DeepTensor: Low-Rank Tensor Decomposition With Deep Network Priors

Vishwanath Saragadam, Randall Balestriero, Member, IEEE, Ashok Veeraraghavan, Fellow, IEEE, and Richard G. Baraniuk, Fellow, IEEE

Abstract—DeepTensor is a computationally efficient framework for low-rank decomposition of matrices and tensors using deep generative networks. We decompose a tensor as the product of low-rank tensor factors (e.g., a matrix as the outer product of two vectors), where each low-rank tensor is generated by a deep network (DN) that is trained in a self-supervised manner to minimize the mean-square approximation error. Our key observation is that the implicit regularization inherent in DNs enables them to capture nonlinear signal structures (e.g., manifolds) that are out of the reach of classical linear methods like the singular value decomposition (SVD) and principal components analysis (PCA). Furthermore, in contrast to the SVD and PCA, whose performance deteriorates when the tensor’s entries deviate from additive white Gaussian noise, we demonstrate that the performance of DeepTensor is robust to a wide range of distributions. We validate that DeepTensor is a robust and computationally efficient drop-in replacement for the SVD, PCA, nonnegative matrix factorization (NMF), and similar decompositions by exploring a range of real-world applications, including hyperspectral image denoising, 3D MRI tomography, and image classification. In particular, DeepTensor offers a 6 dB signal-to-noise ratio improvement over standard denoising methods for signal corrupted by Poisson noise and learns to decompose 3D tensors 60 times faster than a single DN equipped with 3D convolutions.

Index Terms—Tensor decomposition, matrix factorization, low-rank completion, deep network, self-supervised learning.

I. INTRODUCTION

Low-rank representations of matrices and tensors are truly ubiquitous and applied across all fields of science and engineering, from statistics [1], [2], [3], [4], [5] to control systems [6], [7] to computer vision [8], [9], [10], and beyond. Low-rank representations seek to represent a large matrix/tensor as a product of smaller (and hence lower rank) matrices/fibers. For instance, the classical approach to representing matrices in a low-rank manner is via the singular value decomposition (SVD), which expresses a matrix as a product of two smaller orthonormal matrices (containing the singular vectors) and a diagonal matrix (containing the singular values). Thresholding the singular values creates a matrix that inhabits a lower dimensional subspace. The SVD is a pervasive technique for data preprocessing and dimensionality reduction across a wide range of machine learning (ML) applications, including principal component analysis (PCA) and data whitening.

Nevertheless, the SVD and PCA are not without their shortcomings. SVD/PCA is an optimal low-rank decomposition technique only under a narrow set of assumptions on the statistics of the signal and noise in the task at hand [11]. When the signal or noise is non-Gaussian, the resulting decomposition is not optimal and results in a subspace that differs from the true low-rank approximation of the underlying matrix. These issues have been addressed somewhat successfully in the past with several signal- and application-specific regularizers that include sparsity on error [12], [13], [14], total variation penalty [15], [16], [17], and data-driven approaches [18], [19]. The key observation is that a good signal model can act as a strong regularizer for estimating the low-rank factors. Unfortunately, finding a useful signal model/regularizer for a new application can be a daunting task. Does there exist a generalized regularizer that can encompass a large class of signals and applications? We found the answer to be hidden implicitly in deep networks (DNs).

In this paper, we propose DeepTensor, a new approach for low-rank matrix/tensor decomposition that is robust to a wide class of signal and noise models: Our core enabling observation is that DNs produce signals that are implicitly regularized due to the networks’ inherent inductive bias. We exploit DNs as priors by representing a matrix/tensor in terms of factors output from a set of untrained generative networks (see Fig. 1). The parameters of the networks are learned in a self-supervised manner for each matrix/tensor using a simple MSE loss between the resulting decomposition and the product of the deeply generated factors. The inductive bias of the generative networks enables DeepTensor to better identify the underlying low-dimensional subspace while rejecting noise, resulting in a more accurate estimate of the noise-free matrix/tensor.

DeepTensor is a computationally efficient, drop-in replacement for many existing matrix/tensor factorization approaches
that combines the simplicity of low-rank decomposition with the power of deep generative networks: Further, DeepTensor can significantly improve the performance of downstream tasks, such as image classification, that rely on low dimensional representation. We empirically back these claims via experiments on a wide variety of real-world tasks, including denoising with low-rank approximation, “eigenfaces” [20] for facial recognition (see Fig. 2), solving linear inverse problems with compressively sensed videos, tensor denoising, and recovery of 3D volumes from computed tomographic (CT) measurements. We also highlight DeepTensor’s computational efficiency and scalability for large and higher-order tensor decomposition by demonstrating that it offers a 60 × or more speedup for decomposing 3D tensors as compared to a single generative network equipped with full 3D convolutions.

II. BACKGROUND AND PRIOR WORK

DeepTensor leverages classical work on low-rank approximation and recent self-supervised learning techniques with DNs [21], [22]. We provide a brief overview of these topics to introduce our notation and intuition and to set context for our work.

