Tensor Super-Resolution with Generative Adversarial Nets: A Large Image Generation Approach

Zihan Ding, Xiao-Yang Liu, Miao Yin and Linghe Kong

August 12, 2019

Tensor Super-Resolution with Generative Adversarial Nets: A Large Image Generation Approach

Zihan Ding[1], Xiao-Yang Liu[2], Miao Yin[3], and Linghe Kong[4]

[1] Imperial College London, the Unitied Kindom
zd2418@ic.ac.uk
[2] Columbia University, U.S.A.
xl2427@columbia.edu
[3] University of Electronic Science
and Technology of China, China.
yinmiaothink@gmail.com
[4] Shanghai Jiao Tong University, China
linghe.kong@sjtu.edu.cn

Abstract

Deep generative models have been successfully applied to many applications. However, existing methods experience limitations when generating large images (the literature usually generates small images, e.g., 32×32 or 128×128). In this paper, we propose a novel scheme using tensor super-resolution with adversarial generative nets (TSRGAN), to generate large high-quality images by exploring tensor structures. Essentially, the super resolution process of TSRGAN is based on tensor representation. First, we impose tensor structures for concise image representation, which is superior in capturing the pixel proximity information and the spatial patterns of elementary objects in images, over the vectorization preprocess in existing works. Secondly, we propose TSRGAN that integrates deep convolutional generative adversarial networks and tensor super-resolution in a cascading manner, to generate high-quality images from random distributions. More specifically, we design a tensor super-resolution process that consists of tensor dictionary learning and tensor coefficients learning. Finally, on three datasets, the proposed TSRGAN generates images with more realistic textures, compared with state-of-the-art adversarial autoencoders and super-resolution methods. The size of the generated images is increased by over 8.5 times, namely 374×374 in PASCAL2.

Keywords: GAN, generative model, super-resolution, tensor sparse coding, tensor representation.

Contents

1 Introduction

With the great successes of deep learning, deep generative models have been investigated widely. However, existing generative adversarial nets (GAN) experience limitations when generating large images. With the growing scale of images, conventional GAN is hard to produce high-quality natural images because it is difficult for the generator and the discriminator to achieve the optimality simultaneously. When processing high-dimensional images, the computational complexity and the training time increase significantly. The challenge is that the image has too many pixels and it is hard for a single generator G to learn the empirical distribution. Therefore, the traditional GAN [7] does not scale well for the generation of large images.

The variations of GAN such as deep convolutional GAN (DCGAN) [20], super-resolution GAN (SRGAN) [16], Laplacian Pyramid GAN (LAPGAN) [2] and StackGAN [27] are promising candidates for generative models in unsupervised learning. Moreover, other types of generative models include adversarial autoencoders (AAE) [17], combining the GANs and variational autoencoders (VAE) [12], etc [8]. However, even with DCGAN, the bottleneck of GAN exposes easily for large images, since increasing the complexity of the generator does not necessarily improve the image quality. Moreover, StackGAN [27] uses a two-stage GAN to generate images of size 256×256, which are relatively large images for state-of-the-art generative models. Another work is progressive growing of GANs [11], which applies an approach of growing network size for both the generator and the discriminator to generate high-resolution images. However, this method requires large amounts of computation during the progressively increasing of network size. Our proposed method only needs much less computation resources to obtain high-resolution images of large size.

For image super-resolution, approaches including deep neural networks [3][4][23], sparsity-based [26], local regression [9] are proposed in recent years. The super-resolution via sparse representation (ScSR) [26] applies the sparse representation with an over-complete dictionary in super-resolution. The enforced similarity of sparse representations between the low-resolution and high-resolution images makes it possible to achieve super-resolution. Instead of applying popular deep neural networks for super-resolution, our work is based on sparse representation,and changes the representation space from pixel space to tensor space, to exploit the benefits of tensor representation for large images generation.

To solve the challenge of efficient large image generation, we apply a new perspective to change the representation space through combining the GAN and tensor super-resolution process. Traditional GAN-based methods operates in pixel space to generate images while tensor-based methods work in tensor space. Tensor representation [13] and its derivative methods such as tensor sparse coding [6] and tensor super-resolution have a better representation of images, especially for large images. Multi-dimensional tensor sparse coding uses t-linear combination to obtain a more concise and smaller dictionary for representing the images, and the corresponding coefficients have richer physical explanations than the traditional methods.

In this paper, we present a novel generative model called TSRGAN as shown in Fig. 1, cascading a DCGAN and tensor-based super-resolution to generate large high-quality images

