Deep Learning and Its Applications in Signal Processing

Lesson 6: Generative Models and Generative Adversarial Networks

Liang Dong, ECE

Baylor University
SCHOOL OF ENGINEERING & COMPUTER SCIENCE
Department of Electrical & Computer Engineering

Discriminative Models vs. Generative Models

Discriminative Model

- Classification: Given the features of an instance of data $x \sim p_{data}(x)$, it predicts a label or category y to which that data belongs.

- The discriminative model learns a function that maps the input data x to some output class label y.

- It learns the conditional distribution $p(y|x)$.

Discriminative Models vs. Generative Models

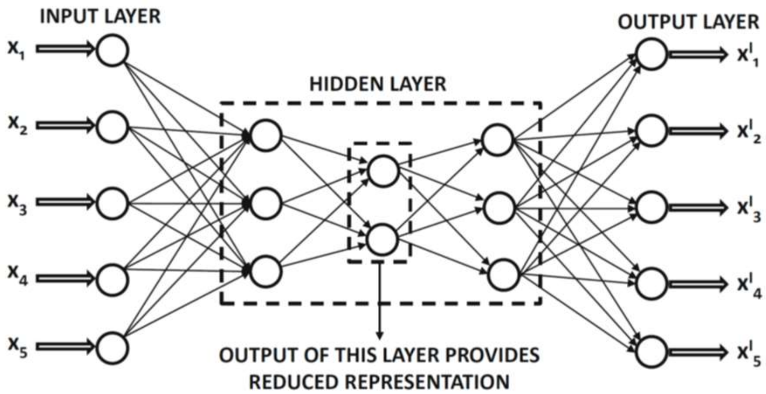
Generative Model

- Modeling Data Distribution: Given finite samples of the data distribution $X = \{x|x \sim p_{data}(x)\}$, it finds a model such that $p_{model}(x; \theta) \approx p_{data}(x)$.

- The generative model tries to learn the underlying structure of the input data and can generate synthetic data.

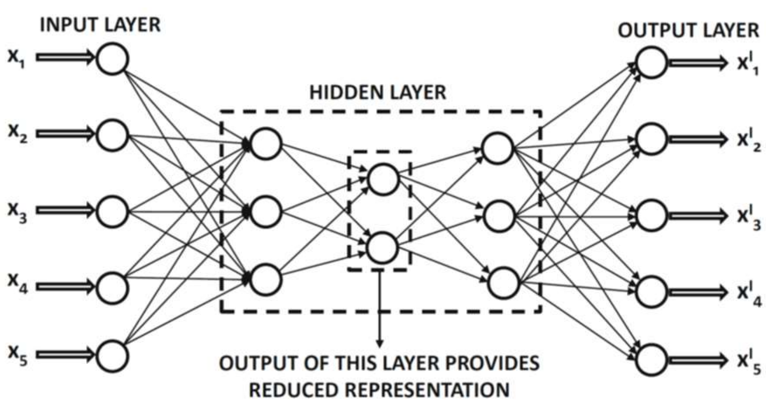
- It learns the joint probability of the input data and label $p(x, y)$.

Autoencoder



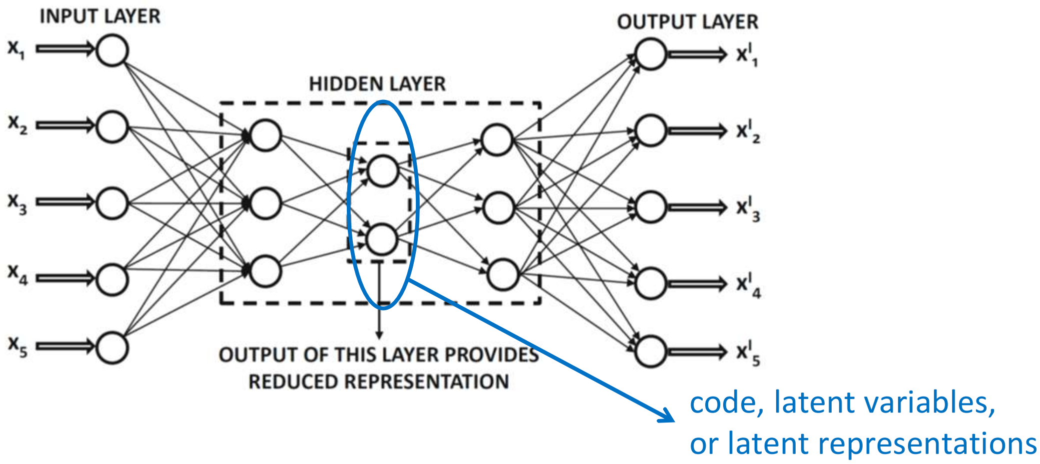
Autoencoder can be used for learning generative models of data.

Autoencoder



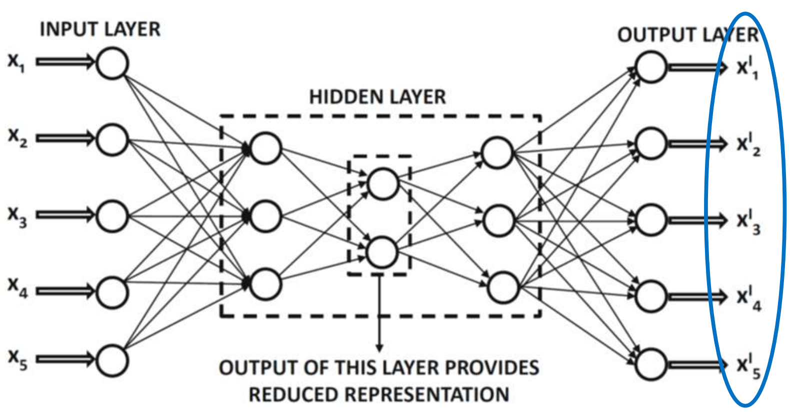
An autoencoder has an output layer with the same dimensionality as the input.
The number of units in each middle layer is *constricted*. These units in the middle layer hold a reduced representation of the data (dimensionality reduction).

Autoencoder



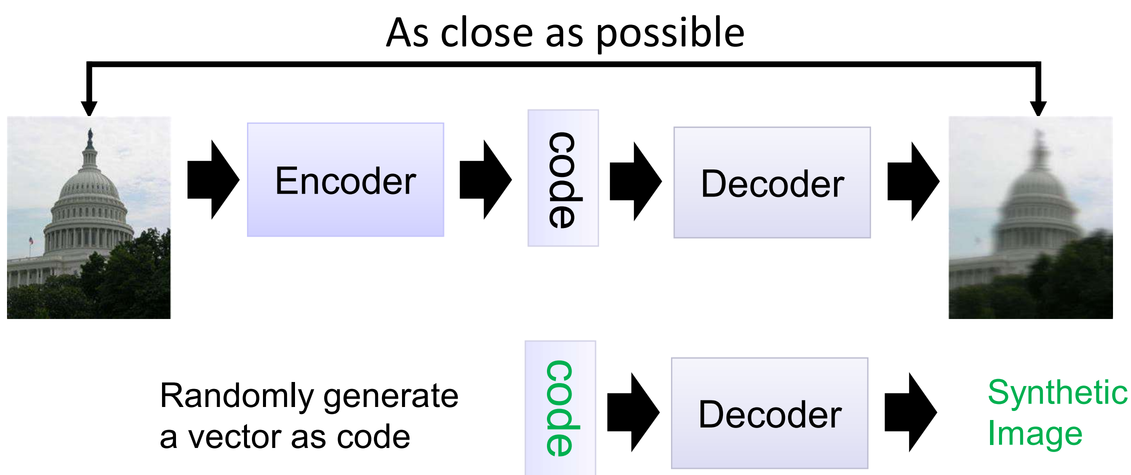
The autoencoder learns to compress data from the input into a short code.

