The large learning rate phase of deep learning: the catapult mechanism

Aitor Lewkowycz [1] **Yasaman Bahri** [2] **Ethan Dyer** [1] **Jascha Sohl-Dickstein** [2] **Guy Gur-Ari** [1]

Abstract

The choice of initial learning rate can have a profound effect on the performance of deep networks. We present a class of neural networks with solvable training dynamics, and confirm their predictions empirically in practical deep learning settings. The networks exhibit sharply distinct behaviors at small and large learning rates. The two regimes are separated by a phase transition. In the small learning rate phase, training can be understood using the existing theory of infinitely wide neural networks. At large learning rates the model captures qualitatively distinct phenomena, including the convergence of gradient descent dynamics to flatter minima. One key prediction of our model is a narrow range of large, stable learning rates. We find good agreement between our model's predictions and training dynamics in realistic deep learning settings. Furthermore, we find that the optimal performance in such settings is often found in the large learning rate phase. We believe our results shed light on characteristics of models trained at different learning rates. In particular, they fill a gap between existing wide neural network theory, and the nonlinear, large learning rate, training dynamics relevant to practice.

1. Introduction

Deep learning has shown remarkable success across a variety of machine learning tasks. At the same time, our theoretical understanding of deep learning methods remains limited. In particular, the interplay between training dynamics, properties of the learned network, and generalization remains a largely open problem.

In this work we take a step toward addressing these questions. We present a dynamical mechanism that allows deep networks trained using SGD to find flat minima and achieve superior performance. Our theoretical predictions agree well with empirical results in a variety of deep learning settings. In many cases we are able to predict the regime of learning rates where optimal performance is achieved. Figure 1 summarizes our main results. This work builds on several existing results, which we now review.

1.1. Large learning rate SGD improves generalization

SGD training with large initial learning rates often leads to improved performance over training with small initial learning rates (see Li et al. (2019); Leclerc & Madry (2020); Xie et al. (2020); Frankle et al. (2020); Jastrzebski et al. (2020) for recent discussions). It has been suggested that one of the mechanisms underlying the benefit of large learning rates is that noise from stochastic gradient descent leads to flat minima, and that flat minima generalize better than sharp minima (Hochreiter & Schmidhuber, 1997; Keskar et al., 2016; Smith & Le, 2018; Jiang et al., 2020; Park et al., 2019) (though see Dinh et al. (2017) for discussion of some caveats). According to this suggestion, training with a large learning rate (or with a small batch size) can improve performance because it leads to more stochasticity during training (Mandt et al., 2017; Smith et al., 2017; Smith & Le, 2018; Smith et al., 2018).

We will develop a connection between large learning rate and flatness of minima in models trained via SGD. Unlike the relationship explored in most previous work though, this connection is not driven by SGD noise, but arises solely as a result of training with a large initial learning rate, and holds even for full batch gradient descent.

[1]Google [2]Google Brain. Correspondence to: Guy Gur-Ari <guyga@google.com>.

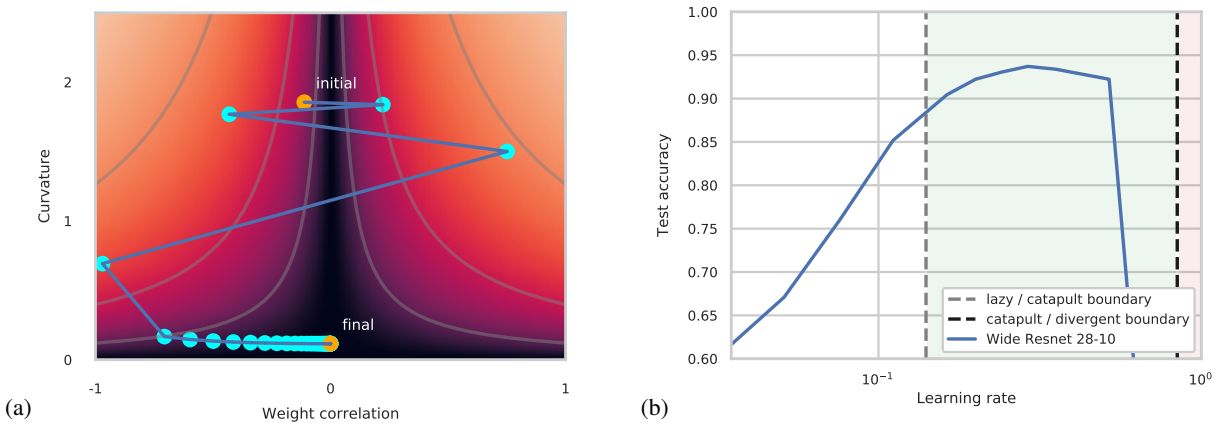


Figure 1. A summary of our main results. (a) A visualization of gradient descent dynamics derived in our theoretical setup. A 2D slice of parameter space is shown, where lighter color indicates higher loss and dots represents points visited during optimization. Initially, the loss grows rapidly while local curvature decreases. Once curvature is sufficiently low, gradient descent converges to a flat minimum. We call this the *catapult effect*. See Figures 2 and S1 for more details. (b) Confirmation of our theoretical predictions in a practical deep learning setting. Line shows the test accuracy of a Wide ResNet trained on CIFAR-10 as a function of learning rate, each trained for a fixed number of steps. Dashed lines show our predictions for the boundaries of the large learning rate regime (the *catapult phase*), where we expect optimal performance to occur. Maximal performance is achieved between the dashed lines, confirming our predictions. See Section 3 for details.

1.2. The existing theory of infinite width networks is insufficient to describe large learning rates

A recent body of work has investigated the gradient descent dynamics of deep networks in the limit of infinite width (Daniely, 2017; Jacot et al., 2018; Lee et al., 2019; Du et al., 2019; Zou et al., 2018; Allen-Zhu et al., 2019; Li & Liang, 2018; Chizat et al., 2019; Mei et al., 2018; Rotskoff & Vanden-Eijnden, 2018; Sirignano & Spiliopoulos, 2018; Woodworth et al., 2019; Naveh et al.). Of particular relevance is the work by Jacot et al. (2018) showing that gradient flow in the space of functions is governed by a dynamical quantity called the Neural Tangent Kernel (NTK) which is fixed at its initial value in this limit. Lee et al. (2019) showed this result is equivalent to training the linearization of a model around its initialization in parameter space. Finally, moving away from the strict limit of infinite width by working perturbatively, Dyer & Gur-Ari (2020); Huang & Yau (2019) introduced an approach to computing the finite-width corrections to network evolution.

Despite this progress, it seems these results are insufficient to capture the full dynamics of deep networks, as well as their superior performance, in regimes applicable to practice. Prior work has focused on comparisons between various infinite-width kernels associated with deep networks and their finite-width, SGD-trained counterparts (Lee et al., 2018; Novak et al., 2019; Arora et al., 2019). Specific findings vary depending on precise choices for architecture and hyperparameters. However, dramatic performance gaps are consistently observed between non-linear CNNs and their limiting kernels, implying that the theory is not sufficient to explain the performance of deep networks in this realistic setup. Furthermore, some hyperparameter settings in finite-width models have no known analogue in the infinite width limit, and it is these settings that often lead to optimal performance.

In particular, finite width networks are often trained with large learning rates that would cause divergence for infinite width linearized models. Further, these large learning rates cause finite width networks to converge to flat minima. For infinite width linearized models, trained with MSE loss, all minima have the same curvature, and the notion of flat minima does not apply. We argue that the reduction in curvature during optimization, and support for learning rates that are infeasible for infinite width linearized models, may thus partially explain performance gaps observed between linear and non-linear models.

1.3. Our contribution: three learning rate regimes

In this work, we identify a dynamical mechanism which enables finite-width networks to stably access large learning rates. We show that this mechanism causes training to converge to flatter minima and is associated with improved generalization. We further show that this same mechanism can describe the behavior of infinite width networks, if training time is increased

along with network width.

This new mechanism enables a characterization of gradient descent training in terms of three learning rate regimes, or phases: the *lazy phase*, the *catapult phase*, and the *divergent phase*. In Section 2 we analytically derive the behavior in these three learning rate regimes for one hidden layer linear networks with large but finite width, trained with MSE loss. We confirm experimentally in Section 3 that these phases also apply to deep nonlinear fully- connected, convolutional, and residual architectures. In Section 4 we study additional predictions of the analytic solution.

We now summarize all three phases, using η to indicate the learning rate, and λ_0 to indicate the initial curvature (defined precisely in Section 2.1). The phase is determined by the curvature at initialization and by the learning rate, despite the fact that the curvature may change significantly during training. Based on the experimental evidence we expect the behavior described below to apply in typical deep learning settings, when training sufficiently wide networks using SGD.

Lazy phase: $\eta < 2/\lambda_0$ **.** For sufficiently small learning rate, the curvature λ_t at training step t remains constant during the initial part of training. The model behaves (loosely) as a model linearized about its initial parameters (Lee et al., 2019); this becomes exact in the infinite width limit, where these dynamics are sometimes called *lazy training* (Jacot et al., 2018; Lee et al., 2019; Du et al., 2019; Li & Liang, 2018; Zou et al., 2018; Allen-Zhu et al., 2019; Chizat et al., 2019; Dyer & Gur-Ari, 2020). For a discussion of trainability and the connection to the NTK in the lazy phase see Xiao et al. (2019).

Catapult phase: $2/\lambda_0 < \eta < \eta_{\max}$ **.** In this phase, the curvature at initialization is too high for training to converge to a nearby point, and the linear approximation quickly breaks down. Optimization begins with a period of exponential growth in the loss, coupled with a rapid decrease in curvature, until curvature stabilizes at a value $\lambda_{\text{final}} < 2/\eta$. Once the curvature drops below $2/\eta$, training converges, ultimately reaching a minimum that is flatter than those found in the lazy phase. This initial period lasts for a number of training steps that is of order $\log(n)$, where n is the network width, and is therefore quite short for realistic networks (often lasting less than a single epoch). Optimal performance is often achieved when the initial learning rate is in this range. The gradient descent dynamics in this phase are visualized in SM Figure S1 and in Figure 1.

The maximum learning rate is approximately given by $\eta_{\max} = c_{\text{act.}}/\lambda_0$, where $c_{\text{act.}}$ is an architecture-dependent constant. Empirically, we find that this constant depends strongly on the non-linearity but only weakly on other aspects of the architecture. For networks with ReLU non-linearity we find empirically that $c_{\text{act.}} \approx 12$. For the theoretical model, we show that $c_{\text{act.}} = 4$.

Divergent phase: $\eta > \eta_{\max}$ **.** When the learning rate is above the maximum learning rate of the model, the loss diverges and the model does not train.

2. Theoretical results

We now present our main theoretical result, an analysis of gradient descent dynamics for a neural network with large but finite width.

Given a network function $f : \mathbb{R}^d \to \mathbb{R}$ with model parameters $\theta \in \mathbb{R}^p$, and a training set $\{(x_\alpha, y_\alpha)\}_{\alpha=1}^m$, the MSE loss is

$$L = \frac{1}{2m} \sum_{\alpha=1}^{m} (f(x_\alpha) - y_\alpha)^2 \,. \tag{1}$$

The NTK $\Theta : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is defined by

$$\Theta(x, x') := \frac{1}{m} \sum_{\mu=1}^{p} \frac{\partial f(x)}{\partial \theta_\mu} \frac{\partial f(x')}{\partial \theta_\mu} \,. \tag{2}$$

We denote by λ the maximum eigenvalue of the kernel. In large width models, λ provides a local measure of the loss landscape curvature that is similar to the top eigenvalue of the Hessian (Dyer & Gur-Ari, 2020).

In this section, we will consider a network with one hidden layer and linear activations, where the network function f is given by

$$f(x) = n^{-1/2} v^T u x \,. \tag{3}$$

Here n is the width (number of neurons in the hidden layer), $v \in \mathbb{R}^n$ and $u \in \mathbb{R}^{n \times d}$ are the model parameters (collectively denoted θ), and $x \in \mathbb{R}^d$ is the training input. At initialization, the weights are drawn from $\mathcal{N}(0, 1)$.

