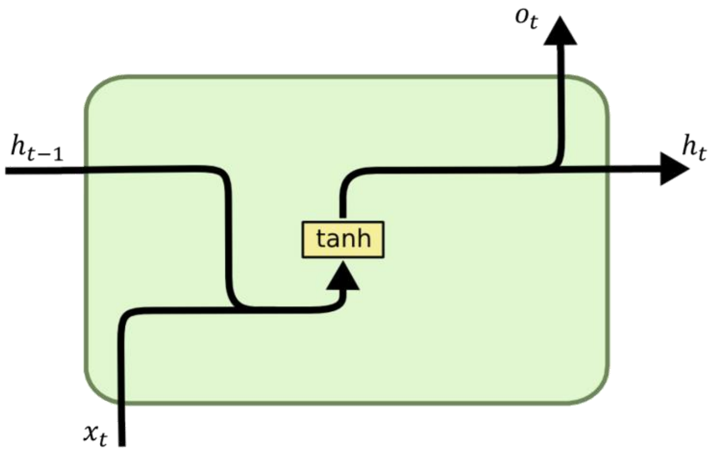
LECTURE 7:

GENERATIVE ADVERSARIAL NETWORKS

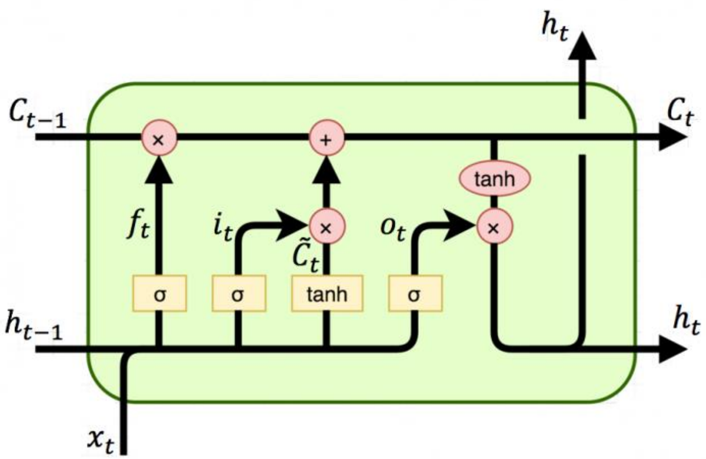
University of Washington, Seattle

Fall 2024

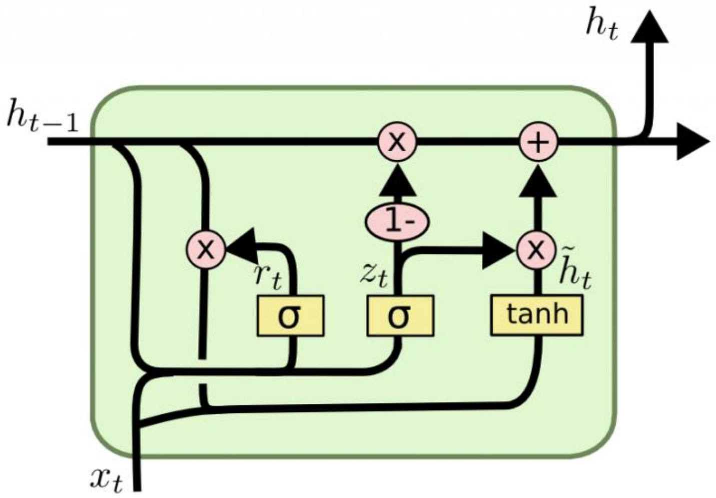
Previously in EEP 596...



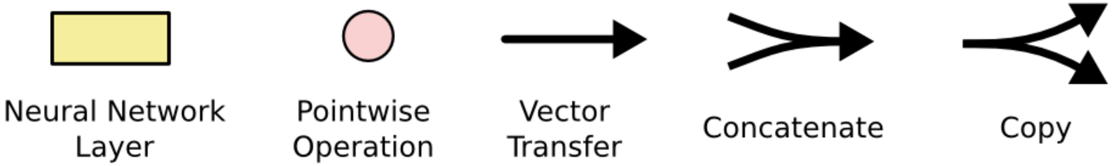
Vanilla RNN



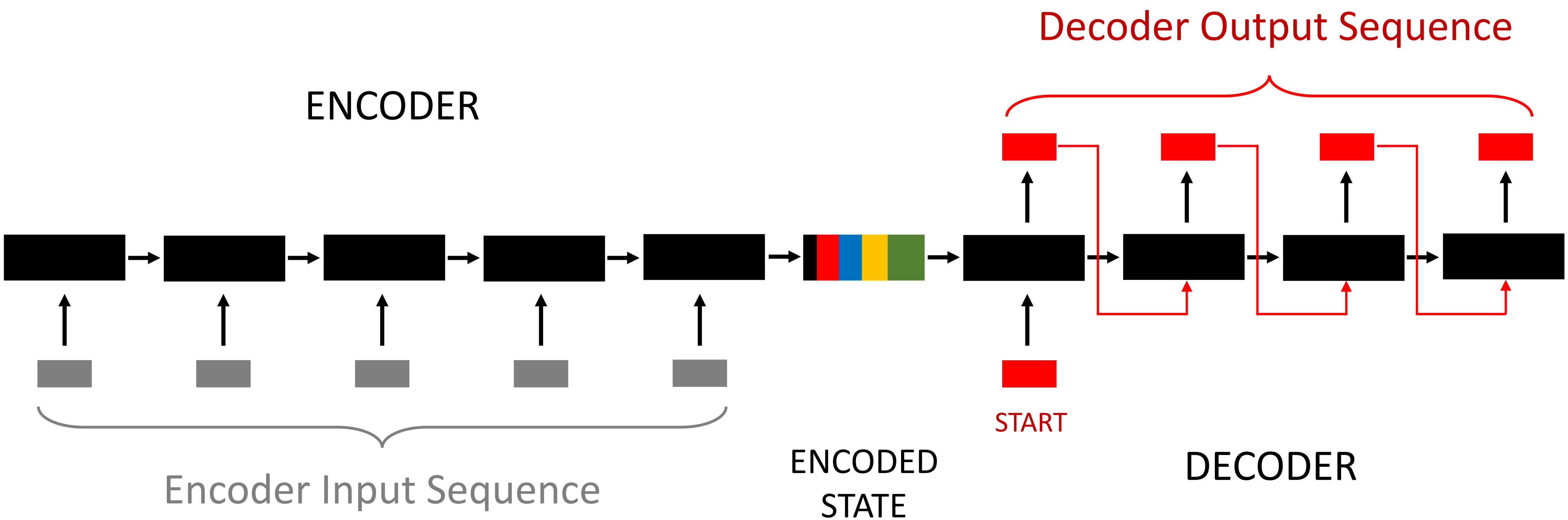
LSTM



GRU



Previously in EEP 596…



OUTLINE

Part 1: Unsupervised Learning

- Supervised vs unsupervised
- Unsupervised learning in with NN

Part 2: Generative Model Taxidermy

- FVBN
- Variational Autoencoder
- GAN

Part 3: Generative Adversarial Networks

- GAN architecture
- Two-player game
- Generator network
- Discriminator network

Part 4: GAN Optimization and Applications

- Competing cost function
- Minmax game optimization
- GAN variations

Unsupervised Learning

Supervised vs Unsupervised

Unsupervised Learning in NN

Supervised vs Unsupervised Learning

Supervised

Data:
{x} x: inputs **WITH labels**

Neural Network Goal:
Minimize specific **error**

Examples: Classification,
Regression, Detection, Prediction

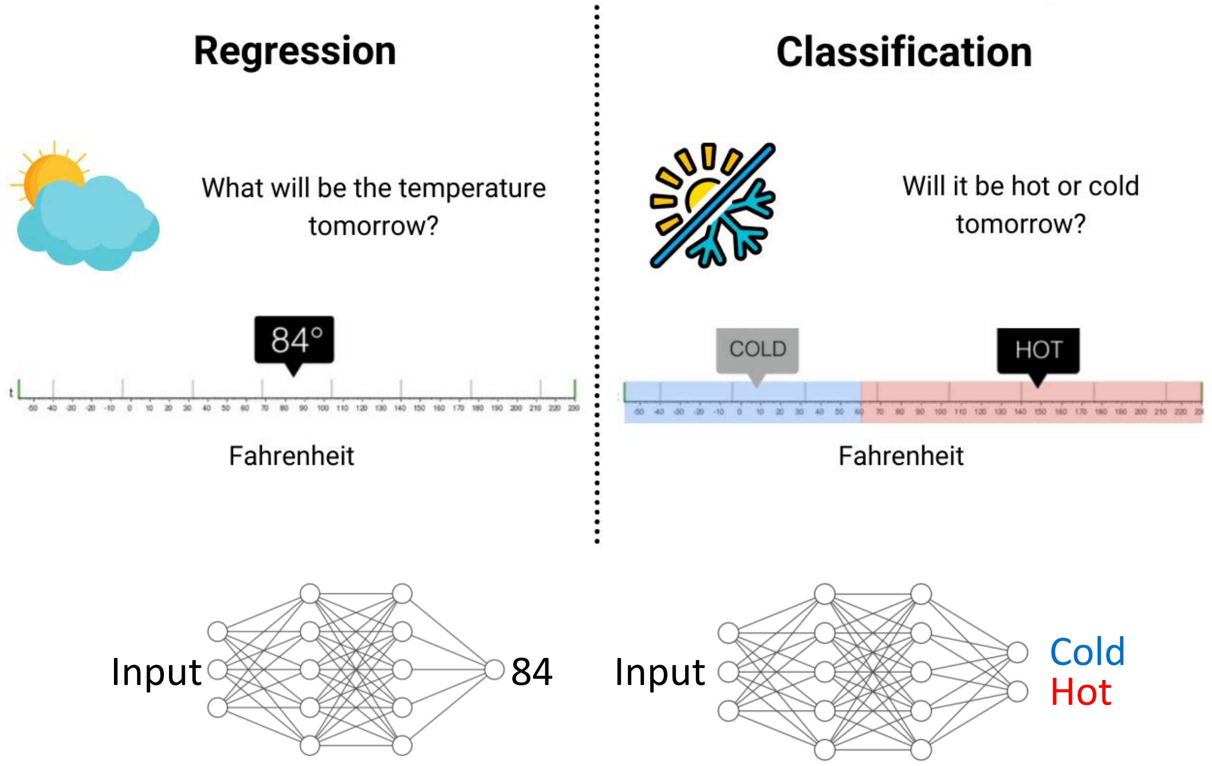
Supervised vs Unsupervised Learning

Supervised

Data:
{x} x: inputs **WITH labels**

Neural Network Goal:
Minimize specific **error**

Examples: Classification, Regression, Detection, Prediction



Unsupervised Learning in NN

Unsupervised

Data:

