arXiv:2103.04217v1 [cs.LG] 7 Mar 2021

Spectral Tensor Train Parameterization of Deep Learning Layers

Anton Obukhov
ETH Zurich

Maxim Rakhuba
HSE University

Alexander Liniger
ETH Zurich

Zhiwu Huang
ETH Zurich

Stamatios Georgoulis
ETH Zurich

Dengxin Dai
ETH Zurich

Luc Van Gool
ETH Zurich, KU Leuven

Abstract

We study low-rank parameterizations of weight matrices with embedded spectral properties in the Deep Learning context. The low-rank property leads to parameter efficiency and permits taking computational shortcuts when computing mappings. Spectral properties are often subject to constraints in optimization problems, leading to better models and stability of optimization. We start by looking at the compact SVD parameterization of weight matrices and identifying redundancy sources in the parameterization. We further apply the Tensor Train (TT) decomposition to the compact SVD components, and propose a non-redundant differentiable parameterization of fixed TT-rank tensor manifolds, termed the Spectral Tensor Train Parameterization (STTP). We demonstrate the effects of neural network compression in the image classification setting and both compression and improved training stability in the generative adversarial training setting. Project website: obukhov.ai/sttp.

1 Introduction

Deep neural networks have become ubiquitous over the past decade in many computer science domains such as computer vision (Krizhevsky et al., 2012) and natural language processing (Vaswani et al., 2017). Much of the research was dedicated to improving model performance on various datasets and benchmarks, which has led to models with billions of parameters. On the other hand, research productization has led to advances in model compression and energy efficiency, required by constrained computational environments such as edge devices. Adapting research models for production is a challenging task, often involving a ground-up redesign of the model architecture, as seen in Howard et al. (2017). Changes to the model often lead to a vastly different optimization landscape, which may present an additional challenge, especially in unstable settings, such as Generative Adversarial Networks (GAN) (Goodfellow et al., 2014). Therefore, there is a demand for parameter-efficient drop-in neural network components (such as the linear and convolutional layers) with variable capacity and improved training stability for a wide range of optimization settings.

To tackle these challenges, we introduce a principled way to construct low-rank convolutional and linear layers with embedded spectral properties through weight matrix reparameterization. The usage of low-rank layers in place of the original ones introduces network compression in terms of the number of parameters. A layer rank can be treated as a hyperparameter, which defines the layer capacity and its computational cost, and does not affect the layer dimensions. Embedded spectral properties permit efficient rank utilization within the layer and prevent the growth of the layer's Lipschitz constant during training, which improves the optimization process stability and the final model performance.

To this end, we propose two parameterizations, which represent a weight matrix as a product of the compact SVD components: $W = U\Sigma V^\top$. In SVD Parameterization (SVDP), we directly parameterize Σ using free parameters, and U, V using a parameterization of orthonormal frames, such as the Householder parameterization. Fixing or penalizing Σ corresponds to embedding spectral properties into the layer. W is differentiable with respect to parameters of the components, and hence parameter gradients can be computed using auto-differentiation and updated using a standard optimizer such as SGD. Next, we propose a Spectral

Proceedings of the 24$^{\text{th}}$ International Conference on Artificial Intelligence and Statistics (AISTATS) 2021, San Diego, California, USA. PMLR: Volume 130. Copyright 2021 by the author(s).

Tensor Train Parameterization (STTP), further representing U and V through several parameterizations of much smaller orthonormal frames. STTP introduces sparsity into U and V, leading to fewer parameters than SVDP with the same rank. It is worth noting that both proposed parameterizations of weight matrices are non-redundant. To the best of our knowledge, the case of a differentiable non-redundant parameterization of fixed TT-rank tensor manifolds is a novel result. From the practical point of view, given the same budget of parameters, STTP spans a different submanifold of weight matrices than SVDP, which results in more expressive layers in certain rank ranges.

Our scenario of interest includes following a pre-defined training protocol without transfer learning, with a few modifications. Before the training begins, we replace selected layers with low-rank ones of compatible dimensions. During the training, an optional spectral penalty is added to the loss function before backpropagation. After the training, the network can be stored in the parameterized form, or the layers can be decompressed into the original convolutional and linear types.

The paper is structured as follows. Sec. 2 defines the notation and key terms. SVDP is introduced in Sec. 3. The parameterization of orthonormal components arising from SVDP is discussed in Sec. 3.1. Sec. 3.2 highlights parameter redundancy in SVDP and proposes a non-redundant modification. We introduce STTP in Sec. 3.3 and reuse the results from Sec. 3.2 to remove the redundancy. Sec. 3.4 discusses the spectral constraints applicable in both SVDP and STTP. We compare parameterizations and spectral constraints in the context of training GAN and image classification networks in Sec. 4. Sec. 5 concludes the paper.

Related Work Zhang et al. (2018) explore SVD and Householder parameterizations in the context of vanishing gradients in the transition matrix of recurrent neural networks (RNN) and study representation power and generalization bounds of spectral RNN layers. Although other orthogonal parameterization approaches exist, such as exponential maps (Lezcano-Casado and Martínez-Rubio, 2019) and Givens rotations, Householder transformation is found to be the most efficient (Shepard et al., 2015). The Tensor Train (TT) decomposition by Oseledets (2011) is used to parameterize weight matrices in a low-rank fashion in Yang et al. (2017); Garipov et al. (2016); Novikov et al. (2015). Both convolutional and linear layers are shown to have an adequate low-rank parameterization, although with no regard to the spectral properties or the redundancy of the proposed parameterizations. Other low-rank tensor parameterizations have been used for network compression (Obukhov et al., 2020; Wang et al., 2018; Lebedev et al., 2015), offering high compression rates at the cost of undefined spectral properties and representation redundancy. In another vein, Phan et al. (2020) address instabilities arising during CP decomposition of weight matrices during training. Holtz et al. (2012) provide the exact dimensionality of fixed TT-rank tensor manifolds; however, parameterizations are not discussed. Despite the similar naming of Bigoni et al. (2016), their paper is concerned with spectral approximation theory and extending TT to functions of continuous variables. Finally, spectrum control effectively addresses multiple neural network training problems such as representation degeneration (Wang et al., 2020) and mode collapse (Miyato et al., 2018).

2 Preliminaries

We are concerned with the class of neural network models $\mathcal{F}_\theta$ with learned parameters θ composed of affine and non-linear mappings. For example, a feed-forward network with L layers takes the form $\omega_L \circ a_{L-1} \circ \omega_{L-1} \circ \cdots \circ a_1 \circ \omega_1$, where ω_k are learned affine mappings, a_k are non-linear mappings (activations), and $\circ$ denotes composition, meaning that the output of an i-th layer is the input of the $(i+1)$-th layer.

The *Lipschitz constant* of a mapping $\mu : \mathbb{R}^M \to \mathbb{R}^N$ is such a constant K_μ (if it exists) that the inequality $\|\mu(x) - \mu(y)\|_2 \leq K_\mu \|x - y\|_2$ holds for any $x, y \in \mathbb{R}^M$. Most non-linearities (such as ReLU, sigmoid, etc.) have their Lipschitz constant equal to 1. If every layer of the feed-forward model is Lipschitz-continuous, so is the composition of the layers, and thus the upper bound of the Lipschitz constant of such network is given by:

$$K_{\mathcal{F}_\theta} \leq \prod_{\ell=1}^{L} K_{\omega_\ell}. \tag{1}$$

Similar bounds can be derived for most computational graphs corresponding to popular deep architectures such as CNNs, RNNs, Transformers, and others.

A *linear layer* is an affine mapping: $\omega_\ell \colon \mathbb{R}^{d_\ell} \to \mathbb{R}^{d_{\ell+1}} : x \mapsto W_\ell x + b_\ell$, with $W_\ell \in \mathbb{R}^{d_{\ell+1} \times d_\ell}$ (weight matrix), $b_\ell \in \mathbb{R}^{d_{\ell+1}}$ (bias term), $\ell = 1, \ldots, L$. The Lipschitz constant of a linear layer is equal to the largest singular value of the layer's weight matrix: $K_{\omega_\ell} = \sigma_1(W_\ell)$.

An N-dimensional *convolutional layer* acting in spatial dimensions $S_1, \ldots, S_N$ is an affine mapping: $\omega \colon \mathbb{R}^{C_{\text{in}} S_1 \ldots S_N} \to \mathbb{R}^{C_{\text{out}} S_1 \ldots S_N}$, given by the kernel tensor $\mathcal{W} \in \mathbb{R}^{C_{\text{out}} \times C_{\text{in}} \times M_1 \times \cdots \times M_N}$. Following[1] the conventions set by Miyato et al. (2018) when dealing

[1] The Lipschitz constant of a convolutional layer may be larger than the largest singular value of the kernel matrix, as noted in Sedghi et al. (2019). Nevertheless, the empirical observations in Sanyal et al. (2020) suggest that it does not often happen in practice.

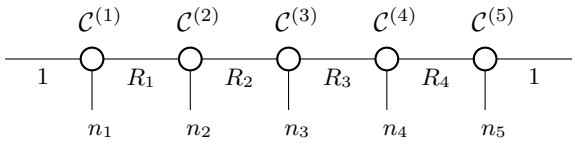


Figure 1: A tensor diagram corresponding to the TT decomposition of a 5D tensor in $\mathbb{R}^{n_1 \times n_2 \times n_3 \times n_4 \times n_5}$.

with convolutional layers, we are concerned with the kernel tensor reshaped into a kernel matrix of size $C_{\text{out}} \times (C_{\text{in}} \cdot M_1 \cdots M_N)$, also called the weight matrix.

TT decomposition (Oseledets, 2011) is a representation for a low-rank approximation of an arbitrary D-dimensional array (tensor) $A \in \mathbb{R}^{n_1 \times \cdots \times n_D}$ through several three-dimensional tensors (TT-cores) $\mathcal{C}^{(i)} \in \mathbb{R}^{R_{i-1} \times n_i \times R_i}, i = 1, \ldots, D$, with TT-rank $(R_0, \ldots, R_D)$. Its elements are expressed as follows:

$$A_{i_1,\ldots,i_D} = \sum_{\beta_0,\ldots,\beta_D=1}^{R_0,\ldots,R_D} \mathcal{C}^{(1)}_{\beta_0,i_1,\beta_1} \cdot \mathcal{C}^{(2)}_{\beta_1,i_2,\beta_2} \cdots \mathcal{C}^{(D)}_{\beta_{D-1},i_D,\beta_D} \quad (2)$$

The *TT-rank* R defines the degree of compression of A. By convention, $R_0 = R_D = 1$, and the rest of the rank values are bounded (Holtz et al., 2012, Eq. (20)):

$$1 \leq R_k \leq R_k^{\max} \equiv \min\left(\prod_{j=1}^{k} n_j, \prod_{j=k+1}^{D} n_j\right). \quad (3)$$

We define the *order of elements* in the tensor A by associating each element $A_{i_1,\ldots,i_D}$ with a multi-index:

$$\overline{i_1 \ldots i_D} = 1 + \sum_{p=1}^{D} (i_p - 1) \prod_{q=p+1}^{D} n_q. \quad (4)$$

Reshaping preserves the order of elements. In what follows, *tensorization* refers to the reshaping of a vector or a matrix into a tensor. *Matricization* (e.g., of TT-cores) refers to the reshaping of a tensor into a matrix.

