NAIST-IS-DD1761016

Doctoral Dissertation

Analysis of Regularization and Optimization for Deep Learning

Yasutaka Furusho

March 13, 2020

Graduate School of Information Science
Nara Institute of Science and Technology

A Doctoral Dissertation
submitted to Graduate School of Information Science,
Nara Institute of Science and Technology
in partial fulfillment of the requirements for the degree of
Doctor of ENGINEERING

Yasutaka Furusho

Analysis of Regularization and Optimization for Deep Learning[*]

Yasutaka Furusho

Abstract

Keys to the recent success of deep learning are regularization methods and optimization algorithms. For example, deep neural networks (DNNs) such as the MLP and ResNet overfit noise in the training set and training the DNNs takes a long time due to slow convergence of the gradient descent (GD) while deeper networks have higher expressive power. Various regularization methods prevent the DNNs from over-fitting noise in the training set and optimization algorithms such as the batch normalization (BN) and stochastic depth accelerate the convergence of the GD. However, their mechanisms are not fully understood. In this thesis, toward understanding the mechanisms, we provide theoretical novel insight into the above phenomena as follows. The weight decay, L1, and path regularizations discourage the MLP from learning high-frequency components of training data. When we assume that the high-frequency components come from noise in the training set, the regularization methods prevent the MLP from over-fitting the noise while learning a true target function behind the noisy data. It is necessary for the convergence of parameters of the ResNet into minima to set a learning rate exponentially small with respect to the depth. The stochastic depth relaxes the exponential term of this decrease and the BN relaxes the exponential decrease into the polynomial order. Moreover, our experiments confirmed that the BN and stochastic depth enabled the GD to use a large learning rate compared with that of the vanilla ResNet and accelerated their training.

Keywords:

Deep learning, ResNet, Batch normalization, Stochastic depth, Regularization

[*]Doctoral Dissertation, Graduate School of Information Science, Nara Institute of Science and Technology, March 13, 2020.

深層学習の正則化と最適化の解析*

古庄 泰隆

内容梗概

深層学習は多くの分野に適用されその優れた性能が報告されている．近年の深層学習の目覚ましい成果は正則化と最適化アルゴリズムに基づいている．例えば多層パーセプトロン (MLP) や ResNet などの深いニューラルネットワーク (DNN) は高い表現能力を持つ一方，データセットに含まれるノイズに過剰適合したり学習速度が遅いという問題がある．多様な正則化手法はこれらの DNN が訓練データに含まれるノイズに過剰適合するのを防ぎ，バッチ正規化 (BN) や stochastic depth などの最適化手法は DNN の学習を加速させる．しかしながら，これらの手法が上記のように性能を改善する理由は理論的には十分に明らかでなかった．そこで我々はこの改善理由の理解に向けて，上記の正則化手法と最適化手法に関する以下の理論的な結果を示した．まず荷重減衰や L1 正則化，パス正則化は MLP がデータの高周波成分を学習するのを防ぐ．高周波成分の大部分がノイズで構成されていると仮定すると，この結果は正則化により MLP がノイズを学習するのを防ぐことを示唆している．また ResNet が勾配降下法により極小解へ収束する必要条件としてその学習率を層の数に対し指数的に小さく設定する必要がある．そして stochastic depth はこの減少の指数部を小さくし,BN は指数的減少を多項式的減少へ緩和する．更に数値実験でも BN と stochastic depth により大きな学習率が使え ResNet の学習を加速することを確認した．

キーワード

深層学習, ResNet, バッチ正規化, Stochastic depth, 正則化

*奈良先端科学技術大学院大学 情報科学研究科 博士論文, 2020 年 3 月 13 日.

Contents

List of Figures

1. Introduction

1.1 Machine learning and deep learning

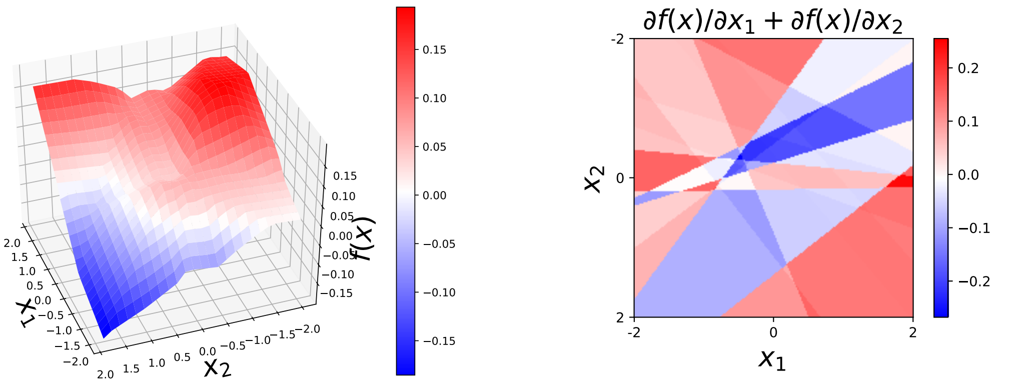
Machine learning technologies are widely used in modern society from search engine on the internet [1, 2] to handwritten zip code recognition system in the post office [3, 4] and automate various manual works [5]. This spread of machine learning is owing to its wide applicability and super-human ability.

Before the machine learning age, we have to write a program and tell a computer an algorithm, a set of exact and unambiguous instructions, for what we automate. It is often said that you don't really understand something until you can express it as an algorithm. However, we don't completely understand almost all the things we do in everyday life because we unconsciously do these things. For example, we can recognize individual persons but cannot exactly explain how to recognize them because our recognition ability is implicit knowledge: it is surjective, grows through our individual experience, and thus it is difficult to explain explicitly. On the other hand, machine learning feeds data, face images and the labels that correspond to the individuals, and automatically writes this recognition algorithm such that it maps the face images into the corresponding labels in the data. In this sense, machine learning is algorithm that creates an algorithm based on data. Of course, machine learning can be applied to other tasks such as object recognition [6, 7], language translation [8, 9, 10], and sport analysis [11, 12] because all it needs is only data, inputs and the correspond targets. This wide applicability is one reason why machine learning technologies are around us. The other reason is its super-human performance achieved by one machine learning algorithm called deep learning.

Although conventional machine learning can be applied to various applications, it has limitations in their performance. Constructing a machine learning system with a better performance required careful engineering and expert domain knowledge to design a good feature extractor, a transformation of raw input, for predicting the target. Deep learning automatically creates a good feature extractor that captures complex patters in the data by composing a computational model with stacking many layers that each transforms the features captured by a previous layer [13, 14, 15, 16], which is called deep neural network (DNN), and learning this model based on the given data, which is called training set. During the training, the DNN predicts the corresponding targets for given inputs and an optimization algorithm optimizes its parameters to reduce an empirical risk, which measures a discrepancy between the predictions and true targets in the training set. The most famous optimization algorithm for DNN is the gradient descent (GD). Landscape of the empirical risk on the parameters is like a hill. The GD calculates the gradient of this hill at current parameters, which looks at the steepest descent direction, and slightly moves the parameters into this direction. Iterating this procedure makes the parameters converge into a point that achieves a small empirical risk. Deep learning can be applied to complex tasks that conventional machine learning didn't make a good job [17, 18, 19] and has been changing the history of machine learning in performance [20, 21, 22].

1.2 How fast deep learning methods are developing!

Deeper neural networks have higher expressive power owing to their depth. Consider a DNN with the rectified linear unit (ReLU) activation functions, which represents a piecewise linear function (Fig. 1). Its expressive ability can be measured by the number of its linear regions and it increases exponentially with respect to the depth of the DNN [23, 24, 25, 26]. Although DNNs have massive expressive power, it was thought that their training was difficult and this diffi-



(a) Output of the DNN (b) Gradient of the output

Figure 1. Landscape of a 2-layer DNN with the ReLU activation $f : \mathbb{R}^2 \to \mathbb{R}$.

culty closed the neural network boom at the end of the twenty century. At the beginning of the twenty-first century, thanks to massive computational resources, it was empirically observed that many heuristics can overcome this difficulty. And now, hundreds of heuristics: regularization methods and optimization algorithms for training extremely DNNs are developing day by day as follows.

1.2.1 Regularization methods

The extremely DNNs tend to over-fit noise in the training set (Fig. 2) although their predictions should be generalized to unseen data, in other words, the DNNs should minimize a discrepancy between their predictions and true targets for unseen inputs, which is called expected risk. This is because the extremely DNNs have too much expressive power such that they can perfectly fit even input–target relation of randomly generated data [28]. Various regularization techniques prevented the DNNs from over-fitting noise in the training set while learning a true target function behind the noisy data (Fig. 2) [29]. These regularization methods modify optimization algorithm explicitly such as the dropout [30] or

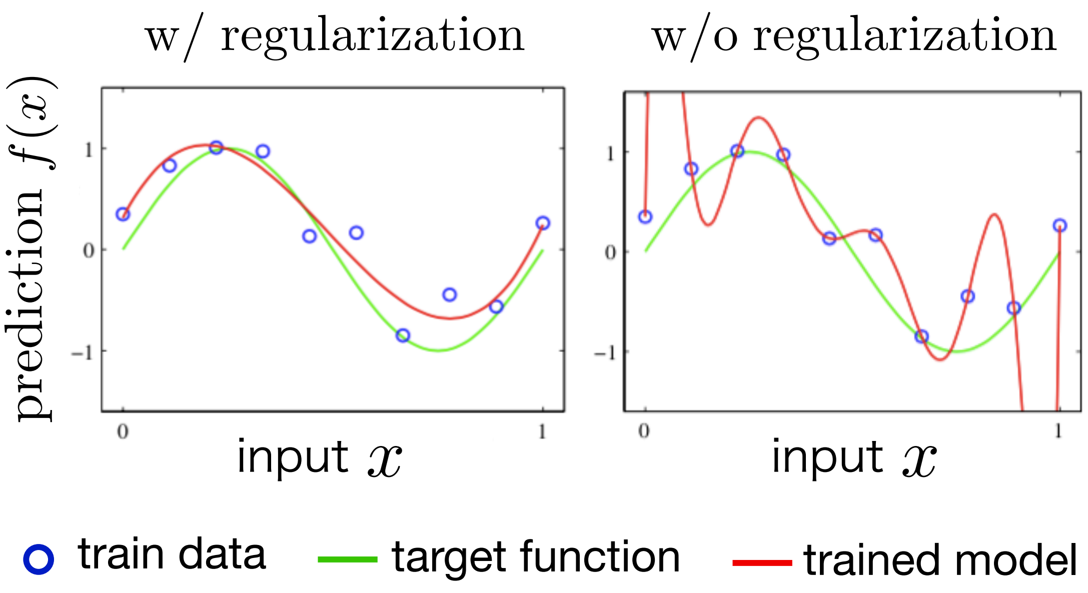


Figure 2. Over-fitting of the DNN $f : \mathbb{R} \to \mathbb{R}$ and effects of the regularization technique (Modification of Fig. 1.4 in [27]).

add the regularization term to the loss function such as the weight decay.

Effects of these regularization methods on the generalization ability of DNNs were theoretically analyzed by evaluating the model complexity measure such as the Rademacher complexity [31, 32, 33, 34]. However, a recent experiment showed that this measure is trivial and fail to explain their effects on the generalization ability [28].

This has prompted recent works on implicit regularization, which is the inherent mechanism in the optimization algorithm and the model architecture for high generalization ability [35]. For example, the SGD learns simple patterns [29] and low-frequency signals first [36]. Moreover, the SGD with a small mini-batch size converges the parameters into the flat minima [37], which have high generalization ability [38, 39, 40]. In the ResNet, skipping two layers by its shortcut makes the loss landscape around the minima flatter than skipping one or no layer [41]. If the training data are linearly separable, the GD converges a linear feedforward network into a maximum margin classifier [42, 43]. In the case of a linear convolutional neural network, the GD converges this linear model into a sparse classifier in the frequency domain [43]. In the matrix factorization, adding depth to the linear model provides a low-rank solution [44].

However, the mechanism of the explicit regularization methods such as the weight decay have been still unclear.

1.2.2 Optimization algorithms

Training deep conventional feedforward neural networks like the multi-layer perceptron (MLP) (Fig. 3) degraded even an empirical risk compared with shallow one while deeper networks have higher expressive power [23, 24, 25, 26]. For example, this network with 56 layers had a larger empirical risk than one with 20 layers [45]. This is because some layers lose important information for predicting the target. Some numerical experiments showed that the transformation by lay-

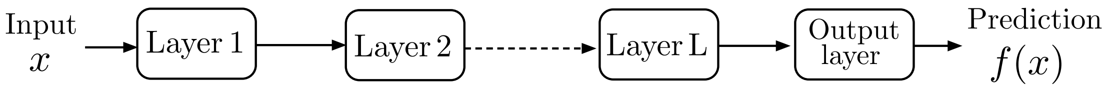


Figure 3. Multi-layer perceptron (MLP).

ers of the MLP decreased mutual information between transformed inputs and the targets [46, 47]. Moreover, the transformation decreases the angle between two data points, which makes the classification of data points difficult [48, 49].

To overcome this problem, the ResNet incorporates skip-connections between layers [45, 50] (Fig. 4), which add an input of a layer into its output, and propagate input information even at higher layers. These skip-connections enabled an extremely DNN (1202 layers) to be trained with a small empirical risk. However, training a deep ResNet takes a long time due to its slow convergence. For example, even a modern computer with multiple GPUs requires several weeks to train a 152-layer ResNet to learn the ImageNet dataset [45].

One reason for this slow convergence was said to be the internal covariate shift: an optimization problem for each layer is changing during training because a probability distribution of inputs of each layer is changing by updating parameters of its previous layer. The batch normalization (BN) was proposed to relax this internal covariate shift by normalizing inputs of each layer and accelerated the convergence of the GD with a large learning rate [51]. However, a recent experiment showed that the internal covariate shift doesn't occur [52].

Another reason for the slow convergence is the vanishing gradient problem: norms of the gradients with respect to parameters vanish in early layers [53, 54, 55]. The stochastic depth trains a shallow ResNet by dropping layers at random from the full ResNet for each iteration of training (Fig. 5) and relaxes this vanishing gradient problem [56]. It reduces the training time by 25 percent with

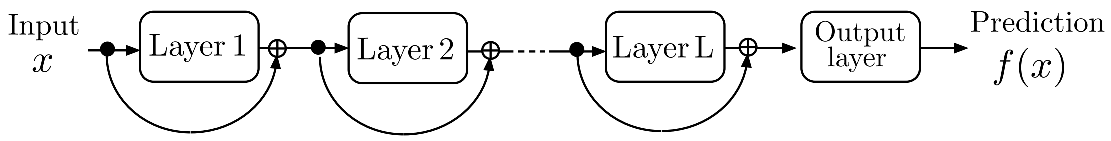


Figure 4. ResNet ($\oplus$ depicts element-wise vector addition).

