Positive Semi-Definite Convolution: A Novel Structural Regularization Technique for Deep Convolutional Networks

Hosein Hasani
Department of Computer Engineering
Sharif University of Technology
Tehran, Iran
hasanih@ce.sharif.edu

Mahdiyar Shahbazi
Department of Electrical Engineering
Sharif University of Technology
Tehran, Iran
mahdiyar.shahbazi@ee.sharif.edu

Behrad Moniri
Department of Electrical Engineering
Sharif University of Technology
Tehran, Iran
bemoniri@ee.sharif.edu

Mahdieh Soleymani Baghshah
Department of Computer Engineering
Sharif University of Technology
Tehran, Iran
soleymani@sharif.edu

Hamid Aghajan
Department of Electrical Engineering
Sharif University of Technology
Tehran, Iran
aghajan@ee.sharif.edu

Abstract

Various regularization methods have been introduced to improve the training of deep neural networks and increase their generalization capability. In this paper, we propose a novel structural regularization technique through imposing Positive Semi-Definite (PSD) constraints on convolution kernels of deep Convolutional Neural Networks (CNNs). We also introduce a proper initialization scheme for PSD kernels. Our experiments on image classification benchmarks show that utilizing PSD convolutions as a hard regularization constraint enhances the classification accuracy and decreases the generalization gap of networks. We discuss how rank constraints can be incorporated into PSD convolutions and study the effect of such constraints on the number of parameters and network accuracy. We also demonstrate that networks equipped with PSD convolutions are more robust to adversarial attacks. Finally, we show how PSD constraints can also enhance the optimization procedure for very deep networks.

1 Introduction

Successful training of deep neural networks is challenging since several issues and aspects should be taken into consideration. The generalization capability, the robustness of the trained network to adversarial input perturbation, and optimization efficiency are some of the most critical issues. Many studies have focused on one or some of these properties and tried to achieve these goals. However,

Preprint. Under review.

some aspects of training deep neural networks are still considered difficult, from both theoretical and experimental points of view.

To enhance the generalization property, various techniques have already been proposed. Some methods, e.g., drop-out and weight decay, can help but they adversely affect optimization efficiency [21]. Batch Normalization [23] and skip connections [17] are examples of the most successful methods introduced to address training issues. To relax the remaining difficulties of training deep networks, a line of research that focuses on imposing structural regularization has received attention in the past few years. Orthogonality of weight matrices is a structural constraint studied in [32, 15, 25, 31, 21, 24] for Recurrent Neural Networks (RNNs) and fully connected networks. More recently imposing this constraint for CNNs has been addressed in [4, 36]. Symmetry of weights [19] is another structural constraint introduced in recent years.

On the other hand, robustness of neural networks to adversarial attacks has received much attention due to security implications. Several defenses against adversarial attacks, including regularization techniques, have been proposed, e.g., in [27, 9, 13, 37, 7]. However, designing a network with both a good generalization capability and a high level of robustness to attacks is still considered challenging.

In this paper, we propose *Positive Semi-Definite* (PSD) constraints as a structural regularization method applied to convolution layers. We demonstrate that PSD convolution layers provide a proper foundation for more appropriate initialization schemes. We also introduce a method of applying rank constraints on PSD convolutions. Through experiments on image classification benchmarks, we show that PSD convolutions enhance the classification accuracy and decrease the generalization gap. We also discuss that by enforcing rank constraints on PSD convolutions, the number of parameters is reduced, and at the same time, network accuracy is maintained. We demonstrate that CNNs equipped with PSD convolutions exhibit higher robustness to adversarial attacks. Finally, we show that PSD convolutions can enhance the optimization procedure of very deep networks.

1.1 Related works

Orthogonality of weight matrices is an example of structural regularization techniques. The paper [4] introduces a regularization term in the loss function to penalize the distance of weights from the Stiefel manifold, e.g., by penalizing for each weight matrix W, the distance between WW^T and I. Different soft orthogonality regularizations have been introduced in [4] and it is shown that employing such methods results in faster and more stable convergence of training in ResNets.

Symmetric weight constraint is another example of structural regularization techniques. [19] has introduced several methods of imposing symmetry on the weights of a CNN. Surprisingly, it is shown that despite significantly decreasing the number of parameters, symmetry constraints have a small adverse effect on the accuracy in deep networks.

Enforcing rank constraints as another structural regularization method has recieved much attention in recent years [35, 22, 28]. Models based on these constraints are proved to be very effective in reducing the number of parameters, speeding up training, and reducing storage and hardware implementation requirements. Although these methods can maintain the generalization capability, they are unable to enhance it.

