Conditional Transferring Features: Scaling GANs to Thousands of Classes with 30% Less High-Quality Data for Training

Chunpeng Wu and Hai Li
Department of Electrical and Computer Engineering, Duke University, Durham, NC 27708, USA
{chunpeng.wu, hai.li}@duke.edu

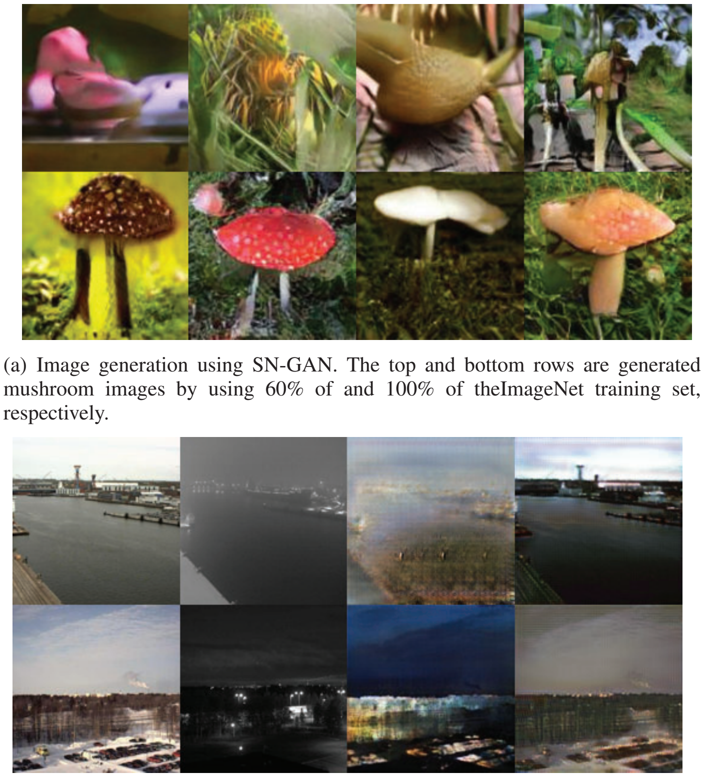
***Abstract*—Generative adversarial network (GAN) can greatly improve the quality of unsupervised image generation. Previous GAN-based methods often require a large amount of high-quality training data. This work aims to reduce the use of high-quality data in training, meanwhile scaling up GANs to thousands of classes. We propose an image generation method based on conditional transferring features, which can capture pixel-level semantic changes when transforming low-quality images into high-quality ones. Self-supervision learning is then integrated into our GAN architecture to provide more label-free semantic supervisory information observed from the training data. As such, training our GAN architecture requires much fewer high-quality images with a small number of additional low-quality images. Experiments show that even removing 30% high-quality images from the training set, our method can still achieve better image synthesis quality on CIFAR-10, STL-10, ImageNet, and CASIA-HWDB1.0, compared to previous competitive methods. Experiments on ImageNet with 1,000 classes of images and CASIA-HWDB1.0 with 3,755 classes of Chinese handwriting characters also validate the scalability of our method on object classes. Ablation studies further validate the contribution of our conditional transferring features and self-supervision learning to the quality of our synthesized images.**

***Index Terms*—Conditional transferring feature, generative adversarial network, high-quality image, image generation, low-quality image, self-supervision.**

I. Introduction

As one of the most exciting breakthroughs in unsupervised machine learning, *generative adversarial network* (GAN) [1] has been successfully applied to a variety of applications, such as face verification [2], human pose estimation [3], and small object detection [4]. In principle, GANs are trained in an adversarial manner: a *generator* produces new data by mimicking a targeted distribution; and a *discriminator* measures the similarity between the generated and targeted distributions, which in turn is used to adapt the generator.

A major performance metric of GAN is the quality of generated data, which can be measured by *Inception score* [5] and *Fréchet inception distance* (FID) [6]. Higher Inception or lower FID score indicates better image quality. The quality of generated data highly relies on both *volume* and *quality* of the training data. For example, our experiments on GAN-based image generation and image-to-image translation show dramatic performance degradation when reducing the number of high-quality training images.



(a) Image generation using SN-GAN. The top and bottom rows are generated mushroom images by using 60% of and 100% of theImageNet training set, respectively.

(b) Image-to-image translation (*Day→Night*) using Cycle-GAN. Columns from left to right: input day images, input night images, generated image with respectively 60% of and 100% of training data.

Fig. 1: Our experiments on (a) image generation and (b) image-to-image translation. We compare the quality of generated images by using 60% of and 100% of training data.

Figure 1(a) shows several mushroom images generated by SN-GAN [7] trained with 60% of (top row) or 100% of (bottom row) ImageNet training data [8]. The images in the bottom row obtained by using the entire training dataset present a more distinguishable appearance (*e.g.*, cap and stem of mushroom) and have much better quality. When removing 40% of the training data, the Inception score decreases from 21.1 to 14.8, and FID increases from 90.4 to 141.2. Figure 1(b) shows the image-to-image translation (*Day→Night*) by using

This work was in part supported by National Science Foundation (NSF) under grant CCF-1725456 and Department of Energy (DOE) under grant SC0017030.

