Stacked Generative Adversarial Networks

Xun Huang[1] Yixuan Li[2] Omid Poursaeed[2] John Hopcroft[1] Serge Belongie[1,3]
[1]Department of Computer Science, Cornell University
[2]School of Electrical and Computer Engineering, Cornell University [3]Cornell Tech
{xh258,yl2363,op63,sjb344}@cornell.edu jeh@cs.cornell.edu

Abstract

In this paper, we propose a novel generative model named Stacked Generative Adversarial Networks (SGAN), which is trained to invert the hierarchical representations of a bottom-up discriminative network. Our model consists of a top-down stack of GANs, each learned to generate lower-level representations conditioned on higher-level representations. A representation discriminator is introduced at each feature hierarchy to encourage the representation manifold of the generator to align with that of the bottom-up discriminative network, leveraging the powerful discriminative representations to guide the generative model. In addition, we introduce a conditional loss that encourages the use of conditional information from the layer above, and a novel entropy loss that maximizes a variational lower bound on the conditional entropy of generator outputs. We first train each stack independently, and then train the whole model end-to-end. Unlike the original GAN that uses a single noise vector to represent all the variations, our SGAN decomposes variations into multiple levels and gradually resolves uncertainties in the top-down generative process. Based on visual inspection, Inception scores and visual Turing test, we demonstrate that SGAN is able to generate images of much higher quality than GANs without stacking.

1. Introduction

Recent years have witnessed tremendous success of deep neural networks (DNNs), especially the kind of bottom-up neural networks trained for discriminative tasks. In particular, Convolutional Neural Networks (CNNs) have achieved impressive accuracy on the challenging ImageNet classification benchmark [30, 56, 57, 21, 52]. Interestingly, it has been shown that CNNs trained on ImageNet for classification can learn representations that are transferable to other tasks [55], and even to other modalities [20]. However, bottom-up discriminative models are focused on learning useful representations from data, being incapable of capturing the data distribution.

Learning top-down generative models that can explain complex data distribution is a long-standing problem in machine learning research. The expressive power of deep neural networks makes them natural candidates for generative models, and several recent works have shown promising results [28, 17, 44, 36, 68, 38, 9]. While state-of-the-art DNNs can rival human performance in certain discriminative tasks, current best deep generative models still fail when there are large variations in the data distribution.

A natural question therefore arises: can we leverage the hierarchical representations in a discriminatively trained model to help the learning of top-down generative models? In this paper, we propose a generative model named *Stacked Generative Adversarial Networks* (SGAN). Our model consists of a top-down stack of GANs, each trained to generate "plausible" lower-level representations conditioned on higher-level representations. Similar to the image discriminator in the original GAN model which is trained to distinguish "fake" images from "real" ones, we introduce a set of *representation discriminators* that are trained to distinguish "fake" representations from "real" representations. The *adversarial loss* introduced by the representation discriminator forces the intermediate representations of the SGAN to lie on the manifold of the bottom-up DNN's representation space. In addition to the adversarial loss, we also introduce a *conditional loss* that imposes each generator to use the higher-level conditional information, and a novel *entropy loss* that encourages each generator to generate diverse representations. By stacking several GANs in a top-down way and using the top-most GAN to receive labels and the bottom-most GAN to generate images, SGAN can be trained to model the data distribution conditioned on class labels. Through extensive experiments, we demonstrate that our SGAN is able to generate images of much higher quality than a vanilla GAN. In particular, our model obtains state-of-the-art Inception scores on CIFAR-10 dataset.

2. Related Work

Deep Generative Image Models. There has been a large body of work on generative image modeling with deep

learning. Some early efforts include Restricted Boltzmann Machines [22] and Deep Belief Networks [23]. More recently, several successful paradigms of deep generative models have emerged, including the auto-regressive models [32, 16, 58, 44, 45, 19], Variational Auto-encoders (VAEs) [28, 27, 50, 64, 18], and Generative Adversarial Networks (GANs) [17, 5, 47, 49, 53, 33]. Our work builds upon the GAN framework, which employs a generator that transforms a noise vector into an image and a discriminator that distinguishes between real and generated images.

However, due to the vast variations in image content, it is still challenging for GANs to generate diverse images with sufficient details. To this end, several works have attempted to factorize a GAN into a series of GANs, decomposing the difficult task into several more tractable sub-tasks. Denton *et al*. [5] propose a LAPGAN model that factorizes the generative process into multi-resolution GANs, with each GAN generating a higher-resolution residual conditioned on a lower-resolution image. Although both LAPGAN and SGAN consist of a sequence of GANs each working at one scale, LAPGAN focuses on generating *multi-resolution images* from coarse to fine while our SGAN aims at modeling *multi-level representations* from abstract to specific. Wang and Gupta [62] propose a S^2-GAN, using one GAN to generate surface normals and another GAN to generate images conditioned on surface normals. Surface normals can be viewed as a specific type of image representations, capturing the underlying 3D structure of an indoor scene. On the other hand, our framework can leverage the more general and powerful multi-level representations in a pre-trained discriminative DNN.

There are several works that use a pre-trained discriminative model to aid the training of a generator. [31, 7] add a regularization term that encourages the reconstructed image to be similar to the original image in the feature space of a discriminative network. [59, 26] use an additional "style loss" based on Gram matrices of feature activations. Different from our method, all the works above only add loss terms to regularize the generator's *output*, without regularizing its *internal representations*.

Matching Intermediate Representations Between Two DNNs. There have been some works that attempt to "match" the intermediate representations between two DNNs. [51, 20] use the intermediate representations of one pre-trained DNN to guide another DNN in the context of knowledge transfer. Our method can be considered as a special kind of knowledge transfer. However, we aim at transferring the knowledge in a bottom-up DNN to a top-down generative model, instead of another bottom-up DNN. Also, some auto-encoder architectures employ layer-wise reconstruction loss [60, 48, 67, 66]. The layer-wise loss is usually accompanied by lateral connections from the encoder to the decodery. On the other hand, SGAN is a generative model and does not require any information from the encoder once training completes. Another important difference is that we use adversarial loss instead of $L2$ reconstruction loss to match intermediate representations.