{x} x: inputs **WITHOUT labels**

Neural Network Goal:

Learn a **structure** of the data

Unsupervised Learning in NN

Training Data

Training Data ~ $\mathbf{P}_{\mathbf{data}}(\mathbf{x})$

Unsupervised Learning in NN

Training Data

Generated Samples

http://www.whichfaceisreal.com/

Training Data ~ $\mathbf{P_{data}(x)}$

Generate Samples ~ $\mathbf{P_{model}(x)}$

Unsupervised Learning in NN

Training Data

Generated Samples

http://www.whichfaceisreal.com/

Training Data ~ $\mathbf{P_{data}(x)}$

Generate Samples ~ $\mathbf{P_{model}(x)}$

Goal: Model estimated density ≈ Real world density

Core problem in unsupervised learning

Unsupervised Learning in NN

Unsupervised

Data:

{x} x: inputs **WITHOUT labels**

Neural Network Goal:

Learn a **structure** of the data

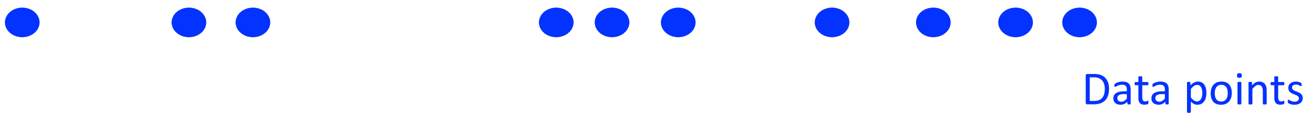
+ No need for labeling → More data

- Challenge: Cost?

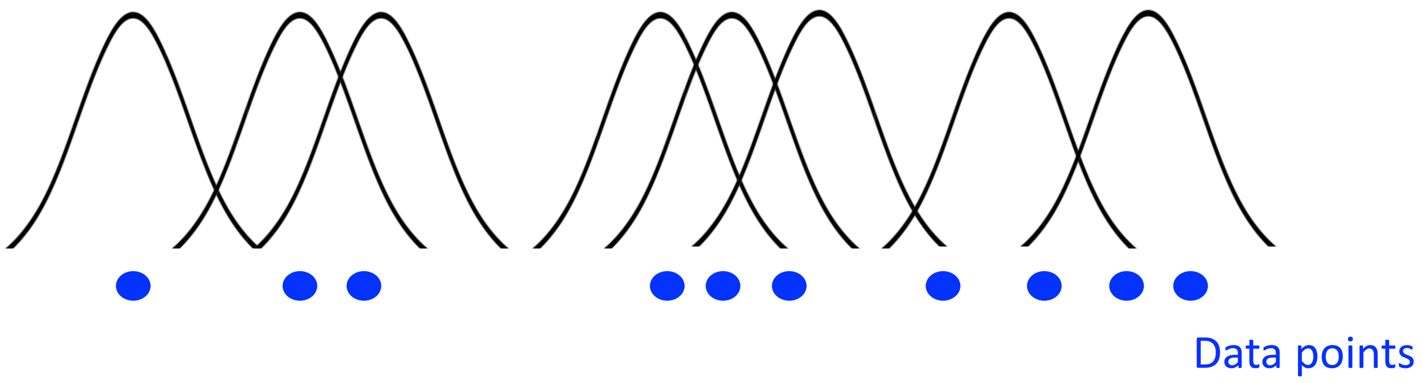
+ Has the potential to learn the real world

- Challenge: Optimization?

