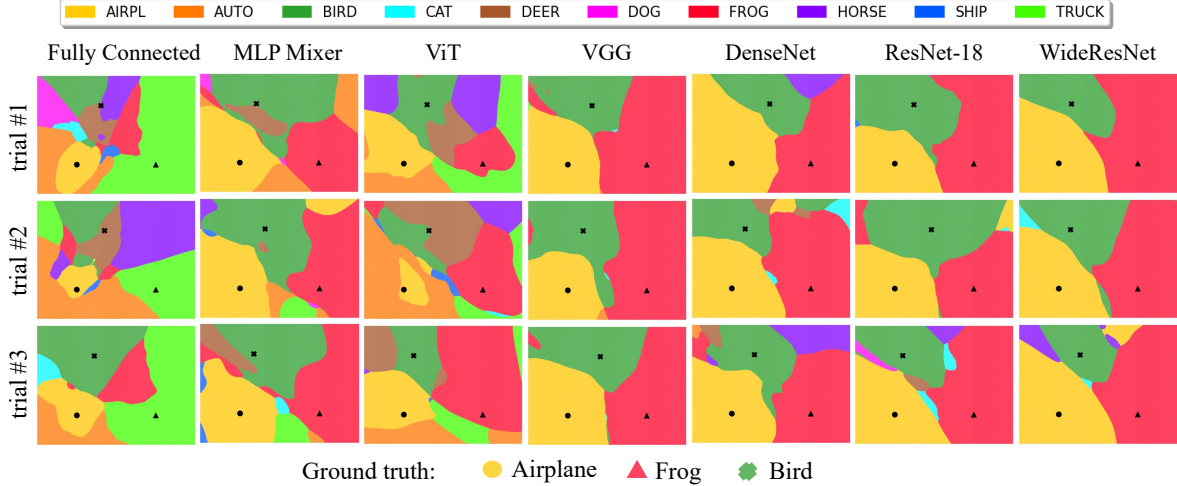


Figure 3. Decision regions through a triplet of images, for various architectures (columns) and initialization seeds (rows).

2.3. Experimental Setup:

Architectures used: We select several well-known networks from diverse architecture families¹. We consider a simple Fully Connected Network with 5 hidden layers and ReLU non-linearities, DenseNet-121 [20], ResNet-18 [17], WideResNet-28x10, WideResNet-28x20, WideResNet-28x30 [38], ViT [9], MLP Mixer [35], and VGG-19 [32]. For fast training, our ViT has only 6 layers, 8 heads, and patchsize 4. The custom MLP Mixer we use has 12 hidden layers with hidden embedding dimension 512 and patch size 4. Unless otherwise stated, architectures are trained for 100 epochs using SGD optimizer, and 3 multi-step learning rate drops. Random Crop and Horizontal Flip data augmentations are used in training. For distillation experiments, we also use a ViT-S/16 pretrained on ImageNet [7] as a teacher [36]. Some experiments use the Sharpness-Aware Minimization (SAM) optimizer [12] adversarial radius set to $\rho = 0.01$.

We select learning rates using a grid search across $\{0.001, 0.002, 0.005, 0.01, 0.02, 0.05\}$ for each architecture and optimizer (Adam [22] and SGD) combination, and training for 200 epochs. Mean test accuracy over 3 runs per model is reported in Table 1.

3. Model reproducibility and inductive bias

It is known that neural networks can easily overfit complex datasets, and can even interpolate randomly labeled images [39]. Despite this flexibility, networks have an important *inductive bias* – they have a strong tendency to converge on decision boundaries that generalize well. Our goal in this section is to display the inductive bias phenomenon

using decision boundary visualizations. We ask two questions:

- Can a model replicate the same decision boundaries twice, given different random initializations?
- Are there disparities between the inductive biases of different model families that result in different decision boundaries?

Below, we consider various sources of inductive bias, including neural architecture family, network width, and the choice of optimizer.

3.1. Inductive bias depends on model class

We choose three random images from the CIFAR-10 training set, construct the associated plane through input space, and plot the decision regions for 7 different architectures in Figure 3. For each model, we run the training script three times with different random initializations.

Several interesting trends emerge in this visualization. First, we observe systematic differences between model families. Convolutional models all share similar decision boundaries, while the boundaries of Fully Connected Nets, ViT, and MLP Mixer show noticeable differences. For example, ViT and MLP Mixer consistently show the presence of an orange “Automobile” region that CNNs do not. Fully Connected Nets show considerably more complex and fragmented decision regions than other model families.

At the same time, we observe strong reproducibility trends across runs with different random seeds. This trend is particularly high for convolutional architectures, and the effect is quite strong for WideResNet, which leads us to hypothesize that there may be a link between model width and reproducibility – an issue that we will investigate in more detail below.

¹Architecture implementations from <https://github.com/kuangliu/pytorch-cifar> and <https://github.com/lucidrains/vit-pytorch>

3.2. Quantitative analysis of decision regions

The visualizations in Figure 3 suggest that reproducibility is high within a model class, while differences in inductive bias result in low similarities across model families. To validate our intuitions, we use quantitative metrics derived from the decision plots averaged over many trials to provide a more sensitive and conclusive analysis.

Reproducibility Score: We define a metric of similarity between the decision boundaries of pairs of models. We first sample triplets $T_i = (x_0, x_1, x_2)_i$ of i.i.d. images from the training distribution. Let S_i be the set of points in the plane defined by T_i at which the decision regions are evaluated. We define the reproducibility score:

$$R(\theta_1, \theta_2) = \mathbb{E}_{T_i \sim \mathcal{D}} \left[(|f(S_i, \theta_1) \cap f(S_i, \theta_2)|) / |S_i| \right] \quad (1)$$

where for notation simplicity we denote the set of class predictions within each decision region as $f(S_i, \theta) = \{(x, f(x; \theta))\}_{x \in S_i}$ for a model with parameters θ . Practically, we estimate the expectation in Eq. (1) by sampling 500 triplets and 2500 points in each truncated plane for a total of $1.25M$ forward passes. Simply put, this corresponds to the “intersection over union” score for two decision boundary plots.

This score can quantify reproducibility of decision regions across architectures, initializations, minibatch ordering, etc. In earlier work [3], variability of the decision boundaries is studied by examining the similarity of predictions at test points. In contrast, our method gives a much richer picture of the variance of the classification regions not just at the input points, but also in the regions around them and can be applied to both train and test data.

Measuring architecture-dependent bias We apply the reproducibility score to measure model similarity between different training runs with the same architecture and across different architectures. For each model pair, we compute the reproducibility score across 5 different training runs and 500 local decision regions, each containing 2,500 sampled points (6.25M total forward passes compared).

