Deep Generative Models

Lecture 6

Roman Isachenko

Ozon Masters

Spring, 2022

Recap of previous lecture

Likelihood-based models so far...

Autoregressive models

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

- tractable likelihood,
- no inferred latent factors.

Latent variable models

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

- latent feature representation,
- intractable likelihood.

How to build a model with latent variables and tractable likelihood?

Recap of previous lecture

Change of variable theorem

Let $\mathbf{x}$ be a random variable with density function $p(\mathbf{x})$ and $f : \mathbb{R}^m \to \mathbb{R}^m$ is a differentiable, invertible function (diffeomorphism). If $\mathbf{z} = f(\mathbf{x})$, $\mathbf{x} = f^{-1}(\mathbf{z}) = g(\mathbf{z})$, then

$$p(\mathbf{x}) = p(\mathbf{z}) \left| \det \left(\frac{\partial \mathbf{z}}{\partial \mathbf{x}} \right) \right| = p(f(\mathbf{x})) \left| \det \left(\frac{\partial f(\mathbf{x})}{\partial \mathbf{x}} \right) \right|$$

$$p(\mathbf{z}) = p(\mathbf{x}) \left| \det \left(\frac{\partial \mathbf{x}}{\partial \mathbf{z}} \right) \right| = p(g(\mathbf{z})) \left| \det \left(\frac{\partial g(\mathbf{z})}{\partial \mathbf{z}} \right) \right|.$$

Inverse function theorem

If function f is invertible and Jacobian is continuous and non-singular, then

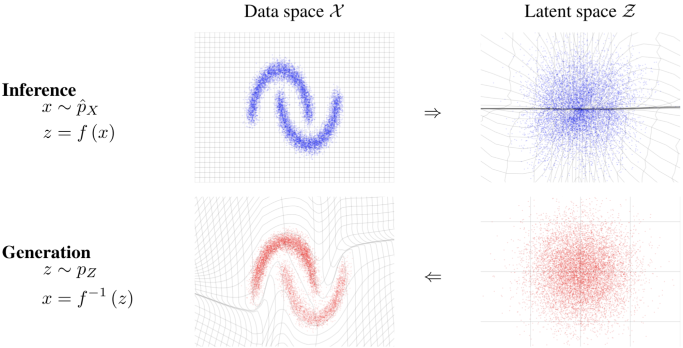
$$\left| \det \left(\frac{\partial f(\mathbf{x})}{\partial \mathbf{x}} \right) \right| = \left| \det \left(\frac{\partial g^{-1}(\mathbf{x})}{\partial \mathbf{x}} \right) \right| = \left| \det \left(\frac{\partial g(\mathbf{z})}{\partial \mathbf{z}} \right) \right|^{-1}$$

Recap of previous lecture

MLE problem for fitting flows

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

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



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

Recap of previous lecture

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

Definition

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

- **Normalizing** means that the inverse flow takes samples from $p(\mathbf{x})$ and normalizes them into samples from density $p(\mathbf{z})$.
- **Flow** refers to the trajectory followed by samples from $p(\mathbf{z})$ as they are transformed by the sequence of transformations

$$\mathbf{z} = f_K \circ \cdots \circ f_1(\mathbf{x}); \quad \mathbf{x} = f_1^{-1} \circ \cdots \circ f_K^{-1}(\mathbf{z}) = g_1 \circ \cdots \circ g_K(\mathbf{z})$$

$$p(\mathbf{x}) = p(f_K \circ \cdots \circ f_1(\mathbf{x})) \left| \det \left(\frac{\partial f_K \circ \cdots \circ f_1(\mathbf{x})}{\partial \mathbf{x}} \right) \right| =$$

$$= p(f_K \circ \cdots \circ f_1(\mathbf{x})) \prod_{k=1}^{K} \left| \det \left(\frac{\partial \mathbf{f}_k}{\partial \mathbf{f}_{k-1}} \right) \right|.$$

Recap of previous lecture

Forward KL for flow model

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

Reverse KL for flow model

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

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;
- $\mathbf{z} \sim p(\mathbf{z})$, $\mathbf{x} = g(\mathbf{z}, \boldsymbol{\theta})$, $\mathbf{x} \sim p(\mathbf{x}|\boldsymbol{\theta})$;
- $\mathbf{x} \sim \pi(\mathbf{x})$, $\mathbf{z} = f(\mathbf{x}, \boldsymbol{\theta})$, $\mathbf{z} \sim p(\mathbf{z}|\boldsymbol{\theta})$;

Papamakarios G. et al. Normalizing flows for probabilistic modeling and inference, 2019

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

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

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

Outline

Autoregressive flows

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

- $\tau(\cdot, \cdot)$ – coupling law (invertible by first argument).
- $c(\cdot)$ – coupling function (do not need to be invertible, could be neural network).

