Spectral Norm Regularization for Improving the Generalizability of Deep Learning

Yuichi Yoshida
National Institute of Informatics
yyoshida@nii.ac.jp

Takeru Miyato
Preferred Networks, Inc.
miyato@preferred.jp

Abstract

We investigate the generalizability of deep learning based on the sensitivity to input perturbation. We hypothesize that the high sensitivity to the perturbation of data degrades the performance on it. To reduce the sensitivity to perturbation, we propose a simple and effective regularization method, referred to as spectral norm regularization, which penalizes the high spectral norm of weight matrices in neural networks. We provide supportive evidence for the abovementioned hypothesis by experimentally confirming that the models trained using spectral norm regularization exhibit better generalizability than other baseline methods.

1 Introduction

Deep learning has been successfully applied to several machine learning tasks including visual object classification [12, 20], speech recognition [13], and natural language processing [6, 16]. A well-known method of training deep neural networks is stochastic gradient descent (SGD). SGD can reach a local minimum with high probability over the selection of the starting point [23], and all local minima attain similar loss values [4, 8, 17]. However, the performance on test data, that is, *generalizability*, can be significantly different among these local minima. Despite the success of deep learning applications, we lack the understanding of generalizability, even though there has been progress [18, 31].

While understanding the generalizability of deep learning is an interesting topic, it is important from a practical point of view. For example, suppose that we are training a deep neural network using SGD and want to parallelize the process using multiple GPUs or nodes to accelerate the training. A well-known method for achieving this is synchronous SGD [7, 3], which requires a large minibatch to effectively exploit parallel computation on multiple GPUs or nodes. However, it is reported that models trained through synchronous SGD with large minibatches exhibit poor generalization [18], and a method is required to resolve this problem.

In this study, we consider the generalizability of deep learning from the perspective of sensitivity to input perturbation. Intuitively, if a trained model is insensitive or sensitive to the perturbation of an input, then the model is confident or not confident about the output, respectively. As the performance on test data is important, models that are insensitive to the perturbation of *test* data are required. Note that adversarial training [29, 9] is designed to achieve insensitivity to the perturbation of *training* data, and it is not always effective for achieving insensitivity to the perturbation of test data.

To obtain insensitivity to the perturbation of test data, we propose a simple and effective regularization method, referred to as spectral norm regularization. As the name suggests, spectral norm regularization prevents the weight matrices used in neural networks from having large spectral norms. Through this, even though test data are not known in advance, a trained model is ensured to exhibit slight sensitivity to the perturbation of test data.

Using several real-world datasets, we experimentally confirm that models trained using spectral norm regularization exhibit better generalizability than models trained using other baseline methods. It is claimed in [18] that the maximum eigenvalue of the Hessian predicts the generalizability of a trained model. However, we show that the insensitivity to the perturbation of test data is a more important factor for predicting generalizability, which further motivates the use of spectral norm regularization. Finally, we show that spectral norm regularization effectively reduces the spectral norms of weight matrices.

The rest of this paper is organized as follows: We review related works in Section 2. In Section 3, we explain spectral norm regularization and compare it with other regularizing techniques . Experimental results are provided in Section 4, and conclusions are stated in Section 5.

2 Related Works

A conventional method of understanding the generalizability of a trained model is the notion of the flatness/sharpness of a local minimum [14]. A local minimum is (informally) referred to as *flat* if its loss value does not increase significantly when it is perturbed; otherwise, it is referred to as *sharp*. In general, the high sensitivity of a training function at a sharp local minimizer negatively affects the generalizability of the trained model. In [14], this is explained in more detail through the minimum description length theory, which states that statistical models that require fewer bits to describe generalize better [27].

It is known that SGD with a large minibatch size leads to a model that does not generalize well [22]. In [18], this problem is studied based on the flatness/sharpness of the obtained local minima. They formulated a flat local minimum as a local minimum at which all eigenvalues of the Hessian are small (note that all eigenvalues are non-negative at a local minimum). Using a proxy measure, they experimentally showed that SGD with a smaller minibatch size tends to converge to a flatter minimum.

The notion of flat/sharp local minima considers the sensitivity of a loss function against the perturbation of model parameters. However, it is natural to consider the sensitivity of a loss function against the perturbation of input data, as we discuss in this paper. In [29], the perturbation to training data that increases the loss function the most is considered, and the resulting perturbed training data are referred to as *adversarial examples*. It is reported in [9] that training using adversarial examples improves test accuracy.

Recently, [31] showed that the classical notions of Rademacher complexity and the VC dimension are not adequate for understanding the generalizability of deep neural networks.

Note that the spectral norm of a matrix is equal to its largest singular value. Singular values have attracted attention in the context of training recurrent neural networks (RNN). In [1, 30], it is shown that by restricting the weight matrices in RNN to be unitary or orthogonal, that is, matrices with all singular values equal to one, the problem of diminishing and exploding gradients can be prevented and better performance can be obtained.

3 Spectral Norm Regularization

In this section, we explain spectral norm regularization and how it reduces the sensitivity to test data perturbation.

