Deep Generative Models

Lecture 5

Roman Isachenko

AI Masters

2024, Spring

Recap of previous lecture

Forward pass (Loss function)

$$\mathbf{z} = \mathbf{x} + \int_{t_1}^{t_0} f_{\boldsymbol{\theta}}(\mathbf{z}(t), t)dt, \quad L(\mathbf{z}) = -\log p(\mathbf{x}|\boldsymbol{\theta})$$

$$L(\mathbf{z}) = -\log p(\mathbf{z}) + \int_{t_0}^{t_1} \operatorname{tr}\left(\frac{\partial f_{\boldsymbol{\theta}}(\mathbf{z}(t), t)}{\partial \mathbf{z}(t)}\right) dt$$

Adjoint functions

$$\mathbf{a}_{\mathbf{z}}(t) = \frac{\partial L}{\partial \mathbf{z}(t)}; \quad \mathbf{a}_{\boldsymbol{\theta}}(t) = \frac{\partial L}{\partial \boldsymbol{\theta}(t)}.$$

These functions show how the gradient of the loss depends on the hidden state $\mathbf{z}(t)$ and parameters $\boldsymbol{\theta}$.

Theorem (Pontryagin)

$$\frac{d\mathbf{a}_{\mathbf{z}}(t)}{dt} = -\mathbf{a}_{\mathbf{z}}(t)^T \cdot \frac{\partial f_{\boldsymbol{\theta}}(\mathbf{z}(t), t)}{\partial \mathbf{z}}; \quad \frac{d\mathbf{a}_{\boldsymbol{\theta}}(t)}{dt} = -\mathbf{a}_{\mathbf{z}}(t)^T \cdot \frac{\partial f_{\boldsymbol{\theta}}(\mathbf{z}(t), t)}{\partial \boldsymbol{\theta}}.$$

Chen R. T. Q. et al. Neural Ordinary Differential Equations, 2018

Recap of previous lecture

Forward pass

$$\mathbf{z} = \mathbf{z}(t_0) = \int_{t_0}^{t_1} f_{\boldsymbol{\theta}}(\mathbf{z}(t), t)dt + \mathbf{x} \quad \Rightarrow \quad \text{ODE Solver}$$

Backward pass

$$\left.\begin{aligned} \frac{\partial L}{\partial \boldsymbol{\theta}(t_1)} &= \mathbf{a}_{\boldsymbol{\theta}}(t_1) = -\int_{t_0}^{t_1} \mathbf{a}_{\mathbf{z}}(t)^T \frac{\partial f_{\boldsymbol{\theta}}(\mathbf{z}(t), t)}{\partial \boldsymbol{\theta}(t)} dt + 0 \\ \frac{\partial L}{\partial \mathbf{z}(t_1)} &= \mathbf{a}_{\mathbf{z}}(t_1) = -\int_{t_0}^{t_1} \mathbf{a}_{\mathbf{z}}(t)^T \frac{\partial f_{\boldsymbol{\theta}}(\mathbf{z}(t), t)}{\partial \mathbf{z}(t)} dt + \frac{\partial L}{\partial \mathbf{z}(t_0)} \\ \mathbf{z}(t_1) &= -\int_{t_1}^{t_0} f_{\boldsymbol{\theta}}(\mathbf{z}(t), t)dt + \mathbf{z}_0. \end{aligned}\right\} \Rightarrow \text{ODE Solver}$$

Note: These scary formulas are the standard backprop in the discrete case.

Chen R. T. Q. et al. Neural Ordinary Differential Equations, 2018

Recap of previous lecture

Bayes theorem

$$p(\mathbf{t}|\mathbf{x}) = \frac{p(\mathbf{x}|\mathbf{t})p(\mathbf{t})}{p(\mathbf{x})} = \frac{p(\mathbf{x}|\mathbf{t})p(\mathbf{t})}{\int p(\mathbf{x}|\mathbf{t})p(\mathbf{t})d\mathbf{t}}$$

- $\mathbf{x}$ – observed variables, $\mathbf{t}$ – unobserved variables (latent variables/parameters);
- $p(\mathbf{x}|\mathbf{t})$ – likelihood;
- $p(\mathbf{x}) = \int p(\mathbf{x}|\mathbf{t})p(\mathbf{t})d\mathbf{t}$ – evidence;
- $p(\mathbf{t})$ – prior distribution, $p(\mathbf{t}|\mathbf{x})$ – posterior distribution.

Posterior distribution

$$p(\boldsymbol{\theta}|\mathbf{X}) = \frac{p(\mathbf{X}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\mathbf{X})} = \frac{p(\mathbf{X}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{\int p(\mathbf{X}|\boldsymbol{\theta})p(\boldsymbol{\theta})d\boldsymbol{\theta}}$$

Recap of previous lecture

Latent variable models (LVM)

$$p(\mathbf{x}|\boldsymbol{\theta}) = \int p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta}) d\mathbf{z} = \int p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta}) p(\mathbf{z}) d\mathbf{z}.$$

MLE problem for LVM

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} \log p(\mathbf{X}|\boldsymbol{\theta}) = \arg\max_{\boldsymbol{\theta}} \sum_{i=1}^{n} \log p(\mathbf{x}_i|\boldsymbol{\theta}) =$$

$$= \arg\max_{\boldsymbol{\theta}} \sum_{i=1}^{n} \log \int p(\mathbf{x}_i|\mathbf{z}_i, \boldsymbol{\theta}) p(\mathbf{z}_i) d\mathbf{z}_i.$$

Naive Monte-Carlo estimation

$$p(\mathbf{x}|\boldsymbol{\theta}) = \int p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta}) p(\mathbf{z}) d\mathbf{z} = \mathbb{E}_{p(\mathbf{z})} p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta}) \approx \frac{1}{K} \sum_{k=1}^{K} p(\mathbf{x}|\mathbf{z}_k, \boldsymbol{\theta}),$$

where $\mathbf{z}_k \sim p(\mathbf{z})$.