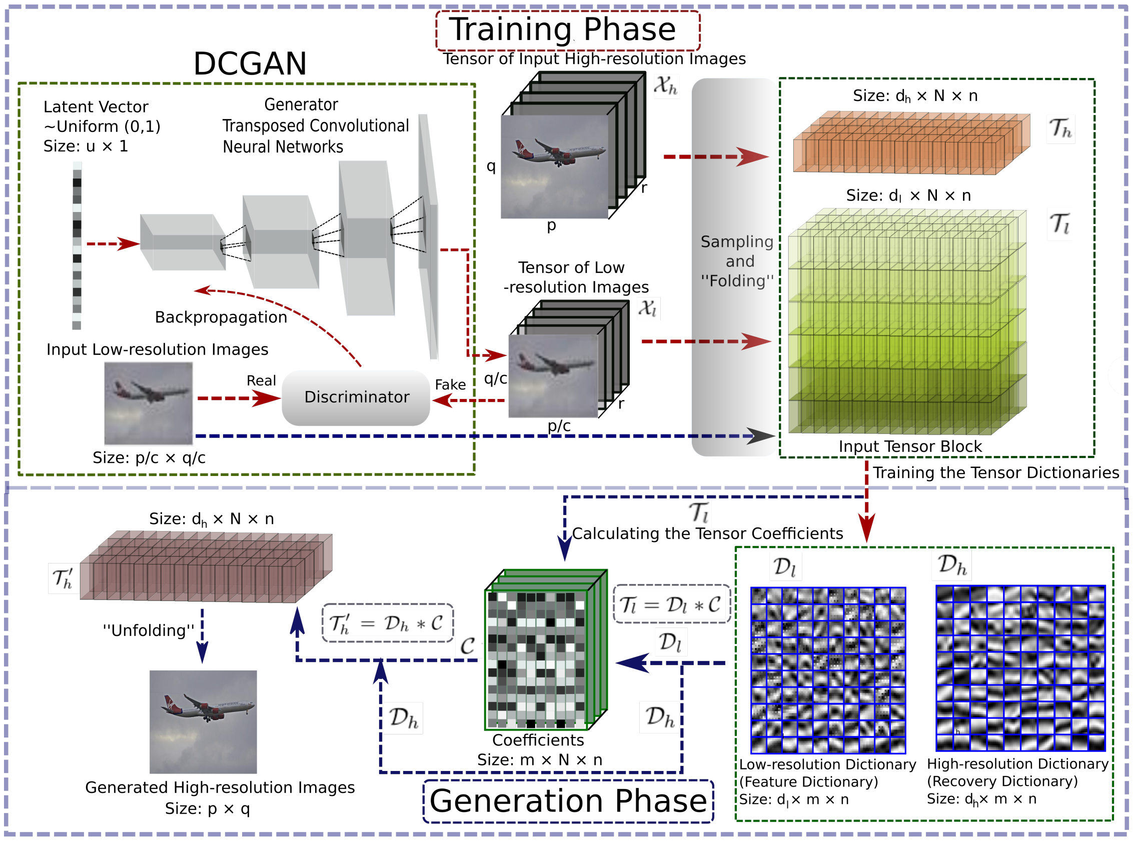

Figure 1: The architecture of TSRGAN. The latent vectors are sampled from random distributions. During the training phase, the DCGAN are trained with the input low-resolution images, to generate low-resolution image tensors from latent vectors. Through a sampling and "folding" process, the high-resolution and low-resolution image tensors are transformed into tensor blocks, $\mathcal{T}_h$ and $\mathcal{T}_l$, respectively. The feature dictionary (low-resolution) $\mathcal{D}_l$ and recovery dictionary (high-resolution) $\mathcal{D}_h$ are trained with these input tensor blocks. In the generation phase, low-resolution tensor images are generated with DCGAN from the latent vectors. The tensor coefficients $\mathcal{C}$ are obtained using $\mathcal{T}_l = \mathcal{D}_l * \mathcal{C}$, where $\mathcal{D}_l$ is the low-resolution tensor feature dictionary derived from the training phase. High-resolution tensor images can be obtained via $\mathcal{T}'_h = \mathcal{D}_h * \mathcal{C}$, where $\mathcal{D}_h$ is the trained high-resolution tensor recovery dictionary. The final 2D images X' are transformed from the high-resolution tensor images $\mathcal{T}'_h$. (Note that during the training phase $\mathcal{T}_l$ is derived from input low-resolution images while for the generation phase it is from images generated with DCGAN.)

(e.g. 374×374). The contribution of the proposed TSRGAN has threefold: (i) We apply tensor representation and tensor sparse coding for images representation in generative models. This is testified to have advantages of more concise and efficient representation of images with less loss on spatial patterns. (ii) We incorporate the tensor representation into the super-resolution process, which is called tensor super-resolution. The tensor super-resolution is cascaded after a DCGAN with transposed convolutional layers, which generates low-resolution images directly from random distributions. (iii) The DCGAN and tensor dictionaries in tensor super-resolution are both pretrained with a large number of high-resolution and low-resolution images. The size of dictionaries is smaller with tensor representation than traditional methods, which accelerates the dictionary learning process in tensor super-resolution. The generation performance of TSRGAN surpasses traditional generative models including adversarial autoencoders [17] in inception score [21] on test datasets, especially for large images. Our codes will be available online.

2 Notations and Preliminaries

We apply the tensor representation and tensor sparse coding in our proposed TSRGAN scheme.

2.1 Tensor Product

We use boldface capital letters to denote matrices, e.g., $\mathbf{A}$, and calligraphic letters to denote tensors, e.g. $\mathcal{T}$. An order-3 tensor is denoted as $\mathcal{T} \in R^{n_1 \times n_2 \times n_3}$. The expansion of $\mathcal{T}$ along the third dimension is represented as $\underline{\mathcal{T}} = [\mathcal{T}^{(1)}; \mathcal{T}^{(2)}; ...\mathcal{T}^{(k)}; ...\mathcal{T}^{(n_3)}] \in R^{n_1 n_2 \times n_3}$, where $\mathcal{T}^{(k)}$ denotes the k-th frontal slice, for $k \in [n_3]$ with $[n]$ denoting the set $\{1, 2, ..., n\}$. The circular matrix representation of tensor $\mathcal{T}$ is defined as

$$\underline{\mathcal{T}}^c = \begin{bmatrix} \mathcal{T}^{(1)} & \mathcal{T}^{(n_3)} & \dots & \mathcal{T}^{(2)} \\ \mathcal{T}^{(2)} & \mathcal{T}^{(1)} & \dots & \dots \\ \dots & \dots & \dots & \mathcal{T}^{(n_3)} \\ \mathcal{T}^{(n_3)} & \mathcal{T}^{(n_3-1)} & \dots & \mathcal{T}^{(1)} \end{bmatrix}. \tag{1}$$

The tensor product [10] of two tensors $\mathcal{A} \in R^{n_1 \times n_2 \times n_3}$ and $\mathcal{B} \in R^{n_2 \times n_4 \times n_3}$ is defined as

$$\mathcal{T} = \mathcal{A} * \mathcal{B} \in R^{n_1 \times n_4 \times n_3}, \tag{2}$$

where $\mathcal{T}(i, j, :) = \sum_{s=1}^{n_2} \mathcal{A}(i, s, :) \bullet \mathcal{B}(s, j, :)$ for $i \in [n_1]$ and $j \in [n_4]$, and $\bullet$ denotes the circular convolution operation. In addition, the tensor product has an equivalent matrix-product form:

$$\underline{\mathcal{T}} = \underline{\mathcal{A}}^c \underline{\mathcal{B}}. \tag{3}$$

