Tight and Efficient Upper Bound on Spectral Norm of Convolutional Layers

Ekaterina Grishina, Mikhail Gorbunov, and Maxim Rakhuba

HSE University
ergrishina@edu.hse.ru

Abstract. Controlling the spectral norm of the Jacobian matrix, which is related to the convolution operation, has been shown to improve generalization, training stability and robustness in CNNs. Existing methods for computing the norm either tend to overestimate it or their performance may deteriorate quickly with increasing the input and kernel sizes. In this paper, we demonstrate that the tensor version of the spectral norm of a four-dimensional convolution kernel, up to a constant factor, serves as an upper bound for the spectral norm of the Jacobian matrix associated with the convolution operation. This new upper bound is independent of the input image resolution, differentiable and can be efficiently calculated during training. Through experiments, we demonstrate how this new bound can be used to improve the performance of convolutional architectures.

Keywords: Spectral norm · Convolutional layer · Lipschitz constant

1 Introduction

Controlling spectral norm of convolutional layers' Jacobians is a way to make models more robust to input perturbations [2,20,22], increase generalization performance [16], prevent explosion of the gradients during back propagation [28]. In addition, bounds on spectral norm have been used to construct orthogonal convolutional layers [14, 21].

Despite the advantages offered by controlling the spectral norm, its efficient computation for convolutional layers remains a challenging task. Finding the spectral norm is equivalent to finding the largest singular value of a matrix and the straightforward computation of the singular value decomposition (SVD) is not feasible due to the size of the convolution Jacobian T. Most of the other existing techniques rely on the input sizes of images, which can result in a significant computational load for high-resolution images or images with more than two spatial dimensions.

In this work, we derive a new accurate upper bound for the spectral norm of the Jacobian $\|T\|_2$. For a kernel tensor $K \in \mathbb{R}^{c_{in} \times c_{out} \times h \times w}$, this bound can be computed with $\mathcal{O}(c_{in}c_{out}hw)$ operations, where c_{in}, c_{out} are the number of input and output channels and h, w are the filter sizes. Note that this complexity does

not depend on the input or output image resolution. As a result, controlling the spectral norm during training comes at almost no additional cost.

To be more precise, our bound is based on a norm denoted as $\|K\|_\sigma$, which is induced by the kernel tensor K as a multilinear functional. Specifically, we show that $\|K\|_\sigma \leq \|T\|_2 \leq \sqrt{hw}\|K\|_\sigma$. The tensor norm $\|K\|_\sigma$ is well-known in applied multilinear algebra and can be efficiently computed by a higher-order generalization of a power method (HOPM) [3]. We will refer to our tensor norm upper bound $\sqrt{hw}\|K\|_\sigma$ as TN and will use it for several new regularization strategies. Compared to the existing approach [20], our bound is guaranteed to be more accurate. It is also applicable for zero-padded convolutions and can be modified to account for strided convolutions, which are both widely used in convolutional architectures. Moreover, it straightforwardly generalizes to convolutions with more than 2 spatial dimensions, for which efficient estimation of spectral properties is much more time-consuming.

2 Related work

The work [18] was the first to develop an algorithm for the exact computation of singular values for the circular convolution. In this method, the authors obtain n^2 matrices of the shape $c_{out} \times c_{in}$ after applying the Fourier transform to a zero-padded kernel. Then the SVD of n^2 matrices is computed. The complexity (see Table 1) grows polynomially with n and can be computationally demanding for datasets with higher resolutions. Recently [7] extended the method of [18] for non-circular convolutions. The authors of [19] generalized the formula proposed in [18] to convolutions with more than two dimensions and arbitrary strides.

Several works [8, 17] adapted power iteration to approximate the spectral norm of the convolution map. In this case, time complexity also depends on the input size n. The authors of [1] proposed an upper bound for the singular value of a doubly Toeplitz matrix, which can be efficiently computed, yet may be not very accurate for all filter sizes (see [6]). Works [5, 6] introduced the so-called Gram iteration with superlinear convergence as opposed to the power iteration. However, it is applied to a padded kernel and, hence, [6] has a complexity similar to [18]. Both algorithms [5,6] can also be memory consuming, for example, in the algorithm [5] the spatial size of the kernel increases almost twice every iteration.

A differentiable upper bound independent of input size was introduced in [20] for the spectral norm of circular convolutions. It is computed as the minimum spectral norm of four unfoldings of convolution kernel multiplied by a constant factor $\sqrt{hw}$. Besides being a computationally efficient upper bound, it also provides new insights into the spectral normalization of GANs [15] and the regularization of CNNs from [30]. Using the Toeplitz matrix theory, [29] proved that the bound also holds in the non-circular case (zero-padded convolutions). In this paper, we provide a new upper bound that combines the benefits of [20], while being both theoretically more precise and demonstrating noticeably higher accuracy in experiments. This new bound is based on the observation that the norm of a tensor can be bounded from above by the spectral norms of its unfoldings.

Table 1: Comparison of existing methods for computing the spectral norm of a convolution Jacobian. "Acc." stands for "accuracy", "Mem." for memory, "Diff." for "differentiable", "Ind. of n" for "independence of n". Accuracy and speed are measured experimentally. Our bound is differentiable, fast, accurate and independent of the input size n.

