Lecture 12: Image synthesis
• PS1 grades out
• Submit regrade requests ASAP (due in a week)
• New submission format for PS5. Deadline extend 12 hours in case you need more time for this.
Recall: parametric texture synthesis

Start with a noise image as output.

**Iterative algorithm** [Heeger & Bergen, 95]:
- Match pixel histogram of output image to input
- Decompose input/output images using a Steerable Pyramid
- Match histograms of input and output pyramids
- Reconstruct image and repeat

Source: A. Efros
Recall: steerable filter features

Filter bank

Why these features?

Input image

Source: A. Efros
Recall: AlexNet units

96 Units in conv1

Source: Isola, Torralba, Freeman
Extracting features from a trained network

Unit activations

dog
0.1

cat
0.2

gecko
0.5

alligator
0.0

fish
0.2
Extracting neural net features

$(112 \times 112) \times 128$ for conv2 of VGG19
Capturing feature correlations

Gram (≈ covariance) matrix:

\[ G_{ij} = \sum_{x=1}^{w} \sum_{y=1}^{h} c_i(x, y)c_j(x, y) \]

[Gatys et al. 2016]

Idea: correlations between unit activations convey texture. Discard global spatial information.
Matching image statistics

\[ \sum_{i=1}^{128} \sum_{j=1}^{128} (G_{ij}(I) - G_{ij}(\hat{I}))^2 \]

Find \( \hat{I} \) by minimizing:

Implementation details:

- Use many layers of network.
- Minimize with gradient descent
- How do we compute gradients? Backprop!
Texture captures artistic style

Can we transfer the style of a painting to a photo?

[Gatys et al. 2016]
Match the **style** of the painting.

... and the **content** of the photo.

Perceptual loss:
usually distance in feature space

\[
\sum_{i=1}^{128} \sum_{j=1}^{128} (G_{ij}(\hat{I}) - G_{ij}(I))^2
\]

\[
\sum_{i} \sum_{x,y} (c_i(x, y) - \hat{c}_i(x, y))^2
\]
London during the day.

New York at night.
Neural networks that generate images
Image classification

\[ \text{image } x \rightarrow \text{Classifier} \rightarrow \text{"Duck"} \]

Source: Isola, Freeman, Torralba
Image synthesis

“Duck” $\rightarrow$ Generator $\rightarrow$ image $\mathbf{x}$

label $y$

Source: Isola, Freeman, Torralba
Neural networks as distribution transformers

Source distribution

$\mathcal{G}$

Target distribution

$\mathcal{G}$

Source: Isola, Freeman, Torralba
Gaussian noise
\( z \sim \mathcal{N}(\tilde{0}, 1) \)

Synthesized image

Source: Isola, Freeman, Torralba
Neural networks as distribution transformers

Gaussian noise

\[ z \sim \mathcal{N}(\vec{0}, 1) \]

Source: Isola, Freeman, Torralba

Synthesized image
Generative adversarial networks (GANs)
$G$ tries to synthesize fake images that fool $D$

$D$ tries to identify the fakes

Source: Isola, Freeman, Torralba
$\arg\max_D \mathbb{E}_{z,x} \left[ \log D(G(z)) + \log (1 - D(x)) \right]$ 

Source: Isola, Freeman, Torralba

[Goodfellow et al., 2014]
\( G \) tries to synthesize fake images that \textit{fool} \( D \):

\[
\arg\min_G \mathbb{E}_{z,x} \left[ \log D(G(z)) + \log (1 - D(x)) \right]
\]

Source: Isola, Freeman, Torralba  

[Goodfellow et al., 2014]
G tries to synthesize fake images that fool the best D:

$$\arg\min_G \max_D \mathbb{E}_{z,x}[\log D(G(z)) + \log (1 - D(x))]$$

Source: Isola, Freeman, Torralba [Goodfellow et al., 2014]
Training

\[ G(z) \]

\[ D \]

\( G \) tries to synthesize fake images that fool \( D \)

\( D \) tries to identify the fakes

- Training: iterate between training \( D \) and \( G \) with backprop.
- Global optimum when \( G \) reproduces data distribution (see book)
Samples from BigGAN

[Brock et al. 2018]


Source: Isola, Freeman, Torralba
Latent space (Gaussian)

\[ Z \]

Data space (Natural image manifold)

\[ X \]

[BigGAN, Brock et al. 2018]
Rapid progress due mostly to better architectures

ACGAN [Odena et al. 2016]

BigGAN [Brock et al. 2018]

Both trained on ImageNet

Source: Isola, Freeman, Torralba
Architectures

DCGAN
[Radford, Metz, Chintala 2016]

Transpose convolution + batch norm + nonlinearities

StyleGAN
[Karras, Laine, Aila 2019]

Similar but bigger and lots of engineering details.

