Neural Network Training Techniques Regularize Optimization Trajectory: An Empirical Study

Cheng Chen
Department of ECE
University of Utah
u0952128@utah.edu

Junjie Yang
Department of ECE
The Ohio State University
yang.4972@osu.edu

Yi Zhou
Department of ECE
University of Utah
yi.zhou@utah.edu

***Abstract*—Modern deep neural network (DNN) trainings utilize various training techniques, e.g., nonlinear activation functions, batch normalization, skip-connections, etc. Despite their effectiveness, it is still mysterious how they help accelerate DNN trainings in practice. In this paper, we provide an empirical study of the regularization effect of these training techniques on DNN optimization. Specifically, we find that the optimization trajectories of successful DNN trainings consistently obey a certain regularity principle that regularizes the model update direction to be aligned with the trajectory direction. Theoretically, we show that such a regularity principle leads to a convergence guarantee in nonconvex optimization and the convergence rate depends on a regularization parameter. Empirically, we find that DNN trainings that apply the training techniques achieve a fast convergence and obey the regularity principle with a large regularization parameter, implying that the model updates are well aligned with the trajectory. On the other hand, DNN trainings without the training techniques have slow convergence and obey the regularity principle with a small regularization parameter, implying that the model updates are not well aligned with the trajectory.**

***Index Terms*—Neural network, training techniques, nonconvex optimization, optimization trajectories, regularity principle**

I. Introduction

Deep learning has been successfully applied to various domains such as computer vision, natural language processing, etc, and has achieved state-of-art performance in solving challenging tasks. Although deep neural networks (DNNs) have been well-known for decades to have great expressive power [5], the empirical success of training DNNs postponed to recent years when sufficient computation power is accessible and effective DNN *training techniques* are developed.

The milestone developments of DNN training techniques can be roughly divided into two categories. First, various techniques have been developed at different levels of *neural network design*. Specifically, at the neuron level, various functions have been applied to activate the neurons, e.g., sigmoid function, hyperbolic tangent (tanh) function and the more popular rectified linear unit (ReLU) function [19]. At the layer level, batch normalization (BN) has been widely applied to the hidden layers of DNNs to stabilize the training [13]. Moreover, at the architecture level, skip-connections have been introduced to enable successful training of deep networks [11], [12], [28], [29]. Second, various efficient stochastic *optimization algorithms* have been developed for DNN training, e.g., stochastic gradient descent (SGD) [24], [26], SGD with momentum [20], [22] and Adam [14], etc. Table I provides a summary of these important DNN training techniques.

TABLE I
SUMMARY OF DNN TRAINING TECHNIQUES

Neuron activation	Layer normalization	Network architecture	Optimization algorithm
sigmoid, tanh, ReLU	Batch normalization	Skip-connection	SGD, SGD-momemtum, Adam

Although these training techniques have been widely applied in practical DNN training, there is limited understanding of how they help facilitate the training to achieve a global minimum of the network. In the existing literature, it is known that the sigmoid and tanh activation functions can cause the vanishing gradient problem, and the ReLU activation function is a popular replacement that avoids this issue [9], [19]. On the other hand, the batch normalization was originally proposed to reduce the internal covariance shift [13], and more recent studies show that it allows to use a large learning rate [2] and improves the loss landscape [27]. The skip-connection has been shown to help eliminate singularities and degeneracies [21] and improve the loss landscape [10]. Moreover, regarding the optimization algorithm, the momentum scheme has been well-known to accelerate convex optimization [20] and is also widely applied to accelerate nonconvex optimization [8], whereas the Adam algorithm normalizes the update in each dimension to accelerate deep learning optimization [14]. While these existing studies provide partial explanations to the effectiveness of DNN training techniques, their reasonings are from very different perspectives and are lack of a principled understanding. In particular, these studies do not fully explain why these diverse types of DNN training techniques can facilitate the training in practice.

In this paper, we take a step toward understanding DNN training techniques by providing a systematic empirical study of their regularization effect in the perspective of optimization. We empirically show that the training techniques regularize the optimization trajectory in practical DNN training following a regularity principle, which quantifies the regularization effect

978-1-7281-6251-5/20/$31.00 ©2020 IEEE

that further determines the convergence rate in nonconvex optimization.

