Deep Generative Models

Lecture 6

Roman Isachenko

Moscow Institute of Physics and Technology

Autumn, 2021

Recap of previous lecture

Flow log-likelihood

$$\log p(\mathbf{x}|\boldsymbol{\theta}) = \log p(f(\mathbf{x}, \boldsymbol{\theta})) + \log \left| \det \left(\frac{\partial f(\mathbf{x}, \boldsymbol{\theta})}{\partial \mathbf{x}} \right) \right|$$

The main challenge is a determinant of the Jacobian.

Residual flows: planar/Sylvester

$$g(\mathbf{z}, \boldsymbol{\theta}) = \mathbf{z} + \mathbf{u}\, h(\mathbf{w}^T \mathbf{z} + b); \quad g(\mathbf{z}, \boldsymbol{\theta}) = \mathbf{z} + \mathbf{A}\, h(\mathbf{B}\mathbf{z} + \mathbf{b}).$$

Matrix determinant lemma for calculating the Jacobian.

Autoregressive flows

$$x_i = \tau(z_i, c(\mathbf{z}_{1:i-1})) \quad \Leftrightarrow \quad z_i = \tau^{-1}(x_i, c(\mathbf{z}_{1:i-1}))$$

Jacobian is triangular.

Rezende D. J., Mohamed S. Variational Inference with Normalizing Flows, 2015
Berg R. et al. Sylvester normalizing flows for variational inference, 2018
Papamakarios G., Pavlakou T., Murray I. Masked Autoregressive Flow for Density Estimation, 2017

Recap of previous lecture

Gaussian autoregressive flow (MAF)

$$\mathbf{x} = g(\mathbf{z}, \boldsymbol{\theta}) \quad \Rightarrow \quad x_i = \sigma_i(\mathbf{x}_{1:i-1}) \cdot z_i + \mu_i(\mathbf{x}_{1:i-1}).$$

$$\mathbf{z} = f(\mathbf{x}, \boldsymbol{\theta}) \quad \Rightarrow \quad z_i = (x_i - \mu_i(\mathbf{x}_{1:i-1})) \cdot \frac{1}{\sigma_i(\mathbf{x}_{1:i-1})}.$$

Generation function $g(\mathbf{z}, \boldsymbol{\theta})$ is **sequential**. Inference function $f(\mathbf{x}, \boldsymbol{\theta})$ is **not sequential**.

Inverse autoregressive flow (IAF)

$$\mathbf{x} = g(\mathbf{z}, \boldsymbol{\theta}) \quad \Rightarrow \quad x_i = \tilde{\sigma}_i(\mathbf{z}_{1:i-1}) \cdot z_i + \tilde{\mu}_i(\mathbf{z}_{1:i-1})$$

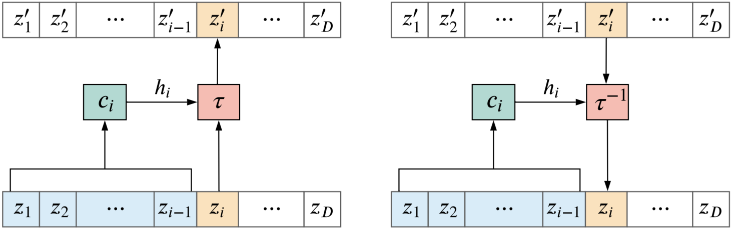
$$\mathbf{z} = f(\mathbf{x}, \boldsymbol{\theta}) \quad \Rightarrow \quad z_i = (x_i - \tilde{\mu}_i(\mathbf{z}_{1:i-1})) \cdot \frac{1}{\tilde{\sigma}_i(\mathbf{z}_{1:i-1})}.$$

Papamakarios G., Pavlakou T., Murray I. Masked Autoregressive Flow for Density Estimation, 2017

Kingma D. P. et al. Improving Variational Inference with Inverse Autoregressive Flow, 2016

Recap of previous lecture

Autoregressive flows



RealNVP: Affine coupling law

$$\begin{cases} \mathbf{z}_{1:d} = \mathbf{x}_{1:d}; \\ \mathbf{z}_{d:m} = \tau(\mathbf{x}_{d:m}, c(\mathbf{x}_{1:d})); \end{cases} \Leftrightarrow \quad \begin{cases} \mathbf{x}_{1:d} = \mathbf{z}_{1:d}; \\ \mathbf{x}_{d:m} = \tau^{-1}(\mathbf{z}_{d:m}, c(\mathbf{z}_{1:d})). \end{cases}$$