2.2 Tensor Sparse Coding for Images

Considering r input images X of size $p \times q$, we first sample the image tensor $\mathcal{X} \in R^{p \times q \times r}$ using tensor cubes and reshape it to be the input tensor block $\mathcal{T} \in R^{d \times N \times n}$ (detailed relationships of d, N, n with p, q, r and the tensor cubes are shown in Section 4). $\mathcal{T}$ can be approximated with an overcomplete tensor dictionary $\mathcal{D} \in R^{d \times m \times n}$, $m > d$ as follows [6]:

$$\mathcal{T} = \mathcal{D} * \mathcal{C} = \mathcal{D}_1 * \mathcal{C}_1 + ... + \mathcal{D}_m * \mathcal{C}_m, \tag{4}$$

where $\mathcal{C} \in R^{m \times N \times n}$ is the tensor coefficient with slice $\mathcal{C}_j = \mathcal{C}(j, :, :)$.

One of the proposed schemes for tensor sparse coding is based on the ℓ_1-norm of the coefficient. The sparse coding problem in tensor representation is as follows:

$$\begin{aligned} \min_{\mathcal{D}, \mathcal{C}} \quad & \frac{1}{2} ||\mathcal{T} - \mathcal{D} * \mathcal{C}||_F^2 + \lambda ||\mathcal{C}||_1 \\ \text{s.t.} \quad & ||\mathcal{D}(:, j, :)||_F^2 1, j \in [m], \end{aligned} \tag{5}$$

where the size of the dictionary $\mathcal{D}$ is $d \times m \times n$, $m > d$. However, traditional sparse coding requires the size of the dictionary to be $(d \times n) \times m, m > d \times n$, which significantly increases with the increase in dimensionality, as shown in [6]. A smaller dictionary is easier to learn in tensor sparse coding, which is a more efficient way to encode images compared with traditional sparse coding methods. Our proposed TSRGAN scheme are based on tensor representation, to reduce the dictionary size and improve the dictionary learning efficiency, which is the key advantage of the proposed method over other more advanced approaches like BigGANs [1], Progressive GAN [11], SNGAN [18], etc. Moreover, our proposed framework of cascading tensor super-resolution with DCGAN is actually orthogonal to those advanced GAN methods, which means the DCGAN can actually be replaced by more advanced GANs to leverage the advantages of tensor sparse coding as well.

3 TSRGAN Scheme

As a generative model with tensor super-resolution, the TSRGAN scheme could be divided into two phases: the training phase and the generation phase. First of all, two-dimensional (2D) images are transformed into the tensor space as a preprocess. In the generation phase: using pretrained DCGAN to generate low-resolution image tensors from random distributions, we apply tensor super-resolution for transforming low-resolution image tensors to high-resolution image tensors. High-resolution 2D images can be derived from the obtained high-resolution image tensors. The tensor dictionaries we used in the tensor super-resolution process and the DCGAN are both pretrained with large numbers of high-resolution and low-resolution image tensors in the training phase. The training phase is ahead of the generation phase in implementations. Our proposed approach combines DCGAN with tensor-based super-resolution, to directly generate high-resolution images.

3.1 Tensor Representation in TSRGAN

An important motivation of applying the tensor representation for large-image generation is the advantage of smaller size of both the dictionary and the sparse coefficients, as shown in Sec. 2.2: For instance, a tensor of size $\mathcal{T} \in R^{d \times N \times n}$ with tensor sparse coding has both the dictionary size and size of sparse coefficients at least n times smaller than the traditional sparse coding methods. Moreover, there are other advantages including the invariance of shifting [6] and more concise representation of images, which make the proposed TSRGAN scheme to have larger potential to be effective in practice. We make the assumption [22] that the inner patterns of images can be at least approximately sparsely represented with a learned dictionary. For tensor dictionary representation, $\mathcal{T} = \mathcal{D} * \mathcal{C}$, where $\mathcal{T} \in R^{d \times N \times n}, \mathcal{D} \in R^{d \times m \times n}, \mathcal{C} \in R^{m \times N \times n}$. Therefore, tensor representation of images is necessary, which acts as the main representation of images in our workflows.

3.2 Data Preprocess: "Folding" and "Unfolding"

We obtain the tensor input block $\mathcal{T}$ with original images $X \in R^{p \times q}$ with the "folding" process as follows. We first concatenate r images shifted from the same original image $X \in R^{p \times q}$ for high-resolution or $X \in R^{\frac{p}{c} \times \frac{q}{c}}$ for low-resolution (first upsampling it to be $X \in R^{p \times q}$ in the generation phase) with different pixels to obtain the image representation tensor $\mathcal{X} \in R^{p \times q \times r}$, as shown in Fig. 2. Then we sample N_0 image tensors $\mathcal{T}$ in all dimensions with the tensor block of size $a \times a \times a$ to obtain N sample blocks, where $N = N_0 \times (p-a+1) \times (q-a+1) \times (r-a+1)$. Therefore,