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

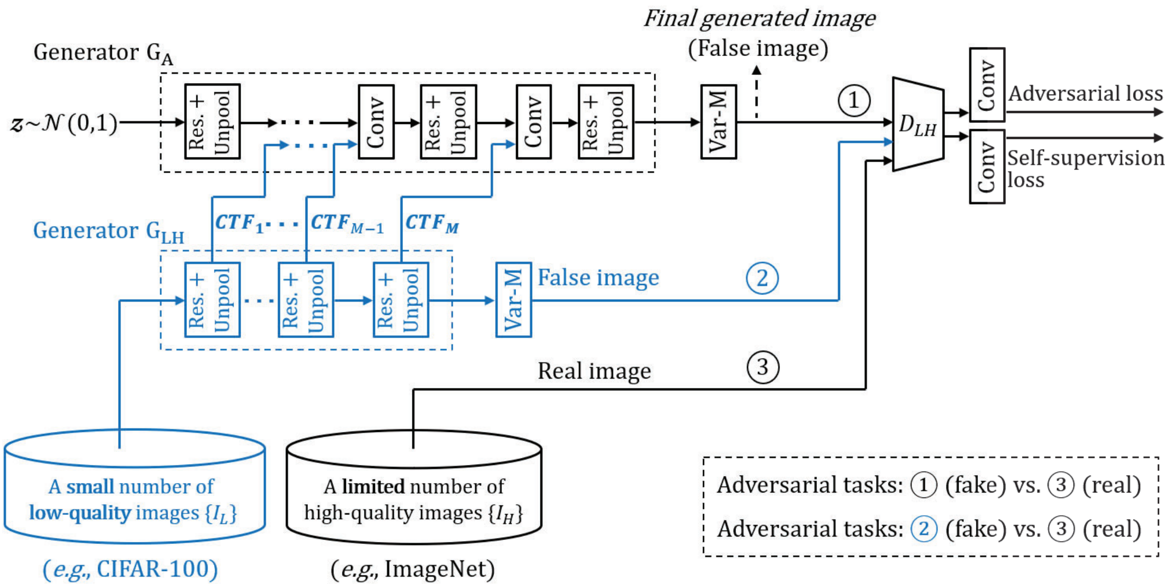


Fig. 2: Our proposed image generation method. The data-flow in blue extracts our proposed CTFs and provides CTFs to generator G_A. Tasks ② (in blue) and ③ (in dotted lines) are first adversarially trained, followed by the training of adversarial tasks ① and ③. The number of high-quality images I_H and low-quality images I_L are not required to be the same.

CycleGAN [9]. Comparably, the generated *Night* images in the fourth column trained with the full set of training data maintain both *Day* and *Night* features well, while those in the third column that use only 60% of the training data are blurred and miss some *Day* details. The high demand for high-quality training data has emerged as a major challenge of GAN-based methods—it is very difficult or even impossible to collect sufficient data for producing satisfactory results in real-world applications.

Another major challenge of GAN-based image generation is the scalability of object classes. Traditional image generation methods [10]–[15] produce images for tens of categories, such as MNIST classes, CIFAR-10 classes, STL-10 classes, and a subset of LSUN scenes [16]. Recently, AC-GAN [17], SN-GAN [7], and SN-GAN+Projection [7], [18] presented the results of all 1,000 ImageNet classes. AC-GAN does not directly tackle so many classes as a whole. Instead, it splits them into 10 subsets of 100 classes and processes each subset using a different GAN. Both SN-GAN and Projection run directly on all 1,000 classes. Compared to AC-GAN with Inception score 28.5 ± 0.20 and FID 260.0, SN-GAN has a worse Inception score 21.1 ± 0.35 and a better FID 141.2. SN-GAN+Projection gets both better Inception score 36.8 ± 0.44 and FID 92.4, while the results of using Projection only were not reported [18].

To address these two major challenges, we propose an image generation method based on *conditional transferring features* (CTFs) with three key solutions. First, we construct the training data with a portion of the original high-quality images and a small number of low-quality images. Second, our method extracts the CTFs by transforming low-quality images into high-quality images. Third, we further enhance our method with more label-free supervisory information observed from the training data. Our major contributions are:

- Our proposed CTFs reduce the required amount of high-quality training samples without sacrificing image synthesis quality. The self-supervision strategy further enhances the image synthesis quality. Ablation studies validate the effectiveness of our CTFs and self-supervision learning strategy.
- Experiments show that even removing 30% high-quality images from the training set, our method can still achieve better image synthesis quality (measured by Inception score and FID) on CIFAR-10, STL-10, ImageNet, and CASIA-HWDB1.0, compared to previous competitive methods [7], [18].
- Experiments on ImageNet (1,000 classes) and CASIA-HWDB1.0 (3,755 classes) also validate the scalability of our method on object classes.

II. Related Work

Many GAN research studies explore how to stabilize GAN training by modifying network architecture [11], [14] and optimizing algorithms [13], [15]. This is because the original GAN [1] suffers from instability-induced vanishing gradients and model collapse [19]. DC-GAN [11], as one of the early successful attempts, provides guidelines for designing a stable GAN such as the use of strided convolutions. Based on DC-GAN, Gulrajani *et al.* [14] further improve the discriminator design using deep residual blocks. LS-GAN [15] replaces previous cross-entropy loss functions with the least square loss functions to alleviate vanishing gradients. Wasserstein-GAN [13] theoretically improves the GAN architecture (*e.g.*,

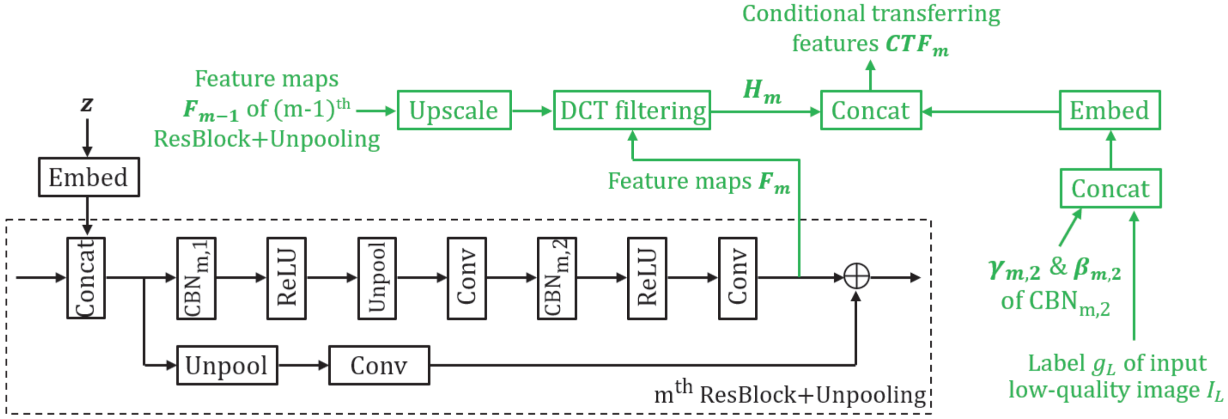

Conditional transferring features $\boldsymbol{CTF_m}$
Feature maps $\boldsymbol{F_{m-1}}$ of (m-1)th ResBlock+Unpooling
Upscale
DCT filtering
$\boldsymbol{H_m}$
Concat
Embed
Feature maps $\boldsymbol{F_m}$
Concat
z
Embed
Concat
CBN$_{m,1}$
ReLU
Unpool
Conv
CBN$_{m,2}$
ReLU
Conv
Unpool
Conv
mth ResBlock+Unpooling
$\boldsymbol{\gamma_{m,2}}$ & $\boldsymbol{\beta_{m,2}}$ of CBN$_{m,2}$
Label g_L of input low-quality image I_L

Fig. 3: Data-flow (in green) for extracting the conditional transferring features $\boldsymbol{CTF_m}$ in m^{th} *ResBlock+Unpooling* block in the generator G_{LH}.

removing sigmoid at the end of the discriminator) and the optimization algorithm (*e.g.*, avoiding the use of momentum and Adam [20]). SN-GAN [7] stabilizes the discriminator by using a weight normalization method. Projection [18] improves the way of incorporating conditional information into a discriminator. In this work, we propose a new approach that is rarely considered in traditional GAN-based methods, that is, using low-quality training data to facilitate the generation of high-quality images. Our GAN-based method can also scale to thousands of classes with significantly fewer high-quality training data.