3.1 General idea

We consider feed-forward neural networks as a simple example to explain the intuition behind spectral norm regularization. A feed-forward neural network can be represented as cascaded computations, $\boldsymbol{x}^\ell = f^\ell(W^\ell \boldsymbol{x}^{\ell-1} + \boldsymbol{b}^\ell)$ for $\ell = 1, \ldots, L$ for some L, where $\boldsymbol{x}^{\ell-1} \in \mathbb{R}^{n_{\ell-1}}$ is the input feature of the ℓ-th layer, $f^\ell : \mathbb{R}^{n_\ell} \to \mathbb{R}^{n_\ell}$ is a (non-linear) activation function, and $W^\ell \in \mathbb{R}^{n_\ell \times n_{\ell-1}}$ and $b^\ell \in \mathbb{R}^{n_\ell}$ are the layer-wise weight matrix and bias vector, respectively. For a set of parameters, $\Theta = \{W^\ell, \boldsymbol{b}^\ell\}_{\ell=1}^L$, we denote by $f_\Theta : \mathbb{R}^{n_0} \to \mathbb{R}^{n_L}$ the function defined as $f_\Theta(\boldsymbol{x}^0) = \boldsymbol{x}^L$. Given training data, $(\boldsymbol{x}_i, \boldsymbol{y}_i)_{i=1}^K$, where $\boldsymbol{x}_i \in \mathbb{R}^{n_0}$ and $\boldsymbol{y}_i \in \mathbb{R}^{n_L}$, the loss function is defined as $\frac{1}{K}\sum_{i=1}^K L(f_\Theta(\boldsymbol{x}_i), \boldsymbol{y}_i)$, where L is frequently selected to be cross entropy and the squared ℓ_2-

distance for classification and regression tasks, respectively. The model parameter to be learned is Θ.

Let us consider how we can obtain a model that is insensitive to the perturbation of the input. Our goal is to obtain a model, Θ, such that the ℓ_2-norm of $f(\boldsymbol{x}+\boldsymbol{\xi}) - f(\boldsymbol{x})$ is small, where $\boldsymbol{x} \in \mathbb{R}^{n_0}$ is an arbitrary vector and $\boldsymbol{\xi} \in \mathbb{R}^{n_0}$ is a perturbation vector with a small ℓ_2-norm. A key observation is that most practically used neural networks exhibit nonlinearity only because they use piecewise linear functions, such as ReLU , maxout [10], and maxpooling [26], as activation functions. In such a case, function f_Θ is a piecewise linear function. Hence, if we consider a small neighborhood of $\boldsymbol{x}$, we can regard f_Θ as a linear function. In other words, we can represent it by an affine map, $\boldsymbol{x} \mapsto W_{\Theta,\boldsymbol{x}}\boldsymbol{x} + \boldsymbol{b}_{\Theta,\boldsymbol{x}}$, using a matrix, $W_{\Theta,\boldsymbol{x}} \in \mathbb{R}^{n_0 \times n_L}$, and a vector, $\boldsymbol{b}_{\Theta,\boldsymbol{x}} \in \mathbb{R}^{n_L}$, which depend on Θ and $\boldsymbol{x}$. Then, for a small perturbation, $\boldsymbol{\xi} \in \mathbb{R}^{n_0}$, we have

$$\frac{\|f_\Theta(\boldsymbol{x}+\boldsymbol{\xi}) - f(\boldsymbol{x})\|_2}{\|\boldsymbol{\xi}\|_2} = \frac{\|(W_{\Theta,\boldsymbol{x}}(\boldsymbol{x}+\boldsymbol{\xi}) + \boldsymbol{b}_{\Theta,\boldsymbol{x}}) - (W_{\Theta,\boldsymbol{x}}\boldsymbol{x} + \boldsymbol{b}_{\Theta,\boldsymbol{x}})\|_2}{\|\boldsymbol{\xi}\|_2} = \frac{\|W_{\Theta,\boldsymbol{x}}\boldsymbol{\xi}\|_2}{\|\boldsymbol{\xi}\|_2} \leq \sigma(W_{\Theta,\boldsymbol{x}}),$$

where $\sigma(W_{\Theta,\boldsymbol{\xi}})$ is the spectral norm of $W_{\Theta,\boldsymbol{\xi}}$. The *spectral norm* of a matrix $A \in \mathbb{R}^{m \times n}$ is defined as

$$\sigma(A) = \max_{\boldsymbol{\xi} \in \mathbb{R}^n, \boldsymbol{\xi} \neq \boldsymbol{0}} \frac{\|A\boldsymbol{\xi}\|_2}{\|\boldsymbol{\xi}\|_2},$$

which corresponds to the largest singular value of A. Hence, the function f_Θ is insensitive to the perturbation of $\boldsymbol{x}$ if the spectral norm of $W_{\Theta,\boldsymbol{x}}$ is small.

The abovementioned argument suggests that model parameter Θ should be trained so that the spectral norm of $W_{\Theta,\boldsymbol{x}}$ is small for any $\boldsymbol{x}$. To further investigate the property of $W_{\Theta,\boldsymbol{x}}$, let us assume that each activation function, f^ℓ, is an element-wise ReLU (the argument can be easily generalized to other piecewise linear functions). Note that, for a given vector, $\boldsymbol{x}$, f^ℓ acts as a diagonal matrix, $D^\ell_{\Theta,\boldsymbol{x}} \in \mathbb{R}^{n_\ell \times n_\ell}$, where an element in the diagonal is equal to one if the corresponding element in $\boldsymbol{x}^{\ell-1}$ is positive; otherwise, it is equal to zero. Then, we can rewrite $W_{\Theta,\boldsymbol{x}}$ as $W_{\Theta,\boldsymbol{x}} = D^L_{\Theta,\boldsymbol{x}} W^L D^{L-1}_{\Theta,\boldsymbol{x}} W^{L-1} \cdots D^1_{\Theta,\boldsymbol{x}} W^1$. Note that $\sigma(D^\ell_{\Theta,\boldsymbol{x}}) \leq 1$ for every $\ell \in \{1, \ldots, L\}$. Hence, we have

$$\sigma(W_{\Theta,\boldsymbol{x}}) \leq \sigma(D^L_{\Theta,\boldsymbol{x}})\sigma(W^L)\sigma(D^{L-1}_{\Theta,\boldsymbol{x}})\sigma(W^{L-1}) \cdots \sigma(D^1_{\Theta,\boldsymbol{x}})\sigma(W^1) \leq \prod_{\ell=1}^{L} \sigma(W^\ell).$$

It follows that, to bound the spectral norm of $W_{\Theta,\boldsymbol{x}}$, it suffices to bound the spectral norm of W^ℓ for each $\ell \in \{1, \ldots, L\}$. This motivates us to consider spectral norm regularization, which is described in the next section.

