Generative Models

Generative Model

What you will learn in this course

Low-Level Vision

Image Processing:
Photometric Image formation
Point Operations
Histogram Equalization
Linear & Non-Linear filters
Convolution & Cross correlation

Feature Detection:
Edge detection
Corner detection
Line detection

Feature Description:
SIFT
Feature Matching
Image Stitching
Panorama

Geometric Vision

Transformation and Camera Mode:
Geometric Image Formation
Pinhole Camera Model

Camera Calibration:
Estimating intrinsics
Estimating extrinsics

Visual Understanding

Object Detection & Tracking:
Single-stage detectors
Two-stage detectors
YOLO
Object tracking

Image Segmentation:
Traditional Image Segmentation
Learning-based Image Segmentation

Machine Learning for CV

Machine Learning for Classification:
k-NN
Linear Classifier

Deep Learning:
Optimization
Neural Network
CNN
CNN architectures:

Generative & Geometric Learning

Generative Models:
AE, VAE, GAN

3D Vision and 3D Deep Learning:
3D shape representations
Geometric deep learning overview
3D Deep Learning models : pointnet

What we will learn today

- Autoencoder
- Variational Autoencoder
- Generative Adversarial Network

Supervised vs. Unsupervised Learning

Supervised Learning

Data: (x, y)

x is data, y is label

Data Labeling is needed

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

$$y = f(x)$$

Examples: classification, regression, object detection, semantic segmentation, image segmentation etc.

Classification: Input x

$y = 8$

Object Detection

DOG, DOG, CAT

Semantic Segmentation

GRASS, CAT, TREE, SKY

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Supervised vs. Unsupervised Learning

Supervised Learning

Data: (x, y)

x is data, y is label

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

$$y = f(x)$$

Examples: classification, regression, object detection, semantic segmentation, image segmentation 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.

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Supervised vs. Unsupervised Learning

Clustering
(e.g. k-Means)

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.

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Supervised vs. Unsupervised Learning

Dimensionality Reduction
(e.g. Principal Component Analysis- PCA)

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.

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Supervised vs. Unsupervised Learning

Feature Learning
(e.g. Autoencoders)

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.

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Discriminative vs. Generative Models

Discriminative Model: Learn a probability distribution $P(y|x)$

Generative Model: Learn a probability distribution $P(x)$

Conditional Generative Model: Learn a probability distribution $P(x|y)$

Data: x

Label: y = Cat

Probability Recap:

Density Function

$P(x)$ assigns a positive number to each possible x; higher numbers mean x is more likely

Density functions are **normalized**:

$$\int_x P(x)dx = 1$$

Different values of x **complete** for density

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Discriminative vs. Generative Models

Discriminative Model: Learn a probability distribution $P(y|x)$

Generative Model: Learn a probability distribution $P(x)$

Conditional Generative Model: Learn a probability distribution $P(x|y)$

Density Function

$P(x)$ assigns a positive number to each possible x; higher numbers mean x is more likely

Density functions are **normalized**:

$$\int_x P(x)dx = 1$$

Different values of x **complete** for density

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Discriminative vs. Generative Models

Discriminative Model: Learn a probability distribution $P(y|x)$

Generative Model: Learn a probability distribution $P(x)$

Conditional Generative Model: Learn a probability distribution $P(x|y)$

Discriminative model: the possible **labels** for each input "**compete**" for probability mass. But no competition between **images**

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Discriminative vs. Generative Models

Discriminative Model: Learn a probability distribution $P(y|x)$

Generative Model: Learn a probability distribution $P(x)$

Conditional Generative Model: Learn a probability distribution $P(x|y)$

Discriminative model: No way for the model to handle unreasonable inputs; it must give label distributions for all images

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Discriminative vs. Generative Models

Discriminative Model: Learn a probability distribution $P(y|x)$

Generative Model: Learn a probability distribution $P(x)$

Conditional Generative Model: Learn a probability distribution $P(x|y)$

Generative model: All possible images compete with each other for probability mass

Requires deep image understanding! Is a dog more likely to sit or stand? How about 3-legged dog vs 3-armed monkey?