Visualizing Deep Representations. Our work is also related to the recent efforts in visualizing the internal representations of DNNs. One popular approach uses gradient-based optimization to find an image whose representation is close to the one we want to visualize [37]. Other approaches, such as [8], train a top-down deconvolutional network to reconstruct the input image from a feature representation by minimizing the Euclidean reconstruction error in image space. However, there is inherent uncertainty in the reconstruction process, since the representations in higher layers of the DNN are trained to be invariant to irrelevant transformations and to ignore low-level details. With Euclidean training objective, the deconvolutional network tends to produce blurry images. To alleviate this problem, Dosovitskiy abd Brox [7] further propose a feature loss and an adversarial loss that enables much sharper reconstructions. However, it still does not tackle the problem of uncertainty in reconstruction. Given a high-level feature representation, the deconvolutional network deterministically generates a single image, despite the fact that there exist many images having the same representation. Also, there is no obvious way to sample images because the feature prior distribution is unknown. Concurrent to our work, Nguyen *et al*. [42] incorporate the feature prior with a variant of denoising auto-encoder (DAE). Their sampling relies on an iterative optimization procedure, while we are focused on efficient feed-forward sampling.

3. Methods

In this section we introduce our model architecture. In Sec. 3.1 we briefly overview the framework of Generative Adversarial Networks. We then describe our proposal for Stacked Generative Adversarial Networks in Sec. 3.2. In Sect. 3.3 and 3.4 we will focus on our two novel loss functions, conditional loss and entropy loss, respectively.

3.1. Background: Generative Adversarial Network

As shown in Fig. 1 (a), the original GAN [17] is trained using a two-player min-max game: a discriminator D trained to distinguish generated images from real images, and a generator G trained to fool D. The discriminator loss $\mathcal{L}_D$ and the generator loss $\mathcal{L}_G$ are defined as follows:

$$\mathcal{L}_D = \mathbb{E}_{x\sim P_{data}}[-\log D(x)] + \mathbb{E}_{z\sim P_z}[-\log(1-D\big(G(z)\big))] \quad (1)$$

$$\mathcal{L}_G = \mathbb{E}_{z\sim P_z}[-\log(D\big(G(z)\big))] \quad (2)$$

In practice, D and G are usually updated alternately. The training process matches the generated image distribution

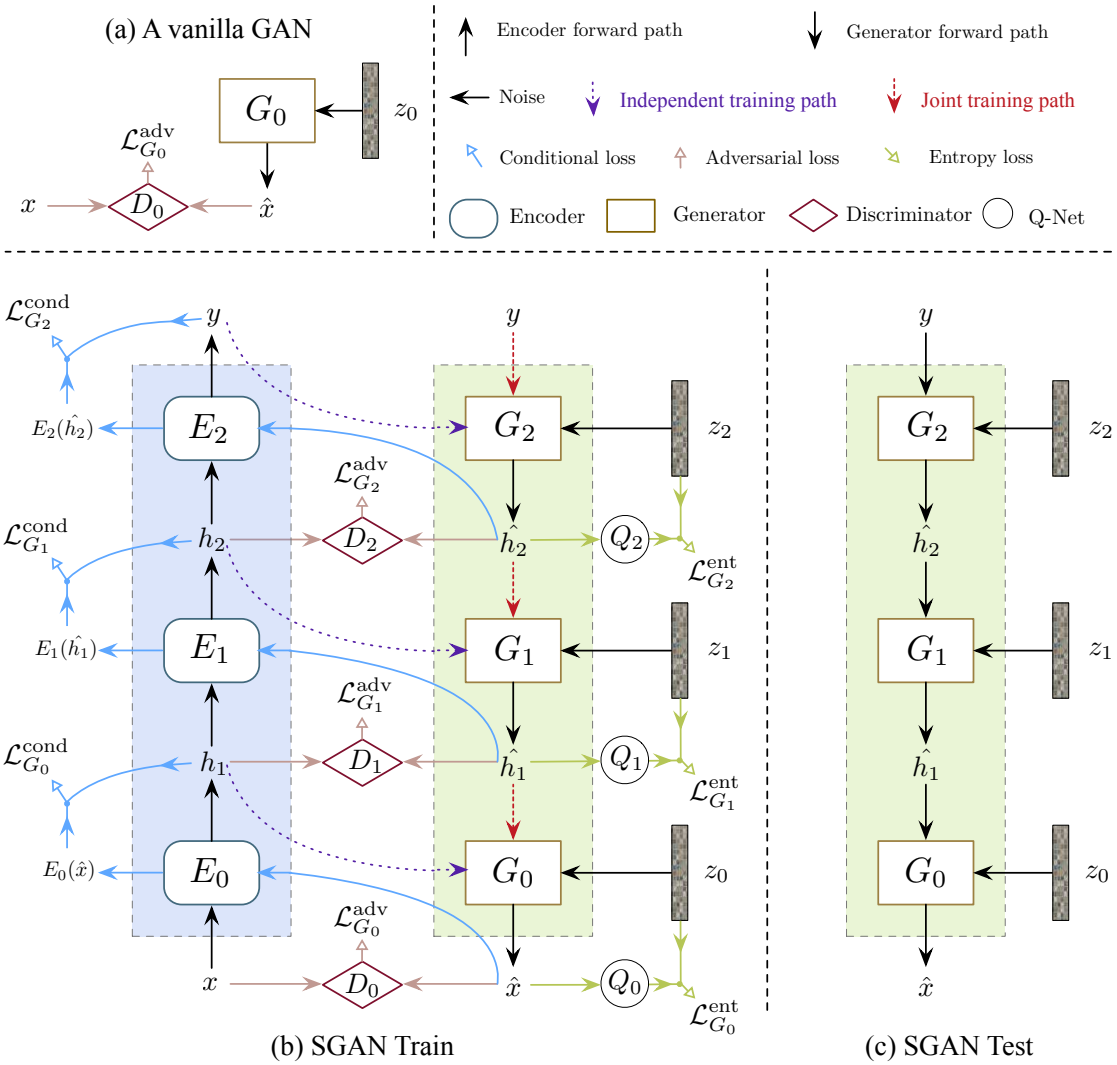


Figure 1: **An overview of SGAN.** (a) The original GAN in [17]. (b) The workflow of training SGAN, where each generator G_i tries to generate plausible features that can fool the corresponding representation discriminator D_i. Each generator receives conditional input from encoders in the independent training stage, and from the upper generators in the joint training stage. (c) New images can be sampled from SGAN (during test time) by feeding random noise to each generator G_i.

