Unsupervised learning

EEML 2020
Mihaela Rosca
DeepMind & University College London

Unsupervised learning

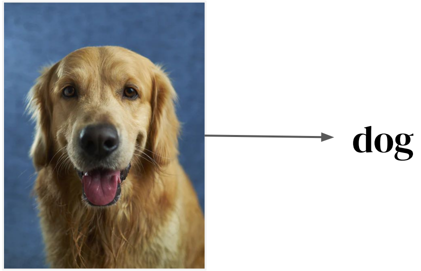
Aim: learn structure from data.

Types of learning

Supervised learning

Learn a mapping from input **x** to output **y.**

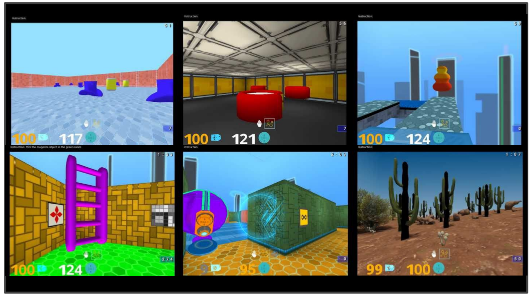
Challenge: generalization, having a flexible enough parametrization to learn the mapping.



Reinforcement learning

Learn behaviours to maximize rewards.

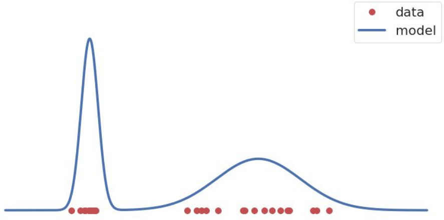
Challenge: finding rewarding behaviour (exploration), generalization, transfer.



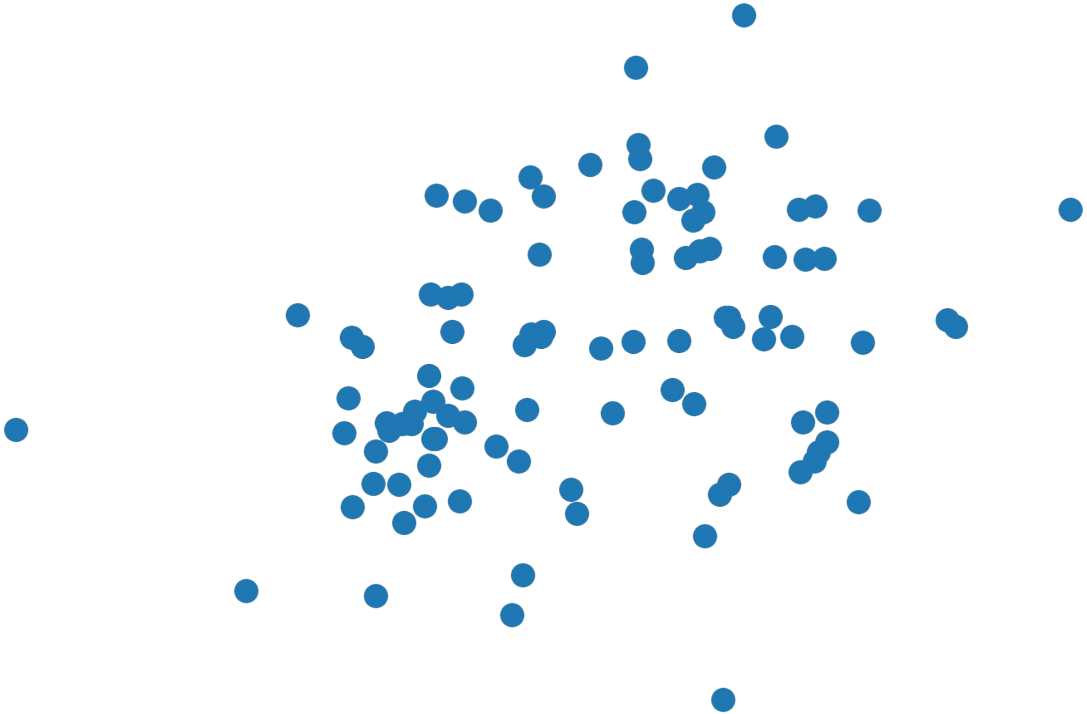
Unsupervised learning

Learn structure from data.

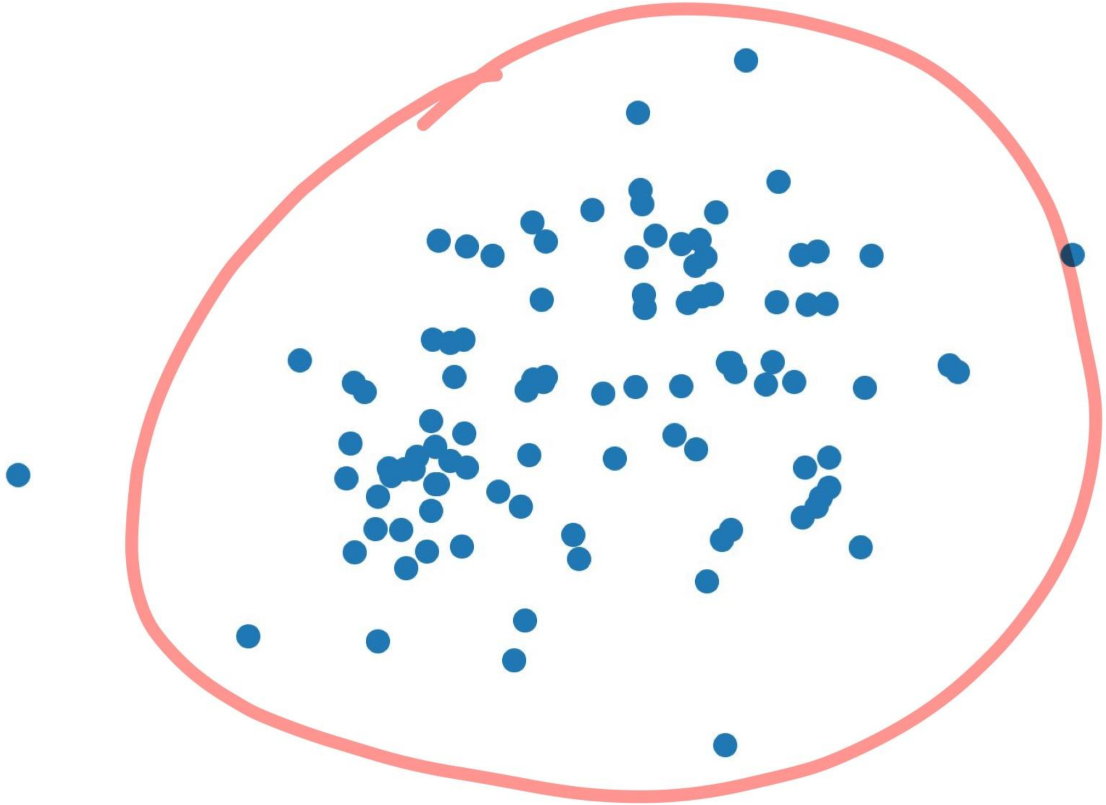
Challenge: No labels, no rewards. Generalization.