Tensor diagram notation (Fig. 1) is a convenient tool for visualizing interactions of tensors like in (2). Each node represents a tensor with the number of legs matching the number of dimensions: 1 – vector, 2 – matrix, 3 – 3D array (e.g., TT-core). Connected legs represent summation over the corresponding indices in (2). Size-1 legs may be omitted. *Contraction* of the tensor diagram is the operation of computing the elements of the tensor product of all nodes involved in the operation (e.g., contraction of Fig. 1 gives a tensor of size $n_1 \times \ldots \times n_5$).

3 Method

In this section, we describe the proposed parameterization of neural network layers. Given a weight matrix $W \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ of a layer implementing an affine mapping and the rank hyperparameter $r \leq \min(d_{\text{out}}, d_{\text{in}})$, we represent W using the compact SVD with rank r:

$$W = U \Sigma V^\top, \quad (5)$$

where $U \in \mathbb{R}^{d_{\text{out}} \times r}$ and $V \in \mathbb{R}^{d_{\text{in}} \times r}$ have orthonormal columns, and $\Sigma = \text{diag}(\sigma_1, \ldots, \sigma_r)$ is a matrix of singular values, parameterized by r parameters. Matrices U and V belong to the real Stiefel manifold

$$\text{St}(d, r) \equiv \{X \in \mathbb{R}^{d \times r} : X^\top X = I_r\}$$

of orthonormal real r-frames ($r \leq d$), referred to as *orthonormal frames*. In what follows, we will use U of the size $d \times r$, implying either U of the size $d_{\text{out}} \times r$ or V of the size $d_{\text{in}} \times r$, unless stated otherwise.

We directly parameterize the arising $U \in \text{St}(d, r)$ by certain mappings $\phi : \mathbb{R}^q \to \text{St}(d, r)$ such that $U = \phi(\theta)$ and where q is the dimensionality of a submanifold of $\text{St}(d, r)$, chosen according to a parameterization type.

We consider two types of weight matrix parameterizations: (1) SVDP, requiring both U and V parameterized as orthonormal frames (Sec. 3.1, Fig. 2a), and (2) STTP, a parameterization of a reshaped weight matrix with a fixed TT-rank or, equivalently, SVDP with TT-compressed U and V (Sec. 3.3, Fig. 2b).

3.1 SVDP

To construct the mapping ϕ from the parameter space to orthonormal frames, we utilize a sequence of Householder reflections (Shepard et al., 2015). It is known that any matrix $A \in \mathbb{R}^{d \times r}$ can be represented using QR-decomposition $A = QR$, where $Q \in \text{St}(d, r)$ and $R \in \mathbb{R}^{r \times r}$ is upper-triangular. The matrix Q can be given as a product of Householder reflections:

$$Q = H^{(1)} H^{(2)} \ldots H^{(r)} I_{d \times r}, \quad (6)$$

where $I_{d \times r}$ is a truncated identity matrix of size $d \times r$, and the Householder reflector $H^{(i)}$ is written as $H^{(i)} = I_d - 2u^{(i)} u^{(i)\top}$ for some $u^{(i)} \in \mathbb{R}^d$: $\|u^{(i)}\|_2 = 1$ and $u^{(i)}_\alpha = 0$, $\alpha = 1, \ldots, i-1$.

A QR decomposition of $U \in \text{St}(d, r)$ results in a diagonal matrix R with $R_{ii} = \{-1, 1\}, i = 1, \ldots, r$. To ensure the uniqueness and differentiability of SVDP, one has to choose R_{ii} carefully; it affects the numerical stability of the parameterization of certain regions of $\text{St}(d, r)$. As a result, $U = QR$, and hence it is given by the matrices $H^{(i)}$. The number of *degrees of freedom* (DOF) to represent $H^{(i)}$ is $(d - i)$ since we only need to store nonzero entries of $u^{(i)}$, and there is an additional requirement $\|u^{(i)}\|_2 = 1$. The total number of parameters to represent all the $H^{(i)}$, $i = 1, \ldots, r$ is

$$\text{DOF}(U) = \sum_{i=1}^{r} (d - i) = dr - \frac{r(r+1)}{2},$$

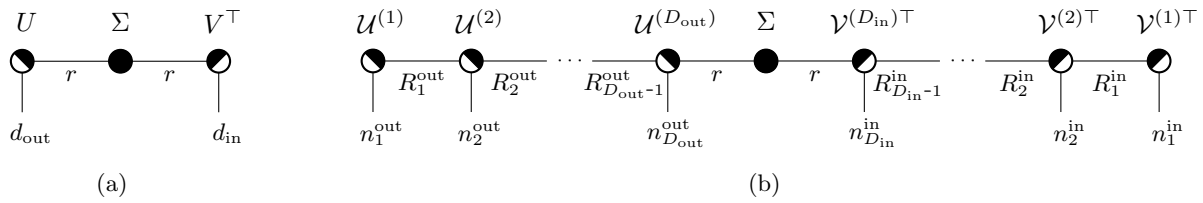


Figure 2: Tensor diagrams of (a) SVDP and (b) STTP of a weight matrix $W \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$, where $d_{\text{out}} = \prod n_i^{\text{out}}$, $d_{\text{in}} = \prod n_j^{\text{in}}$. Half-filled nodes represent TT-cores whose matricizations ($\mathcal{M}$) are orthonormal frames (with shaded area enumerating columns). Filled nodes represent diagonal matrices. In both SVDP and STTP, we can choose any r subject to rank constraints while still controlling the spectral properties due to the exposed matrix Σ. STTP provides an extra degree of compression given the same rank r due to the induced sparsity in U and V.

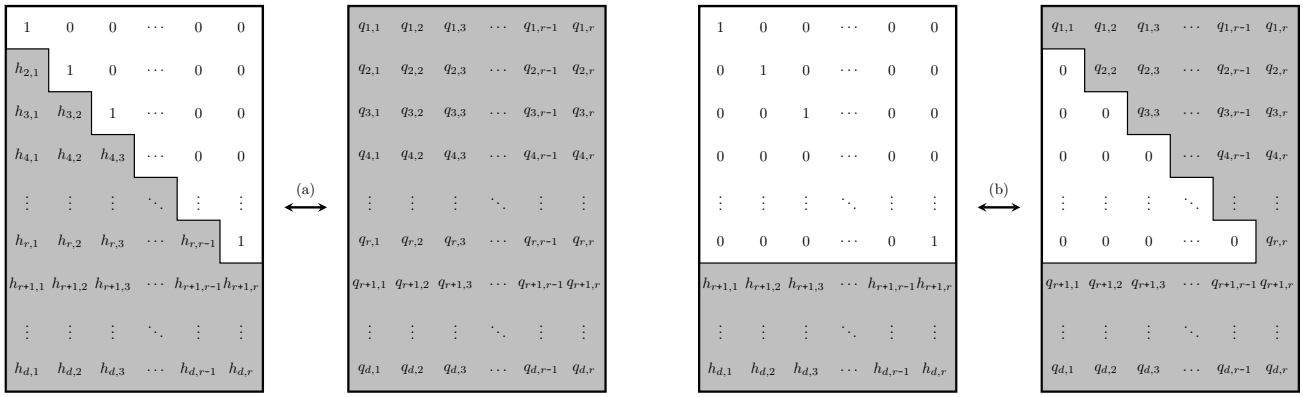


Figure 3: Visualization of Householder parameterizations of orthonormal frames as per the LAPACK convention: (a) Full parameterization, (b) Reduced parameterization. Shaded areas with $h_{i,j}$ values represent parameters of reflectors; $q_{i,j}$ values represent orthonormal frame elements, affected by the corresponding parameterization.

which coincides with the dimensionality of $\text{St}(d, r)$.

We use the LAPACK convention for parameter layout in a matrix with columns $h^{(i)}, i = 1, ..., r$ (Fig. 3a) and obtain $u^{(i)} = h^{(i)}/\|h^{(i)}\|_2$.

Once both $U \in \text{St}(d_{\text{out}}, r)$ and $V \in \text{St}(d_{\text{in}}, r)$ are parameterized as is described above, SVDP spans the whole manifold of weight matrices of ranks not higher than r. Thus, the total number of degrees of freedom required to parameterize the matrix $W \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ using SVDP with rank r adds up from the numbers of parameters required to parameterize orthonormal frames $U \in \mathbb{R}^{d_{\text{out}} \times r}$, $V \in \mathbb{R}^{d_{\text{in}} \times r}$ and r singular values:

$$\text{DOF}(W) = r(d_{\text{out}} + d_{\text{in}}) - r^2.$$

3.2 The Case of Identity Spectrum

When all r singular values are fixed to 1 ($\Sigma = I_r$), independent parameterizations of U and V lead to redundancy in W. This is due to the fact that for any orthogonal matrix $Q \in \mathbb{R}^{r \times r}$, the following holds:

$$W = UV^\top = UQ^\top QV^\top = (UQ)(VQ)^\top, \qquad (7)$$

which leads to a new $\widetilde{U} = UQ \in \text{St}(d_{\text{out}}, r)$ and $\widetilde{V} = VQ \in \text{St}(d_{\text{in}}, r)$ that produce the same W.

To eliminate this redundancy of parameters, we impose additional constraints on either U or V (we choose U for concreteness). We follow the Grassmann manifold parameterization (Shepard et al., 2015) and require the leading $r \times r$ sub-matrix of $U \in \text{St}(d_{\text{out}}, r)$ to be upper triangular. We denote the subset of all such matrices by $\text{St}_{\mathsf{U}}(d_{\text{out}}, r) \subset \text{St}(d_{\text{out}}, r)$ (subscript U for "upper").

To parameterize a matrix $U \in \text{St}_{\mathsf{U}}(d_{\text{out}}, r)$, we propose a reduced form of the Householder parameterization (Fig. 3b). It differs from the full parameterization by setting the entries $h_j^{(i)}, i + 1 \leq j < r$ to zero. This saves us $r(r-1)/2$ parameters to store the vectors $h^{(i)}$, $i = 1, \ldots, r$, leading to $(d_{\text{out}} r - r^2)$ effective parameters.