Implicit regularization when minimizing an underdetermined quadratic function over a PSD matrix X with gradient descent on a factorization of X has been studied in [14]. In this paper, in order to solve the optimization problem $\min_{X \succcurlyeq 0} ||\mathcal{A}(X) - y||_2^2$, where $\mathcal{A} : \mathbb{R}^{n \times n} \to \mathbb{R}^m$ is a linear operator and $y \in \mathbb{R}^m$, the matrix X is factorized as $X = UU^T$ with $U \in \mathbb{R}^{n \times d}$. It is shown that when d is sufficiently large and $m \ll n^2$, a global minima of the original problem is obtained by applying gradient descent on U.

Appropriate initialization is a crucial part of training and can have significant effects on the outcome. Thus, a large body of research has focused on the initialization of the weights. [16] generalizes the idea of filter-size dependent initialization [11] for ReLU non-linearity, showing that VGGNets [33] can be trained in a single optimization run. The initialization method introduced in [16] is also applied in ResNets [17]. Orthogonal initialization [20] is another example of initialization methods.

1.2 Our contributions

- We introduce PSD convolutions and propose an effective way of imposing PSD constraints on convolution layers. PSD convolution can be utilized in various well-known CNN architectures.
- An initialization method is presented for PSD convolutions that helps enforce the training to start from a model with very low complexity and gradually become more complex. We demonstrate the potential benefits of PSD weights and our initialization scheme for deeper structures.
- We show that PSD convolutions remarkably enhance the generalization capability of well-known CNN architectures in standard image classification benchmarks.
- We demonstrate that low-rank constraints can also be easily enforced on PSD convolution layers to decrease the number of parameters. The low-ranked version achieves higher speed and accuracy compared to the baselines on image classification benchmarks.
- We show that models utilizing PSD convolutions are more robust to adversarial attacks.

2 Positive Semi-Definite Constraints

In this section, we introduce PSD convolutions and an effective way of imposing PSD constraints on the kernels of a convolution layer. A method of imposing low-rank constraints on these kernels is also illustrated. Finally, an initialization method for PSD convolutions is proposed.

2.1 PSD convolutions

Let $W \in \mathbb{R}^{N \times N}$ be a positive semi-definite matrix. W can be decomposed into $W = UU^T$, where $U \in \mathbb{R}^{N \times N}$. Conversely, any matrix of the form UU^T is positive semi-definite. The decomposition of W into UU^T is not unique and hence, UU^T is an overparameterized representation for W. Besides, any matrix of the form VV^T where $V \in \mathbb{R}^{N \times K}$ and $N > K$ is low-rank and positive semi-definite [3]. Multiplying the matrix W with a vector $x \in \mathbb{R}^N$ needs N^2 multiplications. By using the factorization $W = VV^T$, the operation can be carried out with $2NK$ multiplications.

As described below and further motivated throughout the rest of the paper, we propose to constrain the weight matrices in layers of a neural network to be of the form UU^T (i.e. to be positive semi-definite). In this section, we will present an extension of this idea for CNNs. Note that, here, all filters are 4D tensors and $[k]$ denotes the set $\{1, \dots, k\}$.

We say that a $k \times k$ convolution kernel F with the same number of input and output channels N is a PSD convolution if there exist k^2 number of 1×1 filters f_i , $i \in [k^2]$ such that for every input feature map X, the output map $Y = F * X$ can be computed as follows: Let X_i , $i \in [k^2]$ denote k^2 intermediate feature maps $X_i = \Delta_i(X) * f_i$, $i \in [k^2]$, in which $*$ and Δ_i denote the convolution and placement operators, respectively. A graphical illustration of the function of the placement operator is provided in Figure (1). The output feature map Y equals $\sum_{i=1}^{k^2} X_i * f_i^{\top}$ [1]. More specifically, when we convolve the input feature map with f_i and convolve the resulting map with its transpose, $f_i^{\top}$, we are multiplying each feature vector along the channel dimension by a PSD weight matrix. Each PSD matrix is in the form of $W_i = U_i U_i^T$, where the matrix U_i is the squeezed version of f_i. It is worth noting that we do not use any bias parameters for PSD convolution layers.

In PSD layers, the filters f_i contain the parameters optimized during training and F can be easily calculated from the f_is. First, the filter $g_i = f_i \times f_i^{\top}$ is calculated,[2] then the output Y is calculated as $Y = \sum_{i=1}^{k^2} \Delta_i(X) * g_i$.

Remark 1. *There exist two methods of convolving a PSD convolution F and a given input feature map X in the test time. The first method is to directly convolve F and X, and the second method is based on using the f_is as described above. Both methods yield the same results, but they have different computational complexities. This will be discussed in more detail in Section* (2.2)*.*

[1] In this equation, $A^{\top}$ is the channel-wise transpose of the tensor A. The channel-wise transpose of A with a shape $[n_1 \times n_2 \times n_3 \times n_4]$ has the shape $[n_1 \times n_2 \times n_4 \times n_3]$.

