

Deep Generative Models

Lecture 6

Roman Isachenko



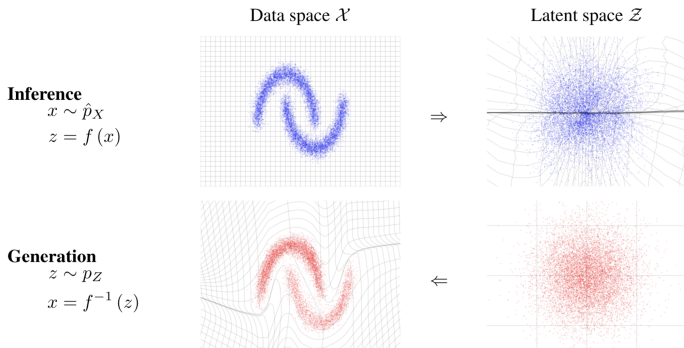
AI Masters

Autumn, 2022

Recap of previous lecture

Definition

Normalizing flow is a *differentiable, invertible* mapping from data \mathbf{x} to the noise \mathbf{z} .



Log likelihood

$$\log p(\mathbf{x}|\boldsymbol{\theta}) = \log p(f_K \circ \dots \circ f_1(\mathbf{x})) + \sum_{k=1}^K \log |\det(\mathbf{J}_{f_k})|$$

Recap of previous lecture

Forward KL for flow model

$$\log p(\mathbf{x}|\boldsymbol{\theta}) = \log p(f(\mathbf{x}, \boldsymbol{\theta})) + \log |\det(\mathbf{J}_f)|$$

Reverse KL for flow model

$$KL(p||\pi) = \mathbb{E}_{p(\mathbf{z})} [\log p(\mathbf{z}) - \log |\det(\mathbf{J}_g)| - \log \pi(g(\mathbf{z}, \boldsymbol{\theta}))]$$

Flow KL duality

$$\arg \min_{\boldsymbol{\theta}} KL(\pi(\mathbf{x})||p(\mathbf{x}|\boldsymbol{\theta})) = \arg \min_{\boldsymbol{\theta}} KL(p(\mathbf{z}|\boldsymbol{\theta})||p(\mathbf{z}))$$

- ▶ $p(\mathbf{z})$ is a base distribution; $\pi(\mathbf{x})$ is a data distribution;
- ▶ $\mathbf{z} \sim p(\mathbf{z})$, $\mathbf{x} = g(\mathbf{z}, \boldsymbol{\theta})$, $\mathbf{x} \sim p(\mathbf{x}|\boldsymbol{\theta})$;
- ▶ $\mathbf{x} \sim \pi(\mathbf{x})$, $\mathbf{z} = f(\mathbf{x}, \boldsymbol{\theta})$, $\mathbf{z} \sim p(\mathbf{z}|\boldsymbol{\theta})$.

Recap of previous lecture

Flow log-likelihood

$$\log p(\mathbf{x}|\boldsymbol{\theta}) = \log p(f(\mathbf{x}, \boldsymbol{\theta})) + \log |\det(\mathbf{J}_f)|$$

The main challenge is a determinant of the Jacobian.

Linear flows

$$\mathbf{z} = f(\mathbf{x}, \boldsymbol{\theta}) = \mathbf{W}\mathbf{x}, \quad \mathbf{W} \in \mathbb{R}^{m \times m}, \quad \boldsymbol{\theta} = \mathbf{W}, \quad \mathbf{J}_f = \mathbf{W}^T$$

- ▶ LU-decomposition

$$\mathbf{W} = \mathbf{P}\mathbf{L}\mathbf{U}.$$

- ▶ QR-decomposition

$$\mathbf{W} = \mathbf{Q}\mathbf{R}.$$

Decomposition should be done only once in the beginning. Next, we fit decomposed matrices (**P/L/U** or **Q/R**).