Dinh L., Krueger D., Bengio Y. NICE: Non-linear Independent Components Estimation, 2014

Dinh L., Sohl-Dickstein J., Bengio S. Density estimation using Real NVP, 2016

Linear flows

RealNVP

$$\begin{cases} \mathbf{z}_{1:d} = \mathbf{x}_{1:d}; \\ \mathbf{z}_{d:m} = \tau(\mathbf{x}_{d:m}, c(\mathbf{x}_{1:d})); \end{cases} \quad \Leftrightarrow \quad \begin{cases} \mathbf{x}_{1:d} = \mathbf{z}_{1:d}; \\ \mathbf{x}_{d:m} = \tau^{-1}(\mathbf{z}_{d:m}, c(\mathbf{z}_{1:d})). \end{cases}$$

- First step is a **split** operator which decouples a variable into 2 subparts: $\mathbf{x}_1$ and $\mathbf{x}_2$ (usualy channel-wise).
- We should **permute** components between different layers.

$$\mathbf{z} = \mathbf{W}\mathbf{x}, \quad \mathbf{W} \in \mathbb{R}^{m \times m}$$

In general, we need $O(m^3)$ to invert matrix.

Invertibility

- Diagonal matrix $O(m)$.
- Triangular matrix $O(m^2)$.
- It is impossible to parametrize all invertible matrices.

Linear flows

$$\mathbf{z} = \mathbf{W}\mathbf{x}, \quad \mathbf{W} \in \mathbb{R}^{m \times m}$$

Matrix decompositions

- LU-decomposition
$$\mathbf{W} = \mathbf{P}\mathbf{L}\mathbf{U},$$
where $\mathbf{P}$ is a permutation matrix, $\mathbf{L}$ is lower triangular with positive diagonal, $\mathbf{U}$ is upper triangular with positive diagonal.
- QR-decomposition
$$\mathbf{W} = \mathbf{Q}\mathbf{R},$$
where $\mathbf{Q}$ is an orthogonal matrix, $\mathbf{R}$ is an upper triangular matrix with positive diagonal.

Kingma D. P., Dhariwal P. Glow: Generative Flow with Invertible 1x1 Convolutions, 2018
Hoogeboom E., Van Den Berg R., and Welling M. Emerging convolutions for generative normalizing flows, 2019

Glow samples

Kingma D. P., Dhariwal P. Glow: Generative Flow with Invertible 1x1 Convolutions, 2018

MAF/IAF pros and cons

MAF

- Sampling is slow.
- Likelihood evaluation is fast.

IAF

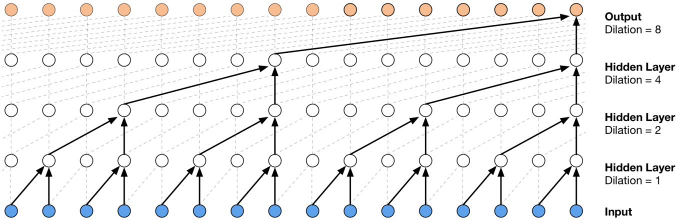
- Sampling is fast.
- Likelihood evaluation is slow.

How to take the best of both worlds?

WaveNet

Autoregressive model with caused dilated convolutions for raw audio waveforms generation.

$$p(\mathbf{x}|\boldsymbol{\theta}) = \prod_{t=1}^{T} p(x_t|\mathbf{x}_{1:t-1}, \boldsymbol{\theta}).$$



Oord A. et al. Wavenet: A generative model for raw audio, 2016

Parallel WaveNet

- 24kHz instead of 16kHz using increased dilated convolution filter size from 2 to 3.
- 16-bit signals with mixture of logistics instead of 8-bit signal with 256-way categorical distribution.

Probability density distillation

1. Train usual WaveNet (MAF) via MLE (teacher network).
2. Train IAF WaveNet (student network), which attempts to match the probability of its own samples under the distribution learned by the teacher.