3.2 Details of spectral norm regularization

In this subsection, we explain spectral norm regularization. The notations are the same as those used in Section 3.1. To bound the spectral norm of each weight matrix, W^ℓ, we consider the following empirical risk minimization problem:

$$\underset{\Theta}{\text{minimize}} \ \frac{1}{K} \sum_{i=1}^{K} L(f_\Theta(\boldsymbol{x}_i), \boldsymbol{y}_i) + \frac{\lambda}{2} \sum_{\ell=1}^{L} \sigma(W^\ell)^2, \tag{1}$$

where $\lambda \in \mathbb{R}_+$ is a regularization factor. We refer to the second term as the *spectral norm regularizer*. It decreases the spectral norms of the weight matrices.

When performing SGD, we need to calculate the gradient of the spectral norm regularizer. To this end, let us consider the gradient of $\sigma(W^\ell)_2^2/2$ for a particular $\ell \in \{1, 2, \ldots, L\}$. Let $\sigma_1 = \sigma(W^\ell)$ and σ_2 be the first and second singular values, respectively. If $\sigma_1 > \sigma_2$, then the gradient of $\sigma(W^\ell)^2/2$ is $\sigma_1 \boldsymbol{u}_1 \boldsymbol{v}_1^\top$, where $\boldsymbol{u}_1$ and $\boldsymbol{v}_1$ are the first left and right singular vectors, respectively. If $\sigma_1 = \sigma_2$, then $\sigma(W^\ell)_2^2$ is not differentiable. However, for practical purposes, we can assume that this case never occurs because numerical errors prevent σ_1 and σ_2 from being exactly equal.

As it is computationally expensive to compute σ_1, $\boldsymbol{u}_1$, and $\boldsymbol{v}_1$, we approximate them using the power iteration method. Starting with a randomly initialized $\boldsymbol{v} \in \mathbb{R}^{n_{\ell-1}}$, we iteratively perform the following procedure a sufficient number of times: $\boldsymbol{u} \leftarrow W^\ell \boldsymbol{v}$ and $\boldsymbol{v} \leftarrow (W^\ell)^\top \boldsymbol{u}$, and $\sigma \leftarrow \|\boldsymbol{u}\|_2 / \|\boldsymbol{v}\|_2$.

Algorithm 1 SGD with spectral norm regularization

```
1: for $\ell = 1$ to $L$ do
2:     $\boldsymbol{v}^\ell \leftarrow$ a random Gaussian vector.
3: for each iteration of SGD do
4:     Consider a minibatch, $\{(\boldsymbol{x}_{i_1}, y_{i_1}), \ldots, (\boldsymbol{x}_{i_k}, y_{i_k})\}$, from training data.
5:     Compute the gradient of $\frac{1}{k}\sum_{i=1}^{k} L(f_\Theta(\boldsymbol{x}_{i_j}), y_{i_j})$ with respect to $\Theta$.
6:     for $\ell = 1$ to $L$ do
7:         for a sufficient number of times do        ▷ One iteration was adequate in our experiments
8:             $\boldsymbol{u}^\ell \leftarrow W^\ell \boldsymbol{v}^\ell$, $\boldsymbol{v}^\ell \leftarrow (W^\ell)^\top \boldsymbol{u}^\ell$, $\sigma^\ell \leftarrow \|\boldsymbol{u}^\ell\| / \|\boldsymbol{v}^\ell\|$
9:         Add $\lambda \sigma^\ell \boldsymbol{u}^\ell (\boldsymbol{v}^\ell)^\top$ to the gradient of $W^\ell$.
10:    Update $\Theta$ using the gradient.
```

Then, σ, $\boldsymbol{u}$, and $\boldsymbol{v}$ converge to σ_1, $\boldsymbol{u}_1$, and $\boldsymbol{v}_1$, respectively (if $\sigma_1 > \sigma_2$). To approximate σ_1, $\boldsymbol{u}_1$, and $\boldsymbol{v}_1$ in the next iteration of SGD, we can reuse $\boldsymbol{v}$ as the initial vector. in our experiments, which are explained in Section 4, we performed only one iteration because it was adequate for obtaining a sufficiently good approximation. A pseudocode is provided in Algorithm 1.

Convolutions. We describe how to handle convolutions because they are used widely in recent applications of deep neural networks. Consider a convolutional layer with a input channels, b output channels, and a $k_w \times k_h$-sized kernel. This implies that the convolution has abk_wk_h parameters. Note that a value in an output channel is determined using ak_wk_h values in the input channels. Hence, we align the parameters as a matrix of size $b \times ak_wk_h$ and apply the abovementioned power iteration method to the matrix to calculate its spectral norm and gradient.

3.3 Comparison with other regularization methods

We now compare the spectral norm regularization with other regularization techniques.

Weight decay. *Weight decay*, or the *Frobenius norm regularization*, is a well-known regularization technique for deep learning. It considers the following problem:

$$\underset{\Theta}{\text{minimize}} \ \frac{1}{K}\sum_{i=1}^{K} L(f_\Theta(\boldsymbol{x}_i), \boldsymbol{y}_i) + \frac{\lambda}{2}\sum_{\ell=1}^{L} \|W^\ell\|_F^2, \tag{2}$$

where $\lambda \in \mathbb{R}_+$ is a regularization factor. We note that $\|W^\ell\|_F^2 = \sum_{i=1}^{\min\{n_{\ell-1}, n_\ell\}} \sigma_i(W^\ell)^2$, where $\sigma_i(W^\ell)$ is the i-th singular value of W^ℓ. Hence, the Frobenius norm regularization reduces the sum of squared singular values. Even though it will be effective to train models that are insensitive to input perturbation, a trained model may lose important information about the input because each trained weight matrix, W^ℓ, acts as an operator that shrinks in all directions. In contrast, spectral norm regularization focuses only on the first singular value, and each trained weight matrix, W^ℓ, does not shrink significantly in the directions orthogonal to the first right singular vector.

