A. Correlation between Lipschitz and growing risk

Our adaptive growing strategy relies on the Lipschitz dynamics to guide the grow, since it can be shown that the growing risks can be theoretically controlled by the Lipschitz constant. We have proposed two types of risks, the performance degeneration right after grow, *i.e.*, $\left|\varepsilon(\mathcal{F}^{(2N\to N)}) - \varepsilon(\mathcal{F}^{(N)})\right|$, and the performance gap between networks trained with varied depth and with static depth, *i.e.*, $\left|\varepsilon(\mathcal{F}^{(2N\to N)}) - \varepsilon(\mathcal{F}^{(2N)})\right|$. Hereby we shall refer to the former as temporal risk and the latter as final risk. Based on Theorem 3, assuming the Lipschitz constant is relatively small, the temporal error is $e^{(N,2N)} \lesssim \exp\left[\mathcal{L}(f^{(N)}(t_e - t_s))\right] - 1 \approx \mathcal{L}(f^{(N)})(t_e - t_s))$, which reveals a linear correlation approximately. And since the aforementioned two risks can both be bounded by the temporal error, the linear correlation should be observed as well.

Extensive experiments show empirically that the growing risks are high correlated to the Lipschitz constant. To reduce the influence of varied learning rate and better comply with the theory, we adopt constant learning rate (0.1) and pre-activation architecture to train a ResNet-38 on CIFAR-10 with static training and adaptive training starting from ResNet-20, where the one-time grow is scheduled at different epochs. Figure 6 shows the training error rate with respect to the epoch for different training schedules. The Lipschitz constant of ResNet-20 during training is presented jointly. One can find that as the Lipschitz constant increases during training, the temporal risk increases accordingly. To further verify it, we measure the correlation magnitude by r^2, *i.e.*, the coefficient of determination, between the scaled Lipschitz constant we used in LipGrow, and the temporal risk captured in each experiment. As shown in Figure 6, the correlation is statistically significant.

To fairly measure the final risk, we continue the training of ResNet-38 for additional 50 epochs after grow, and compare the yielded performance with the performance of ResNet-38 trained without grow at corresponding epochs. As shown in Figure 6, the final risk is highly correlated with the Lipschitz constant as well. The statistical significance is not as high as the temporal risk because now the term $C^{(N,2N)}$ in Theorem 3 is not negligible, whereas for temporal risk this term is rigorously zero, as the block functions are copied identically.

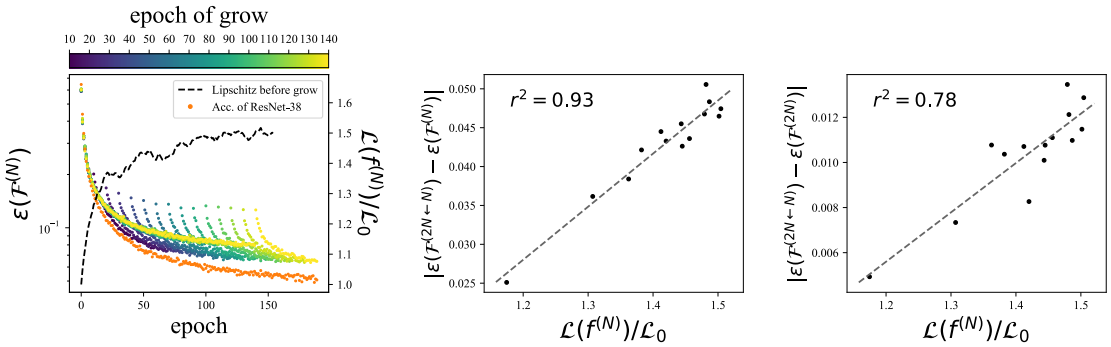


Figure 6: The Correlations between Growing Risks and Lipschitz Constant. ε here refers to the training error rate. $\left|\varepsilon(\mathcal{F}^{(2N\to N)}) - \varepsilon(\mathcal{F}^{(N)})\right|$ and $\left|\varepsilon(\mathcal{F}^{(2N\to N)}) - \varepsilon(\mathcal{F}^{(2N)})\right|$ refer to the temporal risk and final risk, respectively. Details can be found in the text.

