AN EXPONENTIAL LEARNING RATE SCHEDULE FOR BATCH NORMALIZED NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Intriguing empirical evidence exists that deep learning can work well with exotic schedules for varying the learning rate. This paper suggests that the phenomenon may be due to Batch Normalization or BN(Ioffe & Szegedy, 2015), which is ubiquitous and provides benefits in optimization and generalization across all standard architectures. The following new results are shown about BN with weight decay and momentum (in other words, the typical use case which was not considered in earlier theoretical analyses of stand-alone BN (Ioffe & Szegedy, 2015; Santurkar et al., 2018; Arora et al., 2018)

- Training can be done using SGD with momentum and an exponentially *increasing* learning rate schedule, i.e., learning rate increases by some $(1+\alpha)$ factor in every epoch for some $\alpha > 0$. (Precise statement in the paper.) To the best of our knowledge this is the first time such a rate schedule has been successfully used, let alone for highly successful architectures. As expected, such training rapidly blows up network weights, but the net stays well-behaved due to normalization.
- Mathematical explanation of the success of the above rate schedule: a rigorous proof that it is *equivalent* to the standard setting of BN + SGD + Standard Rate Tuning + Weight Decay + Momentum. This equivalence holds for other normalization layers as well, Group Normalization(Wu & He, 2018), Layer Normalization(Ba et al., 2016), Instance Norm(Ulyanov et al., 2016), etc.
- A worked-out toy example illustrating the above linkage of hyperparameters. Using either weight decay or BN alone reaches global minimum, but convergence fails when both are used.

1 INTRODUCTION

Batch Normalization (BN) offers significant benefits in optimization and generalization across architectures, and has become ubiquitous. Usually best performance is attained by adding weight decay and momentum in addition to BN.

Usually weight decay is thought to improve generalization by controlling the norm of the parameters. However, it is fallacious to try to separately think of optimization and generalization because we are dealing with a nonconvex objective with multiple optima. Even slight changes to the training surely lead to a different trajectory in the loss landscape, potentially ending up at a different solution! One needs trajectory analysis to have a hope of reasoning about the effects of such changes.

In the presence of BN and other normalization schemes, including GroupNorm, LayerNorm, and InstanceNorm, the optimization objective is *scale invariant* to the parameters, which means rescaling parameters would not change the prediction[1]. The current paper introduces new modes of analysis for such settings. This rigorous analysis yields the surprising conclusion that the original learning rate (LR) schedule and weight decay can be folded into a new exponential schedule for learning rate: in each iteration multiplying it by $(1+\alpha)$ for some $\alpha > 0$ that depends upon the momentum and weight decay rate.

[1]Often the parameters that compute the output do not have BN and are thus not scale-invariant. In our experiments we found that fixing the output layer randomly doesn't harm the performance of the network. So the trainable parameters satisfy scale-invariance

Theorem 1.1 (Main, Informal)**.** *SGD on a scale-invariant objective with initial learning rate η, weight decay factor λ, and momentum γ is equivalent to SGD where at iteration t, the learning rate $\tilde{\eta}_t$ in the new exponential learning rate schedule is defined as $\tilde{\eta}_t = \alpha^{-2t-1}\eta$, where α is a non-zero root of equation*

$$x^2 - (1 + \gamma - \lambda\eta)x + \gamma = 0.$$

Specifically, when momentum $\gamma = 0$, the above schedule can be simplified as $\tilde{\eta}_t = (1 - \lambda\eta)^{-2t-1}\eta$ and $\tilde{\gamma} = 0$.

The above theorem requires that the product of learning rate and weight decay factor, $\lambda\eta$, is small compared to $1 - \gamma$, which is almost always satisfied in practice. The rigorous and most general version of above theorem is Theorem 2.5, which deals with adaptive learning rate schedule, momentum and weight decay.

Such an exponential increase in learning rate seems absurd at first sight and to the best of our knowledge, no deep learning success has been reported using such an idea before. It does highlight the above-mentioned viewpoint that in deep learning, optimization and regularization are not easily separated. Of course, the exponent trumps the effect of initial lr very fast, which serves as another explanation of the standard wisdom that initial lr is unimportant when training with BN.

Note that it is customary in BN to switch to a lower learning rate upon reaching a plateau in the validation loss. According to the analysis in the above theorem, this corresponds to an exponential growth with a smaller exponent, except for a transient effect when a correction term is needed for the two processes to be equivalent (see discussion around Theorem 2.4).

Thus the final training algorithm is roughly as follows: *Start from a convenient learning rate like* 0.1*, and grow it at an exponential rate with a suitable exponent. When validation loss plateaus, switch to an exponential growth of lr with a lower exponent. Repeat the procedure until the training loss saturates.*

In Section 3, we demonstrate on a toy example how weight decay and normalization are inseparably involved in the optimization process. With either weight decay or normalization alone, SGD is guaranteed to achieve zero training error. But with both turned on, SGD fails to converge to global minimum.

In Section 4 of experiments, we verify our theoretical findings on CNNs and ResNets. We also construct better Exponential learning rate schedules by incorporating the Cosine learning rate schedule, which opens the possibility of even more general theory of rate schedule tuning towards better performance.

1.1 Related Work

There have been other theoretical analyses of training models with scale-invariance. (Cho & Lee, 2017) proposed to run Riemmanian gradient descent on Grassmann manifold $\mathcal{G}(1, n)$ since the weight matrix is scaling invariant to the loss function. (Hoffer et al., 2018) observed that the effective stepsize is proportional to $\frac{\eta_w}{\|\boldsymbol{w}_t\|^2}$. (Arora et al., 2019) show the gradient is always perpendicular to the current parameter vector which has the effect that norm of each scale invariant parameter group increases monotonically, which has an auto-tuning effect. (Wu et al., 2018) proposes a new adaptive learning rate schedule motivated by scale-invariance property of Weight Normalization.

Previous work for understanding Batch Normalization. (Santurkar et al., 2018) suggested that the success of BNhas does not derive from reduction in Internal COvariate Shift, but by making landscape smoother. (Kohler et al., 2018) shows 2-layer linear nets with BN could achieve expoential convergence rate, but their analysis is for a variant of GD with an inner optimization loop rather than GD itself. (Bjorck et al., 2018) observe that the higher learning rates enabled by BN empirically improves generalization. (Arora et al., 2019) prove that with certain mild assumption, (S)GD with BN finds approximate first order stationary point with any fixed learning rate. None of the above analyses incorporated weight decay, but (Zhang et al., 2019) argued qualitatively that weight decay makes parameters have smaller norms, and thus larger effective learning rate. None of the above analyses deals with momentum.

1.2 PRELIMINARIES AND NOTATIONS

For batch $\mathcal{B} = \{x_i\}_{i=1}^{B}$, network parameter $\boldsymbol{\theta}$, we use $f_{\boldsymbol{\theta}}$ to denote the network and use $L_t(f_{\boldsymbol{\theta}}) = L(f_{\boldsymbol{\theta}}, \mathcal{B}_t)$ to denote the loss function at iteration t. When there's no ambiguity, we also use $L_t(\boldsymbol{\theta})$ for convenience.

We say a loss function $L(\boldsymbol{\theta})$ is *scale invariant* to its parameter $\boldsymbol{\theta}$ is for any $c \in \mathbb{R}^+$, $L(\boldsymbol{\theta}) = L(c\boldsymbol{\theta})$. In practice, the source of scale invariance is usually different types of normalization layers, including Batch Normalization(Ioffe & Szegedy, 2015), Group Normalization(Wu & He, 2018), Layer Normalization(Ba et al., 2016), Instance Norm(Ulyanov et al., 2016), etc.

Implementations of SGD with Momentum/Nesterov comes with subtle variations in literature. We adopt the variant from (Sutskever et al., 2013), also the default in PyTorch(Paszke et al., 2017). $L2$ *regularization* (a.k.a. *Weight Decay*) is another common trick used in deep learning. Combining them together, we get the one of the mostly used optimization algorithms below.

