A Generalization Bound for Nearly-Linear Networks

Anonymous authors
Paper under double-blind review

Abstract

We consider nonlinear networks as perturbations of linear ones. Based on this approach, we present novel generalization bounds that become non-vacuous for networks that are close to being linear. The main advantage over the previous works which propose non-vacuous generalization bounds is that our bounds are *a-priori*: performing the actual training is not required for evaluating the bounds. To the best of our knowledge, they are the first non-vacuous generalization bounds for neural nets possessing this property.

1 Introduction

Despite huge practical advancements of deep learning, the main object of this field, a neural network, is not yet fully understood. As we do not have a complete understanding of how neural networks learn, we are not able to answer the main question of deep learning theory: why do neural networks generalize on unseen data?

While the above question is valid for any supervised learning model, it is notoriously difficult to answer for neural nets. The reason is that modern neural nets have billions of parameters and as a result, huge capacity. Therefore among all parameter configurations that fit the training data, there provably exist such configurations that do not fit the held-out data well (Zhang et al., 2016). This is the reason why classical approaches for bounding a generalization gap, i.e. the difference between distribution and train errors, fall short on neural networks: such approaches bound the gap uniformly over a model class. That is, if weights for which the network performs poorly exist, we bound our trained network's performance by performance of that poor one.

As we observe empirically, networks commonly used in practice do generalize, which means that training algorithms we use (i.e. gradient descent or its variants) choose "good" parameter configurations despite the existence of poor ones. In other words, these algorithms are *implicitly biased* towards good solutions.

Unfortunately, implicit bias of gradient descent is not fully understood yet. This is because the training dynamics is very hard to integrate, or even characterize, analytically. There are two main obstacles we could identify. First, modern neural networks have many layers, resulting in a high-order weight evolution equation. Second, activation functions we use are applied to hidden representations elementwise, destroying a nice algebraic structure of stacked linear transformations.

If we remove all activation functions, the training dynamics of gradient descent can be integrated analytically (Saxe et al., 2013) under certain assumptions. However, the resulting model, a linear network, is as expressive as a linear model, thus loosing one of the crucial advantages of neural nets.

Idea. The idea we explore in the present paper is to consider nonlinear nets as perturbations of linear ones. We show that the original network can be approximated with a proxy-model whose parameters can be computed using parameters of a linear network trained the same way as the original one. Since the proxy-model uses the corresponding linear net's parameters, its generalization gap can be meaningfully bounded with classical approaches. Indeed, if the initial weights are fixed, the result of learning a linear network to minimize square loss on a dataset ($X \in \mathbb{R}^{d\times m}, Y \in \mathbb{R}^{d_{out}\times m}$) is solely determined by $YX^\top \in \mathbb{R}^{d_{out}\times d}$ and $XX^\top \in \mathbb{R}^{d\times d}$, where d, d_{out} are the input and output dimensions, and m is the dataset size. The number of parameters in these two matrices is much less than the total number of parameters in the network, making classical counting-based approaches meaningful.

Contributions. Our main contribution is a generalization bound given by Theorem 4.2, which is ready to apply in the following setting: (1) fully-connected networks, (2) gradient descent with vanishing learning rate (gradient flow), (3) binary classification with MSE loss. The main disadvantage of our bound is that it diverges as training time t goes to infinity. We discuss how to choose the training time such that the bound stays not too large while the training risk reduces significantly, in Section 4.3. Our generalization bound becomes non-vacuous as long as (1) a linear model provably generalizes well under the same setting, (2) the network we consider is close enough to being linear. We validate our bound on a simple fully-connected network trained on a downsampled MNIST dataset, and demonstrate that it becomes non-vacuous in this scenario (Section 5). We discuss assumption we use, as well as possible improvements of our bound, in Appendix A. Finally, we discuss how far the approach we use, i.e. generalization bounds based on deviation from specific proxy-models, could lead us in the best scenario (Appendix B).

2 Comparison to previous work

Our bounds has the following advantages over some other non-vacuous bounds available in the literature (e.g. Biggs & Guedj (2022); Galanti et al. (2023); Dziugaite & Roy (2017); Zhou et al. (2019)):

1. They are a-priori bounds, i.e. getting the actual trained the model is not required for evaluating them. To the best of our knowledge, they are the first non-vacuous a-priori bounds available for neural nets. All works mentioned above, despite providing non-vacuous bounds, could be evaluated only on a trained network thus relying on the implicit bias phenomenon which is not well understood yet.

2. The bound of our Theorem 4.2 does not require a held-out dataset to evaluate it, compared to Galanti et al. (2023) and coupled bounds of Biggs & Guedj (2022). Indeed, if one had a held-out dataset, one could just use it directly to evaluate the trained model, thus questioning the practical utility of such bounds.

3. It does not grow with network width. In contrast, PAC-Bayesian bounds, Biggs & Guedj (2022); Dziugaite & Roy (2017); Zhou et al. (2019), might grow with width.

4. Similarly to Biggs & Guedj (2022); Galanti et al. (2023), we bound the generalization gap of the original trained model, not its proxy. In contrast, Dziugaite & Roy (2017) introduces a Gaussian noise to the learned weights, while Zhou et al. (2019) crucially relies on quantization and compression after training.

For a fair comparison, we also list the disadvantages our present bound has:

1. The bound of our Theorem 4.2 becomes non-vacuous only when the following two conditions hold. First, a simple counting-based generalization bound for a linear model evaluated in the same setting should be non-vacuous. Such a bound is vacuous even for binary classification on the standard MNIST dataset, but becomes non-vacuous if we downsample the images.

2. Second, the activation function has to be sufficiently close to being linear. To be specific, for a two-layered leaky ReLU neural net trained on MNIST downsampled to 7x7, one needs the negative slope to be not less than 0.99 (1 corresponds to a linear net, while ReLU corresponds to 0).

3. One may hope for the bound to be non-vacuous only for a partially-trained network, while for a fully-trained network the bound diverges.

4. Even in the most optimistic scenario, when we manage to tighten the terms of our bound as much as possible, our bound stays non-vacuous only during the early stage of training when the network has not started "exploiting" its nonlinearity yet (Appendix B). However, the minimal negative ReLU slope for which the bound stays non-vacuous is much smaller, 0.6.

3 Related work

Non-vacuous generalization bounds. While first generalization bounds date back to Vapnik & Chervonenkis (1971), the bounds which are non-vacuous for realistically large neural nets appeared quite recently. Specifically, Dziugaite & Roy (2017) constructed a PAC-Bayesian bound which was non-vacuous for small fully-connected networks trained on MNIST and FashionMNIST. The bound of Zhou et al. (2019), also of PAC-Bayesian nature, relies on quantization and compression after training. It ends up being non-vacuous for VGG-like nets trained on large datasets of ImageNet scale. The bound of Biggs & Guedj (2022) is the first non-vacuous bound that applies directly to the learned model and not to its proxy. It is not obvious whether their construction could be generalized to neural nets with more than two layers. The bound of Galanti et al. (2023) is not PAC-Bayesian in contrast to the previous three. It is based on the notion of effective depth: it assumes that a properly trained network has small effective depth, even if it is deep. Therefore its effective capacity is smaller than its total capacity, which allows for a non-vacuous bound. See the previous section for discussion of some of the features of these bounds.

A-priori generalization bounds. Most of the generalization bounds available in the literature are a-posteriori. That is, some PAC-Bayesian bounds Dziugaite & Roy (2017); Neyshabur et al. (2018); Biggs & Guedj (2022) and Rademacher complexity-based bounds Bartlett et al. (2017) depend on norms of learned weights, or distances between the learned weights and their initializations, so they cannot be computed before these norms or distances are known. The bound of Zhou et al. (2019) depends on the trained model size after compression and quantization, which cannot be predicted before training and compression. That is, this bound implicitly relies on the fact that the training procedure prefers well-compressible models; to the best of our knowledge, this fact does not have a solid explanation yet. The bound of Galanti et al. (2023) depends on a so-called *effective depth* of a learned model, which also cannot be predicted in advance.

The only a-priori bounds we are aware of are the classical uniform bounds based on VC-dimension and Rademacher complexity, see Vapnik & Chervonenkis (1971). These bounds grow with model expressivity, hence with both width and depth, which make them vacuous in most of the practical scenarios involving neural nets. The reason they are vacuous are also related to the fact that among all models that interpolate the data, one can often find a model that does not generalize well, see empirical results of Zhang et al. (2016). Uniform bounds bound the generalization gap in worst case, hence if a "bad" model exists in our model class, the whole generalization bound has no chance to be good.

Overall, generalization bounds that use some complexity of the whole function class fall short due to the trade-off they impose: low complexity = provably good generalization, high complexity = maybe poor generalization. Modern neural architectures are deep and wide, hence they have high complexity, while they still generalize well. The reason is that the training procedure we often use is likely to be implicitly biased towards well-generalizing solutions. If we bound the generalization gap uniformly over the whole function class, we ignore this effect. The a-posteriori bounds we discuss above do take some form of implicit bias into account (i.e. low weight norm, compressibility etc.). So does our bound: we exploit the fact that a nearly linear network is close to a proxy which uses only weights of a linear network. In their turn, the trained weights of a linear network depend only on a few parameters as long as their initialization is given. The difference is that our bound does not need any prior info about the learned weights (only on their initialization and the optimization procedure); hence it is a-priori.

Linear networks training dynamics. The pioneering work that integrates the training dynamics of a linear network under gradient flow to optimize square loss is Saxe et al. (2013). This work crucially assumes that the initial weights are well aligned with the eigenvectors of the optimal linear regression weights YX^+, where (X, Y) is the train dataset. As the initialization norm approaches zero, the learning process becomes more sequential: components of the data are learned one by one starting from the strongest. Li et al. (2020) conjecture that the same happens for any initialization approaching zero excluding some set of directions of measure zero. They prove this result for the first, the strongest, component, but moving further seems more challenging. See also Yun et al. (2020); Jacot et al. (2021). We note a recent result of Tu et al. (2024) who propose a so-called *mixed* dynamics for two-layered linear networks that does not suffer from this issue.

Nearly-linear neural nets. Our generalization gap bound decreases as activation functions get closer to linearity. While nearly-linear activations do not conform with the usual practice, nearly-linear networks have been studied before. That is, Li et al. (2022) demonstrated that when width n and depth L go to infinity at the same time proportionally, the hidden layer covariances at initialization admit a meaningful limit as long as the ReLU slopes behave as $1 \pm \frac{c}{\sqrt{n}}$, i.e. become closer and closer to linearity. See also Noci et al. (2023) for a similar limit for Transformers (Vaswani et al., 2017).

Linearized training of neural networks and NTK. It is important to emphasize that linear networks we consider in the present work are fundamentally different from the NTK model introduced by Jacot et al. (2018) resulted from linearized training of a neural network. That is, by a linear network we mean a network $f_\theta(x)$ who is linear in its input, x, but not necessarily in its weights θ. Whereas if we linearize the training procedure, the trained model becomes linear in θ, but not in x. Jacot et al. (2018) demonstrate that training a neural network becomes equivalent to training a kernel method under specific (non-standard) parameterization as width goes to infinity (NTK limit). Since training a kernel method is equivalent to training a random feature model with sufficiently large number of features, the corresponding training procedure is indeed linear in weights. See also Yang & Littwin (2021) for proving NTK limit for general architectures, and Chizat et al. (2018) for demonstrating linearized training for large weight initializations.

As in the NTK limit training of a neural net becomes equivalent to training of a kernel method, features the model uses stay the same over the whole process of training. In other words, the model does not enjoy *feature learning* when training is linearized. In contrast, even a *linear* network with two layers trained with gradient descent does demonstrate feature learning: see Saxe et al. (2013); Tu et al. (2024). The feature learning it exhibits is kernel alignment and a so-called *momentum effect*: the associated kernel (NTK), or equivalently, the features the model uses, aligns over strong components of the data and extends along them Tu et al. (2024). This way, these strong components get learned first, before weak components, often resulted from data noise.

We also emphasize that the proxy models we use to prove our results, while exploiting weights of a trained linear network, are not linear themselves; neither in x, nor in weights. Therefore they do not suffer from expressivity issues as linear networks (they are all funtionally equivalent to linear models $x \to Wx$), while they do enjoy feature learning (since linear networks do).

Therefore our approach is quite orthogonal to NTK and generalizability of kernel methods. In terms of high-level methodology, the closest paper we could mention is Arora et al. (2019), where they consider a two-layer NTK-parameterized MLP, and prove a generalization bound for it when width is sufficiently large. For this, they combine a generalization bound for the infinite-width limit when the kernel is constant, and a term that takes into account the fact that the kernel for finite-width deviates from the limit one. Whereas what we do is we combine a generalization bound for a proxy model that exploits the weights of a linear network (linear in inputs, not in weights as in NTK), and a proxy deviation bound. The bound of Arora et al. (2019) is good when the width is large (hence the finite-width NTK is close to the limit NTK), while our bound is good when the activation functions are close to be linear (hence the proxy does not deviate from the original model much).

4 Main result

Notations. For integer $L \geq 0$, $[L]$ denotes the set $\{1, \ldots, L\}$. For integers $l \leq L$, $[l, L]$ denotes the set $\{l, \ldots, L\}$. For a vector x, $\|x\|$ denotes its Euclidean norm. For a matrix X, $\|X\|$ denotes its maximal singular value, while $\|X\|_F$ denotes its Frobenius norm.

4.1 Setup

Model. The model we study is a fully-connected LeakyReLU network with L layers:

$$f_\theta^\epsilon(x) = W_L x_{\theta,L-1}^\epsilon(x), \qquad x_{\theta,0}^\epsilon(x) = x, \qquad x_{\theta,l}^\epsilon(x) = \phi^\epsilon\left(W_l x_{\theta,l-1}^\epsilon(x)\right) \quad \forall l \in [L-1], \tag{1}$$

where $\theta \in \mathbb{R}^N$ denotes the vector of all weights, i.e. $\theta = \mathrm{cat}(\{\mathrm{vec}(W_l)\}_{l=1}^L)$, $W_l \in \mathbb{R}^{n_l \times n_{l-1}}$ $\forall l \in [L]$, and ϕ^ϵ is a Leaky ReLU with a negative slope $1-\epsilon$, that is:

$$\phi^\epsilon(x) = x - \epsilon \min(0, x). \tag{2}$$

Since we are going to consider only binary classification in the present work, we take $n_L = 1$. We also define $d = n_0$ to denote the input dimension.

Data. Data points (x, y) come from a distribution $\mathcal{D}$. We assume all x from $\mathcal{D}$ to lie on a unit ball, $\|x\| \leq 1$, and all y to equal ± 1. During the training phase, we sample a set of m points iid from $\mathcal{D}$ to form a dataset (X, Y), where $X \in \mathbb{R}^{d \times m}$ and $Y \in \mathbb{R}^{1 \times m}$.

Training. Assuming $\mathrm{rk}\, X = d$ (which implies $m \geq d$, i.e. the data is abundant), we train our model on (X, Y) with gradient flow to optimize square loss on whitened data, i.e. on $(\tilde{X}, Y)$ for $\tilde{X} = \Sigma_X^{-1/2} X$, where $\Sigma_X = \frac{1}{m} X X^\top \in \mathbb{R}^{d \times d}$ is an empirical feature correlation matrix. That is,

$$\frac{dW_l^\epsilon(t)}{dt} = -\frac{\partial \left(\frac{1}{2m} \left\| Y - f^\epsilon_{\theta^\epsilon(t)}(\tilde{X}) \right\|_F^2 \right)}{\partial W_l} \quad \forall l \in [L]. \tag{3}$$

Note that $\tilde{X}\tilde{X}^\top = m I_d$.