B. Deficiency of Network Morphism

In Section 3.2, we present a favored initialization scheme based on the performance measure proposed in Theorem 2. Under the same framework, we can also analyze the deficiencies of other initialization methods. For example, network morphism essentially ensures the new blocks are inserted such that the input-output function is preserved. For residual network particularly, a simple solution will be initializing all new blocks to be 0.

Nonetheless, this operation first implicitly involves cloning since part of the residual functions are inherited. Thus besides the similar functional error due to the misalignment between time points as introduced in3.2, newly inserted blocks may introduce additional error since they are far away from the optimal ODE and thus demand extra optimization. Moreover, the regularity condition is broken after the grow since the adjacent functional difference with respect to the new blocks can not be bounded, which implies that the performance retained by the morphism may only be impermanent, and may pose difficulty to the subsequent optimization.

In previous work, network morphism is reported to be less effective in a continual growing setting (Wen et al., 2019). Note that for shallow grow, such initialization (including random) may be favoured, as C_t^* may not be smaller than

$\sup_t \|f^*(t) - f_r\|_\infty$, where f_r is some manually retrieved residual function.

C. Continuous Transformation

The residual connection is a discrete transformation, *i.e.*

$$\boldsymbol{z}_{n+1}^{(N)} = \boldsymbol{z}_n^{(N)} + h^{(N)} f_n^{(N)}(\boldsymbol{z}_n^{(N)}), \tag{7}$$

where the blocks functions are only defined at specific time points $t_n^{(N)} = t_s + (t_e - t_s)(n/N)$. One can intentionally extend the definition to be continuous on t using Dirac measure (Thorpe & van Gennip, 2018), such as

$$f^{(N)}(t, \boldsymbol{z}) = h^{(N)} \sum_n f_n^{(N)}(\boldsymbol{z}) \mathbb{1}_{t_n^{(N)}}(t), n \in \{0, ..., N-1\}$$

where $\mathbb{1}$ is the indicator function defined as

$$\mathbb{1}_{t_n^{(N)}}(t) := \begin{cases} 1 & \text{if } t = t_n^{(N)}, \\ 0 & \text{o.w.} \end{cases}$$

The associated ODE, *i.e.*,

$$\frac{d\boldsymbol{z}^{(N)}(t)}{dt} = f^{(N)}(t, \boldsymbol{z}^{(N)}(t)),$$

now gives a solution of feature maps defined continuously on t

$$\boldsymbol{z}^{(N)}(t) = \boldsymbol{z}^{(N)}(t_s) + h^{(N)} \sum_n f_n^{(N)}(\boldsymbol{z}^{(N)}(t_n^{(N)})) \mathcal{H}_{t_n^{(N)}}(t), \tag{8}$$

where $\mathcal{H}$ is the step function

$$\mathcal{H}_{t_n^{(N)}}(t) := \begin{cases} 1 & \text{if } t > t_n^{(N)}, \\ 0 & \text{o.w.} \end{cases}$$

Alternatively, (8) can be viewed as an interpolation of the discrete feature maps.

D. Additional proofs

Proof. (**Theorem 1**) This theorem is a rephrased proposition derived in Thorpe & van Gennip (2018).

Proof. (**Theorem 2**)

A network of depth N follows a residual connection as

$$\boldsymbol{z}_{n+1}^{(N)} = \boldsymbol{z}_n^{(N)} + h^{(N)} f_n^{(N)}(\boldsymbol{z}_n^{(N)}).$$

Compared to the Taylor expansion of the continuous solution at the corresponding time points

$$\boldsymbol{z}^*\left(t_{n+1}^{(N)}\right) = \boldsymbol{z}^*\left(t_n^{(N)}\right) + h^{(N)} f^*\left(t_n^{(N)}, \boldsymbol{z}^*\left(t_n^{(N)}\right)\right) + \frac{{h^{(N)}}^2}{2} \boldsymbol{z}^{*\prime\prime}(t_\eta),$$

where $t_n^{(N)} = t_s + (t_e - t_s)(n/N)$ and $t_n^{(N)} < t_\eta < t_{n+1}^{(N)}$, we will have