Definition 1.2. [SGD with Momentum and Weight Decay][cite] At iteration t, with randomly sampled batch $\mathcal{B}_t$, update the parameters $\boldsymbol{\theta}_t$ and momentum $\boldsymbol{v}_t$ as following:

$$\boldsymbol{\theta}_t = \boldsymbol{\theta}_{t-1} - \eta_t \boldsymbol{v}_t \tag{1}$$

$$\boldsymbol{v}_t = \gamma \boldsymbol{v}_{t-1} + \nabla_{\boldsymbol{\theta}} \left(L_t(\boldsymbol{\theta}_{t-1}) + \frac{\lambda_{t-1}}{2} \|\boldsymbol{\theta}_{t-1}\|^2 \right), \tag{2}$$

where η_t is the learning rate at epoch t, γ is the momentum coefficient, and λ is the factor of weight decay. Usually, $\boldsymbol{v}_t$ is initialized to be 0.

For ease of analysis, we will use the following equivalent of Definition 1.2.

$$\frac{\boldsymbol{\theta}_t - \boldsymbol{\theta}_{t-1}}{\eta_{t-1}} = \gamma \frac{\boldsymbol{\theta}_{t-1} - \boldsymbol{\theta}_{t-2}}{\eta_{t-2}} - \nabla_{\boldsymbol{\theta}} \left((L(\boldsymbol{\theta}_{t-1}) + \frac{\lambda_{t-1}}{2} \|\boldsymbol{\theta}_{t-1}\|_2^2 \right), \tag{3}$$

where η_{-1} and $\boldsymbol{\theta}_{-1}$ must be chosen in a way such that $\boldsymbol{v}_0 = \frac{\boldsymbol{\theta}_0 - \boldsymbol{\theta}_{-1}}{\eta_{-1}}$ is satisfied, e.g. $\boldsymbol{\theta}_{-1} = \boldsymbol{\theta}_0$ and η_{-1} could be arbitrary.

2 DERIVING EXPONENTIAL LEARNING RATE SCHEDULE

As warmup in Section 2.1 we show how to interpret *Fixed LR + Fixed WD + Fixed Momentum* as an equivalent *Exponential LR + Fixed Momentum.* However, usually in deep learning fixed LR is insufficient to reach full training accuracy and instead one needs a few phases where LR is reduced by some factor between phases. Section 2.2 shows how to interpret such a multiphase LR schedule + WD as a certain multiphase exponential LR schedule.

In principle all results can be derived from the Main Theorem 2.5 but that is harder to understand. Hence the simpler Theorems 2.1 and Theorem 2.4 are given separately. There are also other possible variants where momentum can also be changed (as discussed later in experiments section) but for simplicity momentum is left unchanged.

2.1 REPLACING WD BY EXPONENTIAL LEARNING RATE: CASE OF CONSTANT LR

In this subsection, we use notation of Section 1.2 and assume η (LR), γ (Momentum) and λ (WD) are fixed. The following lemma shows how to replace WD with an exponential LR schedule.

Theorem 2.1. *The following two sequences of parameters ,$\{\boldsymbol{\theta}_t\}_{t=0}^{\infty}$ and $\{\widetilde{\boldsymbol{\theta}}_t\}_{t=0}^{\infty}$, define the same sequence of network functions, i.e. $f_{\boldsymbol{\theta}_t} = f_{\widetilde{\boldsymbol{\theta}}_t}$, $\forall t \in \mathbb{N}$, given $\widetilde{\boldsymbol{\theta}}_0 = \boldsymbol{\theta}_0$, $\widetilde{\boldsymbol{\theta}}_{-1} = \boldsymbol{\theta}_{-1}\alpha$.*

1. $\boldsymbol{\theta}_t - \boldsymbol{\theta}_{t-1} = \gamma(\boldsymbol{\theta}_{t-1} - \boldsymbol{\theta}_{t-2}) - \eta \nabla_{\boldsymbol{\theta}} (L(\boldsymbol{\theta}_{t-1}) + \frac{\lambda}{2} \|\boldsymbol{\theta}_{t-1}\|_2^2)$
2. $\widetilde{\boldsymbol{\theta}}_t - \widetilde{\boldsymbol{\theta}}_{t-1} = \gamma(\widetilde{\boldsymbol{\theta}}_{t-1} - \widetilde{\boldsymbol{\theta}}_{t-2}) - \alpha^{-2t-1} \eta \nabla_{\boldsymbol{\theta}} L(\widetilde{\boldsymbol{\theta}}_{t-1})$

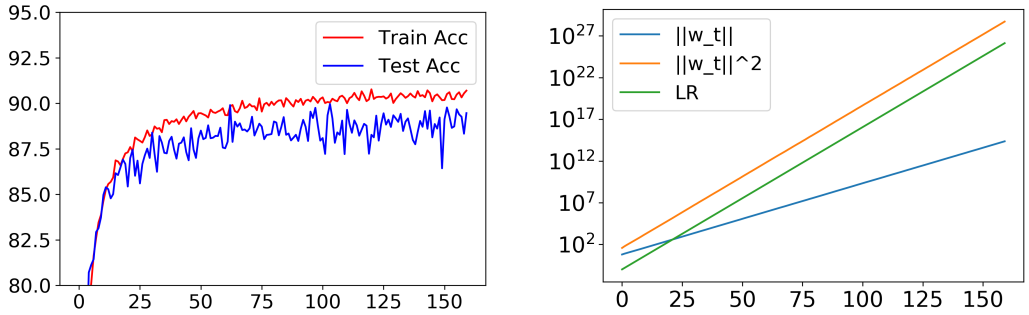

Figure 1: Taking PreResNet32 with published hyperparameters and replacing WD during first phase (Fixed LR) by exponential LR according to Theorem 2.1 to the schedule $\widetilde{\eta}_t = 0.1 \times 1.481^t$, momentum 0.9. Plot on right shows weight norm $\boldsymbol{w}$ of the first convolutional layer in the second residual block grows exponentially, satisfying $\frac{\|\boldsymbol{w}_t\|^2}{\widetilde{\eta}_t} =$ constant. Reason being that according to the proof it is essentially the norm square of the weights when trained with Fixed LR + WD + Momentum, and published hyperparameters kept this norm roughly constant during training.

where α is a positive root of equation

$$x^2 - (1 + \gamma - \lambda\eta)x + \gamma = 0, \tag{4}$$

which is always smaller than 1. When $\lambda = 0$, then $\alpha = \gamma$ is the unique non-zero solution.

Remark 2.2. *We implicitly assume λ and γ are small enough such that α_t are always positive, which is always true in practice, and so is P_t. (In case of constant LR, we need the quadratic equation to have at least one positive root.) The reason behind this requirement is that our definition of "scale invariance" only holds for all positive scalar c, since in neural networks, multiplying activations by -1 before a normalization layer usually yields completely different outputs. Using very high rate of weight decay can flip the sign of the weight.*

2.2 Replacing WD by Exponential LR: Case of multiple LR phases

Usual practice in deep learning shows that reaching full training accuracy requires reducing the learning rate a few times.

Definition 2.3. *Step Decay* is the (standard) learning rate schedule, where training has K phases, where phase I starts at iteration T_I, and all iterations in phase I use a fixed learning rate of η_I.

Translating WD into the learning rate here leads to the following result. We give an informal version here, and the exact version is Theorem 2.7 later in Section 2.2. The version below is correct up to a correction term in the learning rate schedule. The correction is nontrivial only in the first few iterations of the phase. It is also nontrivial at the end of training when learning rate in the original schedule falls a lot, like 0.001. These effects are also explored in the experiments, where we empirically find on CIFAR10 that ignoring the correction term does not change performance much.

Theorem 2.4 (Tapered-Exponential LR Schedule)**.** *(Informal) If WD is turned off, the following sequence of learning rates $\{\widetilde{\eta}_t\}$ is almost equivalent throughout phase I, ($\widetilde{\eta}_0 = \eta_1$)*

$$\widetilde{\eta}_{t+1} = \begin{cases} \widetilde{\eta}_t \times \left(\frac{1+\gamma-\lambda\eta_I+\sqrt{(1+\gamma-\lambda\eta_I)^2-4\gamma}}{2} \right)^{-2} & \text{if } t \neq T_I \text{ for some } I \\ \widetilde{\eta}_t \times \frac{\eta_I}{\eta_{I-1}} \times \left(\frac{1+\gamma-\lambda\eta_I+\sqrt{(1+\gamma-\lambda\eta_I)^2-4\gamma}}{2} \right)^{-2} & \text{if } t = T_I \text{ for some } I \end{cases} \tag{5}$$

2.3 Proof Sketch

In this subsection, we will first give the most general statement of the equivalence .