[2] This multiplication is equivalent to multiplying U_i, the squeezed version of f_i, by U_i^T and then inserting two new axes at the beginning of it. The resulting tensor will have the shape of $[1 \times 1 \times N \times N]$.

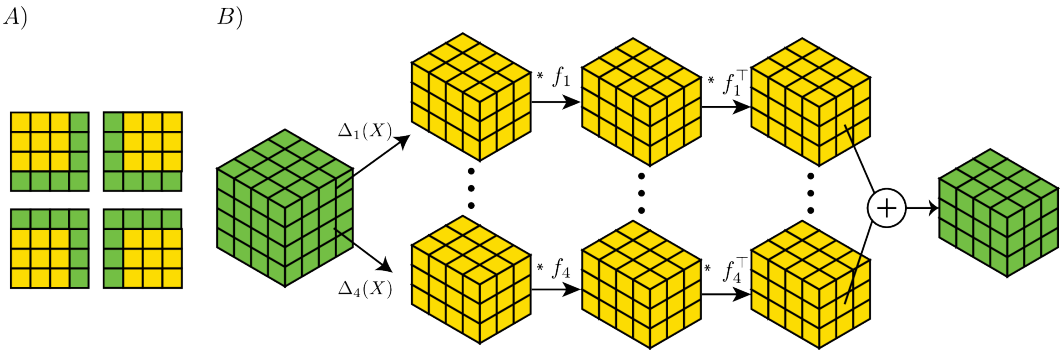

Figure 1: A) A toy example for explaining the placement operator. Consider a 2D slice of a convolution filter with kernel size 2×2. In the convolution operation, when the filter is moved through the feature map, each element of the filter only sees a specific region of the input map. For instance, the first element of the filter only can see the yellow region indicated in the upper left square. In other words, the yellow region in the upper left square represents $\Delta_1(X)$, where X is a 2D slice of the input feature map. This operator can be easily defined mathematically. B) A toy example summarizing the PSD convolution operations.

2.2 Low-ranked PSD convolutions

A PSD convolution layer has a lower degree of freedom than a normal convolution layer. However, our method of imposing PSD constraints is over-parameterized, and as a result, the number of parameters of U_i, which are the matrices that are optimized in training, is equal to the number of parameters of a normal convolution layer. The degree of freedom of a PSD convolution layer and also the number of its parameters can be reduced by imposing rank constraints.

In order to limit the rank of the filters, the number of output channels of f_is are set to be less than their input channels. This is equivalent to forcing U_is to have more rows than columns and hence, forcing $W_i = U_i U_i^T$s to be low-rank.

As mentioned in Remark (1), there exist two methods of calculating the convolution of a feature map with a PSD convolution filter in test time. In the first approach, F is directly convolved with the feature map. The second approach is the approach mentioned in (2.1). These two methods have different numbers of multiplications. Suppose that we have access to f_is and F. When $K \geq \frac{\mathrm{N}}{2}$, where N and K are the number of input and output channels of filters f_i, respectively, taking advantage of the first approach leads to a lower number of multiplications. On the other hand, when $\mathrm{K} \leq \frac{\mathrm{N}}{2}$, the second approach is more efficient.

2.3 Initialization of PSD convolutions

In training of PSD ResNets, the matrix U_i is initialized using orthonormal initialization. The orthogonal matrix used for initialization is obtained by calculating the QR decomposition of a random matrix with i.i.d. normal entries. For the full-rank version of PSD convolutions, this implies that initially, the weights $W_i = U_i U_i^T$ are equal to the identity matrix. In the low-rank case, the rectangular matrices U_i have orthonormal columns and the weights $W_i = U_i U_i^T$ can have non-zero off-diagonal entries. However, W_is are still near-identity as on average, their diagonal values are significantly larger than the off-diagonals.

This initialization technique forces the training to start from a model with low complexity. In training PSD models, only a small deviation from identity occurs during training, and the weight matrices remain near identity. These near identity weight matrices can act as a pseudo-skip-connection inside a single layer. Identity and near-identity initialization have been studied in several papers; for example, [6] proves that deep linear residual networks with identity initialization can learn positive semi-definite linear transformations efficiently via gradient descent. [5] shows that a large class of functions can be exactly written as the composition of "near-identity" mappings.

3 Experiments

To evaluate the performance of models under our proposed constraints, we set up four types of experiments. First, we apply the models to standard image classification tasks on four different benchmarks and compare their accuracy and loss functions. Then, we study the effects of rank constraints on CNNs equipped with PSD convolutions. After that, we evaluate the robustness of the models to adversarial attacks. Finally, we study the effectiveness of utilizing PSD weights in enhancing the optimization of very deep networks.