2.1. Warmup: a simplified model

Before analyzing the dynamics of the model, we analyze a simpler setting which captures the most important aspects of the full solution. Consider a dataset with 1D inputs, and with a single training sample $x = 1$ with label $y = 0$. The network function evaluated on this input is then $f = n^{-1/2} v^T u$, with $u, v \in \mathbb{R}^n$, and the loss is $L = f^2/2$. The gradient descent equations at training step t are

$$u_{t+1} = u_t - \eta n^{-1/2} f_t v_t \,, \; v_{t+1} = v_t - \eta n^{-1/2} f_t u_t \,. \tag{4}$$

Next, consider the update equations in function space. These can be written in terms of the Neural Tangent Kernel. For this model, the kernel evaluated on the training set is a scalar which is equal to λ, its top eigenvalue, and is given by

$$\Theta(1,1) = \lambda = n^{-1} \left(\|v\|_2^2 + \|u\|_2^2 \right) \,. \tag{5}$$

At initialization, both f^2 and λ scale as $n^0 = 1$ with width. The following update equations for f and λ at step t can be derived from (4).

$$f_{t+1} = \left(1 - \eta \lambda_t + \frac{\eta^2 f_t^2}{n} \right) f_t \,, \tag{6}$$

$$\lambda_{t+1} = \lambda_t + \frac{\eta f_t^2}{n} \left(\eta \lambda_t - 4 \right) \,. \tag{7}$$

It is important to note that these are the exact update equations for this model, and that no higher-order terms were neglected. We now analyze these dynamical equations assuming the width n is large. Two learning rates that will be important in the analysis are $\eta_{\rm crit} = 2/\lambda_0$ and $\eta_{\rm max} = 4/\lambda_0$. In terms of the notation introduced above, the architecture-dependent constant that determines that maximum learning rate in this model is $c_{\rm act.} = 4$.

2.1.1. Lazy phase

Taking the strict infinite width limit, equations (6) and (7) become

$$f_{t+1} = (1 - \eta \lambda_t) f_t \,, \quad \lambda_{t+1} = \lambda_t \,. \tag{8}$$

When $\eta < \eta_{\rm crit}$, λ remains constant throughout training. This is a special case of NTK dynamics, where the kernel is constant and the network evolves as a linear model (Lee et al., 2019). The function and the loss both shrink to zero because the multiplicative factor obeys $|1 - \eta \lambda_t| < 1$. This convergence happens in $\mathcal{O}(n^0) = \mathcal{O}(1)$ steps.

2.1.2. Catapult phase

When $\eta_{\rm crit} < \eta < \eta_{\rm max}$, the loss diverges in the infinite width limit. Indeed, from (8) we see that the kernel is constant in the limit, while f receives multiplicative updates where $|1 - \eta \lambda_t| > 1$. This is the well known instability of gradient descent dynamics for linear models with MSE loss. However, the underlying model is not linear in its parameters, and finite width contributions turn out to be important. We therefore relax the infinite width limit and analyze equations (6,7) for large but finite width, $n \gg 1$.

First, note that $\eta \lambda_0 - 4 < 0$ by assumption, and therefore the (additive) kernel updates are negative for all t. During early training, $|f_t|$ grows (as in the infinite width limit) while λ_t remains constant up to small $\mathcal{O}(n^{-1})$ updates. After $t \sim \log(n)$ steps, $|f_t|$ grows to order $n^{1/2}$. At this point, the kernel updates are no longer negligible because f_t^2/n is of order n^0. The kernel λ_t receives negative, non-negligible updates while both f_t and the loss continue to grow (for now, we ignore the term in (6) with an explicit $1/n$ dependence). This continues until the kernel is sufficiently small that the condition $\eta \lambda_t \lesssim 2$ is met.[1] We call this curvature-reduction effect the *catapult effect*. Beyond this point, $|1 - \eta \lambda_t| < 1$ holds, $|f_t|$ shrinks, and the loss converges to a global minimum. The n dependence of the steps until optimization converges is $\log(n)$.

It remains to show that the term in (6) with an explicit n^{-1} dependence does not affect these conclusions. Once $|f_t|$ grows to order $n^{1/2}$, this term is no longer negligible and can cause the multiplicative factor in front of f_t to become smaller than 1

[1] The bound is not exact because of the term we neglected.

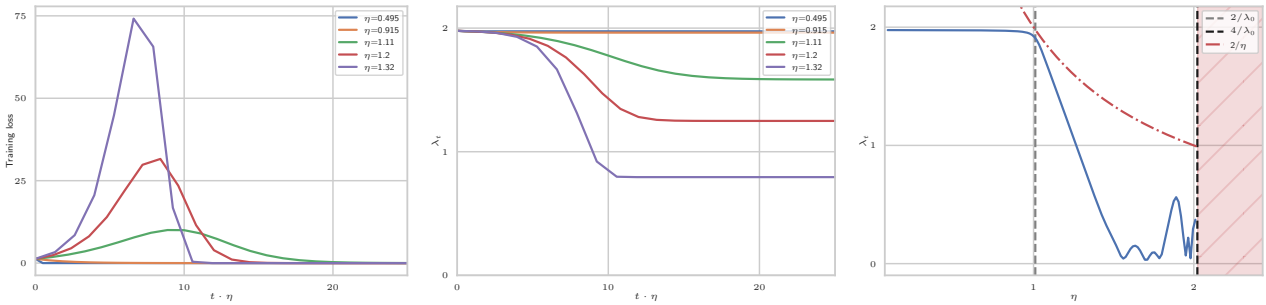
in absolute value, causing $|f_t|$ to start shrinking. However, once $|f_t|$ shrinks sufficiently this term again becomes negligible. Therefore, the loss will not converge to zero unless the curvature eventually drops below $2/\eta$. Conversely, notice that this term cannot cause $|f_t|$ to diverge for learning rates below $\eta_{\max}$. Indeed, if this were to happen then equation (7) would drive λ_t to negative values, leading to a contradiction. This completes the analysis in this phase.

Let us make a few comments about the catapult phase.

It is important for the analysis that we take a modified large width limit, in which the number of training steps grows like $\log(n)$ as n becomes large. This is different than the large width limit commonly studied in the literature, in which the number of steps is kept fixed as the width is taken large. When using this modified limit, the analysis above holds even in the limit. Note as well that the catapult effect takes place over $\log(n)$ steps, and for practical networks will occur within the first 100 steps or so of training.

In the catapult phase, the kernel at the end of training is smaller by an order n^0 amount compared with its value at initialization. The kernel provides a local measure of the loss curvature. Therefore, the minima that SGD finds in the catapult phase are flatter than those it finds in the lazy phase. Contrast this situation, in which the kernel receives non-negligible updates, with the conclusions of Jacot et al. (2018) where the kernel is constant throughout training. The difference is due to the large learning rate, which leads to a breakdown of the linearized approximation even at large width.

Figure 2 illustrates the dynamics in the catapult phase. For learning rates $\eta_{\text{crit}} < \eta < \eta_{\max}$ we observe the catapult effect: the loss goes up before converging to zero. The curvature exhibits the expected sharp transitions as a function of the learning rate: it is constant in the lazy phase, decreases in the catapult phase, and diverges for $\eta > \eta_{\max}$.



(a) (b) (c)

Figure 2. Empirical results for the gradient descent dynamics of the warmup model with $n = 10^3$, for which $\eta_{\text{crit}} \approx 1$. (a) Training loss for different learning rates. (b) Maximum NTK eigenvalue as a function of time. For $\eta > 1$, λ_t decreases rapidly to a fixed value. (c) Maximum NTK eigenvalue at $t = 25/\eta$. The shaded area indicates learning rates for which training diverges empirically. The results are presented as a function of $t \cdot \eta$ (rather than t) for convenience.

2.1.3. Divergent phase

Completing the analysis of this model, when $\eta > \eta_{\max}$ the loss diverges because the kernel receives positive updates, accelerating the rate of growth of the function. Therefore, $\eta_{\max} = 4/\lambda_0$ is the maximum learning rate of the model.

2.2. Full model

We now turn to analyzing the model presented at the beginning of this section, with d-dimensional inputs and m training samples with general labels. The full analysis is presented in SM Section D.1; here we summarize the argument. The conclusions are essentially the same as those of the warmup model.

We introduce the notation $f_\alpha := f(x_\alpha)$ for the function evaluated on a training sample, $\tilde{f}_\alpha := f_\alpha - y_\alpha$ for the error, and $\Theta_{\alpha\beta} := \Theta(x_\alpha, x_\beta)$ for the kernel elements. We will treat $f, \tilde{f}$ evaluated on the training set as vectors in $\mathbb{R}^m$, whose elements are $f_\alpha, \tilde{f}_\alpha$. Consider the following update equation for the error, which can be derived from the update equations for the

parameters. Note that this is the exact update equation for this model; no higher-order terms were neglected.

$$\tilde{f}_\alpha^{t+1} = \sum_\beta (\delta_{\alpha\beta} - \eta\Theta_{\alpha\beta})\tilde{f}_\beta + \frac{\eta^2}{nm}(x_\alpha^T\zeta)(f^T\tilde{f})\,. \tag{9}$$

Here, $\zeta := \sum_\alpha \tilde{f}_\alpha x_\alpha/m \in \mathbb{R}^d$, and all variables are implicitly evaluated at step t unless specified otherwise.

We again take the modified large width limit $n \to \infty$, allowing the number of steps to scale logarithmically in the width. At initialization, f_α, $\tilde{f}_\alpha$, and $\Theta_{\alpha\beta}$ are all of order n^0. We now analyze the gradient descent dynamics as a function of the learning rate.

The maximum eigenvalue of the kernel at step t is λ_t. When $\eta < \eta_{\rm crit}$, the norm $\|\tilde{f}^t\|_2$ shrinks to zero in $\mathcal{O}(n^0)$ time while the kernel receives $\mathcal{O}(n^{-1})$ corrections. Therefore, in the limit the kernel remains constant until convergence. This is a special case of the NTK result (Jacot et al., 2018), and the model evolves as a linear model.

Next, suppose that $\eta_{\rm crit} < \eta < \eta_{\rm max}$. Early during training $\|\tilde{f}\|_2$ grows, with the fastest growth taking place along the direction of the top kernel eigenvector, $e_t^{\rm max} \in \mathbb{R}^m$. During this part of training the kernel receives $\mathcal{O}(n^{-1})$ updates, and so $e_t^{\rm max}$ does not change much. As a result, $\tilde{f}_t$ becomes aligned with $e_t^{\rm max}$. In addition, f_t becomes close to $\tilde{f}_t$ because f_t grows while the label is constant. We therefore consider the following approximate update equations for $\tilde{f}^{\rm max} := \sum_\alpha \tilde{f}_\alpha e_\alpha^{\rm max}$ and for the maximum eigenvalue λ, which can be approximated by $\tilde{f}^T\Theta\tilde{f}/\|\tilde{f}\|_2^2$.

$$\tilde{f}_{t+1}^{\rm max} \approx (1 - \eta\lambda_t)\,\tilde{f}_t^{\rm max} + \mathcal{O}(n^{-1})\,, \tag{10}$$

$$\lambda_{t+1} \approx \lambda_t + \frac{\eta\|\zeta\|_2^2}{n}(\eta\lambda_t - 4)\,. \tag{11}$$

We note in passing the similarity between these equations and (6), (7). We see that once $\tilde{f}^{\rm max}$ and ζ become of order $n^{1/2}$, λ_t receives non-negligible negative corrections of order n^0. This evolution continues until $\lambda_t \lesssim 2/\eta$, after which the error converges to zero. Finally, if $\eta > \eta_{\rm max}$, the error grows while λ_t receives positive updates, and the loss diverges. This concludes the discussion of the theoretical model; further details can be found in Section 4 and in SM Section D.1.

3. Experimental results

In this section we test the extent to which the behavior of our theoretical model describes the dynamics of deep networks in practical settings. The theoretical results of Section 2, describing distinct learning rate phases, are not guaranteed to hold beyond the model analyzed there. We treat these results as predictions to be tested empirically, including the values $\eta_{\rm crit}$ and $\eta_{\rm max}$ of the learning rates that separate the three phases.

In a variety of deep learning settings, we find clear evidence of the different phases predicted by the model. The experiments all use MSE loss, sufficiently wide networks, and SGD[2]. Parameters such as network architecture, choice of non-linearity, weight parameterization, and regularization, do not significantly affect this conclusion.

In terms of the learning rates that determine the location of the transitions, the only modification needed to obtain good agreement with experiment is to replace the theoretical maximum learning rate, $4/\lambda_0$, with a 1-parameter function $\eta_{\rm max} = c_{\rm act.}/\lambda_0$, where $c_{\rm act.}$ is an architecture-dependent constant. We find that $c_{\rm act.} \approx 12$ for all network that use ReLU non-linearity, and it seems this parameter depends only weakly on other details of the architecture. We find the level of agreement with the experiments surprising, given that our theoretical model involves a shallow network without non-linearities.

Building on the observed correlation between lower curvature and generalization performance (Keskar et al., 2016; Jiang et al., 2020), we conjecture that optimal performance occurs in the large learning rate (catapult) phase, where the loss converges to a flatter minimum. For a fixed amount of computational budget, we find that this conjecture holds in all cases we tried. Even when comparing different learning rates trained for a fixed amount of *physical time* $t_{\rm phys} = t \cdot \eta$, we find that performance of models trained in the catapult phase either matches or exceeds that of models trained in the lazy phase.