Recap of previous lecture

ELBO derivation 1 (inequality)

$$\log p(\mathbf{x}|\boldsymbol{\theta}) = \log \int p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta}) d\mathbf{z} \geq \mathbb{E}_q \log \frac{p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta})}{q(\mathbf{z})} = \mathcal{L}(q, \boldsymbol{\theta})$$

ELBO derivation 2 (equality)

$$\mathcal{L}(q, \boldsymbol{\theta}) = \int q(\mathbf{z}) \log \frac{p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta})}{q(\mathbf{z})} d\mathbf{z} = \int q(\mathbf{z}) \log \frac{p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta}) p(\mathbf{x}|\boldsymbol{\theta})}{q(\mathbf{z})} d\mathbf{z} =$$
$$= \log p(\mathbf{x}|\boldsymbol{\theta}) - KL(q(\mathbf{z})||p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta}))$$

Variational decomposition

$$\log p(\mathbf{x}|\boldsymbol{\theta}) = \mathcal{L}(q, \boldsymbol{\theta}) + KL(q(\mathbf{z})||p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta})) \geq \mathcal{L}(q, \boldsymbol{\theta}).$$

Recap of previous lecture

Variational lower Bound (ELBO)

$$\log p(\mathbf{x}|\boldsymbol{\theta}) = \mathcal{L}(q, \boldsymbol{\theta}) + KL(q(\mathbf{z})||p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta})) \geq \mathcal{L}(q, \boldsymbol{\theta}).$$

$$\mathcal{L}(q, \boldsymbol{\theta}) = \int q(\mathbf{z}) \log \frac{p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta})}{q(\mathbf{z})} d\mathbf{z} = \mathbb{E}_q \log p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta}) - KL(q(\mathbf{z})||p(\mathbf{z}))$$

Log-likelihood decomposition

$$\log p(\mathbf{x}|\boldsymbol{\theta}) = \mathbb{E}_q \log p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta}) - KL(q(\mathbf{z})||p(\mathbf{z})) + KL(q(\mathbf{z})||p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta})).$$

- Instead of maximizing incomplete likelihood, maximize ELBO

$$\max_{\boldsymbol{\theta}} p(\mathbf{x}|\boldsymbol{\theta}) \quad \rightarrow \quad \max_{q, \boldsymbol{\theta}} \mathcal{L}(q, \boldsymbol{\theta})$$

- Maximization of ELBO by variational distribution q is equivalent to minimization of KL

$$\arg\max_{q} \mathcal{L}(q, \boldsymbol{\theta}) \equiv \arg\min_{q} KL(q(\mathbf{z})||p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta})).$$

Outline

1. EM-algorithm
   Amortized inference
   ELBO gradients, reparametrization trick

2. Variational autoencoder (VAE)

3. Data dequantization

Outline

1. EM-algorithm
   Amortized inference
   ELBO gradients, reparametrization trick
2. Variational autoencoder (VAE)
3. Data dequantization

EM-algorithm

$$\mathcal{L}(q, \theta) = \mathbb{E}_q \log p(\mathbf{x}|\mathbf{z}, \theta) - KL(q(\mathbf{z})||p(\mathbf{z})) =$$

$$= \mathbb{E}_q \left[\log p(\mathbf{x}|\mathbf{z}, \theta) - \log \frac{q(\mathbf{z})}{p(\mathbf{z})} \right] d\mathbf{z} \to \max_{q, \theta}.$$

Block-coordinate optimization

- Initialize θ^*;
- **E-step** ($\mathcal{L}(q, \theta) \to \max_q$)

$$q^*(\mathbf{z}) = \arg\max_q \mathcal{L}(q, \theta^*) =$$

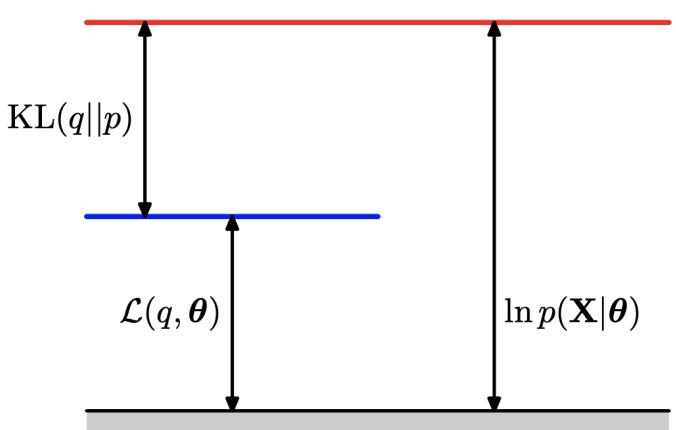
$$= \arg\min_q KL(q(\mathbf{z})||p(\mathbf{z}|\mathbf{x}, \theta^*)) = p(\mathbf{z}|\mathbf{x}, \theta^*);$$

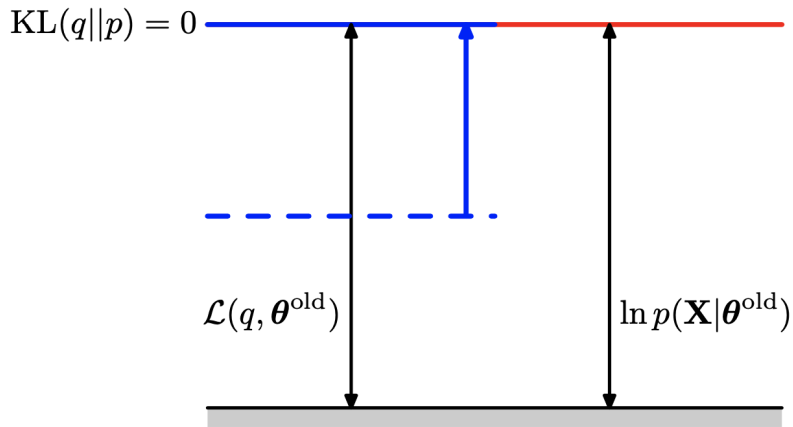
- **M-step** ($\mathcal{L}(q, \theta) \to \max_\theta$)

