CAI 4104/6108 – Machine Learning Engineering:

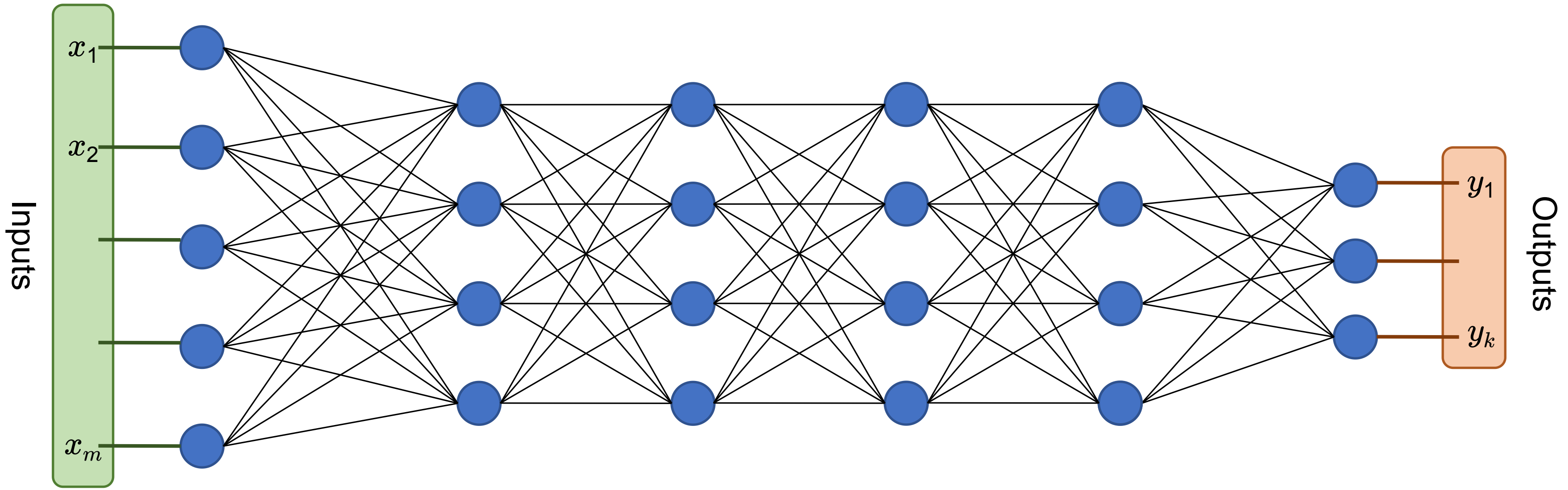
Auto-Encoders & GANs

Prof. Vincent Bindschaedler

Spring 2024

Reminder: Deep Neural Networks

- What is a deep neural network?
  - Any neural network with two or more hidden layers
  - Nowadays, the best neural networks architectures for many applications & problems are deep
    - E.g.: AlexNet (2012) has 8 layers, ResNet18 has 18 layers, GPT-2 has 48 layers



AutoEncoders and GANs

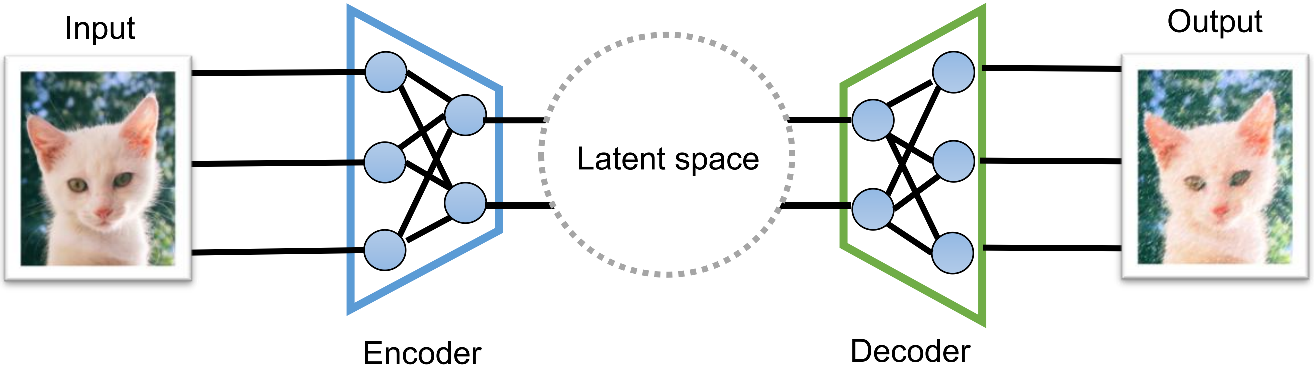
- AutoEncoders
  - Architecture combining an encoder network and a decoder network
  - Learn efficient representations of the data
    - Each data point can be represented in the latent space
  - Applications: dimensionality reduction and feature learning