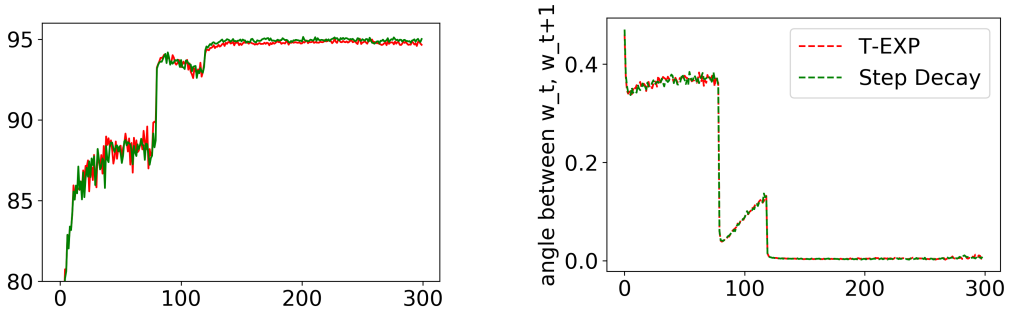

Figure 2: PreResNet32 trained with standard Step Decay and its corresponding Tapered-Exponential LR schedule. As predicted by Theorem 2.4, they have similar trajectories and performances.

Theorem 2.5 (Main Theorem)**.** *The following two sequences of parameters ,*$\{\boldsymbol{\theta}_t\}_{t=0}^{\infty}$ *and* $\{\widetilde{\boldsymbol{\theta}}_t\}_{t=0}^{\infty}$*, define the same sequence of network functions, i.e.* $f_{\boldsymbol{\theta}_t} = f_{\widetilde{\boldsymbol{\theta}}_t}$*,* $\forall t \in \mathbb{N}$*, given the initial conditions,* $\widetilde{\boldsymbol{\theta}}_0 = P_0\boldsymbol{\theta}_0$*,* $\widetilde{\boldsymbol{\theta}}_{-1} = P_{-1}\boldsymbol{\theta}_{-1}$*.*

1. $\frac{\boldsymbol{\theta}_t - \boldsymbol{\theta}_{t-1}}{\eta_{t-1}} = \gamma\frac{\boldsymbol{\theta}_{t-1} - \boldsymbol{\theta}_{t-2}}{\eta_{t-2}} - \nabla_{\boldsymbol{\theta}}\left((L(\boldsymbol{\theta}_{t-1}) + \frac{\lambda_{t-1}}{2}\|\boldsymbol{\theta}_{t-1}\|_2^2\right)$*, for* $t = 1, 2, \ldots$*;*
2. $\frac{\widetilde{\boldsymbol{\theta}}_t - \widetilde{\boldsymbol{\theta}}_{t-1}}{\widetilde{\eta}_{t-1}} = \gamma\frac{\widetilde{\boldsymbol{\theta}}_{t-1} - \widetilde{\boldsymbol{\theta}}_{t-2}}{\widetilde{\eta}_{t-2}} - \nabla_{\boldsymbol{\theta}}L(\widetilde{\boldsymbol{\theta}}_{t-1})$*, for* $t = 1, 2, \ldots$*,*

where $\widetilde{\eta}_t = P_tP_{t+1}\eta_t$*,* $P_t = \prod_{i=-1}^{t}\alpha_i^{-1}$*,* $\forall t \geq -1$ *and* α_t *recursively defined as*

$$\alpha_t = -\eta_{t-1}\lambda_{t-1} + 1 + \frac{\eta_{t-1}}{\eta_{t-2}}\gamma(1 - \alpha_{t-1}^{-1}), \forall t \geq 1. \tag{6}$$

needs to be always positive. Here α_0, α_{-1} *are free parameters. Different choice of* α_0, α_{-1} *would lead to different trajectory for* $\{\widetilde{\boldsymbol{\theta}}_t\}$*, but the equality that* $\widetilde{\boldsymbol{\theta}}_t = P_t\boldsymbol{\theta}_t$ *is always satisfied. If the initial condition is given via* $\boldsymbol{v}_0$*, then it's also free to choose* $\lambda_{-1}, \boldsymbol{\theta}_{-1}$*, as long as* $\frac{\boldsymbol{\theta}_0 - \boldsymbol{\theta}_{-1}}{\lambda_{-1}} = \boldsymbol{v}_0$*.*

Now we are ready to give the exact version of Theorem 2.4.

Definition 2.6 (Tapered-Exponential LR Schedule, Full version)**.** Given a Step Decay LR schedule with $\{T_I\}_{I=1}^{K}, \{\eta_I\}_{I=1}^{K}$, the corresponding *Tapered-Exponential* schedule is the following ($\alpha_0 = \alpha_{-1} = 1$):

$$\alpha_t = \begin{cases} -\eta_{I-1}\lambda + 1 + \gamma(1 - \alpha_{t-1}^{-1}), \forall T_{I-1} \leq t \leq T_I - 1; \\ -\eta_I\lambda + 1 + \frac{\eta_I}{\eta_{I-1}}\gamma(1 - \alpha_{t-1}^{-1}), \forall t = T_I; \end{cases}$$
$$P_t = \Pi_{i=1}^{t}\alpha_t^{-1};$$
$$\hat{\eta}_t = P_tP_{t+1}\eta_t.$$

Theorem 2.7 (Rigorous Theorem 2.4)**.** *The schedule of Theorem 2.4 is the same as that of Definition 2.6 throughout phase I,* ($\widetilde{\eta}_0 = \eta_1$)*, in the sense that*

$$\left|\frac{\hat{\eta}_{t-1}}{\hat{\eta}_t} - \frac{\widetilde{\eta}_{t-1}}{\widetilde{\eta}_t}\right| < \frac{3}{2}\left(\frac{\gamma}{(z_1^1)^2}\right)^{t-T_I} = \frac{3}{2}\left(\frac{z_2}{z_1}\right)^{t-T_I}, \quad \forall T_I + 1 \leq t \leq T_{I+1} - 1.$$

where z_1^1 *is the larger root of* $x^2 - (1+\gamma-\lambda\eta_1)x + \gamma = 0$*. In Appendix A, we show that* $z_1 \geq 1 - \frac{\eta}{1-\gamma}$*. When* $\lambda\eta$ *is small compared to* $1 - \gamma$*, which is usually the case in practice, one could approximate* z_1^1 *by 1.*

Before sketching the proof, we restate a simple but key lemma from (Arora et al., 2019).

Lemma 2.8 (Scale Invariance). *If for any* $c \in \mathbb{R}^+$, $L(\boldsymbol{\theta}) = L(c\boldsymbol{\theta})$, *then*
(1). $\langle \nabla_{\boldsymbol{\theta}} L, \boldsymbol{\theta} \rangle = 0$;
(2). $\nabla_{\boldsymbol{\theta}} L\big|_{\boldsymbol{\theta}=\boldsymbol{\theta}_0} = c\nabla_{\boldsymbol{\theta}} L\big|_{\boldsymbol{\theta}=c\boldsymbol{\theta}_0}$, *for any* $c > 0$

Proof Sketch of Theorem 2.5. This core of this proof relies on the property of scale invariance property of normalization layers, which allows us to have access to the gradients of $\boldsymbol{\theta}_{t-1}$ from its scaled version, $\widetilde{\boldsymbol{\theta}}_{t-1}$.

The proof is based on induction — assuming $\widetilde{\boldsymbol{\theta}}_{t-1} = P_t\boldsymbol{\theta}_{t-1}$, $\widetilde{\boldsymbol{\theta}}_{t-2} = P_{t-2}\boldsymbol{\theta}_{t-2}$, and using Lemma 2.8 we could replace $\nabla_{\boldsymbol{\theta}}(L(\widetilde{\boldsymbol{\theta}}_{t-1})$ by $\frac{\nabla_{\boldsymbol{\theta}}(L(\boldsymbol{\theta}_{t-1})}{P_{t-1}}$, and thus have all three basis for $\boldsymbol{\theta}_t$ in hand. The goal is now reduced to pick a suitable $\widetilde{\eta}_t$ such that the coefficients in update rule 2 are the same of those in Update rule 1, under global rescaling. This rescaling will be called P_t. In the full proof we show we can always find such $\widetilde{\eta}_t$ given proper initial condition. □

3 Example illustrating interplay of WD and BN

The paper so far has shown that effects of different hyperparameters in training are not easily separated, since their combined effect on the trajectory is complicated. We give a simple example to illustrate this, where convergence is guaranteed if we use either BatchNorm or weight decay in isolation, but convergence fails if both are used. (Momentum is turned off for clarity of presentation, but the analysis below could be easily extended to the case with momentum. See Appendix A).