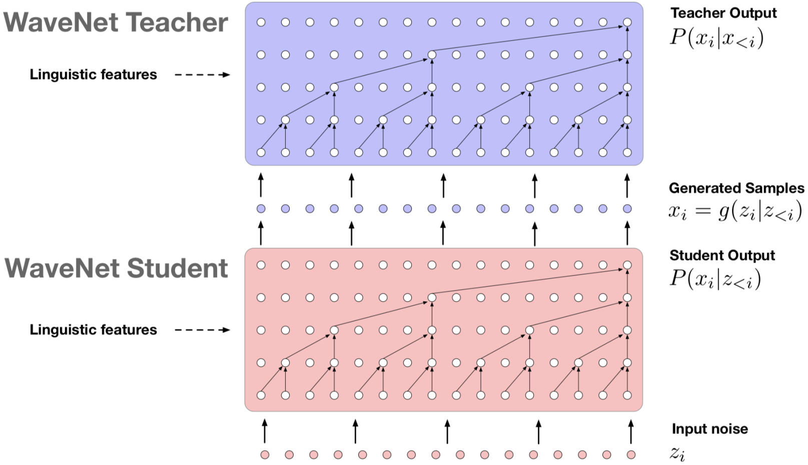
Student objective

$$KL(p_s||p_t) = H(p_s, p_t) - H(p_s).$$

More than 1000x speed-up relative to original WaveNet!

Oord A. et al. Parallel WaveNet: Fast High-Fidelity Speech Synthesis, 2017

Parallel WaveNet



Oord A. et al. Parallel WaveNet: Fast High-Fidelity Speech Synthesis, 2017

Likelihood-based models

Exact likelihood evaluation

- Autoregressive models (PixelCNN, WaveNet);
- Flow models (NICE, RealNVP, Glow).

Approximate likelihood evaluation

- Latent variable models (VAE).

What are the pros and cons of each of them?

VAE recap

ELBO

$$\log p(\mathbf{x}|\boldsymbol{\theta}) \geq \mathcal{L}(\phi, \boldsymbol{\theta}) = \mathbb{E}_{q(\mathbf{z}|\mathbf{x},\phi)} \log \frac{p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta})}{q(\mathbf{z}|\mathbf{x}, \phi)} \rightarrow \max_{\phi,\boldsymbol{\theta}}.$$

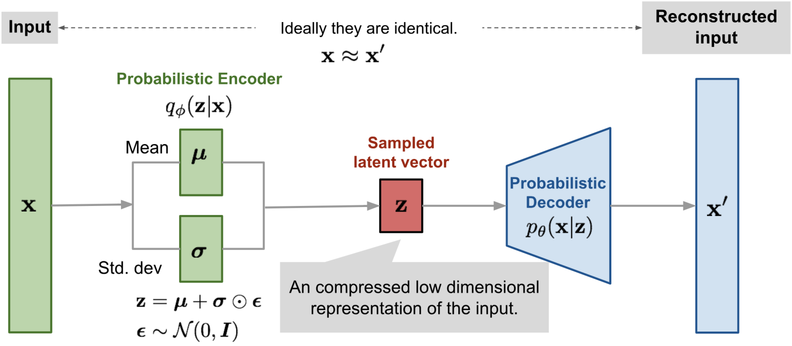


image credit: https://lilianweng.github.io/lil-log/2018/08/12/from-autoencoder-to-beta-vae.html

VAE limitations

- Poor variational posterior distribution (encoder)

$$q(\mathbf{z}|\mathbf{x}, \phi) = \mathcal{N}(\mathbf{z}|\boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x}), \boldsymbol{\sigma}^2_{\boldsymbol{\phi}}(\mathbf{x})).$$

- Poor prior distribution

$$p(\mathbf{z}) = \mathcal{N}(0, \mathbf{I}).$$

- Poor probabilistic model (decoder)

$$p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{z}), \boldsymbol{\sigma}^2_{\boldsymbol{\theta}}(\mathbf{z})) \quad (\text{or Softmax}(\pi(\mathbf{z}))).$$

- Loose lower bound

$$\log p(\mathbf{x}|\boldsymbol{\theta}) - \mathcal{L}(q, \boldsymbol{\theta}) = (?).$$

Posterior collapse

Representation learning

"Identifies and disentangles the underlying causal factors of the data, so that it becomes easier to understand the data, to classify it, or to perform other tasks".

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

If the decoder model $p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta})$ is powerful enough to model $p(\mathbf{x}|\boldsymbol{\theta})$ the latent variables $\mathbf{z}$ becomes irrelevant.