[2]While our theoretical framework focused on (full-batch) gradient descent, we expect these the phases to happen at similar points for SGD as long as evolution is not noise dominated, in which case we expect all phases to be shifted towards smaller learning rates.

3.1. Early time curvature dynamics

Our theoretical model makes detailed predictions for the gradient descent evolution of λ, the top eigenvalue of the NTK. Here we test these predictions against empirical results in a variety of deep learning models (see the Supplement for additional experimental results).

Figure 3 shows λ during the early part of training for two deep learning settings. The results are compared against the theoretical predictions of a phase transition at $\eta_{\rm crit} = 2/\lambda_0$, and a maximum learning rate of $4/\lambda_0$. Here λ_0 is the top eigenvalue of the empirical NTK at initialization.

For learning rates $\eta < \eta_{\rm crit}$, we find that λ is independent of the learning rate and constant throughout training, as expected in the lazy phase. For $\eta_{\rm crit} < \eta < 4/\lambda_0$ we find that λ decreases during training to below $2/\eta$, matching the predicted behavior in the catapult phase (note that in the Wide ResNet example, λ initially increases before reaching its stable value).

The large learning rate behavior predicted by the model appears to persist up to the maximum learning rate, which is larger in these experiments than in the theoretical model. In these and other experiments involving ReLU networks, we find that $\eta_{\rm max} \approx 12/\lambda_0$ is a good predictor of the maximum learning rate (in the SM C.4 we discuss other nonlinearities). We conjecture that this is the typical maximum learning rate of networks with ReLU non-linearities.

Figure 3 also shows the loss initially increasing before converging in the catapult phase, confirming another prediction of the model. This transient behavior is very short, taking less than 10 steps to complete.

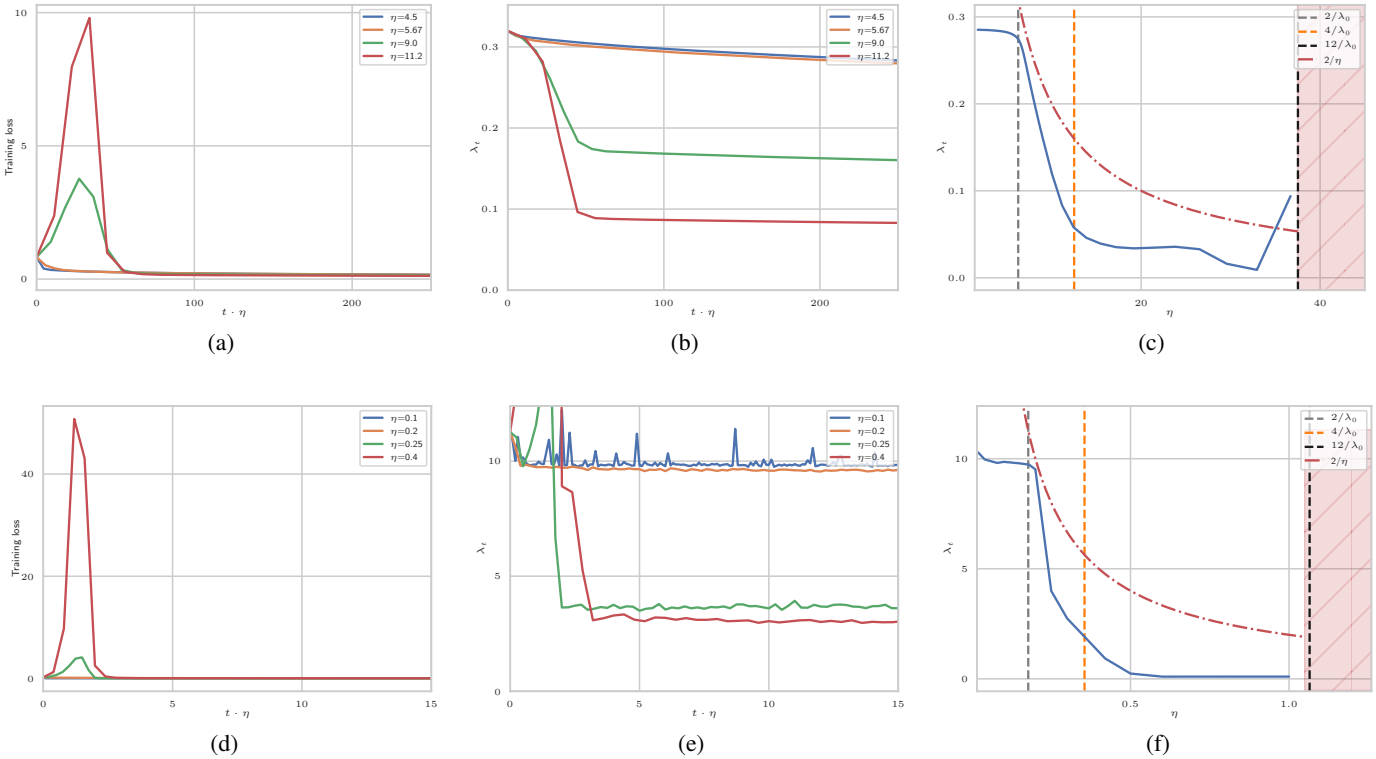


Figure 3. Early time dynamics. (a,b,c) A 3 hidden layer fully-connected network with ReLU non-linearity trained on MNIST ($\eta_{\rm crit} = 6.25$). (d,e,f) Wide ResNet 28-10 trained on CIFAR-10 ($\eta_{\rm crit} = 0.18$). Both networks are trained with vanilla SGD; for more experimental details see SM Section A. (a,d) Early time dynamics of the training loss for learning rates in the linear and catapult phases. (b,e) Early time dynamics of the curvature for learning rates in the linear and catapult phase. (c,f) λ_t measured at $t \cdot \eta = 250$ (for FC) and $t \cdot \eta = 30$ (for WRN), as a function of learning rate, compared with theoretical predictions for the locations of phase transitions. Training diverges for learning rates in the shaded region.

3.2. Generalization performance

We now consider the performance of trained models in the different phases discussed in this work. Keskar et al. (2016) observed a correlation between the flatness of a minimum found by SGD and the generalization performance (see Jiang et al. (2020) for additional empirical confirmation of this correlation). In this work, we showed that the minima SGD finds are flatter in the catapult phase, as measured by the top kernel eigenvalue. Our measure of flatness differs from that of Keskar et al. (2016), but we expect that these measures are correlated.

We therefore conjecture that optimal performance is often obtained for learning rates above $\eta_{\rm crit}$ and below the maximum learning rate.

In this section we test this conjecture empirically. We find that performance in the large learning rate range always matches or exceeds the performance when $\eta < \eta_{\rm crit}$. For a fixed compute budget, we find that the best performance is always found in the catapult phase.

Figure 4 shows the accuracy as a function of the learning rate for a fully-connected ReLU network trained on a subset of MNIST. We find that the optimal performance is achieved above $\eta_{\rm crit}$ and close to $\eta_{\rm max} = 12/\lambda_0$, the expected maximum learning rate.

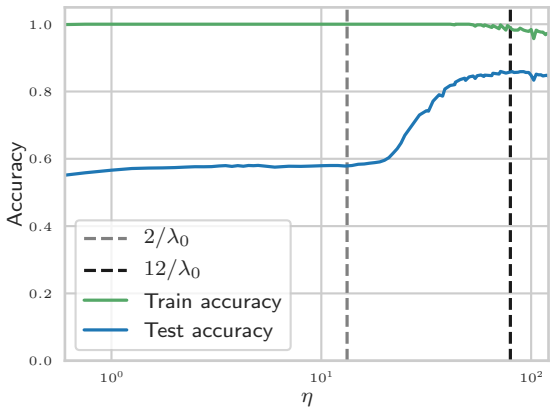


Figure 4. Final accuracy versus learning rate for a fully-connected 1 hidden layer ReLU network, trained on 512 samples of MNIST with full-batch gradient descent until training accuracy reaches 1 or 700k physical steps (see SM Section A for details). We used a subset of samples to accentuate the performance difference between phases. The optimal performance is obtained when the learning rate is above $\eta_{\rm crit}$, and close to $\eta_{\rm max}$.

Next, Figure 5 shows the performance of a convolutional network and a Wide ResNet (WRN) trained on CIFAR-10. The experimental setup, which we now describe, was chosen to ensure a fair comparison of the performance across different learning rates. The network is trained with different initial learning rates, followed by a decay at a fixed physical time $t \cdot \eta$ to the same final learning rate. This schedule is introduced in order to ensure that all experiments have the same level of SGD noise toward the end of training.

We present results using two different stopping conditions. In Figure 5a, 5c, all models were trained for a fixed number of training steps. We find a significant performance gap between small and large learning rates, with the optimal learning rate above $\eta_{\rm crit}$ and close to $\eta_{\rm max}$. Beyond this learning rate, performance drops sharply.

The fixed compute stopping condition, while of practical interest, biases the results in favor of large learning rates. Indeed, in the limit of small learning rate, training for a fixed number of steps will keep the model close to initialization. To control for this, in Figure 5b,5d models were trained for the same amount of physical time $t \cdot \eta$. For the CNN of figure 5b, decaying the learning rate does not have a significant effect on performance and we observe that performance is flat up to $\eta_{\rm max}$, and there is no correlation between our measure of curvature and generalization performance. Figure 5d shows the analogous experiment for WRN. When decaying the learning rate toward the end of training to control for SGD noise, we find that optimal performance is achieved above $\eta_{\rm crit}$. In all these cases, $\eta_{\rm max}$ is a good predictor of the maximal learning rate, despite significant differences in the architectures. Notice that by tuning the learning rate to the catapult phase, we are able to achieve performance using MSE loss, and without momentum, that is competitive with the best reported results for this

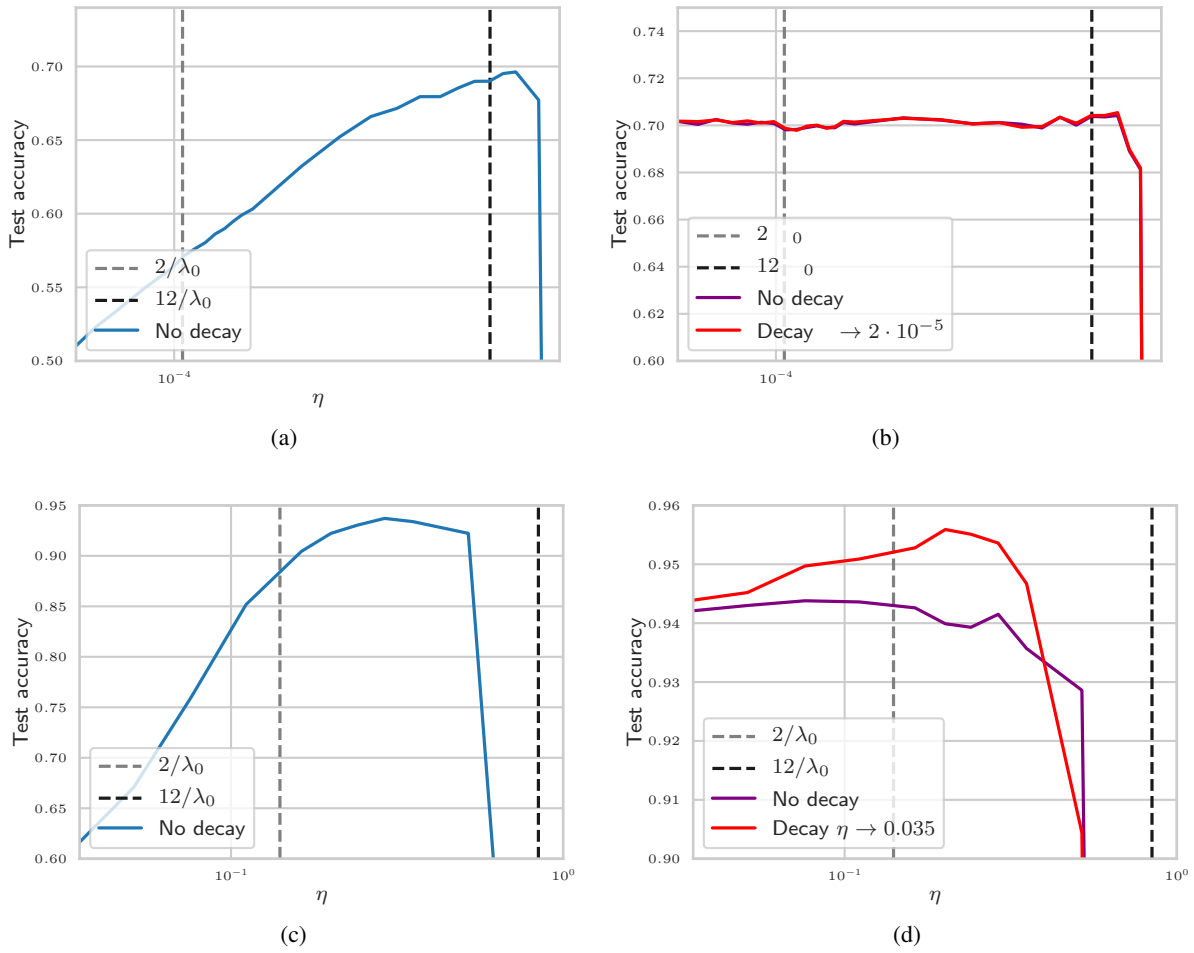


Figure 5. Test accuracy vs learning rate for (a,b) a CNN trained on CIFAR-10 using SGD with batch size 256 and L_2 regularization ($\eta_{\text{crit}} \approx 10^{-4}$) and (c,d) WRN28-10 trained on CIFAR-10 using SGD with batch size 1024, L_2 regularization, and data augmentation ($\eta_{\text{crit}} \approx 0.14$); see SM A for details. (a,c) have a fixed compute budget: (a) 437k steps and (b) 12k steps. (b,d) have been evolved for a fixed amount of physical time: (b) was evolved for $475/\eta$ steps (purple) and evolved for 50k more steps at learning rate $2 \cdot 10^{-5}$ (red) and (d) was evolved for $3360/\eta$ steps with learning rate η (purple) and then evolved for 4800 more steps at learning rate 0.035 (red). In all cases, optimal performance is achieved above η_{crit} and close to the expected maximum learning rate, in agreement with our predictions.

model (Zagoruyko & Komodakis, 2016).