Method	Acc.	Mem.	Diff.	Ind. of n	Padding	Complexity ($\mathcal{O}$)
Power method [8, 17]	+	+	+	-	zero	$n^2 c_{out} c_{in} hw$
Sedghi *et al.* [18]	+	-	-	-	circular	$n^2 c_{out} c_{in} (\log n + c_{in})$
F4 (Fantastic four) [20, 29]	-	+	+	+	any	$c_{out} c_{in} hw$
LipBound [1]	-	+	+	+	zero	$c_{out} c_{in} hw$
Gram iteration [6]	+	-	+	-	circular	$n^2 c_{out} c_{in} (\log n + c_{in})$
PowerQR [7]	+	+	-	-	zero	$n^2 c_{out} c_{in} hw$
Tensor Norm (ours)	+	+	+	+	any	$c_{out} c_{in} hw$

3 Preliminaries and notation

3.1 Background on convolutions

For simplicity, let us first consider a one-dimensional convolution of a vector $X \in \mathbb{R}^n$ and a kernel vector $K \in \mathbb{R}^w$. In this paper we consider two types of convolution – with zero and circular paddings. Thanks to the linearity of convolution, it may be expressed as a matrix-vector product $Y = TX$. In the case of zero padding with an integer parameter $p \geq 0$, we have $Y_j = \sum_i K_{p+i} X_{j+i}$ and T is a matrix with constant values along diagonals – a Toeplitz matrix. In the case of circular padding, we have $Y_j = \sum_i K_{\lfloor \frac{w}{2} \rfloor + i} X_{(j+i) \mod n}$ and T is a circulant, which is a special case of a Toeplitz matrix, where each column is a cyclic permutation of a previous one. Below we give examples of T for $n = 4, w = 3, p = 1$:

$$T = \begin{bmatrix} K_1 & K_2 & 0 & 0 \\ K_0 & K_1 & K_2 & 0 \\ 0 & K_0 & K_1 & K_2 \\ 0 & 0 & K_0 & K_1 \end{bmatrix} \begin{pmatrix} \text{zero} \\ \text{padding} \end{pmatrix}, \quad T = \begin{bmatrix} K_1 & K_2 & 0 & K_0 \\ K_0 & K_1 & K_2 & 0 \\ 0 & K_0 & K_1 & K_2 \\ K_2 & 0 & K_0 & K_1 \end{bmatrix} \begin{pmatrix} \text{circular} \\ \text{padding} \end{pmatrix}.$$

Let us now consider multi-channel two-dimensional convolution with stride 1 for ease of presentation. Larger stride values will be discussed separately. Let the convolution operation be given by a kernel tensor $K \in \mathbb{R}^{c_{out} \times c_{in} \times h \times w}$ applied to an input $X \in \mathbb{R}^{c_{in} \times n \times n}$ resulting in an output $Y \in \mathbb{R}^{c_{out} \times n \times n}$. The Jacobian of the convolution can be represented as a matrix $T \in \mathbb{R}^{c_{out} n^2 \times c_{in} n^2}$ satisfying

$$vec(Y) = T vec(X), \tag{1}$$

where $vec(X)$ represents reshaping into a vector with $c_{in} n^2$ entries using colexicographic order (column-major).

Similarly to the one-dimensional case, zero padding that preserves spatial size leads to a matrix T that can be represented as a doubly block Toeplitz

matrix. In more detail, each of its blocks B_k is a block Toeplitz matrix with unstructured dense blocks $T_{k,l} = K_{:,:,k+h_1,l+w_1} \in \mathbb{R}^{c_{out} \times c_{in}}$.

$$T = \begin{bmatrix} B_0 & \dots & B_{h_2} & 0 & \dots & 0 \\ \vdots & B_0 & \ddots & \ddots & \ddots & \vdots \\ B_{-h_1} & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \ddots & \ddots & \ddots & \ddots & B_{h_2} \\ \vdots & \ddots & \ddots & \ddots & B_0 & \vdots \\ 0 & \dots & 0 & B_{-h_1} & \dots & B_0 \end{bmatrix}, B_k = \begin{bmatrix} T_{k,0} & \dots & T_{k,w_2} & 0 & \dots & 0 \\ \vdots & T_{k,0} & \ddots & \ddots & \ddots & \vdots \\ T_{k,-w_1} & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \ddots & \ddots & \ddots & \ddots & T_{k,w_2} \\ \vdots & \ddots & \ddots & \ddots & T_{k,0} & \vdots \\ 0 & \dots & 0 & T_{k,-w_1} & \dots & T_{k,0} \end{bmatrix}$$

where h_1, h_2 and w_1, w_2 subject to $h = h_1 + h_2 + 1, w = w_1 + w_2 + 1$ depend on the padding size in height and width. Depending on the order of vectorization, T may be either doubly block Toeplitz (as described above) or multi-block doubly Toeplitz. Nevertheless, singular values of the Jacobian remain the same, see, e.g. [29, Lemma 1].

Another way to maintain the input size is by using circular padding. In this case the layer "wraps around" and takes pixels from the opposite side of the image when the kernel gets beyond the edges. The corresponding Jacobian is a doubly block circulant matrix. It is a special case of a doubly block Toeplitz matrix with the entries satisfying $C_{k,l} = C_{(n-l) \bmod n,\ (n-k) \bmod n}$.

3.2 Matrix and tensor norms

Let us introduce several definitions from multilinear algebra. First, we define the Frobenius inner product for the d-dimensional tensors $A, B \in \mathbb{C}^{n_1 \times \dots \times n_d}$ and the Frobenius norm:

$$\langle A, B \rangle_F = \sum_{i_1, \dots i_d} \overline{A}_{i_1, \dots i_d} B_{i_1, \dots i_d}, \quad \|A\|_F = \sqrt{\langle A, A \rangle_F}.$$

Given also vectors $u_i \in \mathbb{C}^{n_i}$, we introduce the following multilinear functional:

$$[\![A; u_1, u_2 \dots u_d]\!] = \sum_{i_1, \dots i_d} A_{i_1, \dots i_d} u_{1_{i_1}} u_{2_{i_2}} \dots u_{d_{i_d}}. \tag{2}$$

Spectral (second) norm of a matrix $A \in \mathbb{C}^{n_1 \times n_2}$ can be defined using this notation [11]:

$$\|A\|_2 = \sup_{\substack{u_i \in \mathbb{C}^{n_i} : \|u_i\|_2 = 1 \\ i=1,2}} |u_1^T A u_2| = \sup_{\substack{u_i \in \mathbb{C}^{n_i} : \|u_i\|_2 = 1 \\ i=1,2}} |[\![A; u_1, u_2]\!]|. \tag{3}$$

It is also well-known that $\|A\|_2$ equals to the largest singular value $\sigma_1(A)$ and the vectors u_1, u_2 on which the equality is attained are respectively the left and right singular vectors of A. Both the largest singular value and the respective singular vectors can be computed using, e.g., the power method.