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

Hoogeboom E., et al. Emerging convolutions for generative normalizing flows, 2019

Outline

1. Autoregressive flows

- Gaussian autoregressive flows

- Inverse gaussian autoregressive flows

- RealNVP: coupling layer

2. Normalizing flows as VAE model

Outline

1. Autoregressive flows

- Gaussian autoregressive flows

- Inverse gaussian autoregressive flows

- RealNVP: coupling layer

2. Normalizing flows as VAE model

Outline

1. Autoregressive flows

Gaussian autoregressive flows

Inverse gaussian autoregressive flows

RealNVP: coupling layer

2. Normalizing flows as VAE model

Gaussian autoregressive model

Consider an autoregressive model

$$p(\mathbf{x}|\boldsymbol{\theta}) = \prod_{j=1}^m p(x_j|\mathbf{x}_{1:j-1}, \boldsymbol{\theta}), \quad p(x_j|\mathbf{x}_{1:j-1}, \boldsymbol{\theta}) = \mathcal{N}(\mu_j(\mathbf{x}_{1:j-1}), \sigma_j^2(\mathbf{x}_{1:j-1})).$$

Sampling: reparametrization trick

$$x_j = \sigma_j(\mathbf{x}_{1:j-1}) \cdot z_j + \mu_j(\mathbf{x}_{1:j-1}), \quad z_j \sim \mathcal{N}(0, 1).$$

Inverse transform

$$z_j = (x_j - \mu_j(\mathbf{x}_{1:j-1})) \cdot \frac{1}{\sigma_j(\mathbf{x}_{1:j-1})}.$$

- ▶ We have an **invertible** and **differentiable** transformation from $p(\mathbf{z})$ to $p(\mathbf{x}|\boldsymbol{\theta})$.
- ▶ It is an autoregressive (AR) flow with the base distribution $p(\mathbf{z}) = \mathcal{N}(0, 1)$!
- ▶ Jacobian of such transformation is triangular!

Gaussian autoregressive flow

$$\mathbf{x} = g(\mathbf{z}, \boldsymbol{\theta}) \quad \Rightarrow \quad \mathbf{x}_j = \sigma_j(\mathbf{x}_{1:j-1}) \cdot \mathbf{z}_j + \mu_j(\mathbf{x}_{1:j-1}).$$

$$\mathbf{z} = f(\mathbf{x}, \boldsymbol{\theta}) \quad \Rightarrow \quad \mathbf{z}_j = (\mathbf{x}_j - \mu_j(\mathbf{x}_{1:j-1})) \cdot \frac{1}{\sigma_j(\mathbf{x}_{1:j-1})}.$$

Generation function $g(\mathbf{z}, \boldsymbol{\theta})$ is **sequential**.

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

Forward KL for NF

$$KL(\pi||p) = -\mathbb{E}_{\pi(\mathbf{x})} [\log p(f(\mathbf{x}, \boldsymbol{\theta})) + \log |\det(\mathbf{J}_f)|] + \text{const}$$

- ▶ We need to be able to compute $f(\mathbf{x}, \boldsymbol{\theta})$ and its Jacobian.
- ▶ We need to be able to compute the density $p(\mathbf{z})$.
- ▶ We don't need to think about computing the function $g(\mathbf{z}, \boldsymbol{\theta}) = f^{-1}(\mathbf{z}, \boldsymbol{\theta})$ until we want to sample from the model.

Outline

1. Autoregressive flows

Gaussian autoregressive flows

Inverse gaussian autoregressive flows

RealNVP: coupling layer

2. Normalizing flows as VAE model

Inverse gaussian autoregressive flow (IAF)

Let use the following reparametrization: $\tilde{\sigma} = \frac{1}{\sigma}$; $\tilde{\mu} = -\frac{\mu}{\sigma}$.

Gaussian autoregressive flow

$$x_j = \sigma_j(\mathbf{x}_{1:j-1}) \cdot z_j + \mu_j(\mathbf{x}_{1:j-1}) = (z_j - \tilde{\mu}_j(\mathbf{x}_{1:j-1})) \cdot \frac{1}{\tilde{\sigma}_j(\mathbf{x}_{1:j-1})}$$
$$z_j = (x_j - \mu_j(\mathbf{x}_{1:j-1})) \cdot \frac{1}{\sigma_j(\mathbf{x}_{1:j-1})} = \tilde{\sigma}_j(\mathbf{x}_{1:j-1}) \cdot x_j + \tilde{\mu}_j(\mathbf{x}_{1:j-1}).$$

Let just swap \mathbf{z} and \mathbf{x} .