$P_G(x)$ with the real image distribution $P_{data}(x)$ in the training set. In other words, The adversarial training forces G to generate images that reside on the natural images manifold.

3.2. Stacked Generative Adversarial Networks

Pre-trained Encoder. We first consider a bottom-up DNN pre-trained for classification, which is referred to as the encoder E throughout. We define a stack of bottom-up deterministic nonlinear mappings: $h_{i+1} = E_i(h_i)$, where $i \in \{0, 1, ..., N-1\}$, E_i consists of a sequence of neural layers (*e.g.*, convolution, pooling), N is the number of hierarchies (stacks), $h_i (i \neq 0, N)$ are intermediate representations, $h_N = y$ is the classification result, and $h_0 = x$ is the input image. Note that in our formulation, each E_i can contain multiple layers and the way of grouping layers together into E_i is determined by us. The number of stacks N is therefore less than the number of layers in E and is also determined by us.

Stacked Generators. Provided with a pre-trained encoder E, our goal is to train a top-down generator G that inverts E. Specifically, G consists of a top-down *stack* of generators G_i, each trained to invert a bottom-up mapping E_i. Each G_i takes in a higher-level feature and a noise vector as inputs, and outputs the lower-level feature $\hat{h}_i$. We first train each GAN independently and then train them jointly in an end-to-end manner, as shown in Fig. 1. Each gener-

ator receives conditional input from encoders in the independent training stage, and from the upper generators in the joint training stage. In other words, $\hat{h}_i = G_i(h_{i+1}, z_i)$ during independent training and $\hat{h}_i = G_i(\hat{h}_{i+1}, z_i)$ during joint training. The loss equations shown in this section are for independent training stage but can be easily modified to joint training by replacing h_{i+1} with $\hat{h}_{i+1}$.

Intuitively, the total variations of images could be decomposed into multiple levels, with higher-level semantic variations (*e.g.*, attributes, object categories, rough shapes) and lower-level variations (*e.g.*, detailed contours and textures, background clutters). Our model allows using different noise variables to represent different levels of variations.

The training procedure is shown in Fig. 1 (b). Each generator G_i is trained with a linear combination of three loss terms: adversarial loss, conditional loss, and entropy loss.

$$\mathcal{L}_{G_i} = \lambda_1 \mathcal{L}_{G_i}^{adv} + \lambda_2 \mathcal{L}_{G_i}^{cond} + \lambda_3 \mathcal{L}_{G_i}^{ent}, \tag{3}$$

where $\mathcal{L}_{G_i}^{adv}$, $\mathcal{L}_{G_i}^{cond}$, $\mathcal{L}_{G_i}^{ent}$ denote adversarial loss, conditional loss, and entropy loss respectively. λ_1, λ_2, λ_3 are the weights associated with different loss terms. In practice, we find it sufficient to set the weights such that the magnitude of different terms are of similar scales. In this subsection we first introduce the adversarial loss $\mathcal{L}_{G_i}^{adv}$. We will then introduce $\mathcal{L}_{G_i}^{cond}$ and $\mathcal{L}_{G_i}^{ent}$ in Sec. 3.3 and 3.4 respectively.

For each generator G_i, we introduce a *representation discriminator* D_i that distinguishes generated representations $\hat{h}_i$, from "real" representations h_i. Specifically, the discriminator D_i is trained with the loss function:

$$\begin{aligned}\mathcal{L}_{D_i} = &\mathbb{E}_{h_i \sim P_{data,E}}[-\log D_i(h_i)] + \\ &\mathbb{E}_{z_i \sim P_{z_i},\, h_{i+1} \sim P_{data,E}}[-\log\left(1 - D_i(G_i(h_{i+1}, z_i))\right)]\end{aligned} \tag{4}$$

And G_i is trained to "fool" the representation discriminator D_i, with the adversarial loss defined by:

$$\mathcal{L}_{G_i}^{adv} = \mathbb{E}_{h_{i+1} \sim P_{data,E},\, z_i \sim P_{z_i}}[-\log(D_i(G_i(h_{i+1}, z_i)))] \tag{5}$$

During joint training, the adversarial loss provided by representational discriminators can also be regarded as a type of deep supervision [35], providing intermediate supervision signals. In our current formulation, E is a discriminative model, and G is a generative model conditioned on labels. However, it is also possible to train SGAN without using label information: E can be trained with an unsupervised objective and G can be cast into an unconditional generative model by removing the label input from the top generator. We leave this for future exploration.

Sampling. To sample images, all G_is are stacked together in a top-down manner, as shown in Fig. 1 (c). Our SGAN is capable of modeling the data distribution conditioned on the class label: $p_G(\hat{x}|y) = p_G(\hat{h}_0|\hat{h}_N) \propto p_G(\hat{h}_0, \hat{h}_1, ..., \hat{h}_{N-1}|\hat{h}_N) = \prod_{0 \le i \le N-1} p_{G_i}(\hat{h}_i|\hat{h}_{i+1})$, where each $p_{G_i}(\hat{h}_i|\hat{h}_{i+1})$ is modeled by a generator G_i. From an information-theoretic perspective, SGAN factorizes the total entropy of the image distribution $H(x)$ into multiple (smaller) conditional entropy terms: $H(x) = H(h_0, h_1, ..., h_N) = \sum_{i=0}^{N-1} H(h_i|h_{i+1}) + H(y)$, thereby decomposing one difficult task into multiple easier tasks.

3.3. Conditional Loss

At each stack, a generator G_i is trained to capture the distribution of lower-level representations $\hat{h}_i$, conditioned on higher-level representations h_{i+1}. However, in the above formulation, the generator might choose to ignore h_{i+1}, and generate plausible $\hat{h}_i$ from scratch. Some previous works [40, 15, 5] tackle this problem by feeding the conditional information to both the generator and discriminator. This approach, however, might introduce unnecessary complexity to the discriminator and increase model instability [46, 54].

Here we adopt a different approach: we regularize the generator by adding a loss term $\mathcal{L}_{G_i}^{cond}$ named *conditional loss*. We feed the generated lower-level representations $\hat{h}_i = G_i(h_{i+1}, z_i)$ back to the encoder E, and compute the recovered higher-level representations. We then enforce the recovered representations to be similar to the conditional representations. Formally:

$$\mathcal{L}_{G_i}^{cond} = \mathbb{E}_{h_{i+1} \sim P_{data,E},\, z_i \sim P_{z_i}}[f(E_i(G_i(h_{i+1}, z_i)), h_{i+1})] \tag{6}$$

where f is a distance measure. We define f to be the Euclidean distance for intermediate representations and cross-entropy for labels. Our conditional loss $\mathcal{L}_{G_i}^{cond}$ is similar to the "feature loss" used by [7] and the "FCN loss" in [62].