- Generative Adversarial Networks (GANs)
  - Novel idea: adversarial learning/training

- Generative models
  - Some models can actually generate new data instances
  - E.g.: some autoencoders, GANs, etc.

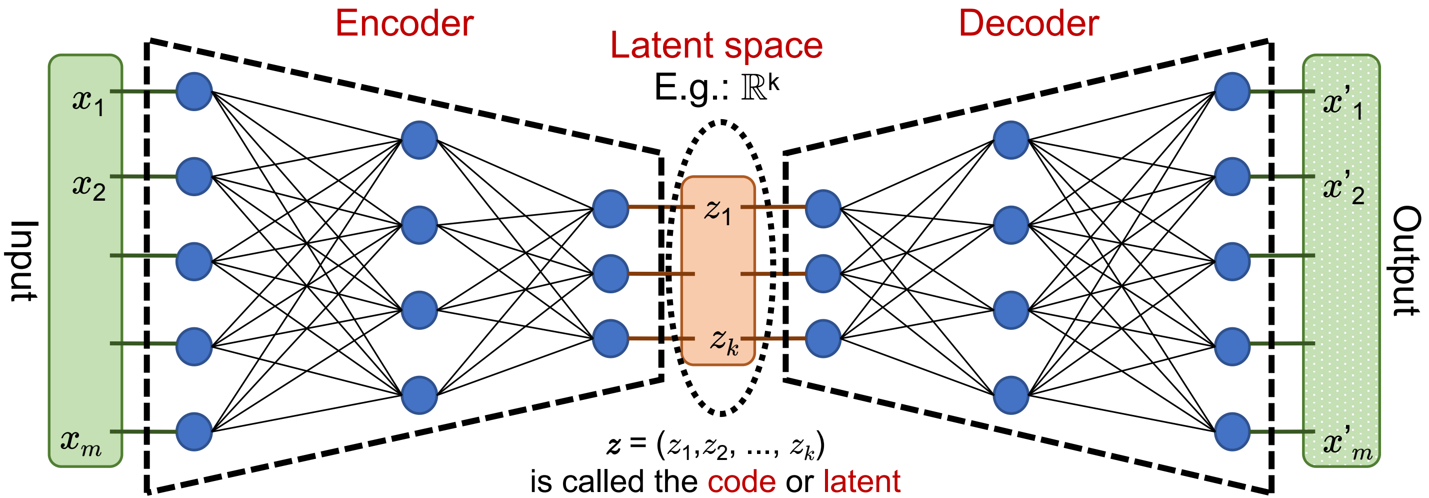
AutoEncoders

- Encoder-Decoder network
  - Goal: learn to reproduce the input as output
  - Constraints:
    - The latent representation (aka codings) is constrained (e.g., must have lower dimensionality than input)
  - Effect: network must learn an efficient way to represent the information



