Computer Vision – TP13
Generative Models

Miguel Coimbra, Hélder Oliveira

Outline

- Generative Models
- Autoencoders
- Variational autoencoders (VAEs)
- Generative adversial networks (GANs)

Outline

- Generative Models
- Autoencoders
- Variational autoencoders (VAEs)
- Generative adversial networks (GANs)

Supervised vs. Unsupervised

- Supervised learning
  - We have access to a set of training data for which we know the correct class/answer
  - Training data:$\{(x_i, y_i)\}_{i=1}^{N}$
  - x_i: data (e.g., image)
  - y_i: label
- Examples
  - Image classification
  - Image segmentation
  - Object detection
  - Etc.

DOG, DOG, CAT

Object Detection

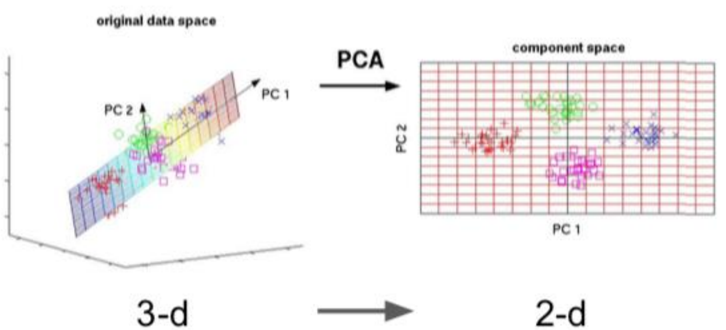
GRASS, CAT, TREE, SKY

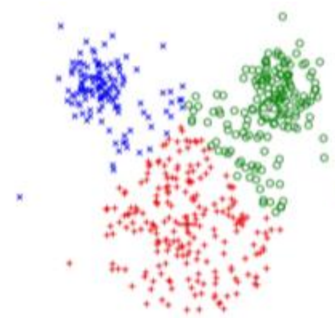
Semantic Segmentation

Supervised vs. Unsupervised

- Unsupervised learning
  - Discover hidden structures in the data
  - Training data: $\{x_i\}_{i=1}^{N}$
  - x_i: only data (e.g., image), no label!

- Examples
  - Clustering
  - Dimensionality reduction
  - Generative models
  - Etc.




K-means clustering

Generative models

- Given a set of training data, learn their distribution and generate new data from a similar distribution

Training data ~ $p_{data}(x)$ Generated samples ~ $p_{model}(x)$

Want to learn $p_{model}(x)$ similar to $p_{data}(x)$

- Explicit: returns $p_{model}(x)$
- Implicit: generate samples only

Applications

- Generate new data for simulations
  - Reinforcement learning
- Generate new data for model training
  - Data augmentation
- Fill in the gaps of measured data
  - Super-resolution
  - Colorization
- Inference of latent representation

Outline

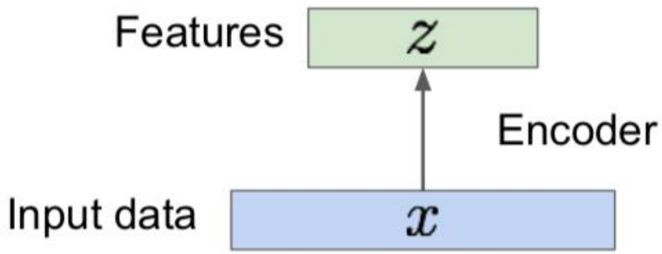
- Generative Models
- Autoencoders
- Variational autoencoders (VAEs)
- Generative adversial networks (GANs)

Autoencoders

- Objective
  - Find representative features of the data
- Unsupervised learning
  - No data labels required
- Simple idea
  - Learn a representation of the data and try to recover the original data from that!