Spectral norm (3) naturally extends from matrices to tensors with more than two dimensions [13]. For a d-dimensional tensor $A \in \mathbb{C}^{n_1 \times \cdots \times n_d}$, it is defined as a norm of a multilinear functional (2):

$$\|A\|_\sigma = \sup_{\substack{u_i \in \mathbb{C}^{n_i} : \|u_i\|_2 = 1 \\ i=1,\ldots,d}} |[\![A; u_1, u_2 \ldots u_d]\!]|. \tag{4}$$

We will further mean that the supremum above is taken over complex vectors u_i even for real A, which may be different from supremum over real u_i, see Appendix B for details. We use the notation $\|A\|_2$ for matrices and $\|A\|_\sigma$ for d-dimensional tensors when $d > 2$.

Our main result will be formulated in terms of the norm (4). We note that this expression also defines the largest singular value $\sigma_1(A)$ of a tensor A and is associated with the best rank-1 approximation problem [13]:

$$\inf_{\substack{\sigma \in \mathbb{R}_+, \|u_i\|_2 = 1 \\ i=1,\ldots,d}} \|A - \sigma\, u_1 \circ \cdots \circ u_d\|_F, \tag{5}$$

which admits the solution $\sigma = \sigma_1(A)$ ($\circ$ denotes tensor product). As we will discuss in more detail later, there is an analog of the power method called HOPM (Algorithm 1) that can be used to solve the best rank-1 approximation problem and calculate the spectral norm of a tensor.

3.3 Tensor unfoldings

We call a matrix

$$A_{(i_1 \ldots i_k; j_1 \ldots j_{d-k})} \in \mathbb{R}^{\left(n_{i_1} \cdot \ldots \cdot n_{i_k}\right) \times \left(n_{i_k} \cdot \ldots \cdot n_{j_{d-k}}\right)}$$

an unfolding of a tensor $A \in \mathbb{R}^{n_1 \times \cdots \times n_d}$, if it is obtained by first permuting indices of the tensor and then by reshaping the tensor into a matrix in a column-major order. In a Python numpy-like notation it reads as:

$$A = A.\texttt{transpose}(i_1, \ldots, i_k, j_1, \ldots, j_{d-k}),$$
$$A_{(i_1, i_2 \ldots i_k; j_1, \ldots j_{d-k})} = A.\texttt{reshape}(n_{i_1} \ldots n_{i_k}, n_{j_1} \ldots n_{j_{d-k}}, \texttt{ order='f'}).$$

The following lemma establishes a connection between the spectral norm of a tensor and the spectral norm of its unfoldings. We will later use this result to prove the lower bound on $\|T\|_2$ and to demonstrate how our result relates to [20].

Lemma 1 (Prop. 4.1, [27]). *$\|K\|_\sigma \le \|R\|_2$ for any unfolding matrix R of the tensor K.*

3.4 Spectral density matrix

The approach we use to prove the main result of this paper is based on the following techniques. Following [29, Theorem 1], for a doubly block Toeplitz

matrix $T \in \mathbb{R}^{c_{out}n^2 \times c_{in}n^2}$ with the blocks $T_{k,l} \in \mathbb{R}^{c_{out} \times c_{in}}$, we consider a matrix-valued generating function $F\colon [-\pi, \pi]^2 \to \mathbb{C}^{c_{out} \times c_{in}}$ such that

$$T_{k,l} = \frac{1}{(2\pi)^2} \int_{-\pi}^{\pi} \int_{-\pi}^{\pi} F(\tau_1, \tau_2) e^{-i(k\tau_1 + l\tau_2)} d\tau_1 d\tau_2.$$

The generating function can be explicitly written as [29]:

$$F(\tau_1, \tau_2) = \sum_{k=-h_1}^{h_2} \sum_{l=-w_1}^{w_2} T_{k,l} e^{i(k\tau_1 + l\tau_2)},$$

which is a generalization of [23, 24]. The function $F(\tau_1, \tau_2)$ is also called the spectral density matrix. The spectral norm of $F(\tau_1, \tau_2)$ is defined [23, 29] as

$$\|F\|_2 = \sup_{\tau_1, \tau_2 \in [-\pi, \pi]} \|F(\tau_1, \tau_2)\|_2.$$

Lemma 2 (Lemma 4, [29]). $\|T\|_2 \leq \|F\|_2$.

The set of singular values of circular convolution is obtained as $\sigma_j(F(\tau_1, \tau_2))$ with $(\tau_1, \tau_2) = (-\pi + \frac{2\pi j_1}{n}, -\pi + \frac{2\pi j_2}{n}), \forall j_1, j_2 \in [n] - 1$ [29]. However, by contrast to the circular case, for the zero-padded convolution the values of τ_1, τ_2 that lead to the singular values are not known a priori.

4 Main results

The Lipschitz constant of a neural network f with respect to the Euclidean norm for an input space $\mathcal{X}$ is defined as

$$\mathrm{Lip}(f) = \sup_{x, x' \in \mathcal{X}, x \neq x'} \frac{\|f(x) - f(x')\|_2}{\|x - x'\|_2}.$$

It determines, e.g., how much the output value of the network changes relative to input perturbations. Computation of the Lipschitz constant of the whole network f is a challenging task [25]. However, assuming that f is a composition of maps ϕ_i:

$$f = \phi_\ell \circ \cdots \circ \phi_1,$$

it can be bounded from above with the product of the Lipschitz constants of individual layers:

$$\mathrm{Lip}(f) \leq \prod_{i=1}^{\ell} \mathrm{Lip}(\phi_i). \tag{6}$$

Lipschitz constant of a linear layer $\phi(x) = Wx + b$ has the explicit representation:

$$\mathrm{Lip}(\phi) = \|W\|_2 = \sigma_1(W).$$

In the case of a convolutional layer, we have $W = T$ from (1). Naive computation of its largest singular value by computing SVD appears to be intractable, as such a matrix does not even fit into GPU memory for practical sizes of layers and the computational complexity grows cubically with its size. Alternative iterative approaches, e.g., the power method that take the structure of T into account require multiple applications of convolution, so their complexity depends on input image shape n.

