Analysis of the Stable Rank Normalization for Convolutional Kernels

Zhe Zheng†, Valéry Dewil†, Gabriele Facciolo†, Pablo Arias‡

† Université Paris-Saclay, CNRS, ENS Paris-Saclay, Centre Borelli, 91190 Gif-sur-Yvette, France

‡ University Pompeu Fabra, Department of Engineering, Barcelona, Spain

Résumé – La normalisation du rang stable des couches linéaires dans les réseaux neuronaux a récemment été associée à leur capacité de généralisation ainsi qu'à la stabilité des réseaux de neurones récurrents. Cela a été appliqué aux couches non convolutionnelles. Cependant, l'application du SRN aux réseaux convolutionnels n'est pas triviale, car elle nécessite de calculer les vecteurs singuliers dominants de la matrice circulante associée au noyau. Dans ce travail, nous passons en revue les approches précédentes en soulignant certaines limitations, et nous proposons une analyse du problème, montrant qu'il est faisable pour certaines valeurs du rang stable cible. Enfin, nous proposons une approximation du problème sous la forme d'une minimisation sur une sphère.

Abstract – The normalization of the stable rank of linear layers in neural networks has been linked recently with their ability to generalize and with the stability of recurrent neural networks. This has been applied in practice for non-convolutional layers. However, the application of SRN to convolutional networks is not straightforward, as it requires computing the leading singular vectors of the circulant matrix associated to the convolution kernel. In this work we review previous approaches pointing out some limitations, and propose an analysis of the problem, showing that it is feasible for certain values of the target stable rank. Finally, we propose an approximation of the problem as a minimization on a sphere.

1 Introduction

Stable rank normalization has recently gained attention in deep learning for controlling the behavior of neural networks , such as improving their generalization ability [4, 6] and stability [1, 7]. The stable rank was first associated with the generalization error bound for neural networks in [4]. The authors in [6] proposed normalizing the stable rank as a way to control the network generalization, and found an explicit optimal solution to the Stable Rank Normalization (SRN) problem. The SRN algorithm was then used for the linear layers of neural networks, together with Spectral Normalization (SN) [3], and it was empirically shown that controlling the stable rank and operator norm of each layer of the network improves generalization behavior and classification performance [6].

Controlling the stable rank also benefits the stability of recurrent *convolutional* neural networks, which are often used in video processing tasks such as denoising and super-resolution [1, 7]. Recurrent neural networks are prone to instabilities during inference, which can result in divergence for long signals [7] where some pixel values blow up, despite the model's stable behavior on short training sequences. It is argued that layers with smaller stable rank lead to better stability at inference time [7].

The SRN algorithm was originally only used for linear layers, because the size of the underlying linear mapping for a convolutional layer can be intractable. The authors in [7] proposed to adapt SRN for linear layers to convolutional layers (named SRN-C), leveraging the equivalence between matrix multiplication and kernel convolution [2]. However, this adaptation is not as straightforward as presented. In this paper, we first show that the stable rank property is not preserved by the original SRN-C algorithm and analyze in detail the stable rank normalization problem for convolutional kernels. We show that, with a correctly specified desired stable rank value, the problem is solvable.

In the rest of the paper we first introduce the related concepts to SRN, analyze the original SRN-C algorithm, leading to a reformulation of the problem, which we study in detail.

2 Stable Rank Normalization

Stable Rank of a matrix. The stable rank of a matrix W is a continuous, scale-invariant relaxation of the standard matrix rank [6, 5], defined as the ratio of the squared Frobenius norm to the squared spectral norm:

$$\text{srank}(W) = \frac{\|W\|_F^2}{\|W\|_2^2} = \frac{\sum_{i=1}^{p} \sigma_i^2(W)}{\sigma_1^2(W)}, \tag{1}$$

where p is the rank of W, and σ_i are the singular values of W.