Adversarial training. Adversarial training [9] considers the following problem:

$$\underset{\Theta}{\text{minimize}} \ \alpha \cdot \frac{1}{K}\sum_{i=1}^{K} L(f_\Theta(\boldsymbol{x}_i), \boldsymbol{y}_i) + (1-\alpha) \cdot \frac{1}{K}\sum_{i=1}^{K} L(f_\Theta(\boldsymbol{x}_i + \boldsymbol{\eta}_i), \boldsymbol{y}_i), \tag{3}$$

where

$$\boldsymbol{\eta}_i = \epsilon \cdot \frac{\boldsymbol{g}_i}{\|\boldsymbol{g}_i\|_2} \quad \text{and} \quad \boldsymbol{g}_i = \nabla_{\boldsymbol{x}} L(f_\Theta(\boldsymbol{x}), \boldsymbol{y}_i)|_{\boldsymbol{x}=\boldsymbol{x}_i},$$

and α and $\epsilon \in \mathbb{R}_+$ are hyperparameters. It considers the perturbation toward the direction that increases the loss function the most. Hence, a trained model is insensitive to the adversarial perturbation of training data. In contrast, spectral norm regularization automatically trains a model that is insensitive to the perturbation of training data and test data.

Jacobian regularization. Another method of reducing the sensitivity of a model against input perturbation is penalizing the ℓ_2-norm of the derivative of the output with respect to the input. Let us denote the Jacobian matrix of $\boldsymbol{y}$ with respect to $\boldsymbol{x}$ by $\partial \boldsymbol{y}/\partial \boldsymbol{x}$. The *Jacobian regularization* considers the following regularization term:

$$\frac{1}{K}\sum_{i=1}^{K}\left\|\frac{\partial \boldsymbol{y}}{\partial \boldsymbol{x}}\Big|_{\boldsymbol{x}=\boldsymbol{x}_i}\right\|_F^2.$$

The Jacobian regularization promotes the smoothness of a model against input perturbation. However, this regularization is impractical because calculating the derivative of a Jacobian with respect to parameters is computationally expensive. To resolve the issue, Gu *et al.* [11] proposed an alternative method that regularizes layer-wise Jacobians:

$$\frac{1}{K}\sum_{i=1}^{K}\sum_{\ell=1}^{L}\left\|\frac{\partial \boldsymbol{x}^{\ell}}{\partial \boldsymbol{x}^{\ell-1}}\Big|_{\boldsymbol{x}^{\ell-1}=\boldsymbol{x}_i^{\ell-1}}\right\|_F^2,$$

where $\boldsymbol{x}_i^{\ell}$ is the input to the ℓ-th layer calculated using $\boldsymbol{x}_i$. Note that, if we neglect the effect of activation functions between layers, this regularization coincides with weight decay. Hence, we exclude this regularization from our experiments.

4 Experiments

In this section, we experimentally demonstrate the effectiveness of spectral norm regularization on classification tasks over other regularization techniques, and confirm that the insensitivity to test data perturbation is an important factor for generalization.

All the training methods discussed here are based on stochastic gradient descent (SGD). We consider two regimes for the choice of the mini-batch size. In the small-batch regime, we set the mini-batch size to $B = 64$, and in the large-batch regime, we set it to $B = 4096$. In our experiments, we compared the following four problems:

- Vanilla problem (vanilla): As a vanilla problem, we considered empirical risk minimization without any regularization, that is, minimize$_\Theta$ $\frac{1}{K}\sum_{i=1}^{K} L(f_\Theta(\boldsymbol{x}_i), \boldsymbol{y}_i)$, where L is the cross entropy.
- Weight decay (decay): We considered the problem (2), where L is the cross entropy. We selected the regularization factor $\lambda = 10^{-4}$.
- Adversarial training (adversarial): We considered the problem (3), where L is the cross entropy. We selected $\alpha = 0.5$ and $\epsilon = 1$, as suggested in [9].
- Spectral norm regularization (spectral): We considered the problem (1), where L is the cross entropy. We selected the regularization factor $\lambda = 0.01$.

We trained models using Nesterov's accelerated gradient descent [2] with momentum 0.9. We decreased the learning rate by a factor of $1/10$ when the half and the three quarters of the training process have passed. We optimized the hyper-parameters through a grid search and selected those that showed a reasonably good performance for every choice of neural network and mini-batch size.

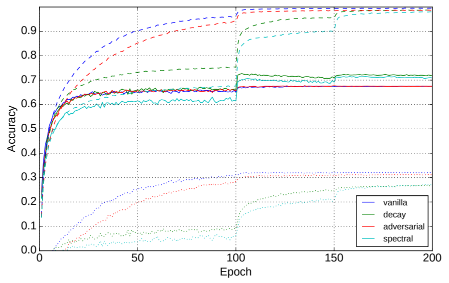
In our experiments, we used the following four settings on the model and dataset.

- The VGG network (VGGNet for short) [28] on the CIFAR-10 dataset [19].
- The network in network (NIN) model [24] on the CIFAR-100 dataset [19].
- The densely connected convolutional network (DenseNet) [15] having a depth of 40 on the CIFAR-100 dataset.
- DenseNet having a depth of 22 on the STL-10 dataset [5]. The depth was decreased because of the memory consumption issue.