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Discriminative vs. Generative Models

Discriminative Model: Learn a probability distribution $P(y|x)$

Generative Model: Learn a probability distribution $P(x)$

Conditional Generative Model: Learn a probability distribution $P(x|y)$

Generative model: All possible images compete with each other for probability mass

Model can "reject" unreasonable inputs by assigning them small values

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Discriminative vs. Generative Models

Discriminative Model: Learn a probability distribution $P(y|x)$

Generative Model: Learn a probability distribution $P(x)$

Conditional Generative Model: Learn a probability distribution $P(x|y)$

Conditional Generative Model: Each possible label induces a competition among all images

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Discriminative vs. Generative Models

Discriminative Model: Learn a probability distribution $P(y|x)$

Generative Model: Learn a probability distribution $P(x)$

Conditional Generative Model: Learn a probability distribution $P(x|y)$

These models may seem to be very distinct.

Although, they are actually **not fully** distinct.

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Discriminative vs. Generative Models

Discriminative Model: Learn a probability distribution $P(y|x)$

Generative Model: Learn a probability distribution $P(x)$

Conditional Generative Model: Learn a probability distribution $P(x|y)$

Recall **Bayes' Rule:**

$$\underbrace{\boxed{P(x \mid y)}}_{\text{Conditional Generative Model}} = \frac{\overbrace{\boxed{P(y \mid x)}}^{\text{Discriminative Model}}}{\underbrace{\boxed{P(y)}}_{\text{Prior over labels}}} \overbrace{\boxed{P(x)}}^{\text{(Unconditional) Generative Model}}$$

We can build a conditional generative model from other components!

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Discriminative vs. Generative Models

Discriminative Model: Learn a probability distribution $P(y|x)$ → Assign labels to data; Feature learning (with labels)

Generative Model: Learn a probability distribution $P(x)$ → Detect outliers; Feature learning (without labels); **Sample** to generate new data

Conditional Generative Model: Learn a probability distribution $P(x|y)$ → Assign labels, while rejecting outliers!; Generate new data conditioned on input labels

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Taxonomy of Generative Models

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Taxonomy of Generative Models

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Autoencoder

Representation Learning

The choice of representation is **critical**. Each type of representation makes some operations easy and others hard.

Q: How can we **come up with the best representation possible**?

Image from: Torralba, A. and Isola, P. and Freeman, W.T., Foundations of Computer Vision, MIT Press, https://visionbook.mit.edu

Representation Learning

Goal of representation learning

The goal is to learn to map from datapoints, $\mathbf{x} \in \mathcal{X}$, to abstract representation, $\mathbf{z} \in \mathcal{Z}$

$$f : \mathrm{X} \to \mathcal{Z}$$

$\mathbf{z}$ is called a **vector embedding** of $\mathbf{x}$

Image from: Torralba, A. and Isola, P. and Freeman, W.T., Foundations of Computer Vision, MIT Press, https://visionbook.mit.edu

Representation Learning

What Makes for a Good Representation?

A good representation is one that makes subsequent problem solving easier.

- **Compression**
  - Compressed representations require less memory to store.
  - Compression is a way to achieve invariance to nuisance factors.
- **Prediction**
  - Facilitating the prediction of important world properties—the future, the past, mental states, cause and effect, and so on—is perhaps the defining property of a good representation.

Image from: Torralba, A. and Isola, P. and Freeman, W.T., Foundations of Computer Vision, MIT Press, https://visionbook.mit.edu

Autoencoder

Autoencoders are one of the **oldest and most common kinds of representation learners**.

Image from: Torralba, A. and Isola, P. and Freeman, W.T., Foundations of Computer Vision, MIT Press, https://visionbook.mit.edu

Autoencoder

Architecture

Unsupervised method for learning feature vectors from raw data x, without any labels.
"Autoencoding" = encoding itself

Features should extract useful information (maybe object identities, properties, scene type, etc) that we can use for downstream tasks

Problem: How can we **learn** this **feature transform** from raw data?