Stable Rank Normalization. The stable rank normalization (SRN) problem is stated as follows [6], given a matrix $W \in \mathbb{R}^{m\times n}$ with rank p, we look for an approximation $\widehat{W}_k$:

$$\begin{aligned} \underset{\widehat{W}_k \in \mathbb{R}^{m\times n}}{\arg\min} \quad & \left\| W - \widehat{W}_k \right\|_F^2 \\ \text{s.t.} \quad & \text{srank}(\widehat{W}_k) = r, \quad \lambda_i = \sigma_i,\ \forall i \in \{1, \cdots, k\}, \end{aligned} \tag{2}$$

where, $1 \le r < \text{srank}(W)$ is the desired stable rank, k denotes the number of singular values preserved, and λ_i and σ_i are the singular values of $\widehat{W}_k$ and W, respectively. It essentially seeks a matrix: 1) with a pre-specified stable rank value r; 2) that preserves the k largest singular values; 3) and that approximates well the given matrix W in the sense of Frobenius norm. The optimal solution for the above problem is explicitly given by [6], which we will describe in the next subsection.

This problem can be regarded as a generalization of rank-k approximation of a matrix, which approximates a given matrix W as a sum of k rank-1 matrices that are associated to the first largest singular values of W.

Lipschitz constant. The Lipschitz constant L is a key concept for stability in dynamic systems and neural networks. For a function h, the Lipschitz constant is defined as the smallest value L, s.t. $\|h(x) - h(y)\| \leq L\|x - y\|$, for any x, y in the domain of h. When $L < 1$, the function f is called contractive. For a neural network consisting of l layers with weights $W_k, k = 1, \ldots, l$ and ReLU activation function, the Lipschitz constant satisfies $L \leq \Pi_{k=1}^{l} \|W_k\|$. Prior work has proposed constraining the spectral norm of each layer to ensure the network being contractive and enforce stability [3]. However, such constraints can be too restrictive because the upper bound can be much larger than the true Lipschitz constant. Instead, minimizing the stable rank of the weight matrices provides a more flexible mechanism for improving stability. Lower stable rank implies that the energy of the operator is concentrated in fewer directions, which in turn reduces the amplification of hidden feature perturbations and encourages contraction [1, 7]. A more practical estimate of its behavior is the empirical Lipschitz constant. It has been shown experimentally that lower stable rank leads to a smaller empirical Lipschitz constant [6].

2.1 Application to neural networks

Linear Layer. The stable rank was associated in [4] with the generalization error bound of neural networks leading to the error bound $\mathcal{O}\left(\sqrt{\prod_i^d \|W_i\|_2^2 \sum_{i=1}^d \operatorname{srank}(W_i)}\right)$. Inspired by this result, [6] proposed applying SRN to each linear layer of a neural network alongside spectral normalization (SN) [3]. The resulting algorithm first applies spectral normalization by dividing the weight matrix by its largest singular value, then solves SRN based on the SN-normalized weight matrix. When applied to neural networks, only the largest singular value is preserved, i.e., $k = 1$ in (2). The close form solution to SRN for a spectral normalized ($\sigma_1 = 1$) matrix W is given by [6] (note $\|S_1\|_F = \sigma_1 = 1$):

$$\widehat{W}_k = S_1 + \frac{\sqrt{r-1}}{\|S_2\|_F} S_2, \tag{3}$$

where $S_1 = u_1 v_1^T$, u_1, v_1 are the leading singular vectors of W, and $S_2 = W - S_1$. The SRN algorithm requires decomposition of the matrix into two matrices S_1 and S_2, through singular value decomposition,

Convolutional Layer. While most architectures employed in vision are CNN-based, SRN is not directly applicable to a convolutional layer, as the size of the underlying linear mapping (represented as a doubly circulant matrix) is too big. Let us denote the circulant/doubly circulant matrix that corresponds to a kernel a in convolution as $C(a)$. Indeed, for a kernel a that convolves with an input feature map (m_{in}, N, N) to produce an output features map (m_{out}, N, N), the corresponding linear operator $C(a)$ is of size $m_{\text{out}} \cdot N^2 \times m_{\text{in}} \cdot N^2$. Some methods, such as [3], reshape the convolutional kernel into a matrix as an approximation. However, this does not properly normalize the operator norm or the stable rank of the linear mapping associated with the kernel [2, 7].