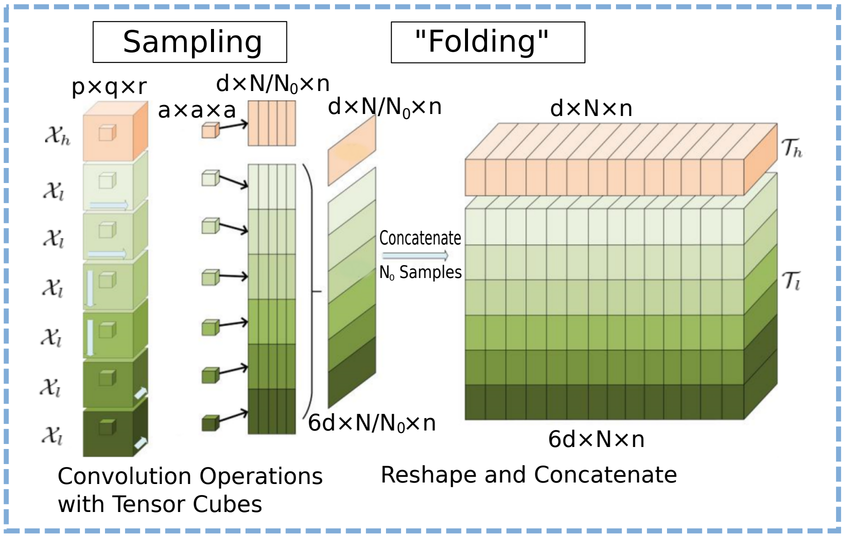

Figure 2: Preparation of the tensor blocks for tensor dictionary learning, including sampling and "folding". With the concatenated high-resolution image tensors $\mathcal{X}_h$ and low-resolution image tensors $\mathcal{X}_l$ (upsampled to have same size with $\mathcal{X}_h$) from the same original image, we sample (through a convolution operation) $\mathcal{X}_l$ in all dimensions and $\mathcal{X}_h$ in one dimension with the tensor cubes of size $a \times a \times a$, to obtain N sample blocks and reshape them. With a batch of original images, we could obtain the tensor blocks $\mathcal{T}_l \in R^{d_l \times N \times n}$ and $\mathcal{T}_h \in R^{d_h \times N \times n}$, where $d_l = 6 \times d, d_h = d$.

the size of image representation tensor is $(a \times a \times a) \times (p-a+1) \times (q-a+1) \times (r-a+1)$. The tensor is reshaped to be input tensor blocks $\mathcal{T} \in R^{d \times N \times n}$, where $d = a \times a, n = a$. For training phase, X are 2D images from the training set; and for generation phase, X are generated with DCGAN from random distributions. Note that the tensor dictionary $\mathcal{D} \in R^{d \times m \times n}$ is independent of the number of samples N.

The inverse process of the above "folding" process is called the "unfolding" process, which is used for recovering the high-resolution 2D images from the obtained high-resolution tensor output blocks. The "unfolding" is just a trivial combination of inversing each step in "folding".

3.3 The Training Phase: DCGAN Training and Tensor Dictionary Learning

In our model, we first downsample the original images $X \in R^{p \times q}$ in the training set to high-resolution images $X_h \in R^{p \times q}$ and low-resolution images $X_l \in R^{\frac{p}{c} \times \frac{q}{c}}$ at the downsampling rate c, and we further transform them into tensor representation $\mathcal{X}_l, \mathcal{X}_h$. Then we train DCGAN with X_l to generate low-resolution tensor images $\mathcal{T}_G \in R^{\frac{p}{c} \times \frac{q}{c} \times r}$ from random distributions $\mathbf{r} \sim$Uniform$(0,1)$. The reconstruction loss and adversarial loss are formulated as a minimax game:

$$\begin{aligned}\min_G \max_D L(G,D) =& E_{\mathbf{r} \sim U}\left[\log(1 - D(G(\mathbf{r})))\right] \\ &+ E\left[\log D(X_l)\right], \end{aligned} \tag{6}$$

where G, D denote generator and discriminator of DCGAN, and $\mathbf{r}, U$ denote the latent vector and uniform distributions. The images in tensor representation $\mathcal{X}_l$ and $\mathcal{X}_h$ are further transformed to be input tensor blocks $\mathcal{T}_l$ and $\mathcal{T}_h$ (as shown in the data preprocess of Section 3.2) for training the dictionaries $\mathcal{D}_l$ and $\mathcal{D}_h$ in tensor super-resolution. We have tensor product relationships in tensor sparse coding: $\mathcal{T}_h = \mathcal{D}_h * \mathcal{C}_h$ and $\mathcal{T}_l = \mathcal{D}_l * \mathcal{C}_l$, where $\mathcal{C}_h, \mathcal{C}_l$ denotes tensor sparse coefficients for high-resolution images and low-resolution images respectively. Note that, in tensor super-resolution, it is reasonable to set $\mathcal{C}_h = \mathcal{C}_l$ and denote it with $\mathcal{C}$ [26]. This treatment for the coefficients in super-resolution is derived from the intuition that images can be represented well with sparse codes and appropriate over-complete dictionaries, for both low- and high-resolution images.

3.4 Details about Tensor Super Resolution

The goal for tensor super-resolution is to transform low-resolution images X_l into high-resolution images X_h through the tensor spares coding approach. For an input tensor $\mathcal{T} \in R^{d\times N\times n}$, tensor dictionary learning is similar to (the only difference is the dimensions) the tensor sparse coding in Section 3.2, where $\mathcal{D} \in R^{d\times m\times n}$ is the tensor dictionary, and its slice $\mathcal{D}(:, j, :)$ is a basis, $\mathcal{C} \in R^{m\times N\times n}$ is the tensor sparse coefficient. The first and second term uses the Frobenius norm and ℓ_1-norm in Equ. (5), respectively.

If taking the sparse coding process of different resolution images as similar patterns with respect to different bases, we could consider that high-resolution and low-resolution tensor images from the same origins have sparse and approximate tensor coefficients $\mathcal{C}$. Therefore the constraints of two dictionaries could be combined as follows:

$$\mathcal{D} = \arg\min_{\mathcal{D},\mathcal{C}} ||\mathcal{T} - \mathcal{D} * \mathcal{C}||_F^2 + \lambda||\mathcal{C}||_1, \tag{7}$$

where

$$\mathcal{T} = \begin{bmatrix} \frac{1}{\sqrt{N_h}}\mathcal{T}_h \\ \frac{1}{\sqrt{N_l}}\mathcal{T}_l \end{bmatrix}, \mathcal{D} = \begin{bmatrix} \frac{1}{\sqrt{N_h}}\mathcal{D}_h \\ \frac{1}{\sqrt{N_l}}\mathcal{D}_l \end{bmatrix}, \lambda = \frac{\lambda_h}{N_h} + \frac{\lambda_l}{N_l}, \tag{8}$$