Low-Rank Approximation: Low-rank approximation seeks to represent a matrix $X \in \mathbb{R}^{M \times N}$ of rank $R = \min(M,N)$ as a product of two smaller matrices, $U \in \mathbb{R}^{M \times k}, V \in \mathbb{R}^{N \times k}$ where $k$ is generally taken to be smaller than $R$. The specific constraints on $U, V$, and the desired objective give rise to different types of low-rank approximation algorithms. For example, one recovers PCA [23], nonnegative matrix factorization (NMF) [24] and k-means [25] via

$$\begin{align*}
\min_{U, V} \|X - UV^T\|_F \quad \text{s.t.} \quad U = V^T \quad \text{(PCA/SVD)} \quad (1) \\
\min_{U, V} \|X - UV^T\|_F \quad \text{s.t.} \quad U \geq 0, V \geq 0 \quad \text{(NMF)} \quad (2) \\
\min_{U, V} \|X - UV^T\|_F \quad \text{s.t.} \quad [V]_{:,k} \in \{e_1, \ldots, e_n\} \quad \text{(k-means)} \quad (3)
\end{align*}$$

where $e_k$ is the $k^{th}$ euclidean canonical basis vector, and $[V]_{:,k}$ is the $k^{th}$ column of $V$. Applications of low-rank approximation are extremely diverse ranging from denoising [26], [27], compression [28], clustering for anomaly detection [29], and forecasting [30].

The Achilles’ heel of low-rank approximation is non-Gaussian signal and/or noise statistics. There are been many extensions and variants of the SVD algorithms, such as Robust PCA [12] that improves robustness in learning the low-rank matrices against outliers in the data. Similarly, other non-Gaussian noise settings, different metrics and constraints could be employed to obtain the most adapted $U, V$ decomposition to solve the task at hand. This can be well understood based on the generative models corresponding to low-rank approximation techniques such as Probabilistic PCA [31] from which it is clear that PCA is optimal under a Gaussian noise model, and in the presence of say a Laplacian noise, an $\ell_1$ reconstruction loss should be used instead. Prior work identified approaches to tackling tensor decomposition with unknown noise statistics via bayesian optimization [32], [33], modeling noise distribution as a mixture of Gaussian [34], and using a decomposition inspired by Kronecker product [35], [36]. While the approaches are
promising for several tensor decomposition applications, none of the previous works leverage inductive bias of DNs.

There has been some research in exploiting trained DNs for matrix factorization especially in the settings of nonnegative matrix factorization [37], magnetic resonance imaging (MRI) denoising [19], and tensor completion [18]. The key idea is that the statistics of training data can help regularize the inverse problem of matrix factorization. Such techniques however suffer from dataset biases, and require very large pools of data to be effective.

Deep Networks as Implicit Regularizers: DNs have emerged very rapidly from classification and regression applications where they have reached superhuman performance across a wide range of datasets and tasks [38]. More recently, the use of DNs has diversified—an illustrative and important example for this paper is the Deep Image Prior (DIP) model [21]. In this setting, a DN $f$ is used as a constrained projection of a random noise vector $z$ to fit a target sample $x$ as follows

$$
\min_{\theta} \| f_{\theta}(z) - x \|_2^2.
$$

When the architecture of the DN is carefully picked, the estimation of the input $x$ is denoised, and well reconstructed even in the presence of missing values. From the implicit regularization field it is understood that the above problem is equivalent to some problem

$$
\min_W \| Wz - x \|_2^2 + R(W),
$$

where $W$ is a weight matrix for an equivalent fully connected deep network, and $R(W)$ is a regularization term that directly depends on the choice of the DN architecture [39]. The key result that we will leverage throughout this paper is that searching for the DN parameters producing the desired result in (4) is equivalent to searching in the space of regularizers in (5).

Related Work Combining DNs and Low-Rank Decomposition: Applying DNs in a self-supervised manner for matrix decomposition has received surprisingly little attention. The closest work is by Aittala et al. [40], which regularized a matrix factorization for a specific video imaging problem using generative networks. The video sequence was represented by a generative network equipped with 3D convolutional kernels, while the light transport matrix was represented as a linear combination of the input video sequence multiplied by another generative network equipped by 2D convolutional kernels. DeepTensor is in many ways inspired by the factorization idea proposed by Aittala et al. [40] but goes beyond light transport matrices and can be applied to a wide variety of problem settings.

A related but different approach to DeepTensor is the work by Bacca et al. [41], who identified that hyperspectral images (which are modeled as 3D tensors) can be represented as the output of a single generative network equipped with 3D convolutional kernels. The work by Bacca et al. [41] is a promising framework for solving inverse problems in hyperspectral imaging, but are not aimed at low-rank matrix factorization—which is the focus of this paper. The key difference between their work and DeepTensor is that the input to their 3D network is a low-rank Tucker tensor; in contrast we output a low-rank Tucker tensor.

III. DEEPTENSOR DECOMPOSITION

A. Low-Rank Decomposition With Deep Network Prior

Matrix Decomposition: Consider the low-rank decomposition in (1). If we include regularizers for $U$ and $V$, we obtain the following optimization function

$$
\min_{U,V} \| X - UV^\top \|_F^2 + R(U,V),
$$

where $R(U,V)$ is a regularizer for $U,V$. Instead of having explicit regularizers on the left and right matrices, we model $U,V$ as the outputs of generative networks $f_U,f_V$, which yields

$$
\min_{\theta_U,\theta_V} \| X - f_U(z_U) f_V(z_V)^\top \|_F^2,
$$

where $z_U$ and $z_V$ are randomly initialized inputs to the networks $f_U,f_V$ respectively that have parameters $\theta_U,\theta_V$. The networks’ output are of the same shape as the desired $U$ and $V$ matrices from (6). We note here that there is no further regularizer on the matrices—any regularization comes from the implicit prior of the DN itself, which makes it an appealing choice to solve a diverse type of signals and noise settings. Fig. 3 visualizes the regularization offered by DNs for the task of rank-20 decomposition. Over 100 iterations, the error for signal reduces by two orders of magnitude, while the error reduces by less than one order for noise. This slow fitting to noise is a result of implicit bias of DNs which is leveraged by DeepTensor.