In this work, we take benefit of the matrix structure and obtain an upper bound that is independent of n, valid for any padding and can be efficiently computed in practice. This bound is formulated in the next theorem.

Theorem 1. *Let $T \in \mathbb{R}^{c_{out}n^2 \times c_{in}n^2}$ be the Jacobian of a convolutional layer with stride $s = 1$ and zero padding with the parameter $p \geq 0$ or circular padding. Let $K \in \mathbb{R}^{c_{out} \times c_{in} \times h \times w}$ be the convolution kernel. Then*

$$\|K\|_\sigma \leq \|T\|_2 \leq \sqrt{hw}\|K\|_\sigma.$$

Proof. Firstly, let us prove the upper bound. We can rewrite $F(\tau_1, \tau_2)$ as a convolution of the kernel with vectors z_1, z_2:

$$F(\tau_1, \tau_2) = \sum_{k=-h_1}^{h_2} \sum_{l=-w_1}^{w_2} K_{:,:,h_1+k,w_1+l} e^{i(k\tau_1 + l\tau_2)} = [\![K; I_{c_{out}}, I_{c_{in}}, z_1, z_2]\!],$$

$$z_1 = [e^{-h_1(i\tau_1)}, e^{(-h_1+1)(i\tau_1)} \dots e^{h_2(i\tau_1)}], z_2 = [e^{-w_1(i\tau_2)}, e^{(-w_1+1)(i\tau_2)} \dots e^{w_2(i\tau_2)}]$$

and for the matrices in square brackets the summation is done along their last indices. For complex vectors u_i, we have,

$$\begin{aligned}
&\|F\|_2 = \sup_{\substack{\|u_i\|_2=1 \\ i=1,2}} |u_1^T F u_2| = \sup_{\substack{\|u_i\|_2=1 \\ i=1,2}} |[\![F; u_1, u_2]\!]| = \sup_{\substack{\|u_i\|_2=1 \\ i=1,2}} |[\![K; u_1, u_2, z_1, z_2]\!]| = \\
&= \sup_{\substack{\|u_i\|_2=1 \\ i=1,2}} \left| \sqrt{hw} \left[\!\!\left[K; u_1, u_2, \frac{z_1}{\sqrt{h}}, \frac{z_2}{\sqrt{w}} \right]\!\!\right] \right| \leq \sup_{\substack{\|u_i\|_2=1 \\ i=1,\dots,4}} \left| \sqrt{hw} [\![K; u_1, u_2, u_3, u_4]\!] \right| = \\
&= \sqrt{hw}\, \|K\|_\sigma.
\end{aligned} \tag{7}$$

Thus, $\|T\|_2 \leq \sqrt{hw}\|K\|_\sigma$.

Let us consider an unfolding $R = K_{(1,234)}$. Note that it is a submatrix of a doubly Toeplitz matrix T up to a permutation of columns. In particular, we can obtain R by choosing a block row in T which starts with $T_{-h_1,-w_1}$ and excluding zero entries from it, see Sec. 3.1. Since the spectral norm of a matrix upper bounds spectral norm of any of its submatrices, we have

$$\|K\|_\sigma \leq \|R\|_2 = \|K_{(1,234)}\|_2 \leq \|T\|_2,$$

which completes the proof.

Remark 1. Note that the matrix F and the vectors z_1, z_2 from (7) are complex. Therefore, all suprema in this context are taken over complex vectors u_i. As a result, we need a complex rank-1 approximation of the real kernel K to find $\|K\|_\sigma$ when applying HOPM Algorithm 1. This is because complex and real spectral norms of a real tensor do not always coincide [9, 10]. In Appendix B we provide an example of a tensor for which $\sqrt{hw}\|K\|_\sigma$ with supremum taken over real vectors does not bound from above the spectral norm of a convolution.

Remark 2. As a direct corollary of Theorem 1, the bounds are exact for $h = w = 1$, meaning that $\|T\|_2 = \|K\|_\sigma$.

Note that Theorem 1 also suggests that in the worst case, our TN bound does not overestimate the exact singular value of the convolution Jacobian by more than $\sqrt{hw}$ times, i.e., $\sqrt{hw}\|K\|_\sigma \leq \sqrt{hw}\|T\|_2$.

Let us now discuss how our bound compares to [20, 29]. The work [20] proposed to bound the singular value of circular convolution using the minimum of spectral norms of four unfoldings of the kernel:

$$\|T\|_2 \leq \sqrt{hw}\min\left(\|K_{(13;24)}\|_2, \|K_{(14,23)}\|_2, \|K_{(1;234)}\|_2, \|K_{(2;134)}\|_2\right). \quad (8)$$

Lemma 1 shows that the tensor spectral norm never exceeds the norm of any of its unfoldings, including the ones that are not present in (8). Therefore, our bound is provably more accurate than the one in [20].

Theorem 1 naturally extends to convolutions with any number of dimensions.

Theorem 2. *Let us consider convolution with a kernel* $K \in \mathbb{R}^{c_{out}\times c_{in}\times h_1\times\ldots\times h_d}$, $d \in \mathbb{N}_+$, *with arbitrary padding and stride* $s = 1$. *Then*

$$\|K\|_\sigma \leq \|T\|_2 \leq \sqrt{h_1\ldots h_d}\|K\|_\sigma.$$

Proof. The proof is provided in Appendix D.

The following theorem extends Theorem 1 to the case of strided convolutions.