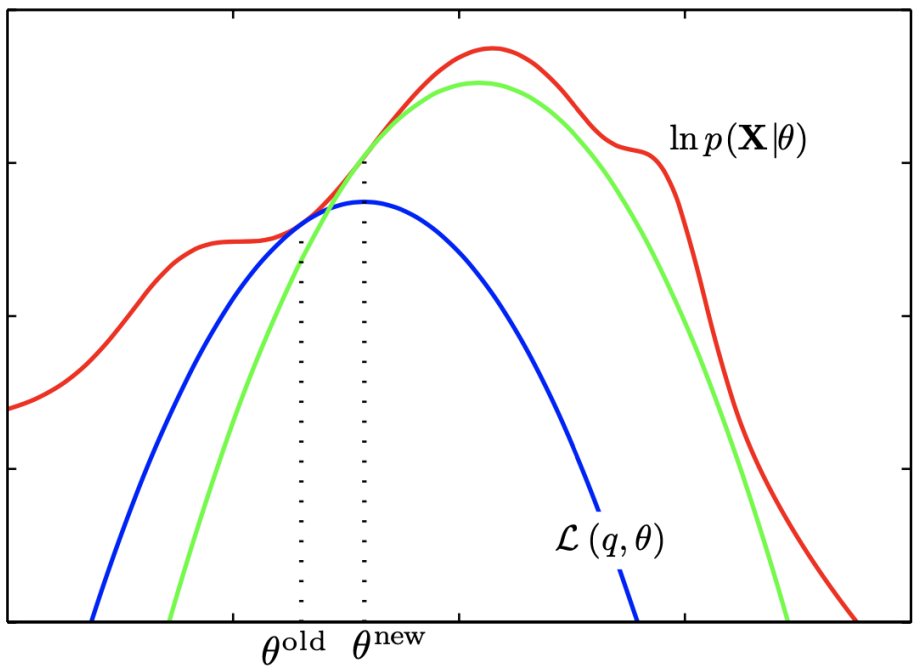
$$\theta^* = \arg\max_\theta \mathcal{L}(q^*, \theta);$$

- Repeat E-step and M-step until convergence.

EM-algorithm illustration







Bishop C. Pattern Recognition and Machine Learning, 2006

Outline

Amortized variational inference

E-step

$$q(\mathbf{z}) = \arg\max_q \mathcal{L}(q, \boldsymbol{\theta}^*) = \arg\min_q KL(q||p) = p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta}^*).$$

- $q(\mathbf{z})$ approximates true posterior distribution $p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta}^*)$, that is why it is called **variational posterior**;
- $p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta}^*)$ could be **intractable**;
- $q(\mathbf{z})$ is different for each object $\mathbf{x}$.

Idea

Restrict a family of all possible distributions $q(\mathbf{z})$ to a parametric class $q(\mathbf{z}|\mathbf{x}, \boldsymbol{\phi})$ conditioned on samples $\mathbf{x}$ with parameters $\boldsymbol{\phi}$.

Variational Bayes

- E-step

$$\boldsymbol{\phi}_k = \boldsymbol{\phi}_{k-1} + \eta \cdot \nabla_{\boldsymbol{\phi}} \mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta}_{k-1})|_{\boldsymbol{\phi}=\boldsymbol{\phi}_{k-1}}$$

- M-step

$$\boldsymbol{\theta}_k = \boldsymbol{\theta}_{k-1} + \eta \cdot \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\phi}_k, \boldsymbol{\theta})|_{\boldsymbol{\theta}=\boldsymbol{\theta}_{k-1}}$$

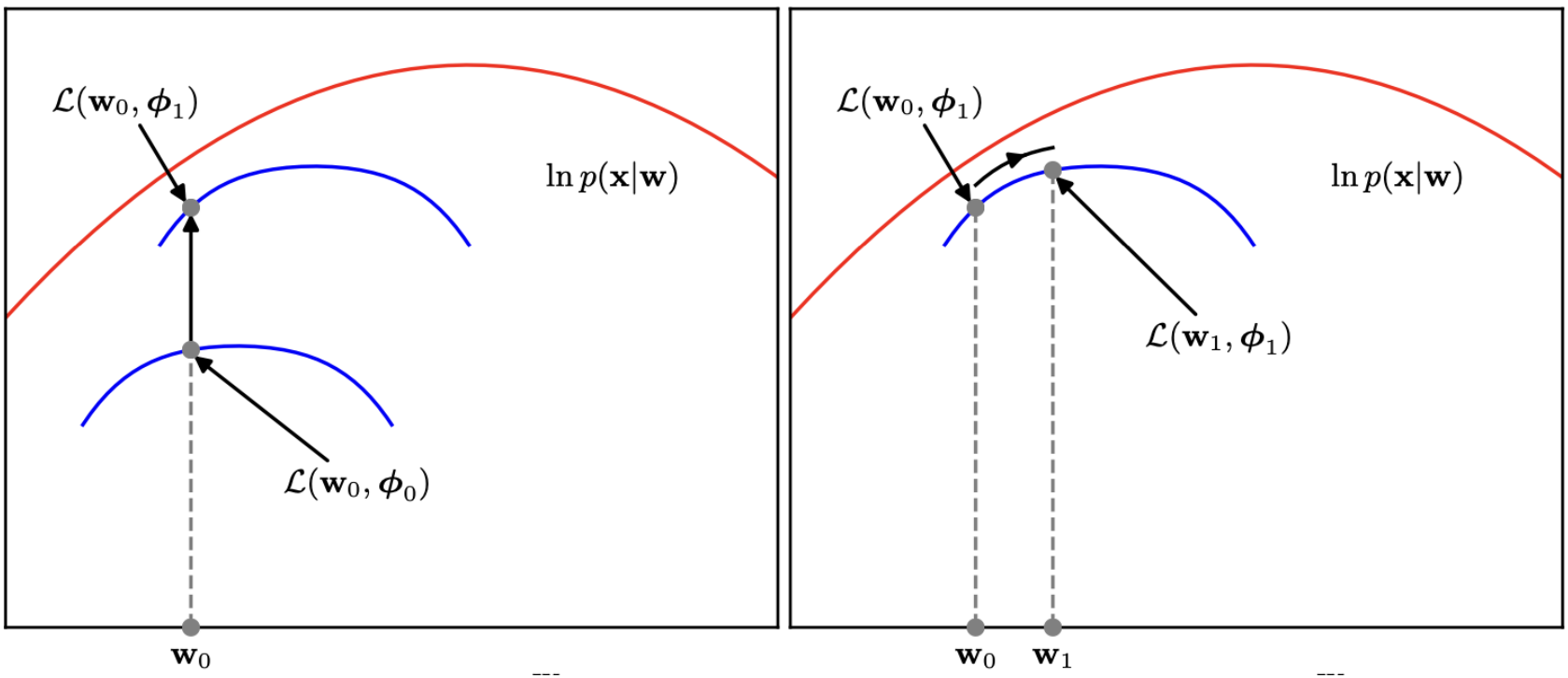
Variational EM illustration

- E-step

$$\phi_k = \phi_{k-1} + \eta \nabla_{\phi} \mathcal{L}(\phi, \theta_{k-1})|_{\phi=\phi_{k-1}}$$

