Lazy vs hasty: linearization in deep networks impacts learning schedule based on example difficulty

Thomas George [1] **Guillaume Lajoie** [1] **Aristide Baratin** [2]

Abstract

A recent line of work has identified a so-called 'lazy regime' where a deep network can be well approximated by its linearization around initialization throughout training. Here we investigate the comparative effect of the lazy (linear) and feature learning (non-linear) regimes on subgroups of examples based on their difficulty. Specifically, we show that easier examples are given more weight in feature learning mode, resulting in faster training compared to more difficult ones. We illustrate this phenomenon across different ways to quantify example difficulty, including c-score, label noise, and in the presence of spurious correlations.

1. Introduction

Understanding the performance of deep learning algorithms has been the subject of intense research efforts in the past few years, driven in part by observed phenomena that seem to defy conventional statistical wisdom (Neyshabur et al., 2015; Zhang et al., 2017; Belkin et al., 2019). Notably, many such phenomena have been analyzed rigorously in simpler contexts of high dimensional linear or random feature models (Bartlett et al., 2021), which shed a new light on the crucial role of overparametrization in the performance of such systems. These results also apply to lazy training (Chizat et al., 2018), a specific regime where a deep network can be well approximated by its linearization at initialization, characterized by the neural tangent kernel (NTK) (Jacot et al., 2018; Du et al., 2019; Allen-Zhu et al., 2019).

Actual deep learning algorithms have been shown to depart from these analytical linear models (Chizat et al., 2018). However, they remain a convenient tool for analysis of deep learning mechanisms, with several recent works studying the time varying tangent kernel (Kopitkov & Indelman, 2020; Baratin et al., 2021; Paccolat et al., 2021; Fort et al., 2020), showing how it specializes to the task by stretching in a few task-relevant directions, while shrinking in less useful directions. It was argued in (Baratin et al., 2021) that such a mechanism acts as implicit regularizer, by allowing large models to adapt their capacity to the task.

Here we continue this line of work, by investigating the comparative effect of the lazy and feature learning regimes on the training dynamics of various groups of examples of increasing difficulty. We highlight what we believe to be an important aspect of the non-linear dynamics: to hasten towards learning simple patterns and easy examples, thereby inducing some form of curriculum or sequentialization of learning that is less stressed in the lazy (linear) regime. In the context where easy examples are the ones that display correlations to some spurious features (Sagawa et al., 2020b), this observation suggests an enhanced sensitivity to spurious correlations in the feature learning regime.

2. Preliminaries

We consider neural networks $f_{\boldsymbol{\theta}}$ parametrized by $\boldsymbol{\theta} \in \mathbb{R}^p$ (i.e. weights and biases for all layers), and trained by minimizing some task-dependent loss function $\ell(\boldsymbol{\theta}) := \sum_{i=1}^n \ell_i(f_{\boldsymbol{\theta}}(\mathbf{x}_i))$ computed on a training dataset, using variants of gradient descent,

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \eta \nabla_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}^{(t)}), \tag{1}$$

with some random initialization $\boldsymbol{\theta}^0$ and a chosen learning rate $\eta > 0$. A Taylor expansion and the chain rule give the corresponding updates $f^{(t)} := f_{\boldsymbol{\theta}^{(t)}}$ for any network output, at first order in the learning rate,

$$f^{(t+1)}(\mathbf{x}) \simeq f^{(t)}(\mathbf{x}) - \eta \sum_{i=1}^{n} K^{(t)}(\mathbf{x}, \mathbf{x}_i) \nabla \ell_i \tag{2}$$

which depend on the time-varying tangent kernel $K^{(t)}(\mathbf{x}, \mathbf{x}') := \nabla_{\boldsymbol{\theta}} f^{(t)}(\mathbf{x})^\top \nabla_{\boldsymbol{\theta}} f^{(t)}(\mathbf{x}')$. The *lazy regime* is one where this kernel remains nearly constant throughout training. Training the network in this regime thus corresponds to training the linear predictor defined by

$$\bar{f}_{\boldsymbol{\theta}}(\mathbf{x}) := f_{\boldsymbol{\theta}_0}(\mathbf{x}) + (\boldsymbol{\theta} - \boldsymbol{\theta}_0)^\top \nabla_{\boldsymbol{\theta}} f_{\boldsymbol{\theta}_0}(\mathbf{x}). \tag{3}$$

[1]Mila - Université de Montréal [2]SAIT AI Lab, Montréal. Correspondence to: Thomas George <tfjgeorge@gmail.com>.