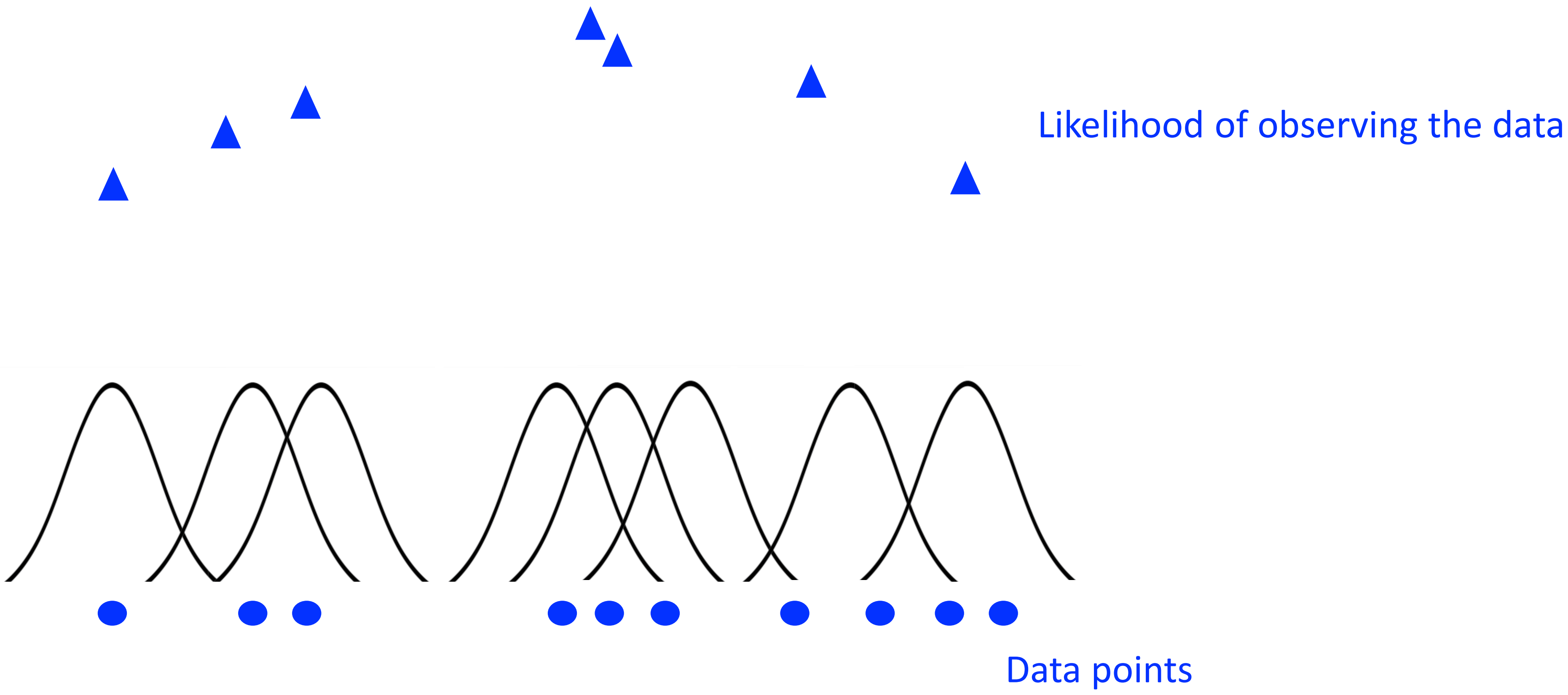
Maximum Likelihood Estimation



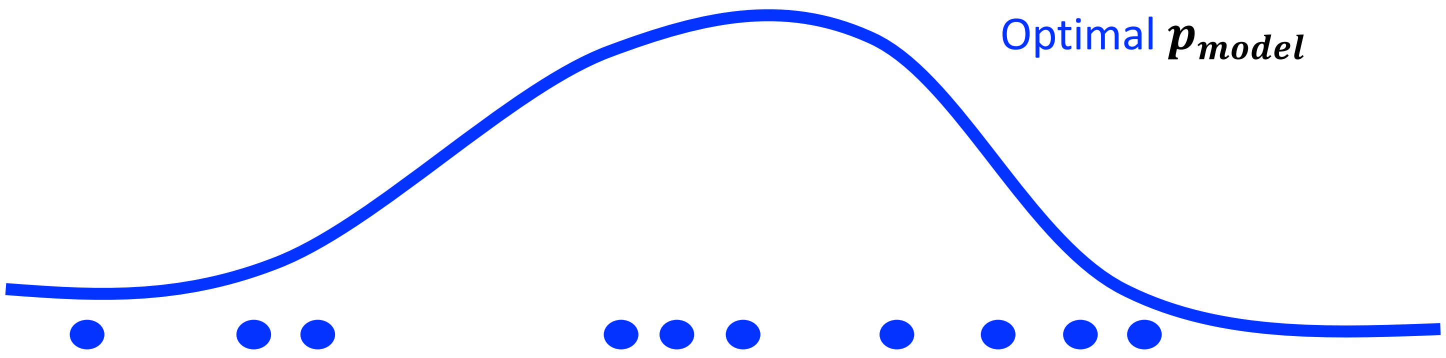
Maximum Likelihood Estimation



Maximum Likelihood Estimation



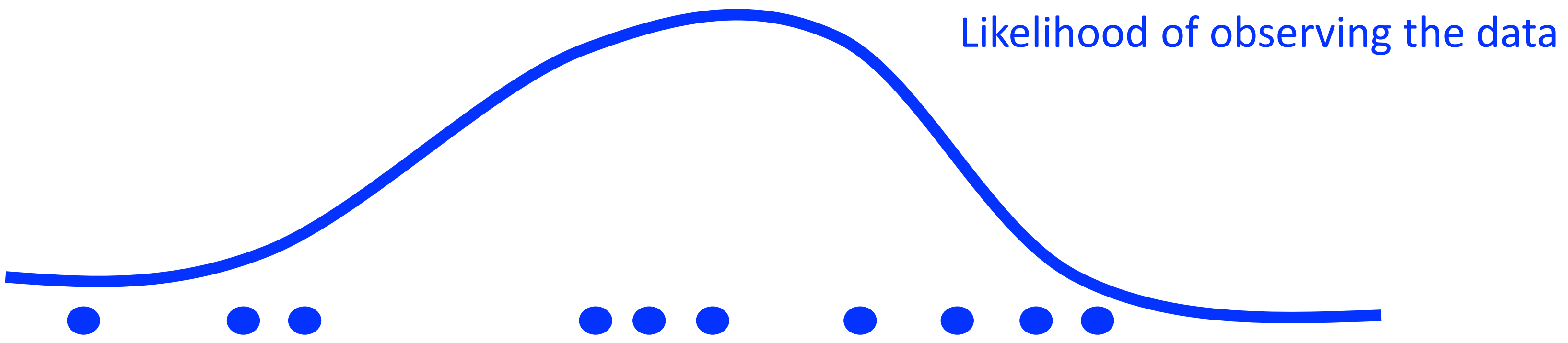
Unsupervised Learning in NN



Model params

$$\theta^* = argmax_\theta \, E_{x \sim p_{data}} \log p_{model}(x|\theta)$$

Unsupervised Learning in NN

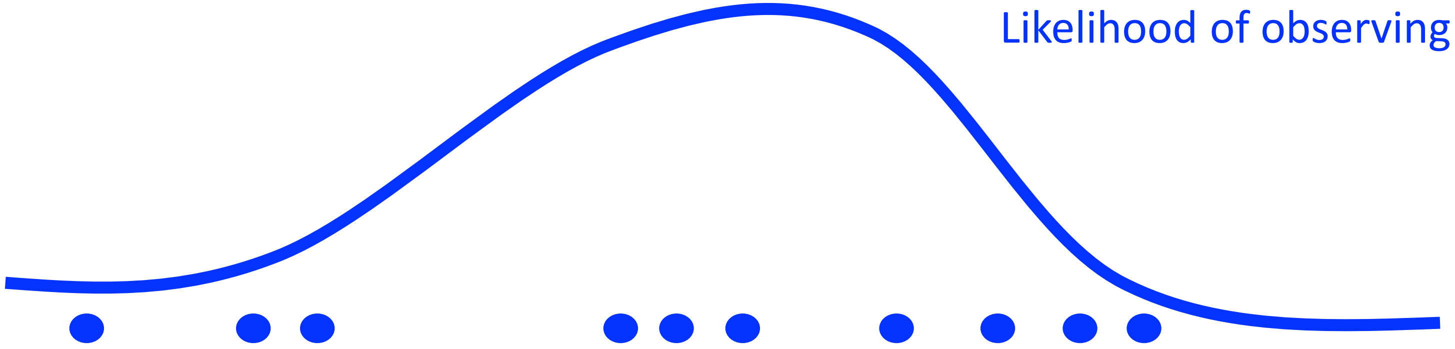


Model params

$$\theta^* = argmax_\theta\, E_{x \sim p_{data}}\, log\, p_{model}(x|\theta)$$

Goal: Find the optimal distribution $p_{model}(x|\theta)$ that best fit the data

Unsupervised Learning in NN



Likelihood of observing the data

Model params

$$\theta^* = \mathrm{argmax}_\theta\, E_{x \sim p_{data}} \log p_{model}(x|\theta)$$

Explicit – explicitly define and generate P_{model}

Implicit - generate P_{model} without defining P_{model} exactly

Generative Model Taxidermy

Supervised vs Unsupervised

Unsupervised Learning in NN

Unsupervised Learning in NN

Ian Goodfellow

Density Estimation

Explicit Density

Implicit Density

Direct

GAN

Tractable Density

Fully Visible BN

Approximate Density

Markov Chain

Variational

Markov Chain

Variational Autoencoder

Fully Visible BN

- Explicitly formula based on chain rule:

$$\boldsymbol{p_{model}}(\boldsymbol{x}) = \boldsymbol{p_{model}}(\boldsymbol{x_1})\Pi_{i=2}^{n}\boldsymbol{p_{model}}\,(\boldsymbol{x_i}\,|\boldsymbol{x_1}, \boldsymbol{x_2}, \ldots, \boldsymbol{x_{i-1}})$$

- O(n) generation cost

- No control through hidden variables

Language Model

Language model: probability distribution over sequences of words. Given such a sequence, say of length m, it assigns a probability to the whole sequence.

Language Model

Language model: probability distribution over sequences of words. Given such a sequence, say of length m, it assigns a probability to the whole sequence.

P(W=NASA will **take** me to Moon) = p1
P(W= NASA will **bake** me to Moon) = p2
p2 << p1

Language Model

Language model: probability distribution over sequences of words. Given such a sequence, say of length m, it assigns a probability to the whole sequence.

P(W=NASA will **take** me to Moon) = p1
P(W= NASA will **bake** me to Moon) = p2
p2 << p1

Chain rule is used to estimate probability:

$$P(w_1 w_2 \ldots w_n) = \prod_i P(w_i | w_1 w_2 \ldots w_{i-1})$$

P(W) = P(**NASA**) P(**will**| NASA) P(**take**| NASA will) P(**me**| NASA will take) P(**to**| NASA will take me) P(**Moon** | NASA will take me to)

Variational Autoencoder

$$p_{model}(x) = \prod_{i=2}^{n} p_{model}\left(x_i \mid x_1, x_2, \dots, x_{i-1}\right)$$

$$\downarrow$$

$$p_{model}(x) = \int p_{model}(z) p_{model}(x|z)\, dz$$

Variational Autoencoder

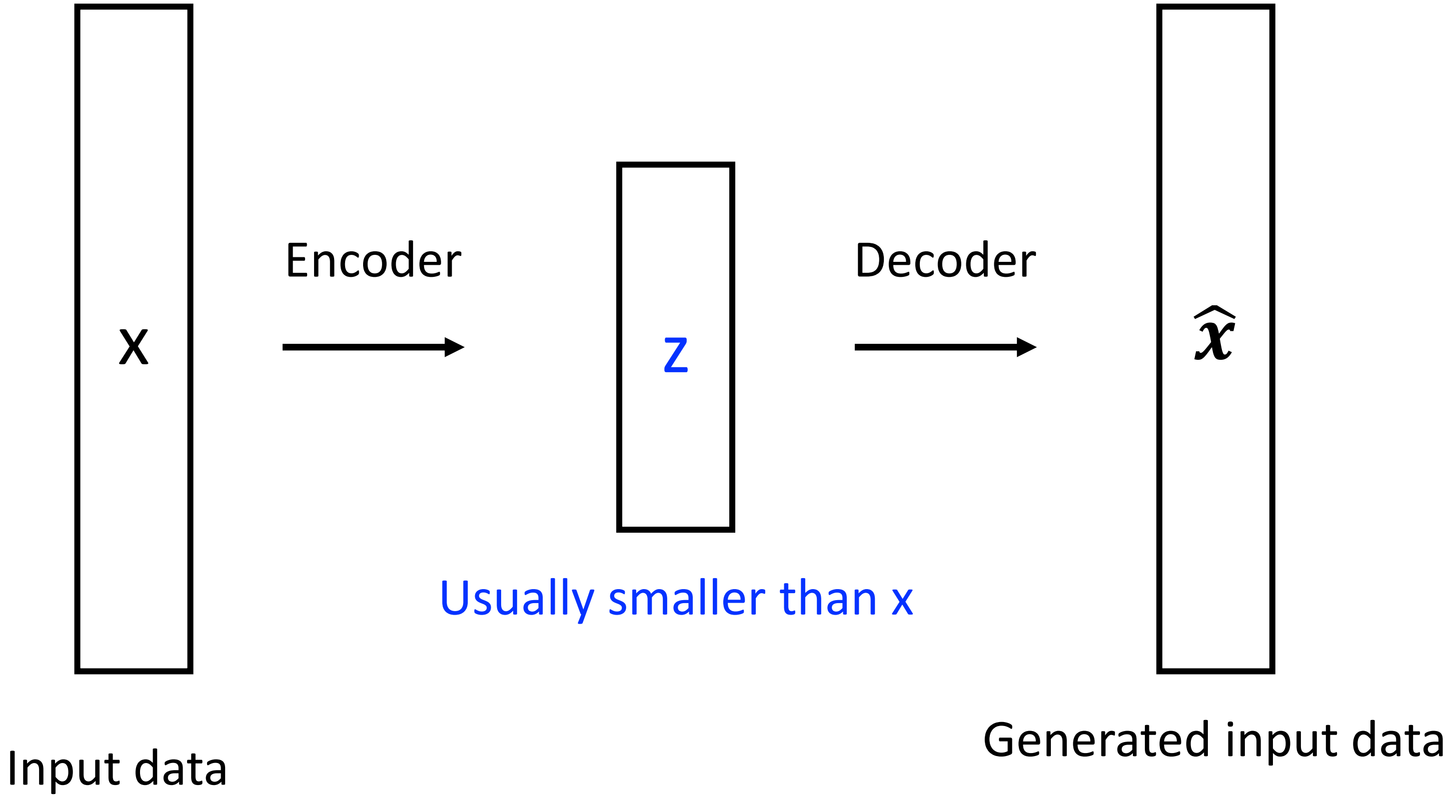
$$p_{model}(x) = \prod_{i=2}^{n} p_{model}\left(x_i \mid x_1, x_2, \ldots, x_{i-1}\right)$$