AutoEncoders

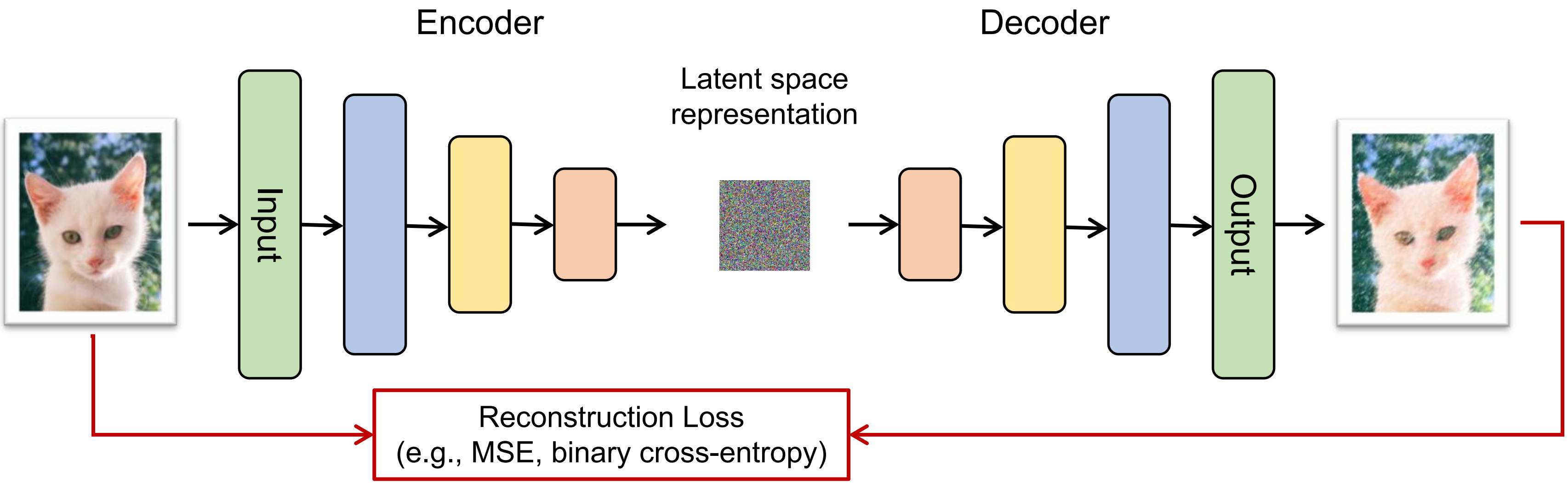
- Encoder-Decoder network
  - Goal: learn to reproduce the input as output
  - Constraints:
    - The latent representation (aka codings) is constrained (e.g., must have lower dimensionality than input)
  - Effect: network must learn an efficient way to represent the information



$\boldsymbol{z} = (z_1, z_2, \ldots, z_k)$ is called the code or latent representation of $\boldsymbol{x}$

Training AutoEncoders

- Encoder-Decoder network
  - Goal: learn to reproduce the input as output
  - Constraints:
    - The latent representation (aka codings) is constrained (e.g., must have lower dimensionality than input)
  - Effect: network must learn an efficient way to represent the information

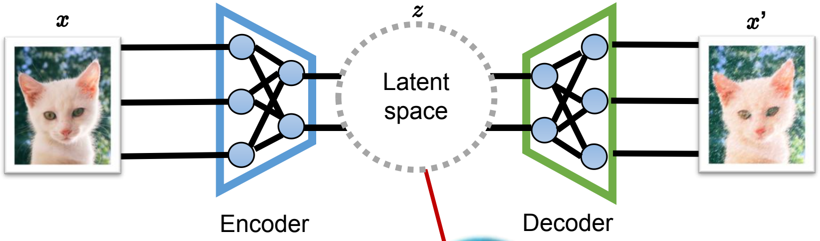


Types of AutoEncoders

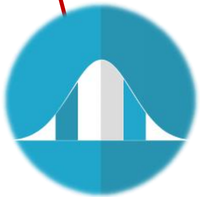
- Deep (aka Stacked) AutoEncoders:
  - Multiple hidden layers for encoder and decoder
  - Note: layers could be fully-connected, convolutional, recurrent, etc.
- Sparse AutoEncoders:
  - Use a large bottleneck layer, but with a sparsity constraint (e.g., enforced through regularization)
- Denoising AutoEncoders:
  - Add noise (typically Gaussian) to the input (or use dropout) to force the network to learn “robust” features and how to remove noise in the output
- Variational AutoEncoders:
  - Probabilistic AutoEncoder, which makes it a generative model
  - Idea: a data point is encoded as a mean μ and standard deviation σ
    - Then, we sample from a Gaussian with mean μ and standard deviation σ
    - Training loss: reconstruction loss (as before) + KL-divergence of latent space distribution and isotropic gaussian
- Many others...