$$\begin{aligned} &\boldsymbol{z}^*\left(t_{n+1}^{(N)}\right) - \boldsymbol{z}_{n+1}^{(N)} \\ =&\left[\boldsymbol{z}^*\left(t_n^{(N)}\right) - \boldsymbol{z}_n^{(N)}\right] + h^{(N)}\left[f^*\left(t_n^{(N)}, \boldsymbol{z}^*\left(t_n^{(N)}\right)\right) - f_n^{(N)}(\boldsymbol{z}_n^{(N)}))\right] + \frac{{h^{(N)}}^2}{2} \boldsymbol{z}^{*\prime\prime}(t_\eta) \\ =&\left[\boldsymbol{z}^*\left(t_n^{(N)}\right) - \boldsymbol{z}_n^{(N)}\right] \\ &+ h^{(N)}\left[f^*\left(t_n^{(N)}, \boldsymbol{z}^*\left(t_n^{(N)}\right)\right) - f^*\left(t_n^{(N)}, \boldsymbol{z}_n^{(N)}\right) + f^*\left(t_n^{(N)}, \boldsymbol{z}_n^{(N)}\right) - f_n^{(N)}(\boldsymbol{z}_n^{(N)}))\right] + \frac{{h^{(N)}}^2}{2} \boldsymbol{z}^{*\prime\prime}(t_\eta). \end{aligned} \tag{9}$$

The deviation of the discrete feature map $\boldsymbol{z}^*\left(t_n^{(N)}\right) - \boldsymbol{z}_n^{(N)}$ can also be viewed as the local truncation error.

We assume the l^∞ distance between $f^{(N)}$ and f^* is bounded, *i.e.*,

$$\sup_{n,\boldsymbol{z}} \left| f^*(t_n^{(N)}, \boldsymbol{z}) - f_n^{(N)}(\boldsymbol{z}) \right| = C^{(N,*)}, \tag{10}$$

and f^* is l-lipshitz, the curvature of continuous dynamics is bounded, *i.e.*,

$$\begin{aligned} \mathcal{L}(f^*) &= \sup_{t,\boldsymbol{z}_1 \neq \boldsymbol{z}_2} \frac{|f^*(t,\boldsymbol{z}_1) - f^*(t,\boldsymbol{z}_2)|}{|\boldsymbol{z}_1 - \boldsymbol{z}_2|}, \\ M(\boldsymbol{z}^*) &= \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}} \sup_t |\boldsymbol{z}^{*\prime\prime}(t)|. \end{aligned} \tag{11}$$

Given the above three bounds, we now have

$$\left| \boldsymbol{z}^*\left(t_{n+1}^{(N)}\right) - \boldsymbol{z}_{n+1}^{(N)} \right| \leq \left[1 + h^{(N)}\mathcal{L}(f^*)\right] \left| \boldsymbol{z}^*\left(t_n^{(N)}\right) - \boldsymbol{z}_n^{(N)} \right| + h^{(N)} \left[\frac{M(\boldsymbol{z}^*)}{2} h^{(N)} + C^{(N,*)} \right], \tag{12}$$

Recursively apply this contraction relation yields

$$\begin{aligned} e^{(N)} &:= \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}} \left| \boldsymbol{z}^*(t_e) - \boldsymbol{z}^{(N)}(t_e) \right| \\ &\leq \left[1 + h^{(N)}\mathcal{L}(f^*)\right]^N \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}} \left| \boldsymbol{z}^*(t_s) - \boldsymbol{z}_0^{(N)} \right| + h^{(N)} \left[\frac{M(\boldsymbol{z}^*)}{2} h^{(N)} + C^{(N,*)} \right] \sum_{n=0}^{N} \left[1 + h^{(N)}\mathcal{L}(f^*)\right]^n \\ &= \left[\frac{M(\boldsymbol{z}^*)}{2} h^{(N)} + C^{(N,*)} \right] \frac{[1 + h^{(N)}\mathcal{L}(f^*)]^N - 1}{\mathcal{L}(f^*)} \\ &\leq \left[\frac{M(\boldsymbol{z}^*)}{2} h^{(N)} + C^{(N,*)} \right] \frac{\exp\left[h^{(N)}\mathcal{L}(f^*)N\right] - 1}{\mathcal{L}(f^*)} \\ &= \left[\frac{M(\boldsymbol{z}^*)}{2} h^{(N)} + C^{(N,*)} \right] \frac{\exp\left[\mathcal{L}(f^*)(t_e - t_s)\right] - 1}{\mathcal{L}(f^*)} \end{aligned} \tag{13}$$