Autoencoder

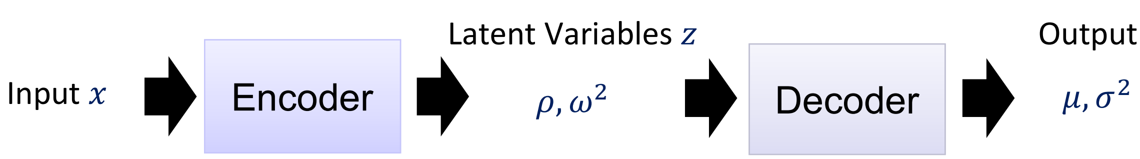


Along with the reduction side, a reconstructing side is learnt. The autoencoder tries to uncompress the code into something that closely matches the original data.

Autoencoder as Generative Model



Variational Autoencoder



Strong assumptions concerning the distribution of latent variables.

The prior over the latent variables is usually set to be the centered isotropic multivariate Gaussian.

$q(z|x) = N(\rho, \omega^2 I)$, and the posterior distribution is $p(x|z) = N(\mu, \sigma^2 I)$.

Variational Autoencoder

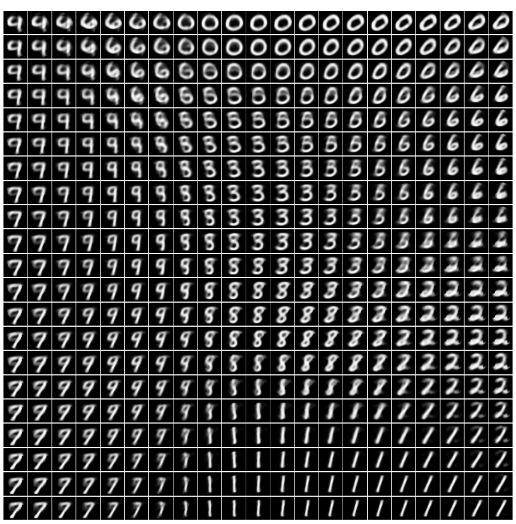
Input x → Encoder → Latent Variables z ρ, ω^2 → Decoder → Output μ, σ^2

The objective function is $D_{KL}(q(z|x)||p(z)) - E_{q(z|x)}(\log p(x|z))$

(Kullback-Leibler divergence)

However, this model only shows the mean of the distributions rather than a sample of the learned Gaussian distribution.

Variational Autoencoder Working on MNIST Dataset



Learned MNIST manifold

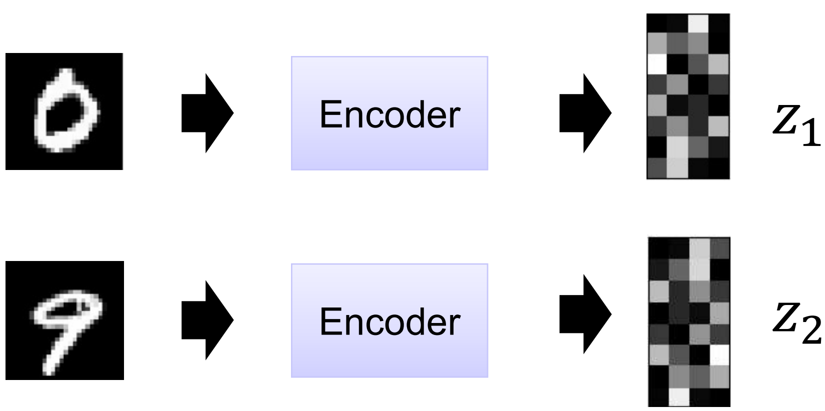
Diederik P Kingma, Max Welling, Auto-Encoding Variational Bayes, 2014. arXiv:1312.6114v10

Variational Autoencoder Working on MNIST Dataset

1st epoch 9th epoch original

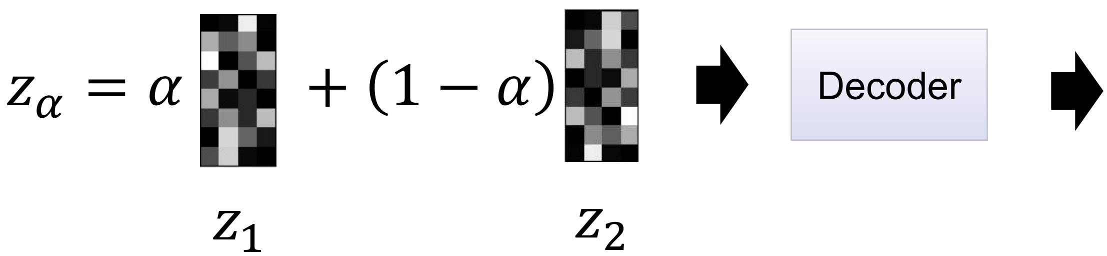
Autoencoder for Latent Space Modeling

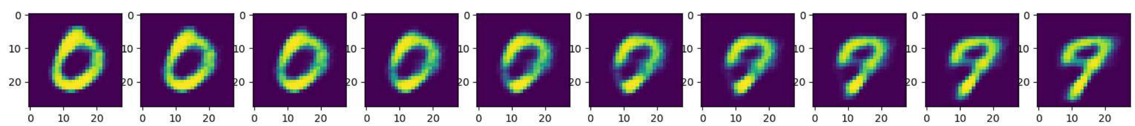
Example of latent space interpolation



Autoencoder for Latent Space Modeling

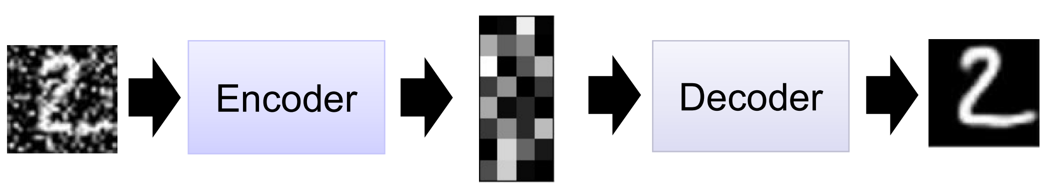
Example of latent space interpolation





Denoising Autoencoder

DAE tries to undo the effect of corruption process stochastically applied to the input.



Noisy Input $\tilde{x}$

$\tilde{x}$ is a corrupted copy of x.

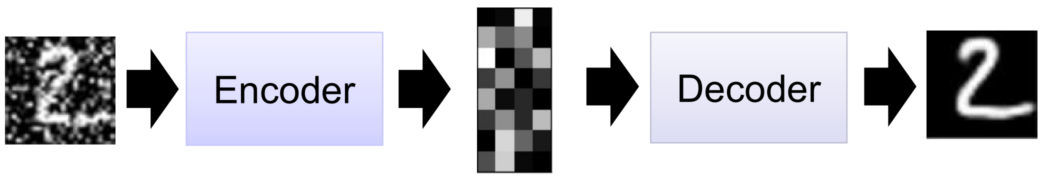
Denoised Output

$\hat{x} = g(f(\tilde{x}))$

Minimization of the loss $L(x, g(f(\tilde{x}))$

Denoising Autoencoder

DAE tries to undo the effect of corruption process stochastically applied to the input.