The authors in [7] proposed to adapt the SRN for linear layers to convolutional layers by introducing SRN-C, a method that leverages the equivalence between matrix multiplication and kernel convolution [2]

$$\begin{aligned} \text{vec}(v) = C(a)\text{vec}(u) &\Longleftrightarrow v = a * u, \\ \text{vec}(u) = C(a)^T\text{vec}(v) &\Longleftrightarrow v = a^T * u, \end{aligned} \tag{4}$$

where $\text{vec}(\cdot)$ denotes the vectorization of a feature tensor. This equivalence makes the computation practical. The SRN-C algorithm follows an analogous procedure to SRN, but uses a decomposition of the kernel a instead of the associated matrix $C(a)$. The output is given as a weighted sum of kernels

$$\hat{a} = a_1 + \gamma(r) a_2, \tag{5}$$

where $a_1 = \nabla_{\tilde{a}}\Big(u_1^T(\tilde{a} * v_1)\Big)$, $a_2 = a - a_1$ and $\gamma(r)$ is the scaling factor determined by the stable rank value r. The vectors u_1, v_1 are computed using the power iteration with (4), thus they are the singular vectors of the underlying linear mapping. The algorithm leverages the fact that the power iteration method does not require to explicitly construct the underlying linear operators. The definition of a_1 will be discussed next.

2.2 Analysis of the SRN-C

The kernels produced by the analogue proposed in [7] do not verify the conditions of (3) $S_1 := u_1 v_1^T \neq C(a_1)$. The problems comes from the step

$$a_1 = \nabla_{\tilde{a}}(u_1^T(\tilde{a} * v_1)). \tag{6}$$

Ideally, this step should yield a kernel that can produce the rank-1 matrix $u_1 v_1^T$ corresponding to the maximum singular value of the true linear mapping. However, we argue that:

1. When writing out the gradient term on the right-hand side, a Jacobian matrix appears, meaning the result is no longer exactly uv^T;
2. The circulant matrix generated by the resulting kernel does not have rank 1;

In practice, we implemented this algorithm using PyTorch and applied it to a kernel of shape $(1, 1, 3, 3)$, i.e. a 3×3 kernel and one input/output channel. The input image size is $N \times N$, thus resulting in a matrix of shape $N^2 \times N^2$. The desired stable rank is controlled by $\beta \in (0, 1)$, targeting a stable rank βN^2. As expected, the matrix corresponding to the computed a_1 does not have a maximum singular value of 1, whereas it ideally should; the final normalized kernel does not have spectral norm 1, and the stable rank differs from the target value βN^2. [1]

3 Stable Rank Normalization for Circulant Matrices

The original SRN-C algorithm missed an important point, whether the normalized matrix is still circulant. Furthermore,

1. Experiments are available at `https://github.com/d-zhe/srnc`.

if it is circulant, can it be represented by a kernel that has the same shape as the original one?

For sake of simplicity let us restrict ourselves to 1-d kernels. We define the *Stable Rank Normalization Problem* for convolutional kernels as follows. Assume that we have a circulant matrix $C(a)$ that is generated by a kernel $a \in \mathbb{R}^q$. The problem concerns finding a circulant matrix $C(x)$ satisfying the following conditions:

— Staying as close as possible to the given matrix $C(a)$ in terms of the Frobenius norm;

— Being generated by a kernel x that has the same length q as a (the same support set after padding zeros to match the signal to be convolved);

— Meeting the desired stable rank value r;

— Preserving the largest singular value.

Mathematically, the problem is formulated as follows if the kernel is assumed to have length q (this can be generalized to high dimension):

$$\begin{aligned} \underset{x\in\mathbb{R}^q}{\text{minimize}} \quad & \|C(x) - C(a)\|_F^2 \\ \text{subject to} \quad & \operatorname{srank}(C(x)) = r \\ & \sigma^x_{\max} = \sigma^a_{\max}, \end{aligned} \tag{7}$$

where $\sigma^x_{\max}$ and $\sigma^a_{\max}$ are the largest singular value of the matrices $C(x)$ and $C(a)$.