Coupling law $\tau(\cdot, \cdot)$

- $\tau(x, c) = x + c$ – additive;
- $\tau(x, c) = x \odot \exp c_1 + c_2$ – affine.

What is the Jacobian for the additive/affine coupling law?

Jacobian

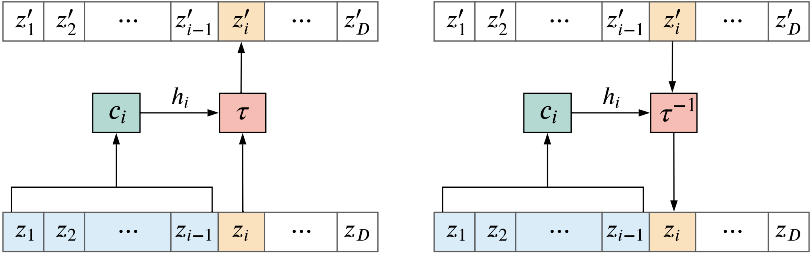
$$\det\left(\frac{\partial \mathbf{x}}{\partial \mathbf{z}}\right) = \prod_{i=1}^{m} \frac{\partial x_i}{\partial z_i} = \prod_{i=1}^{m} \frac{\partial \tau(z_i, c(\mathbf{z}_{1:i-1}))}{\partial z_i}$$

Papamakarios G. et al. Normalizing flows for probabilistic modeling and inference, 2019

Autoregressive flows

Forward and inverse transforms

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



- Forward transform is **not sequential**.
- Inverse transform is **sequential**.

Papamakarios G. et al. Normalizing flows for probabilistic modeling and inference, 2019

Gaussian autoregressive model

Consider an autoregressive model

$$p(\mathbf{x}|\boldsymbol{\theta}) = \prod_{i=1}^{m} p(x_i|\mathbf{x}_{1:i-1}, \boldsymbol{\theta}), \quad p(x_i|\mathbf{x}_{1:i-1}, \boldsymbol{\theta}) = \mathcal{N}\left(\mu_i(\mathbf{x}_{1:i-1}), \sigma_i^2(\mathbf{x}_{1:i-1})\right).$$

Sampling: reparametrization trick

$$x_i = \sigma_i(\mathbf{x}_{1:i-1}) \cdot z_i + \mu_i(\mathbf{x}_{1:i-1}), \quad z_i \sim \mathcal{N}(0, 1).$$

Inverse transform

$$z_i = (x_i - \mu_i(\mathbf{x}_{1:i-1})) \cdot \frac{1}{\sigma_i(\mathbf{x}_{1:i-1})}.$$

We have got an autoregressive flow with base distribution $\mathbf{z} = \mathcal{N}(0, 1)$ and affine coupling law $\tau(x, \mu, \sigma) = x \odot \sigma + \mu$.

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

Gaussian autoregressive flow

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

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

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

Forward KL for flow model

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

- 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 flow.

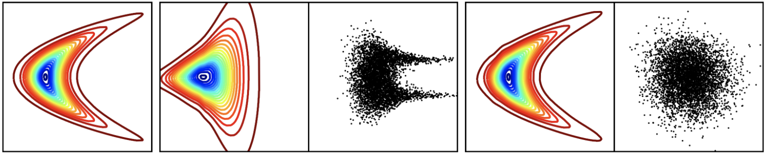
Papamakarios G., Pavlakou T., Murray I. Masked Autoregressive Flow for Density Estimation, 2017

Masked autoregressive flow (MAF)

Gaussian autoregressive model

$$p(\mathbf{x}|\boldsymbol{\theta}) = \prod_{i=1}^{m} p(x_i|\mathbf{x}_{1:i-1}, \boldsymbol{\theta}) = \prod_{i=1}^{m} \mathcal{N}\left(x_i|\mu_i(\mathbf{x}_{1:i-1}), \sigma_i^2(\mathbf{x}_{1:i-1})\right).$$

We could use MADE for the conditionals. Samples from the base distribution could be an indicator of how good the flow was fitted.



(a) Target density (b) MADE with Gaussian conditionals (c) MAF with 5 layers

MAF is just a stacked MADE model with different ordering.

- Parallel density estimation.
- Sequential sampling.