Published at the ICML 2022 Workshop on Spurious Correlations, Invariance, and Stability. Baltimore, Maryland, USA. Copyright 2022 by the author(s).

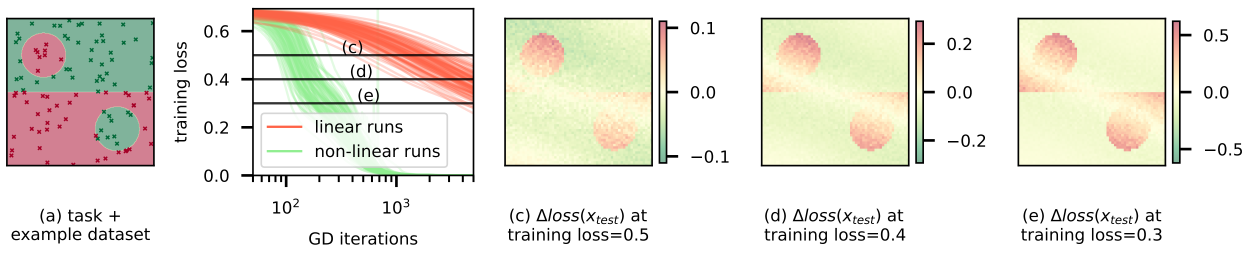

Figure 1. 100 randomly initialized runs of a 4 layers MLP trained on the yin-yang dataset **(a)** using GD in both a non-linear ($\alpha = 1$) and linearized ($\alpha = 100$) setting. The training losses **(b)** show a speed-up in the non-linear regime: in order to compare both regimes at equal progress, we normalize by comparing models extracted at equal training loss thresholds **(c)**, **(d)** and **(e)**. We observe the differences $\Delta \text{loss}\left(x_{\text{test}}\right) = \text{loss} f_{\text{non-linear}}\left(x_{\text{test}}\right) - \text{loss} f_{\text{linear}}\left(x_{\text{test}}\right)$ for test points paving the 2d square $\left[-1, 1\right]^2$ using a color scale. These differences are not uniformly spread across examples: instead they suggest a comparative bias of the non-linear regime towards correctly classifying easy examples (large areas of the same class), whereas difficult examples (e.g. the small disks) are boosted in the linear regime.

In our experiments, following (Chizat et al., 2018), we modulate the level of "non-linearity" during training with a scalar parameter $\alpha \geq 1$, by replacing our prediction $f_{\boldsymbol{\theta}}$ by

$$f_{\boldsymbol{\theta}}^{\alpha}\left(\mathbf{x}\right) := \alpha\left(f_{\boldsymbol{\theta}}\left(\mathbf{x}\right) - f_{\boldsymbol{\theta}_0}\left(\mathbf{x}\right)\right) \tag{4}$$

and by rescaling the learning rate as $\eta_\alpha = \eta/\alpha^2$. In this setup, gradient descent steps in parameter space are rescaled by $1/\alpha$ while steps in function space (up to first order) are in $O(1)$ in α. α can also be viewed as controlling the *level of feature adaptativity*, where large values of α result in linear training where features are not learned. Linearity is assessed empirically using various metrics (See Appendix A), including *sign similarity* that counts the proportion of ReLUs that keep the same sign since initialization; and *matrix alignment* of the tangent kernel and the last non-softmax layer representation with their initial values. We will also experiment with $\alpha < 1$ below, which enhances adaptivity compared to the standard regime ($\alpha = 1$).

3. A motivating example on a toy dataset

We first explore the effect of modulating the training regime for a binary classification task on a toy dataset with 2d inputs for which we can get a visual intuition. For 100 independent runs, we generate 100 training examples uniformly on $\left[-1, 1\right]^2$ from the yin-yang dataset (fig. 1.a). We use a fully-connected network with 4 layers and ReLU activations. We perform 2 training runs for $\alpha = 1$ and $\alpha = 100$ using (full-batch) gradient descent with learning rate 0.01.