Figure 4 shows reproducibility scores for various architectures, and we see that quantitative results strongly reflect the trends observed in the decision regions of Figure 3. In particular, it becomes clear that

- The inductive biases of all the convolutional architectures are highly similar. Meanwhile, MLP Mixer, ViT and FC models have substantially different decision regions from convolutional models and from each other.
- Wider convolutional models appear to have higher reproducibility in their decision regions, with WideRN30 being both the widest and most reproducible model in this study.

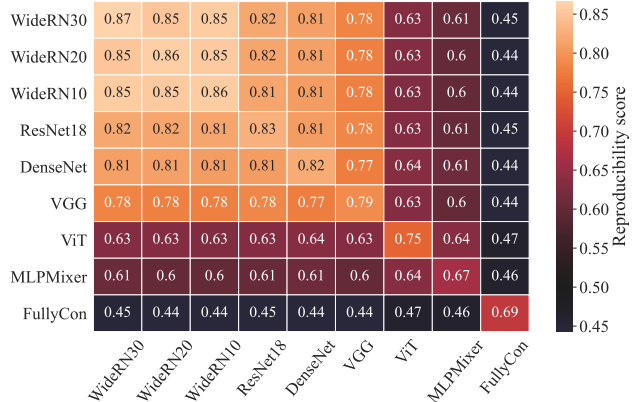


Figure 4. Reproducibility across several popular architectures.

- Skip connections have little impact on the shape of decision regions. ResNet (with residual connections across blocks), DenseNet (with many convolutional connections within blocks), and VGG (no skip connections) all share very similar decision regions. However it is worth noting that skip connection architectures achieve slightly higher reproducibility scores than the very wide VGG network.

3.3. Does distillation preserve decision boundaries?

Distillation [19] involves training a student model on the outputs of an already trained teacher model. Some believe that distillation does indeed convey information about the teacher’s decision boundary to the student [15], while others argue distillation improves generalization through other mechanisms [34]. We calculate the relative similarity of the student’s decision boundary to its teacher’s boundary and compare this to the similarity between teacher network and a network of the student’s architecture and initialization but trained in a standard fashion. Across the board, distilled students exhibit noticeably higher similarity to their teachers compared with their vanilla trained counterparts. In Figure 5, we see that *almost every* student-teacher combination has a higher reproducibility score than the same teacher compared to an identically initialized model trained without distillation.

3.4. The effect of the optimizer

In addition to the influence of initialization, data ordering, and architecture, the choice of optimizer/regularizer used during training can greatly impact the resulting model [13]. Thus, we study the effect of optimizer choice on the reproducibility of a network’s decision boundary. In Table 1, we can see that SAM [12] induces more reproducible decision boundaries than standard optimizers such as SGD and Adam. This observation suggests that SAM has a stronger regularization effect. However, more reg-

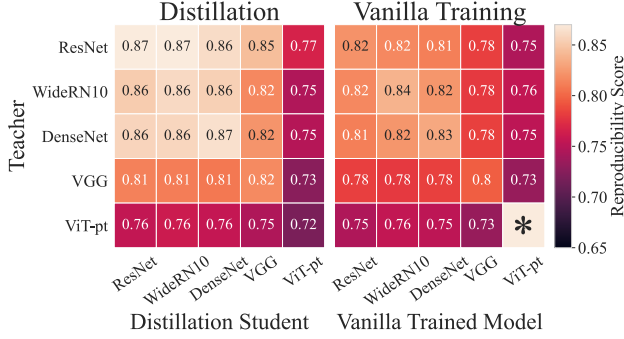


Figure 5. Differences in reproducibility comparing distilled model to vanilla trained model. *The reproducibility score is not applicable for this diagonal entry because we start from the same pre-trained model.

Reproducibility			
	Adam	SGD	SGD + SAM
ResNet-18	79.81%	83.74%	87.22%
VGG	81.19%	80.92%	84.21%
MLPMixer	67.80%	66.51%	68.06%
VIT	69.55%	75.13%	75.19%

Test Accuracy			
	Adam	SGD	SGD + SAM
ResNet-18	93.04	95.30	95.68
VGG	92.87	93.13	93.90
MLPMixer	82.22	82.04	82.18
VIT	70.89	75.49	74.72

Table 1. Reproducibility of different models when using different optimizers. SGD produces more reproducible decision boundaries relative to Adam, and SGD+SAM almost always consistently increase reproducibility of the model relative to SGD.

ularization doesn’t always mean better test accuracy. For example, for MLPMixer and ViT, using SAM does not always achieve the highest test accuracy but does achieve the highest reproducibility.

4. Double descent

In classical learning theory, it is thought that models with too few parameters (e.g., low width) generalize poorly because they are not expressive enough to fit the data, while models with too many parameters generalize poorly because of over-fitting. This is known as the Bias-Variance trade-off [14]. In contrast, the strong inductive bias of neural networks enables them to achieve good performance even with extremely large numbers of parameters. Belkin et al. [4] and Nakkiran et al. [27] have shown that under the right training conditions, we can see neural models oper-

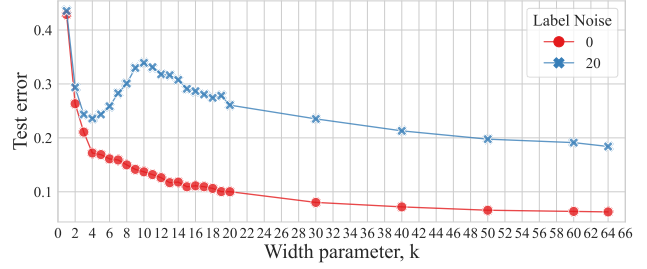


Figure 6. Test error curves with 0 and 20% label noise in training.

ating in both the classical and over-parameterized regimes. This is depicted in Fig. 6, which plots test error as a function of model width on CIFAR-10. We observe a classic U-shaped curve for widths less than 10 (the underparametrized regime). For models of width greater than 10, the test error fall asymptotically (overparametrized regime). This behaviour is referred to as “double descent” and discussed in generality in Belkin et al. [4]. Between the two regimes is a model that lives at the “interpolation threshold”; here, the model has too many parameters to benefit from classical simplicity bias, but too few parameters to be regularized by the inductive bias of the over-parameterized regime. Double descent has been studied rigorously for several simple and classical model families, including kernel methods, linear models, and simple MLPs [2, 6, 16, 26, 29, 30, 33]. Double descent is now well described for linear models and random feature networks in [1, 6, 10, 16]. In the classical regime, bias decreases with increased model complexity, while the variance increases at the same time, resulting in a U-shaped curve. Then, in the overparameterized regime, the variance decreases rapidly while bias remains low [28, 37]. In our studies above, we visualized the over-parameterized regime and saw that models become highly reproducible, with wide architectures producing nearly identical models across training runs. These visualizations captured the low-variance of the over-parameterized regime.