$$\downarrow$$

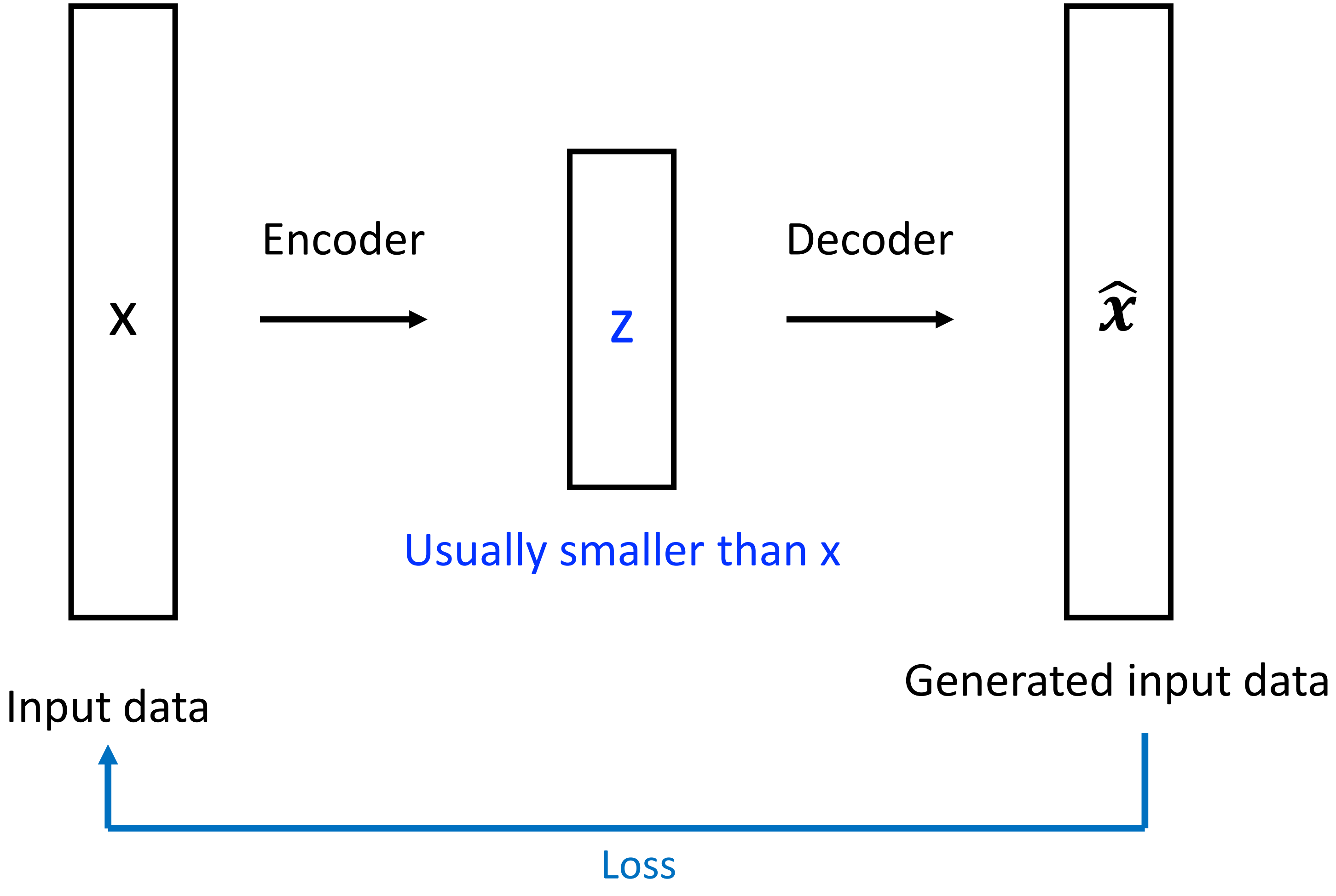
$$p_{model}(x) = \int p_{model}(z) p_{model}(x|z)\, dz$$

$p_{model}(x)$ is controlled by hidden state z

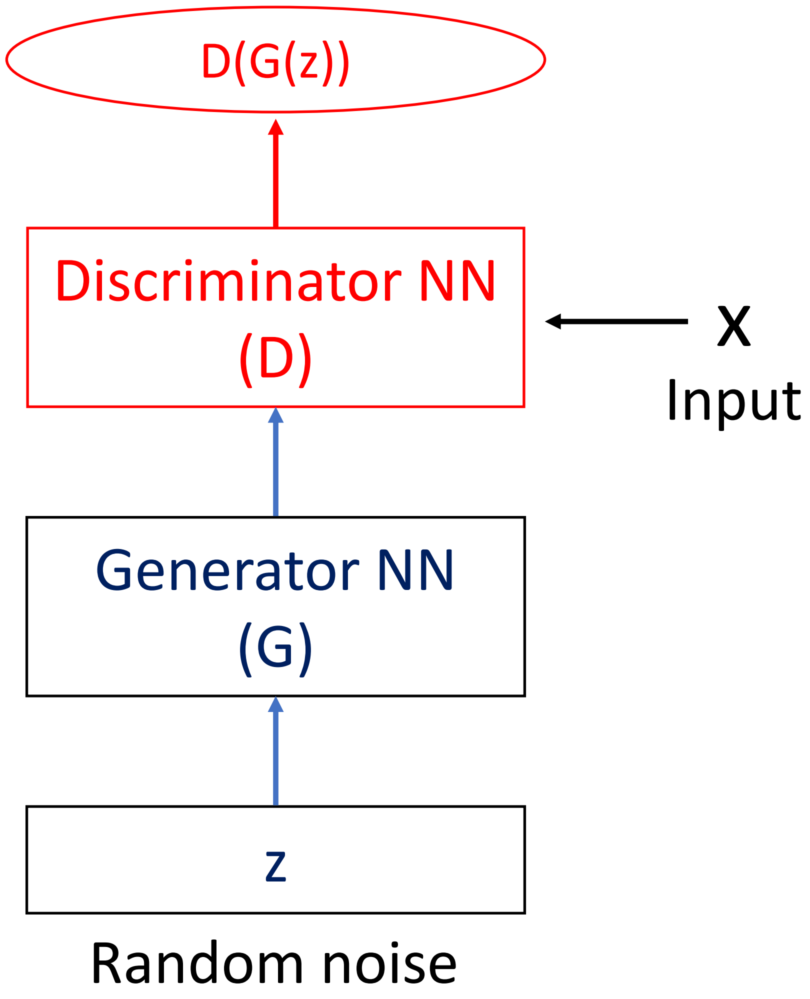
Variational Autoencoder



Variational Autoencoder



GAN



- Instead of sampling from **high dimensional, complex and unknown distribution**
- Sample from **simple distribution**, e.g. normal distribution (random noise) and **find transformation** to the distribution we want to learn.
- **Learn the transformation using a NN**

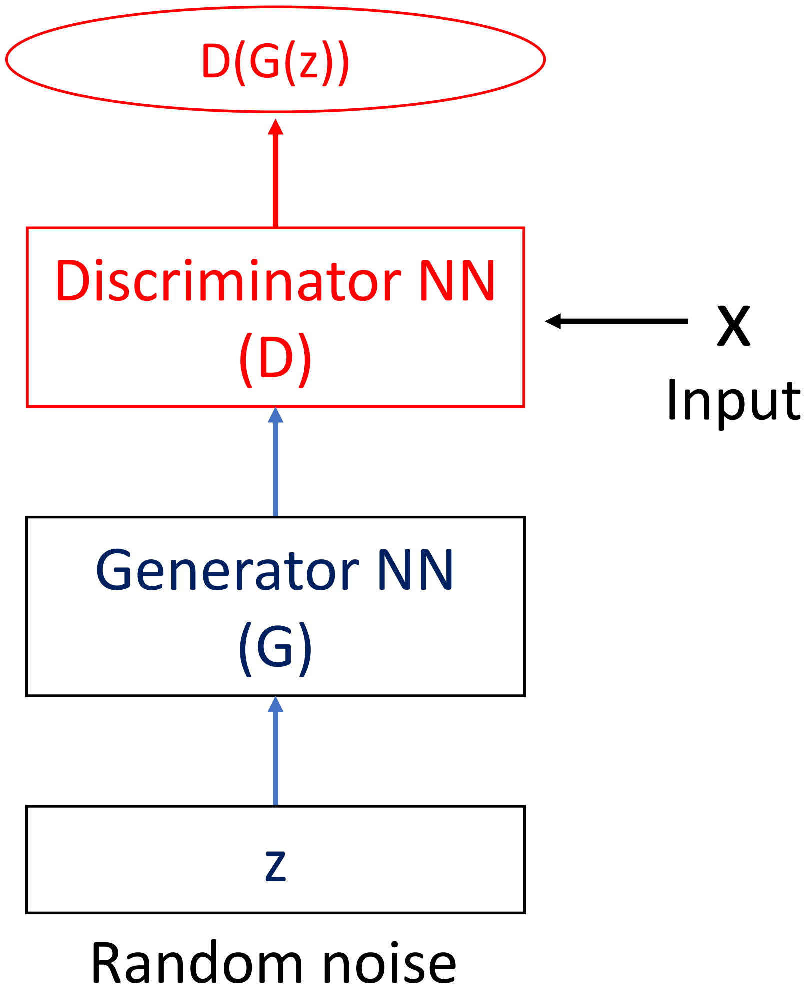
not learning classification but transformation