- M-step

$$\theta_k = \theta_{k-1} + \eta \nabla_{\theta} \mathcal{L}(\phi_k, \theta)|_{\theta=\theta_{k-1}}$$



Bishop C. Deep Learning: Foundations and Concepts, 2024

Variational EM-algorithm

ELBO

$$\log p(\mathbf{x}|\boldsymbol{\theta}) = \mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta}) + KL(q(\mathbf{z}|\mathbf{x}, \boldsymbol{\phi})||p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta})) \geq \mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta}).$$

- **E-step**

$$\boldsymbol{\phi}_k = \boldsymbol{\phi}_{k-1} + \eta \cdot \nabla_{\boldsymbol{\phi}} \mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta}_{k-1})|_{\boldsymbol{\phi}=\boldsymbol{\phi}_{k-1}},$$

  where $\boldsymbol{\phi}$ – parameters of variational posterior distribution $q(\mathbf{z}|\mathbf{x}, \boldsymbol{\phi})$.

- **M-step**

$$\boldsymbol{\theta}_k = \boldsymbol{\theta}_{k-1} + \eta \cdot \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\phi}_k, \boldsymbol{\theta})|_{\boldsymbol{\theta}=\boldsymbol{\theta}_{k-1}},$$

  where $\boldsymbol{\theta}$ – parameters of the generative distribution $p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta})$.

Now all that is left is to obtain gradients: $\nabla_{\boldsymbol{\phi}} \mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta})$, $\nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta})$.
Challenge: Number of samples n could be huge (we need derive the **unbiased** stochastic gradients).

Outline

1. EM-algorithm
   - Amortized inference
   - ELBO gradients, reparametrization trick
2. Variational autoencoder (VAE)
3. Data dequantization

ELBO gradients, (M-step, $\nabla_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta})$)

$$\mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta}) = \mathbb{E}_{q(\mathbf{z}|\mathbf{x},\boldsymbol{\phi})}\left[\log p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta}) - \log \frac{q(\mathbf{z}|\mathbf{x}, \boldsymbol{\phi})}{p(\mathbf{z})}\right] \to \max_{\boldsymbol{\phi},\boldsymbol{\theta}}.$$

M-step: $\nabla_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta})$

$$\nabla_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta}) = \int q(\mathbf{z}|\mathbf{x}, \boldsymbol{\phi})\nabla_{\boldsymbol{\theta}} \log p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta})d\mathbf{z} \approx$$

$$\approx \nabla_{\boldsymbol{\theta}} \log p(\mathbf{x}|\mathbf{z}^*, \boldsymbol{\theta}), \quad \mathbf{z}^* \sim q(\mathbf{z}|\mathbf{x}, \boldsymbol{\phi}).$$

Naive Monte-Carlo estimation

$$p(\mathbf{x}|\boldsymbol{\theta}) = \int p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta})p(\mathbf{z})d\mathbf{z} \approx \frac{1}{K}\sum_{k=1}^{K} p(\mathbf{x}|\mathbf{z}_k, \boldsymbol{\theta}), \quad \mathbf{z}^* \sim p(\mathbf{z}).$$

The variational posterior $q(\mathbf{z}|\mathbf{x}, \boldsymbol{\phi})$ assigns typically more probability mass in a smaller region than the prior $p(\mathbf{z})$.

---

image credit: https://jmtomczak.github.io/blog/4/4_VAE.html

ELBO gradients, (E-step, $\nabla_\phi \mathcal{L}(\phi, \theta)$)

E-step: $\nabla_\phi \mathcal{L}(\phi, \theta)$

Difference from M-step: density function $q(\mathbf{z}|\mathbf{x}, \phi)$ depends on the parameters ϕ, it is impossible to use the Monte-Carlo estimation:

$$\nabla_\phi \mathcal{L}(\phi, \theta) = \nabla_\phi \int q(\mathbf{z}|\mathbf{x}, \phi) \left[\log p(\mathbf{x}|\mathbf{z}, \theta) - \log \frac{q(\mathbf{z}|\mathbf{x}, \phi)}{p(\mathbf{z})} \right] d\mathbf{z}$$

$$\neq \int q(\mathbf{z}|\mathbf{x}, \phi) \nabla_\phi \left[\log p(\mathbf{x}|\mathbf{z}, \theta) - \log \frac{q(\mathbf{z}|\mathbf{x}, \phi)}{p(\mathbf{z})} \right] d\mathbf{z}$$

Reparametrization trick (LOTUS trick)

- $r(x) = \mathcal{N}(0, 1)$, $y = \sigma \cdot x + \mu$, $p(y|\theta) = \mathcal{N}(\mu, \sigma^2)$, $\theta = [\mu, \sigma]$.
- $\epsilon^* \sim r(\epsilon), \quad \mathbf{z} = g_\phi(\mathbf{x}, \epsilon), \quad \mathbf{z} \sim q(\mathbf{z}|\mathbf{x}, \phi)$

$$\nabla_\phi \int q(\mathbf{z}|\mathbf{x}, \phi) f(\mathbf{z}) d\mathbf{z} = \nabla_\phi \int r(\epsilon) f(\mathbf{z}) d\epsilon \Big|_{\mathbf{z}=g_\phi(\mathbf{x},\epsilon)}$$

$$= \int r(\epsilon) \nabla_\phi f(g_\phi(\mathbf{x}, \epsilon)) d\epsilon \approx \nabla_\phi f(g_\phi(\mathbf{x}, \epsilon^*))$$

