CS 2770: Computer Vision
Generative Adversarial Networks

Prof. Adriana Kovashka
University of Pittsburgh
April 12, 17, 2018
Plan for today

• Generative models: What are they?
• Techniques: Generative Adversarial Networks
  – Conditional GANs
  – Cycle-consistency loss
  – Dealing with sparse data
  – *Applications interleaved with techniques*
• Generation before GANs
• Generation for synthetic training data
Supervised vs Unsupervised Learning

Supervised Learning

**Data:** \((x, y)\)

\(x\) is data, \(y\) is label

**Goal:** Learn a *function* to map \(x \rightarrow y\)

**Examples:** Classification, regression, object detection, semantic segmentation, image captioning, etc.
Supervised vs Unsupervised Learning

Supervised Learning

**Data:** \((x, y)\)

\(x\) is data, \(y\) is label

**Goal:** Learn a *function* to map \(x \rightarrow y\)

**Examples:** Classification, regression, object detection, semantic segmentation, image captioning, etc.
Supervised vs Unsupervised Learning

Supervised Learning

**Data**: \((x, y)\)
\(x\) is data, \(y\) is label

**Goal**: Learn a *function* to map \(x \rightarrow y\)

**Examples**: Classification, regression, object detection, semantic segmentation, image captioning, etc.

DOG, DOG, CAT

Object Detection
Supervised vs Unsupervised Learning

Supervised Learning

Data: \((x, y)\)
\(x\) is data, \(y\) is label

Goal: Learn a \textit{function} to map \(x \rightarrow y\)

Examples: Classification, regression, object detection, semantic segmentation, image captioning, etc.
Supervised vs Unsupervised Learning

Supervised Learning

Data: \((x, y)\)
x is data, y is label

Goal: Learn a function to map \(x \rightarrow y\)

Examples: Classification, regression, object detection, semantic segmentation, image captioning, etc.
Supervised vs Unsupervised Learning

Unsupervised Learning

**Data:** $x$
Just data, no labels!

**Goal:** Learn some underlying hidden *structure* of the data

**Examples:** Clustering, dimensionality reduction, feature learning, density estimation, etc.
Unsupervised Learning

**Data:** $x$
Just data, no labels!

**Goal:** Learn some underlying hidden *structure* of the data

**Examples:** Clustering, dimensionality reduction, feature learning, density estimation, etc.
Unsupervised Learning

**Data:** $x$
Just data, no labels!

**Goal:** Learn some underlying hidden *structure* of the data

**Examples:** Clustering, dimensionality reduction, feature learning, density estimation, etc.

Principal Component Analysis
(Dimensionality reduction)
Unsupervised Learning

Data: $x$
Just data, no labels!

Goal: Learn some underlying hidden structure of the data

Examples: Clustering, dimensionality reduction, feature learning, density estimation, etc.

Supervised vs Unsupervised Learning
Supervised vs Unsupervised Learning

Unsupervised Learning

**Data**: $x$
Just data, no labels!

**Goal**: Learn some underlying hidden *structure* of the data

**Examples**: Clustering, dimensionality reduction, feature learning, density estimation, etc.

1-d density estimation

2-d density estimation
Supervised vs Unsupervised Learning

Supervised Learning

**Data**: (x, y)

x is data, y is label

**Goal**: Learn a *function* to map x -> y

**Examples**: Classification, regression, object detection, semantic segmentation, image captioning, etc.

Unsupervised Learning

**Data**: x

Just data, **no labels**!

**Goal**: Learn some underlying hidden *structure* of the data

**Examples**: Clustering, dimensionality reduction, feature learning, density estimation, etc.
Generative Models

Training data $\sim p_{\text{data}}(x)$

Generated samples $\sim p_{\text{model}}(x)$

Want to learn $p_{\text{model}}(x)$ similar to $p_{\text{data}}(x)$
Generative Models

Addresses density estimation, a core problem in unsupervised learning

**Several flavors:**
- Explicit density estimation: explicitly define and solve for $p_{model}(x)$
- Implicit density estimation: learn model that can sample from $p_{model}(x)$ w/o explicitly defining it
Why Generative Models?

- Realistic samples for artwork, super-resolution, colorization, etc.