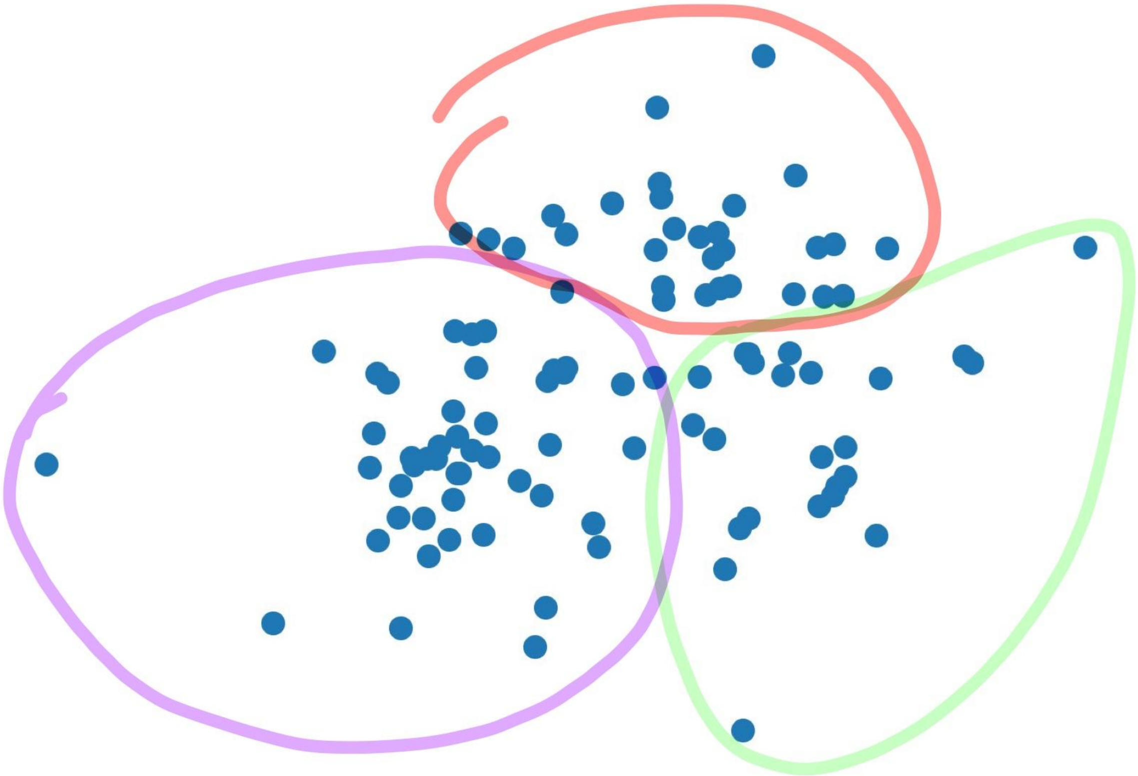
Unsupervised learning is hard



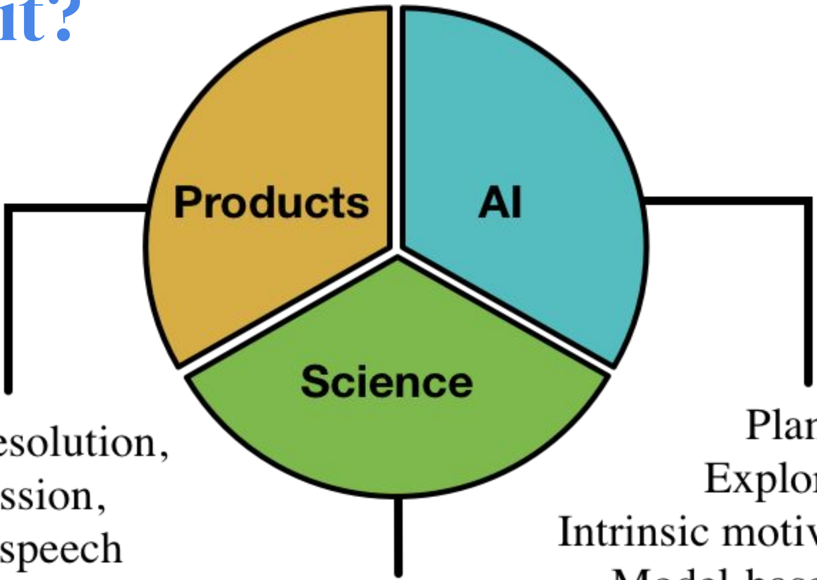
Unsupervised learning is hard

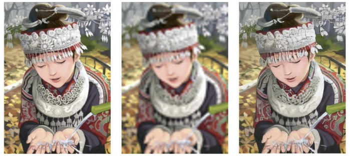


Unsupervised learning is hard



Why do we need it?

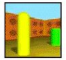




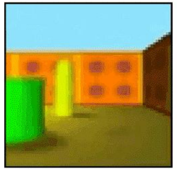
Super-resolution,
Compression,
Text-to-speech

Planning,
Exploration
Intrinsic motivation
Model-based RL

observation



neural rendering



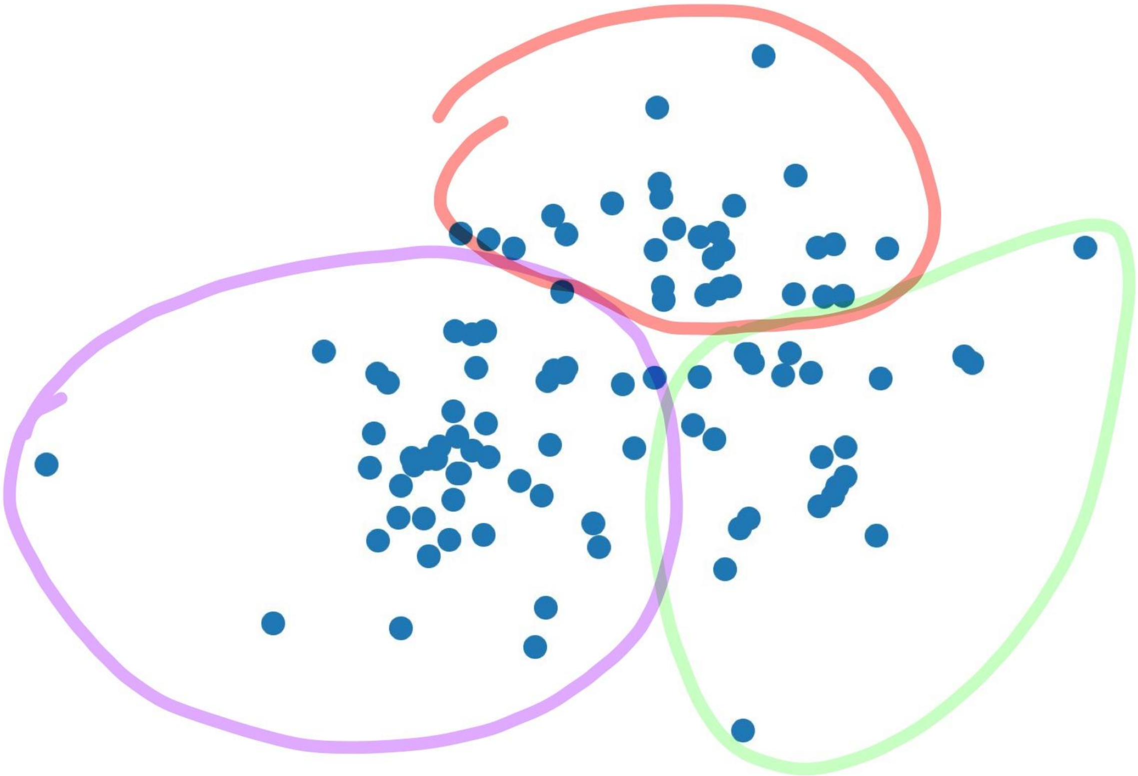
Proteomics,
Drug Discovery,
Astronomy,
High-energy physics

Types of unsupervised learning

- Clustering
- Generative modeling
- Representation learning

Often the lines can be blurry.

Clustering



Generative modeling

Learn a model of the true underlying data distribution $p^*(x)$ from samples

$$x_1, x_2 \dots x_n$$

Generative modeling

Learn a model of the true underlying data distribution $p^*(x)$ from samples

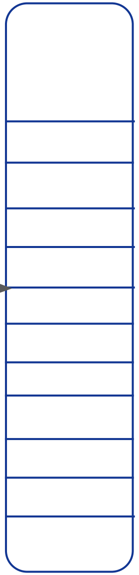
$x_1, x_2 \dots x_n$

Representation learning

The success of deep learning tells us about the importance of learning representations.

Easier for downstream tasks to work with learned representations rather than high dimensional data.

Representation learning



semi supervised learning

reinforcement learning

learned representations

unsupervised

supervised, RL

How to do unsupervised learning: a recipe

- Find an objective
- Find a model
- Find a way to learn the model using your objective
- Find an evaluation metric

The problem

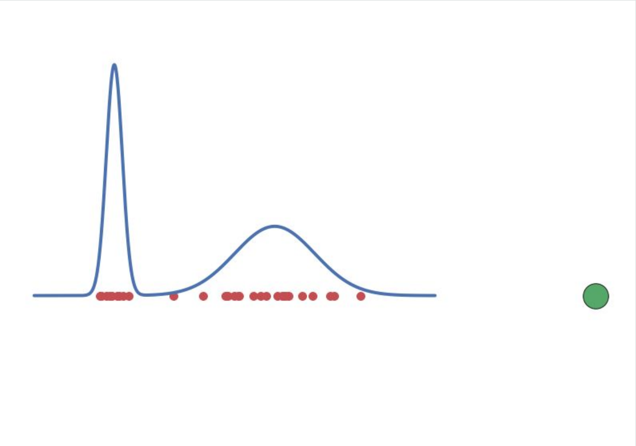
Case study: generative modeling

Learn a model of the true underlying data distribution $p^*(x)$ from samples

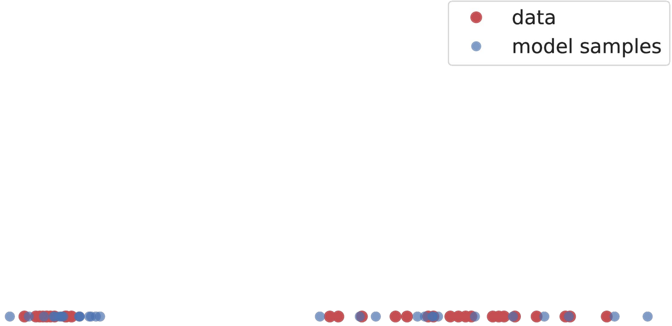
$$x_1, x_2 \dots x_n$$

What can we do with a distribution?

Query it



Sample from it



Types of data

Continuous data:

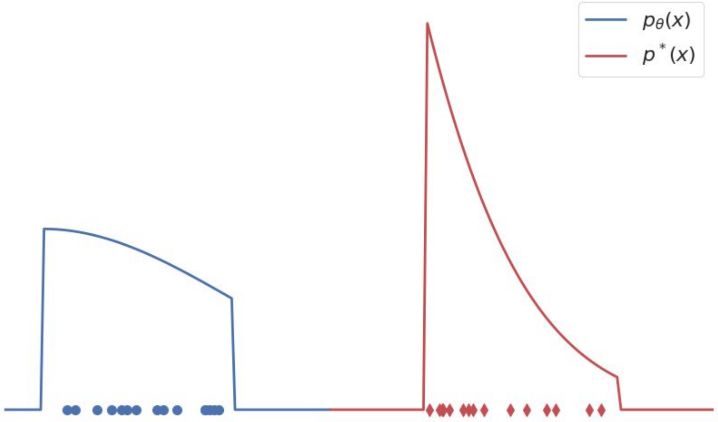
- Images
- Audio/ Speech
- Health data:
  - Age
  - Weight

Discrete data:

- Text
- Images
- Health data:
  - # of times someone was admitted to hospital
  - is smoker?