Generative Adversarial Networks

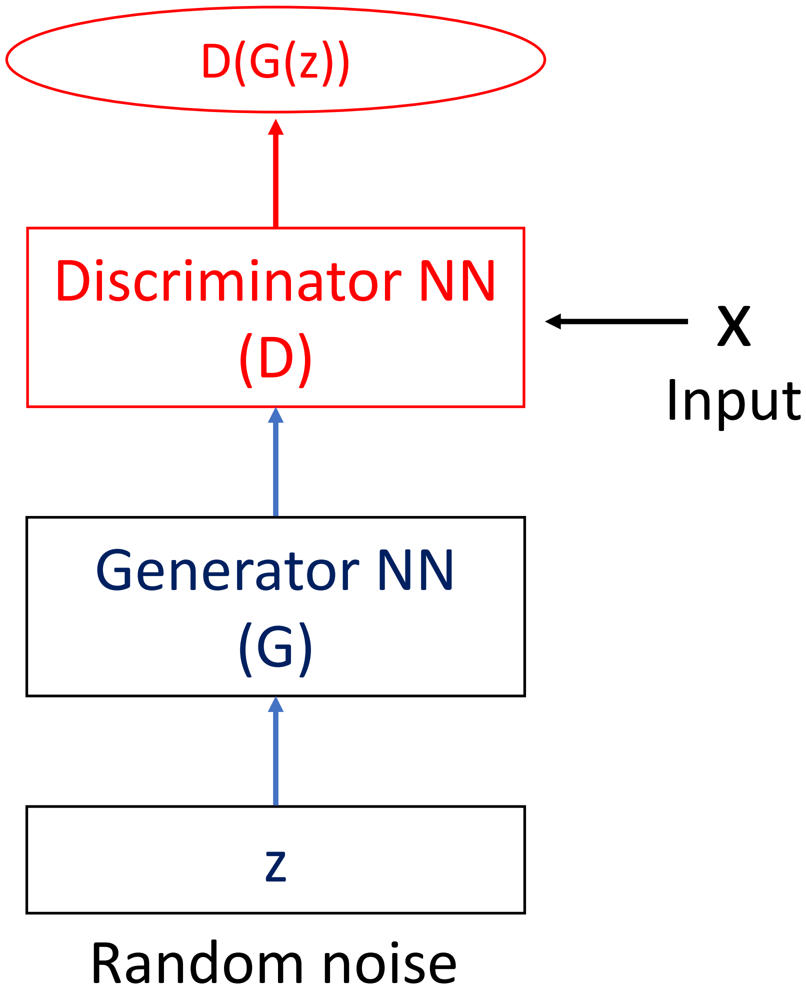
Supervised vs Unsupervised

Unsupervised Learning in NN

GAN



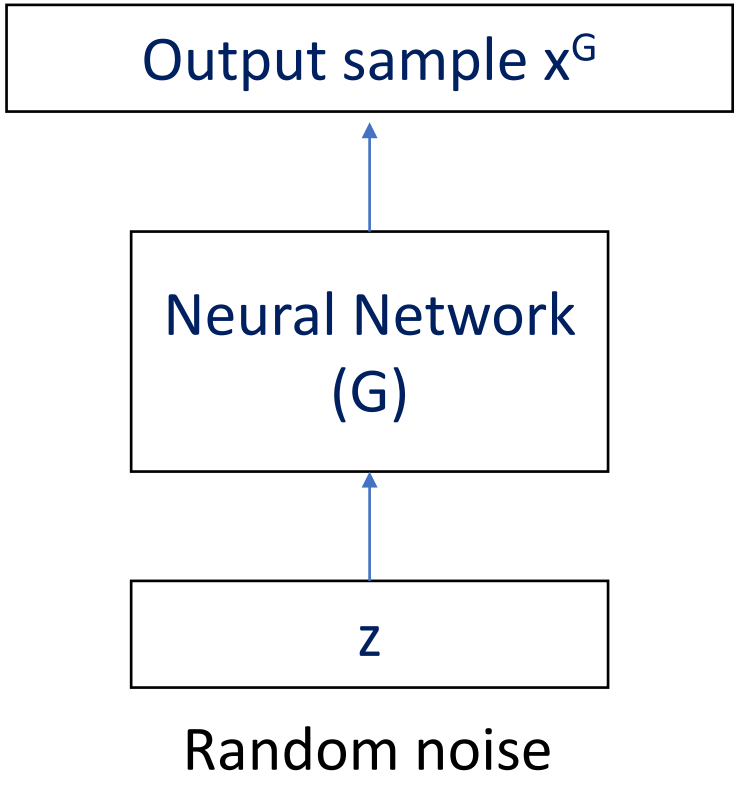
GAN



Discriminator – try to distinguish between **x (real) and generated (fake)** images

Generator – try to generate **samples and present them as real world** and fool the discriminator

Generator (G)



Training data has distribution $\mathbf{p}_{\mathbf{data.}}$
Sample $\mathbf{x} \sim \mathbf{p}_{\mathbf{data.}}$

Goal: Output sample $\mathbf{x}^{\mathbf{G}}$ is of similar dimensions as $\mathbf{x}$ and distribution $\mathbf{p}_{\mathbf{data.}}$

Examples

Face:

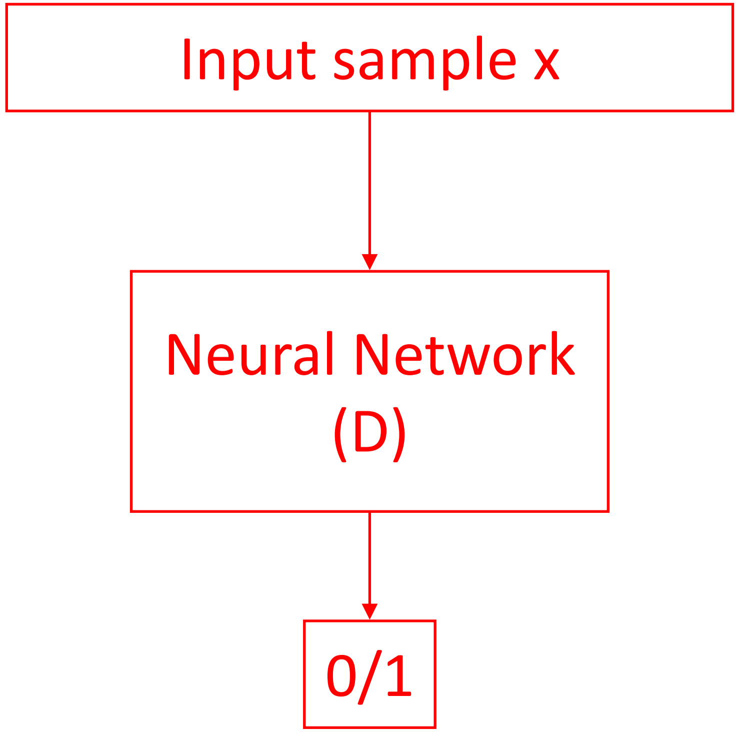
Car:

Bedroom:

Discriminator (D)

Receives input of same dimensions as $\mathbf{p}_{data}$.

Goal: Distinguish sample from $\mathbf{p}_{data}$ (1) or not (0).



Examples

Discriminator

Face (gen):