Thus the total number of independent parameters required to parameterize W with $U \in \text{St}_{\mathsf{U}}(d_{\text{out}}, r)$, $V \in \text{St}(d_{\text{in}}, r)$, and $\Sigma = I_r$ becomes

$$\text{DOF}(W) = r(d_{\text{out}} + d_{\text{in}}) - \frac{r(3r+1)}{2}.$$

Apart from a smaller parameter footprint, redundancy removal may also benefit the optimization landscape, as the redundant parameters introduce plateau regions.

3.3 STTP

We assume that the dimensions of the weight matrix $W \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ factorize (see discussion in Sec. 8): $d_{\text{out}} = n_1^{\text{out}} \cdots n_{D_{\text{out}}}^{\text{out}}$, $d_{\text{in}} = n_1^{\text{in}} \dots n_{D_{\text{in}}}^{\text{in}}$, for example, prime factors with repetition. Thus we can consider parameterizing the weight matrix W tensorized into a tensor $\widetilde{W}$ with factored dimensions $n_1^{\text{out}} \times \cdots \times n_{D_{\text{out}}}^{\text{out}} \times n_{D_{\text{in}}}^{\text{in}} \times \cdots \times n_1^{\text{in}}$. Upon obtaining $\widetilde{W}$ from the underlying parameters, the matrix structure of W can be recovered through matricization.

As previously discussed, simply parameterizing $\widetilde{W}$ as a TT decomposition with unconstrained parameterizations of TT-cores (2) as done in the prior art does not specify the spectral properties of W. However, since the TT decomposition is inherently redundant, $\widetilde{W}$ can have multiple equivalent TT decompositions, including the one shown in Fig. 2b. Here matricizations $\mathcal{M}\colon \mathbb{R}^{a\times b\times c} \to \mathbb{R}^{ab\times c}$ of TT-cores $\mathcal{U}^{(i)}$ and $\mathcal{V}^{(j)}$ are[2] orthonormal frames, and Σ is a matrix of singular values of the weight matrix W (Holtz et al., 2012).

Let us show that this TT decomposition can be reduced to the SVD form (5), with U and V being orthonormal frames. The elements of the matrix $U \in \mathbb{R}^{n_1 \cdots n_D \times r}$ are products of TT-cores $\mathcal{U}^{(k)}$ to the left of Σ in Fig. 2b:

$$U_{\overline{i_1 \dots i_D},\alpha} = \sum_{\beta_0,\dots,\beta_D=1}^{R_0,\dots,R_D} \mathcal{U}^{(1)}_{\beta_0,i_1,\beta_1} \cdot \mathcal{U}^{(2)}_{\beta_1,i_2,\beta_2} \dots \mathcal{U}^{(D)}_{\beta_{D-1},i_D,\alpha}, \quad (8)$$

where $\overline{i_1 \dots i_D}$ is computed as in (4), $\alpha \in [1, r]$. The next proposition illustrates that our choice of TT-cores $\mathcal{U}^{(k)}$ leads to $U \in \text{St}(d, r)$. For convenience, we use the notation $U = \mathcal{T}(\mathcal{U}^{(1)}, \dots, \mathcal{U}^{(D)})$ as a shorthand to (8).

Proposition 1. *Let the matricizations $\mathcal{M}(\mathcal{U}^{(k)}) \in \mathbb{R}^{R_{k-1}n_k \times R_k}$ of the TT-cores $\mathcal{U}^{(k)} \in \mathbb{R}^{R_{k-1}\times n_k \times R_k}$ satisfy $\mathcal{M}(\mathcal{U}^{(k)}) \in \text{St}(R_{k-1}n_k, R_k)$, $k = 1, \dots, D$. Then $\mathcal{T}(\mathcal{U}^{(1)}, \dots, \mathcal{U}^{(D)}) \in \text{St}(n_1 \dots n_D, r)$.*

Proof. Follows from Oseledets (2011, Lemma 3.1); see the proof in Sec. 10 for completeness. □

Proposition 1 gives us a framework to perform parameterization of TT-cores, leading to parameterizations of U and V, and in the end, W with a given spectrum. It follows that we can parameterize the matricized TT-cores $\mathcal{M}(\mathcal{U}^{(k)}) \in \text{St}(R_{k-1}n_k, R_k)$ using the procedure described in Sec. 3.1. Nevertheless, the following proposition suggests that parameterizing each $\mathcal{M}(\mathcal{U}^{(k)})$ simply as an element of the Stiefel manifold leads to over-parameterization similar to (7), which is a direct consequence of TT decomposition non-uniqueness.

[2]Similar to how matrices U and V are treated identically in Sec. 3.1, so are TT-cores of U and V; however, the transposed V in (5) leads to the transposed $\mathcal{M}(\mathcal{V}^{(k)})$. Therefore, TT-cores of $V^\top$ are denoted as $\mathcal{V}^{(k)\top}$ in Fig. 2b.

Proposition 2. *Let $Q_k \in \mathbb{R}^{R_k \times R_k}$ be orthogonal for $k = 1, \dots, D-1$, $Q_0 = 1$, $Q_D = I_r$. We also assume that $\mathcal{U}^{(k)} \in \mathbb{R}^{R_{k-1}\times n_k \times R_k}$, $k = 1, \dots, D$ are such that $\mathcal{M}(\mathcal{U}^{(k)}) \in \text{St}(R_{k-1}n_k, R_k)$. We define $\widetilde{\mathcal{U}}^{(k)} \in \mathbb{R}^{R_{k-1}\times n_k \times R_k}$: $\widetilde{\mathcal{U}}^{(k)}_{:,i_k,:} = Q_{k-1}^\top \mathcal{U}^{(k)}_{:,i_k,:} Q_k$ (where $\mathcal{U}^{(k)}_{:,i_k,:} \in \mathbb{R}^{R_{k-1}\times R_k}$). Then*

$$\mathcal{M}(\widetilde{\mathcal{U}}^{(k)}) \in \text{St}(R_{k-1}n_k, R_k), \quad k = 1, \dots, D, \quad \text{and}$$
$$\mathcal{T}\left(\mathcal{U}^{(1)}, \dots, \mathcal{U}^{(D)}\right) = \mathcal{T}\left(\widetilde{\mathcal{U}}^{(1)}, \dots, \widetilde{\mathcal{U}}^{(D)}\right).$$

Proof. See a complete proof in Sec. 11. □

To avoid over-parameterization, we reuse the approach from Sec. 3.2 and require all TT-cores except for the two adjacent to Σ in Fig. 2b to have reduced parameterizations: $\mathcal{M}(\mathcal{U}^{(k)}) \in \text{St}_{\text{U}}(R_{k-1}n_k, R_k)$, $k = 1, \dots, D-1$ and $\mathcal{M}(\mathcal{U}^{(D)}) \in \text{St}(R_{D-1}n_D, r)$. For the edge case of identity spectrum, we additionally require $\mathcal{M}(\mathcal{U}^{(D)}) \in \text{St}_{\text{U}}(R_{D-1}n_D, r)$ just for the last TT-core of U (but not for the last TT-core of V). The algorithm to enforce such parameterizations is described in Sec. 3.1.

The total number of independent parameters required to parameterize W with STTP without redundancy is:

$$\text{DOF}(W) = \sum_{k=1}^{D_{\text{out}}+D_{\text{in}}} R_{k-1}n_kR_k - \sum_{k=1}^{D_{\text{out}}+D_{\text{in}}-1} R_k^2, \quad \text{where}$$
$$R = (1, R_1^{\text{out}}, \dots, R_{D_{\text{out}}-1}^{\text{out}}, r, R_{D_{\text{in}}-1}^{\text{in}}, \dots, R_1^{\text{in}}, 1)^{[3]},$$
$$n = (n_1^{\text{out}}, \dots, n_{D_{\text{out}}}^{\text{out}}, n_{D_{\text{in}}}^{\text{in}}, \dots, n_1^{\text{in}}) \quad (9)$$

are the TT-rank and dimensions of $\widetilde{W}$ respectively. Notably, $\text{DOF}(W)$ with learned spectrum matches the dimensionality of the fixed TT-rank R tensor manifold given in (Holtz et al., 2012) (see derivation in Sec. 12).

An edge case of STTP happens with the values of TT-rank R (excluding r in the middle) set to $R_k^{\max}$ subject to (3): such parameterization spans the same manifold of rank-r matrices W as SVDP. A careful inspection of this edge case reveals that it is an SVDP in disguise: all TT-cores except for the two adjacent to Σ have square matricizations. Given that $\forall p\ \text{St}_{\text{U}}(p, p)$ does not require any learned parameters (Fig. 3b), all of them are concentrated in Σ and the two adjacent TT-cores with matricizations of sizes $d_{\text{out}} \times r$ and $d_{\text{in}} \times r$.

The degree of compression of W is defined by the TT-rank R. In practice, we treat r as the only hyperparameter and compute TT-rank values as $R_k = \min(r, R_k^{\max})$ using (3). As such, $\text{DOF}(W) = \mathcal{O}(r^2 \log(d_{\text{out}}d_{\text{in}}))$; the number of parameters is logarithmic in the size of W.

[3]TT-rank of a tensor diagram Fig. 2b made compatible with the form of TT decomposition introduced in (2) and Fig. 1 (consisting only of TT-cores) by contracting the matrix Σ into either left or right adjacent TT-core.

3.4 Spectral Constraints

We consider two distinct cases: the identity spectrum (Sec. 3.1) and learned parameterization of the diagonal matrix Σ, with optional regularization. As was previously shown, the former case results in a more compact (also more restricted) parameterization.

The learned singular values Σ are parameterized with a vector $S \in \mathbb{R}^r$. To implement a Lipschitz-1 constraint (1), we initialize $S = I_r$ and compute Σ to keep all singular values constrained in the $[-1, 1]$ range:

$$\Sigma = \text{diag}(S/\|S\|_\infty)$$

Furthermore, we investigate whether an additional regularization term associated with Σ helps to learn a better model. To this end, we explore the D-optimal regularizer (10) as in Jiang et al. (2018), which penalizes the learned singular values of small magnitude:

$$\mathcal{R}(\Sigma) = -\sum_{i=1} \log |\Sigma_i|. \tag{10}$$

As will be discussed in the experiments section, the spectra of neural network layers are crucial to the stability of optimization and good model performance. A near-zero element in Σ effectively reduces the rank of the whole weight matrix W. Therefore, embedding spectral properties through regularization or identity spectrum is a more versatile approach than enforcing only the Lipschitz-1 constraint on the model.

4 Experiments

We evaluate STTP in the Generative Adversarial Networks (GANs) training setting, which is unstable due to its non-convex non-concave optimization objective. The *Mode Collapse* problem goes back to Goodfellow et al. (2014) where it manifested in the generator producing data samples of limited variety (Fig. 4, top).