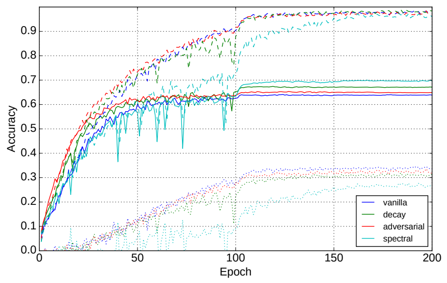
We preprocessed all the datasets using global contrast normalization. We further applied data augmentation with cropping and horizontal flip on STL-10 because the number of training data samples is only 5,000, which is small considering the large mini-batch size of 4096. The learning rate of SGD was initialized to 0.1 in the small-batch regime and 1.0 in the large-batch regime for the NIN and DenseNet models on the CIFAR-100 dataset, and was initialized to 0.01 in the small-batch regime and 0.1 in the large-batch regime for the VGGNet and DenseNet models on the CIFAR-10 dataset.

Table 1: Test accuracy and generalization gap (best values in bold).

Model	B	Test accuracy vanilla	decay	adver.	spectral	Generalization gap α	vanilla	decay	adver.	spectral
VGGNet	64	0.898	0.897	0.884	**0.904**	0.88	0.079	0.074	0.109	**0.068**
	4096	0.858	0.863	0.870	**0.885**	0.85	0.092	0.064	0.064	**0.045**
NIN	64	0.626	**0.672**	0.627	0.669	0.62	0.231	0.120	0.253	**0.090**
	4096	0.597	0.618	0.607	**0.640**	0.59	0.205	0.119	0.196	**0.090**
DenseNet (CIFAR100)	64	0.675	**0.718**	0.675	0.709	0.67	0.317	0.080	0.299	**0.095**
	4096	0.639	0.671	0.649	**0.697**	0.63	0.235	0.111	0.110	**0.051**
DenseNet (STL-10)	64	0.724	0.723	0.707	**0.735**	0.70	**0.063**	0.073	0.069	0.068
	4096	0.686	0.689	0.676	**0.697**	0.67	0.096	0.057	**0.015**	0.042



(a) $B = 64$



(b) $B = 4096$

Figure 1: Accuracy of the DenseNet model on the CIFAR-100 dataset. The solid, dashed, and dotted lines indicate the test accuracy, training accuracy, and generalization gap, respectively.

4.1 Accuracy and Generalization Gap

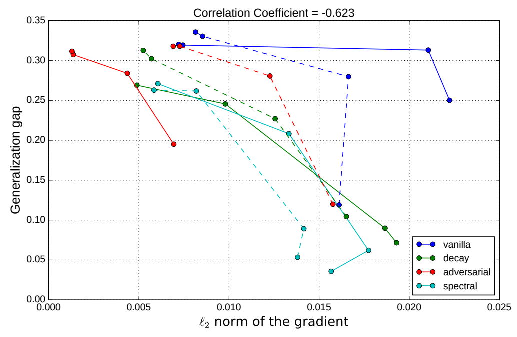
First, we look at the test accuracy obtained by each method, which is summarized in the left columns of Table 1. In the small-batch regime, decay and spectral show better test accuracies than the other two methods. In the large-batch regime, spectral clearly achieves the best test accuracy for every model. Although the test accuracy decreases as the mini-batch size increases, as reported in [18], the decrease in the test accuracy of spectral is less significant than those of the other three methods.

Next, we look at the generalization gap. We define the generalization gap at a threshold α as the minimum difference between the training and test accuracies when the test accuracy exceeds α. The generalization gap of each method is summarized in the right columns of Table 1. For each setting, we selected the threshold α so that every method achieves a test accuracy that (slightly) exceeds this threshold. For each of the settings, except for the DenseNet model on the STL-10 dataset, spectral clearly shows the smallest generalization gap followed by decay, which validates the effectiveness of spectral.

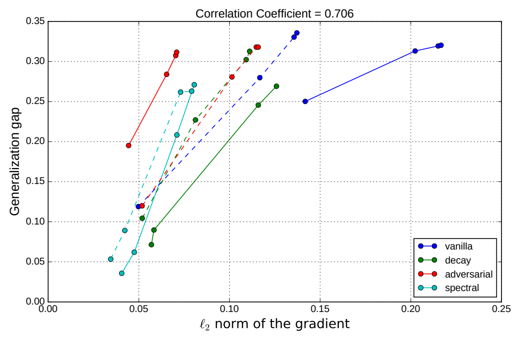
Figure 1 shows the training curve of the DenseNet model on the CIFAR-100 dataset. The results for other settings are given in Appendix A. As we can observe, in both the small-batch and large-batch regimes, spectral shows the smallest generalization gaps. The generalization gap of decay increases as the mini-batch size increases, whereas that of spectral does not increase significantly. Investigating the reason behind this phenomena is an interesting future work.

The choice of the mini-batch size and the training method is not important to obtain a model with a good training accuracy; all of them exceeds 95%. However, obtaining a model with a good test accuracy, or a small generalization gap, is important. In the subsequent sections, we investigate which property of a trained model determines its generalization gap.

To summarize, spectral consistently achieves the small generalization gap and shows the best test accuracy, especially in the large-batch regime.



(a) Train



(b) Test

Figure 2: Relation between the generalization gap and the ℓ_2-norm of the gradient of the DenseNet model on the CIFAR-100 dataset. The solid and dashed lines indicate the results for the small-batch and large-batch regimes, respectively.

4.2 Sensitivity to the perturbation of the input

To understand the relation between the sensitivity to the input perturbation and the generalization gap of the trained model, we look at the gradient of the loss function, defined with the training data or the test data, with respect to the input. Figure 2 shows the transition of the ℓ_2-norm when training the DenseNet model on the CIFAR-100 dataset. The results for other settings are given in Appendix B.

We can observe in Figure 2(b) that the ℓ_2-norm of the gradient with respect to the test data is well correlated with the generalization gap. In particular, the ℓ_2-norm of the gradient gradually increases as training proceeds, which matches the behavior of the generalization gap, which also increases as training proceeds.