- Generative models can be used to enhance training datasets with diverse synthetic data
- Generative models of time-series data can be used for simulation

Adapted from Serena Young
Taxonomy of Generative Models

Generative models

Explicit density

Implicit density

Tractable density

Approximate density

Markov Chain

Fully Visible Belief Nets
- NADE
- MADE
- PixelRNN/CNN

Change of variables models (nonlinear ICA)

Variational

Markov Chain

Variational Autoencoder

Boltzmann Machine

Figure copyright and adapted from Ian Goodfellow, Tutorial on Generative Adversarial Networks, 2017.
Generative Adversarial Networks

Ian Goodfellow et al., “Generative Adversarial Nets”, NIPS 2014

Problem: Want to sample from complex, high-dimensional training distribution. No direct way to do this!

Solution: Sample from a simple distribution, e.g. random noise. Learn transformation to training distribution.

Q: What can we use to represent this complex transformation?
Problem: Want to sample from complex, high-dimensional training distribution. No direct way to do this!

Solution: Sample from a simple distribution, e.g. random noise. Learn transformation to training distribution.

Q: What can we use to represent this complex transformation?

A: A neural network!
Training GANs: Two-player game

Generator network: try to fool the discriminator by generating real-looking images

Discriminator network: try to distinguish between real and fake images

Ian Goodfellow et al., “Generative Adversarial Nets”, NIPS 2014
Training GANs: Two-player game

**Generator network**: try to fool the discriminator by generating real-looking images

**Discriminator network**: try to distinguish between real and fake images

Ian Goodfellow et al., “Generative Adversarial Nets”, NIPS 2014
Adversarial Networks Framework

- Differentiable function $D$ tries to output 1.
- $D$ tries to output 0.
- Differentiable function $D$.
- $x$ sampled from data.
- $x$ sampled from model.
- Differentiable function $G$.
- Input noise $Z$.

---

Ian Goodfellow
Adversarial Networks Framework

Generator
\[ x \sim G(z) \]

Discriminator
Real vs. Fake

[Goodfellow et al. 2014]

Jun-Yan Zhu
Training GANs: Two-player game

Generator network: try to fool the discriminator by generating real-looking images

Discriminator network: try to distinguish between real and fake images

Train jointly in minimax game

Minimax objective function:

$$\min_{\theta_g} \max_{\theta_d} \left[ \mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log (1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$
Training GANs: Two-player game

Ian Goodfellow et al., “Generative Adversarial Nets”, NIPS 2014

**Generator network**: try to fool the discriminator by generating real-looking images

**Discriminator network**: try to distinguish between real and fake images

Train jointly in **minimax game**

Minimax objective function:

\[
\min_{\theta_g} \max_{\theta_d} \left[ \mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log (1 - D_{\theta_d}(G_{\theta_g}(z))) \right]
\]

- **Discriminator output** for real data \( x \)
- **Discriminator output** for generated fake data \( G(z) \)
Training GANs: Two-player game

Generator network: try to fool the discriminator by generating real-looking images
Discriminator network: try to distinguish between real and fake images

Train jointly in minimax game

Minimax objective function:

\[
\min_{\theta_g} \max_{\theta_d} \left[ \mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log (1 - D_{\theta_d}(G_{\theta_g}(z))) \right]
\]

- Discriminator (\(\theta_d\)) wants to maximize objective such that \(D(x)\) is close to 1 (real) and \(D(G(z))\) is close to 0 (fake)
- Generator (\(\theta_g\)) wants to minimize objective such that \(D(G(z))\) is close to 1 (discriminator is fooled into thinking generated \(G(z)\) is real)

Ian Goodfellow et al., “Generative Adversarial Nets”, NIPS 2014
Training GANs: Two-player game

Minimax objective function:

$$\min_{\theta_g} \max_{\theta_d} \left[ \mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log (1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

Alternate between:

1. **Gradient ascent** on discriminator

   $$\max_{\theta_d} \left[ \mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log (1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

2. **Gradient descent** on generator

   $$\min_{\theta_g} \mathbb{E}_{z \sim p(z)} \log (1 - D_{\theta_d}(G_{\theta_g}(z)))$$
Minimax objective function:

$$\min_{\theta_g} \max_{\theta_d} \left[ \mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

Alternate between:

1. **Gradient ascent** on discriminator
   $$\max_{\theta_d} \left[ \mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

2. **Gradient descent** on generator
   $$\min_{\theta_g} \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z)))$$

In practice, optimizing this generator objective does not work well!