This method of imposing PSD constraints can be applied to various well-known CNN architectures. Any convolution layer with an equal number of input and output channels can benefit from PSD convolutions. Most popular CNN architectures such as ResNets [17], VGGNets [33], and MobileNets [18] consist of multiple blocks, each of which may contain one or more of such layers. The details of the PSD versions of ResNet VGGNets can be found in the supplementary materials. In CNN experiments, we report all results on ResNet-18 and ResNet-34 architectures. In PSD ResNets, PSD kernels are added to the second layer of each building block. Results for each network are compared with baseline ResNet-18 or ResNet-34 models.

Training of all models utilizes the typical optimization algorithm for ResNets, which is Stochastic Gradient Descent (SGD), with a momentum equal to 0.9 and a weight decay protocol [17]. In all models, an L_2 regularization loss is applied on the convolution kernels with $\lambda = 10^{-5}$. The weights of PSD convolution layers are initialized using the method introduced in Section (2.3). Normal convolution layers are all initialized identical to [16]. For all experiments, we set the batch size to 128. All of the experiments are carried out with Tensorflow [1].

We use the CIFAR-10/100 [26], SVHN [29], and a variant of ImageNet [8] datasets for our experiments. CIFAR-10 and CIFAR-100 consist of 50,000 training and 10,000 test color images of size 32×32, in 10 and 100 classes, respectively. SVHN is a dataset with more than 600,000 images of size 32×32 in 10 classes for training and 26,032 images for testing. We also use a small and balanced variant of ImageNet, which we call Small-ImageNet. It consists of 100 randomly chosen categories from ImageNet. For each category, 500 instances are randomly selected for training, 50 instances for validation, and 100 instances as the test set. All images of this dataset are cropped around their centers and resized to 160×160 pixels. In all experiments, datasets are normalized with respect to the mean and standard deviation of the training sets. For CIFAR and ImageNet experiments, just a horizontal flip is applied and no further augmentation is applied to the datasets.

3.1 Image classification experiments

The accuracies of the PSD versions of ResNet-18 and ResNet-34 alongside their baselines are reported in Table (1). The reported accuracies are averaged over five trials. In both the ResNet-18 and ResNet-34 architectures, the PSD version of each network achieves higher accuracy than its baseline on all datasets. As it is seen, most of the result are statistically significant.

Table 1: Classification accuracy of the PSD and baseline networks on different datasets.

Architecture	Model	Dataset			
		CIFAR-10	CIFAR-100	SVHN	Small-ImageNet
ResNet-18	PSD	$\mathbf{89.7}_{\pm \mathbf{0.2}}$	$\mathbf{66.9}_{\pm \mathbf{0.6}}$	$\mathbf{97.85}_{\pm \mathbf{0.01}}$	$\mathbf{69.5}_{\pm \mathbf{0.3}}$
	Baseline	$89.0_{\pm 0.2}$	$65.9_{\pm 0.1}$	$97.74_{\pm 0.06}$	$69.4_{\pm 0.3}$
ResNet-34	PSD	$\mathbf{90.9}_{\pm \mathbf{0.2}}$	$\mathbf{69.0}_{\pm \mathbf{0.3}}$	$\mathbf{98.01}_{\pm \mathbf{0.02}}$	$\mathbf{70.9}_{\pm \mathbf{0.2}}$
	Baseline	$89.4_{\pm 0.2}$	$66.7_{\pm 0.4}$	$97.91_{\pm 0.05}$	$70.7_{\pm 0.2}$

To better demonstrate the superior performance of PSD models, the loss function and accuracy of PSD and baseline ResNet-34 networks on the test and training data of SVHN is illustrated in Figure (2). It is observed that the baseline ResNet-34 achieves a higher training accuracy than PSD ResNet-34. However, PSD models exhibit a smaller generalization gap and achieve a higher test accuracy in the entire steps of training. This reduction in generalization gap can be attributed to the model being less complex and having a lower degrees of freedom.

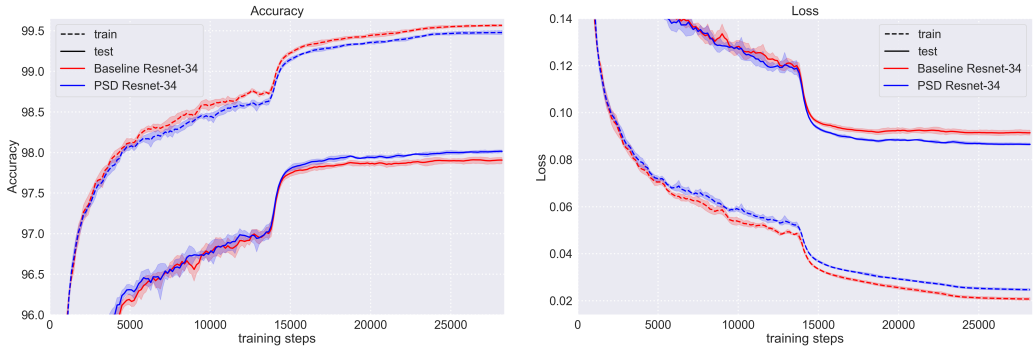

Figure 2: The accuracy (left) and loss function (right) of PSD and baseline ResNet-34 on test and train data in the SVHN experiment. The pale margin indicates standard deviation across different trials.

