GAN
Conditional GAN
Algo GAN generator G, discriminator D

for number of training iterations do
    for k steps do
        • Sample minibatch of $m$ noise samples $\{z^{(1)}, \ldots, z^{(m)}\}$ from noise prior $p_g(z)$.
        • Sample minibatch of $m$ examples $\{x^{(1)}, \ldots, x^{(m)}\}$ from data generating distribution $p_{data}(x)$.
        • Update the discriminator by ascending its stochastic gradient:
          \[
          \nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[ \log D(x^{(i)}) + \log \left(1 - D\left(G\left(z^{(i)}\right)\right)\right) \right].
          \]
    end for
    • Sample minibatch of $m$ noise samples $\{z^{(1)}, \ldots, z^{(m)}\}$ from noise prior $p_g(z)$.
    • Update the generator by descending its stochastic gradient:
      \[
      \nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log \left(1 - D\left(G\left(z^{(i)}\right)\right)\right).
      \]
end for

The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

Which architectures for G and D?
Basic Archi for G and D and expe

Models
\( G \) and \( D \) fully connected nets or convolutional for \( D \), (de)convolutional for \( G \) (as seen for segmentation nets)
ReLU and/or sigmoids, dropout

Datasets
MNIST, Toronto Face Database, CIFAR-10

GAN - Evaluation

- Approximate \( p_g \) by fitting a Gaussian Parzen window on the generated images.
- Cross-validate \( \sigma \) to maximize likelihood of validation set
- Compute the likelihood of the test set

Evaluation not trivial, can be done using generated images as inputs for deep nets \( \Rightarrow \) inception scores
Figure: Right col nearest from dataset. a) MNIST, b) TFD, c) CIFAR-10 (fully connected), d) CIFAR-10 (convolutional D, deconvolutional G)
GAN - Qualitative results 2/2

Figure: Linear interpolation between 2 points in z space

- Advantages:
  - Computational advantages (no complex likelihood inference)
  - Can represent sharper distributions

- Disadvantages:
  - $G$ and $D$ must be well synchronized for the algorithm to converge correctly
GAN architectures

• How to improve result quality?
  • Spatial resolution
    ⇒ Cascade of GAN
  • Object quality
    ⇒ Progressive growing of spatial resolution in G
Laplacian Pyramid GANs (LAPGANs)

- GANs do not work well for complex / high level / natural images.
- Idea: decompose the generation in successive tasks using Laplacian Pyramid (of GANs)

Let \( d(l) \) and \( u(l) \) be down-sampling and up-sampling operations. Gaussian pyramid:

\[
G(l) = [l_0, l_1, \ldots, l_K], \quad l_k = d^{(k)}(l)
\]

Laplacian pyramid:

\[
h_k = L_k(l) = l_k - u(l_{k+1})
\]

Reconstruction:

\[
l_k = u(l_{k+1}) + h_k
\]
LAPGAN model - sampling

- Set of generative convnets: $G_0, ..., G_K$
- Generated details: $\tilde{h}_k = G_k(z_k, u(\tilde{l}_{k+1}))$
- Reconstructed image: $\tilde{l}_k = u(\tilde{l}_{k+1}) + \tilde{h}_k$ ($\tilde{l}_{K+1} = 0$)
LAPGAN model - training

- Low-pass version of $l_0$: $l_0 = u(d(l_0)$
- Either:
  - High-pass version of $l_0$: $h_0 = l_0 - l_0$
  - Generate $\tilde{h}_0 = G_0(z_0, l_0)$
- Forward $D_0(l_0 + h_0$ or $\tilde{h}_0$)
- Backward $D_0$ and $G_0$
- $G_K$ and $D_K$ are trained as a simple GAN
LAPGAN model - Experiments