In SM B.1, we present additional results for WRN on CIFAR-100, with similar conclusions as those for WRN on CIFAR-10.

4. Additional properties of the model

So far we have focused on the generalization performance and curvature of the large learning rate phase. Here we investigate additional predictions made by our model.

4.1. Restoration of linear dynamics

One striking prediction of the model is that after a period of excursion, the logit differences settle back to $\mathcal{O}(1)$ values, the NTK stops changing, and evolution is again well approximated by a linear model with constant kernel at large width.

We speculate that the return to linearity and constancy of the kernel may hold asymptotically in width for more general models for a range of learning rates above η_{crit}. We test this by evolving the model for order $\log(n)$ steps until the catapult effect is over, linearizing the model, and comparing the evolution of the two models beyond this point. Figure 6 shows an

example of this. At fixed width, the accuracy of the linear and non-linear networks match for a range of learning rates above the transition up to $4/\lambda_0$. We present additional evidence for this asymptotic linearization behavior in the Supplement.

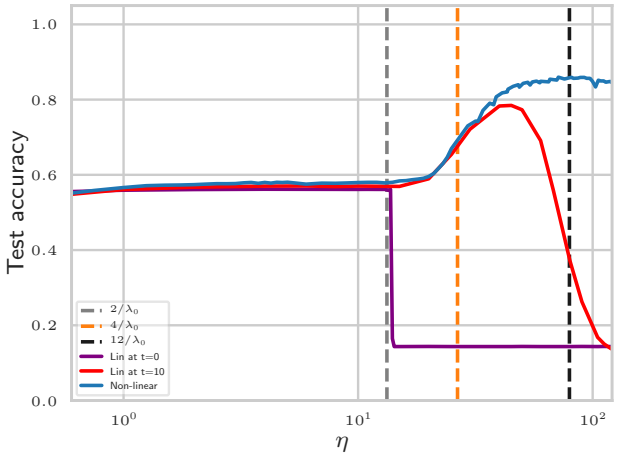


Figure 6. Evidence for linear dynamics after the catapult effect is over. Here we show the same model as in Figure 4 with the addition of models linearized at step 0 and another linearized at step 10. We observe that the model linearized after 10 steps tracks the non-linear performance in the catapult phase up to $\eta \approx 4/\lambda_0$.

4.2. Non-perturbative phase transition

The large width analysis of the small learning rate phase has been the subject of much work. In this phase, at infinite width, the network map evolves as a linear random features model, $f^{(0)}_{t+1} = f^{(0)}_t - \Theta f^{(0)}_t$, where $f^{(0)}$ is the function of the linearized model. At large but finite width, corrections to this linear evolution can be systematically incorporated via a perturbative expansion (Taylor expansion) around infinite width (Dyer & Gur-Ari, 2020; Huang & Yau, 2019).

$$f_t = f^{(0)}_t + \frac{1}{n} f^{(1)}_t + \cdots . \tag{12}$$

The evolution equations (10) and (11) of the solvable model are an example of this. At large width and in the small learning rate phase, the $O(n^{-1})$ terms are suppressed for all times. In contrast, the leading order dynamics of $f^{(0)}_t$ diverge when $\eta > \eta_{\text{crit}}$, and so the true evolution cannot be described by the linear model. Indeed, the logits grow to $\mathcal{O}(n^{1/2})$ and thus all terms in (10) and (11) are of the same order. Similarly, the growth observed empirically in the catapult phase for more general models cannot be described by truncating the series (12) at any order, because the terms all become comparable.

5. Discussion

In this work we took a step toward understanding the role of large learning rates in deep learning. We presented a dynamical mechanism that allows deep networks to be trained at larger learning rates than those accessible to their linear counterparts. For MSE loss, linear model training diverges when the learning rate is above the critical value $\eta_{\text{crit}} = 2/\lambda_0$, where λ_0 is the curvature at initialization. We showed that deep networks can train for larger learning rates by navigating to an area of the landscape that has sufficiently low curvature. Perhaps counterintuitively, training in this regime involves an initial period during which the loss increases before converging to its final, small value. We call this the *catapult effect*.

5.1. A tractable model illustrating catapult dynamics

These observations are made concrete in our theoretical model, where we fully analyze the gradient descent dynamics as a function of the learning rate. The analysis involves a modified large width limit, in which both the width and training time are taken to be large. Sweeping the learning rate from small to large, and working in the limit, we find sharp transitions from a *lazy phase* where linearized model training is stable, to a *catapult phase* in which only the full model converges, and finally to a *divergent phase* in which training is unstable. These transitions have the hallmarks of phase transitions that commonly appear in physical systems such as ferromagnets or water, as one changes parameters such as temperature. In

particular, these transitions are non-perturbative: a Taylor series expansion of the linearized model that takes into account finite width corrections is not sufficient to describe the behavior beyond the critical learning rate.

We derive the learning rates at which these transitions occur as a function of the curvature at initialization. We then treat these theoretical results as predictions, to be tested beyond the regime where they are guaranteed to hold, and find good quantitative agreement with empirical results across a variety of realistic deep learning settings.

We find it striking that a relatively simple theoretical model can correctly predict the behavior of realistic deep learning models. In particular, we conjecture that the maximum learning rate is typically a simple function of the curvature at initialization, with a single parameter $c_{\mathrm{act.}}$ that seems to depend only on the non-linearity. For ReLU networks, we conjecture that the maximum learning rate is approximately $12/\lambda_0$, which we confirm in many cases.

5.2. Reducing misalignment of activations and gradients

The catapult dynamics for the simplified model in Section 2.1 reduce curvature by shrinking the component of the first layer weights u which is orthogonal to the second layer weights v, and shrinking the component of the second layer weights v which is orthogonal to the first layer weights u. We can rewrite the simplified model in terms of a hidden layer $h = ux$, where $f(x) = n^{-1/2} v^\top h$. The gradient with respect to this hidden layer is $\frac{\partial L}{\partial h} = n^{-1/2} f(x) v$. These hidden layer gradients $\frac{\partial L}{\partial h}$ thus point in the same direction as v, while the hidden activations h point in the same direction as u. An alternative interpretation of the catapult dynamics is then that they reduce the components of h and $\frac{\partial L}{\partial h}$ which are orthogonal to each other. The catapult dynamics thus serve, in this simplified model, to reduce the misalignment between feedforward activations h, and backpropagated gradients $\frac{\partial L}{\partial h}$. We hypothesize that this reduction of misalignment between activations and gradients may be a feature of large learning rates and catapult dynamics in deep, as well as shallow, networks. We further hypothesize that it may play a directly beneficial role in generalization, for instance by making the model output less sensitive to orthogonal, out-of-distribution, perturbations of activations.

5.3. Catapult dynamics often improve generalization

Our results shed light on the regularizing effect of training at large learning rates. The effect presented here is independent of the regularizing effect of stochastic gradient noise, which has been studied extensively. Building on previous works, we noted the observed correlation between flatness and generalization performance. Based on these observations, we expect the optimal performance to often occur for learning rates larger than η_{crit}, where the linearized model is unstable. Observing this effect required controlling for several confounding factors that affect the comparison of performance between different learning rates. Under a fair comparison, and also for a fixed compute budget, we find that this expectation holds in practice.

5.4. Beyond infinite linear models

One outcome of our work is to address the performance gap between ordinary neural networks, and linear models inspired by the theory of wide networks. Optimal performance is often obtained at large learning rates which are inaccessible to linearized models. In such cases, we expect the performance gap to persist even at arbitrarily large widths. We hope our work can further improve the understanding of deep learning methods.

5.5. Other open questions

There are several remaining open questions. While the model predicts a maximum learning rate of $4/\lambda_0$, for models with ReLU activations we find that the maximum learning rate is consistently higher. This may be due to a separate dynamical curvature-reduction mechanism that relies on ReLU. In addition, we do not explore the degree to which our results extend to softmax classification. While we expect qualitatively similar behavior there, the non-constant Hessian of the softmax cross entropy makes controlled experiments more challenging. Similarly, behavior for other optimizers such as SGD with momentum may differ. For example, the maximum learning rate when training a linear model is larger for gradient descent with momentum than for vanilla gradient descent, and therefore the transition to the catapult phase (if it exists) will occur at a higher learning rate. We leave these questions to future work.

Acknowledgements

The authors would like to thank Kyle Aitken, Dar Gilboa, Justin Gilmer, Boris Hanin, Tengyu Ma, Andrea Montanari, and Behnam Neyshabur for useful discussions. We would also like to thank Jaehoon Lee for early discussions about empirical properties of the lazy phase.

References

Allen-Zhu, Z., Li, Y., and Song, Z. A convergence theory for deep learning via over-parameterization. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 242–252, Long Beach, California, USA, 09–15 Jun 2019. PMLR.

Arora, S., Du, S. S., Hu, W., Li, Z., Salakhutdinov, R. R., and Wang, R. On exact computation with an infinitely wide neural net. In *Advances in Neural Information Processing Systems*, pp. 8139–8148, 2019.

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., and Wanderman-Milne, S. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/google/jax.

Chizat, L., Oyallon, E., and Bach, F. On lazy training in differentiable programming. In Wallach, H., Larochelle, H., Beygelzimer, A., dÁlché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 32*, pp. 2933–2943. Curran Associates, Inc., 2019. URL http://papers.nips.cc/paper/8559-on-lazy-training-in-\differentiable-programming.pdf.

Daniely, A. Sgd learns the conjugate kernel class of the network. In *Advances in Neural Information Processing Systems*, pp. 2422–2430, 2017.

Dinh, L., Pascanu, R., Bengio, S., and Bengio, Y. Sharp minima can generalize for deep nets. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 1019–1028. JMLR. org, 2017.

Du, S. S., Lee, J. D., Li, H., Wang, L., and Zhai, X. Gradient descent finds global minima of deep neural networks. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, pp. 1675–1685, 2019. URL http://proceedings.mlr.press/v97/du19c.html.

Dyer, E. and Gur-Ari, G. Asymptotics of wide networks from feynman diagrams. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=S1gFvANKDS.

Frankle, J., Schwab, D. J., and Morcos, A. S. The early phase of neural network training. *arXiv preprint arXiv:2002.10365*, 2020.

Hochreiter, S. and Schmidhuber, J. Flat minima. *Neural Computation*, 9(1):1–42, 1997.

Huang, J. and Yau, H.-T. Dynamics of Deep Neural Networks and Neural Tangent Hierarchy. *arXiv e-prints*, art. arXiv:1909.08156, Sep 2019.

Jacot, A., Gabriel, F., and Hongler, C. Neural tangent kernel: Convergence and generalization in neural networks. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 31*, pp. 8571–8580. Curran Associates, Inc., 2018.

Jastrzebski, S., Szymczak, M., Fort, S., Arpit, D., Tabor, J., Cho, K., and Geras, K. The break-even point on optimization trajectories of deep neural networks. *arXiv preprint arXiv:2002.09572*, 2020.

Jiang, Y., Neyshabur, B., Krishnan, D., Mobahi, H., and Bengio, S. Fantastic generalization measures and where to find them. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=SJgIPJBFvH.

Keskar, N. S., Mudigere, D., Nocedal, J., Smelyanskiy, M., and Tang, P. T. P. On large-batch training for deep learning: Generalization gap and sharp minima. *CoRR*, abs/1609.04836, 2016. URL http://arxiv.org/abs/1609.04836.