With a motivation to further reduce the complexity of the networks, we introduce a new version of PSD ResNets called the Linear PSD ResNets. In Linear PSD ResNets, we remove batch normalization and use a linear activation function, instead of ReLU after all PSD convolution layers of the networks. The activation functions for other convolution layers remain intact. The other details of the networks and training are identical to the PSD ResNet. The accuracy of PSD ResNets, Linear PSD ResNets, and the baselines on CIFAR-10 are reported in Table (2). This table also includes the accuracy of models with the channel-wise symmetry constraints, denoted as Symmetric ResNets, as introduced in [19]. The reported accuracies are averaged over five trials. It is shown that Linear PSD ResNets can even achieve a higher accuracy than the PSD networks.

In these experiments, it is observed that models employing PSD convolutions exhibit higher accuracies than models that take advantage of symmetry constraints. This observation proves that the performance boost observed as a result of imposing PSD constraints is a direct result of additional attributes of PSD matrices and not just the symmetry that PSD constraints also impose on kernels of a PSD convolution layer.

Table 2: Test accuracy of the PSD, Linear (Lin.) PSD, Symmetric (Sym.) and Baseline ResNet-18 and ResNet-34 on CIFAR-10.

	ResNet-18				ResNet-34			
	PSD	Lin PSD	Sym	Baseline	PSD	Lin PSD	Sym	Baseline
Acc.	$89.7_{\pm 0.2}$	$\mathbf{90.2}_{\pm \mathbf{0.1}}$	$88.7_{\pm 0.1}$	$89.0_{\pm 0.2}$	$90.9_{\pm 0.2}$	$\mathbf{91.3}_{\pm \mathbf{0.1}}$	$89.4_{\pm 0.2}$	$89.4_{\pm 0.2}$

Figure (3) illustrates the accuracy and loss function of PSD, linear PSD, symmetric and baseline versions of ResNet-18 on CIFAR-10 as a function of training steps.

3.2 Low-rank experiments

Table (3) shows the comparison of the test accuracy between the baseline, PSD and two versions of low-rank PSD ResNet-18 and ResNet-34 on the CIFAR-10 dataset. PSD/2 and PSD/4 denote two low-rank ResNet models utilizing PSD convolutions. In PSD/2 and PSD/4, the matrices $W_i = U_i U_i^T$, $U_i \in \mathbb{R}^{N \times K}$ have $\frac{K}{N}$ equal to $\frac{1}{2}$ and $\frac{1}{4}$, respectively. Alongside these models, two control models UV/2 and UV/4 have also been introduced. In these models, the matrices W_i are decomposed as $W_i = UV$, where $U, V \in \mathbb{R}^{N \times K}$. In UV/2 and UV/4, the ratio $\frac{K}{N}$ is equal to $\frac{1}{2}$ and $\frac{1}{4}$, respectively. The matrices U and V are initialized independently in the same way done for U in PSD convolutions.

It can be seen that each low-rank PSD model (PSD/2 and PSD/4) achieves a higher accuracy than its low-rank counterpart (UV/2 and UV/4) despite having fewer parameters. This brings us to the conclusion that PSD + low-rank configurations are more effective than low-rank configuration in

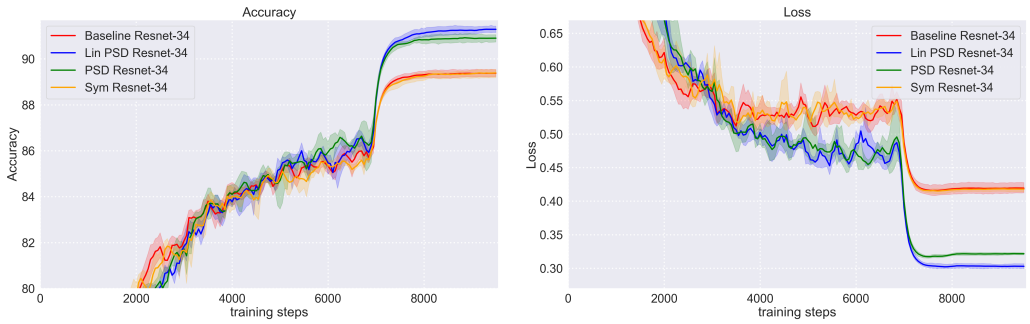

Figure 3: The accuracy (left) and loss function (right) of PSD, linear PSD, symmetric and baseline versions of ResNet-18 on CIFAR-10 as a function of training steps.

parameter reduction and maintaining the accuracy. The low-rank PSD model also achieves a higher accuracy than the baseline, but slightly less than the PSD model, leading us to the conclusion that low-rank PSD constraints are an effective way of decreasing network parameters and increasing its speed, but at the same time, maintaining the generalization capability of PSD networks.