Inverse gaussian autoregressive flow

$$\mathbf{x} = g(\mathbf{z}, \boldsymbol{\theta}) \quad \Rightarrow \quad \mathbf{x}_j = \tilde{\sigma}_j(\mathbf{z}_{1:j-1}) \cdot z_j + \tilde{\mu}_j(\mathbf{z}_{1:j-1})$$
$$\mathbf{z} = f(\mathbf{x}, \boldsymbol{\theta}) \quad \Rightarrow \quad z_j = (\mathbf{x}_j - \tilde{\mu}_j(\mathbf{z}_{1:j-1})) \cdot \frac{1}{\tilde{\sigma}_j(\mathbf{z}_{1:j-1})}.$$

Inverse gaussian autoregressive flow (IAF)

Gaussian autoregressive flow: $g(\mathbf{z}, \theta)$

$$x_j = \sigma_j(\mathbf{x}_{1:j-1}) \cdot z_j + \mu_j(\mathbf{x}_{1:j-1}).$$

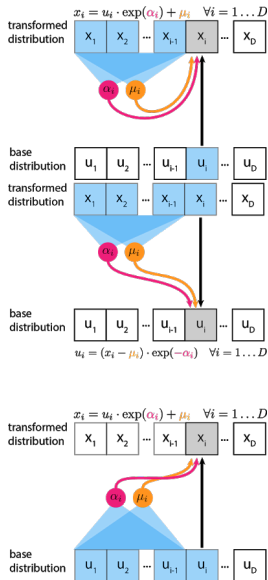
Inverse transform: $f(\mathbf{x}, \theta)$

$$z_j = (x_j - \mu_j(\mathbf{x}_{1:j-1})) \cdot \frac{1}{\sigma_j(\mathbf{x}_{1:j-1})};$$

$$z_j = \tilde{\sigma}_j(\mathbf{x}_{1:j-1}) \cdot x_j + \tilde{\mu}_j(\mathbf{x}_{1:j-1}).$$

Inverse gaussian autoregressive flow:
 $g(\mathbf{z}, \theta)$

$$x_j = \tilde{\sigma}_j(\mathbf{z}_{1:j-1}) \cdot z_j + \tilde{\mu}_j(\mathbf{z}_{1:j-1}).$$



Inverse gaussian autoregressive flow (IAF)

Inverse gaussian autoregressive flow

$$\begin{aligned}\mathbf{x} = g(\mathbf{z}, \boldsymbol{\theta}) &\Rightarrow \mathbf{x}_j = \tilde{\sigma}_j(\mathbf{z}_{1:j-1}) \cdot z_j + \tilde{\mu}_j(\mathbf{z}_{1:j-1}) \\ \mathbf{z} = f(\mathbf{x}, \boldsymbol{\theta}) &\Rightarrow z_j = (\mathbf{x}_j - \tilde{\mu}_j(\mathbf{z}_{1:j-1})) \cdot \frac{1}{\tilde{\sigma}_j(\mathbf{z}_{1:j-1})}.\end{aligned}$$

Reverse KL for NF

$$KL(p||\pi) = \mathbb{E}_{p(\mathbf{z})} [\log p(\mathbf{z}) - \log |\det(\mathbf{J}_g)| - \log \pi(g(\mathbf{z}, \boldsymbol{\theta}))]$$

- ▶ We need to be able to compute $g(\mathbf{z}, \boldsymbol{\theta})$ and its Jacobian.
- ▶ We need to be able to sample from the density $p(\mathbf{z})$ (do not need to evaluate it) and to evaluate(!) $\pi(\mathbf{x})$.
- ▶ We don't need to think about computing the function $f(\mathbf{x}, \boldsymbol{\theta})$.

