Towards Theoretically Inspired Neural Initialization Optimization (Appendix)

A Proofs of Theoretical Results

A.1 Lemma 1 Proof

Please refer to [59] for the proof.

A.2 Theorem 2 Proof

Let θ^* denote an overall local optimum of the training loss $\mathcal{L} = \frac{1}{n}\sum_i^n l(f_\theta(x_i), y_i)$ via gradient descent from the initialization θ_0. Since we have assumed that $\forall k \in [m], i \in [n], [\nabla^2 l(f_\theta(x_i), y_i)]_{k,k} \le \mathcal{H}$, through the expansion of $l(f_{\theta^*}(x_i), y_i)$ at the sample optimum θ_i^*, we have:

$$l(f_{\theta^*}(x_i), y_i) \le l(f_{\theta_i^*}(x_i), y_i) + \nabla_\theta l(f_{\theta_i^*}(x_i), y_i)^T(\theta^* - \theta_i^*) + \frac{\mathcal{H}}{2}||\theta^* - \theta_i^*||_2^2. \tag{9}$$

At the sample optimum θ_i^*, we have $l(f_{\theta_i^*}(x_i), y_i) = 0$ and $\nabla_\theta l(f_{\theta_i^*}(x_i), y_i) = \mathbf{0}$. Then the above inequality can be rewrote as:

$$l(f_{\theta^*}(x_i), y_i) \le \frac{\mathcal{H}}{2}||\theta^* - \theta_i^*||_2^2, \tag{10}$$

and thus we have $\mathcal{L} \le \frac{\mathcal{H}}{2n}\sum_i^n ||\theta^* - \theta_i^*||_2^2$. Due to the assumption that the sample-wise optimization landscapes are nearly convex and semi-smooth at the neighborhood of the initialization θ_0, the overall optimization landscape is also convex and semi-smooth, and the overall local optimum θ^* lies in the convex hull of the sample-wise optima $\theta_i^*, \forall i \in [1, n]$. Then we have:

$$||\theta^* - \theta_i^*||_2^2 \le \sum_j ||\theta_j^* - \theta_i^*||_2^2, \tag{11}$$

and

$$\mathcal{L} \le \frac{\mathcal{H}}{2n}\sum_{i,j} ||\theta_i^* - \theta_j^*||_2^2. \tag{12}$$

We use α and β to denote the maximal and minimal ℓ_2-norms of the sample-wise optimization paths, respectively, *i.e.,* $\alpha = \max(||\theta_i^* - \theta_0||_2), \beta = \min(||\theta_i^* - \theta_0||_2), \forall i \in [1, n]$. We further denote the normalized sample-wise optimization paths as $\mathbf{p}_i = \frac{\theta_i^* - \theta_0}{\alpha}$, $0 < ||\mathbf{p}_i||_2 \le 1$. Then we have:

$$\begin{aligned}
\mathcal{L} &\le \frac{\mathcal{H}}{2n}\sum_{i,j} ||\theta_i^* - \theta_j^*||_2^2 \\
&= \frac{\mathcal{H}}{2n}\sum_{i,j} ||\theta_i^* - \theta_0 - (\theta_j^* - \theta_0)||_2^2 \\
&= \frac{\mathcal{H}\alpha^2}{2n}\sum_{i,j} \frac{||\theta_i^* - \theta_0 - (\theta_j^* - \theta_0)||_2^2}{\alpha^2} \\
&= \frac{\mathcal{H}\alpha^2}{2n}\sum_{i,j} ||\mathbf{p}_i - \mathbf{p}_j||_2^2 \\
&\le \frac{\mathcal{H}\alpha^2}{2n}\sum_{i,j} \frac{||\mathbf{p}_i||_2^2 + ||\mathbf{p}_j||_2^2 - 2\mathbf{p}_i^T\mathbf{p}_j}{||\mathbf{p}_i||_2 \cdot ||\mathbf{p}_j||_2} \\
&= \frac{\mathcal{H}\alpha^2}{2n}\sum_{i,j} \left(\frac{||\mathbf{p}_i||_2}{||\mathbf{p}_j||_2} + \frac{||\mathbf{p}_j||_2}{||\mathbf{p}_i||_2} - 2\frac{\mathbf{p}_i^T\mathbf{p}_j}{||\mathbf{p}_i||_2 \cdot ||\mathbf{p}_j||_2}\right) \\
&\overset{a}{\le} \frac{\mathcal{H}\alpha^2}{2n}\sum_{i,j} \left(2\frac{\alpha}{\beta} - 2\cos\angle(\mathbf{p}_i, \mathbf{p}_j)\right)
\end{aligned} \tag{13}$$