3.4. Entropy Loss

Simply adding the conditional loss $\mathcal{L}_{G_i}^{cond}$ leads to another issue: the generator G_i learns to ignore the noise z_i, and compute $\hat{h}_i$ deterministically from h_{i+1}. This problem has been encountered in various applications of conditional GANs, *e.g.*, synthesizing future frames conditioned on previous frames [39], generating images conditioned on label maps [25], and most related to our work, synthesizing images conditioned on feature representations [7]. All the above works attempted to generate *diverse* images/videos by feeding noise to the generator, but failed because the conditional generator simply ignores the noise. To our knowledge, there is still no principled way to deal with this issue. It might be tempting to think that *minibatch discrimination* [53], which encourages sample diversity in each minibatch, could solve this problem. However, even if the generator generates $\hat{h}_i$ deterministically from h_{i+1}, the generated samples in each minibatch are still diverse since generators are conditioned on different h_{i+1}. Thus, there is no ob-

vious way minibatch discrimination could penalize a collapsed conditional generator.

Variational Conditional Entropy Maximization. To tackle this problem, we would like to encourage the generated representation $\hat{h}_i$ to be sufficiently diverse when conditioned on h_{i+1}, *i.e.*, the conditional entropy $H(\hat{h}_i|h_{i+1})$ should be as high as possible. Since directly maximizing $H(\hat{h}_i|h_{i+1})$ is intractable, we propose to maximize instead a *variational lower bound* on the conditional entropy. Specifically, we use an auxiliary distribution $Q_i(z_i|\hat{h}_i)$ to approximate the true posterior $P_i(z_i|\hat{h}_i)$, and augment the training objective with a loss term named *entropy loss*:

$$\mathcal{L}_{G_i}^{ent} = \mathbb{E}_{z_i \sim P_{z_i}}[\mathbb{E}_{\hat{h}_i \sim G_i(\hat{h}_i|z_i)}[-\log Q_i(z_i|\hat{h}_i)]] \quad (7)$$

Below we give a proof that minimizing $\mathcal{L}_{G_i}^{ent}$ is equivalent to maximizing a variational lower bound for $H(\hat{h}_i|h_{i+1})$.

$$\begin{aligned}
H(\hat{h}_i|h_{i+1}) &= H(\hat{h}_i, z_i|h_{i+1}) - H(z_i|\hat{h}_i, h_{i+1}) \\
&\geq H(\hat{h}_i, z_i|h_{i+1}) - H(z_i|\hat{h}_i) \\
&= H(z_i|h_{i+1}) + \underbrace{H(\hat{h}_i|z_i, h_{i+1})}_{0} - H(z_i|\hat{h}_i) \\
&= H(z_i|h_{i+1}) - H(z_i|\hat{h}_i) \\
&= H(z_i) - H(z_i|\hat{h}_i) \\
&= \mathbb{E}_{\hat{h}_i \sim G_i}[\mathbb{E}_{z_i' \sim P_i(z_i'|\hat{h}_i)}[\log P_i(z_i'|\hat{h}_i)]] + H(z_i) \\
&= \mathbb{E}_{\hat{h}_i \sim G_i}[\mathbb{E}_{z_i' \sim P_i(z_i'|\hat{h}_i)}[\log Q_i(z_i'|\hat{h}_i)] \\
&+ \underbrace{KLD(P_i \| Q_i)}_{\geq 0}] + H(z_i) \\
&\geq \mathbb{E}_{\hat{h}_i \sim G_i}[\mathbb{E}_{z_i' \sim P_i(z_i'|\hat{h}_i)}[\log Q_i(z_i'|\hat{h}_i)]] + H(z_i) \\
&= \mathbb{E}_{z_i' \sim P_{z_i'}}[\mathbb{E}_{\hat{h}_i \sim G_i(\hat{h}_i|z_i')}[\log Q_i(z_i'|\hat{h}_i)]] + H(z_i) \\
&\triangleq -\mathcal{L}_{G_i}^{ent} + H(z_i)
\end{aligned} \quad (8)$$

In practice, we parameterize Q_i with a deep network that predicts the posterior distribution of z_i given $\hat{h}_i$. Q_i shares most of the parameters with D_i. We treat the posterior as a diagonal Gaussian with fixed standard deviations, and use the network Q_i to only predict the posterior mean, making $\mathcal{L}_{G_i}^{ent}$ equivalent to the Euclidean reconstruction error. In each iteration we update both G_i and Q_i to minimize $\mathcal{L}_{G_i}^{ent}$.

Our method is similar to the variational mutual information maximization technique proposed by Chen *et al*. [2]. A key difference is that [2] uses the Q-network to predict only a small set of deliberately constructed "latent code", while our Q_i tries to predict *all* the noise variables z_i in each stack. The loss used in [2] therefore maximizes the *mutual information* between the output and the latent code, while ours maximizes the *entropy* of the output $\hat{h}_i$, conditioned on h_{i+1}. [6, 10] also train a separate network to map images back to latent space to perform unsupervised feature learning. Independent of our work, [4] proposes to regularize EBGAN [68] with entropy maximization in order to prevent the discriminator from degenerating to uniform prediction. Our entropy loss is motivated from generating multiple possible outputs from the same conditional input.

4. Experiments

In this section, we perform experiments on a variety of datasets including MNIST [34], SVHN [41], and CIFAR-10 [29]. Code and pre-trained models are available at: https://github.com/xunhuang1995/SGAN. Readers may refer to our code repository for more details about experimental setup, hyper-parameters, *etc*.

Encoder: For all datasets we use a small CNN with two convolutional layers as our encoder: `conv1-pool1-conv2-pool2-fc3-fc4`, where `fc3` is a fully connected layer and `fc4` outputs classification scores before softmax. On CIFAR-10 we apply horizontal flipping to train the encoder. No data augmentation is used on other datasets.

