Wasserstein Introspective Neural Networks

Kwonjoon Lee Weijian Xu Fan Fan Zhuowen Tu
University of California San Diego
{kwl042, wex041, f1fan, ztu}@ucsd.edu

Abstract

We present Wasserstein introspective neural networks (WINN) that are both a generator and a discriminator within a single model. WINN provides a significant improvement over the recent introspective neural networks (INN) method by enhancing INN's generative modeling capability. WINN has three interesting properties: (1) A mathematical connection between the formulation of the INN algorithm and that of Wasserstein generative adversarial networks (WGAN) is made. (2) The explicit adoption of the Wasserstein distance into INN results in a large enhancement to INN, achieving compelling results even with a single classifier — e.g., providing nearly a 20 times reduction in model size over INN for unsupervised generative modeling. (3) When applied to supervised classification, WINN also gives rise to improved robustness against adversarial examples in terms of the error reduction. In the experiments, we report encouraging results on unsupervised learning problems including texture, face, and object modeling, as well as a supervised classification task against adversarial attacks. Our code is available online[1].

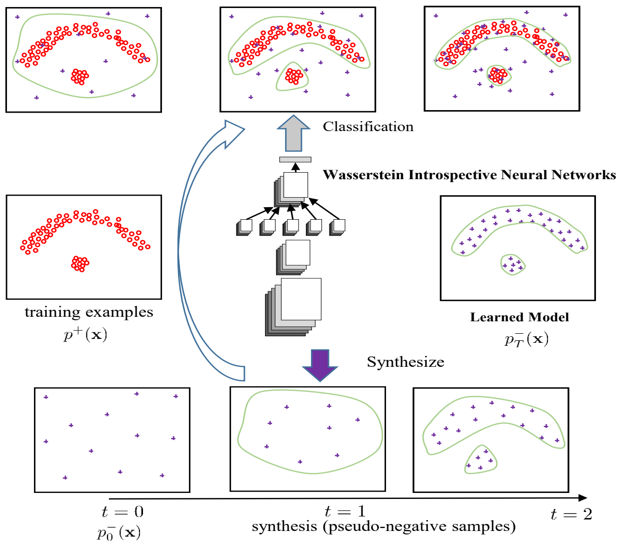


Figure 1: Schematic illustration of Wasserstein introspective neural networks for unsupervised learning. The left figure shows the input examples; the bottom figures show the pseudo-negatives (purple crosses) being progressively synthesized; the top figures show the classification between the given examples (positives) and synthesized pseudo-negatives (negatives). The right figure shows the model learned to approach the target distribution based on the given data.

1. Introduction

Performance within the task of supervised image classification has been vastly improved in the era of deep learning using modern convolutional neural network (CNN) [30] based discriminative classifiers [26, 43, 31, 42, 16, 52, 18]. On the other hand, unsupervised generative models in deep learning were previously attained using methods under the umbrella of graphical models — e.g., the Boltzmann machine [17] or autoencoder [4, 24] architectures. However, the rich representational power seen within convolution-based (discriminative) models is not being directly enjoyed in these generative models. Later, inverting convolutional neural networks in order to convert internal representations into a real image was investigated in [6, 27]. Recently, generative adversarial networks (GAN) [11] and followup works [38, 2, 14] have attracted a tremendous amount of attention in machine learning and computer vision by producing high quality synthesized images by training a pair of competing models against one another in an adversarial manner. While a generator tries to create "fake" images to fool the discriminator, the discriminator attempts to discern between these "real" (given training) and "fake" images. After convergence, the generator is able to produce images faithful to the underlying data distribution.

Before the deep learning era [17], generative modeling had been an area with a steady pace of development [1, 5, 54, 36, 49, 53]. These models were guided by rigorous statistical theories which, although nice in theory, did not succeed in producing synthesized images with practical quality.

In terms of building generative models from discriminative classifiers, there have been early attempts in [48, 44].

[1] https://github.com/kjunelee/WINN

In [48], a generative model was obtained from a repeatedly trained boosting algorithm [8] using a weak classifier whereas [44] used a strong classifier in order to self-generate negative examples or "pseudo-negatives".

To address the lack of richness in representation and efficiency in synthesis, convolutional neural networks were adopted in introspective neural networks (INN) [29, 21] to build a single model that is simultaneously generative and discriminative. The generative modeling aspect was studied in [29] where a sequence of CNN classifiers $(10-60)$ were trained, while the power within the classification setting was revealed in [21] in the form of introspective convolutional networks (ICN) that used only a single CNN classifier. Although INN models [29, 21] point to a promising direction to obtain a single model being both a good generator and a strong discriminative classifier, a sequence of CNNs were needed to generate realistic synthesis. As a result, this requirement may serve as a possible bottleneck with respect to training complexity and model size.

Recently, a generic formulation [2] was developed within the GAN model family to incorporate a Wasserstein objective to alleviate the well-known difficulty in GAN training. Motivated by introspective neural networks (INN) [29] and this Wasserstein objective [2], we propose to adopt the Wasserstein term into the INN formulation to enhance the modeling capability. The resulting model, Wasserstein introspective neural networks (WINN) shows greatly enhanced modeling capability over INN by having $20\times$ reduction in the number of CNN classifiers.

2. Significance and Related Work

We make several interesting observations for WINN:

- A mathematical connection between the WGAN formulation [2] and the INN algorithm [29] is made to better understand the overall objective function within INN.
- By adopting the Wasserstein distance into INN, we are able to generate images using a single CNN in WINN with even higher quality than those by INN that uses 20 CNNs (as seen in Figure 2, 4, 5, 6, and 7; the similar underlying CNN architectures are used in WINN and INN). WINN achieves a significant reduction in model complexity over INN, making the generator more practical.
- Within texture modeling, INN and WINN are able to inherently model the input image space, making the synthesis of large texture images realistic, whereas GAN projects a noise vector onto the image space making the image patch stitching more difficult (although extensions exist), as demonstrated in Figure 2.
- To compare with the family of GAN models, we compute Inception scores using the standard procedure on the CIFAR-10 datasets and observed modest results. Here, we typically train 4-5 cascades to boost the numbers but WINN with one CNN is already promising. Overall, modern GAN variants (e.g., [14]) still outperform our WINN with better quality images. Some results are shown in Figure 7.
- To test the robustness of the discriminative abilities of WINN, we directly make WINN into a discriminative classifier by training it on the standard MNIST and SVHN datasets. Not only are we able to improve over the previous ICN [21] classifier for supervised classification, we also observe a large improvement in robustness against adversarial examples compared with the baseline CNN, ResNet, and the competing ICN.

In terms of other related work, we briefly discuss some existing methods below.

Wasserstein GAN. A closely related work to our WINN algorithm is the Wasserstein generative adversarial networks (WGAN) method [2, 14]. While WINN adopts the Wasserstein distance as motivated by WGAN, our overall algorithm is still within the family of introspective neural networks (INN) [29, 21]. WGAN on the other hand is a variant of GAN with an improvement over GAN by having an objective that is easier to train. The level of difference between WINN and WGAN is similar to that between INN [29, 21] and GAN [12]. The overall comparisons between INN and GAN have been described in [29, 21].