$$
\begin{aligned}
&= \frac{\mathcal{H}\alpha^2}{n} \sum_{i,j} \left(\frac{\alpha}{\beta} - \cos \angle(\theta_i^* - \theta_0, \theta_j^* - \theta_0) \right) \\
&= \Theta_{S,l}(f_{\theta_0}(\cdot)),
\end{aligned}
$$

where $\overset{a}{\leq}$ holds because for any pair of (i, j), $i, j \in [1, n]$, we have $\frac{||\mathbf{p}_i||_2}{||\mathbf{p}_j||_2} = \frac{||\theta_i^* - \theta_0||_2}{||\theta_j^* - \theta_0||_2} \leq \frac{\alpha}{\beta}$.

A.3 Theorem 3 Proof

We use the population error $\mathbb{E}_{(x_u,y_u)\sim\mathcal{D}}[l(f_{\theta^*}(x_u), y_u)]$ to measure the generalization performance, where $\mathcal{D}$ is the data distribution for both training and test data, and (x_u, y_u) refers to the test sample. With probability $1 - \delta$, we have:

$$
\begin{aligned}
\mathbb{E}_{(x_u,y_u)\sim\mathcal{D}}[l(f_{\theta^*}(x_u), y_u)] &\leq \frac{\mathcal{H}}{2}\mathbb{E}_{(x_u,y_u)\sim\mathcal{D}}[||\theta^* - \theta_u^*||_2^2] && (14) \\
&\leq \frac{\mathcal{H}}{2n}\sum_i^n ||\theta^* - \theta_i^*||_2^2 + \frac{\sigma}{\sqrt{n\delta}} && (15) \\
&\leq \Theta_{S,l}(f_{\theta_0}(\cdot)) + \frac{\sigma}{\sqrt{n\delta}}, && (16)
\end{aligned}
$$

where n is the number of training samples and σ is a positive constant such that $Var_{(x_u,y_u)\sim\mathcal{D}}[||\theta^* - \theta_u^*||_2^2] \leq \sigma^2$. Eq. (14) is based on Eq. (10), Eq. (15) holds following Chebyshev's inequality, and Eq. (16) is derived from the result of Theorem 2. With Theorem 2 and Theorem 3, we can conclude that $\Theta_{S,l}(f_{\theta_0}(\cdot))$ is the upper bound for both training and generalization error, and thus can be used as an indicator to evaluate the quality of the initialization θ_0.

B Experimental Details

In this section, we provide more information about the implementation details, the architectures, the configurations of NIO, and the training details after NIO.

B.1 Implementation Details

Selection of Hyper-parameters. The hypeparameters used in the NIO Algorithm 3 are mainly selected according to the following principles: (1) For the number of sub-batches D and overlap ratio r in the batch-wise relaxation, while a smaller D corresponds to a faster inference speed, it also means more violation of the sample-wise analysis. Therefore, the smaller D is, the larger r should be adopted to relieve the violation. Our ablation study also validates this practice. For all experiments, we set D as 2 to obtain the fastest initialization speed and r is correspondingly selected from $0.6 \sim 0.8$, proportional to the batch size B. (2) For the iteration steps T, we set it to a number that traverses all the training samples on CIFAR-10/100. For the large-scale dataset ImageNet, traversing all the training samples is time-consuming. So we set T as 100 for experiments on ImageNet. (3) For the upper bound of the gradient norm γ, we follow the choices suggested in [60]. The learning rate τ of scale coefficients is set within the range from 10^{-3} to 1. Networks with more parameters are initialized with a smaller learning rate τ.

B.2 On CIFAR-10/CIFAR-100

Architecture Details. The architectures adopted include a ResNet-110 with an initial width of 16 and two convolution layers in each residual block; a 28-layer Wide ResNet with a widen factor of 10 (WRN-28-10); and a DenseNet-100. Following [60], for experiments without BN layers, we replace the BN layers with learnable bias parameters in the corresponding places to isolate the normalization effect from the affine function in BN. When testing NIO on these architectures, we first adopt the non-zero Kaiming initialization to all convolution and FC layers, and then perform NIO upon the initialized parameters. These architecture settings remain unchanged for CIFAR-100 except that the channel size of the last linear layer is adjusted according to the number of categories.