Leclerc, G. and Madry, A. The two regimes of deep network training, 2020.

Lee, J., Bahri, Y., Novak, R., Schoenholz, S., Pennington, J., and Sohl-dickstein, J. Deep neural networks as gaussian processes. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=B1EA-M-0Z.

Lee, J., Xiao, L., Schoenholz, S., Bahri, Y., Novak, R., Sohl-Dickstein, J., and Pennington, J. Wide neural networks of any depth evolve as linear models under gradient descent. In Wallach, H., Larochelle, H., Beygelzimer, A., d' Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 32*, pp. 8570–8581. Curran Associates, Inc., 2019. URL http://papers.nips.cc/paper/9063-wide-neural-networks-of-\any-depth-evolve-as-linear-models-\under-gradient-descent.pdf.

Li, Y. and Liang, Y. Learning overparameterized neural networks via stochastic gradient descent on structured data. In *Advances in Neural Information Processing Systems*, pp. 8157–8166, 2018.

Li, Y., Wei, C., and Ma, T. Towards explaining the regularization effect of initial large learning rate in training neural networks. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 32*, pp. 11669–11680. Curran Associates, Inc., 2019.

Mandt, S., Hoffman, M. D., and Blei, D. M. Stochastic gradient descent as approximate bayesian inference. *The Journal of Machine Learning Research*, 18(1):4873–4907, 2017.

May, R. M. Simple mathematical models with very complicated dynamics. *Nature*, 261(5560):459–467, 1976.

Mei, S., Montanari, A., and Nguyen, P.-M. A mean field view of the landscape of two-layer neural networks. 115(33): E7665–E7671, 2018. doi: 10.1073/pnas.1806579115.

Naveh, Ben-David, Sompolinsky, and Ringel. to be published.

Novak, R., Xiao, L., Bahri, Y., Lee, J., Yang, G., Abolafia, D. A., Pennington, J., and Sohl-dickstein, J. Bayesian deep convolutional networks with many channels are gaussian processes. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=B1g30j0qF7.

Novak, R., Xiao, L., Hron, J., Lee, J., Alemi, A. A., Sohl-Dickstein, J., and Schoenholz, S. S. Neural tangents: Fast and easy infinite neural networks in python. In *International Conference on Learning Representations*, 2020. URL https://github.com/google/neural-tangents.

Park, D. S., Sohl-Dickstein, J., Le, Q. V., and Smith, S. L. The effect of network width on stochastic gradient descent and generalization: an empirical study. *CoRR*, abs/1905.03776, 2019. URL http://arxiv.org/abs/1905.03776.

Rotskoff, G. and Vanden-Eijnden, E. Parameters as interacting particles: long time convergence and asymptotic error scaling of neural networks. In *Advances in neural information processing systems*, pp. 7146–7155, 2018.

Sirignano, J. and Spiliopoulos, K. Mean field analysis of neural networks. *arXiv preprint arXiv:1805.01053*, 2018.

Smith, S. L. and Le, Q. V. A bayesian perspective on generalization and stochastic gradient descent. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=BJij4yg0Z.

Smith, S. L., Kindermans, P.-J., Ying, C., and Le, Q. V. Don't Decay the Learning Rate, Increase the Batch Size. *arXiv e-prints*, art. arXiv:1711.00489, Nov 2017.

Smith, S. L., Duckworth, D., Rezchikov, S., Le, Q. V., and Sohl-Dickstein, J. Stochastic natural gradient descent draws posterior samples in function space. *arXiv preprint arXiv:1806.09597*, 2018.

Woodworth, B., Gunasekar, S., Lee, J., Soudry, D., and Srebro, N. Kernel and deep regimes in overparametrized models. *arXiv preprint arXiv:1906.05827*, 2019.

Xiao, L., Pennington, J., and Schoenholz, S. S. Disentangling trainability and generalization in deep learning, 2019.

Xie, Z., Sato, I., and Sugiyama, M. A diffusion theory for deep learning dynamics: Stochastic gradient descent escapes from sharp minima exponentially fast. *arXiv preprint arXiv:2002.03495*, 2020.

Zagoruyko, S. and Komodakis, N. Wide residual networks. *CoRR*, abs/1605.07146, 2016. URL http://arxiv.org/abs/1605.07146.

Zou, D., Cao, Y., Zhou, D., and Gu, Q. Stochastic gradient descent optimizes over-parameterized deep relu networks. *arXiv preprint arXiv:1811.08888*, 2018.

Supplementary materials

A. Experimental details

We are using JAX (Bradbury et al., 2018) and the Neural Tangents Library for our experiments (Novak et al., 2020).

All the models have been trained with Mean Squared Error normalized as $\mathcal{L}(\{x,y\}_B) = \frac{1}{2k|B|}\sum_{(x,y)\in B,i}(f^i(x)-y^i)^2$, where k is the number of classes and y^i are one-targets.

In a similar way, we have normalized the NTK as $\Theta_{ij}(x,x') = \frac{1}{k|B|}\sum_\alpha \partial_\alpha f^i(x)\partial_\alpha f^j(x')$ so that the eigenvalues of the NTK are the same as the non-zero eigenvalues of the Fisher information: $\frac{1}{k|B|}\sum_{x\in B,i}\partial_\alpha f^i(x)\partial_\beta f^i(x)$.

In our experiments we measure the top eigenvalue of the NTK using Lanczos' algorithm. We construct the NTK on a small batch of data, typically several hundred samples, compute the top eigenvalue, and then average over batches. In this work, we do not focus on precision aspects such as fluctuations in the top eigenvalue across batches.

All experiments that compare different learning rates use the same seed for the weights at initialization and we consider only one such initialization (unless otherwise stated) although we have not seen much variance in the phenomena described. We let σ_w, σ_b denote the constant (width-independent) coefficient of the standard deviation of the weight and bias initializations, respectively.

Here we describe experimental settings specific to a figure.

Figure 3a,3b,3c. Fully connected, three hidden layers $w = 2048$, ReLU non-linearity trained using SGD (no momentum) on MNIST. Batch size= 512, using NTK normalization, $\sigma_w = \sqrt{2}, \sigma_b = 0$.

Figures 3d,3e,3f. Wide ResNet 28-18 trained on CIFAR10 with SGD (no momentum). Batch size of 128, LeCun initialization with $\sigma_w = \sqrt{2}, \sigma_b = 0$, $L_2 = 0$.

Figures 4,6 Fully connected network with one hidden layer and ReLU non-linearity trained on 512 samples of MNIST with SGD (no momentum). Batch size of 512, NTK initialization with $\sigma_w = \sqrt{2}, \sigma_b = 0$.

Figures 5a,5b. The convolutional network has the following architecture: $\text{Conv}_1(320) \to$ ReLU $\to \text{Conv}_2(320) \to$ ReLU $\to$ MaxPool((2,2), 'VALID') $\to \text{Conv}_1(320) \to$ ReLU $\to \text{Conv}_2(128) \to$ MaxPool((2,2), 'VALID') $\to$ Flatten() $\to \text{Dense}(256) \to$ ReLU $\to \text{Dense}(10)$. $\text{Dense}(n)$ denotes a fully-connected layer with output dimension n. $\text{Conv}_1(n), \text{Conv}_2(n)$ denote convolutional layers with 'SAME' or 'VALID' padding and n filters, respectively; all convolutional layers use $(3,3)$ filters. MaxPool((2,2), 'VALID') performs max pooling with 'VALID' padding and a (2,2) window size. LeCun initialization is used, with the standard deviation of the weights and biases drawn as $\sigma_w = \sqrt{2}$, $\sigma_b = 0.05$, respectively. Trained on CIFAR-10 with SGD, batch size of 256 and L2 regularization = 0.001.

Figures 1, 5c,5d. Wide ResNet on CIFAR10 using SGD (no momentum). Training on v3-8 TPUs with a total batch size of 1024 (and per device batch size of 128). They all use L_2 regularization= 0.0005, LeCun initialization with $\sigma_w = 1, \sigma_b = 0$. There is also data augmentation: we use flip, crop and mixup. With softmax classification, these models can get test accuracy of 0.965 if one uses cosine decay, so we don't observe a big performance decay due to using MSE. Furthermore, we are using JAX's implementation of Batch Norm which doesn't keep track of training batch statistics for test mode evaluation. We have not hyperparameter tuned for learning rates nor L_2 regularization parameter.

Figures S2,S3. Wide ResNet on CIFAR100 using SGD (no momentum). Same setting as figure 5c, 5d except for the different dataset, different L2 regularization $= 0.000025$ and label smoothing (we have subtracted 0.01 from the target one-hot labels).

Figure S7. Two hidden layer, ReLU network for one data point $x = 1, y = 1$.

Figure S10. Fully connected network with two hidden layers and tanh non-linearity trained on MNIST with SGD (no momentum). Batch size of 512, LeCun initialization with $\sigma_w = 1, \sigma_b = 0$.

Figure S8a. Two-hidden layer fully connected network trained on MNIST with batch size 512, NTK normalization with $\sigma_w = \sqrt{2}, \sigma_b = 0$. Trained using both momenta $\gamma = 0.9$ and vanilla SGD for three different non-linearities: tanh, ReLU and identity (no non-linearity). The learning rate for each non-linearity was chosen to correspond to $\eta = \frac{1}{\lambda_0}$.

Rest of SM figures. Small modifications of experiments in previous figures, specified in captions.

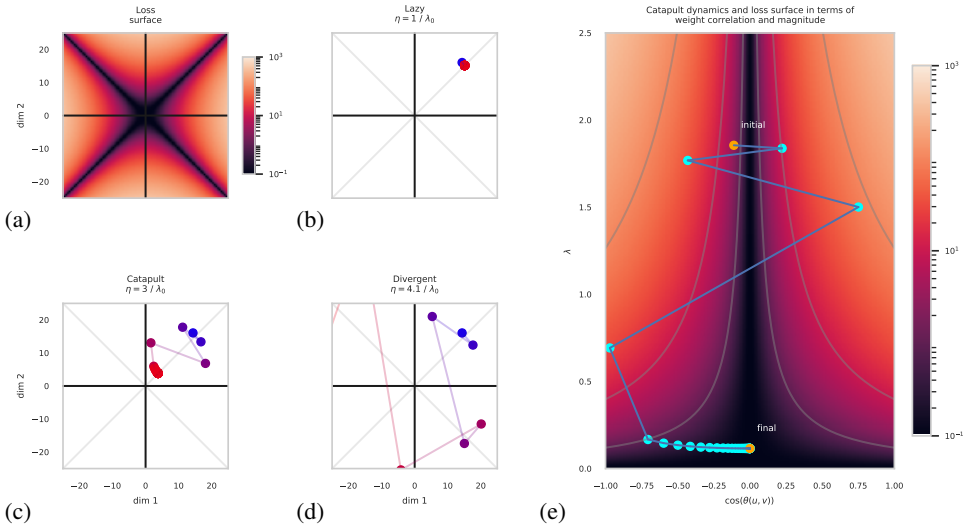


Figure S1. Visualization of training dynamics in all three phases. In the **lazy phase**, the network is approximately linear in its parameters, and converges exponentially to a global minimum. In the **catapult phase**, the loss initially grows, while the weight norm and curvature decrease. Once the curvature is low enough, optimization converges. In the **divergent phase**, both the loss and parameter magnitudes diverge. *(a)-(d)* Loss surface and training dynamics visualized in a 2d linear subspace. The network has a single hidden layer with width $n = 500$, linear activations, and is trained with MSE loss on a single 1D sample $x = 1$ with label $y = 0$. The parameter subspace is defined by $u = [\mathrm{dim1}]\, r + [\mathrm{dim2}]\, s, v = [\mathrm{dim1}]\, r - [\mathrm{dim2}]\, s$, where r and s are orthonormal vectors, $u, v \in \mathbb{R}^n$ are the weight vectors, and $[\mathrm{dim1}], [\mathrm{dim2}]$ are the coordinates in the subspace. If initialized in this 2d subspace, u_t and v_t remain in the subspace throughout training, and so training dynamics can be fully visualized with a two dimensional plot. *(e)* Visualization of the loss surface and training dynamics in terms of a nonlinear reparameterization, providing interpretable properties: *x-axis* correlation between weight vectors, *y-axis* curvature λ. The trajectory shown is identical to that in (c), and in Figure 1.

B. Experimental results: Late time performance

B.1. CIFAR-100 performance

We can also repeat the performance experiments for CIFAR-100 and the same Wide ResNet 28-10 setup. In this case, using MSE and SGD we require to evolve the system for longer times, which requires a smaller L_2 regularization. We didn't tune for it, but found that 2.5×10^{-5} works. With only one decay we can get within 3% of the Zagoruyko & Komodakis (2016) performance that used softmax classification and two learning rate decays. However, evolution for longer time is needed: we found that different learning rates converge at ≈ 2000 physical epochs. Similar to the main text experiments, we observe that if we decay after evolving for the same amount of physical epochs, larger learning rates do better. See figure S2.