When sample is likely fake, want to learn from it to improve generator. But gradient in this region is relatively flat!
Training GANs: Two-player game

Ian Goodfellow et al., “Generative Adversarial Nets”, NIPS 2014

Minimax objective function:

\[
\min_{\theta_g} \max_{\theta_d} \left[ \mathbb{E}_{x \sim p_{\text{data}}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]
\]

Alternate between:

1. **Gradient ascent** on discriminator

\[
\max_{\theta_d} \left[ \mathbb{E}_{x \sim p_{\text{data}}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]
\]

2. **Instead: Gradient ascent** on generator, different objective

\[
\max_{\theta_g} \mathbb{E}_{z \sim p(z)} \log(D_{\theta_d}(G_{\theta_g}(z)))
\]

Instead of minimizing likelihood of discriminator being correct, now maximize likelihood of discriminator being wrong. Same objective of fooling discriminator, but now higher gradient signal for bad samples => works much better! Standard in practice.
Training GANs: Two-player game

Putting it together: GAN training algorithm

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:
          \[
          \nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[ \log D_{\theta_d}(x^{(i)}) + \log(1 - D_{\theta_d}(G_{\theta_g}(z^{(i)}))) \right]
          \n          \]
    end for
    • Sample minibatch of \( m \) noise samples \( \{z^{(1)}, \ldots, z^{(m)}\} \) from noise prior \( p_g(z) \).
    • Update the generator by ascending its stochastic gradient (improved objective):
      \[
      \nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log(D_{\theta_d}(G_{\theta_g}(z^{(i)})))
      \]
end for
Training GANs: Two-player game

Generator network: try to fool the discriminator by generating real-looking images
Discriminator network: try to distinguish between real and fake images

After training, use generator network to generate new images

Serena Young
Generative Adversarial Nets

Samples from the model look amazing!

Radford et al, ICLR 2016
Generative Adversarial Nets: Interpretable Vector Math

Samples from the model

Smiling woman Neutral woman Neutral man

Radford et al, ICLR 2016
Generative Adversarial Nets: Interpretable Vector Math

---

Smiling woman  Neutral woman  Neutral man

Samples from the model

Average Z vectors, do arithmetic

Radford et al, ICLR 2016
Generative Adversarial Nets: Interpretable Vector Math

Samples from the model

Smiling woman

Neutral woman

Neutral man

Average Z vectors, do arithmetic

= Smiling Man

Radford et al, ICLR 2016
Generative Adversarial Nets: Interpretable Vector Math

Glasses man  No glasses man  No glasses woman

Radford et al, ICLR 2016

Serena Young
Generative Adversarial Nets: Interpretable Vector Math

Glasses man  No glasses man  No glasses woman

Woman with glasses

Radford et al, ICLR 2016
What is in this image?

(Yeh et al., 2016)
Generative modeling reveals a face

(Yeh et al., 2016)
Creative Adversarial Networks

CAN: Top ranked by human subjects

(Elgammal et al., 2017)
Artificial Fashion: vue.ai
Celebrities Who Never Existed

Karras et al., “Progressive Growing of GANs for Improved Quality, Stability, and Variation”, ICLR 2018
GANs for Privacy (Action Detection)

Ren et al., “Learning to Anonymize Faces for Privacy Preserving Action Detection”, arxiv 2018
2017: Year of the GAN

Better training and generation

Source->Target domain transfer

Text -> Image Synthesis


Reed et al. 2017.

Many GAN applications

Plan for today

• Generative models: What are they?
• Techniques: Generative Adversarial Networks
  – Conditional GANs
  – Cycle-consistency loss
  – Dealing with sparse data
  – Applications interleaved with techniques
• Generation before GANs
• Generation for synthetic training data
GANs

$G$: generate fake samples that can fool $D$
$D$: classify fake samples vs. real images

[Goodfellow et al. 2014]

Jun-Yan Zhu
Conditional GANs

G(x)

real or fake pair?
Edges → Images

Input | Output | Input | Output | Input | Output
--- | --- | --- | --- | --- | ---
[Image of handbag] | [Image of handbag] | [Image of handbag] | [Image of handbag] | [Image of handbag] | [Image of handbag]
[Image of purse] | [Image of purse] | [Image of purse] | [Image of purse] | [Image of purse] | [Image of purse]
[Image of backpack] | [Image of backpack] | [Image of backpack] | [Image of backpack] | [Image of backpack] | [Image of backpack]

Edges from [Xie & Tu, 2015]
Sketched → Images

Input → Output

Input → Output

Input → Output

Data from [Eitz, Hays, Alexa, 2012]

Pix2pix / CycleGAN
#edges2cats

[Christopher Hesse]