0

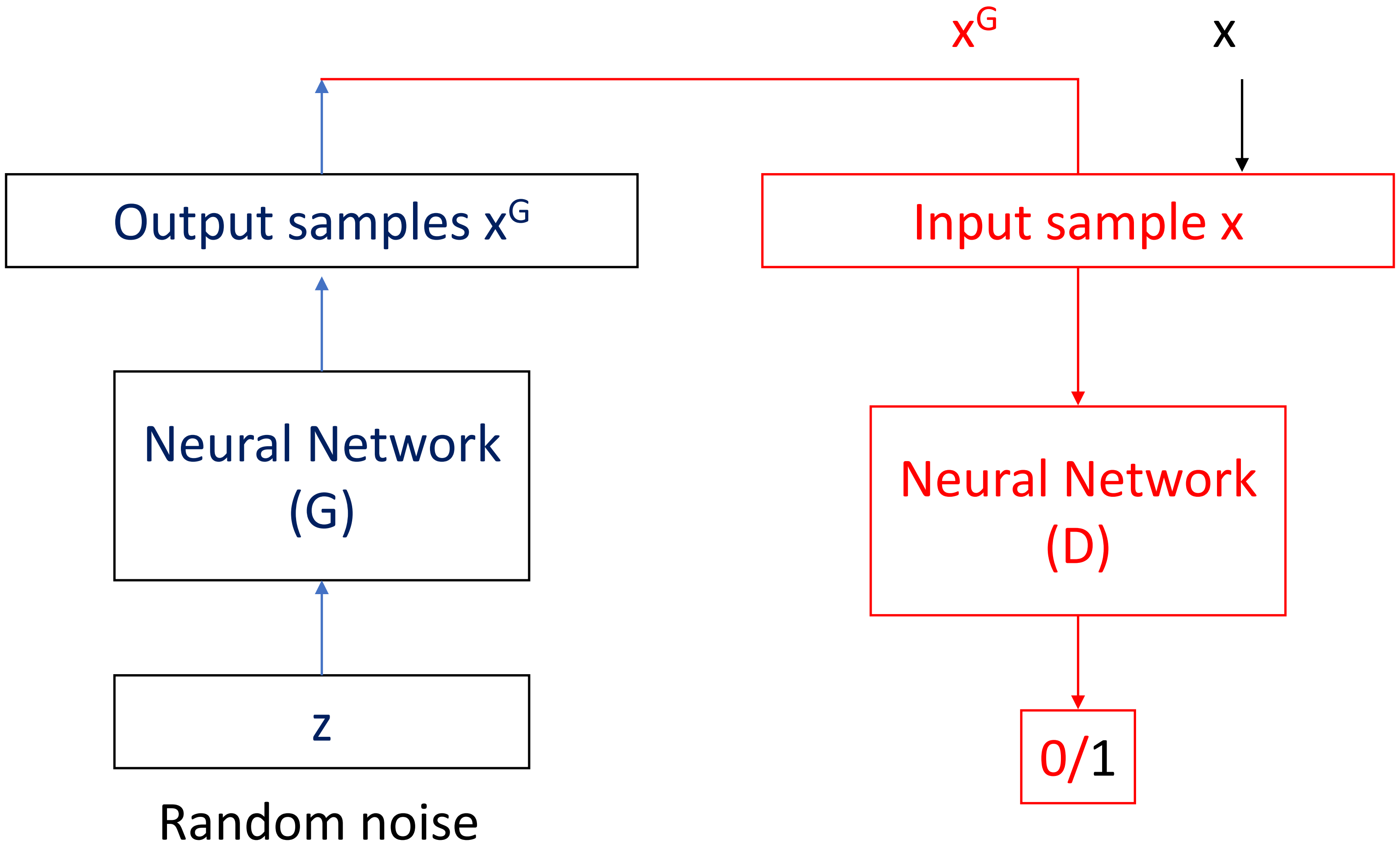
Car (real):

1

Bedroom (gen):

0

Full Architecture



GAN Optimization and Applications

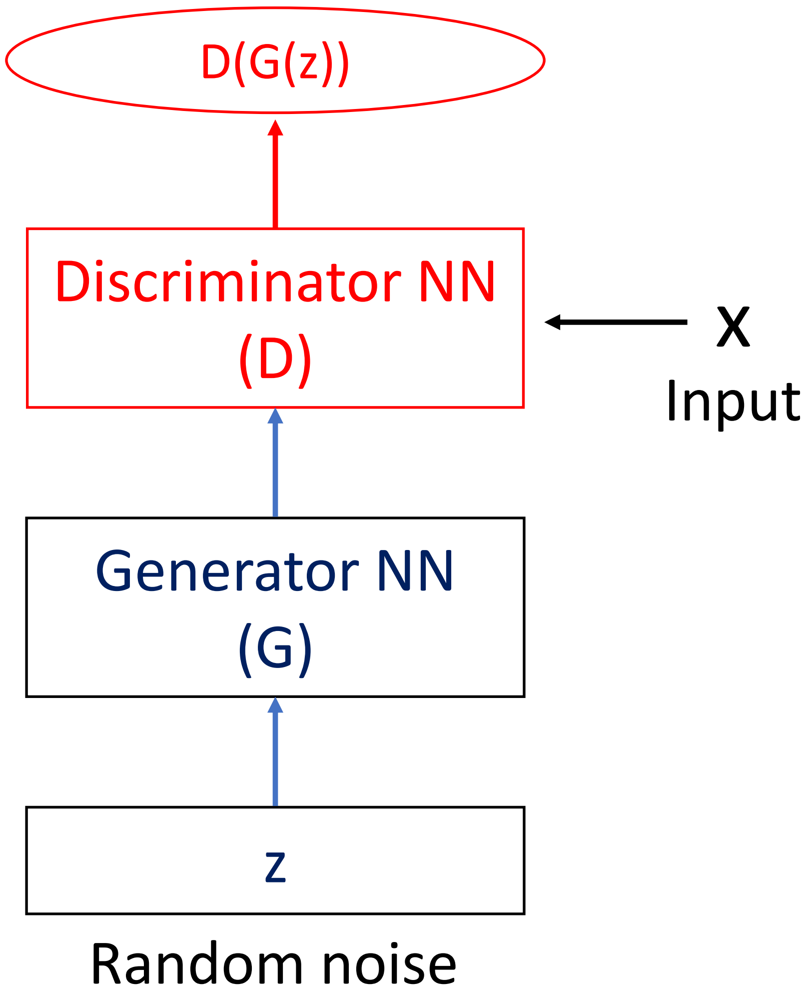
Supervised vs Unsupervised

Unsupervised Learning in NN

Competing cost functions

"Cost": $\|D(G(z))\|$

"Cost": $\|D(G(z)) - 1\|$



Competing cost functions

Binary Cross Entropy Loss

$$J^{(D)} = -\frac{1}{2}\mathbb{E}_{x\sim p_{data}} \log D_{\theta_d}(x) - \frac{1}{2}\mathbb{E}_{z\sim p_{model}} \log\left(1 - D_{\theta_d}(G_{\theta_g}(z))\right)$$

$$J^{(G)} = -J^{(D)}$$

$$J^{(D)} = -\frac{1}{2}\int p_{data}(x)\log D(x)dx - \frac{1}{2}\int p_{model}(x)\log(1 - D(x))dx$$

Competing cost functions

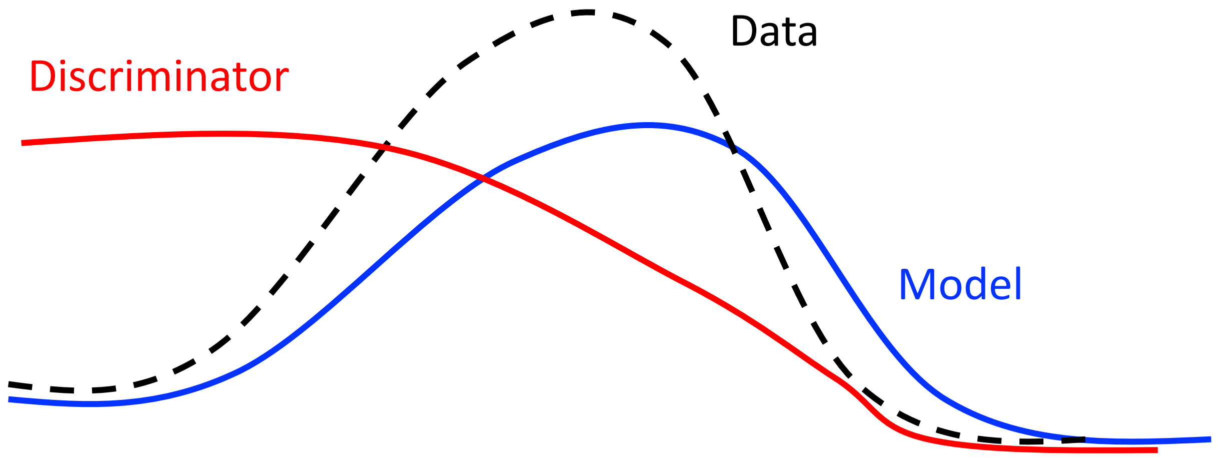
Optimal $D(x)$ is

$$D(x) = \frac{p_{data}}{p_{model} + p_{data}}$$

Assumption: p_{model}, p_{data} are nonzero everywhere

Equilibrium: $p_{model} = p_{data}$ then $\mathrm{E}(D(x)) = \frac{1}{2}$

Competing cost functions



Discriminator learns an approximation of $p_{data}(x)/p_{model}(x)$

vs

learning p_{model} (x) directly (or indirectly via latent variable models).

Minmax Game Optimization

$$\min_{\theta_g} \max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p_{model}} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

Discriminator output for real data

Discriminator output for generated data

Solution:

Saddle point in the parameter space (Nash Equillibrium)

- One player (Discriminator) is at **maximum**,
- Other player (Generator) is at **minimum**

Optimization in NN

- **Gradient ascent** for the discriminator on J

$$J^{(D)} = \frac{1}{2}\mathbb{E}_{x\sim p_{data}} \log D_{\theta_d}(x) + \frac{1}{2}\mathbb{E}_{z\sim p_{model}} \log\left(1 - D_{\theta_d}(G_{\theta_g}(z))\right)$$

$$\theta_d \leftarrow \arg\min_{\theta_d} J^{(D)}$$

- **Gradient descent** for the generator

$$J^{(G)} = -\frac{1}{2}\mathbb{E}_{z\sim p_{model}} \log\left(1 - D_{\theta_d}(G_{\theta_g}(z))\right)$$

$$\theta_g \leftarrow \arg\min_{\theta_g} J^{(G)}$$

Optimization in NN

Take ***k*** gradient steps for the discriminator (k a hyperparameter), each doing the following:

- Sample *m* noise samples, $\{z^{(1)},z^{(2)},\ldots,z^{(m)}\}$ from $p_{model}(z)$.
- Sample *m* actual samples, $\{x^{(1)},x^{(2)},\ldots,x^{(m)}\}$ from $p_{data}(x)$: (a minibatch of your input data.)

