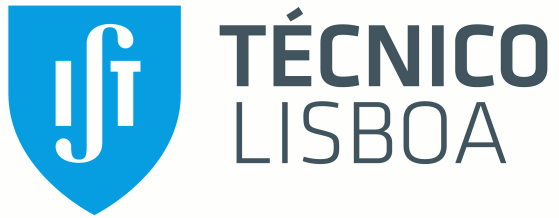
Lecture 13: Deep Generative Models

André Martins, Francisco Melo, Mário Figueiredo



Deep Learning Course, Winter 2021-2022

Announcements

- This Friday, 15:00–17:00, Centro de Congressos (Pav. Civil, Alameda): guest lecture by Prof. Chrysoula Zerva!

Today's Roadmap

Most of the course was about supervised learning.

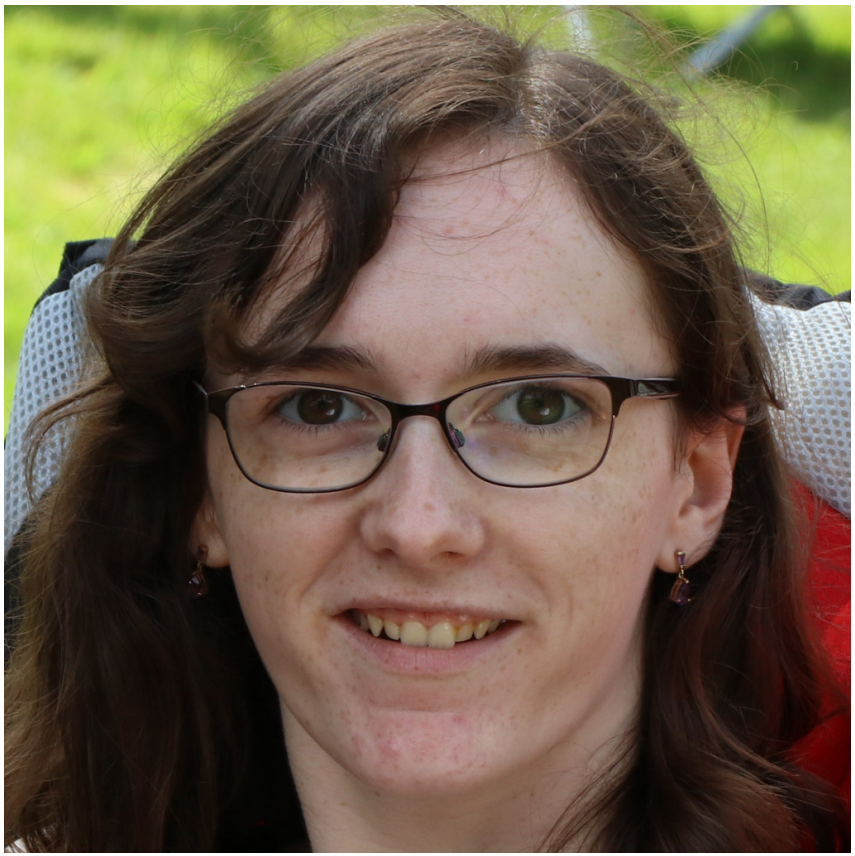
Today we'll talk about deep generative models and unsupervised learning.

- Deep auto-regressive models
- Boltzmann machines
- Evidence lower bound (ELBO) and variational inference
- Variational auto-encoders
- Generative adversarial networks

Which of these people is real?

(http://www.whichfaceisreal.com)

Which of these people is real?



(http://www.whichfaceisreal.com)

Which of these people is real?

(http://www.whichfaceisreal.com)

Which of these people is real?

(http://www.whichfaceisreal.com)

Which of these people is real?

(`http://www.whichfaceisreal.com`)

Which of these people is real?

(http://www.whichfaceisreal.com)

Which of these people is real?

(http://www.whichfaceisreal.com)

Which of these people is real?

(http://www.whichfaceisreal.com)

Which of these people is real?

(http://www.whichfaceisreal.com)

Which of these people is real?

(http://www.whichfaceisreal.com)

Generative Modeling

- Modelling complex high-dimensional data is a hard, open problem
- Deep generative models are currently making progress on this.
- **Goal:** model $\mathbb{P}(\boldsymbol{x})$ (unsupervised learning) or $\mathbb{P}(\boldsymbol{x}, \boldsymbol{y})$ (supervised learning)
- Often, deep generative models also use latent variables $\boldsymbol{h}$, in which case they may model $\mathbb{P}(\boldsymbol{x}, \boldsymbol{h})$ or $\mathbb{P}(\boldsymbol{x}, \boldsymbol{h}, \boldsymbol{y})$, such that

$$\mathbb{P}(\boldsymbol{x}) = \sum_{\boldsymbol{h}} \mathbb{P}(\boldsymbol{x}, \boldsymbol{h}) \quad \text{or} \quad \mathbb{P}(\boldsymbol{x}, \boldsymbol{y}) = \sum_{\boldsymbol{h}} \mathbb{P}(\boldsymbol{x}, \boldsymbol{h}, \boldsymbol{y})$$

Examples of Deep Generative Models

- Auto-Regressive Networks
- Restricted Boltzmann Machines
- Deep Belief Networks
- Deep Boltzmann Machines
- Gaussian-Bernoulli RBMs
- Convolutional Boltzmann Machines
- Sigmoid Belief Nets
- Variational Auto-Encoders
- Generative Adversarial Networks
- Convolutional Generative Networks
- Generative Stochastic Networks

Examples of Deep Generative Models

- Auto-Regressive Networks
- Restricted Boltzmann Machines
- Deep Belief Networks
- Deep Boltzmann Machines
- Gaussian-Bernoulli RBMs
- Convolutional Boltzmann Machines
- Sigmoid Belief Nets
- Variational Auto-Encoders
- Generative Adversarial Networks
- Convolutional Generative Networks
- Generative Stochastic Networks

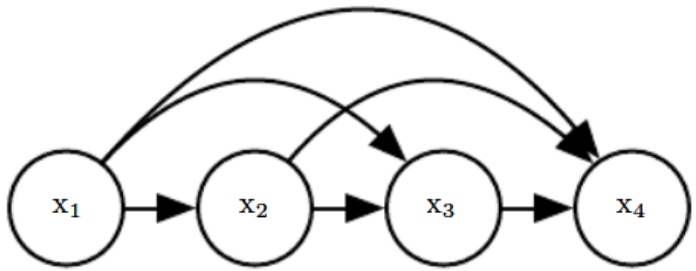
Outline

1. Deep Auto-Regressive Models
2. Boltzmann Machines
3. Variational Auto-Encoders
   - Variational Inference and ELBO
   - Gradients and Reparameterization Trick
4. Generative Adversarial Networks
5. Conclusions

Deep Auto-Regressive Models

- Deep auto-regressive (AR) models have no latent variables.
- Use the chain rule of probabilities to decompose:

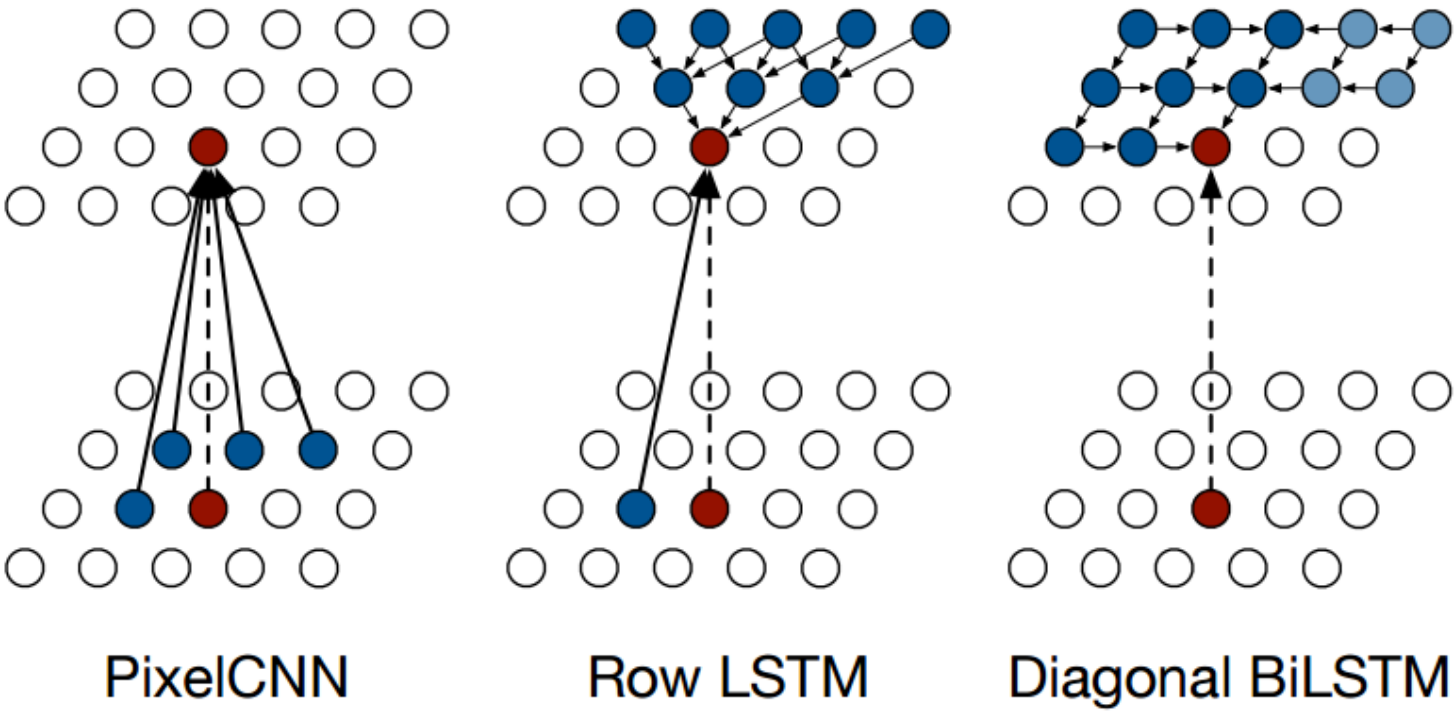
$$\mathbb{P}(\boldsymbol{x}) = \mathbb{P}(x_1)\,\mathbb{P}(x_2 \mid x_1) \cdots \mathbb{P}(x_D \mid x_1, \ldots, x_{D-1})$$



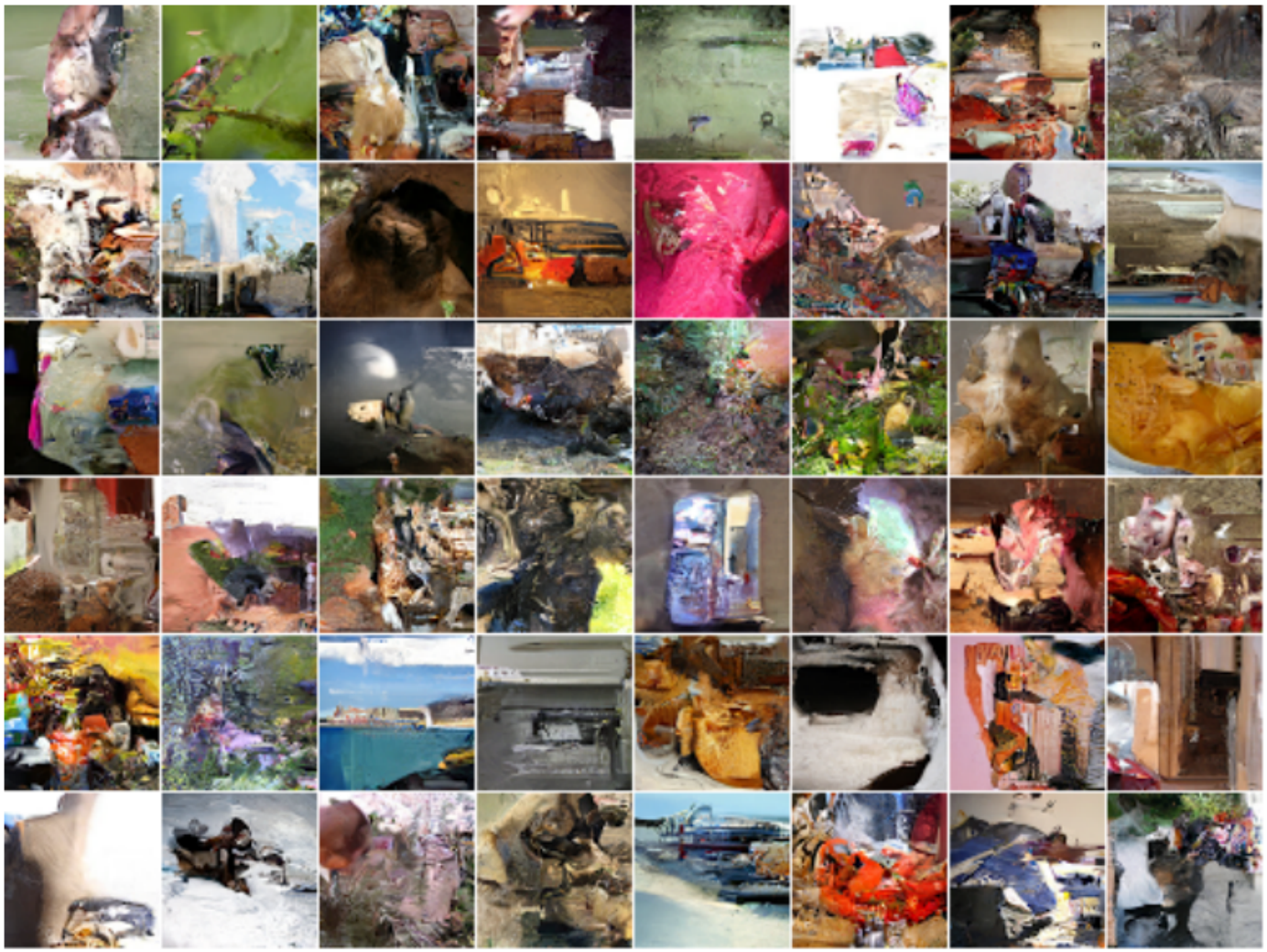
- Also called fully-visible Bayes networks.
- We saw examples already: RNNs, Pixel RNNs, Pixel CNNs, ...

Examples: PixelCNNs and PixelRNNs

- Input-to-state and state-to-state mappings for PixelCNN and two PixelRNN models (Oord et al., 2016):



RNNs for Generating Images



(Oord et al., 2016)

Summary

- Despite their simplicity, deep AR models can be very powerful.
- However, they may require too many parameters/complex functions due to the assumption all variables are observed.
- Models with latent variables are an appealing alternative: they can represent "clusters", yielding simpler representations.

Outline

1. Deep Auto-Regressive Models
2. Boltzmann Machines
3. Variational Auto-Encoders
   - Variational Inference and ELBO
   - Gradients and Reparameterization Trick
4. Generative Adversarial Networks
5. Conclusions

Energy Based Models

- A probability distribution (mixing observed and latent variables) via an energy function $E(\boldsymbol{x}, \boldsymbol{h}; \boldsymbol{\theta})$:

$$\mathbb{P}(\boldsymbol{x}, \boldsymbol{h}) = \frac{\exp(-E(\boldsymbol{x}, \boldsymbol{h}; \boldsymbol{\theta}))}{Z(\boldsymbol{\theta})}$$

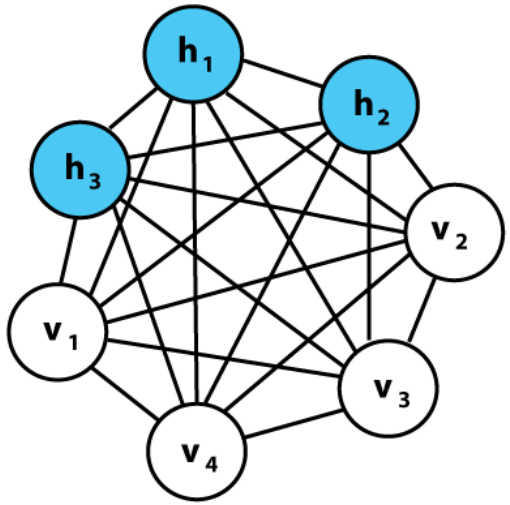
- Maximizing probability corresponds to minimizing the energy.