Ivy Tasi @ivymyt

Vitaly Vidmirov @vvid

https://affinelayer.com/pixsrv/
Cycle Consistency

Discriminator $D_Y$: $L_{GAN}(G(x), y)$
Real zebras vs. generated zebras

Zhu et al., “Unpaired Image-To-Image Translation Using Cycle-Consistent Adversarial Networks”, ICCV 2017
Cycle Consistency

Discriminator $D_Y: L_{GAN}(G(x), y)$
Real zebras vs. generated zebras
Discriminator $D_X: L_{GAN}(F(y), x)$
Real horses vs. generated horses

Zhu et al., “Unpaired Image-To-Image Translation Using Cycle-Consistent Adversarial Networks”, ICCV 2017
Zhu et al., “Unpaired Image-To-Image Translation Using Cycle-Consistent Adversarial Networks”, ICCV 2017
Zhu et al., “Unpaired Image-To-Image Translation Using Cycle-Consistent Adversarial Networks”, ICCV 2017
Cycle Consistency

Forward cycle loss: $\|F(G(x)) - x\|_1$

Zhu et al., “Unpaired Image-To-Image Translation Using Cycle-Consistent Adversarial Networks”, ICCV 2017
Cycle Consistency

Forward cycle loss: $\|F(G(x)) - x\|_1$

Zhu et al., "Unpaired Image-To-Image Translation Using Cycle-Consistent Adversarial Networks", ICCV 2017
Cycle Consistency

Forward cycle loss: \( \|F(G(x)) - x\|_1 \)

Backward cycle loss: \( \|G(F(y)) - y\|_1 \)

Zhu et al., “Unpaired Image-To-Image Translation Using Cycle-Consistent Adversarial Networks”, ICCV 2017
Training Details: Objective

\[ 
\mathcal{L}_{\text{GAN}}(G, D_Y, X, Y) = \mathbb{E}_{y \sim p_{\text{data}}(y)}[\log D_Y(y)] + \mathbb{E}_{x \sim p_{\text{data}}(x)}[\log(1 - D_Y(G(x))], 
\]

\[ 
\mathcal{L}_{\text{cyc}}(G, F') = \mathbb{E}_{x \sim p_{\text{data}}(x)}[\|F(G(x)) - x\|_1] + \mathbb{E}_{y \sim p_{\text{data}}(y)}[\|G(F(y)) - y\|_1]. 
\]

\[ 
\mathcal{L}(G, F, D_X, D_Y) = \mathcal{L}_{\text{GAN}}(G, D_Y, X, Y) + \mathcal{L}_{\text{GAN}}(F, D_X, Y, X) + \lambda \mathcal{L}_{\text{cyc}}(G, F),
\]

\[ 
G^*, F^* = \arg \min_{G,F} \max_{D_X,D_Y} \mathcal{L}(G, F, D_X, D_Y).
\]

Zhu et al., “Unpaired Image-To-Image Translation Using Cycle-Consistent Adversarial Networks”, ICCV 2017
Pix2pix / CycleGAN
Monet’s paintings → photos

Pix2pix / CycleGAN
Monet’s paintings → photos

Pix2pix / CycleGAN
Pix2pix / CycleGAN
Pix2pix / CycleGAN
#CycleGAN

Monet → Thomas Kinkade @David Fouhey

Resurrecting Ancient Cities @ Jack Clark

Birds @Matt Powell

Bear → Panda @Matt Powell
StarGAN

Choi et al., “StarGAN: Unified Generative Adversarial Networks for Multi-Domain Image-to-Image Translation”, CVPR 2018
Plan for today

• Generative models: What are they?
• Techniques: Generative Adversarial Networks
  – Conditional GANs
  – Cycle-consistency loss
  – Dealing with sparse data
    – Applications interleaved with techniques
• Generation before GANs
• Generation for synthetic training data
Autoencoders

- Encoder
- Code
- Decoder

$L_1/L_2$ reconstruction loss
Adversarial Autoencoders

Improve reconstruction quality by adding a GAN loss.

Adversarial Autoencoders
A. Makhzani, J. Shlens, N.Jaitly, I. Goodfellow, B. Frey
Autoencoder GANs

• Combine the reconstruction power of autoencoders with the sampling power of GANs.
• By construction, autoencoders learn to cover the entire training data.
• Work on the codespace - not data space.