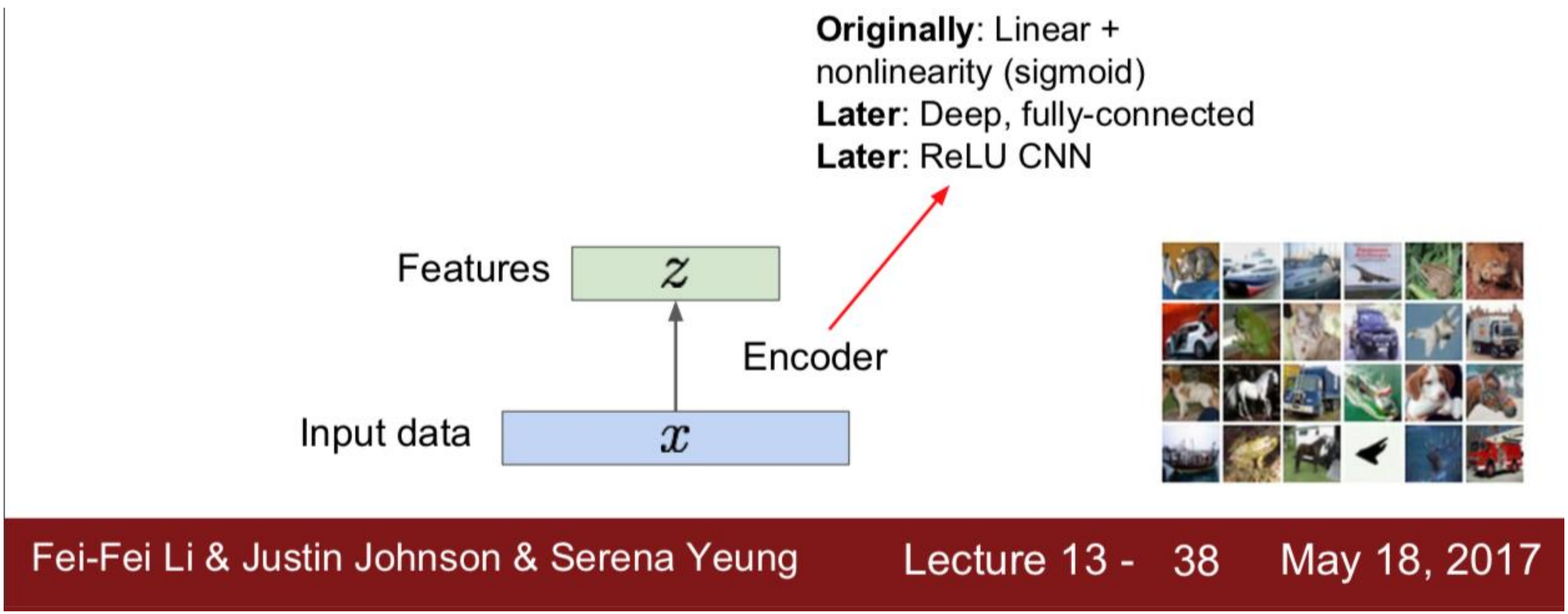
Autoencoders

- Representative features



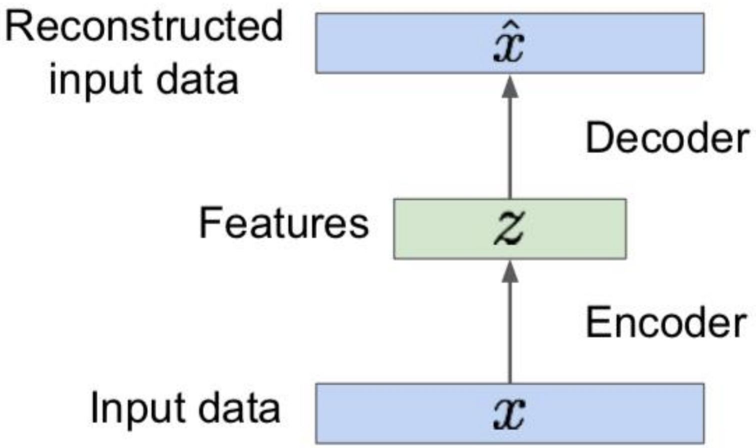
Autoencoders

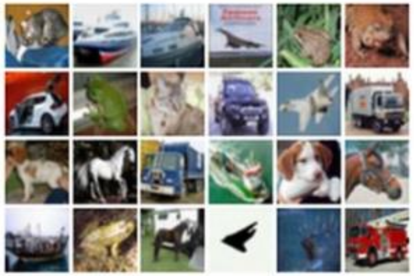
- Representative features



Autoencoders

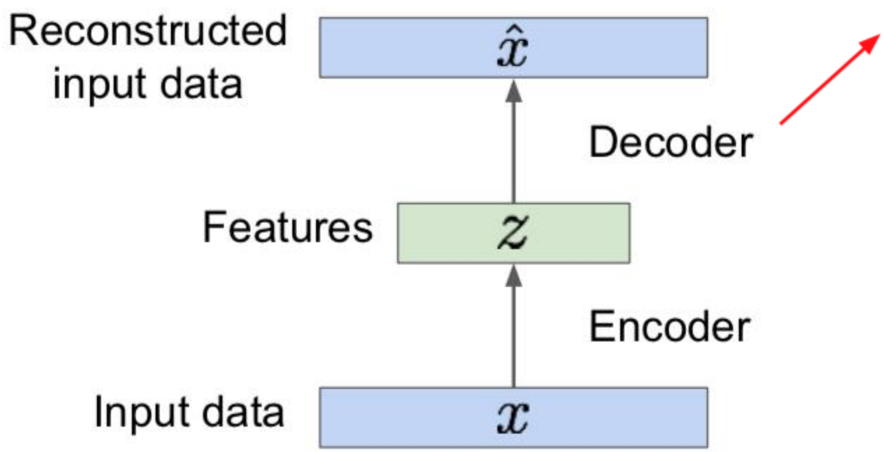
- Reconstruction



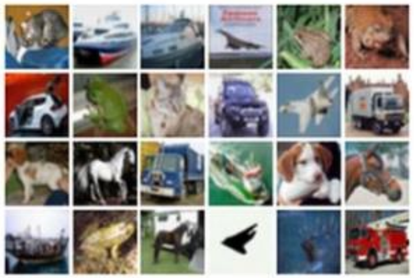


Autoencoders

- Reconstruction

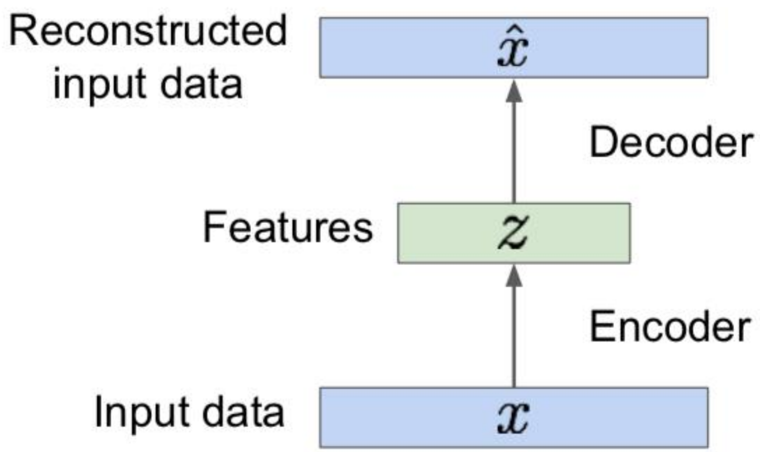


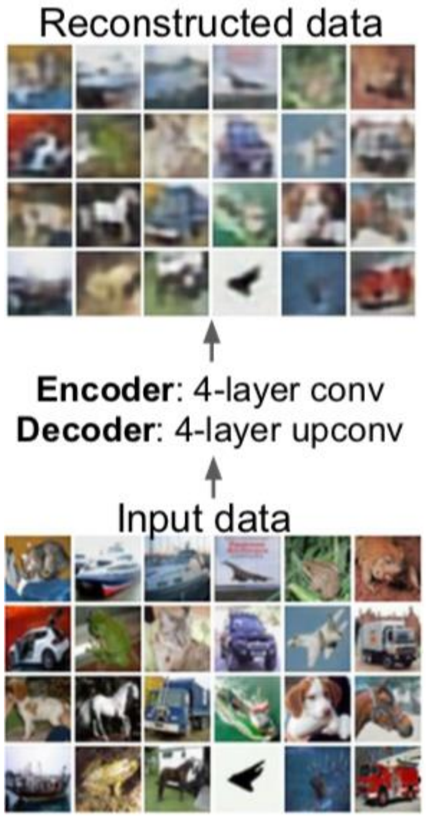
Originally: Linear + nonlinearity (sigmoid)
Later: Deep, fully-connected
Later: ReLU CNN (upconv)



Autoencoders

- Reconstruction

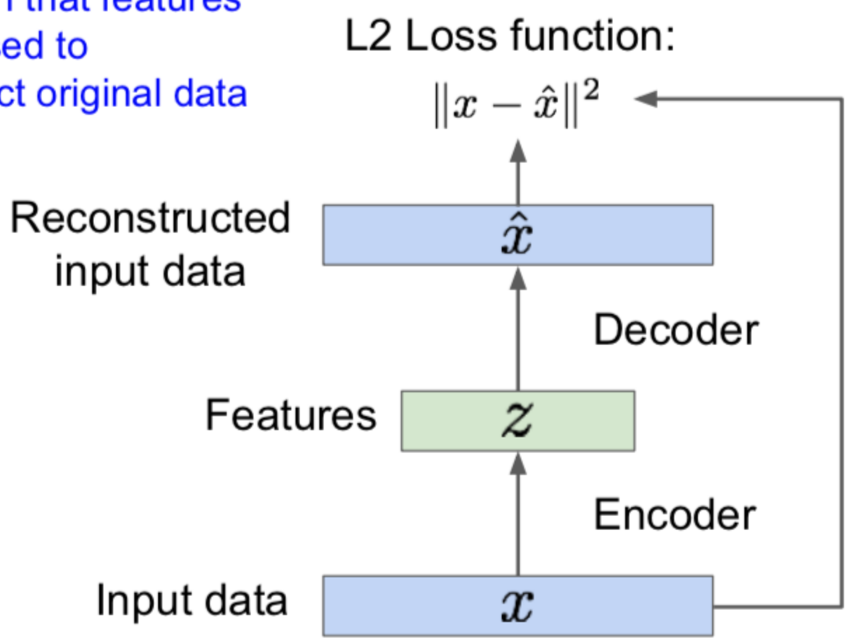


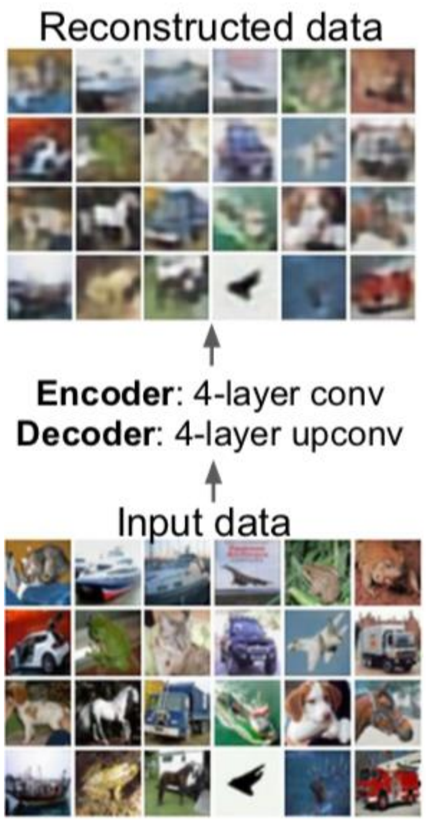


Autoencoders

- Training

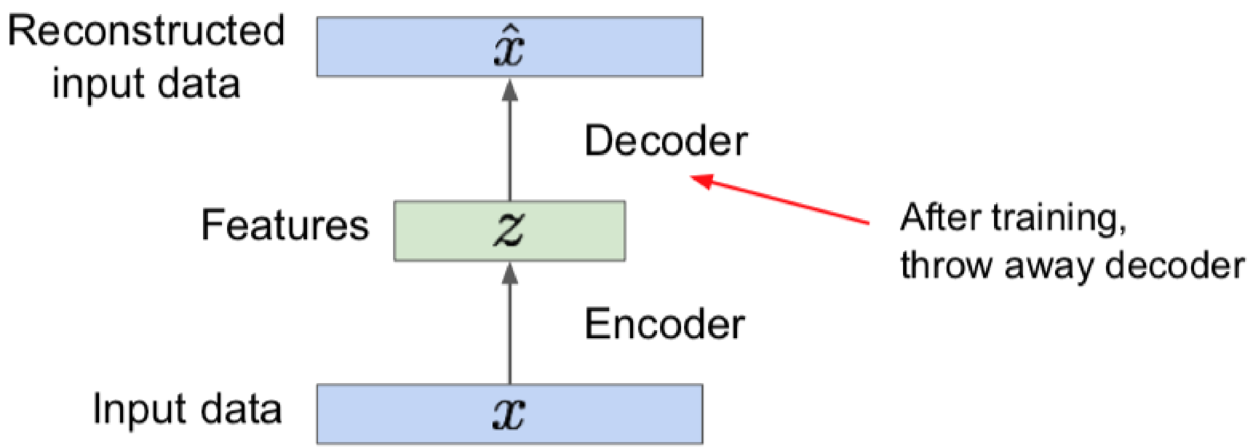
Train such that features
can be used to
reconstruct original data





