CS5670: Computer Vision
Image Manifolds & Image Synthesis (including GANS)

Most content from Abe Davis, with additional credit to Jin Sun and Phillip Isola
Announcements

• In class final on May 10
  – Open book, open note (your own notes – please do not print out whole slide decks)
• Project 5 (Neural Radiance Fields) due tomorrow by 8:00 pm
• Course evaluations are open starting today (May 3)
  – We would love your feedback!
  – Small amount of extra credit for filling out
    • What you write is still anonymous, instructors only see whether students filled it out
  
https://apps.engineering.cornell.edu/CourseEval/
DIMENSIONALITY REDUCTION

By Abe Davis
Linear Dimensionality Reduction: 2D- > 1D

- Consider a bunch of data points in 2D
- Let’s say these points only differ along one line
- If so, we can translate and rotate our data so that it is 1D
Linear Dimensionality Reduction: 3D- >2D

• Similar to 1D case, we can fit a plane to the data, and transform our coordinate system so that plane becomes the x-y plane

• “Plane fitting”

• More generally: look for the 2D subspace that best fits the data, and ignore the remaining dimensions

Think of this as data that sits on a flat sheet of paper, suspended in 3D space. We will come back to this analogy in a couple slides...
Generalizing Linear Dimensionality Reduction

- **Principal Components Analysis (PCA)**: find and order orthogonal axes by how much the data varies along each axis.

- The axes we find (ordered by variance of our data) are called *principal components*.

- Dimensionality reduction can be done by using only the first $k$ principal components.

Side Note: principal components are closely related to the eigenvectors of the covariance matrix for our data.
Manifolds

• Think of a piece of paper as a 2D subspace
• If we bend & fold it, it’s still locally a 2D subspace...
• A “manifold” is the generalization of this concept to higher dimensions...
Autoencoders: Dimensionality Reduction for Manifolds

• Learn a non-linear transformation into some lower-dimensional space (encoder)
• Learn a transformation from lower-dimensional space back to original content (decoder)
• Loss function measures difference between input & output

• **Unsupervised**
  – No labels required!

Feature space at bottleneck is often called “latent space”
Autoencoders: Dimensionality Reduction for Manifolds

- Transformations that reduce dimensionality **cannot be invertible** in general

- An autoencoder tries to learn a transformation that is **invertible for points on some manifold**.
The Space of All Images

• Lets consider the space of all 100x100 images

• Now lets randomly sample that space...

• Conclusion: Most images are noise

pixels = np.random.rand(100,100,3)

Question:
What do we expect a random uniform sample of all images to look like?
Natural Image Manifolds

• Most images are “noise”

• “Meaningful” images tend to form some manifold within the space of all images

• Images of a particular class fall on manifolds within that manifold...
Denoising & the “Nullspace” of Autoencoders

• The autoencoder tries to learn a dimensionality reduction that is invertible for our data (data on some manifold)
• Most noise will be in the non-invertible part of image space (off the manifold)
• If we feed noisy data in, we will often get denoised data out

Examples from: https://blog.keras.io/building-autoencoders-in-keras.html
Problem

• Autoencoders can compress because data sits on a manifold

• This doesn’t mean that every point in the latent space will be on the manifold...

• GANs (later this lecture) will learn a loss function that helps with this.
Abe Davis, with slides from Jin Sun, Phillip Isola, and Richard Zhang

IMAGE-TO-IMAGE APPLICATIONS
Image prediction ("structured prediction")

**Object labeling**
- Image of a person and a horse
- Long et al. 2015, ...

**Depth prediction**
- Single RGB Image
- Depth Map
- Eigen et al. 2014, ...

**Text-to-photo**
- "this small bird has a pink breast and crown…"
- Reed et al. 2016, ...

**Style transfer**
- Image of a cityscape
- Gatys et al. 2016, ...
Image classification vs. image translation

• For image classification, we map an image to a label (e.g., “cat”)
• For image prediction/translation tasks, we map an image to another image-shaped thing (e.g., a depth map)
• What kind of convolutional neural network architecture can do this?
U-Net