Global training speed-up and normalization After the very first few iterations, we observe a speed-up in training progress of the non-linear regime (fig. 1.b). This is consistent with previously reported numerical experiments on the lazy training regime (Chizat et al., 2018, section 3). This raises the question of whether this acceleration comes from a global scaling in all directions, or if it prioritizes certain particular groups of examples. We answer this question by comparing the training dynamics at equal progress: we counteract the difference in training speed by normalizing by the mean training loss, and we compare the linear and non-linear regimes after both cross common thresholds ((c), (d) and (e) horizontal lines in fig. 1.b).

Comparing linear and non-linear regimes At every threshold value, we compute the predictions on test examples uniformly paving the 2d square. We compare the loss of both regimes on individual examples by plotting the difference in loss values $\Delta \text{loss}\left(x_{\text{test}}\right) = \text{loss} f_{\text{non-linear}}\left(x_{\text{test}}\right) - \text{loss} f_{\text{linear}}\left(x_{\text{test}}\right)$ (fig. 1.c, 1.d, 1.e). Red (resp. green) areas indicate a lower test loss for the linearized (resp. non-linear) model. Remarkably, the resulting picture is not uniform: these plots suggest that compared to the linear regime, the non-linear training dynamics speeds up for specific groups of examples (the large top-right and bottom-left areas) at the expenses on examples in more intricate areas (both the disks and the area between the disks and the horizontal boundary).

4. Hastening easy examples

We now experiment with deeper convolutional networks on CIFAR10, in two setups where the training examples are split into groups of varying difficulty. For the results shown below, we trained a ResNet18 with SGD with learning rate 0.01, momentum 0.9 and bach size 125.

C-scores Here we quantify example difficulty using consistency scores (C-scores) (Jiang et al., 2021). Intuitively, examples with high C-scores are more likely to be well-classified by models trained on subsets of the dataset that do not contain them, which can be interpreted as the fact that they are representative of a large group of examples in