Noisy Input $\tilde{x}$

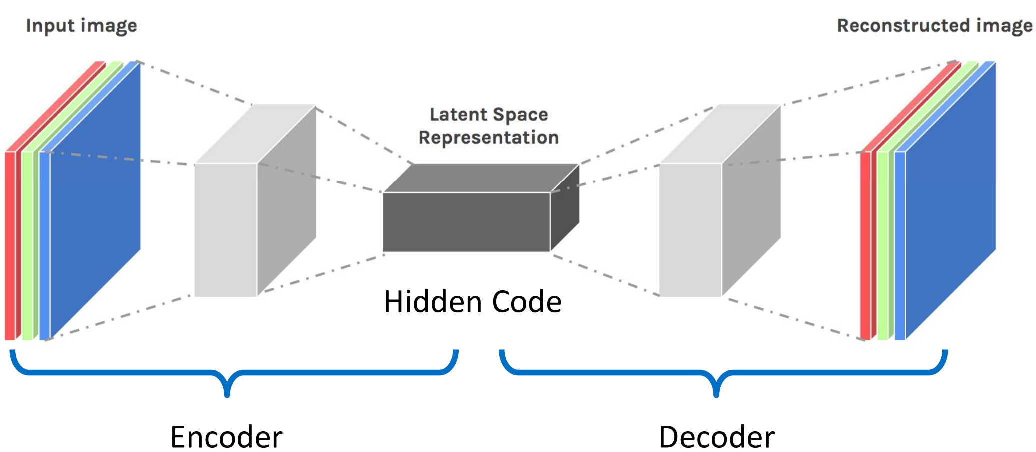
$\tilde{x}$ is a corrupted copy of x.

Denoised Output

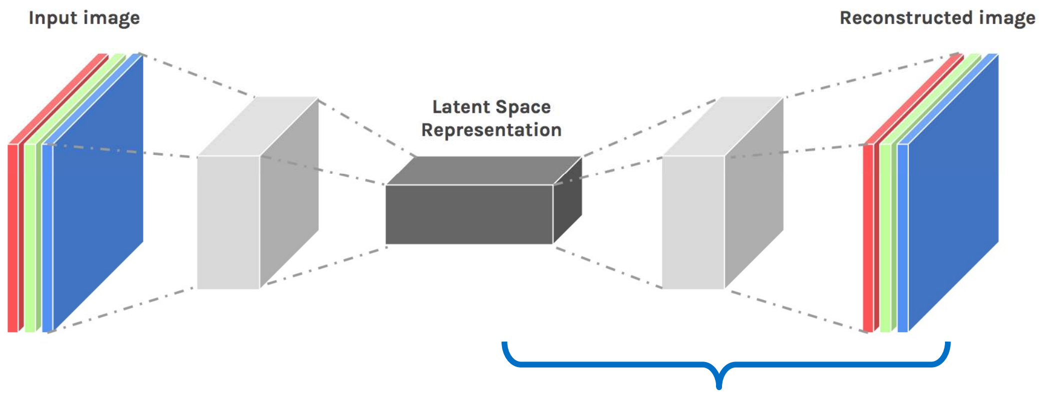
$\hat{x} = g(f(\tilde{x}))$

The DAE forces the hidden layer to learn a generalized structure of the data, or concentrates the data near a lower dimensional manifold.

Convolutional Autoencoder

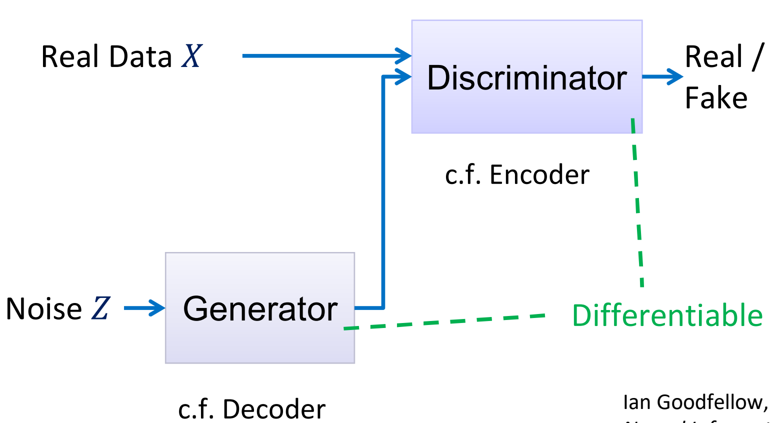


Convolutional Autoencoder



Deconvolutional Layer: Using *fractionally strided convolutions* or *transposed convolutions* at a fractional value, e.g., 0.5.

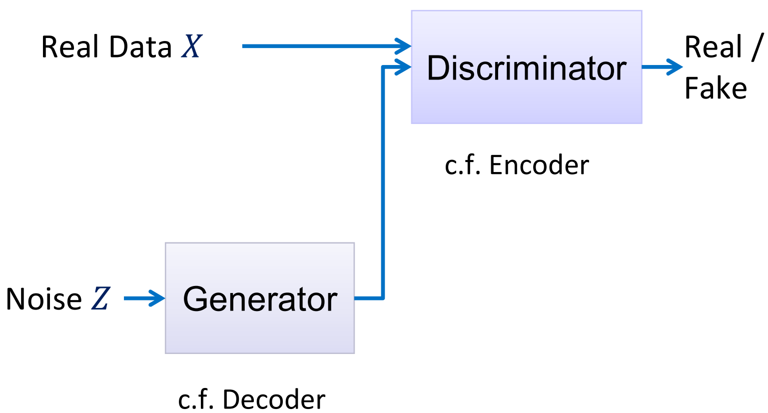
Generative Adversarial Networks



Generative adversarial networks (GANs) are deep neural network architectures comprised of two networks, competing against each other in a zero-sum game framework.

Ian Goodfellow, et al. “Generative Adversarial Nets”, *Proc. Advances in Neural Information Processing Systems*, pp. 2672-2680, 2014

Generative Adversarial Networks



One network, called the *generator*, generates new data instances, while the other, the *discriminator*, evaluates them for authenticity.

The discriminator decides whether each instance of data that it reviews belongs to the actual training dataset or not.

Z is random noise and can be viewed as the latent representation of the data.

Generative Adversarial Networks

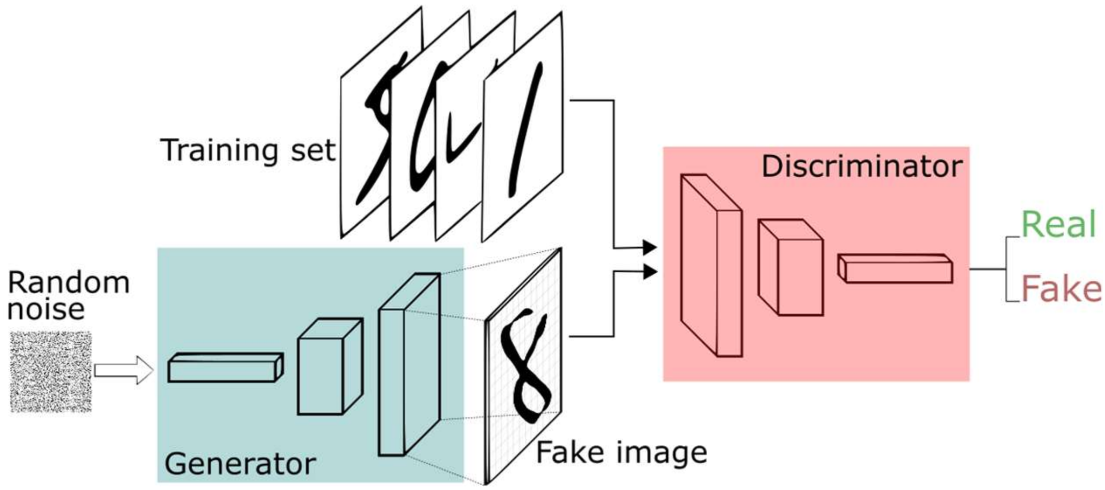
The data distribution is $x \sim p_{data}(x), x \in X$.

The generator G has a latent prior $z \sim p_z(z), z \in Z$ and maps this to sample space $G: Z \to X$.

G implicitly defines a distribution $p_{model}(x; \theta_G)$.