Setting: Suppose we are fine-tuning the last linear layer of the network, where the input of the last layer is assumed to follow a standard Gaussian distribution $\mathcal{N}(0, I_m)$, where m is the input dimension of last layer. We also assume this is a binary classification task with logistic loss, $l(u, y) = \ln(1+\exp(-uy))$, where label $y \in \{-1, 1\}$ and $u \in \mathbb{R}$ is the output of the neural net. For simplicity we assume the the input of the last layer are already separable, and w.l.o.g. we assume the label is equal to the sign of the first coordinate of $\boldsymbol{x} \in \mathbb{R}^m$, namely sign (x_1) . Thus the training loss and training error are simply

$$L(\boldsymbol{w}) = \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{N}(0,I_m), y=\text{sign}(x_1)} \left[\ln(1+\exp(-\boldsymbol{x}^\top \boldsymbol{w} y))\right], \qquad \Pr_{\boldsymbol{x}\sim\mathcal{N}(0,I_m), y=\text{sign}(x_1)} \left[\boldsymbol{x}^\top \boldsymbol{w} y \le 0\right] = \arccos \frac{w_1}{\|\boldsymbol{w}\|}$$

Case 1: WD alone: Since both the above function and L2 regularization are convex $\boldsymbol{w}$,vanilla SGD *with suitably small learning rate* could get arbitrarily close to the global minimum for this regularized objective, which has 100% training accuracy.

Case 2: BN alone: Add a BN layer after the linear layer(here we use global batch statistics for simplicity), and fix scalar and bias term to 1 and 0. The objective becomes

$$L_{BN}(\boldsymbol{w}) = \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{N}(0,I_m), y=\text{sign}(x_1)} \left[\ln(1+\exp(-\boldsymbol{x}^\top \frac{\boldsymbol{w}}{\|\boldsymbol{w}\|} y))\right].$$

The following lower bound holds for the norm of the stochastic gradient, (see Appendix A) where $c_{\boldsymbol{w}}(\boldsymbol{x}) \in R$ is a random variable with constant distribution (independent of $\boldsymbol{w}$).

$$\|\nabla_{\boldsymbol{w}} L_{BN}(\boldsymbol{w}, \boldsymbol{x})\| \ge \frac{c_{\boldsymbol{w}}(\boldsymbol{x})}{\|\boldsymbol{w}\|}, \forall \boldsymbol{w} \in \mathbb{R}^m,$$

By Pythagorean Theorem, $\|\boldsymbol{w}_{t+1}\|^2 = \|\boldsymbol{w}_t\|^2 + \eta^2\|\nabla_{\boldsymbol{w}} L_{BN}(\boldsymbol{w}, \boldsymbol{x})\|^2 \ge \|\boldsymbol{w}_t\|^2 + \eta^2 \frac{c_{\boldsymbol{w}}^2}{\|\boldsymbol{w}_t\|^2}$. As a result, for any *fixed learning rate*, $\|\boldsymbol{w}_{t+1}\|^4 \ge \|\boldsymbol{w}_t\|^4 + 2\eta^2 c_{\boldsymbol{w}}^2$ grows linearly with high probability. Following the analysis of (Arora et al., 2019), this is like reducing the effective learning rate, and when $\|\boldsymbol{w}_t\|$ is large enough, the effective learning rate is small enough, and thus SGD can find the local minimum, which is the unique global minimum.

Case 3: Both BN and WD: Suppose weight decay factor is λ, learning rate is η, the width of the last layer is $m \geq 2$, Now the SGD updates have the form

$$\begin{aligned}\boldsymbol{w}_{t+1} &= \boldsymbol{w}_t - \eta \nabla \left(\ln(1+\exp(-\boldsymbol{x}_t^\top \tfrac{\boldsymbol{w}_t}{\|\boldsymbol{w}_t\|} y_t)) + \tfrac{\lambda}{2}\|\boldsymbol{w}_t\|^2 \right) \\ &= (1-\lambda\eta)\boldsymbol{w}_t - \eta \tfrac{y_t}{1+\exp(\boldsymbol{x}_t^\top \frac{\boldsymbol{w}_t}{\|\boldsymbol{w}_t\|} y_t)} \tfrac{\Pi^\perp_{\boldsymbol{w}_t} \boldsymbol{x}_t}{\|\boldsymbol{w}_t\|},\end{aligned}$$

where $\boldsymbol{x}_t \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, I_m), y_t = \text{sign}\,(x_{t,1})$, and $\Pi^\perp_{\boldsymbol{w}_t} = I - \frac{\boldsymbol{w}_t \boldsymbol{w}_t^\top}{\|\boldsymbol{w}_t\|^2}$.

Theorem 3.1. *[Nonconvergence] Starting from iteration any T_0, with probability arbitrarily close to $1-\delta$ over the randomness of samples, the training error will be larger than ε at least once for the following consecutive $\frac{1}{2(\eta\lambda - 2\varepsilon^2)} \ln \frac{32\|w_{T_0}\|^2 \varepsilon}{\eta\sqrt{m}} + 6\ln\frac{1}{\delta}$ iterations.*

Sketch. (See full proof in Appendix A.) The high level idea of this proof is that if the test error is low, the weight is restricted in a small cone around the global minimum, and thus the amount of the gradient update is bounded by the size of the cone. In this case, the growth of the norm of the weight by Pythagorean Theorem is not large enough to cancel the shrinkage brought by weight decay. As a result, the norm of the weight converges to 0 geometrically.

Again we need to use the lower bound for size of the gradient, that $\|\nabla_{\boldsymbol{w}} L_t\| = \Theta(\frac{\eta\sqrt{m}}{\|\boldsymbol{w}_t\|})$ holds with constant probability. Thus the size of the gradient will grow along with the shrinkage of $\|\boldsymbol{w}_t\|$ until they're comparable, forcing the weight to leave the cone in next iteration. □

It's interesting that the only property of the global minimum we use is that the if both $\boldsymbol{w}_t, \boldsymbol{w}_{t+1}$ are ϵ optimal, then the angle between $\boldsymbol{w}_t$ and $\boldsymbol{w}_{t+1}$ is at most 2ϵ. Thus we indeed have proved a stronger statement: *At least once in every $\frac{1}{2(\eta\lambda - 2\varepsilon^2)} \ln \frac{32\|w_{T_0}\|^2 \varepsilon}{\eta\sqrt{m}} + 6\ln\frac{1}{\delta}$ iterations, the angle between $\boldsymbol{w}_t$ and $\boldsymbol{w}_{t+1}$ will be larger than 2ϵ.* In other words, if the the amount of the update stablizes in terms of angle, then this angle must be larger than $\sqrt{2\eta\lambda}$ for this simple model.

4 Experiments

The translation to exponential lr schedule is exact except for correction term which happens when original schedule reduces lr a lot. The experiments explore the effect of this correction term. The Tapered-Exponential LR schedule contains two parts when entering a new phase I: an instant LR decay ($\frac{\eta_I}{\eta_{I-1}}$) and an adjustment of the growth factor. The first part is relative small compared to the huge exponential growing. Thus a natural question arises: *Can we simplify T-EXP LR schedule by dropping the part of instant LR decay?*

Also, previously we have only verified our equivalence theorem in Step Decay LR schedules. But it's not sure how would the Exponential LR schedule behave on more rapid time-varying LR schedules such as Cosine LR schedule.

Settings: The initial learning rate is 0.1 and the momentum is 0.9 in all settings. We fix all the scalar and bias of BN, because otherwise they together with the following conv layer grow exponentially, sometimes exceeding the range of float32 when trained with large growth rate for a long time. We fix the parameters in the last fully connected layer for scale invariance of the objective.