- Challenges:
  - Computing the partition function $Z(\boldsymbol{\theta})$
  - Computing the evidence $\mathbb{P}(\boldsymbol{x})$
  - Computing the posterior $\mathbb{P}(\boldsymbol{h} \mid \boldsymbol{x})$
  - Sampling from $\mathbb{P}(\boldsymbol{x}, \boldsymbol{h})$ or from $\mathbb{P}(\boldsymbol{x})$

Boltzmann Machine (Ackley et al., 1985)

- Energy-based model over binary vectors

- Some variables are observed ($\boldsymbol{v}$), others are latent/hidden ($\boldsymbol{h}$)



- Probability distribution over $(\boldsymbol{v}, \boldsymbol{h}) \in \{0,1\}^{N+M}$:

$$\mathbb{P}(\boldsymbol{v}, \boldsymbol{h}) = \frac{\exp(-E(\boldsymbol{v}, \boldsymbol{h}; \boldsymbol{\theta}))}{Z(\boldsymbol{\theta})}$$

- Energy function, with $\boldsymbol{\theta} = (\boldsymbol{R}, \boldsymbol{W}, \boldsymbol{S}, \boldsymbol{b}, \boldsymbol{c})$,

$$\begin{aligned} E(\boldsymbol{v}, \boldsymbol{h}; \boldsymbol{\theta}) &= -\begin{bmatrix} \boldsymbol{v}^\top & \boldsymbol{h}^\top \end{bmatrix} \begin{bmatrix} \boldsymbol{R} & \boldsymbol{W}/2 \\ \boldsymbol{W}^\top/2 & \boldsymbol{S} \end{bmatrix} \begin{bmatrix} \boldsymbol{v} \\ \boldsymbol{h} \end{bmatrix} - \begin{bmatrix} \boldsymbol{b}^\top & \boldsymbol{c}^\top \end{bmatrix} \begin{bmatrix} \boldsymbol{v} \\ \boldsymbol{h} \end{bmatrix} \\ &= -\boldsymbol{v}^\top \boldsymbol{R} \boldsymbol{v} - \boldsymbol{v}^\top \boldsymbol{W} \boldsymbol{h} - \boldsymbol{h}^\top \boldsymbol{S} \boldsymbol{h} - \boldsymbol{b}^\top \boldsymbol{v} - \boldsymbol{c}^\top \boldsymbol{h} \end{aligned}$$

Boltzmann Machine

- The Boltzmann machine (BM) is a universal approximator of probability mass functions over discrete variables (Le Roux and Bengio, 2008)

- Emulates the idea in Hebbian learning: "neurons that fire together wire together."

- However, in general,
  - Sampling is hard,
  - Inference is hard,
  - Learning is hard.

How to Learn a Boltzmann Machine?

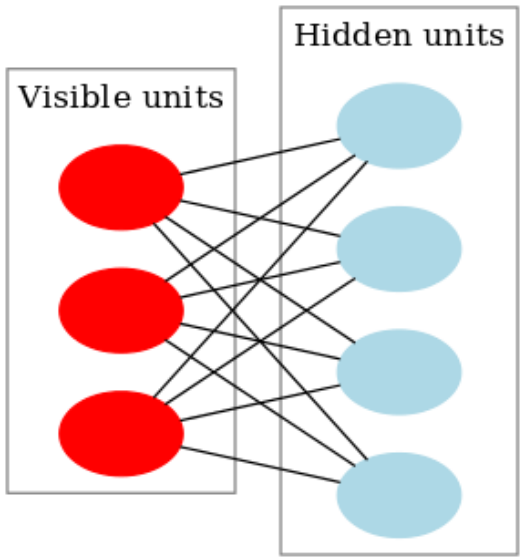
- Learning is usually based on maximum likelihood.
- The partition function $Z(\boldsymbol{\theta})$ is intractable: for learning, the gradient must be approximated:
  - contrastive divergence
  - pseudo-likelihood
  - noise-contrastive estimation
  - annealed importance sampling
- Not covered here, but check Goodfellow et al. (2016, Chapter 18).
- In a nutshell, learning a fully general BM is usually very challenging, so we typically use a particular version.

Some Particular Cases

- Restricted Boltzmann machines
- Deep belief networks
- Deep Boltzmann machines
- ...

Restricted Boltzmann Machines

- Also called harmonium (Smolensky, 1986)
- A layer of observable variables
- A single layer of latent variables.



- Bipartite graph, no intra-layer connections: $\boldsymbol{R} = 0$; $\boldsymbol{S} = 0$.
- The energy function becomes:

$$E(\boldsymbol{v}, \boldsymbol{h}; \boldsymbol{\theta}) = -\boldsymbol{v}^\top \boldsymbol{W}\boldsymbol{h} - \boldsymbol{b}^\top \boldsymbol{v} - \boldsymbol{c}^\top \boldsymbol{h}$$

- What is the advantage?

Restricted Boltzmann Machines

- Unfortunately, the partition function $Z(\boldsymbol{\theta})$ is still intractable
- ... but the conditionals $\mathbb{P}(\boldsymbol{h} \mid \boldsymbol{v})$ and $\mathbb{P}(\boldsymbol{v} \mid \boldsymbol{h})$ are now tractable!
  - easy to compute!
  - easy to sample!
  - using Markov-Chain Monte Carlo (MCMC) with Gibbs sampling.
- Why are these easy? Conditional independence (next slide)

Restricted Boltzmann Machines

- Why are the conditionals tractable?
- Because, without intra-layer connections, $h_1, \ldots, h_N$ are conditionally independent given $\boldsymbol{v}$:

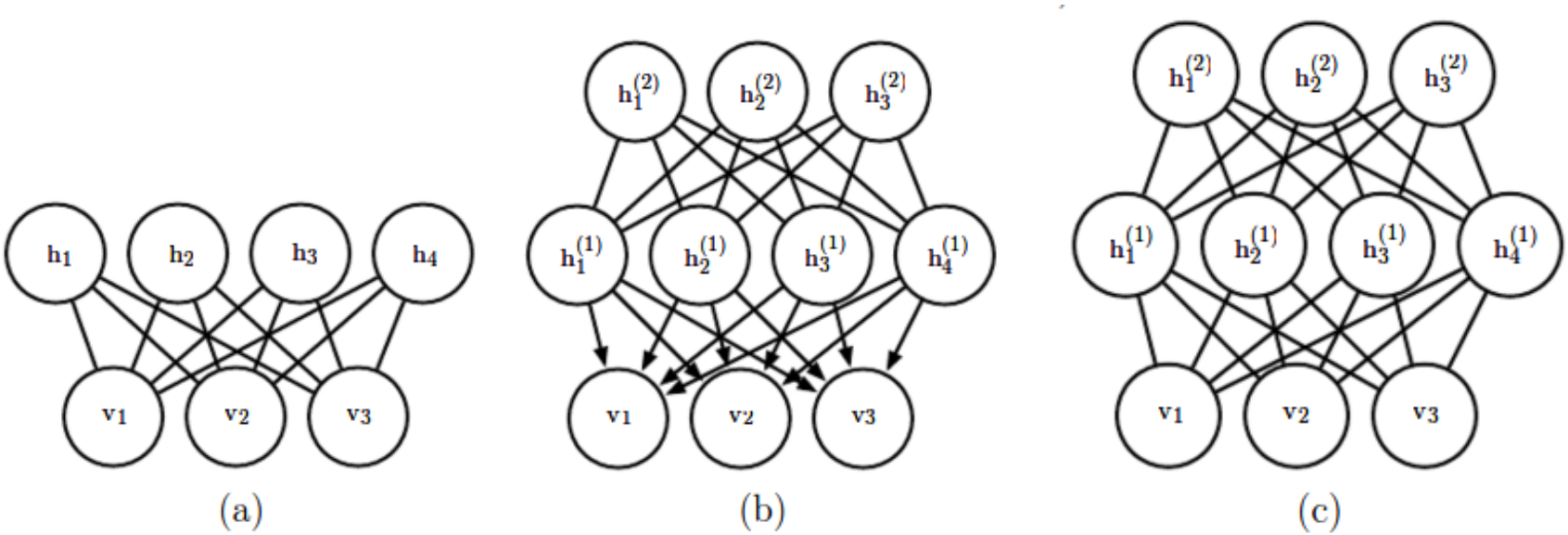
$$\mathbb{P}(\boldsymbol{h} \mid \boldsymbol{v}) = \prod_{j=1}^{M} \mathbb{P}(h_j \mid \boldsymbol{v})$$

where

$$\mathbb{P}(h_j = 1 \mid \boldsymbol{v}) \quad = \quad \sigma\big(c_j + (\boldsymbol{W}\boldsymbol{v})_j\big), \quad \forall j = 1, \ldots, M.$$

- Reciprocally for $\mathbb{P}(\boldsymbol{v} \mid \boldsymbol{h})$.
- RBMs are relatively easy to train by approximating $Z(\boldsymbol{\theta})$ (see Goodfellow et al. (2016, Chapter 18)).
- RBMs may be stacked to form deeper models.

Some RBM's Friends



(Image from Goodfellow et al. (2016))

(a) Restricted Boltzmann machine (RBM)

(b) Deep belief network (DBN): hybrid directed/undirected GM with multiple latent layers

(c) Deep Boltzmann machine (DBM): undirected GM with several layers of latent variables.

Examples of Deep Generative Models

- Auto-Regressive Networks
- Restricted Boltzmann Machines
- Deep Belief Networks
- Deep Boltzmann Machines
- Gaussian-Bernoulli RBMs
- Convolutional Boltzmann Machines
- Sigmoid Belief Nets
- Variational Auto-Encoders
- Generative Adversarial Networks
- Convolutional Generative Networks
- Generative Stochastic Networks

Examples of Deep Generative Models

- Auto-Regressive Networks ✓
- Restricted Boltzmann Machines ✓
- Deep Belief Networks
- Deep Boltzmann Machines
- Gaussian-Bernoulli RBMs
- Convolutional Boltzmann Machines
- Sigmoid Belief Nets
- Variational Auto-Encoders
- Generative Adversarial Networks
- Convolutional Generative Networks
- Generative Stochastic Networks

Next: Differentiable Generator Networks

- Several recent models are based on differentiable generator networks.
- This is a differentiable function $G(\boldsymbol{h}; \boldsymbol{\theta})$ that maps latent variables $\boldsymbol{h}$ into sample reconstructions $\boldsymbol{x}$ (or distributions $\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{h})$).
- This idea underlies
  - Variational auto-encoders (VAE)
  - Generative adversarial networks (GAN)
- We'll cover those next.

Outline

1. Deep Auto-Regressive Models
2. Boltzmann Machines
3. Variational Auto-Encoders
   - Variational Inference and ELBO
   - Gradients and Reparameterization Trick
4. Generative Adversarial Networks
5. Conclusions

Variational Auto-Encoders

- Many latent variable models have:
  - intractable evidence $\mathbb{P}(\boldsymbol{x})$
  - intractable posterior $\mathbb{P}(\boldsymbol{h} \mid \boldsymbol{x})$.
- Variational inference (e.g. mean field approximation) is a technique used to approximate these quantities.
- Widely used in Bayesian inference, topic models, etc...
- Auto-encoders: effective to learn data representations or codes (i.e. $\boldsymbol{x} \longrightarrow \boldsymbol{h} \longrightarrow \hat{\boldsymbol{x}}$)
- Key idea: combine auto-encoders with variational inference.

Assumptions

- Henceforth, we assume that:
  - the prior $\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h})$ is tractable (e.g.. zero-mean, unit-variance Gaussian)
  - the conditional $\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{h})$ is tractable (e.g. a feed-forward neural network or an RNN).
- However,
  - he evidence $\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x})$ (i.e. marginalizing out $\boldsymbol{h}$) is still intractable
  - computing the posterior $\mathbb{P}(\boldsymbol{h} \mid \boldsymbol{x})$ is still intractable.
- Next: using variational inference to approximate these computations

Outline

1. Deep Auto-Regressive Models
2. Boltzmann Machines
3. Variational Auto-Encoders
   - Variational Inference and ELBO
   - Gradients and Reparameterization Trick
4. Generative Adversarial Networks
5. Conclusions

Recap: Shannon's Entropy

- Let $\mathbb{P}$ be a distribution over $\mathcal{X}$. The entropy of $\mathbb{P}$ is

$$H(\mathbb{P}) = -\sum_{\boldsymbol{x} \in \mathcal{X}} \mathbb{P}(\boldsymbol{x}) \log \mathbb{P}(\boldsymbol{x})$$

Recap: Shannon's Entropy

- Let $\mathbb{P}$ be a distribution over $\mathcal{X}$. The entropy of $\mathbb{P}$ is

$$H(\mathbb{P}) = -\sum_{\boldsymbol{x}\in\mathcal{X}} \mathbb{P}(\boldsymbol{x}) \log \mathbb{P}(\boldsymbol{x}) = \mathbb{E}_{\mathbb{P}(\boldsymbol{x})}\big[-\log \mathbb{P}(\boldsymbol{x})\big]$$

Recap: Shannon's Entropy

- Let $\mathbb{P}$ be a distribution over $\mathcal{X}$. The entropy of $\mathbb{P}$ is

$$H(\mathbb{P}) = -\sum_{\boldsymbol{x}\in\mathcal{X}} \mathbb{P}(\boldsymbol{x}) \log \mathbb{P}(\boldsymbol{x}) = \mathbb{E}_{\mathbb{P}(\boldsymbol{x})}\big[-\log \mathbb{P}(\boldsymbol{x})\big]$$

- Always non-negative: $H(\mathbb{P}) \geq 0$
- $H(\mathbb{P}) = 0$ iff $\mathbb{P}(\boldsymbol{x}) = 1$, for some $\boldsymbol{x}$ and $\mathbb{P}(\boldsymbol{x}') = 0$, for any $\boldsymbol{x}' \neq \boldsymbol{x}$.

Recap: Shannon's Entropy

- Let $\mathbb{P}$ be a distribution over $\mathcal{X}$. The entropy of $\mathbb{P}$ is

$$H(\mathbb{P}) = -\sum_{\boldsymbol{x}\in\mathcal{X}} \mathbb{P}(\boldsymbol{x}) \log \mathbb{P}(\boldsymbol{x}) = \mathbb{E}_{\mathbb{P}(\boldsymbol{x})}\big[-\log \mathbb{P}(\boldsymbol{x})\big]$$

- Always non-negative: $H(\mathbb{P}) \geq 0$
- $H(\mathbb{P}) = 0$ iff $\mathbb{P}(\boldsymbol{x}) = 1$, for some $\boldsymbol{x}$ and $\mathbb{P}(\boldsymbol{x}') = 0$, for any $\boldsymbol{x}' \neq \boldsymbol{x}$.
- Upper bound: $H(\mathbb{P}) \leq \log |\mathcal{X}|$
- $H(\mathbb{P}) = \log |\mathcal{X}|$ iff $\mathbb{P}(\boldsymbol{x}) = 1/|\mathcal{X}|$ (uniform distribution)

Recap: Shannon's Entropy

- Let $\mathbb{P}$ be a distribution over $\mathcal{X}$. The entropy of $\mathbb{P}$ is

$$H(\mathbb{P}) = -\sum_{\boldsymbol{x}\in\mathcal{X}} \mathbb{P}(\boldsymbol{x}) \log \mathbb{P}(\boldsymbol{x}) = \mathbb{E}_{\mathbb{P}(\boldsymbol{x})}\big[-\log \mathbb{P}(\boldsymbol{x})\big]$$

- Always non-negative: $H(\mathbb{P}) \geq 0$
- $H(\mathbb{P}) = 0$ iff $\mathbb{P}(\boldsymbol{x}) = 1$, for some $\boldsymbol{x}$ and $\mathbb{P}(\boldsymbol{x}') = 0$, for any $\boldsymbol{x}' \neq \boldsymbol{x}$.
- Upper bound: $H(\mathbb{P}) \leq \log |\mathcal{X}|$
- $H(\mathbb{P}) = \log |\mathcal{X}|$ iff $\mathbb{P}(\boldsymbol{x}) = 1/|\mathcal{X}|$ (uniform distribution)
- Intuition: $H(\mathbb{P})$ measures how close to uniform the distribution is

Recap: Shannon's Entropy

- Let $\mathbb{P}$ be a distribution over $\mathcal{X}$. The entropy of $\mathbb{P}$ is

$$H(\mathbb{P}) = -\sum_{\boldsymbol{x}\in\mathcal{X}} \mathbb{P}(\boldsymbol{x}) \log \mathbb{P}(\boldsymbol{x}) = \mathbb{E}_{\mathbb{P}(\boldsymbol{x})}\big[-\log \mathbb{P}(\boldsymbol{x})\big]$$

- Always non-negative: $H(\mathbb{P}) \geq 0$
- $H(\mathbb{P}) = 0$ iff $\mathbb{P}(\boldsymbol{x}) = 1$, for some $\boldsymbol{x}$ and $\mathbb{P}(\boldsymbol{x}') = 0$, for any $\boldsymbol{x}' \neq \boldsymbol{x}$.
- Upper bound: $H(\mathbb{P}) \leq \log |\mathcal{X}|$
- $H(\mathbb{P}) = \log |\mathcal{X}|$ iff $\mathbb{P}(\boldsymbol{x}) = 1/|\mathcal{X}|$ (uniform distribution)
- Intuition: $H(\mathbb{P})$ measures how close to uniform the distribution is
- Coding perspective: expected number of bits (using $\log_2$) to optimally encode $\boldsymbol{x} \sim \mathbb{P}(\boldsymbol{x})$