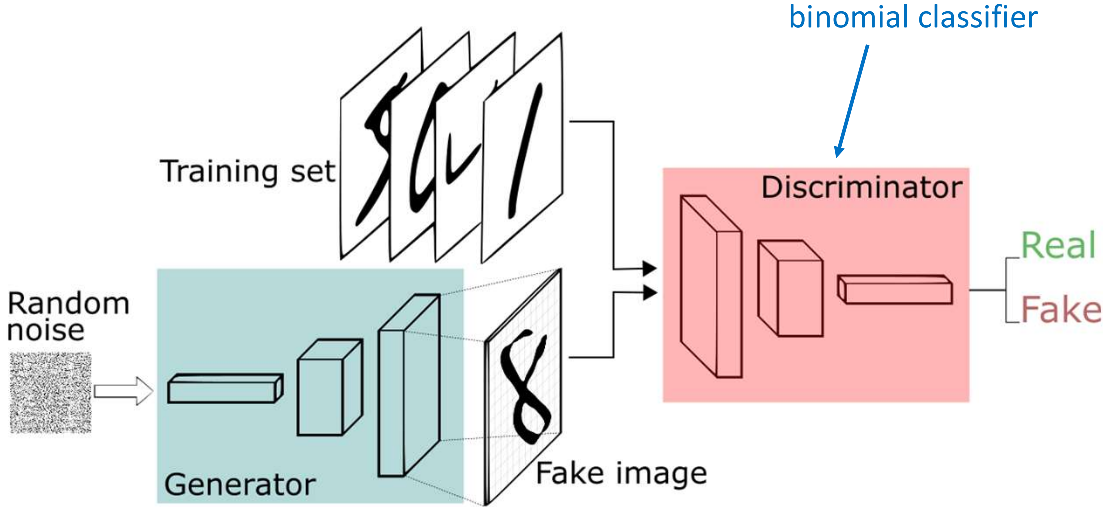
The discriminator D tells how real a sample looks via a score $D: X \to \mathbb{R}$ (It outputs a single scalar – the prob that x comes from p_{data} rather than p_{model}.)

Generating Image with GANs

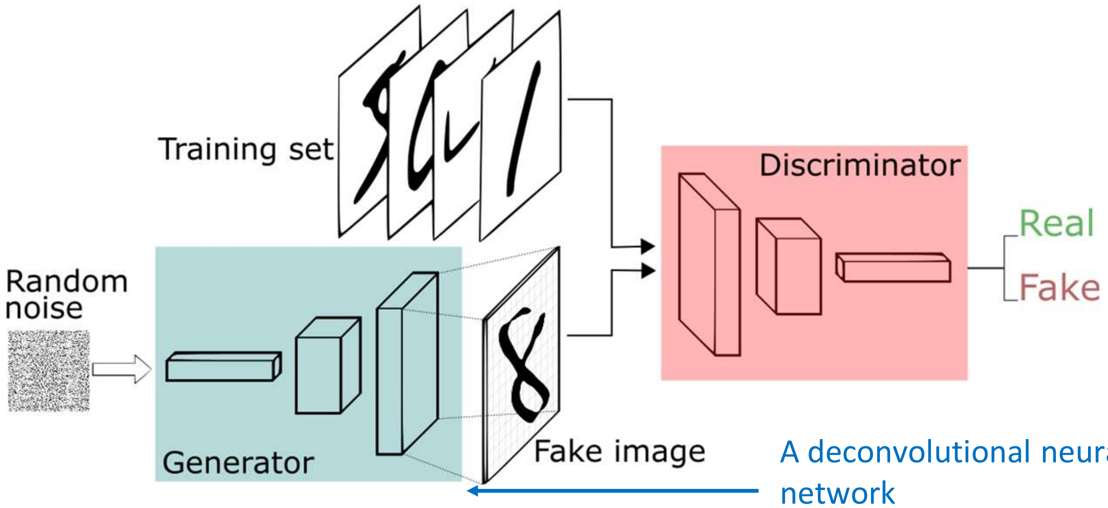


Generating Image with GANs

A standard CNN and a binomial classifier



Generating Image with GANs

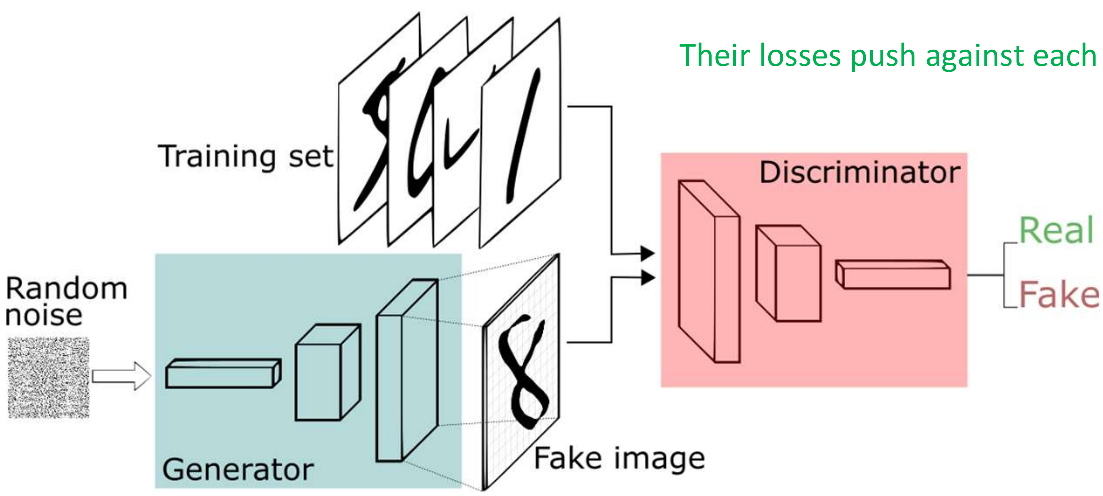


A deconvolutional neural network

Generating Image with GANs

Both nets are trying to optimize a different and opposing objective function in a zero-sum game.

Their losses push against each other.



Generating Image with GANs

The generator takes in a random noise vector and returns an image.

This synthetic image is fed into the discriminator alongside a stream of images taken from the real image dataset.

The discriminator takes in both real and fake (synthetic) images and returns probabilities, a number between 0 and 1, with 1 representing a prediction of authenticity and 0 representing fake.

GAN - “Robotic Artist”

https://www.theverge.com/2018/10/23/18013190/ai-art-portrait-auction-christies-belamy-obvious-robbie-barrat-gans

Training Generative Adversarial Networks

Goal – Find a setting of parameters that makes generated data look like the training data to the discriminator network.

Discriminator training - Backprop from a binary classification loss.

Generator training - Backprop the negation of the binary classification loss of the discriminator.

Training Generative Adversarial Networks

Alternately updating the parameters θ_G and θ_D of the generator G and the discriminator D.

The discriminator is a neural network with *d*-dimensional inputs and a single output in $(0, 1)$, which indicates the probability whether or not the *d*-dimensional input example is real.

A value of 1 indicates that the example is real and a value of 0 indicates that the example is fake (synthetic).

The objective for the discriminator is to correctly classify the real examples to a label of 1 and the synthetically generated examples to a label of 0.

Training Generative Adversarial Networks

The generator takes noise samples from a *p*-dimensional probability distribution as input and uses those to generate *d*-dimensional examples of the data.

The discriminator error is used to train the generator to create other samples like coming from the real data distribution.

The objective for the generator is to generate examples so that they fool the discriminator (i.e., encourage the discriminator to label such examples as 1).

Training Discriminator

The objective function of the discriminator:

$$J_D = \sum_{x \in R_m} \log\,[D(x)] + \sum_{x \in S_m} \log[1 - D(x)]$$

R_m is the set of m randomly sampled examples from the real data set.

S_m is the set of m generated synthetic samples.

Maximization for the discriminator:

$$\text{Maximize}_D\, J_D$$

Training Generator

The objective function of the generator:

$$J_G = \sum_{x \in S_m} \log[1 - D(x)] = \sum_{z \in N_m} \log[1 - D(G(z))]$$

N_m is the set of m input samples $\{z_m\}$.