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

Early in the training the approximate posterior $q(\mathbf{z}|\mathbf{x})$ carries little information about $\mathbf{x}$ and the model sets the posterior to the prior to avoid paying any cost $KL(q(\mathbf{z}|\mathbf{x})||p(\mathbf{z}))$.

PixelVAE

LVM

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

- More powerful $p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta})$ leads to more powerful generative model $p(\mathbf{x}|\boldsymbol{\theta})$.
- Too powerful $p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta})$ could lead to posterior collapse, where variational posterior $q(\mathbf{z}|\mathbf{x})$ will not carry any information about data and close to prior $p(\mathbf{z})$.

How to make the generative model $p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta})$ more powerful?

Autoregressive decoder

$$p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta}) = \prod_{i=1}^{n} p(x_i|\mathbf{x}_{1:i-1}, \mathbf{z}, \boldsymbol{\theta})$$

Gulrajani I. et al. PixelVAE: A Latent Variable Model for Natural Images, 2016

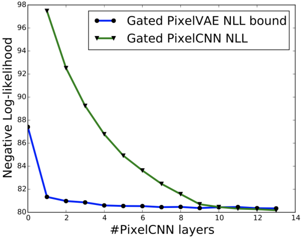
PixelVAE

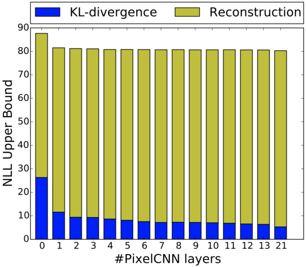
Autoregressive decoder

$$p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta}) = \prod_{i=1}^{n} p(x_i|\mathbf{x}_{1:i-1}, \mathbf{z}, \boldsymbol{\theta})$$

- Global structure is captured by latent variables.
- Local statistics are captured by limited receptive field autoregressive model.

MNIST results





Gulrajani I. et al. PixelVAE: A Latent Variable Model for Natural Images, 2016

PixelVAE

MNIST

Model	NLL Test
DRAW (Gregor et al., 2016)	$\lesssim 80.97$
Discrete VAE (Rolfe, 2016)	$= 81.01$
IAF VAE (Kingma et al., 2016)	≈ 79.88
PixelCNN (van den Oord et al., 2016a)	$= 81.30$
PixelRNN (van den Oord et al., 2016a)	$= 79.20$
Convolutional VAE	$\lesssim 87.41$
PixelVAE	$\lesssim 80.64$
Gated PixelCNN (our implementation)	$= 80.10$
Gated PixelVAE	≈ 79.48 ($\lesssim 80.02$)
Gated PixelVAE without upsampling	$\approx$ **79.02** ($\lesssim 79.66$)

ImageNet 64x64

Model	NLL Validation (Train)
Convolutional DRAW (Gregor et al., 2016)	$\lesssim 4.10$ (4.04)
Real NVP (Dinh et al., 2016)	$= 4.01$ (3.93)
PixelRNN (van den Oord et al., 2016a)	$= 3.63$ (3.57)
Gated PixelCNN (van den Oord et al., 2016b)	$=$ **3.57** (3.48)
Hierarchical PixelVAE	$\lesssim 3.66$ (3.59)

Gulrajani I. et al. PixelVAE: A Latent Variable Model for Natural Images, 2016

Decoder weakening

- Powerful decoder $p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta})$ makes the model expressive, but posterior collapse is possible.
- PixelVAE model uses the autoregressive PixelCNN model with small number of layers to limit receptive field.

How to force the model encode information about $\mathbf{x}$ into $\mathbf{z}$?

KL annealing

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

Start training with $\beta = 0$, increase it until $\beta = 1$ during training.

Free bits

$$\mathcal{L}(q, \boldsymbol{\theta}, \lambda) = \mathbb{E}_{q(\mathbf{z}|\mathbf{x})} \log p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta}) - \max(\lambda, KL(q(\mathbf{z}|\mathbf{x})||p(\mathbf{z}))).$$

It ensures the use of less than λ bits of information and results in $KL(q(\mathbf{z}|\mathbf{x})||p(\mathbf{z})) \geq \lambda$.