NIO Configurations. For architectures with BN layers enabled, we select the learning rate τ from relatively larger values $\{3\times10^{-1}, 2.5\times10^{-1}, 2\times10^{-1}, 1.5\times10^{-1}, 1\times10^{-1}\}$. We find that the best τ for ResNet-110 (w/ BN), DenseNet-100 (w/ BN), and WRN-28-10 (w/ BN) are $1\times10^{-1}, 1.5\times10^{-1}$, and 2.5×10^{-1}, respectively. For architectures without BN layers, we try τ from smaller values $\{3\times10^{-2}, 2.5\times10^{-2}, 2\times10^{-2}, 1.5\times10^{-2}, 1\times10^{-2}\}$. The best τ for ResNet-110 (w/o BN), DenseNet-100 (w/o BN), and WRN-28-10 (w/o BN) are $1.5\times10^{-2}, 1.5\times10^{-2}$, and 3×10^{-1}, respectively. To speed up NIO for a fast initialization, we use the batch setting for all architectures with and without BN layers. We set the number of sub-batches D as 2 under a batch size of $B = 128$. To relieve the violation of the sample-wise analysis, we adopt an overlap between the two sub-batches. Specifically, the best overlap ratios r for ResNet-110 (w/ BN), DenseNet-100 (w/ BN), and WRN-28-10 (w/ BN) are $0.6, 0.6$, and 0.7, respectively. The best overlap ratios r for ResNet-110 (w/o BN), DenseNet-100 (w/o BN), and WRN-28-10 (w/o BN) are $0.7, 0.75$, and 0.8 respectively. Finally, for the upper bound of the gradient norm γ, we follow the choices suggested in [60]. The best γ for ResNet-110 (w/ BN), ResNet-110 (w/o BN), DenseNet-100 (w/ BN), DenseNet-100 (w/o BN), WRN-28-10 (w/ BN), and WRN-28-10 (w/o BN) are $5, 4, 3, 3.5, 2$, and 2, respectively for CIFAR-10. CIFAR-100 has 10^2 classes, so the initial cross entropy loss will be about 2 times as large as the one on CIFAR-10 for the same architecture. Following the principle in [60], we accordingly set γ 2 times as large as that for CIFAR-10. NIO is iterated until all the training samples are covered.

Training details. After NIO, we train these networks on CIFAR-10 and CIFAR-100 for 500 epochs with a batch size of 128 on a single GPU. The initial learning rate is 0.1 and decays after each iteration following the cosine annealing learning rate schedule without restart [33]. We set weight decay to 10^{-4} in all experiments. For networks without BN layers, we apply gradient clipping (with a maximum gradient norm of 1) to prevent from gradient explosion. Following the standard data augmentation schemes, we adopt random cropping, random flipping, and cutout [10] for all networks. We do not use dropout in any of our experiments.

B.3 On ImageNet

Architecture Details. To show the extensibility of NIO to large datasets and novel architectures, we also conduct experiments on ImageNet with ResNet-50 and Swin-Transformer. Specifically, we use the standard ResNet-50 architecture that is stacked by 3-layer bottleneck blocks. BN layers are enabled as default. For experiments of Swin-Transformer, we adopt the Swin-Tiny configuration, where the window size M is 7, the query dimension of each head d is 32, and the expansion ratio of each MLP block is 4. The initial channel number is 96, and the numbers of blocks for four stages are $\{2, 2, 6, 2\}$. We follow the official implementation in [32].

NIO Configurations. Since the networks on ImageNet have more parameters than the architectures on CIFAR10/100, we adopt a relatively smaller learning rate τ. Concretely, we try τ from $\{5\times10^{-2}, 1\times10^{-2}, 5\times10^{-3}, 3\times10^{-3}, 1\times10^{-3}\}$ and choose $\tau = 3\times10^{-3}$ for both ResNet-50 and Swin-Transformer. We set the number of sub-batches $D = 4$ under the batch size $B = 64$ and use an overlap ratio of $r = 0.2$ for both architectures. Following the previous principle, we change the upper bound of the gradient norm γ three times as large as that for CIFAR-10 due to the increased number of categories. Finally, we run NIO for 100 iterations on ImageNet to keep the initialization process efficient.

Training Details. Following the standard implementations, we train ResNet-50 for 100 epochs and Swin-Transformer for 300 epochs. The batchsize is 256 for ResNet-50 and 1024 for Swin-Transformer among 8 NVIDIA A-100 GPUs. ResNet-50 is trained by the SGD optimizer with a weight decay of 10^{-4} and a momentum of 0.9. Swin-Transformer is trained by the AdamW optimizer with a weight decay of 0.05 and a momentum of (0.9, 0.999). The initial learning for ResNet-50 is 0.1, and is divided by a factor of 10 at epoch 30, 60, and 90. The base learning rate for Swin-Transformer is 1×10^{-3}, and follows the cosine annealing learning rate schedule without restart. For Swin-Transformer with warmup enabled, a linear learning rate warmup is used for the first 20 epochs of training. Swin-Transformer adopts gradient clipping with a maximum gradient norm of 5, and the standard augmentation schemes, such as Mixup [57].