• A popular network structure to generate same-sized output

• Similar to a convolutional autoencoder, but with “skip connections” that concatenate the output of earlier layers onto later layers

• Great for learning transformations from one image to another
Image Colorization

from Jin Sun, Richard Zhang, Phillip Isola
arg min_{\mathcal{F}} \mathbb{E}_{x,y}[L(\mathcal{F}(x), y)]

“What should I do”  “How should I do it?”

from Jin Sun, Richard Zhang, Phillip Isola
Training data

\[ \arg \min_{\mathcal{F}} \mathbb{E}_{x,y} [L(\mathcal{F}(x), y)] \]

Objective function (loss)

Neural Network

Color information: ab channels

from Jin Sun, Richard Zhang, Phillip Isola
from Jin Sun, Richard Zhang, Phillip Isola
Basic loss functions

Prediction: \( \hat{y} = \mathcal{F}(x) \)  
Truth: \( y \)

Classification (cross-entropy):
\[
L(\hat{y}, y) = - \sum_i \hat{y}_i \log y_i
\]
How many extra bits it takes to correct the predictions

Least-squares regression:
\[
L(\hat{y}, y) = \| \hat{y} - y \|_2
\]
How far off we are in Euclidean distance

from Jin Sun, Richard Zhang, Phillip Isola
Designing loss functions

Input

Output

Ground truth

\[ L_2(\hat{Y}, Y) = \frac{1}{2} \sum_{h,w} \| Y_{h,w} - \hat{Y}_{h,w} \|^2 \]
\[ L_2(\hat{Y}, Y) = \frac{1}{2} \sum_{h,w} \left\| Y_{h,w} - \hat{Y}_{h,w} \right\|_2^2 \]
Designing loss functions

Color distribution cross-entropy loss with colorfulness enhancing term.

[Zhang, Isola, Efros, ECCV 2016]
Designing loss functions

Image colorization

[Image colorization]

Super-resolution

[L2 regression]

[Johnson, Alahi, Li, ECCV 2016]

[L2 regression]

[Zhang, Isola, Efros, ECCV 2016]

[Johnson, Alahi, Li, ECCV 2016]
Designing loss functions

Image colorization

[Image colorization text]

Cross entropy objective, with colorfulness term

Super-resolution

[Super-resolution text]

Deep feature covariance matching objective

[References]

[Zhang, Isola, Efros, ECCV 2016]

[Johnson, Alahi, Li, ECCV 2016]
Better Loss Function: Sticking to the Manifold

- How do we design a loss function that penalizes images that aren’t on the image manifold?

- Key insight: we will *learn* our loss function by training a network to discriminate between images that are on the manifold and images that aren’t
PART 3: GENERATIVE ADVERSARIAL NETWORKS (GANs)
Generative Adversarial Networks (GANs)

• Basic idea: Learn a mapping from some latent space to images on a particular manifold

• Example of a **Generative Model:**
  – We can think of classification as a way to compute some $P(x)$ that tells us the probability that image $x$ is a member of a class.
  – Rather than simply evaluating this distribution, a generative model tries to learn a way to sample from it
Generative Adversarial Networks (GANs)

• Generator network has similar structure to the decoder of our autoencoder
  – Maps from some latent space to images
• We train it in an adversarial manner against a discriminator network
  – Generator tries to create output indistinguishable from training data
  – Discriminator tries to distinguish between generator output and training data
Example: Randomly Sampling the Space of Face Images
(Using Generative Adversarial Networks (GANs))

Which face is real?
Example: Randomly Sampling the Space of Face Images
(Using Generative Adversarial Networks (GANs))

Which face is real?
Conditional GANs

- Generate samples from a conditional distribution
- Example: generate high-resolution image conditioned on low resolution input

[Ledig et al 2016]
Goodfellow et al., 2014
\( G \) tries to synthesize fake images that fool \( D \)

\( D \) tries to identify the fakes

[Goodfellow et al., 2014]
\[
\arg \max_D \mathbb{E}_{x,y} \left[ \log D(G(x)) + \log(1 - D(y)) \right]
\]

[Goodfellow et al., 2014]
\[ \text{arg min}_G \mathbb{E}_{x,y} \left[ \log D(G(x)) + \log(1 - D(y)) \right] \]
\( G \) tries to synthesize fake images that *fool* the best \( D \):

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

[Goodfellow et al., 2014]
G’s perspective: D is a loss function.