Variational AutoEncoders

- Seminal Paper
  - Kingma and Welling. *"Auto-Encoding Variational*
  - *Bayes."* stat, 10. 2014



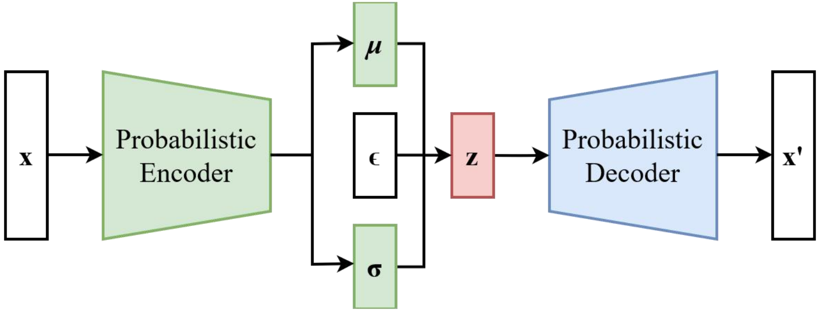
- Probabilistic encoder/decoder
  - Encoder maps an input x to a ***distribution*** in the latent space
    - Posterior $p(z|x)$
    - Approximate posterior $q(z|x)$
  - Prior $p(z)$ over the latent space
    - Usually we choose Gaussian $\mathcal{N}(\mu, \sigma^2)$
  - Likelihood $p(x|z)$



- This is a generative model
  - Q: How do we sample?

Training Variational AutoEncoders

- Seminal Paper
  - Kingma and Welling. *"Auto-Encoding Variational*
  - *Bayes."* stat, 10. 2014



source: wikipedia

- How to train the model?
  - Loss function: reconstruction loss (e.g., MSE or cross entropy) + Kullback-Leibler divergence between $p(z|x)$ and $q(z|x)$
    - Evidence Lower Bound (ELBO)
  - How can we do backpropagation? The latent representation is **random**!?
    - Reparameterization trick: $z = \mu(x) + \epsilon\sigma(x)$
      - Here $\epsilon \sim \mathcal{N}(0, \boldsymbol{I})$ is an ***external input***
      - Sometimes called *"stochastic backpropagation"*

Generative Models

- (Some) generative models allow us to:
  - Sampling — $\boldsymbol{x} \sim p(\boldsymbol{x})$
  - We want to able to **sample new instances** from the **learned** distribution
  - Density estimation — $p(\boldsymbol{x})$
    - We want to **estimate** $p(\boldsymbol{x})$ or compare $p(\boldsymbol{x}_1)$ and $p(\boldsymbol{x}_2)$
  - Learn representations — $z = \mathrm{repr}(\boldsymbol{x})$
    - The representation can be used in **downstream tasks** (e.g., classification or regression)
    - And (in many cases) **reduce dimensionality** (e.g., use an AutoEncoder instead of PCA)

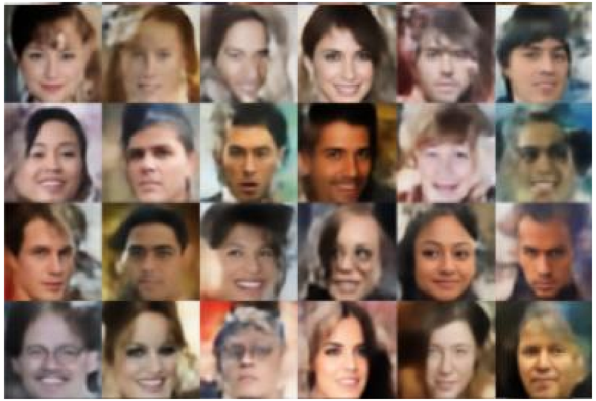
How Good Are AutoEncoders?