Table 3: The accuracy and the number of parameters of low-rank PSD models PSD/2 and PSD/4, low-rank models UV/2 and UV/4, PSD models and the baselines on CIFAR-10.

	ResNet-18		ResNet-34	
Model	# params	Accuracy	# params	Accuracy
PSD	11.2 M	$89.7_{\pm 0.2}$	21.3 M	$90.9_{\pm 0.2}$
PSD/2	8.0 M	$89.3_{\pm 0.3}$	15.6 M	$90.4_{\pm 0.1}$
UV/2	11.2 M	$88.6_{\pm 0.1}$	21.3 M	$89.2_{\pm 0.1}$
PSD/4	6.5 M	$89.2_{\pm 0.2}$	12.8 M	$90.1_{\pm 0.1}$
UV/4	8.0 M	$88.51_{\pm 0.04}$	15.6 M	$88.9_{\pm 0.2}$
Baseline	11.2 M	$89.0_{\pm 0.2}$	21.3 M	$89.4_{\pm 0.2}$

3.3 Robustness to adversarial attacks

We use the test accuracy on adversarial samples generated through Fast Gradient Sign Method (FGSM) [12] as a measure of a network's robustness. FGSM is an attack for an l_∞-bounded adversary, and generates an adversarial example as:

$$\mathbf{x}' = \mathbf{x} + \epsilon \text{sign}(\nabla_{\mathbf{x}} L(\boldsymbol{\theta}, \mathbf{x}, y)),$$

where $\boldsymbol{\theta}$ is the vector of the parameters of the model, $\mathbf{x}$ is the input, y is the output associated with $\mathbf{x}$, and L is the loss function.

Figure (4) demonstrates the accuracy of PSD ResNet-18, PSD ResNet-34, and the baselines trained on CIFAR-10 and CIFAR-100 on samples generated through FGSM as a function of ϵ. It can be seen that ResNet-18 and ResNet-34 models utilizing PSD convolutions enjoy a higher robustness against adversarial perturbations of the inputs.

3.4 PSD weights in deeper structures

In this section, we analyze the impact of PSD weights on deeper structures and its potential effects on the training of such networks. To better assess the sole effect of a PSD configuration in deep networks, we conduct our experiment on multi-layer perceptrons (MLPs) for their simplicity and interpretability.

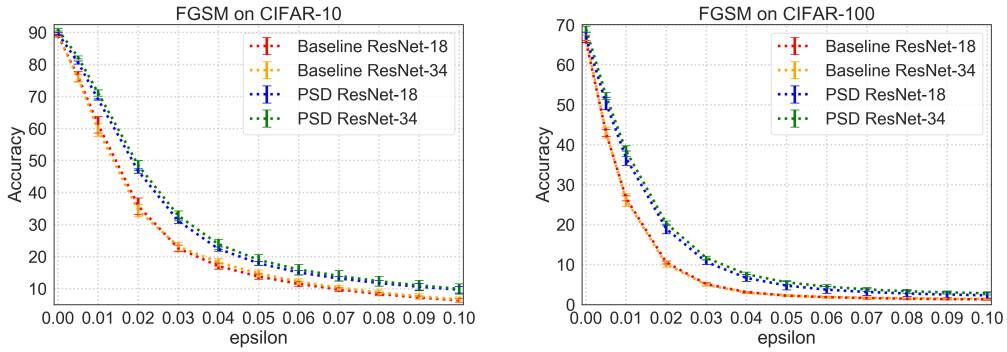

Figure 4: Test accuracy of PSD ResNet-18, PSD ResNet-34, and the baselines on adversarial samples generated through FGSM as a function of ϵ in CIFAR-10 (left) and CIFAR-100 (right). Note that for a better visualization, the error bars shown are twice the standard deviations.

In this experiment, we have used the CIFAR-10 dataset. The first and second layers of MLPs change the dimension from 3072 to 1024 and from 1024 to 512, respectively. All other layers have square weight matrices of size 512×512. In PSD and symmetric MLPs, the square weight matrices are enforced to be positive semi-definite and symmetric, respectively. The weights of the baseline MLPs are not constrained.