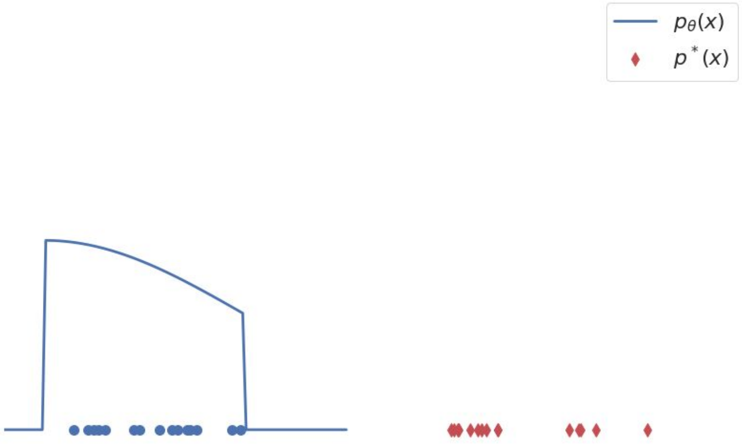
The objective

Measuring distances between distributions



How can we measure the distance between these two distributions?

Measuring distances between distributions



Caveat: we only have samples from the true distribution.

Monte Carlo estimation

How can we incorporate the data distribution in the objective if we only have samples from it?

$$\mathbb{E}_{p^*(\mathbf{x})} f(x) \approx \frac{1}{N} \sum_{i=1}^{N} f(\hat{x}_i)$$

Divergence and distance minimization

- The objective of generative models is often to minimize a divergence or distance.
- Most common: Maximum likelihood (KL divergence).

Why divergence/distance minimization?

$$D(p^* || p_\theta) = 0 \implies p_\theta = p^*$$

KL divergence - maximum likelihood

$$\mathrm{KL}(p^*(\mathbf{x})||p_\theta(\mathbf{x})) = \int p^*(\mathbf{x}) \log \frac{p^*(\mathbf{x})}{p_\theta(\mathbf{x})} d\mathbf{x}$$

$$= C - \int p^*(\mathbf{x}) \log p_\theta(\mathbf{x}) d\mathbf{x}$$

KL divergence - maximum likelihood

min $$\mathrm{KL}(p^*(\mathbf{x})||p_\theta(\mathbf{x})) = \int p^*(\mathbf{x}) \log \frac{p^*(\mathbf{x})}{p_\theta(\mathbf{x})} d\mathbf{x}$$

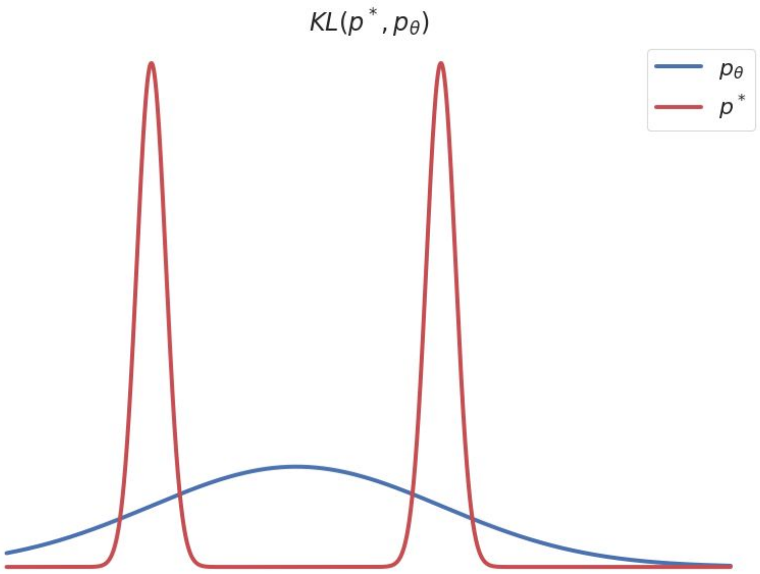
$$= C - \int p^*(\mathbf{x}) \log p_\theta(\mathbf{x}) d\mathbf{x}$$

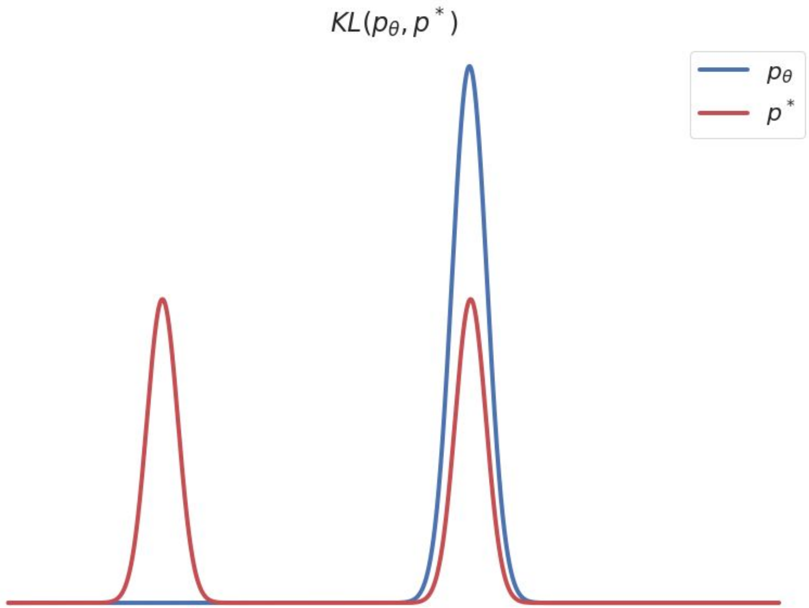
max $$\mathbb{E}_{p^*(\mathbf{x})} \log p_\theta(\mathbf{x})$$

Effects of the choice of divergence

Want to learn more?

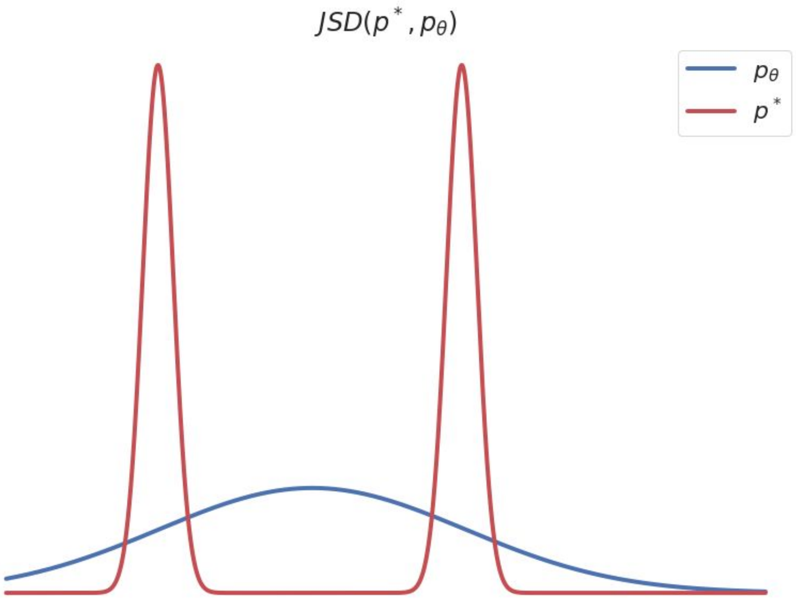
Goodfellow, et al. **NIPS 2016 Tutorial: Generative Adversarial Networks** Arxiv (2016)

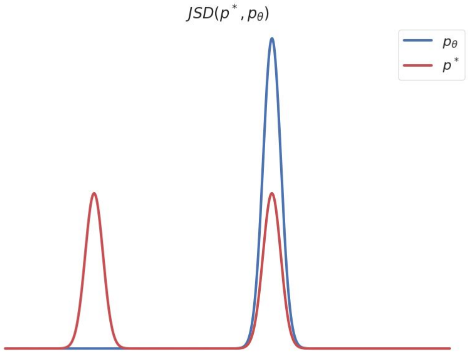


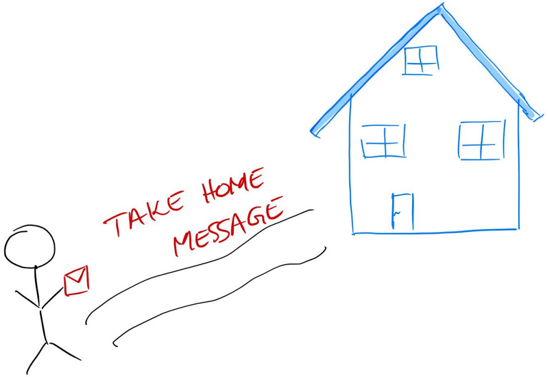


Jensen Shannon divergence

$$\mathrm{JSD}(p_\theta, p^*) = \mathrm{KL}(p_\theta, \frac{p_\theta + p^*}{2}) + \mathrm{KL}(p^*, \frac{p_\theta + p^*}{2})$$







Given a model family, different divergences can lead to a vastly different distribution.