Bowman S. R. et al. Generating Sentences from a Continuous Space, 2015
Kingma D. P. et al. Improving Variational Inference with Inverse Autoregressive Flow, 2016

VAE limitations

- Poor variational posterior distribution (encoder)

$$q(\mathbf{z}|\mathbf{x}, \phi) = \mathcal{N}(\mathbf{z}|\boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x}), \boldsymbol{\sigma}^2_{\boldsymbol{\phi}}(\mathbf{x})).$$

- Poor prior distribution

$$p(\mathbf{z}) = \mathcal{N}(0, \mathbf{I}).$$

- Poor probabilistic model (decoder)

$$p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{z}), \boldsymbol{\sigma}^2_{\boldsymbol{\theta}}(\mathbf{z})).$$

- Loose lower bound

$$\log p(\mathbf{x}|\boldsymbol{\theta}) - \mathcal{L}(q, \boldsymbol{\theta}) = (?).$$

Importance Sampling

Generative model

$$p(\mathbf{x}|\boldsymbol{\theta}) = \int p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta}) d\mathbf{z} = \int \left[\frac{p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta})}{q(\mathbf{z}|\mathbf{x})}\right] q(\mathbf{z}|\mathbf{x}) d\mathbf{z}$$
$$= \int f(\mathbf{x}, \mathbf{z}) q(\mathbf{z}|\mathbf{x}) d\mathbf{z} = \mathbb{E}_{\mathbf{z} \sim q(\mathbf{z}|\mathbf{x})} f(\mathbf{x}, \mathbf{z})$$

Here $f(\mathbf{x}, \mathbf{z}) = \frac{p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta})}{q(\mathbf{z}|\mathbf{x})}$.

ELBO

$$\log p(\mathbf{x}|\boldsymbol{\theta}) = \log \mathbb{E}_{\mathbf{z} \sim q(\mathbf{z}|\mathbf{x})} f(\mathbf{x}, \mathbf{z}) \geq \mathbb{E}_{\mathbf{z} \sim q(\mathbf{z}|\mathbf{x})} \log f(\mathbf{x}, \mathbf{z}) =$$
$$= \mathbb{E}_{\mathbf{z} \sim q(\mathbf{z}|\mathbf{x})} \log \frac{p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta})}{q(\mathbf{z}|\mathbf{x})} = \mathcal{L}(q, \boldsymbol{\theta}).$$

Could we choose better $f(\mathbf{x}, \mathbf{z})$?

IWAE

$$p(\mathbf{x}|\boldsymbol{\theta}) = \int p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta}) d\mathbf{z} = \int \left[\frac{p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta})}{q(\mathbf{z}|\mathbf{x})} \right] q(\mathbf{z}|\mathbf{x}) d\mathbf{z} = \mathbb{E}_{\mathbf{z} \sim q(\mathbf{z}|\mathbf{x})} f(\mathbf{x}, \mathbf{z})$$

Let define

$$f(\mathbf{x}, \mathbf{z}_1, \dots, \mathbf{z}_K) = \frac{1}{K} \sum_{k=1}^{K} \frac{p(\mathbf{x}, \mathbf{z}_k|\boldsymbol{\theta})}{q(\mathbf{z}_k|\mathbf{x})}$$

$$\mathbb{E}_{\mathbf{z}_1, \dots, \mathbf{z}_K \sim q(\mathbf{z}|\mathbf{x})} f(\mathbf{x}, \mathbf{z}_1, \dots, \mathbf{z}_K) = p(\mathbf{x}|\boldsymbol{\theta})$$

ELBO

$$\begin{aligned} \log p(\mathbf{x}|\boldsymbol{\theta}) = \log \mathbb{E}_{\mathbf{z}_1, \dots, \mathbf{z}_K \sim q(\mathbf{z}|\mathbf{x})} f(\mathbf{x}, \mathbf{z}, \dots, \mathbf{z}_K) \geq \\ \geq \mathbb{E}_{\mathbf{z}_1, \dots, \mathbf{z}_K \sim q(\mathbf{z}|\mathbf{x})} \log f(\mathbf{x}, \mathbf{z}, \dots, \mathbf{z}_K) = \\ = \mathbb{E}_{\mathbf{z}_1, \dots, \mathbf{z}_K \sim q(\mathbf{z}|\mathbf{x})} \log \left[\frac{1}{K} \sum_{k=1}^{K} \frac{p(\mathbf{x}, \mathbf{z}_k|\boldsymbol{\theta})}{q(\mathbf{z}_k|\mathbf{x})} \right] = \mathcal{L}_K(q, \boldsymbol{\theta}). \end{aligned}$$