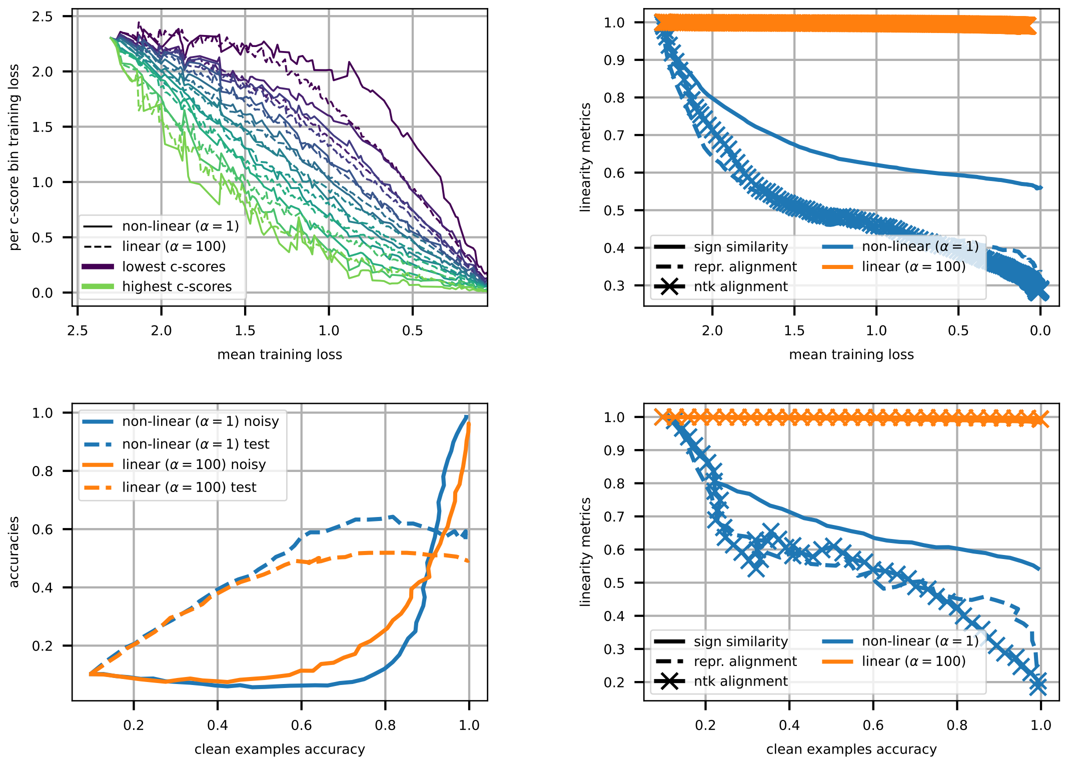

Figure 2. **(Top left)** We compute the training accuracies separately on 10 subgroups of CIFAR10 examples ranked by their C-scores. Training progress is normalized by the mean training loss on the x-axis. Unsurprisingly, in both regimes examples with high C-scores are learned faster. Remarkably, this ranking is more pronounced in the non-linear regime as we can observe by comparing dashed and solid lines of the same color. This suggests an increased tiering between groups of varying difficulty. **(bottom left)** CIFAR10 with 15% label noise. At equal progress (measured by equal clean examples accuracy), the non-linear regime prioritizes learning clean examples and nearly ignores noisy examples compared to the linear regime. Simultaneously, the non-linear test accuracy reaches a higher peak. **(right)** On the same training run, as a sanity check we observe that the $\alpha = 100$ training run actually stays in the linear regime throughout since all metrics stay close to 1, whereas in the $\alpha = 1$ run, the NTK and representation kernels rotate, and a large part of ReLU signs are flipped.

the dataset. Formally, for each (input, label) pair (x, y) in a dataset $\mathcal{D}$, we define its empirical consistency profile as the proportion of models that correctly classify (x, y) when trained on a size n subset of $\mathcal{D}$ that does not include (x, y):

$$\hat{C}_{\mathcal{D},n}(x, y) = \mathbb{E}_{D \overset{n}{\sim} \mathcal{D}} \left[\mathbb{P}\left(f\left(x; D \setminus \{(x, y)\}\right) = y \right) \right] \quad (5)$$

We obtain a scalar C-score by averaging the consistency profile over n. The results are shown in fig.2 (top left). Unsurprisingly, we find that examples with high C-scores are learned faster during training than examples with low C-scores in both regimes. Remarkably, this effect is amplified in the non-linear regime, as we can see by comparing e.g. the top (resp. bottom) decile in light green (resp dark blue). This is an instance of an acceleration of the non-linear regime in the direction of easy examples.

Label noise. Here 15% of the training examples are assigned a wrong (random) label. We compute the loss and accuracy independently on the regular examples with their true label, and the noisy examples whose label is flipped. In parallel, we train a copy of the initial model in the linearized regime with $\alpha = 100$. The results in fig.2 (bottom left) show that noisy examples are essentially ignored in the non-linear regime compared to the linear regime.

5. Spurious correlations

We now examine setups where easy examples are those with correlations between labels and some spurious feature (Sagawa et al., 2020b). We experiment with **Celeb A** (Liu et al., 2015) and **Waterbirds** (Wah et al., 2011) datasets. Our task with CelebA is to classify pictures based on whether the person is blond or not. The attribute ”the person is a woman” is spuriously correlated with the attribute ”the person is blond”, since the attribute ”blond” is overrepresented amongst women (24% are blond) compared to men (2% are blond). Our task with Waterbirds is to distinguish land birds from water birds (Sagawa et al., 2020a),