- Perform an optimization step on the discriminator:

Optimization in NN

Take ***k*** gradient steps for the discriminator (k a hyperparameter), each doing the following:

- Sample *m* noise samples, $\{z^{(1)},z^{(2)},...,z^{(m)}\}$ from $p_{model}(z)$.
- Sample *m* actual samples, $\{x^{(1)},x^{(2)},...,x^{(m)}\}$ from $p_{data}(x)$: (a minibatch of your input data.)
- Perform an optimization step on the discriminator:

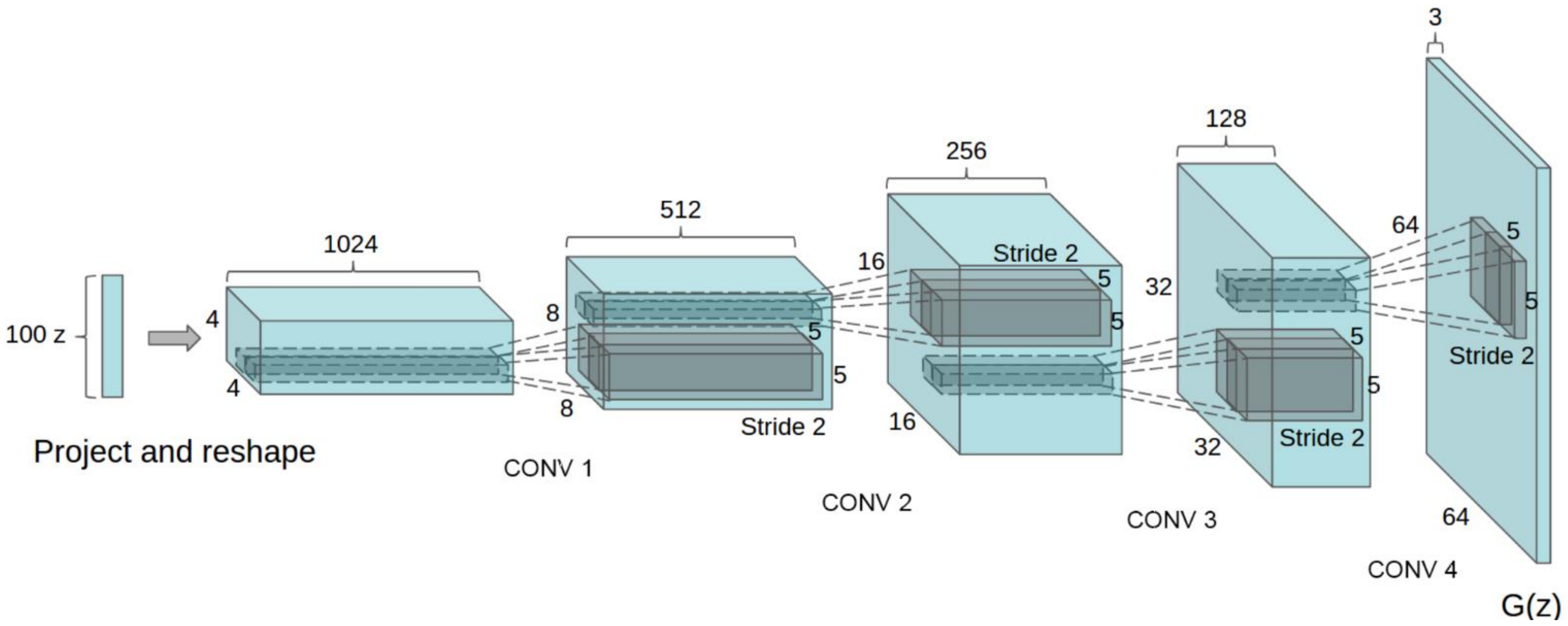
Do gradient descent step for the generator:

- Sample *m* noise samples, $\{z^{(1)},z^{(2)},...,z^{(m)}\}$ from $p_{model}(z)$.
- Perform an optimization step on the **generator**:

GAN Applications: Original GAN

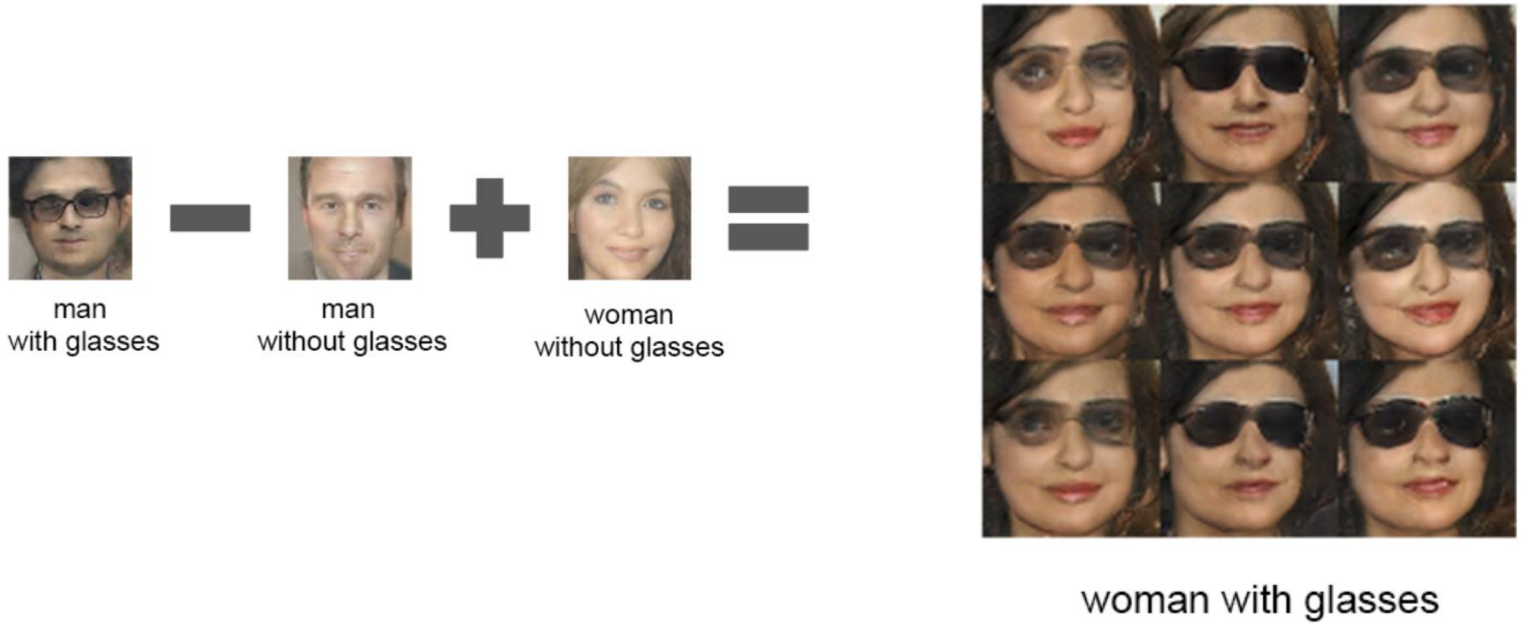
Goodfellow et al. (2014) Generative Adversarial Nets

DCGAN

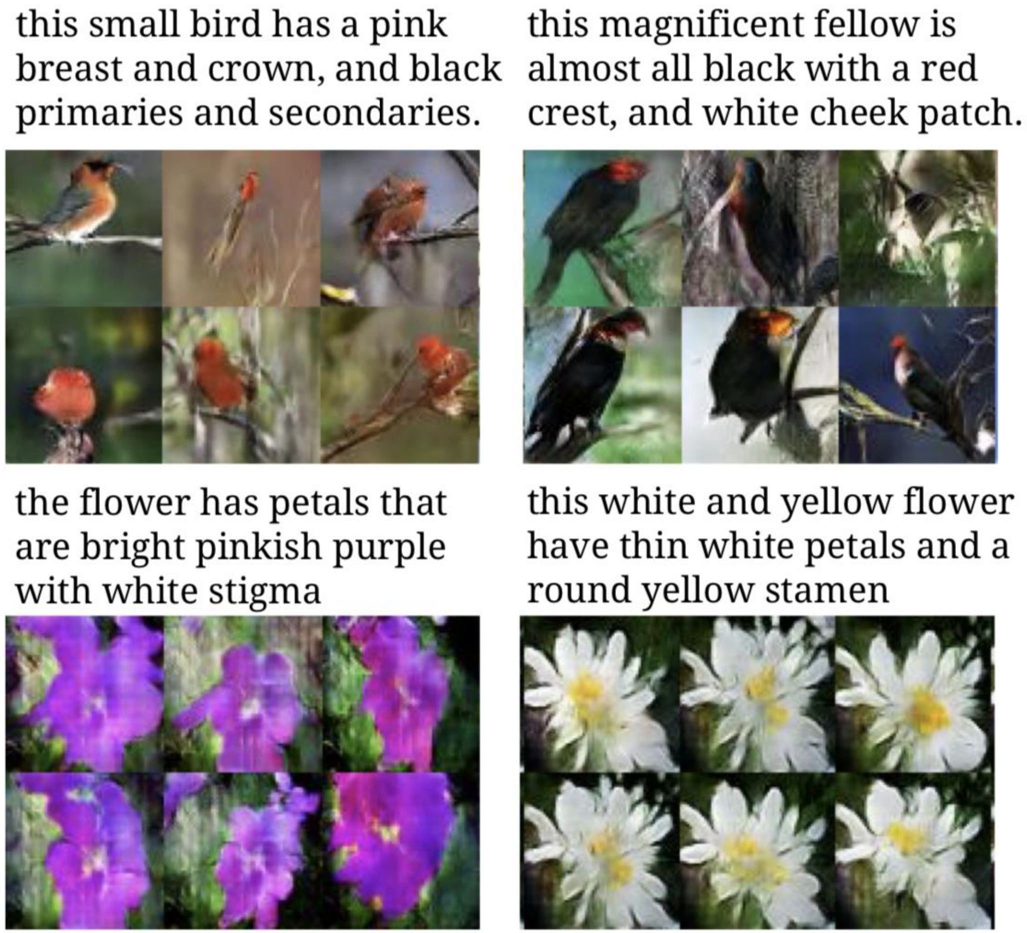


Radford, Alec, Luke Metz, and Soumith Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks." (2015).