Recent analysis (Che et al., 2017; Arjovsky and Bottou, 2017; Gulrajani et al., 2017) suggests that the family of discriminator functions plays a key role in the training dynamics (i.e., training stability) and the quality of the generator parameters updates. Others (Miyato et al., 2018; Jiang et al., 2018) tied up mode collapse in the generator with Lipschitz continuity of the discriminator, and most recently, Liu et al. (2019) associated mode collapse with the *Spectral Collapse*. The latter is a condition of simultaneous growth of the spectral norm and drop of the *stable rank* (Sanyal et al., 2020) of weight matrices in the discriminator layers.

4.1 Setup

We follow closely the unconditional image generation setup of SNGAN from Miyato et al. (2018); in particular, we use a residual generator $\mathcal{G}$ and discriminator $\mathcal{D}$ for image sizes 32×32 and 48×48 and the hinge loss objective $\min_\mathcal{G} \max_\mathcal{D} V(\mathcal{G}, \mathcal{D})$.

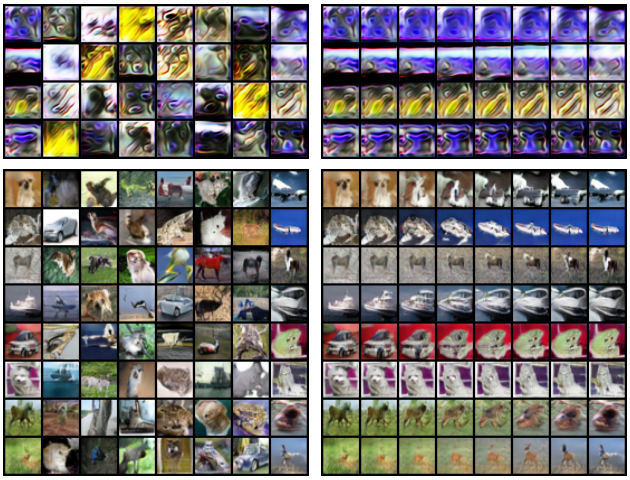


Figure 4: Samples from two trained SNGAN (Miyato et al., 2018) generators (top, bottom). Top: spectral collapse during training leads to degenerate samples at test time. Bottom: diverse samples from the generator. Left: random samples. Right: latent code interpolation between left-most and right-most random samples.

Experiments with 32×32 size are conducted with the CIFAR-10 (Krizhevsky et al., 2009) dataset, consisting of 50K images of 10 classes. For images of 48×48 size, we utilize the unlabeled split of the STL-10 (Coates et al., 2011) dataset, consisting of 100K images.

All experiments are trained on a single 11GB GPU for 100K generator updates, 5 discriminator updates per one generator update, batch size 64, Adam optimizer with betas $[0, 0.9]$, and the learning rate $2e{-}4$, with linear decay of the learning rate to zero towards the end of the training. All code is implemented in `PyTorch` (Paszke et al., 2019) for consistency of comparisons. For spectral normalization, we use the standard `torch.nn.utils.spectral_norm` with one power iteration per update. To transform the learned parameters of SVDP and STTP into weight matrices, we employ `torch-householder` (Obukhov, 2021) to compute (6) and `opt_einsum` (Smith and Gray, 2018) to contract tensor diagrams in Fig. 2. See details in Sec. 6, 7.

For assessing the generated results, we use the *Inception Score* (IS) (Salimans et al., 2016) (higher is better) and the *Fréchet Inception Distance* (FID) (Heusel et al., 2017) (lower is better). Both IS and FID are known to correlate with the human perception of sample quality, however, IS is known to fluctuate inadequately in the presence of synthetic artifacts. We also report the *Kernel Inception Distance* (KID) (Binkowski et al., 2018) (lower is better), as it was shown to have no

Table 1: Results of training the SNGAN model with an unparameterized generator ($\mathcal{G}$) and a reduced discriminator ($\mathcal{D}$) on two image datasets. Methods: SN – Spectral Normalization (Miyato et al., 2018), SR – Spectral Regularization (Liu et al., 2019), SVDP-C, STTP-C – the proposed methods with the identity spectrum and rank $r = 64$. Metrics: $\mathcal{Z}$ – the ratio (11) of counts of the learned discriminator parameters relative to SN (lower is better), IS – Inception Score (higher is better), FID – Fréchet Inception Distance (lower is better), and KID – Kernel Inception Distance (lower is better). The reduction of discriminator features causes a spectral collapse in layer weight matrices in the original (SN) setting. Both SR and our methods prevent spectral collapse.

Dataset	CIFAR10				STL10−48			
Metric	$\mathcal{Z}\downarrow$	IS ↑	FID ↓	KID×100 ↓	$\mathcal{Z}\downarrow$	IS ↑	FID ↓	KID×100 ↓
SN	100	5.48±0.44	55.17±3.65	3.84±0.15	100	2.56±0.24	251.6±26.4	28.78±5.28
SR	100	7.17±0.06	27.24±0.80	1.95±0.06	100	3.77±0.32	193.9±11.5	19.55±1.36
SVDP-C	93.1	7.42±0.20	23.76±2.12	1.77±0.24	89.4	3.91±0.36	204.3±16.9	21.48±1.77
STTP-C	87.7	7.38±0.08	24.45±0.48	1.76±0.08	74.1	4.35±0.16	190.6±13.7	19.59±1.96

Table 2: Results of training the SNGAN model with an unparameterized generator ($\mathcal{G}$) and the full discriminator ($\mathcal{D}$) on two image datasets. Added methods: "-L" – learned spectrum, "-R" – learned and regularized spectrum (Jiang et al., 2018). Metrics: see Table. 1. All models parameterized with SVDP and STTP use rank $r = 64$. Both SVDP and STTP result in smaller models with comparable performance.

Dataset	CIFAR10				STL10−48			
Metric	$\mathcal{Z}\downarrow$	IS ↑	FID ↓	KID×100 ↓	$\mathcal{Z}\downarrow$	IS ↑	FID ↓	KID×100 ↓
SN	100	7.99±0.02	18.38±0.30	1.25±0.04	100	8.38±0.11	99.26±0.58	10.56±0.01
SR	100	8.02±0.05	16.89±0.23	1.17±0.02	100	8.85±0.11	93.14±0.56	09.59±0.11
SVDP-C	51.7	7.97±0.08	17.53±0.63	1.23±0.05	14.2	8.49±0.12	98.50±1.60	10.40±0.22
STTP-C	16.7	7.85±0.01	18.93±0.54	1.37±0.01	3.24	8.59±0.11	96.91±0.89	10.22±0.07
SVDP-L	53.3	7.97±0.06	17.18±1.30	1.24±0.15	14.5	8.71±0.04	96.03±1.53	10.18±0.27
STTP-L	18.3	7.96±0.02	18.00±0.56	1.24±0.03	3.51	8.64±0.07	96.04±1.57	10.12±0.14
SVDP-R	53.3	8.02±0.07	17.17±0.37	1.23±0.03	14.5	8.69±0.07	95.75±0.84	10.14±0.18
STTP-R	18.3	7.96±0.01	18.43±0.75	1.32±0.06	3.51	8.67±0.10	97.12±0.68	10.29±0.22

bias, thus making its values comparable across a wider range of evaluation protocols (e.g., different sample and subset sizes). During the evaluation, we sample 50000 images from the generator and compute IS with ten splits, FID with all samples, KID with 100 subsets each 1000 samples. For all metrics, we report the mean and two standard deviations (68%) confidence interval over three runs with different seeds. Evaluation is performed with `torch-fidelity` (Obukhov et al., 2020), which is shown to be consistent with reference implementations.

For each reparameterized model, we calculate the compression ratio using the following formula:

$$\mathcal{Z}(\mathcal{F}) = 100 \times \frac{\sum_{\ell=1}^{L} \mathrm{DOF}(W_\ell) + C_\ell}{\sum_{\ell=1}^{L} \mathrm{numel}(W_\ell) + C_\ell}, \qquad (11)$$

where $\mathrm{DOF}(W_\ell)$ denotes the number of degrees of freedom of the weight matrix W_ℓ with respect to the chosen parameterization, $\mathrm{numel}(W_\ell)$ is the number of elements in the weight matrix W_ℓ, and C_ℓ is the number of parameters not subject to reparameterization, such as bias terms b_ℓ and parameters of batch norms.

4.2 Effect of Spectral Constraints

We compare with two methods: spectral normalization (SN) (Miyato et al., 2018) and spectral regularization (SR) (Liu et al., 2019) applied in place of SN. First, we want to verify that the proposed parameterization prevents spectral collapse when it is known to happen under SN. We use the same strategy as Liu et al. (2019), who showed that reducing the number of channels in all layers of the discriminator leads to both spectral and mode collapses. We reduce the number of channels in the discriminators of SNGAN-32 and SNGAN-48 by 4× (128 to 32) and 16× (1024 to 64), respectively. Such reduction limits the discriminator's ability to provide good updates to the generator. We use the identity spectrum to leave out the spectrum factor of variation. All models are trained with a rank $r \leq 64$.

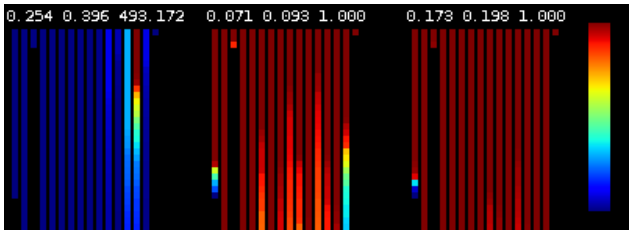


Figure 5: Visualization of singular values (SV) of SNGAN-48 discriminators ($\mathcal{D}$), trained with SN (Miyato et al., 2018) (left), SVDP-L (middle), and SVDP-R (right). Each column represents one layer of $\mathcal{D}$ in the feed-forward order. Column height represents the rank of a particular layer. For example, the first layer is a Conv2D with 64 outputs, 3 inputs, and 3×3 kernel; hence its rank is 27. Only the top 32 SVs are shown. Three annotations at the top correspond to the minimum SV in $\mathcal{D}$, the minimum SV in the observed slice of 32 SVs (blue), and the maximum SV in $\mathcal{D}$ (red). SN spectrum grows unconstrained and unbalanced, which causes a spectral collapse in more restricted configurations, such as the one from Table 1.

The first observation is that SR consistently prevents spectral collapse. We consider SR as an improved version of SN, which performs normalization and maintains weight matrices' stable ranks. It is worth noting that SR could be seen as a full-rank method, as it applies SVD on each weight matrix every training step. Therefore, we group SN and SR in Table 1 as baselines. We aim to demonstrate that our low-rank method is better than SN and around or better than SR. Results of the reduced discriminator experiments confirm that both SVDP and STTP consistently outperform SN even with the identity spectrum.

In the second set of experiments, we use the original networks with unaltered channels of the discriminators; hence both SN and SR perform well with no spectral collapse. We compare various strategies of spectrum control in Table 2. As before, our method outperforms SN even with the identity spectrum (-C suffix). Letting singular values loose (-L suffix) improves our method over the identity spectrum, but the best performance is achieved with the D-optimal regularizer (-R suffix). We conjecture that it allows the optimizer to take shortcuts when traversing the manifolds of W; however, such a regularizer makes singular values end up close to 1, as confirmed by Fig. 5. All models from Table 2 produce visually appealing results (Fig. 4, bottom).