4.1 The benefit of instant LR decay

We tried the following LR schedule (we call it *T-EXP v2*). Interestingly, up to fluctuations of growth factor when entering a new phase, this schedule is equivalent to a constant LR schedule, but with the weight decay coefficient reduced correspondingly at the beginning of each phase.

$$\widetilde{\eta}_{t+1} = \widetilde{\eta}_t \times \left(\frac{1+\gamma-\lambda\eta_I + \sqrt{(1+\gamma-\lambda\eta_I)^2 - 4\gamma}}{2} \right)^{-2}, \quad \forall T_I \leq t < T_{I+1} \tag{7}$$

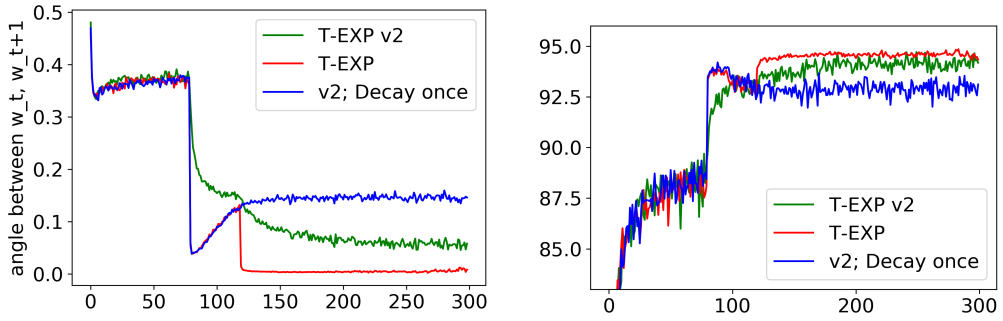

Figure 3: Instant LR decay is crucial when $\frac{\widetilde{\eta}_t}{\widetilde{\eta}_{t-1}} - 1$ is very small. When $\frac{\widetilde{\eta}_t}{\widetilde{\eta}_{t-1}} - 1$ is divided by 100, it would take T-EXP hundreds of epochs to reach its equilibrium. As a result, T-EXP achieves better test accuracy than T-EXP in shorter time. As a comparison, when $\frac{\widetilde{\eta}_t}{\widetilde{\eta}_{t-1}} - 1$ is divided by 10, it only takes 70 epochs to return to equilibrium. It's even faster without growth rate decay.

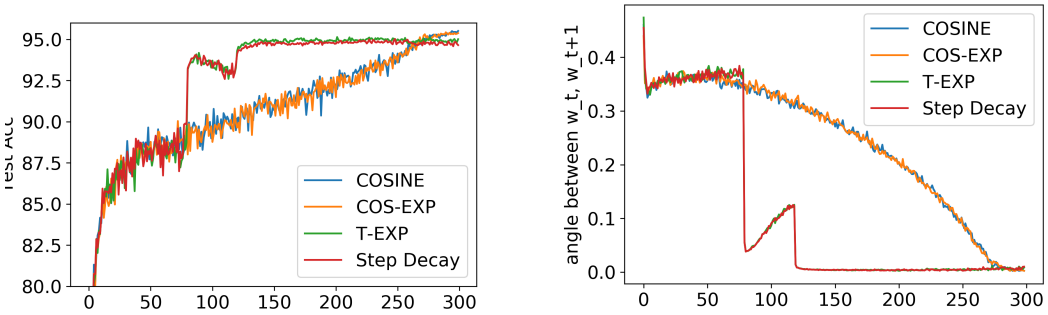

Figure 4: Both Cosine and Step Decay schedule behaves almost the same as their exponential counterpart, as predicted by our equivalence theorem. The (exponential) Cosine LR schedule achieves better test accuracy, with a entirely different trajectory.

4.2 Better Exponential LR Schedule with Cosine LR

We apply the T-EXP LR schedule (Theorem 2.4) on the Cosine LR schedule by (Loshchilov & Hutter, 2016), where the learning rate changes every epoch, and thus correction terms cannot be ignored. Learning rate at epoch $t \leq T$ is defined as:

$$\text{Cosine LR schedule:} \qquad \eta_t = \eta_0 \frac{1 + \cos(\frac{t}{T}\pi)}{2}. \tag{8}$$

Our experiments show this hybrid schedule with Cosine LR performs better on CIFAR10, but this finding needs to be verified on other datasets.

5 Conclusions

The paper shows rigorously how BN allows a host of very exotic learning rate schedules in deep learning, and verifies these effects in experiments. The lr increases exponentially in almost every iteration during training. The exponential increase derives from use of weight decay.

This also is a substantial improvement over earlier theoretical analyses of BN, since it accounts for weight decay and momentum, which are always combined in practice.

Our tantalising experiments with a hybrid of exponential and cosine rates suggest that more surprises may lie out there. Our theoretical analysis of interrelatedness of hyperparameters could also lead to faster hyperparameter search.

REFERENCES

Sanjeev Arora. Is optimization a sufficient language for understanding deep learning? URL http://www.offconvex.org/2019/06/03/trajectories/.

Sanjeev Arora, Nadav Cohen, and Elad Hazan. On the optimization of deep networks: Implicit acceleration by overparameterization. In *International Conference on Machine Learning*, pp. 244–253, 2018.

Sanjeev Arora, Zhiyuan Li, and Kaifeng Lyu. Theoretical analysis of auto rate-tuning by batch normalization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=rkxQ-nA9FX.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Nils Bjorck, Carla P Gomes, Bart Selman, and Kilian Q Weinberger. Understanding batch normalization. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 31*, pp. 7705–7716. Curran Associates, Inc., 2018.

Minhyung Cho and Jaehyung Lee. Riemannian approach to batch normalization. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 30*, pp. 5225–5235. Curran Associates, Inc., 2017.

Sanjoy Dasgupta and Anupam Gupta. An elementary proof of a theorem of johnson and lindenstrauss. *Random Structures & Algorithms*, 22(1):60–65, 2003.

Elad Hoffer, Ron Banner, Itay Golan, and Daniel Soudry. Norm matters: efficient and accurate normalization schemes in deep networks. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 31*, pp. 2164–2174. Curran Associates, Inc., 2018.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pp. 448–456, 2015.

Jonas Kohler, Hadi Daneshmand, Aurelien Lucchi, Ming Zhou, Klaus Neymeyr, and Thomas Hofmann. Exponential convergence rates for batch normalization: The power of length-direction decoupling in non-convex optimization. *arXiv preprint arXiv:1805.10694*, 2018.

Ilya Loshchilov and Frank Hutter. SGDR: Stochastic Gradient Descent with Warm Restarts. *arXiv e-prints*, art. arXiv:1608.03983, Aug 2016.

David Page. How to train your resnet 6: Weight decay? URL https://myrtle.ai/how-to-train-your-resnet-6-weight-decay/.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

Shibani Santurkar, Dimitris Tsipras, Andrew Ilyas, and Aleksander Madry. How does batch normalization help optimization? In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 31*, pp. 2488–2498. Curran Associates, Inc., 2018.

Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28*, ICML'13, pp. III–1139–III–1147. JMLR.org, 2013. URL http://dl.acm.org/citation.cfm?id=3042817.3043064.

Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016.

Xiaoxia Wu, Rachel Ward, and Léon Bottou. WNGrad: Learn the Learning Rate in Gradient Descent. *arXiv preprint arXiv:1803.02865*, 2018.

Yuxin Wu and Kaiming He. Group normalization. In *The European Conference on Computer Vision (ECCV)*, September 2018.

Yang You, Igor Gitman, and Boris Ginsburg. Large Batch Training of Convolutional Networks. *arXiv e-prints*, art. arXiv:1708.03888, Aug 2017.

Guodong Zhang, Chaoqi Wang, Bowen Xu, and Roger Grosse. Three mechanisms of weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=B1lz-3Rct7.