Minimization for the generator:

$$\text{Minimize}_G \, J_G$$

Training Generative Adversarial Networks

This is a two-person zero-sum minimax game, which has an inner maximization by D and an outer minimization by G.

$$\min_G \max_D V(D, G)$$

$$V(D, G) = \mathbb{E}_{x \sim p_{data}(x)}[\log D(x; \theta_D)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z; \theta_G); \theta_D))]$$

Training Generative Adversarial Networks

Theoretical Results (given enough capacity and non-parametric)

For fixed G, the optimal discriminator D is

$$D_G^*(x) = \frac{p_{data}(x)}{p_{data}(x) + p_{model}(x; \theta_G)}$$

Training Generative Adversarial Networks

Theoretical Results (given enough capacity and non-parametric)

Find the global minimum w.r.t G for the optimal discriminator D

$$\begin{aligned} C(G) &= \mathbb{E}_{x \sim p_{data}} \left[\log \frac{p_{data}}{p_{data} + p_{model}} \right] + \mathbb{E}_{x \sim p_{model}} \left[\frac{p_{model}}{p_{data} + p_{model}} \right] \\ &= -\log(4) + KL \left(p_{data} \middle\| \frac{p_{data} + p_{model}}{2} \right) + KL \left(p_{model} \middle\| \frac{p_{data} + p_{model}}{2} \right) \\ &= -\log(4) + 2 \cdot JSD(p_{data} \| p_{model}) \end{aligned}$$

The Jensen-Shannon divergence (JSD) between two distributions is non-negative and zero *iff* the distributions are equal.

Therefore, the unique global minimum is $C(G) = -\log 4$, when $p_{data} = p_{model}$.

Training Generative Adversarial Networks

Stochastic gradient ascent is used for learning the parameters θ_D of the discriminator.

Stochastic gradient descent is used for learning the parameters θ_G of the generator.

The gradient update steps are alternated between the generator and the discriminator.

k steps of optimizing D and one step of optimizing G

– To maintain D near its optimal solution while G changes slowly.

Training Generative Adversarial Networks

At the discriminator:

- (Repeat $k < 5$ times): A mini-batch of size $2m$ is constructed with an equal number of real and synthetic examples.
- Stochastic gradient ascent is performed on the parameters of the discriminator so as the maximize the likelihood that the discriminator correctly classifies both the real and synthetic examples.
- For each update step, performing backpropagation on the discriminator network with respect to the mini-batch of $2m$ real/synthetic examples.

Training Generative Adversarial Networks

At the Generator:

- Provide the generator with m noise inputs so as to create m synthetic examples (current mini-batch).
- Stochastic gradient descent is performed on the parameters of the generator so as to minimize the likelihood that the discriminator correctly classifies the synthetic examples.
- Even though the discriminator is connected to the generator, the gradient updates (during backpropagation) are performed with respect to the parameters of only the generator network.

Training Generative Adversarial Networks

This iterative process is repeated to convergence until *Nash equilibrium* is reached. At this point, the discriminator will be unable to distinguish between the real and synthetic examples.

The training of the generator and discriminator are done simultaneously with interleaving.

The generator may produce poor samples in early iterations and therefore $D(G(z))$ will be close to 0. In this case, we can train G to maximize $\log D(G(z))$ instead of minimizing $\log(1 - D(G(z)))$ during the early stages.

GAN Compared to Variational Autoencoder

A GAN is not designed to reconstruct specific input samples like a variational AE.

However, both models can generate images like the base data, because the hidden space has a known structure (typically Gaussian) from which points can be sampled.

In general, the GAN produces samples of better quality (e.g., less blurry images) than a variational AE. This is because the adversarial approach is specifically designed to produce realistic images, whereas the regularization of the variational AE actually hurts the quality of the generated objects.

Major Problems of GANs

GANs do not naturally have a metric for convergence. Networks are difficult to converge on large problems.

Ideally, all losses go to $-\log\left(\frac{1}{2}\right) \approx 0.69$. But that usually does not happen in practice.

Generator and Discriminator reach some desired equilibrium but this is rare.

Common Failure Cases

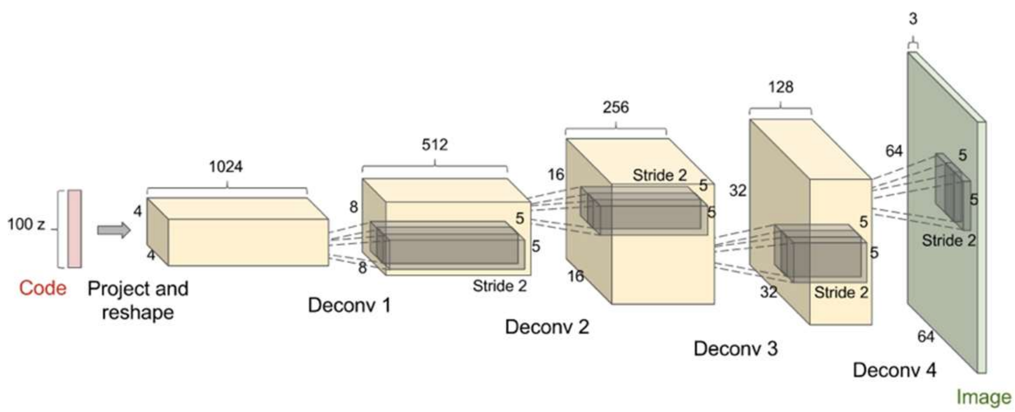
The discriminator becomes too strong too quickly and the generator ends up not learning anything.

Mode Collapse – The generator learns only a very small subset of the true data distribution. It produces only one mode of data distribution.

Vanishing/Exploding gradients from the discriminator to the generator.

The generator learns very specific weaknesses of the discriminator. It produces garbage that fools the discriminator.

Deep Convolutional GAN (DCGAN)



(Radford *et al.*, 2015)

Deep Convolutional GAN (DCGAN)

Fully connected layers are not used in either the discriminator or the generator.

Replace pooling layers with strided convolutions (discriminator) and fractional-strided convolutions (generator).

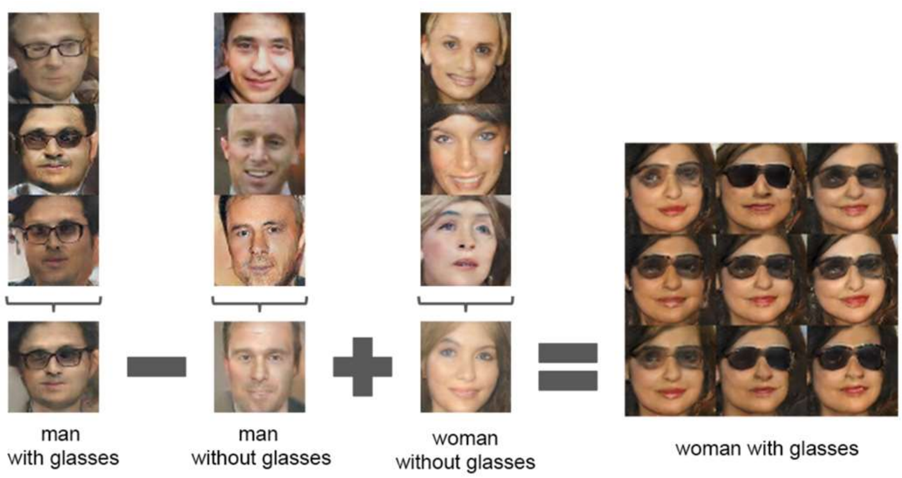
Batch normalization is used in order to reduce any problems with the vanishing and exploding gradient problems.

The generator uses ReLU activation for all layers expect for the output (Tanh).

The discriminator uses a convolutional neural network architecture, except that the leaky ReLU is used instead of the ReLU.