4.3 Rank Utilization Study

For simplicity, all our experiments use the same maximum rank hyperparameter (denoted as r) in all layers. This design does not limit the model's ability to learn low-rank projections when learning the spectrum (e.g., with D-optimal regularizer), as any rank-ρ projection ($\rho < r$) can be achieved by setting a subset of $r - \rho$ singular values to zero. Thus, increasing r leads to a higher expressive power in the layers where this is required. We analyze both SVDP and STTP on SNGAN: reparameterizing only the discriminator and reparameterizing both the discriminator and the generator.

Reparameterizing Discriminator We keep the generator intact and unconstrained; for the discriminator, we apply both SVDP and STTP with $r \in \{32, 64\}$.

Note that the reported performance reflects the quality of the same unconstrained generator, trained together with varying discriminator constraints. This can be seen as another form of limiting the discriminator's capacity (by rank instead of the number of features), similar to the reduced discriminator setting (Table 1).

Table 3 (left) shows the results of this group of experiments: $r = 64$ gives the best performance, suggesting that it is the optimal low-rank regime for SNGAN-32 in the given settings. STTP with $r = 64$ has a similar performance to SVDP with $r = 32$, with only $\sim$66% of the number of parameters.

Reparameterizing Generator It remains unclear what kind of effect the rank reduction may have on the generator. To this end, we choose a sufficiently good discriminator setting with $r = 64$ SVDP and explore the same range of ranks with SVDP and STTP, only now varying just the generator's parameterization. To allow for the large magnitude of inputs to the last tanh layer, we leave the first fully-connected layer in its original, unparameterized form.

The results can be seen in Table 3 (right): STTP is more sensitive to rank reduction than SVDP; however, matching SVDP performance is possible at a lower parameter count with twice a larger rank. We additionally visualize the performance-compression frontier (Fig. 6), which confirms that STTP is more rank-efficient than SVDP in the examined setting and low-rank regime.

4.4 Image Classification

We additionally study the effect of applying both SVDP and STTP in a more traditional image classification setting. To this end, we train an image classification CNN on the CIFAR-10 dataset and report the Top-1 accuracy over the validation split. For the CNN, we use a Wide ResNet (Zagoruyko and Komodakis, 2016) with 28 layers and widening factor 10 (WRN-28-10). We train for 100K steps with SGD, the initial learning rate 0.1, decaying linearly to 0, momentum 0.9, weight

Table 3: Results of varying the rank hyperparameter (r) in GAN modules on CIFAR-10 image generation with SVDP and STTP separately applied to the generator ($\mathcal{G}$) and the discriminator ($\mathcal{D}$). Experiments with parameterized $\mathcal{D}$ (left) use unconstrained $\mathcal{G}$. Additionally, experiments with parameterized $\mathcal{G}$ (right) use SVDP of $\mathcal{D}$ with rank $r = 64$. All experiments use the D-optimal regularizer. Performance of STTP with $2-3\times$ rank is similar to that of SVDP, but with higher compression $\mathcal{Z}$ (11) in terms of the number of parameters (DOF).

	Scope	Discriminator ($\mathcal{D}$)				Generator ($\mathcal{G}$)			
	Metric	$\mathcal{Z}\downarrow$	IS $\uparrow$	FID $\downarrow$	KID$\times$100 $\downarrow$	$\mathcal{Z}\downarrow$	IS $\uparrow$	FID $\downarrow$	KID$\times$100 $\downarrow$
SVDP	r=32	27.71	7.94±0.08	18.33±0.44	1.36±0.01	25.14	7.82±0.07	19.46±0.64	1.39±0.03
	r=64	53.36	8.02±0.07	17.17±0.37	1.23±0.03	37.13	8.05±0.07	17.09±0.11	1.23±0.05
STTP	r=32	06.44	7.51±0.11	25.16±0.57	1.88±0.10	14.61	7.07±0.09	31.32±2.09	2.29±0.13
	r=64	18.33	7.91±0.06	17.89±0.65	1.28±0.01	18.82	7.71±0.03	22.06±0.05	1.61±0.02

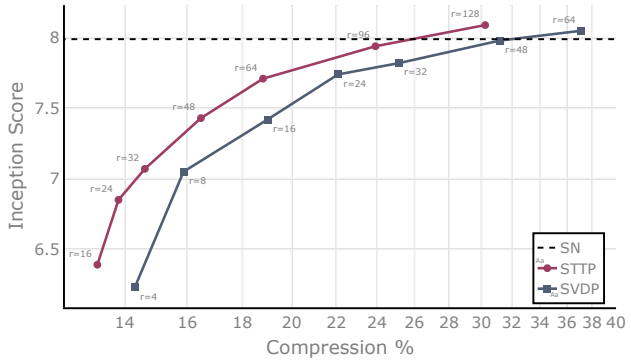


Figure 6: Performance-compression curves of the SNGAN generator, while the discriminator is statically parameterized with SVDP $r = 64$. Performance is the Inception Score of $\mathcal{G}$ (higher is better); compression is the ratio (11) of the parameterized and the original $\mathcal{G}$ parameter counts (lower is better). Each data point is annotated with the rank r of $\mathcal{G}$ used to produce the score. STTP consistently outperforms SVDP, achieves higher scores with larger ranks and fewer parameters, suggesting a better rank-efficiency of STTP.

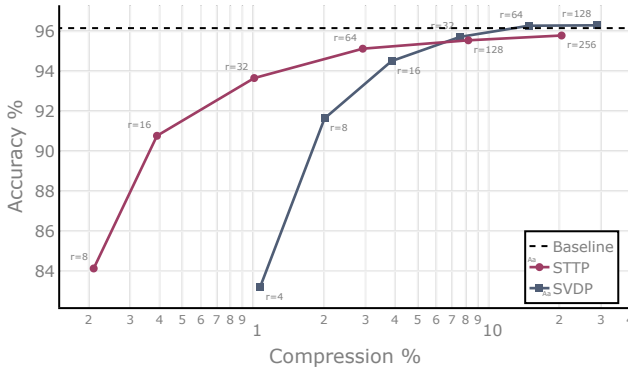


Figure 7: Performance-compression curves of the proposed methods in image classification with WRN-28-10 on CIFAR10. Performance is the Top-1 Accuracy of the image class prediction model (higher is better); compression is the ratio (11) of the parameterized and the original model's parameter counts (lower is better). Each data point is annotated with the respective rank r hyperparameter used within model layers to produce the score. STTP outperforms SVDP in the low-rank regime while achieving higher scores with larger ranks.

decay 1e-4. The spectrum is learned and regularized in all experiments (the '-R' flavor of models). The first convolutional layer is kept unparameterized to allow for a large global Lipschitz constant required to approximate one-hot distributions in the softmax layer.

As shown in the performance-compression plot in Fig. 7, STTP produces highly compressed models with less than 10% parameters of the original model, which still achieve competitive performance. In this extreme compression regime, STTP clearly outperforms SVDP by a large margin (e.g., 10% performance gap at a 1% compression ratio). However, SVDP is suitable for moderate compression of models, capable of achieving full uncompressed performance. These observations agree with those made in the GAN setting (Sec. 4.3).

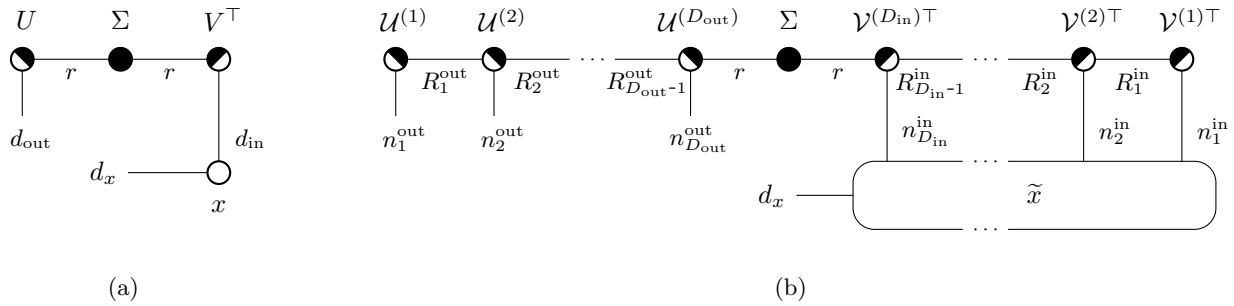
5 Conclusion

We presented the Spectral Tensor Train Parameterization (STTP), a novel low-rank parameterization of weight matrices of convolutional and linear layers with embedded spectral properties. We analyzed the parameter efficiency of the proposed parameterization in the image classification setting, compared it to the SVD parameterization, and concluded that it permits efficient rank utilization with fewer learned parameters. Finally, we analyzed our parameterization in the GAN setting and showed that it leads to better training stability. Future research directions may include finding optimal per-layer rank selection policies (e.g., using Neural Architecture Search) and analyzing the induced sparsity effect on model biases and rare class performance.

Acknowledgements

This work is funded by Toyota Motor Europe via the research project TRACE-Zurich. We thank NVIDIA for GPU donations and Amazon Activate for EC2 credits. Computations were also performed on the Leonhard cluster at ETH Zurich. We also thank the anonymous reviewers for the valuable feedback and time spent.

Spectral Tensor Train Parameterization of Deep Learning Layers Supplementary Materials



(a) (b)

Figure 8: Tensor diagrams of a product of (a) SVDP and (b) STTP of a weight matrix $W \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ and an input $x \in \mathbb{R}^{d_{\text{in}} \times d_x}$, where d_x is a batch or any other second dimension. See legend in Fig. 2 caption. $\widetilde{x}$ is obtained by tensorizing x along the dimension d_{in} using the same dimension factorization as the corresponding dimension of the matrix W. The factorized dimensions of $\widetilde{x}$ are connected with the recipient dimensions of TT-cores $\mathcal{V}^{(k)}$. The contraction order of all connected edges defines FLOPs and memory requirements for computing $y = Wx$. After the contraction, dimensions of the output $\widetilde{y}$ can be flattened to recover y.

6 Computational Shortcuts of Low-Rank Affine Mappings

Both SVDP and STTP support "decompression" of the weight matrix $W \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$, which can be used for computing the mapping $\omega(x)$ directly as $y = Wx$ for some input $x \in \mathbb{R}^{d_{\text{in}} \times d_x}$. The last dimension of the input x may correspond to the batch dimension, so its value may be large during neural network training or equal to 1 during inference. The low-rank structure of the proposed parameterizations allows for taking certain computational shortcuts for computing either W or the mapping output y, as measured in floating-point operations (*FLOPs*).