Rather than being hand-designed, it is *learned*.

[Goodfellow et al., 2014]
[Isola et al., 2017]
$\arg \min_G \max_D \mathbb{E}_{x,y} \left[ \log D(G(x)) + \log(1 - D(y)) \right]$
arg\ min_G\ max_D\ \mathbb{E}_{x,y}\left[\log D(G(x)) + \log(1 - D(y))\right]

[Goodfellow et al., 2014]
\[
\arg\min_G \max_D \mathbb{E}_{x,y} \left[ \log D(G(x)) + \log(1 - D(y)) \right]
\]

[Goodfellow et al., 2014]
[Isola et al., 2017]
arg min_G \max_D \mathbb{E}_{x,y} \left[ \log D(x, G(x)) + \log(1 - D(x, y)) \right] [Goodfellow et al., 2014] [Isola et al., 2017]
arg\ min_G \ max_D \ \mathbb{E}_{x,y} \left[ \log D(x, G(x)) + \log(1 - D(x, y)) \right]

[Goodfellow et al., 2014]
[Isola et al., 2017]
\[
\arg\min_G \max_D \mathbb{E}_{x,y} \left[ \log D(x, G(x)) + \log(1 - D(x, y)) \right]
\]

[Goodfellow et al., 2014]
[Isola et al., 2017]
\[
\arg \min_G \max_D \mathbb{E}_{x,y} \left[ \log D(x, G(x)) + \log(1 - D(x, y)) \right]
\]

[Goodfellow et al., 2014]
[Isola et al., 2017]
More Examples of Image-to-Image Translation with GANs

• We have pairs of corresponding training images
• Conditioned on one of the images, sample from the distribution of likely corresponding images
Data from [Russakovsky et al. 2015]
Data from [maps.google.com]
Labels → Street Views

Data from [Wang et al, 2018]
Day → Night

Input  | Output | Input  | Output | Input  | Output
--- | --- | --- | --- | --- | ---

Data from [Laffont et al., 2014]
Edges → Images

Edges from [Xie & Tu, 2015]
Demo

https://affinelayer.com/pixsrv/
Image Inpainting

Data from [Pathak et al., 2016]
Pose-guided Generation

Data from [Ma et al., 2018]
Challenges —> Solutions

• Output is high-dimensional, structured object
  – Approach: Use a deep net, D, to analyze output!

• Uncertainty in mapping; many plausible outputs
  – Approach: D only cares about “plausibility”, doesn’t hedge

• Lack of supervised training data
  – Approach: ?

“this small bird has a pink breast and crown…”
Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks

Jun-Yan Zhu*  Taesung Park*  Phillip Isola  Alexei A. Efros

UC Berkeley

In ICCV 2017

[Paper] [Code (Torch)] [Code (PyTorch)]

https://junyanz.github.io/CycleGAN/
StyleGAN

A Style-Based Generator Architecture for Generative Adversarial Networks
Tero Karras, Samuli Laine, Timo Aila
https://github.com/NVlabs/stylegan
Real-time image stylization

https://stadia.dev/blog/behind-the-scenes-with-stadias-style-transfer-ml/
Analyzing and Improving the Image Quality of StyleGAN
Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, Timo Aila

https://github.com/NVlabs/stylegan2
StyleGAN3 [2021]

Alias-Free Generative Adversarial Networks (StyleGAN3)
Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, Timo Aila
Questions?