B.2. Different learning rates converge at the same physical time

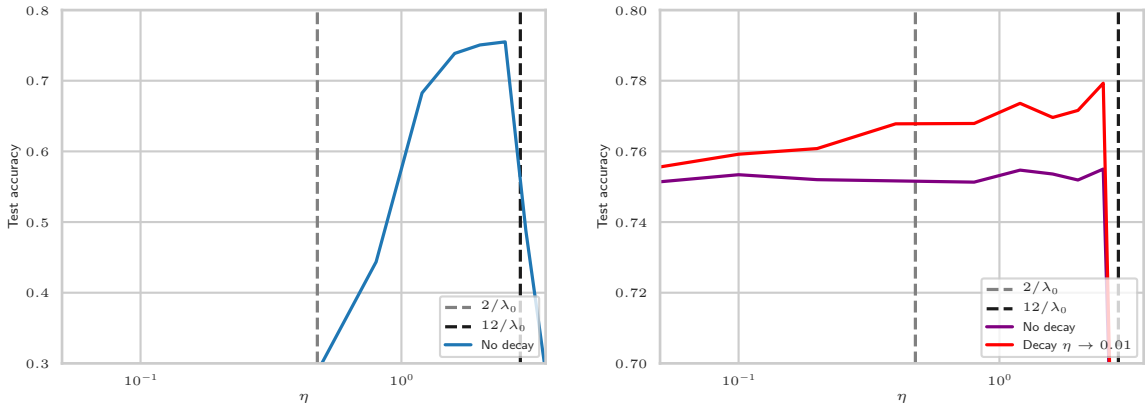
We can also plot the test accuracy versus physical time for different learning rates to show that for vanilla SGD, the performance curves of different learning rates are basically on top of each other if we plot them in physical time, which is why we find that the fair comparison between learning rates should be at the same physical time.

We have picked a subset of learning rates of the previous WRN28-18 CIFAR100 experiment of SM B.1. In figure S3, we see how even if the curves are slightly different they converge to roughly the same accuracy. The only curve which is slightly different is $\eta = 2.5$ which is a rather high learning rate (close to $\frac{12}{\lambda_0}$).

B.3. Comparison of learning rates for different L_2 regularization for WRN28-10 on CIFAR10

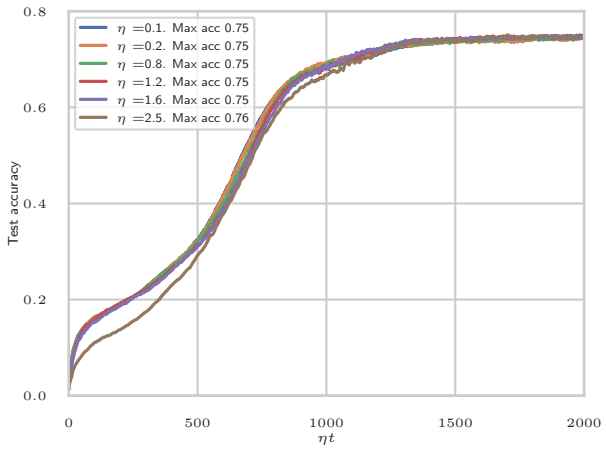
Even if in the main section we have considered a model with fixed L_2 regularization, we can study the effect without L_2 or with a different value. In these two examples, we will be considering the same setup as figures 5c,5d.

Without L_2 regularization, we see that the larger learning rate does better even in the absence of learning rate decay, although training takes a really long time. In our experience, comparing this setup with state of the art, $L_2 = 0$ regularization makes



(a) (b)

Figure S2. Test accuracy vs learning rate for WRN28-10 and CIFAR100 with vanilla SGD, L_2 regularization, data augmentation, label smoothing and batch size 1024. The critical learning rate is $\eta_{\rm crit} \approx 0.4$. (a) Evolved for 38400 steps. (b) Evolved for $96000/\eta$ steps with learning rate η (blue) and then evolved for 7200 more steps at learning rate 0.01 (red).



(a)

Figure S3. Test accuracy vs physical time for different learning rates in the WRN CIFAR100 experiment of the previous section B.1

the experiment take longer before convergence but does not influence performance much.

In the presence of L_2 regularization we picked the particular value $L_2 = 0.0005$ in order to make sure that our conclusion is not dependent on the choice of L_2, the only hyperparameter (other than η), we have considered a larger $L_2 = 0.001$. We see that the optimal performance in physical time is also peaked in the catapult phase, although the difference here is smaller.

B.4. Training accuracy plots

The training accuracies of the previous experiments are shown in figure S6.

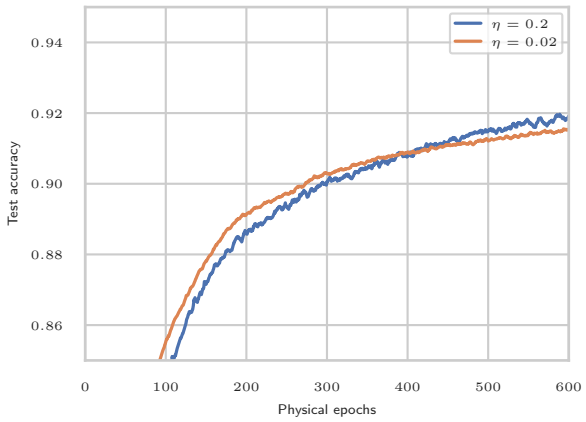


Figure S4. WRN28-10 on CIFAR10 without L_2. Same setup as 5d but evolved for longer times.

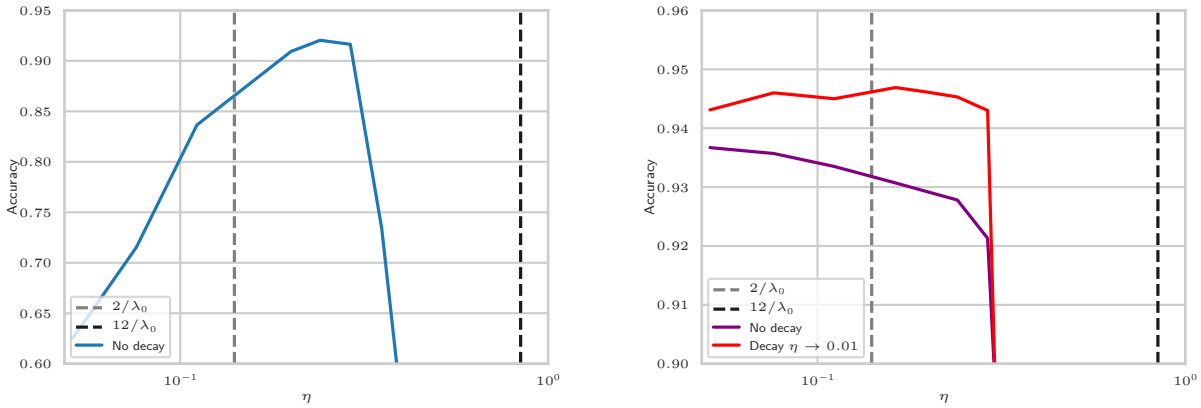


Figure S5. Test accuracies for a larger L_2 CIFAR10 experiment like that of the main section. (a) WRN CIFAR-10 7200 steps as in figure 5c. (b) WRN CIFAR10 2400 physical steps and then 4800 more steps at learning rate 0.01 as in figure 5d.

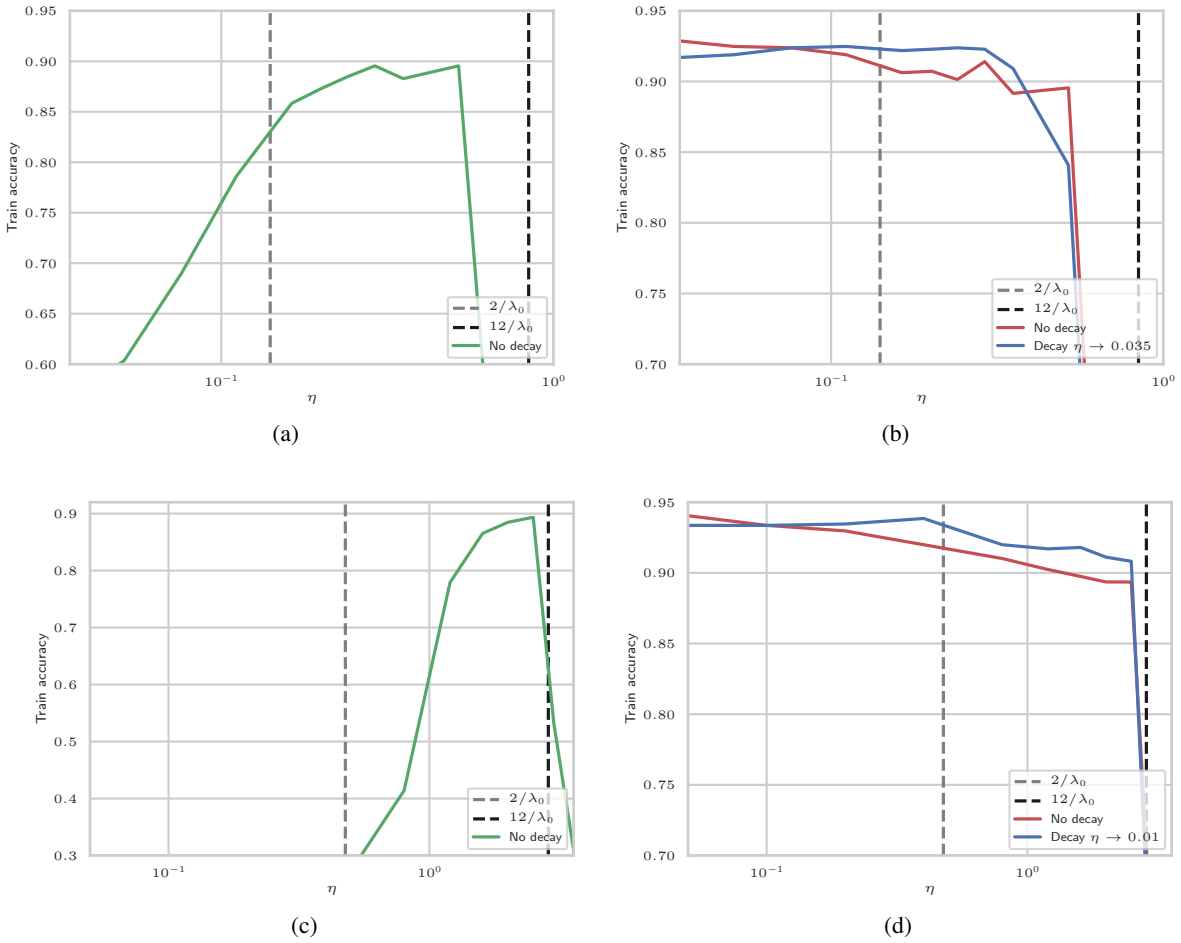
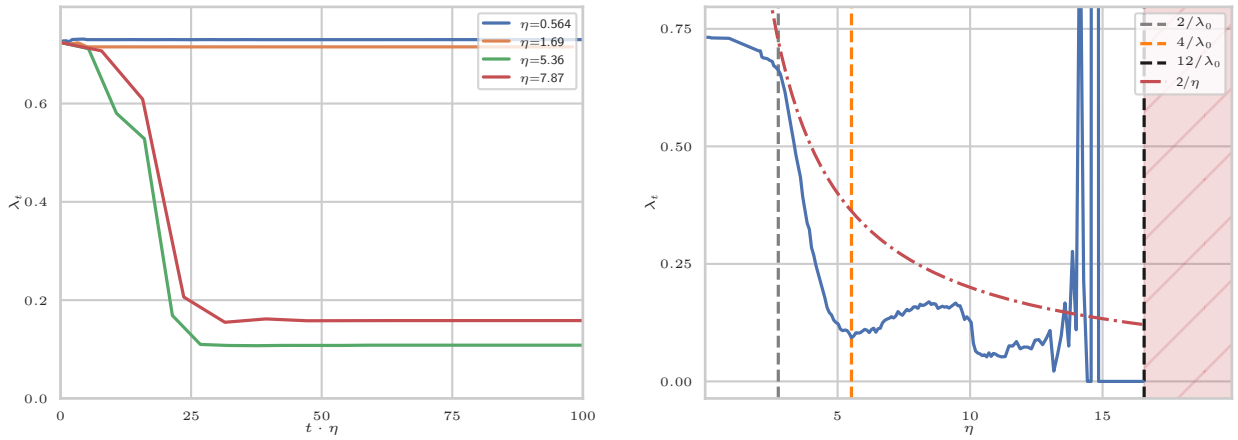


Figure S6. Training accuracies for the performance experiments. Smaller learning rates have higher training accuracy when compared in physical time. However, they still perform worse for a fixed number of steps. (a) WRN CIFAR-10 12000 steps as in figure 5c. (b) WRN CIFAR10 3360 physical steps as in figure 5d. (c) WRN CIFAR100 38400 steps as in figure S2a.(d) WRN CIFAR100 96000 physical steps as in figure S2b.