SVDP The number of FLOPs required to decompress W given U, Σ, and V is $r \min(d_{\text{out}}, d_{\text{in}}) + 2rd_{\text{out}}d_{\text{in}}$. Computing $y = Wx$ then takes another $2d_{\text{out}}d_{\text{in}}d_x$ FLOPs. When $r \ll \min(d_{\text{in}}, d_{\text{out}})$, and $d_x = 1$, computing $y = U(\Sigma(V^\top x))$ following the order indicated by parentheses is preferred. Indeed, such computation brings the number of FLOPs down to $r(2d_{\text{in}} + 2d_{\text{out}} + 1)$. Overall, the *optimal contraction order* of a tensor diagram shown in Fig. 8a (which corresponds to arranging parentheses in the expression $U\Sigma V^\top x$) is defined by the sizes of all operands involved in the expression and can be precomputed upon the layer initialization.

STTP After computing the TT-cores of matrices U and V from the underlying parameterizations, there are more than two ways to compute the mapping $\omega(x)$. As before, one can contract the tensor diagram of the matrix W first and then perform the regular computation of $y = Wx$. A slightly more efficient way is to contract matrices U and V and then re-use the approach to the low-rank mapping of SVDP.

Finally, the most efficient approach consists of the following steps: (1) factorization of the first dimension of x into D_{in} factors: $d_{\text{in}} = n_1 \dots n_{D_{\text{in}}}$, (2) tensorization of x into a tensor $\widetilde{x}$ according to the dimension factorization, (3)

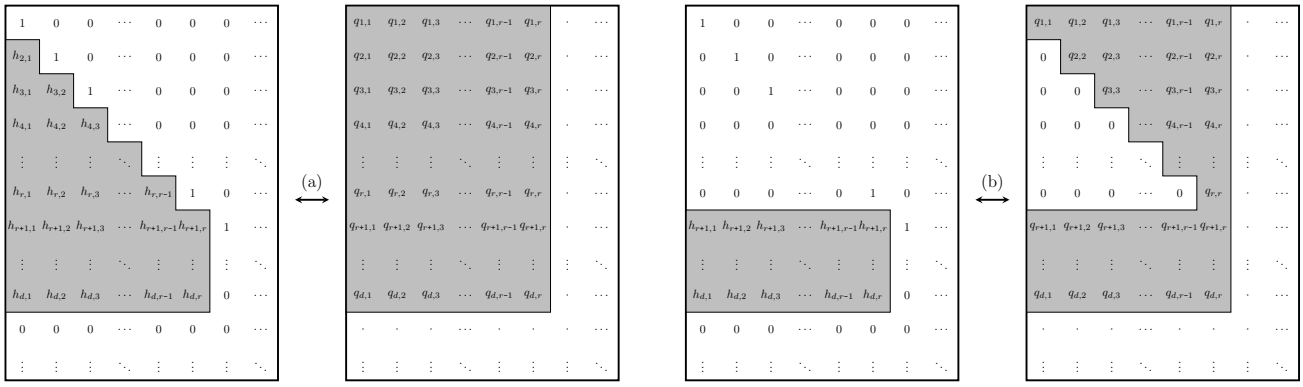


Figure 9: Visualization of the Padded Householder parameterization of orthonormal frames: (a) Full parameterization, (b) Reduced parameterization. Shaded areas with $h_{i,j}$ values represent parameters of reflectors; $q_{i,j}$ values represent orthonormal frame elements, affected by the parameterization. The padded parameterization allows for more efficient batching of orthonormal frames of different sizes for better parallelism at the expense of memory.

connecting factorized dimensions of $\widetilde{x}$ with the respective dimensions of the TT-cores $\mathcal{V}$, and finally, (4) contracting the resulting tensor diagram in Fig. 8b according to the optimal contraction order. While finding the optimal contraction order of a generic tensor diagram is an NP-hard problem, efficient algorithms exist for certain classes of graphs (Smith and Gray, 2018). This approach gives us the lowest possible FLOPs count of computing ω directly in the low-rank space, as both simpler approaches belong to the search space of the contraction order. As in the case of SVDP, the optimal contraction order depends on the topology of the tensor diagram and node sizes. Since the layer dimensions are known in advance, the mapping complexity is not increased at runtime.

7 Batch Householder Transformation

Computations involving the proposed parameterizations are dominated by orthogonal transformations (6). In this section, we discuss some aspects that make our approach feasible as the size and the number of neural network layers grow. Despite Householder transformation being more amenable to SIMD implementation than Givens rotations and matrix exponential maps (Shepard et al., 2015), prior works avoid using them altogether due to the lack of framework support[4], complex implementations, and hardness to scale beyond a handful of layers. We overcome these limitations by utilizing a joint parameterization of orthonormal frames of the same size (Obukhov, 2021). In the context of SVDP, it allows us to generate multiple orthonormal frames of the same size $d \times r$ (potentially belonging to different layers) in a sequence of a total of r batched Householder reflections.

In the context of STTP, all matricized TT-cores are represented as orthonormal frames of a limited set of sizes and can be computed independently of each other. Specifically, the sizes are of the form $R_a n_b \times R_c$, where n_b belongs to the set of all possible factors of weight matrices' sizes in the whole network, and R_a, R_c belong to the set of all possible TT-rank values induced by matrix dimensions and rank r. In practice, we can reduce the set of different sizes of orthonormal frames used in the model by following high-level design recommendations discussed in Sec. 8. These observations lead to the improvement of TT-cores computation parallelism by having fewer different orthonormal frame sizes and a higher number of frames in each batch of a fixed size.

Padded Householder Although batching orthonormal frames of the same size improves parallelism, batches of different sizes are still processed in sequence. Here we show how to trade memory for parallelism and perform parameterizations of multiple orthonormal frames of different sizes in a single batch. Concretely, given a set of sizes $d_1 \times r_1, \dots, d_k \times r_k$, we parameterize the respective orthonormal frames using the proposed Padded Householder parameterization in a batch of k matrices of size $\max(d_1, \dots, d_k) \times \max(r_1, \dots, r_k)$, as shown in Fig. 9a. Indeed, by zeroing $h_{p,i,j} : \forall i > d_p, \forall j > r_p$ in the p-th matrix of parameters, propagating 1 on the diagonal for $j > r_p$, and applying the Householder transformation, the resulting leading sub-matrix is in $\mathrm{St}(d_p, r_p), p = 1 \dots k$.

Householder parameterization padding can also be used together with the Reduced parameterization introduced

[4]Even though an orthogonal transformation implementing (6) can be found in modern automatic differentiation packages as LAPACK bindings (`?ORGQR`), these functions rarely support batching or differentiation with respect to inputs.

in Sec. 3.1 (Fig. 9b) to describe elements in $\mathrm{St}_{\mathsf{U}}(d, r)$. Indeed, both Reduced and Padded variations employ the same transformation and differ only in the placement of constants $\{0, 1\}$. Thus it is possible to perform parameterization of all orthonormal frames needed by the model in a single batch of rank-r orthonormal frames.

8 Neural Architecture Design for Efficient Batching of Orthonormal Frames

Sec. 7 points out the possibility of parallel computation of orthonormal frames, potentially belonging to different layers (and cores in STTP). The ability to compute most of the orthonormal frames of weight matrices in parallel is the defining factor of the compute throughput during training. There are a few neural architecture design traits, which have a direct impact on the effectiveness of such batching with or without padding.

Recall that SVDP of a 2D convolution with the weight matrix $W \in \mathbb{R}^{C_{\text{out}} \times C_{\text{in}} K^2}$ requires the computation of orthonormal frames $U \in \mathbb{R}^{C_{\text{out}} \times R}$ and $V \in \mathbb{R}^{C_{\text{in}} K^2 \times R}$, where $R = \min(r, C_{\text{out}}, C_{\text{in}} K^2)$. To ensure that orthonormal frames from different layers can be batched, one should aim to reduce the amount of variation in the dimensions of matrices U and V belonging to different layers. For example, this is achieved with most residual architectures such as He et al. (2016); Zagoruyko and Komodakis (2016), which contain repetitions of residual blocks. Each unique orthonormal frame size forms a separate batch, which in turn requires a separate function call during training. Such cases include, for example, components of the preamble layer attaching to RGB inputs or layers with unique d_{out} or d_{in} less than r, causing rank demotion to satisfy the constraint $r \leq \min(d_{\text{out}}, d_{\text{in}})$.

STTP declares less strict constraints on the overall network architecture and weight matrix sizes than SVDP. Recall the TT parameterization of an orthonormal frame $U \in \mathbb{R}^{d \times r}$ requires computing dimension factorization $d = n_1 \dots n_D$ for some $n_1, \dots, n_D \in \mathbb{N} \setminus \{1\}$, for example, prime factors of d with repetition. Then U can have a low-rank parameterization through a number of TT-cores (8) with matricized dimensions $R_{k-1} n_k \times R_k$ for $k = 1, \dots, D$. The ranks R_k are defined as $\min(r, R_k^{\max})$ (3). To give a concrete example, consider a convolutional layer with the weight matrix $W \in \mathbb{R}^{16 \times 8 \cdot 3 \cdot 3}$, and $r = 4$. Then matrices $U \in \mathbb{R}^{16 \times 4}$ and $V^\top \in \mathbb{R}^{8 \cdot 3 \cdot 3 \times 4}$ will be tensorized into tensors $\widetilde{U} \in \mathbb{R}^{2 \times 2 \times 2 \times 2 \times 4}$ and $\widetilde{V} \in \mathbb{R}^{2 \times 2 \times 2 \times 3 \times 3 \times 4}$ using the prime factors of the first dimensions of matrices U and $V^\top$. The TT-rank and dimensions of $\widetilde{W}$ induced by such dimensions factorization (9) will be $R = (1, 2, 4, 4, 4, 4, 4, 2, 1)$, $n = (2, 2, 2, 2, 3, 3, 2, 2, 2)$, and the complete set of matricized core sizes will contain: $\{(2 \times 2)^2, (4 \times 4)^2, (8 \times 4)^3, (12 \times 4)^2\}$ (upper index indicates the size of the batch). Thus, the relation of r to the size of the layer and the ability to factorize dimensions of the weight matrices play crucial roles in reducing the variation of orthonormal frame sizes involved in the parameterization. With this in mind, here are a few neural architecture design rules for maximum computation throughput and efficiency with STTP:

- r should be substantially smaller than the maximum dimension of a weight matrix in the whole network (e.g., $r = 64$ with 512 features in the largest layer);
- the set of convolutional filter sizes (e.g., $\{1, 3\}$) in the entire network should be small;
- usage of large (e.g., greater than 3) prime factors should be avoided in r, channel, and filter sizes;
- best throughput can be achieved with r, channel, and filter sizes being powers of a small factor (e.g., 2 or 3).