Inference. To conform with the above training procedure, we take the model output at a point x to be $f^\epsilon_\theta\left(\Sigma^{-1/2} x\right)$, where Σ is a (distribution) feature correlation matrix: $\Sigma = \mathbb{E}_{(x,y)\sim\mathcal{D}}(x x^\top) \in \mathbb{R}^{d \times d}$. We assume this matrix to be known; in practice, we could substitute it with Σ_X.

Performance measure. We take the risk function to be misclassification loss: $r(z, y) = \mathrm{Ind}[zy < 0]$, where $z, y \in \mathbb{R}$ and $\mathrm{Ind}[\cdot]$ is an indicator of a condition. The empirical (train) risk on the dataset (X, Y) and the distribution risk of the model trained for time t are then defined as

$$\hat{R}^\epsilon(t) = \frac{1}{m} \sum_{k=1}^m r\left(f^\epsilon_{\theta^\epsilon(t)}(\Sigma^{-1/2} x_k), y_k \right), \qquad R^\epsilon(t) = \mathbb{E}_{(x,y)\sim\mathcal{D}} r\left(f^\epsilon_{\theta^\epsilon(t)}(\Sigma^{-1/2} x), y \right). \tag{4}$$

For a given model f, we also define $R(f)$ and $\hat{R}(f)$ accordingly.

In addition, for $\gamma > 0$, we define γ-margin loss: $r_\gamma(z, y) = \mathrm{Ind}[z \operatorname{sgn} y < \gamma]$, and its continuous version:

$$r^C_\gamma(z, y) = \begin{cases} 1, & zy < 0, \\ 1 - zy/\gamma, & zy \in [0, \gamma], \\ 0, & zy > \gamma \end{cases} \tag{5}$$

We define R^γ and R^C_γ analogously to R, and $\hat{R}^\gamma$ and $\hat{R}^C_\gamma$ analogously to $\hat{R}$.

4.2 Generalization bound

We will need the following assumption on the training process:

Assumption 4.1. $\forall t \geq 0$ $\left\| \left(Y - f^\epsilon_{\theta^\epsilon(t)}(\tilde{X}) \right) \tilde{X}^\top \right\|_F^2 \leq \left\| \left(Y - f^\epsilon_{\theta^\epsilon(0)}(\tilde{X}) \right) \tilde{X}^\top \right\|_F^2$.

Note that $\left\| Y - f^\epsilon_{\theta^\epsilon(t)}(\tilde{X}) \right\|_F^2 \leq \left\| Y - f^\epsilon_{\theta^\epsilon(0)}(\tilde{X}) \right\|_F^2$ since we minimize the loss monotonically with gradient flow. This implies that the above assumption holds automatically, whenever $\tilde{X}$ is a (scaled) orthogonal matrix (which happens when $m = d$). It is easy to show that it provably holds for a linear network ($\epsilon = 0$), see Appendix E, and we found this assumption to hold empirically for all of our experiments with nonlinear networks too, see Figure 7.

We are now ready to formulate our main results:

Theorem 4.2. *Fix $\beta, \gamma > 0$, $t \geq 0$, $\delta \in (0,1)$, $\epsilon \in [0,1]$, and $\kappa \in \{1,2\}$. Let p be the floating point arithmetic precision (32 by default). Under the setting of Section 4.1 and Assumption 4.1, for any weight initialization satisfying $\|W_l^\epsilon(0)\| \leq \beta$ $\forall l \in [L]$, w.p. $\geq 1-\delta$ over sampling the dataset (X,Y),*

$$R^\epsilon(t) \leq \hat{R}_\gamma^\epsilon(t) + \Upsilon_\kappa + \frac{\Delta_{\kappa,\beta}(t)\epsilon^\kappa}{\gamma}, \tag{6}$$

where

$$\Upsilon_\kappa = \sqrt{\frac{\kappa p d \ln 2 + \ln(1/\delta)}{2m}}, \tag{7}$$

and

$$\Delta_{\kappa,\beta}(t) = \Phi_\kappa v_\beta(t) u_\beta^{L-1}(t), \tag{8}$$

where

$$\Phi_\kappa = \begin{cases} L\sqrt{d} + 1 + (L-1)\rho, & \kappa = 1; \\ (L-1)\left[(L+1+(L-1)\rho)\sqrt{d} + 2(1+(L-1)\rho)\right], & \kappa = 2, \end{cases} \tag{9}$$

where $\rho = \frac{\|W_1^\epsilon(0)\|_F}{\|W_1^\epsilon(0)\|}$ is the square root of the stable rank of the input layer at initialization, and the definitions of u_β and v_β are given below.

Define

- $s = \left\| Y X^\top \Sigma_X^{-1/2} \right\|$; $\quad \bar{1} = 1 + \rho\beta^L$;
- *For a given $r \geq 0$, $\bar{s}_r = (1-\epsilon)(s + \rho\beta^L) + \epsilon\sqrt{r}(L-1)(1+\rho\beta^L)$;*
- $\bar{s} = \bar{s}_1$; $\quad \hat{s} = \frac{L}{1+(L-1)\rho}\bar{s}$; $\quad \bar{\beta} = \beta\frac{\bar{s}_\rho}{\bar{s}_1}$.

We have

$$u_\beta(t) = \begin{cases} \bar{\beta} e^{\bar{s}t}, & L = 2; \\ \left(\bar{\beta}^{2-L} - (L-2)\bar{s}t\right)^{\frac{1}{2-L}}, & L \geq 3; \end{cases} \tag{10}$$

$$v_\beta(t) = \frac{L-1}{L}\hat{s}^{\frac{2-L}{L}} u_\beta^{L-1}(t)[w(u_\beta(t)) - w(\beta)] e^{\frac{u_\beta^L(t)}{\hat{s}}}, \tag{11}$$

where [1]

$$w(u) = -\frac{\bar{1}}{\bar{s}}\Gamma\left(\frac{2-L}{L}, \frac{u^L}{\hat{s}}\right) - \rho\frac{\hat{s}}{\bar{s}}\Gamma\left(\frac{2}{L}, \frac{u^L}{\hat{s}}\right). \tag{12}$$

This theorem gives an a-priori bound for the generalization gap $R^\epsilon - \hat{R}_\gamma^\epsilon$, i.e. it could be computed without performing the actual training.

Proof idea and meaning of terms. In order to prove Theorem 4.2, we approximate our nonlinear network with a model which uses weights of a linear network trained the same way as the original one we consider. We consider two proxy-models, of order 1 and 2, indicated by κ. Since they use linear networks weights, the number of parameters of these proxy-models happens to be only κ times larger than the one of a linear model. This gives Υ_κ, which is a simple counting-based bound for the generalization gap. The last term of the bound accounts for a deviation of these proxy-models from the original one.

Dimension dependency. Our bound does not depend on width n_l $\forall l \in [L-1]$, in contrast to the bounds of Dziugaite & Roy (2017); Zhou et al. (2019); Biggs & Guedj (2022). However, both penalty terms of Theorem 4.2, Υ_κ and $\Delta_{\beta,\kappa}(t)$, grow as $\sqrt{d}$ with the input dimension.

[1] Γ is an upper-incomplete gamma-function defined as $\Gamma(s,x) = \int_x^\infty t^{s-1}e^{-t}\,dt$.

4.3 Choosing the training time

As we see, $\Delta_\kappa(t)$ diverges super-exponentially as $t \to \infty$ for $L = 2$ and as $t \to \frac{\bar{\beta}^{2-L}}{(L-2)\bar{s}}$ for $L \geq 3$, so the bound eventually becomes vacuous. For the bound to make sense, we should be able to find t small enough for $\Delta_\kappa(t)$ to stay not too large, and at the same time, large enough for the training risk $\hat{R}^\epsilon_\gamma(t)$ to get considerably reduced.

In the present section, we are going to demonstrate that for values of t which correspond to partially learning the dataset (i.e. for which $\hat{R}^\epsilon_\gamma(t) \in (0,1)$), the last term of the bound, $\Delta_{\kappa,\beta}(t)/\gamma$, admits a finite limit as $\beta \to 0$ (the saddle-to-saddle regime of Jacot et al. (2021)).

We do not know how $\hat{R}^\epsilon_\gamma(t)$ decreases with t in our case. However, when the network is linear, this can be computed explicitly when either the weight initialization is properly aligned with the data, or the initialization norm β vanishes. We are going to perform our whole analysis for $L = 2$ in the main, and defer the case $L \geq 3$ to Appendix D.2.

That is, consider an SVD: $Y\tilde{X}^+ = \frac{1}{\sqrt{m}} YX^\top (XX^\top)^{-1/2} = PSQ^\top$, where P and Q are orthogonal and S is diagonal; note that $S_{11} = s$. Observe also that $s = \|Y\tilde{X}^+\| \leq \|Y\|\|\tilde{X}^+\| \leq 1$ since $y = \pm 1$. Saxe et al. (2013)[2] have demonstrated for linear nets ($\epsilon = 0$) and $L = 2$ that when the weight initialization is properly aligned with P and Q[3], $\|W_1^0(t)\| = \|W_2^0(t)\| = \bar{u}(t)$, where $\bar{u}$ satisfies[4]

$$\frac{d\bar{u}(t)}{dt} = \bar{u}(t)(s - \bar{u}^2(t)), \qquad \bar{u}(0) = \beta, \tag{13}$$

which gives the solution in implicit form:

$$t = \int \frac{d\bar{u}}{\bar{u}(s - \bar{u}^2)} = \frac{1}{2s} \ln \left(\frac{\bar{u}^2(t)(s - \beta^2)}{\beta^2(s - \bar{u}^2(t))} \right). \tag{14}$$

One could resolve $\bar{u}$ explicitly to get

$$\bar{u}(t) = \frac{se^{2st}}{e^{2st} - 1 + s/\beta^2}. \tag{15}$$

Saxe et al. (2013) observed this expression to hold with good precision for random (not aligned) initializations when β is small enough. This is supported by further works (Li et al., 2020; Jacot et al., 2021): when a linear network is initialized close to the origin and the initial weights do not lie on a "bad" subspace, gradient flow nearly follows the same trajectory as studied by Saxe et al. (2013).

Plugging $\bar{u}^2(t) = \alpha s$ into Equation (14), we get the time required for a linear network to learn a fraction α of the strongest mode:

$$t^*_\alpha(\beta) = \frac{1}{2s} \ln \left(\frac{\alpha(s - \beta^2)}{(1 - \alpha)\beta^2} \right). \tag{16}$$

Clearly, the learning time $t^*_\alpha(\beta)$ diverges whenever $\beta \to 0$, or $\alpha \to 1$.

Since $t^*_\alpha(\beta)$ is the time sufficient to learn a network for $\epsilon = 0$, we suppose it also suffices to learn a nonlinear network. So, we are going to evaluate our bound at $t = t^*_\alpha$. Since we need $\hat{R}_\gamma(t^*_\alpha) < 1$ for the bound to be non-vacuous, we should take γ small relative to α. We consider $\gamma = \alpha^\nu/q$ for $\nu, q \geq 1$.

Since the linear network learning time $t^*_\alpha(\beta)$ is correct for almost all initialization only when β vanishes, we are going to work in the limit of $\beta \to 0$. Since we need $\alpha \in (\beta^2/s, 1)$, otherwise the linear training time is negative, we take $\alpha = \frac{r}{s}\beta^\lambda$ for $\lambda \in (0, 2]$ and $r > 1$.

Let us compute the quantities which appear in our bounds, at $t = t^*_\alpha(\beta)$:

$$u = \bar{\beta} \left(\frac{\alpha(s - \beta^2)}{(1 - \alpha)\beta^2} \right)^{\frac{\bar{s}}{2s}} = \frac{\bar{s}_\rho}{\bar{s}_1} r^{\frac{\bar{s}}{2s}} \beta^{1 + \frac{\bar{s}}{s}\left(\frac{\lambda}{2} - 1\right)} (1 + O(\beta)), \tag{17}$$

[2]Saxe et al. (2013) assumed $XX^\top = I_d$ and $\|YX^\top\| = s$, while not introducing the factor $\frac{1}{m}$ as we do in Equation (3). It is easy to see that the our gradient flow has exactly the same dynamics as the one studied by Saxe et al. (2013).

[3]That is, when $W_2(0) = P\bar{S}_2^{1/2}R^\top$, $W_1(0) = R\bar{S}_1^{1/2}Q^\top$, where R is orthogonal, and $\bar{S}_1$ and $\bar{S}_2$ are constructed from S by adding or removing zero rows and columns.

[4]Our $\bar{u}$ corresponds to $u^{1/(N_l - 1)}$ of Saxe et al. (2013).

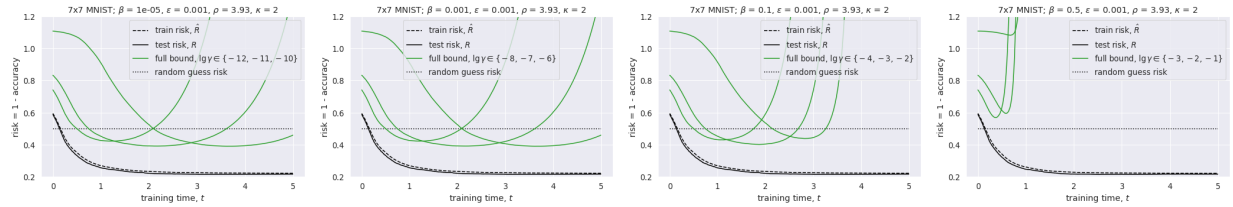

Figure 1: We consider 7x7 binary MNIST, $L = 2$, $\kappa = 2$, $\epsilon = 0.001$, and vary β. The bound of Theorem 4.2 converges as β vanishes and increases as β grows. The bound stays non-vacuous for a small enough β and a properly choosen γ. We consider $\gamma = \beta^2/q$ for $q \in \{1, 10, 100\}$.

where we omitted the argument $t^*_\alpha(\beta)$ for brevity.

Since $\Gamma(0, x) = -\operatorname{Ei}(-x) = -\ln x + O_{x\to 0}(1)$ and $\Gamma(1, x) = e^{-x} = O_{x\to 0}(1)$, we have $w(\beta) = \frac{\bar{1}}{\bar{s}} \ln(\beta^2) + O(1)$. Similarly, whenever $1 + \frac{\bar{s}}{s}\left(\frac{\lambda}{2} - 1\right) > 0$, we have $u = o(\beta)$, and $w(u) = \frac{\bar{1}}{\bar{s}} \ln\left(\beta^{2+\frac{\bar{s}}{s}(\lambda-2)}\right) + O(1)$.