In this section, our goal is to gain insight into the model behaviors that emerge at the interpolation threshold, causing double descent. We observe closely what is happening at critical points (i.e., the transition between the under and overparameterized regimes), and how the class boundaries transition as we increase the capacity of the model class. We find that the behaviour of class boundaries aligns with the bias-variance decomposition findings of [28, 37], however the model instabilities that cause variance to spike in neural networks is manifested as a complex fragmentation of decision space that is not, to the best of our knowledge, described in the literature on classical models.

Experimental setup: We follow the experimental setting from Nakkiran et al. [27] to replicate the double descent phenomenon for ResNet-18 [17]. We increase model capac-

ity by varying the number of filters in the convolutional layers by a “width” parameter, k . Note that a standard ResNet-18 model has $k = 64$ and lives in the over-parameterized regime on CIFAR-10. We train models with cross-entropy loss and the Adam optimizer with learning-rate 0.0001 for 4000 epochs. This gentle but long training regiment ensures stability and convergence for the wide range of models needed for this study.

It was observed in [27] that label noise is important for creating easily observable double descent in realistic models. We train two sets of models, one with a clean training set and another with 20% label noise (uniform random incorrect class labels). In both cases, we use the standard (clean) test set. For noisy experiments, the same label errors are used across epochs and experiments. RandomCrop and RandomHorizontalFlip augmentations are used while training. We observe a pronounced double-descent when label noise is present. See Figure 6, which replicates the double-descent curve of Nakkiran et al. [27].

We focus on several important model widths: $k = 4$ is the local minimum of test error in the underparametrized regime, and $k = 10$ achieves peak error ($\approx$ interpolation threshold) beyond which the test error will continually fall. We refer the reader to Appendix D for training error plots showing the onset of interpolation near $k = 10$.

4.1. How do decision boundaries change as we cross the interpolation threshold?

In Figure 7, we plot decision boundaries for models trained with and without label noise and with varying capacities. As above, visualizations take place in the plane spanned by three data points. We present examples using two different methods for sampling – one with all three images from the same class and one with three different classes. The three images are drawn from the training set and are correctly labeled (even for the experiments involving label noise). Similar behaviours are observed for other randomly sampled images and with other combinations of classes. See Appendix D for additional examples.

As we move from left to right in the figure, model capacity sweeps from $k = 1$ (under-parameterized) to $k = 64$ (standard ResNet-18, which is over-parameterized). As the models become increasingly over-parameterized, the models are getting confident about their predictions, as seen by the intensity of the color. When models are trained with clean labels, the model fits all three points with high confidence by the time $k = 4$, and the decision boundaries change little beyond this point.

The mechanism behind the error spike in the double descent curve is captured by the visualizations using label noise. In this case, the under-fitting behavior of the classical regime is apparent at $k = 4$, as the model fits only 1 out of 3 points correctly, and confidence in predictions is

low. When we reached $k = 10$ (the interpolation threshold), the model fits most of the training data, including the three points in the visualization plane. As we cross this threshold, the decision regions become chaotic and fragmented. By the time we reach $k = 20$, the fragmentation is reduced and class boundaries become smooth as we enter the over-parameterized regime.

To refine our picture of double descent, we visualize the class boundaries at $k = 10$ for a range of different image triplets in Figure 8, both with and without label noise. We see that in the label noise case, where double descent is observed, there is a clear instability in the classification behavior at the interpolation threshold.

Let’s now see what happens to the decision boundaries around mislabeled images. Figure 9 shows decision boundaries around three points from the *Automobile* class, where one of the points is mislabeled in the training set. When $k = 10$, we see chaotic boundaries. The mislabeled points are assigned their (incorrect) dataset label, but they are just barely interpolated in the sense that they lie very near the decision boundary. For $k = 64$, the boundaries are seemingly regularized by inductive bias; the mislabeled points lie in the center of their respective regions, and boundaries are much more smooth.

Having observed the qualitative behaviour of correctly labeled and mislabeled points in models with and without label noise at various capacities, we ask the following questions:

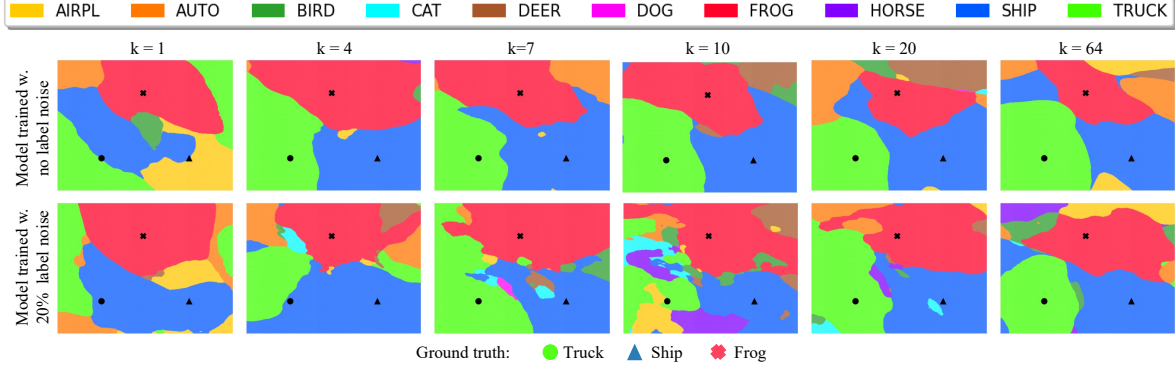
- Can quantitative methods validate that fragmentation behavior persists across multiple decision regions at the interpolation threshold and vanishes elsewhere?
- Is the fragmentation at the interpolation threshold indeed caused by model variance? In other words, do we observe different decision boundaries across training runs, or are the chaotic regions reproducible like the regions we observed in the over-parameterized regime?
- What is the mechanism for the decrease in test error in the wide model regime? Is it caused by shrinkage of the misclassified regions around mislabeled points, causing them to stop contaminating the test accuracy? Or is it merely caused by the vanishing of unnecessary fragmentation behavior for large k ?

In the subsequent sub-sections, we investigate these issues using quantitative measurements of decision regions.

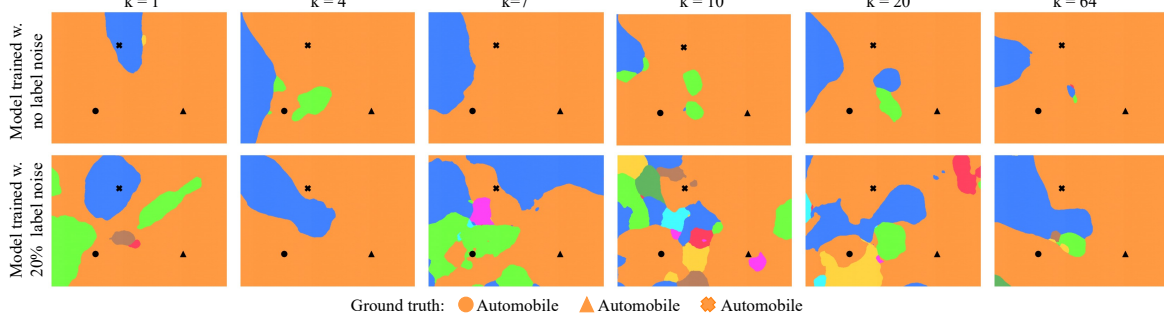
4.2. Quantifying fragmentation

We have observed that decision regions appear to become highly fragmented as we cross the interpolation threshold. To verify that our results are repeatable across many experiments and triples, we introduce the *fragmentation score*, which counts the number of connected class regions in the plane spanned by a triplet of images.