Recap: Kullback-Leibler (KL) Divergence

- Let $\mathbb{P}$ and $\mathbb{Q}$ be two distributions over $\mathcal{X}$.

$$KL(\mathbb{P}\|\mathbb{Q}) = \sum_{\boldsymbol{x}} \mathbb{P}(\boldsymbol{x}) \log \frac{\mathbb{P}(\boldsymbol{x})}{\mathbb{Q}(\boldsymbol{x})}$$

Recap: Kullback-Leibler (KL) Divergence

- Let $\mathbb{P}$ and $\mathbb{Q}$ be two distributions over $\mathcal{X}$.

$$\begin{aligned} KL(\mathbb{P}\|\mathbb{Q}) &= \sum_{\boldsymbol{x}} \mathbb{P}(\boldsymbol{x}) \log \frac{\mathbb{P}(\boldsymbol{x})}{\mathbb{Q}(\boldsymbol{x})} \\ &= \mathbb{E}_{\mathbb{P}(\boldsymbol{x})}\big[-\log \mathbb{Q}(\boldsymbol{x})\big] + \mathbb{E}_{\mathbb{P}(\boldsymbol{x})}\big[\log \mathbb{P}(\boldsymbol{x})\big] \end{aligned}$$

Recap: Kullback-Leibler (KL) Divergence

- Let $\mathbb{P}$ and $\mathbb{Q}$ be two distributions over $\mathcal{X}$.

$$\begin{aligned} KL(\mathbb{P}\|\mathbb{Q}) &= \sum_{\boldsymbol{x}} \mathbb{P}(\boldsymbol{x}) \log \frac{\mathbb{P}(\boldsymbol{x})}{\mathbb{Q}(\boldsymbol{x})} \\ &= \mathbb{E}_{\mathbb{P}(\boldsymbol{x})}\big[-\log \mathbb{Q}(\boldsymbol{x})\big] + \mathbb{E}_{\mathbb{P}(\boldsymbol{x})}\big[\log \mathbb{P}(\boldsymbol{x})\big] \\ &= \mathbb{E}_{\mathbb{P}(\boldsymbol{x})}\big[-\log \mathbb{Q}(\boldsymbol{x})\big] - H(\mathbb{P}) \end{aligned}$$

Recap: Kullback-Leibler (KL) Divergence

- Let $\mathbb{P}$ and $\mathbb{Q}$ be two distributions over $\mathcal{X}$.

$$\begin{aligned} KL(\mathbb{P}\|\mathbb{Q}) &= \sum_{\boldsymbol{x}} \mathbb{P}(\boldsymbol{x}) \log \frac{\mathbb{P}(\boldsymbol{x})}{\mathbb{Q}(\boldsymbol{x})} \\ &= \mathbb{E}_{\mathbb{P}(\boldsymbol{x})}\big[-\log \mathbb{Q}(\boldsymbol{x})\big] + \mathbb{E}_{\mathbb{P}(\boldsymbol{x})}\big[\log \mathbb{P}(\boldsymbol{x})\big] \\ &= \mathbb{E}_{\mathbb{P}(\boldsymbol{x})}\big[-\log \mathbb{Q}(\boldsymbol{x})\big] - H(\mathbb{P}) \end{aligned}$$

- Always non-negative: $KL(\mathbb{P}\|\mathbb{Q}) \geq 0$

Recap: Kullback-Leibler (KL) Divergence

- Let $\mathbb{P}$ and $\mathbb{Q}$ be two distributions over $\mathcal{X}$.

$$\begin{aligned} KL(\mathbb{P}\|\mathbb{Q}) &= \sum_{\boldsymbol{x}} \mathbb{P}(\boldsymbol{x}) \log \frac{\mathbb{P}(\boldsymbol{x})}{\mathbb{Q}(\boldsymbol{x})} \\ &= \mathbb{E}_{\mathbb{P}(\boldsymbol{x})}\big[-\log \mathbb{Q}(\boldsymbol{x})\big] + \mathbb{E}_{\mathbb{P}(\boldsymbol{x})}\big[\log \mathbb{P}(\boldsymbol{x})\big] \\ &= \mathbb{E}_{\mathbb{P}(\boldsymbol{x})}\big[-\log \mathbb{Q}(\boldsymbol{x})\big] - H(\mathbb{P}) \end{aligned}$$

- Always non-negative: $KL(\mathbb{P}\|\mathbb{Q}) \geq 0$
- $KL(\mathbb{P}\|\mathbb{Q}) = 0$ iff $\mathbb{P}(\boldsymbol{x}) = \mathbb{Q}(\boldsymbol{x})$, for all $\boldsymbol{x} \in \mathcal{X}$

Recap: Kullback-Leibler (KL) Divergence

- Let $\mathbb{P}$ and $\mathbb{Q}$ be two distributions over $\mathcal{X}$.

$$\begin{aligned} KL(\mathbb{P}\|\mathbb{Q}) &= \sum_{\boldsymbol{x}} \mathbb{P}(\boldsymbol{x}) \log \frac{\mathbb{P}(\boldsymbol{x})}{\mathbb{Q}(\boldsymbol{x})} \\ &= \mathbb{E}_{\mathbb{P}(\boldsymbol{x})}\big[-\log \mathbb{Q}(\boldsymbol{x})\big] + \mathbb{E}_{\mathbb{P}(\boldsymbol{x})}\big[\log \mathbb{P}(\boldsymbol{x})\big] \\ &= \mathbb{E}_{\mathbb{P}(\boldsymbol{x})}\big[-\log \mathbb{Q}(\boldsymbol{x})\big] - H(\mathbb{P}) \end{aligned}$$

- Always non-negative: $KL(\mathbb{P}\|\mathbb{Q}) \geq 0$
- $KL(\mathbb{P}\|\mathbb{Q}) = 0$ iff $\mathbb{P}(\boldsymbol{x}) = \mathbb{Q}(\boldsymbol{x})$, for all $\boldsymbol{x} \in \mathcal{X}$
- Not symmetric: in general, $KL(\mathbb{P}\|\mathbb{Q}) \neq KL(\mathbb{Q}\|\mathbb{P})$

Recap: Kullback-Leibler (KL) Divergence

- Let $\mathbb{P}$ and $\mathbb{Q}$ be two distributions over $\mathcal{X}$.

$$\begin{aligned} KL(\mathbb{P}\|\mathbb{Q}) &= \sum_{\boldsymbol{x}} \mathbb{P}(\boldsymbol{x}) \log \frac{\mathbb{P}(\boldsymbol{x})}{\mathbb{Q}(\boldsymbol{x})} \\ &= \mathbb{E}_{\mathbb{P}(\boldsymbol{x})}\big[-\log \mathbb{Q}(\boldsymbol{x})\big] + \mathbb{E}_{\mathbb{P}(\boldsymbol{x})}\big[\log \mathbb{P}(\boldsymbol{x})\big] \\ &= \mathbb{E}_{\mathbb{P}(\boldsymbol{x})}\big[-\log \mathbb{Q}(\boldsymbol{x})\big] - H(\mathbb{P}) \end{aligned}$$

- Always non-negative: $KL(\mathbb{P}\|\mathbb{Q}) \geq 0$
- $KL(\mathbb{P}\|\mathbb{Q}) = 0$ iff $\mathbb{P}(\boldsymbol{x}) = \mathbb{Q}(\boldsymbol{x})$, for all $\boldsymbol{x} \in \mathcal{X}$
- Not symmetric: in general, $KL(\mathbb{P}\|\mathbb{Q}) \neq KL(\mathbb{Q}\|\mathbb{P})$
- Intuition: $KL(\mathbb{P}\|\mathbb{Q})$ measure how different $\mathbb{Q}$ is from $\mathbb{P}$

Recap: Kullback-Leibler (KL) Divergence

- Let $\mathbb{P}$ and $\mathbb{Q}$ be two distributions over $\mathcal{X}$.

$$\begin{aligned} KL(\mathbb{P}\|\mathbb{Q}) &= \sum_{\boldsymbol{x}} \mathbb{P}(\boldsymbol{x}) \log \frac{\mathbb{P}(\boldsymbol{x})}{\mathbb{Q}(\boldsymbol{x})} \\ &= \mathbb{E}_{\mathbb{P}(\boldsymbol{x})}\big[-\log \mathbb{Q}(\boldsymbol{x})\big] + \mathbb{E}_{\mathbb{P}(\boldsymbol{x})}\big[\log \mathbb{P}(\boldsymbol{x})\big] \\ &= \mathbb{E}_{\mathbb{P}(\boldsymbol{x})}\big[-\log \mathbb{Q}(\boldsymbol{x})\big] - H(\mathbb{P}) \end{aligned}$$

- Always non-negative: $KL(\mathbb{P}\|\mathbb{Q}) \geq 0$
- $KL(\mathbb{P}\|\mathbb{Q}) = 0$ iff $\mathbb{P}(\boldsymbol{x}) = \mathbb{Q}(\boldsymbol{x})$, for all $\boldsymbol{x} \in \mathcal{X}$
- Not symmetric: in general, $KL(\mathbb{P}\|\mathbb{Q}) \neq KL(\mathbb{Q}\|\mathbb{P})$
- Intuition: $KL(\mathbb{P}\|\mathbb{Q})$ measure how different $\mathbb{Q}$ is from $\mathbb{P}$
- Coding perspective: expected number of extra bits needed to encode $\boldsymbol{x} \sim \mathbb{P}(\boldsymbol{x})$ using a code that is optimal for $\mathbb{Q}(\boldsymbol{x})$.

Recap: Entropy and KL Divergence in the Continuous Case

- Let $\mathbb{P}$ be a probability density over $\mathcal{X}$. The differential entropy of $\mathbb{P}$ is

$$H(\mathbb{P}) = -\int_{\mathcal{X}} \mathbb{P}(\boldsymbol{x}) \log \mathbb{P}(\boldsymbol{x}) \, d\boldsymbol{x}$$

...no longer guaranteed to be non-negative.

Recap: Entropy and KL Divergence in the Continuous Case

- Let $\mathbb{P}$ be a probability density over $\mathcal{X}$. The differential entropy of $\mathbb{P}$ is

$$H(\mathbb{P}) = -\int_{\mathcal{X}} \mathbb{P}(\boldsymbol{x}) \log \mathbb{P}(\boldsymbol{x})\, d\boldsymbol{x} = \mathbb{E}_{\mathbb{P}(\boldsymbol{x})}\big[-\log \mathbb{P}(\boldsymbol{x})\big]$$

...no longer guaranteed to be non-negative.

Recap: Entropy and KL Divergence in the Continuous Case

- Let $\mathbb{P}$ be a probability density over $\mathcal{X}$. The differential entropy of $\mathbb{P}$ is

$$H(\mathbb{P}) = -\int_{\mathcal{X}} \mathbb{P}(\boldsymbol{x}) \log \mathbb{P}(\boldsymbol{x})\, d\boldsymbol{x} = \mathbb{E}_{\mathbb{P}(\boldsymbol{x})}\big[-\log \mathbb{P}(\boldsymbol{x})\big]$$

  ...no longer guaranteed to be non-negative.

- Let $\mathbb{P}$ and $\mathbb{Q}$ be two probability densities over $\mathcal{X}$.

$$\begin{aligned} KL(\mathbb{P}\|\mathbb{Q}) &= \int_{\mathcal{X}} \mathbb{P}(\boldsymbol{x}) \log \frac{\mathbb{P}(\boldsymbol{x})}{\mathbb{Q}(\boldsymbol{x})} d\boldsymbol{x} \\ &= \mathbb{E}_{\mathbb{P}(\boldsymbol{x})}\big[-\log \mathbb{Q}(\boldsymbol{x})\big] + \mathbb{E}_{\mathbb{P}(\boldsymbol{x})}\big[\log \mathbb{P}(\boldsymbol{x})\big] \\ &= \mathbb{E}_{\mathbb{P}(\boldsymbol{x})}\big[-\log \mathbb{Q}(\boldsymbol{x})\big] - H(\mathbb{P}) \end{aligned}$$

Recap: Entropy and KL Divergence in the Continuous Case

- Let $\mathbb{P}$ be a probability density over $\mathcal{X}$. The differential entropy of $\mathbb{P}$ is

$$H(\mathbb{P}) = -\int_{\mathcal{X}} \mathbb{P}(\boldsymbol{x}) \log \mathbb{P}(\boldsymbol{x})\, d\boldsymbol{x} = \mathbb{E}_{\mathbb{P}(\boldsymbol{x})}\big[-\log \mathbb{P}(\boldsymbol{x})\big]$$

  ...no longer guaranteed to be non-negative.

- Let $\mathbb{P}$ and $\mathbb{Q}$ be two probability densities over $\mathcal{X}$.

$$\begin{aligned} KL(\mathbb{P}\|\mathbb{Q}) &= \int_{\mathcal{X}} \mathbb{P}(\boldsymbol{x}) \log \frac{\mathbb{P}(\boldsymbol{x})}{\mathbb{Q}(\boldsymbol{x})} d\boldsymbol{x} \\ &= \mathbb{E}_{\mathbb{P}(\boldsymbol{x})}\big[-\log \mathbb{Q}(\boldsymbol{x})\big] + \mathbb{E}_{\mathbb{P}(\boldsymbol{x})}\big[\log \mathbb{P}(\boldsymbol{x})\big] \\ &= \mathbb{E}_{\mathbb{P}(\boldsymbol{x})}\big[-\log \mathbb{Q}(\boldsymbol{x})\big] - H(\mathbb{P}) \end{aligned}$$

- Also, always non-negative: $KL(\mathbb{P}\|\mathbb{Q}) \geq 0$

Recap: Entropy and KL Divergence in the Continuous Case

- Let $\mathbb{P}$ be a probability density over $\mathcal{X}$. The differential entropy of $\mathbb{P}$ is

$$H(\mathbb{P}) = -\int_{\mathcal{X}} \mathbb{P}(\boldsymbol{x}) \log \mathbb{P}(\boldsymbol{x})\, d\boldsymbol{x} = \mathbb{E}_{\mathbb{P}(\boldsymbol{x})}\big[-\log \mathbb{P}(\boldsymbol{x})\big]$$

  ...no longer guaranteed to be non-negative.

- Let $\mathbb{P}$ and $\mathbb{Q}$ be two probability densities over $\mathcal{X}$.

$$\begin{aligned} KL(\mathbb{P}\|\mathbb{Q}) &= \int_{\mathcal{X}} \mathbb{P}(\boldsymbol{x}) \log \frac{\mathbb{P}(\boldsymbol{x})}{\mathbb{Q}(\boldsymbol{x})} d\boldsymbol{x} \\ &= \mathbb{E}_{\mathbb{P}(\boldsymbol{x})}\big[-\log \mathbb{Q}(\boldsymbol{x})\big] + \mathbb{E}_{\mathbb{P}(\boldsymbol{x})}\big[\log \mathbb{P}(\boldsymbol{x})\big] \\ &= \mathbb{E}_{\mathbb{P}(\boldsymbol{x})}\big[-\log \mathbb{Q}(\boldsymbol{x})\big] - H(\mathbb{P}) \end{aligned}$$

- Also, always non-negative: $KL(\mathbb{P}\|\mathbb{Q}) \geq 0$
- $KL(\mathbb{P}\|\mathbb{Q}) = 0$ iff $\mathbb{P}(\boldsymbol{x}) = \mathbb{Q}(\boldsymbol{x})$, almost everywhere

Recap: Entropy and KL Divergence in the Continuous Case

- Let $\mathbb{P}$ be a probability density over $\mathcal{X}$. The differential entropy of $\mathbb{P}$ is

$$H(\mathbb{P}) = -\int_{\mathcal{X}} \mathbb{P}(\boldsymbol{x}) \log \mathbb{P}(\boldsymbol{x})\, d\boldsymbol{x} = \mathbb{E}_{\mathbb{P}(\boldsymbol{x})}\big[-\log \mathbb{P}(\boldsymbol{x})\big]$$

  ...no longer guaranteed to be non-negative.