Autoencoders

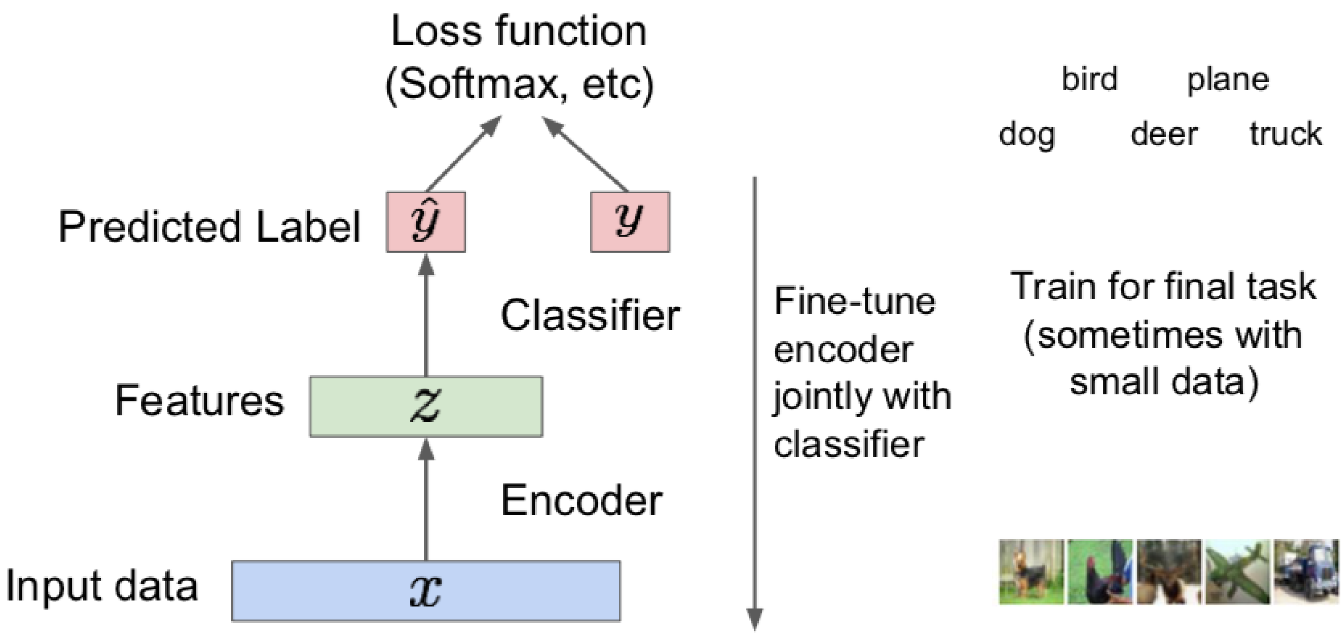
- Use the learned features for other tasks!



Autoencoders

- Use the learned features for other tasks!

Encoder can be used to initialize a **supervised** model



Avoid trivial solutions

- Undercomplete: dim(z) << dim(x)
  - Forces to capture the most salient features
  - Dimensionality reduction
  - Capture meaningful factors of variation
- Regularized
  - Encourage the model to have some properties

Sparse Autoencoders

- Code sparsity

$$LOSS = \|x - \hat{x}\|_2^2 + \|z\|_1$$

  - Helps learning good features for classification
  - Forces a (Laplace) prior on latent representation
  - Different from weight regularization! Why?

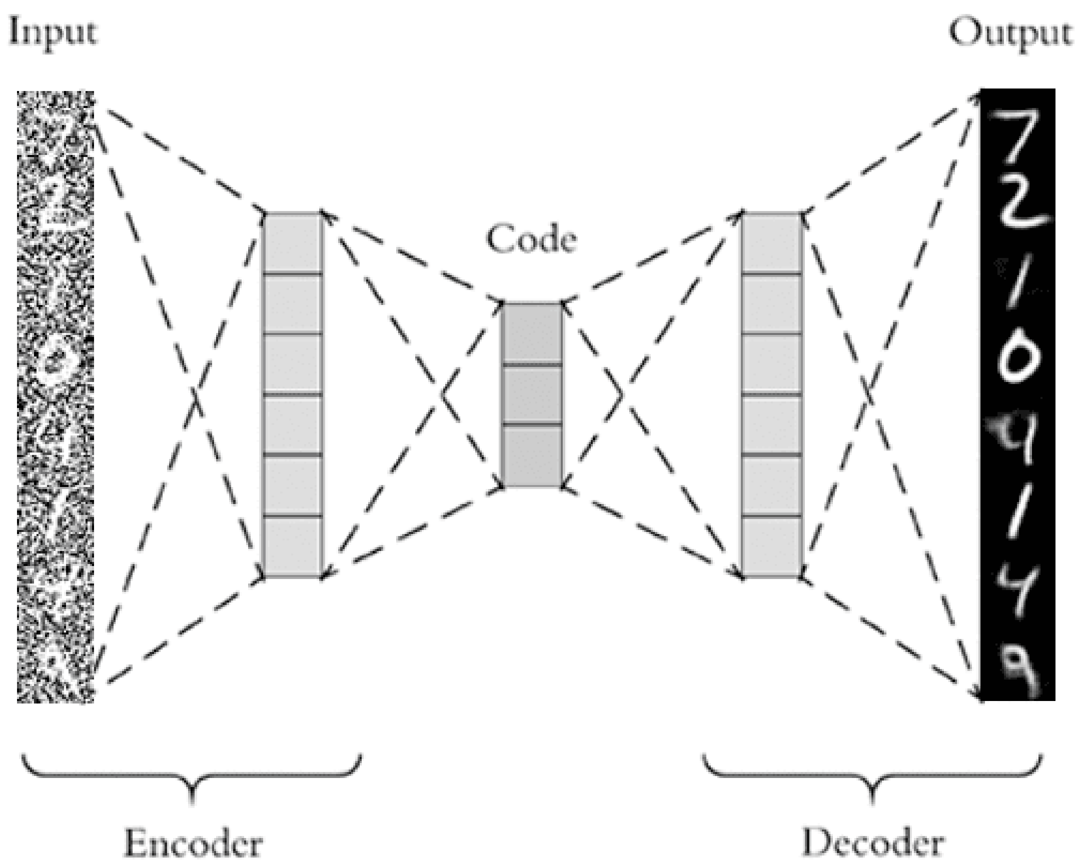
Denoising Autoencoders

- Definition
  - Encoder function: $z = E(x)$
  - Decoder function: $\hat{x} = D(z)$
  - Noisy version of data: $\tilde{x} = x + noise$
  - Denoising autoencoder:

$$LOSS_{den} = \|x - D(E(\tilde{x}))\|_2^2$$

- Implicitly learns the structure of the data

Denoising Autoencoders



https://www.pyimagesearch.com/2020/02/24/denoising-autoencoders-with-keras-tensorflow-and-deep-learning/

Outline

- Generative Models
- Autoencoders
- Variational autoencoders (VAEs)
- Generative adversial networks (GANs)

Variational Autoencoders

- Idea: we can use the autoencoder approach to generate data from a specific distribution
- Training: data sampled from such distribution
- Use autoencoder to generate the statistical description of the data

Variational Autoencoders

- Idea
  - Encoder and decoder provide **distributions** (their parameters), not data points!
- Assumptions
  - Training data $\{x_i\}_{i=1}^{N}$
  - $p(z)$ Gaussian distribution
  - $p(x|z)$ Gaussian distribution (Encoder)
  - $p(z|x)$ approximated by a Gaussian distribution (Decoder)

Variational Autoencoders

- Training
  - Use a variational lower bound of the log-likelihood $\log p(x_i)$
- Generate data
  - Sample z from a Gaussian prior
  - Use decoder to get (Gaussian) $p(x|z)$
  - Sample $x|z$ from $p(x|z)$

Training

- Data likelihood intractable to compute:

$$p_\theta(x) = \int p_\theta(z) p_\theta(x|z) dz$$

- Use a network to model encoder distribution

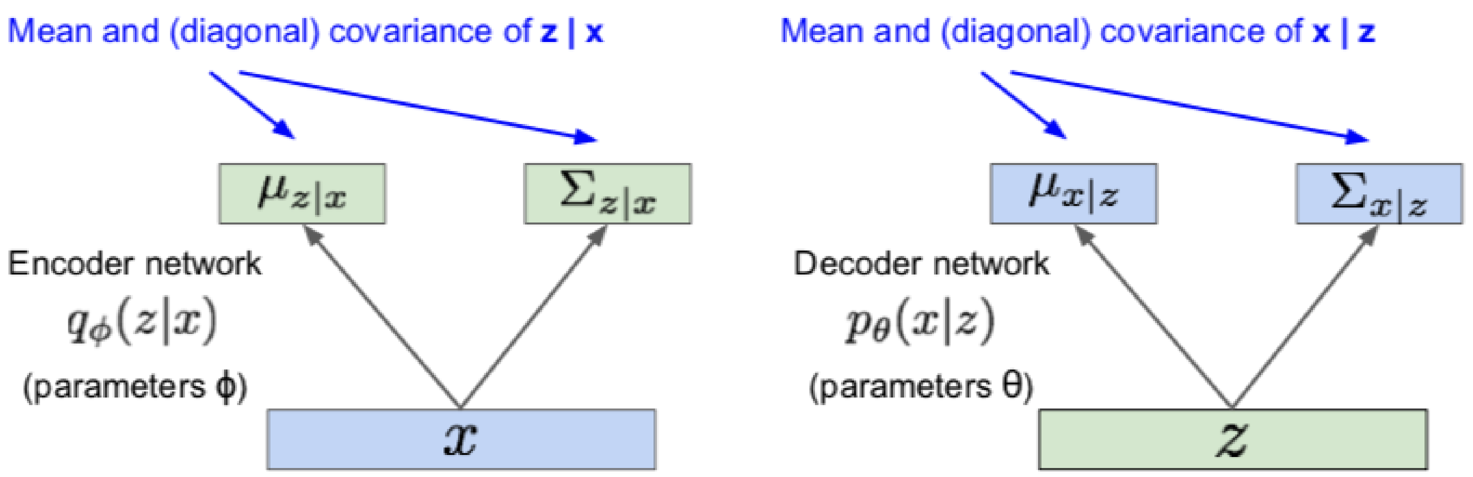
$$q_\phi(z|x) \approx p(z|x)$$

- Then, optimize a lower bound of the data likelihood:

$$\log p_\theta(x) \geq E_z[\log p_\theta(x|z)] - D_{KL}\left(q_\phi(z|x) \middle\| p_\theta(z)\right)$$

Training

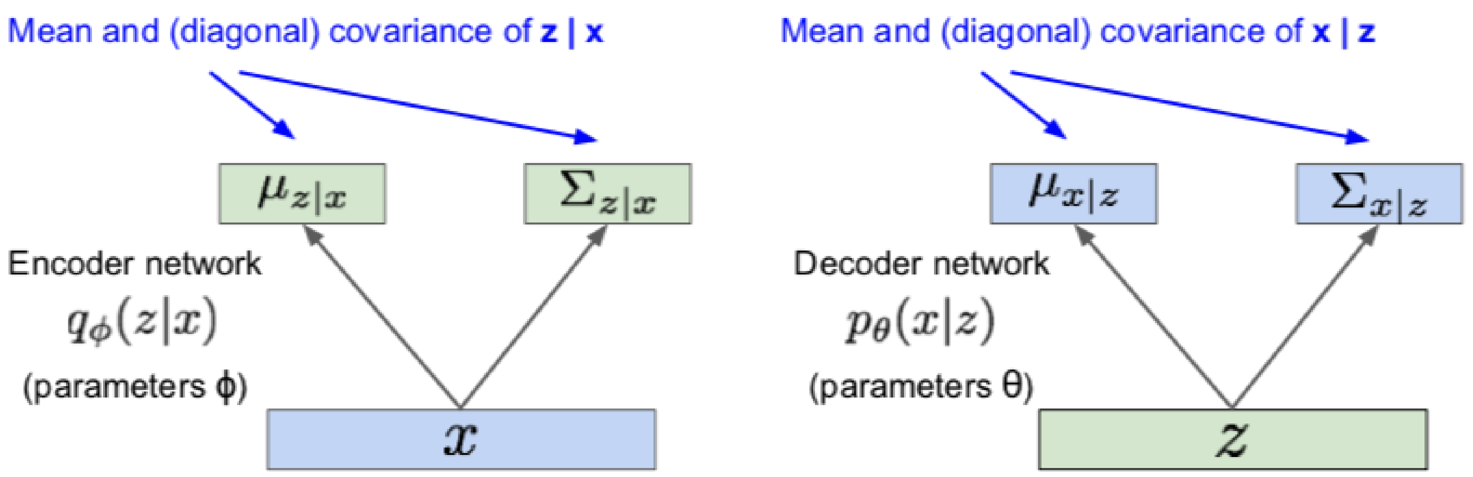
- Encoder and decoder networks return distribution parameters



Kingma and Welling, "Auto-Encoding Variational Bayes", ICLR 2014

Training

- Sample from these distributions to get latent z and image x



Kingma and Welling, "Auto-Encoding Variational Bayes", ICLR 2014

Training

Putting it all together: maximizing the likelihood lower bound

$$\underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)}, \theta, \phi)}$$

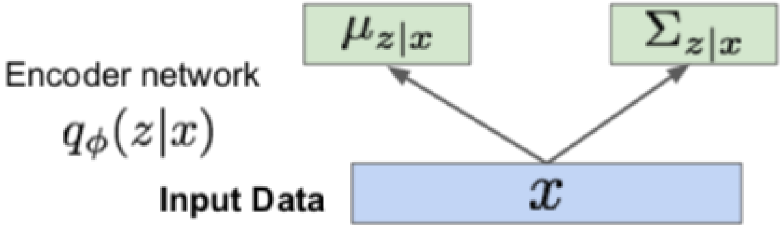
Let's look at computing the bound (forward pass) for a given minibatch of input data

Input Data x

Training

Putting it all together: maximizing the likelihood lower bound

$$\underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)}, \theta, \phi)}$$

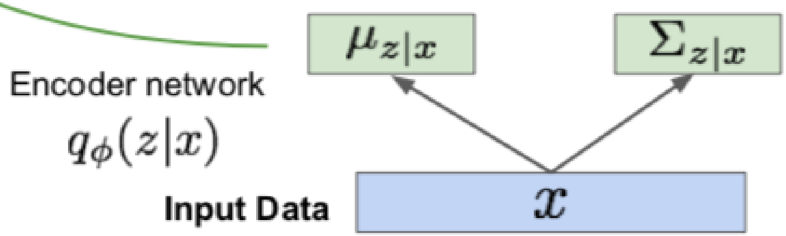


Training

Putting it all together: maximizing the likelihood lower bound

$$\underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)},\theta,\phi)}$$

Make approximate
posterior distribution
close to prior

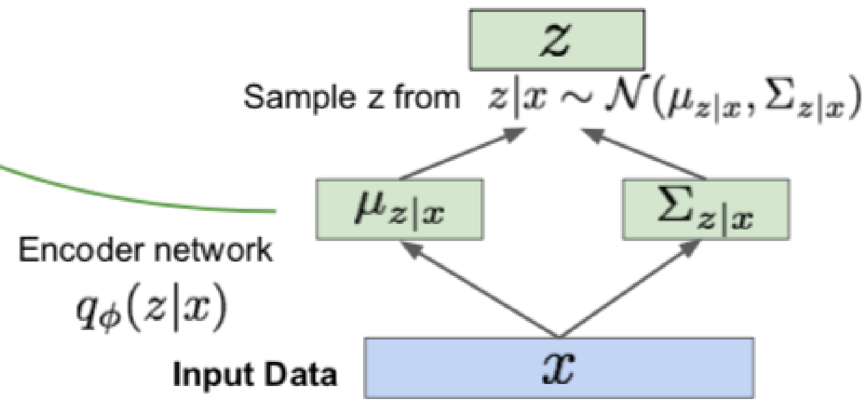


Training

Putting it all together: maximizing the likelihood lower bound

$$\underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)}, \theta, \phi)}$$