Similarities in Hidden Space

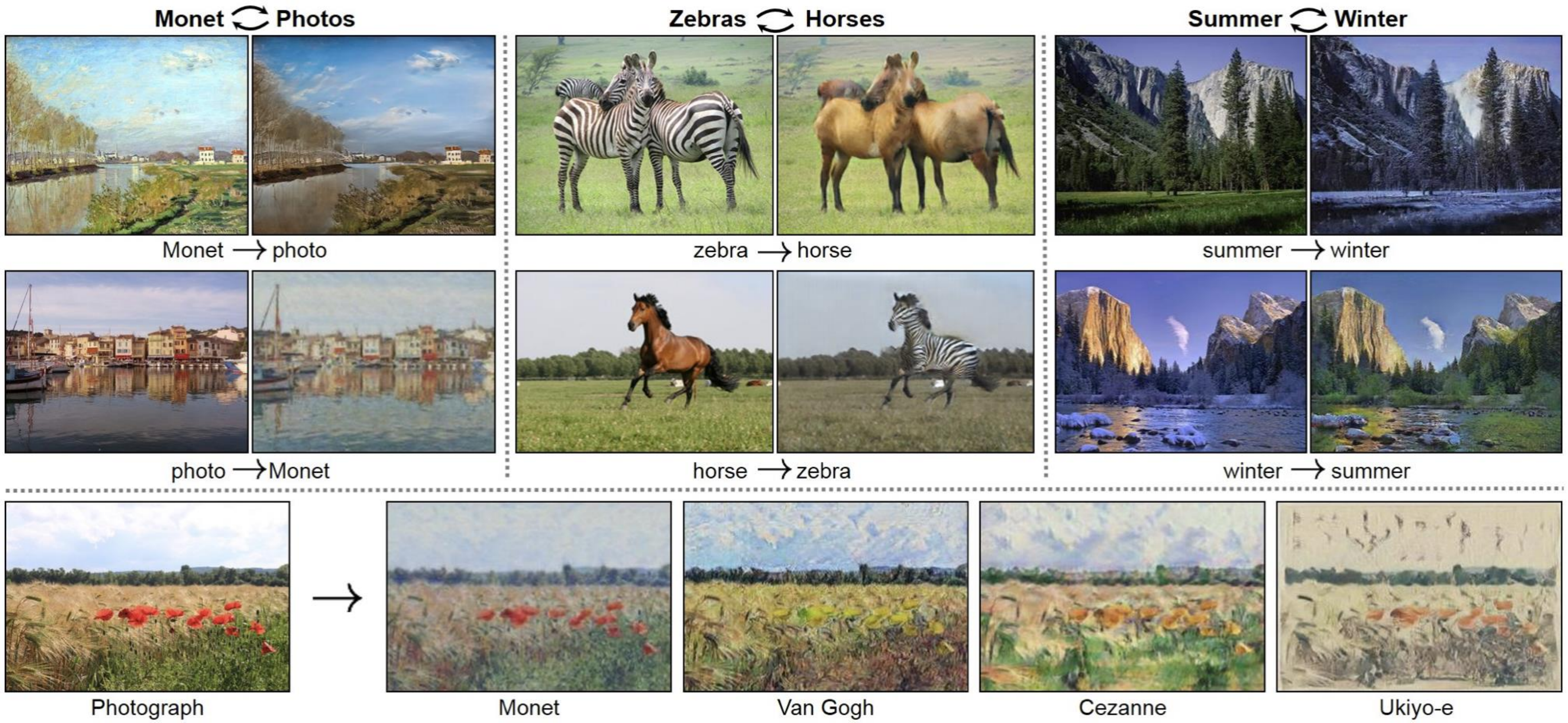


Text to Image Synthesis



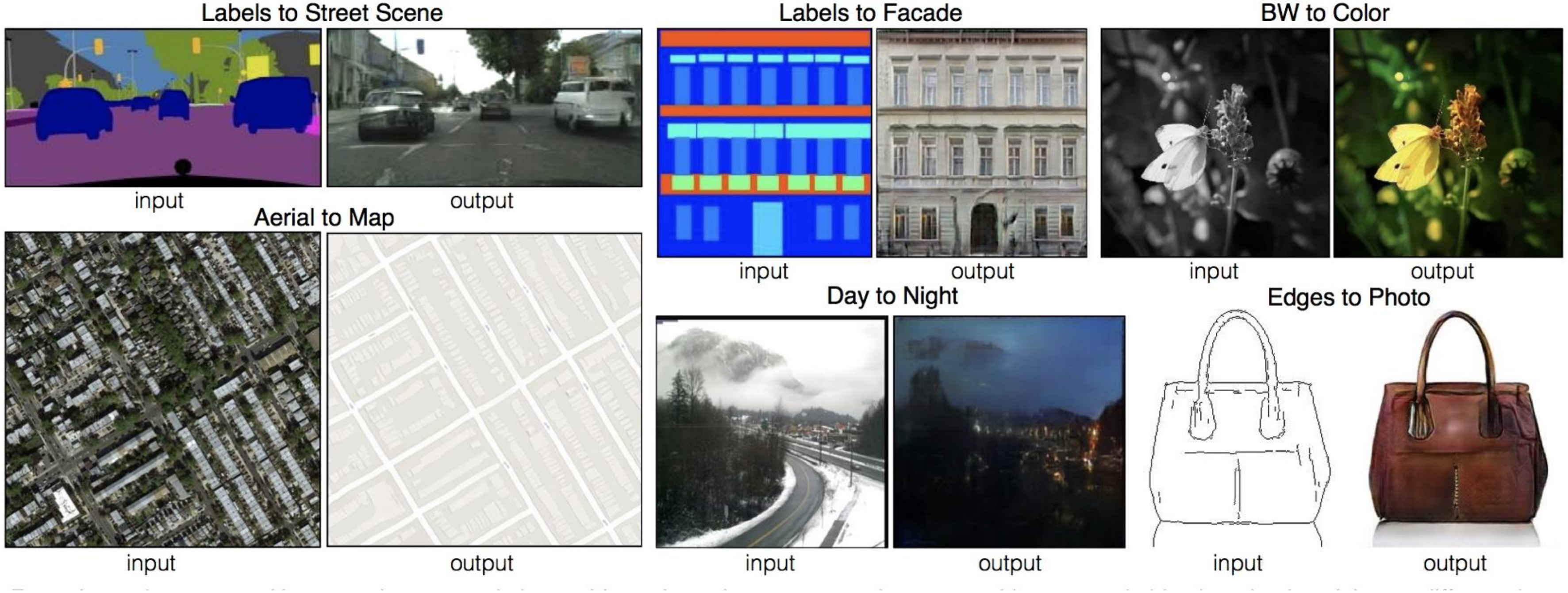
Reed et al. Generative Adversarial Text to Image Synthesis (2017)

CycleGAN



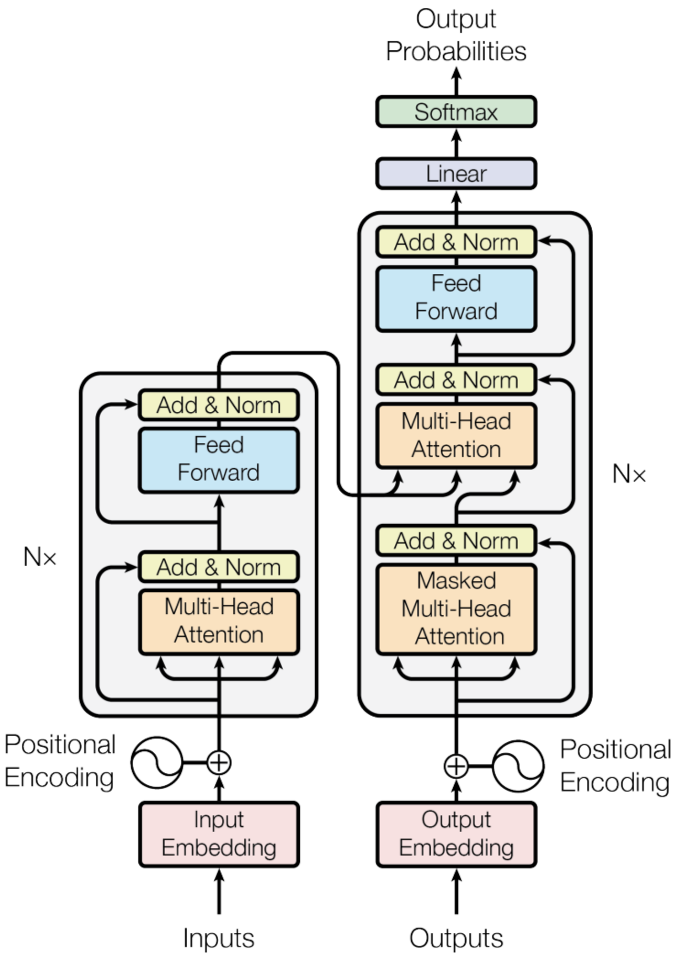
Zhu et al., Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks, ICCV 2017

Pix2Pix



P. Isola et al. Image-to-Image Translation with Conditional Adversarial Nets, CVPR 2017

Next episode in EEP 596…



Attention and **Transformer**