Generative ConvNets. Recently, there has also been a cluster of algorithms developed in [50, 51, 15] where Langevin dynamics are adopted in generator CNNs. However, the models proposed in [50, 51, 15] do not perform introspection (Figure 1) and their generator and discriminator components are still somewhat separated; thus, their generators are not used as effective discriminative classifiers to perform state-of-the-art classification on standard supervised machine learning tasks. Their training processes are also more complex than those of INN and WINN.

Deep energy models (DEMs) [35]. DEM [35] extends the standard density estimation by using multi-layer neural networks (MLNN) with a rather complex training procedure. The probability model in DEM includes both the raw input and the features computed by MLNN. WINN instead takes a more general and simplistic form and is easier to train (see Eq. (1)). In general, DEM belongs to the minimum description length (MDL) family models in which the maximum likelihood is achieved. WINN, instead, has a formulation being simultaneously discriminative and generative.

3. Introspective Neural Networks

3.1. Brief introduction of INN

We first briefly introduce the introspective neural network method (INNg) [29] for generative modeling and its companion model [21] which focuses on the classification aspect. The main motivation behind the INN work [29, 21] is to make a convolutional neural network classifier simultaneously discriminative and generative. A single CNN classifier is trained in an introspective manner to improve the standard supervised classification result [21], however, a sequence of CNNs (typically $10-60$) is needed to be able to synthesize images of good quality [29].

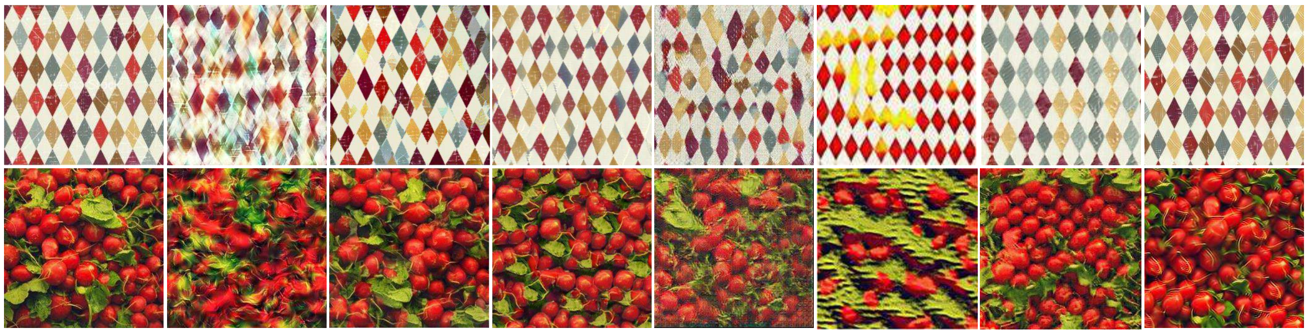


Figure 2: Comparison of texture synthesis algorithms. Gatys *et al.* [9], and TextureNet [45] results are from [45].

Figure 1 shows a brief illustration of a single introspective CNN classifier [21]. We discuss our basic unsupervised formulation next. Suppose we are given a set of training examples: $S = \{\mathbf{x}_i \mid i = 1, \ldots, n\}$ where we assume each $x_i \in \mathbb{R}^m$ — e.g., $m = 4096$ for images of size 64×64. These will constitute positive examples of the patterns/targets we wish to model. The main idea of INN is to define pseudo-negative examples that are to be self-generated by the discriminative classifier itself. We therefore define label y for each example $\mathbf{x}$, $y = +1$ if $\mathbf{x}$ is from the given training set and $y = -1$ if $\mathbf{x}$ is self-generated. Motivated by the generative via discriminative learning (GDL) framework [44], one could try to learn the generative model for the given examples, $p(\mathbf{x}|y = +1)$, by a sequentially learned distribution of the **pseudo-negative** samples, $p_t(\mathbf{x}|y = -1; \mathsf{W}_t)$ which is abbreviated as $p^-_{\mathsf{W}_t}(\mathbf{x})$ where W_t includes all the model parameters learned at step t.

$$p^-_{\mathsf{W}_t}(\mathbf{x}) = \frac{1}{Z_t} \exp\{\mathbf{w}_t^{(1)} \cdot \phi(\mathbf{x}; \mathbf{w}_t^{(0)})\} \cdot p^-_0(\mathbf{x}),\ t = 1, \ldots, T \tag{1}$$

where $Z_t = \int \exp\{\mathbf{w}_t^{(1)} \cdot \phi(\mathbf{x}; \mathbf{w}_t^{(0)})\} \cdot p^-_0(\mathbf{x}) d\mathbf{x}$ and the initial distribution $p^-_0(\mathbf{x})$ such as a Gaussian distribution over the entire space of $\mathbf{x} \in \mathbb{R}^m$. The discriminative classifier is a convolutional neural network (CNN) parameterized by $\mathsf{W}_t = (\mathbf{w}_t^{(0)}, \mathbf{w}_t^{(1)})$ where $\mathbf{w}_t^{(1)}$ denotes the weights of the top layer combining the features through $\phi(\mathbf{x}; \mathbf{w}_t^{(0)})$ (e.g., softmax layer) and $\mathbf{w}_t^{(0)}$ parameterizing the internal representations. The synthesis process through which pseudo-negative samples are generated is carried out by stochastic gradient Langevin dynamics [47] as

$$\Delta\mathbf{x} = \frac{\epsilon}{2} \nabla(\mathbf{w}_t^{(1)} \cdot \phi(\mathbf{x}; \mathbf{w}_t^{(0)})) + \eta$$

where $\eta \sim \mathcal{N}(0, \epsilon)$ is a Gaussian distribution and ϵ is the step size that is annealed in the sampling process. Overall, we desire

$$p^-_{\mathsf{W}_t}(\mathbf{x}) \overset{t=\infty}{\to} p(\mathbf{x}|y = +1), \tag{2}$$

using the iterative reclassification-by-synthesis process [21, 29] guided by Eq. (1).

3.2. Connection to the Wasserstein distance

The overall training process, reclassification-by-synthesis, is carried out iteratively without an explicit objective function. The generative adversarial network (GAN) model [11] instead has an objective function formulated in a minimax fashion with the generator and discriminator competing against each other. The Wasserstein generative adversarial network (WGAN) work [2] improves GAN [11] by replacing the Jensen-Shannon distance with an efficient approximation of the Earth-Mover distance [2]. Also, there has been further generalization of the GAN family models in [32].