Optimising Reverse KL & JSD - not so easy

$$\mathrm{KL}(p^*(\mathbf{x})||p_\theta(\mathbf{x})) = \int p^*(\mathbf{x}) \log \frac{p^*(\mathbf{x})}{p_\theta(\mathbf{x})} d\mathbf{x}$$

$$= C - \int \underbrace{p^*(\mathbf{x}) \log p_\theta(\mathbf{x}) d\mathbf{x}}_{\text{Monte Carlo estimation}}$$

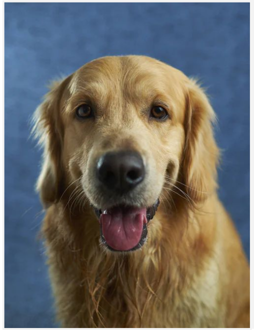
$$\mathrm{KL}(p_\theta(\mathbf{x})||p^\star(\mathbf{x})) = \int p_\theta(\mathbf{x}) \log \frac{p_\theta(\mathbf{x})}{p^\star(\mathbf{x})} d\mathbf{x}$$

$$= \int \underbrace{p_\theta(\mathbf{x}) \log p_\theta(\mathbf{x}) d\mathbf{x}}_{\text{Entropy hard to estimate}} - \int \underbrace{p_\theta(\mathbf{x}) \log p^\star(\mathbf{x}) d\mathbf{x}}_{\text{Need access to the true data distribution}}$$

Beyond divergence minimization - two player games

Discriminator

Learns to distinguish between real and generated data.



vs

Created by Minus icons
from Noun Project

Generator

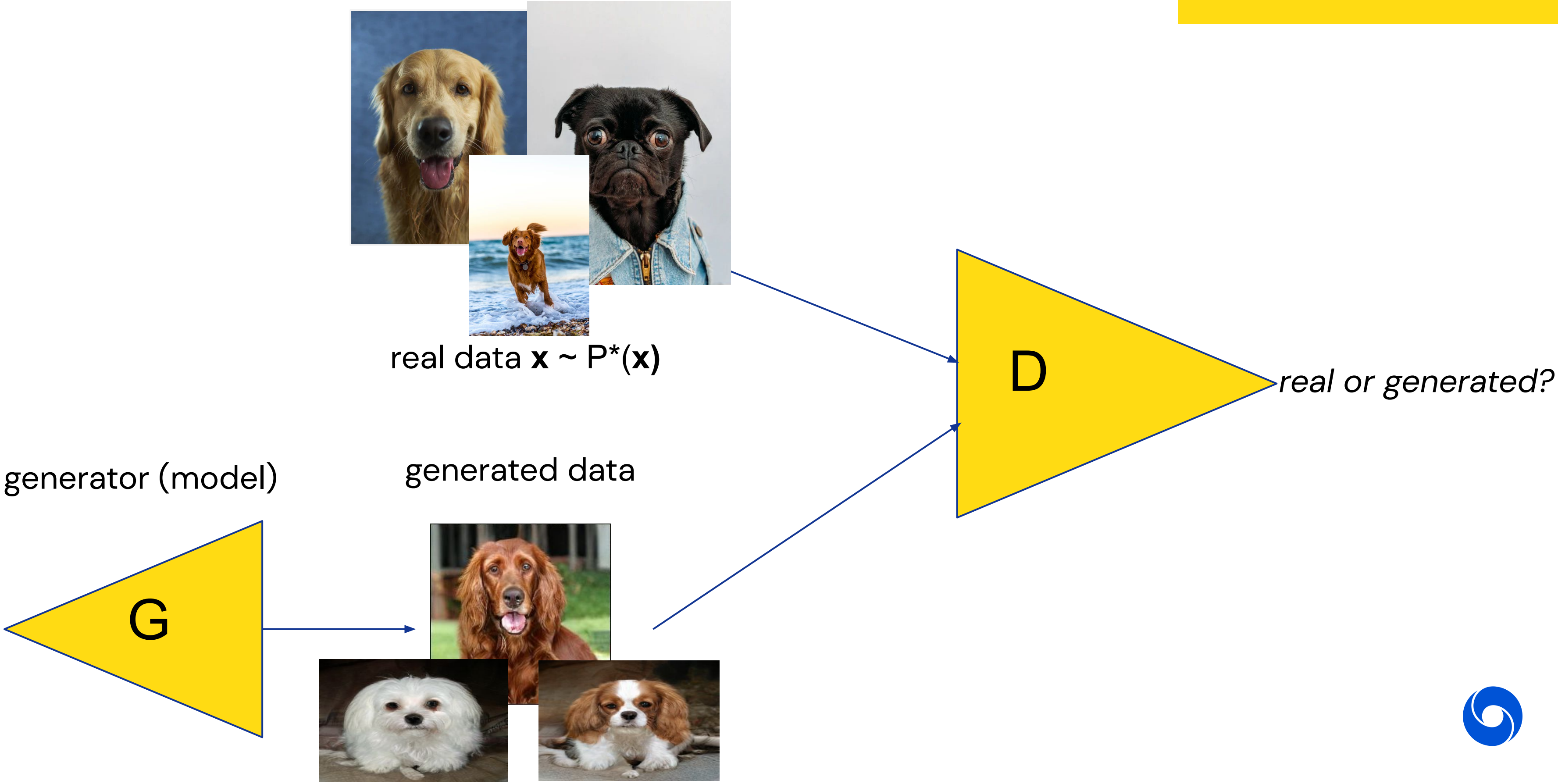
Learns to generate data to “fool” the discriminator.

Created by Minus icons
from Noun Project

Generative adversarial networks

Want to learn more?

Goodfellow, et al. **Generative adversarial networks..** Neural Information Processing Systems (**2014**)



Generative adversarial networks

Want to learn more?

Goodfellow, et al. **Generative adversarial networks..** Neural Information Processing Systems (**2014**)

$$\min_{\boldsymbol{\theta}} \max_{\boldsymbol{\phi}} \underbrace{\mathbb{E}_{p^*(\mathbf{x})}\left[\log \mathcal{D}_{\boldsymbol{\phi}}(\mathbf{x})\right]}_{\text{log-probability that D correctly predicts real data } \mathbf{x} \text{ are real}} + \underbrace{\mathbb{E}_{p_{\boldsymbol{\theta}}(\mathbf{x})}\left[\log(1 - \mathcal{D}_{\boldsymbol{\phi}}(\mathbf{x}))\right]}_{\text{log-probability that D correctly predicts generated data are generated}}$$

Are GANs doing divergence minimization?

$$\min_{\boldsymbol{\theta}} \max_{\boldsymbol{\phi}} \mathbb{E}_{p^*(\mathbf{x})}\left[\log \mathcal{D}_{\boldsymbol{\phi}}(\mathbf{x})\right] + \mathbb{E}_{p_{\boldsymbol{\theta}}(\mathbf{x})}\left[\log(1 - \mathcal{D}_{\boldsymbol{\phi}}(\mathbf{x}))\right]$$

If the discriminator (D) is optimal:
the generator is minimizing the Jensen Shannon divergence between the true and generated distributions.

Connection to optimality:

$$JSD(p^* || p_\theta) = 0 \implies p_\theta = p^*$$

Other divergences and distances

Want to learn more?

Arjovsky,, et al **Wasserstein GAN** ICML **(2017)**

Wasserstein Distance

$$W(p^*, p_\theta) = \sup_{||f||_L \leq 1} \mathbb{E}_{p^*(\mathbf{x})} f(\mathbf{x}) - \mathbb{E}_{p_\theta(\mathbf{x})} f(\mathbf{x})$$

$$|f(x) - f(y)| \leq |x - y|$$

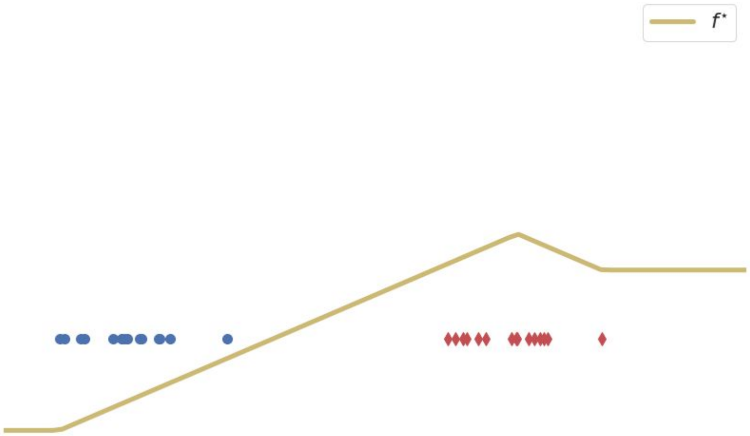
Other divergences and distances

Want to learn more?

Arjovsky,, et al **Wasserstein GAN**
ICML **(2017)**

Wasserstein Distance

$$W(p^*, p_\theta) = \sup_{||f||_L \leq 1} \mathbb{E}_{p^*(\mathbf{x})} f(\mathbf{x}) - \mathbb{E}_{p_\theta(\mathbf{x})} f(\mathbf{x})$$



Other divergences and distances

Wasserstein Distance

Estimation

$$W(p^*, p_\theta) = \sup_{||f||_L \leq 1} \mathbb{E}_{p^*(\mathbf{x})} f(\mathbf{x}) - \mathbb{E}_{p_\theta(\mathbf{x})} f(\mathbf{x})$$

Learning

$$\min_{\theta} \max_{\substack{\phi, \\ ||D_\phi||_L \leq 1}} \mathbb{E}_{p^*(\mathbf{x})} D_\phi(\mathbf{x}) - \mathbb{E}_{p_\theta(\mathbf{x})} D_\phi(\mathbf{x})$$

GANs: More than divergence minimization

In practice D is not optimal:

→ limited computational resources

→ we do not have access to the true data distribution (just samples)

Discriminators as learned “distances”

Want to learn more?

Arora, et al **Generalization and Equilibrium in Generative Adversarial Nets.** ICLR **(2017)**

$$\min_G \boxed{\max_D V(D,G)}$$

We can think of D (the discriminator) as learning a “distance” between the data and model distribution that can provide useful gradients to the model.