Image generation and image-to-image translation are two typical use scenarios of GANs. GAN-based image generation methods [5], [6], [10], [17], [21]–[28] tackle the issues of multi-resolution, variation observation, architecture changing, energy estimation for samples, embedding recursive structures, integrating condition information into GANs, and quality evaluation of generated images. Recently, BigGAN [29] dramatically improves image synthesis quality by adding orthogonal regularization to the generator. Both subjective [10] and objective methods [5], [6], [11], [17] have been developed for quality evaluation. Here, we take the widely adopted objective methods, Inception score and FID, to measure the variation of generated images, and detect model collapse.

Traditional image-to-image translation studies [30]–[33] typically focus on a specific task. The prevalence of deep neural networks (DNNs), especially conditional GAN, further encourages higher translation accuracy and more translation tasks [23], [34]–[39]. A recent work [9] designs a common model for general transfer tasks and studies unpaired image-to-image translation. Our proposed CTFs are inspired by traditional image-to-image translation methods but take a different approach. Specifically, our CTFs are obtained by translating low-quality images to high-quality images. In contrast, traditional image-to-image translation methods often translate a set of images to another set of images with the same quality.

III. Image Generation Based on CTFs

Figure 2 shows our proposed image generation method. Following traditional GAN-based methods, the proposed design consists of a generator G_A and a discriminator D_{LH}. Our discriminator is also used for *self-supervision* (SP) learning. We introduce a generator G_{LH} for extracting CTFs. There are three learning tasks in our method:

- Task ① adopts G_A and D_{LH} to produce images that are similar to the high-quality I_H by using noise $z_A \sim \mathcal{N}(0,1)$ and the conditional transferring features $\boldsymbol{CTF_m}\,(m = 1, 2, ..., M)$ as G_A's input. The *Conv* layers in G_A convolute $\boldsymbol{CTF_m}\,(m = 1, 2, ..., M)$ by taking the output from the previous layer under the same resolution.
- Task ② (highlight in blue) adopts G_{LH} and D_{LH} to transform the low-quality images to high-quality images similar to I_H and provides the extracted $\boldsymbol{CTF_m}\,(m = 1, 2, ..., M)$ to G_A. Noises $z_m\,(m = 1, 2, ..., M)$ are injected into each *Res.+Unpool* (*ResBlock+Unpooling*) block in G_{LH}, respectively, to increase the randomness of the generated images.
- Task ③ distinguishes the real images from the synthetic images using D_{LH}.

During operation, the adversarial tasks ② and ③ are first trained for extracting the CTFs until no significant improvement can be observed. Afterwards, tasks ① and ③ are adversarially trained for image generation based on the CTFs.

A. Extracting CTFs

In G_{LH}, the m^{th} *ResBlock+Unpooling* block is used to extract the conditional transferring feature $\boldsymbol{CTF_m}$. Figure 3 shows the detailed extraction data-flow (in green). The random noise $z_m \sim \mathcal{N}(0,1)$ is embedded and concatenated to the input of the block. The Embed operator is previously described in [7], [18], [40]. We replace the batch-normalization layers in traditional ResBlock with *conditional batch-normalization* (CBN) [41] layers $CBN_{m,1}$ and

$CBN_{m,2}$ in *ResBlock+Unpooling*. $CBN_{m,1}$ and $CBN_{m,2}$ are conditional to the label information $c \in \{1, ..., c_H\}$ of the high-quality images where c_H is the class number of the high-quality images. According to the CBN's definition [41], for the layer $CBN_{m,1}$, an input activation $x_{m,1}$ is transformed into a normalized activation $y_{m,1}$ specific to a class $c \in \{1, ..., c_H\}$ calculated as:

$$y_{m,1} = \gamma^c_{m,1} \frac{x_{m,1} - \mu}{\sigma} + \beta^c_{m,1}\,, \tag{1}$$

where μ and σ are respectively the mean and standard deviation taken across spatial axes, and $\gamma^c_{m,1}$ and $\beta^c_{m,1}$ are trainable parameters specific to class c of $CBN_{m,1}$. Thus, the trainable parameters of $CBN_{m,1}$ are $\boldsymbol{\gamma_{m,1}} = \{\gamma^c_{m,1}\}^{C_H}_{c=1}$ and $\boldsymbol{\beta_{m,1}} = \{\beta^c_{m,1}\}^{C_H}_{c=1}$. Similarly, $\boldsymbol{\gamma_{m,2}} = \{\gamma^c_{m,2}\}^{C_H}_{c=1}$ and $\boldsymbol{\beta_{m,2}} = \{\beta^c_{m,2}\}^{C_H}_{c=1}$ denote the trainable parameters across all the classes of $CBN_{m,2}$.

The label information of both low-quality and high-quality images are concatenated to feature the differences between adjacent blocks of *ResBlock+Unpooling*. $\boldsymbol{CTF_m}$ is calculated by:

$$\boldsymbol{CTF_m} = \mathrm{Concat}\Big(\boldsymbol{H_m},\, \mathrm{Embed}\big(\mathrm{Concat}(\boldsymbol{\gamma_{m,2}}, \boldsymbol{\beta_{m,2}}, \boldsymbol{g_L})\big)\Big), \tag{2}$$

where $\boldsymbol{H_m} = \{H^t_m\}^T_{t=1}$ is the aggregated difference maps between the feature maps $\boldsymbol{F_m} = \{F^t_m\}^T_{t=1}$ and $\boldsymbol{F_{m-1}} = \{F^s_{m-1}\}^S_{s=1}$ respectively in the m^{th} and the $(m-1)^{th}$ blocks of *ResBlock+Unpooling*. Note that T might not be equal to S. To make it more clear, given a feature map F^t_m, the difference map between F^t_m and each $F^s_{m-1}\,(s = 1, 2, ..., S)$ is calculated, and then H^t_m is obtained by aggregating all S difference maps together. $\boldsymbol{g_L}$ is the labels of input low-quality images I_L, and $\boldsymbol{\gamma_{m,2}}$ and $\boldsymbol{\beta_{m,2}}$ include label information of high-quality images. The class information of low-quality and high-quality images are first concatenated together before they are concatenated to the difference maps $\boldsymbol{H_m}$. The class conditional parameters $\boldsymbol{\gamma_{m,1}}$ and $\boldsymbol{\beta_{m,1}}$ of the layer $CBN_{m,1}$ are not used in Equation (2) because the layer $CBN_{m,2}$ is in front of the *Unpooling* layer as shown in Figure 3, *i.e.*, its resolution corresponds to $\boldsymbol{F_{m-1}}$ but not $\boldsymbol{F_m}$. The feature maps $\boldsymbol{F_{m-1}}$ will be upsampled to the same size of $\boldsymbol{F_m}$ using bilinear interpolation. For the first block of *ResBlock+Unpooling* ($m = 1$), the previous $\boldsymbol{F_{m-1}}$ is replaced by the gray-level version of low-quality image I_L. The differences between a pair of feature maps are evaluated in a DCT-based frequency domain $\mathscr{D}$. $H^t_m\,(t = 1, 2, ..., T)$ is calculated as shown in Equation (3) when $m = 1$:

$$H^t_m = \mathscr{D}^{-1}\Big(\mathscr{D}(F^t_m) - \mathscr{D}(\mathrm{Upscale}(\mathrm{CvtGray}(I_L)))\Big), \tag{3}$$

where $\mathscr{D}(\cdot)$ and $\mathscr{D}^{-1}(\cdot)$ are DCT and inverse DCT transforms, Upscale function unsamples an image using bilinear interpolation, and CvtGray function converts a color image into a gray-level image. $H^t_m\,(t = 1, 2, ..., T)$ is calculated as shown in Equation (4) when $1 < m \leqslant M$:

$$H^t_m = \mathscr{D}^{-1}\left(\frac{\sum^S_{s=1}(\mathscr{D}(F^t_m) - \mathscr{D}(\mathrm{Upscale}(F^s_{m-1})))}{S}\right). \tag{4}$$

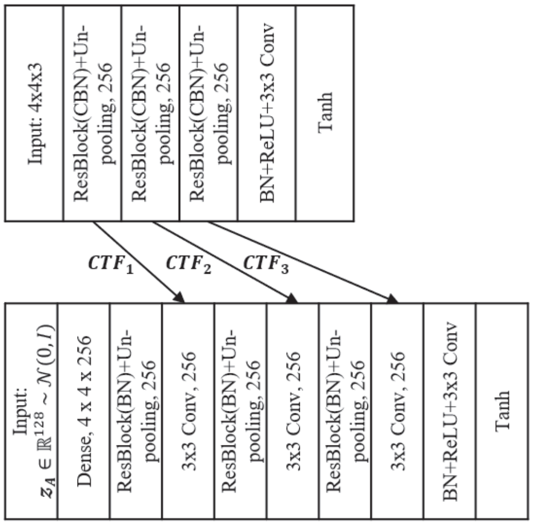

Input: 4x4x3
ResBlock(CBN)+Unpooling, 256
ResBlock(CBN)+Unpooling, 256
ResBlock(CBN)+Unpooling, 256
BN+ReLU+3x3 Conv
Tanh
CTF_1
CTF_2
CTF_3
Input: $z_A \in \mathbb{R}^{128} \sim \mathcal{N}(0,I)$
Dense, 4 x 4 x 256
ResBlock(BN)+Unpooling, 256
3x3 Conv, 256
ResBlock(BN)+Unpooling, 256
3x3 Conv, 256
ResBlock(BN)+Unpooling, 256
3x3 Conv, 256
BN+ReLU+3x3 Conv
Tanh

Fig. 4: The generators G_{LH} (top) and G_A (bottom) for synthesizing CIFAR-10.

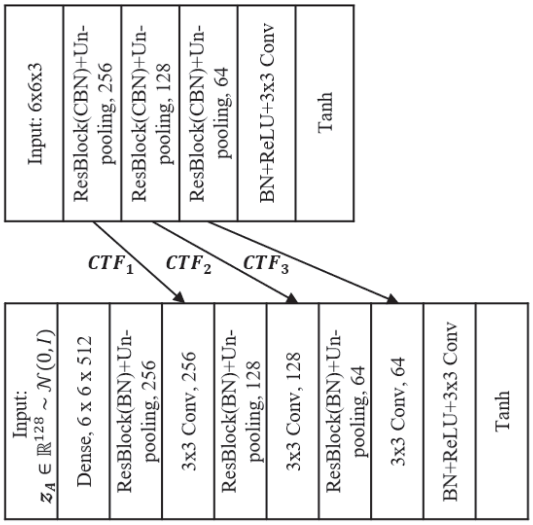

Input: 6x6x3
ResBlock(CBN)+Unpooling, 256
ResBlock(CBN)+Unpooling, 128
ResBlock(CBN)+Unpooling, 64
BN+ReLU+3x3 Conv
Tanh
CTF_1
CTF_2
CTF_3
Input: $z_A \in \mathbb{R}^{128} \sim \mathcal{N}(0,I)$
Dense, 6 x 6 x 512
ResBlock(BN)+Unpooling, 256
3x3 Conv, 256
ResBlock(BN)+Unpooling, 128
3x3 Conv, 128
ResBlock(BN)+Unpooling, 64
3x3 Conv, 64
BN+ReLU+3x3 Conv
Tanh

Fig. 5: The generators G_{LH} (top) and G_A (bottom) for synthesizing STL-10 (CASIA-HWDB1.0).

B. Self-supervision (SP) loss

We adopt a different SP learning task compared to traditional tasks [42]–[44] on predicting chromatic transformations, rotation, scaling, relative position of the image patches, *etc*. Specifically, for any image I_{LH} sampled from the high-quality image dataset $\{I_H\}$ or generated by G_A (or G_{LH}), we randomly cut an image patch from the image, paste it to a random location of the image, and record the bounding-box coordinates $Coor(I_{LH})$ of the patch. The "cut and paste" operation is denoted as $SP(\cdot)$. Formally, our SP loss is defined as $min||Conv(D_{LH}(SP(I_{LH}))) - Coor(I_{LH})||^2_2$. The SP loss is thus to minimize MSE loss between the recorded coordinates $Coor(I_{LH})$ and coordinates predicted by the layers $Conv(D_{LH}(\cdot))$ with transformed image $SP(I_{LH})$ as input.

TABLE I: Conditional image generation on CIFAR-10 and STL-10. Higher Inception score means higher image quality.