Consider first $\lambda < 2$ and suppose $\nu = 1$. In this case, $w(u) - w(\beta) = \frac{\bar{1}}{s}\left(\frac{\lambda}{2} - 1\right)\ln(\beta^2) + O(1)$, and

$$\frac{2uv}{\gamma} = \frac{\bar{s}_\rho^2}{\bar{s}_1^2} \frac{q r^{\frac{\bar{s}}{s}-1}\bar{1}}{\beta^{\left(\frac{\bar{s}}{s}-1\right)(2-\lambda)}} \left(\frac{\lambda}{2} - 1\right) \ln(\beta^2)(1 + O(1/\ln\beta)). \tag{18}$$

Since $\bar{s} \geq s$, this expression diverges as $\beta \to 0$. Note that we get the same order divergence even also for $1 + \frac{\bar{s}}{s}\left(\frac{\lambda}{2} - 1\right) \leq 0$. Clearly, for $\nu > 1$, we get even faster divergence.

On the other hand, if $\lambda = 2$ then $w(u) - w(\beta) = \frac{\bar{1}}{\bar{s}} \ln\left(\frac{\bar{s}_\rho^2}{\bar{s}_1^2} r^{\frac{\bar{s}}{s}}\right) + O(\beta)$, and

$$\frac{2uv}{\gamma} = \frac{s^\nu \bar{s}_\rho^2}{\bar{s}_1^3} q r^{\frac{\bar{s}}{s}-\nu}\bar{1} \ln\left(\frac{\bar{s}_\rho^2}{\bar{s}_1^2} r^{\frac{\bar{s}}{s}}\right) \beta^{2(1-\nu)}(1 + O(\beta)). \tag{19}$$

We get a finite limit only for $\nu = 1$, i.e. when $\gamma \propto \alpha$. In this case, the last term of the bound Equation (6) becomes

$$\frac{\Delta_{\kappa,\beta}\epsilon^\kappa}{\gamma} = \frac{\bar{1} s \bar{s}_\rho^2}{2\bar{s}_1^3} q r^{\frac{\bar{s}}{s}-1} \Lambda_\kappa \ln\left(\frac{\bar{s}_\rho^2}{\bar{s}_1^2} r^{\frac{\bar{s}}{s}}\right) \epsilon^\kappa (1 + O(\beta)). \tag{20}$$

Summing up, we expect the empirical risk $\hat{R}_\gamma(t^*_\alpha)$ to be smaller than $\frac{1}{2}$ for large enough q (i.e. for small γ compared to α), and the last term to stay finite as $\beta \to 0$ and vanish as ϵ^κ. Therefore as long as Υ_κ is not large enough (i.e. when $\kappa p d$ is small compared to m), the overall bound becomes non-vacuous for small enough ϵ at least at the (linear) training time t^*_α. We are going to evaluate our bound empirically in the upcoming section.

5 Experiments

Setup. We consider an L-layer bias-free fully-connected network of width 64 and train it to classify 0-4 versus 5-9 digits of MNIST (i.e. $m = 60000$). In order to approximate the gradient flow dynamics, we run gradient descent with learning rate 0.001. By default, we take $L = 2$, the floating point precision to be $p = 32$, downsample the images to 7x7, and initialize the layers randomly in a standard Pytorch way (plus, we rescale the weights to match the required layer norm β). For some experiments, we consider deeper networks, half-precision $p = 16$, downsample not so aggresively, or enforce the input layer weight matrix to have rank 1.

Observations:

- When we take $\kappa = 2$ and $\beta \leq 0.1$, the bound becomes nonvacuous up to $\epsilon = 0.001$ (Figure 1) and even up to $\epsilon = 0.01$ if we enforce $\rho = 1$ (Figure 3);

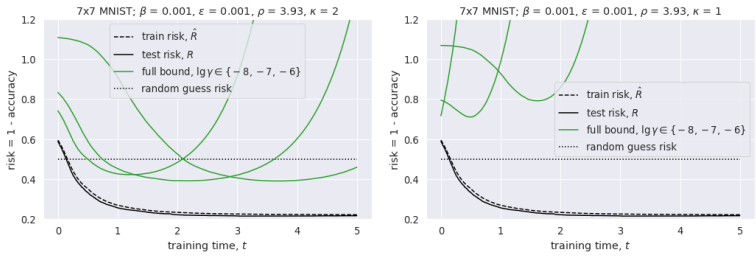

Figure 2: We consider 7x7 binary MNIST, $L = 2$, $\beta = 0.001$, $\epsilon = 0.001$, and compare different kappas of Theorem 4.2. The bound for $\kappa = 2$ is much stronger than that for $\kappa = 1$.

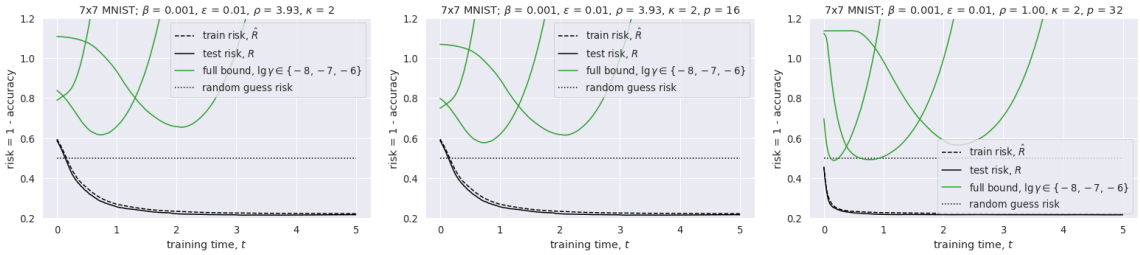

Figure 3: We consider 7x7 binary MNIST, $L = 2$, $\beta = 0.001$, $\epsilon = 0.01$, $\kappa = 2$, and vary the stable rank at initialization ρ and floating point precision p. Initializing the input layer with a rank one matrix considerably improves the bound. Moreover, it also improves the convergence speed.

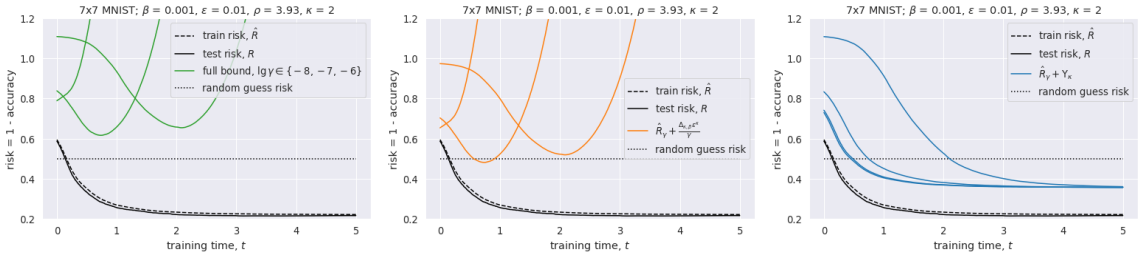

Figure 4: We consider 7x7 binary MNIST, $L = 2$, $\beta = 0.001$, $\epsilon = 0.01$, $\kappa = 2$, and compare different components of the bound. The left figure corresponds to the full bound, while for the central one we forget about the generalization gap bound for the proxy model Υ_κ, and for the rightmost one, we forget about the deviation term $\frac{\Delta_{\kappa,\beta}\epsilon^\kappa}{\gamma}$. We see that both terms are of the same order; one therefore has to work on reducing both in order to reduce the overall bound.

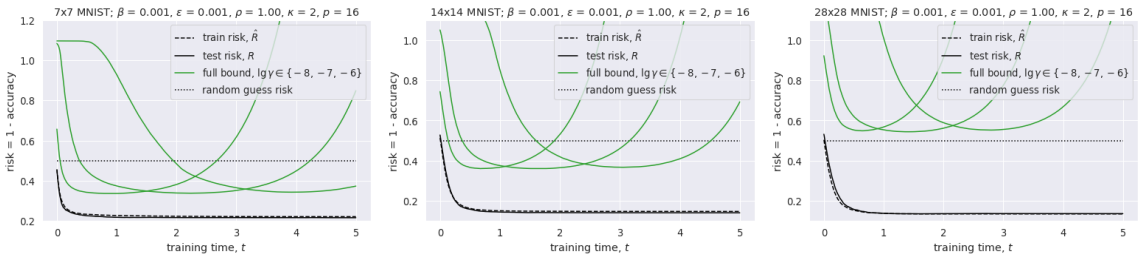

Figure 5: We consider binary MNIST, $L = 2$, $\beta = 0.001$, $\epsilon = 0.001$, $\kappa = 2$, $p = 16$, rank one input layer initialization, and vary image dimensions. In this "gentle" scenario, the bound stays non-vacuous for 14x14 MNIST, and only slightly exceeds the random guess risk for the full-sized, 28x28 MNIST.

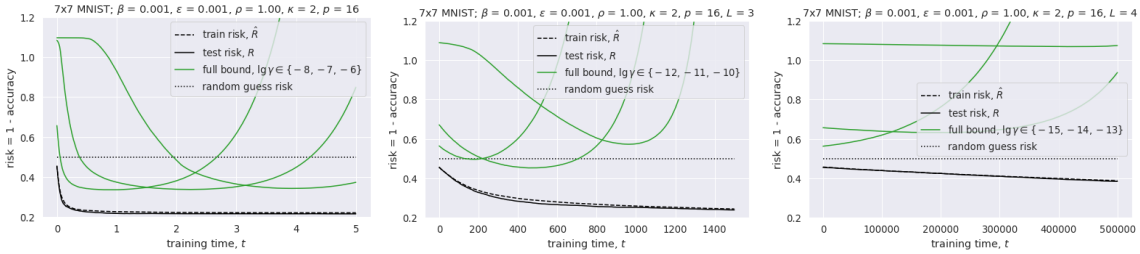

Figure 6: We consider binary 7x7 MNIST, $\beta = 0.001$, $\epsilon = 0.001$, $\kappa = 2$, $p = 16$, rank one input layer initialization, and vary depth. Even in this "gentle" scenario, the bound gets considerably worse with depth.

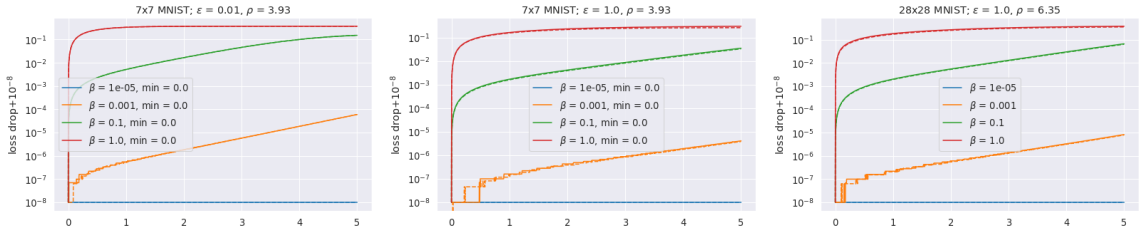

Figure 7: We consider binary MNIST, $L = 2$, $\beta = 0.001$, $\epsilon = 0.001$, and measure the differences $\mathcal{L}(0) - \mathcal{L}(t)$ (solid lines) and $\mathcal{L}_X(0) - \mathcal{L}_X(t)$ (dashed lines). Here $\mathcal{L}(t) = \frac{1}{2m}\left\|Y - f^{\epsilon}_{\theta^{\epsilon}(t)}(\tilde{X})\right\|_F^2$ is the loss at time t, while $\mathcal{L}_X(t) = \frac{1}{2m}\left\|\left(Y - f^{\epsilon}_{\theta^{\epsilon}(t)}(\tilde{X})\right)\tilde{X}^{\top}\right\|_F^2$ is the "projected" loss at time t. While $\mathcal{L}(t)$ should clearly decrease for gradient descent with small enough steps, it is not a-priori clear that $\mathcal{L}_X(t)$ also does. As we see from the plots, it does for β large and small, and for ϵ up to 1, which corresponds to conventional ReLU activations. These results validate our Assumption 4.1. Note that we added a small quantity 10^{-8} in order to make zero visible.

- We can have a non-vacuous bound also for 14x14 MNIST if we take $\epsilon = 0.001$, half-precision ($p = 16$), and enforce $\rho = 1$ (Figure 5); the bound slightly exceeds the random guess risk for the full-sized, 28x28 MNIST;
- The bound for $\kappa = 2$ is much stronger than that for $\kappa = 1$ (Figure 2);
- The bound improves and converges as β vanishes (Figure 1);
- Assumption 4.1 holds empirically (Figure 7);
- The bound improves if we enforce $\rho = 1$; this also results in faster convergence (Figure 3);
- The bound slightly improves for the half floating point precision (Figure 3);
- For $\epsilon = 0.01$ at the training time, all components of our bound are of the same order, while for larger ϵ, the last one starts to dominate (Figure 4);
- The bound deteriorates consideraly when we increase depth (Figure 6).

6 Proof of the main result

6.1 Proof of Theorem 4.2 for $\kappa = 1$

We start with approximating our learned network $f^{\epsilon}_{\theta^{\epsilon}}$ with $f^{\epsilon}_{\theta^0}$, i.e. with a nonlinear network that uses the weights learned by the linear one. This approximation deviates from the original network the following way: $\forall x \in \mathbb{R}^d, \epsilon \in [0, 1]$,

$$\frac{1}{\epsilon} \left\| f^{\epsilon}_{\theta^{\epsilon}}(x) - f^{\epsilon}_{\theta^0}(x) \right\| = \frac{1}{\epsilon} \left\| \int_0^{\epsilon} \frac{\partial f^{\epsilon}_{\theta^{\tau}}(x)}{\partial \tau} \, d\tau \right\| \leq \sup_{\tau \in [0,\epsilon]} \left\| \frac{\partial f^{\epsilon}_{\theta^{\tau}}(x)}{\partial \tau} \right\| \leq \sup_{\tau \in [0,\epsilon]} \sum_{l=1}^{L} \left\| \frac{\partial W_l^{\tau}}{\partial \tau} \right\| \prod_{k \neq l} \|W_k^{\tau}\| \|x\| \tag{21}$$

since we use Leaky ReLUs. We omitted the argument t for brevity. We get a similar deviation bound on the training dataset:

$$\frac{1}{\epsilon} \left\| f^{\epsilon}_{\theta^{\epsilon}}(\tilde{X}) - f^{\epsilon}_{\theta^0}(\tilde{X}) \right\| \leq \sup_{\tau \in [0,\epsilon]} \left\| \frac{\partial W_1^{\tau} \tilde{X}}{\partial \tau} \right\|_F \prod_{k=2}^{L} \|W_k^{\tau}\| + \sup_{\tau \in [0,\epsilon]} \sum_{l=2}^{L} \left\| \frac{\partial W_l^{\tau}}{\partial \tau} \right\| \|W_1^{\tau} \tilde{X}\|_F \prod_{k \in [2:L] \setminus \{l\}} \|W_k^{\tau}\|. \tag{22}$$

We complete the above deviation bound with bounding weight norms and norms of weight derivatives:

Lemma 6.1. *Under Assumption 4.1, $\forall \tau \in [0, 1], t \geq 0$ $\frac{1}{\sqrt{m}} \|W_1^{\tau}(t) \tilde{X}\|_F \leq \rho u(t)$, $\frac{1}{\sqrt{m}} \left\| \frac{\partial W_1^{\tau} \tilde{X}}{\partial \tau}(t) \right\|_F \leq v(t)$, and $\forall l \in [L]$ $\|W_l^{\tau}(t)\| \leq u(t)$, $\left\| \frac{\partial W_l^{\tau}}{\partial \tau}(t) \right\| \leq v(t)$ for u, v defined in Theorem 4.2.*

See Section 6.3 and Appendix D.1 for the proof.