We will first show that, if the stable rank r is chosen properly the problem (7) is feasible. Then we propose a practical way to find the solution and show numerical results.

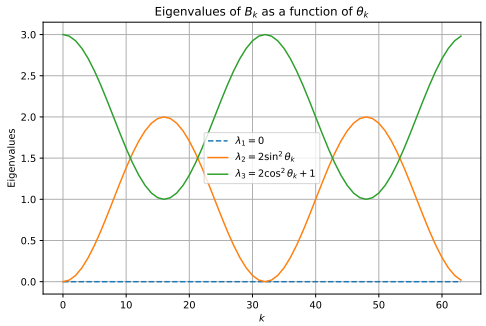

Figure 1 – The eigenvalues of B_k for frequency $0 \le k \le N-1$ over the sphere. $\lambda_{\min}(B_k) \le g_k(x) \le \lambda_{\max}(B_k)$, and $\max_k g_k(x)$ equals 3 in this case. Consequently, $r \ge \frac{N}{3}$.

3.1 Feasible Set

For circulant matrices, the singular values can be computed from the Discrete Fourier Transform (DFT). Let us consider the DFT matrix $F = [f^0, f^1, \cdots, f^{N-1}]$, where $f^k = (\xi_N^{kl})_{l=0,\cdots,N-1}^N$, $\xi = \exp(\frac{2\pi i}{N})$, is the k-th column. For a kernel $x = (x_0, \ldots, x_{N-1})$, the eigenvalues of its circulant matrix are given by $\lambda = Fx$ and therefore the singular values are the magnitudes

$$\lambda_l = \sum_{k=0}^{N-1} x_k \xi_N^{lk}, \quad \sigma_l = |\lambda_l|, \quad l \in \{0, 1, \cdots, N-1\}. \tag{8}$$

Let us denote the l-th squared eigenvalue $g_l(x) = \sigma_l^2 = |\sum_{j=0}^{q-1} x_j f_l^j|^2$, and represent the stable rank of a circulant matrix associated with a kernel x by $f(x) = \operatorname{srank}(C(x))$. Then the constraints in (7) can be rewritten as follows:

$$\begin{cases} f(x) = \dfrac{\sum_i \sigma_i^2}{\max_l \sigma_l^2} = \dfrac{N\|x\|^2}{\max_l g_l(x)} = r & (9) \\ \sigma^x_{\max} = \sqrt{\max_l g_l(x)} = \sigma^a_{\max} \triangleq M & (10) \end{cases}$$

$$\iff \|x\|^2 = \frac{rM^2}{N}, \qquad \max_l g_l(x) = M^2. \tag{11}$$

This shows that the feasible set consists of points on a sphere of radius $\sqrt{\frac{rM^2}{N}}$ that satisfy $\max_l g_l(x) = M^2$. Equivalently, we can

— first find a vector x that verifies the stable rank condition over the unit sphere $\|x\| = 1$,

— then scale it to the desired radius $\tilde{x} = \sqrt{\frac{rM^2}{N}} x$ so as to preserve the largest singular value.

The stable rank function $f(x)$ is continuous w.r.t. x, therefore, when the desired stable rank value r is assigned appropriately, there must be a point on the unit sphere such that $f(x) = r$. The stable rank of any matrix $W \in \mathbb{R}^{N\times N}$ satisfies

$$1 \le \operatorname{srank}(W) \le N,$$

the maximum value $f(x) = N$ can be easily reached, for example, taking $x = (t, 0, \cdots, 0), t \in \mathbb{R}$. However, the minimum value of $f(x)$ is not obvious.