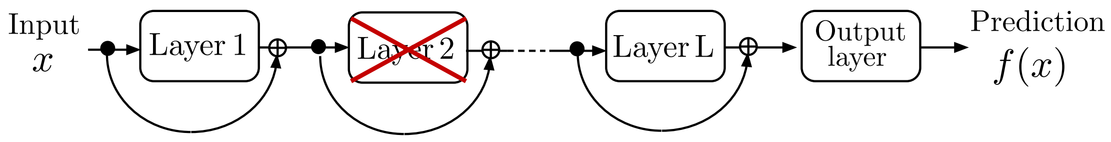


Figure 5. Stochastic depth.

higher performance at the same number of iterations. However, the mechanism of this acceleration is not fully understood.

Convergence property of the GD depends on the smoothness of the loss landscape of the DNN. The loss landscape is not only convex but also tends to have a large number of sharp kinks, which makes the GD unstable [57]. For example, a large step in updating the parameters by the GD tends to degrade its loss experimentally due to the sharp kink. This forces us to carefully set a small learning rate, which makes the convergence slow [58]. Theoretically, a small learning rate is necessary for the convergence when the maximum eigenvalue of the Hessian matrix is large at minima and a large condition number (ratio of the maximum eigenvalue to the minimum eigenvalue) makes the convergence slow around the minima [59]. Note that the relationship between the condition number and the convergence speed holds only around the minima θ_* but the GD and SGD make the parameters converge into the minima under some conditions [60, 61, 62]. Karakida et al. [63, 64] theoretically calculated the eigenvalues of the Fisher information matrix (FIM) of the randomly initialized MLP and MLP with BN because these eigenvalues approximate the eigenvalues of the Hessian matrix at the minima [60, 63]. Their results showed that the BN decreases the maximum eigenvalue and is helpful to set a large learning rate of the GD compared with that of the vanilla MLP for its convergence. However, the counterparts of the ResNet, ResNet with BN, and ResNet with stochastic depth haven't been analyzed yet.

1.3 Our contribution

Although some regularization methods prevent DNNs from over-fitting noise in the training set and the optimization algorithms accelerate the convergence of the GD, their mechanisms are not fully understood. Toward understanding these mechanisms, we aim to provide some novel theoretical insights into the prevention of the over-fitting by the regularization methods and the fast convergence of the GD by the BN and stochastic depth in this thesis.

First, toward understanding the prevention of the over-fitting by the weight decay, L1 regularization, and path regularization, we rethink these regularization methods from the perspective of the Fourier spectra and show that they penalize the MLP to discourage it from learning high-frequency components of data. When we assume that the high-frequency components come from noise in the training set, this result implies that these regularization methods prevent the DNN from over-fitting the noise while learning a true target function behind the noisy data.

Second, toward understanding mechanisms for the fast convergence of the GD by the BN and stochastic depth, we calculated the maximum eigenvalues of the FIMs of the randomly initialized ResNet, ResNet with BN, and ResNet with stochastic depth. Our results show that the maximum eigenvalue of the ResNet grows exponentially with respect to the depth, the stochastic depth relaxes this growth by decreasing exponential term, and the BN relaxes this exponential growth into at most the polynomial order. Thanks to this relaxation of the maximum eigenvalue, the BN and stochastic depth are helpful to set a large learning rate of the GD compared with that of the vanilla ResNet for its convergence. Moreover, our experiments confirmed the acceleration of training by the BN and stochastic depth with their optimal learning rates of the GD.

2. Problem formulation

2.1 Samples for training

Let a training set be denoted by $S = \{(x(n), y(n))\}_{n=1}^{N}$. Each training example is a pair of an input $x(n) \in \mathcal{X}$ and the corresponding target $y(n) \in \mathcal{Y}$, which is independently identically distributed from a probability distribution $\mathcal{D}$, where $\mathcal{X}$ and $\mathcal{Y}$ are an input space and target space. The indices of the set are omitted if they are clear from the context.

2.2 Training for deep neural network

A DNN $f : \mathcal{X} \times \Theta \to \mathcal{Y}$ with parameters $\theta \in \Theta$ predicts a corresponding target $y \in \mathcal{Y}$ for a given input $x \in \mathcal{X}$, where Θ is a parameter space. Its performance is measured by an expected risk

$$R(\theta) = \mathbb{E}_{x,y}\left[\ell(f(x, \theta), y)\right], \tag{1}$$

where $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+ \cup \{0\}$ is a loss function and we consider the squared loss $\ell(f(x, \theta), y) = \frac{1}{2}(f(x, \theta) - y)^2$ unless otherwise noted. The expectation is taken over the probability distribution $\mathcal{D}$. The parameters θ are trained by the GD

$$\theta_{t+1} = \theta_t - \eta \nabla_\theta R_S(\theta_t) \tag{2}$$

to minimize an empirical risk

$$R_S(\theta) = \frac{1}{N} \sum_{n=1}^{N} \ell(f(x(n), \theta), y(n)) \tag{3}$$

instead of the expected risk because we cannot calculate it directly due to inaccessibility of the data distribution $\mathcal{D}$. Note θ_t is an output of the GD at tth update and η is a learning rate of the GD. A more popular gradient-based training algorithm is the stochastic GD (SGD), which randomly selects one or more examples (a mini-batch) from the training set and updates the parameters using the gradient of the loss with the mini-batch. And more practical gradient-based training algorithm is the Adam [65], which uses learning rates for different parameters of the DNN and adaptively changes these learning rates based on estimated second moment of the gradients. Initialization methods of the parameters θ_0 are specified in each subsequent analysis section.

2.3 Regularization methods

The L-layer MLP $f : (x, \theta) \to \hat{y}$ with parameters $\theta = (W^1, W^2, ..., W^L) \in \Theta$ predicts a corresponding target $y \in \mathbb{R}$ for an input $x \in \mathbb{R}^D$,

$$h^l = \phi(W^l h^{l-1}), \quad \hat{y} = W^L h^{L-1}, \tag{4}$$

where $h^0 = x$ is the input, $\phi(x)_i = \max\{0, x_i\}$ is the element-wise ReLU function, and the width of l-th hidden layer is denoted by D_l. Due to the ReLU function, the MLP represents piecewise linear function and the number of linear regions grows with respect to the depth, which implies that deeper MLP has higher expressive power [23, 24, 25, 26]. Because of this high expression ability, a deep MLP tends to over-fit noise in the training set.

To prevent the DNNs from the over-fitting, a regularizer $\psi : \Theta \to \mathbb{R}_+ \cup \{0\}$ is added to the empirical risk,

$$R_S(\theta) + C \cdot \psi(\theta), \tag{5}$$

where C is the regularization coefficient, and the gradient based training algorithm is applied to this objective. In the weight decay and L1 regularization, their regularizers are the sum over the L2 norms of weights and that over the L1 norms, respectively.

$$\psi_{L2}(\theta) = \sum_{l=1}^{L} \left\| W^l \right\|_2, \quad \psi_{L1}(\theta) = \sum_{l=1}^{L} \left\| W^l \right\|_1 \tag{6}$$

In the path regularization, its regularizer is the sum over the Lp norms of path weights from the input nodes into the output node [33]. We consider the L1 norm path regularizer.

$$\psi_{P1}(\theta) = \sum_{\alpha \in [D_0] \times [D_1] \times \cdots \times [D_L]} \left| \prod_{l=1}^{L} W^l_{\alpha_{l+1}, \alpha_l} \right| \tag{7}$$

The path regularizer involves an exponential number of terms. However, it can be computed efficiently by dynamic programming in a single forward step [66].

These regularization methods made the MLP achieve a small test loss as well as a small empirical risk empirically and their mechanisms were theoretically analyzed by using the Rademacher complexity [33, 34, 67]. Let

$$\mathcal{H} \circ S = \{(f(x(1), \theta), f(x(2), \theta), ..., f(x(N), \theta)) \mid \theta \in \Theta\} \tag{8}$$

be the set of all possible evaluation of the MLP on the training set S. Then, its Rademacher complexity is defined as

$$\mathcal{R}(\mathcal{H} \circ S) = \frac{1}{N}\mathbb{E}_{\sigma}\left[\sup_{\theta \in \Theta} \sum_{n=1}^{N} \sigma_n f(x(n), \theta)\right], \tag{9}$$

where $\sigma_n \in \{-1, +1\}$ is the uniformly random variable and the expectation is taken with respect to these random variables $\{\sigma_n\}_{n=1}^{N}$. It measures ability of the model to fit random binary labels and this measure is upper bounded.

Theorem 1. *(Theorem 1 in [34]). Suppose that a parameter space is restricted such that* $\|W^L\|_2 = 1$ *and* $\prod_{l=1}^{L-1} \|W^l\|_2 \leq K$. *Then, the following inequality holds.*

$$\mathcal{R}(\mathcal{H} \circ S) \leq \frac{1}{\sqrt{N}} 2^{L-1/2} K \max_{n \in [N]} \|x(n)\|_2. \tag{10}$$

The regularizers penalize norms of the weights, confine learning to a subset of parameters with small norms, and thus it decreases the Rademacher complexity. Thanks to this property, the regularizers decrease a gap between the expected risk and empirical risk because the gap is upper bounded by the Rademacher complexity.

Theorem 2. *(Modification of Theorem. 26.5 in [32]). Assume that the loss function is* α*-Lipschitz and absolute value of the loss is upper bounded* $|\ell(f(x, \theta), y)| \leq c$ *for any* $\theta \in \Theta$ *and* $(x, y) \in \mathcal{X} \times \mathcal{Y}$. *Then, the following inequality holds with probability of at least* $1 - \delta$ *for all* $\theta \in \Theta$.

$$R(\theta) - R_S(\theta) \leq 2\alpha \mathcal{R}(\mathcal{H} \circ S) + 4c\sqrt{\frac{2 \log(4/\delta)}{N}}. \tag{11}$$

However, a recent experiment showed that the MLP can fit the training set with random binary labels perfectly even when the weight decay was used [28]. This implies that the Rademacher complexity is trivial for evaluating the effect of the regularization methods on the generalization ability. Thus, we need to rethink the regularization methods from different perspectives.

2.4 Optimization algorithms

Training a deep MLP degraded even its empirical risk compared with shallow one while deeper MLP has higher expressive power [45]. To overcome this degradation problem, the ResNet $f : (x, \theta) \mapsto \hat{y}$ incorporates skip-connections between layers, which add an input of a layer into its output (Fig. 6).

$$h^l = W^l \phi\left(h^{l-1}\right) + h^{l-1}, \quad \hat{y} = 1^T h^L, \tag{12}$$

where $h^0 = u^0 = W^0 x \in \mathbb{R}^D$ is the linear transform of the input. Although an extremely deep ResNet can be trained with high performance, it takes a long time due to its slow convergence [45]. To accelerate this training, the BN [51] and stochastic depth [56] were proposed.

The BN normalizes inputs of each layer (Fig. 6).

$$h^l = W^l \phi(\mathrm{BN}(h^{l-1})) + h^{l-1} \quad \mathrm{BN}\left(h^l\right)_i = \frac{h_i^l - \mathbb{E}_x\left[h_i^l\right]}{\sqrt{\mathrm{Var}\left(h_i^l\right)}}, \quad \hat{y} = 1^T h^L, \tag{13}$$

where the expectation is taken with respect to the input in the batch of the GD, in other words, the input in the training set.

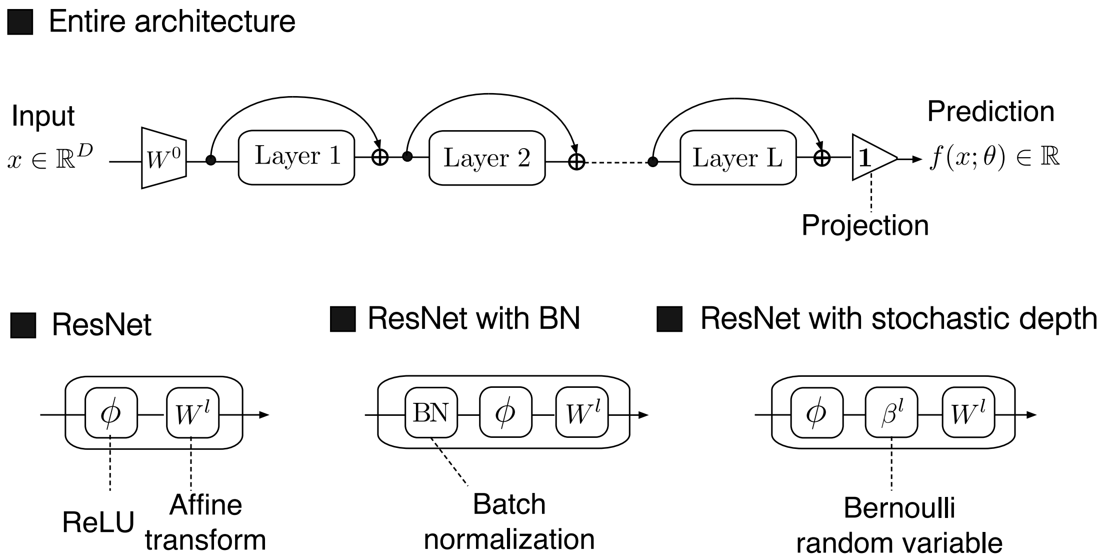


Figure 6. Architectures of the ResNet, ResNet with BN, and ResNet with stochastic depth.

The stochastic depth trains a shallow ResNet by dropping layers at random from full ResNet for each iteration of training (Fig. 6).

$$h^l = W^l \beta^l \cdot \phi\left(h^{l-1}\right) + h^{l-1} \text{ where } \beta^l \sim \mathrm{Ber}(p), \quad \hat{y} = 1^T h^L, \tag{14}$$

where p is the survival probability of each layer. Note the ResNet does not drop layers and instead multiplies the output of each layer by the survival probability for evaluation of its performance.