A. Related Work

DNN training techniques: Various training techniques have been developed for DNN training. Examples include piecewise linear activation functions, e.g., ReLU [19], ELU [4], leaky ReLU [18], batch normalization [13], skip-connection [11], [28], [29] and advanced optimizers such as SGD with momentum [22], Adagrad [7], Adam [14], AMSgrad [23]. The ReLU activation function and skip connection have been shown to help avoid the vanishing gradient problem and improve the loss landscape [10], [31], [33], [36]. The batch normalization has been shown to help avoid the internal covariance shift problem [13]. More recent studies show that batch normalization allows to adopt a large learning rate [2] and improves the loss landscape in the training [27]. The convergence properties of the advanced optimizers have been studied in nonconvex optimization [3].

Optimization properties of nonconvex ML: Many nonconvex ML models have amenable properties for accelerating the optimization. For example, nonconvex problems such as phase retrieval [30], low-rank matrix recovery, blind deconvolution [16] and neural network sensing [32] satisfy the local regularity geometry around the global minimum [16], [32], [33], [35], which guarantees the linear convergence of gradient-based algorithms. Recently, DNN trainings that use SGD have been shown to follow a star-convex optimization path [34]. Moreover, there are many works that study the convergence of gradient methods in training neural networks, e.g., [1], [6], [17]

II. Regularity Principle for Nonconvex Optimization

We first introduce a regularity principle that regularizes the optimization trajectory of stochastic algorithms.

A. Optimization Trajectory of Stochastic Algorithm

The goal of a machine learning task is to search for a good ML model θ that minimizes the total loss f on a set of training data samples $\mathcal{Z}: \{z_i\}_{i=1}^n$. The problem is formally written as

$$\min_{\theta\in\mathbb{R}^d} \ f(\theta;\mathcal{Z}) := \frac{1}{n}\sum_{i=1}^{n} \ell(\theta; z_i), \tag{P}$$

where $\ell(\cdot; z_i): \mathbb{R}^d \to \mathbb{R}$ corresponds to the loss on the i-th data sample z_i. Consider a generic stochastic algorithm (SA) that is initialized with certain model θ_0. In each iteration k, the SA samples a data sample $z_{\xi_k} \in \mathcal{Z}$, where $\xi_k \in \{1, ..., n\}$ is obtained via random sampling with reshuffle. Based on the current model θ_k and the sampled data z_{ξ_k}, the SA generates a stochastic update $U(\theta_k; z_{\xi_k})$ and applies it to update the model with a learning rate $\eta > 0$ according to the update rule

$$\text{(SA)}: \quad \theta_{k+1} = \theta_k - \eta U(\theta_k; z_{\xi_k}), \ k = 0, 1, 2, ... \tag{1}$$

Equation (1) covers the update rule of many existing optimizers for DNN training. For example, the stochastic gradient descent (SGD) algorithm chooses the update $U(\theta_k; z_{\xi_k})$ to be the stochastic gradient $\nabla\ell(\theta_k; z_{\xi_k})$. In comparison, the SGD with momentum algorithm generates the update using an extra momentum step, and the Adam algorithm generates the update as a moving average of the stochastic gradients normalized by the moving average of their second moments. We formally define the optimization trajectory of SA as follows.

Definition 1 (Optimization trajectory)**.** *The optimization trajectory of SA is the generated sequence of model parameters $\{\theta_k\}_k$ in the stochastic optimization process.*

We note that the stochastic optimization trajectory generally depends on the specific update rule U, which is affected by the training techniques. For example, neuron activation functions, batch normalization and skip-connections affect the back-propagation process in the computation of the model update U. On the other hand, optimization algorithms such as SGD with momentum and Adam specify the model update U in different forms. Therefore, we are motivated to understand *how the training techniques affect the optimization trajectory*.

B. Regularity Principle for Optimization Trajectory

We next propose a regularity principle for the optimization trajectory generated by SA.

Definition 2 (Regularity principle for SA)**.** *Apply SA to solve the problem (P) for T iterations and generate an optimization trajectory $\{\theta_0, \theta_1, ..., \theta_T\}$. We say that the trajectory satisfies the regularity principle with parameter $\gamma > 0$ if for all $k = 0, 1, ...T-1$,*

$$\langle \theta_k - \theta_T, U(\theta_k; z_{\xi_k})\rangle \geq \frac{\eta}{2}\|U(\theta_k; z_{\xi_k})\|^2 + \gamma\big(\ell(\theta_k; z_{\xi_k}) - \inf_{\theta\in\mathbb{R}^d} \ell(\theta)\big). \tag{2}$$