A Omitted Proofs

A.1 Omitted Proof in Section 2

Some Facts about Equation 4: Suppose $z_1, z_2 (z_1 \geq z_2)$ are the two real roots of the the following equation, we have

$$x^2 - (1 + \gamma - \lambda\eta)x + \gamma = 0$$

1. $0 \leq z_2 \leq z_1 \leq 1$;
2. Let $t = \frac{\lambda\eta}{1-\gamma}$, we have $1 - z_1 \leq \frac{t}{1+t}$;
3. if we view z_1 as a function of

Proof. Let $f(x) = x^2 - (1 + \gamma - \lambda\eta)x + \gamma$, we have $f(1) = f(\gamma) = \lambda\eta \geq 0$. Note the minimum of f is taken at $x = \frac{1+\gamma-\lambda\eta}{2} \in [0, 1]$, the both roots of $f(x) = 0$ must lie between 0 and 1, if exists.

$$\begin{aligned}
1 - z_1 &= \frac{1 - \gamma + \lambda\eta + \sqrt{(1-\gamma)^2 - 2(1+\gamma)\lambda\eta + \lambda^2\eta^2}}{2} \\
&= (1-\gamma)\frac{1 + t - \sqrt{1 - \frac{1+\gamma}{1-\gamma}t + t^2}}{2} \\
&= (1-\gamma)\frac{2t + 2\frac{1+\gamma}{1-\gamma}t}{2(1 + t + \sqrt{1 - \frac{1+\gamma}{1-\gamma}t + t^2})} \\
&\leq (1-\gamma)\frac{\frac{4}{1-\gamma}t}{4(1+t)} \\
&= \frac{t}{(1+t)}
\end{aligned}$$

□

Proof of Theorem 2.5. We will prove by induction. By assumption $S(t) : P_t\boldsymbol{\theta}_t = \widetilde{\boldsymbol{\theta}}_t$ for $t = -1, 0$. Now we will show that $S(t) \Longrightarrow S(t+1), \forall t \geq 0$.

$$
\begin{aligned}
&\frac{\boldsymbol{\theta}_t-\boldsymbol{\theta}_{t-1}}{\eta_{t-1}}=\gamma\frac{\boldsymbol{\theta}_{t-1}-\boldsymbol{\theta}_{t-2}}{\eta_{t-2}}-\nabla_{\boldsymbol{\theta}}\left((L(\boldsymbol{\theta}_{t-1})+\frac{\lambda_{t-1}}{2}\|\boldsymbol{\theta}_{t-1}\|_2^2\right)\\
\xRightarrow{\text{Take gradient}}\ &\frac{\boldsymbol{\theta}_t-\boldsymbol{\theta}_{t-1}}{\eta_{t-1}}=\gamma\frac{\boldsymbol{\theta}_{t-1}-\boldsymbol{\theta}_{t-2}}{\eta_{t-2}}-\nabla_{\boldsymbol{\theta}}L(\boldsymbol{\theta}_{t-1})+\lambda_{t-1}\boldsymbol{\theta}_{t-1}\\
\xRightarrow{\text{Scale Invariance}}\ &\frac{\boldsymbol{\theta}_t-\boldsymbol{\theta}_{t-1}}{\eta_{t-1}}=\gamma\frac{\boldsymbol{\theta}_{t-1}-\boldsymbol{\theta}_{t-2}}{\eta_{t-2}}-P_{t-1}\nabla_{\boldsymbol{\theta}}L(\widetilde{\boldsymbol{\theta}}_{t-1})+\lambda_{t-1}\boldsymbol{\theta}_{t-1}\\
\xRightarrow{\text{Rescaling}}\ &\frac{P_t(\boldsymbol{\theta}_t-\boldsymbol{\theta}_{t-1})}{P_tP_{t-1}\eta_{t-1}}=\gamma\frac{P_{t-2}(\boldsymbol{\theta}_{t-1}-\boldsymbol{\theta}_{t-2})}{P_{t-1}P_{t-2}\eta_{t-2}}-\nabla_{\boldsymbol{\theta}}L(\widetilde{\boldsymbol{\theta}}_{t-1})-\lambda_{t-1}\frac{\boldsymbol{\theta}_{t-1}}{P_{t-1}}\\
\xRightarrow{\text{Simplfying}}\ &\frac{P_t\boldsymbol{\theta}_t-\alpha_t^{-1}\widetilde{\boldsymbol{\theta}}_{t-1}}{\widetilde{\eta}_{t-1}}=\gamma\frac{\alpha_{t-1}\widetilde{\boldsymbol{\theta}}_{t-1}-\widetilde{\boldsymbol{\theta}}_{t-2}}{\widetilde{\eta}_{t-2}}-\nabla_{\boldsymbol{\theta}}L(\widetilde{\boldsymbol{\theta}}_{t-1})-\eta_{t-1}\lambda_{t-1}\frac{P_t\boldsymbol{\theta}_{t-1}}{\eta_{t-1}P_{t-1}P_t}\\
\xRightarrow{\text{Simplfying}}\ &\frac{P_t\boldsymbol{\theta}_t-\alpha_t^{-1}\widetilde{\boldsymbol{\theta}}_{t-1}}{\widetilde{\eta}_{t-1}}=\gamma\frac{\alpha_{t-1}\widetilde{\boldsymbol{\theta}}_{t-1}-\widetilde{\boldsymbol{\theta}}_{t-2}}{\widetilde{\eta}_{t-2}}-\nabla_{\boldsymbol{\theta}}L(\widetilde{\boldsymbol{\theta}}_{t-1})-\eta_{t-1}\lambda_{t-1}\frac{\alpha_t^{-1}\widetilde{\boldsymbol{\theta}}_{t-1}}{\widetilde{\eta}_{t-1}}\\
\xRightarrow{\text{Simplfying}}\ &\frac{P_t\boldsymbol{\theta}_t-\alpha_t^{-1}(1-\eta_{t-1}\lambda_{t-1})\widetilde{\boldsymbol{\theta}}_{t-1}}{\widetilde{\eta}_{t-1}}=\gamma\frac{\alpha_{t-1}\widetilde{\boldsymbol{\theta}}_{t-1}-\widetilde{\boldsymbol{\theta}}_{t-2}}{\widetilde{\eta}_{t-2}}-\nabla_{\boldsymbol{\theta}}L(\widetilde{\boldsymbol{\theta}}_{t-1})
\end{aligned}
$$

To conclude that $P_t\boldsymbol{\theta}_t=\widetilde{\boldsymbol{\theta}}_t$, it suffices to show that the coefficients before $\widetilde{\boldsymbol{\theta}}_{t-1}$ is the same to that in (2). In other words, we need to show

$$
\frac{-1+\alpha_t^{-1}(1-\eta_{t-1}\lambda_{t-1})}{\widetilde{\eta}_{t-1}}=\frac{\gamma(1-\alpha_{t-1})}{\widetilde{\eta}_{t-2}},
$$

which is equivalent to the definition of α_t, (equation 6). □

Proof of Theorem 2.4. Assuming z_1^I and $z_2^I(z_1^I>z_2^I)$ are the roots of Equation4 with $\eta=\eta_I$ we can rewrite the recursion in Theorem 2.5 as the following:

$$
\alpha_t=-\eta_I\lambda+1+\gamma(1-\alpha_{t-1}^{-1})=-(z_1^I+z_2^I)+z_1^Iz_2^I\alpha_{t-1}^{-1}. \tag{9}
$$

In other words, we have

$$
\alpha_t-z_1^I=\frac{z_2^I}{\alpha_{t-1}}(\alpha_{t-1}-z_1^I),t\geq 1, \tag{10}
$$

which means that if we enter each phase with $\alpha_{t_I}\geq z_1^I$, we have $\alpha_t\geq z_1^I$ for the whole phase. Thus we conclude that α_t will be larger than z_1^1 for phase 1. It's not hard to show that since $\eta_I>\eta_J,\forall I<J$, $z_1^I>z_1^J$ and $z_2^I<z_2^J$. Now we will prove α_t is always larger than z_1^1. If $\alpha_t\geq z_1^1$, then we have $\alpha_t\geq z_2^1\geq z_2^I$. In each phase, due to the above inequality, $|\alpha_t-z_1^I|$ is decreasing, which guarantees that if $\alpha_t>z_1^1$ when entering the phase, then $\alpha_t>z_1^1$ for every iteration in this phase.

Note that the when entering a new phase, since $\eta_I<\eta_{I-1}$, α_{t_I} will be larger than it would be without changing phase, which is already larger than z_1^1. Thus for the whole process, α_t will be larger than z_1^1.

Thus for $\alpha_{t-1}\in[z_1^1,\infty)$, $\alpha_t-z_1^I=\frac{z_2^I}{\alpha_{t-1}}(\alpha_{t-1}-z_1^I)\leq\frac{z_2^1}{z_1^1}(\alpha_{t-1}-z_1^I)=\frac{\gamma}{z_1^1}(\alpha_{t-1}-z_1^I)$, which means α_t geometrically converges to its stable fixed point z_1. With small $\lambda\eta$, one could approximate z_1^1 by 1 and thus $\frac{\gamma}{z_1^1}$ by γ.