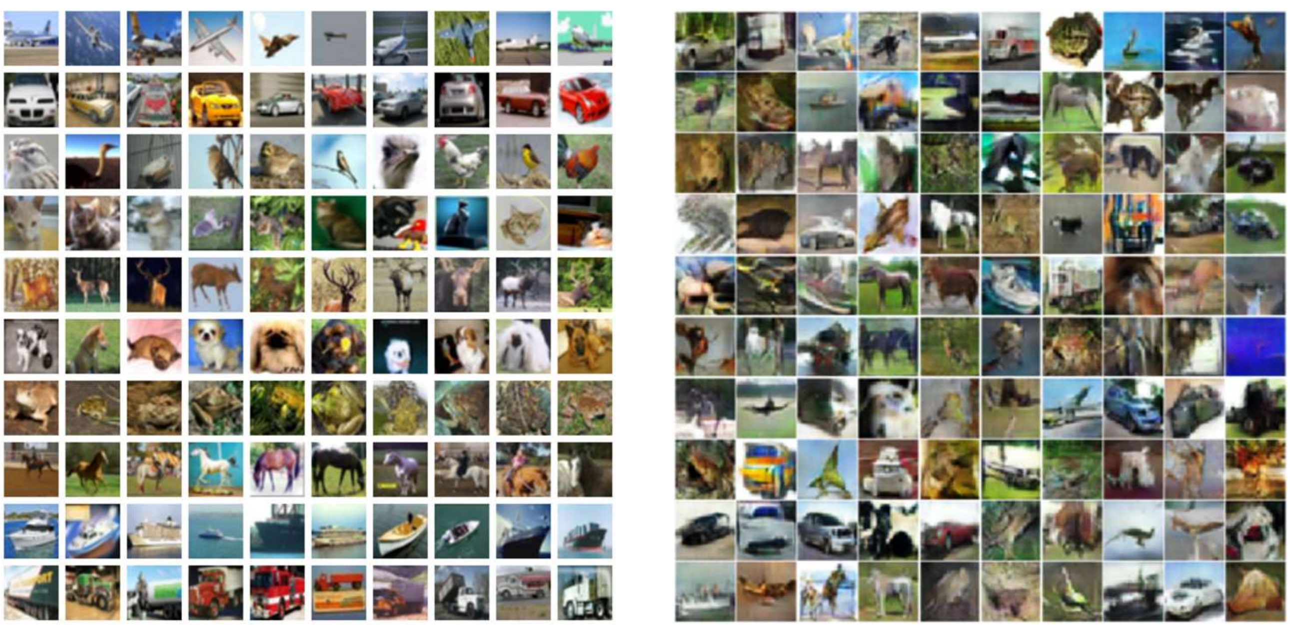
The final convolutional layer of the discriminator is flattened and fed into a single sigmoid output.

Deep Convolutional GAN (DCGAN)



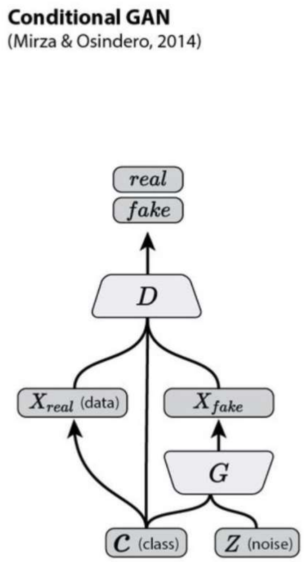
Radford, Alec, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. 2015.

Generated bedrooms. Source: “Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks”, 2016. https://arxiv.org/abs/1511.06434v2



Original CIFAR-10 vs. Generated CIFAR-10 samples
Source: “Improved Techniques for Training GANs”, 2016. https://arxiv.org/abs/1606.03498

Conditional GAN



Idea: Leverage side information to produce better quality or conditional samples.

In conditional GANs, both the generator and the discriminator are conditioned on an additional input, which can be a class label, a caption, or another object of the same type.

Force G to generate a particular type of output.

The generator learns side-information conditional distributions, as it is able to disentangle this from the overall latent space.

Conditional GAN

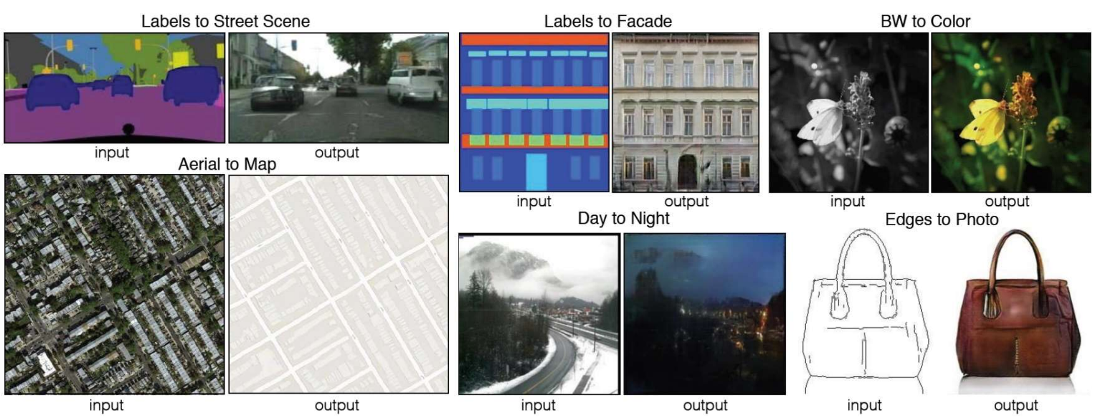
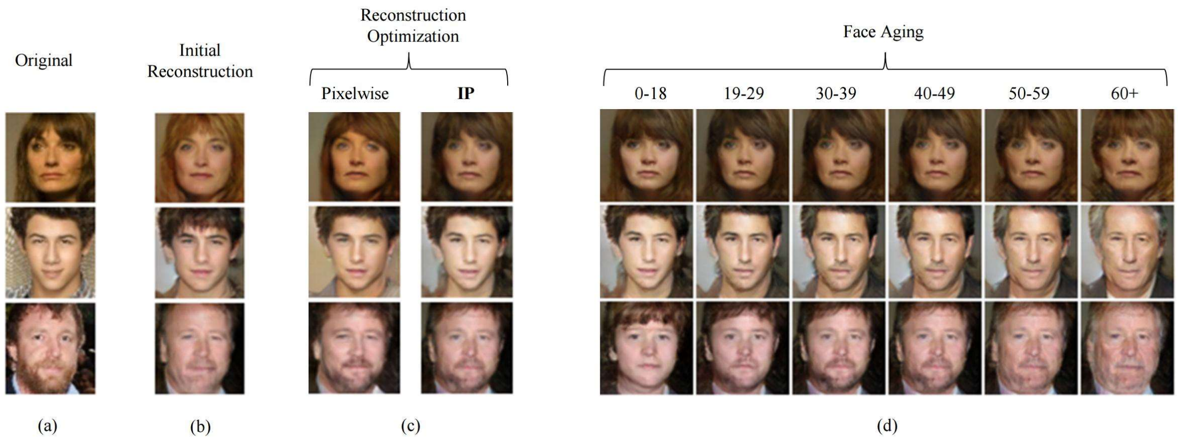


Image-to-Image Translation, pix2pix

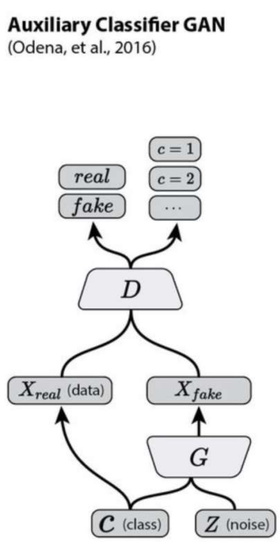
Phillip Isola, et al. Image-to-Image Translation with Conditional Adversarial Networks, CVPR 2017.