To elaborate, the left hand side of (2) measures the coherence between $\theta_k - \theta_T$ and the corresponding model update $U(\theta_k; z_{\xi_k})$. Intuitively, a positive coherence implies that the model update is well aligned with the corresponding trajectory direction $\theta_k - \theta_T$. On the other hand, the right hand side of (2) regularizes the coherence by two non-negative terms. Hence, a larger value of γ implies that the update $U(\theta_k; z_{\xi_k})$ is more coherent with the trajectory direction $\theta_k - \theta_T$. Fig. 1 illustrates two optimization trajectories that satisfy the regularity principle with large and small γ, respectively. Intuitively, the left trajectory is close to a straight line where the update direction $-U(\theta_k; z_{\xi_k})$ is well aligned with the trajectory direction $\theta_k - \theta_T$, implying a large γ. As a comparison, the right trajectory is curved and the update direction $-U(\theta_k; z_{\xi_k})$ diverges from the trajectory direction $\theta_k - \theta_T$, implying a small γ. Therefore, the left trajectory is well regularized by the regularity principle and the trajectory length is much shorter than the curved trajectory, implying a faster convergence.

Next, we analyze the convergence of SA under the regularity principle in over-parameterized nonconvex optimization. In specific, we assume the model θ is over-parameterized so that the global minimum of the total loss $f(\theta; \mathcal{Z})$ interpolates all the sample losses $\{\ell(\theta; z_i)\}_{i=1}^n$, i.e., $\inf_{\theta\in\mathbb{R}^d} f(\theta) =$

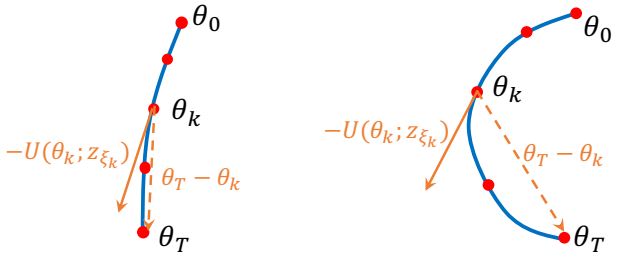

Fig. 1. Illustration of optimization trajectories. The left trajectory satisfies regularity principle with large γ and the right trajectory satisfies regularity principle with small γ.

$\frac{1}{n}\sum_{i=1}^{n} \inf_{\theta\in\mathbb{R}^d} \ell(\theta; z_i)$. Such a scenario is common in deep learning where neural networks typically have more parameters than samples and the training can overfit all the data samples.

Theorem 1 (Convergence under Regularity Principle)**.** *Apply SA to solve the over-parameterized problem (P) and generate an optimization trajectory* $\{\theta_0, \theta_1, ..., \theta_T\}$*. If the optimization trajectory satisfies the regularity principle with parameter* $\gamma > 0$*, then after* $T = nB, B \in \mathbb{N}$ *iterations (i.e., B epochs), the average loss converges to the global minimum at the rate*

$$\frac{1}{T}\sum_{k=0}^{T-1} \ell(\theta_k; z_{\xi_k}) - \inf_{\theta\in\mathbb{R}^d} f(\theta; \mathcal{Z}) \leq \frac{1}{2\eta T}\frac{\|\theta_0 - \theta_T\|^2}{\gamma}. \quad (3)$$

In Theorem 1, the convergence rate involves the factor γ^{-1}. Intuitively, if the optimization trajectory is well-regularized by the regularity principle with a large γ, then the trajectory is close to a straight line and the loss achieves a fast convergence to the global minimum. Based on the convergence rate in (3), we can quantify the regularization effect of the regularity principle on the optimization trajectory by evaluating the problem-dependent constant factor $\|\theta_0 - \theta_T\|^2\gamma^{-1}$. In the subsequent sections, we empirically compute such a factor to explore the regularization effect of training techniques in deep learning optimization.

III. Experiments on Network-level Training Techniques

In this section, we examine the validity of the regularity principle in training DNNs with different neural network-level training techniques. We outline the exploration plan below.

Exploration plan: We train the network for a sufficient number of epochs to achieve an approximate global minimum. We store the optimization trajectory $\{\theta_k\}_k$, loss $\{\ell(\theta_k; z_{\xi_k})\}_k$ and update $\{U(\theta_k; z_{\xi_k})\}_k$ that are generated in each training. Then, we compute the upper bound for γ in each iteration according to (2), where θ_T corresponds to the network parameters produced in the last training iteration. We report the convergence rate factor $\gamma/\|\theta_0 - \theta_T\|^2$ in (3).

A. Effect of Neuron Activation Function on Regularity Principle

Experiment setup: We train a ResNet-18 [11] with different choices of activation functions for all the nonlinear neurons. The activation functions that we explore include sigmoid, tanh, ReLU and leaky ReLU (with slope 10^{-2}). We apply the standard SGD optimizer with a fixed initialization point, a mini-batch size 128 and a constant learning rate $\eta = 0.05$ to train these networks for 150 epochs on the CIFAR-10 and CIFAR-100 datasets [15] and use the cross-entropy loss.