Proof. (**Theorem 3**)

Special case:

Suppose we now have two networks of depth N and $2N$. The residual architectures are respectively

$$\begin{aligned} \boldsymbol{z}_{n+1}^{(N)} &= \boldsymbol{z}_n^{(N)} + h^{(N)} f_n^{(N)}(\boldsymbol{z}_n^{(N)}), \\ \boldsymbol{z}_{2(n+1)}^{(2N)} &= \boldsymbol{z}_{2n}^{(2N)} + h^{(2N)} f_{2n}^{(2N)}(\boldsymbol{z}_{2n}^{(2N)}) + h^{(2N)} f_{2n+1}^{(2N)}(\boldsymbol{z}_{2n+1}^{(2N)}). \end{aligned} \tag{14}$$

Assume the L^∞ distance of their block functions at corresponding time points is bounded by

$$\sup_{t,\boldsymbol{z}} \left| f^{(N)}(t,\boldsymbol{z}) - f^{(2N)}(t,\boldsymbol{z}) \right| = C^{(N,2N)}, \tag{15}$$

the block functions are l-Lipschitz, as

$$\sup_{n,\boldsymbol{z}_1 \neq \boldsymbol{z}_2} \frac{\left| f_n^{(N)}(\boldsymbol{z}_1) - f_n^{(N)}(\boldsymbol{z}_2) \right|}{|\boldsymbol{z}_1 - \boldsymbol{z}_2|} = \mathcal{L}(f^{(N)}), \tag{16}$$

and the residuals are bounded by

$$M(f^{(N)}) = \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}} \sup_n \left| f_n^{(N)}(\boldsymbol{z}_n^{(N)}) \right|. \tag{17}$$

By bound (17) we have

$$\begin{aligned}\left|\boldsymbol{z}_{2n+1}^{(2N)}-\boldsymbol{z}_{n}^{(N)}\right| &\leq \left|\boldsymbol{z}_{2n+1}^{(2N)}-\boldsymbol{z}_{2n}^{(2N)}\right|+\left|\boldsymbol{z}_{2n}^{(2N)}-\boldsymbol{z}_{n}^{(N)}\right| \\ &= h^{(2N)}\left|f_{2n}^{(2N)}(\boldsymbol{z}_{2n}^{(2N)})\right|+\left|\boldsymbol{z}_{2n}^{(2N)}-\boldsymbol{z}_{n}^{(N)}\right| \\ &\leq h^{(2N)}M(f^{(2N)})+\left|\boldsymbol{z}_{2n}^{(2N)}-\boldsymbol{z}_{n}^{(N)}\right|.\end{aligned} \tag{18}$$

The error propagation between their feature maps can be formulated as

$$\begin{aligned}\boldsymbol{z}_{2(n+1)}^{(2N)}-\boldsymbol{z}_{(n+1)}^{(N)} &= \left[\boldsymbol{z}_{2n}^{(2N)}-\boldsymbol{z}_{n}^{(N)}\right]+\frac{h^{(N)}}{2}\left[f_{2n}^{(2N)}(\boldsymbol{z}_{2n}^{(2N)})+f_{2n+1}^{(2N)}(\boldsymbol{z}_{2n+1}^{(2N)})-2f_{n}^{(N)}(\boldsymbol{z}_{n}^{(N)})\right] \\ &= \left[\boldsymbol{z}_{2n}^{(2N)}-\boldsymbol{z}_{n}^{(N)}\right]+\frac{h^{(N)}}{2}\left[f_{2n}^{(2N)}(\boldsymbol{z}_{2n}^{(2N)})-f_{n}^{(N)}(\boldsymbol{z}_{2n}^{(2N)})+f_{n}^{(N)}(\boldsymbol{z}_{2n}^{(2N)})-f_{n}^{(N)}(\boldsymbol{z}_{n}^{(N)})\right] \\ &\quad+\frac{h^{(N)}}{2}\left[f_{2n+1}^{(2N)}(\boldsymbol{z}_{2n+1}^{(2N)})-f_{n}^{(N)}(\boldsymbol{z}_{2n+1}^{(2N)})+f_{n}^{(N)}(\boldsymbol{z}_{2n+1}^{(2N)})-f_{n}^{(N)}(\boldsymbol{z}_{n}^{(N)})\right]\end{aligned} \tag{19}$$