Let $p^+(\mathbf{x}) \equiv p(\mathbf{x}|y = +1)$ be the target distribution and $p^-_{\mathsf{W}}(\mathbf{x}) \equiv p(\mathbf{x}|y = -1; \mathsf{W})$ be the pseudo-negative distribution parameterized by W. Next, we show a connection between the INN framework and the WGAN formulation [2], whose objective (rewritten with our notations) can be defined as

$$\min_{\mathsf{W}} \max_{||f||_L \le 1} E_{\mathbf{x} \sim p^+}[f(\mathbf{x})] - E_{\mathbf{x} \sim p^-_{\mathsf{W}}}[f(\mathbf{x})], \tag{3}$$

where $||f||_L \le 1$ denotes the space of 1-Lipschitz functions. To build the connection between Eq. (3) of WGAN and Eq. (1) of INN, we first present the following lemma.

Lemma 1 *Considering $f(\mathbf{x}) = \ln \frac{p^+(\mathbf{x})}{p^-_{\mathsf{W}}(\mathbf{x})}$ and assuming its 1-Lipschitz property, we have a lower bound on the Wasserstein distance by*

$$\begin{aligned} &\max_{||f||_L \le 1} E_{\mathbf{x} \sim p^+}[f(\mathbf{x})] - E_{\mathbf{x} \sim p^-_{\mathsf{W}}}[f(\mathbf{x})] \\ &\ge \ KL(p^+||p^-_{\mathsf{W}}) + KL(p^-_{\mathsf{W}}||p^+) \end{aligned} \tag{4}$$

where $KL(p||q)$ denotes the Kullback-Leibler divergence between the two distributions p and q, and $KL(p^+||p^-_{\mathsf{W}}) + KL(p^-_{\mathsf{W}}||p^+)$ is the Jeffreys divergence.

Proof: see Appendix A.
Note that using the Bayes' rule, the ratio of the generative probabilities $\frac{p(\mathbf{x}|y=+1)}{p(\mathbf{x}|y=-1)}$ in Lemma 1 can be turned into the ratio of the discriminative probabilities $\frac{p(y=+1|\mathbf{x})}{p(y=-1|\mathbf{x})}$ assuming equal priors $p(y=+1)=p(y=-1)$.

Theorem 1 *The introspective neural network formulation (Eq. (1)) implicitly minimizes a lower bound of the WGAN objective [2] (Eq. (3)).*
Proof: see Appendix B.

4. Wasserstein Introspective Networks

Algorithm 1 Outline of WINN-single training algorithm. We use $k=3$ and $\lambda=10$.

```
while $\mathsf{W}_t$ has not converged do
    // Classification-step:
    for $k$ steps do
        Sample $m$ positive samples $\{\mathbf{x}_1^+,\cdots,\mathbf{x}_m^+\}$ from $S_+$.
        Sample $m$ pseudo-negative samples $\{\mathbf{x}_1^-,\cdots,\mathbf{x}_m^-\}$ from $S_-^t$.
        Sample $m$ random numbers $\{\alpha_1,\cdots,\alpha_m\}$ from $U[0,1]$.
        $\hat{\mathbf{x}}_i \leftarrow \alpha_i \mathbf{x}_i^+ + (1-\alpha_i)\mathbf{x}_i^-, i=1,\ldots,m$.
        Perform stochastic gradient descent:
        $\nabla_{\mathsf{W}_t} \frac{1}{m}\sum_{i=1}^m\{[f_{\mathsf{W}_t}(\mathbf{x}_i^-)-f_{\mathsf{W}_t}(\mathbf{x}_i^+)]+\lambda(\|\nabla_{\hat{\mathbf{x}}_i} f_{\mathsf{W}_t}(\hat{\mathbf{x}}_i)\|_2-1)^2\}$.
    end for
    // Synthesis-step:
    Sample $r$ noise samples $\{\mathbf{x}_1,\cdots,\mathbf{x}_r\}$ from $p_0^-(\mathbf{x})$.
    Perform stochastic gradient ascent with early-stopping.
    $S_-^{t+1} \leftarrow S_-^t \cup \{\mathbf{x}_1,\cdots,\mathbf{x}_r\}$.
    $t \leftarrow t+1$.
end while
```

Here we present the formulation for WINN building upon the formulation of the prior introspective learning works presented in Section 3.

4.1. WINN algorithm

We denote our unlabeled input training data as $S_+ = \{\mathbf{x}_i | y_i = +1, i=1,\ldots,n\}$. Also, we denote the set of all the self-generated pseudo-negative samples up to step t as $S_-^t = \{\mathbf{x}_i | y_i = -1, i=1,\ldots,l\}$. In other words, S_-^t consists of pseudo-negatives $\mathbf{x}$ sampled from our model $p_{\mathsf{W}_t}^-(\mathbf{x})$ for $t \geq 1$ where W_t is the model parameter vector at step t.

Classification-step. The classification-step can be viewed as training a classifier to approximate the Wasserstein distance between S_+ and S_-^t for $t \geq 1$. Note that we also keep pseudo-negatives from earlier stages – which are essentially the mistakes of the earlier stages – to prevent the classifier forgetting what it has learned in previous stages. We use CNNs parametrized by W_t as base classifiers. Let $f_{\mathsf{W}_t}(\cdot)$ denote the output of final fully connected layer (without passing through sigmoid nonlinearity) of the CNN. In the previous introspective learning frameworks [29, 21], the classifier learning objective was to minimize the following standard cross-entropy loss function on $S_+ \cup S_-^t$:

$$\mathcal{L}(\mathsf{W}_t) = -\{\mathbb{E}_{\mathbf{x}^+\sim p^+} \ln \sigma[+f_{\mathsf{W}_t}(\mathbf{x}^+)] + \mathbb{E}_{\mathbf{x}^-\sim p_{\mathsf{W}_t}^-} \ln \sigma[-f_{\mathsf{W}_t}(\mathbf{x}^-)]\}$$

where $\sigma(\cdot)$ denotes the sigmoid nonlinearity. Motivated by Section 3.2, in WINN training we wish to minimize the following Wasserstein loss function by the stochastic gradient descent algorithm via backpropagation:

$$\mathcal{L}(\mathsf{W}_t) = -\left[\mathbb{E}_{\mathbf{x}^+\sim p^+} f_{\mathsf{W}_t}(\mathbf{x}^+) - \mathbb{E}_{\mathbf{x}^-\sim p_{\mathsf{W}_t}^-} f_{\mathsf{W}_t}(\mathbf{x}^-)\right] \quad (5)$$

To enforce the function f_{W_t} to be 1-Lipschitz, we add the following gradient penalty term [14] to $\mathcal{L}(\mathsf{W}_t)$:

$$\lambda \mathbb{E}_{\hat{\mathbf{x}}\sim p_{\hat{\mathbf{x}}}}[(\|\nabla_{\hat{\mathbf{x}}} f_{\mathsf{W}_t}(\hat{\mathbf{x}})\|_2 - 1)^2]$$

where $\hat{\mathbf{x}} = \alpha \mathbf{x}^+ + (1-\alpha)\mathbf{x}^-$, $\mathbf{x}^+ \sim p^+$, $\mathbf{x}^- \sim p_{\mathsf{W}_t}^-$, and $\alpha \sim U[0,1]$.