Make approximate posterior distribution close to prior

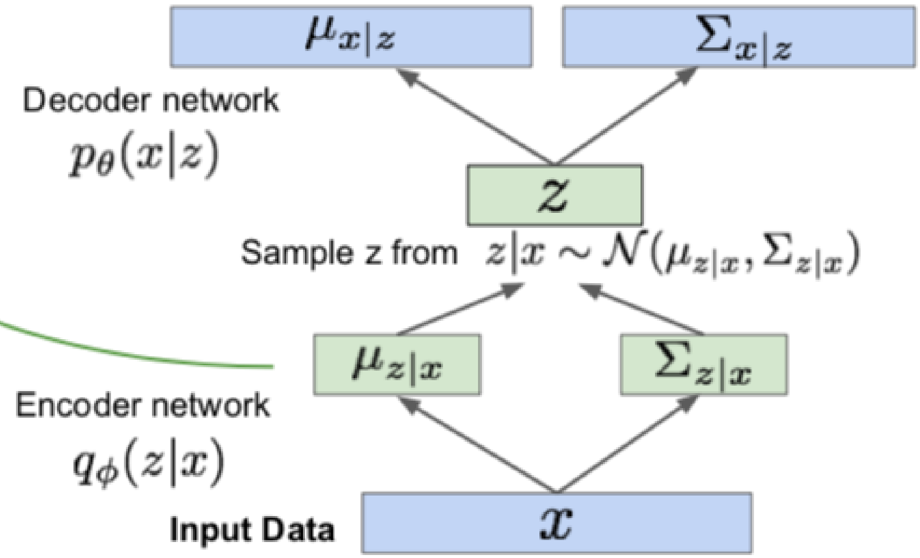


Training

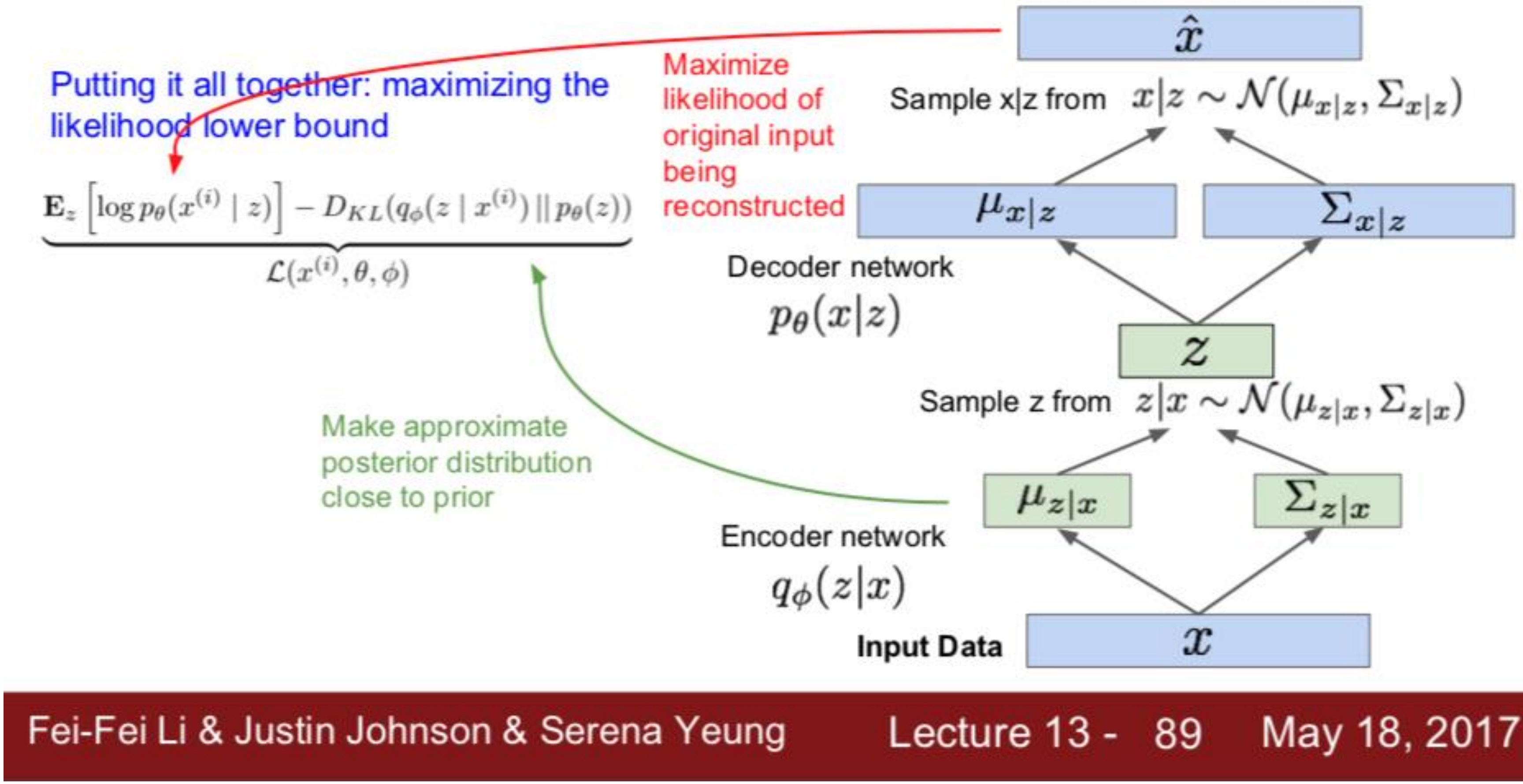
Putting it all together: maximizing the likelihood lower bound

$$\underbrace{\mathbf{E}_z \left[\log p_\theta(x^{(i)} \mid z) \right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)}, \theta, \phi)}$$

Make approximate posterior distribution close to prior



Training


Putting it all together: maximizing the likelihood lower bound
$\underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)},\theta,\phi)}$
Maximize likelihood of original input being reconstructed
Make approximate posterior distribution close to prior
$\hat{x}$
Sample x|z from $x|z \sim \mathcal{N}(\mu_{x|z}, \Sigma_{x|z})$
$\mu_{x|z}$
$\Sigma_{x|z}$
Decoder network
$p_\theta(x|z)$
z
Sample z from $z|x \sim \mathcal{N}(\mu_{z|x}, \Sigma_{z|x})$
$\mu_{z|x}$
$\Sigma_{z|x}$
Encoder network
$q_\phi(z|x)$
Input Data
x
Fei-Fei Li & Justin Johnson & Serena Yeung
Lecture 13 - 89
May 18, 2017

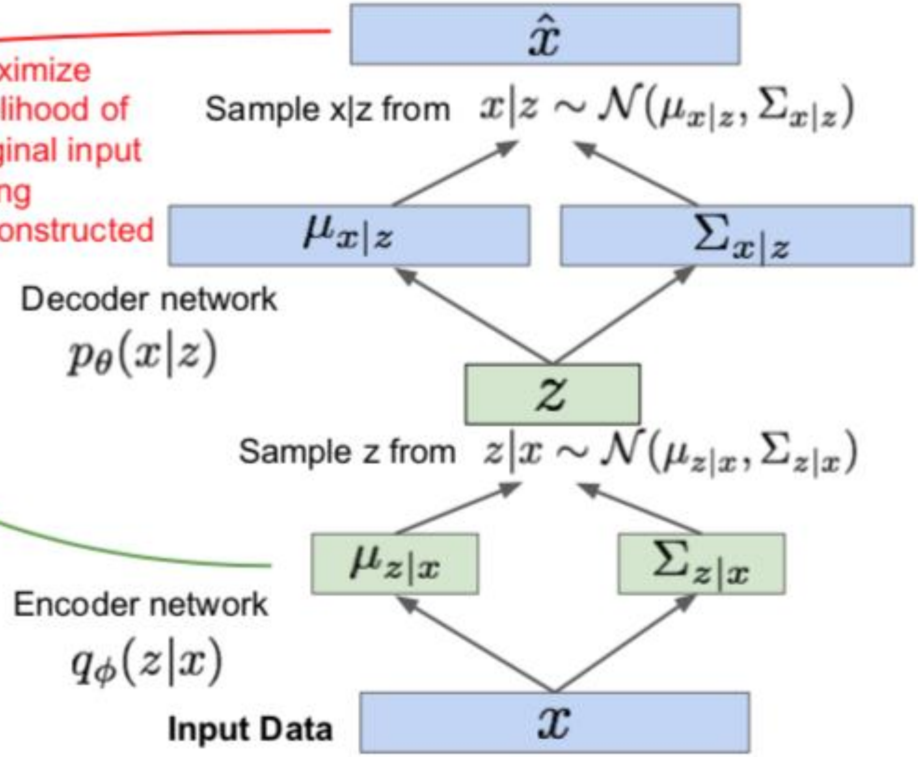
Training

Putting it all together: maximizing the likelihood lower bound

$$\underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)},\theta,\phi)}$$

Maximize likelihood of original input being reconstructed

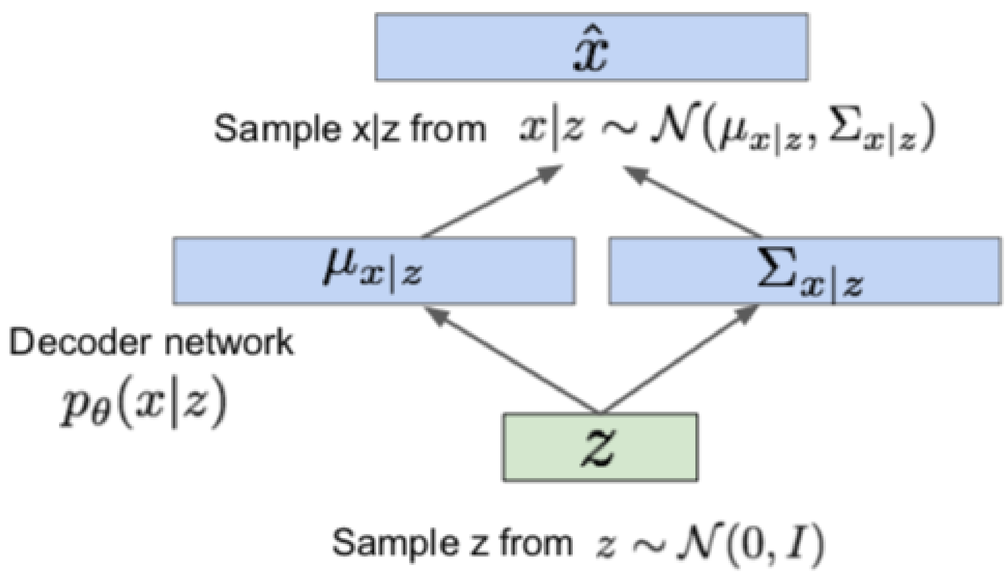
Make approximate posterior distribution close to prior



For every minibatch of input data: compute this forward pass, and then backprop!