where $\mathcal{T}_h, \mathcal{T}_l$ represent input tensor blocks of high-resolution and low-resolution images and N_h, N_l denote the first dimensional size of $\mathcal{T}_h, \mathcal{T}_l$ respectively. We then apply the Lagrange dual method and the iterative shrinkage threshold algorithm based on tensor-product to solve the tensor dictionaries and tensor sparse coefficients. The minimization problem can be rewritten as:

$$\min_{\mathcal{C}} f(\mathcal{C}) + \lambda g(\mathcal{C}) \tag{9}$$

where $f(\mathcal{C})$ stands for $\frac{1}{2}||\mathcal{T} - \mathcal{D} * \mathcal{C}||_F^2$ and $g(\mathcal{C})$ stands for $||\mathcal{C}||_1$ (coefficient $\frac{1}{2}$ can be absorbed in λ). At the $(s+1)$-th iteration,

$$\begin{aligned} \mathcal{C}_{s+1} = \arg\min_{\mathcal{C}} & f(\mathcal{C}_s) + \langle \nabla f(\mathcal{C}_s), \mathcal{C} - \mathcal{C}_s \rangle \\ & + \frac{L_{s+1}}{2}||\mathcal{C} - \mathcal{C}_s||_F^2 + \lambda g(\mathcal{C}), \end{aligned} \tag{10}$$

where L_{s+1} is a Lipschitz constant. Therefore,

$$\begin{aligned} \mathcal{C}_{s+1} = & \arg\min_{\mathcal{C}} \frac{1}{2}||\mathcal{C} - (\mathcal{C}_s - \frac{1}{L_{s+1}}\nabla f(\mathcal{C}_s))||_F^2 \\ & + \frac{\lambda}{L_{s+1}}||\mathcal{C}||_1, \end{aligned} \tag{11}$$

We can obtain the Lipschitz constant that $L = \sum_{b=1}^{n} ||\tilde{\mathcal{D}}^{(b)H}\tilde{\mathcal{D}}^{(b)}||_F^2$, $\tilde{\mathcal{D}}^{(b)}$ is the discrete fourier transformation (DFT) of the third-dimension slice $\mathcal{D}^{(b)}(:, j), b \in [n]$, and subscript H implies that it is a conjugate transpose. In the implemented algorithm for the training process of $\mathcal{C}$, we use $\mathbf{Prox}_{\beta/L}$ to solve the above equations, which is the proximal operator [19]. We therefore obtain the tensor sparse coding coefficients $\mathcal{C}$ through iteratively solving Equ. (11).

For learning the dictionary $\mathcal{D}$ with fixed $\mathcal{C}$, the optimization problem w.r.t each of the n slices of $\mathcal{D}$ becomes

$$\begin{aligned} \min_{\mathcal{D}^{(b)} \in R^{d\times m}, b\in[n]} & ||\mathcal{T}^{(b)} - \mathcal{D}^{(b)} * \mathcal{C}^{(b)}||_F^2 \\ \text{s.t. } & ||\tilde{\mathcal{D}}^{(b)}(:, j)||_F^2 1, j \in [m], b \in [n]. \end{aligned} \tag{12}$$

Transform the above equations into the frequency domain,

$$\begin{aligned} \min_{\mathcal{D}^{(b)} \in R^{d\times m}, b\in[n]} & ||\tilde{\mathcal{T}}^{(b)} - \tilde{\mathcal{D}}^{(b)} \cdot \tilde{\mathcal{C}}^{(b)}||_F^2 \\ \text{s.t. } & ||\tilde{\mathcal{D}}^{(b)}(:, j)||_F^2 1, j \in [m], b \in [n]. \end{aligned} \tag{13}$$

Therefore, with the Langrange dual, we obtain

$$\begin{aligned} \mathcal{L}(\tilde{\mathcal{D}}, \Omega) = & \sum_{b=1}^{n} ||\tilde{\mathcal{T}}^{(b)} - \tilde{\mathcal{D}}^{(b)} \cdot \tilde{\mathcal{C}}^{(b)}||_F^2 \\ & + \sum_{j=1}^{m} \omega_j (\sum_{b=1}^{n} ||\tilde{\mathcal{D}}^{(b)}(:, j)||_F^2 - n). \end{aligned} \tag{14}$$

Thus, the optimal formulation of $\widehat{\mathcal{D}}^{(b)}$ satisfies:

$$\tilde{\mathcal{D}}^{(b)} = (\tilde{\mathcal{T}}^{(b)} \tilde{\mathcal{C}}^{(b)^H})(\tilde{\mathcal{C}}^{(b)} \tilde{\mathcal{C}}^{(b)^H} + \Omega)^{-1}. \tag{15}$$

Therefore,

$$\mathcal{L}(\Omega) = -\sum_{b=1}^{n} \mathbf{Tr}(\tilde{\mathcal{C}}^{(b)^H} \tilde{\mathcal{T}}^{(b)} \tilde{\mathcal{D}}^{(b)^H}) - n \sum_{j=1}^{m} \omega_j. \tag{16}$$

Equ. (16) can be solved with Newton's method. Substitute the derived Lagrange multiplier $\Omega = \{\omega_j\}, j \in [m]$ in Equ. (15). Thus, we can derive the dictionary $\mathcal{D}$ through inverse fourier transformation of $\tilde{\mathcal{D}}^{(b)}$.

3.5 The Generation Phase

In the generation phase, we first generate low-resolution images $T_G \in R^{p\times q}$ with the trained DCGAN model directly from latent vectors $\mathbf{r}$ in random distribution, and concatenate them to make image tensors $\mathcal{T}_G \in R^{p\times q\times r}$. Then, we set $\mathcal{T}_l = \mathcal{T}_G$ to derive the tensor sparse coefficients $\mathcal{C}$ with the relationship $\mathcal{T}_l = \mathcal{D}_l * \mathcal{C}$ and trained dictionary $\mathcal{D}_l$ with $\mathcal{C}$. Finally we use $\mathcal{T}_h' = \mathcal{D}_h * \mathcal{C}$ to generate high-resolution output tensor block $\mathcal{T}_h'$ with derived dictionary $\mathcal{D}_h$. The output high-resolution 2D images $X' \in R^{p\times q}$ are obtained through "unfolding" the generated high-resolution tensor block $\mathcal{T}_h'$.