Synthesis-step. Obtaining increasingly difficult pseudo-negative samples is an integral part of the introspective learning framework, as it is crucial for tightening the decision boundary. To this end, we develop an efficient sampling procedure under the Wasserstein formulation. After the classification-step, we obtain the following distribution of pseudo-negatives:

$$p_{\mathsf{W}_t}^-(\mathbf{x}) = \frac{1}{Z_t}\exp\{f_{\mathsf{W}_t}(\mathbf{x})\} \cdot p_0^-(\mathbf{x}), \ t=1,\ldots,T \quad (6)$$

where $Z_t = \int \exp\{f_{\mathsf{W}_t}(\mathbf{x})\} \cdot p_0^-(\mathbf{x}) d\mathbf{x}$; the initial distribution $p_0^-(\mathbf{x})$ is a Gaussian distribution $G(\mathbf{x}; 0, \sigma^2)$ or the distribution defined in Appendix **D**. We find that the distribution of Appendix **D** encourages the diversity of sampled images.
The following equivalence is shown in [29, 21]:

$$\frac{p(y=+1|\mathbf{x};\mathsf{W}_t)}{p(y=-1|\mathbf{x};\mathsf{W}_t)} = \exp\{f_{\mathsf{W}_t}(\mathbf{x})\}. \quad (7)$$

The sampling strategy of [29, 21] was to carry out gradient ascent on the term $\ln \frac{p(y=+1|\mathbf{x};\mathsf{W}_t)}{p(y=-1|\mathbf{x};\mathsf{W}_t)}$. In Lemma 1 we chose $f(\mathbf{x})$ to be $\ln \frac{p^+(\mathbf{x})}{p_{\mathsf{W}}^-(\mathbf{x})}$. Using Bayes' rule, it is easy to see that $\nabla \ln \frac{p(y=+1|\mathbf{x};\mathsf{W}_t)}{p(y=-1|\mathbf{x};\mathsf{W}_t)}$ is loosely connected to $\nabla \ln \frac{p^+(\mathbf{x})}{p_{\mathsf{W}_t}^-(\mathbf{x})}$. Also, [2, 14] argue that $f_{\mathsf{W}_t}(\mathbf{x})$ correlates with the quality of the sample $\mathbf{x}$. This motivates us to use the following sampling strategy. After initializing $\mathbf{x}$ by drawing a fair sample from $p_0^-(\mathbf{x})$, we increase $f_{\mathsf{W}_t}(\mathbf{x})$ using gradient ascent on the image $\mathbf{x}$ via backpropagation. Specifically, as shown in [47], we can obtain fair samples from the distribution $p_{\mathsf{W}_t}^-$ using the following update rule:

$$\Delta \mathbf{x} = \frac{\epsilon}{2}\nabla f_{\mathsf{W}_t}(\mathbf{x}) + \eta$$

where ϵ is a time-varying step size and η is a random variable following the Gaussian distribution $N(0, \epsilon)$. Gaussian noise term is added to make samples cover the full distribution. Inspired by [20], we found that injecting noise in the image space could be substituted by applying Dropout to the higher layers of CNN. In practice, we were able obtain the samples of enough diversity without step size annealing and noise injection.

As an early stopping criterion, we empirically find that the following is effective: (1) we measure the minimum and maximum $f_{\mathsf{W}_t}(\cdot)$ of positive examples; (2) we set the early stopping threshold to a random number from the uniform distribution between these two numbers. Intuitively, by matching the value of $f_{\mathsf{W}_t}(\cdot)$ positives and pseudo-negatives, we expect to obtain pseudo-negative samples that match the quality of positive samples.

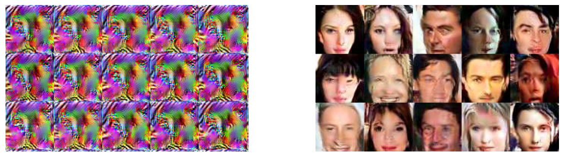
4.2. Expanding model capacity

In practice, we find that the version with the single classifier – which we call WINN-single – is expressive enough to capture the generative distribution under variety of applications. The introspective learning formulation [44, 29, 21] allows us to model more complex distributions by adding a sequence of cascaded classifiers parameterized by $(\mathsf{W}^1, \ldots, \mathsf{W}^K)$. Then, we can model the distribution as:

$$p^-_{\mathsf{W}^k}(\mathbf{x}) = \frac{1}{Z_t} \exp\{f_{\mathsf{W}^k}(\mathbf{x})\} \cdot p^-_{\mathsf{W}^{k-1}}(\mathbf{x}),\ k = 2, \ldots, K \tag{8}$$

In the next sections, we demonstrate the modeling capability of WINN under cascaded classifiers, as well as its agnosticy to the type of base classifier.

4.3. GAN's discriminator vs. WINN's classifier



(a) WGAN-GP discriminator (b) WINN-single

Figure 3: Synthesized images from the discriminators of WGAN and WINN trained on the CelebA dataset. Both use the same ResNet architecture for the discriminator as the one adopted in [14].

GAN uses the competing discriminator and the generator whereas WINN maintains a single model being both generative and discriminative. Some general comparisons between GAN and INN have been provided in [29, 21]. Below we make a few additional observations that are worth future exploration and discussions.

- First, the generator of GAN is a cost-effective option for image patch synthesis, as it works in a feed-forward fashion. However the generator of GAN is not meant to be trained as a classifier to perform the standard classification task, while the generator in the introspective framework is also a strong classifier. Section 5.6 shows WINN to have significant robustness to external adversarial examples.
- Second, the discriminator in GAN is meant to be a critic but not a generator. To show whether or not the discriminator in GAN can also be used as a generator, we train WGAN-GP [14] on the CelebA face dataset. Using the same CNN architecture (ResNet from [14]) that was used as GAN's discriminator, we also train a WINN-single model, making GAN's discriminator and WINN-single to have the identical CNN architecture. Applying the sampling strategy to WGAN-GP's discriminator allows us to synthesize image form WGAN-GP's discriminator as well and we show some samples in Figure 3 (a). These synthesized images are not like faces, yet they have been classified by the discriminator of WGAN-GP as "real" faces; this demonstrates the separation between the generator and the discriminator in GAN. In contrast, images synthesized by WINN-single's CNN classifier are faces like, as shown in Figure 3 (b).
- Third, the discriminator of GAN may not be used as a direct discriminative classifier for the standard supervised learning task. As shown and discussed in ICN [21], the introspective framework has the ability of classification for discriminator.

5. Experiments

5.1. Implementation

Classification-step. For training the discriminator network, we use Adam [23] with a mini-batch size of 100. The learning rate was set to 0.0001. We set $\beta_1 = 0.0$, and $\beta_2 = 0.9$, inspired by [14]. Each batch consists of 50 positive images sampled from the set of positives S_+ and 50 pseudo-negative images sampled from the set of pseudo-negatives S_-. In each iteration, we limit total number of training images to $10,000$.