C More Explanations and Discussions

As suggested by the reviewers of this paper, we offer more explanations and discussions here for some contents in the main paper.

1. Explanations for Figure 1.

In Figure 1 (c) and (d), it seems that an initialization with a smaller cosine similarity (larger angle) is closer to the global optimum of the two samples. But we should note that:

(1) The optimum of a given network is not unique. It is dependent on its initialization. The model with different initialization points will converge to different optimal parameters, even though their performances may be close. So, what we do here is to look for an initialization whose sample-wise optimization paths are more consistent (smaller angle), instead of the inverse way—looking for a point closer to the global optimum of some given sample-wise optima and landscapes, which is impossible for a real network. The larger optimization path consistency induces a smaller gradient variance, whose benefits have been supported by prior studies [31, 61].

(2) The illustration in Figure 1 is a toy example with only two samples and simple landscapes. In a real network, the landscapes are highly non-convex in a high dimension, and there are a larger number of training samples. In this case, the global optimum does not necessarily lie in the convex hull. The point with a larger angle to sample-wise optima is not ensured to be closer to the global optimum.

In conclusion, an initialization with more consistent sample-wise optimization paths is more favorable. Our aim is to look for such initialization, instead of the global optimum of given sample-wise landscapes. Figure 1 (c) and (d) are toy examples that are only used to show that the metric Eq. (1) is agnostic of initialization, while our Eq. (2) reflects the optimization path consistency and is a function of initialization.

2. Explanations for some concepts.

Sample-wise optimization refers to training the model only on each single sample. So, the optimization path for sample i can be characterized by $\theta_i^* - \theta_0$, where θ_0 is the initialized point, and θ_i^* is the converged optimum of only training on sample i. Because all sample-wise optimization paths have the same starting point θ_0, we use the averaged Cosine similarity to measure their consistency. The consistency of sample-wise optimization path can be formulated as $\frac{1}{n^2}\sum_{i,j}\cos\angle\left(\theta_i^* - \theta_0, \theta_j^* - \theta_0\right)$. So, if the angle between the paths from the initialization θ_0 to each local optimum θ_i^* is small, we will have more consistent sample-wise optimization paths. It is approximated by our GradCosine in Line 6 in Algorithm 1.

It is shown that our proposed Eq. (2) reflects the optimization path consistency and is a function of initialization θ_0. The aim of our NIO is to look for an initialization θ_0 that maximizes GradCosine.

3. Discussions about the rationality of the approximation in Eq. (3).

Note that θ_i^* in Eq. (3) is not the global optimum. It is the local optimum for sample i. If we optimize a model on only one training sample, it is very easy to finish the training. Only several iterations are needed to attain a zero loss. So, we make the first-order approximation for the sample-wise optimization, *i.e.*, the sample-wise optimum can be reached via only one step of gradient descent.

4. Discussions about the comparison with GradSign [59].

In GradSign [59], the metric of sample-wise optima density (Eq. (1) in our paper) is proposed and proved to be an upper bound of training and generalization error. Because sample-wise optima are not tractable, they approximate it by counting the gradient sign, which is non-differentiable. The metric is used to rank neural architectures without training for a neural architecture search purpose.

However, the goal of our work is to develop a method for initialization optimization, instead of architecture search. So, our theoretical metric needs to be differentiable and be a function of initialization. The differences between [59] and our work are summarized as follows:

(1) The metric Eq. (1) proposed in [59] is agnostic of initialization (as shown in Figure 1 (c) and (d)). Its approximated quantity, GradSign, is non-differentiable. So, it cannot be used for initialization optimization. In contrast, our metric Eq. (2) reflects the optimization path consistency as a function

of initialization. It is approximated by the differentiable GradCosine, and thus can serve for our initialization optimization purpose.

(2) Our method considers optimization path consistency, which is intimately related to a favorable training dynamic. The larger optimization path consistency induces a smaller gradient variance, whose benefits have been supported by prior studies [31, 61]. See more details in the "Relations to Favorable Training Dynamic" paragraph. As a comparison, [59] does not enjoy this advantage.

(3) Although both metrics Eq. (1) and Eq. (2) are related to model performance, because our work and [59] aim for different tasks, the other parts of our work including the GradCosine quantity, the objective of our neural initialization optimization, and the algorithm to solve it, are original and have no similarity with [59].