Now we can relate the risk of the original model with the risk of the approximation. Since $r \leq r_{\gamma}^C \leq r_{\gamma}$ and r_{γ}^C is $1/\gamma$-Lipschitz,

$$\begin{aligned} R(f^{\epsilon}_{\theta^{\epsilon}}) - \hat{R}_{\gamma}(f^{\epsilon}_{\theta^{\epsilon}}) &\leq R_{\gamma}^C(f^{\epsilon}_{\theta^{\epsilon}}) - \hat{R}_{\gamma}^C(f^{\epsilon}_{\theta^{\epsilon}}) \\ &\leq R_{\gamma}^C(f^{\epsilon}_{\theta^0}) - \hat{R}_{\gamma}^C(f^{\epsilon}_{\theta^0}) + \frac{1}{\gamma} \mathbb{E} \left\| f^{\epsilon}_{\theta^{\epsilon}}(\tilde{x}) - f^{\epsilon}_{\theta^0}(\tilde{x}) \right\| + \frac{1}{\gamma m} \left\| f^{\epsilon}_{\theta^{\epsilon}}(\tilde{X}) - f^{\epsilon}_{\theta^0}(\tilde{X}) \right\|_1, \end{aligned} \tag{23}$$

where the expectation is over the data distribution $\mathcal{D}$, and $\tilde{x} = \Sigma^{-1/2} x$ is the actual input of the network.

As for the last term, the deviation on the train dataset, we use that $\|z\|_1 \leq \sqrt{m} \|z\|$ for any $z \in \mathbb{R}^m$, and Equation (22). As for the deviation on the test dataset, due to Equation (21) and Lemma 6.1, in order to bound the last term, it suffices to bound $\mathbb{E} \|\tilde{x}\|$. Since $\Sigma = \mathbb{E}\left[x x^{\top}\right]$, we get

$$\mathbb{E} \|\Sigma^{-1/2} x\|^2 = \mathbb{E} \left[x^{\top} \Sigma^{-1} x \right] = \mathbb{E} \operatorname{tr} \left[x x^{\top} \Sigma^{-1} \right] = \operatorname{tr}[I_d] = d. \tag{24}$$

This gives $\mathbb{E}\,\|\Sigma^{-1/2}x\| \leq \sqrt{\mathbb{E}\,\|\Sigma^{-1/2}x\|^2} \leq \sqrt{d}$.

The first two terms is a generalization gap of the proxy-model. We use a simple counting-based bound:

$$R_\gamma^C\left(f^\epsilon_{\theta^0(t)}\right) - \hat{R}_\gamma^C\left(f^\epsilon_{\theta^0(t)}\right) \leq \sqrt{\frac{\ln|\mathcal{F}_1^\epsilon(t;\theta(0))| - \ln\delta}{2m}}, \tag{25}$$

w.p. $\geq 1-\delta$ over sampling the dataset (X, Y), where $\mathcal{F}_1^\epsilon(t;\theta(0))$ denotes the set of functions representable with $f^\epsilon_{\theta^0(t)}$ for a given initial weights $\theta(0)$, where $\theta^0(t)$ is a result of running the gradient flow Equation (3) for time t. As long as we work with finite precision, this class is finite:

Lemma 6.2. $\forall\theta(0), \epsilon, t \geq 0$, $|\mathcal{F}_1^\epsilon(t;\theta(0))| \leq 2^{pd}$.

Proof. Since we run our gradient flow Equation (3) on whitened data to optimize squared loss, the initial weights are fixed, and the network we train is linear, the resulting weights depend only on $Y\tilde{X}^\top$ which has d parameters. Since each function in our class is completely defined with the resulting weights, and each weight occupies p bits, we get the above class size. □

We could have used a classical VC-dimension-based bound instead. However, we found it to be numerically larger compared to the counting-based bound above.

This finalizes the proof of Theorem 4.2 for $\kappa = 1$.

6.2 Proof of Theorem 4.2 for $\kappa = 2$

What changes for $\kappa = 2$ is a proxy-model. Consider the following:

$$\tilde{f}^\epsilon_{\theta^0,\theta^\epsilon}(x) = f^\epsilon_{\theta^0}(x) + \left(f^\epsilon_{\theta^\epsilon}(\tilde{X}) - f^\epsilon_{\theta^0}(\tilde{X})\right)\tilde{X}^+x. \tag{26}$$

That is, we take the same proxy-model as before, but we add a linear correction term. This correction term aims to fit the proxy model to the original one, $f^\epsilon_{\theta^\epsilon}$, on the training dataset $\tilde{X}$. We prove the following lemma in Appendix C.1:

Lemma 6.3. *Under the premise of Lemma 6.1,* $\forall t \geq 0\ \forall\epsilon \in [0,1]\ \forall x \in \mathbb{R}^d$ *we have*

$$\left\|f^\epsilon_{\theta^\epsilon(t)}(x) - \tilde{f}^\epsilon_{\theta^0(t),\theta^\epsilon(t)}(x)\right\| \leq (L-1)(L+1+\rho(L-1))u^{L-1}(t)v(t)\|x\|\epsilon^2; \tag{27}$$

$$\left\|f^\epsilon_{\theta^\epsilon(t)}(\tilde{X}) - \tilde{f}^\epsilon_{\theta^0(t),\theta^\epsilon(t)}(\tilde{X})\right\| \leq 2(L-1)(1+\rho(L-1))u^{L-1}(t)v(t)\sqrt{m}\epsilon^2. \tag{28}$$

The deviation now scales as ϵ^2 instead of ϵ.

What remains is to bound the size of $\mathcal{F}_2^\epsilon(t;\theta(0))$, which denotes the set of functions representable with our new proxy-model $\tilde{f}^\epsilon_{\theta^0(t),\theta^\epsilon(t)}$ for given initial weights $\theta(0)$. Since $\tilde{f}^\epsilon_{\theta^0(t),\theta^\epsilon(t)}$ is a sum of $f^\epsilon_{\theta^0(t)}$ and a linear model, its size is at most 2^{pd} times larger:

$$|\mathcal{F}_2^\epsilon(t;\theta(0))| \leq |\mathcal{F}_1^\epsilon(t;\theta(0))|2^{pd} \leq 2^{2pd}. \tag{29}$$

This finalizes the proof of Theorem 4.2 for $\kappa = 2$.

6.3 Proof of Lemma 6.1

We are going to present the proof for $L = 2$ in the present section, and defer the case of $L \geq 3$ to Appendix D.1.

6.3.1 Weight norms

Let us expand the weight evolution Equation (3):

$$\frac{dW_1^\tau}{dt} = \left[D^\tau \odot W_2^{\tau,\top}\Xi^\tau\right]\tilde{X}^\top, \qquad \frac{dW_2^\tau}{dt} = \Xi^\tau\left[D^\tau \odot W_1^\tau \tilde{X}\right]^\top, \tag{30}$$

where we define

$$\Xi^\tau = \frac{1}{m}\left(Y - W_2^\tau\left[D^\tau \odot W_1^\tau \tilde{X}\right]\right), \tag{31}$$

and $D^\tau = (\phi^\epsilon)'\left(W_1^\tau \tilde{X}\right)$ is a $n \times m$ matrix with entries equal to 1 or $1-\tau$. We express it as $D^\tau = (1-\tau)1_{n\times m} + \tau\Delta$, where Δ is a $n \times m$ 0-1 matrix.

Let us bound the evolution of weight norms:

$$\frac{d\left\|W_1^\tau\right\|}{dt} \le \left\|\frac{dW_1^\tau}{dt}\right\| \le (1-\tau)\left\|W_2^\tau\right\|\left\|\Xi^\tau\tilde{X}^\top\right\| + \tau\left\|\Delta \odot W_2^{\tau,\top}\Xi^\tau\right\|\left\|\tilde{X}\right\|. \tag{32}$$

Since multiplying by a 0-1 matrix elementwise does not increase Frobenius norm, we get

$$\left\|\Delta \odot W_2^{\tau,\top}\Xi^\tau\right\| \le \left\|\Delta \odot W_2^{\tau,\top}\Xi^\tau\right\|_F \le \|W_2^\tau\|\|\Xi^\tau\|_F. \tag{33}$$

Noting that Ξ^τ and $\Xi^\tau\tilde{X}^\top$ are row matrices, this results in

$$\frac{d\left\|W_1^\tau\right\|}{dt} \le \left((1-\tau)\|\Xi^\tau\tilde{X}^\top\| + \tau\|\Xi^\tau\|\|\tilde{X}\|\right)\|W_2^\tau\|. \tag{34}$$

By a similar reasoning,

$$\frac{d\left\|W_2^\tau\right\|}{dt} \le (1-\tau)\|\Xi^\tau\tilde{X}^\top\|\|W_1^\tau\| + \tau\|\Xi^\tau\|\|W_1^\tau\tilde{X}\|_F. \tag{35}$$

Consider the following system of ODEs:

$$\frac{dg_1(t)}{dt} = \bar{s}^2 g_2(t), \qquad g_1(0) = \beta\bar{s}_\rho; \tag{36}$$

$$\frac{dg_2(t)}{dt} = g_1(t), \qquad g_2(0) = \beta. \tag{37}$$

We then make use of the following lemma which we prove in Appendix C.2:

Lemma 6.4. $\|\Xi_\tau\| = \|\Xi_\tau\|_F \le \frac{1}{\sqrt{m}}(1+\rho\beta^L)$. *If we additionally take Assumption 4.1 then* $\|\Xi_\tau\tilde{X}^\top\| = \|\Xi_\tau\tilde{X}^\top\|_F \le s + \rho\beta^L$.

This lemma implies $g_1(t) \ge (1-\tau)(s+\rho\beta^2)\|W_1^\tau(t)\| + \tau(1+\rho\beta^2)\|W_1^\tau(t)\tilde{X}\|_F$ and $g_2(t) \ge \|W_2^\tau(t)\|$.

The above system of ODEs could be solved analytically:

$$g_1(t) = \beta\sqrt{\bar{s}_\rho - \bar{s}_1}\cosh\left(\bar{s}t + \tanh^{-1}\left(\frac{\bar{s}_1}{\bar{s}_\rho}\right)\right); \tag{38}$$

$$g_2(t) = \sqrt{\bar{\beta}^2 - \beta^2}\sinh\left(\bar{s}t + \tanh^{-1}\left(\frac{\bar{s}_1}{\bar{s}_\rho}\right)\right), \tag{39}$$

where $\bar{\beta} = \beta\frac{\bar{s}_\rho}{\bar{s}_1}$. This gives the bound for the input layer weight norms:

$$\|W_1^\tau(t)\| \le \beta + \bar{s}\int_0^t g_2(t)\,dt = \beta - \bar{\beta} + \sqrt{\bar{\beta}^2 - \beta^2}\cosh\left(\bar{s}t + \tanh^{-1}\left(\frac{\bar{s}_1}{\bar{s}_\rho}\right)\right). \tag{40}$$

For further analysis, we will need simpler-looking bounds. Consider a looser bound: $g_2(t) \leq u(t)$ and $g_1(t) \leq \bar{s}u(t)$, where

$$\frac{du(t)}{dt} = \bar{s}u(t), \qquad u(0) = \beta\frac{\bar{s}_\rho}{\bar{s}_1}. \tag{41}$$

This ODE solves as $u(t) = \bar{\beta}e^{\bar{s}t}$.

So, we have $\|W_2^\tau(t)\| \leq u(t)$. As for the input layer weights, we get

$$\|W_1^\tau(t)\| \leq \beta + \bar{s}\int_0^t u(t)\,dt = \beta - \bar{\beta} + u(t) \tag{42}$$

Similarly, $\frac{1}{\sqrt{m}}\|W_1^\tau(t)\tilde{X}\|_F = \beta\rho - \bar{\beta} + u(t)$. As we do not want additive terms, we bound from above: $\|W_1^\tau(t)\| \leq u(t)$ and $\frac{1}{\sqrt{m}}\|W_1^\tau(t)\tilde{X}\| \leq \rho u(t)$.

6.3.2 Norms of weight derivatives

In Appendix C.3, we follow the same logic to demonstrate that $\left\|\frac{dW_1^\tau}{d\tau}\right\|$, $\frac{1}{\sqrt{m}}\left\|\frac{dW_1^\tau}{d\tau}\tilde{X}\right\|_F$, and $\left\|\frac{dW_2^\tau}{d\tau}\right\|$ are all bounded by the same v which satisfies

$$\frac{dv}{dt} = v\left[(1+\rho)u^2 + \bar{s}\right] + u\left[\bar{1} + \rho u^2\right], \quad v(0) = 0. \tag{43}$$

It is a linear ODE in $v(t)$, and $u(t)$ is given: $u(t) = \bar{\beta}e^{\bar{s}t}$. We solve it in Appendix C.4 to get $v(t)$ from Theorem 4.2.

7 Conclusion

We have derived a novel generalization bound for LekyReLU networks. Our bound could be evaluated before the actual training and does not depend on network width. Our bound becomes non-vacuous for partially-trained nets with activation functions close to being linear.

Broader Impact Statement

This is a theoretical work studying generalization in neural nets; its main result is a form of a generalization guarantee. Having a good generalization guarantee is necessary to make the models we use in practice more reliable. We do not foresee any negative societal or ethical impact our research could potentially cause.

References

Sanjeev Arora, Simon Du, Wei Hu, Zhiyuan Li, and Ruosong Wang. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. In *International Conference on Machine Learning*, pp. 322–332, 2019.

Peter L Bartlett, Dylan J Foster, and Matus J Telgarsky. Spectrally-normalized margin bounds for neural networks. In *Advances in Neural Information Processing Systems*, pp. 6240–6249, 2017.

Felix Biggs and Benjamin Guedj. Non-vacuous generalisation bounds for shallow neural networks. In *International Conference on Machine Learning*, pp. 1963–1981. PMLR, 2022.

Lenaic Chizat, Edouard Oyallon, and Francis Bach. On lazy training in differentiable programming. arxiv e-prints, page. *arXiv preprint arXiv:1812.07956*, 2018.

Gintare Karolina Dziugaite and Daniel M Roy. Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. *arXiv preprint arXiv:1703.11008*, 2017.

Tomer Galanti, Liane Galanti, and Ido Ben-Shaul. Comparative generalization bounds for deep neural networks. *Transactions on Machine Learning Research*, 2023.

Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in neural information processing systems*, pp. 8571–8580, 2018.

Arthur Jacot, François Ged, Franck Gabriel, Berfin Şimşek, and Clément Hongler. Deep linear networks dynamics: Low-rank biases induced by initialization scale and l2 regularization. *arXiv preprint arXiv:2106.15933*, 2021.

Mufan Li, Mihai Nica, and Dan Roy. The neural covariance sde: Shaped infinite depth-and-width networks at initialization. *Advances in Neural Information Processing Systems*, 35:10795–10808, 2022.

Zhiyuan Li, Yuping Luo, and Kaifeng Lyu. Towards resolving the implicit bias of gradient descent for matrix factorization: Greedy low-rank learning. *arXiv preprint arXiv:2012.09839*, 2020.

Behnam Neyshabur, Srinadh Bhojanapalli, and Nathan Srebro. A PAC-bayesian approach to spectrally-normalized margin bounds for neural networks. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=Skz_WfbCZ`.