Let S_i be a local classification region spanned by a triplet



(a) All the points in the triple are from different classes, and are correctly labeled in the train set (even in the label noise case).



(b) All points in the triple are from the same class, Automobile, and are correctly labeled in the train set (even in the label noise case).

Figure 7. **Decision boundaries for models of varying width.** Label noise induces chaotic fragmentation of decision regions as we cross the threshold of interpolation ($k=10$), while very narrow and wide models remain smooth.

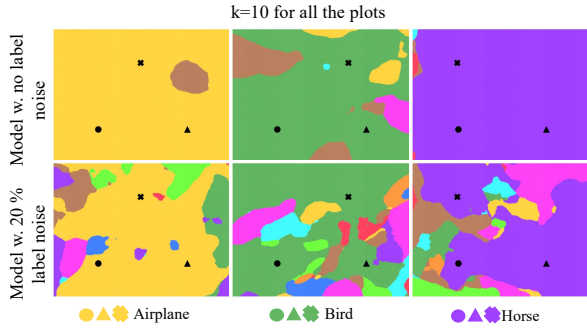


Figure 8. Decision boundaries of 3 correctly labeled points at $k=10$ on models with and without label noise.

T_i . We create a decomposition $S_i(\theta) = \cup_{j=1}^{n_i} P_j(\theta)$ where each $P_j(\theta)$ is a disjoint, maximal, path-connected component corresponding to a *single* predicted class label for the model with parameters θ . The fragmentation score $F(\theta, T_i)$ of model θ within the decision region defined by T_i is then the number of path-connected regions. The overall fragmentation score for a model is

$$F(\theta) = \mathbb{E}_{T_i \sim \mathcal{D}} F(\theta, T_i). \quad (2)$$

In practice, we compute the fragmentation score of a model using a watershed method to find connected regions in de-

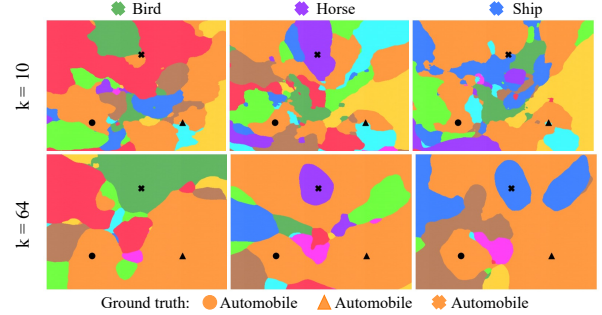


Figure 9. Decision boundaries with 1 mislabeled automobile and 2 correctly labeled automobiles. Each column represents a different image triplet. The mislabeled point is marked by x.

cision region spanned by the triplet and then by averaging such fragmentation counts over 1000 triplets.

Note that prior work [11] proposes a metric to understand class connectivity that requires solving a non-convex optimization problem to find an explicit path between any two given points. In contrast, our fragmentation score is scalable, does not require any backward passes to approximate the complexity of decision boundaries, and can be averaged over a large number of input triplets.

Fragmentation scores as a function of model width are depicted in Figure 10. With label noise, we see a sharp peak

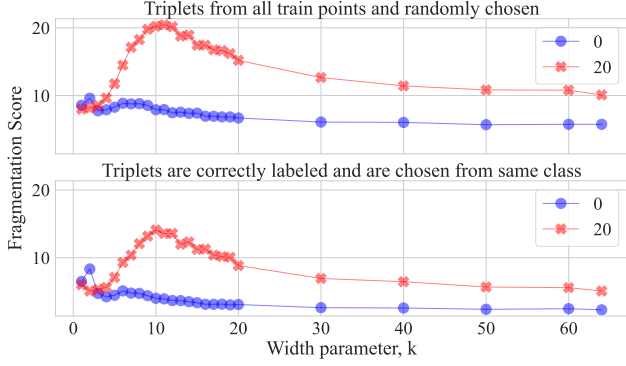


Figure 10. Fragmentation scores as a function of model width for models trained with and without label noise.

in fragmentation score as the model capacity crosses the interpolation threshold, confirming our observations from the visualization in the figures above. Interestingly, this highly sensitive analysis is also able to detect a peak (around $k = 7$) in the fragmentation score for models trained without label noise. The bottom part of Figure 10 quantifies the fragmentation trend for the decision regions spanned by triplets of the same class (like in Figure 8).

4.3. Quantifying class region stability



Figure 11. Reproducibility with respect to random initialization for models of different widths.

Theoretical studies of double descent predict that, for simple linear model classes, model variance spikes near the interpolation threshold as decision regions become highly unstable with respect to noise in the data sampling process. Using reproducibility scores, we observe that the fragmentation of neural decision boundaries at the interpolation threshold is associated with high variance and model instability. Figure 11 shows reproducibility scores across model capacities with and without label noise. We see that reproducibility across training runs is high in the under- and over-parameterized regimes, but breaks down at the interpolation threshold. Interestingly, our quantifications are sensitive enough to detect a dip in reproducibility even without label noise, although the variance introduced by this effect is not strong enough to cause double descent. Note that the

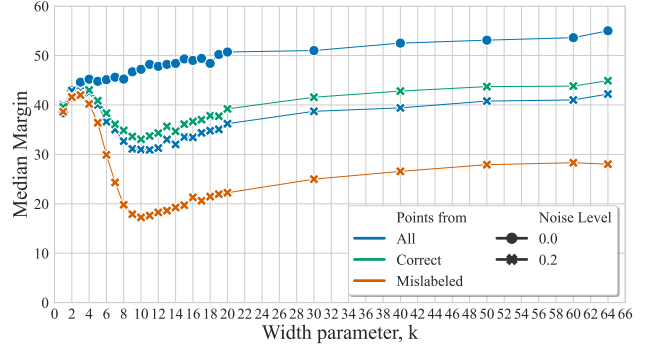


Figure 12. Median Margins - models with and without label noise. Y-axis reflects the average perturbation size needed to reach decision boundary in a random direction.

model variance in Figure 11 is caused by differences in random initialization. Classical convex learning theory studies variance with respect to random data sampling. We find that a similar curve is produced by freezing initialization and randomizing the sampling process (see Appendix D).