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

Inverse autoregressive flow (IAF)

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

Gaussian autoregressive flow

$$x_i = \sigma_i(\mathbf{x}_{1:i-1}) \cdot z_i + \mu_i(\mathbf{x}_{1:i-1}) = (z_i - \tilde{\mu}_i(\mathbf{x}_{1:i-1})) \cdot \frac{1}{\tilde{\sigma}_i(\mathbf{x}_{1:i-1})}$$

$$z_i = (x_i - \mu_i(\mathbf{x}_{1:i-1})) \cdot \frac{1}{\sigma_i(\mathbf{x}_{1:i-1})} = \tilde{\sigma}_i(\mathbf{x}_{1:i-1}) \cdot x_i + \tilde{\mu}_i(\mathbf{x}_{1:i-1}).$$

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

Inverse autoregressive flow

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

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

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

Inverse autoregressive flow (IAF)

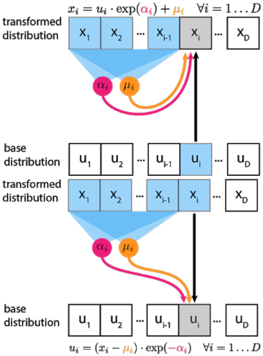
Gaussian autoregressive flow: $f(\mathbf{x}, \boldsymbol{\theta})$

$$x_i = \sigma_i(\mathbf{x}_{1:i-1}) \cdot z_i + \mu_i(\mathbf{x}_{1:i-1}).$$

Inverse transform: $g(\mathbf{z}, \boldsymbol{\theta})$

$$z_i = (x_i - \mu_i(\mathbf{x}_{1:i-1})) \cdot \frac{1}{\sigma_i(\mathbf{x}_{1:i-1})};$$

$$z_i = \tilde{\sigma}_i(\mathbf{x}_{1:i-1}) \cdot x_i + \tilde{\mu}_i(\mathbf{x}_{1:i-1}).$$



Inverse autoregressive flow: $f(\mathbf{x}, \boldsymbol{\theta})$

$$x_i = \tilde{\sigma}_i(\mathbf{z}_{1:i-1}) \cdot z_i + \tilde{\mu}_i(\mathbf{z}_{1:i-1}).$$

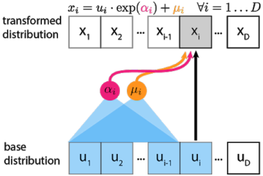


image credit: https://blog.evjang.com/2018/01/nf2.html

Autoregressive flows

Forward and inverse transforms in MAF

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

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

- Sampling is sequential.
- Density estimation is parallel.

Forward and inverse transforms in IAF

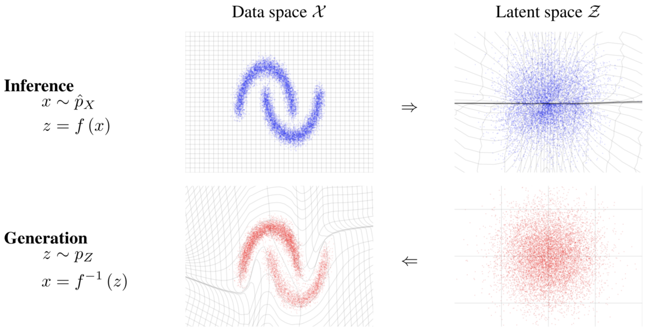
$$\mathbf{x} = g(\mathbf{z}, \boldsymbol{\theta}) \quad \Rightarrow \quad x_i = \tilde{\sigma}_i(\mathbf{z}_{1:i-1}) \cdot z_i + \tilde{\mu}_i(\mathbf{z}_{1:i-1}).$$

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

- Sampling is parallel.
- Density estimation is sequential.

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

Autoregressive flows



- MAF performs parallel inference that is useful for density estimation tasks (forward KL or MLE).
- IAF performs parallel generation that is useful for optimization of reverse KL.

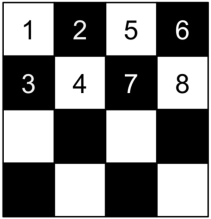
Dinh L., Sohl-Dickstein J., Bengio S. Density estimation using Real NVP, 2016

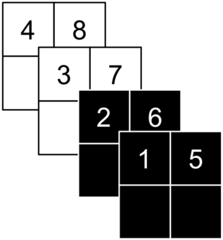
RealNVP

Coupling layer

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

Image partitioning





Masked convolutions are used to define ordering.

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

RealNVP

Affine coupling law

$$\begin{cases} \mathbf{z}_{1:d} = \mathbf{x}_{1:d}; \\ \mathbf{z}_{d:m} = \mathbf{x}_{d:m} \odot \exp\left(c_1(\mathbf{x}_{1:d}, \boldsymbol{\theta})\right) + c_2(\mathbf{x}_{1:d}, \boldsymbol{\theta}). \end{cases}$$

$$\begin{cases} \mathbf{x}_{1:d} = \mathbf{z}_{1:d}; \\ \mathbf{x}_{d:m} = \left(\mathbf{z}_{d:m} - c_2(\mathbf{z}_{1:d}, \boldsymbol{\theta})\right) \odot \exp(-c_1(\mathbf{z}_{1:d}, \boldsymbol{\theta})). \end{cases}$$