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Autoencoder

Architecture

Idea: Use the features to **reconstruct the input data** with a decoder

"Autoencoding" = encoding itself

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Autoencoder

Architecture

Idea: Use the features to **reconstruct the input data** with a decoder

"Autoencoding" = encoding itself

"Encoder" learns mapping from the data, $\mathcal{X}$, to a low-dimensional latent space, $\mathcal{Z}$

"Decoder" learns mapping back from latent space, $\mathcal{Z}$, to reconstructed observation, $\hat{x}$

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Autoencoder

Architecture

Loss: L2 distance between input and reconstructed data.

Does not use any labels! Just raw data!

Features need to be **lower dimensional** than the data

x

Input data

Encode

z

Features

Decode

$\hat{x}$

Reconstructed input data

$$\|\hat{x} - x\|_2^2$$

Loss function

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Autoencoder

Latent vector

Autoencoding is a form of compression! **Size of latent vector** affects the reconstructed data quality.

2D latent space

5D latent space

Ground truth

Slide adopted from: A. Amini, et al., MIT 6.S191 Introduction to Deep Learning, MIT, http://introtodeeplearning.com/

Autoencoder

Representation (Latent) Space

Latent Space of MNIST dataset

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Image from: Torralba, A. and Isola, P. and Freeman, W.T., Foundations of Computer Vision, MIT Press, https://visionbook.mit.edu

Autoencoder

Autoencoders (Regular, non-variational)

After training, **throw away decoder** and use encoder for a downstream task

Autoencoder

Autoencoders (Regular, non-variational)

After training, **throw away decoder** and use encoder for a downstream task

Encoder can be used to initialize a supervised model.

Train for final task
(sometimes with small data)

Autoencoder

Applications

- Dimensionality Reduction and Feature Extraction
- Denoising Data
- Anomaly detections

Image from: https://www.v7labs.com/blog/autoencoders-guide

Autoencoder

Autoencoders (Regular, non-variational)

Autoencoders learn **latent features** for data without any labels!

Can use features to initialize a supervised model

Not probabilistic: No way to sample **new data** from learned model

Autoencoder

Autoencoders (Regular, non-variational)

There are "gaps" in the latent space, where data is never mapped to.

If we sample a latent vector from a **region** in the **latent space that was never seen** by the decoder during training, the output might not make any sense at all.

the latent space, z, can become **disjoint** and **non-continuous**.

Image from: https://avandekleut.github.io/vae/

Variational

Autoencoder

Generative Modeling meets Representation Learning

- Deep nets transform datapoints, layer by layer
- Each layer is a different representation of the data

Representation Learning (analysis):
maps data to abstract representations (**embeddings**).

Generative Modeling (synthesis):
map abstract representations to data

Image from: Torralba, A. and Isola, P. and Freeman, W.T., Foundations of Computer Vision, MIT Press, https://visionbook.mit.edu

Generative Modeling meets Representation Learning

Latent Variables as Representations

Image from: Torralba, A. and Isola, P. and Freeman, W.T., Foundations of Computer Vision, MIT Press, https://visionbook.mit.edu

Variational Autoencoder

The Decoder of an Autoencoder is a Data Generator

Image from: Torralba, A. and Isola, P. and Freeman, W.T., Foundations of Computer Vision, MIT Press, https://visionbook.mit.edu

Variational autoencoders (VAEs)

are a way to turn an autoencoder into a proper generative model, which can be sampled from and which maximizes data likelihood under a formal probabilistic model.

The trick is very simple: just take a vanilla autoencoder and (1) regularize the latent distribution to squish it into a Gaussian (or some other base distribution), (2) add noise to the output of the encoder.

Variational Autoencoder

Variational autoencoders are a probabilistic twist on autoencoders!
Sample from the mean and standard deviation to compute latent sample

Overview of VAE : Instead of learning an arbitrary function, the network learns the parameters of **probability distribution** modeling the data

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Variational Autoencoder

Intuition

Image from: https://www.jeremyjordan.me/variational-autoencoders/

Variational Autoencoder

Intuition

Using a variational autoencoder, we can describe latent attributes in probabilistic terms.

Image from: https://www.jeremyjordan.me/variational-autoencoders/

Variational Autoencoder

Intuition

After training with lots of samples, the probability distribution of each attribute is learned. So that we can represent each latent attribute as a probability distribution.