Tensor Decomposition: The task of tensor decomposition finds numerous applications and is an active area of research, where the major difficulty rises from defining an appropriate regularizer/basis constraint (recall (3)). Any constraint on the factor matrices can be expressed as regularization functions. Hence, given the following general decomposition problem

$$
\min_{U,V,\ldots,W} \| X - U \otimes V \otimes \cdots \otimes W \|_F^2 + R(U,V,\ldots,W),
$$

with $X$ a $k$-dimensional tensor, one needs to specify the correct regularizer ($R$) such that the produced decomposition is adapted for the task at hand. This search is mostly understood in
Fig. 4. Separable convolutions leads to speedup. An added benefit of deep low rank decomposition is that the convolutions are separated. Hence for a 3D volume, we can exploit one 2D and one 1D network for factorization instead of a 3D network – resulting in two orders of magnitude speed up.

narrowly defined settings such as Gaussian noise and Gaussian latent space factors. Instead, DeepTensor seeks to solve the the following optimization to compute the decomposition

$$\min_{\theta_U, \theta_V, \ldots, \theta_W} \|X - f_U(z_U) \otimes f_V(z_V) \otimes \cdots \otimes f_W(z_W)\|_2,$$

where $f_U, f_V, \ldots, f_W$ are $k$ DNs that are fed with input vectors $z_U, z_V, \times, z_W$ with parameters $\theta_U, \theta_V, \theta_W$ respectively. Note that this decomposition closely resembles the parallel factor analysis (PARAFAC); one can instead formulate a Tucker representation which includes an extra core tensor.

Dimensionality Splitting Brings Tractability Without Sacrificing Performance: It is possible to extend 2D DN-based priors to higher dimensions such as 3D and 4D tensors. However, doing so requires convolutional networks equipped with 3D and 4D convolutional kernels.

The goal of this section is to study the exponential computational time gain offered by separating the dimensions of the tensor and performing the tensor products of multiple independent DN outputs as per (8) versus evaluating a DN equipped with high-dimensional convolutional kernels. For high-dimensional DN, the form will be an extension of Ulyanov et al. [21] and can be written as,

$$f_{U,V,W}(z_U \otimes z_V \otimes \cdots \otimes z_W).$$

To understand the benefits of those two different cases we approximate a 3D magnetic resonance imaging (MRI) volume of size $128 \times 128 \times 150$ with three types of networks — one 2D and 1D that is a natural parametrization that captures the dependence of the first two dimensions $f_{U,V}(z_{U,V}) \otimes f_W(z_W)$, three 1D (8), and one 3D networks (9). Fig. 4 shows the plot of mean squared error (MSE) as a function of time for the three approaches. We note that using three 1D networks is faster than using a 2D and 1D network, and both are two orders of magnitude faster than using one 3D network. Additionally, our (separate) parametrization has the benefit of keeping interpretability since one has access to each generated low-rank matrix that combine to form the observed data matrix $X$. The choice between using networks equipped with 2D and 1D convolutions, and networks equipped with 1D convolutions is specific to the task at hand. Tensors such as videos and hyperspectral images benefit from networks equipped with 2D and 1D convolutions. In contrast, tensors from multi-dimensional face databases [43] or computed tomographic (CT) images benefit from a full tensor decomposition.

Unless explicitly specified, we utilize overparameterized networks for the factor matrices, similar to DIP [21] (see Fig. 5). However, it is possible to use underparameterized networks instead, similar to the deep decoder architecture [22]. The advantage of the latter is that the learned parameters can be used as the compressed version of the tensor being decomposed. In contrast, for pure data imputation and denoising the DIP version should be preferred. We compare the two choices in the upcoming section.

Optimization Procedure: We optimized for the parameters of the networks that output the factor matrices using stochastic gradient descent. Specifically, we employed the ADAM optimizer [44] and implemented the optimization using PyTorch framework [45]. For all experiments, we optimized an $\ell_2$ loss function between the data and the product of outputs of DNs.
We generated random $N$ to $\[11\] \times X$ that is common in $N$. The performance of SVD-diagonalization shows a plot of peak signal-to-noise ratio (PSNR) as $X \times Y = X \times Z$ are equivalent choices – a very low learning rate of $N$. The matrix. The entries of the matrix $X$ are iid Gaussian noise with standard deviation of 0.1 units: $Y$ shows results for varying learning rate and its schedule. $X$ and the intrinsic dimension (rank) was 10. We generated data $Y$ that is suited for a range of non-Gaussian noise models.

The primary purpose of choosing a simple loss function was to emphasize on the regularization capabilities of the DNs. In practice, it is possible to choose a more appropriate loss function for each specific problem.