Conditional GAN



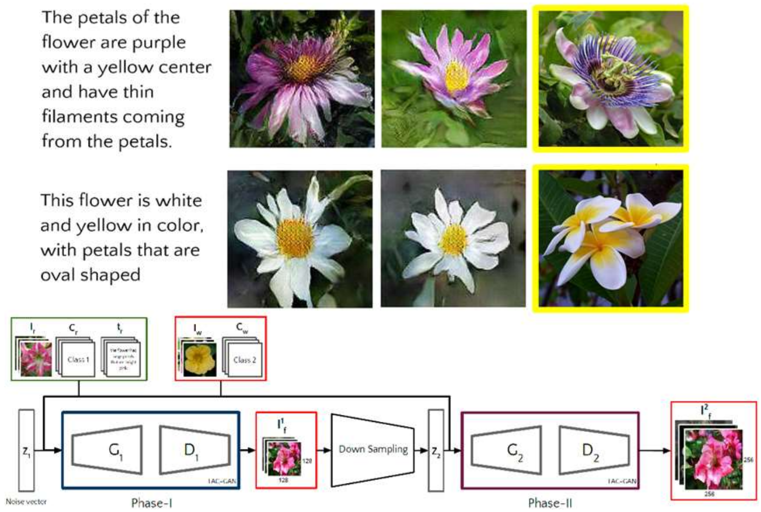
Antipov, G., Baccouche, M., & Dugelay, J. L., Face Aging With Conditional Generative Adversarial Networks, 2017.

Auxiliary Classifier GAN



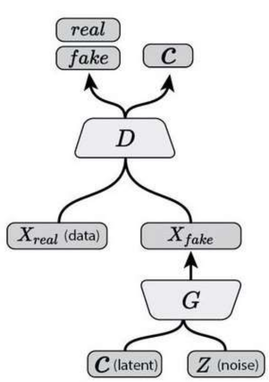
Discriminator is tasked with jointly learning real-vs-fake and the ability to reconstruct the latent variable being passed in.

Auxiliary Classifier GAN



Ayushman Dash, et al. TAC-GAN - Text Conditioned Auxiliary Classifier Generative Adversarial Network, 2017.

InfoGAN



z vector captures slight variations in the object.

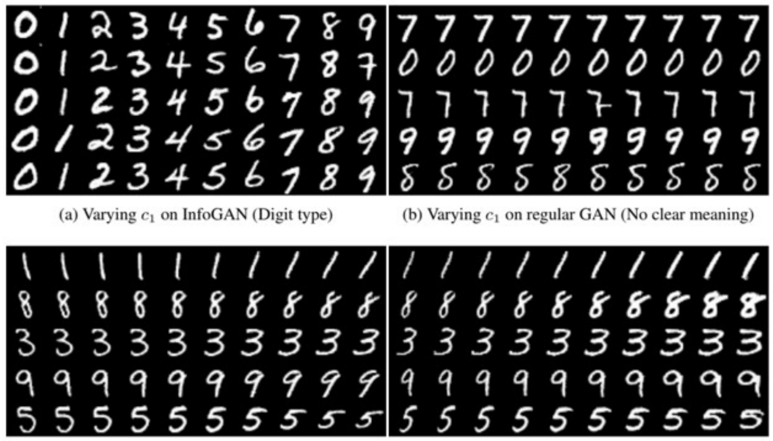
c vector captures the main attributes of the object.

Want to maximize the mutual information I between c and $x = G(z, c)$.

$$\min_G \max_D V(D, G) - \lambda I(c; G(z, c))$$

Chen, Xi, *et al*. InfoGAN: Interpretable representation learning by information maximizing generative adversarial nets. NIPS, 2016.

InfoGAN



(a) Varying c_1 on InfoGAN (Digit type) (b) Varying c_1 on regular GAN (No clear meaning)

(c) Varying c_2 from -2 to 2 on InfoGAN (Rotation) (d) Varying c_3 from -2 to 2 on InfoGAN (Width)

Chen, Xi, *et al*. InfoGAN: Interpretable representation learning by information maximizing generative adversarial nets. NIPS, 2016.

Variants of GANs

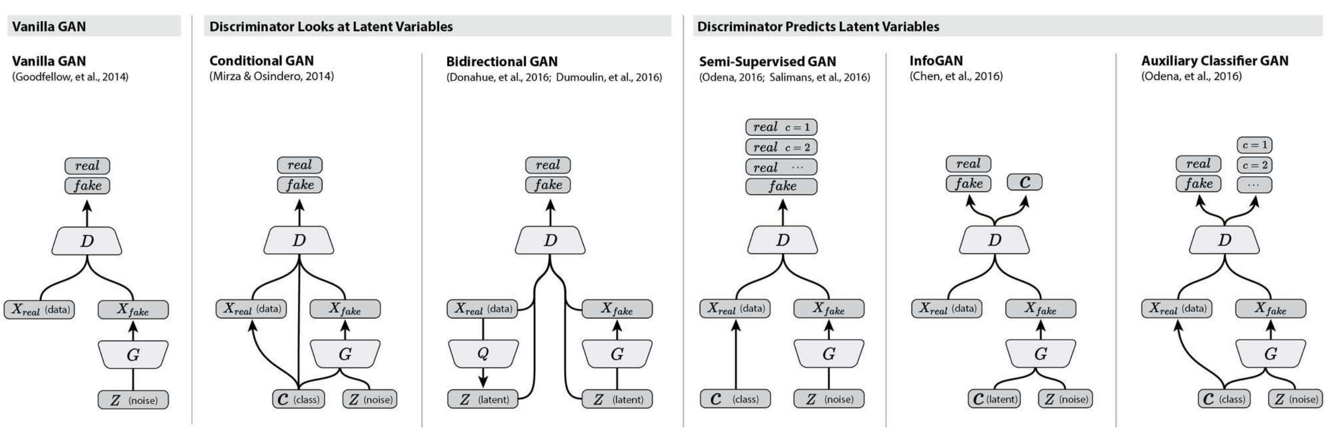
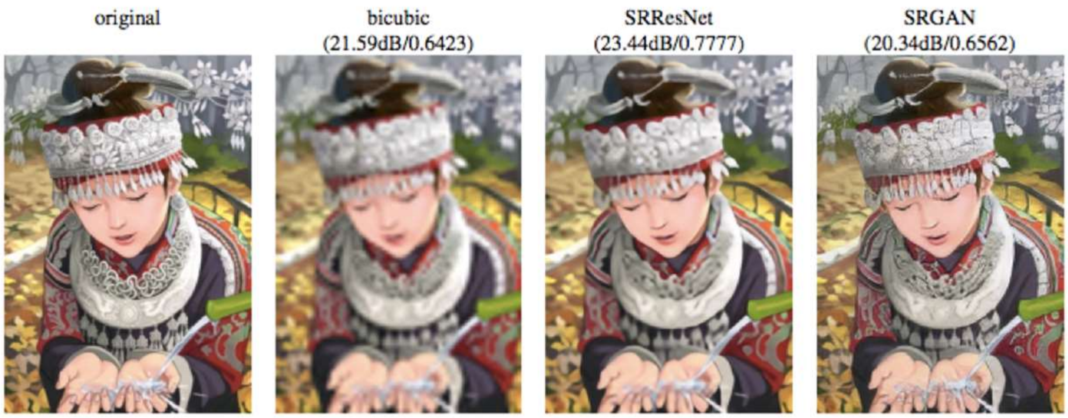