4.4. Why does label noise amplify double descent?

The dramatic effect of label noise near the interpolation threshold could be caused by two factors: (i) the necessary regions of incorrect class labels that must emerge around mislabeled points for the model to interpolate them, or (ii) instability in the class boundaries, resulting in oscillations that are not needed to interpolate the data. Quantitative evidence presented above suggests that (ii) is the predominant mechanism of double descent. The lower fragmentation scores in over-parameterized regime (where almost all mislabeled points are interpolated) compared to critical regime as seen in Figure 10 shows that the extra regions are not needed for interpolation.

To lend more strength to this conclusion, we investigate hypothesis (i) by measuring the “mean margin,” which we define to be the average distance between an image and the edge of its class region in a random direction. For each image, we approximate this value using a bisection search in 10 random directions. We compute the mean margin for 5000 data points and report the median for models with and without label noise in Figure 12.

Both with and without label noise, the margins are increasing for $k \geq 10$ (the over-parameterized regime). The interesting observation is, when we computed margins of only the mislabeled points, they go up too! The fact that test error descends, even as the regions around mislabeled points grow, lends further strength to the notion that double descent is predominantly driven by the “unnecessary” oscillations resulting from model instability, and not by the error bubbles around mislabeled points.

5. Conclusion

In this article, we use visualizations and quantitative methods to investigate model reproducibility, inductive bias, and double descent from an empirical/scientific perspective. These explorations reveal several interesting behaviors of neural models that we do not think have been previously observed. Curiously, the results of Section 3 indicate that different model families achieve low test error by different inductive strategies; While ResNet-18 and ViT make similar predictions on test data, there are dramatic differences in the decision boundaries they draw. Also, while our studies of double descent found that the model instability predicted for linear models is also observed for neural networks, we saw that this instability is manifested as the dramatic fragmentation of class regions. These oscillations in the model output are reminiscent of “Gibbs phenomenon,” and do not appear to be described in the theoretical literature.

6. Acknowledgements

This work was supported by the ONR MURI program, the Office of Naval Research, the National Science Foundation (DMS-1912866), and DARPA GARD (HR00112020007). Additional funding was provided by Capital One Bank and Kulkarni Summer Research Fellowship.

References

- [1] Ben Adlam and Jeffrey Pennington. Understanding double descent requires a fine-grained bias-variance decomposition. *arXiv preprint arXiv:2011.03321*, 2020. 5
- [2] Madhu S Advani, Andrew M Saxe, and Haim Sompolinsky. High-dimensional dynamics of generalization error in neural networks. *Neural Networks*, 132:428–446, 2020. 5
- [3] Anonymous. Decision boundary variability and generalization in neural networks. In *Submitted to The Tenth International Conference on Learning Representations*, 2022. under review. 4
- [4] Mikhail Belkin, Daniel Hsu, Siyuan Ma, and Soumik Mandal. Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences*, 116(32):15849–15854, 2019. 5
- [5] Pratik Chaudhari, Anna Choromanska, Stefano Soatto, Yann LeCun, Carlo Baldassi, Christian Borgs, Jennifer Chayes, Levent Sagun, and Riccardo Zecchina. Entropy-sgd: Biasing gradient descent into wide valleys. *Journal of Statistical Mechanics: Theory and Experiment*, 2019(12):124018, 2019. 1
- [6] Yehuda Dar, Vidya Muthukumar, and Richard G Baraniuk. A farewell to the bias-variance tradeoff? an overview of the theory of overparameterized machine learning. *arXiv preprint arXiv:2109.02355*, 2021. 5
- [7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 3
- [8] Laurent Dinh, Razvan Pascanu, Samy Bengio, and Yoshua Bengio. Sharp minima can generalize for deep nets. In *International Conference on Machine Learning*, pages 1019–1028. PMLR, 2017. 1
- [9] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 3
- [10] Stéphane d’Ascoli, Maria Refinetti, Giulio Biroli, and Florent Krzakala. Double trouble in double descent: Bias and variance (s) in the lazy regime. In *International Conference on Machine Learning*, pages 2280–2290. PMLR, 2020. 5
- [11] Alhussein Fawzi, Seyed-Mohsen Moosavi-Dezfooli, Pascal Frossard, and Stefano Soatto. Empirical study of the topology and geometry of deep networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3762–3770, 2018. 2, 7
- [12] Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. *arXiv preprint arXiv:2010.01412*, 2020. 3, 4
- [13] Jonas Geiping, Micah Goldblum, Phillip E Pope, Michael Moeller, and Tom Goldstein. Stochastic training is not necessary for generalization. *arXiv preprint arXiv:2109.14119*, 2021. 4
- [14] Stuart Geman, Elie Bienenstock, and René Doursat. Neural networks and the bias/variance dilemma. *Neural computation*, 4(1):1–58, 1992. 5
- [15] Micah Goldblum, Liam Fowl, Soheil Feizi, and Tom Goldstein. Adversarially robust distillation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 3996–4003, 2020. 4
- [16] Trevor Hastie, Andrea Montanari, Saharon Rosset, and Ryan J Tibshirani. Surprises in high-dimensional ridgeless least squares interpolation. *arXiv preprint arXiv:1903.08560*, 2019. 5

- [17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 3, 5
- [18] Warren He, Bo Li, and Dawn Song. Decision boundary analysis of adversarial examples. In *International Conference on Learning Representations*, 2018. 2
- [19] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 4
- [20] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017. 3
- [21] Hamid Karimi, Tyler Derr, and Jiliang Tang. Characterizing the decision boundary of deep neural networks. *arXiv preprint arXiv:1912.11460*, 2019. 2
- [22] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 3
- [23] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 2
- [24] Michel Ledoux. *The concentration of measure phenomenon*. Number 89. American Mathematical Soc., 2001. 2, 1
- [25] Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. *arXiv preprint arXiv:1712.09913*, 2017. 1
- [26] Vidya Muthukumar, Kailas Vodrahalli, Vignesh Subramanian, and Anant Sahai. Harmless interpolation of noisy data in regression. *IEEE Journal on Selected Areas in Information Theory*, 1(1):67–83, 2020. 5
- [27] Preetum Nakkiran, Gal Kaplun, Yamini Bansal, Tristan Yang, Boaz Barak, and Ilya Sutskever. Deep double descent: Where bigger models and more data hurt. *arXiv preprint arXiv:1912.02292*, 2019. 5, 6
- [28] Brady Neal, Sarthak Mittal, Aristide Baratin, Vinayak Tantia, Matthew Scicluna, Simon Lacoste-Julien, and Ioannis Mitliagkas. A modern take on the bias-variance tradeoff in neural networks. *arXiv preprint arXiv:1810.08591*, 2018. 5
- [29] Manfred Opper. Statistical mechanics of learning: Generalization. *The handbook of brain theory and neural networks*, pages 922–925, 1995. 5
- [30] Manfred Opper. Learning to generalize. *Frontiers of Life*, 3(part 2):763–775, 2001. 5
- [31] Ali Shafahi, W Ronny Huang, Christoph Studer, Soheil Feizi, and Tom Goldstein. Are adversarial examples inevitable? *arXiv preprint arXiv:1809.02104*, 2018. 2, 1
- [32] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 3
- [33] Stefano Spigler, Mario Geiger, Stéphane d’Ascoli, Levent Sagun, Giulio Biroli, and Matthieu Wyart. A jamming transition from under-to over-parametrization affects loss landscape and generalization. *arXiv preprint arXiv:1810.09665*, 2018. 5
- [34] Samuel Stanton, Pavel Izmailov, Polina Kirichenko, Alexander A Alemi, and Andrew Gordon Wilson. Does knowledge distillation really work? *arXiv preprint arXiv:2106.05945*, 2021. 4
- [35] Ilya Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, et al. Mlp-mixer: An all-mlp architecture for vision. *arXiv preprint arXiv:2105.01601*, 2021. 3
- [36] Ross Wightman. Pytorch image models. <https://github.com/rwightman/pytorch-image-models>, 2019. 3
- [37] Zitong Yang, Yaodong Yu, Chong You, Jacob Steinhardt, and Yi Ma. Rethinking bias-variance trade-off for generalization of neural networks. In *International Conference on Machine Learning*, pages 10767–10777. PMLR, 2020. 5
- [38] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016. 3
- [39] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, 64(3):107–115, 2021. 3
- [40] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017. 2