Lorenzo Noci, Chuning Li, Mufan Bill Li, Bobby He, Thomas Hofmann, Chris Maddison, and Daniel M Roy. The shaped transformer: Attention models in the infinite depth-and-width limit. *arXiv preprint arXiv:2306.17759*, 2023.

Andrew M Saxe, James L McClelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *arXiv preprint arXiv:1312.6120*, 2013.

Zhenfeng Tu, Santiago Aranguri, and Arthur Jacot. Mixed dynamics in linear networks: Unifying the lazy and active regimes. *Advances in Neural Information Processing Systems*, 2024.

V N Vapnik and A Ya Chervonenkis. O ravnomernoj skhodimosti chastot pojavlenija sobytiy k ih veroyatnostiam. *Teorija verojatnostiej i jejo primenenija*, 16(2):264–279, 1971.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Greg Yang and Etai Littwin. Tensor programs iib: Architectural universality of neural tangent kernel training dynamics. *arXiv preprint arXiv:2105.03703*, 2021.

Chulhee Yun, Shankar Krishnan, and Hossein Mobahi. A unifying view on implicit bias in training linear neural networks. *arXiv preprint arXiv:2010.02501*, 2020.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*, 2016.

Wenda Zhou, Victor Veitch, Morgane Austern, Ryan P. Adams, and Peter Orbanz. Non-vacuous generalization bounds at the imagenet scale: a PAC-bayesian compression approach. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=BJgqqsAct7`.

A Discussion

A.1 Assumptions

Whitened data. Overall, the assumption on whitened data is not necessary for a result similar to Theorem 4.2 to hold. We assume the data to be whitened for two reasons. First, it legitimates the choice of training time $t^*_\alpha(\beta)$ since it is based on the analysis of Saxe et al. (2013), which assumed the data to be

whitened. If we dropped it, we could still evaluate the bound of Theorem 4.2 at $t = t^*_\alpha(\beta)$, but it would be less clear whether the trainining risk $\hat{R}_\gamma$ becomes already small by this time.

Second, we had to bound $\|\tilde{X}\|$ throughout the proof of Theorem 4.2 and $\|\tilde{X}^+\|$ in the proof of Lemma 6.3. For whitened data, these are simply $\sqrt{m}$ and $1/\sqrt{m}$, which is a clear dependence on m, making the final bound look cleaner. Otherwise, they would be random variables whose dependence on m would be not obvious.

Third, if we considered training on the original dataset (Y, X) instead of the whitened one, $(Y, \tilde{X})$, we would have to know $YX^\top$ and $XX^\top$ in order to determine $\theta^0(t)$ for a given t and initialization $\theta^0(0)$. These two matrices have $d + \frac{d(d+1)}{2}$ parameters, compared to just d for $Y\tilde{X}^\top$. This way, Υ_κ would grow as d instead of $\sqrt{d}$.

Gradient flow. We expect our technique to follow through smoothly for finite-step gradient descent. Introducing momentum also seems doable. However, generalizing it to other training procedures, e.g. the ones which use normalized gradients, might pose problems since it is not clear how to reasonably upper-bound the norm of the elementwise ratio of two matrices.

Assumption 4.1. We use this assumption to prove the second part of Lemma 6.4. If we dropped it, the bound would be $\|\Xi^\tau \tilde{X}^\top\|_F \leq \|\Xi^\tau\|_F \|\tilde{X}^\top\| \leq 1 + \rho\beta^L$ instead of $s + \rho\beta^L$. This would result in larger exponents for the definition u in Theorem 4.2.

As an argument in favor of this assumption, we demonstrate it empirically first (Figure 7), and second, we prove it for the linear case, see Appendix E.

A.2 Proof

We expect the bounds on weight norms $u(t)$ to be quite loose since we use Lemma 6.4 to bound the loss. This lemma bounds the loss with its value at the initialization, while the loss should necessarily decrease. If we could account for the loss decrease, the resulting $u(t)$ would increase with a lower exponent, or even stay bounded as $\bar{u}(t)$, which corresponds to a linear model, does. This way, we would not have to assume ϵ to vanish as β vanishes in order to keep the bound non-diverging for small β at the training time $t^*_\alpha(\beta)$. Also, the general bound of Theorem 4.2 would diverge with training time t much slower. We leave it for future work.

As we see from our estimates, Υ_κ becomes the main bottleneck of our bound for small ϵ. The bound we used for Υ_κ is very naive; we believe that better bounds are possible. Improving the bound for Υ_κ will increase the maximal ϵ for which our bound is non-vacuous.

A.3 Other architectures

As becomes apparent from the proof in Appendix C.1, the proxy-model for $\kappa = 2$, Equation (26), deviates from the original model $f^\epsilon_{\theta^\epsilon}$ as $O(\epsilon^2)$ for any map $(\epsilon, \theta, x) \to f^\epsilon_\theta(x)$ as long as the following holds:

1. $f^0_\theta(x)$ is linear in x for any θ;
2. $\frac{\partial^2 f^\epsilon_\theta(x)}{\partial\epsilon\partial\theta}$ is continuous as a function of (ϵ, θ, x);
3. the result of learning $\theta^\epsilon(t)$ is differentiable in ϵ for any t.

This is directly applicable to convolutional networks with no other nonlinearities except for ReLU's; in partiular, without max-pooling layers. One may introduce max-poolings by interpolating between average-poolings (which are linear) for $\epsilon = 0$ and max-poolings for $\epsilon = 1$. This is not applicable to Transformers (Vaswani et al., 2017) since attention layers are inherently nonlinear: queries and keys have to be multiplied.

Compared to the fully-connected case of the present work, our bound might become even tighter for convolutional nets since d becomes the number of color channels (up to 3) instead of the whole image size in pixels. However, the corresponding proxy-models might be over-simplistic: the linear net they will deviate

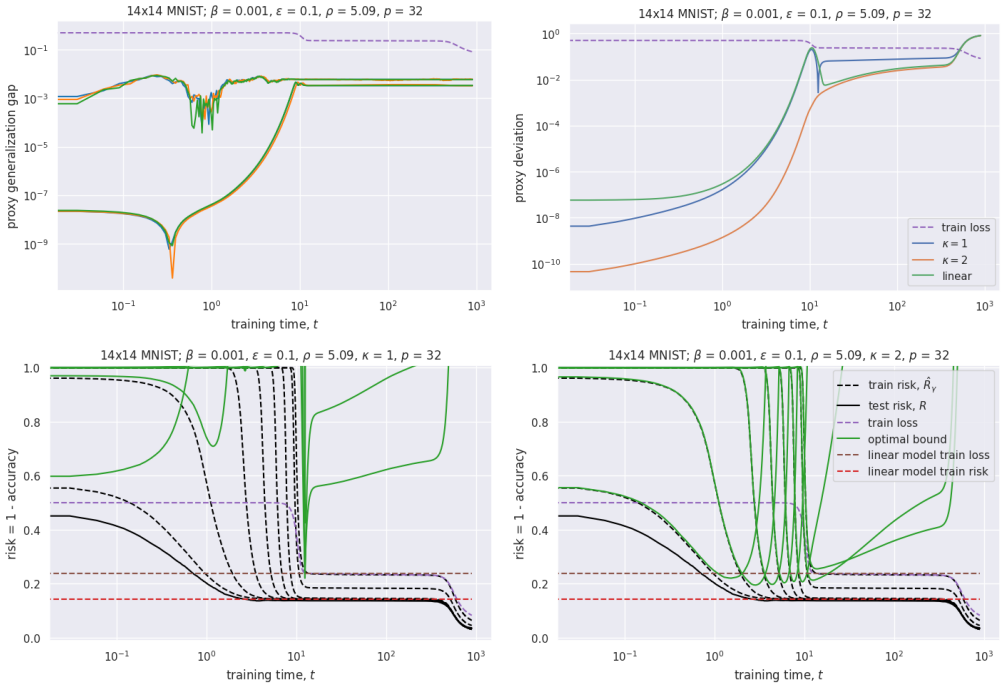

Figure 8: We examine the optimistic bound of Equation (44) for the proxy-models proposed in our paper: the $\kappa = 1$ one, $f^{\epsilon}_{\theta^0}$, the $\kappa = 2$ one of Equation (26), and also a linear proxy, $f^{0}_{\theta^0}$. The bottom row demonstrates the full bound (green lines), while the top one depicts the two components of the bound, namely, the proxy model generalization gap $R^C_\gamma(g) - \hat{R}^C_\gamma(g)$ and the proxy model deviation $\mathbb{E}\,|f(x) - g(x)| + \hat{\mathbb{E}}\,|f(x) - g(x)|$, separately. Different lines of the same color (e.g. solid green and dashed black lines on the bottom row) correspond to different values of γ. Proxy generalization gap stays low during the whole training (top left figure), while the train risk and the model deviation over gamma contribute significantly (bottom row, two groups of lines correspond to the minimal and the maximal γ we considered). The optimistic bound for the 1st-order proxy (bottom left) gets non-vacuous only at the moment when GF escapes the origin and reaches the linear model loss. The bound for the 2nd-order proxy (bottom right) becomes non-vacuous soon after the original model becomes non-vacuous (but still stays near the origin), and stays non-vacuous until the model starts exploiting its nonlinearity to reduce the loss below the optimal linear model loss level (the last drop of purple and black lines).

from is just a global average-pooling followed by a linear $\mathbb{R}^d \to \mathbb{R}$ map. We leave exploring the convolutional net case for future work.

B How far could we get with our approach?

Broadly speaking, the approach we apply in this work could be described as follows. Suppose we have a model f learned on a dataset (X, Y), and another "proxy" model g which we construct having an access to the same dataset (X, Y). We bound the distribution risk of the "original" model using the following relation:

$$R(f) - \hat{R}_\gamma(f) \leq R^C_\gamma(f) - \hat{R}^C_\gamma(f) \leq R^C_\gamma(g) - \hat{R}^C_\gamma(g) + \frac{\mathbb{E}_{x\sim\mathcal{D}}|g(x) - f(x)| + \frac{1}{m}\|g(X) - f(X)\|_1}{\gamma}. \tag{44}$$

In words, we say that performance of f is worse than that of g at most by some deviation term.

The bound ends up to be good whenever (a) the generalization gap of g could be well-bounded, and (b) g does not deviate from f much. That is why we considered proxy-models based on linear learned weights: their generalization gap could be easily bounded analytically, and they do not deviate much from corresponding leaky ReLU nets as long as ReLU negative slopes are close to one.

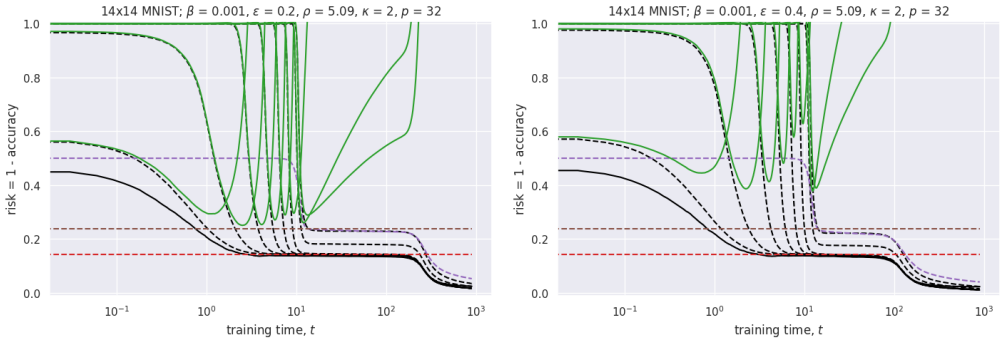

Figure 9: The optimistic bound of Equation (44) based on our $\kappa = 2$ proxy stays non-vacuous up to $\epsilon = 0.4$ until the gradient flow starts "exploiting" the nonlinearity (the last drop of purple and black lines).

The biggest conceptual disadvantage of this approach is that, given both f and g learn the training dataset, we have no chance proving that f performs better than g, we could only prove that f performs *not much worse* than g. Do the proxy-models we use in the present paper perform well, and how much do they deviate from original models? Our main theoretical result, Theorem 4.2, bounds the proxy-model generalization gap and the deviation from above. These bounds are arguably not optimal. It is therefore instructive to examine how well the bound would perform, if we could estimate Equation (44) exactly.

B.1 Empirical validation

Setup. We work under the same setup as in Section 5[5], but instead of evaluating the bound of Theorem 4.2, we actually train a linear model with exactly the same procedure as for the original model, in order to get trained linear weights θ^0. We then evaluate the proxy-models considered in the present work: (1) the one for $\kappa = 1$, $f^{\epsilon}_{\theta^0}$, (2) the one for $\kappa = 2$, see Equation (26), and also (3) the linear network, $f^{0}_{\theta^0}$. We then evaluate the rhs of Equation (44) using a test part of the MNIST dataset. For this "optimistic" bound, we consider much larger values of epsilon: $\epsilon \geq 0.1$, i.e. our model much less "nearly-linear" now than before.

Figures. We present the results on Figures 8 and 9. Dashed lines correspond to the train set, while solid ones correspond to the test set. Black lines are risks of the actual trained model $f^{\epsilon}_{\theta^{\epsilon}(t)}$: $R(t)$ and $\hat{R}_{\gamma}(t)$, respectively. Green lines are our "optimistic" bound Equation (44) evaluated at different values of γ. Purple lines denote MSE loss of the actual trained model.

We also put three baselines on the plots. The dotted black line is the classification risk (and the MSE loss) of a zero model $f \equiv 0$. The brown dashed line is the MSE (train) loss of the optimal linear model, $f : x \to Y\tilde{X}^{+}x$. Finally, the red dashed line is the (train) classification risk of the optimal linear model.

Training phases. As we observe on risk plots (Figure 9 and the bottom row of Figure 8), the training process could be divided into three phases. During the first phase, the risk decreases until it reaches the risk of the optimal linear model, while the loss stays at the level of $f \equiv 0$. This indicates that while the weights stay very close to the origin, the network outputs already align with the outputs of the optimal linear model. During the second phase, both the loss and the risk stay at the level of the optimal linear model. As for the following phase, both the risk and the loss drop below the optimal linear model level. Therefore from this point on, the network starts to "exploit" its nonlinearity in order to reduce the train loss.

Observations:

- The generalization gap stays negligible for all models and γ's considered (Figure 8, top left);

[5]We also downsample MNIST images to 14x14 instead of 7x7. The reason why we do it is that on one hand, we wanted to test our bounds on more realistic scenarios, while on the other, X does not appear to be full-rank for the original 28x28 MNIST.

- The proxy-model for $\kappa = 2$ approximates the original model best among all three proxies considered (Figure 8, top right);
- While the linear approximation deviates from the original model more than the one for $\kappa = 2$ during the first phase, their deviations are similar during the subsequent phases;
- At the same time, the $\kappa = 1$ approximation deviates more than that of $\kappa = 2$ during the second phase;
- The transition between the first and the second phases results in a nonmonotonic behavior of the deviation from the original model for $\kappa = 1$ and linear proxy-models;
- The resulting optimistic bound for $\kappa = 2$ (green lines of Figure 8, bottom right) stays non-vacuous during the first two phases for $\epsilon = 0.1$, while this is not the case for $\kappa = 1$ (green lines of Figure 8, bottom left);
- The optimistic bound for $\kappa = 2$ stays non-vacuous up to $\epsilon = 0.4$ (green lines of Figure 9).