ELBO gradient (E-step, $\nabla_\phi \mathcal{L}(\phi, \theta)$)

$$\nabla_\phi \mathcal{L}(\phi, \theta) = \nabla_\phi \int q(\mathbf{z}|\mathbf{x}, \phi) \log p(\mathbf{x}|\mathbf{z}, \theta) d\mathbf{z} - \nabla_\phi \mathrm{KL}(q(\mathbf{z}|\mathbf{x}, \phi)||p(\mathbf{z}))$$

$$= \int r(\epsilon) \nabla_\phi \log p(\mathbf{x}|g_\phi(\mathbf{x}, \epsilon), \theta) d\epsilon - \nabla_\phi \mathrm{KL}(q(\mathbf{z}|\mathbf{x}, \phi)||p(\mathbf{z}))$$

$$\approx \nabla_\phi \log p(\mathbf{x}|g_\phi(\mathbf{x}, \epsilon^*), \theta) - \nabla_\phi \mathrm{KL}(q(\mathbf{z}|\mathbf{x}, \phi)||p(\mathbf{z}))$$

Variational assumption

$$r(\epsilon) = \mathcal{N}(0, \mathbf{I}); \quad q(\mathbf{z}|\mathbf{x}, \phi) = \mathcal{N}(\boldsymbol{\mu}_\phi(\mathbf{x}), \boldsymbol{\sigma}^2_\phi(\mathbf{x})).$$

$$\mathbf{z} = g_\phi(\mathbf{x}, \epsilon) = \boldsymbol{\sigma}_\phi(\mathbf{x}) \odot \epsilon + \boldsymbol{\mu}_\phi(\mathbf{x}).$$

Here $\boldsymbol{\mu}_\phi(\cdot), \boldsymbol{\sigma}_\phi(\cdot)$ are parameterized functions (outputs of neural network).

- $p(\mathbf{z})$ – prior distribution on latent variables $\mathbf{z}$. We could specify any distribution that we want. Let say $p(\mathbf{z}) = \mathcal{N}(0, \mathbf{I})$.
- $p(\mathbf{x}|\mathbf{z}, \theta)$ - generative distibution. Since it is a parameterized function let it be neural network with parameters θ.

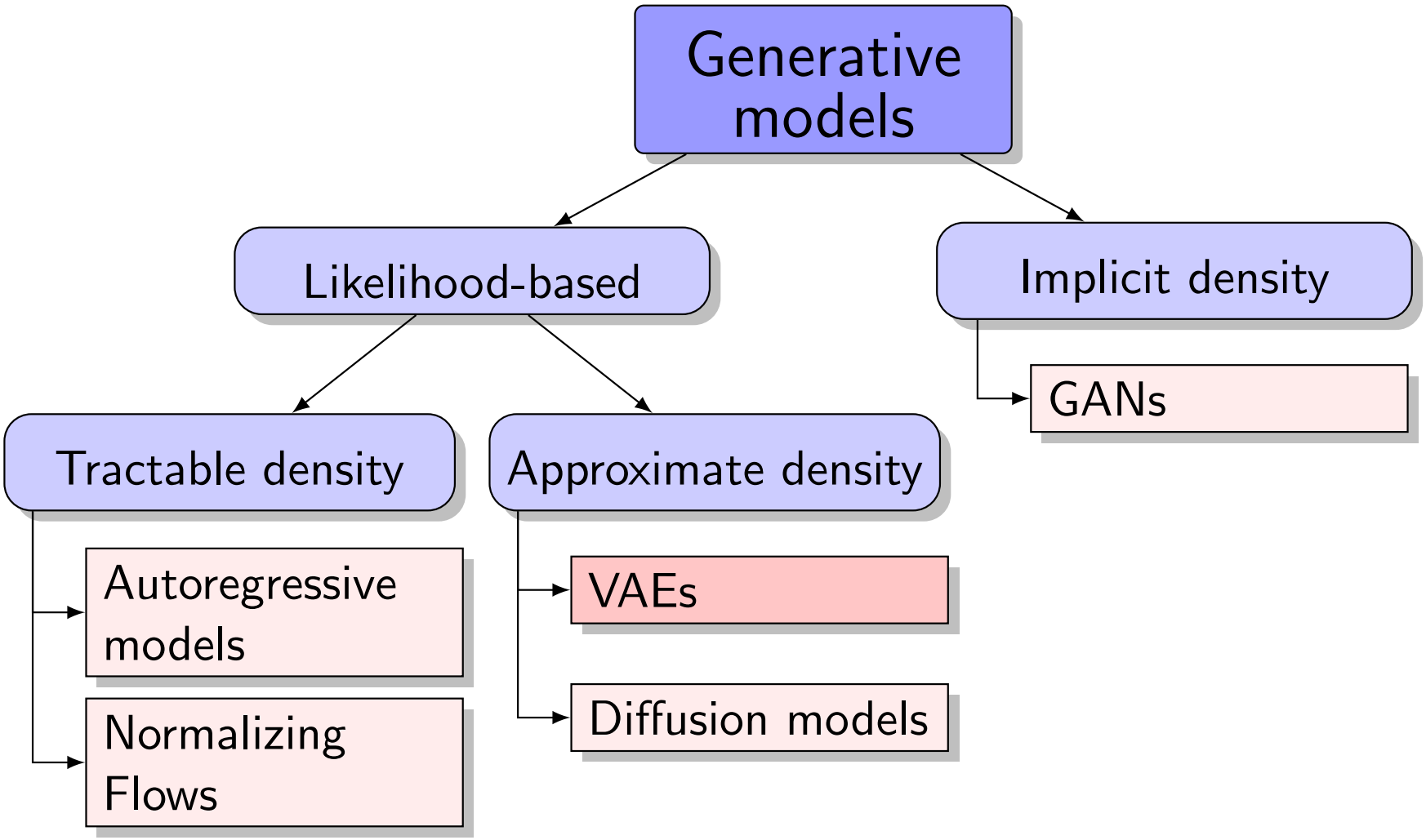
Outline

1. EM-algorithm
   Amortized inference
   ELBO gradients, reparametrization trick

2. Variational autoencoder (VAE)

3. Data dequantization

Generative models zoo



Variational autoencoder (VAE)