Convergence property of the GD depends on the smoothness of the loss landscape of the DNN. The empirical risk $R_S(\theta_t)$ is approximated by the second-order Taylor expansion around the minima θ_* such that third and subsequent-order terms can be ignored and hence the empirical risk on this region can be written as

$$R_S(\theta_t) \simeq R_S(\theta_*) + \frac{1}{2}(\theta_t - \theta_*)^T H(\theta_*)(\theta_t - \theta_*), \tag{15}$$

where $H(\theta_*) = \nabla_\theta \nabla_\theta R_S(\theta_*)$ is the Hessian matrix. The Hessian matrix is decomposed as $H(\theta_*) = U \Lambda U^T$, where U and Λ are a unitary square matrix with the eigenvectors and a diagonal matrix filled with the eigenvalues, which simplifies the empirical risk to

$$R_S(\theta_t) \simeq R_S(\theta_*) + \frac{1}{2} v_t^T \Lambda v_t, \tag{16}$$

where $v_t = U^T(\theta_t - \theta_*)$ and its update by the GD is

$$v_{t+1} = (I - \eta \Lambda)\, v_t. \tag{17}$$

Let $\lambda_{\min}$ and $\lambda_{\max}$ be the minimum and maximum eigenvalues of $H(\theta_*)$. It is necessary for the convergence of the GD into the minima to set a learning rate smaller than $2/\lambda_{\max}$ (Fig. 7). Moreover, it converges fastest when $\eta = 1/\lambda_{\max}$ around the minima and a small condition number $\lambda_{\max}/\lambda_{\min}$ induces fast convergence [59].

However, calculating the Hessian matrix and its eigenvalues are difficult owing to the complicated structure of the DNNs. Karakida et al. [63, 64] found that the FIM of probability distribution $p(x, y; \theta)$ represented by the MLP $f(x, \theta)$,

$$F(\theta) = \mathbb{E}_{x,y}\left[\nabla_\theta \log p(x, y; \theta)\ \nabla_\theta \log p(x, y; \theta)^T\right], \tag{18}$$

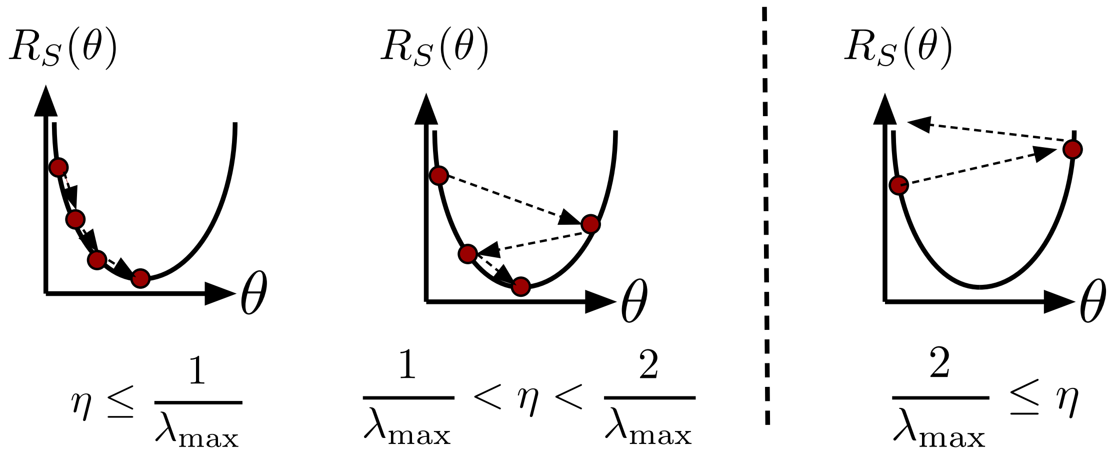

$R_S(\theta)$
$R_S(\theta)$
$R_S(\theta)$
θ
θ
θ
$\eta \leq \frac{1}{\lambda_{\max}}$
$\frac{1}{\lambda_{\max}} < \eta < \frac{2}{\lambda_{\max}}$
$\frac{2}{\lambda_{\max}} \leq \eta$

Figure 7. Convergence property of the GD and its learning rate.

approximates the Hessian matrix at the minima. Consider $p(x, y; \theta) = p(x)p(y|x; \theta)$, where $p(x)$ is the probability of the input in the training set and $p(y|x; \theta) = \mathcal{N}(f(x; \theta), 1)$. In this case, the FIM is rewritten as

$$F(\theta) = \mathbb{E}_x \left[\nabla_\theta f(x, \theta) \; \nabla_\theta f(x, \theta)^T \right] \tag{19}$$

and the following holds:

$$H(\theta) = F(\theta) - \mathbb{E}_x \left[(y - f(x, \theta) \cdot \nabla_\theta \nabla_\theta f(x, \theta) \right], \tag{20}$$

where the second term is negligible when the error is small. In addition, the eigenvalues of the FIM of a sufficiently wide MLP don't change during training.

Karakida et al. [63, 64] also calculated the maximum eigenvalue of the FIM of the randomly initialized MLP and MLP with BN. Their results showed that the BN decreases the maximum eigenvalue and is helpful to set a large learning rate of the GD compared with the vanilla MLP for its convergence under some condition. However, the counterpart of the ResNet, ResNet with BN, and ResNet with stochastic depth haven't been analyzed yet.

3. Analysis of regularization methods

3.1 Fourier spectra analysis

We rethink the regularization methods from the perspective of the Fourier spectra of the MLP. Let the MLP $f(\cdot, \theta)$ be denoted by $f_\theta(\cdot) : \mathbb{R}^D \to \mathbb{R}$. Then, the Fourier transform of the MLP is written as

$$f_\theta(x) = (2\pi)^{-D/2} \int \tilde{f}_\theta(k)\, e^{i\langle k,x\rangle} dk, \quad \text{where} \quad \tilde{f}_\theta(k) = \int f_\theta(x)\, e^{-i\langle k,x\rangle} dx. \tag{21}$$

Rahaman et al. [36] calculated the Fourier coefficients and their upper bounds as follows.

Theorem 3. *(Modification of Theorem 1 in [36].) The Fourier coefficients of the MLP f_θ are upper bounded by the Lipschitz constant of the MLP $\|f_\theta\|_{\text{Lip}}$ and the number of linear regions represented by the MLP $K(\theta)$.*

$$\tilde{f}_\theta(k) \le \sum_{d=1}^{D} \|k\|_2^{-(d+1)}\, O\Big(\|f_\theta\|_{\text{Lip}} \cdot K(\theta)\Big). \tag{22}$$

Theorem 3 shows the implicit regularization of the MLP: the Fourier coefficient of the MLP shrinks polynomially with respect to the frequency. Moreover, this theorem implies that the MLP needs a large Lipschitz constant to represent the high frequency components. We focused on this property and analyzed the relationship between the Lipschitz constant and the regularizers toward understanding the effects of the explicit regularization methods on the Fourier coefficients of the MLP.

3.2 Theoretical analysis

The Lipschitz constant of the MLP were theoretically analyzed by focusing on its layered structure [68] and Theorem 4 is application of this result.

Theorem 4. *The Lipschitz constant of the MLP is upper bounded by the product over the L2 norms of the weights and that over the L1 norms.*

$$\|f_\theta\|_{\text{Lip}} \le \prod_{l=1}^{L} \|W^l\|_2 \le \prod_{l=1}^{L} \|W^l\|_1 \tag{23}$$

In contrast, we derive Theorem 5 by focusing on its path-sum structure, that is, the output of the MLP can be written as the sum over path weights from input nodes to the output node.

Theorem 5. *The Lipschitz constant of the MLP is upper bounded by the path and L1 regularizers.*

$$\|f_\theta\|_{\mathrm{Lip}} \leq \psi_{P1}(\theta) \leq \psi_{L1}(\theta)^L \tag{24}$$

Remark 1. *Theorems 3-5 imply that the weight decay, L1 regularization, and path regularization decrease the Lipschitz constant and hence give a penalty for the MLP to learn high-frequency components. When we assume that the high frequency components come from noise in the training set, this result implies that these regularization methods prevent the MLP from over-fitting the noise while learning a target function behind the noisy data.*

3.3 Numerical experiments

3.3.1 Toy examples

Our theory shows that the regularizations prevent the MLP from learning high-frequency components. To confirm its validity, we made toy examples (Fig. 8). Two hundred inputs $x \in \mathbb{R}$ were sampled from $[0, 1]$ at equall intervals and the targets $y \in \mathbb{R}$ were generated by the target function f_0 and the noise function ϵ:

$$y = f_0(x) + \epsilon(x), \text{ where } f_0(x) = \sum_{i=1}^{3} \sin(2\pi k_i x + \varphi_i), \ \epsilon(x) = \sum_{i=4}^{6} 0.4 \sin(2\pi k_i x + \varphi_i), \tag{25}$$

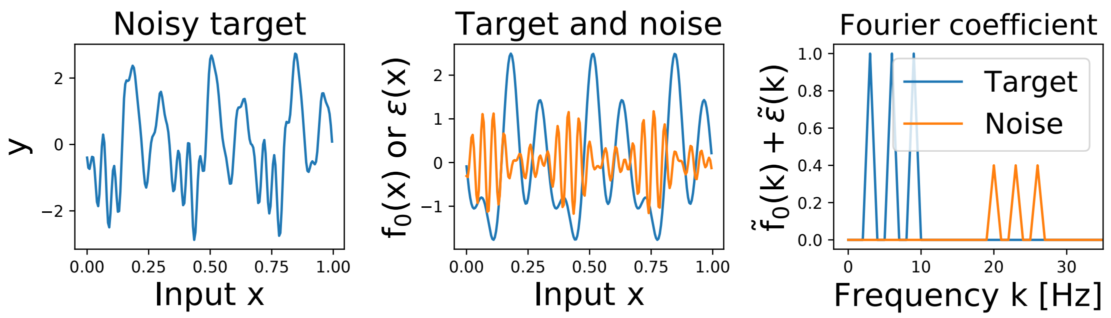
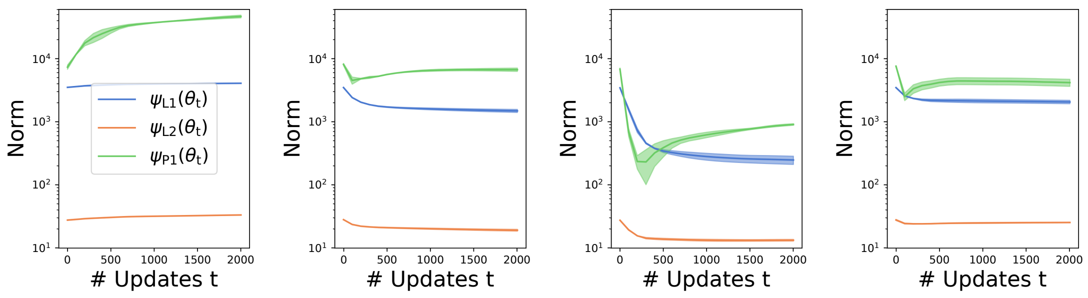


Figure 8. Inputs and targets in the toy example and its Fourier coefficients.

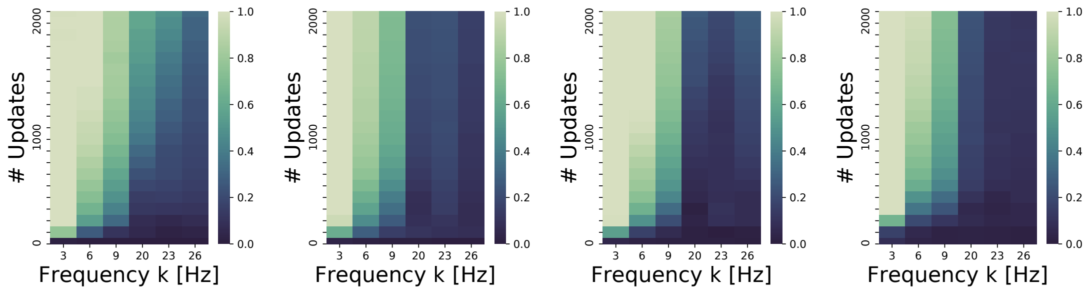
where $k = (3, 6, 9, 20, 23, 26)$ are the frequencies and $\varphi = (\varphi_1, \varphi_2, ..., \varphi_6) \in [0, 2\pi]^6$ are the phases, which are randomly sampled. We initialized 5-layer MLP with 100 hidden units by the He initialization [69] and trained this model by the Adam with a learning rate 0.001 to minimize the squared loss on this dataset. Regularization coefficients were $C_{L2} = 0.03$, $C_{L1} = 0.002$, and $C_{P1} = 0.00001$, respectively. Every 100 updates, the norms (Fig. 9) and the Fourier coefficients of the MLP (Fig. 10) were calculated. Note that the Fourier coefficients were calculated by applying the fast Fourier transform (FFT) to the pair of the input $x \in \mathbb{R}$ and the output of the MLP $f_\theta(x) \in \mathbb{R}$.

The norms of the MLP without the regularization grew (Fig. 9a) and the MLP learned the high-frequency noise components (Fig. 10a). On the other hand, the regularizers suppressed the norms (Figs. 9b-d) and prevented the MLP from learning the high-frequency noise (Figs. 10b-d). These results agreed with our analysis.



(a) w/o regularization (b) Weight decay (c) L1 regularization (d) Path regularization

Figure 9. Norms (solid line: average, shaded area: within one s.d. over 3 trials).



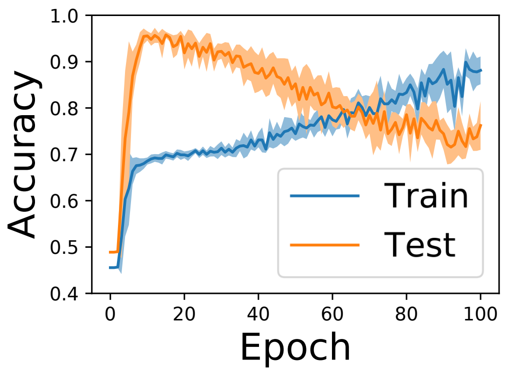
(a) w/o regularization (b) Weight decay (c) L1 regularization (d) Path regularization

Figure 10. Fourier coefficients of the MLP (color map: average over 3 trials).

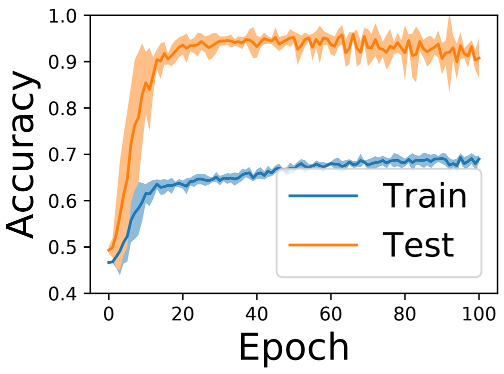
3.3.2 Handwritten digit dataset

In real-life applications, similar inputs have similar targets, that is, the true target function is smooth. Therefore, our theory implies that the regularizations prevent the MLP from over-fitting the noise while learning the unknown target function behind noisy data. To confirm this, we carried out the following experiment.