9 Training Considerations

Optimizer weight decay, L2 regularization The role of regularization with SVDP and STTP is fundamentally different from the regularization of the regular affine layers. Whereas the latter results in simpler models due to the reduction of the Frobenius norm of weight matrices, the former will reduce individual reflectors' magnitudes, which promotes a truncated diagonal structure in weight matrices. This may or may not be the desired effect, depending on higher-level design decisions, such as the presence of skip connections; this topic is well beyond the scope of the current work. Frobenius regularization of the parameterized weight matrices can still be implemented by simply imposing an L2 penalty or performing weight decay of the learned singular values.

Initialization Most weight matrix initialization schemes in deep learning are motivated by norm preservation of the layer mapping (He et al., 2015). While SVDP and STTP achieve the same goal through the embedded spectral properties, a good initialization still plays an important role in the convergence speed. We experimented with three different ways of initializing orthonormal frames in both SVDP and STTP: (1) truncated identity

matrix $I_{d\times r}$; (2) orthogonal initialization with QR decomposition of a random normal matrix (Saxe et al., 2014): $\mathrm{QR}(N_{d\times r})$, where elements of $N_{d\times r}$ are i.i.d. sampled from $\mathcal{N}(0,1)$; (3) orthogonal initialization with a noisy identity matrix $\mathrm{QR}(I_{d\times r}+\alpha N_{d\times r})$. All initialization schemes resulted in a good model performance at the end of the training; however, the noisy identity scheme with $\alpha = 1e{-}4$ exhibited faster convergence in the considered experiments with SNGAN. We conjecture that the best value of α depends on the dimensions d and r.

10 Proof of Proposition 1

The fact that $\mathcal{M}(\mathcal{U}^{(k)}) \in \mathrm{St}(R_{k-1}n_k, R_k)$ implies $I_{R_k} = \mathcal{M}(\mathcal{U}^{(k)})^\top \mathcal{M}(\mathcal{U}^{(k)})$, or in the index notation,

$$\delta_{\mu\nu} = \sum_{\beta_{k-1}=1}^{R_{k-1}} \sum_{i_k=1}^{n_k} \mathcal{M}(\mathcal{U}^{(k)})_{\overline{\beta_{k-1}i_k},\mu}\, \mathcal{M}(\mathcal{U}^{(k)})_{\overline{\beta_{k-1}i_k},\nu} = \sum_{\beta_{k-1}=1}^{R_{k-1}} \sum_{i_k=1}^{n_k} \mathcal{U}^{(k)}_{\beta_{k-1},i_k,\mu}\, \mathcal{U}^{(k)}_{\beta_{k-1},i_k,\nu}, \tag{12}$$

where $\delta_{\mu\nu}$ is the Kronecker delta. To show that $\mathcal{T}(\mathcal{U}^{(1)},\dots,\mathcal{U}^{(D)}) \in \mathrm{St}(n_1\cdots n_D, r)$, let us write the orthogonality condition in index notation (for the ease of notation, we omit ranges in which indices vary):

$$\begin{aligned}
&\sum_{i_1,\dots,i_D} \mathcal{T}(\mathcal{U}^{(1)},\dots,\mathcal{U}^{(D)})_{\overline{i_1\dots i_D},\mu}\; \mathcal{T}(\mathcal{U}^{(1)},\dots,\mathcal{U}^{(D)})_{\overline{i_1\dots i_D},\nu} \\
&\quad= \sum_{i_1,\dots,i_D} \left(\sum_{\beta_0,\dots,\beta_{D-1}} \mathcal{U}^{(1)}_{\beta_0,i_1,\beta_1} \mathcal{U}^{(2)}_{\beta_1,i_2,\beta_2} \cdots \mathcal{U}^{(D)}_{\beta_{D-1},i_D,\mu} \right) \left(\sum_{\beta_0,\dots,\beta_{D-1}} \mathcal{U}^{(1)}_{\beta_0,i_1,\beta_1} \mathcal{U}^{(2)}_{\beta_1,i_2,\beta_2} \cdots \mathcal{U}^{(D)}_{\beta_{D-1},i_D,\nu} \right) \\
&\quad= \sum_{i_1,\dots,i_D} \sum_{\beta_0,\dots,\beta_{D-1}} \sum_{\tilde\beta_0,\dots,\tilde\beta_{D-1}} \mathcal{U}^{(1)}_{\beta_0,i_1,\beta_1} \mathcal{U}^{(1)}_{\tilde\beta_0,i_1,\tilde\beta_1} \cdots \mathcal{U}^{(D)}_{\beta_{D-1},i_D,\mu} \mathcal{U}^{(D)}_{\tilde\beta_{D-1},i_D,\nu},
\end{aligned} \tag{13}$$

and note that β_0 and $\tilde\beta_0$ vary from 1 to 1, so with (12) for $k=1$, we get

$$\sum_{i_1} \mathcal{U}^{(1)}_{1,i_1,\beta_1} \mathcal{U}^{(1)}_{1,i_1,\tilde\beta_1} = \delta_{\beta_1\tilde\beta_1}.$$

The latter expression implies that in the last line of (13), after summing over i_1, only the terms with $\tilde\beta_1 = \beta_1$ remain. We can now apply (12) for $k=2$:

$$\sum_{\beta_1,i_2} \mathcal{U}^{(2)}_{\beta_1,i_2,\beta_2} \mathcal{U}^{(2)}_{\beta_1,i_2,\tilde\beta_2} = \delta_{\beta_2\tilde\beta_2}.$$

Proceeding recursively, we obtain that (13) equals $\delta_{\mu\nu}$, which completes the proof.

11 Proof of Proposition 2

Let us first show that $\mathcal{T}\left(\mathcal{U}^{(1)},\mathcal{U}^{(2)},\dots,\mathcal{U}^{(D)}\right) = \mathcal{T}\left(\widetilde{\mathcal{U}}^{(1)},\widetilde{\mathcal{U}}^{(2)},\dots,\widetilde{\mathcal{U}}^{(D)}\right)$.

Since $\widetilde{\mathcal{U}}^{(k)}_{:,i_k,:} = Q_{k-1}^\top \mathcal{U}^{(k)}_{:,i_k,:} Q_k$, $Q_k^\top Q_k = I$, and Q_0, Q_D are identity matrices of appropriate sizes, we have:

$$\begin{aligned}
\mathcal{T}\left(\widetilde{\mathcal{U}}^{(1)},\widetilde{\mathcal{U}}^{(2)},\dots,\widetilde{\mathcal{U}}^{(D)}\right)_{\overline{i_1\dots i_D},:} &= \left(Q_0^\top \mathcal{U}^{(1)}_{:,i_1,:} Q_1\right)\left(Q_1^\top \mathcal{U}^{(2)}_{:,i_2,:} Q_2\right)\dots\left(Q_{D-1}^\top \mathcal{U}^{(D)}_{:,i_D,:} Q_D\right) \\
&= \mathcal{U}^{(1)}_{:,i_1,:} \mathcal{U}^{(2)}_{:,i_2,:} \dots \mathcal{U}^{(D)}_{:,i_D,:} = \mathcal{T}\left(\mathcal{U}^{(1)},\mathcal{U}^{(2)},\dots,\mathcal{U}^{(D)}\right)_{\overline{i_1\dots i_D},:}.
\end{aligned}$$

Next, let us finally show that $\mathcal{M}(\widetilde{\mathcal{U}}^{(k)}) \in \mathrm{St}(R_{k-1}n_k, R_k)$. Indeed, $\mathcal{M}(\widetilde{\mathcal{U}}^{(k)}) = (Q_{k-1}^\top \otimes I_{n_k})\mathcal{M}(\mathcal{U}^{(k)})Q_k$ since

$$\begin{aligned}
\left(\mathcal{M}(\widetilde{\mathcal{U}}^{(k)})\right)_{\overline{\alpha_{k-1}i_k},\alpha_k} &= \sum_{\alpha_{k-1},\alpha_k=1}^{R_{k-1},R_k} (Q_{k-1})_{\alpha_{k-1},\beta_{k-1}} \mathcal{U}^{(k)}_{\alpha_{k-1},i_k,\alpha_k} (Q_k)_{\alpha_k,\beta_k} \\
&= \sum_{\alpha_{k-1},\alpha_k=1}^{R_{k-1},R_k} \sum_{j_k=1}^{n_k} (Q_{k-1})_{\alpha_{k-1},\beta_{k-1}} \delta_{i_k j_k} \mathcal{U}^{(k)}_{\alpha_{k-1},j_k,\alpha_k} (Q_k)_{\alpha_k,\beta_k} =
\end{aligned}$$

$$= \sum_{\alpha_{k\text{-}1},\alpha_k=1}^{R_{k\text{-}1},R_k} \sum_{j_k=1}^{n_k} (Q_{k\text{-}1})_{\alpha_{k\text{-}1},\beta_{k\text{-}1}} \delta_{i_k j_k} \left(\mathcal{M}(\mathcal{U}^{(k)})\right)_{\overline{\alpha_{k-1} j_k},\alpha_k} (Q_k)_{\alpha_k,\beta_k}$$
$$= \left((Q_{k\text{-}1}^\top \otimes I_{n_k})\mathcal{M}(\mathcal{U}^{(k)})Q_k\right)_{\overline{\alpha_{k-1} i_k},\alpha_k}.$$

Hence,

$$\mathcal{M}(\widetilde{\mathcal{U}}^{(k)})^\top \mathcal{M}(\widetilde{\mathcal{U}}^{(k)}) = \left((Q_{k\text{-}1}^\top \otimes I_{n_k})\mathcal{M}(\mathcal{U}^{(k)})Q_k\right)^\top \left((Q_{k\text{-}1}^\top \otimes I_{n_k})\mathcal{M}(\mathcal{U}^{(k)})Q_k\right)$$
$$= Q_k^\top \mathcal{M}(\mathcal{U}^{(k)})^\top (Q_{k\text{-}1}\left(Q_{k\text{-}1}^\top\right) \otimes I_{n_k})\, \mathcal{M}(\mathcal{U}^{(k)})Q_k = Q_k^\top \mathcal{M}(\mathcal{U}^{(k)})^\top \mathcal{M}(\mathcal{U}^{(k)})Q_k = Q_k^\top Q_k = I_{R_k},$$

which completes the proof.