- Variational Auto-Encoder (VAE)

Input

Reconstruction

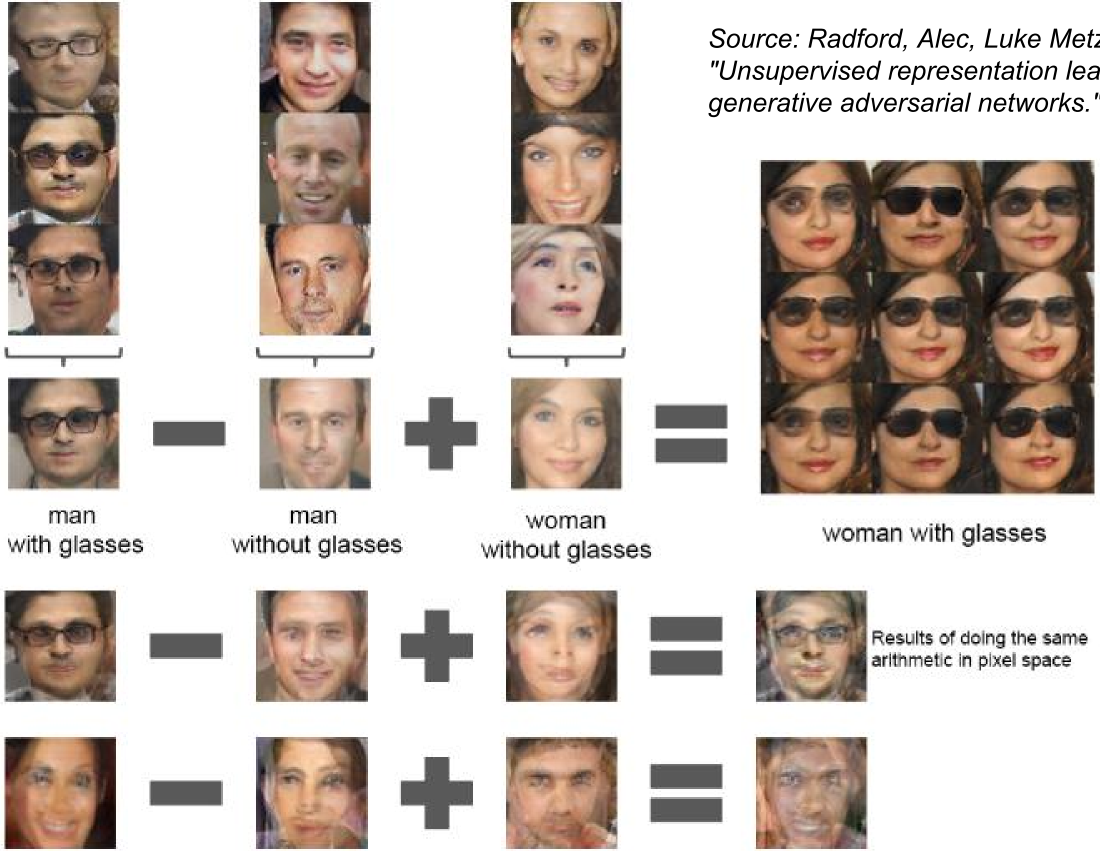
Samples



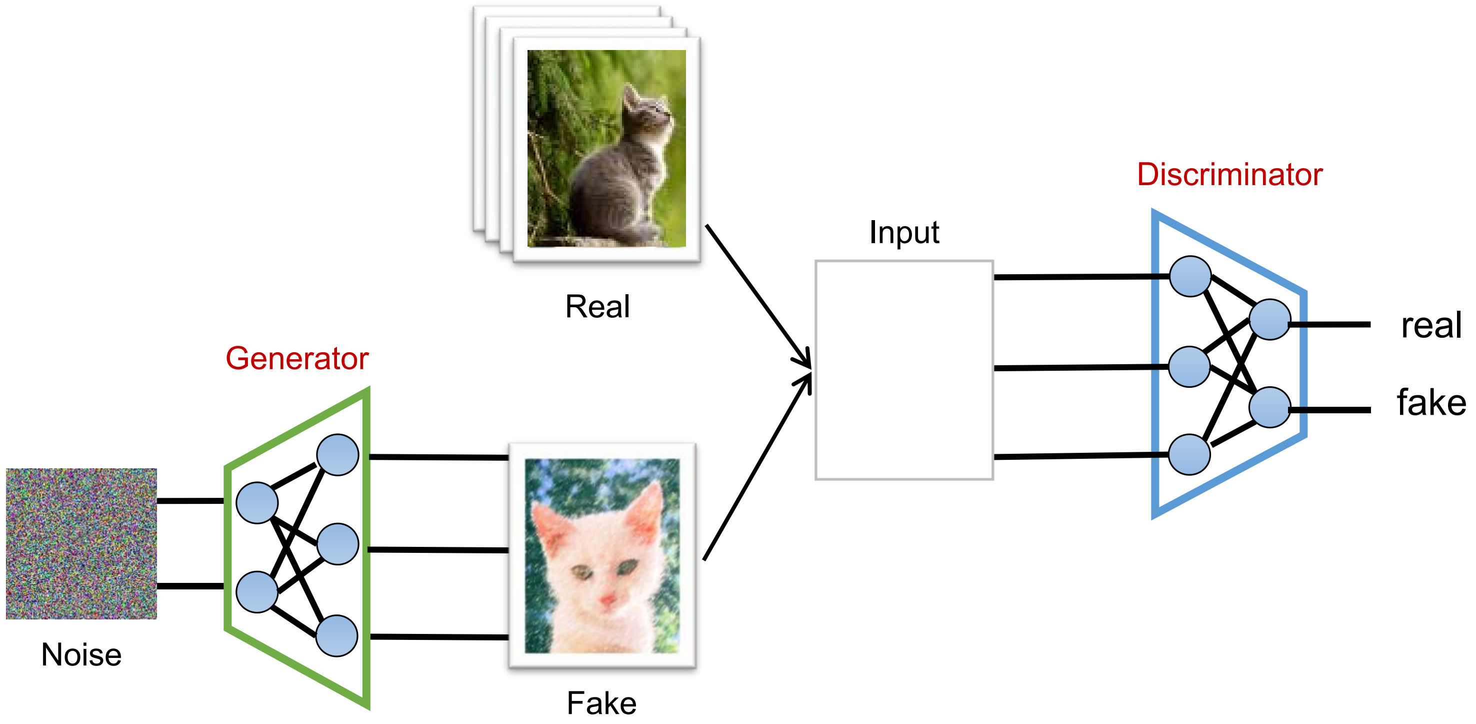
Source: Tolstikhin, Ilya, Olivier Bousquet, Sylvain Gelly, and Bernhard Schoelkopf. "Wasserstein auto-encoders." 2017.

Manipulating Latent Features