Given the bounds given by (15), (16) and (18), we wil have

$$\begin{aligned}\left|\boldsymbol{z}_{2(n+1)}^{(2N)}-\boldsymbol{z}_{(n+1)}^{(N)}\right| &\leq \left|\boldsymbol{z}_{2n}^{(2N)}-\boldsymbol{z}_{n}^{(N)}\right|+h^{(N)}C^{(N,2N)}+\frac{h^{(N)}}{2}\mathcal{L}(f^{(N)})\left[\left|\boldsymbol{z}_{2n}^{(2N)}-\boldsymbol{z}_{n}^{(N)}\right|+\left|\boldsymbol{z}_{2n+1}^{(2N)}-\boldsymbol{z}_{n}^{(N)}\right|\right] \\ &\leq \left|\boldsymbol{z}_{2n}^{(2N)}-\boldsymbol{z}_{n}^{(N)}\right|+h^{(N)}C^{(N,2N)})+h^{(N)}\mathcal{L}(f^{(N)})\left|\boldsymbol{z}_{2n}^{(2N)}-\boldsymbol{z}_{n}^{(N)}\right|+0.25{h^{(N)}}^{2}\mathcal{L}(f^{(N)})M(f^{(2N)}) \\ &= (1+h^{(N)}\mathcal{L}(f^{(N)}))\left|\boldsymbol{z}_{2n}^{(2N)}-\boldsymbol{z}_{n}^{(N)}\right|+h^{(N)}\left[C^{(N,2N)})+0.25\mathcal{L}(f^{(N)})M(f^{(2N)})h^{(N)}\right]\end{aligned} \tag{20}$$

Recursively apply this relation we will have

$$\begin{aligned}\left|\boldsymbol{z}_{2N}^{(2N)}-\boldsymbol{z}_{N}^{(N)}\right| \leq & \left[1+h^{(N)}\mathcal{L}(f^{(N)})\right]^{N}\left|\boldsymbol{z}_{0}^{(2N)}-\boldsymbol{z}_{0}^{(N)}\right| \\ &+h^{(N)}\left[C^{(N,2N)})+0.25\mathcal{L}(f^{(N)})M(f^{(2N)})h^{(N)}\right]\sum_{n=0}^{N}\left[1+h^{(N)}\mathcal{L}(f^{(N)})\right]^{n}.\end{aligned} \tag{21}$$

Since $\left|\boldsymbol{z}_{0}^{(2N)}-\boldsymbol{z}_{0}^{(N)}\right|=|\boldsymbol{x}-\boldsymbol{x}|=0$, we have

$$\begin{aligned}e^{(N,2N)} &:= \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\left|\mathcal{F}^{(2N)}(\boldsymbol{x})-\mathcal{F}^{(N)}(\boldsymbol{x})\right| \\ &\equiv \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\left|\boldsymbol{z}_{2N}^{(2N)}-\boldsymbol{z}_{N}^{(N)}\right| \\ &\leq h^{(N)}\left[0.25\mathcal{L}(f^{(N)})M(f^{(2N)}))h^{(N)}+C^{(N,2N)})\right]\sum_{n=0}^{N}\left[1+h^{(N)}\mathcal{L}(f^{(N)})\right]^{n} \\ &= \left[0.25\mathcal{L}(f^{(N)})M(f^{(2N)}))h^{(N)}+C^{(N,2N)})\right]\frac{[1+h^{(N)}\mathcal{L}(f^{(N)})]^{N}-1}{\mathcal{L}(f^{(N)})} \\ &\leq \left[0.25\mathcal{L}(f^{(N)})M(f^{(2N)}))h^{(N)}+C^{(N,2N)})\right]\frac{\exp\left[\mathcal{L}(f^{(N)})(t_e-t_s)\right]-1}{\mathcal{L}(f^{(N)})}.\end{aligned} \tag{22}$$

One can easily prove that the last term preserves monotonicity with respect to $\mathcal{L}(f^{(N)})$.