It is tempting to assume that the weights θ^ϵ follow the same trajectory as the weights of the linear model, θ^0, during the first two phases. However, if it was the case, the $\kappa = 1$ proxy-model, $f^\epsilon_{\theta^0}$, would coincide with the original one, $f^\epsilon_{\theta^\epsilon}$, during this period. Then their quality would be the same; however, Figure 8, top right, demonstrates the opposite.

C Missing calculations in Section 6

C.1 Proof of Lemma 6.3

We have:

$$\frac{1}{\epsilon^2}\left\|f^\epsilon_{\theta^\epsilon}(x) - \tilde{f}^\epsilon_{\theta^0,\theta^\epsilon}(x)\right\| = \frac{1}{\epsilon^2}\left\|\int_0^\epsilon \left(\frac{\partial f^\epsilon_{\theta^\tau}(x)}{\partial\tau} - \frac{\partial f^\epsilon_{\theta^\tau}(\tilde{X})\tilde{X}^+x}{\partial\tau}\right) d\tau\right\| \le \frac{1}{\epsilon}\sup_{\tau\in[0,\epsilon]}\left\|\frac{\partial f^\epsilon_{\theta^\tau}(x)}{\partial\tau} - \frac{\partial f^\epsilon_{\theta^\tau}(\tilde{X})}{\partial\tau}\tilde{X}^+x\right\|. \tag{45}$$

Since $f^0_{\theta^\tau}$ is a linear network and $\operatorname{rk}\tilde{X} = d$, we have $f^0_{\theta^\tau}(x) = f^0_{\theta^\tau}(\tilde{X})\tilde{X}^+x$ and $\frac{\partial f^0_{\theta^\tau}(x)}{\partial\tau} = \frac{\partial f^0_{\theta^\tau}(\tilde{X})\tilde{X}^+x}{\partial\tau}$. This implies

$$\begin{aligned}\frac{1}{\epsilon}\left\|\frac{\partial f^\epsilon_{\theta^\tau}(x)}{\partial\tau} - \frac{\partial f^\epsilon_{\theta^\tau}(\tilde{X})}{\partial\tau}\tilde{X}^+x\right\| &= \frac{1}{\epsilon}\left\|\int_0^\epsilon \left(\frac{\partial^2 f^\rho_{\theta^\tau}(x)}{\partial\rho\partial\tau} - \frac{\partial^2 f^\rho_{\theta^\tau}(\tilde{X})\tilde{X}^+x}{\partial\rho\partial\tau}\right) d\rho\right\| \\ &\le \sup_{\rho\in[0,\epsilon]}\left\|\frac{\partial^2 f^\rho_{\theta^\tau}(x)}{\partial\rho\partial\tau} - \frac{\partial^2 f^\rho_{\theta^\tau}(\tilde{X})}{\partial\rho\partial\tau}\tilde{X}^+x\right\| \\ &\le \sup_{\rho\in[0,\epsilon]}\left\|\frac{\partial^2 f^\rho_{\theta^\tau}(x)}{\partial\rho\partial\tau}\right\| + \sup_{\rho\in[0,\epsilon]}\left\|\frac{\partial^2 f^\rho_{\theta^\tau}(\tilde{X})}{\partial\rho\partial\tau}\right\|\|\tilde{X}^+x\|.\end{aligned} \tag{46}$$

Since we use LeakyReLU,

$$\left\|\frac{\partial^2 f^\rho_{\theta^\tau}(x)}{\partial\rho\partial\tau}\right\| \le (L-1)\sum_{l=1}^L\left\|\frac{\partial W^\tau_l}{\partial\tau}\right\|\prod_{k\neq l}\|W^\tau_k\|\|x\|; \tag{47}$$

$$\left\|\frac{\partial^2 f^\rho_{\theta^\tau}(\tilde{X})}{\partial\rho\partial\tau}\right\| \le (L-1)\left\|\frac{\partial W^\tau_1\tilde{X}}{\partial\tau}\right\|_F\prod_{k\in[2,L]}\|W^\tau_k\| + (L-1)\|W^\tau_1\tilde{X}\|_F\sum_{l=2}^L\left\|\frac{\partial W^\tau_l}{\partial\tau}\right\|\prod_{k\in[2,L]\setminus\{l\}}\|W^\tau_k\|. \tag{48}$$

Finally, since $\tilde{X}\tilde{X}^\top = mI_d$, we have $\|\tilde{X}^+\| = \frac{1}{\sqrt{m}}$. Combining everything together, we arrive into

$$\begin{aligned}\frac{\left\|f^\epsilon_{\theta^\epsilon}(x) - \tilde{f}^\epsilon_{\theta^0,\theta^\epsilon}(x)\right\|}{(L-1)\epsilon^2\|x\|} &\leq \sup_{\tau\in[0,\epsilon]}\left[\sum_{l=1}^{L}\left\|\frac{\partial W_l^\tau}{\partial\tau}\right\|\prod_{k\neq l}\|W_k^\tau\|\right] \\ &+ \sup_{\tau\in[0,\epsilon]}\left[\frac{1}{\sqrt{m}}\|W_1^\tau\tilde{X}\|_F\sum_{l=2}^{L}\left\|\frac{\partial W_l^\tau}{\partial\tau}\right\|\prod_{k\in[2,L]\setminus\{l\}}\|W_k^\tau\|\right] \\ &+ \sup_{\tau\in[0,\epsilon]}\left[\frac{1}{\sqrt{m}}\left\|\frac{\partial W_1^\tau\tilde{X}}{\partial\tau}\right\|_F\prod_{k\in[2,L]}\|W_k^\tau\|\right].\end{aligned} \tag{49}$$

Plugging the bounds from Lemma 6.1 then gives

$$\left\|f^\epsilon_{\theta^\epsilon}(x) - \tilde{f}^\epsilon_{\theta^0,\theta^\epsilon}(x)\right\| \leq (L-1)(L+1+(L-1)\rho)u^{L-1}v\|x\|\epsilon^2. \tag{50}$$

By a similar reasoning,

$$\begin{aligned}\frac{\left\|f^\epsilon_{\theta^\epsilon}(\tilde{X}) - \tilde{f}^\epsilon_{\theta^0,\theta^\epsilon}(\tilde{X})\right\|}{(L-1)\epsilon^2} &\leq 2\sup_{\tau\in[0,\epsilon]}\left[\|W_1^\tau\tilde{X}\|_F\sum_{l=2}^{L}\left\|\frac{\partial W_l^\tau}{\partial\tau}\right\|\prod_{k\in[2,L]\setminus\{l\}}\|W_k^\tau\|\right] \\ &+ 2\sup_{\tau\in[0,\epsilon]}\left[\left\|\frac{\partial W_1^\tau\tilde{X}}{\partial\tau}\right\|_F\prod_{k\in[2,L]}\|W_k^\tau\|\right].\end{aligned} \tag{51}$$

Plugging the same bounds,

$$\left\|f^\epsilon_{\theta^\epsilon}(\tilde{X}) - \tilde{f}^\epsilon_{\theta^0,\theta^\epsilon}(\tilde{X})\right\| \leq 2(L-1)(1+(L-1)\rho)u^{L-1}v\epsilon^2\sqrt{m}. \tag{52}$$

C.2 Proof of Lemma 6.4

Recall the definition of Ξ^τ:

$$\Xi^\tau(t) = \frac{1}{m}\left(Y - W_L^\tau(t)\left[D_{L-1}^\tau(t)\odot W_{L-1}^\tau(t)\left[\ldots W_2^\tau(t)\left[D_1^\tau(t)\odot W_1^\tau(t)\tilde{X}\right]\right]\right]\right). \tag{53}$$

Since $m\|\Xi^\tau\|_F^2$ is the loss function we optimize, it does not increase under gradient flow. Hence, since multiplying by D_l^τ elementwise does not increase Frobenius norm,

$$m\|\Xi^\tau(t)\|_F \leq m\|\Xi^\tau(0)\|_F \leq \|Y\|_F + \|W_1^\tau(0)\tilde{X}\|_F\prod_{l=2}^{L}\|W_l^\tau(0)\|. \tag{54}$$

We have $\|W_1^\tau(0)\tilde{X}\|_F = \rho\|W_1^\tau(0)\tilde{X}\| \leq \|W_1^\tau(0)\|\|\tilde{X}\|$. Since all y from $\operatorname{supp}\mathcal{D}$ are ±1 and $\tilde{X}\tilde{X}^\top = mI_d$, we get $\|\Xi^\tau(t)\|_F = \frac{1}{\sqrt{m}}(1+\rho\beta^L)$.

Due to Assumption 4.1, $m\|\Xi^\tau\tilde{X}^\top\|_F^2$ does not increase under gradient flow:

$$m\|\Xi^\tau(t)\tilde{X}^\top\|_F \leq m\|\Xi^\tau(0)\tilde{X}^\top\|_F \leq \|Y\tilde{X}^\top\|_F + \|\tilde{X}\|\|W_1^\tau(0)\tilde{X}\|_F\prod_{l=2}^{L}\|W_l^\tau(0)\|. \tag{55}$$

Since $Y\tilde{X}^+ = s$, we have $Y\tilde{X}^\top = ms$. Therefore $\|\Xi^\tau(t)\tilde{X}^\top\|_F = s + \rho\beta^L$.

Finally, since $\Xi^\tau \in \mathbb{R}^{1\times m}$, $\|\Xi^\tau\| = \|\Xi^\tau\|_F$ and $\|\Xi^\tau\tilde{X}^\top\| = \|\Xi^\tau\tilde{X}^\top\|_F$.

C.3 Bounding derivatives in the proof of Lemma 6.1

Let us start with W_1^τ:

$$\frac{d^2W_1^\tau}{dtd\tau} = \left[D^\tau \odot \left(\frac{dW_2^{\tau,\top}}{d\tau}\Xi^\tau + W_2^{\tau,\top}\frac{d\Xi^\tau}{d\tau}\right)\right]\tilde{X}^\top - \left[\bar{\Delta}\odot W_2^{\tau,\top}\Xi^\tau\right]\tilde{X}^\top; \tag{56}$$

$$\left\|\frac{d^2W_1^\tau}{dtd\tau}\right\| \le \left\|\frac{dW_2^\tau}{d\tau}\right\|\left((1-\tau)\|\Xi^\tau\tilde{X}^\top\| + \tau\|\Xi^\tau\|_F\|\tilde{X}\|\right) + \|W_2^\tau\|\left(\left\|\frac{d\Xi^\tau}{d\tau}\right\|_F + \|\Xi^\tau\|_F\right)\|\tilde{X}\|. \tag{57}$$

We need a bound for $\left\|\frac{d\Xi^\tau}{d\tau}\right\|$:

$$m\frac{d\Xi^\tau}{d\tau} = W_2^\tau\left[\bar{\Delta}\odot W_1^\tau\tilde{X}\right] - W_2^\tau\left[D^\tau\odot\frac{dW_1^\tau}{d\tau}\tilde{X}\right] - \frac{dW_2^\tau}{d\tau}\left[D^\tau\odot W_1^\tau\tilde{X}\right]; \tag{58}$$

$$\begin{aligned} m\left\|\frac{d\Xi^\tau}{d\tau}\right\| &\le \|W_2^\tau\|\|W_1^\tau\tilde{X}\|_F + \|W_2^\tau\|\left\|\frac{dW_1^\tau}{d\tau}\tilde{X}\right\|_F + \left\|\frac{dW_2^\tau}{d\tau}\right\|\|W_1^\tau\tilde{X}\|_F \\ &\le u^2\rho\sqrt{m} + u\left\|\frac{dW_1^\tau}{d\tau}\tilde{X}\right\|_F + u\rho\sqrt{m}\left\|\frac{dW_2^\tau}{d\tau}\right\|. \end{aligned} \tag{59}$$

This results in

$$\frac{d}{dt}\left\|\frac{dW_1^\tau}{d\tau}\right\| \le \left\|\frac{d^2W_1^\tau}{dtd\tau}\right\| \le u\left(1+\rho\beta^2+\rho u^2 + u\frac{1}{\sqrt{m}}\left\|\frac{dW_1^\tau}{d\tau}\tilde{X}\right\|_F\right) + \left\|\frac{dW_2^\tau}{d\tau}\right\|(s+(1-s)\tau+\rho\beta^2+\rho u^2). \tag{60}$$

Similarly, we have an evolution of $W_1^\tau\tilde{X}$:

$$\begin{aligned} \frac{d}{dt}\left\|\frac{dW_1^\tau}{d\tau}\tilde{X}\right\|_F &\le \left\|\frac{dW_2^\tau}{d\tau}\right\|\left((1-\tau)\|\Xi^\tau\tilde{X}^\top\|_F\|\tilde{X}\| + \tau\|\Xi^\tau\|_F\|\tilde{X}^\top\tilde{X}\|\right) + \|W_2^\tau\|\left(\left\|\frac{d\Xi^\tau}{d\tau}\right\|_F + \|\Xi^\tau\|_F\right)\|\tilde{X}^\top\tilde{X}\| \\ &\le u\left(1+\rho\beta^2+\rho u^2 + u\frac{1}{\sqrt{m}}\left\|\frac{dW_1^\tau}{d\tau}\tilde{X}\right\|_F\right)\sqrt{m} + \left\|\frac{dW_2^\tau}{d\tau}\right\|(s+(1-s)\tau+\rho\beta^2+\rho u^2)\sqrt{m}. \end{aligned} \tag{61}$$

Finally, consider W_2^τ:

$$\frac{d^2W_2^\tau}{dtd\tau} = \frac{d\Xi^\tau}{d\tau}\left[D^\tau\odot W_1^\tau\tilde{X}\right]^\top + \Xi^\tau\left[D^\tau\odot\frac{dW_1^\tau}{d\tau}\tilde{X} - \bar{\Delta}\odot W_1^\tau\tilde{X}\right]^\top; \tag{62}$$

$$\left\|\frac{d^2W_2^\tau}{dtd\tau}\right\| \le \left\|\frac{dW_1^\tau}{d\tau}\right\|\left((1-\tau)\|\Xi^\tau\tilde{X}^\top\| + \tau\|\Xi^\tau\|_F\|\tilde{X}\|\right) + \left(\|\Xi^\tau\| + \left\|\frac{d\Xi^\tau}{d\tau}\right\|\right)\|W_1^\tau\tilde{X}\|_F; \tag{63}$$

$$\begin{aligned} \frac{d}{dt}\left\|\frac{dW_2^\tau}{d\tau}\right\| &\le \left\|\frac{d^2W_2^\tau}{dtd\tau}\right\| \\ &\le u\left(1+\rho\beta^2+\rho u^2 + u\frac{1}{\sqrt{m}}\left\|\frac{dW_1^\tau}{d\tau}\tilde{X}\right\|_F\right) + \left\|\frac{dW_1^\tau}{d\tau}\right\|(s+(1-s)\tau+\rho\beta^2) + \rho u^2\left\|\frac{dW_2^\tau}{d\tau}\right\|. \end{aligned} \tag{64}$$

We see that $\left\|\frac{dW_1^\tau}{d\tau}\right\|$, $\frac{1}{\sqrt{m}}\left\|\frac{dW_1^\tau}{d\tau}\tilde{X}\right\|_F$, and $\left\|\frac{dW_2^\tau}{d\tau}\right\|$ are all bounded by the same v which satisfies

$$\frac{dv(t)}{dt} = v(t)(\bar{s} + (1+\rho)u^2(t)) + u(t)(\bar{1} + \rho u^2(t)), \qquad v(0) = 0, \tag{65}$$

where $\bar{s} = s + (1-s)\tau + \rho\beta^2$ and $\bar{1} = 1 + \rho\beta^2$.