### B. Validation Experiments

#### Low-Rank Completion

We generated random $64 \times 64$ dimensional matrices with rank varying from 10 to 60. The left and right matrices were generated as either iid Gaussian values (with a standard deviation of 1 unit), or random, piecewise constant signals, emulating visual signals. We then added one of the three types of noise:

- iid Gaussian noise with standard deviation of 0.1 units: $Y = X + N(0, 0.1)$
- Poisson noise with mean at each entry of the matrix lying in the range $[0, 1000]$: $Y = \mathcal{P}(1000X)$, which is common in visual signals
- Rician noise with standard deviation of 0.02 units: $Y = \sqrt{(X + N(0, 0.02))^2 + N(0, 0.02)^2}$ that is common in MRI measurements.

Fig. 6 shows a plot of peak signal-to-noise ratio (PSNR) as a function of rank for various signal and noise types averaged over 10 realizations. We make two observations. First, when the noise is Gaussian, DeepTensor has similar performance to SVD/PCA. This is expected, since SVD/PCA is known to be the optimal low-rank decomposition for white Gaussian matrices [11]. Second, for other noise settings, such as Poisson or Rician, DeepTensor has a far superior performance across all rank values. This empirically establishes that DeepTensor is well suited for a range of non-Gaussian noise models.

**Principal Component Analysis**: The performance of SVD-based PCA degrades under noisy conditions or when samples are limited. To verify how DeepTensor can benefit PCA, we generated variable number of data points with a known intrinsic matrix generated as iid Gaussian random variables with mean 0 and standard deviation of 1. The feature dimension was 64 and the intrinsic dimension (rank) was 10. We generated data via a linear combination of the columns where the weights were drawn from an iid Gaussian distribution with mean 0 and standard deviation. We then added iid Gaussian noise to the data matrix with varying levels of standard deviation. Fig. 7 plots the accuracy of estimating the PCA components for varying noise levels and number of samples averaged over 10 realizations. We observe that with low noise or a large number of samples, SVD-based PCA and DeepTensor have similar performance. However, with high noise or a limited number of samples, DeepTensor significantly outperforms SVD-based PCA.

### C. Sensitivity to Learning Parameters and Architectures

We now discuss how the choice of learning rate scheduling and the network architecture affect the final accuracy obtained by DeepTensor.

**Learning Rate Scheduling**: The choice of learning rate and its scheduling directly affects the maximum achievable accuracy. To gain empirical insight into how the rate and its scheduling affect the training process, we performed a rank-16 low-rank decomposition of a $64 \times 64$ matrix. The entries of the matrix were drawn from iid Gaussian distribution. We then varied the learning rates from $10^{-5}$ to $10^{-2}$, and chose four learning rate schedulers, namely fixed with no change in learning rate, step scheduler where the learning rate was multiplied by 0.99 after every 2,000 epochs, exponential scheduler with a multiplication factor of 0.9999, and cosine annealing-based scheduling [46].

Table I shows results for varying learning rate and its schedule. We note that a fixed scheduler results in higher PSNR across all learning rates. Moreover, learning rates ranging from $10^{-4}$ to $10^{-2}$ are equivalent choices – a very low learning rate of $10^{-5}$ resulted in poorer PSNR. The two observations above imply that DeepTensor does not require complex learning rate scheduling and is robust to the learning rates.

**Stopping Criterion**: The stopping criterion for optimal approximation accuracy is a function of input noise distribution and network architecture. To demonstrate this dependence, we performed a rank-16 low-rank decomposition of $64 \times 64$ matrices with entries drawn from iid Gaussian distribution. We then utilized an under-parameterized, and over-parameterized...
DeepTensor is a more robust alternative for PCA under low SNR or small sample size. The SVD is the core computation of PCA. We generated data with varying number of samples drawn from a Gaussian distribution with varying standard deviation. We then computed PCA components via SVD and DeepTensor decomposition. We observe that DeepTensor is particularly advantageous when the data is corrupted by a large noise, or the number of samples are limited.

| Max learning rate | Scheduler | Fixed  | Step  | Exponential | Cosine annealing |
|-------------------|-----------|--------|-------|-------------|-----------------|
| $10^{-2}$         |           | 17.6 ± 0.7 | 17.1 ± 0.4 | 16.6 ± 0.4 | 16.5 ± 0.3 |
| $10^{-3}$         |           | 17.5 ± 0.6 | 17.1 ± 0.5 | 16.6 ± 0.4 | 16.6 ± 0.3 |
| $10^{-4}$         |           | 17.6 ± 0.6 | 17.1 ± 0.4 | 16.6 ± 0.4 | 16.5 ± 0.3 |
| $10^{-5}$         |           | 16.5 ± 0.3 | 16.5 ± 0.2 | 16.5 ± 0.4 | 16.6 ± 0.3 |

The table presents the best achievable learning rate for low-rank approximation of a toy matrix. We repeated each experiment five times. SVD accuracy was 14.4 dB. The choice affects the final achievable accuracy—fixed scheduling with a high learning rate performs better than other choices.