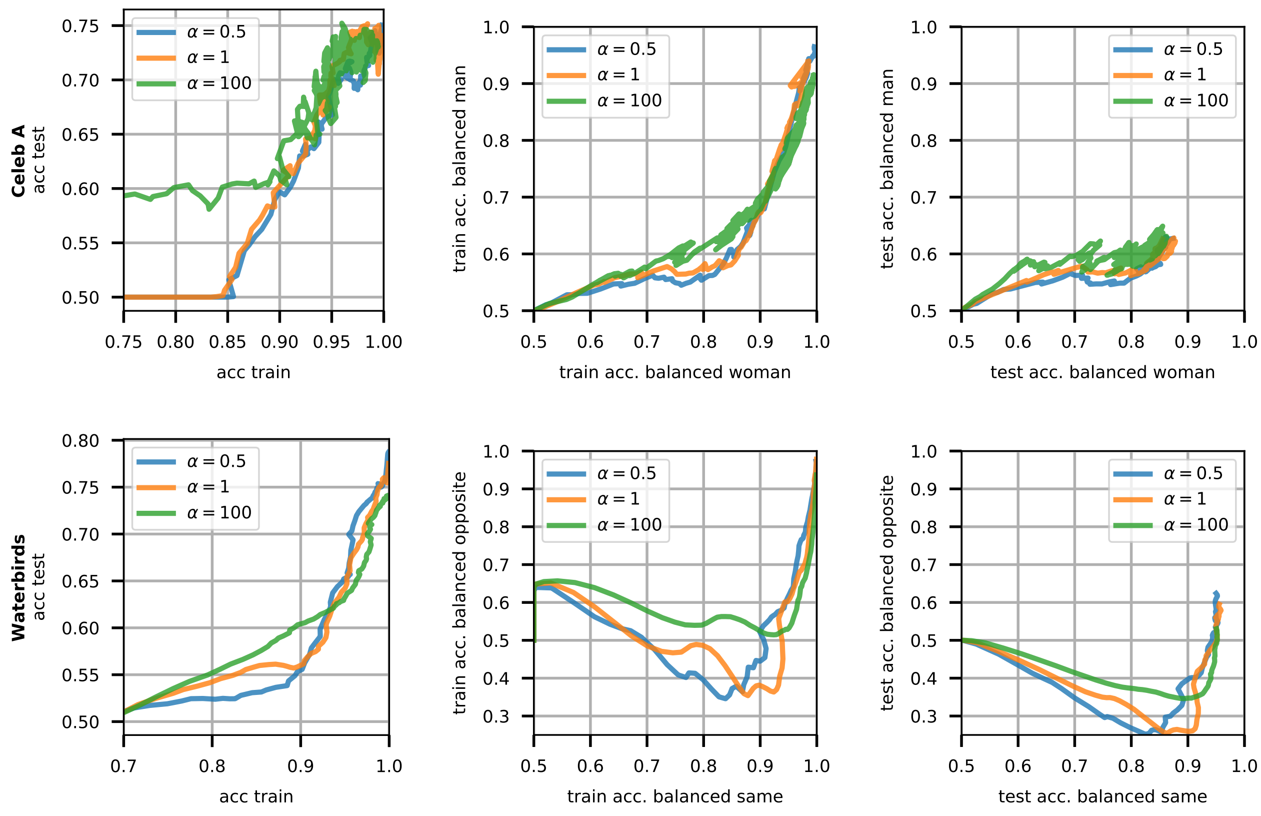

Figure 3. On Celeb A **(top row)**, as training progresses we observe in a first phase that **(middle)** the training accuracy on the balanced dataset for men increases faster in the linear regime compared to the training set accuracy amongst women, and **(right)** this translates to the balanced test dataset, showing that in this first part the linear regime avoids learning the spurious correlation between blond hair and woman gender, thus gets better generalization **(left)** In parallel, the test accuracy on the whole (unbalanced) dataset is better for the linear run $\alpha = 100$ during this first phase, suggesting that in this setup, a more linear run incurs more robustness, at least at the beginning of training. **(bottom row)** On Waterbirds, we also observe the same hierarchy between the linear $\alpha = 100$ run and other runs, in the first training phase. The linear regime is less prone to learning the spurious correlation since the accuracy for the *opposite* dataset (land birds on water background and water birds on land background) stays higher while training progresses in the first phase (**(middle)** and **(right)**).

whose labels often display spurious correlations with the background (e.g, lakes for waterbirds, forests for landbirds).

Looking at fig. 3 left, in both setups we identify 2 phases: in the first phase the test accuracy is higher for the linear $\alpha = 100$ run than for other runs. In the second phase all 3 runs seem to converge, with a slight advantage for non-linear runs in Waterbirds. Taking a closer look at the first phase in fig. 3 middle and right, we understand this difference in test accuracy in light of spurious and non-spurious features: in the non-linear regime, the training dynamics learn the majority examples faster, at the cost of being more prone to spurious correlations. This can be seen both on the balanced training set (fig. 3 middle) and the balanced test set (fig. 3 right).