Burda Y., Grosse R., Salakhutdinov R. Importance Weighted Autoencoders, 2015

IWAE

VAE objective

$$\log p(\mathbf{x}|\boldsymbol{\theta}) \geq \mathcal{L}(q, \boldsymbol{\theta}) = \mathbb{E}_{q(\mathbf{z}|\mathbf{x})} \log \frac{p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta})}{q(\mathbf{z}|\mathbf{x})} \rightarrow \max_{q, \boldsymbol{\theta}}$$

$$\mathcal{L}(q, \boldsymbol{\theta}) = \mathbb{E}_{\mathbf{z}_1, \dots, \mathbf{z}_K \sim q(\mathbf{z}|\mathbf{x})} \left(\frac{1}{K} \sum_{k=1}^{K} \log \frac{p(\mathbf{x}, \mathbf{z}_k|\boldsymbol{\theta})}{q(\mathbf{z}_k|\mathbf{x})} \right) \rightarrow \max_{q, \boldsymbol{\theta}}.$$

IWAE objective

$$\mathcal{L}_K(q, \boldsymbol{\theta}) = \mathbb{E}_{\mathbf{z}_1, \dots, \mathbf{z}_K \sim q(\mathbf{z}|\mathbf{x})} \log \left(\frac{1}{K} \sum_{k=1}^{K} \frac{p(\mathbf{x}, \mathbf{z}_k|\boldsymbol{\theta})}{q(\mathbf{z}_k|\mathbf{x})} \right) \rightarrow \max_{q, \boldsymbol{\theta}}.$$

If $K = 1$, these objectives coincide.

Burda Y., Grosse R., Salakhutdinov R. Importance Weighted Autoencoders, 2015

IWAE

Theorem

1. $\log p(\mathbf{x}|\boldsymbol{\theta}) \geq \mathcal{L}_K(q, \boldsymbol{\theta}) \geq \mathcal{L}_M(q, \boldsymbol{\theta}), \quad \text{for } K \geq M;$
2. $\log p(\mathbf{x}|\boldsymbol{\theta}) = \lim_{K\to\infty} \mathcal{L}_K(q, \boldsymbol{\theta})$ if $\frac{p(\mathbf{x},\mathbf{z}|\boldsymbol{\theta})}{q(\mathbf{z}|\mathbf{x})}$ is bounded.

Proof of 1.

$$
\begin{aligned}
\mathcal{L}_K(q, \boldsymbol{\theta}) &= \mathbb{E}_{\mathbf{z}_1, \dots, \mathbf{z}_K} \log \left(\frac{1}{K} \sum_{k=1}^{K} \frac{p(\mathbf{x}, \mathbf{z}_k|\boldsymbol{\theta})}{q(\mathbf{z}_k|\mathbf{x})} \right) = \\
&= \mathbb{E}_{\mathbf{z}_1, \dots, \mathbf{z}_K} \log \mathbb{E}_{k_1, \dots, k_M} \left(\frac{1}{M} \sum_{m=1}^{M} \frac{p(\mathbf{x}, \mathbf{z}_{k_m}|\boldsymbol{\theta})}{q(\mathbf{z}_{k_m}|\mathbf{x})} \right) \geq \\
&\geq \mathbb{E}_{\mathbf{z}_1, \dots, \mathbf{z}_K} \mathbb{E}_{k_1, \dots, k_m} \log \left(\frac{1}{M} \sum_{m=1}^{M} \frac{p(\mathbf{x}, \mathbf{z}_{k_m}|\boldsymbol{\theta})}{q(\mathbf{z}_{k_m}|\mathbf{x})} \right) = \\
&= \mathbb{E}_{\mathbf{z}_1, \dots, \mathbf{z}_M} \log \left(\frac{1}{M} \sum_{m=1}^{M} \frac{p(\mathbf{x}, \mathbf{z}_m|\boldsymbol{\theta})}{q(\mathbf{z}_m|\mathbf{x})} \right) = \mathcal{L}_M(q, \boldsymbol{\theta})
\end{aligned}
$$

$$
\frac{a_1 + \dots + a_K}{K} = \mathbb{E}_{k_1, \dots, k_M} \frac{a_{k_1} + \dots + a_{k_M}}{M}, \quad k_1, \dots, k_M \sim U[1, K]
$$

Burda Y., Grosse R., Salakhutdinov R. Importance Weighted Autoencoders, 2015

IWAE

Theorem

1. $\log p(\mathbf{x}|\boldsymbol{\theta}) \geq \mathcal{L}_K(q, \boldsymbol{\theta}) \geq \mathcal{L}_M(q, \boldsymbol{\theta}), \quad \text{for} K \geq M;$
2. $\log p(\mathbf{x}|\boldsymbol{\theta}) = \lim_{K\to\infty} \mathcal{L}_K(q, \boldsymbol{\theta})$ if $\frac{p(\mathbf{x},\mathbf{z}|\boldsymbol{\theta})}{q(\mathbf{z}|\mathbf{x})}$ is bounded.