Adapted from Mihaela Rosca
Learning the code distribution

Assumption
learning the code distribution is simpler than learning data distribution

Encoder

Decoder

learn $p(\text{codes})$

reconstructing

sampling

Decoder

Mihaela Rosca

OUR WORK
Plan for today

• Generative models: What are they?
• Techniques: Generative Adversarial Networks
  – Conditional GANs
  – Cycle-consistency loss
  – Dealing with sparse data
  – Applications interleaved with techniques
• Generation before GANs
• Generation for synthetic training data
Neural Style Transfer

Image Quilting for Texture Synthesis & Transfer

Alexei Efros (UC Berkeley)
Bill Freeman (MERL)
[Shannon, ’48] proposed a way to generate English-looking text using N-grams:

• Assume a generalized Markov model
• Use a large text to compute prob. distributions of each letter given N-1 previous letters
• Starting from a seed repeatedly sample this Markov chain to generate new letters
• Also works for whole words

WE NEED TO EAT CAKE
Assuming Markov property, compute $P(p | N(p))$

- Building explicit probability tables infeasible
- Instead, let’s search the input image for all similar neighborhoods — that’s our histogram for $p$

To synthesize $p$, just pick one match at random
Texture Transfer

Take the texture from one object and “paint” it onto another object

• This requires separating texture and shape
• That’s HARD, but we can cheat
• Assume we can capture shape by boundary and rough shading

Then, just add another constraint when sampling: similarity to underlying image at that spot

Efros
parmisan

+ +

= =

rice

= =

Efros
Plan for today

• Generative models: What are they?

• Techniques: Generative Adversarial Networks
  – Conditional GANs
  – Cycle-consistency loss
  – Dealing with sparse data
  – Applications interleaved with techniques

• Generation before GANs

• Generation for synthetic training data
Generating Training Data

Unlabeled Real Images

Synthetic → Refiner → Refined

Shrivastava et al., “Learning From Simulated and Unsupervised Images Through Adversarial Training”, CVPR 2017
Generating Training Data

(a) Image examples from the Linemod dataset.

(b) Examples generated by our model, trained on Linemod.

Bousmalis et al., “Unsupervised Pixel-Level Domain Adaptation With Generative Adversarial Networks”, CVPR 2017
Generating Training Data

Huang and Ramanan, “Expecting the Unexpected: Training Detectors for Unusual Pedestrians With Adversarial Imposters”, CVPR.
Generating Training Data

Varol et al., “Learning From Synthetic Humans”, CVPR 2017
Recognizing atypical objects

- Computer vision models cannot recognize objects in unusual modalities well

*Thomas and Kovashka, in submission to ECCV 2018*
Recognizing atypical objects

• Training a network with two source modalities helps it generalize to novel target modalities

• We can obtain good secondary modalities for free:
  • Automatic transformation of a photo to a cartoon/sketch (style transfer by Johnson et al. ECCV 2016)
Recognizing atypical objects

• Train a network with additional modalities:
  • The outlines of objects (segmentation)
  • Use a convolutional neural network to transform a photo into a cartoon/sketch

Thomas and Kovashka, in submission to ECCV 2018
Recognizing atypical objects

• We outperform prior domain adaptation methods
  • Including a generous modification of Bousmalis et al. 2017 ~
  our approach but with GAN instead of style transfer

<table>
<thead>
<tr>
<th></th>
<th>PACS: Art, Cartoons, Sketches (Li 2017)</th>
<th>Sketchy Database (Sangkloy 2016)</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Adaptation</td>
<td>0.388605333</td>
<td>0.093055</td>
</tr>
<tr>
<td>Ganin &amp; Lempitsky 2015</td>
<td>0.537944333</td>
<td>0.112329</td>
</tr>
<tr>
<td>Castrejon et al. 2016</td>
<td>0.400783</td>
<td>0.085107</td>
</tr>
<tr>
<td>Ghifary et al. 2016</td>
<td>0.462287276</td>
<td>0.22547</td>
</tr>
<tr>
<td>Long et al. 2016</td>
<td>0.566674</td>
<td>0.303163</td>
</tr>
<tr>
<td>Bousmalis et al. 2017</td>
<td>0.547297333</td>
<td>0.284765</td>
</tr>
<tr>
<td></td>
<td><strong>Ours</strong></td>
<td><strong>0.587813333</strong></td>
</tr>
<tr>
<td></td>
<td><strong>Supervised</strong></td>
<td><strong>0.904064333</strong></td>
</tr>
<tr>
<td></td>
<td>(upper bound)</td>
<td></td>
</tr>
</tbody>
</table>

Thomas and Kovashka, in submission to ECCV 2018