General case:

Now let us extend the above theorem to general cases, *i.e.*, the comparison between networks of depth N and N^{+}. The residual architectures are

$$\begin{aligned}\boldsymbol{z}^{(N)}(t_{n+1}^{(N)}) &= \boldsymbol{z}^{(N)}(t_{n}^{(N)})+h^{(N)}f_{n}^{(N)}(\boldsymbol{z}_{n}^{(N)}), \\ \boldsymbol{z}^{(N^{+})}\left(t_{\overline{n}+1}^{(N^{+})}\right) &= \boldsymbol{z}^{(N^{+})}\left(t_{\underline{n}}^{(N^{+})}\right)+h^{(N^{+})}\sum_{n=\overline{n}}^{\underline{n}}f_{n^{+}}^{(N^{+})}\left(\boldsymbol{z}_{n^{+}}^{(N^{+})}\right),\end{aligned} \tag{23}$$

where for each $n^+ \in \{\underline{n}, ..., \overline{n}\}$, $\lfloor (t_{n^+}^{(N^+)} - t_s)/h^{(N)} + 1/2 \rfloor = n$. In other words, n is associated with the block in network $f^{(N)}$ that is the nearest neighbour of block n^+ in network $f^{(N^+)}$.

Now we can align $f_{n^+}^{(N^+)}$ and $f_n^{(N)}$ to derive the functional difference, as we did above. Note that $\sum_{n=\underline{n}}^{\overline{n}} h^{(N^+)}$ is generally not exactly equal to $h^{(N)}$, but the discrepancy should be bounded by the maximum residual, *i.e.*, $M(f^{(N^+)})$. Therefore, the bound of the temporal error in general case differs from (22) up to a scaling factor related to N and N^+.

E. Adaptive Cosine Annealing Learning Rate Scheduler

Our adaptive growing strategy prevents the determination of grow epochs prior to the training, calling for a consistent learning rate scheduler that can adjust its configuration *in situ*. Adaptive Cosine Annealing Learning Rate Scheduler, as a variant of Cosine Annealing Learning Rate Scheduler, is thus proposed. Figure 7 shows the difference of our proposed learning rate scheduler, where the annealing period is reduced every time a grow takes place.

We demonstrate that the proposed learning rate scheduler yields comparable performance. Since the performance of vanilla model in our experiment is the baseline to align with, it is sufficient to demonstrate our proposed learning rate scheduler works for vanilla training, *i.e.*, training of the deep network from scratch. On the same Nvidia GeForce GTX 1080 Ti GPU, we use our proposed scheduler and the regular Cosine Learning Rate Scheduler to train ResNet-74 on CIFAR-10, CIFAR-100 for 164 epochs, and ResNet-66 on Tiny Imagenet for 90 epochs, where the milestones are fixed to 60, 110 and 30, 60 respectively. Each run will be repeated for 3 times. Table 4 presents the average validation and test accuracy. One may find our proposed learning rate scheduler is generally better than regular one, which can be attributed to larger weight space searched given higher learning rates at the beginning of the training.

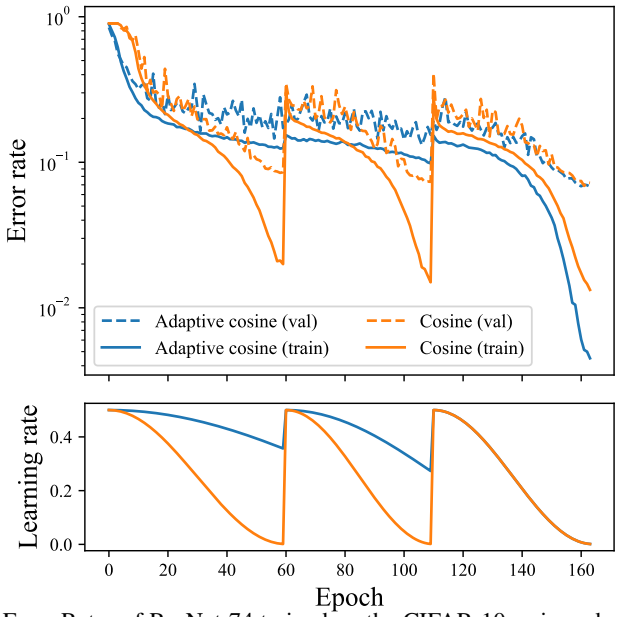


Figure 7: Training and Validation Error Rates of ResNet-74 trained on the CIFAR-10, using adaptive and regular Cosine Annealing Learning rate scheduler, respectively