Generator: We use generators with two stacks throughout our experiments. Note that our framework is generally applicable to the setting with multiple stacks, and we hypothesize that using more stacks would be helpful for large-scale and high-resolution datasets. For all datasets, our top GAN G_1 generates `fc3` features from some random noise z_1, conditioned on label y. The bottom GAN G_0 generates images from some noise z_0, conditioned on `fc3` features generated from GAN G_1. We set the loss coefficient parameters $\lambda_1 = \lambda_2 = 1$ and $\lambda_3 = 10$.[1]

4.1. Datasets

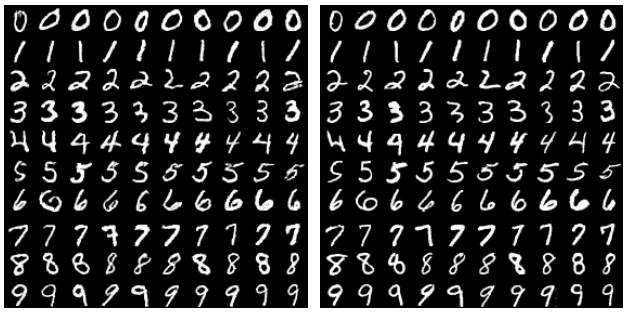
We thoroughly evaluate SGAN on three widely adopted datasets: MNIST [34], SVHN [41], and CIFAR-10 [29]. The details of each dataset is described in the following.

MNIST: The MNIST dataset contains $70,000$ labeled images of hand-written digits with $60,000$ in the training set and $10,000$ in the test set. Each image is sized by 28×28.

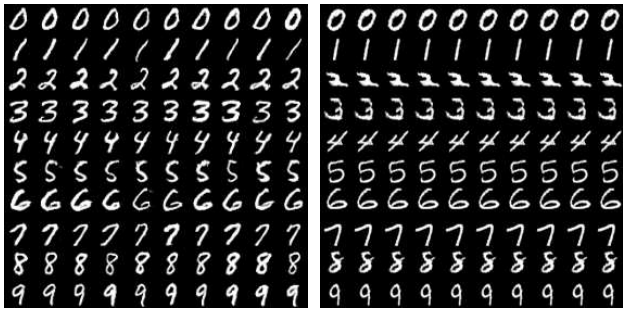
SVHN: The SVHN dataset is composed of real-world color images of house numbers collected by Google Street View [41]. Each image is of size 32×32 and the task is to classify the digit at the center of the image. The dataset contains $73,257$ training images and $26,032$ test images.

CIFAR-10: The CIFAR-10 dataset consists of colored natural scene images sized at 32×32 pixels. There are 50,000 training images and 10,000 test images in 10 classes.

[1] The choice of the parameters are made so that the magnitude of each loss term is of the same scale.



(a) SGAN samples (conditioned on labels) (b) Real images (nearest neighbor)



(c) SGAN samples (conditioned on generated `fc3` features) (d) SGAN samples (conditioned on generated `fc3` features, trained *without* entropy loss)

Figure 2: **MNIST results.** (a) Samples generated by SGAN when conditioned on class labels. (b) Corresponding nearest neighbor images in the training set. (c) Samples generated by the bottom GAN when conditioned on a fixed `fc3` feature activation, generated by the top GAN. (d) Same as (c), but the bottom GAN is trained without entropy loss.

(a) SGAN samples (conditioned on labels) (b) Real images (nearest neighbor)

(c) SGAN samples (conditioned on generated `fc3` features) (d) SGAN samples (conditioned on generated `fc3` features, trained *without* entropy loss)

Figure 3: **SVHN results.** (a) Samples generated by SGAN when conditioned on class labels. (b) Corresponding nearest neighbor images in the training set. (c) Samples generated by the bottom GAN when conditioned on a fixed `fc3` feature activation, generated by the top GAN. (d) Same as (c), but the bottom GAN is trained without entropy loss.

4.2. Samples

In Fig. 2 (a), we show MNIST samples generated by SGAN. Each row corresponds to samples conditioned on a given digit class label. SGAN is able to generate diverse images with different characteristics. The samples are visually indistinguishable from real MNIST images shown in Fig. 2 (b), but still have differences compared with corresponding nearest neighbor training images.

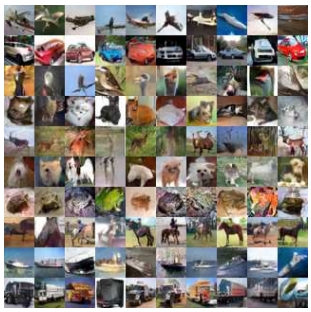
We further examine the effect of entropy loss. In Fig. 2 (c) we show the samples generated by bottom GAN when conditioned on a fixed `fc3` feature generated by the top GAN. The samples (per row) have sufficient low-level variations, which reassures that bottom GAN learns to generate images without ignoring the noise z_0. In contrast, in Fig. 2 (d), we show samples generated without using entropy loss for bottom generator, where we observe that the bottom GAN ignores the noise and instead deterministically generates images from `fc3` features.

An advantage of SGAN compared with a vanilla GAN is its interpretability: it decomposes the total variations of an image into different levels. For example, in MNIST it decomposes the variations into y that represents the high-level digit label, z_1 that captures the mid-level coarse pose of the digit and z_0 that represents the low-level spatial details.

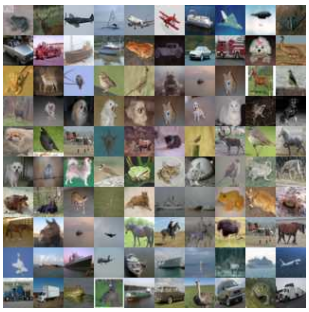
The samples generated on SVHN and CIFAR-10 datasets can be seen in Fig. 3 and Fig. 4, respectively. Provided with the same `fc3` feature, we see in each row of panel (c) that SGAN is able to generate samples with similar coarse outline but different lighting conditions and background clutters. Also, the nearest neighbor images in the training set indicate that SGAN is not simply memorizing training data, but can truly generate novel images.