Synthesis-step. For synthesizing pseudo-negative images via back-propagation, we perform gradient ascent on the image space. In the first cascade, each image is initialized with a noise sampled from the distribution described in Appendix **D**. In the later cascades, images are initialized with the images sampled from the last cascade. We use Adam with a mini-batch size of 100. The learning rate was set to 0.01. We set $\beta_1 = 0.9$, and $\beta_2 = 0.99$.

5.2. Texture modeling

We evaluate the texture modeling capability of WINN. For a fair comparison, we use the same 7 texture images presented in [9] where each texture image has a size of 256 $\times$ 256. We follow the training method of Section 5.1 except that positive images are constructed by cropping 64 $\times$ 64 patches from the source texture image at random positions. We use network architecture of Appendix **C**. After training is done on the 64$\times$64 patch-based model, we try to synthesize texture images of arbitrary size using the anysize-image-generation method following [29]. During the synthesis process, we keep a single working image of

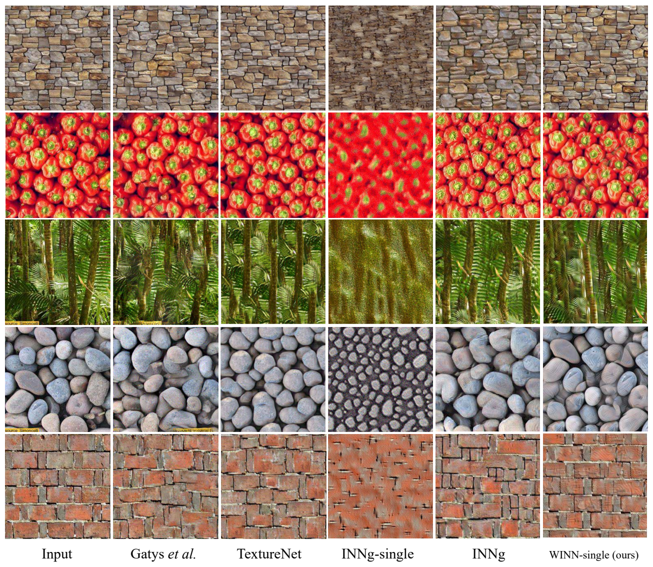


Figure 4: More texture synthesis results. Gatys *et al.* [9] and TextureNets [45] results are from [45].

size 320×320. Note that we expand the image so that center 256×256 pixels are covered with equal probability. In each iteration, we sample 200 patches from the working image, and perform gradient ascent on the chosen patches. For the overlapping pixels between patches, we take the average of the gradients assigned to such pixels. We show synthesized texture images in Figure 2 and 4. WINN-single shows a significant improvement over INNg-single and comparable results to INNg (using 20 CNNs). It is worth noting that [10, 45] leverage rich features of VGG-19 network pretrained on ImageNet. WINN and INNg instead train networks from scratch.

5.3. CelebA face modeling

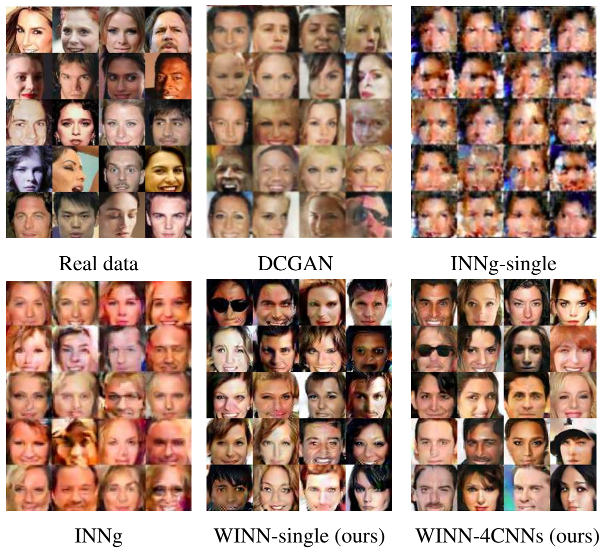


Figure 5: Images generated by various models trained on CelebA.

The CelebA dataset [33] consists of 202, 599 face images of celebrities. This dataset has been widely used in the previous generative modeling works since it contains large pose variations and background clutters. The network architecture adopted here is described in Appendix **C**. In Figure 5, we show some synthesized face images using WINN-single and WINN, as well as those by DCGAN [38], INNg-single, and INNg [29]. WINN-single attains image quality even higher than that of INNg (12 CNNs).

5.4. SVHN modeling

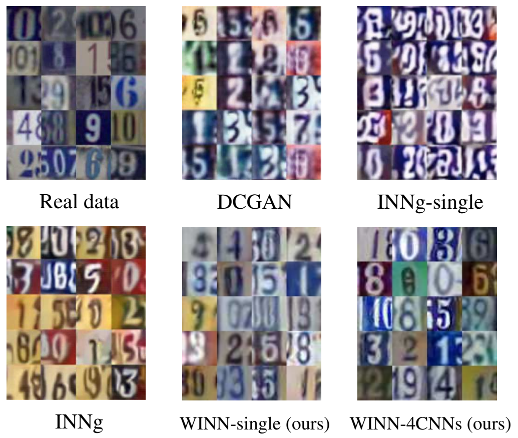


Figure 6: Images generated by various models trained on SVHN. DCGAN [38] result is from [29].

SVHN [34] consists of 32 × 32 images from Google Street View. It contains 73, 257 training images, 26, 032 test images, and 531, 131 extra images. We use only the training images for the unsupervised SVHN modeling. We use the ResNet architecture described in [14]. Generated images by WINN-single and WINN (4 CNN classifiers) as well as DCGAN and INN are shown in Figure 6. The improvement of WINN over INNg is evident.

5.5. CIFAR-10 modeling

Table 1: Inception score on CIFAR-10. "-L" in Improved GANs means without labels.

Method	Score
Real data	$11.95 \pm .20$
WGAN-GP [14]	$\mathbf{7.86} \pm .07$
WGAN [2]	$5.88 \pm .07$
DCGAN [38] (in [19])	$6.16 \pm .07$
ALI [7] (in [46])	$5.34 \pm .05$
Improved GANs (-L) [40]	$4.36 \pm .04$
INNg-single [29]	$1.95 \pm .01$
INNg [29]	$3.04 \pm .02$
WINN-single (ours)	$4.62 \pm .05$
WINN-5CNNs (ours)	$5.58 \pm .05$

CIFAR-10 [25] consists of 50, 000 training images and 10, 000 test images of size 32 × 32 in 10 classes. We use training images augmented by horizontal flips [26] for unsupervised CIFAR-10 modeling. We use the ResNet given in [14]. Figure 7 shows generated images by various models.

To measure the semantic discriminability, we compute the Inception scores [40] on 50, 000 generated images.