6. Conclusion

Our work sheds light on qualitative differences between linear models amenable to analytical treatment (e.g. through the NTK formalism) and the standard deep learning training regime. Specifically, we showed certain subgroups of examples are given more weight in linear training mode than in non-linear (deep learning) training. Through experiments on both toy and image datasets, we exposed a unified picture on these groups of examples: in the spurious correlation experiments, our results suggest that the linear training dynamics is less prone to overfitting on the spurious features, whereas in the noisy example case, the non-linear training dynamics is more robust to noise which results in a higher peak in test accuracy. These two phenomena are however two sides of the same coin, resulting from a priorization of easy examples by the non-linear training dynamics.

References

Allen-Zhu, Z., Li, Y., and Song, Z. A convergence theory for deep learning via over-parameterization. volume 97 of *Proceedings of Machine Learning Research*, pp. 242–252, Long Beach, California, USA, 09–15 Jun 2019. PMLR. URL http://proceedings.mlr.press/v97/allen-zhu19a.html.

Baratin, A., George, T., Laurent, C., Devon Hjelm, R., Lajoie, G., Vincent, P., and Lacoste-Julien, S. Implicit regularization via neural feature alignment. In Banerjee, A. and Fukumizu, K. (eds.), *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pp. 2269–2277. PMLR, 13–15 Apr 2021. URL https://proceedings.mlr.press/v130/baratin21a.html.

Bartlett, P. L., Montanari, A., and Rakhlin, A. Deep learning: a statistical viewpoint. *arXiv:2103.09177 [math.ST]*, 2021.

Belkin, M., Hsu, D., Ma, S., and Mandal, S. Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences*, 116(32):15849–15854, 2019.

Chizat, L., Oyallon, E., and Bach, F. On lazy training in differentiable programming. *arXiv preprint arXiv:1812.07956*, 2018.

Cristianini, N., Shawe-Taylor, J., Elisseeff, A., and Kandola, J. On kernel-target alignment. In Dietterich, T., Becker, S., and Ghahramani, Z. (eds.), *Advances in Neural Information Processing Systems*, volume 14. MIT Press, 2001. URL https://proceedings.neurips.cc/paper/2001/file/1f71e393b3809197ed66df836fe833e5-Paper.pdf.

Du, S. S., Zhai, X., Poczos, B., and Singh, A. Gradient descent provably optimizes over-parameterized neural networks. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=S1eK3i09YQ.

Fort, S., Dziugaite, G. K., Paul, M., Kharaghani, S., Roy, D. M., and Ganguli, S. Deep learning versus kernel learning: an empirical study of loss landscape geometry and the time evolution of the neural tangent kernel. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 5850–5861. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/405075699f065e43581f27d67bb68478-Paper.pdf.

Jacot, A., Gabriel, F., and Hongler, C. Neural tangent kernel: Convergence and generalization in neural networks. In *NIPS*, pp. 8571–8580. 2018.

Jiang, Z., Zhang, C., Talwar, K., and Mozer, M. C. Characterizing structural regularities of labeled data in overparameterized models. In Meila, M. and Zhang, T. (eds.), *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pp. 5034–5044. PMLR, 2021. URL http://proceedings.mlr.press/v139/jiang21k.html.

Kopitkov, D. and Indelman, V. Neural spectrum alignment: Empirical study. In *International Conference on Artificial Neural Networks*, pp. 168–179. Springer, 2020.

Liu, Z., Luo, P., Wang, X., and Tang, X. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

Neyshabur, B., Tomioka, R., and Srebro, N. In search of the real inductive bias: On the role of implicit regularization in deep learning. *ICLR workshop track*, 2015.

Paccolat, J., Petrini, L., Geiger, M., Tyloo, K., and Wyart, M. Geometric compression of invariant manifolds in neural networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2021(4):044001, 2021.

Sagawa, S., Koh, P. W., Hashimoto, T. B., and Liang, P. Distributionally robust neural networks. In *International Conference on Learning Representations*, 2020a. URL https://openreview.net/forum?id=ryxGuJrFvS.