C.4 Solving the ODE for $v(t)$

Recall $u(t) = \bar{\beta} e^{\bar{s}t}$. We solve the homogeneous equation to get

$$v(t) = C(t) e^{\bar{s}t + \frac{1+\rho}{2\bar{s}} \bar{\beta}^2 e^{2\bar{s}t}} = C(t) e^{(L-1)\ln u(t) + \frac{1+(L-1)\rho}{\bar{s}L} u^L(t)} = C(t) \bar{\beta}^{-1} u(t) e^{\frac{1+\rho}{2\bar{s}} u^2(t)}, \tag{66}$$

where $C(t)$ satisfies

$$\frac{dC(t)}{dt} u(t) e^{\frac{1+\rho}{2\bar{s}} u^2(t)} = \bar{\beta} u(t) \left[\bar{1} + \rho u^2(t) \right]. \tag{67}$$

Recall for $L = 2$, $\hat{s} = \frac{2}{1+\rho} \bar{s}$. Then

$$\begin{aligned} C(t) &= \bar{\beta} \int e^{-\frac{u^2(t)}{\hat{s}}} \left[\bar{1} + \rho u^2(t) \right] dt \\ &= \frac{\bar{\beta}}{2} \left[\frac{\bar{1}}{\bar{s}} \left(\mathrm{Ei}\left(-\frac{u^2(t)}{\hat{s}} \right) - \mathrm{Ei}\left(-\frac{\bar{\beta}^2}{\hat{s}} \right) \right) - \rho \frac{\hat{s}}{\bar{s}} \left(e^{-\frac{u^2(t)}{\hat{s}}} - e^{-\frac{\bar{\beta}^2}{\hat{s}}} \right) \right]. \end{aligned} \tag{68}$$

This gives the final solution:

$$v(t) = \frac{1}{2} u(t) [w(u(t)) - w(\beta)] e^{\frac{u^2(t)}{\hat{s}}}, \tag{69}$$

where we took

$$w(u) = \frac{\bar{1}}{\bar{s}} \mathrm{Ei}\left(-\frac{u^2(t)}{\hat{s}} \right) - \frac{2\rho}{1+\rho} e^{-\frac{u^2(t)}{\hat{s}}}. \tag{70}$$

D Deep networks

D.1 Proof of Lemma 6.1 for $L \geq 3$

Bounding weight norms. For $l \in [2, L]$,

$$\frac{dW_l^\tau}{dt} = \left[D_l^\tau \odot W_{l+1}^{\tau,\top} \dots \left[D_{L-1}^\tau \odot W_L^{\tau,\top} \Xi^\tau \right] \right] \left[D_{l-1}^\tau \odot W_{l-1}^\tau \dots \left[D_1^\tau \odot W_1^\tau \tilde{X} \right] \right]^\top; \tag{71}$$

$$\left\| \frac{dW_l^\tau}{dt} \right\| \leq \left((1-\tau) \|\Xi^\tau \tilde{X}^\top\| \|W_1^\tau\| + \tau(L-1) \|\Xi^\tau\|_F \|W_1^\tau \tilde{X}\|_F \right) \prod_{k \in [2,L] \setminus \{l\}} \|W_k^\tau\|. \tag{72}$$

For $l = 1$,

$$\frac{dW_1^\tau}{dt} = \left[D_1^\tau \odot W_2^{\tau,\top} \dots \left[D_{L-1}^\tau \odot W_L^{\tau,\top} \Xi^\tau \right] \right] \tilde{X}^\top; \tag{73}$$

$$\left\| \frac{dW_1^\tau}{dt} \right\| \leq \left((1-\tau) \|\Xi^\tau \tilde{X}^\top\| + \tau(L-1) \|\Xi^\tau\|_F \|\tilde{X}\| \right) \prod_{k \in [2,L]} \|W_k^\tau\|; \tag{74}$$

$$\left\| \frac{dW_1^\tau \tilde{X}}{dt} \right\|_F \leq \left((1-\tau) \|\Xi^\tau \tilde{X}^\top\|_F \|\tilde{X}\| + \tau(L-1) \|\Xi^\tau\|_F \|\tilde{X}^\top \tilde{X}\| \right) \prod_{k \in [2,L]} \|W_k^\tau\|. \tag{75}$$

Using Lemma 6.4, we get $(1-\tau)\|\Xi^\tau \tilde{X}^\top\| \|W_1^\tau\| + \tau(L-1)\|\Xi^\tau\| \|W_1^\tau \tilde{X}\|_F \leq g_1(t)$ and $\|W_l^\tau\| \leq g_2(t)$ $\forall l \in [2:L]$, where

$$\frac{dg_1(t)}{dt} = \bar{s}^2 g_2^{L-1}(t), \quad \frac{dg_2(t)}{dt} = g_1(t) g_2^{L-2}(t), \qquad g_1(0) = \beta((1-\tau)(s+\beta^L) + \tau(L-1)(1+\beta^L)\rho), \quad g_2(0) = \beta. \tag{76}$$

We have the following first integral:

$$\frac{d}{dt} \left(g_1^2(t) - \bar{s}^2 g_2^2(t) \right) = 0, \qquad g_1^2(0) - \bar{s}^2 g_2^2(0) = \beta^2 \left(((1-\tau)(s+\beta^L) + \tau(L-1)(1+\beta^L)\rho)^2 - \bar{s}^2 \right). \tag{77}$$

Therefore

$$\frac{dg_2(t)}{dt} = g_2^{L-2}(t)\sqrt{\bar{s}^2 g_2^2(t) - \bar{s}^2 g_2^2(0) + g_1^2(0)}, \qquad g_2(0) = \beta. \tag{78}$$

Suppose $\rho > 1$. Then $g_1^2(0) - \bar{s}^2 g_2^2(0) > 0$ and the solution is given in the following implicit form:

$$t = \frac{g_2^{3-L}(t)\,{}_2F_1\left(\frac{1}{2}, \frac{3-L}{2}, \frac{5-L}{2}, -\frac{\bar{s}^2 g_2^2(t)}{g_1^2(0)-\bar{s}^2\beta^2}\right) - \beta_2^{3-L} F_1\left(\frac{1}{2}, \frac{3-L}{2}, \frac{5-L}{2}, -\frac{\bar{s}^2\beta^2}{g_1^2(0)-\bar{s}^2\beta^2}\right)}{(3-L)\sqrt{g_1^2(0) - \bar{s}^2\beta^2}}. \tag{79}$$

The above expression cannot be made explicit for general L (but we could get explicit expression for $L \in \{2, 3, 4\}$). As an alternative, we consider a looser bound $u(t)$:

$$\frac{du(t)}{dt} = \bar{s}_1 u^{L-1}(t), \qquad u(0) = \beta\frac{\bar{s}_\rho}{\bar{s}_1}, \tag{80}$$

where $\bar{s}_\rho := (1-\tau)(s+\beta^L) + \tau(L-1)(1+\beta^L)\rho$; note that $\bar{s}_1 = \bar{s}$. We have $u(t) \geq g_2(t)$ and $\bar{s}u(t) \geq g_1(t)$ $\forall t \geq 0$. This ODE solves as

$$u(t) = \left(\bar{s}(2-L)t + \bar{\beta}^{2-L}\right)^{\frac{1}{2-L}}, \tag{81}$$

where $\bar{\beta} = \beta\frac{\bar{s}_\rho}{\bar{s}_1}$. Note that the solution exists only for $t < \frac{\bar{\beta}^{2-L}}{(L-2)\bar{s}}$.

For the input layer weights, we get

$$\frac{d\|W_1^\tau(t)\|}{dt} \leq \bar{s}u^{L-1}(t). \tag{82}$$

The solution is given by

$$\|W_1^\tau(t)\| \leq \beta + \bar{s}\int_0^t \left(\bar{s}(2-L)t + \bar{\beta}^{2-L}\right)^{\frac{L-1}{2-L}} dt = \beta - \bar{\beta} + \left(\bar{s}(2-L)t + \bar{\beta}^{2-L}\right)^{\frac{1}{2-L}} = \beta - \bar{\beta} + u(t). \tag{83}$$

Similarly, we have

$$\frac{1}{\sqrt{m}}\|W_1^\tau(t)\tilde{X}\|_F \leq \beta\rho + \bar{s}\int_0^t \left(\bar{s}(2-L)t + \bar{\beta}^{2-L}\right)^{\frac{L-1}{2-L}} dt = \beta\rho - \bar{\beta} + u(t). \tag{84}$$

For brevity, we define

$$b = \beta - \bar{\beta} = \beta\left(1 - \frac{\bar{s}_\rho}{\bar{s}_1}\right) = -\beta\tau(L-1)(\rho-1)\frac{1+\beta^L}{\bar{s}}, \tag{85}$$

and

$$b_\rho = \beta\rho - \bar{\beta} = \beta\left(\rho - \frac{\bar{s}_\rho}{\bar{s}_1}\right) = \beta(1-\tau)(\rho-1)\frac{s+\beta^L}{\bar{s}}. \tag{86}$$

As a simpler option, we could just say $\|W_1^\tau(t)\| \leq u(t)$ and $\frac{1}{\sqrt{m}}\|W_1^\tau(t)\tilde{X}\| \leq \rho u(t)$.

Bounding norms of weight derivatives. Recall the definition of Ξ^τ:

$$\Xi^\tau = \frac{1}{m}\left(Y - W_L^\tau\left[D_{L-1}^\tau \odot W_{L-1}^\tau \dots \left[D_1^\tau \odot W_1^\tau \tilde{X}\right]\right]\right); \tag{87}$$

$$\begin{aligned} m\left\|\frac{d\Xi^\tau}{d\tau}\right\| &\leq (L-1)\|W_1^\tau\tilde{X}\|_F \prod_{l=2}^L \|W_l^\tau\| + \left\|\frac{dW_1^\tau\tilde{X}}{d\tau}\right\| \prod_{l=2}^L \|W_l^\tau\| + \|W_1^\tau\tilde{X}\|_F \sum_{k=2}^L \left\|\frac{dW_k^\tau}{d\tau}\right\| \prod_{l\in[2:L]\setminus\{k\}} \|W_l^\tau\| \\ &\leq u^{L-1}\left[(L-1)\sqrt{\rho m}u + \left\|\frac{dW_1^\tau\tilde{X}}{d\tau}\right\|_F + \sqrt{\rho m}\sum_{k=2}^L \left\|\frac{dW_k^\tau}{d\tau}\right\|\right]. \end{aligned} \tag{88}$$

For $l \in [2, L]$,

$$\begin{aligned}
\left\| \frac{d^2 W_l^\tau}{dt d\tau} \right\| &\leq \left(\left\| \frac{d\Xi^\tau}{d\tau} \right\|_F + (L-1)\|\Xi^\tau\|_F \right) \|W_1^\tau \tilde{X}\|_F \prod_{k\in[2,L]\setminus\{l\}} \|W_k^\tau\| \\
&+ \sum_{j\in[2:L]\setminus\{l\}} \left((1-\tau)\|\Xi^\tau \tilde{X}^\top\| \|W_1^\tau\| + \tau(L-1)\|\Xi^\tau\|_F \|W_1^\tau \tilde{X}\|_F \right) \left\| \frac{dW_j^\tau}{d\tau} \right\| \prod_{k\in[2,L]\setminus\{l,j\}} \|W_k^\tau\| \\
&+ \left((1-\tau)\|\Xi^\tau \tilde{X}^\top\| \left\| \frac{dW_1^\tau}{d\tau} \right\| + \tau(L-1)\|\Xi^\tau\|_F \left\| \frac{dW_1^\tau \tilde{X}}{d\tau} \right\|_F \right) \prod_{k\in[2,L]\setminus\{l\}} \|W_k^\tau\| \\
&\leq \left(u^{L-1} \left[(L-1)\rho u + \frac{1}{\sqrt{m}} \left\| \frac{dW_1^\tau \tilde{X}}{d\tau} \right\|_F + \rho \sum_{k=2}^{L} \left\| \frac{dW_k^\tau}{d\tau} \right\| \right] + \bar{1}(L-1) \right) \rho u^{L-1} \\
&+ \bar{s} u^{L-2} \sum_{j\in[2:L]\setminus\{l\}} \left\| \frac{dW_j^\tau}{d\tau} \right\| + \left((1-\tau)(s+\beta^L) \left\| \frac{dW_1^\tau}{d\tau} \right\| + \tau \bar{1}(L-1) \frac{1}{\sqrt{m}} \left\| \frac{dW_1^\tau \tilde{X}}{d\tau} \right\|_F \right) u^{L-2}.
\end{aligned} \tag{89}$$

For $l = 1$,

$$\begin{aligned}
\left\| \frac{d^2 W_1^\tau}{dt d\tau} \right\| &\leq \left(\left\| \frac{d\Xi^\tau}{d\tau} \right\|_F + (L-1)\|\Xi^\tau\|_F \right) \|\tilde{X}\| \prod_{k\in[2,L]} \|W_k^\tau\| \\
&+ \sum_{j=2}^{L} \left((1-\tau)\|\Xi^\tau \tilde{X}^\top\| + \tau(L-1)\|\Xi^\tau\|_F \|\tilde{X}\| \right) \left\| \frac{dW_j^\tau}{d\tau} \right\| \prod_{k\in[2,L]\setminus\{j\}} \|W_k^\tau\| \\
&\leq \left(u^{L-1} \left[(L-1)\rho u + \frac{1}{\sqrt{m}} \left\| \frac{dW_1^\tau \tilde{X}}{d\tau} \right\|_F + \rho \sum_{k=2}^{L} \left\| \frac{dW_k^\tau}{d\tau} \right\| \right] + \bar{1}(L-1) \right) u^{L-1} + \bar{s} u^{L-2} \sum_{j=2}^{L} \left\| \frac{dW_j^\tau}{d\tau} \right\|.
\end{aligned} \tag{90}$$

$\left\| \frac{dW_l^\tau}{d\tau} \right\|$ $\forall l \in [L]$, as well as $\frac{1}{\sqrt{m}} \left\| \frac{dW_1^\tau \tilde{X}}{d\tau} \right\|_F$, are all bounded by $v(t)$ which satisfies

$$\begin{aligned}
\frac{dv}{dt} &= \left[(L-1)\rho u^L + (1 + (L-1)\rho) u^{L-1} v + (L-1)\bar{1} \right] u^{L-1} + \bar{s}(L-1) u^{L-2} v \\
&= v \left[(1 + (L-1)\rho) u^{2L-2} + \bar{s}(L-1) u^{L-2} \right] + (L-1) u^{L-1} \left[\bar{1} + \rho u^L \right], \qquad v(0) = 0,
\end{aligned} \tag{91}$$

where $\bar{1} = 1 + \beta^L$.