Source: Radford, Alec, Luke Metz, and Soumith Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks." arXiv, 2015.



Generative Adversarial Networks (GANs)

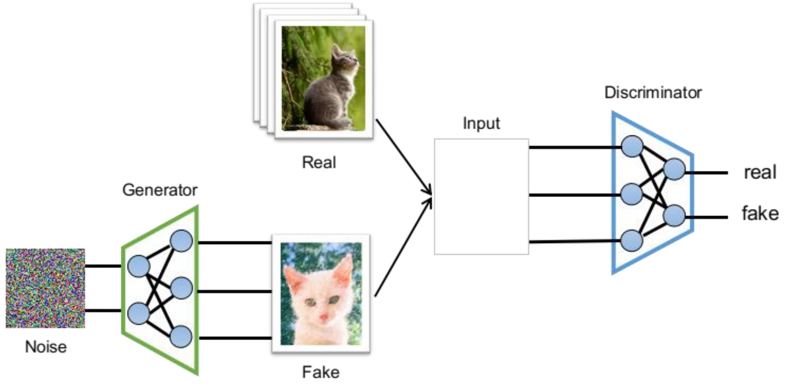


Generative Adversarial Networks

- Origins:
  - Goodfellow et al. *"Generative Adversarial Nets."* in NeurIPS, 2014.