Image from: https://www.jeremyjordan.me/variational-autoencoders/

Variational Autoencoder

Intuition

Values which are nearby to one another in latent space should correspond with very similar reconstructions.

Image from: https://www.jeremyjordan.me/variational-autoencoders/

Variational Autoencoder

Probabilistic spin on autoencoders: we want to do

1. **Learn latent features** z from raw data
2. Sample from the model to **generate new data**

Variational Autoencoder

Probabilistic spin on autoencoders: we want to do

1. **Learn latent features** z from raw data
2. Sample from the model to **generate new data**

After training, we want to **sample** a hidden variable z which **generate** an observation x

We can only see x, but we want to infer the attributes of z, i.e., we want to compute

$$p(z|x) = \frac{p(x|z)p(z)}{p(x)}$$

Variational Autoencoder

Probabilistic spin on autoencoders: we want to do

1. **Learn latent features** z from raw data
2. Sample from the model to **generate new data**

After training, we want to **sample** a hidden variable z which **generate** an observation x

We can only see x, but we want to infer the attributes of z, i.e., we want to compute

compute with
Decoder network

we assumed
Gaussian prior

$$p(z|x) = \frac{p(x|z)p(z)}{p(x)}$$

Very difficult to compute (technically possible, but practical impossible)

Variational Autoencoder

Probabilistic spin on autoencoders: we want to do

1. **Learn latent features** z from raw data
2. Sample from the model to **generate new data**

After training, we want to **sample** a hidden variable z which **generate** an observation x

We can only see x, but we want to infer the attributes of z, i.e., we want to compute

compute with
Decoder network

we assumed
Gaussian prior

$$p(z|x) = \frac{p(x|z)p(z)}{p(x)}$$

$$p(x) = \int p(x|z)p(z)dz$$

Impossible to integrate over all z (intractable)

Variational Autoencoder

Probabilistic spin on autoencoders: we want to do

1. **Learn latent features** z from raw data
2. Sample from the model to **generate new data**

We can approximate $p(z|x)$ by another distribution $q(z|x)$, i.e.

Compute with Encoder Network

$$\boxed{q(z|x)} \approx p(z|x)$$

the KL divergence is a measure of difference between two probability distributions. Thus, if we wanted to ensure that $q(z|x)$ was similar to $p(z|x)$, we could minimize the KL divergence between the two distributions.

$$minD_{KL}(q(z|x)||p(z|x))$$

Variational Autoencoder

Common choice of prior - Gaussian Distribution

$$p(z) = \mathcal{N}(\mu, \sigma)$$

Probabilistic spin on autoencoders: we want to do

1. **Learn latent features** z from raw data
2. Sample from the model to **generate new data**

The reconstruction likelihood

$$\boxed{E_{q(z|x)} \log p(x|z)} - \boxed{D_{KL}(q(z|x)||p(z))}$$

Regularization Loss: ensures that our **learned distribution** is similar to the true **prior distribution**

Variational Autoencoder

Why regularization term is needed

Only reconstruction loss

Only KL divergence

Combination

Image from: https://www.jeremyjordan.me/variational-autoencoders/

Variational Autoencoder

Why regularization term is needed

Autoencoder

Variational Autoencoder

Image from: https://www.jeremyjordan.me/variational-autoencoders/

Variational Autoencoder

How to perform backpropagation when training training the model

$$E_{q(z|x)} \log p(x|z) - D_{KL}(q(z|x)||p(z))$$

Variational Autoencoder

Reparameterization trick

Image from: https://www.jeremyjordan.me/variational-autoencoders/

Variational Autoencoder

Latent perturbation

Slowly increase or decrease a **single latent variable**. Keep all other variables fixed.

Head pose

Different dimension of z encodes **different interpretable latent features**

Image from: https://www.jeremyjordan.me/variational-autoencoders/

Variational Autoencoder

Latent perturbation

Image from: https://www.jeremyjordan.me/variational-autoencoders/

Ideally, we want latent variables that are uncorrelated with each other.