GANs (learned distance) or divergence minimization?

GANs

- good samples
- learned loss function

- hard to analyze dynamics (game theory)
- (in practice) no optimal convergence guarantees

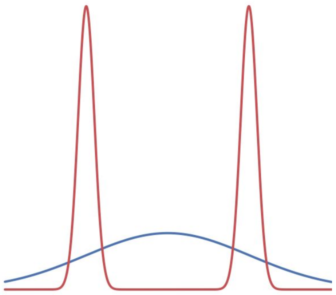
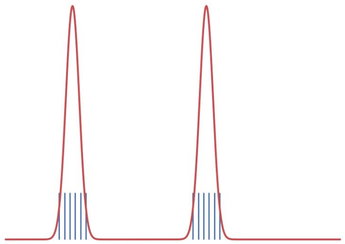
Divergence minimization

- optimal convergence guarantees
- easy to analyze loss properties

- hard to get good samples
- loss functions don't correlate with human evaluation

The model

The importance of the model (in maximum likelihood training)

Model	Not powerful enough	→	Too powerful
Behaviour	 bad model fit		 overfitting

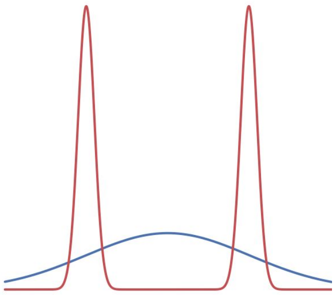
The importance of the model (in maximum likelihood training)

Model

Not too powerful

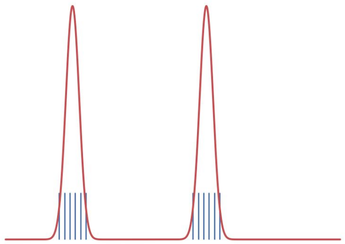
Too powerful

Behaviour



bad model fit

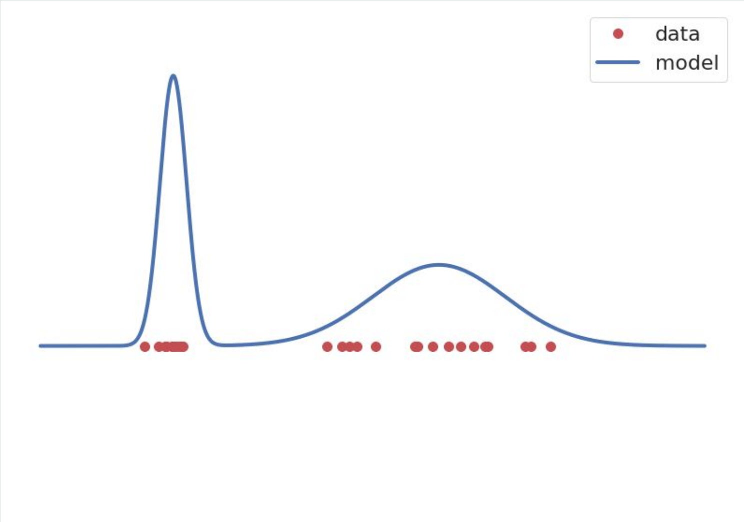
where we want to be



overfitting

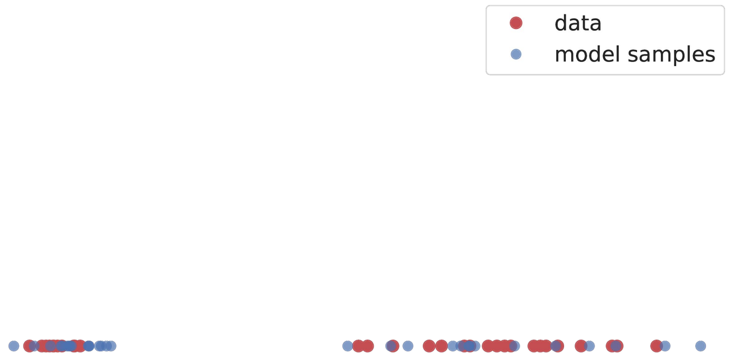
Explicit likelihood models

Model the density p(x).

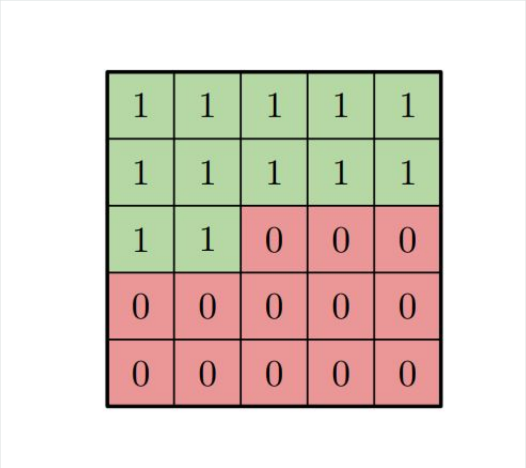


Implicit models

Do not model the density, but the sampling path.