Final EM-algorithm

- pick random sample $\mathbf{x}_i, i \sim U[1, n]$.
- compute the objective:

$$\epsilon^* \sim r(\epsilon); \quad \mathbf{z}^* = g_\phi(\mathbf{x}, \epsilon^*);$$

$$\mathcal{L}(\phi, \theta) \approx \log p(\mathbf{x}|\mathbf{z}^*, \theta) - KL(q(\mathbf{z}^*|\mathbf{x}, \phi)||p(\mathbf{z}^*)).$$

- compute a stochastic gradients w.r.t. ϕ and θ

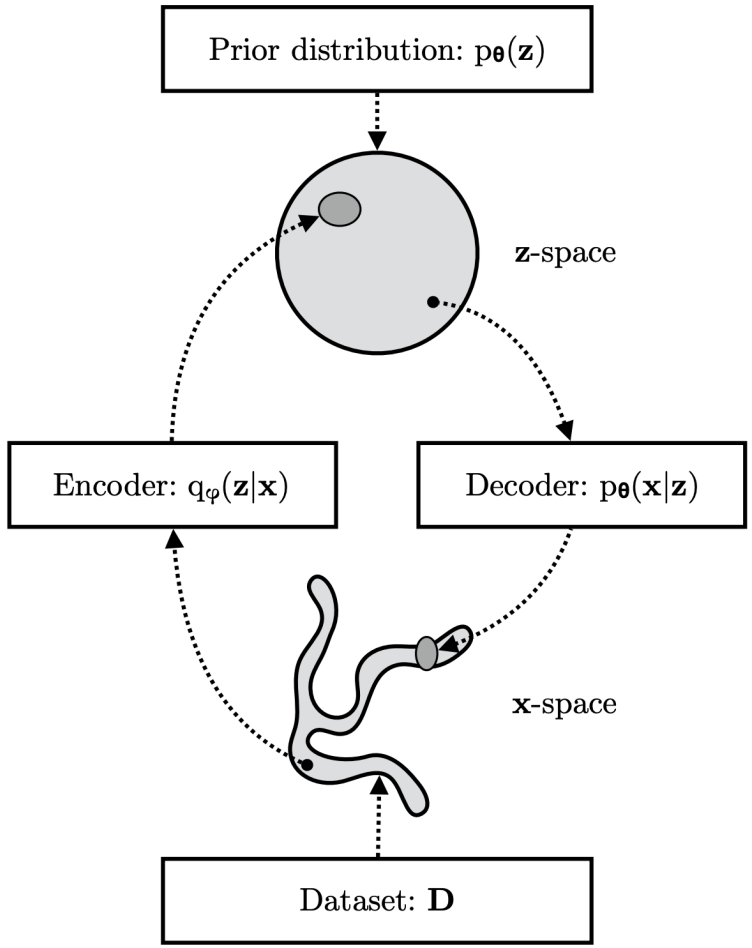
$$\nabla_\phi \mathcal{L}(\phi, \theta) \approx \nabla_\phi \log p(\mathbf{x}|g_\phi(\mathbf{x}, \epsilon^*), \theta) - \nabla_\phi \mathrm{KL}(q(\mathbf{z}|\mathbf{x}, \phi)||p(\mathbf{z}));$$
$$\nabla_\theta \mathcal{L}(\phi, \theta) \approx \nabla_\theta \log p(\mathbf{x}|\mathbf{z}^*, \theta).$$

- update θ, ϕ according to the selected optimization method (SGD, Adam, etc):

$$\phi := \phi + \eta \cdot \nabla_\phi \mathcal{L}(\phi, \theta),$$
$$\theta := \theta + \eta \cdot \nabla_\theta \mathcal{L}(\phi, \theta).$$

Variational autoencoder (VAE)

- VAE learns stochastic mapping between $\mathbf{x}$-space, from complicated distribution $\pi(\mathbf{x})$, and a latent $\mathbf{z}$-space, with simple distribution.
- The generative model learns a joint distribution $p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta}) = p(\mathbf{z})p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta})$, with a prior distribution $p(\mathbf{z})$, and a stochastic decoder $p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta})$.
- The stochastic encoder $q(\mathbf{z}|\mathbf{x}, \boldsymbol{\phi})$ (inference model), approximates the true but intractable posterior $p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta})$ of the generative model.



Kingma D. P., Welling M. An introduction to variational autoencoders, 2019

Variational Autoencoder

$$\mathcal{L}(\phi, \theta) = \mathbb{E}_{q(\mathbf{z}|\mathbf{x},\phi)} \left[\log p(\mathbf{x}|\mathbf{z}, \theta) - \log \frac{q(\mathbf{z}|\mathbf{x}, \phi)}{p(\mathbf{z})} \right] \to \max_{\phi,\theta}.$$