- Let $\mathbb{P}$ and $\mathbb{Q}$ be two probability densities over $\mathcal{X}$.

$$\begin{aligned} KL(\mathbb{P}\|\mathbb{Q}) &= \int_{\mathcal{X}} \mathbb{P}(\boldsymbol{x}) \log \frac{\mathbb{P}(\boldsymbol{x})}{\mathbb{Q}(\boldsymbol{x})} d\boldsymbol{x} \\ &= \mathbb{E}_{\mathbb{P}(\boldsymbol{x})}\big[-\log \mathbb{Q}(\boldsymbol{x})\big] + \mathbb{E}_{\mathbb{P}(\boldsymbol{x})}\big[\log \mathbb{P}(\boldsymbol{x})\big] \\ &= \mathbb{E}_{\mathbb{P}(\boldsymbol{x})}\big[-\log \mathbb{Q}(\boldsymbol{x})\big] - H(\mathbb{P}) \end{aligned}$$

- Also, always non-negative: $KL(\mathbb{P}\|\mathbb{Q}) \geq 0$
- $KL(\mathbb{P}\|\mathbb{Q}) = 0$ iff $\mathbb{P}(\boldsymbol{x}) = \mathbb{Q}(\boldsymbol{x})$, almost everywhere
- Intuition: $KL(\mathbb{P}\|\mathbb{Q})$ measure how different $\mathbb{Q}$ is from $\mathbb{P}$

Evidence Lower Bound (ELBO)

- ELBO is a central concept in variational inference.
- True posterior and evidence: $\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h} \mid \boldsymbol{x})$ and $\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x})$

Evidence Lower Bound (ELBO)

- ELBO is a central concept in variational inference.
- True posterior and evidence: $\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h} \mid \boldsymbol{x})$ and $\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x})$
- For any distribution $\mathbb{Q}(\boldsymbol{h})$,

$$\begin{aligned} 0 &\geq -KL(\mathbb{Q}(\boldsymbol{h}) \| \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h} \mid \boldsymbol{x})) \\ &= \mathbb{E}_{\mathbb{Q}(\boldsymbol{h})}[\log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h} \mid \boldsymbol{x})] - \overbrace{\mathbb{E}_{\mathbb{Q}(\boldsymbol{h})}[\log \mathbb{Q}(\boldsymbol{h})]}^{H(\mathbb{Q})} \end{aligned}$$

Evidence Lower Bound (ELBO)

- ELBO is a central concept in variational inference.
- True posterior and evidence: $\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h} \mid \boldsymbol{x})$ and $\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x})$
- For any distribution $\mathbb{Q}(\boldsymbol{h})$,

$$
\begin{aligned}
0 \quad &\geq \quad -KL(\mathbb{Q}(\boldsymbol{h}) \| \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h} \mid \boldsymbol{x})) \\
&= \quad \mathbb{E}_{\mathbb{Q}(\boldsymbol{h})}[\log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h} \mid \boldsymbol{x})] - \overbrace{\mathbb{E}_{\mathbb{Q}(\boldsymbol{h})}[\log \mathbb{Q}(\boldsymbol{h})]}^{H(\mathbb{Q})} \\
&\overset{(a)}{=} \quad \underbrace{\mathbb{E}_{\mathbb{Q}(\boldsymbol{h})}[\log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{h})] - \mathbb{E}_{\mathbb{Q}(\boldsymbol{h})}[\log \mathbb{Q}(\boldsymbol{h})]}_{\text{ELBO}(\mathbb{Q})} - \log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x}).
\end{aligned}
$$

where (a) uses Bayes law: $\log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h} \mid \boldsymbol{x}) = \log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{h}) - \log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x})$

Evidence Lower Bound (ELBO)

- ELBO is a central concept in variational inference.
- True posterior and evidence: $\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h} \mid \boldsymbol{x})$ and $\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x})$
- For any distribution $\mathbb{Q}(\boldsymbol{h})$,

$$\begin{aligned} 0 &\geq -KL(\mathbb{Q}(\boldsymbol{h}) \| \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h} \mid \boldsymbol{x})) \\ &= \mathbb{E}_{\mathbb{Q}(\boldsymbol{h})}[\log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h} \mid \boldsymbol{x})] - \overbrace{\mathbb{E}_{\mathbb{Q}(\boldsymbol{h})}[\log \mathbb{Q}(\boldsymbol{h})]}^{H(\mathbb{Q})} \\ &\overset{(a)}{=} \underbrace{\mathbb{E}_{\mathbb{Q}(\boldsymbol{h})}[\log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{h})] - \mathbb{E}_{\mathbb{Q}(\boldsymbol{h})}[\log \mathbb{Q}(\boldsymbol{h})]}_{\text{ELBO}(\mathbb{Q})} - \log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x}). \end{aligned}$$

  where (a) uses Bayes law: $\log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h} \mid \boldsymbol{x}) = \log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{h}) - \log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x})$

- Moving $\log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x})$ to the l.h.s.,

$$\log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x}) = \text{ELBO}(\mathbb{Q}) + \overbrace{KL(\mathbb{Q}(\boldsymbol{h}) \| \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h} \mid \boldsymbol{x}))}^{\geq 0}$$

Evidence Lower Bound (ELBO)

- ELBO is a central concept in variational inference.
- True posterior and evidence: $\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h} \mid \boldsymbol{x})$ and $\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x})$
- For any distribution $\mathbb{Q}(\boldsymbol{h})$,

$$
\begin{aligned}
0 \quad &\geq \quad -KL(\mathbb{Q}(\boldsymbol{h}) \| \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h} \mid \boldsymbol{x})) \\
&= \quad \mathbb{E}_{\mathbb{Q}(\boldsymbol{h})}[\log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h} \mid \boldsymbol{x})] - \overbrace{\mathbb{E}_{\mathbb{Q}(\boldsymbol{h})}[\log \mathbb{Q}(\boldsymbol{h})]}^{H(\mathbb{Q})} \\
&\overset{(a)}{=} \quad \underbrace{\mathbb{E}_{\mathbb{Q}(\boldsymbol{h})}[\log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{h})] - \mathbb{E}_{\mathbb{Q}(\boldsymbol{h})}[\log \mathbb{Q}(\boldsymbol{h})]}_{\text{ELBO}(\mathbb{Q})} - \log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x}).
\end{aligned}
$$

  where (a) uses Bayes law: $\log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h} \mid \boldsymbol{x}) = \log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{h}) - \log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x})$

- Moving $\log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x})$ to the l.h.s.,

$$
\log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x}) = \text{ELBO}(\mathbb{Q}) + \overbrace{KL(\mathbb{Q}(\boldsymbol{h}) \| \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h} \mid \boldsymbol{x}))}^{\geq 0} \geq \text{ELBO}(\mathbb{Q})
$$

Variational Inference

- Evidence lower bound (ELBO):

$$\log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x}) = \text{ELBO}(\mathbb{Q}) + KL(\mathbb{Q}(\boldsymbol{h}) \| \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h} \mid \boldsymbol{x})) \geq \text{ELBO}(\mathbb{Q}).$$

Variational Inference

- Evidence lower bound (ELBO):

$$\log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x}) = \text{ELBO}(\mathbb{Q}) + KL(\mathbb{Q}(\boldsymbol{h}) \| \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h} \mid \boldsymbol{x})) \geq \text{ELBO}(\mathbb{Q}).$$

- Equality achieved for $\mathbb{Q}(\boldsymbol{h}) = \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h} \mid \boldsymbol{x})$, but intractable in general

Variational Inference

- Evidence lower bound (ELBO):

$$\log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x}) = \text{ELBO}(\mathbb{Q}) + KL(\mathbb{Q}(\boldsymbol{h}) \| \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h} \mid \boldsymbol{x})) \geq \text{ELBO}(\mathbb{Q}).$$

- Equality achieved for $\mathbb{Q}(\boldsymbol{h}) = \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h} \mid \boldsymbol{x})$, but intractable in general
- Key idea:
  1. constrain $\mathbb{Q}(\boldsymbol{h})$ to a chosen tractable family;
  2. look for the $\mathbb{Q}(\boldsymbol{h})$ in this family that maximizes the ELBO.

Variational Inference

- Evidence lower bound (ELBO):

$$\log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x}) = \text{ELBO}(\mathbb{Q}) + KL(\mathbb{Q}(\boldsymbol{h}) \| \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h} \mid \boldsymbol{x})) \geq \text{ELBO}(\mathbb{Q}).$$

- Equality achieved for $\mathbb{Q}(\boldsymbol{h}) = \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h} \mid \boldsymbol{x})$, but intractable in general
- Key idea:
  1. constrain $\mathbb{Q}(\boldsymbol{h})$ to a chosen tractable family;
  2. look for the $\mathbb{Q}(\boldsymbol{h})$ in this family that maximizes the ELBO.
- Since $\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x})$ fixed,

$$\text{maximizing ELBO}(\mathbb{Q}) \quad \Leftrightarrow \quad \text{minimizing} \quad KL(\mathbb{Q}(\boldsymbol{h}) \| \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h} \mid \boldsymbol{x}))$$

Variational Inference

- Evidence lower bound (ELBO):

$$\log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x}) = \text{ELBO}(\mathbb{Q}) + KL(\mathbb{Q}(\boldsymbol{h}) \| \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h} \mid \boldsymbol{x})) \geq \text{ELBO}(\mathbb{Q}).$$

- Equality achieved for $\mathbb{Q}(\boldsymbol{h}) = \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h} \mid \boldsymbol{x})$, but intractable in general
- Key idea:
  1. constrain $\mathbb{Q}(\boldsymbol{h})$ to a chosen tractable family;
  2. look for the $\mathbb{Q}(\boldsymbol{h})$ in this family that maximizes the ELBO.
- Since $\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x})$ fixed,

$$\text{maximizing ELBO}(\mathbb{Q}) \quad \Leftrightarrow \quad \text{minimizing } KL(\mathbb{Q}(\boldsymbol{h}) \| \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h} \mid \boldsymbol{x}))$$

- Old roots in calculus of variations (Newton, Bernoulli, Euler, Lagrange, ...)

Evidence Lower Bound

- The ELBO can be written in different ways:

$$\begin{aligned}
\text{ELBO}(\mathbb{Q}) &= \mathbb{E}_{\mathbb{Q}(\boldsymbol{h})}[\log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{h})] - \mathbb{E}_{\mathbb{Q}(\boldsymbol{h})}[\log \mathbb{Q}(\boldsymbol{h})] \\
&= \mathbb{E}_{\mathbb{Q}(\boldsymbol{h})}[\log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{h})] - \mathbb{E}_{\mathbb{Q}(\boldsymbol{h})}\left[\log \frac{\mathbb{Q}(\boldsymbol{h})}{\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h})}\right] \\
&= \mathbb{E}_{\mathbb{Q}(\boldsymbol{h})}[\log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{h})] - KL(\mathbb{Q}(\boldsymbol{h}) \| \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h})).
\end{aligned}$$

Evidence Lower Bound

- The ELBO can be written in different ways:

$$\begin{aligned}
\text{ELBO}(\mathbb{Q}) &= \mathbb{E}_{\mathbb{Q}(\boldsymbol{h})}[\log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{h})] - \mathbb{E}_{\mathbb{Q}(\boldsymbol{h})}[\log \mathbb{Q}(\boldsymbol{h})] \\
&= \mathbb{E}_{\mathbb{Q}(\boldsymbol{h})}[\log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{h})] - \mathbb{E}_{\mathbb{Q}(\boldsymbol{h})}\left[\log \frac{\mathbb{Q}(\boldsymbol{h})}{\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h})}\right] \\
&= \mathbb{E}_{\mathbb{Q}(\boldsymbol{h})}[\log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{h})] - KL(\mathbb{Q}(\boldsymbol{h}) \| \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h})).
\end{aligned}$$

- Which values of $\boldsymbol{h}$ is $\mathbb{Q}(\boldsymbol{h})$ encouraged to place its mass on?
  - First term: expected likelihood: encourages placing mass on latent variables $\boldsymbol{h}$ that explain the observed data $\boldsymbol{x}$.
  - Second term: negative KLD between $\mathbb{Q}(\boldsymbol{h})$ and the prior: encourages staying close to the prior.

Evidence Lower Bound

- The ELBO can be written in different ways:

$$\begin{aligned}
\text{ELBO}(\mathbb{Q}) &= \mathbb{E}_{\mathbb{Q}(\boldsymbol{h})}[\log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{h})] - \mathbb{E}_{\mathbb{Q}(\boldsymbol{h})}[\log \mathbb{Q}(\boldsymbol{h})] \\
&= \mathbb{E}_{\mathbb{Q}(\boldsymbol{h})}[\log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{h})] - \mathbb{E}_{\mathbb{Q}(\boldsymbol{h})}\left[\log \frac{\mathbb{Q}(\boldsymbol{h})}{\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h})}\right] \\
&= \mathbb{E}_{\mathbb{Q}(\boldsymbol{h})}[\log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{h})] - KL(\mathbb{Q}(\boldsymbol{h}) \| \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h})).
\end{aligned}$$

- Which values of $\boldsymbol{h}$ is $\mathbb{Q}(\boldsymbol{h})$ encouraged to place its mass on?
  - First term: expected likelihood: encourages placing mass on latent variables $\boldsymbol{h}$ that explain the observed data $\boldsymbol{x}$.
  - Second term: negative KLD between $\mathbb{Q}(\boldsymbol{h})$ and the prior: encourages staying close to the prior.
- The ELBO mirrors the usual trade-off between **likelihood** and **prior**.

Mean Field Approximation

- Which tractable family to use for $\mathbb{Q}(\boldsymbol{h})$?

Mean Field Approximation

- Which tractable family to use for $\mathbb{Q}(\boldsymbol{h})$?

- Mean field approximation (MFA):

$$\mathbb{Q}(\boldsymbol{h}) = \prod_i \mathbb{Q}(h_i).$$

  i.e., model the h_i are independent.

Mean Field Approximation

- Which tractable family to use for $\mathbb{Q}(\boldsymbol{h})$?

- Mean field approximation (MFA):

$$\mathbb{Q}(\boldsymbol{h}) = \prod_i \mathbb{Q}(h_i).$$

  i.e., model the h_i are independent.

- More sophisticated: structured mean field imposes a graphical model on $\mathbb{Q}$ capturing (some) interactions among the h_i, but still tractable. (see the book by Wainwright and Jordan (2008) for details).

Amortized Variational Inference

- As seen so far, the variational distribution $\mathbb{Q}(\boldsymbol{h})$ has to be optimized for every example $\boldsymbol{x}$.

Amortized Variational Inference

- As seen so far, the variational distribution $\mathbb{Q}(\boldsymbol{h})$ has to be optimized for every example $\boldsymbol{x}$.

- This can be expensive: requires several gradient/coordinate ascent iterations per example.

Amortized Variational Inference

- As seen so far, the variational distribution $\mathbb{Q}(\boldsymbol{h})$ has to be optimized for every example $\boldsymbol{x}$.

- This can be expensive: requires several gradient/coordinate ascent iterations per example.

- Alternative: use amortized variational inference!

- Key idea: instead of optimizing $\mathbb{Q}(\boldsymbol{h})$ for every example, use an encoder with shared parameters $\boldsymbol{\phi}$ and define $\mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h} \mid \boldsymbol{x})$.

  For each example:

  - make a forward pass on the encoder to obtain $\mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h} \mid \boldsymbol{x})$
  - backpropagate through the encoder to update $\boldsymbol{\phi}$.