Note that $\frac{\hat{\eta}_{t-1}}{\hat{\eta}_t}=\alpha_t\alpha_{t+1}$ and $\frac{\widetilde{\eta}_{t-1}}{\widetilde{\eta}_t}=(z_I^1)^2$. Since that $0.5\leq z_I^1<\alpha_t\leq 1$, $0.5\leq z_I^1\leq 1$, we have $|\alpha_{T_I}-z_I^1|\leq 0.5$, and thus $|\alpha_t-z_I^1|\leq\frac{1}{2}(\frac{\gamma}{(z_1^1)^2})^{t-T_I},\forall T_I\leq t\leq T_{I+1}-1$. Thus we have

$$
\left|\frac{\hat{\eta}_{t-1}}{\hat{\eta}_t}-\frac{\widetilde{\eta}_{t-1}}{\widetilde{\eta}_t}\right|=\alpha_t\alpha_{t+1}-(z_I^1)^2\leq 3|\alpha_t-z_I^1|\leq\frac{3}{2}\left(\frac{\gamma}{(z_1^1)^2}\right)^{t-T_I}.
$$

□

A.2 PROOF FOR THEOREM B.1

Proof. Let's use R_t, D_t, C_t to denote $\|\boldsymbol{\theta}_t\|^2, \|\boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}_t\|^2, \boldsymbol{\theta}_t^\top(\boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}_t)$ respectively.
The only property we will use about loss is $\nabla_{\boldsymbol{\theta}} L_t^\top \boldsymbol{\theta}_t = 0$.
Expanding the square of $\|\boldsymbol{\theta}_{t+1}\|^2 = \|(\boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}_t) + \boldsymbol{\theta}_t\|^2$, we have

$$S(t): \quad R_{t+1} - R_t = D_t + 2C_t, \forall t.$$

We also have

$$\frac{C_t}{\eta_t} = \boldsymbol{\theta}_t^\top \frac{\boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}_t}{\eta_t} = \boldsymbol{\theta}_t^\top (\gamma \frac{\boldsymbol{\theta}_t - \boldsymbol{\theta}_{t-1}}{\eta_{t-1}} - \lambda_t \boldsymbol{\theta}_t) = \frac{\gamma}{\eta_{t-1}}(D_t + C_{t-1}) - \lambda_t R_t,$$

namely,

$$P(t): \quad \frac{C_t}{\eta_t} - \frac{\gamma C_{t-1}}{\eta_{t-1}} = \frac{\gamma}{\eta_{t-1}} C_{t-1} - \lambda_t R_t.$$

Simplify $\frac{S(t)}{\eta_t} - \frac{\gamma S(t-1)}{\eta_{t-1}} + P(t)$, we have

$$\frac{R_{t+1} - R_t}{\eta_t} - \gamma \frac{R_t - R_{t-1}}{\eta_{t-1}} = \frac{D_t}{\eta_t} + \gamma \frac{D_{t-1}}{\eta_{t-1}} - 2\lambda_t R_t. \tag{11}$$

When $\lambda_t = 0$, we have

$$\frac{R_{t+1} - R_t}{\eta_t} = \gamma^{t+1} \frac{R_0 - R_{-1}}{\eta_{-1}} + \sum_{i=0}^{t} \gamma^{t-i} (\frac{D_i}{\eta_i} + \gamma \frac{D_{i-1}}{\eta_{i-1}}) \geq \gamma^{t+1} \frac{R_0 - R_{-1}}{\eta_{-1}}.$$

Further if $\eta_t = \eta$ is a constant, we have

$$R_{t+1} = \sum_{i=0}^{t} \frac{1 - \gamma^{t-i+1}}{1 - \gamma}(D_i + \gamma D_{i-1}) - \gamma \frac{1 - \gamma^{t+1}}{1 - \gamma}(R_0 - R_{-1}),$$

which covers the result without momentum in (Arora et al., 2019) as a special case:

$$R_{t+1} = \sum_{i=0}^{t} D_i.$$

□

Proof of Theorem B.2. Take average of Equation 11 over t, when the limits $R_\infty = \lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \|\boldsymbol{w}_t\|^2$, $D_\infty = \lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \|\boldsymbol{w}_{t+1} - \boldsymbol{w}_t\|^2$ exists, we have

$$\frac{1 + \gamma}{\eta} D_\infty = 2\lambda R_\infty,$$

which is

$$\sqrt{\frac{D_\infty}{R_\infty}} = \sqrt{\frac{2\eta\lambda}{1 + \gamma}}.$$

□

A.3 OMITTED PROOFS IN SECTION 3

We will need the following lemma when lower bounding the norm of the stochastic gradient.

Lemma A.1 (Concentration of Chi-Square)**.** *Suppose* $X_1, \ldots, X_k \overset{i.i.d.}{\sim} \mathcal{N}(0,1)$*, then*

$$Pr\left[\sum_{i=1}^{k} X_i^2 < k\beta\right] \le \left(\beta e^{1-\beta}\right)^{\frac{k}{2}}. \tag{12}$$

Proof. This Chernoff-bound based proof is a special case of Dasgupta & Gupta (2003).

$$\begin{aligned}\Pr\left[\sum_{i=1}^{k} X_i^2 < k\beta\right] \le \left(\beta e^{1-\beta}\right)^{\frac{k}{2}} =& \Pr\left[\exp\left(kt\beta - t\sum_{i=1}^{k} X_i^2\right) \ge 1\right] \\ \le& \mathbb{E}\left[\exp\left(kt\beta - t\sum_{i=1}^{k} X_i^2\right)\right] \text{ (Markov Inequality)} \\ =& e^{kt\beta}(1+2t)^{-\frac{k}{2}}.\end{aligned} \tag{13}$$

The last equality uses the fact that $\mathbb{E}\left[tX_i^2\right] = \frac{1}{\sqrt{1-2t}}$ for $t < \frac{1}{2}$. The proof is completed by taking $t = \frac{1-\beta}{2\beta}$. □

Proof. **Step 1:** We will use $\hat{\boldsymbol{w}}$ to denote $\frac{\boldsymbol{w}}{\|\boldsymbol{w}\|}$ and $\angle \boldsymbol{u}\boldsymbol{w}$ to $\arccos(\hat{\boldsymbol{u}}^\top \hat{\boldsymbol{w}})$. Note that training error $\le \varepsilon$ is equivalent to $\angle \boldsymbol{e}_1 \boldsymbol{w}_t < \varepsilon$. Let $T_1 = \frac{1}{2(\eta\lambda - 2\varepsilon^2)} \ln \frac{32\|w_{T_0}\|^2 \varepsilon}{\eta\sqrt{m}}$, and $T_2 = 6\ln\frac{1}{\delta}$. Thus if we assume the training error is smaller than ε from iteration T_0 to $T_0 + T_1 + T_2$, then by spherical triangle inequality, $\angle \boldsymbol{w}_t \boldsymbol{w}_{t'} \le \angle \boldsymbol{e}_1 \boldsymbol{w}_{t'} + \angle \boldsymbol{e}_1 \boldsymbol{w}_t = 2\varepsilon$, for $T_0 \le t, t' \le T_0 + T_1 + T_2$.

Now let's define $\boldsymbol{w}_t' = (1-\eta\lambda)\boldsymbol{w}_t$ and for any vector $\boldsymbol{w}$, and we have the following two relationships:

1. $\|\boldsymbol{w}_t'\| = (1-\eta\lambda)\|\boldsymbol{w}\|$.
2. $\|\boldsymbol{w}_{t+1}\| \le \frac{\|\boldsymbol{w}_t\|}{\cos 2\varepsilon}$.

The second property is because by Lemma 2.8, $(\boldsymbol{w}_{t+1} - \boldsymbol{w}_t') \perp \boldsymbol{w}_t'$ and by assumption of small error, $\angle \boldsymbol{w}_{t+1}\boldsymbol{w}_t' \le 2\varepsilon$.