Can You Learn the Same Model Twice? Investigating Reproducibility and Double Descent from the Decision Boundary Perspective

Supplementary Material

A. Proof of Lemma 2.1

For clarity, we restate the lemma here.

Lemma 2.1 *Let $f : [0, 1]^n \rightarrow [0, 1]$ be a neural network satisfying $|f(x) - f(y)| \leq \frac{L}{\sqrt{n}} \|x - y\|$. Let $\bar{f}$ denote the median value of f on the unit hypercube. Then, for an image $x \in [0, 1]^n$ of uniform random pixels, we have $|f(x) - \bar{f}| \leq t$ with probability at least*

$$1 - \frac{Le^{-2\pi t^2/L^2}}{\pi t \sqrt{n}}.$$

Consider a set $\mathcal{A} \subset [0, 1]^n$, and let d denote the ℓ_2 distance metric. We define the ϵ -expansion of the set $\mathcal{A}$ as $\mathcal{A}(\epsilon) = \{x \in [0, 1]^n \mid d(x, \mathcal{A}) \leq \epsilon\}$. In plain words, $\mathcal{A}(\epsilon)$ is the set of all points lying within ϵ units of the set $\mathcal{A}$.

Our proof will make use of the isoperimetric inequality first presented by Ledoux [24]. We use the following variant with tighter constants proved by Shafahi et al. in [31].

Lemma A.1 (Isoperimetric inequality on the unit cube)

Consider a measurable subset of the cube $\mathcal{A} \subset [0, 1]^n$, and a 2-norm distance metric $d(x, y) = \|x - y\|_2$. Let $\Phi(z) = (2\pi)^{-\frac{1}{2}} \int_{-\infty}^z e^{-t^2/2} dt$, and let α be the scalar that satisfies $\Phi(\alpha) = \text{vol}[\mathcal{A}]$. Then

$$\text{vol}[\mathcal{A}(\epsilon)] \geq \Phi\left(\alpha + \epsilon\sqrt{2\pi}\right). \quad (3)$$

In particular, if $\text{vol}(\mathcal{A}) \geq 1/2$, then we simply have

$$\text{vol}[\mathcal{A}(\epsilon)] \geq 1 - \frac{e^{-2\pi\epsilon^2}}{2\pi\epsilon}. \quad (4)$$

To prove Lemma 2.1, we start by choosing $\mathcal{A} = \{x \mid f(x) \leq \bar{f}\}$. Now, consider any $x \in \mathcal{A}\left(t\frac{\sqrt{n}}{L}\right)$. From the Lipschitz bound on f we have

$$|f(x) - f(y)| \leq \frac{L}{\sqrt{n}} \|x - y\|,$$

for any y . If we choose $y = \arg \min_{z \in \mathcal{A}} \|x - z\|$ to be the closest point to x in the set $\mathcal{A}$, we have that $\|x - y\| \leq t\frac{\sqrt{n}}{L}$, and so

$$|f(x) - f(y)| \leq t.$$

But $f(y) \leq \bar{f}$ because $y \in \mathcal{A}$. From this, we see that for any choice of $x \in \mathcal{A}\left(t\frac{\sqrt{n}}{L}\right)$ we have

$$f(x) - \bar{f} \leq t. \quad (5)$$

Recall that $\bar{f}$ is the median value of f on the unit cube, and so we have that

$$\text{vol}[\mathcal{A}] \geq \frac{1}{2}.$$

We can then apply Lemma A.1 with $\epsilon = t\frac{\sqrt{n}}{L}$, and we see that

$$\text{vol}\left[\mathcal{A}\left(t\frac{\sqrt{n}}{L}\right)\right] \geq 1 - \frac{Le^{-2\pi t^2 n/L^2}}{2\pi t \sqrt{n}}.$$

We conclude that a randomly chosen $x \in [0, 1]^n$ will lie in $\mathcal{A}\left(t\frac{\sqrt{n}}{L}\right)$, and therefore satisfy (5) with probability at least $1 - \frac{Le^{-2\pi t^2 n/L^2}}{2\pi t \sqrt{n}}$.

An analogous argument with $\mathcal{A} = \{x \mid f(x) \geq \bar{f}\}$ shows that a randomly chosen $x \in [0, 1]^n$ will satisfy

$$\bar{f} - f(x) \leq t. \quad (6)$$

with the same probability. Applying a union bound, we see that a randomly chosen x will satisfy (5) and (6) simultaneously with probability at least $1 - \frac{Le^{-\pi t^2 n/L^2}}{\pi t \sqrt{n}}$.

B. Decision regions

Off-manifold decision regions We present a few off-manifold decision boundaries in this section. In Fig. 13, we show decision regions of multiple off manifold images where all the pixels are uniformly sampled in the image space. Each row is a model, and each column is a randomly sampled triplet. We observe that the decision regions assigned to such off-manifold images are quite uniform for a given model. For example, in DenseNet, all such images are assigned to *Bird* class, while in ViT, they are assigned to *Frog* or *Automobile*. In Fig. 14, we show decision regions for a multiple triplets of shuffled images. (Expanded version of Fig.2). Even in this type of off-manifold images, we see a similar pattern that the models are assigning the samplings to a certain set of classes. This emphasises that the decision regions are more structured close to the image manifold and are rather uniform farther away from the manifold.