Theorem 3. *Let* $T_s \in \mathbb{R}^{c_{out}\frac{n^2}{s^2}\times c_{in}n^2}$ *be the Jacobian of a convolution with the kernel* $K \in \mathbb{R}^{c_{out}\times c_{in}\times h\times w}$ *and stride* s. *Then*

$$\|Q\|_\sigma \leq \|T_s\|_2 \leq \sqrt{\left\lceil\frac{h}{s}\right\rceil\left\lceil\frac{w}{s}\right\rceil}\|Q\|_\sigma,$$

where $Q \in \mathbb{R}^{c_{out}\times c_{in}s^2\times\lceil\frac{h}{s}\rceil\times\lceil\frac{w}{s}\rceil}$ *is padded with zeros and reshaped kernel* K

$$K_{c,d,a,b} = Q_{c,ds^2+s(a \pmod s)+b \pmod s,\lfloor\frac{a}{s}\rfloor,\lfloor\frac{b}{s}\rfloor}.$$

Proof. The proof is provided in Appendix A.1.

5 Computation of the spectral norm

The power method is a standard algorithm for the computation of the largest singular value and the respective singular vector of a matrix. Works [3, 4] extended the power method from matrices to tensors of higher dimension. The algorithm starts with either randomly initialized vectors $u_1, u_2, \ldots, u_d$, or it can start with a good approximation to the singular vectors. For example, in our case, the kernel weights change slowly during training, so we may utilize vectors from previous iterations. The i-th substep, $i = 1, \ldots, d$ of an iteration is equivalent to the minimization of the functional from (5) with respect to a single u_i. This leads to simple formulas with contractions, see Algorithm 1. Note that according to Remark 1, we need to utilize complex vectors u_i, which is why we have the complex conjugate operation $\text{conj}(u_i)$. The operations in HOPM can be reorganized to speed up computations for the certain hardware. In our repository with code, you can find several implementations of HOPM that have proven to be faster in practice when running on either CPU or GPU. It is also advantageous to rerun the HOPM algorithm from the beginning several times if no good initial guess is available. This will help ensure convergence to the global optimum. Other initialization strategies also exist, see [4].

Algorithm 1 Higher-Order Power Method (HOPM)

Input kernel: $K \in \mathbb{R}^{c_{out} \times c_{in} \times h \times w}$; number of iterations: `n_iters`; initial unit vectors: $u_1 \in \mathbb{C}^{c_{out}}$, $u_2 \in \mathbb{C}^{c_{in}}$, $u_3 \in \mathbb{C}^{h}$, $u_4 \in \mathbb{C}^{w}$.
Return $\|K\|_\sigma$
for $1 \ldots$ `n_iters` **do**
  $u_1 = [\![K; I, u_2, u_3, u_4]\!]$, $u_1 = \text{conj}(u_1)/\|u_1\|$ ▷ $\mathcal{O}(c_{out}c_{in}hw)$
  $u_2 = [\![K; u_1, I, u_3, u_4]\!]$, $u_2 = \text{conj}(u_2)/\|u_2\|$ ▷ $\mathcal{O}(c_{out}c_{in}hw)$
  $u_3 = [\![K; u_1, u_2, I, u_4]\!]$, $u_3 = \text{conj}(u_3)/\|u_3\|$ ▷ $\mathcal{O}(c_{out}c_{in}hw)$
  $u_4 = [\![K; u_1, u_2, u_3, I]\!]$, $u_4 = \text{conj}(u_4)/\|u_4\|$ ▷ $\mathcal{O}(c_{out}c_{in}hw)$
end for
return $|[\![K; u_1, u_2, u_3, u_4]\!]|$ ▷ $\mathcal{O}(c_{out}c_{in}hw)$

Note that, although the algorithm involves complex vectors, the norm $\|K\|_\sigma$ itself is a real number. The following proposition provides explicit formulas for the norm and its gradient, which avoid complex arithmetic calculations.

Proposition 1. *Let u_1, u_2, u_3, u_4 be the complex singular vectors corresponding to $\|K\|_\sigma$. Let $u_j = a_j + ib_j$, where a_j and b_j are the real and imaginary parts. The gradient of the bound $\sqrt{hw}\|K\|_\sigma$ with respect to the kernel is computed as*

$$\nabla_K \sqrt{hw}\|K\|_\sigma = \sqrt{hw}\nabla_K \sqrt{real^2 + im^2} = \frac{\sqrt{hw}}{\|K\|_\sigma}\left(real \nabla_K real + im \nabla_K im\right),$$

where real and im are the real and imaginary parts of $[\![K; u_1, u_2, u_3, u_4]\!]$. For part $\in \{real, im\}$, we can write it and its gradient in terms of predefined tensors

$P_{real}, P_{im} \in \{-1, 0, 1\}^{2\times2\times2\times2}$ *(see Appendix C for their explicit representation):*

$$\begin{aligned} part &= \langle P_{part}, [\![K;\, [a_1, b_1], [a_2, b_2], [a_3, b_3], [a_4, b_4]]\!] \rangle_F, \\ \nabla_K part &= [\![P_{part}; [a_1, b_1]^T, [a_2, b_2]^T, [a_3, b_3]^T, [a_4, b_4]^T]\!] \,. \end{aligned} \tag{9}$$

6 Experiments

The source code is available at `https://github.com/GrishKate/conv_norm`.

6.1 Spectral norm computation

Table 2 compares our TN bound with the closest competitor, "fantastic four" bound [20] ($F4$ for short), in terms of precision. Values in each of the kernels are sampled from $\mathcal{N}(0, 1)$. We observe that TN bound produces close to exact values, whereas $F4$ exceeds exact spectral norm 1.7-2.6 times. Results for strided convolutions are presented in Appendix A.2.

Table 3 shows that the TN bound provides an accurate approximation for the spectral norm of convolutional layers for the pre-trained ResNet18. Our bound performs systematically better for most layers compared to the $F4$ bound.