The test and train accuracy of PSD, symmetric and baseline MLPs as a function of their number of layers is illustrated in Figure (5). All networks were trained using the Adam optimization algorithm. The results are reported for two different learning rates of 10^{-4} and 10^{-5}.

In this experiment, as the number of layers increases, it is expected that the MLPs should be able to fit the data perfectly and achieve 100% train accuracy. However, this is not the case for baseline and symmetric MLPs and a significant drop is observed in their train accuracy when layers are added. This drop is attributed to the fact that very deep baseline and symmetric MLP models cannot be trained properly. Such a significant drop is not observed in the test or train accuracy of networks utilizing PSD weights, and training is possible even for very deep PSD MLPs. This effect suggests that our PSD configuration equipped with its specific initialization scheme can enhance the optimization and gradient flow in very deep networks.

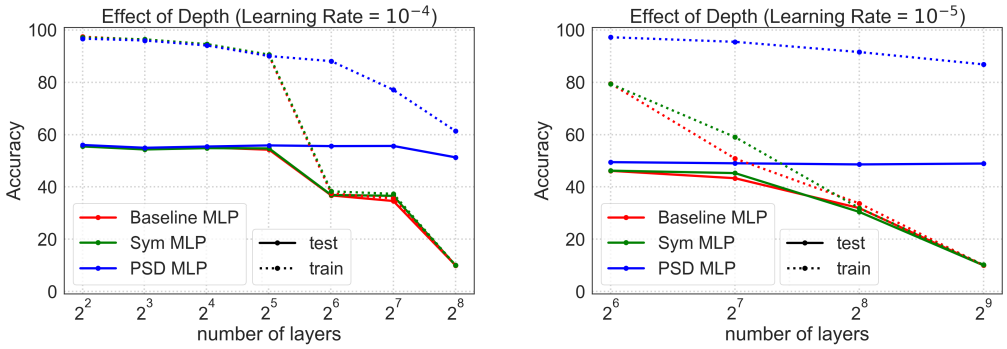

Figure 5: The test and train accuracy of PSD, symmetric, and baseline MLPs trained with a learning rate equal 10^{-4} (left) and 10^{-5} (right) with respect to the number of layers.

4 Discussion

In this paper, we impose PSD constraints on kernels of a convolution layer. We demonstrated that the proposed method can improve the performance of CNNs despite decreasing the degree of freedom

of convolution layers. CNNs utilizing PSD constraints exhibit lower generalization gaps and are more robust to adversarial samples. A low-rank version of PSD convolution is also introduced to decrease the number of parameters optimized in training. It is shown that even under low-rank PSD constraints, the models achieve higher test accuracies than baseline models. Finally, we demonstrated that PSD weight matrices can enhance the optimization of deeper structures. These results and the outcome of our control experiments provide evidence that the proposed structural regularization technique enhances the general performance of a deep neural network. However, several theoretical and experimental questions still remain open for future works.

Although an improved generalization is experimentally demonstrated in this paper, a comprehensive theoretical justification of this phenomenon does not yet exist. The over-parameterized representation of PSD convolutions used in this study could be a potential reason for this enhanced generalization [10, 30, 38, 34, 2].

The enhanced optimization of deeper structures under PSD constraints might be a consequence of a better flow of the gradient in PSD networks. The near-identity PSD kernels implicitly add shortcut connections [17] to each convolution layer and enhance residual learning. More in-depth studies on identity and near-identity initialization, and its potential effect in the training of very deep networks, could serve as an interesting course for further research.

Broader Impact

Nowadays, the importance of convolutional networks is evident to everyone. There are many areas in which CNNs are applied. One of the most notable examples is the analysis of medical images. Recently it has been proved that CNNs can be a "good" assistant to the doctors in diagnosing many types of Cancers. However, in some cases, their diagnoses are not fully trustable. Since our proposed method results in a lower generalization gap, and a higher robustness to input perturbations, the analysis based on them could be more reliable.

To the best of our knowledge, there are no data or algorithmic bias concerned with our methods and experiments. Statistical considerations were taking into account in all experiments.

References

[1] Martin Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek G. Murray, Benoit Steiner, Paul Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. Tensorflow: A system for large-scale machine learning. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, pages 265–283, 2016.

[2] Sanjeev Arora, N Cohen, and Elad Hazan. On the optimization of deep networks: Implicit acceleration by overparameterization. In *35th International Conference on Machine Learning*, 2018.

[3] Sheldon Jay Axler. *Linear Algebra Done Right*. Undergraduate Texts in Mathematics. Springer, New York, 1997.

[4] Nitin Bansal, Xiaohan Chen, and Zhangyang Wang. Can we gain more from orthogonality regularizations in training deep cnns? In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 4266–4276. Curran Associates Inc., 2018.