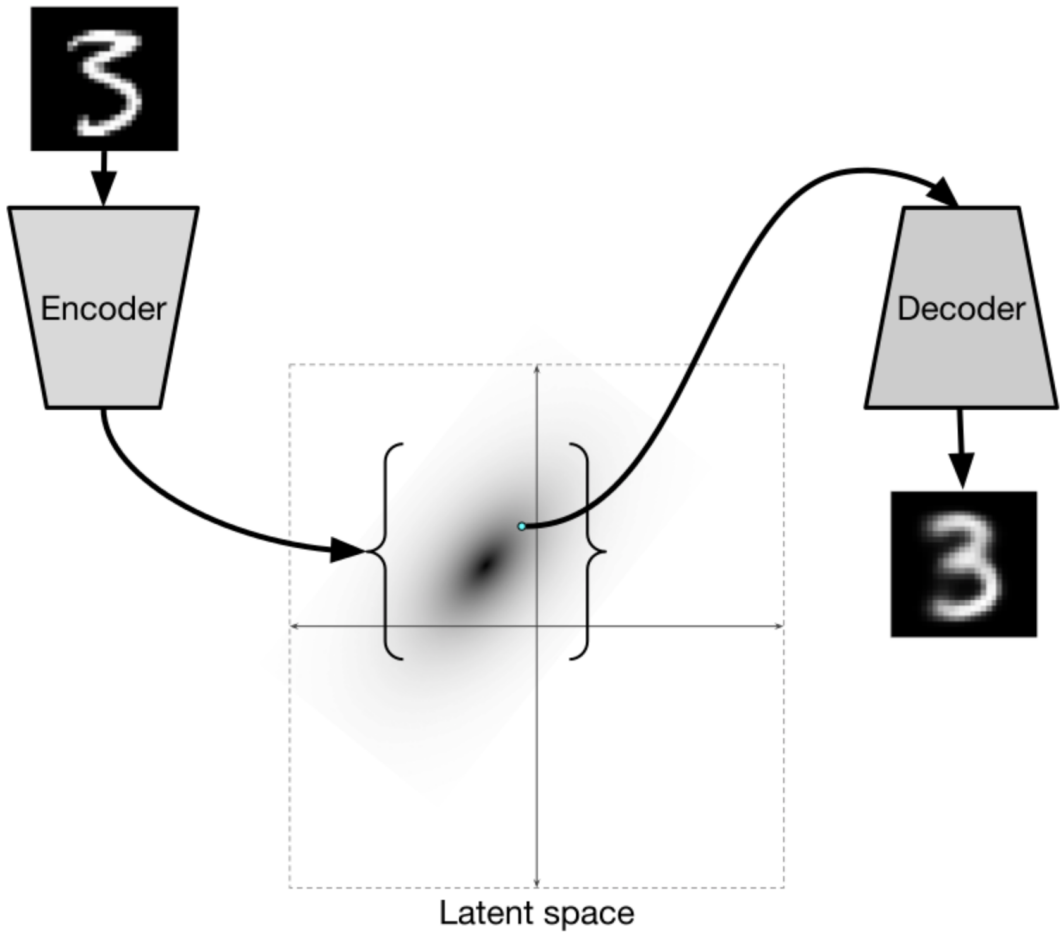


image credit: http://ijdykeman.github.io/ml/2016/12/21/cvae.html

Variational autoencoder (VAE)

- Encoder $q(\mathbf{z}|\mathbf{x}, \phi) = \text{NN}_e(\mathbf{x}, \phi)$ outputs $\boldsymbol{\mu}_\phi(\mathbf{x})$ and $\boldsymbol{\sigma}_\phi(\mathbf{x})$.
- Decoder $p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta}) = \text{NN}_d(\mathbf{z}, \boldsymbol{\theta})$ outputs parameters of the sample distribution.

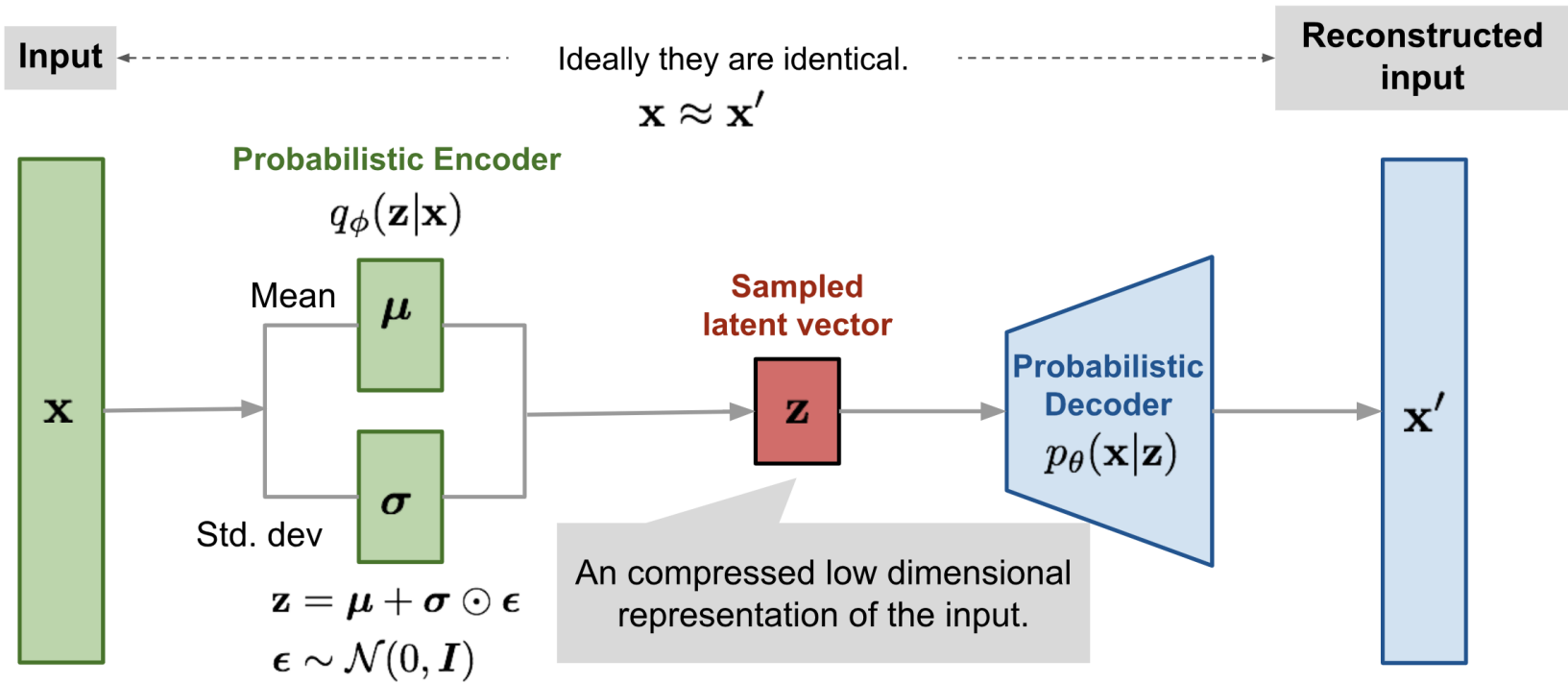


image credit:
https://lilianweng.github.io/lil-log/2018/08/12/from-autoencoder-to-beta-vae.html

Outline

1. EM-algorithm
   Amortized inference
   ELBO gradients, reparametrization trick

2. Variational autoencoder (VAE)

3. Data dequantization

Discrete data vs continuous model

Let our data $\mathbf{y}$ comes from discrete distribution $\Pi(\mathbf{y})$ and we have continuous model $p(\mathbf{x}|\boldsymbol{\theta}) = \text{NN}(\mathbf{x}, \boldsymbol{\theta})$.