- **Datasets:** CIFAR-10, STL
- **Initial scale:**
  - $G_K$ and $D_K$ have 2 hidden layers and ReLU
  - $z_K \sim U_{[-1,1]^{100}}$
  - Trained as GAN
- **Subsequent scales:**
  - $G_k$ and $D_k$ convnets with 3 and 2 layers
  - $z_k \sim U_{[-1,1]|k|}$ ("color" layer)
  - Trained as CGAN
LAPGAN model - Results - CIFAR
Figure 4: STL samples: (a) Random 96x96 samples from our LAPGAN model. (b) Coarse-to-fine generation chain.
LAPGAN model - Results - LSUN
LAPGAN

• Good idea but Genetor structure too weak
• Recent improvement: Pix2PixHD described later
Progressive growing of spatial resolution in G

Deep Convolutional GANs (DCGANs)

GANs are hard to scale => Identify a family of architectures that gives stable training

- Replace any pooling layers with strided convolutions (discriminator) and fractional-strided convolutions (generator)
- Use batchnorm in both the generator and the discriminator
- Remove fully connected hidden layers for deeper architectures
- Use ReLU activation in generator for all layers except for the output, which uses Tanh
- Use LeakyReLU activation in the discriminator for all layers
Progressive growing of spatial resolution in G: DCGAN

Upsampling step by step

Combine with convolutional layers
Conditional GAN
Motivation

Prior distribution $z$

c: train

Generator

Text

this white and yellow flower have thin white petals and a round yellow stamen

Generator

Image

Image

Generator

Image
Conditional GAN

- **Text to image** by traditional supervised learning

1: a dog is running \( \hat{x}^1 \)

2: a bird is flying \( \hat{x}^2 \)

Text: “train”

Target of NN output

A blurry image!

A blurry image!
Conditional GAN

\[ x = G(c, z) \]

Text: “train”

A blurry image!
Conditional GAN

Prior distribution $z$ $\rightarrow$ $G$ $\rightarrow$ Image $x = G(c,z)$

$c$: train $\rightarrow$ $D$ (type 1) $\rightarrow$ scalar $\rightarrow$ $x$ is realistic or not

Positive example: (train, Image)

Negative example: (train, Image)

$D$ (type 2) $\rightarrow$ scalar $\rightarrow$ $x$ is realistic or not + $c$ and $x$ are matched or not

Positive example: (train, Image)

Negative example: (cat, Image)
Conditional GAN (CGAN model)

$$\min_G \max_D \left( \mathbb{E}_{x,y \sim p_{data}(x,y)} \left[ \log D(x, y) \right] + \mathbb{E}_{y \sim p_y, z \sim p_z(z)} \left[ \log (1 - D(G(z, y), y)) \right] \right)$$
Applications

• Text2Im
• Im2Im
• Unpaired transformation
• ...

Text2Image

This flower has small, round violet petals with a dark purple center

$\varphi \xrightarrow{\varphi(t)} \hat{x} := G(z, \varphi(t))$

$z \sim \mathcal{N}(0, 1)$

Generator Network

Discriminator Network

- **Positive samples:**
  - real image + right texts

- **Negative samples:**
  - fake image + right texts
  - Real image + wrong texts

[Reed et al. ICML 2016]
this small bird has a pink breast and crown, and black primaries and secondaries.

the flower has petals that are bright pinkish purple with white stigma

this magnificent fellow is almost all black with a red crest, and white cheek patch.

this white and yellow flower have thin white petals and a round yellow stamen
<table>
<thead>
<tr>
<th>Caption</th>
<th>Image</th>
</tr>
</thead>
<tbody>
<tr>
<td>this flower has white petals and a yellow stamen</td>
<td>![Images]</td>
</tr>
<tr>
<td>the center is yellow surrounded by wavy dark purple petals</td>
<td>![Images]</td>
</tr>
<tr>
<td>this flower has lots of small round pink petals</td>
<td>![Images]</td>
</tr>
</tbody>
</table>
StackGAN: similar idea with LapGan

Figure 2. The architecture of the proposed StackGAN. The Stage-I generator draws a low-resolution image by sketching rough shape and basic colors of the object from the given text and painting the background from a random noise vector. Conditioned on Stage-I results, the Stage-II generator corrects defects and adds compelling details into Stage-I results, yielding a more realistic high-resolution image.