Fig. 2 presents the ResNet training results of the training loss and the factor $\gamma/\|\theta_0 - \theta_T\|^2$ involved in the convergence result (3). It can be seen that all the trainings that adopt different activation functions obey the regularity principle with $\gamma > 0$ in the training process. In particular, observe that the trainings with ReLU types of activation functions converge faster than those with tanh activation function, which is further faster than the trainings with sigmoid activation function. Moreover, one can see that the trainings with ReLU types of activation functions satisfy the regularity principle with the largest γ, whereas the trainings with sigmoid activation function have the smallest γ. These observations are consistent with Theorem 1, where a well-regularized optimization trajectory with a larger γ implies faster convergence.

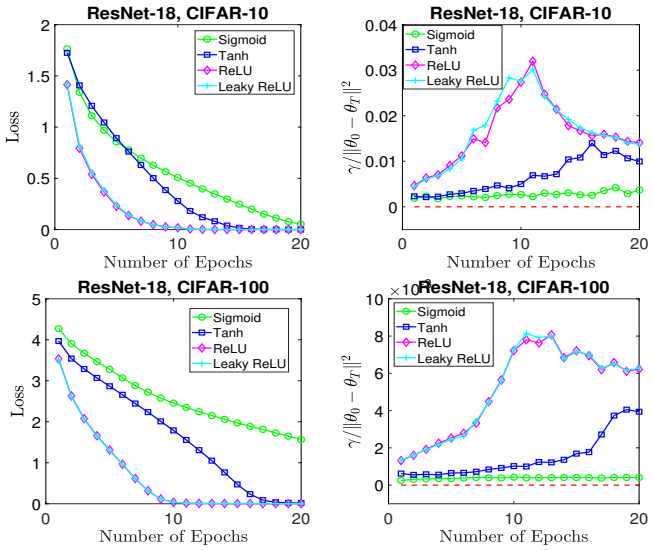

Fig. 2. Training ResNet-18 with different activation functions.

Experiment setup: Next, we train a U-Net [25] with different choices of activation functions on the CIFAR-10 and CIFAR-100 datasets. The activation functions that we explore include sigmoid, tanh, ReLU and leaky ReLU (with slope 10^{-2}). We apply the standard SGD optimizer with a fixed initialization point, a mini-batch size 128 and a constant learning rate $\eta = 0.005$ to train these networks for 150 epochs, and we use the MSE loss.

Fig. 3 shows the training loss curves and their corresponding $\gamma/\|\theta_0 - \theta_T\|^2$ of the regularity principle. In these U-Net trainings, it can be seen that the trainings with the sigmoid activation function converge faster than those with the ReLU types of activation functions, and the trainings with the tanh activation function converge the slowest. Furthermore, one can observe that the training trajectories with the sigmoid activation function obey the regularity principle with the largest γ, whereas the training trajectories with the tanh activation function satisfy the regularity principle with the smallest γ. This is consistent with Theorem 1.

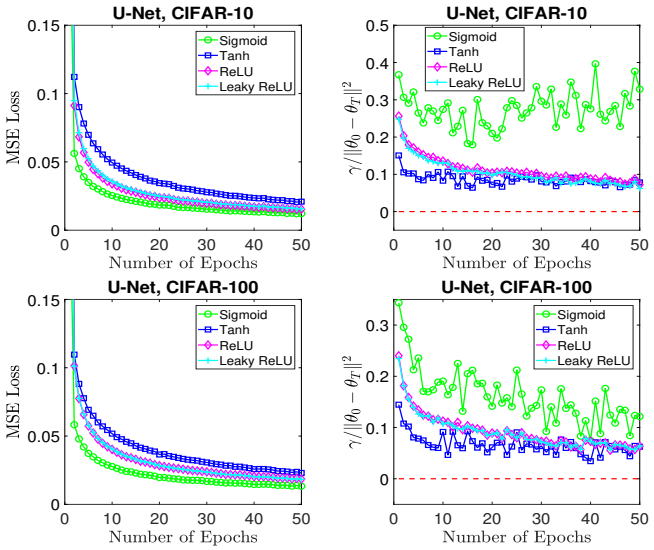

Fig. 3. Training U-Net with different activation functions.