On the contrary, as shown in Figure 2(a), although the ℓ_2-norm of the gradient gradually decreases as training proceeds, the generalization gap actually increases. Hence, the ℓ_2-norm of the gradient with respect to the training data does not predict the generalization gap well.

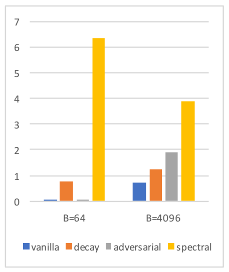
These results motivate us to reduce the ℓ_2-norm of the gradient with respect to the test data to obtain a smaller generalization gap. As we do not know the test data a priori, the spectral norm regularization method is reasonable to achieve this goal because it reduces the gradient at any point.

The superiority of spectral to decay can be elucidated from this experiment. In order to achieve a better test accuracy, we must make the training accuracy high and the generalization gap small. To achieve the latter, we have to penalize the ℓ_2-norm of the gradient with respect to the test data. To this end, decay suppresses all the weights, which decreases model complexity, and hence, we cannot fit the model to the training data well. On the other hand, spectral only suppresses the spectral norm, and hence, we can achieve a greater model complexity than decay and fit the model to the training data better.

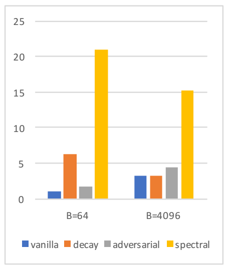
4.3 Maximum eigenvalue of the Hessian with respect to the model parameters

In [18], it is claimed that the maximum eigenvalue of the Hessian of the loss function defined with the training data predicts the generalization gap well. To confirm this claim, we computed the maximum eigenvalue of the DenseNet model trained on the CIFAR-100 dataset, shown in Figure 3. As it is computationally expensive to compute the Hessian, we approximated its maximum eigenvalue by using the power iteration method because we can calculate the Hessian-vector product without explicitly calculating the Hessian [25]. We also computed the maximum eigenvalue of the Hessian of the loss function defined with the test data.

We can observe that, for vanilla, larger eigenvalues (in both the training and test data) are obtained if the mini-batch size is increased, which confirms the claim of [18]. However, the models trained with regularizations tend to have larger eigenvalues, although they have better generalizability. In



(a) Train



(b) Test

Figure 3: Maximum eigenvalue of the Hessian of the DenseNet on CIFAR100

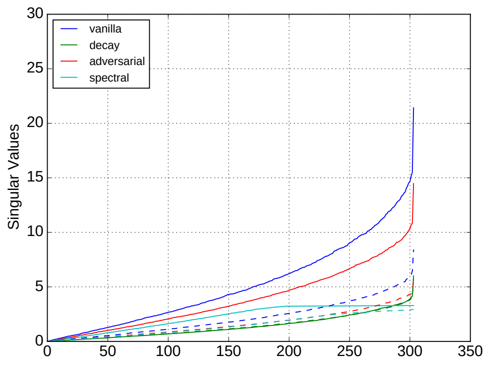


Figure 4: Singular values of weight matrices in the DenseNet on CIFAR100. Solid and dashed lines indicate the results for the small-batch and large-batch regimes, respectively.

particular, the models trained by spectral have the largest eigenvalues, although they have the best generalizability as we have seen in Section 4.1.

This phenomena can be understood as follows: If we train a model without regularization, then a small perturbation does not significantly affect the Frobenius and spectral norms of weight matrices because they are already large. However, if we train a model with regularization, then because of these small norms a small perturbation may significantly affect those norms, which may cause a significant change in the output.

To summarize, this experiment indicates that the maximum eigenvalue of the Hessian of the loss function is not a suggestive measure for predicting generalizability.

4.4 Singular values of weight matrices

Finally, we look at the singular values of weight matrices in the models trained by each method. Figures 4 shows the singular values of a weight matrix taken from the DenseNet model on the CIFAR-100 dataset. The matrix is selected arbitrarily because all matrices showed similar spectra.

We can observe that the spectrum of vanilla is highly skewed, and adversarial and decay shrink the spectrum while maintaining the skewed shape. In contrast, the spectrum of spectral is flat. This behavior is as expected because the spectral norm regularization tries to reduce the largest singular value. Because the maximum singular value obtained by spectral is low, we obtain less sensitivity to the perturbation of the input.

5 Conclusions

In this work, we hypothesized that a high sensitivity to the perturbation of the input data degrades the performance of the data. In order to reduce the sensitivity to the perturbation of the test data, we proposed the spectral norm regularization method, and confirmed that it exhibits a better generalizability than other baseline methods through experiments. Experimental comparison with other methods indicated that the insensitivity to the perturbation of the test data plays a crucial role in determining the generalizability.

There are several interesting future directions to pursue. It is known that weight decay can be seen as a regularization in MAP estimation derived from a Gaussian prior to the model parameters. Is it possible to understand spectral norm regularization as a derivation of a prior? We also need a theoretical understanding of the effect of spectral norm regularization on generalization. It is known that, in some ideal cases, weight decay improves generalizability by preventing neural networks from fitting noises [21]. Can we extend this argument to spectral norm regularization?

Acknowledgement

The authors thank Takuya Akiba and Seiya Tokui for helpful discussions.

References

[1] M. Arjovsky, A. Shah, and Y. Bengio. Unitary evolution recurrent neural networks. In *ICML*, pages 1120–1128, 2016.

[2] Y. Bengio, N. Boulanger-Lewandowski, and R. Pascanu. Advances in optimizing recurrent networks. In *ICASSP*, pages 8624–8628, 2013.

[3] J. Chen, R. Monga, S. Bengio, and R. Jozefowicz. Revisiting distributed synchronous SGD. 2016. arXiv:1604.00981.