To minimize the $f(x)$ in (9), as the numerator is fixed, we have to maximize the denominator on the unit sphere:

$$\max_{\|x\|^2=1} \max_l g_l(x), \tag{12}$$

where $g_l(x) = \left(x^T \cos_l\right)^2 + \left(x^T \sin_l\right)^2 = x^T B_l x$, with

$$\cos_l^T = [\cos(j \cdot \theta_l)]_{j=0}^{q-1}, \quad \sin_l^T = [\sin(j \cdot \theta_l)]_{j=0}^{q-1},$$
$$\theta_l = \frac{2\pi l}{N}, \quad B_l = \cos_l \cos_l^T + \sin_l \sin_l^T$$

The quadratic form $g_l(x)$ is bounded by the minimum and maximum eigenvalues on the unit sphere. The matrix B_l is the sum of two rank-1 matrices, thus the rank of it is at most 2. In fact, we have $\operatorname{Tr}(B_l) = \operatorname{Tr}(\cos_l \cos_l^T) + \operatorname{Tr}(\sin_l \sin_l^T) = \|\cos_l\|^2 + \|\sin_l\|^2 = q$. Thus, the maximum eigenvalue over all B_l must be reached for a rank-1 matrix. In all cases, $l = 0$ (DC) are $l = \frac{N}{2}$ (Nyquist, even N) gives the desired results that

$$\max_{\|x\|^2=1} \max_l g_l(x) = \max_l \lambda_{\max}(b_l) = \|\cos_l\|^2 = q.$$

In conclusion, when the prescribed stable rank r satisfies $\frac{N}{q} \le r \le N$, the feasible set of the optimization problem (7) is non-empty, provided that the length of the kernel is q.

In Figure 1 we show an example for a kernel with $q = 3$ coefficients applied to signals of length $N = 64$. The Figure shows the two leading eigenvalues of B_l for every frequency l. In our case, the maximum value is 3, attained by $x = (1, 1, 1)/\sqrt{3}$, and at the frequency at $k = 0 \quad (\cos\theta_k = 1)$ or $x = (1, -1, 1)/\sqrt{3}$, $k = 32 \quad (\cos\theta_k = -1)$. This result coincides with the intuition that the signal should concentrate the energy as much as possible on one single frequency. For example, if we take $\theta_l = \frac{2\pi l}{N} = 0, (l = 0)$, then $\mathbf{x}$ is a truncated signal that has only one frequency: x is $x_k = \exp(i2\pi f_0 k/N), l = 0, 1, 2; f_0 = 0\ (DC)$.

3.2 Approximating the Constraint

Following the idea to first find a point on the unit sphere that satisfies the stable rank constraint, we have to solve this equation

$$\max_{0\le k\le N-1} |\sum_{j=0}^{q-1} x_j f^j|_k^2 = \max_{0\le k\le N-1} g_k(x) = N/r \qquad (13)$$

on the unit sphere $\mathcal{S}^{q-1} = \{x : \|x\|^2 = 1\}$ for a valid stable rank value r. For a fixed k_0, $g_{k_0}(x) = N/r$ is solvable on the sphere, depending on the max and min eigenvalues of B_{k_0}. However, $\arg\max_k g_k(x) = k_0$ does not necessarily hold. If fact, $\{x \in \mathcal{S}^{q-1} : \max_l g_l(x) = \frac{N}{r}\} = \bigcup_l \{x \in \mathcal{S}^{q-1} : g_l(x) = \frac{N}{r}, g_k(x) \le g_l(x)\}$. Thus, solving (13) is difficult.

We define the function $g_{\text{lse}}(x)$ to approximate $\max_l g_l(x) = \max_l x^T B_l x$ by *log-sum-exp* function, with parameter $\alpha > 0$:

$$g_{\text{lse}}(x) = \frac{1}{\alpha} \log \left(\sum_{l=0}^{N-1} \exp\left(\alpha\, x^T B_l x\right) \right). \qquad (14)$$

The optimization problem (7) can be rewritten as

$$\begin{aligned} &\min_{x\in\mathbb{R}^q} \quad N\|x-a\|^2 \\ &\text{s.t.} \quad \max_l g_l(x) = M^2 \quad \text{and} \quad \|x\|^2 = \frac{rM^2}{N}. \end{aligned} \qquad (15)$$

By penalizing the equality constraint, with changing the variable onto the unit sphere and approximating by f_{lse}, we obtain

$$\min_{\|x\|=1} h(x) = (g_{\text{lse}}(x) - N/r)^2 + \lambda N \|\sqrt{\frac{rM^2}{N}} x - a\|^2. \qquad (16)$$