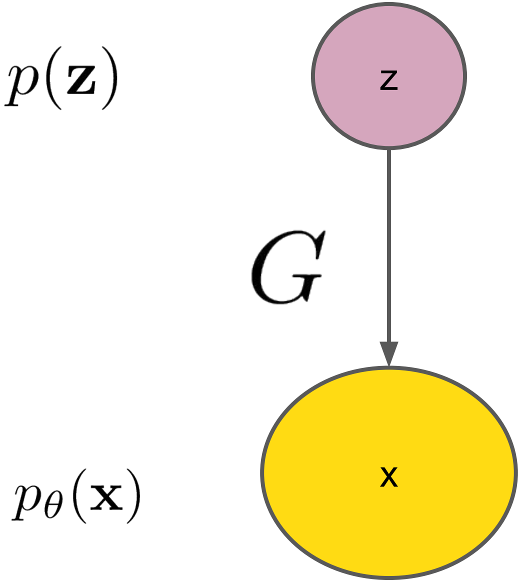


Observed variable models

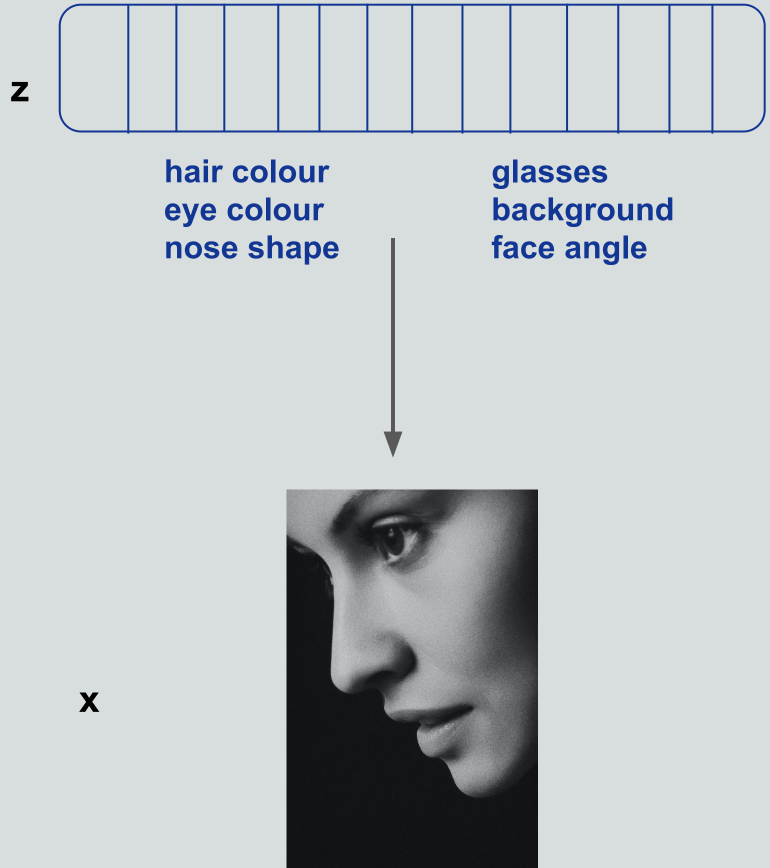


$$p_\theta(\mathbf{x}) = \prod_{i=0}^{N} p_\theta(x_i | x_{<i})$$

Latent variable models

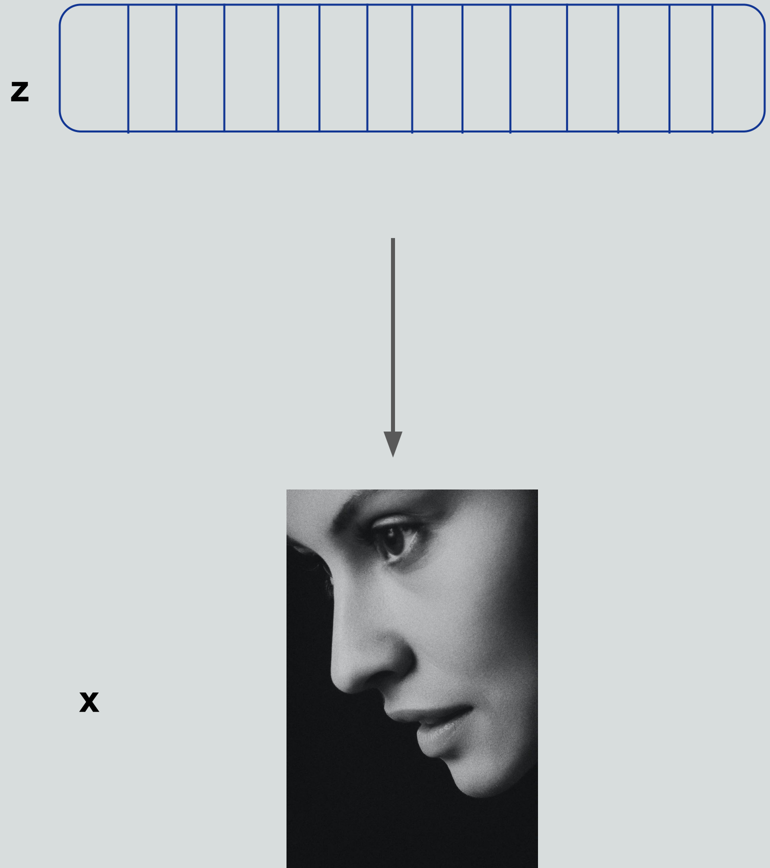


Generation

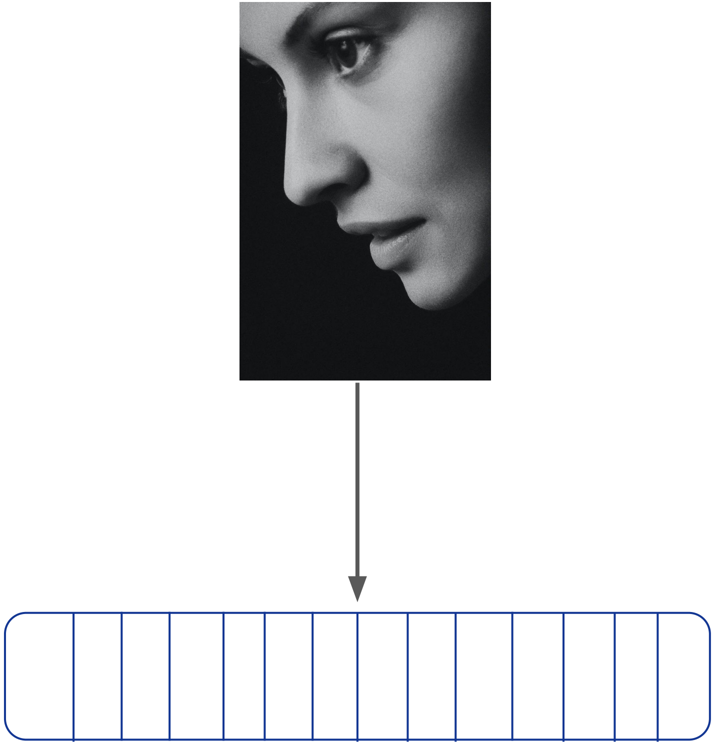


- Challenge: learning the factor unsupervised
- Sampling is often cheap
- Representation learning
  - Inverting the generation process = inference

Generation

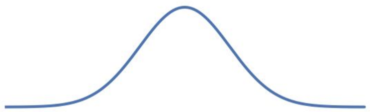


Inference



Explicit models - canonical distributions

$$p_\theta(\mathbf{x}) = \mathcal{N}(\mathbf{x}; \mu, \Sigma)$$



- Learn parameters of canonical distribution
- Example: Gaussian, Poisson
- Pro: Easy to learn
- Con: Not very expressive, especially in high dimensions

Explicit models - mixture models

$$p_{\theta,\pi}(\mathbf{x}) = \sum_{k=1}^{K} \pi_k p_\theta(\mathbf{x}|\mathbf{z} = k)$$

- Pro: models multi modality.
- Con: number of modes are fixed.

- Mixture components can be simple or complex distributions.

Explicit models - autoregressive models

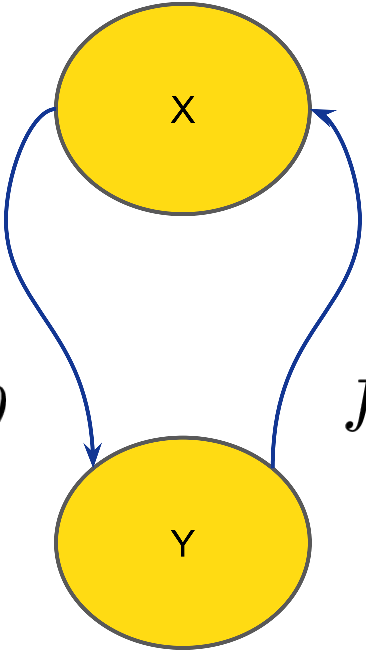
$$p_\theta(\mathbf{x}) = \prod_{i=0}^{N} p_\theta(x_i | x_{<i})$$

- Pro: Very expressive
- Challenge: Slow at sampling (though can be parallelize)
- Modality: great for sequential data, text, audio but have also been used for images

Explicit models - normalizing flows

Want to learn more?

Rezende et all, **Variational Inference with Normalizing Flows**, ICML 2015



f_θ

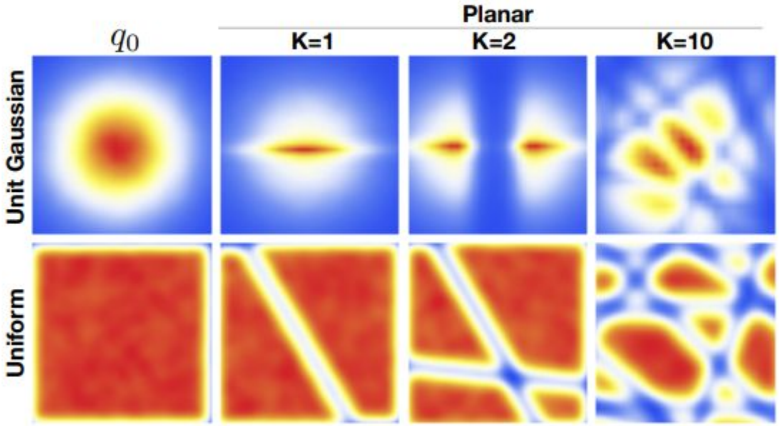
f_θ^{-1}

$$p_{\mathbf{y}}(\mathbf{y}) = p_{\mathbf{x}}(f_\theta^{-1}(\mathbf{y})) \det \left| \frac{df_\theta^{-1}(\mathbf{z})}{d\mathbf{z}} \Big|_{\mathbf{z}=\mathbf{y}} \right|$$

X and Y have the same dimension!

Challenge: modeling invertible functions using neural networks

Explicit models - normalizing flows



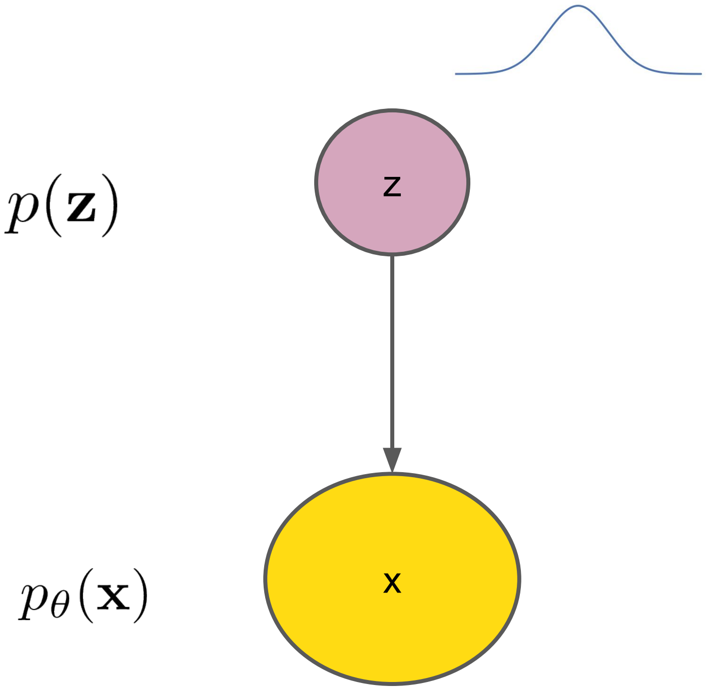
K = number of flows

Composing normalizing flows leads to another flow.

Simple transformations can be used to build complex distributions.

Explicit latent variable models

$$p_\theta(\mathbf{x}) = \int p_\theta(\mathbf{x}|\mathbf{z})p(\mathbf{z})d\mathbf{z}$$



Explicit latent variable models

Want to learn more?

Kingma et al., **Auto-Encoding Variational Bayes,** ICLR 2014

Lower bound on maximum likelihood objective (ELBO):

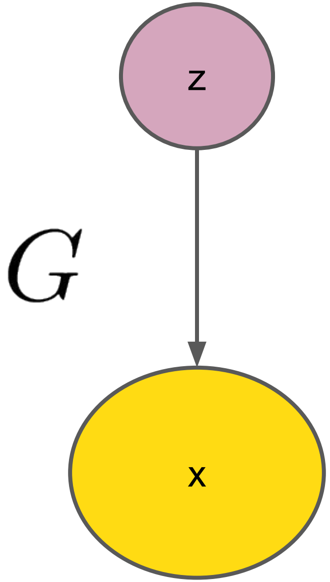
$$\log p_\theta(\mathbf{x}) \geq \mathbb{E}_{q_\eta(\mathbf{z}|\mathbf{x})} \log p_\theta(\mathbf{x}|\mathbf{z}) - \mathrm{KL}(q_\eta(\mathbf{z}|\mathbf{x})||p(\mathbf{z}))$$

Approximate posterior $q_\eta(\mathbf{z}|\mathbf{x})$

Implicit models - latent variable models

Directly the sampling path, without require likelihoods explicitly (no need for the sum rule).

Often not trained with maximum likelihood, but suitable for GAN training.