Sagawa, S., Raghunathan, A., Koh, P. W., and Liang, P. An investigation of why overparameterization exacerbates spurious correlations. In *International Conference on Machine Learning*, pp. 8346–8356. PMLR, 2020b.

Wah, C., Branson, S., Welinder, P., Perona, P., and Belongie, S. The caltech-ucsd birds-200-2011 dataset. 2011.

Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. Understanding deep learning requires rethinking generalization. *ICLR*, 2017.

A. How to assess linearity of training runs

In order to assess that empirically chosen values for the re-scaling factor α actually forces training into a linear regime, we compute 3 different metrics on training examples.

- **Sign similarity** counts the proportion of ReLUs that keep the same sign since initialization.
- **Tangent kernel alignment** measures the similarity of the tangent kernel matrix $\boldsymbol{K}^{(t)}$ with the initial kernel, in terms of matrix alignment
- **Representation alignment** measures the similarity of the last non-softmax layer representation $\phi_R^{(t)}(x)$ with its value at $t = 0$, in terms of the alignment of the corresponding Gram matrices $(\boldsymbol{K}_R^{(t)})_{ij} = \phi_R(\mathbf{x}_i)^\top \phi_R(\mathbf{x}_j)$.

with the kernel alignment (Cristianini et al., 2001) defined as:

$$\mathrm{KA}(\boldsymbol{K}^{(t)}, \boldsymbol{K}^{(0)}) = \frac{\mathrm{Tr}[\boldsymbol{K}^{(t)}\boldsymbol{K}^{(0)}]}{\|\boldsymbol{K}^{(t)}\|_F \|\boldsymbol{K}^{(0)}\|_F} \tag{6}$$

where $\|\cdot\|_F$ is the Froebenius norm.

We use these metrics in figure 2 right.

B. Experimental details

Experimental details Celeb A We use 20 000 examples of CelebA to train a ResNet18 classifier on the task of predicting whether a person is blond or not, using SGD with learning rate 0.01, momentum 0.9 and batch size 100. We also extract a balanced dataset with 180 (i.e. the total number of blond men in the test set) examples of 4 categories: blond man, blond woman, non-blond man and non-blond woman. While training progresses, we measure the loss and accuracy on the subgroups man and woman. Starting from the same initial conditions, we train 3 different classifiers with $\alpha \in \{.5, 1, 100\}$.

Experimental details Waterbirds We use 4 795 training examples of the Waterbirds dataset. Since the dataset is smaller, we start from a pre-trained ResNet18 classifier from default PyTorch models (pre-trained on ImageNet). We replace the last layer with a freshly initialized binary classification layer, and we set batch norm layers to evaluation mode. We train using SGD with learning rate 0.001, momentum 0.9 and minibatch size 100. From the training set, we extract a balanced dataset with 180 examples of each 4 groups: land birds on land background, land birds on water background, water bird on land background, and water bird on land background, that we group in 2 sets: *same* when the type of bird and the background agree, and *opposite* otherwise. While training, we measure the accuracies separately on these 2 sets. We train 3 different classifiers with $\alpha \in \{.5, 1, 100\}$.

Experimental details CIFAR10 C-scores We split the dataset CIFAR10 into 40 000 train examples and 10.000 test examples. Since computing C-scores is computationally expensive, we instead get pre-computed scores from `https://github.com/pluskid/structural-regularity`. We train a ResNet18 network using SGD with learning rate 0.01, momentum 0.9 and batch size 125. While training, we compute the loss and accuracy separately on 10 subsets of the training set ranked by increasing C-scores deciles (e.g. examples in the last subset are top-10% c-scores), for both the training set and test set. We also train a linearized copy of this network (so as to share the same initial conditions) with $\alpha = 100$. The $\alpha = 1$ run is trained for 200 epochs (64 000 SGD iterations) whereas in order to converge the $\alpha = 100$ run is trained for 1 000 epochs (320 000 SGD iterations).

Experimental details CIFAR10 noisy labels We use a ResNet18 network trained on CIFAR10 where 15% of the training examples are assigned a wrong (random) label. We use SGD with learning rate 0.01, momentum 0.9 and bach size 125. We compute the loss and accuracy independently on the regular examples with their true label, and the noisy examples whose label is flipped. In parallel, we train a copy of the initial model in the linearized regime with $\alpha = 100$.