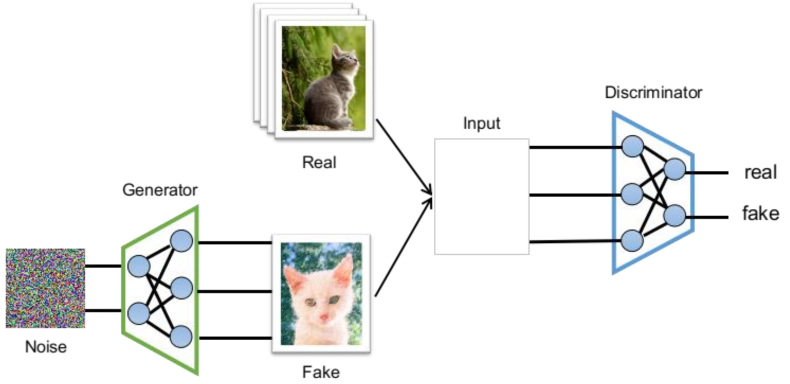
- Generator:
  - Takes random noise from some distribution (e.g., gaussian) and produces a data point
  - Trained using “feedback” from the discriminator

- Discriminator:
  - Given a data point predict real (1) or fake (0)
    - **Real**: data points taken from the dataset
    - **Fake**: data points produced by the generator



Training GANs

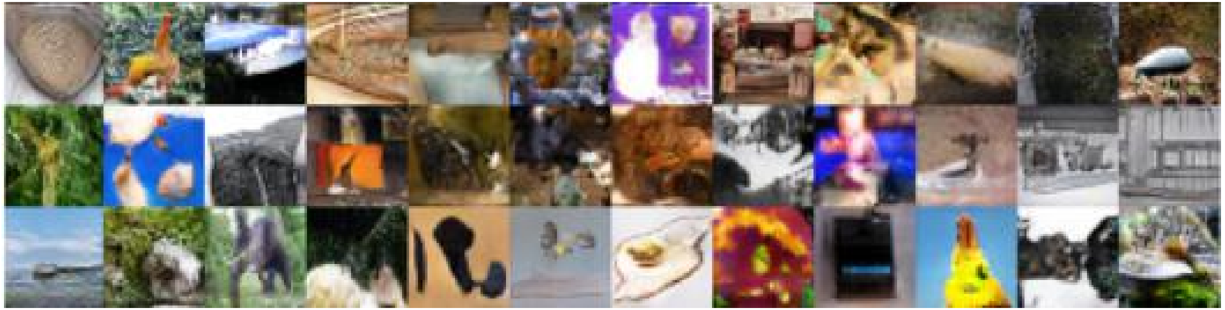
- Challenges:
  - GANs are notoriously difficult to train
  - Generator and discriminator need to learn together at roughly the same pace
    - Otherwise, the training process will fail
- (Informal) training loop (for each epoch):
  - Discriminator:
    - Take k real data points (label 1)
    - Run the generator to produce k fake data points (label 0)
    - Train the discriminator on those $2k$ data points
  - Generator:
    - Freeze the weights of the discriminator (why?)
    - Run the generator to produce k fake data points
    - Give them to the discriminator ***pretending they are real***
    - Backpropagate and update the weights!



How Good Are These Models?

- DCGAN

Source: Radford, Alec, Luke Metz, and Soumith Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks." arXiv, 2015.