- Images (and not only images) are discrete data, pixels lie in the integer domain ($\{0, 255\}$).
- By fitting a continuous density model $p(\mathbf{x}|\boldsymbol{\theta})$ to discrete data $\Pi(\mathbf{y})$, one can produce a degenerate solution with all probability mass on discrete values.

Discrete model

- Use **discrete** model (e.x. $P(\mathbf{y}|\boldsymbol{\theta}) = \text{Cat}(\boldsymbol{\pi}(\boldsymbol{\theta}))$).
- Minimize any suitable divergence measure $D(\Pi, P)$.
- NF works only with continuous data $\mathbf{x}$ (there are discrete NF, see papers below).
- If pixel value is not presented in the train data, it won't be predicted.

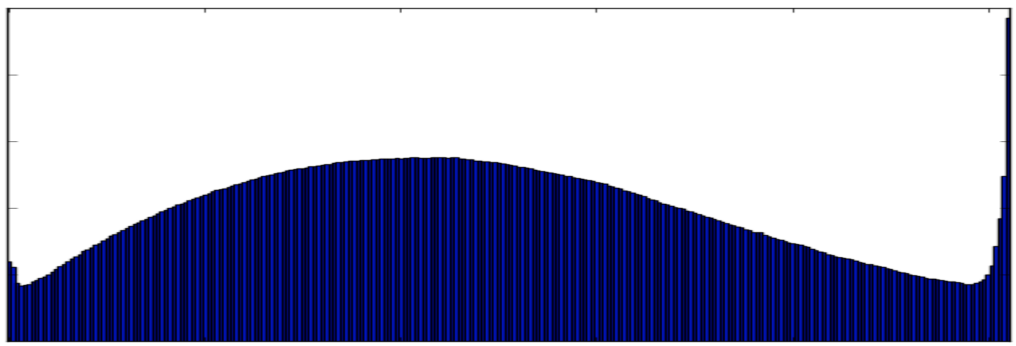
Hoogeboom E. et al. Integer discrete flows and lossless compression, 2019
Tran D. et al. Discrete flows: Invertible generative models of discrete data, 2019

Discrete data vs continuous model

Continuous model

- Use **continuous** model (e.x. $p(\mathbf{x}|\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{x}), \boldsymbol{\sigma}^2_{\boldsymbol{\theta}}(\mathbf{x}))$), but
  - **discretize** model (make the model outputs discrete): transform $p(\mathbf{x}|\boldsymbol{\theta})$ to $P(\mathbf{y}|\boldsymbol{\theta})$;
  - **dequantize** data (make the data continuous): transform $\Pi(\mathbf{y})$ to $\pi(\mathbf{x})$.
- Continuous distribution knows numerical relationships.

CIFAR-10 pixel values distribution



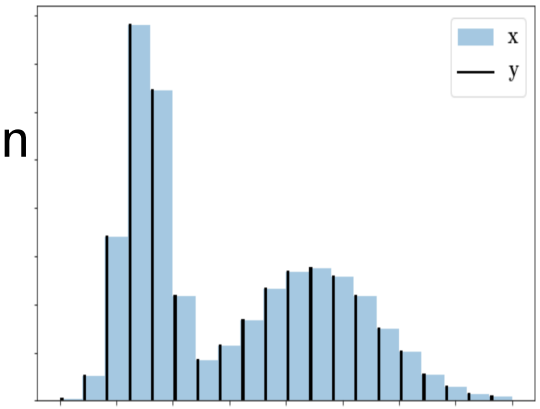
Salimans T. et al. PixelCNN++: Improving the PixelCNN with Discretized Logistic Mixture Likelihood and Other Modifications, 2017

Uniform dequantization

Let dequantize discrete distribution $\Pi(\mathbf{y})$ to continuous distribution $\pi(\mathbf{x})$ in the following way: $\mathbf{x} = \mathbf{y} + \mathbf{u}$, where $\mathbf{u} \sim U[0, 1]$.

Theorem

Fitting continuous model $p(\mathbf{x}|\boldsymbol{\theta})$ on uniformly dequantized data is equivalent to maximization of a lower bound on log-likelihood for a discrete model:



$$P(\mathbf{y}|\boldsymbol{\theta}) = \int_{U[0,1]} p(\mathbf{y} + \mathbf{u}|\boldsymbol{\theta}) d\mathbf{u}$$

Proof

$$\mathbb{E}_\pi \log p(\mathbf{x}|\boldsymbol{\theta}) = \int \pi(\mathbf{x}) \log p(\mathbf{x}|\boldsymbol{\theta}) d\mathbf{x} = \sum \Pi(\mathbf{y}) \int_{U[0,1]} \log p(\mathbf{y}+\mathbf{u}|\boldsymbol{\theta}) d\mathbf{u} \leq$$

$$\leq \sum \Pi(\mathbf{y}) \log \int_{U[0,1]} p(\mathbf{y} + \mathbf{u}|\boldsymbol{\theta}) d\mathbf{u} =$$

$$= \sum \Pi(\mathbf{y}) \log P(\mathbf{y}|\boldsymbol{\theta}) = \mathbb{E}_\Pi \log P(\mathbf{y}|\boldsymbol{\theta}).$$

Theis L., Oord A., Bethge M. A note on the evaluation of generative models. 2015

Summary

- The general variational EM algorithm maximizes ELBO objective for LVM model to find MLE for parameters $\boldsymbol{\theta}$.
- Amortized variational inference allows to efficiently compute the stochastic gradients for ELBO using Monte-Carlo estimation.
- The reparametrization trick gets unbiased gradients w.r.t to the variational posterior distribution $q(\mathbf{z}|\mathbf{x}, \boldsymbol{\phi})$.
- The VAE model is an LVM with two neural network: stochastic encoder $q(\mathbf{z}|\mathbf{x}, \boldsymbol{\phi})$ and stochastic decoder $p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta})$.
- Lots of data are discrete. We able to discretize the model or to dequantize our data to use continuous model.
- Uniform dequantization helps to make discrete data continuous. It gives us lower bound on the log-likelihood.