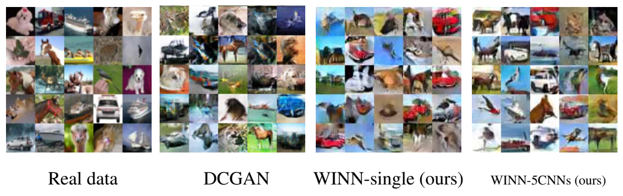


Figure 7: Images generated by models trained on CIFAR-10.

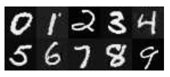
WINN shows its clear advantage over INN. WINN-5CNNs produces a result close to WGAN but there is still a gap to the state-of-the-art results by WGAN-GP.

5.6. Image classification and adversarial examples

To demonstrate the robustness of WINN as a discriminative classifier, we present experiments on the supervised classification tasks.

Table 2: Test errors on MNIST and SVHN. When training on SVHN, we only use training set. All results except WINN-single and baseline ResNet-32 are from [21]. [21] adopted DCGAN [38] discriminator as their CNN architecture. The advantage of WINN over a vanilla CNN is evident. When applied to a stronger baseline such as ResNet-32, WINN is not losing ResNet's superior classification capability in standard supervised classification, while attaining special generative capability and robustness to adversarial attacks (see Table 3 and 4) that do not exist in ResNet. The images below on the right are the generated samples by the corresponding WINN (ResNet-32) classifiers reported in the table.

Method	Error
MNIST	
Baseline vanilla CNN (4 layers)	0.89%
CNN + GDL [44]	0.85%
CNN + DCGAN [38]	0.84%
ICN [21]	0.81%
WINN-single vanilla (ours)	0.67%
Baseline ResNet-32	**0.45%**
WINN-single ResNet-32 (ours)	0.48%
SVHN	
Baseline ResNet-32	4.64%
WINN-single ResNet-32 (ours)	**4.50%**



Training Methods. We add the Wasserstein loss term to the ICN [21] loss function, obtaining the following:

$$\mathcal{L}(\mathsf{W}_t) = -\sum_{\mathbf{x}_i\in S_+}\ln\frac{\exp\{\mathbf{w}_t^{(1)y_i}\cdot\phi(\mathbf{x}_i;\mathbf{w}_t^{(0)})\}}{\sum_{k=1}^{K}\exp\{\mathbf{w}_t^{(1)k}\cdot\phi(\mathbf{x}_i;\mathbf{w}_t^{(0)})\}} +\alpha\Big(\sum_{\mathbf{x}_i\in S_-^t} f_{\mathsf{W}_t}(\mathbf{x}_i)-\sum_{\mathbf{x}_i\in S_+} f_{\mathsf{W}_t}(\mathbf{x}_i) +\lambda\mathbb{E}_{\hat{\mathbf{x}}\sim p_{\hat{\mathbf{x}}}}[(\|\nabla_{\hat{\mathbf{x}}} f_{\mathsf{W}_t}(\hat{\mathbf{x}})\|_2-1)^2]\Big)$$

where $\mathsf{W}_t =< \mathbf{w}_t^{(0)}, \mathbf{w}_t^{(1)_1}, ..., \mathbf{w}_t^{(1)_K} >$, $\mathbf{w}_t^{(0)}$ denotes the internal parameters for the CNN, and $\mathbf{w}_t^{(1)_k}$ denotes the top-layer weights for the k-th class. In the experiments, we set the weight of the WINN loss, α, to 0.01. We use the vanilla network architecture resembling [21] as the baseline CNN, which has less filters and parameters than the one in [21]. We also use a ResNet-32 architecture with Layer Normalization [3] on MNIST and SVHN. In the classification-step, we use Adam with a fixed learning rate of 0.001, β_1 of 0.0. In the synthesis-step, we use the Adam optimizer with a learning rate of 0.02 and β_1 of 0.9. Table 2 shows the errors on MNIST and SVHN.

Table 3: Adversarial examples comparison between the baseline CNN and WINN on MNIST. We first generate N adversarial examples using method A and count the number of adversarial examples misclassified by A ($= N_A$). **Adversarial error** of A is defined as test error rate against adversarial examples ($= N_A/N$). Then among A's mistakes, we count the number of adversarial examples misclassified by B ($= N_{A\cap B}$). Then **error correction rate** by B is $1 - N_{A\cap B}/N_A$. $\uparrow$ means higher the better; $\downarrow$ means lower the better. The comparisons are made in two groups: the first group builds on top a vanilla 4 layer CNN and the second group adopts ResNet-32. Note that the corresponding errors for these classifiers on the standard MNIST supervised classification task can be seen in Table 2.

Method	Adversarial error of **Method** $\downarrow$	Correction rate by **Method** $\uparrow$	Correction rate by **Baseline** $\downarrow$
Baseline vanilla CNN	32.41%	-	-
ICN [21]	19.02%	52.58%	58.68%
WINN-single vanilla (ours)	**7.99%**	**90.00%**	**46.93%**
Baseline ResNet-32	11.28%	-	-
WINN-single ResNet-32 (ours)	**2.05%**	**89.68%**	**43.69%**

Robustness to adversarial examples. It is argued in [11] that discriminative CNN's vulnerability to adversarial examples primarily arises due to its linear nature. Since the reclassification-by-synthesis process helps tighten the decision boundary (Figure 1), one might expect that CNNs trained with the WINN algorithm are more robust to adversarial examples. Note that unlike existing methods for adversarial defenses [13, 28], our method does not train networks with specific types of adversarial examples. With test images of MNIST and SVHN, we adopt "fast gradient sign method" [11] ($\epsilon = 0.125$ for MNIST and $\epsilon = 0.005$ for SVHN) to generate adversarial examples clipped to range $[-1, 1]$, which differs from [21]. We experiment with two networks having the same architecture and only differing in training method (the standard cross-entropy loss vs. the WINN procedure). We call the former as the baseline CNN. We summarize the results in Table 3. Compared to ICN [21], WINN significantly reduces the adversarial error to 7.99% and improves the correction rate to 90.00%. In addition, we have adopted the ResNet-32 architecture into WINN. See Table 3 and 4. We still obtain the adversarial error reduction and correction rate improvement on MNIST and SVHN ($\epsilon = 0.005$) with ResNet-32. Our observation is that WINN is not necessarily improving over a strong baseline for the supervised classification task but its advantage on adversarial attacks is evident.

Table 4: Adversarial examples comparison between the baseline ResNet-32 [16] and WINN on SVHN. Note that the corresponding errors for these classifiers on the standard supervised SVHN classification task can be seen in Table 2.

Method	Adversarial error of **Method**	Correction rate by **Method** $\uparrow$	Correction rate by **Baseline** $\downarrow$
Baseline ResNet-32	29.29%	-	-
WINN-single ResNet-32 (ours)	**19.53%**	74.39%	51.37%

5.7. Agnostic to different architectures

In Figure 8, we demonstrate our algorithm being agnostic to the type of classifier, by varying network architectures to ResNet [16] and DenseNet [18]. Little modification was required to adapt two architectures for WINN.