Figure F.4 in Appendix demonstrates comparison with other existing methods in terms of precision, memory consumption and time performance. Although methods [6,18] obtain a tight upper bound, they require significantly more memory than the other approaches. Algorithms based on power iteration [7, 17] are highly accurate, but their time and memory consumption grows with the input size n. While the bounds [1, 20] are independent of n, they are noticeably less accurate. Our TN bound does not depend on n and therefore is both memory efficient and fast to compute, while being more precise than [1, 20].

6.2 Spectral norm regularization

Following [20], we apply the TN bound for regularization and study its effect on image classification accuracy. We train ResNet18 on CIFAR100 and ResNet34 on ImageNet for 90 epochs. We use weight decay $1e$-4, SGD with momentum 0.9 and initial learning rate of 0.1, which is reduced 10 times every 30 epochs. We apply the sum of estimates of the spectral norms of all convolutional layers of the network as a regularizer. The objective loss function becomes:

$$\mathcal{L} = \mathcal{L}_{train} + \beta \sum_i \sigma_i,$$

where β denotes the regularization coefficient, $\mathcal{L}_{train}$ is cross-entropy loss, σ_i denotes the bound on the largest singular value of the i^{th} convolutional layer computed with different methods. During the epoch, one iteration step is sufficient to update σ_i and the singular vectors. However, for reliability, we recompute singular vectors every epoch from random initialization. We obtain the highest

Table 2: Comparison of the TN bound and the $F4$ bound for Gaussian kernels. Reference values for spectral norm of zero-padded convolution $\|T\|_2$ were computed with power method [17] for the image size 32×32. We used 100 iterations for all methods, stride equals to 1.

Kernel size	$\|T\|_2$ (Reference)	$F4$	TN (Ours)	$\frac{F4}{\|T\|_2}$	$\frac{TN}{\|T\|_2}$
64, 64, 3, 3	49.35	81.26	51.51	1.646	**1.044**
128, 128, 3, 3	67.14	114.68	69.99	1.708	**1.042**
256, 256, 3, 3	95.71	164.44	96.47	1.718	**1.008**
512, 512, 3, 3	135.62	234.55	136.99	1.729	**1.01**
64, 64, 5, 5	81.55	175.74	88.25	2.155	**1.082**
128, 128, 5, 5	113.5	250.91	119.29	2.211	**1.051**
256, 256, 5, 5	160.07	354.46	165.37	2.214	**1.033**
512, 512, 5, 5	227.1	501.8	229.66	2.21	**1.011**
64, 64, 7, 7	112.94	293.34	127.69	2.597	**1.131**
128, 128, 7, 7	158.36	415.81	170.99	2.626	**1.08**
256, 256, 7, 7	222.45	587.98	235.28	2.643	**1.058**
512, 512, 7, 7	315.25	834.35	326.2	2.647	**1.035**

Table 3: Comparison of the TN bound and the $F4$ bound for kernels from ResNet18 pretrained on ImageNet. We use 100 iterations for all methods. For layers with $h = w = 1$ both bounds give an exact value, as proved in Remark 2. The $F4$ bound does not take the stride into account, which results in even looser bound for strided convolution (see the first row, where the ratio $F4/\|T\|_2 = 3.5$).

Kernel size	Stride	$\|T\|_2$ (Exact)	$F4$	TN (Ours)	$\frac{F4}{\|T\|_2}$	$\frac{TN}{\|T\|_2}$
64, 3, 7, 7	2	8.2	28.89	14.91	3.526	**1.819**
64, 64, 3, 3	1	5.99	9.33	8.56	1.558	**1.43**
64, 64, 3, 3	1	5.3	6.3	5.36	1.187	**1.01**
64, 64, 3, 3	1	6.95	8.71	8.63	1.253	**1.242**
64, 64, 3, 3	1	3.8	5.4	4.04	1.42	**1.062**
128, 64, 3, 3	2	2.8	6.01	3.16	2.149	**1.129**
128, 128, 3, 3	1	5.69	7.21	6.67	1.267	**1.173**
128, 64, 1, 1	2	1.91	1.91	1.91	1.0	**1.0**
128, 128, 3, 3	1	4.38	6.78	5.46	1.549	**1.247**
128, 128, 3, 3	1	4.86	7.57	6.07	1.558	**1.248**
256, 128, 3, 3	2	3.95	8.45	4.32	2.14	**1.093**
256, 256, 3, 3	1	6.45	8.04	7.07	1.246	**1.096**
256, 128, 1, 1	2	1.22	1.22	1.22	1.0	**1.0**
256, 256, 3, 3	1	6.23	7.57	6.43	1.216	**1.033**
256, 256, 3, 3	1	7.53	9.17	8.25	1.218	**1.095**
512, 256, 3, 3	2	5.53	11.0	5.99	1.989	**1.082**
512, 512, 3, 3	1	8.38	10.45	9.54	1.247	**1.139**
512, 256, 1, 1	2	2.05	2.05	2.05	1.0	**1.0**
512, 512, 3, 3	1	16.26	18.37	17.95	1.13	**1.104**
512, 512, 3, 3	1	6.8	7.6	7.52	1.117	**1.106**

accuracy with $\beta = 0.0022$ for CIFAR100 and $\beta = 1e\text{-}4$ for ImageNet. The results of hyperparameter tuning are presented in Table E.2 in Appendix. We compare the effect of regularization both with and without weight decay for CIFAR100. Table 4 suggests that TN reduces generalization error and gives systematically better results than $F4$.

Table 4: Test accuracy with different regularizers.

Method	Acc. w/o wd	Acc. w/ wd
Baseline	73.10	73.84
$F4$	**73.96**	74.91
TN (Ours)	**73.96**	**74.99**

(a) ResNet18 trained on CIFAR100.

Method	Acc@1	Acc@5
Baseline	73.368	**91.438**
$F4$	73.388	91.300
TN (Ours)	**73.510**	91.420

(b) ResNet34 trained on ImageNet.

