Image Synthesis
Why generative modeling?

• Understanding the visual world
• Generating samples for learning
• Unsupervised learning
• Graphics and image processing
• Art and creativity
Synthesis for understanding - Analysis by synthesis

Hypothesize and generate

\[ P(y \mid x) = \frac{P(x, y)}{P(x)} \]

Slide adapted from Bruno Olshausen
Generating samples for learning
Graphics and image processing
We have been doing generation for a while...

Perlin noise
We have been doing generation for a while...
We have been doing generation for a while...
Learning generative models of images

- **Standard probabilistic models** (e.g. Gaussian)
  - Can be sampled from
  - Cannot capture arbitrary distributions
- **Neural networks**
  - Can learn to model arbitrary functions
  - Are deterministic
- Key idea 1: neural networks with noise input!
Learning generative models of images
Learning generative models of images

• Two distributions: $p_{\text{real}}$ and $p_{\text{model}}$
• Both highly multimodal, high-dimensional
• Only have samples from each distribution
• Want to minimize the difference between the two distributions
Computing the difference between two distributions

• Idea 1: parzen window estimation
Computing the difference between two distributions

• Idea 1: parzen window estimation
• Plop a Gaussian at each sample to convert samples into a distribution
Computing the difference between two distributions

• Idea 1: parzen window estimation
• Problem: calculating distances an issue in high dimensional spaces
• Need dense sampling
Computing the difference between two distributions

• Idea 1: parzen window estimation
• Problem: calculating distances an issue in high dimensional spaces
• Need dense sampling
The adversary

• Train a classifier to see if the distributions are different!
The adversary

• Essentially the same idea as:
  • Maximum mean discrepancy (MMD)
  • $\mathcal{A}$-distance
Generative Adversarial Nets
Generative Adversarial Nets

• Perspective 1: Two player game
  • Discriminator’s job to distinguish model samples from real thing
  • Generator’s job to fool the discriminator

• Perspective 2: Density estimation

\[
D(x) = \frac{p_{data}(x)}{p_{model}(x) + p_{data}(x)}
\]

• Discriminator estimates density and tells generator which samples are unlikely
• Generator learns to sample
Generative Adversarial Nets

$$\max_G \min_D ( - \mathbb{E}_x \log D(x) - \mathbb{E}_z \log(1 - D(G(z))))$$

- Training:
  - Sample x
  - Sample z
  - Take step(s) along discriminator
  - Take step(s) along generator
Generative Adversarial Nets

\[
\max_G \min_D \left( - \mathbb{E}_x \log D(x) - \mathbb{E}_z \log(1 - D(G(z))) \right)
\]

- Usually, discriminator’s task is easier
- \(D(G(z))\) often close to 0
Generative Adversarial Nets

\[ J_D(x) = -\mathbb{E}_x \log D(x) - \mathbb{E}_z \log(1 - D(G(z))) \]

\[ J_G(x) = -\mathbb{E}_z \log D(G(z)) \]
Generative Adversarial Networks (GANs)
Generative adversarial networks

What do GANs generate?

GAN variants

Table 1: Generator and discriminator loss functions. The main difference whether the discriminator outputs a probability (MM GAN, NS GAN, DRAGAN) or its output is unbounded (WGAN, WGAN GP, LS GAN, BEGAN), whether the gradient penalty is present (WGAN GP, DRAGAN) and where is it evaluated. We chose those models based on their popularity.

<table>
<thead>
<tr>
<th>GAN</th>
<th>DISCRIMINATOR LOSS</th>
<th>GENERATOR LOSS</th>
</tr>
</thead>
<tbody>
<tr>
<td>MM GAN</td>
<td>$L_D^{GAN} = -\mathbb{E}<em>{x \sim p_d}[\log(D(x))] - \mathbb{E}</em>{\hat{x} \sim p_g}[\log(1 - D(\hat{x}))]$</td>
<td>$L_G^{GAN} = \mathbb{E}_{\hat{x} \sim p_g}[\log(1 - D(\hat{x}))]$</td>
</tr>
<tr>
<td>NS GAN</td>
<td>$L_D^{NSGAN} = -\mathbb{E}<em>{x \sim p_d}[\log(D(x))] - \mathbb{E}</em>{\hat{x} \sim p_g}[\log(1 - D(\hat{x}))]$</td>
<td>$L_G^{NSGAN} = -\mathbb{E}_{\hat{x} \sim p_g}[\log(D(\hat{x}))]$</td>
</tr>
<tr>
<td>WGAN</td>
<td>$L_D^{WGAN} = -\mathbb{E}<em>{x \sim p_d}[D(x)] + \mathbb{E}</em>{\hat{x} \sim p_g}[D(\hat{x})]$</td>
<td>$L_G^{WGAN} = -\mathbb{E}_{\hat{x} \sim p_g}[D(\hat{x})]$</td>
</tr>
<tr>
<td>WGAN GP</td>
<td>$L_D^{WANGP} = L_D^{WGAN} + \lambda \mathbb{E}_{\hat{x} \sim p_g}[\left(</td>
<td></td>
</tr>
<tr>
<td>LS GAN</td>
<td>$L_D^{LSGAN} = -\mathbb{E}<em>{x \sim p_d}[(D(x) - 1)^2] + \mathbb{E}</em>{\hat{x} \sim p_g}[D(\hat{x})^2]$</td>
<td>$L_G^{LSGAN} = -\mathbb{E}_{\hat{x} \sim p_g}[(D(\hat{x}) - 1)^2]$</td>
</tr>
<tr>
<td>DRAGAN</td>
<td>$L_D^{DRAGAN} = L_D^{GAN} + \lambda \mathbb{E}_{\hat{x} \sim p_d + N(0,c)}[\left(</td>
<td></td>
</tr>
<tr>
<td>BEGAN</td>
<td>$L_D^{BEGAN} = \mathbb{E}_{x \sim p_d}[</td>
<td></td>
</tr>
</tbody>
</table>

What do GANs generate?

What do GANs generate?

New generation architectures

State-of-the-art generation
State-of-the-art generation
State-of-the-art generation
Some caveats about faces

• Faces are *almost* linear

Original face

Reconstruction from 7 principal components
PCA components of faces

- First few basis elements: shading
- Next basis elements: identity + expression

Basis images 1-4 model the shading
Rest of the basis images (five representatives are shown) model the facial expression

https://grail.cs.washington.edu/cflow/
Variational Autoencoders
Back to generative models

• Need a probabilistic model we can sample from
• Write probabilistic model so that training images are highly likely
  • Probabilistic model maps noise from standard Gaussian to image
  • But which noise generates which training image?
Autoencoders

• Maps each image to a particular latent code
• But not a probabilistic model!

$L_2$ Loss
Variational autoencoders

\[ D_{KL}(Q(Z|x) \| P(z)) \]