Learning

Learn using divergence minimization

Maximum likelihood:

$$\mathbb{E}_{p^*(\mathbf{x})}[\log p_\theta(\mathbf{x})]$$

To learn parameters by gradient descent:

$$\nabla_\theta \mathbb{E}_{p^*(x)}[\log p_\theta(x)] = \underbrace{\mathbb{E}_{p^*(x)} \nabla_\theta [\log p_\theta(x)]}_{\textbf{Monte Carlo estimation}}$$

Stochastic gradient estimation

$$\nabla_\theta \mathbb{E}_{p_\theta(\mathbf{x})} f(\mathbf{x})$$

Cannot put the gradient inside the expectation. But there are other ways to leverage Monte Carlo estimation to compute gradients.

A few training criteria affected

GANs

$$\min_{\boldsymbol{\theta}} \max_{\boldsymbol{\phi}} \mathbb{E}_{p^*(\mathbf{x})}[\log \mathcal{D}_{\boldsymbol{\phi}}(\mathbf{x})] + \boxed{\mathbb{E}_{p_{\boldsymbol{\theta}}(\mathbf{x})}}[\log(1 - \mathcal{D}_{\boldsymbol{\phi}}(\mathbf{x}))]$$

Bound on ML (ELBO)

$$\max_{\theta, \eta} \boxed{\mathbb{E}_{q_\eta(\mathbf{z}|\mathbf{x})}} \log p_\theta(\mathbf{x}|\mathbf{z}) - \mathrm{KL}(q_\eta(\mathbf{z}|\mathbf{x})||p(\mathbf{z}))$$

Options

Want to learn more?

Mohamed, et al. **Monte Carlo gradient estimation in Machine learning**. JMLR (2020)

Score function

- few assumptions on cost
- no assumptions on p
- often high variance
- discrete and continuous data

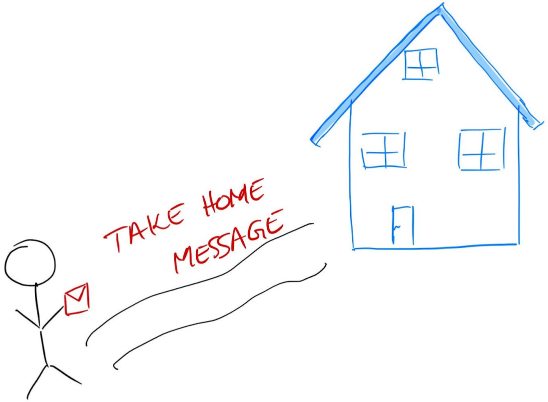
Pathwise

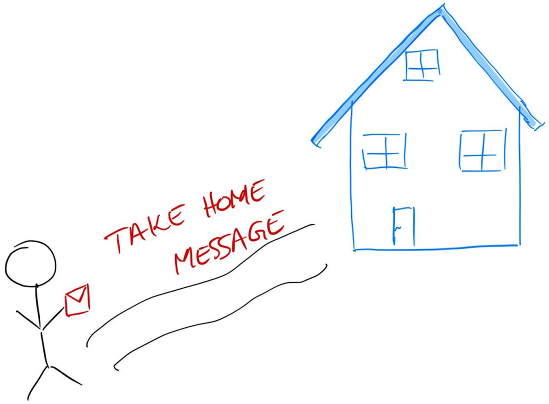
- cost needs to be differentiable
- assumptions on p
- often low variance
- continuous data

Measure valued

- few assumptions on cost
- computationally expensive
- low variance

We are interested in having Monte Carlo estimators not only for the loss, but also to estimate gradients for learning.





Often papers present **algorithms,** which are a choice of:

- objective
- model
- learning choice (parameter update rules)

Models, training and learning criteria

Explicit models are often trained by maximum likelihood:

$$\mathbb{E}_{p^*(\mathbf{x})} \log \underline{p_\theta}(\mathbf{x})$$

Autoregressive models trained by maximum likelihood

Want to learn more?
Radford, et al. **Language Models are Unsupervised Multitask Learners,** OpenAI Blog (2019)

Want to learn more?
van den Oord , et al. **Conditional Image Generation with PixelCNN Decoders,** Neurips (2016)

Want to learn more?
van den Oord , et al. **Pixel Recurrent Neural Networks,** ICML (2016)

Want to learn more?
van den Oord , et al. **WaveNet: A Generative Model for Raw Audio,** arxiv (2016)

- PixelCNN/PixelRNN (image data)
- Wavenet (audio)
- GPT (text)

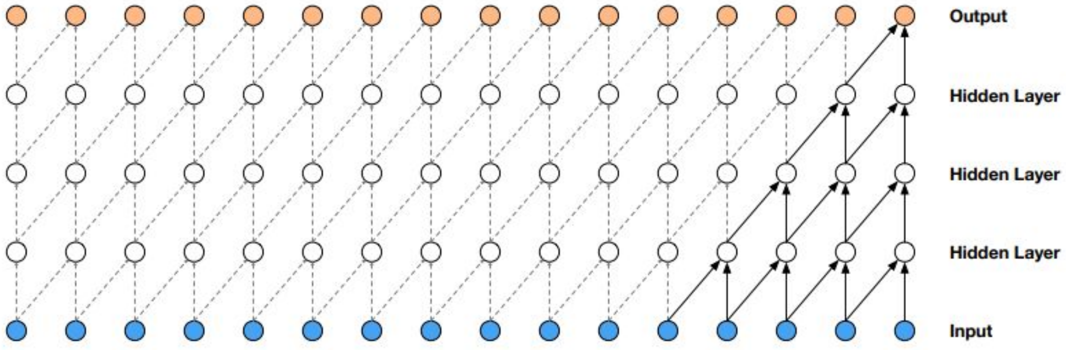


Figure for van den Oord, 2016.

Implicit latent variable models & GAN training

$$\min_{\boldsymbol{\theta}} \max_{\boldsymbol{\phi}} \mathbb{E}_{p^*(\mathbf{x})} \left[\log \mathcal{D}_{\boldsymbol{\phi}}(\mathbf{x})\right] + \underline{\mathbb{E}_{p_{\boldsymbol{\theta}}(\mathbf{x})}} \left[\log(1 - \mathcal{D}_{\boldsymbol{\phi}}(\mathbf{x}))\right]$$

You will often see the GAN criteria written as:

$$\min_{\theta} \max_{\phi} \mathbb{E}_{p^*(\mathbf{x})} \log D_{\phi}(\mathbf{x}) + \underline{\mathbb{E}_{p(\mathbf{z})}} \log(1 - D_{\phi}(G_{\theta}(\mathbf{z})))$$

This assumes:

- The GAN ***model*** is an implicit latent variable model (need not be).
- The model is ***learned*** using the pathwise estimator (need not be).

Evaluation

Why evaluation is hard

Want to learn more?

Theis, et al **A note on the evaluation of generative models**
International Conference for Learning Representations **(2016)**

No evaluation metric is able to capture all desired properties.

- sample quality
- generalization
- representation learning

Evaluate based on end goal

- semi supervised learning: classification accuracy
- reinforcement learning: agent reward
- data generation: human (user) evaluation

Applications

Image generation

Implicit Latent variable + GAN

Photo realistic sample quality.

Modality matters: GANs on discrete data such as text are harder to train.

Want to learn more?

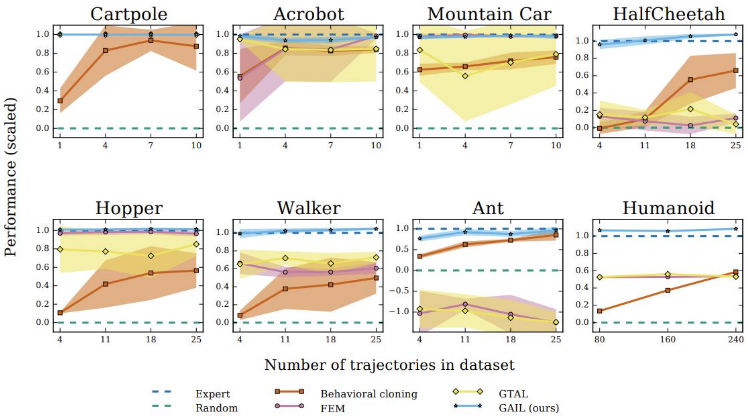
Brock et all **Large scale gan training for high fidelity natural image synthesis** Neurips (2018)

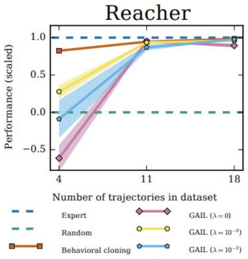
Generative Adversarial Imitation Learning

Want to learn more?

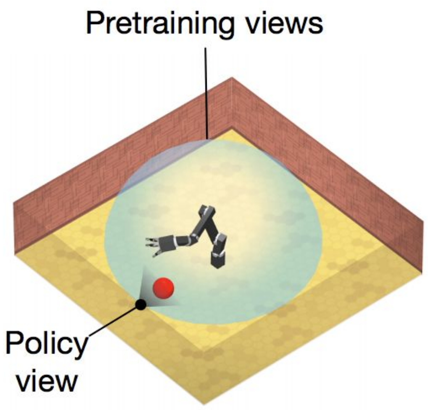
Ho et all, **Generative Adversarial Imitation Learning** Neurips (2016)

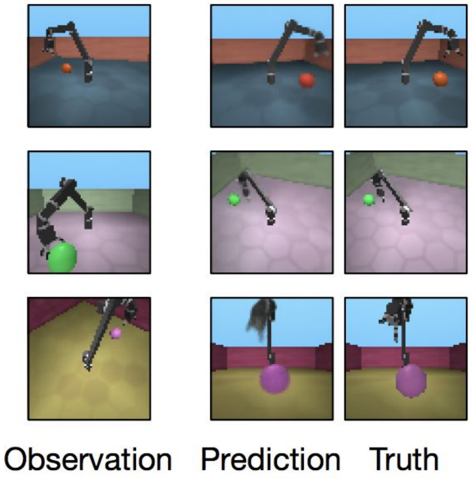
Learn agents to imitate the behaviour of an expert (human), using a discriminator.