12 STTP Degrees of Freedom

We consider a tensor diagram from Fig. 2b made compatible with the TT decomposition introduced in Sec. 2 (consisting only of TT-cores) by contracting the matrix Σ into either left or right adjacent TT-core and recovering size-1 legs on the outer-most TT-cores. Such tensor diagram will have the following TT-rank and dimensions (9):

$$\begin{aligned} R &= (1, R_1^{\text{out}}, \ldots, R_{D_{\text{out}}-1}^{\text{out}}, r, R_{D_{\text{in}}-1}^{\text{in}}, \ldots, R_1^{\text{in}}, 1), \\ n &= (n_1^{\text{out}}, \ldots, n_{D_{\text{out}}}^{\text{out}}, n_{D_{\text{in}}}^{\text{in}}, \ldots, n_1^{\text{in}}). \end{aligned} \tag{14}$$

Recall that R and n are indexed in the ranges $[0, D_{\text{out}} + D_{\text{in}}]$ and $[1, D_{\text{out}} + D_{\text{in}}]$ respectively (Sec. 2) and that STTP is obtained by parameterizing its Σ and TT-cores as follows (left to right in Fig. 2b, Sec. 3.3):

- $\mathcal{M}(\mathcal{U}^{(k)}) \in \text{St}_U(R_{k-1}^{\text{out}} n_k^{\text{out}} \times R_k^{\text{out}}), 1 \leq k < D_{\text{out}}$, or equivalently $\text{St}_U(R_{k-1} n_k \times R_k), 1 \leq k < D_{\text{out}}$ (TT-cores of U excluding the last one) in the notation (14), parameterized by $R_{k-1} n_k R_k - R_k^2$ parameters,
- $\mathcal{M}(\mathcal{U}^{(k)}) \in \text{St}(R_{k-1}^{\text{out}} n_k^{\text{out}} \times R_k^{\text{out}}), k = D_{\text{out}}$, or equivalently $\text{St}(R_{k-1} n_k \times R_k), k = D_{\text{out}}$ (the last TT-core of U) in the notation (14), parameterized by $R_{k-1} n_k R_k - R_k(R_k + 1)\ /\ 2$ parameters,
- $\Sigma = \text{diag}(\sigma_1, \ldots, \sigma_r)$ is a matrix of singular values, parameterized by r parameters,
- $\mathcal{M}(\mathcal{V}^{(k)}) \in \text{St}(R_{k-1}^{\text{in}} n_k^{\text{in}} \times R_k^{\text{in}}), k = D_{\text{in}}$, or equivalently $\text{St}(R_k n_k \times R_{k-1}), k = D_{\text{out}} + 1$ (the last TT-core of V) in the notation (14), parameterized by $R_k n_k R_{k-1} - R_{k-1}(R_{k-1} + 1)\ /\ 2$ parameters,
- $\mathcal{M}(\mathcal{V}^{(k)}) \in \text{St}_U(R_{k-1}^{\text{in}} n_k^{\text{in}} \times R_k^{\text{in}}), D_{\text{in}} > k \geq 1$, or equivalently $\text{St}_U(R_k n_k \times R_{k-1}), D_{\text{out}} + 1 < k \leq D_{\text{out}} + D_{\text{in}}$ (TT-cores of V excluding the last one) in the notation (14), parameterized by $R_k n_k R_{k-1} - R_{k-1}^2$ parameters.

Summing up the degrees of freedom of STTP components listed above, and keeping in mind that $R_{D_{\text{out}}} \equiv r$,

$$\begin{aligned} \text{DOF}(W) = &\left[\sum_{k=1}^{D_{\text{out}}-1} R_{k-1} n_k R_k - R_k^2\right] + \left[R_{k-1} n_k R_k - \frac{R_k(R_k+1)}{2} \;\middle|\; k = D_{\text{out}}\right] + r + \\ &+ \left[R_k n_k R_{k-1} - \frac{R_{k-1}(R_{k-1}+1)}{2} \;\middle|\; k = D_{\text{out}}+1\right] + \left[\sum_{k=D_{\text{out}}+2}^{D_{\text{out}}+D_{\text{in}}} R_k n_k R_{k-1} - R_{k-1}^2\right] = \\ &= \left[r - \frac{r(r+1)}{2} - \frac{r(r+1)}{2}\right] + \sum_{k=1}^{D_{\text{out}}+D_{\text{in}}} R_{k-1} n_k R_k - \sum_{\substack{k \in [1, D_{\text{out}}-1] \cup \\ [D_{\text{out}}+1, D_{\text{out}}+D_{\text{in}}-1]}} R_k^2 \\ &= \sum_{k=1}^{D_{\text{out}}+D_{\text{in}}} R_{k-1} n_k R_k - \sum_{k=1}^{D_{\text{out}}+D_{\text{in}}-1} R_k^2, \end{aligned}$$

we arrive at the same dimensionality of the fixed TT-rank tensor manifold as Holtz et al. (2012).

References

Krizhevsky, A., Sutskever, I., and Hinton, G. E. ImageNet Classification with Deep Convolutional Neural Networks. Advances in Neural Information Processing Systems. 2012.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. Attention is All you Need. Advances in Neural Information Processing Systems. 2017.

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., and Adam, H. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. 2017.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. Advances in neural information processing systems. 2014; pp 2672–2680.

Zhang, J., Lei, Q., and Dhillon, I. Stabilizing Gradients for Deep Neural Networks via Efficient SVD Parameterization. Proceedings of the 35th International Conference on Machine Learning. Stockholmsmässan, Stockholm Sweden, 2018; pp 5806–5814.

Lezcano-Casado, M., and Martínez-Rubio, D. Cheap Orthogonal Constraints in Neural Networks: A Simple Parametrization of the Orthogonal and Unitary Group. Proceedings of the 36th International Conference on Machine Learning. 2019; pp 3794–3803.

Shepard, R., Brozell, S. R., and Gidofalvi, G. (2015) The representation and parametrization of orthogonal matrices. *The Journal of Physical Chemistry A 119*, 7924–7939.

Oseledets, I. V. (2011) Tensor-train decomposition. *SIAM Journal on Scientific Computing 33*, 2295–2317.

Yang, Y., Krompass, D., and Tresp, V. Tensor-Train Recurrent Neural Networks for Video Classification. Proceedings of the 34th International Conference on Machine Learning - Volume 70. 2017; p 3891–3900.

Garipov, T., Podoprikhin, D., Novikov, A., and Vetrov, D. P. (2016) Ultimate tensorization: compressing convolutional and FC layers alike. *CoRR abs/1611.03214*.

Novikov, A., Podoprikhin, D., Osokin, A., and Vetrov, D. P. Tensorizing Neural Networks. Advances in Neural Information Processing Systems. 2015.

Obukhov, A., Rakhuba, M., Georgoulis, S., Kanakis, M., Dai, D., and Van Gool, L. T-Basis: a Compact Representation for Neural Networks. Proceedings of the 37th International Conference on Machine Learning. 2020; pp 7392–7404.

Wang, W., Sun, Y., Eriksson, B., Wang, W., and Aggarwal, V. Wide compression: Tensor ring nets. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018; pp 9329–9338.

Lebedev, V., Ganin, Y., Rakhuba, M., Oseledets, I. V., and Lempitsky, V. S. Speeding-up Convolutional Neural Networks Using Fine-tuned CP-Decomposition. 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings. 2015.

Phan, A.-H., Sobolev, K., Sozykin, K., Ermilov, D., Gusak, J., Tichavský, P., Glukhov, V., Oseledets, I., and Cichocki, A. Stable Low-Rank Tensor Decomposition for Compression of Convolutional Neural Network. Computer Vision – ECCV 2020. Cham, 2020; pp 522–539.

Holtz, S., Rohwedder, T., and Schneider, R. (2012) On manifolds of tensors of fixed TT-rank. *Numerische Mathematik 120*, 701–731.

Bigoni, D., Engsig-Karup, A., and Marzouk, Y. (2016) Spectral Tensor-Train Decomposition. *SIAM Journal on Scientific Computing 38*, A2405–A2439.

Wang, L., Huang, J., Huang, K., Hu, Z., Wang, G., and Gu, Q. Improving Neural Language Generation with Spectrum Control. International Conference on Learning Representations. 2020.

Miyato, T., Kataoka, T., Koyama, M., and Yoshida, Y. Spectral Normalization for Generative Adversarial Networks. International Conference on Learning Representations. 2018.

Sedghi, H., Gupta, V., and Long, P. M. The Singular Values of Convolutional Layers. ICLR 2019, New Orleans, LA, USA, May 6-9, 2019. 2019.

Sanyal, A., Torr, P. H. S., and Dokania, P. K. Stable Rank Normalization for Improved Generalization in Neural Networks and GANs. 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020. 2020.

Jiang, H., Chen, Z., Chen, M., Liu, F., Wang, D., and Zhao, T. (2018) On computation and generalization of generative adversarial networks under spectrum control. *Power 5*.

Liu, K., Tang, W., Zhou, F., and Qiu, G. Spectral Regularization for Combating Mode Collapse in GANs. Proceedings of the IEEE International Conference on Computer Vision. 2019; pp 6382–6390.

Che, T., Li, Y., Jacob, A. P., Bengio, Y., and Li, W. Mode Regularized Generative Adversarial Networks. 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings. 2017.

Arjovsky, M., and Bottou, L. Towards Principled Methods for Training Generative Adversarial Networks. 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings. 2017.

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. C. Improved training of wasserstein gans. Advances in neural information processing systems. 2017; pp 5767–5777.

Krizhevsky, A., Hinton, G., et al. (2009) Learning multiple layers of features from tiny images.

Coates, A., Ng, A., and Lee, H. An analysis of single-layer networks in unsupervised feature learning. Proceedings of the fourteenth international conference on artificial intelligence and statistics. 2011; pp 215–223.

Paszke, A. et al. *Advances in Neural Information Processing Systems 32*; Curran Associates, Inc., 2019; pp 8024–8035.

Obukhov, A. Efficient Householder transformation in PyTorch. 2021; https://github.com/toshas/torch-householder.

Smith, D., and Gray, J. (2018) opt_einsum - A Python package for optimizing contraction order for einsum-like expressions. *Journal of Open Source Software 3*, 753.

Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., and Chen, X. Improved techniques for training gans. Advances in neural information processing systems. 2016; pp 2234–2242.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. Gans trained by a two time-scale update rule converge to a local nash equilibrium. Advances in neural information processing systems. 2017; pp 6626–6637.

Binkowski, M., Sutherland, D. J., Arbel, M., and Gretton, A. Demystifying MMD GANs. 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings. 2018.

Obukhov, A., Seitzer, M., Wu, P.-W., Zhydenko, S., Kyl, J., and Lin, E. Y.-J. High-fidelity performance metrics for generative models in PyTorch. 2020; https://github.com/toshas/torch-fidelity, Version: 0.2.0, DOI: 10.5281/zenodo.3786540.

Zagoruyko, S., and Komodakis, N. Wide Residual Networks. Proceedings of the British Machine Vision Conference (BMVC). 2016; pp 87.1–87.12.

He, K., Zhang, X., Ren, S., and Sun, J. Deep Residual Learning for Image Recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016. 2016; pp 770–778.

He, K., Zhang, X., Ren, S., and Sun, J. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV). USA, 2015; p 1026–1034.

Saxe, A. M., McClelland, J. L., and Ganguli, S. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. 2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings. 2014.