Method[1]	CIFAR-10		STL-10	
	Training data	Inception score	Training data	Inception score
DC-GAN [11]	50,000 (CIFAR-10 training set)	6.58	5,000 (STL-10 training set)	-
Improved-GAN [21]		8.09 ± .07		-
AC-GAN [17]		8.25 ± .07		-
SGAN [45]		8.59 ± .12		-
WGAN-GP [14]		8.67 ± .14		-
Splitting-GAN [27]		8.87 ± .09		-
PROG-GAN [28]		8.88 ± .05		9.34 ± .06
SN-GAN+Projection [7], [18]		9.01 ± .04		9.38 ± .08
Ours (Fewer HQ)	**32,500+10,000**	9.05 ± .06	**3,250+1,000**	9.44 ± .04
Ours (Entire HQ)	50,000+10,000	**9.17 ± .04**	5,000+1,000	**9.63 ± .05**

[1] Inception scores are obtained from the paper PROG-GAN [28] or by running source code of PROG-GAN and SN-GAN+Projection.

IV. Experiments

Our generator G_A has a similar architecture with the generator adopted in SN-GAN+Projection [7], [18], but differs in the followings two components: our G_A has additional layers for convoluting with CTFs provided by the generator G_{LH}; and our G_A uses a regular BN (instead of CBN in SN-GAN+Projection), while our G_{LH} uses CBN.

The training optimization method used in all experiments is Adam [20] with $\alpha = 0.0001$, $\beta_1 = 0$, $\beta_2 = 0.9$, and $\epsilon = 10^{-8}$. The objective function used for the adversarial loss is the standard version described in [1]. The scaling factors of the adversarial loss and auxiliary SP loss are fixed to 0.7 and 0.3, respectively, for all experiments. The mini-batch size is 16 for the experiments on CIFAR-10, STL-10, and CASIA-HWDB1.0, and 8 for the experiments on ImageNet. The training iterations we adopt are 100K for CIFAR-10 and STL-10, 450K for CASIA-HWDB1.0, and 650K for ImageNet.

A. Conditional Image Generation on CIFAR-10 and STL-10

For CIFAR-10, the architectures of the generators G_A and G_{LH} are shown in Figure 4, and the discriminator D_{LH} has the same architecture as SN-GAN+Projection's discriminator for CIFAR-10. For STL-10, the architectures of the generators G_A and G_{LH} are shown in Figure 5, and the discriminator D_{LH} is the same as SN-GAN+Projection's discriminator for STL-10.

Table I compares the quality of image generation of our method with previous works [7], [11], [14], [17], [18], [21], [27], [28], [45]. All the previous approaches take the full CIFAR-10 training set of 50,000 images. Our training data is a mixed-up of *high-quality* (HQ) images sampled from CIFAR-10 or STL-10 training set and *low-quality* (LQ) images. Since CIFAR-10 or STL-10 are already the "simplest" datasets, we take their down-sampled versions as LQ images. For comparison purpose, *Ours (Fewer HQ)* uses 32,500 CIFAR-10 and 10,000 down-sampled images as training data, and *Ours (Entire HQ)* applies the entire CIFAR-10 training set and 10,000 down-sampled images. According to the popular testing protocol [7], [14], [27], we scale all the generated images to 32×32 for CIFAR-10 classes and 48×48 for STL-10 classes.

The experiment shows that *Ours (Fewer HQ)* with fewer training data slightly outperforms previous methods. Using the entire CIFAR-10 or STL-10 training sets further improves the image quality of our method: *Ours (Entire HQ)* is respectively 1.7% and 2.7% better in Inception score, compared to previously best SN-GAN+Projection [7], [18]. Figure 6 visualizes examples of generated images and the corresponding Inception scores at seven sampling points during training. Compared to SN-GAN+Projection, the image quality of *Ours (Entire HQ)* is slightly better at each sampling point. The images after 62K iterations are not presented here, as the increase of Inception-score is not significant.

B. Conditional Image Generation on 3755-Class CASIA-HSWB1.0

This experiment takes the same architecture of STL-10 for the generator (G_A and G_{LH}) and discriminator G_{LH}.

To further validate the scalability on object classes, we compare the generation of 3,755 classes of CASIA-HWDB1.0 Chinese characters by using our method and SN-GAN+Projection [7], [18]. SN-GAN+Projection adopts the entire CASIA-HWDB1.0 training set (1,246,991 images) as the training data. Our training data takes 810,544 CASIA-HWDB1.0 training set as HQ images and 70,000 MNIST handwriting images as LQ images. The total number of our training data is (880,544) is 29.4% smaller than the entire CASIA-HWDB1.0 training set. The resolution of the generated images is set to 48×48 which is the same as original CASIA-HWDB1.0 dataset.

The quantitative comparison in Table II validates that *Ours (Fewer HQ)* and *Ours (Entire HQ)* can produce higher-quality Chinese characters in 3,755 CASIA-HWDB1.0 classes, compared to SN-GAN+Projection. The quality gap between SN-GAN+Projection and *Ours (Fewer HQ)* is larger than the gap presented in Table I, which implies our advantage on

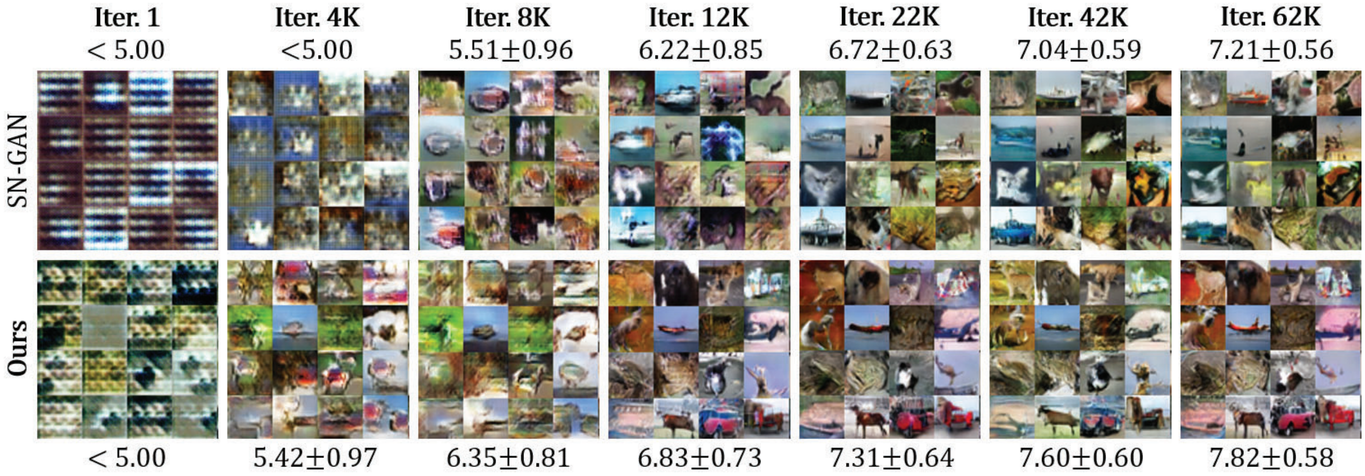


Fig. 6: Quality comparison of generated images of SN-GAN+Projection [7], [18] and *Ours (Entire HQ)* on CIFAR-10 during training. The iteration indexes and averaged Inception scores (the higher the better) are provided.