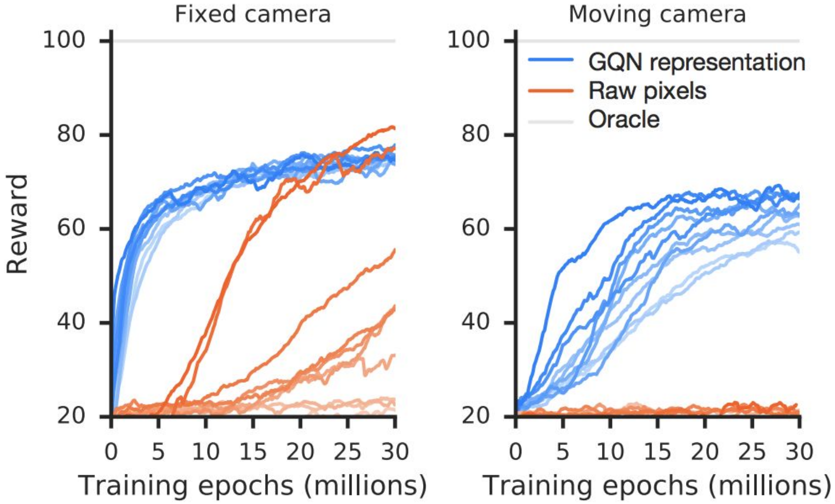




Representation learning with explicit latent variable models (GQN)







Want to learn more?

Eslami et all, **Neural scene representation and rendering**, Science (2018)

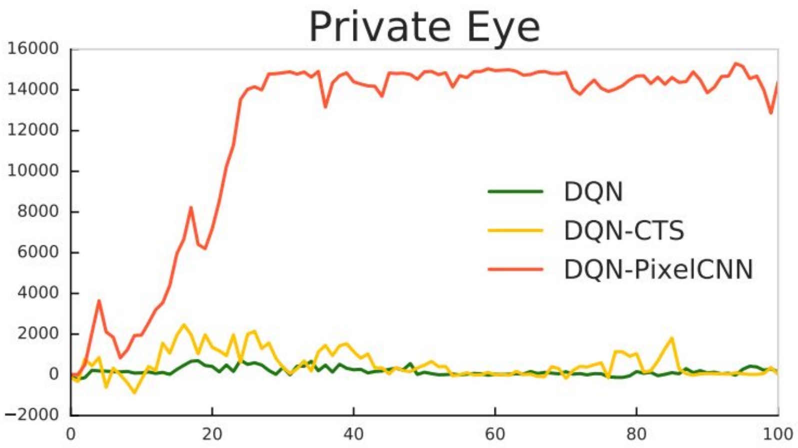
Slide thanks to Ali Eslami.

Exploration in RL

Want to learn more?

Ostrovski, et al **Count-Based Exploration with Neural Density Models** International Conference on Machine Learning **(2016)**

Density estimation can be used to test for out of distribution data.



In RL, this can be used to provide an exploration bonus for unseen states:

have I been here before?

Multi task language learning

Want to learn more?

Radford, et al .**Language Models are Unsupervised Multitask Learners** (2019)

Autoregressive text models trained by maximum likelihood can be used for multiple downstream tasks.

Key: Neural architecture, billions of parameters and large amounts of data

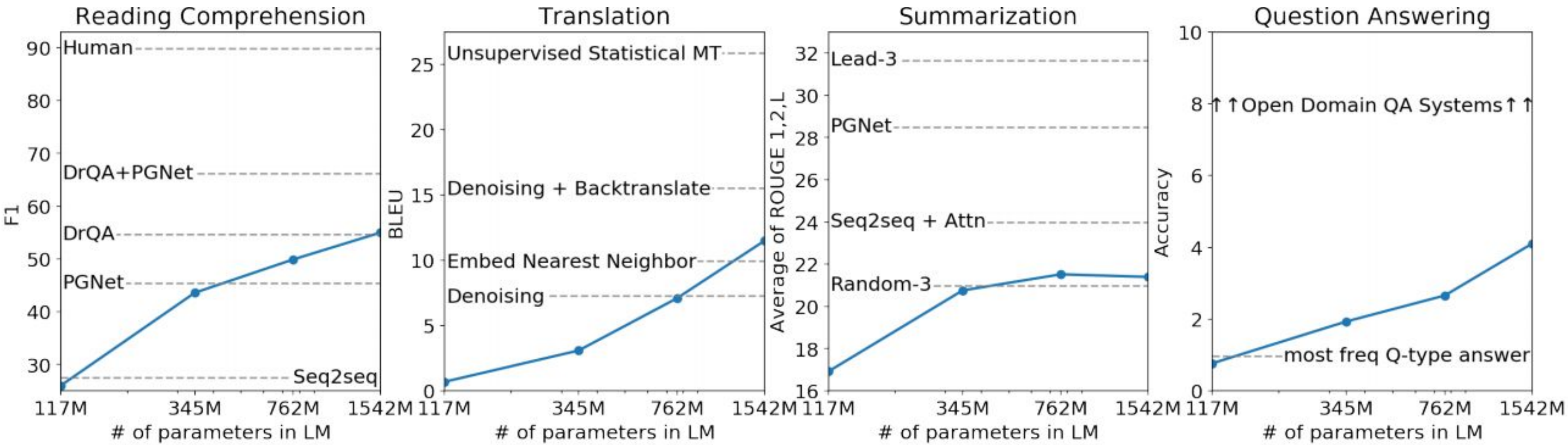
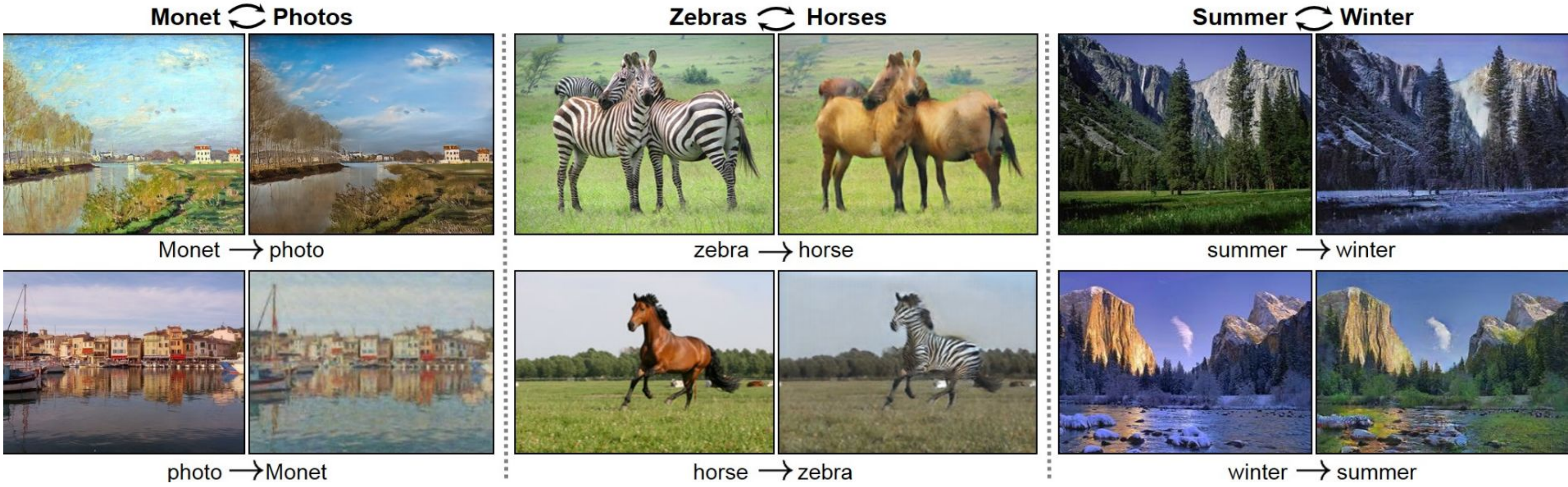


Figure from Radford et al. (2019)

Image to image translation - CycleGAN

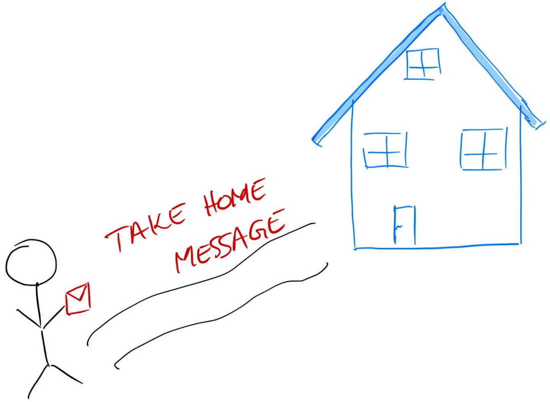
Want to learn more?

Zhu et all, **Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks,** ICCV (2017)



Conclusion

You have choices! Many choices!



Options

Objective

- Divergence minimization
- Adversarial approaches

Model

- Explicit models
- Implicit models
- Observed models
- Latent variable models

Learning

- Monte Carlo estimators

Thank you!