Example: Multivariate Bernoulli with Continuous Latent Variables (Kingma and Welling, 2013)

- Prior: multivariate isotropic Gaussian $\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h}) = \mathcal{N}(\boldsymbol{h}; 0, \boldsymbol{I})$

Example: Multivariate Bernoulli with Continuous Latent Variables (Kingma and Welling, 2013)

- Prior: multivariate isotropic Gaussian $\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h}) = \mathcal{N}(\boldsymbol{h}; 0, \boldsymbol{I})$
- Conditional: multivariate Bernoulli

$$\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{h}) = \prod_{i=1}^{D} \sigma(f_i(\boldsymbol{h}; \boldsymbol{\theta}))^{x_i} (1 - \sigma(f_i(\boldsymbol{h}; \boldsymbol{\theta})))^{1-x_i},$$

where $\boldsymbol{f}(\boldsymbol{h}; \boldsymbol{\theta})$ is an MLP, with parameters $\boldsymbol{\theta}$ and input $\boldsymbol{h}$

Example: Multivariate Bernoulli with Continuous Latent Variables (Kingma and Welling, 2013)

- Prior: multivariate isotropic Gaussian $\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h}) = \mathcal{N}(\boldsymbol{h}; 0, \boldsymbol{I})$
- Conditional: multivariate Bernoulli

$$\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{h}) = \prod_{i=1}^{D} \sigma(f_i(\boldsymbol{h}; \boldsymbol{\theta}))^{x_i} (1 - \sigma(f_i(\boldsymbol{h}; \boldsymbol{\theta})))^{1-x_i},$$

  where $\boldsymbol{f}(\boldsymbol{h}; \boldsymbol{\theta})$ is an MLP, with parameters $\boldsymbol{\theta}$ and input $\boldsymbol{h}$

- The true posterior $\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h} \mid \boldsymbol{x})$ is intractable

Example: Multivariate Bernoulli with Continuous Latent Variables (Kingma and Welling, 2013)

- Prior: multivariate isotropic Gaussian $\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h}) = \mathcal{N}(\boldsymbol{h}; 0, \boldsymbol{I})$
- Conditional: multivariate Bernoulli

$$\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{h}) = \prod_{i=1}^{D} \sigma(f_i(\boldsymbol{h}; \boldsymbol{\theta}))^{x_i} (1 - \sigma(f_i(\boldsymbol{h}; \boldsymbol{\theta})))^{1-x_i},$$

  where $\boldsymbol{f}(\boldsymbol{h}; \boldsymbol{\theta})$ is an MLP, with parameters $\boldsymbol{\theta}$ and input $\boldsymbol{h}$

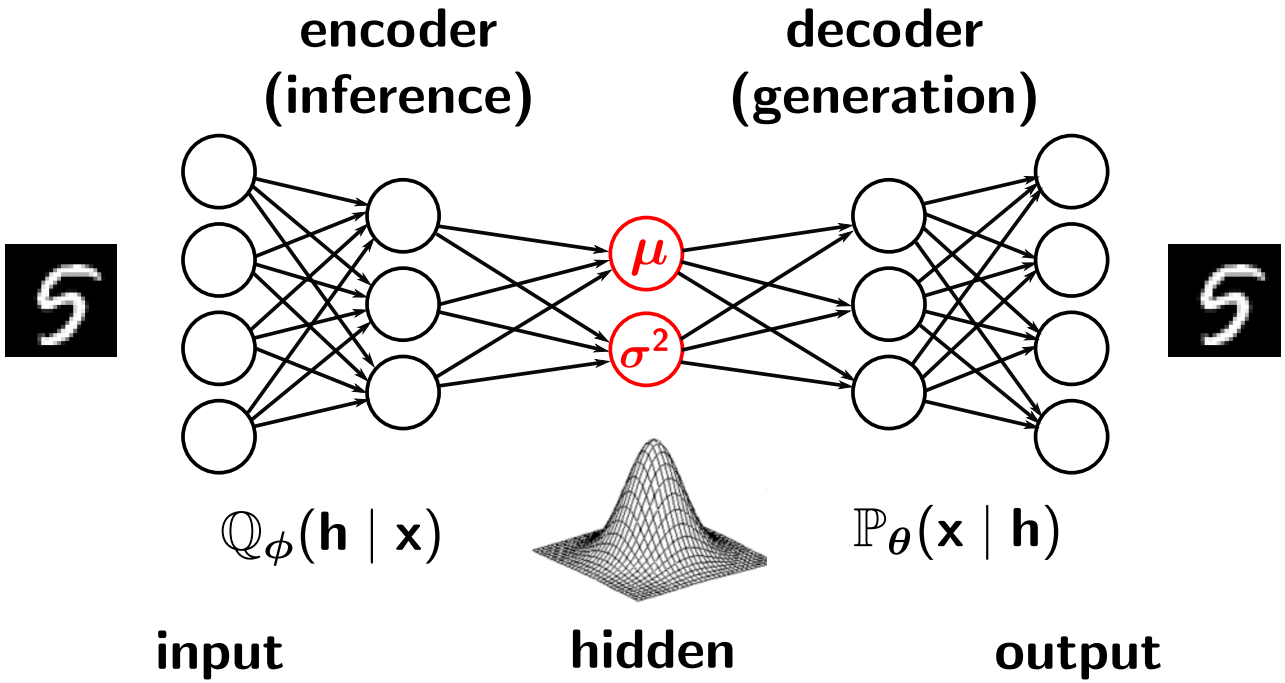
- The true posterior $\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h} \mid \boldsymbol{x})$ is intractable
- Approximate the posterior with a variational distribution

$$\mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h} \mid \boldsymbol{x}) = \mathcal{N}(\boldsymbol{h}; \boldsymbol{\mu}(\boldsymbol{x}; \boldsymbol{\phi}), \boldsymbol{\sigma}^2(\boldsymbol{x}; \boldsymbol{\phi}))$$

  where $\boldsymbol{\mu}(\boldsymbol{x}; \boldsymbol{\phi}$ and $\boldsymbol{\sigma}^2(\boldsymbol{x}; \boldsymbol{\phi})$ are MLPs

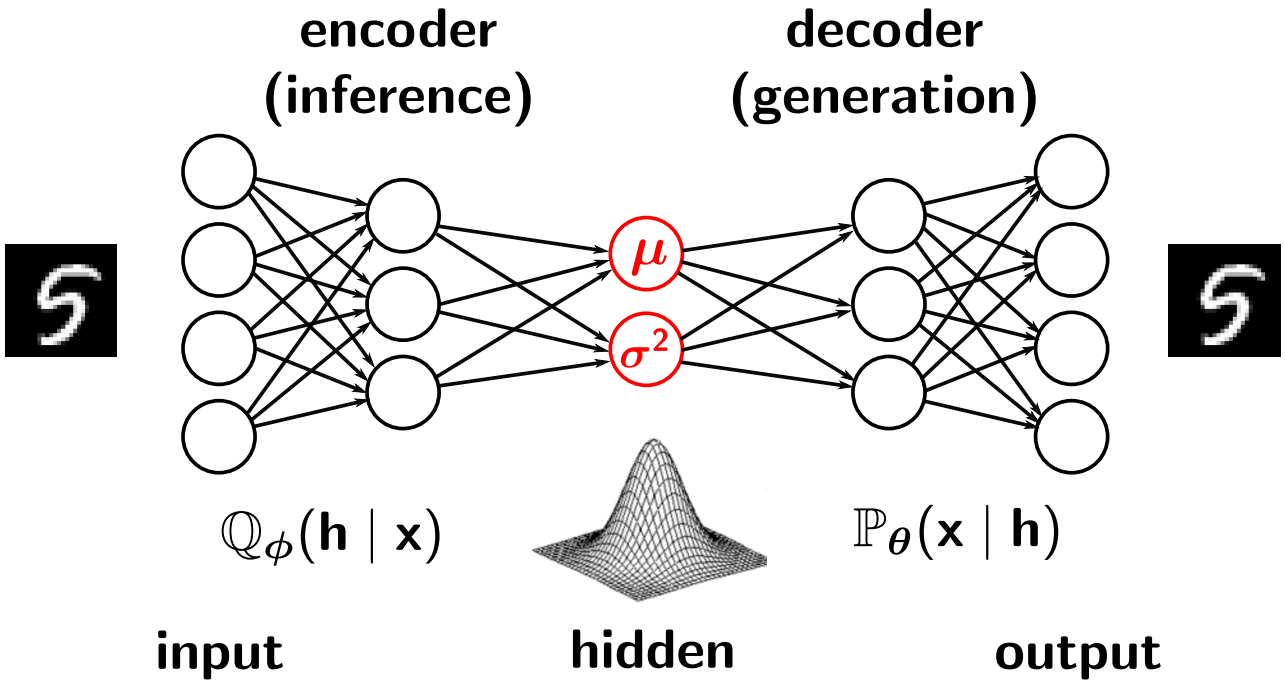
Example: Multivariate Bernoulli with Continuous Latent Variables (Kingma and Welling, 2013)

- This leads to variational auto-encoders:



Example: Multivariate Bernoulli with Continuous Latent Variables (Kingma and Welling, 2013)

- This leads to variational auto-encoders:



- ... we'll come back to this!

Outline

1. Deep Auto-Regressive Models
2. Boltzmann Machines
3. Variational Auto-Encoders
   - Variational Inference and ELBO
   - Gradients and Reparameterization Trick
4. Generative Adversarial Networks
5. Conclusions

Parameter Gradients

- Recall that:

$$\text{ELBO}(\boldsymbol{\phi}; \boldsymbol{\theta}) = \mathbb{E}_{\mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h}|\boldsymbol{x})}[\log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{h}) - \log \mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h} \mid \boldsymbol{x})].$$

Parameter Gradients

- Recall that:

$$\text{ELBO}(\boldsymbol{\phi}; \boldsymbol{\theta}) = \mathbb{E}_{\mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h}|\boldsymbol{x})}[\log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{h}) - \log \mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h} \mid \boldsymbol{x})].$$

- We need to compute gradients with respect to $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$.

Parameter Gradients

- Recall that:

$$\text{ELBO}(\boldsymbol{\phi}; \boldsymbol{\theta}) = \mathbb{E}_{\mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h}|\boldsymbol{x})}[\log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{h}) - \log \mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h} \mid \boldsymbol{x})].$$

- We need to compute gradients with respect to $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$.

- Gradient w.r.t. $\boldsymbol{\theta}$, parameters of the generation network:

$$\nabla_{\boldsymbol{\theta}} \text{ELBO}(\boldsymbol{\phi}; \boldsymbol{\theta}) = \mathbb{E}_{\mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h}|\boldsymbol{x})}[\nabla_{\boldsymbol{\theta}} \log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{h})]$$

  - Follows from linearity of the expectation.

  - This is simple and can be well approximated with Monte Carlo samples.

Parameter Gradients

- Recall that:

$$\text{ELBO}(\boldsymbol{\phi}; \boldsymbol{\theta}) = \mathbb{E}_{\mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h}|\boldsymbol{x})}[\log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{h}) - \log \mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h} \mid \boldsymbol{x})].$$

Parameter Gradients

- Recall that:

$$\text{ELBO}(\boldsymbol{\phi}; \boldsymbol{\theta}) = \mathbb{E}_{\mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h}|\boldsymbol{x})}[\log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{h}) - \log \mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h} \mid \boldsymbol{x})].$$

Parameter Gradients

- Recall that:

$$\text{ELBO}(\boldsymbol{\phi}; \boldsymbol{\theta}) = \mathbb{E}_{\mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h}|\boldsymbol{x})}[\log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{h}) - \log \mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h} \mid \boldsymbol{x})].$$

- Gradient w.r.t. $\boldsymbol{\theta}$, parameters of the inference network:

$$\begin{aligned} &\nabla_{\boldsymbol{\phi}} \text{ELBO}(\boldsymbol{\phi}; \boldsymbol{\theta}) \\ &= \mathbb{E}_{\mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h}|\boldsymbol{x})}[\underbrace{(\log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{h}) - \log \mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h} \mid \boldsymbol{x}))}_{\text{“reward” } R_{\boldsymbol{\theta},\boldsymbol{\phi}}(\boldsymbol{h})} \nabla_{\boldsymbol{\phi}} \log \mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h} \mid \boldsymbol{x})]. \end{aligned}$$

(derivation in the next slide)

  - Problem: Monte Carlo estimators have high variance due to the left part!
  - Requires variance reduction techniques.
  - Resembles REINFORCE (Williams, 1992) (a reinforcement learning algorithm)

Derivation of the Inference Network Gradient

$$
\begin{aligned}
&\nabla_{\boldsymbol{\phi}} \text{ELBO}(\boldsymbol{\phi}; \boldsymbol{\theta}) \\
&= \nabla_{\boldsymbol{\phi}} \mathbb{E}_{\mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h}|\boldsymbol{x})}[\log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{h}) - \log \mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h} \mid \boldsymbol{x})] \\
&= \nabla_{\boldsymbol{\phi}} \sum_{\boldsymbol{h}} \mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h} \mid \boldsymbol{x}) \log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{h}) - \nabla_{\boldsymbol{\phi}} \sum_{\boldsymbol{h}} \mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h} \mid \boldsymbol{x}) \log \mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h} \mid \boldsymbol{x}) \\
&= \sum_{\boldsymbol{h}} \log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{h}) \nabla_{\boldsymbol{\phi}} \mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h} \mid \boldsymbol{x}) - \sum_{\boldsymbol{h}} (1 + \log \mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h} \mid \boldsymbol{x})) \nabla_{\boldsymbol{\phi}} \mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h} \mid \boldsymbol{x}) \\
&= \sum_{\boldsymbol{h}} (\log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{h}) - \log \mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h} \mid \boldsymbol{x})) \nabla_{\boldsymbol{\phi}} \mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h} \mid \boldsymbol{x}) \\
&= \mathbb{E}_{\mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h}|\boldsymbol{x})}[(\log \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{h}) - \log \mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h} \mid \boldsymbol{x})) \nabla_{\boldsymbol{\phi}} \log \mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h} \mid \boldsymbol{x})],
\end{aligned}
$$

where we used the facts:

$$
\sum_{\boldsymbol{h}} \nabla_{\boldsymbol{\phi}} \mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h} \mid \boldsymbol{x}) = \nabla_{\boldsymbol{\phi}} \sum_{\boldsymbol{h}} \mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h} \mid \boldsymbol{x}) = \nabla_{\boldsymbol{\phi}} 1 = 0.
$$

$$
\nabla_{\boldsymbol{\phi}} \mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h} \mid \boldsymbol{x}) = \mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h} \mid \boldsymbol{x}) \nabla_{\boldsymbol{\phi}} \log \mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h} \mid \boldsymbol{x}).
$$

- Summing up, the bottleneck is the gradient w.r.t. ϕ, the parameters of the inference network

- ... the Monte Carlo approximation has large variance.

- Summing up, the bottleneck is the gradient w.r.t. ϕ, the parameters of the inference network
- ... the Monte Carlo approximation has large variance.
- Is there a better strategy?

- Summing up, the bottleneck is the gradient w.r.t. ϕ, the parameters of the inference network
- ... the Monte Carlo approximation has large variance.
- Is there a better strategy?
- Yes: the reparameterization trick.

Reparameterization Trick (Kingma and Welling, 2013)

- How to draw samples from $\mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h} \mid \boldsymbol{x})$?

Reparameterization Trick (Kingma and Welling, 2013)

- How to draw samples from $\mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h} \mid \boldsymbol{x})$?

- Trick:

  - Use an auxiliary random variable $\boldsymbol{\epsilon}$ with fixed distribution $\mathbb{P}(\boldsymbol{\epsilon})$

  - Sample $\boldsymbol{\epsilon} \sim \mathbb{P}(\boldsymbol{\epsilon})$, and obtain $\boldsymbol{h}$ as a deterministic function of $\boldsymbol{\epsilon}$ and $\boldsymbol{x}$

$$\boldsymbol{h} = \mathbf{g}_{\boldsymbol{\phi}}(\boldsymbol{\epsilon}, \boldsymbol{x})$$

Reparameterization Trick (Kingma and Welling, 2013)

- How to draw samples from $\mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h} \mid \boldsymbol{x})$?

- Trick:

  - Use an auxiliary random variable $\boldsymbol{\epsilon}$ with fixed distribution $\mathbb{P}(\boldsymbol{\epsilon})$

  - Sample $\boldsymbol{\epsilon} \sim \mathbb{P}(\boldsymbol{\epsilon})$, and obtain $\boldsymbol{h}$ as a deterministic function of $\boldsymbol{\epsilon}$ and $\boldsymbol{x}$

$$\boldsymbol{h} = \boldsymbol{g}_{\boldsymbol{\phi}}(\boldsymbol{\epsilon}, \boldsymbol{x})$$

- Consequently, for any function f,

$$\mathbb{E}_{\mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h}|\boldsymbol{x})}[f(\boldsymbol{h})] \approx \frac{1}{N} \sum_{i=1}^{N} f(\boldsymbol{g}_{\boldsymbol{\phi}}(\boldsymbol{x}, \boldsymbol{\epsilon}^{(i)}))$$

Reparameterization Trick (Kingma and Welling, 2013)

- How to draw samples from $\mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h} \mid \boldsymbol{x})$?

- Trick:

  - Use an auxiliary random variable $\boldsymbol{\epsilon}$ with fixed distribution $\mathbb{P}(\boldsymbol{\epsilon})$

  - Sample $\boldsymbol{\epsilon} \sim \mathbb{P}(\boldsymbol{\epsilon})$, and obtain $\boldsymbol{h}$ as a deterministic function of $\boldsymbol{\epsilon}$ and $\boldsymbol{x}$

$$\boldsymbol{h} = \boldsymbol{g}_{\boldsymbol{\phi}}(\boldsymbol{\epsilon}, \boldsymbol{x})$$

- Consequently, for any function f,

$$\mathbb{E}_{\mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h}|\boldsymbol{x})}[f(\boldsymbol{h})] \approx \frac{1}{N} \sum_{i=1}^{N} f(\boldsymbol{g}_{\boldsymbol{\phi}}(\boldsymbol{x}, \boldsymbol{\epsilon}^{(i)}))$$

- Gradients w.r.t. $\boldsymbol{\phi}$ can be estimated with regular backpropagation over $\boldsymbol{g}_{\boldsymbol{\phi}}$.

Reparameterization Trick

- This construction is possible in many cases for continuous latent variables:
  - exponential
  - Gaussian
  - location-scale families
  - log-normal
  - etc...

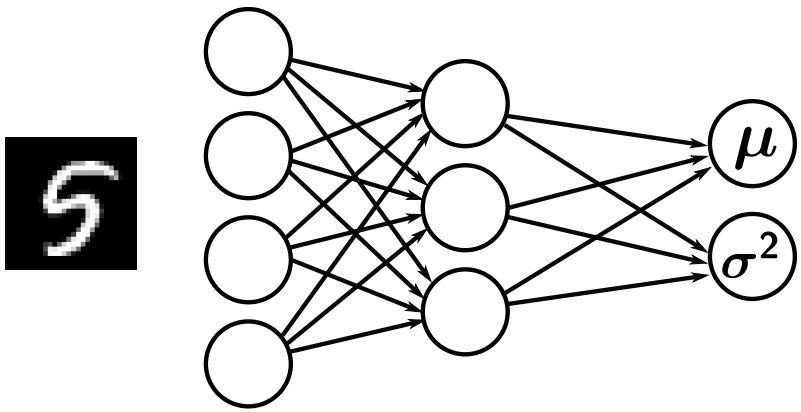
Reparameterization Trick

- This construction is possible in many cases for continuous latent variables:
  - exponential
  - Gaussian
  - location-scale families
  - log-normal
  - etc...
- For discrete latent variables, it is still possible via the Gumbel-softmax trick (not covered in the course).