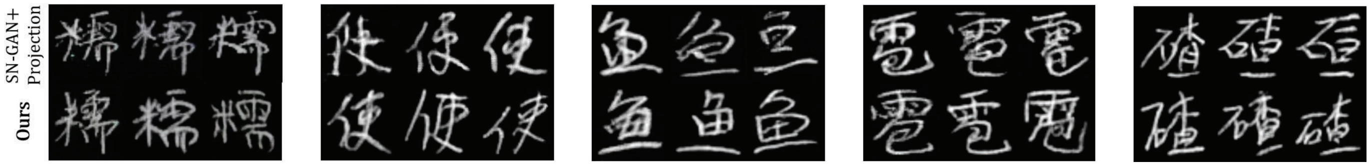


Fig. 7: Comparison of generated Chinese characters using SN-GAN+Projection [7], [18] and Ours (Fewer HQ).

TABLE II: Conditional image generation on CASIA-HWDB1.0. Higher Inception score means higher image quality.

Method	Training data	Inception score
SN-GAN+Projection	1,246,991	10.2 ± .17
Ours (Fewer HQ)	**880,544 (810,544+70,000)**	11.3 ± .13
Ours (Entire HQ)	1,316,991 (1,246,991+70,000)	**13.6 ± .15**

more image classes. Qualitatively, the strokes of our generated characters are more distinguishable, as example images in Figure 7 show.

C. Conditional Image Generation on ImageNet

The architectures of the generators G_A and G_{LH} are shown in Figure 8, and the discriminator G_{LH} is the same as SN-GAN+Projection's discriminator for ImageNet.

We use our method for conditional image generation on ImageNet classes and compare it to AC-GAN [23], SN-GAN [7] and SN-GAN+Projection [7], [18]. The training of the three previous GANs adopt the entire ImageNet training set (1,282,167 images). The training data of *Ours (Fewer HQ)* contains 833,408 ImageNet training set as HQ images and 60,000 CIFAR-100 images as LQ ones. Thus, the total number of *Ours (Fewer HQ)* is 30.3% smaller than the entire ImageNet training set used in previous methods [7], [17], [18]. The resolution of the generated images is set to 128×128 to compare with previous methods.

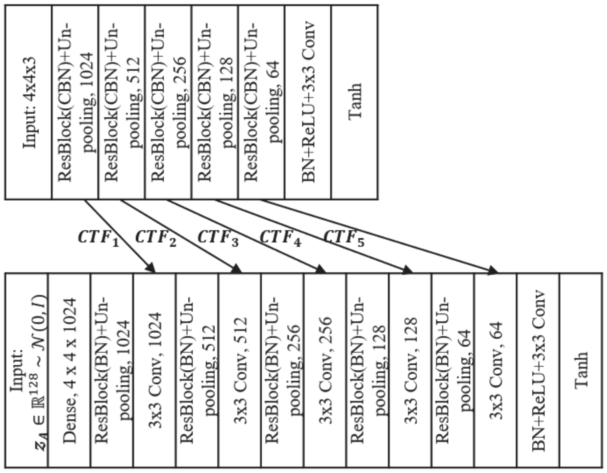

Input: 4x4x3
ResBlock(CBN)+Un-pooling, 1024
ResBlock(CBN)+Un-pooling, 512
ResBlock(CBN)+Un-pooling, 256
ResBlock(CBN)+Un-pooling, 128
ResBlock(CBN)+Un-pooling, 64
BN+ReLU+3x3 Conv
Tanh
CTF_1
CTF_2
CTF_3
CTF_4
CTF_5
Input: $z_A \in \mathbb{R}^{128} \sim \mathcal{N}(0,I)$
Dense, 4 x 4 x 1024
ResBlock(BN)+Un-pooling, 1024
3x3 Conv, 1024
ResBlock(BN)+Un-pooling, 512
3x3 Conv, 512
ResBlock(BN)+Un-pooling, 256
3x3 Conv, 256
ResBlock(BN)+Un-pooling, 128
3x3 Conv, 128
ResBlock(BN)+Un-pooling, 64
3x3 Conv, 64
BN+ReLU+3x3 Conv
Tanh

Fig. 8: The generators G_{LH} (top) and G_A (bottom) for synthesizing ImageNet.

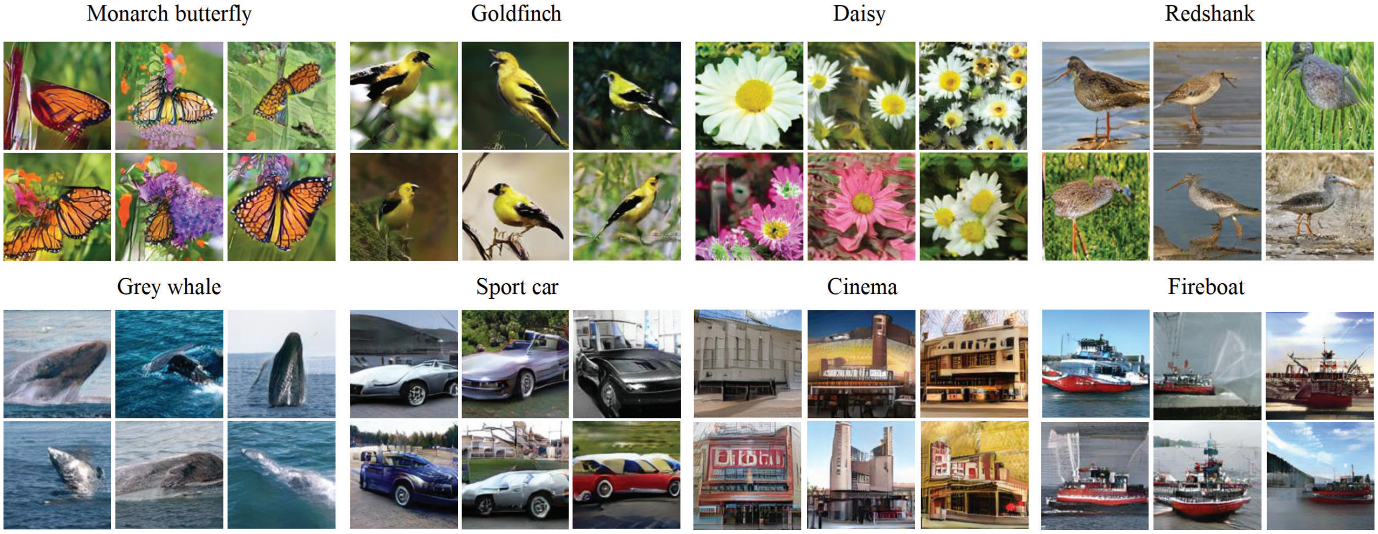
Table III summarizes the comparison with previous methods, and Figure 9 shows some examples of the generated images by *Ours (Entire HQ)*. *Ours (Fewer HQ)* outperforms previous methods, even though it uses 30.3% fewer training data. Using the entire ImageNet training set and CIFAR-100 images, *Ours (Entire HQ)* is 19.3% better than the previous best SN-GAN+Projection [7], [18] in Inception score.