[4] A. Choromanska, M. Henaff, M. Mathieu, G. B. Arous, and Y. LeCun. The loss surfaces of multilayer networks. In *AISTATS*, pages 192–204, 2015.

[5] A. Coates, A. Y. Ng, and H. Lee. An analysis of single-layer networks in unsupervised feature learning. *AISTATS*, 2011.

[6] R. Collobert and J. Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *ICML*, pages 160–167, 2008.

[7] D. Das, S. Avancha, D. Mudigere, K. Vaidynathan, S. Sridharan, D. Kalamkar, B. Kaul, and P. Dubey. Distributed deep learning using synchronous stochastic gradient descent. 2016.

[8] Y. N. Dauphin, R. Pascanu, C. Gulcehre, K. Cho, S. Ganguli, and Y. Bengio. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In *NIPS*, pages 2933–2941, 2014.

[9] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. In *ICLR*, 2015.

[10] I. J. Goodfellow, D. Warde-Farley, M. Mirza, and A. C. Courville. Maxout networks. In *ICML*, pages 1319–1327, 2013.

[11] S. Gu and L. Rigazio. Towards deep neural network architectures robust to adversarial examples. *arXiv preprint arXiv:1412.5068*, 2014.

[12] K. He, X. Zhang, S. Ren, and J. Sun. Identity mappings in deep residual networks. In *European Conference on Computer Vision*, pages 630–645. Springer, 2016.

[13] G. Hinton, L. Deng, D. Yu, G. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97, 2012.

[14] S. Hochreiter and J. Schmidhuber. Flat minima. *Neural Computation*, 9(1):1–42, 1997.

[15] G. Huang, Z. Liu, K. Q. Weinberger, and L. van der Maaten. Densely connected convolutional networks. 2016. arXiv:1608.06993.

[16] R. Jozefowicz, O. Vinyals, M. Schuster, N. Shazeer, and Y. Wu. Exploring the limits of language modeling. 2016. arXiv:1602.02410.

[17] K. Kawaguchi. Deep learning without poor local minima. In *NIPS*, pages 586–594, 2016.

[18] N. S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. T. P. Tang. On large-batch training for deep learning - generalization gap and sharp minima. In *ICLR*, 2017.

[19] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. Technical report, 2009.

[20] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *NIPS*, pages 1097–1105, 2012.

[21] A. Krogh and J. A. Hertz. A simple weight decay can improve generalization. In *NIPS*, pages 950–957, 1991.

[22] Y. A. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller. Efficient backprop. In *Neural networks: Tricks of the trade*, pages 9–48. Springer, 2012.

[23] J. D. Lee, M. Simchowitz, M. I. Jordan, and B. Recht. Gradient descent only converges to minimizers. In *COLT*, pages 1246–1257, 2016.

[24] M. Lin, Q. Chen, and S. Yan. Network in network. In *ICLR*, 2014.

[25] J. Martens. Deep learning via hessian-free optimization. In *ICML*, pages 735–742, 2010.

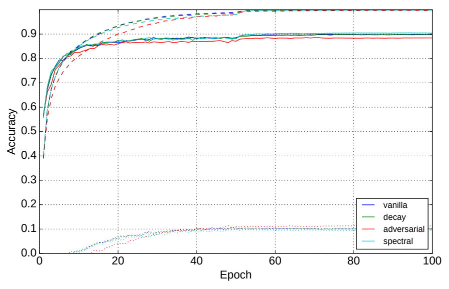
[26] M. Ranzato, F. J. Huang, Y.-L. Boureau, and Y. LeCun. Unsupervised learning of invariant feature hierarchies with applications to object recognition. In *CVPR*, pages 1–8, 2007.

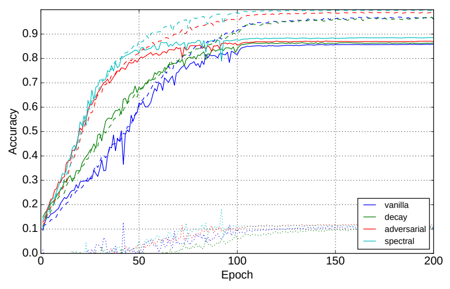
[27] J. Rissanen. A universal prior for integers and estimation by minimum description length. *The Annals of statistics*, 11(2):416–431, 1983.

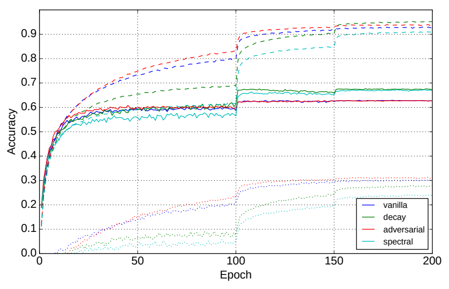
[28] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. 2014. arXiv:1409.1556v6.

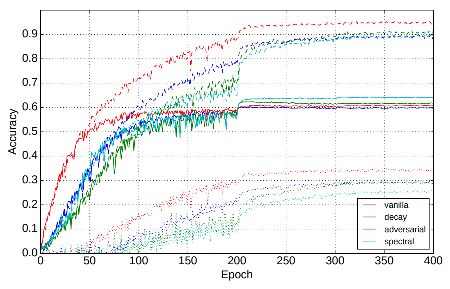
[29] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus. Intriguing properties of neural networks. In *ICLR*, 2014.

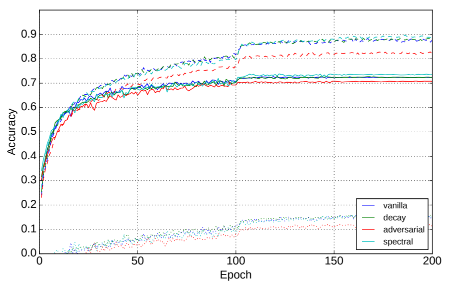
[30] S. Wisdom, T. Powers, J. Hershey, J. Le Roux, and L. Atlas. Full-capacity unitary recurrent neural networks. In *NIPS*, pages 4880–4888, 2016.