Algorithm 1 TSRGAN - Training Phase

```
1: Input: original images $X \in R^{p\times q}$, training iteration $T, S$, sparsity parameter $\lambda$;
2: Initialize the parameters, transform the data into tensor representation space and train the DCGAN.
3: for $k = 1$ to $T$ do
4:     # Solve tensor coefficient $\mathcal{C}$.
5:     for $s = 1$ to $S$ do
6:         Set $L_s = \eta_s(\sum_{b=1}^{n} \|\widehat{\mathcal{D}}^{(b)^H} \widehat{\mathcal{D}}^{(b)}\|_F)$;
7:         Compute $\nabla f(\mathcal{C}_s)$ ;
8:         Compute $\mathcal{C}_s$ via $\mathbf{Prox}_{\beta/L_s}(C_s - \frac{1}{L_s}\nabla f(\mathcal{C}_s))$;
9:         $t_{s+1} = \frac{1+\sqrt{1+4t_s^2}}{2}$;
10:        $\mathcal{C}_{s+1} = \mathcal{C}_s + \frac{t_s-1}{t_{s+1}}(\mathcal{C}_s - \mathcal{C}_{s-1})$;
11:    end for
12:    # Solve tensor dictionaries $\mathcal{D}_h, \mathcal{D}_l$.
13:    Take Fourier transformation for $\mathcal{T} = [1/\sqrt{N_h}\mathcal{T}_h, 1/\sqrt{N_l}\mathcal{T}_l]^T$ to obtain $\tilde{\mathcal{T}}$ and $\tilde{\mathcal{C}}$;
14:    Solve Equ. (16) for $\omega$ via Newton's method;
15:    Derive $\tilde{\mathcal{D}}^{(b)}$ from Equ. (15), $l \in [n]$;
16:    Take inverse Fourier transformation of $\tilde{\mathcal{D}}$ to derive $\mathcal{D}$. $\mathcal{D}$ includes feature dictionary $\mathcal{D}_l$ and recovery dictionary $\mathcal{D}_h$.
17: end for
18: Output: feature and recovery dictionaries: $\mathcal{D}_l$ and $\mathcal{D}_h$.
```

Algorithm 2 TSRGAN - Generation Phase

```
1: Input: $\mathcal{D}_h \in R^{d_h\times m\times n}, \mathcal{D}_l \in R^{d_l\times m\times n}$;
2: Use the trained DCGAN to generate low-resolution images $T_G \in R^{\frac{p}{c}\times\frac{q}{c}}$ from the latent vector $\mathbf{r} \in R^{u\times 1}$ in random distributions, and further concatenate $r$ images $T_G$ to image tensors $\mathcal{T}_G \in R^{\frac{p}{c}\times\frac{q}{c}\times r}$ ;
3: Reshape $\mathcal{T}_G$ to be $\mathcal{T}_l' \in R^{d_l\times N'\times n}$ through sampling and "folding", and use $\mathcal{T}_l' = \mathcal{D}_l * \mathcal{C}$ to obtain tensor sparse coding coefficients $\mathcal{C}$ with feature dictionary $\mathcal{D}_l$ derived above;
4: Derive $\mathcal{T}_h' = \mathcal{D}_h * \mathcal{C}$;
5: Transform $\mathcal{T}_h'$ into 2D images $X' \in R^{p\times q}$ through the "unfolding" process;
6: Output: Generated high-resolution 2D images $X'$.
```

4 Performance Evaluation

In this section, we present the results of proposed TSRGAN scheme on three datasets: MNIST [15], CIFAR10 [14], PASCAL2 VOC [5]. The image size of these three datasets applied in our model is $28\times28, 32\times32, 374\times374$ (downscaled from original 375×500 pixels), repectively.

4.1 Experiments Setting

DCGAN neural network parameters: the generator network has one fully connected layer and three transposed convolutional layers, with a decreasing number of 5×5 filter kernels, decreasing by a factor of 2 from 4×64 to 64 kernels and finally one channel of output images. The discriminator has three convolutional layers, with an increasing number of 5×5 filter kernels consistent with the generator. We use LeakyReLu [25] with parameter $= 0.2$ to avoid

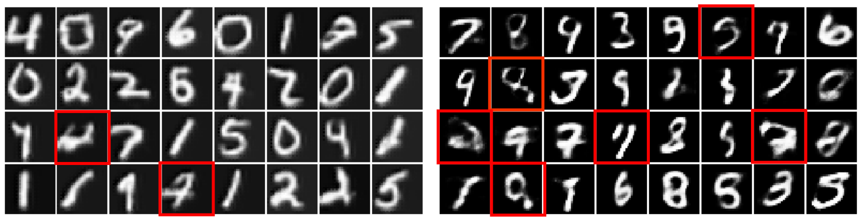

Figure 3: MNIST samples of 28×28 pixels: for TSRGAN and AAE model, we pick the generated digital number images which are hard to recognize (in red borders). The number of the obscure images of TSRGAN (left) and AAE (right) is 2 and 6, respectively.

max-pooling. Strided convolutions of size [1, 2, 2, 1] are used in each convolutional layer and tranposed convolutional layer. The learning rate is set to 1×10^{-4} and stochastic gradient descent is applied with a mini-batch size of 32.

By default, $u = 128, d_h = 16, d_l = 96, m = 128, n = 4, N = 10000, N' = 2500$. The number of directions for pixel-shifting is $r = 7$. The number of iterations $T = 10, S = 50$. The sparsity parameter $\lambda = 0.05$. β in **Prox** method is 0.05. For MNIST data, original images of size $p \times q, p = 28, q = 28$ (size values are set accordingly for other two datasets), downsampling rate $c = 2$.