D. Ablation studies and sensitivity analysis

Our method brings in two new components: CTFs and SP. We evaluate the respective contribution of each component

TABLE III: Conditional image generation on ImageNet. Higher Inception score (or lower FID) means higher image quality.

Method	Training data	Inception score	FID
AC-GAN [23]	1,282,167 (ImageNet training set)	28.5 ± .20	260.00
SN-GAN [7]		21.1 ± .35	141.20
SN-GAN+Projection ([7], [18])		36.8 ± .44	92.40
Ours (Fewer HQ)	**893,408 (833,408+60,000)**	39.4 ± .43	83.0
Ours (Entire HQ)	1,342,167 (1,282,167+60,000)	**43.9 ± .46**	**76.7**


Monarch butterfly

Goldfinch

Daisy

Redshank

Grey whale

Sport car

Cinema

Fireboat

Fig. 9: Our generated images of the ImageNet classes (*Ours (Entire HQ)*).

TABLE IV: Ablation study of *Ours (Entire HQ)* on CASIA-HWDB1.0 and ImageNet. Higher Inception score means higher image quality.

Method	Inception score (CASIA-HWDB1.0)	Inception score (ImageNet)
Ours (Entire HQ) w/o CTFs	7.2 ± .12	32.1 ± .47
Ours (Entire HQ) w/o SP	10.9 ± .16	37.0 ± .50
Ours (Entire HQ)	**13.6 ± .15**	**43.9 ± .46**

TABLE V: Performance comparison of adding more ResBlocks to SN-GAN+Projection. Higher Inception score means higher image quality.

Method	Inception score (ImageNet)
SN-GAN+Projection with 1 more ResBlock	37.3 ± .46
SN-GAN+Projection with 2 more ResBlocks	37.9 ± .44
SN-GAN+Projection with 4 more ResBlocks	37.9 ± .48
SN-GAN+Projection with 6 more ResBlocks	37.6 ± .48
SN-GAN+Projection	36.8 ± .44
Ours (Entire HQ)	**43.9 ± .46**

on CASIA-HWDB1.0 and ImageNet. Table IV presents the ablation studies in Inception score. As can be seen, our image synthesis quality severely drops when CTFs or SP is removed.

Our proposed approach has one more generator G_{LH} compared to SN-GAN+Projection but is not the same as simply adding more layers/blocks in previous methods. For example, we gradually increase the number of ResBlock in SN-GAN+Projection. Table V lists its performance change. The Inception score improves slightly with a maximum of 37.9 ± .48, which is much smaller than 43.9 ± .46 obtained by *Ours (Entire HQ)*. The results demonstrate the effectiveness of CTFs enabled by G_{LH} in our design.

V. Conclusion

Previous GAN-based image generation methods face the challenges of heavy dependency on high-quality training data. In contrast, collecting low-quality images is relatively easier and more economical. By observing the learning process when transforming low-quality images into high-quality ones, we find that combining intermediate output with the class information, or *conditional transferring features* (CTFs), can improve the quality of image generation and the scalability of the object classes of GAN. Moreover, we integrate self-supervision learning into our GAN architecture to further improve the learning ability of the GAN. Experiments on conditional image generation tasks show that our method performs better than previous methods, even after removing 30% high-quality training data. And our method successfully scales GANs to thousands of object classes such as the 1,000 ImageNet classes and 3,755 CASIA-HWDB1.0 classes.

References

[1] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative Adversarial Nets,"

Conference on Neural Information Processing Systems (NeurIPS), pp. 2672–2680, 2014.

[2] Y. Li, L. Song, X. Wu, R. He, and T. Tan, "Anti-Makeup: Learning A Bi-Level Adversarial Network for Makeup-Invariant Face Verification," *AAAI Conference on Artificial Intelligence (AAAI)*, pp. 7057–7064, 2018.

[3] Y. Chen, C. Shen, X.-S. Wei, L. Liu, and J. Yang, "Adversarial PoseNet: A Structure-aware Convolutional Network for Human Pose Estimation," *IEEE Conference on Computer Vision (ICCV)*, pp. 1212–1221, 2017.

[4] J. Li, X. Liang, Y. Wei, T. Xu, J. Feng, and S. Yan, "Perceptual Generative Adversarial Networks for Small Object Detection," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1222–1230, 2017.

[5] T. Salimans and D. P. Kingma, "Weight Normalization: A Simple Reparameterization to Accelerate Training of Deep Neural Networks," *Conference on Neural Information Processing Systems (NeurIPS)*, pp. 1–11, 2016.

[6] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium," *Conference on Neural Information Processing Systems (NeurIPS)*, pp. 6626–6637, 2017.

[7] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral Normalization for Generative Adversarial Network," *International Conference on Learning Representations (ICLR)*, pp. 1–14, 2018.

[8] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and F. F. Li, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115(3), pp. 211–252, 2015.

[9] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks," *IEEE Conference on Computer Vision (ICCV)*, pp. 2223–2232, 2017.

[10] E. Denton, S. Chintala, A. Szlam, and R. Fergus, "Deep Generative Image Models Using a Laplacian Pyramid of Adversarial Networks," *Conference on Neural Information Processing Systems (NeurIPS)*, pp. 1486–1494, 2015.

[11] A. Radford, L. Metz, and S. Chintala, "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks," *International Conference on Learning Representations (ICLR)*, pp. 1–13, 2016.

[12] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, "InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets," *Advances in Neural Information Processing Systems (NIPS)*, 2016.

[13] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein GAN," *arxiv preprint arXiv:1701.07875v3*, pp. 1–32, 2017.