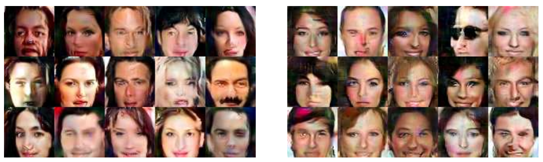


Figure 8: Synthesized CelebA images with varying network architectures. We use a single CNN for each experiment.

6. Conclusion

In this paper, we have introduced Wasserstein introspective neural networks (WINN) that produce encouraging results as a generator and a discriminative classifier at the same time. WINN is able to achieve model size reduction over the previous introspective neural networks (INN) by a factor of 20. In most of the images shown in the paper, we find a single CNN classifier in WINN being sufficient to produce visually appealing images as well as significant error reduction against adversarial examples. WINN is agnostic to the architecture design of CNN and we demonstrate results on three networks including a vanilla CNN, ResNet [16], and DenseNet [18] networks where popular CNN discriminative classifiers are turned into generative models under the WINN procedure. WINN can be adopted in a wide range of applications in computer vision such as image classification, recognition, and generation.

Acknowledgements. This work is supported by NSF IIS-1717431 and NSF IIS-1618477. The authors thank Justin Lazarow, Long Jin, Hubert Le, and Ying Nian Wu for valuable discussions.

7. Appendix

A. Proof of Lemma 1.

Proof. Plugging $f(\mathbf{x}) = \ln \frac{p^+(\mathbf{x})}{p^-_{\mathsf{W}}(\mathbf{x})}$ into Eq. (3), we have

$$\begin{aligned}
& E_{\mathbf{x}\sim p^+}[f(\mathbf{x})] - E_{\mathbf{x}\sim p^-_{\mathsf{W}}}[f(\mathbf{x})] \\
= \; & E_{\mathbf{x}\sim p^+}[\ln \frac{p^+(\mathbf{x})}{p^-_{\mathsf{W}}(\mathbf{x})}] - E_{\mathbf{x}\sim p^-_{\mathsf{W}}}[\ln \frac{p^+(\mathbf{x})}{p^-_{\mathsf{W}}(\mathbf{x})}] \\
= \; & \int p^+(\mathbf{x}) \ln \frac{p^+(\mathbf{x})}{p^-_{\mathsf{W}}(\mathbf{x})} d\mathbf{x} - \int p^-_{\mathsf{W}}(\mathbf{x}) \ln \frac{p^+(\mathbf{x})}{p^-_{\mathsf{W}}(\mathbf{x})} d\mathbf{x} \\
= \; & KL(p^+||p^-_{\mathsf{W}}) + KL(p^-_{\mathsf{W}}||p^+) \qquad \square
\end{aligned}$$

B. Proof of Theorem 1.

Corollary 1 *The Jeffreys divergence in Eq. (4) of lemma 1 is lower and upper bounded by $KL(p^+||p^-_{\mathsf{W}})$ up to some multiplicative constant*

$$(1+\frac{p^+_{min}}{2})KL(p^+||p^-_{\mathsf{W}}) \le JD(p^+;p^-_{\mathsf{W}}) \le (1+\frac{2}{p^+_{min}})KL(p^+||p^-_{\mathsf{W}}),$$

where $JD(p^+;p^-_{\mathsf{W}}) = KL(p^+||p^-_{\mathsf{W}}) + KL(p^-_{\mathsf{W}}||p^+)$ is the Jeffreys divergence.

Proof. Based on the Pinsker's inequality [37, 41], it is observed that

$$KL(p^-_{\mathsf{W}}||p^+) \ge \frac{1}{2}|p^-_{\mathsf{W}} - p^+|^2 \log e,$$

where $|p^-_{\mathsf{W}} - p^+|$ is total variation (TV) distance. From [41] we also have

$$KL(p^-_{\mathsf{W}}||p^+) \le \frac{\log e}{p^+_{min}}|p^-_{\mathsf{W}} - p^+|^2,$$

where $p^+_{min} = \min_{\mathbf{x}} p^+(\mathbf{x})$. Applying the above bounds to $KL(p^+||p^-_{\mathsf{W}})$ and using the symmetry of the TV distance $|p^-_{\mathsf{W}} - p^+| \equiv |p^+ - p^-_{\mathsf{W}}|$,

$$\frac{p^+_{min}}{2}KL(p^+||p^-_{\mathsf{W}}) \le KL(p^-_{\mathsf{W}}||p^+) \le \frac{2}{p^+_{min}}KL(p^+||p^-_{\mathsf{W}}).$$

Plugging the equation above into the Jeffreys divergence, we observe that $KL(p^+||p^-_{\mathsf{W}}) + KL(p^-_{\mathsf{W}}||p^+)$ is upper and lower bounded by by $KL(p^+||p^-_{\mathsf{W}})$. $\square$

Now we can look at theorem 1. It was shown in [21] that Eq. (1) reduces $KL(p^+||p^-_{\mathsf{W}})$, which bounds the Jeffreys divergence $KL(p^+||p^-_{\mathsf{W}}) + KL(p^-_{\mathsf{W}}||p^+)$ as shown in corollary 1. Lemma 1 shows the connection between Jeffreys divergence and the WGAN objective (Eq. (3)) when $f(\mathbf{x}) = \ln \frac{p^+(\mathbf{x})}{p^-_{\mathsf{W}}(\mathbf{x})}$. We therefore can see that the formulation of introspective neural networks (Eq. (1)) connects to a lower bound of the WGAN [2] objective (Eq. (3)). $\square$

C. Texture and CelebA Modeling Architecture. Inspired by [22], we design a CNN architecture for 64×64 image as in Table 5. We use Swish [39] non-linearity after each convolutional layer. We add Layer Normalization [3] after each convolution except the first layer, following [14].

Table 5: Network architecture for the texture and face image modeling.

Textures and CelebA

Layer	Filter size/stride	Output size
Input		64×64×3
Conv3-32	3×3/1	64×64×32
Conv3-64	3×3/1	64×64×64
Avg pool	2×2/2	32×32×64
Conv3-64	3×3/1	32×32×64
Conv3-128	3×3/1	32×32×128
Avg pool	2×2/2	16×16×128
Conv3-128	3×3/1	16×16×128
Conv3-256	3×3/1	16×16×256
Avg pool	2×2/2	8×8×256
Conv3-256	3×3/1	8×8×256
Conv3-512	3×3/1	8×8×256
Avg pool	2×2/2	4×4×512
FC-1		1×1×1

Alternative Initialization

Layer	Filter size/stride	Output size
Input		4×4×512
Conv5-256	5×5/1	4×4×256
Upsample		8×8×256
Conv5-128	5×5/1	8×8×128
Upsample		16×16×128
Conv5-64	5×5/1	16×16×64
Upsample		32×32×64
Conv5-64	5×5/1	32×32×3
Upsample		64×64×3

D. Alternative Initializations. We sample an initial pseudo-negative image by applying an operation defined by the network above to a tensor of size $4 \times 4 \times 512$ sampled from $U[-1, 1]$. The weights of the network are sampled from $G(0, 0.1^2)$. We do not apply any nonlinearities in the network. We add Layer Normalization [3] after each convolution except the last layer.