GANs are Improving

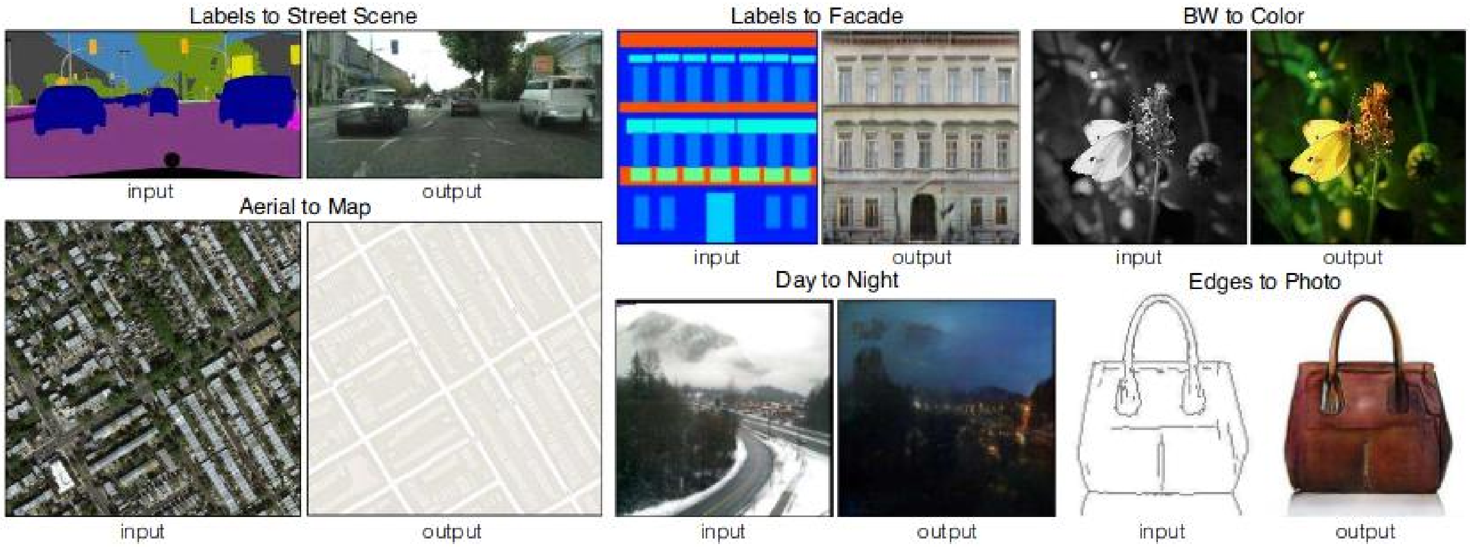
Source: Karras, Tero, Timo Aila, Samuli Laine, and Jaakko Lehtinen. "Progressive growing of gans for improved quality, stability, and variation." ICLR, 2018.

Try: https://www.thispersondoesnotexist.com/

POTTEDPLANT HORSE SOFA BUS CHURCHOUTDOOR BICYCLE TVMONITOR

Other Applications of Generative Models

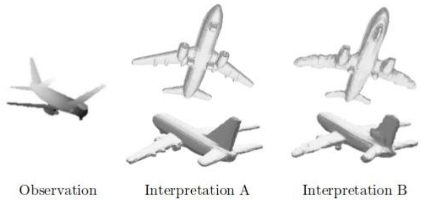
- Image-to-Image translation



Source: Isola, Phillip, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. "Image-to-image translation with conditional adversarial networks." CVPR, 2017.

Other Applications of Generative Models

- Speech/Audio
  - Oord et al. "*Wavenet: A generative model for raw audio*." arXiv, 2016.
- Generating 3D from 2D
  - Wu et al. "Learning shape priors for single-view 3d completion and reconstruction." ECCV, 2018.
- Text-to-image
  - Zhang et al. "Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks." ICCV, 2017.



- And many others...
  - Scene completion
  - Image editing
  - Face aging
  - Super-resolution
  - Video prediction

Next Time

- Wednesday (4/10): Lecture
  - Topic: Diffusion Models
- Upcoming:
  - Homework 5 due 4/12