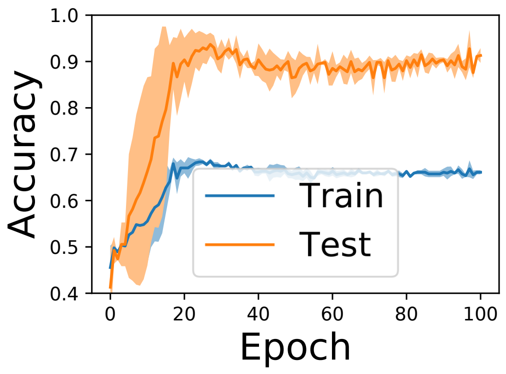
A training set and a test set were made by sampling from the handwritten digit dataset [70] with the target of 0 or 1. We chose this small dataset for easy evaluation of the Fourier coefficients of the MLP as we explain later. In the training set, 65% of the targets were randomly replaced with 0 or 1. We stacked the logistic sigmoid function $\sigma(\hat{y}) = 1/(1 + \exp(-\hat{y}))$ on a 4-layer MLP with 20



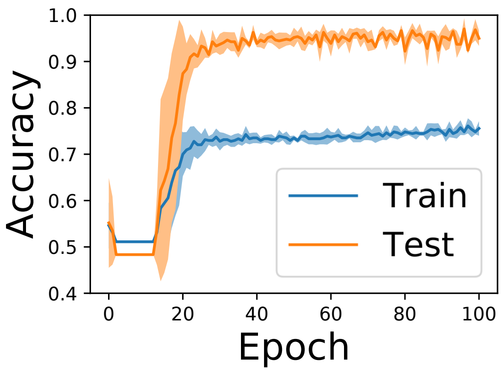
(a) Without regularization



(b) Weight decay



(c) L1 regularization



(d) Path regularization

Figure 11. Accuracy of the MLP on the training set (partially random label) and test set (true label). (solid line: average, shaded area: within one s.d. over 3 trials).

hidden units and trained this model by the SGD with a mini-batch size 32 and a learning rate 0.001 to minimize the binary cross entropy loss on the training set. Regularization coefficients were $C_{L2} = 0.1$, $C_{L1} = 0.01$, and $C_{P1} = 0.0003$, respectively. The training accuracy and the test accuracy were calculated at each epoch (Fig. 11). The Fourier coefficients of the MLP were calculated after training and these coefficients are normalized such that $\sqrt{\sum_k \widetilde{\sigma \circ f_{\theta_T}}(k)^2} = 1$ (Fig. 12). Because the FFT becomes prohibitively expensive for a large input dimension D, we trained the variational autoencoder (VAE) [71] with one dimensional latent space $z \in \mathbb{R}$, and the VAE generated 100 digit images $x(z)$ along with this latent space $z \in [-2, 2]$ (Fig. 13). Then, the Fourier coefficients of the MLP were calculated by applying the FFT to the pair of the latent variable $z \in \mathbb{R}$ and the

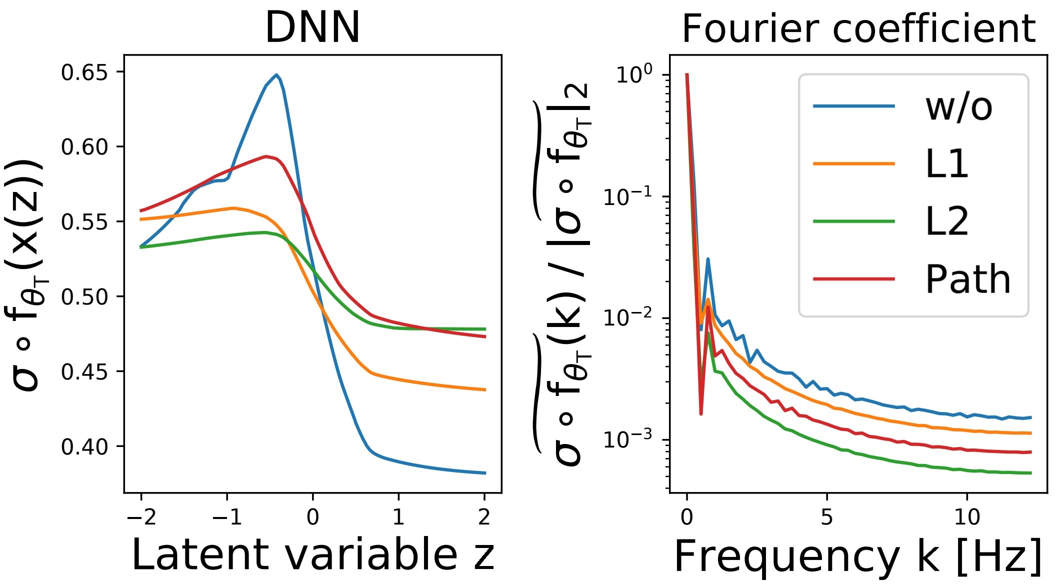


Figure 12. Trained MLP and its normalized Fourier coefficients.

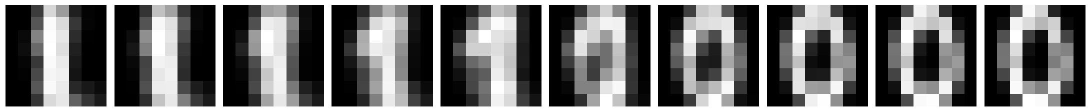


Figure 13. Images generated by the VAE along with the latent space $z \in [-2, 2]$.

output of the MLP $\sigma \circ f(x(z)) \in \mathbb{R}$ (Fig. 12).

The MLP without the regularization first learned the target function but over-fitted the noise after a few epochs (Fig. 11a). On the other hand, the regularizations prevented the MLP from over-fitting the noise while learning the target function (Figs. 11b-d). Fig. 12 also shows that the regularizations made the MLP ignore the spiky noise and learn the smooth target function.

3.4 Conclusion

The DNNs such as the MLP tend to over-fit noise in the training set because they have too much expressive power such that they can fit even input–target relation of randomly generated data. Various regularization techniques prevented a DNN from over-fitting the noise in the training set while learning the target function behind the noisy data.

Their generalization abilities were theoretically analyzed by evaluating the model complexity measures such as the Rademacher complexity. However, a recent experiment showed that this measure is trivial and fails to explain the effects of the regularization methods on the generalization ability. Thus, we need to rethink the regularization methods from different perspectives.

We rethink the regularization methods from the Fourier spectra perspective toward understanding the prevention of the over-fitting by the regularization. We applied the Fourier spectra analysis of the MLP into the weight decay, L1, and path regularizations and showed that they discourage the MLP from learning high-frequency components. When we assume that the high-frequency components come from noise in the training set, they prevent the MLP from over-fitting the noise while learning the target function behind the noisy data. Numerical experiments confirmed the validity of our analysis.

4. Analysis of optimization algorithms

4.1 Structure of expected Fisher information matrix

Convergence property of the GD is related to the maximum eigenvalue of the FIM of the initialized DNN as shown in Sec. 2.4. Under the same setting, we calculated the eigenvalues of FIMs of the ResNet, ResNet with BN, and ResNet with stochastic depth averaged over the random initialization of the parameters [55, 69] (expected FIMs),

$$W^0_{i,j} \sim \mathcal{N}\left(0, \frac{1}{D}\right), \quad W^l_{i,j} \sim \mathcal{N}\left(0, \frac{2}{D}\right) \quad \text{for all } l \in [L], \tag{26}$$

toward understanding effects of the BN and stochastic depth on the convergence property of the GD.

The maximum eigenvalue of the expected FIM is upper bounded by the sum over its diagonal elements and lower bounded by the the mean of its diagonal elements. Let $u^l_i = \phi(h^{l-1})_i$, $u^l_i = \phi(\mathrm{BN}(h^{l-1}))_i$, and $u^l_i = \beta^l \cdot \phi(h^{l-1})_i$ be forward signals of the ResNet, ResNet with BN, and ResNet with stochastic depth respectively for $l \in [L]$. Then, the bounds can be obtained by calculating the second moment of the forward signal u^l_i and that of the backward error signal $\delta^l_i = \frac{\partial f(x;\theta)}{\partial h^l_i}$ under Assumption 1 (Fig. 14).

Assumption 1. *The forward signal u^l_i is independent of the backward error signal δ^l_i, in particular,* $\mathbb{E}_{\theta,x}\left[{u^l_i}^2 \cdot {\delta^l_i}^2\right] = \mathbb{E}_{\theta,x}\left[{u^l_i}^2\right] \cdot \mathbb{E}_{\theta,x}\left[{\delta^l_i}^2\right]$.

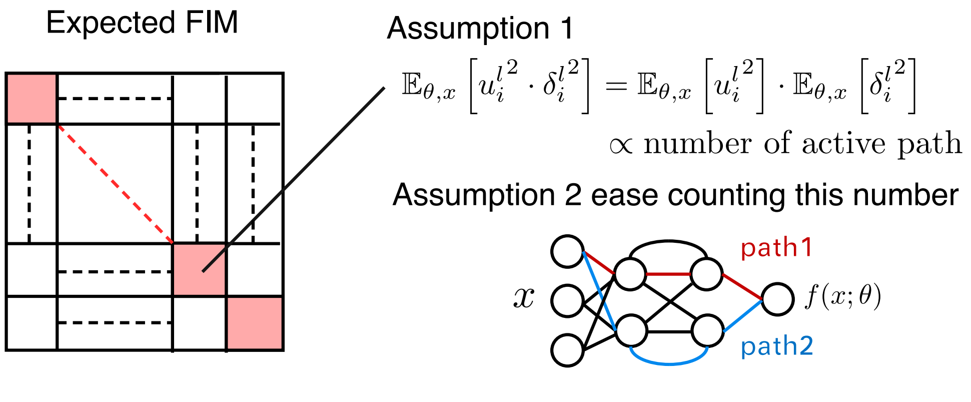

Expected FIM
Assumption 1
$\mathbb{E}_{\theta,x}\left[{u^l_i}^2 \cdot {\delta^l_i}^2\right] = \mathbb{E}_{\theta,x}\left[{u^l_i}^2\right] \cdot \mathbb{E}_{\theta,x}\left[{\delta^l_i}^2\right]$
$\propto$ number of active path
Assumption 2 ease counting this number
path1
x
$f(x;\theta)$
path2

Figure 14. Structure of the expected FIM and roles of Assumptions 1 and 2.

This approximation error $\mathbb{E}_{\theta,x}\left[u_i^{l^2}\cdot\delta_i^{l^2}\right]-\mathbb{E}_{\theta,x}\left[u_i^{l^2}\right]\cdot\mathbb{E}_{\theta,x}\left[\delta_i^{l^2}\right]$ can be written as the 3rd and 4th-order joint cumulants and small joint cumulants decrease the approximation error [72].

The second moments of the forward signal and the backward error signal can be calculated by counting the number of active path of the DNNs (Fig. 14). The active path is the path from an input unit to the output unit on which activations of all hidden units are active. Assumption 2 makes this counting easy.

Assumption 2. *Half of the hidden units are active* $\frac{1}{D}\sum_{i=1}^{D}\phi'(h_i^l)=\frac{1}{2}$ *per layer.*

The numerical experiments on the binary class PCA-whitened MNIST dataset (Fig. 15) confirmed that this assumption is almost satisfied.

4.2 Theoretical analysis

We calculated the maximum eigenvalues of the expected FIMs of the ResNet, ResNet with BN, and ResNet with stochastic depth under Assumptions 1 and 2. Without loss of generality, we suppose that inputs in the training set are normalized $\mathbb{E}_x[x_i]=0$ and $\mathrm{Var}(x_i)=1$.

Theorem 6. *(Theorem 4 in [73]). Under Assumptions 1 and 2, the maximum eigenvalue* $\lambda_{\max}$ *of the expected FIM of the ResNet grows exponentially with depth,*

$$m_\lambda=\frac{L+4}{4L+4}\cdot 2^L,\qquad m_\lambda\le\lambda_{\max}\le(L+1)D^2m_\lambda, \tag{27}$$

where m_λ *is the mean of the eigenvalues* $\{\lambda_i\}_{i=1}^{(L+1)D^2}$.

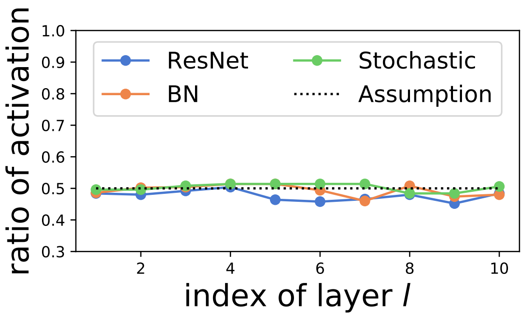


Figure 15. Activation rate of the hidden units in each layer.

Remark 2. *Setting a learning rate exponentially small with respect to the depth is necessary for the convergence of the GD.*

Theorem 7. *(Theorem 5 in [73]). Under Assumptions 1 and 2, the BN relaxes the exponential growth of the maximum eigenvalue to $L \log L$ order at most,*

$$m_\lambda = \frac{H_{L+1} + 1}{2}, \quad m_\lambda \leq \lambda_{\max} \leq (L+1) D^2 m_\lambda, \tag{28}$$

where $H_L = \sum_{k=1}^{L} \frac{1}{k}$ is the harmonic number.

Suppose that the stochastic depth drops $100 \cdot (1-p)$ percent of layers in ResNet rather than dropping each layer with probability $100 \cdot (1-p)$ for simplicity. An extremely deep ResNet satisfied this owing to the law of large numbers.

Theorem 8. *Under Assumptions 1 and 2, the stochastic depth relaxes the exponential growth of the eigenvalues of the ResNet by reducing the exponent,*

$$m_\lambda = \frac{pL + 4}{4L + 4} \cdot 2^{pL}, \quad m_\lambda \leq \lambda_{\max} \leq (L+1) D^2 m_\lambda. \tag{29}$$

Remark 3. *The BN and stochastic depth are helpful to set a large learning rate of the GD compared with that of the ResNet for its convergence.*

4.3 Numerical experiments

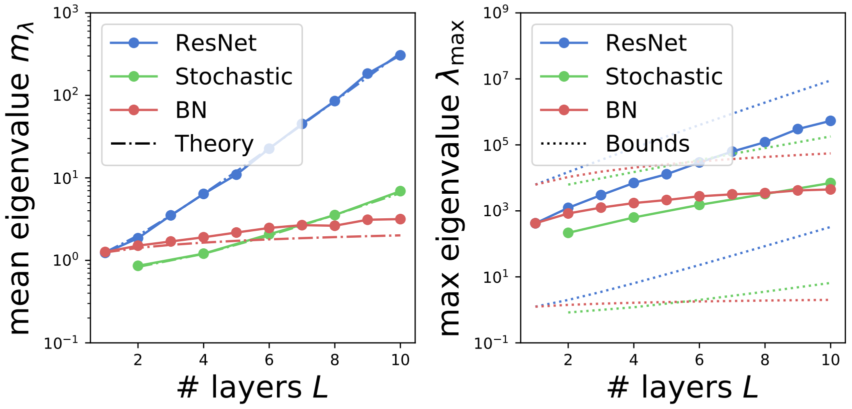
To confirm the validity of the above analysis, some numerical experiments were carried out.