Therefore

$$\frac{\|\boldsymbol{w}_{T_1+T_0}\|^2}{\|\boldsymbol{w}_{T_0}\|^2} \le \left(\frac{1-\eta\lambda}{\cos 2\varepsilon}\right)^{2T_1} \le \left(\frac{1-\eta\lambda}{1-2\varepsilon^2}\right)^{2T_1} \le \left(1-(\eta\lambda - 2\varepsilon^2)\right)^{2T_1} \le e^{-2T_1(\eta\lambda - 2\varepsilon^2)} = \frac{\eta\sqrt{m}}{32\|w_{T_0}\|^2\varepsilon}. \tag{14}$$

In other word, $\|\boldsymbol{w}_{T_0+T_1}\|^2 \le \frac{\eta\sqrt{m}}{32\varepsilon}$. Since $\|\boldsymbol{w}_{T_0+t}\|$ is monotone decreasing, $\|\boldsymbol{w}_{T_0+t}\|^2 \le \frac{\eta\sqrt{m}}{32\varepsilon}$ holds for any $t = T_1, \ldots, T_1 + T_2$.

Step 2: We show that the norm of the stochastic gradient is lower bounded with constant probability.

Note that $\boldsymbol{x}_t^\top \frac{\boldsymbol{w}_t}{\|\boldsymbol{w}_t\|}$ and $\Pi_{\boldsymbol{w}_t}^{\perp}\boldsymbol{x}_t$ are independent gaussian r.v., where $\boldsymbol{x}_t^\top \frac{\boldsymbol{w}_t}{\|\boldsymbol{w}_t\|} \sim \mathcal{N}(0,1)$ and $\|\Pi_{\boldsymbol{w}_t}^{\perp}\boldsymbol{x}_t\|^2 \sim \chi^2(m-1)$.

$$\Pr\left[|\boldsymbol{x}_t^\top \frac{\boldsymbol{w}_t}{\|\boldsymbol{w}_t\|}| < 1\right] \ge \frac{1}{2}, \tag{15}$$

and by Lemma A.1,

$$\Pr\left[\|\Pi^{\perp}_{\boldsymbol{w}_t}\boldsymbol{x}_t\|^2 \geq \frac{m}{16}\right] \geq \Pr\left[\|\Pi^{\perp}_{\boldsymbol{w}_t}\boldsymbol{x}_t\|^2 \geq \frac{m-1}{8}\right] \geq 1-(\frac{1}{8e^{\frac{7}{8}}})^{\frac{m-1}{2}} \geq 1-(\frac{1}{8e^{\frac{7}{8}}})^{\frac{1}{2}} \geq \frac{1}{3}. \quad (16)$$

Thus w.p. at least $\frac{1}{6}$, eventsequation 15 and equation 16 happens together, which implies

$$\|\nabla \ln(1+\exp(-\boldsymbol{x}_t^\top \frac{\boldsymbol{w}_t}{\|\boldsymbol{w}_t\|}y_t))\| = \|\frac{y_t}{1+\exp(\boldsymbol{x}_t^\top \frac{\boldsymbol{w}_t}{\|\boldsymbol{w}_t\|}y_t)}\frac{\Pi^{\perp}_{\boldsymbol{w}_t}\boldsymbol{x}_t}{\|\boldsymbol{w}_t\|}\| \geq \frac{\eta}{1+e}\frac{\sqrt{m}}{4\|\boldsymbol{w}_t\|} \geq \frac{\eta\sqrt{m}}{16\|\boldsymbol{w}_t\|} \quad (17)$$

Step 3. To stay in the cone $\{\boldsymbol{w}|\angle \boldsymbol{w}\boldsymbol{e}_1 \leq \varepsilon\}$, the SGD update $\|\boldsymbol{w}_{t+1}-\boldsymbol{w}'_t\| = \|\nabla \ln(1+\exp(-\boldsymbol{x}_t^\top \frac{\boldsymbol{w}_t}{\|\boldsymbol{w}_t\|}y_t))\|$ has to be smaller than $\|\boldsymbol{w}_t\|\sin 2\varepsilon$ for any $t = T_0+T_1,\ldots,T_0+T_1+T_2$. However, step 1 and 2 together show that $\|\nabla \ln(1+\exp(-\boldsymbol{x}_t^\top \frac{\boldsymbol{w}_t}{\|\boldsymbol{w}_t\|}y_t))\| \geq 2\|\boldsymbol{w}_t\|\varepsilon$ w.p. $\frac{1}{6}$ per iteration. Thus the probability that $\boldsymbol{w}_t$ always stays in the cone for every $t = T_0+T_1,\ldots,T_0+T_1+T_2$ is less than $\left(\frac{5}{6}\right)^{T_2} \leq \delta$. □

B Other Results

Now we rigorously analyze norm growth in this algorithm. This greatly extends previous analyses of effect of normalization schemes (Wu et al., 2018; Arora et al., 2018) for vanilla SGD.

Theorem B.1. *Under the update rule 1.2 with $\lambda_t = 0$, the norm of scale invariant parameter $\boldsymbol{\theta}_t$ satisfies the following property:*

- *Almost Monotone Increasing:* $\|\boldsymbol{\theta}_{t+1}\|^2 - \|\boldsymbol{\theta}_t\|^2 \geq -\gamma^{t+1}\frac{\eta_t}{\eta_0}(\|\boldsymbol{\theta}_0\|^2 - \|\boldsymbol{\theta}_{-1}\|^2)$.
- *Assuming $\eta_t = \eta$ is a constant, then*

$$\|\boldsymbol{\theta}_{t+1}\|^2 = \sum_{i=0}^{t}\frac{1-\gamma^{t-i+1}}{1-\gamma}\left(\|\boldsymbol{\theta}_i - \boldsymbol{\theta}_{i+1}\|^2 + \gamma\|\boldsymbol{\theta}_{i-1}-\boldsymbol{\theta}_i\|^2\right) - \gamma\frac{1-\gamma^{t+1}}{1-\gamma}(\|\boldsymbol{\theta}_0\|^2 - \|\boldsymbol{\theta}_{-1}\|^2)$$

For general deep nets, we have the following result, suggesting that the mean square of the update are constant compared to the mean square of the norm. The constant is mainly determined by $\eta\lambda$, explaining why the usage of weight decay prevents the parameters to converge in direction. [2]

Theorem B.2. *For SGD with constant LR η, weight decay λ and momentum γ, when when the limits $R_\infty = \lim_{T\to\infty}\frac{1}{T}\sum_{t=0}^{T-1}\|\boldsymbol{w}_t\|^2$, $D_\infty = \lim_{T\to\infty}\frac{1}{T}\sum_{t=0}^{T-1}\|\boldsymbol{w}_{t+1}-\boldsymbol{w}_t\|^2$ exists, we have*

$$\sqrt{\frac{D_\infty}{R_\infty}} = \sqrt{\frac{2\eta\lambda}{1+\gamma}}.$$

C Additional Experimental Figures

[2] (Page) had a similar argument for this phenomenon by connecting this to the LARS(You et al., 2017), though it's not rigorous in the way it deals with momentum and equilibrium of norm.

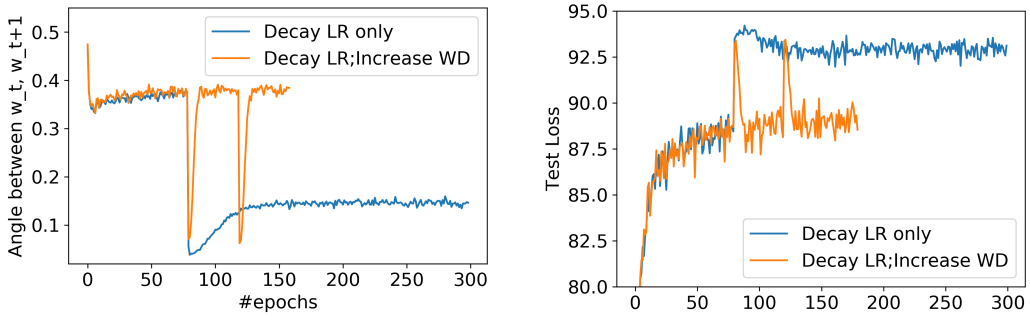

Figure 5: PreResNet32 trained with SGD with 0.9 momentum, 0.0005 WD and 0.1 LR. Here we consider 2 schedules: the orange one divides LR by 10 and multiplies WD by 10 at epoch 80 and 120. The blue divides LR by 10 at epoch 80. Both setting suggest that the instant decay of LR could allow network to stabilize shortly, but in the long run, the average angle between weights in consecutive epochs converges to a value independent of LR, only depending on $LR \times WD$, as predicted by Theorem B.2.