[Zhang et al. 2016]
StackGAN

This bird has a yellow belly and tarsus, grey back, wings, and brown throat, nape with a black face
This bird is white with some black on its head and wings, and has a long orange beak
This flower has overlapping pink pointed petals surrounding a ring of short yellow filaments

(a) Stage-I images

(b) Stage-II images

[Zhang et al. 2016]
<table>
<thead>
<tr>
<th>Caption</th>
<th>Image</th>
</tr>
</thead>
<tbody>
<tr>
<td>a pitcher is about to throw the ball to the batter</td>
<td><img src="image1.png" alt="Images" /></td>
</tr>
<tr>
<td>a group of people on skis stand in the snow</td>
<td><img src="image2.png" alt="Images" /></td>
</tr>
<tr>
<td>a man in a wet suit riding a surfboard on a wave</td>
<td><img src="image3.png" alt="Images" /></td>
</tr>
</tbody>
</table>
Image-to-Image Translation \textit{pix2pix}

- Conditioned on an image of different modality
- No need to specify the loss function

\cite{Isola_2017_CVPR}
Image-to-image pix2pix

\[ x = G(z | c) \]
Image-to-image pix2pix

- Traditional supervised approach

Testing:

It is blurry because it is the average of several images.
Image-to-image

• Conditional GAN

Testing:

input close GAN GAN + close GT
Positive examples

Real or fake pair?

D

G tries to synthesize fake images that fool D

D tries to identify the fakes

Negative examples

Real or fake pair?

D

G

[Isola et al. CVPR 2017]
Label2Image

Input  Ground truth  L1  cGAN  L1 + cGAN

[Isola et al. CVPR 2017]
Edges2Image

[Isola et al. CVPR 2017]
Pix2pixHD [CVPR 2018]

High-Resolution Image Synthesis and Semantic Manipulation with Conditional GANs
Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, Bryan Catanzaro
Pix2pixHD [CVPR 2018]

Coarse-to-fine Generator

Multi-scale Discriminators

Robust Objective

Residual blocks

Residual blocks
Pix2pixHD [CVPR 2018]

Our Method

- Boundary improvement
Feature Embedding: Inference

Image generation network

Feature encoder network

Instance-wise average pooling

Labels

Features

Reals

Features

Synthesized

F

Features

F

Features
Results

• Qualitative comparisons
Unpaired Transformation
Unpaired Transformation - Cycle GAN, Disco GAN

Transform an object from one domain to another without paired data

paired data
Cycle GAN

https://arxiv.org/abs/1703.10593
https://junyanz.github.io/CycleGAN/

Domain X

\[ G_{X\rightarrow Y} \]

Become similar to domain Y

Not what we want

ignore input

Domain Y

\[ D_Y \rightarrow \text{scalar} \]

Input image belongs to domain Y or not
Cycle GAN

Domain X

Domain Y

Cycle GAN

as close as possible

$G_{X\rightarrow Y}$

$G_{Y\rightarrow X}$

$Lack of information for reconstruction$

Input image belongs to domain Y or not

Input image belongs to domain Y or not

Scalar $D_Y$
Cycle GAN

\[ G_{X \rightarrow Y} \xrightarrow{\text{as close as possible}} D_X \xrightarrow{\text{as close as possible}} G_{Y \rightarrow X} \]

\[ G_{Y \rightarrow X} \xrightarrow{\text{as close as possible}} D_Y \xrightarrow{\text{as close as possible}} G_{X \rightarrow Y} \]

scalar: belongs to domain X or not

scalar: belongs to domain Y or not
Results -- Cycle GAN

photo  ➔  van Gogh  ➔  Domain X  ➔  Domain Y
Video GAN


[Vondrick et al. NIPS 2016]
Shape modeling using 3D Generative Adversarial Network