4.3.1 Dataset

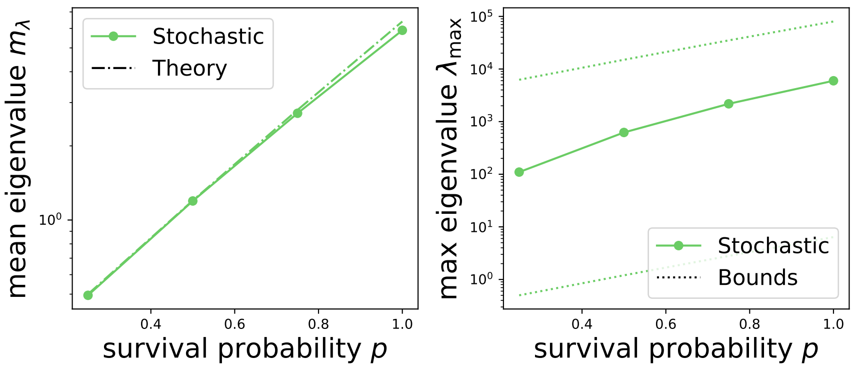
The dataset was a subset of the MNIST dataset with class label of 0 or 1 and its inputs were preprocessed by the PCA-whitening: its inputs were projected into D-dimensional subspace and normalized, that is, $\forall d \in [D], \mathbb{E}[x_d] = 0$ and $\mathrm{Var}(x_d) = 1$.

4.3.2 Eigenvalues of expected FIM

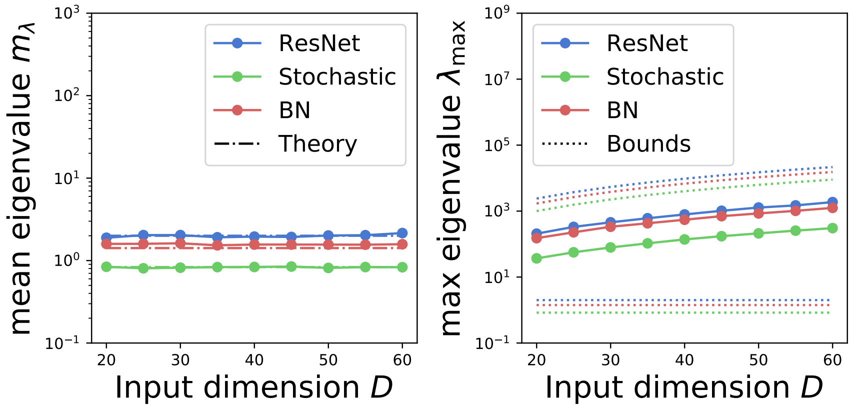
The mean eigenvalues and maximum eigenvalues of the expected FIMs of the ResNet, ResNet with BN, and ResNet with stochastic depth were calculated (Fig. 16). The mean eigenvalues agreed with our theoretical values and the maximum eigenvalues were between the theoretical upper and lower bounds.



(a) Dependence of depth L ($p = 0.5$ and $D = 50$)



(b) Dependence of survival probability p ($L = 4$ and $D = 50$)

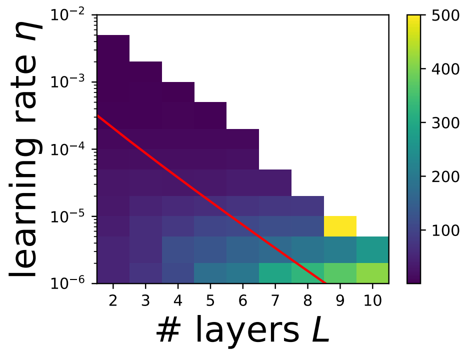


(c) Dependence of width D ($L = 2$ and $p = 0.5$)

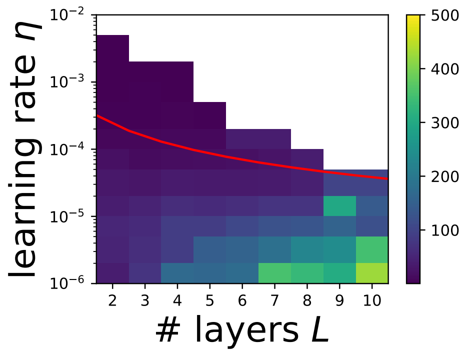
Figure 16. Eigenvalues of the expected FIMs.

4.3.3 Convergence property of GD

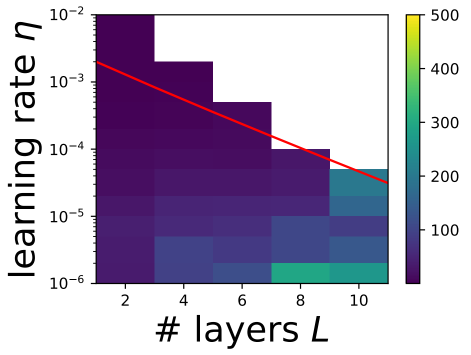
The convergence properties of the ResNet, ResNets with BN, and ResNet with stochastic depth were numerically examined. Each algorithm with various numbers of layers L, input dimension D and learning rates η updated the parameters 50 times for each run and the training loss was calculated by averaging over five runs (Figs. 17 and 18). The maximum learning rate for convergence in theory (red line), calculated as $2/(\text{upper bound of } \lambda_{\max})$, matched the boundary between the convergence (colored) and the divergence, i.e. the loss is larger than 1000 (white). In addition, the BN and stochastic depth enabled the GD to use a larger learning rate than that of the vanilla ResNet when the network is deep.



(a) ResNet

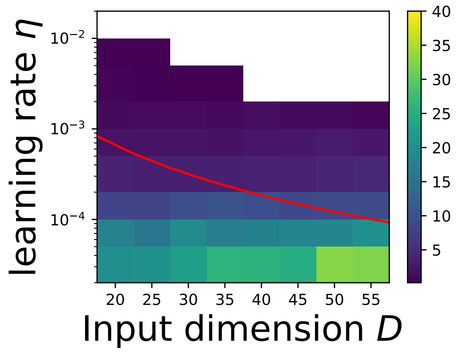


(b) ResNet with BN

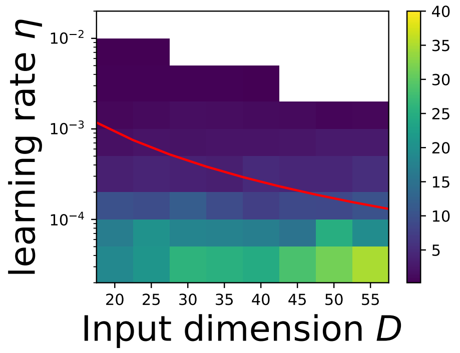


(c) ResNet with stochastic depth

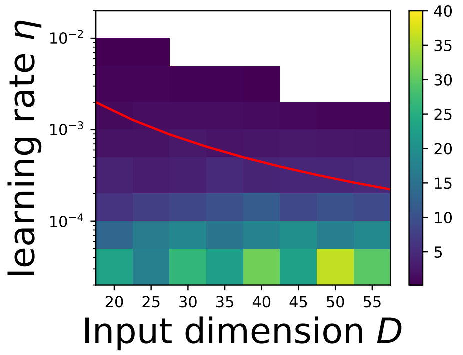
Figure 17. Convergence of the GD on the dataset with $D = 50$.



(a) ResNet



(b) ResNet with BN



(c) ResNet with stochastic depth

Figure 18. Convergence of the GD on the 2-layers DNNs.

4.3.4 Convergence speed

The training loss and test loss of the ResNet, ResNets with BN, and ResNet with stochastic depth at each update were evaluated over 10 runs (Fig. 19). Each algorithm used the optimal learning rate $\eta = 1/(\text{upper bound of } \lambda_{\max})$. These results showed that the BN and stochastic depth accelerates training.

4.4 Conclusion

Although an extremely deep ResNet can be trained with high performance, it takes a long time due to its slow convergence. The BN and stochastic depth empirically accelerated convergence of the GD.

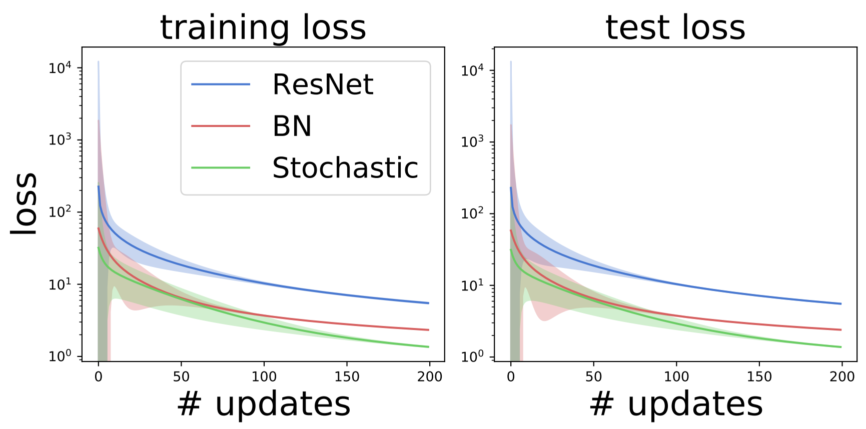


Figure 19. Loss of the 4-layer DNNs on the dataset with $D = 50$ (solid line: average, shadowed area: within one s.d.)

The convergence property of the GD is related to the maximum eigenvalue of the FIM of the randomly initialized DNN. In the case of the MLP, Karakida et al. [63, 64] calculated the eigenvalues of the randomly initialized MLP and MLP with BN and showed that the BN decrease the maximum eigenvalue under some condition. However, the counterpart of the ResNet, ResNet with BN, ResNet with stochastic depth haven't been analyzed yet.

We calculated the eigenvalues of the FIMs of these models and showed that the maximum eigenvalue of the ResNet grows exponentially with respect to the depth, the stochastic depth relaxes this growth by decreasing exponential term, and the BN relaxes this exponential growth into the polynomial order. Thanks to this relaxation of the maximum eigenvalue, the BN and stochastic depth are helpful to set a large learning rate compared with that of the vanilla ResNet for its convergence. Moreover, our experiments confirmed the acceleration of training by the BN and stochastic depth with their optimal learning rates of the GD.

5. Conclusion

Keys to the recent success of deep learning are their regularization methods and optimization algorithms for training extremely DNNs. Some regularization methods prevent the DNN from over-fitting noise in the training set while learning the target function behind the noisy data. The BN and stochastic depth accelerate the convergence of the GD.

Many studies tried to clear these mechanisms. The effects of regularization methods on the generalization ability of the DNNs were theoretically analyzed by evaluating the model complexity measure such as the Rademacher complexity. However, a recent experiment showed that this measure is trivial and fail to explain their generalization abilities. Thus, we need to rethink the regularization method from different perspectives. The fast convergence of the GD by the BN was analyzed in the case of the MLP by calculating the maximum eigenvalues of the FIMs of the randomly initialized MLP and MLP with BN. However, the counterpart of the ResNet, ResNet with BN, and ResNet with stochastic depth haven't been analyzed yet. In this thesis, we provided some novel theoretical insights into the prevention of the over-fitting by the regularization methods from the Fourier spectra perspective and the fast convergence by the BN and stochastic depth in the case of the ResNet.

Toward understanding the prevention of the over-fitting by the regularization methods, we applied the Fourier spectra analysis of the MLP into the weight decay, L1, and path regularization and showed that they penalize the MLP to discourage it from learning high-frequency components. When we assume that the high-frequency components come from noise in the training set, this implies that they prevent the MLP from over-fitting the noise while learning the target function behind the noisy data.

Toward understanding the fast convergence of the GD by the BN and stochastic depth, we calculated the maximum eigenvalue of the FIM of randomly initialized ResNet, ResNet with BN, and ResNet with stochastic depth and showed that the maximum eigenvalue of the ResNet grows exponentially with respect to the depth, the stochastic depth relaxes this growth by decreasing exponential term, and the BN relaxes this exponential growth into the polynomial order. Thanks to this relaxation of the maximum eigenvalue, the BN and stochastic depth are helpful to set a large learning rate compared with that of the vanilla ResNet for its convergence.

Acknowledgements

Before I enrolled in Nara Institute of Science and Technology (NAIST), I studied computer vision and dived into deep learning to seek higher performance. At that time, many heuristics were proposed and I was wondering about reasons why these heuristics make deep learning achieve better performance. To clear my question, I joined the mathematical informatics laboratory in NAIST, learned many things from many peoples, and finally cleared some of them. I think I couldn't do it without the devotion of many peoples who helped with my research activities.

First, I must thank Professor Kazushi Ikeda for giving me interesting research topics which had not been clarified yet and teaching me beautiful analysis tools for clearing these research topics. Thanks to his good lecture, I noticed and understood how mathematics beautiful is, which led me into the doctoral course. In addition, he taught me how to make a good story for my research papers by revision. It makes these papers to be published in a journal article and conference publications. These experiences must be a foundation for my future research activities. I would like to thank him deeply.

I also thank Professor Yuji Matsumoto for reviewing this thesis. In addition, he also accepted my application into his project practice class when I was a master student. At that time, he and his laboratory's members devoted their time to my work and gave me advice from a professional perspective. Thanks to this experience, I got an interest in natural language processing and used these techniques in my research.

I also thank Associate Professor Junichiro Yoshimoto. He always gave me advanced advice from a unique viewpoint whenever I presented my research. This advice gave me inspiration for future research. Actually, his advice directly contributed to my new research and publication.

I also thank Assistant Professor Matthew J. Holland. He was a senior doctor student when I was a master student. At that time, I always respected his passion for research and tried to reference him, which made me move forward and now I reached here.

Lastly, I would like to thank members in the mathematical informatics laboratory for helping my research activities.

References

[1] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, 2019.

[3] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

[4] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[5] Pedro Domingos. *The master algorithm: How the quest for the ultimate learning machine will remake our world.* Basic Books, 2015.

[6] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

[7] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[8] Nicola Ueffing, Gholamreza Haffari, and Anoop Sarkar. Transductive learning for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 25–32, 2007.

[9] I Sutskever, O Vinyals, and QV Le. Sequence to sequence learning with neural networks. *Advances in NIPS*, 2014.

[10] Sebastien Jean, Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio. On using very large target vocabulary for neural machine translation. In *53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL-IJCNLP 2015*, pages 1–10. Association for Computational Linguistics (ACL), 2015.