Image Super-Resolution with GAN



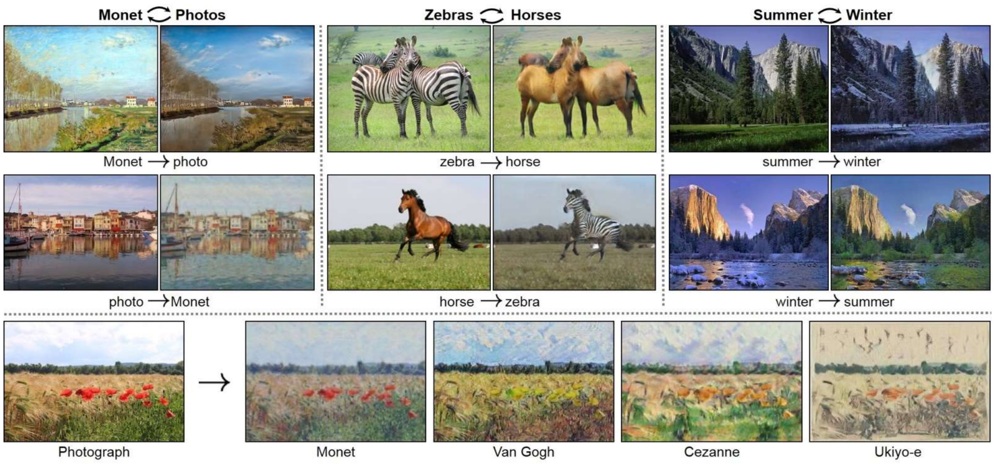
Ledig, Christian, *et al*. Photo-realistic single image super-resolution using a generative adversarial network. CVPR 2016.

Fashion Recommendation and Design with GANs

Find latent representation z that obtains the highest recommendation score.

Kang *et al.*, Visually-Aware Fashion Recommendation and Design With GANs, 2017.

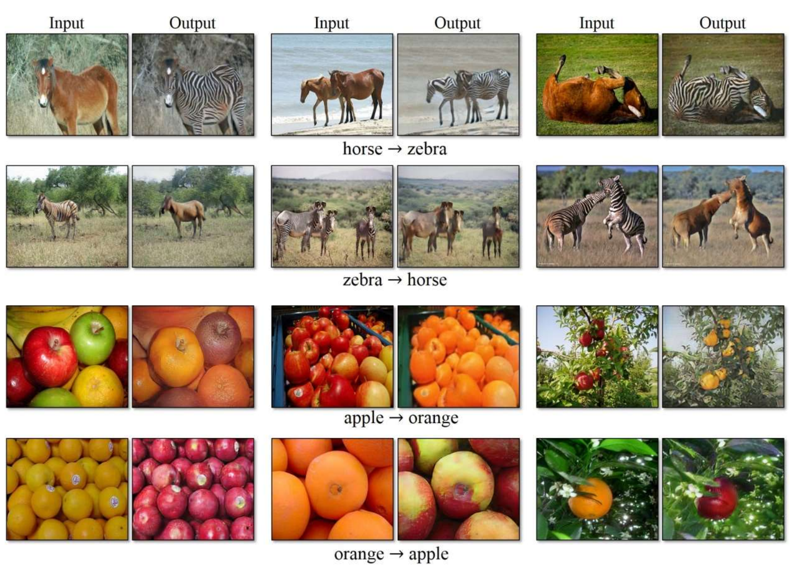
Cycle GANs



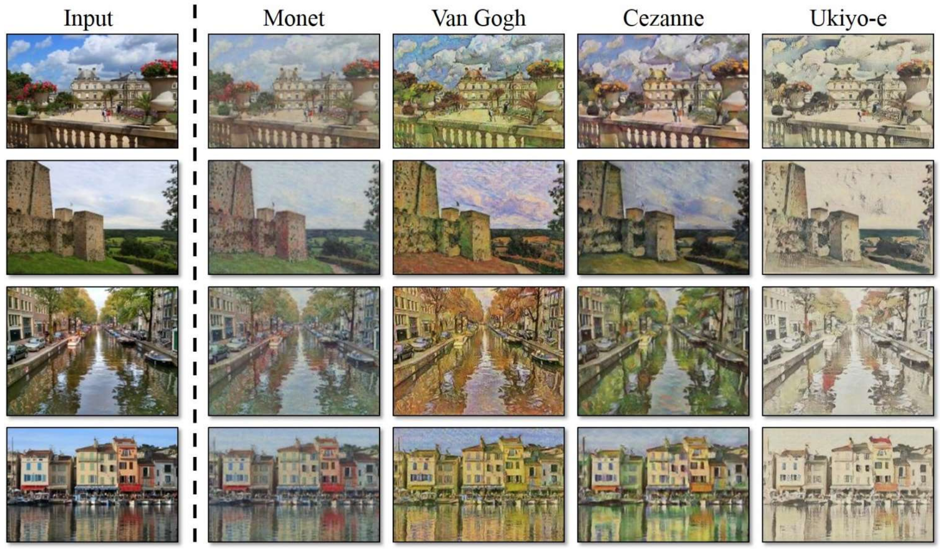
Given two image collections, algorithm learns to translate an image from one collection to the other.

Jun-Yan Zhu et al. Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks, ICCV 2017.

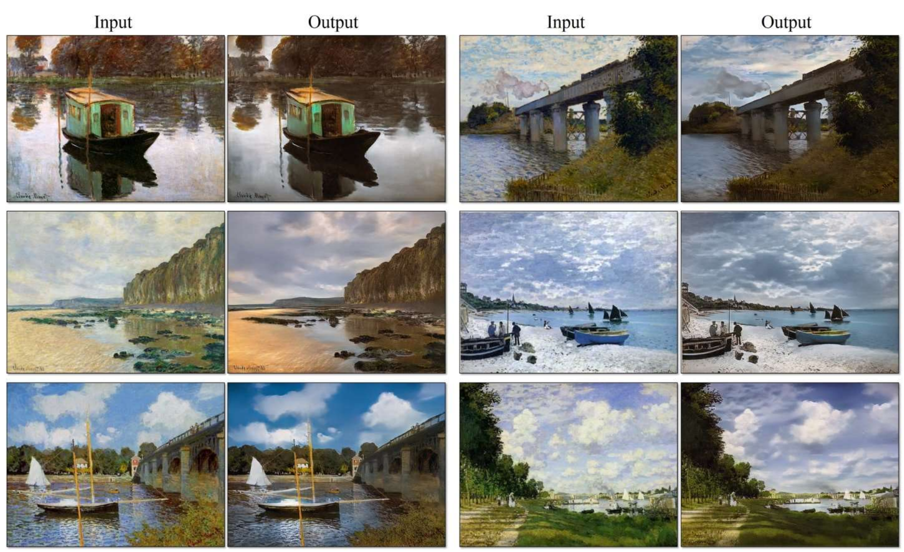
Cycle GANs



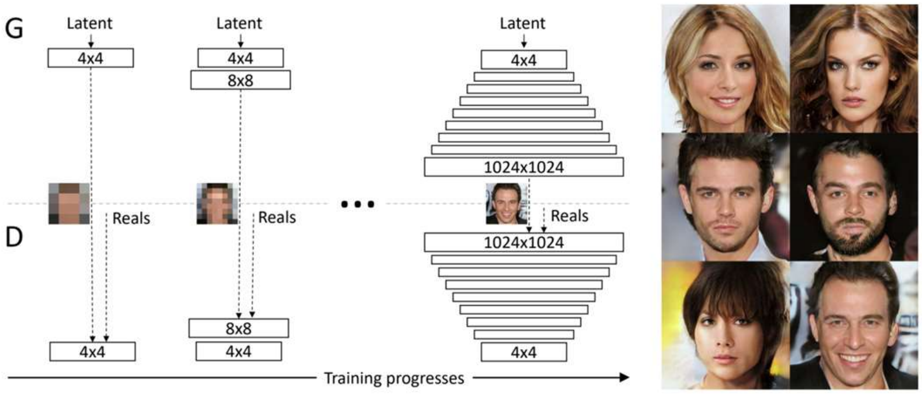
Photos to Paintings



Paintings to Photos



Progressive Growing of GANs



Images are generated by walking through the latent space.

Karras, Tero, *et al*. Progressive growing of GANs for improved quality, stability, and variation. 2017.