[5] Peter L Bartlett, Steven N Evans, and Philip M Long. Representing smooth functions as compositions of near-identity functions with implications for deep network optimization. *arXiv preprint arXiv:1804.05012*, 2018.

[6] Peter L Bartlett, David P Helmbold, and Philip M Long. Gradient descent with identity initialization efficiently learns positive-definite linear transformations by deep residual networks. *Neural computation*, 31(3):477–502, 2019.

[7] Arjun Nitin Bhagoji, Daniel Cullina, Chawin Sitawarin, and Prateek Mittal. Enhancing robustness of machine learning systems via data transformations. In *2018 52nd Annual Conference on Information Sciences and Systems (CISS)*, pages 1–5. IEEE, 2018.

[8] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

[9] Reuben Feinman, Ryan R Curtin, Saurabh Shintre, and Andrew B Gardner. Detecting adversarial samples from artifacts. *arXiv preprint arXiv:1703.00410*, 2017.

[10] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018.

[11] 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.

[12] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[13] Kathrin Grosse, Praveen Manoharan, Nicolas Papernot, Michael Backes, and Patrick McDaniel. On the (statistical) detection of adversarial examples. *arXiv preprint arXiv:1702.06280*, 2017.

[14] Suriya Gunasekar, Blake E Woodworth, Srinadh Bhojanapalli, Behnam Neyshabur, and Nati Srebro. Implicit regularization in matrix factorization. In *Advances in Neural Information Processing Systems*, pages 6151–6159, 2017.

[15] Mehrtash Harandi and Basura Fernando. Generalized backpropagation, Étude de cas: Orthogonality. *arXiv preprint arXiv:1611.05927*, 2016.

[16] 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.

[17] 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.

[18] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[19] Shell Xu Hu, Sergey Zagoruyko, and Nikos Komodakis. Exploring weight symmetry in deep neural networks. *Computer Vision and Image Understanding*, 187:102786, 2019.

[20] Wei Hu, Lechao Xiao, and Jeffrey Pennington. Provable benefit of orthogonal initialization in optimizing deep linear networks. *arXiv preprint arXiv:2001.05992*, 2020.

[21] Lei Huang, Xianglong Liu, Bo Lang, Adams Wei Yu, Yongliang Wang, and Bo Li. Orthogonal weight normalization: Solution to optimization over multiple dependent stiefel manifolds in deep neural networks. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[22] Yani Ioannou, Duncan Robertson, Jamie Shotton, Roberto Cipolla, and Antonio Criminisi. Training cnns with low-rank filters for efficient image classification. *arXiv preprint arXiv:1511.06744*, 2015.

[23] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[24] Kui Jia, Shuai Li, Yuxin Wen, Tongliang Liu, and Dacheng Tao. Orthogonal deep neural networks. *IEEE transactions on pattern analysis and machine intelligence*, 2019.

[25] Kui Jia, Dacheng Tao, Shenghua Gao, and Xiangmin Xu. Improving training of deep neural networks via singular value bounding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4344–4352, 2017.

[26] Alex Krizhevsky et al. Learning multiple layers of features from tiny images. 2009.

[27] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[28] Franck Mamalet and Christophe Garcia. Simplifying convnets for fast learning. In *International Conference on Artificial Neural Networks*, pages 58–65. Springer, 2012.

[29] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011.

[30] Behnam Neyshabur and Zhiyuan Li. Towards understanding the role of over-parametrization in generalization of neural networks. In *International Conference on Learning Representations (ICLR)*, 2019.

[31] Mete Ozay and Takayuki Okatani. Optimization on product submanifolds of convolution kernels. *arXiv preprint arXiv:1701.06123*, 2017.

[32] Tim Salimans and Durk P Kingma. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In *Advances in neural information processing systems*, pages 901–909, 2016.

[33] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[34] Mahdi Soltanolkotabi, Adel Javanmard, and Jason D Lee. Theoretical insights into the optimization landscape of over-parameterized shallow neural networks. *IEEE Transactions on Information Theory*, 65(2):742–769, 2018.

[35] Cheng Tai, Tong Xiao, Yi Zhang, Xiaogang Wang, et al. Convolutional neural networks with low-rank regularization. *arXiv preprint arXiv:1511.06067*, 2015.

[36] Jiayun Wang, Yubei Chen, Rudrasis Chakraborty, and Stella X Yu. Orthogonal convolutional neural networks. *arXiv preprint arXiv:1911.12207*, 2019.

[37] Xiaoyong Yuan, Pan He, Qile Zhu, and Xiaolin Li. Adversarial examples: Attacks and defenses for deep learning. *IEEE transactions on neural networks and learning systems*, 30(9):2805–2824, 2019.

[38] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*, 2016.