[11] Tom Decroos, Lotte Bransen, Jan Van Haaren, and Jesse Davis. Actions speak louder than goals: Valuing player actions in soccer. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1851–1861. ACM, 2019.

[12] Yasutaka Furusho and Kazushi Ikeda. Generation and visualization of tennis swing motion by conditional variational rnn with hidden markov model. In *Asian Conference on Machine Learning: Trajectory, Activiy, and Behaviour workshop*, 2019.

[13] Honglak Lee, Roger Grosse, Rajesh Ranganath, and Andrew Y Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the 26th annual international conference on machine learning*, pages 609–616. ACM, 2009.

[14] Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio. Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research*, 11(Feb):625–660, 2010.

[15] Hugo Larochelle, Yoshua Bengio, Jérôme Louradour, and Pascal Lamblin. Exploring strategies for training deep neural networks. *Journal of machine learning research*, 10(Jan):1–40, 2009.

[16] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.

[17] Frank Seide, Gang Li, and Dong Yu. Conversational speech transcription using context-dependent deep neural networks. In *Twelfth annual conference of the international speech communication association*, 2011.

[18] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.

[19] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484, 2016.

[20] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015.

[21] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, 2015.

[22] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[23] Yoshua Bengio et al. Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2(1):1–127, 2009.

[24] Guido F Montufar, Razvan Pascanu, Kyunghyun Cho, and Yoshua Bengio. On the number of linear regions of deep neural networks. In *Advances in neural information processing systems*, pages 2924–2932, 2014.

[25] Matus Telgarsky. benefits of depth in neural networks. In *Conference on Learning Theory*, pages 1517–1539, 2016.

[26] Maithra Raghu, Ben Poole, Jon Kleinberg, Surya Ganguli, and Jascha Sohl-Dickstein. On the expressive power of deep neural networks. In *International Conference on Machine Learning*, pages 2847–2854, 2017.

[27] Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.

[28] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *International Conference on Learning Representations*, 2017.

[29] Devansh Arpit, Stanisław Jastrzebski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, et al. A closer look at memorization in deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 233–242. JMLR. org, 2017.

[30] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[31] Peter L Bartlett and Shahar Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3(Nov):463–482, 2002.

[32] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

[33] Behnam Neyshabur, Ryota Tomioka, and Nathan Srebro. Norm-based capacity control in neural networks. In *Conference on Learning Theory*, pages 1376–1401, 2015.

[34] Jure Sokolic, Raja Giryes, Guillermo Sapiro, and Miguel RD Rodrigues. Lessons from the rademacher complexity for deep learning. *International Conference on Learning Representations: workshop*, 2016.

[35] Behnam Neyshabur, Ryota Tomioka, and Nathan Srebro. In search of the real inductive bias: On the role of implicit regularization in deep learning. *International Conference on Learning Representations*, 2015.

[36] Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred Hamprecht, Yoshua Bengio, and Aaron Courville. On the spectral bias of neural networks. In *International Conference on Machine Learning*, pages 5301–5310, 2019.

[37] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. *International Conference on Learning Representations*, 2017.

[38] Sepp Hochreiter and Jürgen Schmidhuber. Flat minima. *Neural Computation*, 9(1):1–42, 1997.

[39] Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nati Srebro. Exploring generalization in deep learning. In *Advances in Neural Information Processing Systems*, pages 5947–5956, 2017.

[40] Zachary Charles and Dimitris Papailiopoulos. Stability and generalization of learning algorithms that converge to global optima. *International Conference on Machine Learning*, 2018.

[41] Yasutaka Furusho, Tongliang Liu, and Kazushi Ikeda. Skipping two layers in resnet makes the generalization gap smaller than skipping one or no layer. In *Recent Advances in Big Data and Deep Learning, Proceedings of the INNS Big Data and Deep Learning Conference INNSBDDL 2019, held at Sestri Levante, Genova, Italy 16-18 April 2019*, pages 349–358, 2019.

[42] Daniel Soudry, Elad Hoffer, Mor Shpigel Nacson, Suriya Gunasekar, and Nathan Srebro. The implicit bias of gradient descent on separable data. *The Journal of Machine Learning Research*, 19(1):2822–2878, 2018.

[43] Suriya Gunasekar, Jason D Lee, Daniel Soudry, and Nati Srebro. Implicit bias of gradient descent on linear convolutional networks. In *Advances in Neural Information Processing Systems*, pages 9461–9471, 2018.

[44] Sanjeev Arora, Nadav Cohen, Wei Hu, and Yuping Luo. Implicit regularization in deep matrix factorization. *Advances in neural information processing systems*, 2019.

[45] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[46] Yasutaka Furusho, Takatomi Kubo, and Kazushi Ikeda. Information theoretical analysis of deep learning representations. In *International Conference on Neural Information Processing*, pages 599–605. Springer, 2015.

[47] Yasutaka Furusho, Takatomi Kubo, and Kazushi Ikeda. Roles of pre-training in deep neural networks from information theoretical perspective. *Neurocomputing*, 248:76–79, 2017.

[48] Yasutaka Furusho and Kazushi Ikeda. Additive or concatenating skip-connection improve data separability. In *International Conference on Machine Learning: Understanding and Improving Generalization in Deep Learning*, 2019.

[49] Yasutaka Furusho and Kazushi Ikeda. Resnet and batch-normalization improve data separability. In *Asian Conference on Machine Learning*, pages 94–108, 2019.

[50] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European conference on computer vision*, pages 630–645. Springer, 2016.

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

[52] Shibani Santurkar, Dimitris Tsipras, Andrew Ilyas, and Aleksander Madry. How does batch normalization help optimization? In *Advances in Neural Information Processing Systems*, pages 2483–2493, 2018.

[53] Sepp Hochreiter. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 6(02):107–116, 1998.

[54] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *International conference on machine learning*, pages 1310–1318, 2013.

[55] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256, 2010.

[56] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *European conference on computer vision*, pages 646–661. Springer, 2016.

[57] Kenji Kawaguchi. Deep learning without poor local minima. In *Advances in Neural Information Processing Systems*, pages 586–594, 2016.

[58] Yanzhao Wu, Ling Liu, Juhyun Bae, Ka-Ho Chow, Arun Iyengar, Calton Pu, Wenqi Wei, Lei Yu, and Qi Zhang. Demystifying learning rate polices for high accuracy training of deep neural networks. *arXiv preprint arXiv:1908.06477*, 2019.

[59] Yann A LeCun, Léon Bottou, Genevieve B Orr, and Klaus-Robert Müller. Efficient backprop. In *Neural networks: Tricks of the trade*, pages 9–48. Springer, 2012.

[60] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in neural information processing systems*, pages 8571–8580, 2018.

[61] Rong Ge, Furong Huang, Chi Jin, and Yang Yuan. Escaping from saddle points—online stochastic gradient for tensor decomposition. In *Conference on Learning Theory*, pages 797–842, 2015.

[62] Jason D Lee, Max Simchowitz, Michael I Jordan, and Benjamin Recht. Gradient descent converges to minimizers. *arXiv preprint arXiv:1602.04915*, 2016.

[63] Ryo Karakida, Shotaro Akaho, and Shun-ichi Amari. Universal statistics of fisher information in deep neural networks: Mean field approach. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 1032–1041, 2019.

[64] Ryo Karakida, Shotaro Akaho, and Shun-ichi Amari. The normalization method for alleviating pathological sharpness in wide neural networks. *Advances in neural information processing systems*, 2019.

[65] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 2015.

[66] Behnam Neyshabur, Ruslan R Salakhutdinov, and Nati Srebro. Path-sgd: Path-normalized optimization in deep neural networks. In *Advances in Neural Information Processing Systems*, pages 2422–2430, 2015.

[67] Peter L Bartlett, Dylan J Foster, and Matus J Telgarsky. Spectrally-normalized margin bounds for neural networks. In *Advances in Neural Information Processing Systems*, pages 6240–6249, 2017.

[68] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *International Conference on Learning Representations*, 2018.

[69] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

[70] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

[71] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *International Conference on Learning Representations*, 2014.

[72] James Martens and Roger Grosse. Optimizing neural networks with kronecker-factored approximate curvature. In *International conference on machine learning*, pages 2408–2417, 2015.

[73] Yasutaka Furusho and Kazushi Ikeda. Theoretical analysis of resnet and batch normalization from generalization and optimization perspectives. In *APSIPA Transactions on Signal and Information Processing*, To appear.

[74] David Balduzzi, Marcus Frean, Lennox Leary, JP Lewis, Kurt Wan-Duo Ma, and Brian McWilliams. The shattered gradients problem: If resnets are the answer, then what is the question? In *International Conference on Machine Learning*, pages 342–350, 2017.

Publication list

Journal Article

1. Yasutaka Furusho and Kazushi Ikeda. Theoretical analysis of ResNet and batch normalization from generalization and optimization perspectives. APSIPA Transactions on Signal and Information Processing, to appear.

2. Yasutaka Furusho, Takatomi Kubo, and Kazushi Ikeda. Roles of pretraining in deep neural networks from information theoretical perspective. Neurocomputing, 2017.

3. Yasutaka Furusho and Kazushi Ikeda. Fourier Spectral analysis of explicit regularization for preventing memorization of neural networks. Under review.

4. Yasutaka Furusho and Kazushi Ikeda. Theoretical analysis of neural network with stochastic depth from optimization perspective. Under review.

International Conference and Workshop

1. Yasutaka Furusho and Kazushi Ikeda. ResNet and batch normalization improve data separability. Asian Conference on Machine Learning, 2019.

2. Yasutaka Furusho and Kazushi Ikeda. Generation and visualization of tennis swing motion by conditional variational RNN with hidden Markov model, Asian Conference on Machine Learning: Trajectory, Activity, and Behavior workshop, 2019.

3. Yasutaka Furusho and Kazushi Ikeda. Theoretical analysis of the fixup initialization for fast convergence and high generalization ability. International Conference on Machine Learning: Understanding and Improving Generalization in Deep Learning workshop, 2019.

4. Yasutaka Furusho and Kazushi Ikeda. Additive or concatenating skip-connections improve data separability. International Conference on Machine Learning: Understanding and Improving Generalization in Deep Learning workshop, 2019.

5. Yasutaka Furusho and Kazushi Ikeda. Effects of skip-connection in ResNet and batch normalization on Fisher information matrix. INNS Big Data and Deep Learning, 2019.

6. Yasutaka Furusho, Tongliang Liu, and Kazushi Ikeda. Skipping two layers in ResNet makes the generalization gap smaller than skipping one or no layers. INNS Big Data and Deep Learning, 2019.

7. Yasutaka Furusho and Kazushi Ikeda. Non-asymptotic analysis of Fisher information matrices of multi-layer perceptron, ResNet, and batch normalization. INCF Advances in NeuroInformatics, 2018.

8. Yasutaka Furusho, Takatomi Kubo, Kazushi Ikeda.: Information Theoretical Analysis of Deep Learning Representations. International Conference on Neural Information Processing, 2015.

Other Conference Publications

1. 古庄泰隆, 池田和氏. Stochastic depth が勾配降下法の学習速度に与える影響の理論解析. 情報論的学習理論ワークショップ, 2019.

2. 古庄泰隆, 池田和司. 古典的なニューラルネットワークの問題点と加算型及び結合型ショートカットによる改善. 情報論的学習理論と機械学習研究会, 2019.

3. 古庄泰隆, 池田和司. ResNet と batch normalization によるデータ分離能力の向上. 情報論的学習理論と機械学習研究会, 2019.

4. 古庄泰隆, 池田和司. Fixup initialization の理論解析: 学習の高速化と ResNet の汎化能力向上. 情報論的学習理論と機械学習研究会, 2019.

5. 古庄泰隆, 池田和司. Batch normalization が ResNet のフィッシャー情報行列に与える影響の理論解析. 情報論的学習理論と機械学習研究会, 2019.

6. 古庄泰隆, 池田和司. ニューラルネットワークの隠れ層のユニット数と ReLU 関数が汎化能力に与える影響の理論解析. 情報論的学習理論と機械学習研究会, 2019.

7. 古庄泰隆, 池田和司. Effects of skip-connection in ResNet and batch normalization on Fisher information matrix. 脳と心のメカニズム 冬のワークショップ, 2018.

8. 古庄泰隆, 池田和司. Stable embedding of wide neural networks with ReLU activation and its generalization ability. 脳と心のメカニズム 冬のワークショップ, 2018.

9. 古庄泰隆, 池田和司. ニューラルネットワークの埋め込み安定性と汎化能力の関係. コンピューテーショナル・インテリジェンス研究会, 2018.

10. 古庄泰隆, Tongliang Liu, 池田和司. ResNet のショートカットが学習速度と汎化ギャップに及ぼす影響の理論解析情報論的学習理論ワークショップ, 2018.

11. Yasutaka Furusho, Takatomi Kubo, and Kazushi Ikeda. Roles of pre-training in deep neural networks from information theoretical perspective. Brain and Artificial Intelligence Symposium, Brain Engineering Society of Korea, 2017.

12. 古庄泰隆, 久保孝富, 池田和司. ディープニューラルネットワークの入力符号化能力の情報理論的評価. 計測自動制御学会 システム・情報部門 学術講演会, 2015.

Award

1. 古庄泰隆, 池田和司, SICE 学術奨励賞, 計測自動制御学会, 2019.

2. 古庄泰隆, 池田和司, IBISML 研究会賞ファイナリスト, 情報論的学習理論と機械学習研究会, 2018.

3. Yasutaka Furusho, Takatomi Kubo, and Kazushi Ikeda, Best Poster Paper Award of Artificial Intelligence Symposium, Brain Engineering Society of Korea, 2017.