Gaussian autoregressive NF

Gaussian AR NF

$$\mathbf{x} = g(\mathbf{z}, \boldsymbol{\theta}) \Rightarrow \mathbf{x}_j = \sigma_j(\mathbf{x}_{1:j-1}) \cdot \mathbf{z}_j + \mu_j(\mathbf{x}_{1:j-1}).$$

$$\mathbf{z} = f(\mathbf{x}, \boldsymbol{\theta}) \Rightarrow \mathbf{z}_j = (\mathbf{x}_j - \mu_j(\mathbf{x}_{1:j-1})) \cdot \frac{1}{\sigma_j(\mathbf{x}_{1:j-1})}.$$

- ▶ Sampling is sequential, density estimation is parallel.
- ▶ Forward KL is a natural loss.

Inverse gaussian AR NF

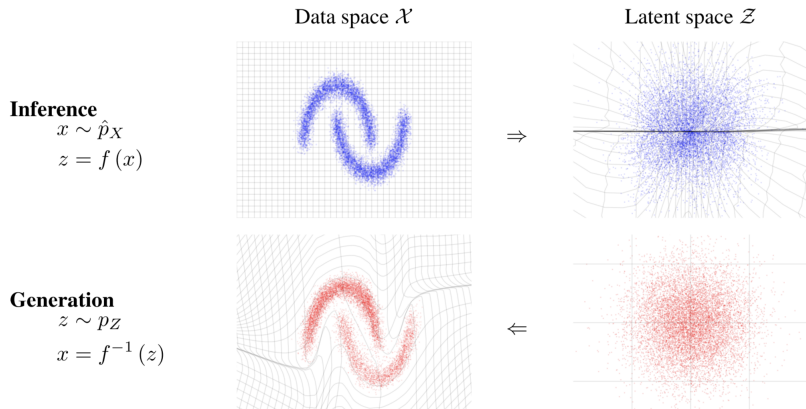
$$\mathbf{x} = g(\mathbf{z}, \boldsymbol{\theta}) \Rightarrow \mathbf{x}_j = \tilde{\sigma}_j(\mathbf{z}_{1:j-1}) \cdot \mathbf{z}_j + \tilde{\mu}_j(\mathbf{z}_{1:j-1})$$

$$\mathbf{z} = f(\mathbf{x}, \boldsymbol{\theta}) \Rightarrow \mathbf{z}_j = (\mathbf{x}_j - \tilde{\mu}_j(\mathbf{z}_{1:j-1})) \cdot \frac{1}{\tilde{\sigma}_j(\mathbf{z}_{1:j-1})}.$$

- ▶ Sampling is parallel, density estimation is sequential.
- ▶ Reverse KL is a natural loss.

Autoregressive flows

Gaussian AR NF and inverse gaussian AR NF are mutually interchangeable.



Outline

1. Autoregressive flows

Gaussian autoregressive flows

Inverse gaussian autoregressive flows

RealNVP: coupling layer

2. Normalizing flows as VAE model

RealNVP

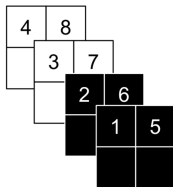
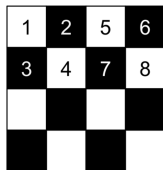
Let split \mathbf{x} and \mathbf{z} in two parts:

$$\mathbf{x} = [\mathbf{x}_1, \mathbf{x}_2] = [\mathbf{x}_{1:d}, \mathbf{x}_{d+1:m}]; \quad \mathbf{z} = [\mathbf{z}_1, \mathbf{z}_2] = [\mathbf{z}_{1:d}, \mathbf{z}_{d+1:m}].$$

Coupling layer

$$\begin{cases} \mathbf{x}_1 = \mathbf{z}_1; \\ \mathbf{x}_2 = \mathbf{z}_2 \odot \sigma(\mathbf{z}_1, \theta) + \mu(\mathbf{z}_1, \theta). \end{cases} \quad \begin{cases} \mathbf{z}_1 = \mathbf{x}_1; \\ \mathbf{z}_2 = (\mathbf{x}_2 - \mu(\mathbf{x}_1, \theta)) \odot \frac{1}{\sigma(\mathbf{x}_1, \theta)}. \end{cases}$$

Image partitioning



- ▶ Checkerboard ordering uses masking.
- ▶ Channelwise ordering uses splitting.