Example: Gaussian

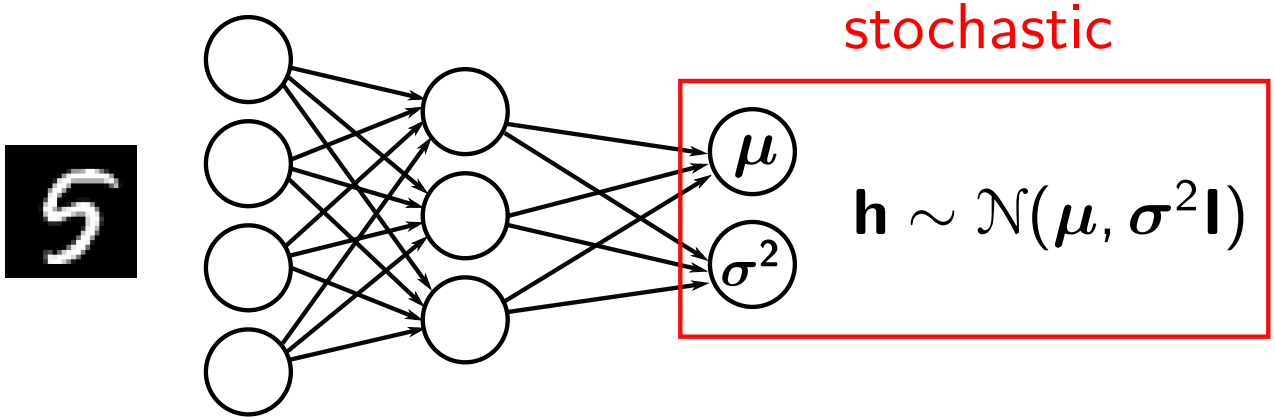
1. Sample $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{\epsilon}; 0, \boldsymbol{I})$
2. Use inference network $\boldsymbol{g}_{\phi}$ with input $\boldsymbol{x}$ to output mean $\boldsymbol{\mu}(\boldsymbol{x})$ and variance $\boldsymbol{\sigma}^2(\boldsymbol{x})$
3. Set $\boldsymbol{h} = \boldsymbol{\mu}(\boldsymbol{x}) + \boldsymbol{\epsilon}\, \boldsymbol{\sigma}(\boldsymbol{x})$.
4. Thus, $\boldsymbol{h} \mid \boldsymbol{x} \sim \mathcal{N}\big(\boldsymbol{\mu}(\boldsymbol{x}), (\boldsymbol{\sigma}(\boldsymbol{x}))^2\big)$

Reparameterization Trick

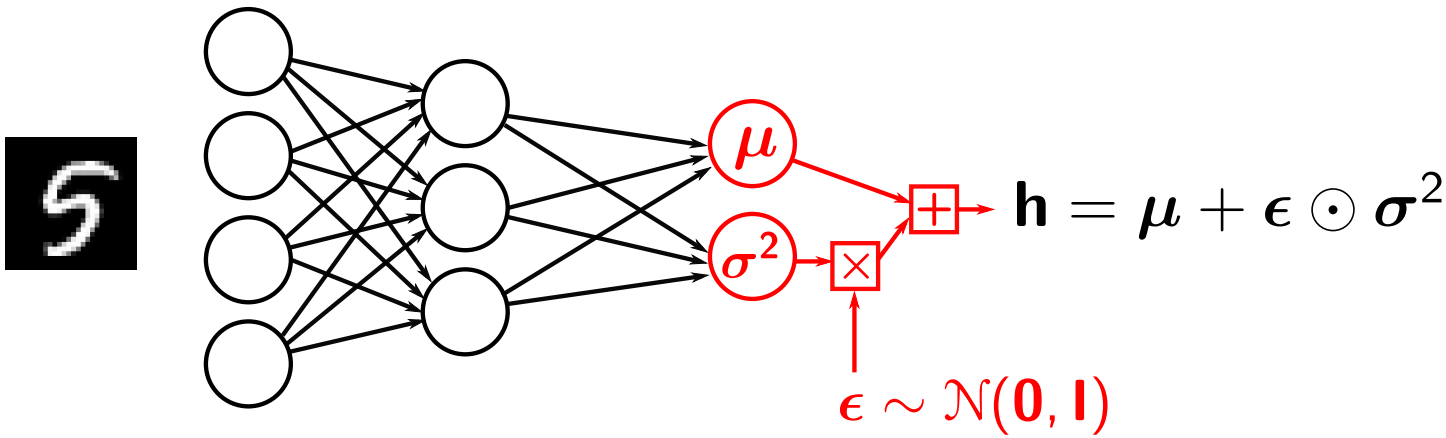


$$\mathbf{h} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2 \mathbf{I})$$

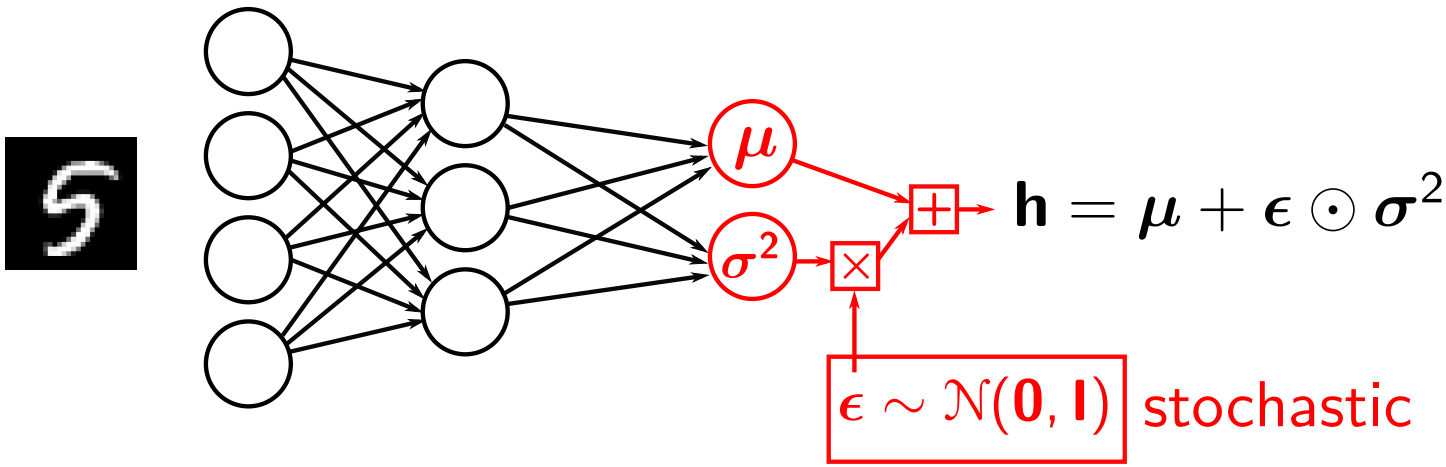
Reparameterization Trick



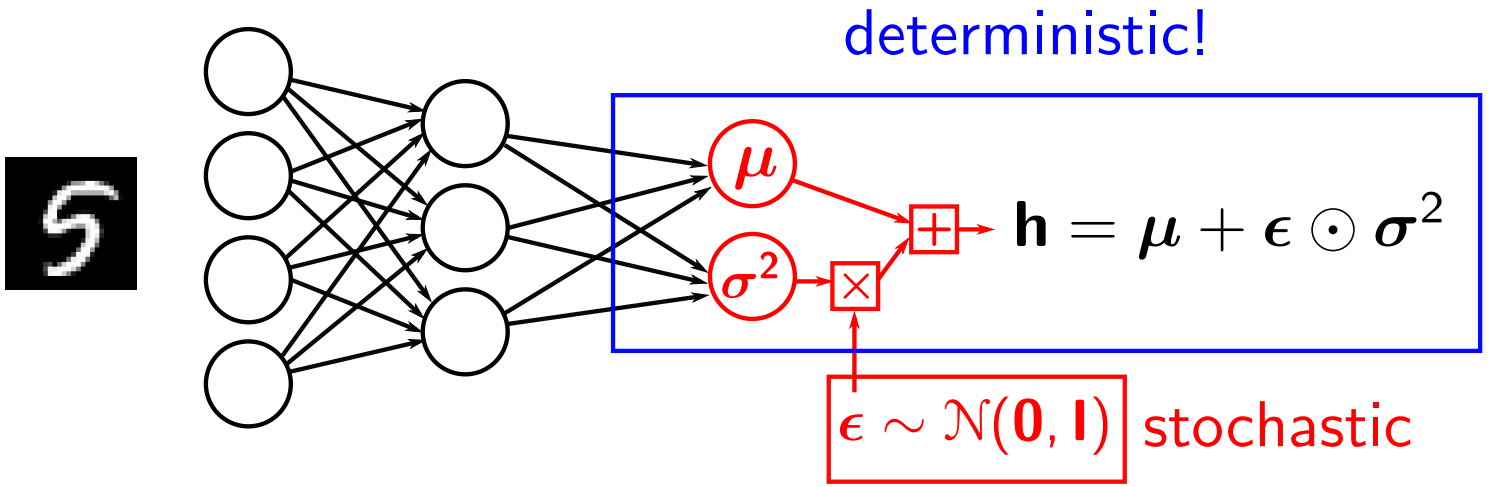
Reparameterization Trick



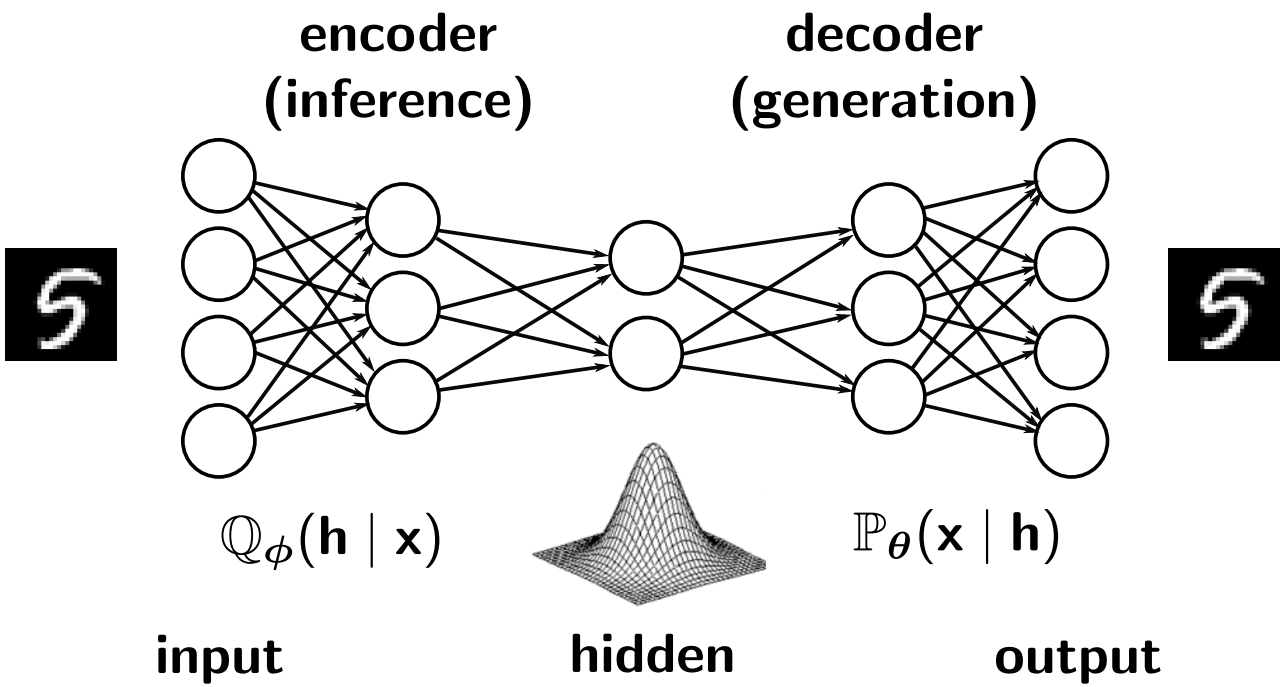
Reparameterization Trick



Reparameterization Trick

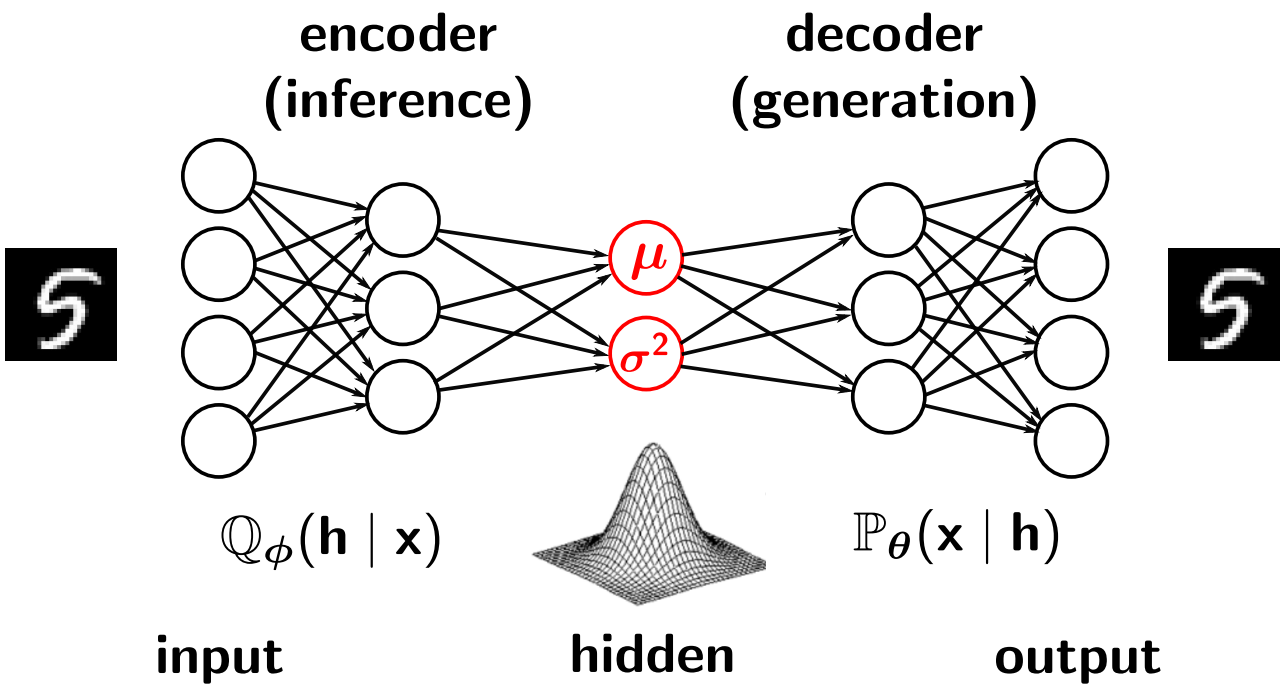


Variational Auto-Encoders (Kingma and Welling, 2013)



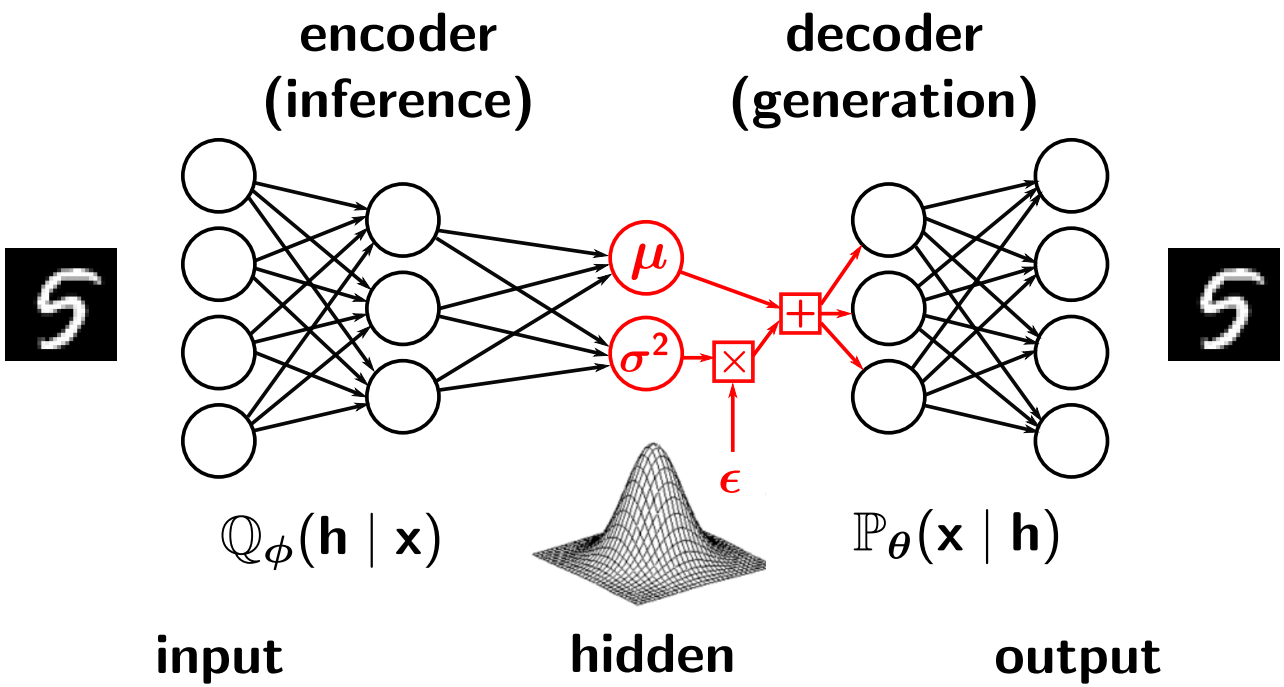
- Decoder computes $\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h})$ and $\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{h})$
- Encoder computes $\mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h} \mid \boldsymbol{x}) = \mathcal{N}(\boldsymbol{h}; \boldsymbol{\mu}_{\boldsymbol{\phi}}(\boldsymbol{x}), \boldsymbol{\sigma}^2_{\boldsymbol{\phi}}(\boldsymbol{x}))$
- Loss function: ELBO.