Enforce diagonal prior on the latent variables to encourage independence

Disentanglement

Generative Adversarial Network

Generative Adversarial Network (GAN)

What if we just want to sample

Idea: do not explicitly model density, and instead just sample to generate new instances.

Problem: want to sample from complex distribution - cannot do this directly

Solution: sample from something simple (e.g., noise), learn a transformation to the data distribution.

Slide adopted from: A. Amini, et al., MIT 6.S191 Introduction to Deep Learning, MIT, http://introtodeeplearning.com/

Generative Adversarial Networks (GANs)

GANs are a way to make a generative model by having two neural networks compete with each other

The **generator** turns noise into an imitation of the data to try to trick the discriminator.

The **discriminator** tries to identify real data from fakes created by the generator.

Slide adopted from: A. Amini, et al., MIT 6.S191 Introduction to Deep Learning, MIT, http://introtodeeplearning.com/

Generative Adversarial Network (GAN)

Intuition behind GANs

Discriminator looks at both real and fake data created by the generator

Generator starts from noise to try to create an imitation of the data

Slide adopted from: A. Amini, et al., MIT 6.S191 Introduction to Deep Learning, MIT, http://introtodeeplearning.com/

Generative Adversarial Network (GAN)

Intuition behind GANs

Discriminator tries to predict what's real and what's fake

Generator starts from noise to try to create an imitation of the data

Slide adopted from: A. Amini, et al., MIT 6.S191 Introduction to Deep Learning, MIT, http://introtodeeplearning.com/

Generative Adversarial Network (GAN)

Intuition behind GANs

Discriminator tries to predict what's real and what's fake

Generator starts from noise to try to create an imitation of the data

Real data

Fake data

Slide adopted from: A. Amini, et al., MIT 6.S191 Introduction to Deep Learning, MIT, http://introtodeeplearning.com/

Generative Adversarial Network (GAN)

Intuition behind GANs

Discriminator tries to predict what's real and what's fake

Generator tries to improve its imitation of the data

Slide adopted from: A. Amini, et al., MIT 6.S191 Introduction to Deep Learning, MIT, http://introtodeeplearning.com/

Generative Adversarial Network (GAN)

Intuition behind GANs

Discriminator tries to predict what's real and what's fake

Generator tries to improve its imitation of the data

Slide adopted from: A. Amini, et al., MIT 6.S191 Introduction to Deep Learning, MIT, http://introtodeeplearning.com/

Generative Adversarial Network (GAN)

Intuition behind GANs

Discriminator tries to identify real data from fakes created by the generator.

Generator tries to create imitations of data to trick the discriminator.

Slide adopted from: A. Amini, et al., MIT 6.S191 Introduction to Deep Learning, MIT, http://introtodeeplearning.com/

Generative Adversarial Networks (GANs)

Training GANs

Training: adversarial objective for **D** and **G**

Global optimum: G reproduces the true data distribution.

G tries to synthesize fake instances that fool **D**

D tries to identify the synthesized instances

Slide adopted from: A. Amini, et al., MIT 6.S191 Introduction to Deep Learning, MIT, http://introtodeeplearning.com/

Generative Adversarial Networks (GANs)

Training GANs

Jointly train generator G and discriminator D with a **minimax game**

Discriminator wants $D(x) = 1$ for real data

Discriminator wants $D(x) = 0$ for fake data

$$\min_{G} \max_{D} \left(E_{x \sim p_{data}} [\log D(x)] + E_{z \sim p(z)} \left[\log \left(1 - D(G(z)) \right) \right] \right)$$

Generator wants $D(x) = 1$ for fake data

Generative Adversarial Networks (GANs)

Training GANs

Jointly train generator G and discriminator D with a **minimax game**

Training G and D using alternating gradient updates

$$\min_{G} \max_{D} \left(E_{x \sim p_{data}} [\log D(x)] + E_{z \sim p(z)} \left[\log \left(1 - D(G(z)) \right) \right] \right)$$

$$= \min_{G} \max_{D} V(G, D)$$

For t in 1, … T:

1. (Update D) $D = D + \alpha_D \frac{\partial V}{\partial D}$
2. (Update G) $G = G - \alpha_G \frac{\partial V}{\partial G}$

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Generative Adversarial Network (GAN)

Generating new data with GANs

After training, use generator network to create **new data** that's never been seen before.