RealNVP

Coupling layer

$$\begin{cases} \mathbf{x}_1 = \mathbf{z}_1; \\ \mathbf{x}_2 = \mathbf{z}_2 \odot \sigma(\mathbf{z}_1, \theta) + \mu(\mathbf{z}_1, \theta). \end{cases} \quad \begin{cases} \mathbf{z}_1 = \mathbf{x}_1; \\ \mathbf{z}_2 = (\mathbf{x}_2 - \mu(\mathbf{x}_1, \theta)) \odot \frac{1}{\sigma(\mathbf{x}_1, \theta)}. \end{cases}$$

Estimating the density takes 1 pass, sampling takes 1 pass!

Jacobian

$$\det \left(\frac{\partial \mathbf{z}}{\partial \mathbf{x}} \right) = \det \begin{pmatrix} \mathbf{I}_d & 0_{d \times m-d} \\ \frac{\partial \mathbf{z}_2}{\partial \mathbf{x}_1} & \frac{\partial \mathbf{z}_2}{\partial \mathbf{x}_2} \end{pmatrix} = \prod_{j=1}^{m-d} \frac{1}{\sigma_j(\mathbf{x}_1, \theta)}.$$

Gaussian AR NF

$$\mathbf{x} = g(\mathbf{z}, \theta) \quad \Rightarrow \quad \mathbf{x}_j = \sigma_j(\mathbf{x}_{1:j-1}) \cdot \mathbf{z}_j + \mu_j(\mathbf{x}_{1:j-1}).$$

$$\mathbf{z} = f(\mathbf{x}, \theta) \quad \Rightarrow \quad \mathbf{z}_j = (\mathbf{x}_j - \mu_j(\mathbf{x}_{1:j-1})) \cdot \frac{1}{\sigma_j(\mathbf{x}_{1:j-1})}.$$

How to get RealNVP coupling layer from gaussian AR NF?

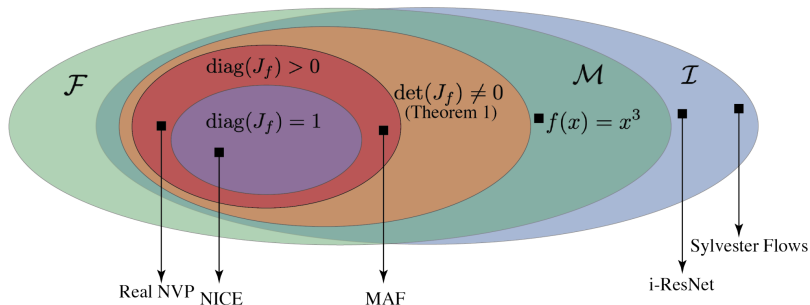
Glow samples

Glow model: coupling layer + linear flows (1x1 convs)



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

Venn diagram for Normalizing flows



- ▶ \mathcal{I} – invertible functions.
- ▶ \mathcal{F} – continuously differentiable functions whose Jacobian is lower triangular.
- ▶ \mathcal{M} – invertible functions from \mathcal{F} .

Outline

1. Autoregressive flows

Gaussian autoregressive flows

Inverse gaussian autoregressive flows

RealNVP: coupling layer

2. Normalizing flows as VAE model

VAE vs Normalizing flows

	VAE	NF
Objective	ELBO \mathcal{L}	Forward KL/MLE
Encoder	stochastic $\mathbf{z} \sim q(\mathbf{z} \mathbf{x}, \phi)$	deterministic $\mathbf{z} = f(\mathbf{x}, \theta)$ $q(\mathbf{z} \mathbf{x}, \theta) = \delta(\mathbf{z} - f(\mathbf{x}, \theta))$
Decoder	stochastic $\mathbf{x} \sim p(\mathbf{x} \mathbf{z}, \theta)$	deterministic $\mathbf{x} = g(\mathbf{z}, \theta)$ $p(\mathbf{x} \mathbf{z}, \theta) = \delta(\mathbf{x} - g(\mathbf{z}, \theta))$
Parameters	ϕ, θ	$\theta \equiv \phi$