C. Additional Reproducibility results

With and without Mixup in training In order to understand how having mixup in the training affects the decision boundaries, we examined 2 cases, ResNet18 and Vision Transformer. In Fig. 15, we show 5 randomly sampled triplets and their decision regions produced by ResNet18



Figure 13. Decision regions when all the images are uniformly sampled. Each row corresponds to a model, while each column is a new sampling of the triplet

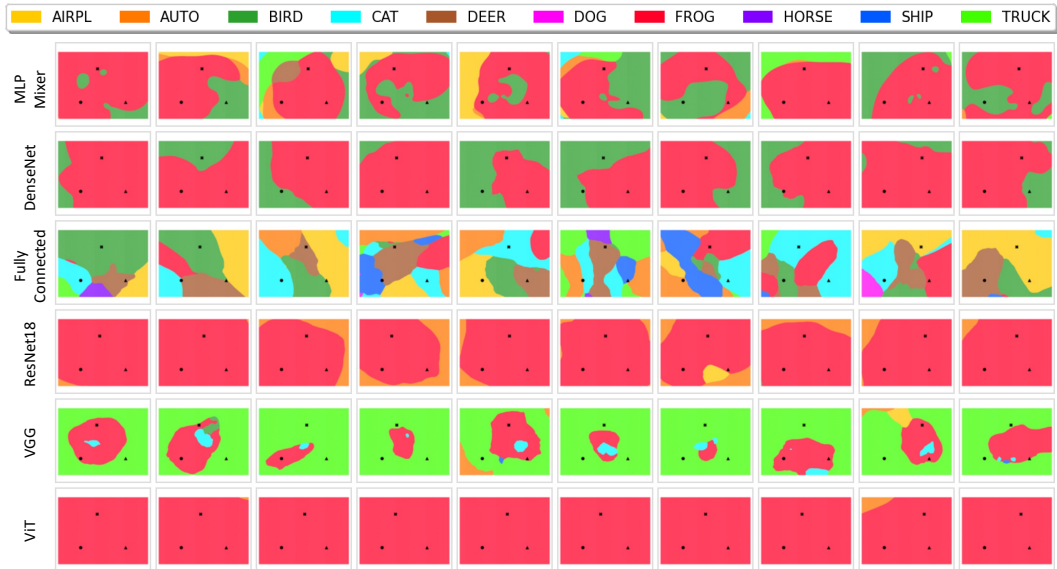


Figure 14. Decision regions when the pixels are randomly shuffled. Each row corresponds to a model, while each column is a new sampling of the triplet. Extended version of Fig 2.

trained with and without mixup. We can see there is a slight difference, but not quite significant. We quantified how “similar” the decision surfaces are with reproducibility score introduced in Section 3.2. The score for Resnet18 is 0.774, and for ViT is 0.808.

D. Additional Double Descent results

Additional error plots In Fig. 6, we have seen how the test errors change as we progressively increase the model capacity. Figure 16 shows how training errors change in addition to test errors. We can see that the train error reaches 0 at much higher k with label noise than without. In model without label noise, the interpolation begins at $k = 10$

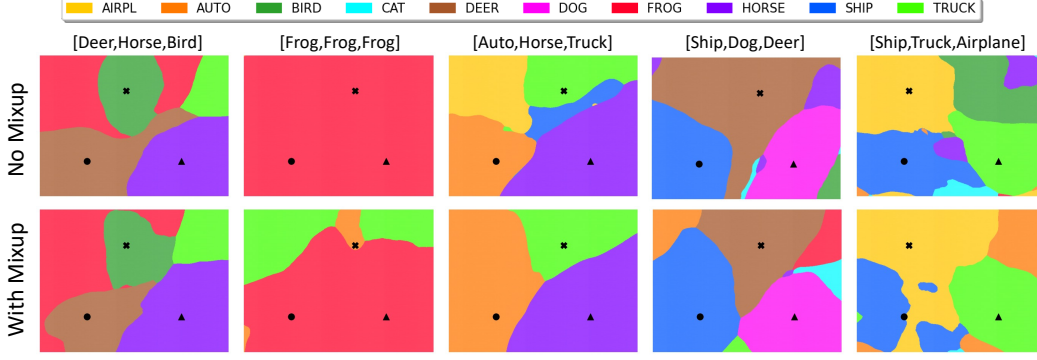


Figure 15. We present decision regions for random triplets sampled from the training set for ResNet18. We see the decision regions are almost same with and without mixup.

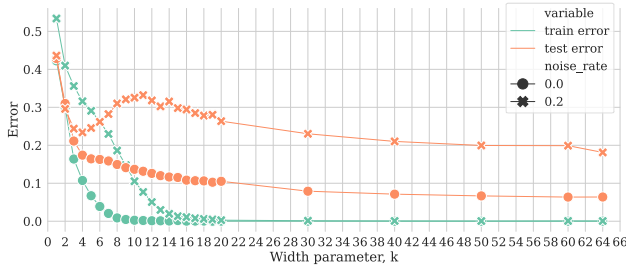


Figure 16. In this figure we show the train and test errors with and without label noise.

which is the true interpolation threshold when there is no label noise. We further examine how correctly labeled and mislabeled points are behaving in Figure 17. The green lines represent the overall train error, while orange shows the error on correctly labeled points. The mislabeled points are shown in grey, and the error is computed as incorrect predictions with respect to assigned class. We see that till $k = 4$ the mislabeled points are not fit to their assigned class which partially explains the low test error of test data. However at $k = 10$ most of the correctly labeled points are fit while some of the mislabeled points are still not fit to their assigned class. This trend diverges from what is seen in simple model families where the second peak of test error coincides with the model capacity with 0 training error. This shows that double descent in more complicated in neural-network architectures than what is seen in simple linear models.

Reproducibility scores from random data sampling In Figure 11, we have seen how decision boundaries change when we compare two runs of the same model architecture with different initializations. In Figure 18, we show how the ordering of the data changes the decision boundaries. We see that the reproducibility across training runs is high in the under- and over-parametrized regimes, but it drops

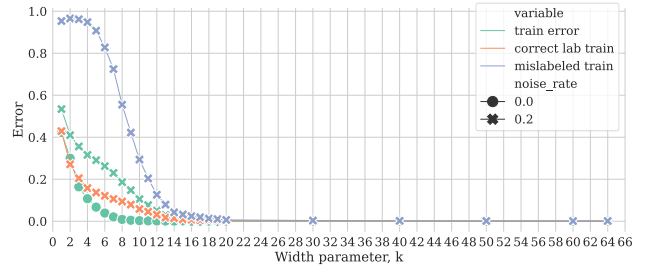


Figure 17. In this figure, we show the train data errors for with and without label noise cases in green color. We also investigate how the errors are changing for correctly labeled points (orange curve) and in mislabeled points (grey curve).