4.3. Comparison with the state of the art

Here, we compare SGAN with other state-of-the-art generative models on CIFAR-10 dataset. The visual quality of generated images is measured by the widely used metric, Inception score [53]. Following [53], we sample $50,000$ images from our model and use the code provided by [53]

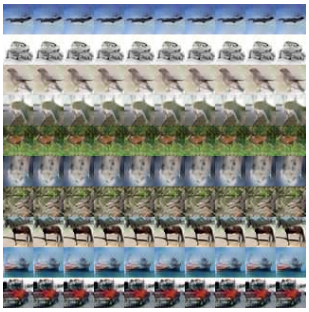


(a) SGAN samples (conditioned on labels)



(b) Real images (nearest neighbor)

(c) SGAN samples (conditioned on generated `fc3` features)



(d) SGAN samples (conditioned on generated `fc3` features, trained *without* entropy loss)

Figure 4: **MNIST results.** (a) Samples generated by SGAN when conditioned on class labels. (b) Corresponding nearest neighbor images in the training set. (c) Samples generated by the bottom GAN when conditioned on a fixed `fc3` feature activation, generated by the top GAN. (d) Same as (c), but the bottom GAN is trained without entropy loss.

to compute the score. As shown in Tab. 1, SGAN obtains a score of 8.59 ± 0.12, outperforming AC-GAN [43] (8.25 ± 0.07) and Improved GAN [53] (8.09 ± 0.07). Also, note that the 5 techniques introduced in [53] are not used in our implementations. Incorporating these techniques might further boost the performance of our model.

4.4. Visual Turing test

To further verify the effectiveness of SGAN, we conduct human visual Turing test in which we ask AMT workers to distinguish between real images and images generated by our networks. We exactly follow the interface used in Improved GAN [53], in which the workers are given 9 images at each time and can receive feedback about whether their answers are correct. With $9,000$ votes for each evaluated model, our AMT workers got 24.4% error rate for samples from SGAN and 15.6% for samples from DCGAN ($\mathcal{L}^{adv} + L^{cond} + \mathcal{L}^{ent}$). This further confirms that our stacked design can significantly improve the image quality over GAN without stacking.

Method	Score
Infusion training [1]	4.62 ± 0.06
ALI [10] (as reported in [63])	5.34 ± 0.05
GMAN [11] (best variant)	6.00 ± 0.19
EGAN-Ent-VI [4]	7.07 ± 0.10
LR-GAN [65]	7.17 ± 0.07
Denoising feature matching [63]	7.72 ± 0.13
DCGAN† (with labels, as reported in [61])	6.58
SteinGAN† [61]	6.35
Improved GAN† [53] (best variant)	8.09 ± 0.07
AC-GAN† [43]	8.25 ± 0.07
DCGAN ($\mathcal{L}^{adv}$)	6.16 ± 0.07
DCGAN ($\mathcal{L}^{adv} + \mathcal{L}^{ent}$)	5.40 ± 0.16
DCGAN ($\mathcal{L}^{adv} + \mathcal{L}^{cond}$)†	5.40 ± 0.08
DCGAN ($\mathcal{L}^{adv} + L^{cond} + \mathcal{L}^{ent}$)†	7.16 ± 0.10
SGAN-no-joint†	**8.37** ± 0.08
SGAN†	**8.59** ± 0.12
Real data	11.24 ± 0.12

† Trained with labels.

Table 1: **Inception Score on CIFAR-10.** SGAN and SGAN-no-joint outperform previous state-of-the-art approaches.

4.5. More ablation studies

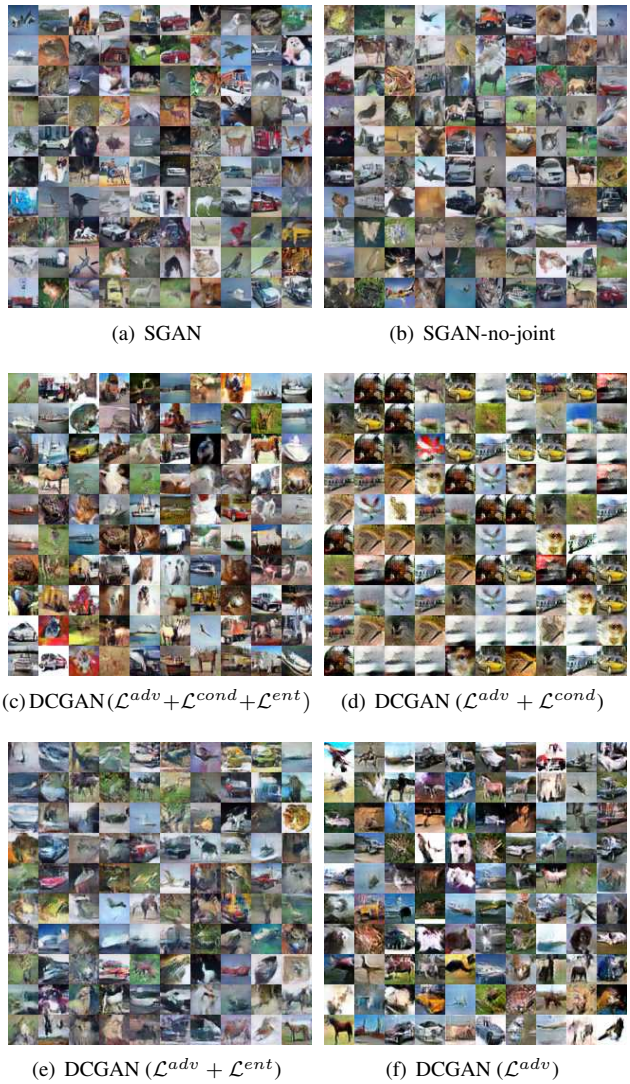
In Sec. 4.2 we have examined the effect of entropy loss. In order to further understand the effect of different model components, we conduct extensive ablation studies by evaluating several baseline methods on CIFAR-10 dataset. If not mentioned otherwise, all models below use the same training hyper-parameters as the full SGAN model.

(a) SGAN: The full model, as described in Sec. 3.

(b) SGAN-no-joint: Same architecture as (a), but each GAN is trained *independently*, and there is no final joint training stage.

(c) DCGAN ($\mathcal{L}^{adv} + \mathcal{L}^{cond} + \mathcal{L}^{ent}$): This is a *single* GAN model with the same architecture as the bottom GAN in SGAN, except that the generator is conditioned on labels instead of `fc3` features. Note that other techniques proposed in this paper, including conditional loss $\mathcal{L}^{cond}$ and entropy loss $\mathcal{L}^{ent}$, are still employed. We also tried to use the full generator G in SGAN as the baseline, instead of only the bottom generator G_0. However, we failed to make it converge, possibly because G is too deep to be trained without intermediate supervision from representation discriminators.