[14] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, "Improved Training of Wasserstein GANs," *Conference on Neural Information Processing Systems (NeurIPS)*, pp. 5767–5777, 2017.

[15] X. Mao, Q. Li, H. Xie, R. Y. K. Lau, Z. Wang, and S. P. Smolley, "Least Squares Generative Adversarial Networks," *IEEE Conference on Computer Vision (ICCV)*, pp. 2794–2802, 2017.

[16] F. Yu, A. Seff, Y. Zhang, S. Song, T. Funkhouser, and J. Xiao, "LSUN: Construction of a Large-Scale Image Dataset Using Deep Learning with Humans in the Loop," *arXiv:1506.03365*, 2017.

[17] A. Odena, C. Olah, and J. Shlens, "Conditional Image Synthesis with Auxiliary Classifier GANs," *IEEE Conference on Machine Learning (ICML)*, pp. 2642–2651, 2017.

[18] T. Miyato and M. Koyama, "cGANS with Projection Discriminator," *International Conference on Learning Representations (ICLR)*, pp. 1–13, 2018.

[19] M. Arjovsky and L. Bottou, "Towards Principled Methods for Training Generative Adversarial Networks," *International Conference on Learning Representations (ICLR)*, pp. 1–17, 2017.

[20] D. P. Kingma and J. L. Ba, "Adam: A Method for Stochastic Optimization," *International Conference on Learning Representations (ICLR)*, pp. 1–11, 2015.

[21] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved Techniques for Training GANs," *Conference on Neural Information Processing Systems (NeurIPS)*, pp. 2226–2234, 2016.

[22] I. Durugkar, I. Gemp, and S. Mahadevan, "Generative Multi-Adversarial Networks," *International Conference on Learning Representations (ICLR)*, pp. 1–10, 2017.

[23] V. Dumoulin, I. Belghazi, B. Poole, O. Mastropietro, A. Lamb, M. Arjovsky, and A. Courville, "Adversarially Learned Inference," *International Conference on Learning Representations (ICLR)*, pp. 1–13, 2017.

[24] Z. Dai, A. Almahairi, P. Bachman, E. H. Hovy, and A. C. Courville, "Calibrating Energy-based Generative Adversarial Networks," *International Conference on Learning Representations (ICLR)*, pp. 1–17, 2017.

[25] J. Yang, A. Kannan, D. Batra, and D. Parikh, "LR-GAN: Layered Recursive Generative Adversarial Networks for Image Generation," *International Conference on Learning Representations (ICLR)*, pp. 1–14, 2017.

[26] D. Warde-Farley and Y. Bengio, "Improving Generative Adversarial Networks with Denoising Feature Matching," *International Conference on Learning Representations (ICLR)*, pp. 1–11, 2017.

[27] G. L. Grinblat, L. C. Uzal, and P. M. Granitto, "Class-Splitting Generative Adversarial Networks," *arXiv preprint arXiv:1709.07359v1*, pp. 1–10, 2017.

[28] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive Growing of GANs for Improved Quality, Stability, and Variation," *International Conference on Learning Representations (ICLR)*, pp. 1–12, 2018.

[29] A. Brock, J. Donahue, and K. Simonyan, "Large Scale GAN Training for High Fidelity Natural Image Synthesis," *International Conference on Learning Representations (ICLR)*, pp. 1–29, 2019.

[30] A. A. Efros and W. T. Freeman, "Image Quilting for Texture Synthesis and Transfer," *ACM SIGGRAPH*, pp. 341–346, 2001.

[31] R. Fergus, B. Singh, A. Hertzmann, S. T. Roweis, and W. T. Freeman, "Removing Camera Shake from a Single Photograph," *ACM Transactions on Graphics (TOG)*, vol. 25(3), pp. 787–794, 2001.

[32] T. Chen, M.-M. Cheng, P. Tan, A. Shamir, and S.-M. Hu, "Sketch2photo: Internet Image Montage," *ACM Transactions on Graphics (TOG)*, vol. 28(5), pp. 1–10, 2009.

[33] P.-Y. Laffont, Z. Ren, X. Tao, C. Qian, and J. Hays, "Transient Attributes for High-Level Understanding and Editing of Outdoor Scenes," *ACM Transactions on Graphics (TOG)*, vol. 33(4), pp. 1–11, 2014.

[34] J. Long, E. Shelhamer, and T. Darrell, "Fully Convolutional Networks for Semantic Segmentation," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3431–3440, 2015.

[35] D. Eigen and R. Fergus, "Predicting Depth, Surface Normals and Semantic Labels with a Common Multi-Scale Convolutional Architecture," *IEEE Conference on Computer Vision (ICCV)*, pp. 2650–2658, 2015.

[36] M.-Y. Liu and O. Tuzel, "Coupled Generative Adversarial Networks," *Conference on Neural Information Processing Systems (NeurIPS)*, pp. 469–477, 2016.

[37] R. Zhang, P. Isola, and A. A. Efros, "Colorful Image Colorization," *European Conference on Computer Vision (ECCV)*, pp. 649–666, 2016.

[38] X. Wang and A. Gupta, "Generative Image Modeling Using Style and Structure Adversarial Networks," *European Conference on Computer Vision (ECCV)*, pp. 318–335, 2016.

[39] J. Johnson, A. Alahi, and F.-F. Li, "Perceptual Losses for Real-Time Style Transfer and Super-Resolution," *European Conference on Computer Vision (ECCV)*, pp. 694–711, 2016.

[40] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, "Generative Adversarial Text to Image Synthesis," *IEEE Conference on Machine Learning (ICML)*, pp. 1060–1069, 2016.

[41] V. Dumoulin, J. Shlens, and M. Kudlur, "A Learned Representation for Artistic Style," *International Conference on Learning Representations (ICLR)*, pp. 1–11, 2017.

[42] A. Dosovitskiy, J. T. Springenberg, M. Riedmiller, and T. Brox, "Discriminative Unsupervised Feature Learning with Convolutional Neural Networks," *Conference on Neural Information Processing Systems (NeurIPS)*, pp. 766–774, 2014.

[43] S. Gidaris, P. Singh, and N. Komodakis, "Unsupervised Representation Learning by Predicting Image Rotations," *International Conference on Learning Representations (ICLR)*, pp. 1–14, 2018.

[44] M. Noroozi and P. Favaro, "Unsupervised Learning of Visual Representations by Solving Jigsaw Puzzles," *European Conference on Computer Vision (ECCV)*, pp. 69–84, 2016.

[45] X.Huang, Y. Li, O. Poursaeed, J. Hopcroft, and S. Belongie, "Stacked Generative Adversarial Networks," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5077–5086, 2017.