Appendix

A. Regularization methods

A.1 Proof of Theorem 4

Proof of Theorem 4. The Lipschitz constant of the composition function is upper bounded by those of the element functions.

$$\|f_\theta\|_{\mathrm{Lip}} \leq \|T^L\|_{\mathrm{Lip}} \cdot \|\phi\|_{\mathrm{Lip}} \cdot \|T^{L-1}\|_{\mathrm{Lip}} \cdots \|\phi\|_{\mathrm{Lip}} \cdot \|T^1\|_{\mathrm{Lip}}, \tag{30}$$

where $T^l(\cdot) = W^l \cdot$ is the affine transformation by the weight $W^l \in \mathbb{R}^{D_l \times D_{l-1}}$. On the basis of the facts that the ReLU function does not increase the Lipschitz constant, the Lipschitz constant of the affine transformation is upper bounded by its L2 norm, and the L2 norm is upper bounded by the L1 norm, the theorem holds.

$$\|f_\theta\|_{\mathrm{Lip}} \leq \prod_{l=1}^{L} \|T^l\|_{\mathrm{Lip}} \leq \prod_{l=1}^{L} \|W^l\|_2 \leq \prod_{l=1}^{L} \|W^l\|_1 \tag{31}$$

□

A.2 Proof of Theorem 5

Proof of Theorem 5. The output of the neural network can be written as the sum over the path weights and activations of these paths from the input nodes to the output node:

$$f_\theta(x) = \sum_{\alpha \in [D_0] \times [D_1] \times \cdots \times [D_L]} \prod_{l=1}^{L} W^l_{\alpha_{l+1}, \alpha_l} \cdot \prod_{l=1}^{L-1} \phi'(u^l_{\alpha_l}) \cdot x_{\alpha_1}, \tag{32}$$

where $u^l = W^l h^{l-1}$ is the output of the l-th linear layer. This path-sum representation relates the regularizers to the Lipschitz constant. The mean value theorem shows that, for any $x, x' \in \mathbb{R}^{D_0}$, there exists $\alpha \in [0, 1]$ such that

$$f(x) - f(x') \leq \nabla_x f(z) \cdot (x - x'), \quad \text{where} \quad z = \alpha \cdot x - (1 - \alpha) \cdot x'. \tag{33}$$

On the basis of the Cauchy–Schwartz inequality and the fact that the L2 norm is upper bounded by the L1 norm, the following inequality holds.

$$\|f_\theta(x) - f_\theta(x')\|_2 \leq \|\nabla_x f_\theta(z)\|_1 \cdot \|x - x'\|_2 \tag{34}$$

Therefore, the Lipschitz constant of the MLP $\|f_\theta\|_{\mathrm{Lip}}$ is upper bounded by

$$\|\nabla_x f(z)\|_1 = \sum_{i=1}^{D_0} \left| \sum_{\alpha \in \{i\} \times [D_1] \times \cdots \times \{D_L\}} \prod_{l=1}^{L} W^l_{\alpha_{l+1},\alpha_l} \cdot \prod_{l=1}^{L-1} \phi'(u^l_{\alpha_l}) \right|. \tag{35}$$

Because the activations of the paths depend on the unknown input z, we consider the worst-case input $z' \in \mathbb{R}^{D_0}$, which induces the path regularizer.

$$\|\nabla_x f_\theta(z)\|_1 \leq \sup_{z' \in \mathbb{R}^{D_0}} \|\nabla_x f_\theta(z')\|_1 = \psi_{P1}(\theta). \tag{36}$$

Theorem 5 of [33] shows that $\psi_{P1}(\theta) \leq \psi_{L1}(\theta)^L$. □

B. Optimization algorithms

B.1 Proof of Theorem 6

Lemma 1. *(Modification of Theorem 1 in [63].) The mean of the eigenvalues m_λ and the maximum eigenvalue $\lambda_{\max}$ of the expected FIM can be written by the forward signal u_i^l and the backward error signal $\delta_i^l = \frac{\partial f(x;\theta)}{\partial h_i^l}$,*

$$\begin{aligned} m_\lambda &= \frac{1}{(L+1)D^2}\left\{\sum_{i,j=1}^{D}\mathbb{E}_{\theta,x}\left[\delta_i^{0\,2} u_j^{0\,2}\right] + \sum_{l=1}^{L}\sum_{i,j=1}^{D}\mathbb{E}_{\theta,x}\left[\delta_i^{l\,2} u_j^{l\,2}\right]\right\}, \\ m_\lambda &\le \lambda_{\max} \le (L+1)D^2 m_\lambda, \end{aligned} \tag{37}$$

where the expectation is taken over the initial parameters θ and the input x in the training set.

Note that $\mathbb{E}_{\theta,x}\left[\delta_i^{l\,2} u_j^{l\,2}\right] = \mathbb{E}_{\theta,x}\left[\delta_i^{l\,2}\right]\cdot\mathbb{E}_{\theta,x}\left[u_j^{l\,2}\right]$ thanks to Assumption 1. Now, we calculate these terms. When you consider that W^0 is initialized by sampling from $\mathcal{N}\left(0, \frac{1}{D}\right)$, the second moment of its transformation of input is

$$E_{\theta,x}\left[u_i^{0\,2}\right] = \frac{1}{D}\sum_{n=1}^{D} E_x\left[x_i^2\right] = 1. \tag{38}$$

We can calculate the remaining terms by neural functional analysis [74].

Lemma 2. *Under Assumption 2, the second moments of the forward signal and the backward error signal of the ResNet are*

$$\begin{aligned} \mathbb{E}_{\theta,x}\left[u_j^{l\,2}\right] &= 2^{l-2} \quad \textit{for all} \;\; l \in [L], \\ \mathbb{E}_{\theta,x}\left[\delta_i^{l\,2}\right] &= 2^{L-l} \quad \textit{for all} \;\; l \in [L]\cup\{0\}, \end{aligned} \tag{39}$$

respectively.

Proof of Lemma 2. First, we calculate the second moment of the forward signal. Define the path set from the hidden units $\{u_i^0\}_{i=1}^D$ to the hidden unit h_j^{l-1} as

$$\text{f-path} := \left\{(F,\alpha) \middle|\; F \subset [l-1],\; \alpha \in [D]^{|F|}\times\{j\}\right\}. \tag{40}$$

Then, the forward signal can be written as the path weights and the activations of these paths,

$$h_j^{l-1} = \sum_{\tilde{\alpha} \in \text{f-path}} W_{\tilde{\alpha}} A_{\tilde{\alpha}} u_{\alpha_1}^0, \quad u_j^l = \phi\left(h_j^{l-1}\right), \tag{41}$$

where

$$W_{\tilde{\alpha}} = \begin{cases} 1 & \text{if } F = \emptyset \\ \prod_{l=1}^{|F|} W_{\alpha_{l+1},\alpha_l}^{F_l} & \text{otherwise,} \end{cases} \tag{42}$$

$$A_{\tilde{\alpha}} = \begin{cases} 1 & \text{if } F = \emptyset \\ \prod_{l=1}^{|F|} \phi'(h_{\alpha_l}^{F_l - 1}) & \text{otherwise.} \end{cases} \tag{43}$$

The path weights are orthogonal such that

$$\mathbb{E}_\theta \left[W_{\tilde{\alpha}1} W_{\tilde{\alpha}2}\right] = \begin{cases} (\frac{2}{D})^{|F|} & \text{if } \tilde{\alpha}1 = \tilde{\alpha}2 \\ 0 & \text{otherwise.} \end{cases} \tag{44}$$

Under Assumption 2, the second moment of h_j^l can be easily calculated using the orthogonality of the path weights,

$$\mathbb{E}_{\theta,x}\left[{h_j^{l-1}}^2\right] = \mathbb{E}_{\theta,x}\left[\sum_{\tilde{\alpha} \in \text{f-path}} W_{\tilde{\alpha}}^2 A_{\tilde{\alpha}}^2 {u_{\alpha_1}^0}^2\right] = \sum_{F \subset [l-1]} \left(\frac{2}{D}\right)^{|F|} \left(\frac{D}{2}\right)^{|F|} = 2^{l-1}. \tag{45}$$

The first equality holds thanks to the orthogonality of the path weights. The second equality holds because the second moment of the path weight and the number of active paths are $\mathbb{E}_\theta\left[W_{\tilde{\alpha}}^2\right] = \left(\frac{2}{D}\right)^{|F|}$ and $\sum_{\tilde{\alpha} \in \text{f-path}} A_{\tilde{\alpha}}^2 = \left(\frac{D}{2}\right)^{|F|}$, respectively, for the length of the path $|F|$ and $\mathbb{E}_{\theta,x}\left[{u_i^0}^2\right] = 1$. Then, the second moment of the forward signal is

$$\mathbb{E}_{\theta,x}\left[{u_j^l}^2\right] = \mathbb{E}_{\theta,x}\left[\phi(h_j^{l-1})^2\right] = \frac{1}{2} \cdot \mathbb{E}_{\theta,x}\left[{h_j^{l-1}}^2\right] = 2^{l-2}. \tag{46}$$

The second equality holds because the weights are initialized with the Gaussian distribution with mean zero and thus h_j^{l-1} is a symmetric random variable around zero.

Next, we calculate the second moment of the backward error signal. Define the path-set from the hidden unit h_i^l into the output unit as

$$\text{b-path} := \left\{(F, \alpha) \middle|\ F \subset [L] \setminus [l],\ \alpha \in \{i\} \times [D]^{|F|}\right\}. \tag{47}$$

Then, the backward error signal can be written as the path weights and the activations of these paths,

$$\delta_i^l = \sum_{\tilde{\alpha} \in \text{b-path}} W_{\tilde{\alpha}} A_{\tilde{\alpha}}. \tag{48}$$

Under Assumption 2, the second moment of δ_i^l can be easily calculated using the orthogonality of the path weights,

$$\mathbb{E}_{\theta,x}\left[\delta_i^{l2}\right] = \mathbb{E}_{\theta,x}\left[\sum_{\tilde{\alpha} \in \text{b-path}} W_{\tilde{\alpha}}^2 A_{\tilde{\alpha}}^2\right] = \sum_{F \subset [L] \setminus [l]} \left(\frac{2}{D}\right)^{|F|} \left(\frac{D}{2}\right)^{|F|} = 2^{L-l}. \tag{49}$$

□

Then, we can derive Theorem 2 by substituting Eqs. 46 and 49 into Eq. 37.

B.2 Proof of Theorem 7

In combination with Lemma 1 and the following lemma, we can derive Theorem 7 in the same way as the proof of Theorem 6.

Lemma 3. *Under Assumption 2, the second moments of the forward signal and the backward error signal of the ResNet with BN are*

$$\begin{aligned} \mathbb{E}_{\theta,x}\left[{u_j^l}^2\right] &= \frac{1}{2} \quad \textit{for all} \quad l \in [L], \\ \mathbb{E}_{\theta,x}\left[\delta_i^{l2}\right] &= \frac{L+1}{l+1} \quad \textit{for all} \quad l \in [L] \cup \{0\}, \end{aligned} \tag{50}$$

respectively.

Proof of Lemma 3. The second moment of the forward signal $u_j^l = \phi\left(\mathrm{BN}\left(h^{l-1}\right)\right)$ is obvious because the mean and variance of the outputs of the BN is zero and one respectively.

Next, we calculate the second moment of the backward error signal. First, we calculated the statistics of the BN. The mean is

$$\begin{aligned} \mathbb{E}_{\theta,x}\left[h_i^l\right] &= \mathbb{E}_{\theta,x}\left[\sum_{j=1}^{D} W_{i,j}^l u_j^l + h_i^{l-1}\right] = \mathbb{E}_{\theta,x}\left[\sum_{j=1}^{D} W_{i,j}^l u_j^l\right] + \mathbb{E}_{\theta,x}\left[h_i^{l-1}\right] \\ &= \mathbb{E}_{\theta,x}\left[h_i^{l-1}\right] = \mathbb{E}_{\theta,x}\left[h_i^0\right] = 0. \end{aligned} \tag{51}$$

The variance is

$$\operatorname{Var}\left(h_i^l\right) = \operatorname{Var}\left(\sum_{j=1}^{D} W_{i,j}^l u_j^l + h_i^{l-1}\right) = \operatorname{Var}\left(\sum_{j=1}^{D} W_{i,j}^l u_j^l\right) + \operatorname{Var}\left(h_i^{l-1}\right) \tag{52}$$

$$= 1 + \operatorname{Var}\left(h_i^{l-1}\right) = l + \operatorname{Var}\left(h_i^0\right) \tag{53}$$

$$= l + \frac{1}{D}\sum_{n=1}^{D} \operatorname{Var}\left(x_i\right) = l + 1. \tag{54}$$

Substitute these values into the statistics of the BN. Define the path-set from the hidden unit h_i^l into the output unit as

$$\text{b-path} := \left\{(F, \alpha) \middle|\ F \subset [L] \setminus [l],\ \alpha \in \{i\} \times [D]^{|F|}\right\}. \tag{55}$$

Then, the back ward error signal can be written as the path weights, the activations of these paths, and the statistics of the BN of these paths,

$$\delta_i^l = \sum_{\tilde{\alpha} \in \text{b-path}} W_{\tilde{\alpha}} A_{\tilde{\alpha}} \Lambda_{\tilde{\alpha}}, \tag{56}$$

where

$$\begin{aligned}
W_{\tilde{\alpha}} &= \begin{cases} 1 & \text{if } F = \emptyset \\ \prod_{l=1}^{|F|} W_{\alpha_{l+1},\alpha_l}^{F_l} & \text{otherwise,} \end{cases} \\
A_{\tilde{\alpha}} &= \begin{cases} 1 & \text{if } F = \emptyset \\ \prod_{l=1}^{|F|} \phi'(h_{\alpha_l}^{F_l - 1}) & \text{otherwise,} \end{cases} \\
\Lambda_{\tilde{\alpha}} &= \begin{cases} 1 & \text{if } F = \emptyset \\ \prod_{l=1}^{|F|} \frac{1}{\sqrt{F_l - 1 + 1}} = \prod_{l=1}^{|F|} \frac{1}{\sqrt{F_l}} & \text{otherwise.} \end{cases}
\end{aligned} \tag{57}$$

Under Assumption 2, the second moment of the backward error signal can be calculated using the orthogonality of the path weights,

$$\begin{aligned}
\mathbb{E}_{\theta,x}\left[{\delta_i^l}^2\right] &= \mathbb{E}_{\theta,x}\left[W_{\tilde{\alpha}}^2 A_{\tilde{\alpha}}^2 \Lambda_{\tilde{\alpha}}^2\right] \\
&= \sum_{F \subset [L] \setminus [l]} \left(\frac{2}{D}\right)^{|F|} \left(\frac{D}{2}\right)^{|F|} \left(\prod_{k \in F} \frac{1}{k}\right) \\
&= \prod_{k=l+1}^{L} \left(1 + \frac{1}{k}\right) = \frac{L+1}{l+1}.
\end{aligned} \tag{58}$$

□

B.3 Proof of Theorem 8

In combination with Lemma 1 and the following lemma, we can derive Theorem 8 in the same way as the proof of Theorem 6.

Lemma 4. *Under Assumption 2, the second moments of the forward signal and the backward error signal of the ResNet with stochastic depth are*

$$\begin{aligned} \mathbb{E}_{\theta,x}\left[{u_j^l}^2\right] &= p \cdot 2^{l-2} \quad \textit{for all} \quad l \in [L], \\ \mathbb{E}_{\theta,x}\left[{\delta_i^l}^2\right] &= 2^{pL-l} \quad \textit{for all} \quad l \in [L] \cup \{0\}, \end{aligned} \tag{59}$$

respectively.