Training time comparison is presented in Table 5. The wall clock time difference between training with and without the TN regularization is almost negligible. The time performance of TN is comparable with $F4$ [20]. Methods of [6, 18] require much more memory and cause "Out of Memory" error for large input sizes. The singular value clipping method [7] is computationally expensive and is applied only once per 50 steps. The advantage of TN over the Gram iteration-based approach [6] is that TN bound can be updated with one iteration of HOPM during training, while the Gram iteration needs to start from scratch every time. Hence, Gram iteration increases time of training multiple times, e.g., see Section 5.5 in [6], where time per epoch increased by a factor of 3. By contrast, with one iteration per training step, TN bound can be used as a computationally efficient regularization with a negligible increase in time and memory.

Table 5: Average time per epoch for training ResNet18 on CIFAR100 with different regularizers. For computing LipBound [1] we use 10 samples. We present time for clipping once per 50 iterations with PowerQR [7].

Method	Baseline	TN (Ours)	$F4$	LipBound	PowerQR	Gram iter.	Sedghi
Time (s)	141.0	143.6	143.7	242.5	149.6	OOM	OOM

Figure 1 (left) presents a comparison of our TN bound with $F4$. Throughout training, the TN bound appears to be more precise. Regularization with the TN bound significantly decreases the singular value of convolutions and makes the bound more precise (middle). One may wonder how regularization affects the Lipschitz constant of the entire network. The right figure suggests that regularization reduces the norm of composition of convolution with the subsequent BatchNorm, meaning that the upper bound (6) on the Lipschitz constant of the

whole network declines. The same effect is observed for all layers, see Figures F.1, F.2, F.3.

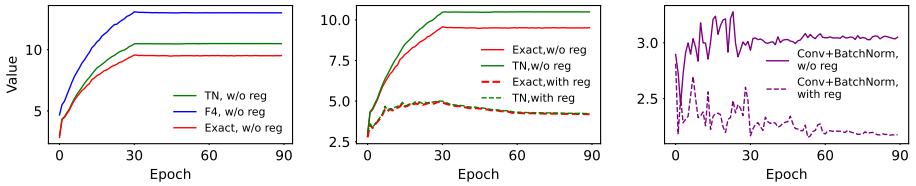

Fig. 1: The behaviour of spectral norm bounds for the third layer of ResNet-18 during training on CIFAR100. The left figure compares tightness of TN and $F4$ bounds when training without regularization. The middle figure shows the effect of training with and without TN regularization. The right one demonstrates the influence of regularization on the spectral norm of composition of convolution and the subsequent BatchNorm layer. Similar plots for all layers are presented in Figures F.1, F.2, F.3 in Appendix.

6.3 Orthogonal regularization

The work [26] proposes to use orthogonal regularization for training CNNs. The method utilizes convolution of the kernel with itself to estimate the divergence of the layer's Jacobian from orthogonal in the Frobenius norm, which leads to the following loss function:

$$\mathcal{L}_{OCNN} = \|Conv(K, K) - I\|_F$$

However, using the spectral norm of the Jacobian may seem more natural in this case, as it is an operator norm. Let T be the Jacobian of the circular convolution with a kernel K. It is known that the product of two circulant matrices is also a circulant, thus $T^T T$ is also a Jacobian of some convolution. Theorem 1 suggests that $\|T^T T - I\|_2 \leq \sqrt{h'w'}\|Conv(K, K, \texttt{padding=``circular''}) - I\|_\sigma$. Thus, the following regularizer penalizes the largest difference between the singular values of the convolution and 1:

$$\mathcal{L}_{2norm} = \|Conv(K, K, \texttt{padding=``circular''}) - I\|_\sigma$$

We introduce one more regularizer based on the spectral norm of the kernel. As is known, the squared Frobenius norm of a matrix is equal to the sum of squares of its singular values and the ratio $\frac{\|A\|_2}{\|A\|_F}$ is minimized when all the singular values of matrix A are equal. The Frobenius norm of the kernel can be used as approximation for the Frobenius norm of convolution Jacobian up to a constant factor, e.g., in the case of circular padding $n\|K\|_F = \|T\|_F$. Hence, we propose

$$\mathcal{L}_{Ratio} = \frac{\sqrt{hw}\|K\|_\sigma}{\|K\|_F}.$$

We train Resnet18 on CIFAR100 dataset with the same hyperparameters as in the previous section and weight decay 1e-4. The regularized objective loss function is given as follows:

$$\mathcal{L} = \mathcal{L}_{train} + \beta \sum_i \mathcal{L}^i_{reg},$$

where $\mathcal{L}^i_{reg}$ denotes one of the regularization losses for the i^{th} layer described above. The results are reported in Table 6. Our regularizers improve generalization performance. $\mathcal{L}_{2norm}$ yields the best top-1 accuracy gain of more than 2%.

Table 6: ResNet18 CIFAR100 accuracy and average training time per epoch with different orthogonal regularizers.

Method	β	Acc@1	Acc@5	Time (s)
Baseline	0	73.84	93.16	141.0
OCNN	0.1	75.32	93.45	150.1
OCNN	0.01	75.32	93.30	150.1
Ratio (Ours)	1.0	74.98	93.50	147.0
Ratio (Ours)	0.1	74.71	93.52	147.0
Ratio (Ours)	0.01	74.71	93.45	147.0
2norm (Ours)	1e-2	74.43	93.47	171.8
2norm (Ours)	5e-3	**75.99**	**94.08**	171.8
2norm (Ours)	1e-3	75.83	93.92	171.8

7 Conclusion

We propose to use the spectral norm of the kernel tensor to bound the spectral norm of convolutional layers. Our bound noticeably improves the accuracy of existing upper bounds that are independent of the resolution of input images. It also provides a trade-off between accuracy and computation efficiency compared with the other methods. We demonstrate that this bound can be used as a regularizer for improving generalization of NNs. Furthermore, we propose two new regularizers based on it that enforce orthogonality of convolutions.