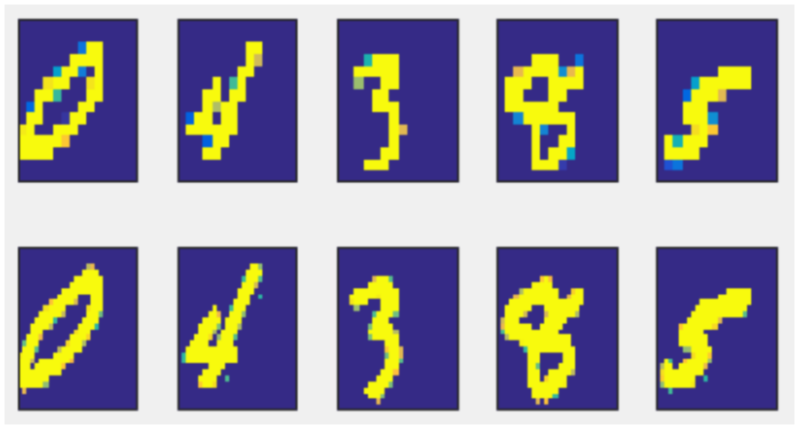

Figure 4: Ablation studies: MNIST samples using TSRGAN with (below) or without (above) tensor super-resolution. This testifies the significant effects of tensor-based super-resolution process.

4.2 Inception Score of Generation Results

We adopt the inception score (IS) metric [21][24] to compare performance of different schemes. The metric compares three kinds of samples, including our generated images, other generated images from similar generative methods and the real images. The inception score metric is $\exp\left[E_{\mathbf{x}\sim X'} KL(p(y|\mathbf{x})||p(y))\right]$. The comparison results of the AAE and our TSRGAN model are shown in Table 1. The proposed TSRGAN achieves better results in all three datasets,

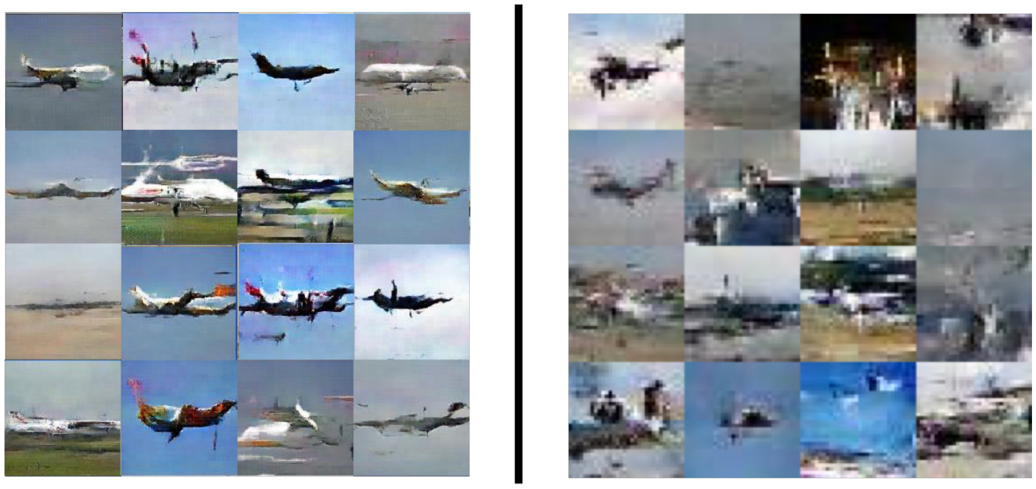

Figure 5: PASCAL2 samples of 374 × 374 pixels: we show the large size airplane samples generated by TSRGAN, compared with the same-sized samples generated by AAE for airplane images in PASCAL2.

especially for larges-sized PASCAL2 images (e.g. 374 × 374). Its inception score of 4.02 for PASCAL2 images significantly outperforms AAE of 3.81.

Dataset	CIFAR 10	Pascal2 VOC
AAE [17]	3.98	3.81
TSRGAN	**4.05**	**4.02**

Table 1: The inception score estimates metric are measured for AAE and proposed TSRGAN on CIFAR10 and Pascal2 VOC datasets.

4.3 Generated Images of TSRGAN

Some of the testing results on benchmark datasets are shown in the end of the paper. Fig. 3 shows the comparison of MNIST images generation with TSRGAN and AAE. The TSRGAN model provides images with more precise features of digital numbers, which benefits from its concise and efficient representation in tensor space. The effects of tensor super-resolution are shown in Fig. 4 for MINIST images with ablation studies. The images generated with general DCGAN have much coarser features without the tensor-based super-resolution process, which testifies that tensor super-resolution can significantly increase the image quality with more convincing details. Fig. 5 and Fig. 6 shows the generation results on PASCAL2 and CIFAR10 datasets, both testify the capability of TSRGAN in generating images with better quality, especially for large images (e.g. 374 × 374) in PASCAL2. Images generated with TSRGAN have more precise features and convincing details than images generated by AAE. This testifies that TSRGAN preserves spatial structures and local proximal information in a better way than traditional methods. Generally, the DCGAN generates basic shapes, structures, and colors of images, while the cascading tensor super-resolution process improves the images with more details.

It is worth noticing that the DCGAN applied in our proposed framework is a simple and straightforward method for illustrating our original ideas of combining tensor super-resolution. And experiments show the advantages of cascading tensor super-resolution with GAN methods for large image generation in a tensor space. However, as the proposed framework is orthogonal to other GAN approaches, like BigGANs [1] and Progressive GAN [11], it is not a necessary or fair comparison between our proposed method with present advanced GAN approaches. The DCGAN can be simply replaced with those more advanced GAN approaches to provide a better performance, with the advantages of leveraging tensor super-resolution as well.