References

[1] D. H. Ackley, G. E. Hinton, and T. J. Sejnowski. A learning algorithm for boltzmann machines. *Cognitive science*, 9(1):147–169, 1985. 1

[2] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein generative adversarial networks. In *ICML*, 2017. 1, 2, 3, 4, 6, 8

[3] L. J. Ba, R. Kiros, and G. E. Hinton. Layer normalization. *CoRR*, abs/1607.06450, 2016. 7, 8

[4] P. Baldi. Autoencoders, unsupervised learning, and deep architectures. In *ICML Workshop on Unsupervised and Transfer Learning*, pages 37–49, 2012. 1

[5] S. Della Pietra, V. Della Pietra, and J. Lafferty. Inducing features of random fields. *IEEE transactions on pattern analysis and machine intelligence*, 19(4):380–393, 1997. 1

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

[7] V. Dumoulin, I. Belghazi, B. Poole, O. Mastropietro, A. Lamb, M. Arjovsky, and A. Courville. Adversarially learned inference. In *ICLR*, 2017. 6

[8] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *J. of Comp. and Sys. Sci.*, 55(1), 1997. 2

[9] L. Gatys, A. S. Ecker, and M. Bethge. Texture synthesis using convolutional neural networks. In *NIPS*, 2015. 3, 5, 6

[10] L. A. Gatys, A. S. Ecker, and M. Bethge. A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576*, 2015. 6

[11] 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, 3, 7

[12] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014. 2

[13] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. In *ICLR*, 2015. 7

[14] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville. Improved training of wasserstein gans. In *NIPS*, 2017. 1, 2, 4, 5, 6, 8

[15] T. Han, Y. Lu, S.-C. Zhu, and Y. N. Wu. Alternating back-propagation for generator network. In *AAAI*, 2017. 2

[16] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 1, 7, 8

[17] G. E. Hinton, S. Osindero, and Y.-W. Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006. 1

[18] G. Huang*, Z. Liu*, L. van der Maaten, and K. Q. Weinberger. Densely connected convolutional networks. In *CVPR*, 2017. 1, 8

[19] X. Huang, Y. Li, O. Poursaeed, J. Hopcroft, and S. Belongie. Stacked generative adversarial networks. In *CVPR*, 2017. 6

[20] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *CVPR*, 2017. 5

[21] L. Jin, J. Lazarow, and Z. Tu. Introspective classification with convolutional nets. In *NIPS*, 2017. 2, 3, 4, 5, 7, 8

[22] T. Karras, T. Aila, S. Laine, and J. Lehtinen. Progressive growing of gans for improved quality, stability, and variation. In *ICLR*, 2018. 8

[23] D. Kingma and J. Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 5

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

[25] A. Krizhevsky, V. Nair, and G. Hinton. Cifar-10 (canadian institute for advanced research). 6

[26] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In *NIPS*, 2012. 1, 6

[27] T. D. Kulkarni, W. F. Whitney, P. Kohli, and J. Tenenbaum. Deep convolutional inverse graphics network. In *NIPS*, 2015. 1

[28] A. Kurakin, I. Goodfellow, and S. Bengio. Adversarial machine learning at scale. In *ICLR*, 2017. 7

[29] J. Lazarow*, L. Jin*, and Z. Tu. Introspective neural networks for generative modeling. In *ICCV*, 2017. 2, 3, 4, 5, 6

[30] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. Howard, W. Hubbard, and L. Jackel. Backpropagation applied to handwritten zip code recognition. In *Neural Computation*, 1989. 1

[31] C.-Y. Lee*, S. Xie*, P. Gallagher, Z. Zhang, and Z. Tu. Deeply-supervised nets. In *AISTATS*, 2015. 1

[32] S. Liu, O. Bousquet, and K. Chaudhuri. Approximation and convergence properties of generative adversarial learning. In *NIPS*, 2017. 3

[33] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *ICCV*, 2015. 6

[34] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, volume 2011, page 5, 2011. 6

[35] J. Ngiam, Z. Chen, P. W. Koh, and A. Y. Ng. Learning deep energy models. In *ICML*, 2011. 2

[36] B. A. Olshausen and D. J. Field. Sparse coding with an overcomplete basis set: A strategy employed by v1? *Vision research*, 37(23):3311–3325, 1997. 1

[37] M. S. Pinsker. Information and information stability of random variables and processes. 1960. 8

[38] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *ICLR*, 2016. 1, 6, 7

[39] P. Ramachandran, B. Zoph, and Q. V. Le. Searching for activation functions. *CoRR*, abs/1710.05941, 2017. 8

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

[41] I. Sason and S. Verdú. f-divergence inequalities. *IEEE Transactions on Information Theory*, 62(11):5973–6006, 2016. 8

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

[43] 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

[44] Z. Tu. Learning generative models via discriminative approaches. In *CVPR*, 2007. 1, 2, 3, 5, 7

[45] D. Ulyanov, V. Lebedev, A. Vedaldi, and V. Lempitsky. Texture networks: Feed-forward synthesis of textures and stylized images. In *ICML*, 2016. 3, 6

[46] D. Warde-Farley and Y. Bengio. Improving generative adversarial networks with denoising feature matching. In *ICLR*, 2017. 6

[47] M. Welling and Y. W. Teh. Bayesian learning via stochastic gradient langevin dynamics. In *ICML*, 2011. 3, 4

[48] M. Welling, R. S. Zemel, and G. E. Hinton. Self supervised boosting. In *NIPS*, 2002. 1, 2

[49] Y. N. Wu, Z. Si, H. Gong, and S.-C. Zhu. Learning active basis model for object detection and recognition. *International journal of computer vision*, 90(2):198–235, 2010. 1

[50] J. Xie, Y. Lu, R. Gao, S.-C. Zhu, and Y. N. Wu. Cooperative learning of energy-based model and latent variable model via mcmc teaching. In *AAAI*, 2018. 2

[51] J. Xie, Y. Lu, S.-C. Zhu, and Y. N. Wu. A theory of generative convnet. In *ICML*, 2016. 2

[52] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He. Aggregated residual transformations for deep neural networks. In *CVPR*, 2017. 1

[53] S.-C. Zhu and D. Mumford. A stochastic grammar of images. *Foundations and Trends® in Computer Graphics and Vision*, 2(4):259–362, 2007. 1

[54] S. C. Zhu, Y. N. Wu, and D. Mumford. Minimax entropy principle and its application to texture modeling. *Neural Computation*, 9(8):1627–1660, 1997. 1