Generating data

1. Sample z from prior
2. Use decoder network
3. Sample from Gaussian posterior



Kingma and Welling, "Auto-Encoding Variational Bayes", ICLR 2014

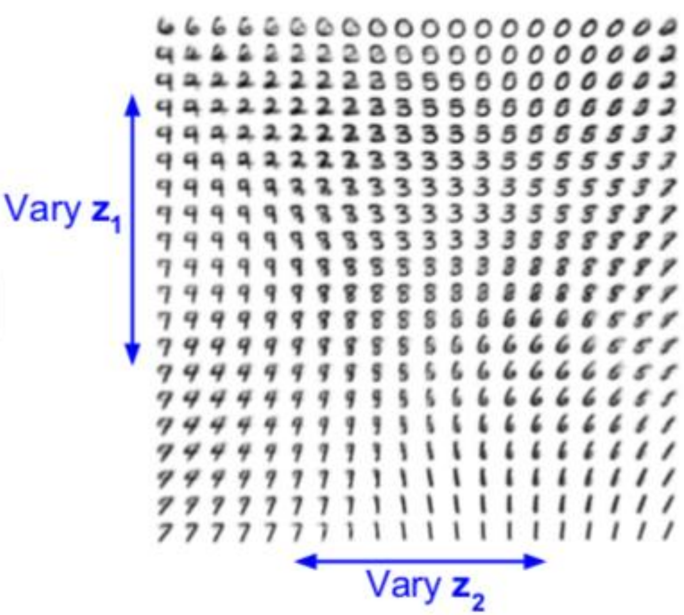
VAEs results

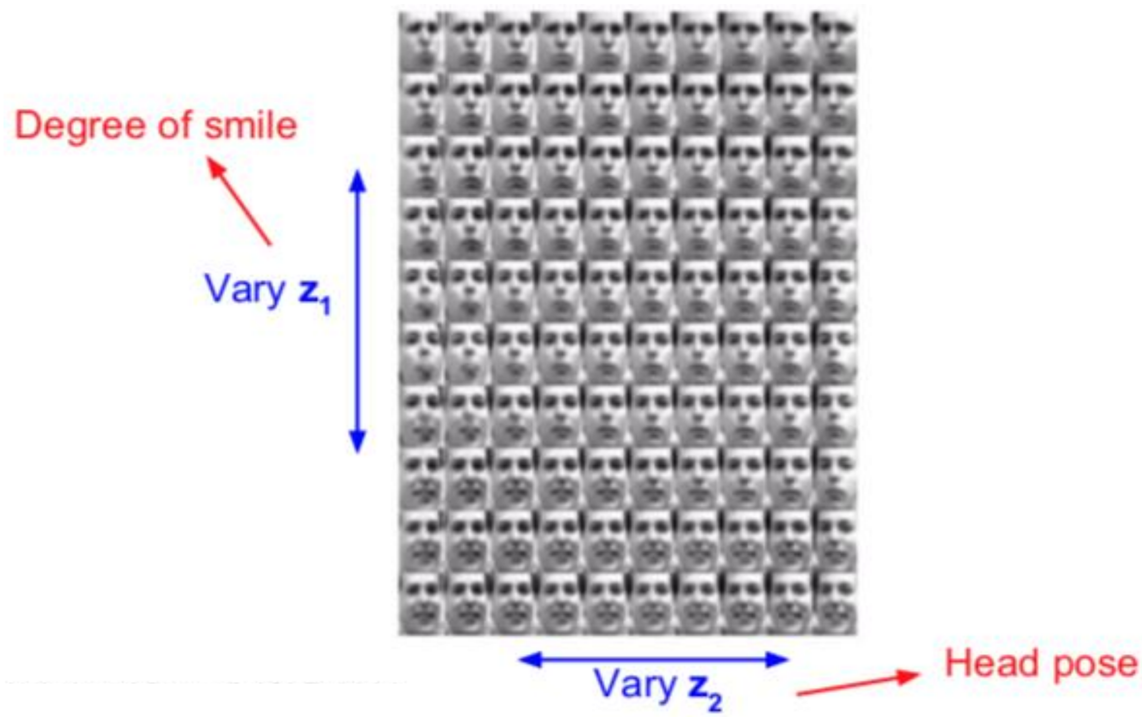
32x32 CIFAR-10

Labeled Faces in the Wild

Latent representations

- z contains independent factors!



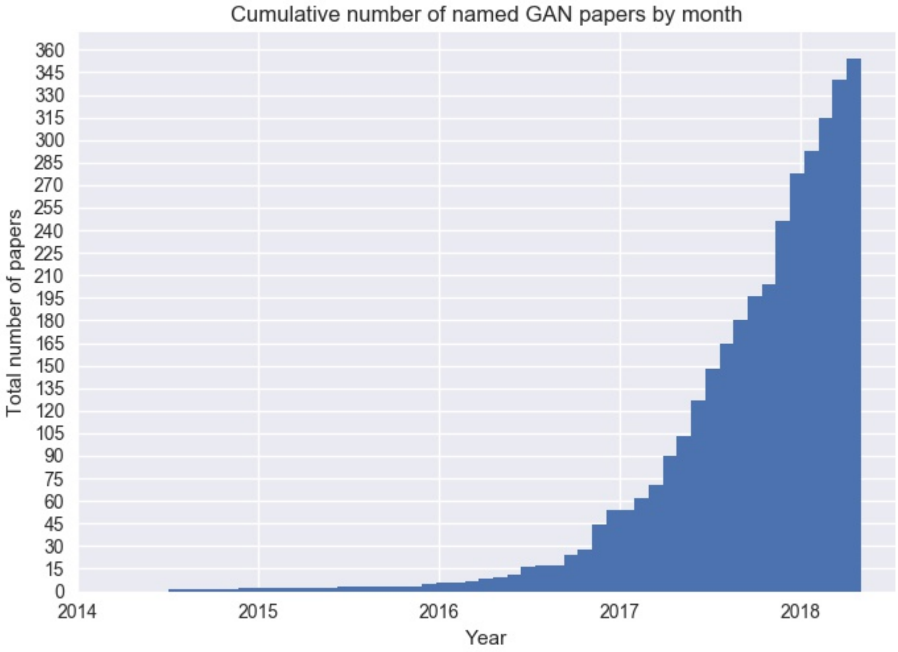


Outline

- Generative Models
- Autoencoders
- Variational autoencoders (VAEs)
- Generative adversial networks (GANs)

GANs taking over

Track updates at the GAN Zoo



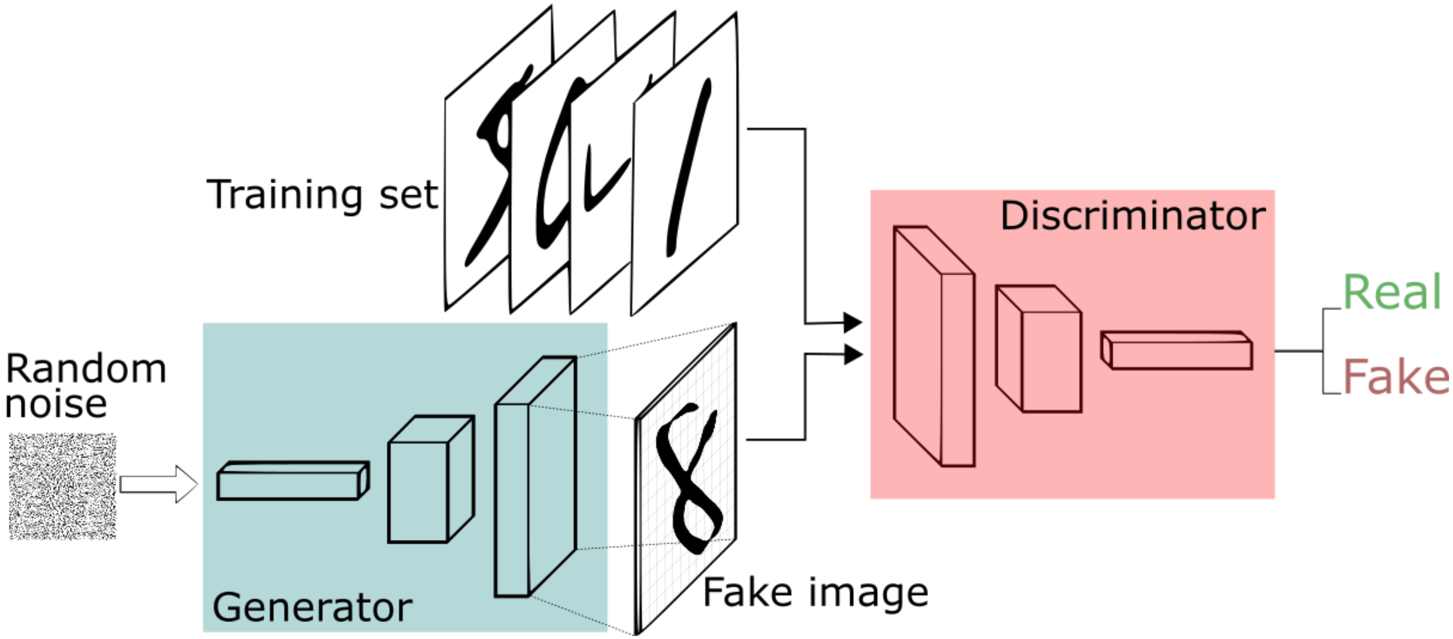
https://github.com/hindupuravinash/the-gan-zoo