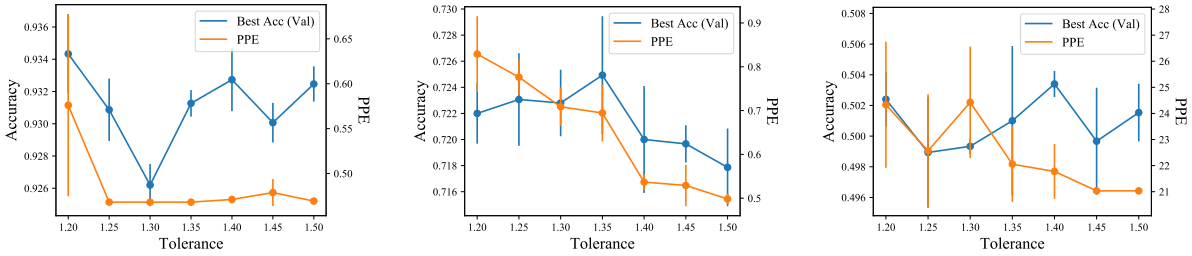
Table 4: Performance comparison between adaptive and regular Cosine Annealing Learning rate scheduler

Scheduler	CIFAR-10 (ResNet-74)		CIFAR-100 (ResNet-74)		Tiny ImageNet (ResNet-66)	
	Val	Test	Val	Test	Val	Test
Cosine	92.95 ± 0.22	93.07 ± 0.38	71.42 ± 0.75	71.67 ± 0.28	49.02 ± 0.98	47.64 ± 0.46
Adaptive cosine	93.09 ± 0.36	93.32 ± 0.67	72.61 ± 0.38	73.13 ± 0.44	50.13 ± 0.77	48.18 ± 0.21

F. Universality and Sensitivity of Lipschitz Tolerance

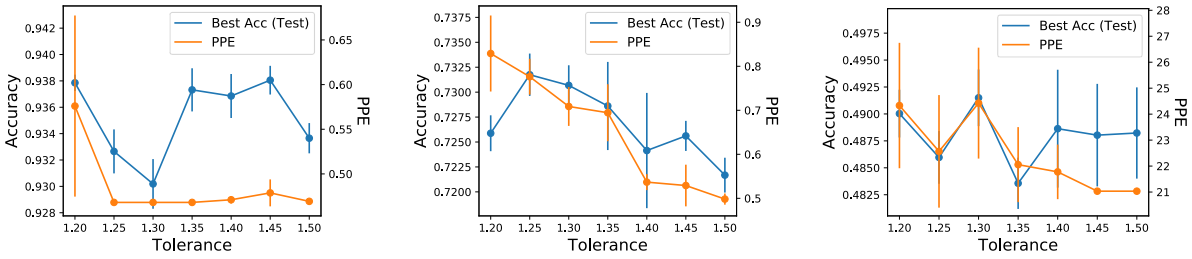
The Lipschitz tolerance is a tunable hyper-parameter in our growing strategy. It is necessary to check if the performance of the grow is sensitive to the specific choice of this hyper-parameter. Here we sample the tolerance within a wide range from 1.2 to 1.5, and test the performance of LipGrow on different dataset (CIFAR-10, CIFAR-100, Tiny ImageNet), different models (ResNet-74, ResNet-66), and different epochs (60, 90).

Figure 8 and 9 shows the best accuracy and PPE achieved given sampled tolerance values. One can find that despite the variation of the tolerance, the performance is generally stable, with the accuracy and efficiency indicators only fluctuating within a small range, which demonstrates that the specific choice of the tolerance is not sensitive. Moreover, despite the variation of the model and dataset, the overall performance, in consideration of the accuracy and efficiency at the same time, generally reaches optimal at tolerance around 1.4, which demonstrates that the reference value is roughly universal.



(a) ResNet-74 on CIFAR-10 (b) ResNet-74 on CIFAR-100 (c) ResNet-66 on Tiny-ImageNet

Figure 8: Validation Accuracy and PPE of *LipGrow*, based on Different Choices of the Tolerance. Error bar shows the mean and standard deviation of 3 repeated runs.