D. Modeling Known Decomposition Constraints With the Last Layer Activation

Different constraints on $U$ and/or $V$ lead to different known low-rank decompositions such as nonnegativity on $U$ and $V$ through different activation functions. The table below shows the average PSNR over 10 runs. DeepTensor with ReLU as the final activation function outperformed standard NMF [51], underscoring the efficacy for matrix factorization with positivity constraints.

| Act. func. | softplus | abs. value | ReLU |
|------------|----------|------------|------|
| DeepTensor | 7.4 ±0.03 | 7.3±0.01   | 7.6 ±0.15 |
| NMF [51]   | 7.5      |            |      |
| CIFAR10     | 8.4 ±0.08 | 8.3±0.06   | 8.8 ±0.17 |
| NMF [51]   | 8.2      |            |      |

We performed NMF on MNIST and CIFAR10 datasets by enforcing nonnegativity on $U$, $V$ through different activation functions. The table below shows the average PSNR over 10 runs. DeepTensor with ReLU as the final activation function outperformed standard NMF [51], underscoring the efficacy for matrix factorization with positivity constraints.

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
Fig. 9. Hyperspectral denoising. Low-rank models are often used for representing hyperspectral images. We simulated noisy hyperspectral images by adding Poisson noise with a maximum mean of 100 counts and then denoised using a rank-20 model. DeepTensor has approximately 6 dB increase in performance over SVD and 3 dB over BM3D [55].

are thus without range constraints. Hence we need to impose nonnegativity constraints on the output.

Nonnegativity can be achieved with various activation functions including ReLU, softplus, and element-wise absolute value, and the exact choice affects the achievable accuracy. To gain an empirical insight into the effect of the activation function, we considered NMF on MNIST [49] and CIFAR [50] datasets. In both cases, we used 2048 images for training. We added a rician noise to the images of the form $n = 0.3(z_1 + z_2^2)$ where $z_1$ and $z_2$ are iid Gaussian random variables with zero mean and unit variance. The input PSNR was evaluated to be 4.8 dB. We then performed NMF on the resultant images with various techniques. Comparisons are tabulated in Table II. We employed a combination of $\ell_2$ loss for data fidelity, and $\ell_1$ penalty for the factor matrices output from the DNs. DeepTensor, particularly when combined with the ReLU activation function achieves higher approximation accuracy compared to baseline NMF algorithm [51].

IV. APPLICATIONS

We now showcase the breadth of DeepTensor’s applicability in several real-world applications.

Training Details: Unless otherwise specified, we used over-parameterized networks with skip connections (similar to DIP [21]), and trained with a learning rate of $10^{-3}$. We implemented our training process in Python using the Pytorch framework [45]. All our experiments were run on a Linux machine equipped with a hexa-core Intel Core i7-6850K CPU, 128GB RAM, and three NVIDIA GeForce GTX 1080 Ti. The network architectures were a modified version of the one used by Ulyanov et al. [21]. Specifically, we changed the number of output channels to be equal to the rank for all factor matrices. For tensor decomposition with 1D fibers, we changed the 2D convolutional architecture in [21] to a 1D architecture. We optimized for both inputs and network parameters, which provided faster convergence; however, there was no other significant difference if we did not optimize for the inputs.

A. Linear Inverse Problems in Computer Vision

Low-rank model finds use in numerous applications in computer vision including sensing of light transport matrices [52], hyperspectral imaging [53], video compressive sensing [54], magnetic resonance imaging (MRI), and positron emission tomography (PET). Most inverse problems involve collection of limited data samples and/or highly noisy samples. In both cases, we expect the DeepTensor to be very effective. We showcase three specific examples here.

Hyperspectral Image Denoising: Low-rank models offer a concise representation of hyperspectral images (HSI) and are used in compression, sensing [53], and dimensionality reduction. HSIs involve imaging the scene across several hundreds of spectral bands, resulting in high photon noise (highly non-Gaussian). Typically, a HSI of dimension $N_x \times N_y \times N_\lambda$ is converted to a matrix of dimension $N_x N_y \times N_\lambda$ which is then approximated using a low-rank model. We denoised a $348 \times 327 \times 260$ HSI from Arad and Ben-Shahar [42] by simulating Poisson noise equivalent to a maximum of 100 photons per spatio-spectral voxel, and a readout noise of 2 photons – settings corresponding to a dull overcast outdoor scene. We then performed a rank-20 decomposition with SVD and DeepTensor. We also compared DeepTensor against the BM3D denoising algorithm [55]. We ran the DeepTensor optimization process for a total of 5000 iterations. Fig. 9 shows the comparisons with various approaches. DeepTensor outperforms SVD by 6 dB, and BM3D by 3 dB and produces visually pleasing results.
outliers such as salt and pepper noise. We performed a 3-way PARAFAC decomposition of a subset of faces from the Yale face database (B) [56] consisting of 160, 192 × 168-dimensional images. We then added 30%, and 60% salt and pepper noise for a fair comparison against the Kronecker Decomposition-based approach (KDRSDL) which is the state-of-the-art in tensor decomposition [35]. We found that an $\ell_1$ penalty worked better than $\ell_2$ for DeepTensor, as the noise was spatially sparse. In both cases, we picked a rank of 2000 for PARAFAC decomposition. Results are shown in Fig. 10. Evidently, DeepTensor outperforms KDRSDL, especially in extremely noisy settings.

**Video Compressive Sensing:** DeepTensor can also be used for full-rank decomposition, which is apt for signals such as videos. To test this, we used our framework on recovery of video frames from spatially multiplexed images. We relied on the setup from Hitomi et al. [54] where each pixel was sampled at an arbitrary time frame across multiple frames. We combined 8 frames into one single coded image and then used DeepTensor to solve the linear inverse problem. For comparison, we recovered video by solving the linear inverse problem with a 2D TV over spatial images as a regularizer. We trained DeepTensor for a total of 10,000 epochs. Fig. 11 shows coded image as well as the 8 recovered images for an example video. DeepTensor not only has higher accuracy than TV, but the recovered images look visibly similar to ground truth.