(Goodfellow 2018)

Generative adversarial networks

- Main idea
  - In contrast with other models (e.g., VAEs), it does not provide explicit information about data distribution (i.e., density function)
  - Generate samples only
- How?
  - Define an adversarial two-player game between a **generator** and a **discriminator**

Generative adversarial networks



https://sthalles.github.io/intro-to-gans/

Generator

Learns a transformation from a simple noise distribution to training distribution (e.g., images)

Training samples

z

Input noise vector

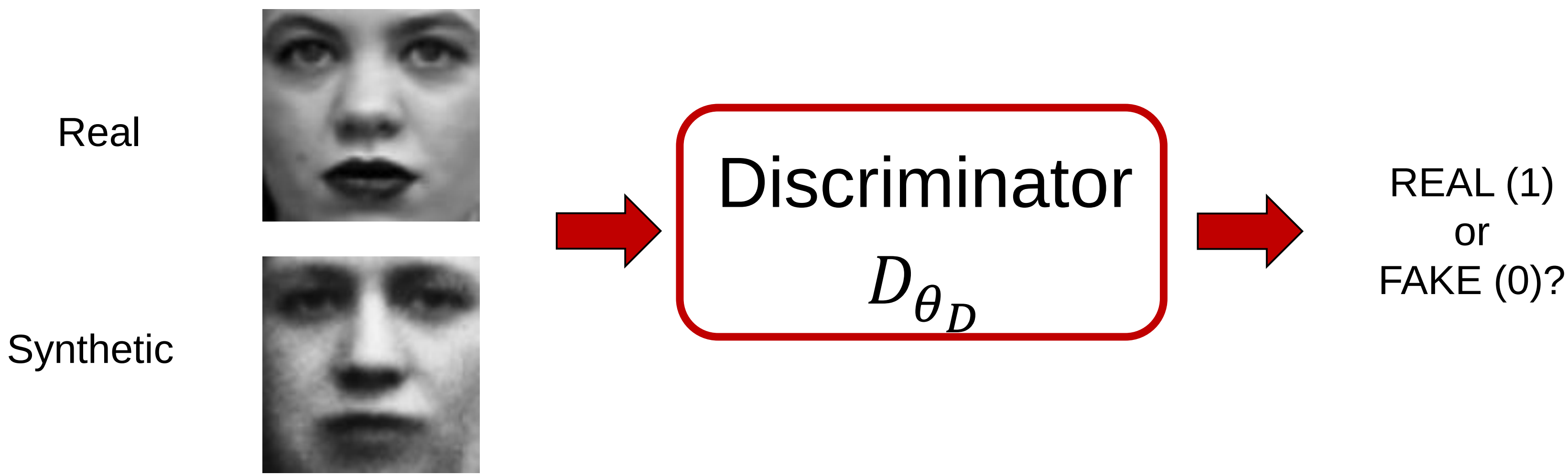
Generator

G_{θ_G}

Synthetic sample

Discriminator

Distinguish between real and fake images



Training GANs

- Train jointly the generator and the discriminator in a minimax game
  - Generator: try to fool the discriminator by generating real-looking images
  - Discriminator: try to distinguish between real and generated images

Training GANs

- Minimax objective function

$$\min_{\theta_G} \max_{\theta_D} E_{p_{train}} \log D_{\theta_D}(x) + E_{p\,(z)} \log(1 - D_{\theta_D}\left(G_{\theta_G}(z)\right))$$

  - **Discriminator**: maximize the objective so that D(x) is close to 1 (real) and D(G(z)) close to 0 (fake)
  - **Generator**: minimize the objective so that D(G(z)) is close to 1 (the discriminator is fooled)
  - Solved via alternating gradient optimization

Generating data

- After training, just use the generator to generate new samples

z

Input noise vector

Generator

G_{θ_G}

Synthetic sample

- Useful side-effect:
  – Discriminator can be used for transfer learning (why?)

GANs architectures

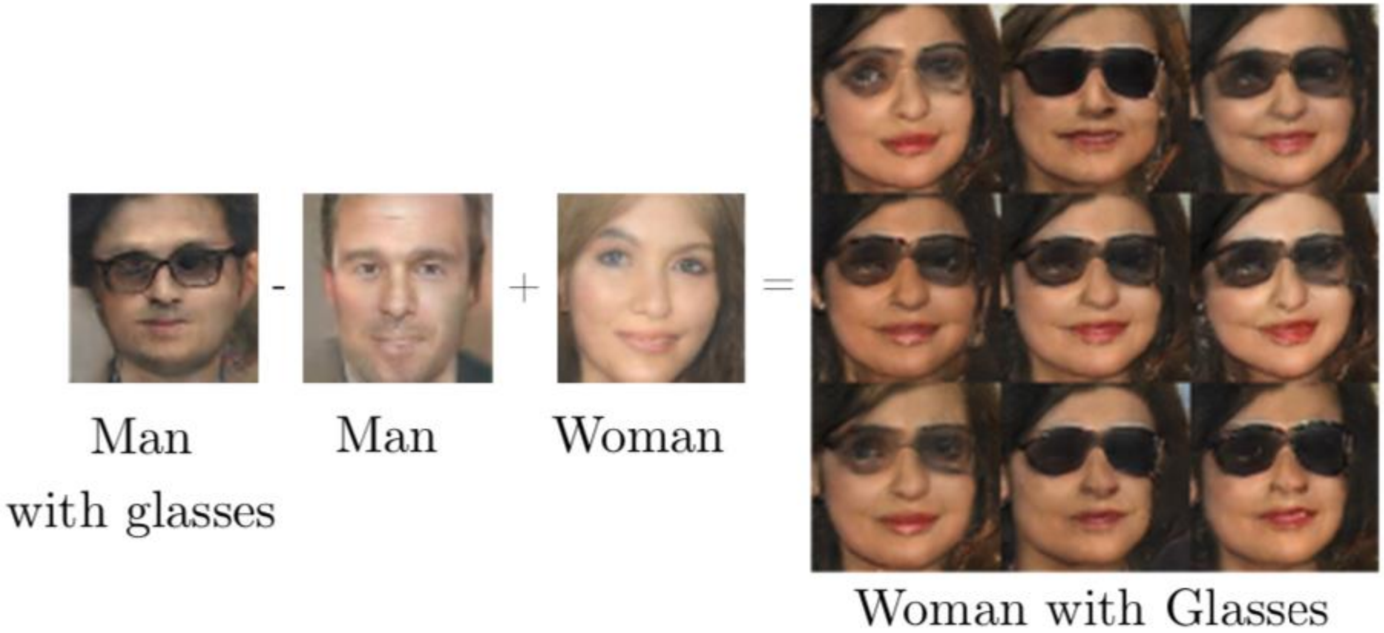
- Image GANs mainly use CNNs
- Generator
  - Upsampling + convolutional layers: similar to decoder in AEs
- Discriminator
  - Similar to classification CNN

GANs results

Karras T, Aila T, Laine S, Lehtinen J. Progressive growing of gans for improved quality, stability, and variation. arXiv preprint arXiv:1710.10196. 2017 Oct 27.