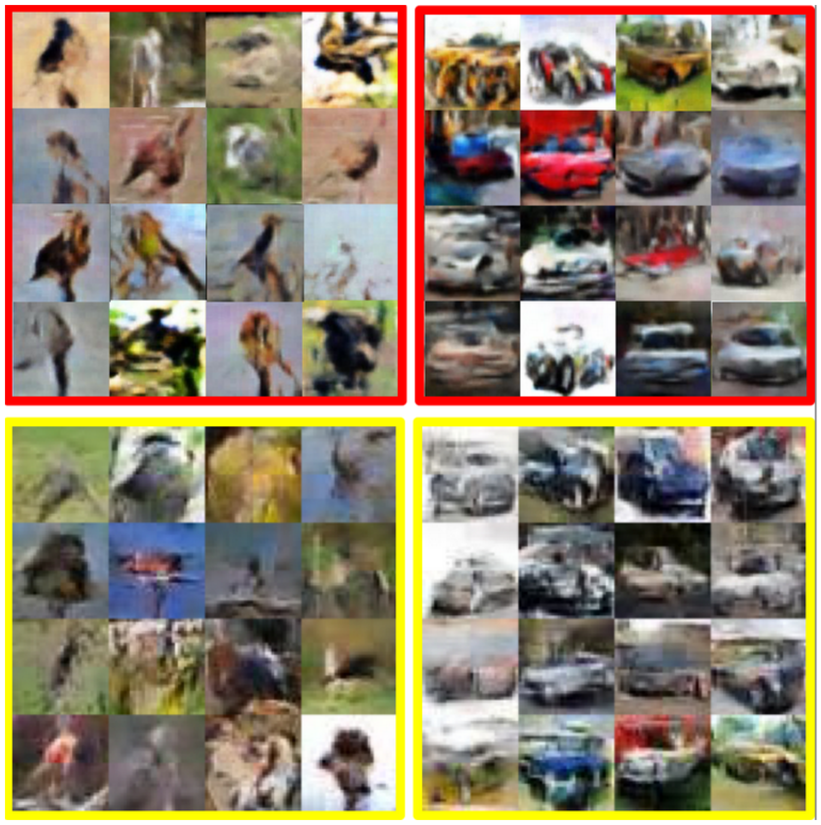

Figure 6: CIFAR10 samples of 128 $\times$ 128 pixels (4 $\times$ 4 image matrix of 32 $\times$ 32 pixels images): TSRGAN and AAE model. We show three kinds of samples: airplane, bird, and car. Above are generated by TSRGAN, and below are AAE.

5 Conclusion

In this paper, we proposed a TSRGAN scheme that integrates DCGAN model and tensor super-resolution, which is able to generate large-sized high-quality images. The proposed scheme applies tensor representation space as main operation space for image generation, which shows better results than traditional generative models working in image pixel space. Essentially, the adversarial process of TSRGAN takes place in a tensor space. Note that in the tensor super-resolution process, tensor sparse coding brings several advantages: (i) the size of dictionary, which accelerates the training process for deriving the representation dictionary; (ii) more concise and efficient representation for images, which is verified in the generated images in our experiments. TSRGAN is superior in preserving spatial structures and local proximity information in images. Accordingly, the tensor super-resolution benefits from tensor representation to generate higher-quality images, especially for large images. Our proposed cascading TSRGAN scheme surpasses the generative model AAE on three datasets (MNIST, CIFAR10, and PASCAL2).

References

[1] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.

[2] Emily L Denton, Soumith Chintala, Rob Fergus, et al. Deep generative image models using a laplacian pyramid of adversarial networks. In *Advances in Neural Information Processing Systems*, pages 1486–1494, 2015.

[3] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Learning a deep convolutional network for image super-resolution. In *European Conference on Computer Vision*, pages 184–199. Springer, 2014.

[4] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Image super-resolution using deep convolutional networks. *IEEE transactions on pattern analysis and machine intelligence*, 38(2):295–307, 2016.

[5] Mark Everingham, Luc Gool, Christopher K. Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *Int. J. Comput. Vision*, 88(2):303–338, June 2010.

[6] Jiang Fei, Xiao-Yang Liu, Hongtao Lu, and Ruimin Shen. Efficient multi-dimensional tensor sparse coding using t-linear combinations. In *Association for the Advancement of Artificial Intelligence*, 2018.

[7] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2014.

[8] Karol Gregor, Ivo Danihelka, Alex Graves, Danilo Jimenez Rezende, and Daan Wierstra. Draw: A recurrent neural network for image generation. *arXiv preprint arXiv:1502.04623*, 2015.

[9] Shuhang Gu, Nong Sang, and Fan Ma. Fast image super resolution via local regression. In *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*, pages 3128–3131. IEEE, 2012.

[10] Ning Hao, Misha E Kilmer, Karen Braman, and Randy C Hoover. Facial recognition using tensor-tensor decompositions. *SIAM Journal on Imaging Sciences*, 6(1):437–463, 2013.

[11] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.

[12] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations*, 2014.

[13] Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM RevIew*, 51(3):455–500, 2009.

[14] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

[15] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[16] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew P Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *CVPR*, volume 2, page 4, 2017.

[17] Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey. Adversarial autoencoders. In *International Conference on Learning Representations*, 2016.

[18] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018.

[19] Neal Parikh, Stephen Boyd, et al. Proximal algorithms. *Foundations and Trends® in Optimization*, 1(3):127–239, 2014.

[20] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep

convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[21] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Advances in Neural Information Processing Systems*, pages 2234–2242, 2016.

[22] Bin She, Yaojun Wang, Jiandong Liang, Zhining Liu, Chengyun Song, and Guangmin Hu. A data-driven avo inversion method via learned dictionaries and sparse representation. *Geophysics*, 83(6):1–91, 2018.

[23] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1874–1883, 2016.

[24] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2818–2826, 2016.

[25] Bing Xu, Naiyan Wang, Tianqi Chen, and Mu Li. Empirical evaluation of rectified activations in convolutional network. *arXiv preprint arXiv:1505.00853*, 2015.

[26] Jianchao Yang, John Wright, Thomas S Huang, and Yi Ma. Image super-resolution via sparse representation. *IEEE Transactions on Image Processing*, 19(11):2861–2873, 2010.

[27] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaolei Huang, Xiaogang Wang, and Dimitris Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *IEEE International Conference on Computer Vision*, pages 5907–5915, 2017.