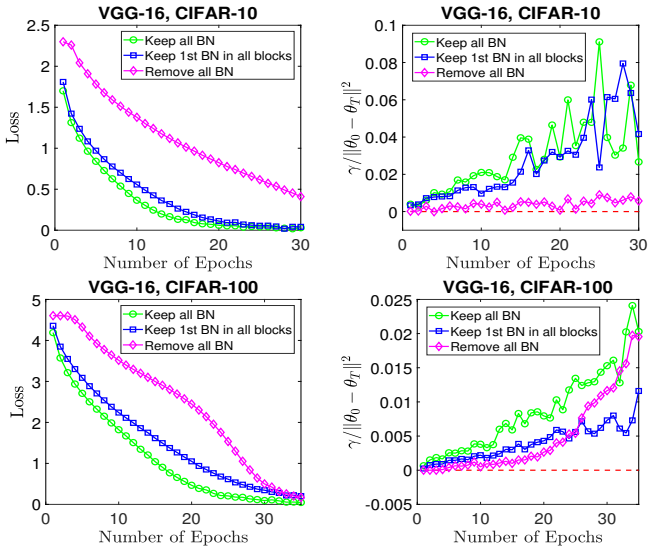

Fig. 4. Training VGG with(out) batch normalization on CIFAR-10/100.

B. *Effect of Batch Normalization on Regularity Principle*

Experiment setup: We train a VGG-16 network with the settings: 1) keep all the BN layers; 2) keep the first BN layer in each block; and 3) remove all the BN layers. We remove all dropout layers in the VGG networks. We apply SGD with a fixed initialization, a constant learning rate ($\eta = 0.01$) and batch size 128 to train on the CIFAR-10 and CIFAR-100 datasets, respectively, and we use the cross-entropy loss.

Fig. 4 shows the VGG training results on the CIFAR-10/100 datasets. It can be seen that the trainings with all BN layers removed suffer from a significant convergence slow down, and the corresponding optimization trajectories obey the regularity principle with a very small γ. On the other hand, the trainings that keep the first BN layer in each block converge as fast as those that keep all the BN layers, and their optimization trajectories obey the regularity principle with a large γ. These empirical observations corroborate the theoretical implication of the regularity principle in Theorem 1.

C. *Effect of Skip-connection on Regularity Principle*

Experiment setup: We train the ResNet-34 with the settings: 1) keep all the skip-connections; and 2) keep the first skip-connection in each block; and 3) keep the first two skip-connections in each block. We apply SGD with a fixed initialization point, a constant learning rate $\eta = 0.05$ and batch size 128 to train for 150 epochs on the CIFAR-10 and CIFAR-100 datasets and use the cross-entropy loss.

Fig. 5 shows the ResNet-34 training results. One can see that the trainings that keep more skip-connections achieve a faster convergence, and the corresponding optimization trajectories obey the regularity principle with a larger γ. These observations are consistent with our theoretical understanding of the regularity principle and they imply that skip-connection helps to regularize the optimization trajectory.

Experiment setup: We also train a U-Net on the CIFAR-10 and CIFAR-100 datasets and consider the settings: 1) keep all the skip-connections; 2) keep the last 3,2,1 skip-connections, respectively and 3) remove all the skip-connections. We apply the standard SGD optimizer with a fixed initialization point, a mini-batch size 128 and a constant learning rate $\eta = 0.005$ to train these networks for 150 epochs, and we use the MSE loss. The training results are shown in Fig. 6, where one can make similar observations as those above.

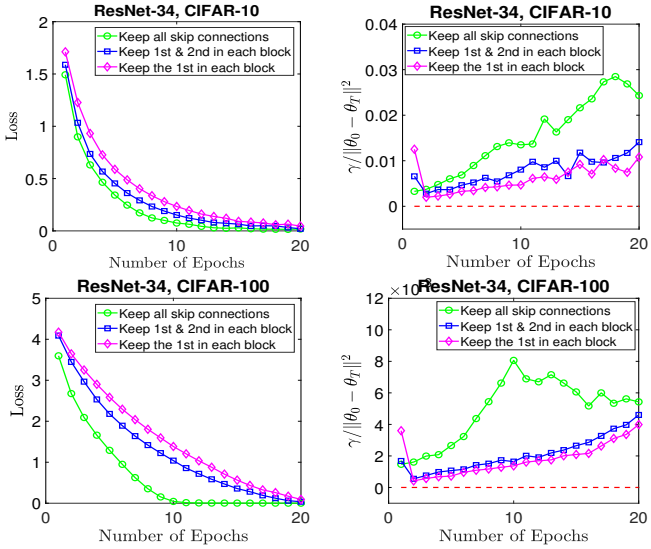

Fig. 5. Training ResNet-34 with and without skip-connections.