Source: Isola, Freeman, Torralba
Image translation

Google Map → Translator → Satellite photo

Source: Isola, Freeman, Torralba
$$\mathbf{x} \xrightarrow{G} G(\mathbf{x})$$

Idea: L1 loss

$$\| G(\mathbf{x}) - \mathbf{y} \|_1$$

Source: Isola, Freeman, Torralba
Input

L1 loss

Source: Isola, Freeman, Torralba
\(G\) tries to synthesize fake images that fool \(D\).

\(D\) tries to identify the fakes.
\[
\arg \max_D \mathbb{E}_{x,y} \left[ \log D(G(x)) + \log(1 - D(y)) \right]
\]

Source: Isola, Freeman, Torralba
G tries to synthesize fake images that fool D:

$$\arg\min_G \mathbb{E}_{x,y} \left[ \log D(G(x)) + \log(1 - D(y)) \right]$$

Source: Isola, Freeman, Torralba
\( \mathbf{G} \) tries to synthesize fake images that *fool* the *best* \( \mathbf{D} \):

\[
\arg \min_G \max_D \mathbb{E}_{x,y} \left[ \log D(G(x)) + \log(1 - D(y)) \right]
\]

Source: Isola, Freeman, Torralba
\[ \arg\min_G \max_D \mathbb{E}_{x,y} \left[ \log D(G(x)) + \log(1 - D(y)) \right] \]
\[
\arg\min_G \max_D \mathbb{E}_{x,y} \left[ \log D(G(x)) + \log(1 - D(y)) \right]
\]

Source: Isola, Freeman, Torralba
arg min_G max_D \mathbb{E}_{x,y} [ \log D(G(x)) + \log(1 - D(y)) ]

Source: Isola, Freeman, Torralba
\[
\arg \min_G \max_D \mathbb{E}_{x,y} \left[ \log D(x, G(x)) + \log(1 - D(x, y)) \right]
\]

Source: Isola, Freeman, Torralba
arg min \max_G \mathbb{E}_{x,y} \left[ \log D(x, G(x)) + \log(1 - D(x, y)) \right]
\[
\begin{align*}
\arg \min_G \max_D & \quad \mathbb{E}_{x,y} \left[ \log D(x, G(x)) + \log(1 - D(x, y)) \right] \\
\text{real or fake pair?}
\end{align*}
\]
Training Details: Loss function

Conditional GAN

\[ G^* = \arg \min_G \max_D \mathcal{L}_{cGAN}(G, D) + \lambda \mathcal{L}_{L1}(G). \]

Helps stabilize training + faster convergence

Source: Isola, Freeman, Torralba
Data from [maps.google.com]
Structured Prediction (with GAN)

Input

Source: Isola, Freeman, Torralba
Training data

\[
\begin{align*}
\{ & \{ \text{shoe 1}, \text{shoe 2} \} , \\
& \{ \text{boot 1}, \text{boot 2} \} , \\
& \{ \text{shoe 3} \} , \\
& \ldots \\
& \{ \text{HED, Xie \& Tu, 2015} \}
\end{align*}
\]

Source: Isola, Freeman, Torralba
edges2cats

[Chris Hess, edges2cats]

Source: Isola, Freeman, Torralba
Ivy Tasi @ivymyt

Vitaly Vidmirov @vvid

Source: Isola, Freeman, Torralba
Architectures

Generator: U-Net

Skip connections between encoder and decoder layers

Figure from [Isola et al., “Image-to-Image Translation with Conditional Adversarial Networks”, 2017]
Architectures

**Discriminator:** fully convolutional network

Sequence of strided convolutions

$n \times n$ output map
Architectures

**Discriminator:** fully convolutional network

Also known as a **Patch GAN**, since only looks at patches
More recent architectures

[“GauGAN”, Park et al., CVPR 2019]
Handling unpaired data

Paired data
\[ x_i \quad y_i \]
\{ \}
\{ \}
\{ \}
\{ \}
\{ \}

Unpaired data
\[ X \quad Y \]
\{ \}
\{ \}
\{ \}
\{ \}
\{ \}

Source: Isola, Freeman, Torralba
arg min\( G \) \( \max\ D \)
\[
\mathbb{E}_{x,y} \left[ \log D(x, G(x)) + \log(1 - D(x, y)) \right]
\]

Source: Isola, Freeman, Torralba
arg min \max_G \mathbb{E}_{x,y} \left[ \log D(x, G(x)) + \log(1 - D(x, y)) \right]

No input-output pairs!

Source: Isola, Freeman, Torralba
GAN loss checks if output is part of an admissible set

\[
\arg\min_G \max_D \mathbb{E}_{x,y} \left[ \log D(G(x)) + \log(1 - D(y)) \right]
\]
Nothing to force output to correspond to input

Source: Isola, Freeman, Torralba
CycleGAN

Source: Isola, Freeman, Torralba
Cycle Consistency Loss

$X \xrightarrow{G} \hat{Y} \xrightarrow{F} \hat{X}$

$D_Y$

$\|F(G(x)) - x\|_1$

Source: Isola, Freeman, Torralba
Cycle Consistency Loss

Source: Isola, Freeman, Torralba