Solving the ODE for v(t). Recall

$$u(t) = \left(\bar{s}(2-L)t + \bar{\beta}^{2-L} \right)^{\frac{1}{2-L}}. \tag{92}$$

We solve the homogeneous equation to get

$$\begin{aligned}
v(t) &= C(t) e^{\frac{L-1}{2-L} \ln\left(\bar{s}(2-L)t + \bar{\beta}^{2-L}\right) + \frac{1+(L-1)\rho}{\bar{s}L} \left(\bar{s}(2-L)t + \bar{\beta}^{2-L}\right)^{\frac{L}{2-L}}} \\
&= C(t) e^{(L-1)\ln u(t) + \frac{1+(L-1)\rho}{\bar{s}L} u^L(t)} = C(t) u^{L-1}(t) e^{\frac{1+(L-1)\rho}{\bar{s}L} u^L(t)},
\end{aligned} \tag{93}$$

where $C(t)$ satisfies

$$\frac{dC(t)}{dt} u^{L-1}(t) e^{\frac{1+(L-1)\rho}{\bar{s}L} u^L(t)} = (L-1) u^{L-1}(t) \left[\bar{1} + \rho u^L(t) \right]. \tag{94}$$

Let us introduce $\hat{s} = \frac{L}{1+(L-1)\rho} \bar{s}$. Then

$$\begin{aligned}
C(t) &= (L-1) \int e^{-\frac{u^L(t)}{\bar{s}}} \left[\bar{1} + \rho u^L(t) \right] dt \\
&= \frac{L-1}{L} \hat{s}^{\frac{2-L}{L}} \left[\frac{\bar{1}}{\bar{s}} \left(\Gamma\left(\frac{2-L}{L}, \frac{\beta^L}{\hat{s}} \right) - \Gamma\left(\frac{2-L}{L}, \frac{u^L(t)}{\hat{s}} \right) \right) + \rho \frac{\hat{s}}{\bar{s}} \left(\Gamma\left(\frac{2}{L}, \frac{\beta^L}{\hat{s}} \right) - \Gamma\left(\frac{2}{L}, \frac{u^L(t)}{\hat{s}} \right) \right) \right].
\end{aligned} \tag{95}$$

This gives the final solution:

$$v(t) = \frac{L-1}{L}\hat{s}^{\frac{2-L}{L}} u^{L-1}(t)[w(u(t)) - w(\beta)]e^{\frac{u^L(t)}{\hat{s}}}, \tag{96}$$

where we took

$$w(u) = -\frac{\bar{1}}{\bar{s}}\Gamma\left(\frac{2-L}{L}, \frac{u^L}{\hat{s}}\right) - \frac{L\rho}{1+(L-1)\rho}\Gamma\left(\frac{2}{L}, \frac{u^L}{\hat{s}}\right). \tag{97}$$

D.2 Evaluating the solution at the learning time

For a properly initialized linear network, $\forall l \in [L]$ $\|W_l^0(t)\| = \bar{u}(t)$, where $u(t)$ satisfies (Saxe et al., 2013)

$$\frac{d\bar{u}(t)}{dt} = \bar{u}^{L-1}(t)(s - \bar{u}^L(t)), \qquad \bar{u}(0) = \beta, \tag{98}$$

which gives the solution in implicit form[6]:

$$t_\alpha^*(\beta) = \int \frac{d\bar{u}}{\bar{u}^{L-1}(s-\bar{u}^L)} = \frac{\bar{u}^{2-L}(t)}{s(2-L)}\,{}_2F_1\left(1, \frac{2}{L}-1, \frac{2}{L}; \frac{\bar{u}^L(t)}{s}\right) - \frac{\beta^{2-L}}{s(2-L)}\,{}_2F_1\left(1, \frac{2}{L}-1, \frac{2}{L}; \frac{\beta^L}{s}\right). \tag{99}$$

Suppose we are going to learn a fixed fraction of the data, i.e. take $\bar{u}(t) = (\alpha s)^{1/L}$ for $\alpha \in (0,1)$. Then

$$t_\alpha^*(\beta) = \frac{\beta^{2-L}\,{}_2F_1\left(1, \frac{2}{L}-1, \frac{2}{L}; \frac{\beta^L}{s}\right) - (\alpha s)^{\frac{2}{L}-1}\,{}_2F_1\left(1, \frac{2}{L}-1, \frac{2}{L}; \alpha\right)}{s(L-2)}. \tag{100}$$

Since $t_\alpha^*(\beta)$ is the time sufficient to learn a network for $\epsilon = 0$, we suppose it also suffices to learn a nonlinear network. So, we are going to evaluate our bound at $t = t_\alpha^*$. Since we need $\hat{R}_\gamma(t_\alpha^*) < 1$ for the bound to be non-vacuous, we should take γ small relative to α. We consider $\gamma = \alpha^\nu/q$ for $\nu, q \geq 1$.

Since the linear network learning time $t_\alpha^*(\beta)$ is correct for almost all initialization only when β vanishes, we are going to work in the limit of $\beta \to 0$. Since we need $\alpha \in (\beta^L/s, 1)$, otherwise the linear training time is negative, we take $\alpha = \frac{r}{s}\beta^\lambda$ for $\lambda \in (0, L]$ and $r > 1$.

Consider first $\lambda < L$:

$$t_\alpha^*(\beta) = \frac{\beta^{2-L}}{s(L-2)} + O(\beta^{2\lambda/L}). \tag{101}$$

Let us evaluate u at this time:

$$u(t_\alpha^*(\beta)) = \left(\bar{\beta}^{2-L} - \frac{\bar{s}}{s}\beta^{2-L} + O(\beta^{2\lambda/L})\right)^{\frac{1}{2-L}} = \beta\left(\frac{\bar{s}_\rho^{2-L}}{\bar{s}_1^{2-L}} - \frac{\bar{s}}{s}\right)^{\frac{1}{2-L}}\left(1 + O\left(\beta^{L-2+2\lambda/L}\right)\right). \tag{102}$$

Apparently, this expression does not make sense for $\rho = 1$ and even close to it, so we switch to $\lambda = L$, which we expect to be the right exponent:

$$t_\alpha^*(\beta) = \beta^{2-L}\frac{1 - r^{\frac{2}{L}-1}}{s(L-2)} + O(\beta^2). \tag{103}$$

Let us evaluate u at this time:

$$u(t_\alpha^*(\beta)) = \left(\bar{\beta}^{2-L} - \frac{\bar{s}}{s}\left(1 - r^{\frac{2}{L}-1}\right)\beta^{2-L} + O(\beta^2)\right)^{\frac{1}{2-L}} = \beta\left(\frac{\bar{s}_\rho^{2-L}}{\bar{s}_1^{2-L}} - \frac{\bar{s}}{s}\left(1 - r^{\frac{2}{L}-1}\right)\right)^{\frac{1}{2-L}}\left(1 + O\left(\beta^L\right)\right). \tag{104}$$

This expression makes sense whenever $\frac{\bar{s}_\rho^{2-L}}{\bar{s}_1^{2-L}} - \frac{\bar{s}}{s}\left(1 - r^{\frac{2}{L}-1}\right) > 0$, i.e. when r is close enough to 1.

[6] ${}_2F_1$ is a hypergeometric function defined as a series ${}_2F_1(a,b,c,z) = 1 + \sum_{k=1}^\infty \frac{(a)_k(b)_k}{(c)_k}\frac{z^k}{k!}$, where $(q)_k = q(q+1)\dots(q+k-1)$.

Let us evaluate w at the training time now. Since $\Gamma(a,x) = \Gamma(a) - \frac{x^a}{a} + O(x^{a+1})$, we get

$$\begin{aligned} w(u(t^*_\alpha(\beta))) - w(\beta) &= \frac{\bar{1}}{\bar{s}} \frac{L}{2-L} \hat{s}^{\frac{L-2}{L}} \left(u^{2-L}(t^*_\alpha(\beta)) - \beta^{2-L} \right) + O(\beta^2). \\ &= \frac{\bar{1}}{\bar{s}} \frac{L}{2-L} \hat{s}^{\frac{L-2}{L}} \beta^{2-L} \left(\frac{\bar{s}_\rho^{2-L}}{\bar{s}_1^{2-L}} - \frac{\bar{s}}{s} \left(1 - r^{\frac{2}{L}-1} \right) - 1 \right) + O(\beta^2). \end{aligned} \tag{105}$$

Then the quantity of interest becomes

$$\begin{aligned} \frac{L u^{L-1}(t^*_\alpha(\beta)) v(t^*_\alpha(\beta))}{\gamma} &= \frac{L-1}{\gamma} u^{2L-2}(t^*_\alpha(\beta)) \frac{\bar{1}}{\bar{s}} \frac{L}{2-L} \beta^{2-L} \left(\frac{\bar{s}_\rho^{2-L}}{\bar{s}_1^{2-L}} - \frac{\bar{s}}{s} \left(1 - r^{\frac{2}{L}-1} \right) - 1 \right) (1 + O(\beta^L)) \\ = \frac{q s^\nu}{r^\nu} \frac{\bar{1}}{\bar{s}} \frac{L(L-1)}{2-L} \beta^{L(1-\nu)} &\left(\frac{\bar{s}_\rho^{2-L}}{\bar{s}_1^{2-L}} - \frac{\bar{s}}{s} \left(1 - r^{\frac{2}{L}-1} \right) \right)^{\frac{2L-2}{2-L}} \left(\frac{\bar{s}_\rho^{2-L}}{\bar{s}_1^{2-L}} - \frac{\bar{s}}{s} \left(1 - r^{\frac{2}{L}-1} \right) - 1 \right) (1 + O(\beta^L)). \end{aligned} \tag{106}$$

This expression does not diverge as $\beta \to 0$ when $\nu = 1$. We will also have a finite $\lim_{L\to\infty} \lim_{\beta\to 0}$ whenever $\epsilon \propto (L-1)^{-1}$.

E Proving Assumption 4.1 for linear nets

We have for $l = 1$,

$$\nabla_1 := \frac{1}{2m} \frac{\partial \left\| Y - f^\epsilon_{\theta^\epsilon}(\tilde{X}) \right\|_F^2}{\partial W_1^\epsilon} = - \left[D_1^\epsilon \odot W_2^{\epsilon,\top} \dots \left[D_{L-1}^\epsilon \odot W_L^{\epsilon,\top} \Xi^\epsilon \right] \right] \tilde{X}^\top. \tag{107}$$

For $l \in [2, L]$,

$$\nabla_l := \frac{1}{2m} \frac{\partial \left\| Y - f^\epsilon_{\theta^\epsilon}(\tilde{X}) \right\|_F^2}{\partial W_l^\epsilon} = - \left[D_l^\epsilon \odot W_{l+1}^{\epsilon,\top} \dots \left[D_{L-1}^\epsilon \odot W_L^{\epsilon,\top} \Xi^\epsilon \right] \right] \left[D_{l-1}^\epsilon \odot W_{l-1}^\epsilon \dots \left[D_1^\epsilon \odot W_1^\epsilon \tilde{X} \right] \right]^\top. \tag{108}$$

We also have

$$\nabla_1^X := \frac{1}{2m} \frac{\partial \left\| \left(Y - f^\epsilon_{\theta^\epsilon}(\tilde{X}) \right) \tilde{X}^\top \right\|_F^2}{\partial W_1^\epsilon} = - \left[D_1^\epsilon \odot W_2^{\epsilon,\top} \dots \left[D_{L-1}^\epsilon \odot W_L^{\epsilon,\top} \Xi^\epsilon \tilde{X}^\top \tilde{X} \right] \right] \tilde{X}^\top. \tag{109}$$

$$\begin{aligned} \nabla_l^X &:= \frac{1}{2m} \frac{\partial \left\| \left(Y - f^\epsilon_{\theta^\epsilon}(\tilde{X}) \right) \tilde{X}^\top \right\|_F^2}{\partial W_l^\epsilon} \\ &= - \left[D_l^\epsilon \odot W_{l+1}^{\epsilon,\top} \dots \left[D_{L-1}^\epsilon \odot W_L^{\epsilon,\top} \Xi^\epsilon \tilde{X}^\top \tilde{X} \right] \right] \left[D_{l-1}^\epsilon \odot W_{l-1}^\epsilon \dots \left[D_1^\epsilon \odot W_1^\epsilon \tilde{X} \right] \right]^\top. \end{aligned} \tag{110}$$

The statement of Assumption 4.1 follows from

Conjecture E.1. $\forall \epsilon \in [0,1]\ \forall t \geq 0\ \sum_{l=1}^L \operatorname{tr}\left[\nabla_l^X \nabla_l^\top \right] \geq 0.$

Indeed, the above conjecture states that loss gradients wrt weights and "projected" loss gradients wrt weights are positively aligned, so, whenever loss does not increase, neither does projected loss. Since we use gradient flow, loss is guaranteed to not increase. Below, we prove the conjecture for linear nets.

Proof of Conjecture E.1 for $\epsilon = 0$. Since ϵ is zero, we omit the corresponding sup-index:

$$\begin{aligned} \operatorname{tr}\left[\nabla_1^X \nabla_1^\top \right] &= \operatorname{tr}\left[\left[W_2^\top \dots W_L^\top \Xi \tilde{X}^\top \tilde{X} \right] \tilde{X}^\top \tilde{X} \left[W_2^\top \dots W_L^\top \Xi \right]^\top \right] \\ &= \operatorname{tr}\left[\left[W_2^\top \dots W_L^\top \Xi \tilde{X}^\top \right] \left[W_2^\top \dots W_L^\top \Xi \tilde{X}^\top \right]^\top \right] = \operatorname{tr}\left[W_L \dots W_2 W_2^\top \dots W_L^\top \right] \operatorname{tr}\left[\Xi X^\top X \Xi^\top \right] \end{aligned} \tag{111}$$

by the circular property of trace, and the fact that Ξ is a matrix with a single row. Since $d_{out} = 1$, both traces are just squared Euclidean norms of vectors, hence they are non-negative: $\operatorname{tr}\left[\nabla_1^X \nabla_1^\top\right] \geq 0$.

Let us do the same for the other layers:

$$\begin{aligned}
\operatorname{tr}\left[\nabla_l^X \nabla_l^\top\right] &= \operatorname{tr}\left[\left[W_{l+1}^\top \dots W_L^\top \Xi \tilde{X}^\top \tilde{X}\right] \left[W_{l-1} \dots W_1 \tilde{X}\right]^\top \left[W_{l-1} \dots W_1 \tilde{X}\right] \left[W_{l+1}^\top \dots W_L^\top \Xi\right]^\top\right] \\
&= \operatorname{tr}\left[\left[W_{l+1}^\top \dots W_L^\top \Xi \tilde{X}^\top W_1^\top \dots W_{l-1}^\top\right] \left[W_{l+1}^\top \dots W_L^\top \Xi \tilde{X}^\top W_1^\top \dots W_{l-1}^\top\right]^\top\right] \\
&= \operatorname{tr}\left[W_L \dots W_{l+1} W_{l+1}^\top \dots W_L^\top\right] \operatorname{tr}\left[\Xi X^\top W_1^\top \dots W_{l-1}^\top W_{l-1} \dots W_1 X \Xi^\top\right] \geq 0
\end{aligned} \tag{112}$$

for the same reasons as before. □

Clearly, Conjecture E.1 should also hold for small enough ϵ whenever it holds for $\epsilon = 0$. However, the bound for the maximal ϵ for which we were able to guarantee the conjecture statement, vanishes with time t, as our weight bounds are too loose. For this reason, we do not include it here. See Section 5 for empirical validation of Assumption 4.1.