Slide adopted from: A. Amini, et al., MIT 6.S191 Introduction to Deep Learning, MIT, http://introtodeeplearning.com/

Generative Adversarial Network (GAN)

GANs are distribution transformers

After training, use generator network to create **new data** that's never been seen before.

Slide adopted from: A. Amini, et al., MIT 6.S191 Introduction to Deep Learning, MIT, http://introtodeeplearning.com/

Generative Adversarial Network (GAN)

GANs are distribution transformers

After training, use generator network to create **new data** that's never been seen before.

Slide adopted from: A. Amini, et al., MIT 6.S191 Introduction to Deep Learning, MIT, http://introtodeeplearning.com/

Generative Adversarial Network (GAN)

GANs are distribution transformers

After training, use generator network to create **new data** that's never been seen before.

Slide adopted from: A. Amini, et al., MIT 6.S191 Introduction to Deep Learning, MIT, http://introtodeeplearning.com/

Generative Adversarial Network (GAN)

Results

Nearest neighbor from training set

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Generative Adversarial Network (GAN)

Variations - DC GAN

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Generative Adversarial Network (GAN)

Variations - DC GAN : Results

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Generative Adversarial Network (GAN)

Variations - DC GAN : Results

Interpolating between points in latent *z* space, show that the space learned has smooth transitions

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Generative Adversarial Network (GAN)

Variations - DC GAN : Vector Math

Smiling woman | Neutral woman | Neutral man

Samples from the model

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Generative Adversarial Network (GAN)

Variations - DC GAN : Vector Math

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Generative Adversarial Network (GAN)

Variations - DC GAN : Vector Math

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Generative Adversarial Network (GAN)

Variations - DC GAN : Vector Math

Samples from the model

Man with glasses

Man w/o glasses

Woman w/o glasses

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Generative Adversarial Network (GAN)

Variations - DC GAN : Vector Math

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Generative Adversarial Network (GAN)

Variations - Progressively Growing GAN

Slide adopted from: A. Amini, et al., MIT 6.S191 Introduction to Deep Learning, MIT, http://introtodeeplearning.com/

Generative Adversarial Network (GAN)

Variations - StyleGAN

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Karras et al, "A Style-Based Generator Architecture for Generative Adversarial Networks", CVPR 2019
Video is licensed under CC BY-NC 4.0.
Source: https://drive.google.com/drive/folders/1NFO7_vH0t98J13ckJYFd7kuaTkyeRJ86

Generative Adversarial Network (GAN)

Variations - StyleGAN

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Cherdsak Kingkan

Karras et al, "A Style-Based Generator Architecture for Generative Adversarial Networks", CVPR 2019
Video is licensed under CC BY-NC 4.0.
Source: https://drive.google.com/drive/folders/1NFO7_vH0t98J13ckJYFd7kuaTkyeRJ86

Generative Adversarial Network (GAN)

Variations - Conditional GAN

Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, Alexei A. Efros
Image-to-Image Translation with Conditional Adversarial Networks, CVPR, 2017

Generative Adversarial Network (GAN)

Variations - Image-to-Image Translation: Pix2Pix

Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, Alexei A. Efros
Image-to-Image Translation with Conditional Adversarial Networks, CVPR, 2017

Generative Adversarial Network (GAN)

Variations - Cycle GAN

Distribution transformations.

Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, Alexei A. Efros
Image-to-Image Translation with Conditional Adversarial Networks, CVPR, 2017

Generative Adversarial Network (GAN)

Variations - Cycle GAN

Domain transformation: CycleGAN learns transformations across domains with unpaired data.

Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, Alexei A. Efros
Image-to-Image Translation with Conditional Adversarial Networks, CVPR, 2017

Generative Adversarial Network (GAN)

Variations - Cycle GAN

Summarize

Autoencoder (AE)

Variational Autoencoder (VAE)

Generative Adversarial Network (GAN)

Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, Alexei A. Efros
Image-to-Image Translation with Conditional Adversarial Networks, CVPR, 2017