IV. Experiments on Optimization-level Training Techniques

Experiment setup: We train the ResNet-18 using SGD, SGD with momentum and Adam, respectively. We apply a fixed initialization point, a constant learning rate $\eta = 0.001$ and batch size 128 to all the optimizers and train these networks on the CIFAR-10 and CIFAR-100 datasets and use the cross-entropy loss. We set the momentum to be 0.5 for the SGD

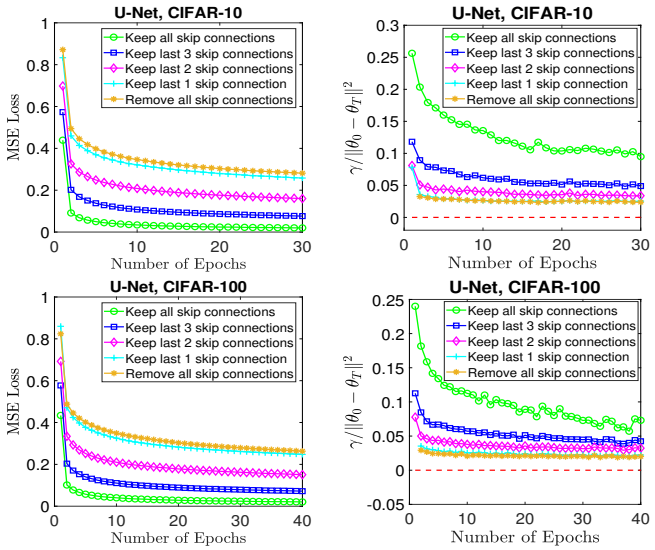

Fig. 6. Training U-Net with and without skip-connections.

with momentum and set $\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-2}$ for the Adam.

Fig. 7 presents the ResNet training results on the CIFAR-10 and CIFAR-100 datasets. In all these trainings, the Adam algorithm achieves the fastest convergence and is followed by the SGD with momentum, whereas the vanilla SGD achieves the slowest convergence. Furthermore, it can be seen that the optimization trajectories driven by Adam obey the regularity principle with the largest γ, and the trajectories driven by SGD types of algorithms is less regularized by the regularity principle. All these observations corroborate the theoretical implication of the regularity principle.

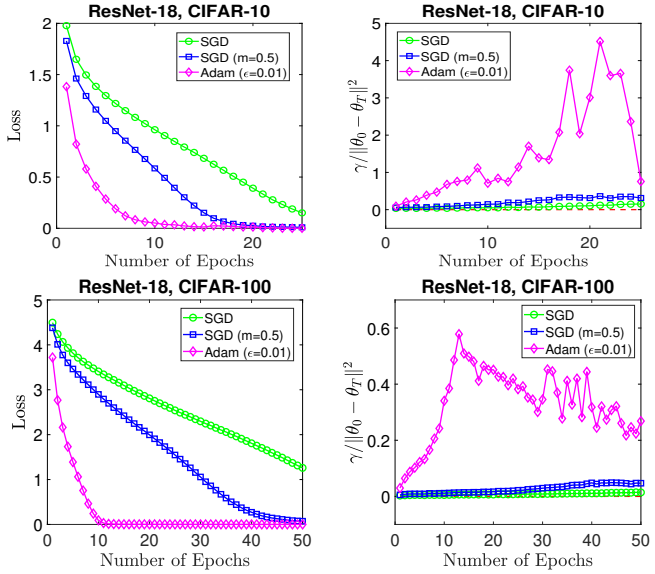

Fig. 7. Training ResNet with different optimizers on CIFAR-10/100.

Experiment setup: Next, we train a U-Net on the CIFAR-10 and CIFAR-100 datasets using SGD, SGD with momentum and Adam, respectively, and use the MSE loss. We apply a fixed initialization, a constant learning rate $\eta = 0.001$ and a mini-batch size 128 to all the optimizers. We set the momentum to be 0.5 for SGD with momentum and $\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-2}$ for the Adam.

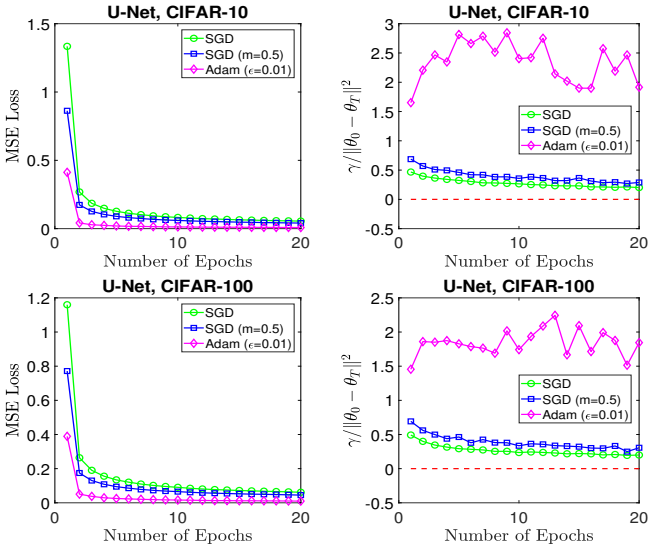

Fig. 8. Training U-Net with different optimizers.