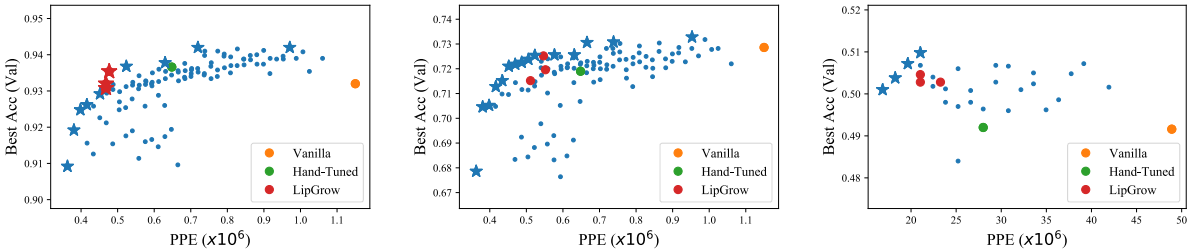
(a) ResNet-74 on CIFAR-10 (b) ResNet-74 on CIFAR-100 (c) ResNet-66 on Tiny-ImageNet

Figure 9: Test Accuracy and PPE of *LipGrow*, based on Different Choices of the Tolerance. Error bar shows the mean and standard deviation of 3 repeated runs.

G. Effectiveness of LipGrow

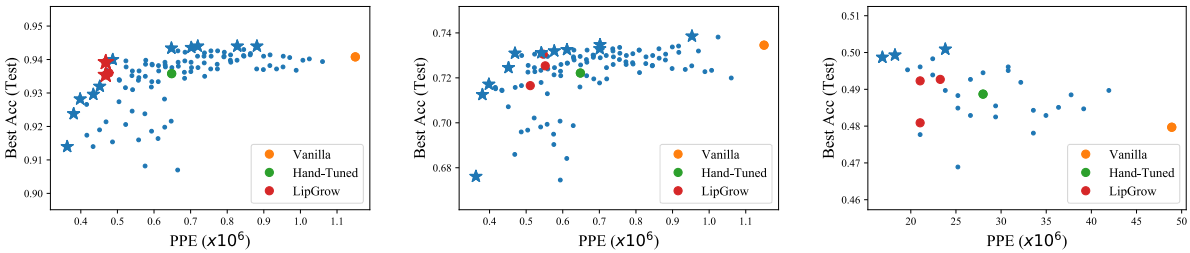
To further explore the effectiveness of LipGrow, we conduct a grid search on possible growing schedulers, for the training of ResNet-74 on CIFAR-10 and CIFAR-100, and ResNet-66 on Tiny-ImageNet. Specifically, We attempt a grow on every epoch that is a multiple of ten, which adds up to 105 trials for a total of 164 epochs, and 28 trials for a total of 90 epochs. Hand-Tuned grow epochs are fixed to 60 and 110 for 164 epochs, and 30 and 60 for 90 epochs, while LipGrow is repeated for 3 times each since it is not deterministic. The Lipschitz tolerance is fixed to 1.4 for all runs. It is worth mentioning that, the Pareto-optimal performances of the grid-search result are not directly comparable to the performance of LipGrow and should be viewed as an “upper-bound" or oracle schedulers.

The results are visualized in Figure 10 and Fig 11. Without requiring any trial-and-error adjustments, LipGrow successfully balances the training cost and the final model performance. Specifically, it achieves or is closer to Pareto-optimal performance, and is generally superior in terms of both test accuracy and training efficiency on each dataset, compared to the hand-tuned scheduler. Besides better model performance, our algorithm does not require a foresight from experts and can better fit the real-world applications. Note that for the Tiny-ImageNet dataset, training with low PPE instead gives higher accuracy. We attribute this to the sparsity of the dataset. Nevertheless, LipGrow still approaches the Pareto-optimal in such an abnormal situation.



(a) ResNet-74 on CIFAR-10 (b) ResNet-74 on CIFAR-100 (c) ResNet-66 on Tiny-ImageNet

Figure 10: Validation Accuracy vs. PPE based on Different Choices of the Growing Epochs. Pareto-optimals are highlighted as stars. Since the adaptive method is not deterministic in terms of grow epoch, *LipGrow* is repeated for 3 times. The tolerance is fixed to 1.4 for all runs.



(a) ResNet-74 on CIFAR-10 (b) ResNet-74 on CIFAR-100 (c) ResNet-66 on Tiny-ImageNet

Figure 11: Test Accuracy vs. PPE based on Different Choices of the Growing Epochs. Specifications are aligned with Figure 10.