C. Experimental results: Early time dynamics

C.1. ReLU activations for the simple model

In the main text we have been using ReLU non-linearities. Compared with the simple model with no non-linearities, ReLU networks have a broader trainability regime after $\eta = \frac{4}{\lambda_0}$. It looks like these networks generically well train until $\eta = \frac{12}{\lambda_0}$. This is a generic feature of deep ReLU networks and can be already observed for the model of section 2 with a target $y = 1$, two hidden layers and a ReLU non-linearity: $f = u.ReLU(w.ReLU(v))$, as shown in figure S7). In this single sample context for $\eta \geq \frac{12}{\lambda}$, the loss doesn't diverge but the neurons die and end up giving the trivial $f = 0$ function. For deep networks with more than one hidden layer and multiple samples, as discussed in the main text, we observe that the loss diverges after $\sim \frac{12}{\lambda}$.



(a) (b)

Figure S7. Simple model ReLU non-linearity ($\eta_{\text{crit}} = 2.54$). (b) is evaluated at physical time 100.

C.2. Momenta

The effect of the optimizer also affects these dynamics. If we consider a similar setup with momenta, first we expect that a linear model converges in a broader range $\eta < \frac{2}{\lambda_0}(1+\gamma)$. For smooth non-linearities, we observe that for $\eta < \frac{2}{\lambda_0}$, the λ_t is constant. However this is not true for ReLU, see figure S8a. In fact, for ReLu networks, we observe that there is a small learning rate, roughly $\eta_{\text{eff,crit}} = \frac{\eta_{\text{crit}}}{1-\gamma}$, below which the time dynamics of λ_t is similar (but non-constant). However, for $\eta > \eta_{\text{eff,crit}}$, there are strong time dynamics, we illustrate this in figure S8b with a 3 hidden layer ReLu network.

C.3. Effect of L_2 regularization to early time dynamics

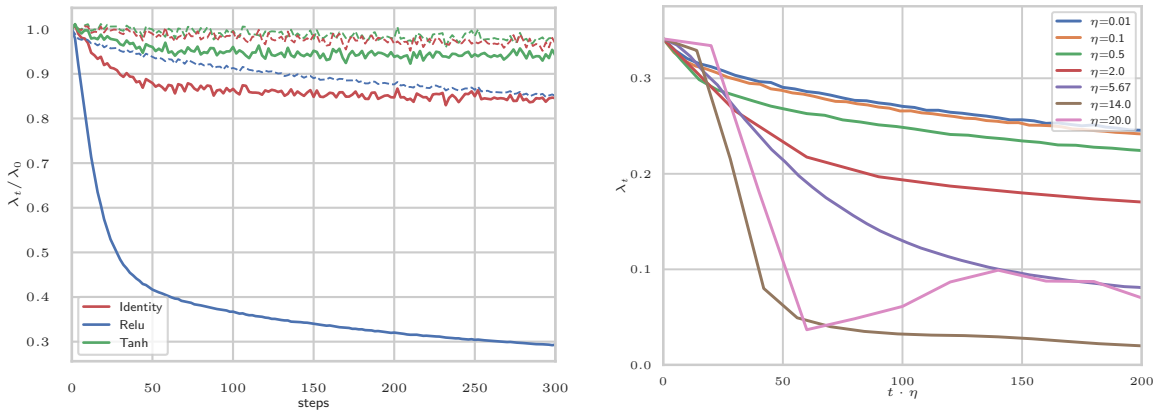
We don't expect L_2 regularization to affect the early time dynamics, but because of the strong rearrangement that goes on in the first steps, it could potentially have a non-trivial effect; among other things, the Hessian spectrum necessarily is decaying. We can see how the dynamics that drives the rearrangement is roughly the same, even in the maximum eigenvalue at early times is decreasing slowly.

C.4. Tanh activations

We observe that for Tanh activation, $\eta_{\max}$ is closer to the simple model expectation $\frac{4}{\lambda_0}$, see figure S10.

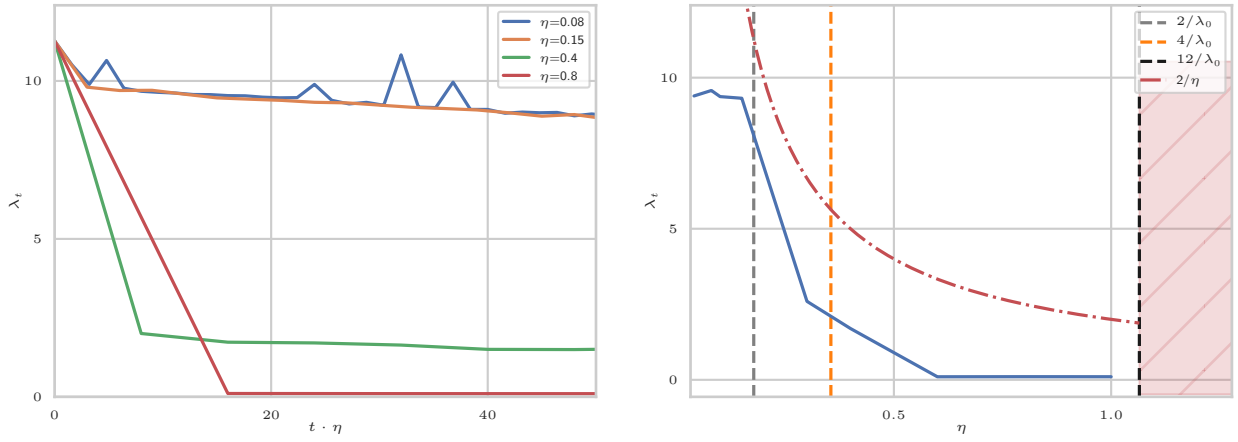
C.5. WRN NTK Normalization

As illustrated in the text in figures 3*b*, 3*c* we also see this behaviour for NTK normalization. For completeness we include the WRN model with NTK normalization. From the linearized intuition, we expect the phases to also be determined by the quantity $\eta\lambda_t$, independently of the normalization. Figure S11 has the same setup as in figure 3.



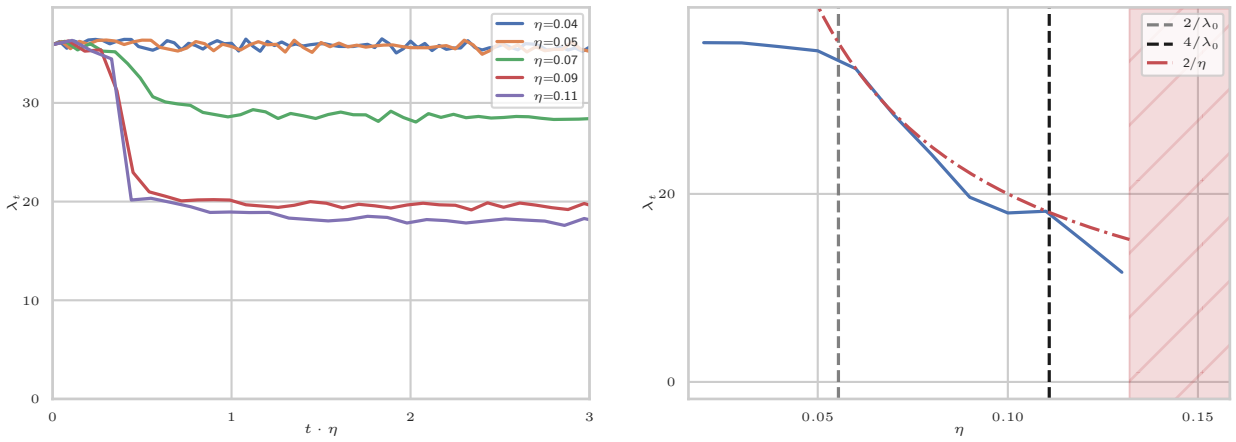
(a) (b)

Figure S8. (a) Evolution of the normalized curvature λ_t/λ_0 for $d = 2$ $w = 2048$ FC connected networks evolved with momenta (same networks with SGD with dashed line for reference) evolved for $\eta = \frac{1}{\lambda_0}$. We observe that ReLU networks evolved with momenta doesn't have a constant kernel in the naive 'lazy' phase. (b) $\eta_{\text{crit}} = 6.96, \eta_{\text{crit,eff}} = 0.69$ Same setup as the FC network of figure 3 with momenta $\gamma = 0.9$: fully connected, three hidden layers $w = 2048$, ReLU non-linearity. η_{crit} is slightly different due to variations at initialization.



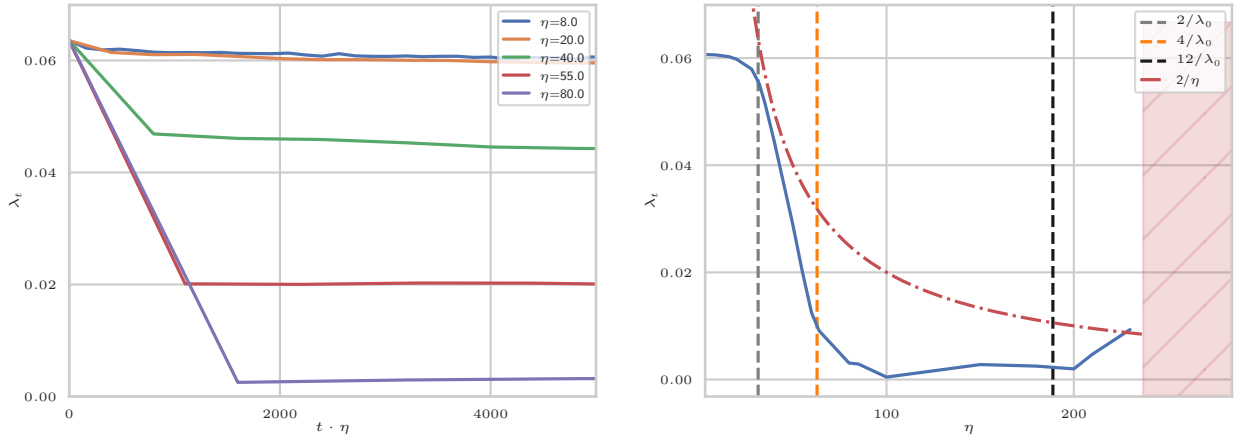
(a) (b)

Figure S9. Same WRN as figure 3d,f with L_2 regularization$= 0.0005$. Dynamics in physical steps of the λ_t and λ_t vs η. $\eta_{\text{crit}} = 0.18$ a) λ_t, b) λ_t at physical time 25



(a) (b)

Figure S10. Maximum NTK eigenvalue λ at early times for a 2 hidden layer fully connected network with tanh non-linearity trained on MNIST, with $\eta_{\text{crit}} = 0.06$. (a) Early time dynamics of the curvature for learning rates in the linear and catapult phase. (b) λ measured at $\eta t = 3$.



(a) (b)

Figure S11. Same as figures 3e,3f but with NTK normalization. a,b) Wide Resnet 28-10. $\eta_{\text{crit}} = 31.47$,$\lambda$ vs η at physical time 4000

D. Theoretical details

D.1. Full model analysis

Here we provide additional details on the theoretical analysis of the full model in Section 2.2. The gradient descent update equations are

$$u_{ia}^{t+1} = u_{ia} - \frac{\eta}{\sqrt{nm}} v_i x_{a\alpha} \tilde{f}_\alpha \,, \quad v_i^{t+1} = v_i - \frac{\eta}{\sqrt{nm}} u_{ia} x_{a\alpha} \tilde{f}_\alpha \,. \tag{S1}$$

and

$$\Theta_{\alpha\beta} = \frac{1}{nm}(|v|^2 x_\alpha^T x_\beta + x_\alpha^T u^T u x_\beta) \tag{S2}$$

The update equations for the error and kernel evaluated on training set inputs are

$$\tilde{f}_\alpha^{t+1} = (\delta_{\alpha\beta} - \eta\Theta_{\alpha\beta})\tilde{f}_\beta + \frac{\eta^2}{nm}(x_\alpha^T \zeta)(f^T \tilde{f}) \,, \tag{S3}$$

$$\begin{aligned}\Theta_{\alpha\beta}^{t+1} = \Theta_{\alpha\beta} &- \frac{\eta}{nm}\left[(x_\beta^T\zeta) f_\alpha + (x_\alpha^T \zeta) f_\beta + \frac{2}{m}(x_\alpha^T x_\beta)(\tilde{f}^T f)\right] \\ &+ \frac{\eta^2}{n^2 m}\left[|v|^2 (x_\alpha^T\zeta)(x_\beta^T\zeta) + (\zeta^T u^T u \zeta)(x_\alpha^T x_\beta)\right] \,.\end{aligned} \tag{S4}$$

Where $\zeta := \sum_\alpha \tilde{f}_\alpha x_\alpha / m \in \mathbb{R}^d$. We now consider the dynamics of the kernel projected onto the $\tilde{f}$ direction, which is given by

$$\tilde{f}^T \Theta_{t+1} \tilde{f} = \tilde{f}^T \Theta \tilde{f} + \frac{\eta}{n} \zeta^T \zeta \left(\eta \tilde{f}^T \Theta \tilde{f} - 4 f^T \tilde{f}\right) \,. \tag{S5}$$

Let us now analyze the phase structure of (S3) and (S5). For now, we neglect the last term on the right-hand side of (S3) (at initialization this term is of order n^{-1} and is negligible at large width). Let λ_0 be the maximal eigenvalue of the kernel at initialization, and let $e^{\max} \in \mathbb{R}^m$ be the corresponding eigenvector. Notice that $\tilde{f}$ projected onto the top eigenvector evolves as

$$(e^{\max})^T \tilde{f}_{t+1} = (1 - \eta\lambda) e^{\max T} \tilde{f} + \mathcal{O}(n^{-1}) \,. \tag{S6}$$

Lazy phase. When $\eta\lambda_0 < 2$, we see that $|e^{\max T} \tilde{f}^t|$ shrinks during training. The kernel updates are of order n^{-1}, while convergence happens in order n^0 steps. Therefore the kernel does not change by much during training. This is a special case of the NTK result (Jacot et al., 2018). Effectively, the model evolves as a linear model in this phase.