Fig. 8 shows the training results, where one can make the same conclusions.

V. Conclusions

In this paper, we propose a regularity principle for general stochastic algorithms. The regularity principle guarantees a sub-linear convergence rate that scales inverse proportionally to γ. Through extensive DNN training experiments, we show that practical DNN training trajectories obey the regularity principle reasonably well, and the regularization parameter γ provides a metric that quantifies the effect of different training techniques on DNN optimization. In the future work, we expect that such an optimization-level regularity principle can be exploited to develop improved training techniques for deep learning.

References

[1] S. Arora, N. Cohen, N. Golowich, and W. Hu. A convergence analysis of gradient descent for deep linear neural networks. In *Proc. International Conference on Learning Representations (ICLR)*, 2019.

[2] N. Bjorck, C. P. Gomes, B. Selman, and K. Q. Weinberger. Understanding batch normalization. In *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, pages 7694–7705, 2018.

[3] X. Chen, S. Liu, R. Sun, and M. Hong. On the convergence of a class of Adam-type algorithms for non-convex optimization. In *Proc. International Conference on Learning Representations (ICLR)*, 2019.

[4] D. Clevert, T. Unterthiner, and S. Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). In *Proc. International Conference on Learning Representations (ICLR)*, 2015.

[5] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2(4):303–314, Dec 1989.

[6] S. Du, J. Lee, H. Li, L. Wang, and X. Zhai. Gradient descent finds global minima of deep neural networks. In *Proc. International Conference on Machine Learning (ICML)*, volume 97, pages 1675–1685, Jun 2019.

[7] J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12:2121–2159, July 2011.

[8] S. Ghadimi and G. Lan. Accelerated gradient methods for nonconvex nonlinear and stochastic programming. *Mathematical Programming*, 156(1):59–99, Mar 2016.

[9] X. Glorot, A. Bordes, and Y. Bengio. Deep sparse rectifier neural networks. In *Proc. International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 315–323, 11–13 Apr 2011.
[10] M. Hardt and T. Ma. Identity matters in deep learning. *arXiv:1611.04231*, 2016.
[11] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2015.
[12] G. Huang, Z. Liu, and K. Weinberger. Densely connected convolutional networks. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2261–2269, 2016.
[13] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proc. International Conference on Machine Learning (ICML)*, pages 448–456, 2015.
[14] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *Proc. International Conference on Learning Representations (ICLR)*, 2015.
[15] A. Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.
[16] X. Li, S. Ling, T. Strohmer, and K. Wei. Rapid, robust, and reliable blind deconvolution via nonconvex optimization. *Applied and Computational Harmonic Analysis*, 2018.
[17] Y. Li and Y. Yuan. Convergence analysis of two-layer neural networks with relu activation. In *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, pages 597–607. 2017.
[18] A. Maas, Y. Hannun, and A. Ng. Rectifier nonlinearities improve neural network acoustic models. In *Proc. ICML Workshop on Deep Learning for Audio, Speech and Language Processing*, 2013.
[19] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proc. International Conference on Machine Learning (ICML)*, pages 807–814, 2010.
[20] Y. Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course*. Springer, 2014.
[21] E. Orhan and X. Pitkow. Skip connections eliminate singularities. In *Proc. International Conference on Learning Representations (ICLR)*, 2018.
[22] N. Qian. On the momentum term in gradient descent learning algorithms. *Neural Networks*, 12(1):145–151, Jan. 1999.
[23] S. Reddi, S. Kale, and S. Kumar. On the convergence of Adam and beyond. In *Proc. International Conference on Learning Representations (ICLR)*, 2018.
[24] H. Robbins and S. Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, 22(3):400–407, 09 1951.
[25] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, editors, *Proc. Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, pages 234–241, 2015.
[26] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning Representations by Back-propagating Errors. *Nature*, 323(6088):533–536, 1986.
[27] S. Santurkar, D. Tsipras, A. Ilyas, and A. Madry. How does batch normalization help optimization? In *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, 2018.
[28] R. Srivastava, K. Greff, and J. Schmidhuber. Highway networks. *ArXiv 1505.00387*, 2015.
[29] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.
[30] H. Zhang, Y. Zhou, Y. Liang, and Y. Chi. A nonconvex approach for phase retrieval: reshaped Wirtinger flow and incremental algorithms. *Journal of Machine Learning Research (JMLR)*, 18(141):1–35, 2017.
[31] X. Zhang, Y. Yu, L. Wang, and Q. Gu. Learning one-hidden-layer relu networks via gradient descent. In *Proc. International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 89, pages 1524–1534, 16–18 Apr 2019.
[32] K. Zhong, Z. Song, P. Jain, P. L. Bartlett, and I. S. Dhillon. Recovery guarantees for one-hidden-layer neural networks. In *Proc. 34th International Conference on Machine Learning (ICML)*, volume 70, pages 4140–4149, Aug 2017.
[33] Y. Zhou and Y. Liang. Characterization of gradient dominance and regularity conditions for neural networks. *ArXiv:1710.06910v2*, Oct 2017.
[34] Y. Zhou, J. Yang, H. Zhang, Y. Liang, and V. Tarokh. SGD converges to global minimum in deep learning via star-convex path. In *Proc. International Conference on Learning Representations(ICLR)*, 2019.
[35] Y. Zhou, H. Zhang, and Y. Liang. Geometrical properties and accelerated gradient solvers of non-convex phase retrieval. *In Proc. 54th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 331–335, 2016.
[36] D. Zou, Y. Cao, D. Zhou, and Q. Gu. Stochastic gradient descent optimizes over-parameterized deep relu networks. *arXiv:1811.08888*, 2018.