Variational Auto-Encoders (Kingma and Welling, 2013)



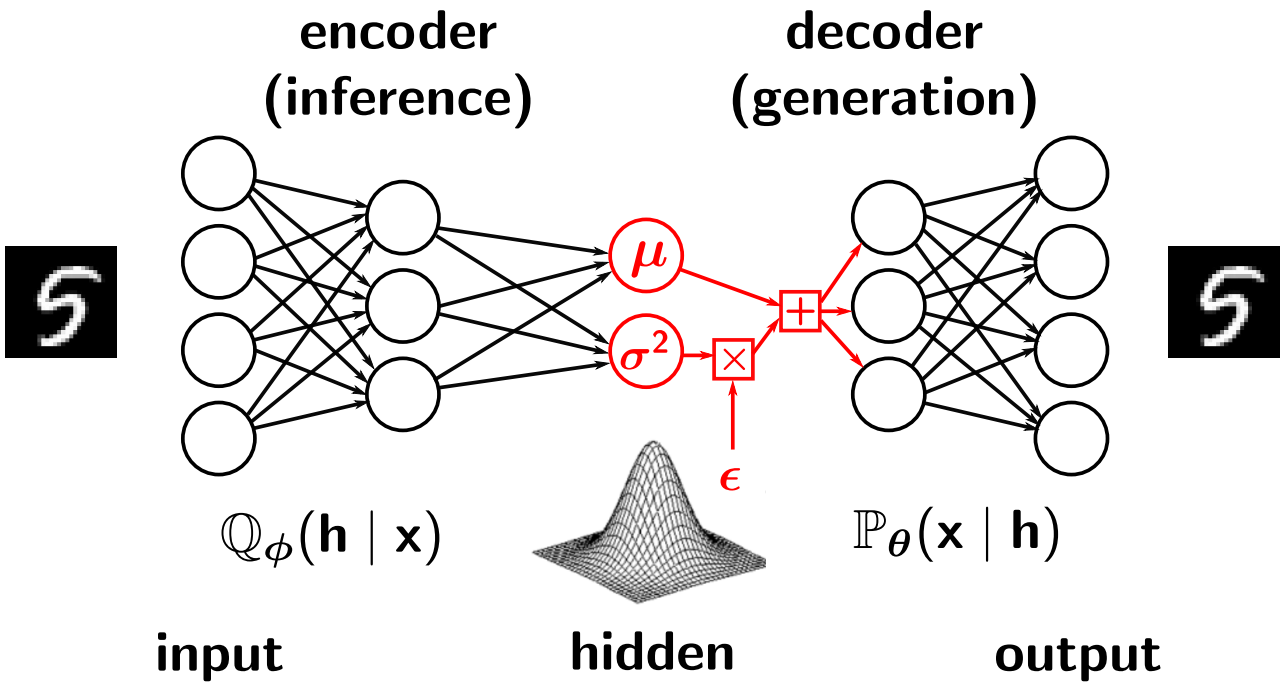
- Decoder computes $\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h})$ and $\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{h})$
- Encoder computes $\mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h} \mid \boldsymbol{x}) = \mathcal{N}(\boldsymbol{h}; \boldsymbol{\mu}_{\boldsymbol{\phi}}(\boldsymbol{x}), \boldsymbol{\sigma}^2_{\boldsymbol{\phi}}(\boldsymbol{x}))$
- Loss function: ELBO.

Variational Auto-Encoders (Kingma and Welling, 2013)

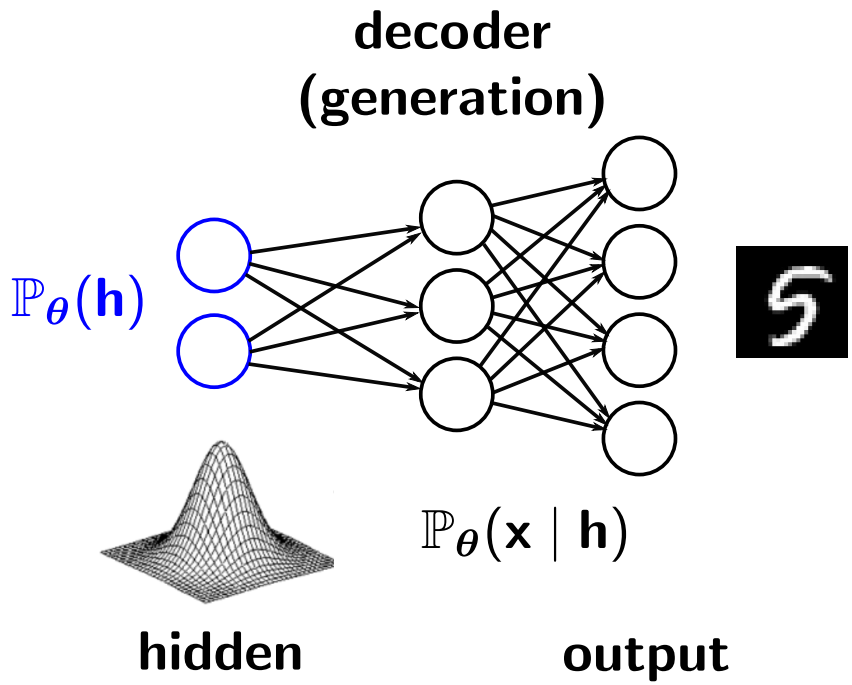


- Decoder computes $\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h})$ and $\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{h})$
- Encoder computes $\mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h} \mid \boldsymbol{x}) = \mathcal{N}(\boldsymbol{h}; \boldsymbol{\mu}_{\boldsymbol{\phi}}(\boldsymbol{x}), \boldsymbol{\sigma}^2_{\boldsymbol{\phi}}(\boldsymbol{x}))$
- Loss function: ELBO.

Summing Up: VAEs at Training Time



Summing Up: VAEs at Test Time



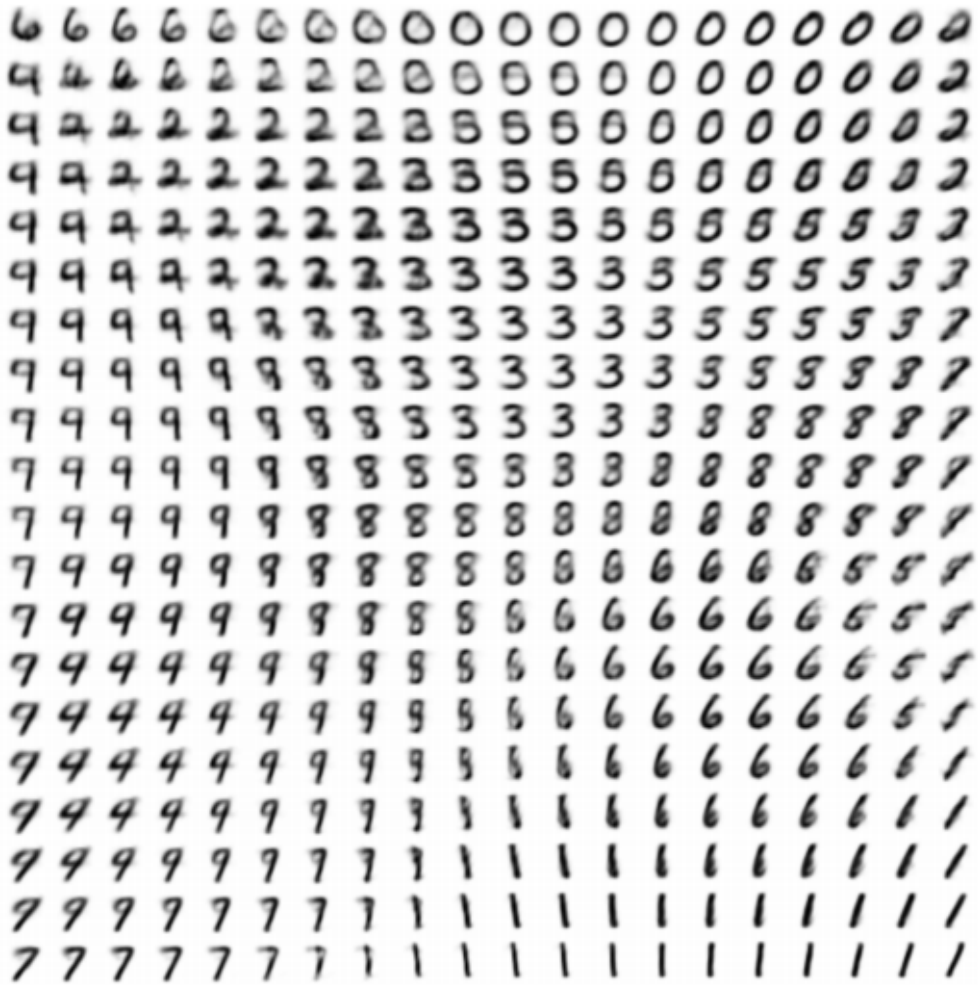
What is the Latent Variable Representing?

- Nice property of VAE: simultaneously training a parametric encoder in combination with a generator network forces the model to learn a predictable coordinate system that the encoder can capture.
- This makes it an excellent manifold learning algorithm.
- Example: the algorithm discovered two independent factors of variation present in images of faces: angle of rotation and emotional expression.

What is the Latent Variable Representing?

From Kingma and Welling (2013).

What is the Latent Variable Representing?



(b) Learned MNIST manifold

From Kingma and Welling (2013).

Issues with VAEs

- Posterior collapse: if the generative part is strong, the model may learn to ignore the latent variables:

$$\begin{aligned} \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{h}) &\approx \mathbb{P}(\boldsymbol{x}) \\ \mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h} \mid \boldsymbol{x}) &\approx \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h}). \end{aligned}$$

Issues with VAEs

- Posterior collapse: if the generative part is strong, the model may learn to ignore the latent variables:

$$\begin{aligned} \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{h}) &\approx \mathbb{P}(\boldsymbol{x}) \\ \mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h} \mid \boldsymbol{x}) &\approx \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h}). \end{aligned}$$

- Can be mitigated with a few tricks:
  - Decrease/anneal the weight of $KL(\mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h} \mid \boldsymbol{x}) \| \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h}))$ in the ELBO objective
  - Use auxiliary losses
  - Combine stochastic and amortized inference.

Issues with VAEs

- Posterior collapse: if the generative part is strong, the model may learn to ignore the latent variables:

$$\begin{aligned} \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{h}) &\approx \mathbb{P}(\boldsymbol{x}) \\ \mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h} \mid \boldsymbol{x}) &\approx \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h}). \end{aligned}$$

- Can be mitigated with a few tricks:
  - Decrease/anneal the weight of $KL(\mathbb{Q}_{\boldsymbol{\phi}}(\boldsymbol{h} \mid \boldsymbol{x}) \| \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h}))$ in the ELBO objective
  - Use auxiliary losses
  - Combine stochastic and amortized inference.
- In general, reporting both reconstruction loss and the KL term is needed to be able to tell if the model makes use of the latent variables.

Examples of Deep Generative Models

- Auto-Regressive Networks ✓
- Restricted Boltzmann Machines ✓
- Deep Belief Networks
- Deep Boltzmann Machines
- Gaussian-Bernoulli RBMs
- Convolutional Boltzmann Machines
- Sigmoid Belief Nets
- Variational Auto-Encoders ✓
- Generative Adversarial Networks
- Convolutional Generative Networks
- Generative Stochastic Networks

Examples of Deep Generative Models

- Auto-Regressive Networks ✓
- Restricted Boltzmann Machines ✓
- Deep Belief Networks
- Deep Boltzmann Machines
- Gaussian-Bernoulli RBMs
- Convolutional Boltzmann Machines
- Sigmoid Belief Nets
- Variational Auto-Encoders ✓
- Generative Adversarial Networks
- Convolutional Generative Networks
- Generative Stochastic Networks

Outline

1. Deep Auto-Regressive Models
2. Boltzmann Machines
3. Variational Auto-Encoders
   - Variational Inference and ELBO
   - Gradients and Reparameterization Trick
4. Generative Adversarial Networks
5. Conclusions

Why Maximum Likelihood?

- All models discussed aim to maximize the likelihood (evidence) $\mathbb{P}(\boldsymbol{x})$
- In fact, since this is intractable, they maximize a lower bound (ELBO)
- But if we want to build a generator, is this really the best criterion?
- Maximum likelihood tends to produce blurry images

Generative Adversarial Networks (GANs) (Goodfellow et al., 2014)

- **Key idea:**
  - keep the generation network $G = \{\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h}), \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{h})\}$
  - drop the inference network and use instead a discriminator network $D : \mathcal{X} \to \{0, 1\}$

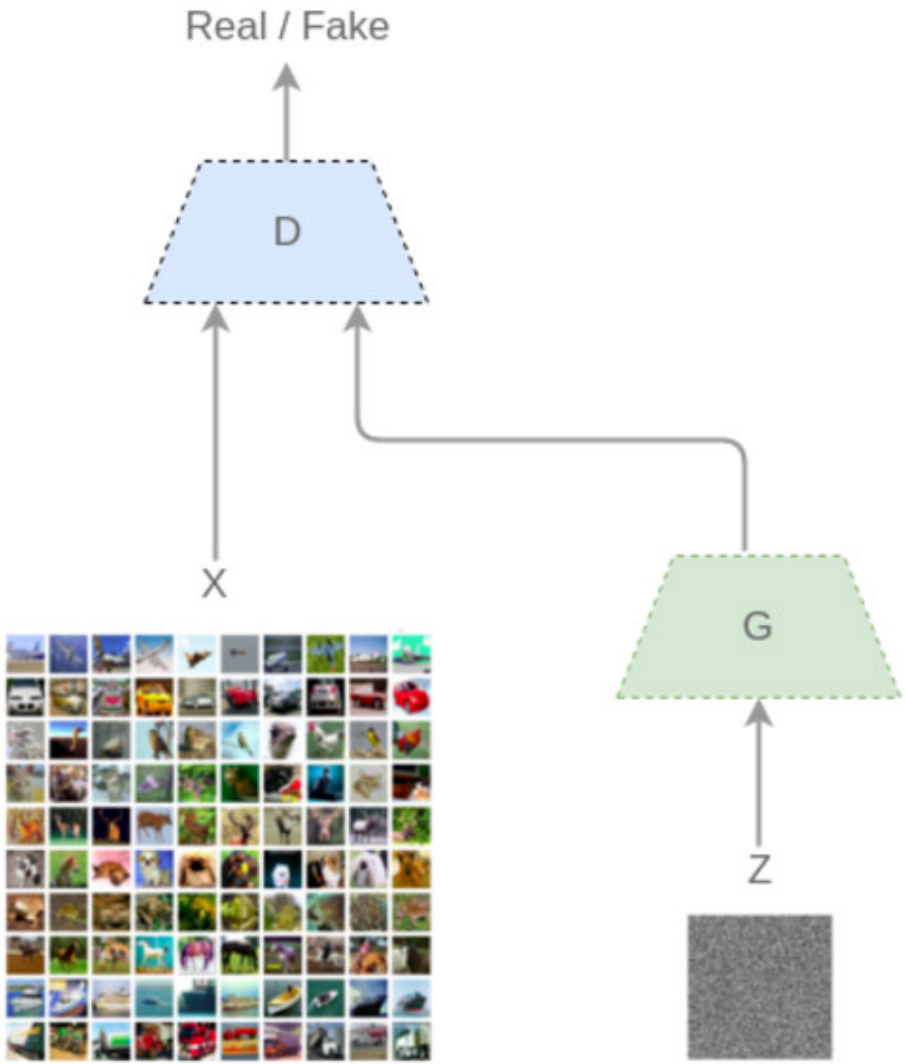
Generative Adversarial Networks (GANs) (Goodfellow et al., 2014)

- **Key idea:**
  - keep the generation network $G = \{\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h}), \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{h})\}$
  - drop the inference network and use instead a discriminator network $D : \mathcal{X} \to \{0, 1\}$
- Formulate the learning problem as a game between two players:
  - the generator's job is to generate data that looks real
  - the discriminator's job is to distinguish between real data and fake data generated by the generator

Generative Adversarial Networks (GANs) (Goodfellow et al., 2014)

- **Key idea:**
  - keep the generation network $G = \{\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h}), \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{h})\}$
  - drop the inference network and use instead a discriminator network $D : \mathcal{X} \to \{0, 1\}$
- Formulate the learning problem as a game between two players:
  - the generator's job is to generate data that looks real
  - the discriminator's job is to distinguish between real data and fake data generated by the generator
- This is like a Turing test: distinguish artificial from real.