(a) VGGNet ($B = 64$)


(b) VGGNet ($B = 4096$)


(c) NIN ($B = 64$)


(d) NIN ($B = 4096$)


(e) DenseNet on STL-10 ($B = 64$)

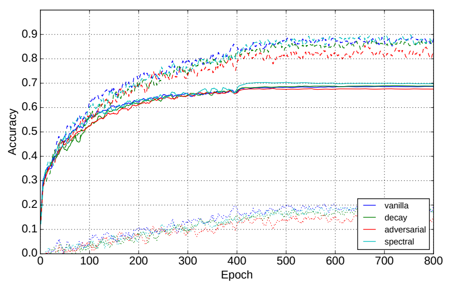

(f) DenseNet on STL-10 ($B = 4096$)

Figure 5: Accuracy. Solid, dashed, and dotted lines indicate testing accuracy, training accuracy, and the generalization gap, respectively.

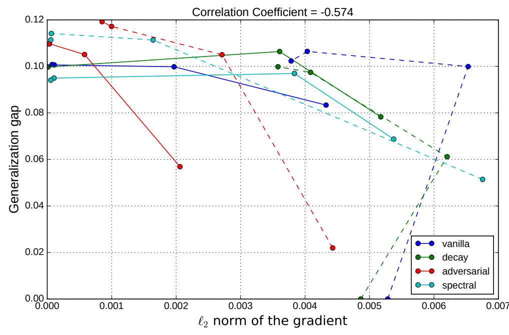
[31] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals. Uunderstanding deep learning requires rethinking generalization. In *ICLR*, 2017.

A Accuracy

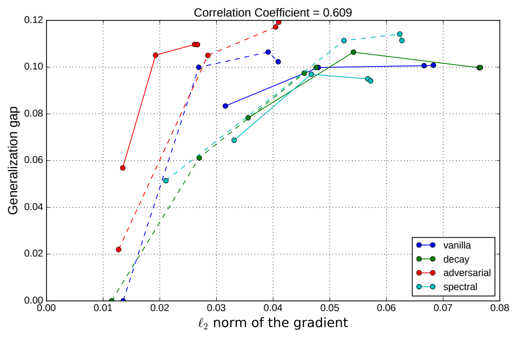
The training curves for the VGGNet, NIN, and DenseNet models on the STL-10 dataset are shown in Figure 5. We can observe that, in every setting, spectral shows the smallest generalization gap or the best test accuracy, which demonstrates that spectral can effectively reduce the generalization gap without suppressing the model complexity significantly.

B Sensitivity to the perturbation of the input

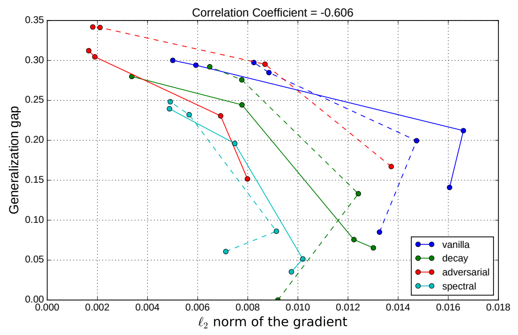
Figure 6 shows the transition of the ℓ_2-norm of the gradient of the loss function, defined with the training data or test data, with respect to the input. For every setting, the ℓ_2-norm of the gradient with respect to the test data is well-correlated with the generalization gap. On the contrary, for the



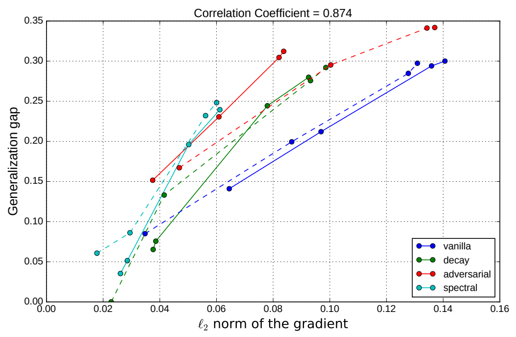
(a) VGGNet (Train)



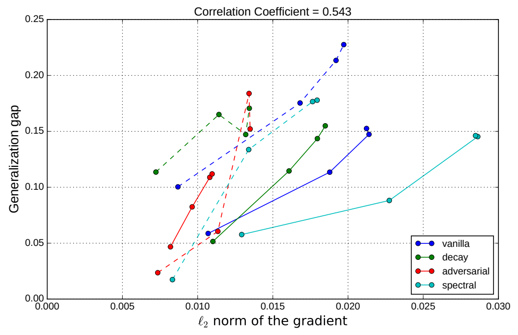
(b) VGGNet (Test)



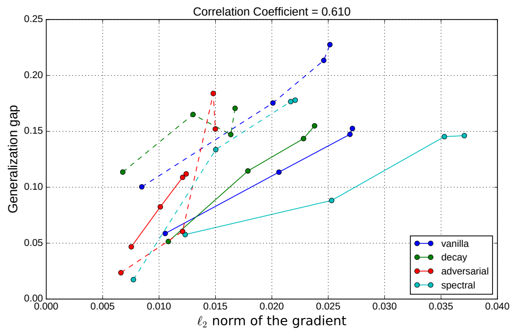
(c) NIN (Train)



(d) NIN (Test)



(e) DenseNet on STL-10 (Train)



(f) DenseNet on STL-10 (Test)

Figure 6: Relation between the generalization gap and the ℓ_2-norm of the gradient. The solid and dashed lines indicate the results for the small-batch and large-batch regimes, respectively.

VGGNet and NIN models, the ℓ_2-norm of the gradient with respect to the training data does not predict generalization gap well.