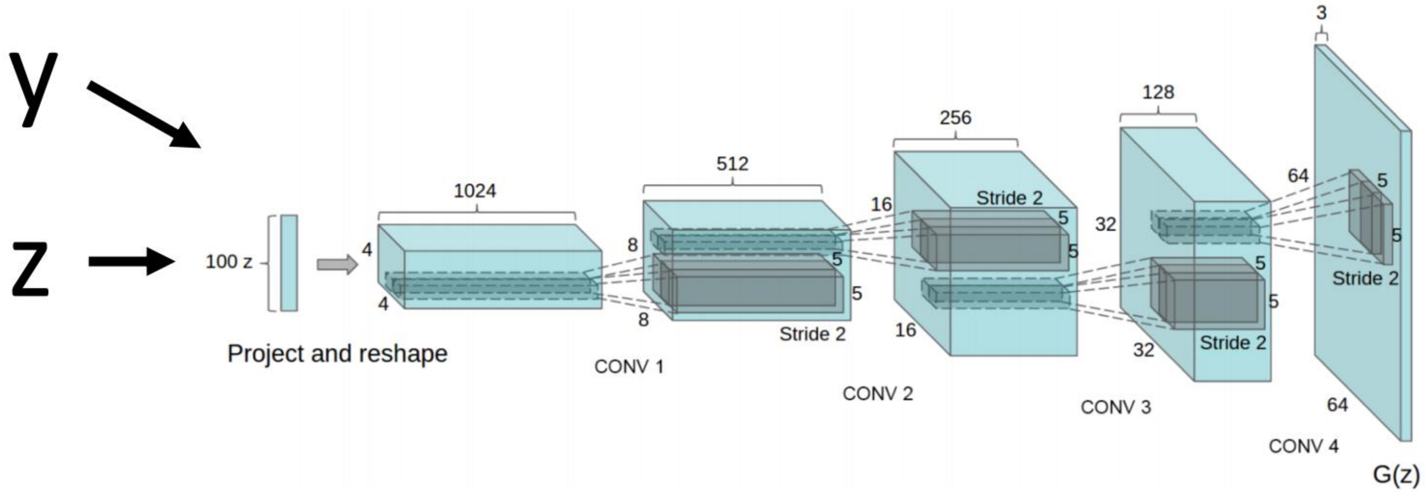
Latent representations

- Interpretable vector math



Radford A, Metz L, Chintala S. Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434. 2015 Nov 19.

Conditional Gans

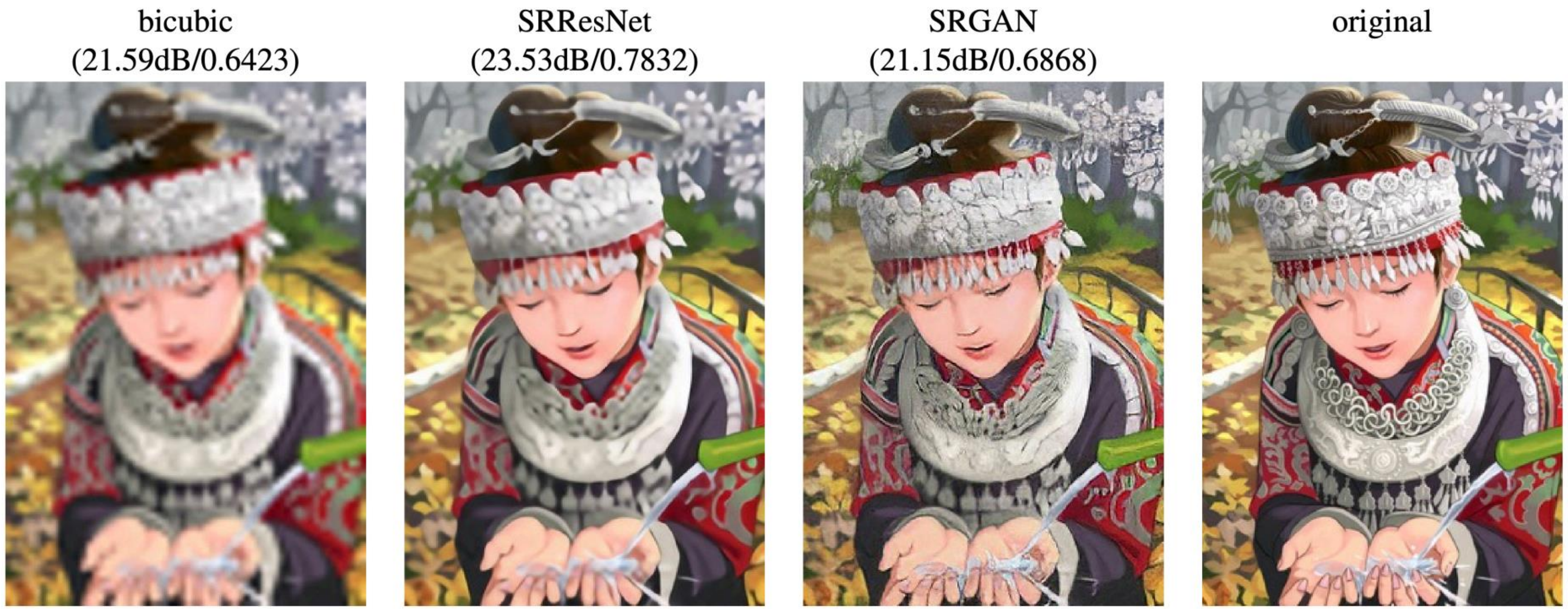


Radford et al, “Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks”, ICLR 2016

Conditional Gans

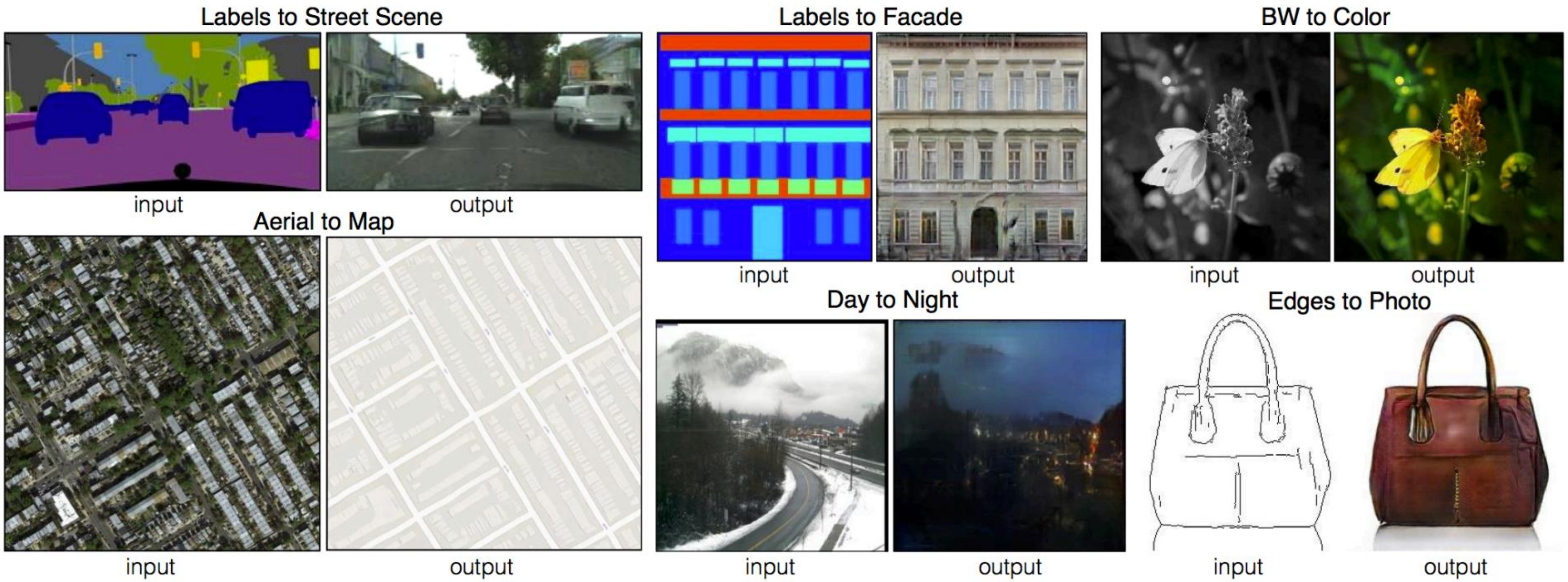
Miyato et al, “Spectral Normalization for Generative Adversarial Networks”, ICLR 2018

Super Resolution



Ledig et al, “Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network”, CVPR 2017

Image to Image



Isola et al, “Image-to-Image Translation with Conditional Adversarial Nets”, CVPR 2017

Cycle Gan

Input Video: Horse | Output Video: Zebra

https://www.youtube.com/watch?v=9reHvktowLY

Generative models: VAEs vs GANs

- VAEs
  - ☺ Principled (max likelihood) approach
  - ☺ Provide explicit distributions
  - ☹ Provide blurry samples
- GANs
  - ☺ Beautiful, state-of-the-art samples
  - ☹ Trickier to train (unstable)
  - ☹ Do not provide explicit distributions

Resources

- F.F. Li, J. Johnson, S. Young. Convolutional Neural Networks for Visual Recognition, Stanford University, 2017
  - Lecture 13- ”Generative models”
  - http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture13.pdf
- I. Goodfellow, Y. Bengio, and A. Courville. Deep learning. Cambridge: MIT press, 2016.
  - Chapter 14 – “Autoencoders”
  - Chapter 20 – “Deep Generative Models”

Some links

- Generative Models I (Michigan Online)
  - https://www.youtube.com/watch?v=Q3HU2vEhD5Y
- Generative Models II (Michigan Online)
  - https://www.youtube.com/watch?v=igP03FXZqgo