Proof of 2.

Consider r.v. $\xi_K = \frac{1}{K}\sum_{k=1}^{K} \frac{p(\mathbf{x},\mathbf{z}_k|\boldsymbol{\theta})}{q(\mathbf{z}_k|\mathbf{x})}$.

If summands are bounded, then (from the strong law of large numbers)

$$\xi_K \xrightarrow[K\to\infty]{a.s.} \mathbb{E}_{q(\mathbf{z}|\mathbf{x})} \frac{p(\mathbf{x},\mathbf{z}|\boldsymbol{\theta})}{q(\mathbf{z}|\mathbf{x})} = p(\mathbf{x}|\boldsymbol{\theta}).$$

Hence $\mathcal{L}_K(q, \boldsymbol{\theta}) = \mathbb{E}\log\xi_K$ converges to $\log p(\mathbf{x}|\boldsymbol{\theta})$ as $K \to \infty$.

Burda Y., Grosse R., Salakhutdinov R. Importance Weighted Autoencoders, 2015

IWAE

$$\log p(\mathbf{x}|\boldsymbol{\theta}) \geq \mathcal{L}_K(q, \boldsymbol{\theta}) \geq \mathcal{L}(q, \boldsymbol{\theta})$$

If $K > 1$ the bound could be tighter.

$$\mathcal{L}(q, \boldsymbol{\theta}) = \mathbb{E}_{q(\mathbf{z}|\mathbf{x})} \log \frac{p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta})}{q(\mathbf{z}|\mathbf{x})};$$

$$\mathcal{L}_K(q, \boldsymbol{\theta}) = \mathbb{E}_{\mathbf{z}_1, \dots, \mathbf{z}_K \sim q(\mathbf{z}|\mathbf{x})} \log \left(\frac{1}{K} \sum_{k=1}^{K} \frac{p(\mathbf{x}, \mathbf{z}_k|\boldsymbol{\theta})}{q(\mathbf{z}_k|\mathbf{x})} \right).$$

- $\mathcal{L}_1(q, \boldsymbol{\theta}) = \mathcal{L}(q, \boldsymbol{\theta})$;
- $\mathcal{L}_\infty(q, \boldsymbol{\theta}) = \log p(\mathbf{x}|\boldsymbol{\theta})$.
- Which $q^*(\mathbf{z}|\mathbf{x})$ gives $\mathcal{L}(q^*, \boldsymbol{\theta}) = \log p(\mathbf{x}|\boldsymbol{\theta})$?
- Which $q^*(\mathbf{z}|\mathbf{x})$ gives $\mathcal{L}(q^*, \boldsymbol{\theta}) = \mathcal{L}_K(q, \boldsymbol{\theta})$?

Burda Y., Grosse R., Salakhutdinov R. Importance Weighted Autoencoders, 2015

IWAE

Theorem

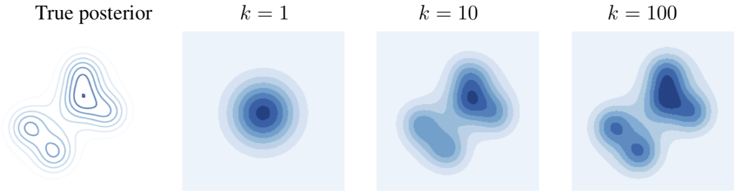
$\mathcal{L}(q_{EW}, \boldsymbol{\theta}) = \mathcal{L}_K(q, \boldsymbol{\theta})$ for the following variational distribution

$$q_{EW}(\mathbf{z}|\mathbf{x}) = \mathbb{E}_{\mathbf{z}_2,\dots,\mathbf{z}_K \sim q(\mathbf{z}|\mathbf{x})} q_{IW}(\mathbf{z}|\mathbf{x}, \mathbf{z}_{2:K}),$$