**3D Reconstruction From CT Images:** DeepTensor is well suited for tasks such as recovery of 3D volume from CT scans. To demonstrate the advantages of using a tensor representation, we rely on PARAFAC decomposition. We approximated a 256 × 256 × 56 PET CT scan from Clark et al. [57] with a rank-1000 PARAFAC tensor. Note that although the rank is much larger than any single dimension, the effective number of parameters are only 15% of the number of elements in the 3D volume. We added Poisson and readout noise of maximum of 100 photons per voxel and 2 photons respectively. We then simulated 40 tomographic measurements for each slice. 2D TV results were obtained by a TV penalty on each slice along z-direction. Bacca et al. [41] results were obtained by representing input as a rank-1000 PARAFAC tensor and using an untrained 2D network which output 56 channels. 2D TV + PARAFAC results were obtained with self-supervised learning by representing the volume via rank-1000 PARAFAC decomposition, and a TV penalty on each slice along z-axis. Algorithmic reconstruction results were obtained by solving the linear inverse problem without any other priors. FBPConvNet [58] results were obtained by using the output of algorithmic reconstruction and then denoising with a pre-trained model. Finally, the results by [59] were obtained with a self-supervised learning by using a 3D convolutional neural network. While [59] utilized an MRI image as input for the network, we used uniform random noise. Results are visualized in Fig. 12. DeepTensor achieves better reconstruction accuracy than other approaches in both PSNR and SSIM. We note that pre-trained approaches such as FBPConvNet [58] can outperform DeepTensor if trained with the appropriate data.

**B. Classification via Low-Dimensional Projection**

A robust low-rank approximation also affects downstream tasks such as classification. To test this, we worked with an Eigenfaces [20] example for facial classification. We took 840 images across 28 subjects from the Weizmann dataset [60]. We used 25% of the data for training an 84-dimensional subspace via PCA, Independent Component Analysis (ICA) [61], and running DeepTensor on the sample covariance matrix. To emulate noisy conditions, we added poisson noise with variable amounts of noise. We then converted the images to the 84-dimensional space and trained linear and kernel support vector machine with radial basis function. We cross-validated to choose penalty that maximized classification accuracy for each individual classifier. Finally, we evaluated the resultant linear subspace along with SVM on the test data to compute average accuracy. Fig. 2 visualizes the learned basis vectors with the three approaches, and the average classification accuracy as a function of input data PSNR. We notice that the basis learned from DeepTensor is smoother and better representative of the underlying data. In contrast, PCA and ICA overfit to the noise in data, resulting in a reduction in classification accuracy.

**C. Time-Frequency Decomposition**

As discussed in Section III-D, DeepTensor can be combined with additional constraints such as nonnegativity on $U, V$. We validated the advantages of DeepTensor by performing an NMF on speech data obtained from the GOOGLECOMMANDS dataset [62]. We first computed spectrograms on all speech data with a window size of 1024 samples and a hop size of 32, resulting in a 512 × 512 dimensional time-frequency image. We then added noise as described in Section III-D, resulting in an input SNR of 4.6 dB. The approximation accuracies with DeepTensor, and a baseline NMF [47] NMF algorithm in Table III. DeepTensor performs significantly better than the baseline. This ability of DeepTensor to perform better in high noise settings is of particular significance when speech is recorded in noisy environments.

**D. Choice of Rank**

As with all matrix and tensor decomposition approaches, the rank of the decomposition is an important parameter for
DeepTensor. However, we now demonstrate that DeepTensor is less sensitive to the choice of rank than the SVD in two different settings. In the first experiment, we truncated a hyperspectral image [63] to rank-20 and swept the rank from 1 to 30. In the second experiment, we truncated a subset of the Yale-B dataset [56] using rank-1000 PARAFAC decomposition and then swept the decomposition rank from 10 to 4000. In both cases, we added Poisson ($\lambda_{\text{max}} = 100$) and Gaussian noise ($\sigma = 2$) equivalent to a 25 dB measurement PSNR. Fig. 13 shows plots of PSNR as a function of rank for both the cases. When the rank
We have demonstrated that self-supervised learning is effective for solving low-rank tensor and matrix decomposition. Across the board, we see that DeepTensor is a superior option compared to SVD/PCA when the input SNR is low, the matrix/tensor values are non-Gaussian distributed, or the inverse problem is ill-conditioned such as in the case of PCA with limited samples, or linear inverse problems. Moreover, our separable approximation approach results in faster approximation of 3D tensors compared to DNs with 2D convolutional filters.

Future Directions: Our experiments relied mostly on 1D or 2D convolutional filters, whose inductive biases are better suited for signals such as time series and images. However, the DNs can be chosen specific to the task at hand, such as fully connected or recurrent networks. For image-like tensors, DIP [21] or deep decoder [22] is a well-suited DN architecture as the outputs resemble images. DIP as a prior may not be appropriate for certain types of signals such as sparse images (see Fig. 14). However, such signals can be approximated with other priors that are well-suited for sparse images. Similarly, if the signals resemble noise (potentially outputs of auto-regressive processes), one can choose a different architecture of DN. As an extension, DeepTensor can also be equipped with trained DNs that are better suited for specific signals like audio clips. Such trained extensions of DeepTensor are exciting future directions. DeepTensor can also be used in non-linear representations, such as local low-rank, or even non-local low-rank. Such settings increase the range of problems that can be tackled effectively and are exciting future directions.