Theorem

MLE for normalizing flow is equivalent to maximization of ELBO for VAE model with deterministic encoder and decoder:

$$p(\mathbf{x}|\mathbf{z}, \theta) = \delta(\mathbf{x} - f^{-1}(\mathbf{z}, \theta)) = \delta(\mathbf{x} - g(\mathbf{z}, \theta));$$

$$q(\mathbf{z}|\mathbf{x}, \theta) = p(\mathbf{z}|\mathbf{x}, \theta) = \delta(\mathbf{z} - f(\mathbf{x}, \theta)).$$

Normalizing flow as VAE

Proof

1. Dirac delta function property

$$\mathbb{E}_{\delta(\mathbf{x}-\mathbf{y})} f(\mathbf{x}) d\mathbf{x} = \int \delta(\mathbf{x}-\mathbf{y}) f(\mathbf{x}) d\mathbf{x} = f(\mathbf{y}).$$

2. CoV theorem and Bayes theorem:

$$p(\mathbf{x}|\boldsymbol{\theta}) = p(\mathbf{z}) |\det(\mathbf{J}_f)|;$$

$$p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta}) = \frac{p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta}) p(\mathbf{z})}{p(\mathbf{x}|\boldsymbol{\theta})}; \quad \Rightarrow \quad p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta}) = p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta}) |\det(\mathbf{J}_f)|.$$

3. Log-likelihood decomposition

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

Normalizing flow as VAE

Proof

ELBO objective:

$$\begin{aligned}\mathcal{L} &= \mathbb{E}_{q(\mathbf{z}|\mathbf{x},\theta)} \left[\log p(\mathbf{x}|\mathbf{z},\theta) - \log \frac{q(\mathbf{z}|\mathbf{x},\theta)}{p(\mathbf{z})} \right] \\ &= \mathbb{E}_{q(\mathbf{z}|\mathbf{x},\theta)} \left[\log \frac{p(\mathbf{x}|\mathbf{z},\theta)}{q(\mathbf{z}|\mathbf{x},\theta)} + \log p(\mathbf{z}) \right].\end{aligned}$$

1. Dirac delta function property:

$$\mathbb{E}_{q(\mathbf{z}|\mathbf{x},\theta)} \log p(\mathbf{z}) = \int \delta(\mathbf{z} - f(\mathbf{x},\theta)) \log p(\mathbf{z}) d\mathbf{z} = \log p(f(\mathbf{x},\theta)).$$

2. CoV theorem and Bayes theorem:

$$\mathbb{E}_{q(\mathbf{z}|\mathbf{x},\theta)} \log \frac{p(\mathbf{x}|\mathbf{z},\theta)}{q(\mathbf{z}|\mathbf{x},\theta)} = \mathbb{E}_{q(\mathbf{z}|\mathbf{x},\theta)} \log \frac{p(\mathbf{z}|\mathbf{x},\theta) |\det(\mathbf{J}_f)|}{q(\mathbf{z}|\mathbf{x},\theta)} = \log |\det \mathbf{J}_f|.$$

3. Log-likelihood decomposition

$$\log p(\mathbf{x}|\theta) = \mathcal{L}(\theta) = \log p(f(\mathbf{x},\theta)) + \log |\det \mathbf{J}_f|.$$

Summary

- ▶ Gaussian autoregressive flow is an autoregressive model with triangular Jacobian. It has fast inference function and slow generation function. Forward KL is a natural loss function.
- ▶ Inverse gaussian autoregressive flow generate new samples fast, but the inference is slow. Reverse KL is a natural loss function.
- ▶ The RealNVP coupling layer is an effective type of flow (special case of AR flows) that has fast inference and generation modes.
- ▶ NF models could be treated as VAE model with deterministic encoder and decoder.