APPENDIX A
PROOF OF THEOREM 1

Note that the SA update rule implies that

$$\begin{aligned}\|\theta_{k+1}-\theta_T\|^2 &= \|\theta_k-\eta U(\theta_k;z_{\xi_k})-\theta_T\|^2\\ &= \|\theta_k-\theta_T\|^2+\eta^2\|U(\theta_k;z_{\xi_k})\|^2\\ &\quad -2\eta\langle\theta_k-\theta_T,U(\theta_k;z_{\xi_k})\rangle\\ &\le \|\theta_k-\theta_T\|^2+\eta^2\|U(\theta_k;z_{\xi_k})\|^2\\ &\quad -\eta^2\|U(\theta_k;z_{\xi_k})\|^2\\ &\quad -2\eta\gamma\big(\ell(\theta_k;z_{\xi_k})-\inf_\theta \ell(\theta;z_{\xi_k})\big), \end{aligned}\tag{4}$$

where the last inequality follows from the regularity principle in Definition 2. Note that the SA algorithm adopts the random sampling with reshuffle scheme. Summing (4) over the B-th epoch (i.e., $k=nB,nB+1,...,n(B+1)-1$) gives that

$$\begin{aligned}\|\theta_{n(B+1)}-\theta_T\|^2 \le \|\theta_{nB}-\theta_T\|^2\\ -2\eta\gamma\sum_{k=nB}^{n(B+1)-1}\big(\ell(\theta_k;z_{\xi_k})-\inf_\theta\ell(\theta;z_{\xi_k})\big).\end{aligned}$$

Further telescoping over the epoch index, we obtain that for all B

$$\begin{aligned}\|\theta_{nB}-\theta_T\|^2 \le \|\theta_0-\theta_T\|^2\\ -2\eta\gamma\sum_{P=0}^{B-1}\sum_{k=nP}^{n(P+1)-1}\big(\ell(\theta_k;z_{\xi_k})-\inf_\theta\ell(\theta;z_{\xi_k})\big).\end{aligned}\tag{5}$$

Therefore, for all $T=nB$, the above inequality further implies that

$$\begin{aligned}0 &\le \|\theta_0-\theta_T\|^2-2\eta\gamma\sum_{k=0}^{T-1}\big(\ell(\theta_k;z_{\xi_k})-\inf_\theta\ell(\theta;z_{\xi_k})\big)\\ &= \|\theta_0-\theta_T\|^2-2\eta\gamma\Big(\sum_{k=0}^{T-1}\ell(\theta_k;z_{\xi_k})-T\inf_\theta f(\theta;\mathcal{Z})\Big),\end{aligned}\tag{6}$$

where the last equality follows from the over-parameterization of the model. Rearranging the above inequality yields that

$$\frac{1}{T}\sum_{k=0}^{T-1}\ell(\theta_k;z_{\xi_k})-\inf_\theta f(\theta;\mathcal{Z})\le\frac{\|\theta_0-\theta_T\|^2}{2\eta\gamma T}.$$