Generative Adversarial Networks (GANs) (Goodfellow et al., 2014)



Minimax Game

- Saddle point problem:

$$\min_G \max_D V(D, G) = \mathbb{E}_{\mathbb{P}_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h})}[\log(1 - D(G(\boldsymbol{h})))].$$

Minimax Game

- Saddle point problem:

$$\min_G \max_D V(D,G) = \mathbb{E}_{\mathbb{P}_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h})}[\log(1 - D(G(\boldsymbol{h})))].$$

- The optimal discriminator (intractable to compute) is:

$$D^\star(\boldsymbol{x}) = \frac{\mathbb{P}_{\text{data}}(\boldsymbol{x})}{\mathbb{P}_{\text{data}}(\boldsymbol{x}) + \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x})}, \qquad \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x}) = \int \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{h}) \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h}).$$

Minimax Game

- Saddle point problem:

$$\min_G \max_D V(D,G) = \mathbb{E}_{\mathbb{P}_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h})}[\log(1 - D(G(\boldsymbol{h})))].$$

- The optimal discriminator (intractable to compute) is:

$$D^\star(\boldsymbol{x}) = \frac{\mathbb{P}_{\text{data}}(\boldsymbol{x})}{\mathbb{P}_{\text{data}}(\boldsymbol{x}) + \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x})}, \qquad \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x}) = \int \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{h}) \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h}).$$

- Given $D^\star(\boldsymbol{x})$, the optimal generator $G^\star(\boldsymbol{x})$ minimizes the Jensen-Shannon divergence between $\mathbb{P}_{\text{data}}(\boldsymbol{x})$ and $\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x})$:

$$JS(\mathbb{P}_{\text{data}}(\boldsymbol{x}), \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x})) = \frac{1}{2} KL\left(\mathbb{P}_{\text{data}}(\boldsymbol{x}) \| \bar{\mathbb{P}}(\boldsymbol{x})\right) + \frac{1}{2} KL\left(\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x}) \| \bar{\mathbb{P}}(\boldsymbol{x})\right),$$

where $\bar{\mathbb{P}}(\boldsymbol{x}) = \frac{\mathbb{P}_{\text{data}}(\boldsymbol{x}) + \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x})}{2}$.

Training GANs

- How to train a GAN?

Training GANs

- How to train a GAN?
- Use stochastic gradient descent! Alternate between:
  - Stochastic gradients updates of the generator parameters $\boldsymbol{\theta}$
  - Stochastic gradients updates of the discriminator D.

Training GANs

- How to train a GAN?
- Use stochastic gradient descent! Alternate between:
  - Stochastic gradients updates of the generator parameters $\boldsymbol{\theta}$
  - Stochastic gradients updates of the discriminator D.
- Several variants and schedules have been proposed.

Training GANs

- How to train a GAN?
- Use stochastic gradient descent! Alternate between:
  - Stochastic gradients updates of the generator parameters $\boldsymbol{\theta}$
  - Stochastic gradients updates of the discriminator D.
- Several variants and schedules have been proposed.
- Caveats:
  - no convergence guarantees
  - optimization in GANs is often difficult

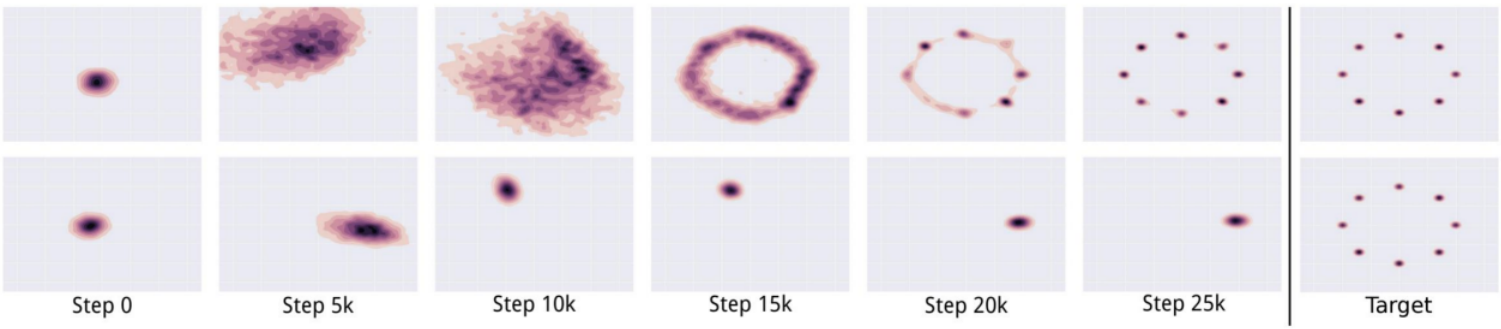
Images Generated by GANs

(https://tryolabs.com/blog/2016/12/06/major-advancements-deep-learning-2016/)

Mode Collapse

$$\min_G \max_D V(D, G) \neq \max_D \min_G V(D, G).$$

- G in inner loop: place all mass on most likely point



(From Metz et al. (2016))

- What prevents the generator from always picking the same example?
- The top row finds all the modes, the bottom finds just one mode.

Mode Collapse

- GANs often seem to collapse to far fewer modes than the model can represent
- This causes low output diversity.
- How to mitigate mode collapse?
- One strategy: minibatch (Salimans et al., 2016)
  - Let the discriminator make a decision by comparing an example to a whole minibatch of fake/real examples
  - Discriminator can now consider diversity.

Wasserstein GANs (WGANs)

- Instead of optimizing the Jensen-Shannon divergence, optimize instead:

$$\min_G \max_D V(D,G) \quad = \quad \mathbb{E}_{\boldsymbol{x}\sim\mathbb{P}_{\text{data}}(\boldsymbol{x})}[D(\boldsymbol{x})] - \mathbb{E}_{\boldsymbol{h}\sim\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{h})}[D(G(\boldsymbol{h}))].$$

- This is related to the Wasserstein distance (also called Earth mover's distance).

- A technical condition is that ∇D is bounded; in practice this is ensured with gradient clipping.

- This improves stability and mitigates the mode collapse problem.

Pros and Cons of GANs

Advantages:

- They currently generate the sharpest images
- They are cheap to train (since no statistical inference is required), and only back-propogation is needed to obtain gradients

Disadvantages:

- GANs are difficult to optimize due to unstable training dynamics.
- No statistical inference can be done with them.

Still Improving...

Ian Goodfellow
@goodfellow_ian

Follow

4 years of GAN progress (source: eff.org/files/2018/02/ …)

2014

2015

2016

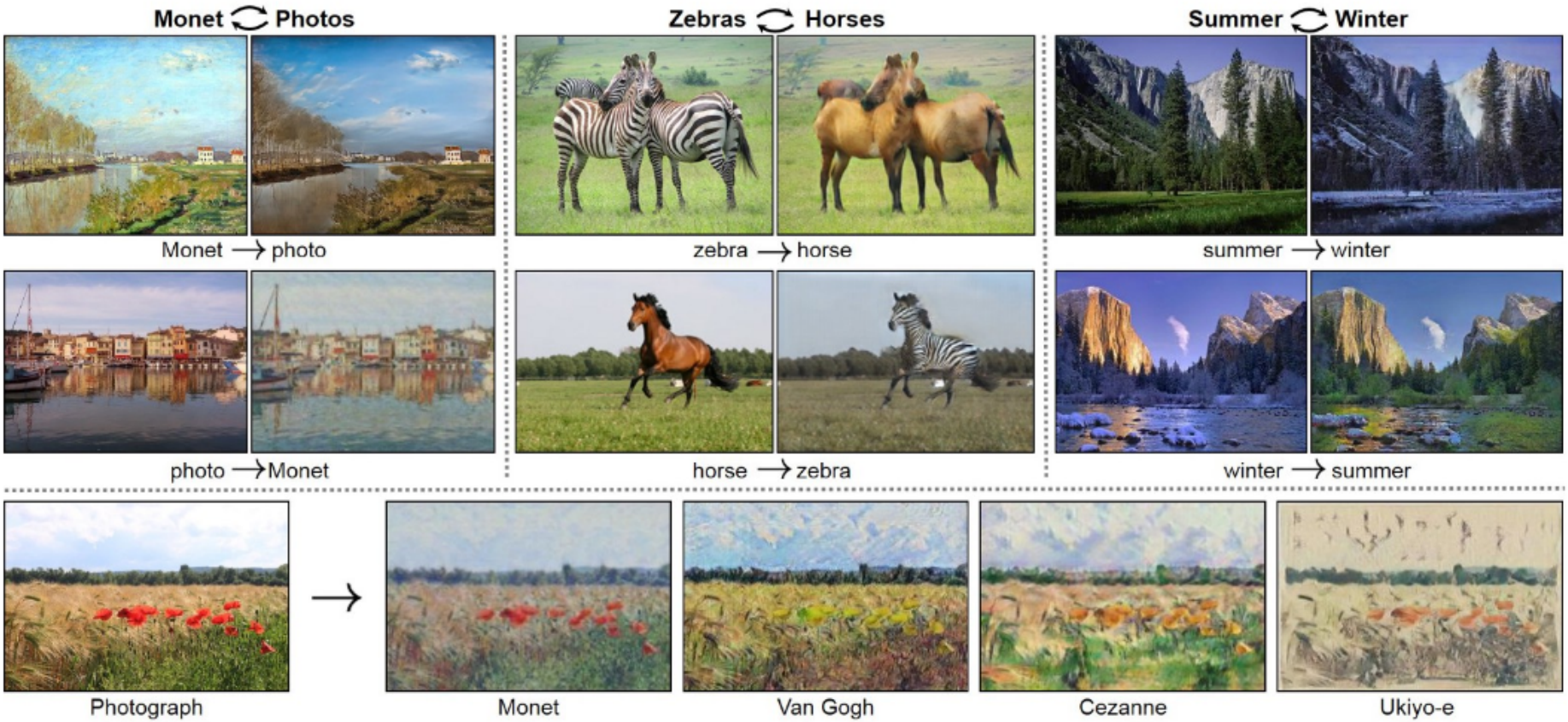
2017

7:26 PM - 2 Mar 2018

Some Extensions of GANs

- Augmenting GANs with an inference network (Dumoulin et al., 2016; Donahue et al., 2016)
- Domain adversarial training for domain adaptation (Ganin et al., 2016)
- Conditional GANs and semi-supervised GANs (Salimans et al., 2016)
- CycleGAN (Zhu et al., 2017): "translate" images from a source domain $\mathcal{X}$ to a target domain $\mathcal{Y}$ without paired examples. Use two generators $G : \mathcal{Y} \to \mathcal{X}$ and $F : \mathcal{X} \to \mathcal{Y}$ and introduce a **cycle consistency loss** to push $F(G(\boldsymbol{y})) \approx \boldsymbol{y}$ and $G(F(\boldsymbol{x})) \approx \boldsymbol{x}$.

Image-to-Image Translation w/ CycleGAN (Zhu et al., 2017)



(https://junyanz.github.io/CycleGAN)

Failure Cases

(https://junyanz.github.io/CycleGAN)

Evaluation

There is no single compelling way to evaluate a generative model.

- Models with good likelihood can produce bad samples
- Models with good samples can have bad likelihood
- There is no standard way to quantify how good samples are
- For GANs, it is also hard to even estimate the likelihood
- See "A note on the evaluation of generative models," Theis et al. (2015), for a good overview.

Discrete Outputs

To train a GAN, G must be differentiable

But G cannot be differentiable if the output is discrete.

Possible workarounds:

- REINFORCE (Williams, 1992)
- Concrete/Gumbel-softmax distribution (Maddison et al., 2016; Jang et al., 2016)
- Learn distribution over continuous embeddings, decode to discrete

How does this compare with VAEs?

- VAEs have trouble with discrete latent variables (cannot differentiate through the inference network)
- GANs have trouble with discrete output variables (cannot differentiate through the generator network).

Connections to Reinforcement Learning

We can regard the discriminator loss as a reward signal for the generator.

- GANs interpreted as actor-critic (Pfau and Vinyals, 2016)
- GANs as inverse reinforcement learning (Finn et al., 2016)

Outline

1. Deep Auto-Regressive Models
2. Boltzmann Machines
3. Variational Auto-Encoders
   - Variational Inference and ELBO
   - Gradients and Reparameterization Trick
4. Generative Adversarial Networks
5. Conclusions

Conclusions

- Generative models are useful to model high-dimensional data
- Latent-variable generative models are appealing since they are more compact ("minimum description length" principle)
- Often, computing evidence and posterior distributions is intractable (e.g. Boltzmann machines)
- A common surrogate for maximum likelihood is the evidence lower bound (ELBO)
- Variational auto-encoders optimize the ELBO with amortized VI
- Their main drawback is posterior collapse
- Generative adversarial networks (GANs) are formulated as a game between a generator and a discriminator
- They manage to generate sharp outputs, but suffer from mode collapse and do not return a likelihood score
- Open problem (both VAEs/GANs): how to deal with discrete data?

Thank you!

Questions?

References I

Ackley, D., Hinton, G., and Sejnowski, T. (1985). A learning algorithm for Boltzmann machines. *Cognitive science*, 9(1):147–169.

Donahue, J., Krähenbühl, P., and Darrell, T. (2016). Adversarial feature learning. *arXiv preprint arXiv:1605.09782*.

Dumoulin, V., Belghazi, I., Poole, B., Mastropietro, O., Lamb, A., Arjovsky, M., and Courville, A. (2016). Adversarially learned inference. *arXiv preprint arXiv:1606.00704*.

Finn, C., Christiano, P., Abbeel, P., and Levine, S. (2016). A connection between generative adversarial networks, inverse reinforcement learning, and energy-based models. *arXiv preprint arXiv:1611.03852*.

Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., and Lempitsky, V. (2016). Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 17(1):2096–2030.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). Deep Learning. Book in preparation for MIT Press.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680.

Jang, E., Gu, S., and Poole, B. (2016). Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*.

Kingma, D. P. and Welling, M. (2013). Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

Le Roux, N. and Bengio, Y. (2008). Representational power of restricted boltzmann machines and deep belief networks. *Neural computation*, 20(6):1631–1649.

Maddison, C. J., Mnih, A., and Teh, Y. W. (2016). The concrete distribution: A continuous relaxation of discrete random variables. *arXiv preprint arXiv:1611.00712*.

Metz, L., Poole, B., Pfau, D., and Sohl-Dickstein, J. (2016). Unrolled generative adversarial networks. *arXiv preprint arXiv:1611.02163*.

Oord, A. v. d., Kalchbrenner, N., and Kavukcuoglu, K. (2016). Pixel Recurrent Neural Networks. In *Proc. of the International Conference on Machine Learning*.

Pfau, D. and Vinyals, O. (2016). Connecting generative adversarial networks and actor-critic methods. *arXiv preprint arXiv:1610.01945*.

Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., and Chen, X. (2016). Improved techniques for training gans. In *Advances in Neural Information Processing Systems*, pages 2234–2242.

References II

Smolensky, P. (1986). Information processing in dynamical systems: Foundations of harmony theory. Technical report, DTIC Document.

Theis, L., Oord, A. v. d., and Bethge, M. (2015). A note on the evaluation of generative models. *arXiv preprint arXiv:1511.01844*.

Wainwright, M. and Jordan, M. (2008). *Graphical Models, Exponential Families, and Variational Inference*. Now Publishers.

Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256.

Zhu, J.-Y., Park, T., Isola, P., and Efros, A. A. (2017). Unpaired image-to-image translation using cycle-consistent adversarial networks. *arXiv preprint*.