C. Experiments variability

We reproduce the experiments in section 5 in order to assess variability with respect to model initialization and minibatch ordering.

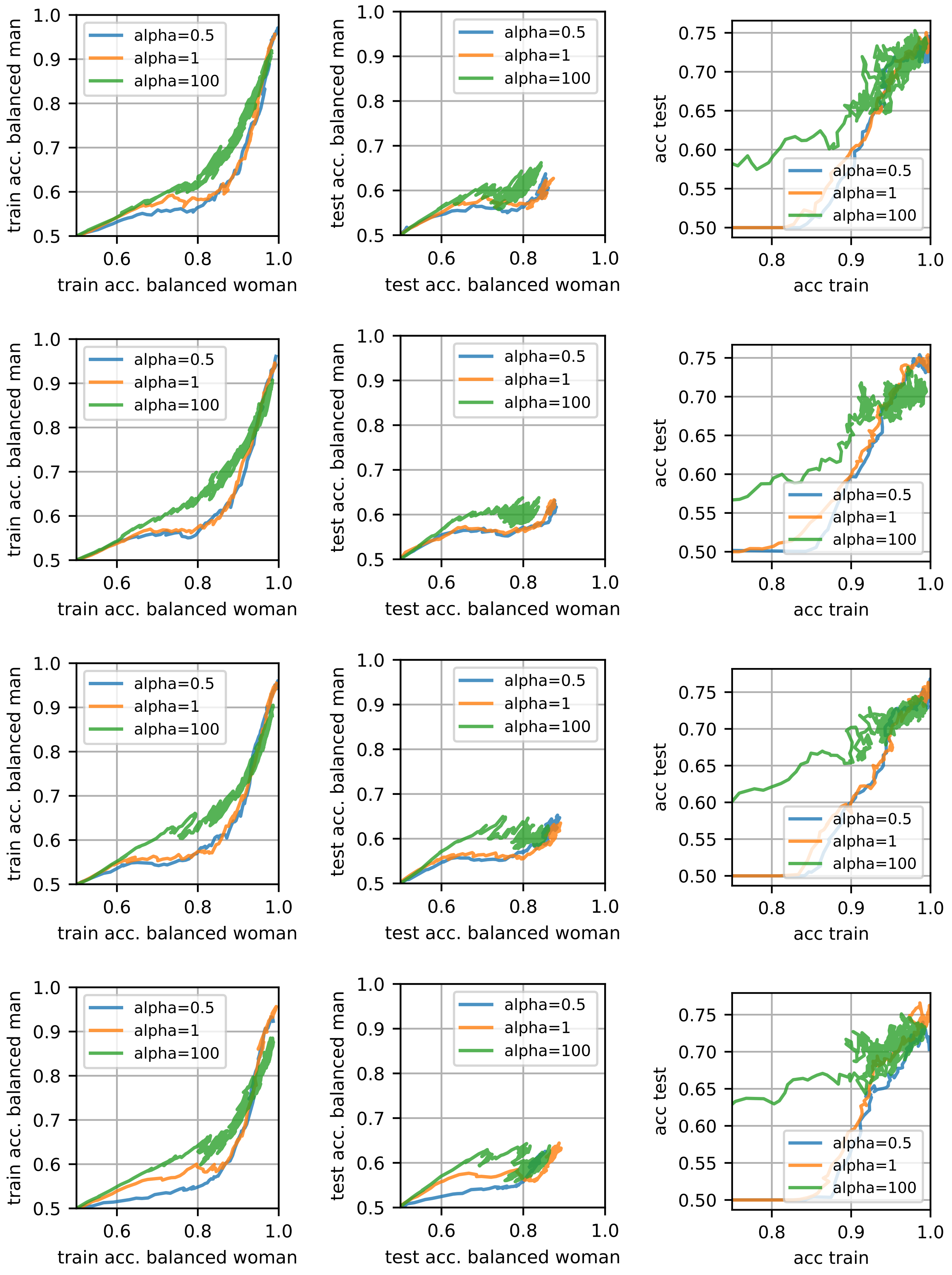

Figure 4. same as fig 3 with varying seed (model initialization and minibatch order)