3.3 Numerical Experiments

We solve the optimization problem on the unit sphere (16) by Riemannian gradient descent. Starting from a point z_0 on the sphere, for each iterates z_t, we have

$$d_t = \nabla h(z_t) \qquad \text{compute the Euclidean gradient} \qquad (17)$$

$$\tilde{d}_t = d_t - \langle d_t, z_t\rangle z_t \qquad \text{project onto the tangent space} \qquad (18)$$

$$z_{t+1} = \frac{z_t - \eta \tilde{d}_t}{\|z_t - \eta\tilde{d}_t\|} \qquad \text{update and normalize} \qquad (19)$$

The procedure is implemented using PyTorch for computing gradient easily. For $q = 3$, we plot the trajectory and the distance and the stable rank of the iterates in Figures 2 and 3. This solves the problem $a = (1, 0, 0), N = 64, \alpha = 10, \lambda = 0.1$, starting from a random point on the sphere. The step size is fixed as $\eta = 1e-5$.

4 Conclusion

Smaller stable rank helps improve the generalizability of networks [6] and stabilize the recurrent networks [1, 7]. However, the adaptation of the SRN algorithm from matrices to tensors is not obvious [7]. This work presents the problem of SRN-C, which aims at normalizing the stable rank for kernels, and discusses the feasibility of the posed problem. The problem is feasible when the desired stable rank r is properly chosen. While without explicit solution to it, it brings difficulty to be applied to CNNs. The future work includes 1) considering 2-d kernels; 2) relaxing the problem; 3) apply to CNN-based networks with inexact solution.

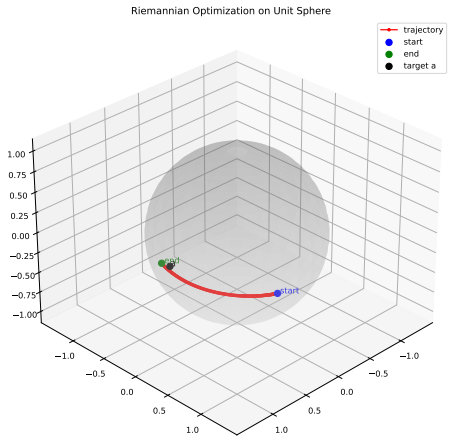

Figure 2 – Trajectory of iterates for solving the problem (16) using Riemannian gradient descent.

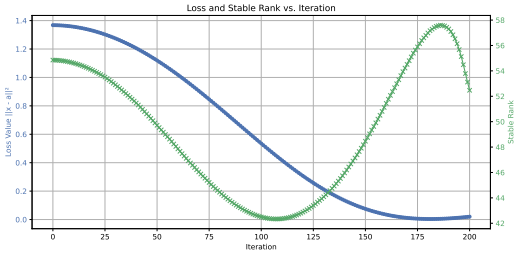

Figure 3 – The loss value and stable rank value versus the iterates.

References

[1] B. N. Chiche, A. Woiselle, J. Frontera-Pons, and J.-L. Starck. Stable long-term recurrent video super-resolution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 837–846, 2022.

[2] H. Gouk, E. Frank, B. Pfahringer, and M. J. Cree. Regularisation of neural networks by enforcing lipschitz continuity. *Machine Learning*, 110:393–416, 2021.

[3] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*, 2018.

[4] B. Neyshabur, S. Bhojanapalli, and N. Srebro. A pac-bayesian approach to spectrally-normalized margin bounds for neural networks. In *International Conference on Learning Representations*, 2018.

[5] M. Rudelson and R. Vershynin. Sampling from large matrices: An approach through geometric functional analysis. *Journal of the ACM (JACM)*, 54(4):21–es, 2007.

[6] A. Sanyal, P. H. Torr, and P. K. Dokania. Stable rank normalization for improved generalization in neural networks and gans. In *International Conference on Learning Representations*, 2020.

[7] T. Tanay, A. Sootla, M. Maggioni, P. K. Dokania, P. Torr, A. Leonardis, and G. Slabaugh. Diagnosing and preventing instabilities in recurrent video processing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(2):1594–1605, 2022.