(d) DCGAN ($\mathcal{L}^{adv} + \mathcal{L}^{cond}$): Same architecture as (c), but trained without entropy loss $\mathcal{L}^{ent}$.



(a) SGAN (b) SGAN-no-joint

(c) DCGAN ($\mathcal{L}^{adv}+\mathcal{L}^{cond}+\mathcal{L}^{ent}$) (d) DCGAN ($\mathcal{L}^{adv}+\mathcal{L}^{cond}$)

(e) DCGAN ($\mathcal{L}^{adv}+\mathcal{L}^{ent}$) (f) DCGAN ($\mathcal{L}^{adv}$)

Figure 5: **Ablation studies on CIFAR-10.** Samples from (a) full SGAN (b) SGAN without joint training. (c) DCGAN trained with $\mathcal{L}^{adv}+\mathcal{L}^{cond}+\mathcal{L}^{ent}$ (d) DCGAN trained with $\mathcal{L}^{adv}+\mathcal{L}^{cond}$ (e) DCGAN trained with $\mathcal{L}^{adv}+\mathcal{L}^{ent}$ (f) DCGAN trained with $\mathcal{L}^{adv}$.

(e) DCGAN ($\mathcal{L}^{adv}+\mathcal{L}^{ent}$): Same architecture as (c), but trained without conditional loss $\mathcal{L}^{cond}$. This model therefore does not use label information.

(f) DCGAN ($\mathcal{L}^{adv}$): Same architecture as (c), but trained with neither conditional loss $\mathcal{L}^{cond}$ nor entropy loss $\mathcal{L}^{ent}$. This model also does not use label information. It can be viewed as a plain unconditional DCGAN model [47] and serves as the ultimate baseline.

We compare the generated samples (Fig. 5) and Inception scores (Tab. 1) of the baseline methods. Below we summarize some of our results:

1) SGAN obtains slightly higher Inception score than SGAN-no-joint. Yet SGAN-no-joint also generates very high quality samples and outperforms all previous methods in terms of Inception scores.

2) SGAN, either with or without joint training, achieves significantly higher Inception score and better sample quality than the baseline DCGANs. This demonstrates the effectiveness of the proposed stacked approach.

3) As shown in Fig. 5 (d), DCGAN ($\mathcal{L}^{adv}+\mathcal{L}^{cond}$) collapses to generating a single image per category, while adding the entropy loss enables it to generate diverse images (Fig. 5 (c)). This further demonstrates that entropy loss is effective at improving output diversity.

4) The single DCGAN ($\mathcal{L}^{adv}+\mathcal{L}^{cond}+\mathcal{L}^{ent}$) model obtains higher Inception score than the conditional DCGAN reported in [61]. This suggests that $\mathcal{L}^{cond}+\mathcal{L}^{ent}$ might offer some advantages compared to a plain conditional DCGAN, even without stacking.

5) In general, Inception score [53] correlates well with visual quality of images. However, it seems to be insensitive to diversity issues . For example, it gives the same score to Fig. 5 (d) and (e) while (d) has clearly collapsed. This is consistent with results in [43, 61].

5. Discussion and Future Work

This paper introduces a top-down generative framework named SGAN, which effectively leverages the representational information from a pre-trained discriminative network. Our approach decomposes the hard problem of estimating image distribution into multiple relatively easier tasks – each generating plausible representations conditioned on higher-level representations. The key idea is to use representation discriminators at different training hierarchies to provide intermediate supervision. We also propose a novel entropy loss to tackle the problem that conditional GANs tend to ignore the noise. Our entropy loss could be employed in other applications of conditional GANs, *e.g.*, synthesizing *different* future frames given the same past frames [39], or generating a *diverse* set of images conditioned on the same label map [25]. We believe this is an interesting research direction in the future.

Acknowledgments

We would like to thank Danlu Chen for the help with Fig. 1. Also, we want to thank Danlu Chen, Shuai Tang, Saining Xie, Zhuowen Tu, Felix Wu and Kilian Weinberger for helpful discussions. Yixuan Li is supported by US Army Research Office W911NF-14-1-0477. Serge Belongie is supported in part by a Google Focused Research Award.

References

[1] F. Bordes, S. Honari, and P. Vincent. Learning to generate samples from noise through infusion training. In *ICLR*, 2017. 7

[2] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *NIPS*, 2016. 5

[3] X. Chen, Y. Sun, B. Athiwaratkun, C. Cardie, and K. Weinberger. Adversarial deep averaging networks for cross-lingual sentiment classification. *arXiv preprint arXiv:1606.01614*, 2016.

[4] Z. Dai, A. Almahairi, P. Bachman, E. Hovy, and A. Courville. Calibrating energy-based generative adversarial networks. In *ICLR*, 2017. 5, 7

[5] E. L. Denton, S. Chintala, R. Fergus, et al. Deep generative image models using a laplacian pyramid of adversarial networks. In *NIPS*, 2015. 2, 4

[6] J. Donahue, P. Krähenbühl, and T. Darrell. Adversarial feature learning. In *ICLR*, 2017. 5

[7] A. Dosovitskiy and T. Brox. Generating images with perceptual similarity metrics based on deep networks. In *NIPS*, 2016. 2, 4

[8] A. Dosovitskiy and T. Brox. Inverting visual representations with convolutional networks. In *CVPR*, 2016. 2

[9] A. Dosovitskiy, J. Tobias Springenberg, and T. Brox. Learning to generate chairs with convolutional neural networks. In *CVPR*, 2015. 1

[10] V. Dumoulin, I. Belghazi, B. Poole, A. Lamb, M. Arjovsky, O. Mastropietro, and A. Courville. Adversarially learned inference. In *ICLR*, 2017. 5, 7

[11] I. Durugkar, I. Gemp, and S. Mahadevan. Generative multi-adversarial networks. In *ICLR*, 2017. 7

[12] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky. Domain-adversarial training of neural networks. *JMLR*, 2016.

[13] L. Gatys, A. S. Ecker, and M. Bethge. Texture synthesis using convolutional neural networks. In *NIPS*, 2015.

[14] L. A. Gatys, A. S. Ecker, and M. Bethge. Image style transfer using convolutional neural networks. In *CVPR*, 2016.