Catapult phase. When $2 < \eta\lambda_0 < 4$, $\|\tilde{f}\|_2$ grows exponentially fast, and it grows fastest in the $e^{\max}$ direction. Therefore, the vector $\tilde{f}$ becomes aligned with $e^{\max}$ after a number of steps that is of order n^0. Also, f itself grows quickly while the label is constant, and so we find that $f \approx \tilde{f} \approx (e^{\max T} \tilde{f}) e^{\max}$ after a similar number of steps. When these approximations hold, notice that $\tilde{f}^T \Theta \tilde{f} \approx \lambda \cdot \|\tilde{f}\|_2^2$. From equation (S5) we can then derive an approximate equation for the evolution of the top NTK eigenvalue.

$$\lambda_{t+1} \approx \lambda + \frac{\eta}{n} \zeta^T \zeta (\eta\lambda - 4) \,. \tag{S7}$$

While $\tilde{f}$ grows exponentially fast, so will ζ. When ζ_t becomes of order $n^{1/2}$, the updates to the top eigenvalue become of order n^0 (and negative), causing λ_t to decrease by a non-negligible amount. This will continue until $\lambda_t < 2/\eta$, at which point $\tilde{f}_t$ will start converging to zero. Eventually, after a number of steps of order $\log(n)$, gradient descent will converge to a global minimum that has a lower curvature than the curvature at initialization.

The justification for dropping the order n^{-1} term in (S6) was explained in the warmup model: While this term may affect the details of the dynamics, eventually the maximum kernel eigenvalue must drop below $2/\eta$ for the component $e^{\max T} \tilde{f}$ of the error (and therefore for the loss) to converge to zero.

Divergent phase. When $\eta\lambda_0 > 4$, both $\|\tilde{f}\|_2^2$ and λ will grow, and optimization will diverge. Therefore, $4/\lambda_0$ is the maximum learning rate for this model.

E. Model dynamics close to the critical learning rate

Here we consider the gradient descent dynamics of the model analyzed in Section 2, for learning rates η that are close to the critical point $\eta_{\rm crit} = 2/\lambda_0$. The analysis reveals that the gradient descent dynamics of the model are qualitatively different above and below this point. For example, the loss decreases monotonically during training when $\eta < \eta_{\rm crit}$, but not when $\eta > \eta_{\rm crit}$. In this section we show that the transition from small to large learning rate becomes sharp once we take the modified large width limit, in the following sense: certain functions of the learning rate become non-analytic at $\eta_{\rm crit}$ in the limit. This sharp transition bears close resemblance to phase transitions of the kind found in physical systems, such as the transition between the liquid and gaseous phases of water. In particular, our case involves a dynamical system, where the dynamics are governed by the gradient descent equations. These dynamics undergo a phase transition as a function of the learning rate — an external parameter. We point to the logistic map (May, 1976) as a well-known example of a dynamical system that undergoes phase transitions as a function of an external parameter.

E.1. Non-perturbative dynamics

A phase transition is a drastic change in a system's behavior incurred under a small change in external parameters. Mathematically, it is a non-analyticity in some property of the system as a function of these parameters. For example, consider the property $\lambda_*(\eta)$, the curvature of the model at the end of training as a function of the learning rate. In the modified large width limit, $\lambda_*(\eta)$ is constant for $\eta < \eta_{\rm crit}$, but not for $\eta > \eta_{\rm crit}$. Therefore, this function is not analytic at $\eta_{\rm crit}$. Notice that this statement is true in the limit but not necessarily at finite width, where the final curvature may be an analytic function of the learning rate even at $\eta_{\rm crit}$. It is well known in physics that phase transitions only occur in a limit where the number of dynamical variables (in this case the number of model parameters) is taken to infinity. One immediate consequence of the non-analyticity at $\eta_{\rm crit}$ is that the large learning rate phase is inaccessible from the small learning rate phase via a perturbative expansion. In other words, we cannot describe all properties of the model for some $\eta > \eta_{\rm crit}$ by doing a Taylor expansion around a point $\eta_0 < \eta_{\rm crit}$ and keeping a finite number of terms.

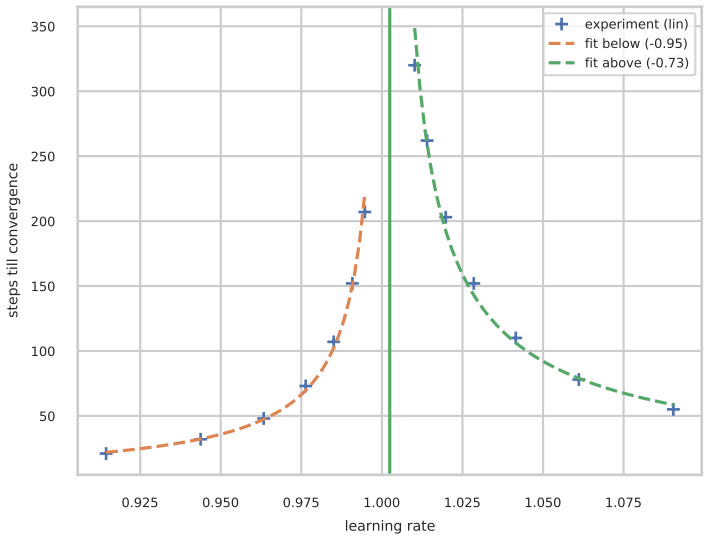
Dyer & Gur-Ari (2020); Huang & Yau (2019) developed a formalism that allows one to compute finite-width corrections to various properties of deep networks, using a perturbative expansion around the infinite width limit. We have argued that the usual infinite width approximation to the training dynamics is not valid for learning rates above $\eta_{\rm crit}$, and that a full analysis must account for large finite-width effects. One may have hoped that including the perturbative finite-width corrections discussed in Dyer & Gur-Ari (2020); Huang & Yau (2019) would allow us to regain analytic control over the dynamics. The results presented here suggest that this is not the case: For $\eta > \eta_{\rm crit}$, we expect that the perturbative expansion will not provide a good approximation to the gradient descent dynamics at any finite order in inverse width.

E.2. Critical exponents

When the external parameters are close to a phase transition, one often finds that the dynamical properties of the system obey power law behavior. The exponents of these power laws (called *critical exponents*) are of interest because they are often found to be universal, in the sense that the same set of exponents is often found to describe the phase transitions of completely different physical systems.

Here we consider $t_*(\eta)$, the number of steps until convergence, as a function of the learning rate. We will now show that t_* exhibits power-law behavior when η is close to $\eta_{\rm crit}$. For simplicity we consider the warmup model studied in Section 2. First, suppose that we are below the transition, setting $\eta\lambda_0 = 2 - \epsilon$ for some small $\epsilon > 0$. From the update equation, $f_{t+1} \approx (1 - \eta\lambda_t) f_t \approx -(1-\epsilon) f_t$ we see that f_t will converge to some fixed small value f_* after time $t_* \approx \epsilon^{-1} \log(f_*^{-1}) \sim \epsilon^{-1}$. Here we assumed that λ_t is constant in t, which is true as long as t_* is independent of n (namely we fix ϵ and then take n large). Therefore, the convergence time below the transition scales as $t_* \sim (\eta_{\rm crit} - \eta)^{-1}$, and the critical exponent is -1.

Next, suppose that $\eta\lambda_0 = 2 + \epsilon$ with $\epsilon > 0$. Now the update equation reads $f_{t+1} \approx -(1+\epsilon) f_t$. This approximation holds early during training, when the curvature updates are small. Initially, $|f_t|$ will grow until it is of order $\sqrt{n}$, at which point the updates to λ_t become of order n^0. This will happen in time $\hat{t} \sim \epsilon^{-1} \log \sqrt{n}$. Following this, the optimizer will converge. At this point $\eta\lambda_t$ is no longer tuned to be close to the transition, and so the convergence time measured from this point on will not be sensitive to ϵ. Therefore, for small ϵ the convergence time will be dominated by the early part of training, namely $t_* \approx \hat{t} \sim \epsilon^{-1}$. The critical exponent is again -1. Figure S12 show an empirical verification of this behavior.

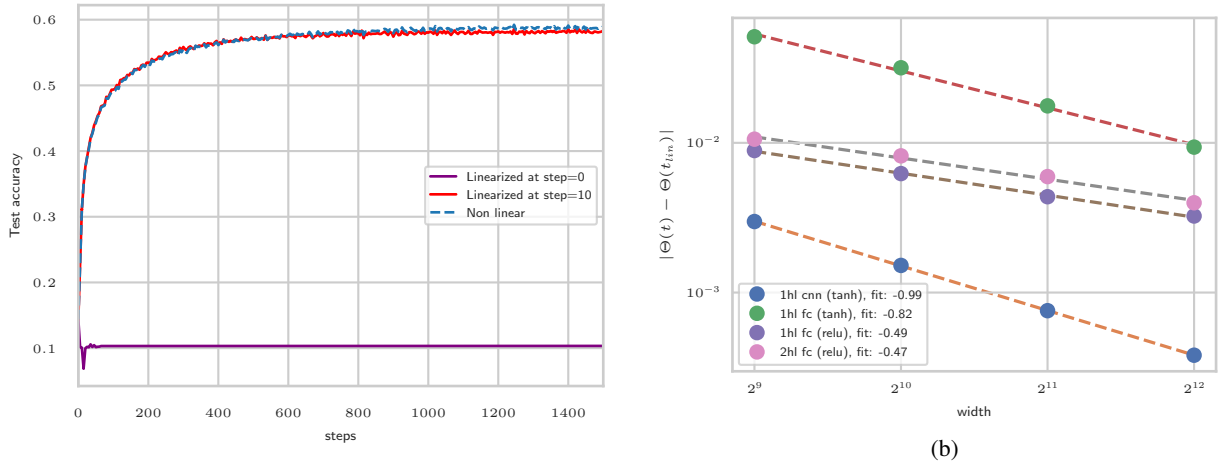


(a)

Figure S12. The convergence time diverges when the learning rate is close to the critical value $\eta_{\rm crit}$, indicated by the solid green line. The measured exponents (shown in parentheses) are close to the predicted value of -1. Experiment involves the warmup model of Section 2 with width 16,000.

F. Additional evidence for linearization in the catapult phase.

Here we present some more detailed evidence for the re-emergence of linear dynamics in the catapult phase. Figure S13 show results for models trained on subsets of MNIST with learning rates $\eta > \eta_{\rm crit}$. In figure Figure S13a we see that for a one-hidden-layer fully connected model trained on 512 MNIST images, the performance of the full non-linear model and model linearized after 10 steps track closely. Models evolve as linear models when the NTK is constant. In Figure S13b we give evidence that as networks become wider, the change in the kernel decreases.



(a) (b)

Figure S13. Evidence for a return of linear dynamics after $t_{\rm lin}$. (a,b) Show the same model as in figure 4 with the addition of linearized models at step 0 and 10. We observe that the linearized model after 10 steps tracks the non-linear performance in the 'catapult' phase up to $\eta \sim \frac{4}{\lambda_0}$ (c) The change in the NTK between $t_{\rm lin} = 50$ steps and $t = 1000$ steps decreases as the width increases. Here we consider 2-class MNIST with 100 samples per class.