Limitations: The current bottleneck of DeepTensor, as opposed to techniques such as SVD, lies in the training computational complexity. While a single pass through a DN is much faster than the training of SVD or NMF, the need to repeatedly iterate between the forward passes and back propagation inherent to gradient based learning slows DeepTensor training. This opens up interesting research directions aimed at discovering simpler networks that are faster to train yet maintain high performance and developing specialized training algorithms that leverage the low-rank decomposition structure.

References

[1] P. H. Schönemann, “A generalized solution of the orthogonal Procrustes problem,” Psychometrika, vol. 31, no. 1, pp. 1–10, 1966.
[2] S. Wold, K. Esbensen, and P. Geladi, “Principal component analysis,” Chemometrics Intell. Lab. Syst., vol. 2, no. 1-3, pp. 37–52, 1987.
[3] P. Jain, P. Netrapalli, and S. Sanghavi, “Low-rank matrix completion using alternating minimization,” in Proc. ACM Symp. Theory Comput., 2013, pp. 665–674.
[4] E. J. Candès and B. Recht, “Exact low-rank matrix completion via convex optimization,” in Proc. Allerton Conf. Commun. Control Comput., 2008, pp. 806–812.
[5] P. Drineas, A. Frieze, R. Kannan, S. Vempala, and V. Vinyay, “Clustering large graphs via the singular value decomposition,” Mach. Learn., vol. 56, no. 1, pp. 9–33, 2004.
[6] A. I. Zecevic and D. D. Siljak, “Global low-rank enhancement of decentralized control for large-scale systems,” IEEE Trans. Autom. Control, vol. 50, no. 5, pp. 740–744, May 2005.
[7] P. Benner and T. Breiten, “Low-rank methods for a class of generalized Lyapunov equations and related issues,” Numerische Mathematik, vol. 124, no. 3, pp. 441–470, 2013.
[8] X. Liang, X. Ren, Z. Zhang, and Y. Ma, “Repairing sparse low-rank texture,” in Proc. Eur. Conf. Comput. Vis., 2012, pp. 482–495.
[9] H. Ji, C. Liu, Z. Shen, and Y. Xu, “Robust video denoising using low-rank matrix completion,” in Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit., 2010, pp. 1791–1798.
[10] H. Zhang, W. He, L. Zhang, H. Shen, and Q. Yuan, “Hyperspectral image restoration using low-rank matrix recovery,” IEEE Trans. Geosci. Remote Sens., vol. 52, no. 8, pp. 4729–4743, Aug. 2014.
[11] C. Eckart and G. Young, “The approximation of one matrix by another of lower rank,” Psychometrika, vol. 1, no. 3, pp. 211–218, 1936.
[12] H. Xu, C. Caramanis, and S. Sanghavi, “Robust PCA via outlier pursuit,” 2010, arXiv:1010.4237.
[13] D. Goldfarb and Z. Qin, “Robust low-rank tensor recovery: Models and algorithms,” SIAM J. Matrix Anal. Appl., vol. 35, no. 1, pp. 225–253, 2014.
[14] A. E. Waters, A. C. Sankaranarayanan, and R. G. Baraniuk, “SpaRCS: Recovering low-rank and sparse matrices from compressive measurements,” in Proc. Int. Conf. Neural Inf. Process. Syst., 2011, pp. 1089–1097.
[15] T.-Y. Ji, T.-Z. Huang, X.-L. Zhao, T.-H. Ma, and G. Liu, “Tensor completion using total variation and low-rank matrix factorization,” Inf. Sci., vol. 226, pp. 243–257, 2016.
[16] W. He, H. Zhang, L. Zhang, and H. Shen, “Total-variation-regularized low-rank matrix factorization for hyperspectral image restoration,” IEEE Trans. Geosci. Remote Sens., vol. 54, no. 1, pp. 178–188, Jan. 2016.
[17] Y. Wang, J. Peng, Q. Zhao, Y. Leung, X.-L. Zhao, and D. Meng, “Hyperspectral image restoration via total variation regularized low-rank tensor decomposition,” IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens., vol. 11, no. 4, pp. 1227–1243, Apr. 2018.

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
A. Paszke et al., “PyTorch: An imperative style, high-performance deep learning library,” in Proc. Int. Conf. Learn. Representations, 2019, pp. 8024–8035.

G. Huang, Y. Li, G. Pleiss, Z. Liu, J. E. Hopcroft, and K. Q. Weinberger, “Snapshot ensembles: Train 1, get M for free,” 2017, arXiv: 1704.00109.

C. H. Ding, T. Li, and M. I. Jordan, “Convex and semi-nonnegative matrix factorizations,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 32, no. 1, pp. 45–55, Jan. 2010.

C. Ding, X. He, and H. D. Simon, “On the equivalence of nonnegative matrix factorization and spectral clustering,” in Proc. SIAM Int. Conf. Data Mining, 2005, pp. 606–610.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” Proc. IEEE, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

A. Krizhevsky et al., “Learning multiple layers of features from tiny images,” 2009. [Online]. Available: https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf

A. Cichocki and A.-H. Phan, “Fast local algorithms for large scale nonnegative matrix and tensor factorizations,” IEICE Trans. Fundam. Electron. Commun. Comput. Sci., vol. 92, no. 3, pp. 708–721, 2009.