[15] J. Gauthier. Conditional generative adversarial nets for convolutional face generation. *Class Project for Stanford CS231N: Convolutional Neural Networks for Visual Recognition, Winter semester*, 2014, 2014. 4

[16] M. Germain, K. Gregor, I. Murray, and H. Larochelle. Made: masked autoencoder for distribution estimation. In *ICML*, 2015. 2

[17] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *NIPS*, 2014. 1, 2, 3

[18] K. Gregor, I. Danihelka, A. Graves, D. Rezende, and D. Wierstra. Draw: A recurrent neural network for image generation. In *ICML*, 2015. 2

[19] K. Gregor, I. Danihelka, A. Mnih, C. Blundell, and D. Wierstra. Deep autoregressive networks. In *ICML*, 2014. 2

[20] S. Gupta, J. Hoffman, and J. Malik. Cross modal distillation for supervision transfer. In *CVPR*, 2016. 1, 2

[21] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 1

[22] G. E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800, 2002. 2

[23] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006. 2

[24] J. Hoffman, D. Wang, F. Yu, and T. Darrell. Fcns in the wild: Pixel-level adversarial and constraint-based adaptation. *arxiv*, 2016.

[25] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. *arxiv*, 2016. 4, 8

[26] J. Johnson, A. Alahi, and L. Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *ECCV*, 2016. 2

[27] D. P. Kingma, S. Mohamed, D. J. Rezende, and M. Welling. Semi-supervised learning with deep generative models. In *NIPS*, 2014. 2

[28] D. P. Kingma and M. Welling. Auto-encoding variational bayes. In *ICLR*, 2014. 1, 2

[29] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. *technical report*, 2009. 5

[30] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 1

[31] A. Lamb, V. Dumoulin, and A. Courville. Discriminative regularization for generative models. In *ICML*, 2016. 2

[32] H. Larochelle and I. Murray. The neural autoregressive distribution estimator. In *AISTATS*, 2011. 2

[33] A. B. L. Larsen, S. K. Sønderby, and O. Winther. Autoencoding beyond pixels using a learned similarity metric. In *ICML*, 2016. 2

[34] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 5

[35] C.-Y. Lee, S. Xie, P. Gallagher, Z. Zhang, and Z. Tu. Deeply-supervised nets. In *AISTATS*, 2015. 4

[36] Y. Li, K. Swersky, and R. Zemel. Generative moment matching networks. In *ICML*, 2015. 1

[37] A. Mahendran and A. Vedaldi. Visualizing deep convolutional neural networks using natural pre-images. *IJCV*, pages 1–23, 2016. 2

[38] A. Makhzani, J. Shlens, N. Jaitly, and I. Goodfellow. Adversarial autoencoders. In *NIPS*, 2016. 1

[39] M. Mathieu, C. Couprie, and Y. LeCun. Deep multi-scale video prediction beyond mean square error. In *ICLR*, 2016. 4, 8

[40] M. Mirza and S. Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014. 4

[41] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. 2011. 5

[42] A. Nguyen, J. Yosinski, Y. Bengio, A. Dosovitskiy, and J. Clune. Plug & play generative networks: Conditional iterative generation of images in latent space. In *CVPR*, 2017. 2

[43] A. Odena, C. Olah, and J. Shlens. Conditional image synthesis with auxiliary classifier gans. *arXiv preprint arXiv:1610.09585*, 2016. 7, 8

[44] A. v. d. Oord, N. Kalchbrenner, and K. Kavukcuoglu. Pixel recurrent neural networks. In *ICML*, 2016. 1, 2

[45] A. v. d. Oord, N. Kalchbrenner, O. Vinyals, L. Espeholt, A. Graves, and K. Kavukcuoglu. Conditional image generation with pixelcnn decoders. In *NIPS*, 2016. 2

[46] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros. Context encoders: Feature learning by inpainting. In *CVPR*, 2016. 4

[47] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *ICLR*, 2016. 2, 8

[48] A. Rasmus, M. Berglund, M. Honkala, H. Valpola, and T. Raiko. Semi-supervised learning with ladder networks. In *NIPS*, 2015. 2

[49] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee. Generative adversarial text to image synthesis. In *ICML*, 2016. 2

[50] D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *ICML*, 2014. 2

[51] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio. Fitnets: Hints for thin deep nets. In *ICLR*, 2015. 2

[52] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, 115(3):211–252, 2015. 1

[53] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training gans. In *NIPS*, 2016. 2, 4, 6, 7, 8

[54] P. Sangkloy, J. Lu, C. Fang, F. Yu, and J. Hays. Scribbler: Controlling deep image synthesis with sketch and color. In *CVPR*, 2017. 4

[55] A. Sharif Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. In *CVPR*, 2014. 1

[56] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 1

[57] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *CVPR*, 2015. 1

[58] L. Theis and M. Bethge. Generative image modeling using spatial lstms. In *NIPS*, 2015. 2

[59] D. Ulyanov, V. Lebedev, A. Vedaldi, and V. Lempitsky. Texture networks: Feed-forward synthesis of textures and stylized images. In *ICML*, 2016. 2

[60] H. Valpola. From neural pca to deep unsupervised learning. *Adv. in Independent Component Analysis and Learning Machines*, pages 143–171, 2015. 2

[61] D. Wang and Q. Liu. Learning to draw samples: With application to amortized mle for generative adversarial learning. *arXiv preprint arXiv:1611.01722*, 2016. 7, 8

[62] X. Wang and A. Gupta. Generative image modeling using style and structure adversarial networks. In *ECCV*, 2016. 2, 4

[63] D. Warde-Farley and Y. Bengio. Improving generative adversarial networks with denoising feature matching. In *ICLR*, 2017. 7

[64] X. Yan, J. Yang, K. Sohn, and H. Lee. Attribute2image: Conditional image generation from visual attributes. In *ECCV*, 2016. 2

[65] J. Yang, A. Kannan, D. Batra, and D. Parikh. Lr-gan: Layered recursive generative adversarial networks for image generation. In *ICLR*, 2017. 7

[66] Y. Zhang, K. Lee, and H. Lee. Augmenting supervised neural networks with unsupervised objectives for large-scale image classification. In *ICML*, 2016. 2

[67] J. Zhao, M. Mathieu, R. Goroshin, and Y. Lecun. Stacked what-where auto-encoders. *ICLR Workshop*, 2016. 2

[68] J. Zhao, M. Mathieu, and Y. LeCun. Energy-based generative adversarial network. In *ICLR*, 2017. 1, 5