[Wu et al. NIPS 2016]
GAN for Inpainting
Inpainting task

- Complete the missing part
Inpainting as unsupervised learning with GAN loss

Reconstruct missing pixels by decoding using context
Loss defined on the predicted patch and the real one (known at training time)
First proposition -- Architecture

- Architecture: Encoder/Fully connected/Decoder

- DC-GAN for inpainting task
- Input: 227 $\times$ 227 $\times$ 3 image
- Output: encoder context features (6 $\times$ 6 $\times$ 256)
Channel-wise fully-connected layer

- **Input / output:** $6 \times 6 \times 256$ channels
- **First layer:** Channel-wise fully-connected (each $6 \times 6$ input connected to the corresponding $6 \times 6$ output)
- **Second layer:** Stride 1 convolution to mix channels

![Diagram of channel-wise fully-connected layer and decoder](image)

Decoder

- **Architecture:** Same as DC-GAN: 5 up-convolutional layers ("deconv" + ReLU)
- **Input:** decoder context features $6 \times 6 \times 256$
- **Output:** $227 \times 227 \times 3$ image
Training: Masking the images

- **How to define the mask?**
  - Center region of the image
  - Random regions (chosen solution)
  - Random segmentation mask from VOC (said to be equivalent to random regions)

- **Formal definition:** Defined by a mask $\hat{M} \in \{0, 1\}^{227 \times 227}$ with 1 if the pixel should be masked
Training: Loss - Overview

- Trained completely from scratch to fill-up the masked areas
- **Problem**: multiple plausible solutions
- **Solution**: combining 2 losses:
  - \( \mathcal{L}_{\text{rec}} \) **L2 reconstruction loss**: learn the structure of the missing region (average multiple modes in prediction)
  - \( \mathcal{L}_{\text{adv}} \) **Adversarial loss**: make it look real (pick a mode from the distribution)

\[
\min_F \mathcal{L} = \lambda_{\text{rec}} \mathcal{L}_{\text{rec}} + \lambda_{\text{adv}} \mathcal{L}_{\text{adv}}
\]

\[
\mathcal{L}_{\text{rec}}(x) = \left\| \hat{M} \odot \left( x - F((1 - \hat{M}) \odot x) \right) \right\|_2
\]

\[
\mathcal{L}_{\text{adv}} = \max_D \mathbb{E}_{x \in \chi} \left[ \log(D(x)) + \log \left( 1 - D(F((1 - \hat{M}) \odot x)) \right) \right]
\]

- Rq: The encoder-decoder is the generator, D is a CNN
Results

Dataset: StreetView Paris and ImageNet
Semantic inpainting - Qualitative results
Generalizing inpainting: missing data encoder
Adding perceptual loss, BB regression loss

\[ r^c = \left( \frac{x^c}{W}, \frac{y^c}{H}, \frac{x^c + w^c}{W}, \frac{y^c + h^c}{H} \right) \]

\[ \mathcal{L}^{HnS}_{\text{disc}}(\theta_d) = \frac{1}{N} \sum_{i=1}^{N} \sum_{c=1}^{C} \| r_i^c - \hat{r}_i^c(\theta_g, \theta_d) \| \]

\[ \mathcal{L}^{HnS}_{\text{gen}}(\theta_g) = \frac{1}{N} \sum_{i=1}^{N} \sum_{c=1}^{C} \| q_i^c - \hat{r}_i^c(\theta_g, \theta_d) \| \]

\[ \mathcal{L}_{\text{tot}}(\theta_g, \theta_d) = \mathcal{L}_{\text{rec}}(\theta_g) + \lambda_{\text{compl}} \mathcal{L}_{\text{compl}}^{vgg}(\theta_g) + \lambda_{\text{adv}} \mathcal{L}_{\text{adv}}(\theta_g, \theta_d) + \lambda_{HnS} \mathcal{L}_{\text{coord}}^{HnS}(\theta_g, \theta_d) \]
Results
Results