M. O’Toole and K. N. Kutulakos, “Optical computing for fast light transport analysis,” ACM Trans. Graph., vol. 29, no. 6, pp. 164:1–164:12, 2010.

V. Saragadam and A. Sankaranarayanan, “KRIS—Krylov subspace-based optical computing of hyperspectral images,” ACM Trans. Graph., vol. 38, no. 5, pp. 148:1–148:14, 2019.

Y. Hinton, J. Gu, M. Ghiassian, T. Mitsuhashi, and S. K. Nayar, “Video from a single coded exposure photograph using a learned over-complete dictionary,” in Proc. IEEE Int. Conf. Comput. Vis., 2011, pp. 287–294.

K. Babov, A. Foi, V. Katkovnik, and K. Egiazarian, “Image denoising by sparse 3D transform-domain collaborative filtering,” IEEE Trans. Image Process., vol. 16, no. 6, pp. 2080–2095, Aug. 2007.

P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman, “Eigenfaces vs. fisherfaces: Recognition using class specific linear projection,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 19, no. 7, pp. 711–720, Jul. 1997.

K. Clark et al., “The cancer imaging archive (TCIA): Maintaining and operating a public information repository,” J. Digit. Imag., vol. 26, no. 6, pp. 1045–1057, 2013.

K. H. Jin, M. T. McCann, E. Froustey, and M. Unser, “Deep convolutional neural network for inverse problems in imaging,” IEEE Trans. Image Process., vol. 26, no. 9, pp. 4509–4522, Sep. 2017.

K. Gong, C. Catana, J. Qi, and Q. Li, “PET image reconstruction using deep image prior,” IEEE Trans. Med. Imag., vol. 38, no. 7, pp. 1655–1665, Jul. 2019.

W. I. of Sciences, “Computer vision databases,” 2004. [Online]. Available: http://www.wisdom.weizmann.ac.il/˜vision/FaceBase/

P. Comon, “Independent component analysis, a new concept?,” Signal Process., vol. 36, no. 3, pp. 287–314, 1994.

P. Warden, “Speech commands: A dataset for limited-vocabulary speech recognition,” 2018, arXiv: 1804.03209.

J. Choi, D. S. Jeon, G. Nam, D. Gutierrez, and M. H. Kim, “High-quality hyperspectral reconstruction using a spectral prior,” ACM Trans. Graph., vol. 36, no. 6, pp. 218:1–218:13, 2017.
Randall Balestriero (Member, IEEE) received the bachelor’s degree in applied mathematics and economics from the University of Toulon, Toulon, France, in 2014, the master’s degree from Pierre and Marie Curie University, Paris, France, and the École Normale Supérieure, Paris, in 2016, and the PhD degree in deep learning from Rice University, Houston, TX, USA, and is now an assistant professor of computer science with Brown University. Prior to joining Brown, he was a postdoctoral researcher with Meta AI in Prof. Yann LeCun’s Group. He did research in signal processing and bioacoustics using the scattering networks and large-scale data sets with the University of Toulon. He did research in signal processing, deep learning, and real-time finite-element PDE approximation at the Pierre and Marie Curie University and the École Normale Supérieure. His most active research is now in deep learning theory, generative modeling, and self-supervised learning. His DeepTensor rank is 53.

Ashok Veeraraghavan (Fellow, IEEE) received the bachelor’s degree in electrical engineering from the Indian Institute of Technology, Madras, Chennai, India, in 2002, and the MS and PhD degrees from the Department of Electrical and Computer Engineering, University of Maryland, College Park, MD, USA, in 2004 and 2008, respectively. He is currently a professor and the chair of the Electrical and Computer Engineering, with Rice University, Houston, TX, USA. Before joining Rice University, he spent three years as a research scientist with Mitsubishi Electric Research Labs, Cambridge, MA, USA. His research interests are broadly in the areas of computational imaging, computer vision, machine learning, and robotics. His thesis received the Doctoral Dissertation Award from the Department of Electrical and Computer Engineering, University of Maryland. He is the recipient of the National Science Foundation CAREER Award in 2017. At Rice University, he directs the Computational Imaging and Vision Lab. His DeepTensor rank is 61.

Richard G. Baraniuk (Fellow, IEEE) received the BS degree from the University of Manitoba, Canada, the MS degree from the University of Wisconsin-Madison, and the PhD degree from the University of Illinois at Urbana-Champaign, all in electrical engineering. He is currently the C. Sidney Burrus professor of electrical and computer engineering with Rice University and the founding director of OpenStax (openstax.org). His research interests lie in new theory, algorithms, and hardware for sensing, signal processing, and machine learning. He is a member of the US National Academy of Engineering and American Academy of Arts and Sciences and a fellow of the National Academy of Inventors, American Association for the Advancement of Science. He has received the DOD Vannevar Bush Faculty Fellow Award, the Harold W. McGraw, Jr. Prize in Education, the IEEE James H. Mulligan, Jr. Education Medal, and the IEEE Signal Processing Society Society, Technical Achievement, Education, Best Paper, Best Magazine Paper, and Best Column Awards. He holds 45 US and 8 foreign patents. His Erdos Number is 3, and his DeepTensor rank is 67.