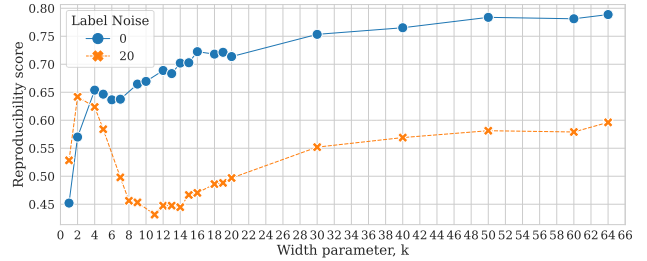
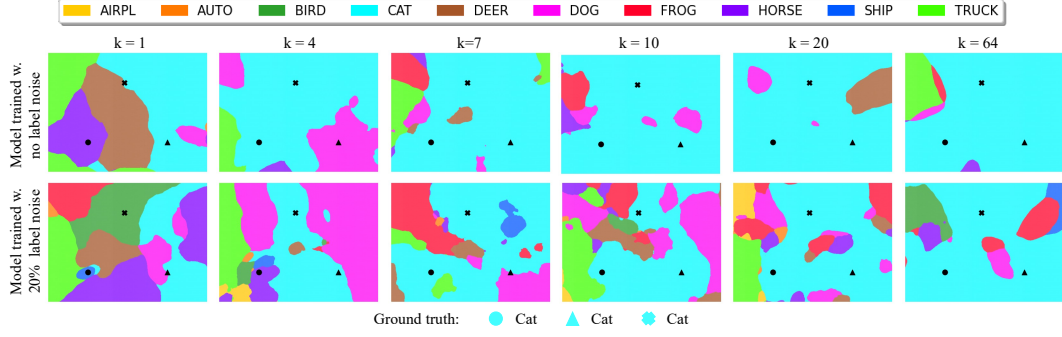


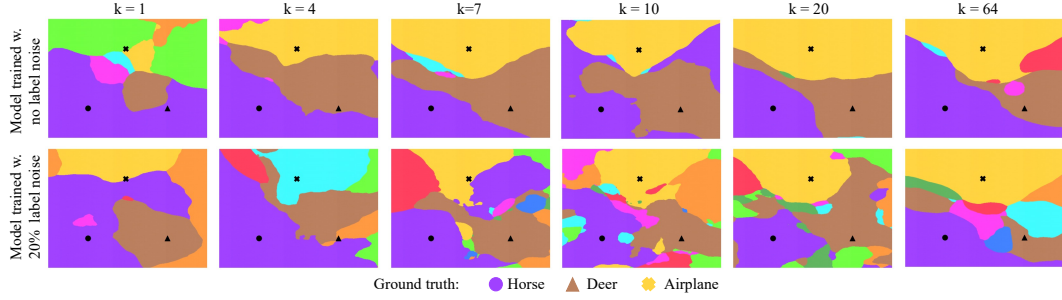
Figure 18. Reproducibility with respect to random data samplings for models of different widths

drastically closer to the interpolation threshold. This is the exact same behaviour observed in Figure 11. This shows that $k = 10$ is a quite unstable with respect to different types of variations in the model training.

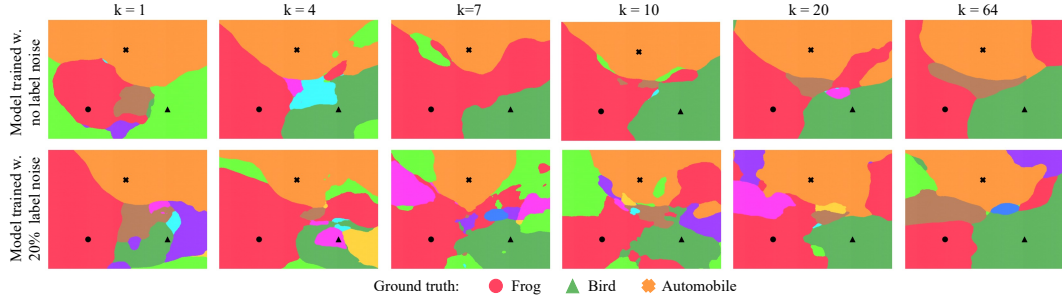
Additional plots across varying model capacities and noise In Figure 19, we show how the decision regions change with and without label noise and with varying model capacities across different samplings of triplets.



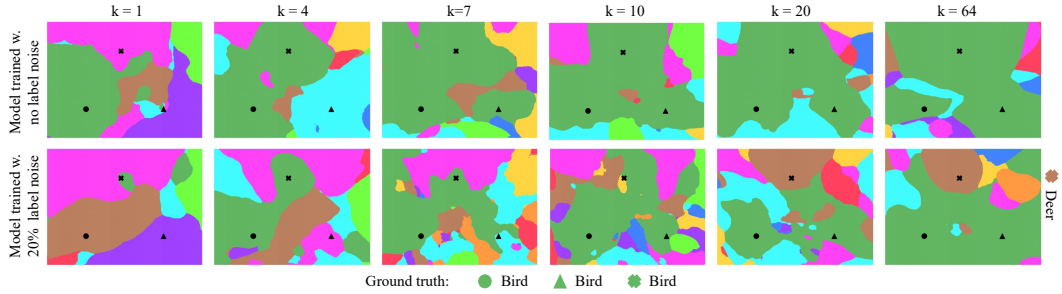
(a) All points are from same class (Cat), and are correctly labeled even in label noise case.



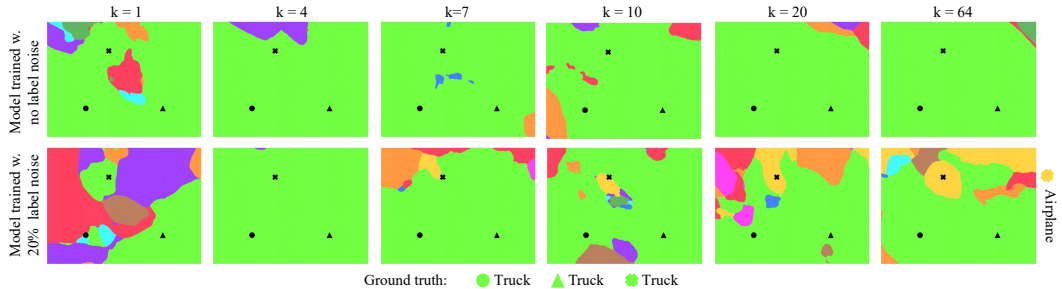
(b) The images are sampled from 3 different classes and are correctly labeled.



(c) The images are sampled from 3 different classes and are correctly labeled. Additional case



(d) In this triplet, when there is no label noise, all three belonged to Bird class. But in the label noise case, the third point is mislabeled as Deer.



(e) In this triplet, when there is no label noise, all three belonged to Truck class. But in the label noise case, the third point is mislabeled as Airplane.

Figure 19. **Decision boundaries for models of varying width.** We show additional decision surfaces with different types of triplets here.