where

$$q_{IW}(\mathbf{z}|\mathbf{x}, \mathbf{z}_{2:K}) = \frac{\frac{p(\mathbf{x},\mathbf{z})}{q(\mathbf{z}|\mathbf{x})}}{\frac{1}{K}\sum_{k=1}^{K} \frac{p(\mathbf{x},\mathbf{z}_k)}{q(\mathbf{z}_k|\mathbf{x})}} q(\mathbf{z}|\mathbf{x}) = \frac{p(\mathbf{x}, \mathbf{z})}{\frac{1}{K}\left(\frac{p(\mathbf{x},\mathbf{z})}{q(\mathbf{z}|\mathbf{x})} + \sum_{k=2}^{K} \frac{p(\mathbf{x},\mathbf{z}_k)}{q(\mathbf{z}_k|\mathbf{x})}\right)}.$$

IWAE posterior



Cremer C., Morris Q., Duvenaud D. Reinterpreting Importance-Weighted Autoencoders, 2017

IWAE

Objective

$$\mathcal{L}_K(q, \boldsymbol{\theta}) = \mathbb{E}_{\mathbf{z}_1, \dots, \mathbf{z}_K \sim q(\mathbf{z}|\mathbf{x}, \phi)} \log \left(\frac{1}{K} \sum_{k=1}^{K} \frac{p(\mathbf{x}, \mathbf{z}_k | \boldsymbol{\theta})}{q(\mathbf{z}_k | \mathbf{x}, \phi)} \right) \to \max_{\phi, \boldsymbol{\theta}} .$$

Gradient

$$\Delta_K = \nabla_{\boldsymbol{\theta}, \phi} \log \left(\frac{1}{K} \sum_{k=1}^{K} \frac{p(\mathbf{x}, \mathbf{z}_k | \boldsymbol{\theta})}{q(\mathbf{z}_k | \mathbf{x}, \phi)} \right), \quad \mathbf{z}_k \sim q(\mathbf{z}|\mathbf{x}, \phi).$$

Theorem

$$\text{SNR}_K = \frac{\mathbb{E}[\Delta_K]}{\sigma(\Delta_K)}; \quad \text{SNR}_K(\boldsymbol{\theta}) = O(\sqrt{K}); \quad \text{SNR}_K(\phi) = O\left(\sqrt{\frac{1}{K}}\right).$$

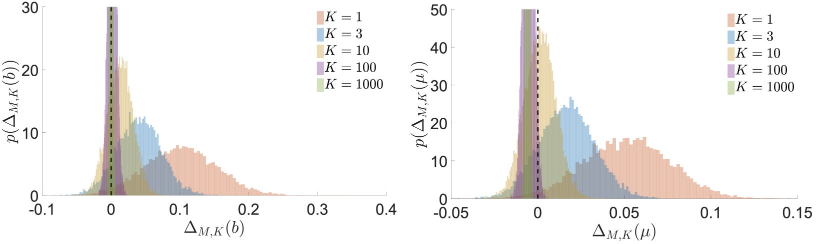
Hence, increasing K vanishes gradient signal of inference network $q(\mathbf{z}|\mathbf{x}, \phi)$.

Rainforth T. et al. Tighter variational bounds are not necessarily better, 2018

IWAE

Theorem

$$\mathrm{SNR}_K = \frac{\mathbb{E}[\Delta_K]}{\sigma(\Delta_K)}; \quad \mathrm{SNR}_K(\boldsymbol{\theta}) = O(\sqrt{K}); \quad \mathrm{SNR}_K(\phi) = O\left(\sqrt{\frac{1}{K}}\right).$$



- IWAE makes the variational bound tighter and extends the class of variational distributions.
- Gradient signal becomes really small, training is complicated.
- IWAE is very popular technique as a quality measure for VAE models.

Rainforth T. et al. Tighter variational bounds are not necessarily better, 2018

Summary

- Linear flows try to parametrize set of invertible matrices via matrix decompositions.
- Gaussian autoregressive model is a special type of flow.
- MAF is an example of such a model which is suitable for density estimation tasks. IAF uses an inverse autoregressive transformation for variational inference task.
- RealNVP is a special case of IAF and MAF.
- There is a duality between forward and reverse KL for flow models.
- To apply a continuous model to a discrete distribution it is standard practice to dequantize data at first.