Acknowledgments

The publication was supported by the grant for research centers in the field of AI provided by the Analytical Center for the Government of the Russian Federation (ACRF) in accordance with the agreement on the provision of subsidies (identifier of the agreement 000000D730321P5Q0002) and the agreement with HSE University №70-2021-00139. The calculations were performed in part through the computational resources of HPC facilities at HSE University [12].

References

1. Araujo, A., Negrevergne, B., Chevaleyre, Y., Atif, J.: On lipschitz regularization of convolutional layers using toeplitz matrix theory. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 35, pp. 6661–6669 (2021)
2. Cisse, M., Bojanowski, P., Grave, E., Dauphin, Y., Usunier, N.: Parseval networks: Improving robustness to adversarial examples. In: International conference on machine learning. pp. 854–863. PMLR (2017)
3. De Lathauwer, L., Comon, P., De Moor, B., Vandewalle, J.: Higher-order power method. Nonlinear Theory and its Applications, NOLTA95 **1**, 4 (1995)
4. De Lathauwer, L., De Moor, B., Vandewalle, J.: On the best rank-1 and rank-$(r_1, r_2, \ldots, r_n)$ approximation of higher-order tensors. SIAM journal on Matrix Analysis and Applications **21**(4), 1324–1342 (2000)
5. Delattre, B., Barthélemy, Q., Allauzen, A.: Spectral norm of convolutional layers with circular and zero paddings. arXiv preprint arXiv:2402.00240 (2024)
6. Delattre, B., Barthélemy, Q., Araujo, A., Allauzen, A.: Efficient bound of lipschitz constant for convolutional layers by gram iteration. In: International Conference on Machine Learning. pp. 7513–7532. PMLR (2023)
7. Ebrahimpour-Boroojeny, A., Telgarsky, M., Sundaram, H.: Spectrum extraction and clipping for implicitly linear layers. In: NeurIPS 2023 Workshop on Mathematics of Modern Machine Learning (2023)
8. Farnia, F., Zhang, J., Tse, D.: Generalizable adversarial training via spectral normalization. In: International Conference on Learning Representations (2019)
9. Friedland, S., Lim, L.H.: Nuclear norm of higher-order tensors. Mathematics of Computation **87**(311), 1255–1281 (2018)
10. Friedland, S., Wang, L.: Spectral norm of a symmetric tensor and its computation. Mathematics of Computation **89**(325), 2175–2215 (2020)
11. Golub, G.H., Van Loan, C.F.: Matrix computations. JHU press (2013)
12. Kostenetskiy, P., Chulkevich, R., Kozyrev, V.: HPC resources of the higher school of economics. In: Journal of Physics: Conference Series. vol. 1740, p. 012050 (2021)
13. Lim, L.H.: Singular values and eigenvalues of tensors: a variational approach. In: 1st IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing, 2005. pp. 129–132. IEEE (2005)
14. Meunier, L., Delattre, B.J., Araujo, A., Allauzen, A.: A dynamical system perspective for lipschitz neural networks. In: International Conference on Machine Learning. pp. 15484–15500. PMLR (2022)
15. Miyato, T., Kataoka, T., Koyama, M., Yoshida, Y.: Spectral normalization for generative adversarial networks. In: International Conference on Learning Representations (2018)
16. Neyshabur, B., Bhojanapalli, S., Srebro, N.: A pac-bayesian approach to spectrally-normalized margin bounds for neural networks. In: International Conference on Learning Representations (2018)
17. Ryu, E., Liu, J., Wang, S., Chen, X., Wang, Z., Yin, W.: Plug-and-play methods provably converge with properly trained denoisers. In: International Conference on Machine Learning. pp. 5546–5557. PMLR (2019)
18. Sedghi, H., Gupta, V., Long, P.M.: The singular values of convolutional layers. In: International Conference on Learning Representations (2018)
19. Senderovich, A., Bulatova, E., Obukhov, A., Rakhuba, M.: Towards practical control of singular values of convolutional layers. Advances in Neural Information Processing Systems **35**, 10918–10930 (2022)

20. Singla, S., Feizi, S.: Fantastic four: Differentiable and efficient bounds on singular values of convolution layers. In: International Conference on Learning Representations (2020)
21. Singla, S., Feizi, S.: Skew orthogonal convolutions. In: International Conference on Machine Learning. pp. 9756–9766. PMLR (2021)
22. Singla, S., Singla, S., Feizi, S.: Improved deterministic l2 robustness on CIFAR-10 and CIFAR-100. In: International Conference on Learning Representations (2021)
23. Tilli, P.: Singular values and eigenvalues of non-hermitian block toeplitz matrices. Linear algebra and its applications **272**(1-3), 59–89 (1998)
24. Tyrtyshnikov, E.E.: A unifying approach to some old and new theorems on distribution and clustering. Linear algebra and its applications **232**, 1–43 (1996)
25. Virmaux, A., Scaman, K.: Lipschitz regularity of deep neural networks: analysis and efficient estimation. Advances in Neural Information Processing Systems **31** (2018)
26. Wang, J., Chen, Y., Chakraborty, R., Yu, S.X.: Orthogonal convolutional neural networks. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 11505–11515 (2020)
27. Wang, M., Duc, K.D., Fischer, J., Song, Y.S.: Operator norm inequalities between tensor unfoldings on the partition lattice. Linear algebra and its applications **520**, 44–66 (2017)
28. Xiao, L., Bahri, Y., Sohl-Dickstein, J., Schoenholz, S., Pennington, J.: Dynamical isometry and a mean field theory of cnns: How to train 10,000-layer vanilla convolutional neural networks. In: International Conference on Machine Learning. pp. 5393–5402. PMLR (2018)
29. Yi, X.: Asymptotic spectral representation of linear convolutional layers. IEEE Transactions on Signal Processing **70**, 566–581 (2022)
30. Yoshida, Y., Miyato, T.: Spectral norm regularization for improving the generalizability of deep learning. stat **1050**, 31 (2017)