Proof of Lemma 4. Consider the pL-layer shallow ResNet sampled by the stochastic depth. First, we calculate the second moment of the forward signal. Define the path set from the hidden units $\{u_i^0\}_{i=0}^D$ to the hidden unit h_j^{l-1} as

$$\text{f-path} := \left\{(F, \alpha) |\ F \subset [l-1],\ \alpha \in [D]^{|F|} \times \{j\}\right\}. \tag{60}$$

Then, the forward signal can be written as the path weights and the activations of these paths,

$$h_j^{l-1} = \sum_{\tilde{\alpha} \in \text{f-path}} W_{\tilde{\alpha}} A_{\tilde{\alpha}} u_{\alpha_1}^0, \qquad u_j^l = \beta^l \cdot \phi\left(h_j^{l-1}\right), \tag{61}$$

where $\beta^l \sim \text{Ber}(p)$ is a Bernoulli random variable and

$$\begin{aligned} W_{\tilde{\alpha}} &= \begin{cases} 1 & \text{if } F = \emptyset \\ \prod_{l=1}^{|F|} W_{\alpha_{l+1},\alpha_l}^{F_l} & \text{otherwise,} \end{cases} \\ A_{\tilde{\alpha}} &= \begin{cases} 1 & \text{if } F = \emptyset \\ \prod_{l=1}^{|F|} \phi'(h_{\alpha_l}^{F_l - 1}) & \text{otherwise,} \end{cases} \end{aligned} \tag{62}$$

Under Assumption 2, the second moment of h_j^{l-1} can be easily calculated using the orthogonality of the path weights,

$$\mathbb{E}_{\theta,x}\left[{h_j^{l-1}}^2\right] = \mathbb{E}_{\theta,x}\left[\sum_{\tilde{\alpha} \in \text{f-path}} W_{\tilde{\alpha}}^2 A_{\tilde{\alpha}}^2 {u_{\alpha_1}^0}^2\right] = \sum_{F \subset [l-1]} \left(\frac{2}{D}\right)^{|F|} \left(\frac{D}{2}\right)^{|F|} = 2^{l-1}, \tag{63}$$

Then, the second moment of the forward signal is

$$\mathbb{E}_{\theta,x}\left[{u_j^l}^2\right] = p \cdot \mathbb{E}_{\theta,x}\left[\phi(h_j^{l-1})^2\right] = \frac{p}{2} \cdot \mathbb{E}_{\theta,x}\left[{h_j^{l-1}}^2\right] = p \cdot 2^{l-2}. \tag{64}$$

Next, we calculate the second moment of the backward error signal. Define the path-set from the hidden unit h_i^l into the output unit as

$$\text{b-path} := \left\{(F,\alpha) \middle| \, F \subset [pL] \setminus [l], \ \alpha \in \{i\} \times [D]^{|F|}\right\}. \tag{65}$$

Then, the backward error signal can be written as the path weights and the activations of these paths,

$$\delta_i^l = \sum_{\tilde{\alpha} \in \text{b-path}} W_{\tilde{\alpha}} A_{\tilde{\alpha}}. \tag{66}$$

Under Assumption 2, the second moment of δ_i^l can be easily calculated using the orthogonality of the path weights,

$$\mathbb{E}_{\theta,n}\left[{\delta_i^l}^2\right] = \mathbb{E}_{\theta,n}\left[\sum_{\tilde{\alpha} \in \text{b-path}} W_{\tilde{\alpha}}^2 A_{\tilde{\alpha}}^2\right] = \sum_{F \subset [pL] \setminus [l]} \left(\frac{2}{D}\right)^{|F|} \left(\frac{D}{2}\right)^{|F|} = 2^{pL-l}. \tag{67}$$

□

B.4 Additional numerical experiments

In Section 4.3, we ran numerical experiments on the MNIST dataset. Here, we apply the same numerical experiments to the CIFAR10 dataset [6].

B.4.1 Dataset

A dataset was made by sampling from the CIFAR10 dataset with class label of 0 or 1 and its inputs were preprocessed by the PCA-whitening: its inputs were projected into D-dimensional subspace and normalized, that is, $\forall d \in [D], \mathbb{E}[x_d] = 0$ and $\text{Var}(x_d) = 1$.

B.4.2 Activation rate

Activation rates of hidden units in each layer of the ResNet, ResNet with BN, and ResNet with stochastic depth were calculated (Fig. 20) to confirm that these DNNs satisfy Assumption 2. These results confirmed the validity of Assumption 2.

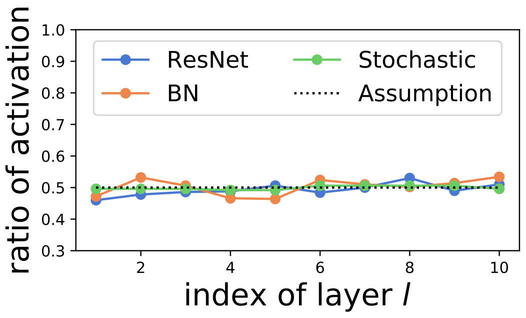


Figure 20. Activation rate of hidden units in each layer.

B.4.3 Eigenvalues of expected FIM

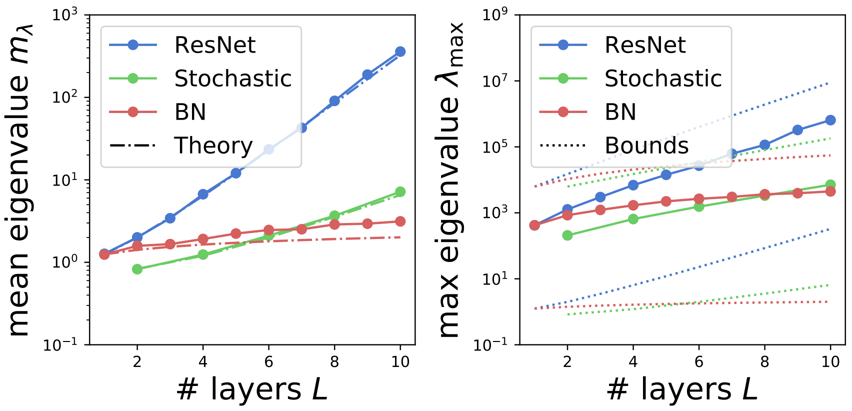
The mean eigenvalues and maximum eigenvalues of the expected FIMs of the ResNet and ResNet with BN and stochastic depth were calculated (Fig. 21). The mean eigenvalues agreed with our theoretical values and the maximum eigenvalues were between the theoretical upper and lower bounds.

B.4.4 Convergence property of GD

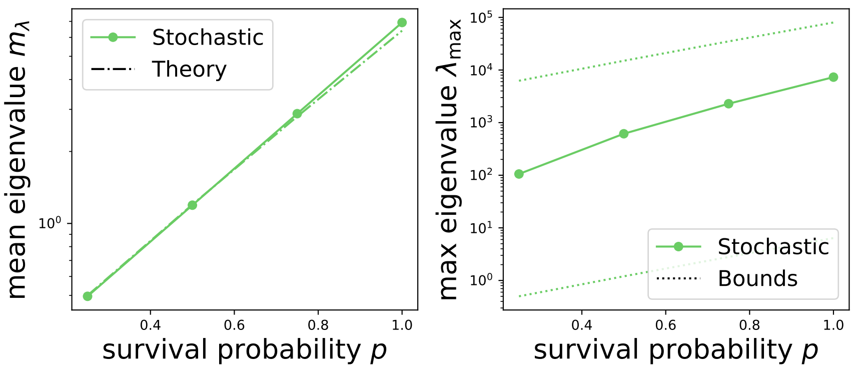
The convergence properties of the ResNet and ResNets with BN and stochastic depth were numerically examined. Each algorithm with various numbers of layers L, input dimensions D, and learning rates η updated the parameters 50 times for each run and the training loss was calculated by averaging over five runs (Figs. 22 and 23). The maximum learning rate for convergence in theory (red line), calculated as $2/(\text{upper bound of } \lambda_{\max})$, matched the boundary between the convergence (colored) and the divergence, i.e. the loss is larger than 1000 (white). In addition, the BN and stochastic depth enabled the GD to use a larger learning rate than that of the vanilla ResNet.

B.4.5 Convergence speed

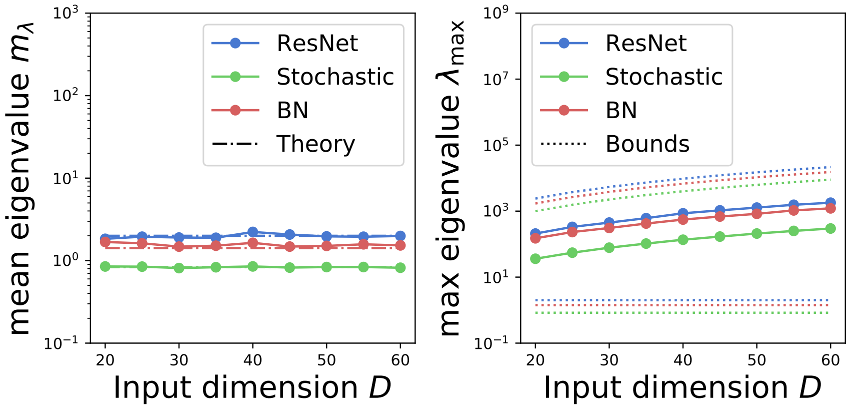
The training loss and test loss of the ResNet and ResNets with BN and stochastic depth at each update were evaluated over 10 runs (Fig. 24). Each algorithm used the optimal learning rate $\eta = 1/(\text{upper bound of } \lambda_{\max})$. These results showed that the BN and stochastic depth accelerates training.



(a) Dependence of depth L ($p = 0.5$ and $D = 50$)

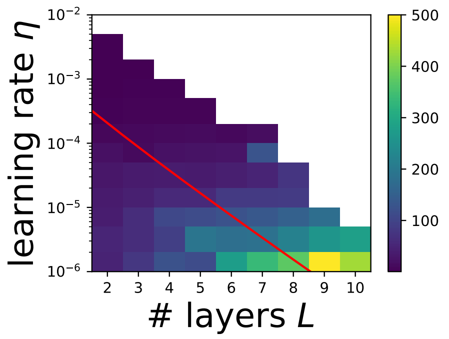


(b) Dependence of survival probability p ($L = 4$ and $D = 50$)

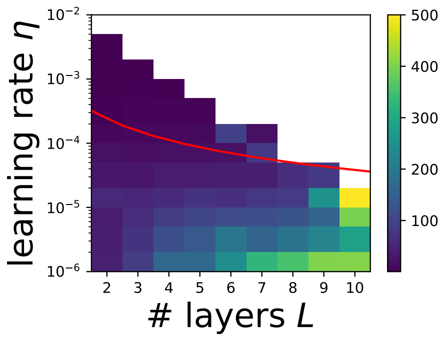


(c) Dependence of width D ($p = 0.5$ and $L = 2$)

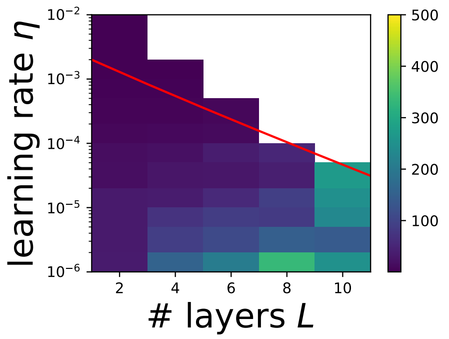
Figure 21. Eigenvalues of the expected FIMs.



(a) ResNet

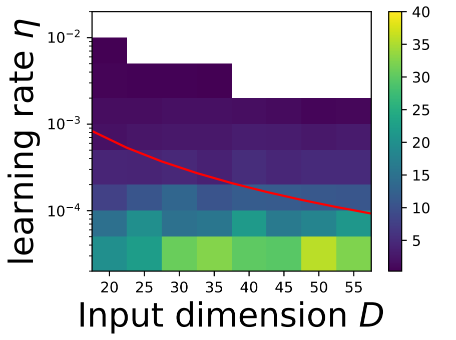


(b) ResNet with BN

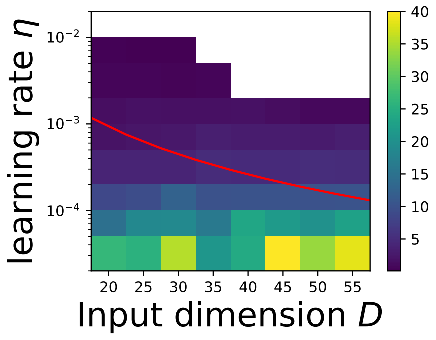


(c) ResNet with stochastic depth

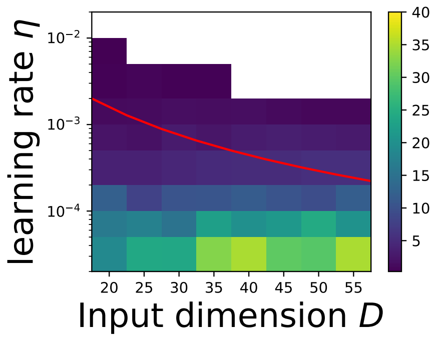
Figure 22. Convergence of the GD on the dataset with $D = 50$.



(a) ResNet



(b) ResNet with BN



(c) ResNet with stochastic depth

Figure 23. Convergence of the GD on the 2-layers DNN.

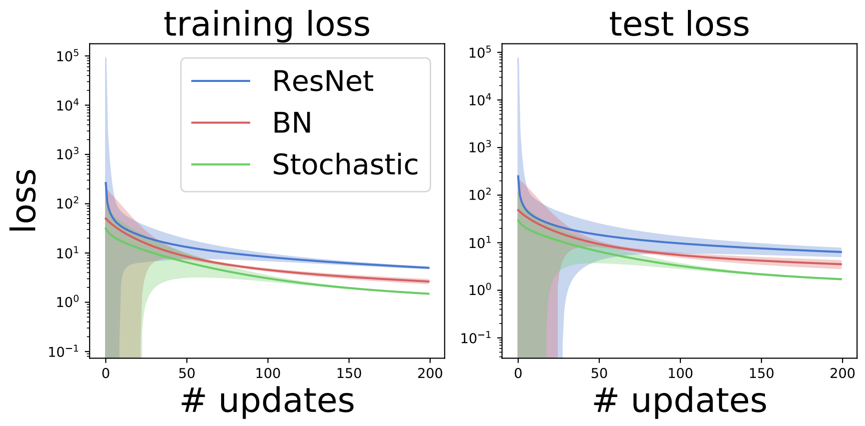


Figure 24. Loss of the 4-layer DNNs (solid line: average, shadowed area: within one s.d.)