Jacobian

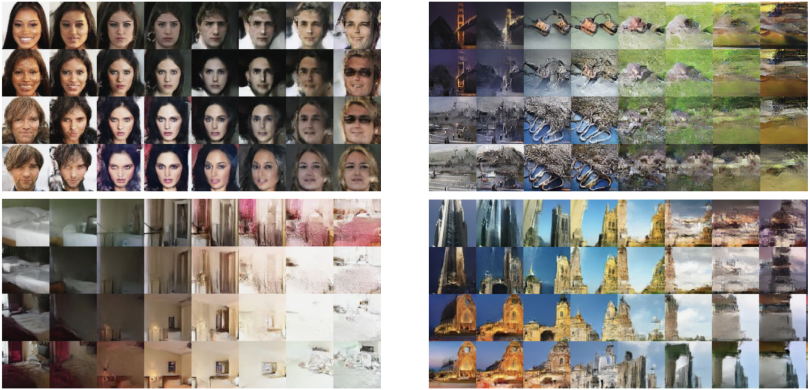
$$\det\left(\frac{\partial \mathbf{z}}{\partial \mathbf{x}}\right) = \det\begin{pmatrix} \mathbf{I}_d & 0_{d \times m-d} \\ \frac{\partial \mathbf{z}_{d:m}}{\partial \mathbf{x}_{1:d}} & \frac{\partial \mathbf{z}_{d:m}}{\partial \mathbf{x}_{d:m}} \end{pmatrix} = \prod_{i=1}^{m-d} \exp(c_1(\mathbf{x}_{1:d}, \boldsymbol{\theta})_i).$$

Non-Volume Preserving (the determinant of Jacobian $\neq 0$).

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

RealNVP

Flow samples



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

MAF vs IAF vs RealNVP

MADE/MAF

$$\mathbf{x} = \boldsymbol{\sigma}(\mathbf{x}) \odot \mathbf{z} + \boldsymbol{\mu}(\mathbf{x}).$$

Estimating the density $p(\mathbf{x}|\boldsymbol{\theta})$ - 1 pass, sampling - m passes.

IAF

$$\mathbf{x} = \tilde{\boldsymbol{\sigma}}(\mathbf{z}) \odot \mathbf{z} + \tilde{\boldsymbol{\mu}}(\mathbf{z}).$$

Estimating the density $p(\mathbf{x}|\boldsymbol{\theta})$ - m passes, sampling - 1 pass.

RealNVP

$$\begin{cases} \mathbf{x}_{1:d} & = \mathbf{z}_{1:d}; \\ \mathbf{x}_{d:m} & = \mathbf{z}_{d:m} \odot \exp\left(c_1(\mathbf{z}_{1:d}, \boldsymbol{\theta})\right) + c_2(\mathbf{z}_{1:d}, \boldsymbol{\theta}). \end{cases}$$

Estimating the density $p(\mathbf{x}|\boldsymbol{\theta})$ - 1 pass, sampling - 1 pass.

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

MAF vs IAF vs RealNVP

RealNVP

$$\begin{cases} \mathbf{x}_{1:d} &= \mathbf{z}_{1:d}; \\ \mathbf{x}_{d:m} &= \mathbf{z}_{d:m} \odot \exp\left(c_1(\mathbf{z}_{1:d}, \boldsymbol{\theta})\right) + c_2(\mathbf{z}_{1:d}, \boldsymbol{\theta}). \end{cases}$$

- Calculating the density $p(\mathbf{x}|\boldsymbol{\theta})$ - 1 pass.
- Sampling - 1 pass.

RealNVP is a special case of MAF and IAF:

MAF

$$\begin{cases} \mu_i = 0, \sigma_i = 1, \; i = 1, \dots, d; \\ \mu_i, \sigma_i \text{ – functions of } \mathbf{x}_{1:d}, \; i = d+1, \dots, m. \end{cases}$$

IAF

$$\begin{cases} \tilde{\mu}_i = 0, \tilde{\sigma}_i = 1, \; i = 1, \dots, d; \\ \tilde{\mu}_i, \tilde{\sigma}_i \text{ – functions of } \mathbf{z}_{1:d}, \; i = d+1, \dots, m. \end{cases}$$

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

MAF/IAF pros and cons

MAF

- Sampling is slow.
- Likelihood evaluation is fast.

IAF

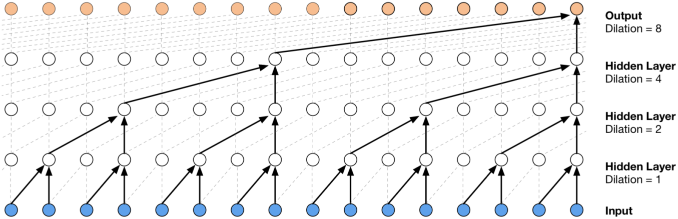
- Sampling is fast.
- Likelihood evaluation is slow.

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.

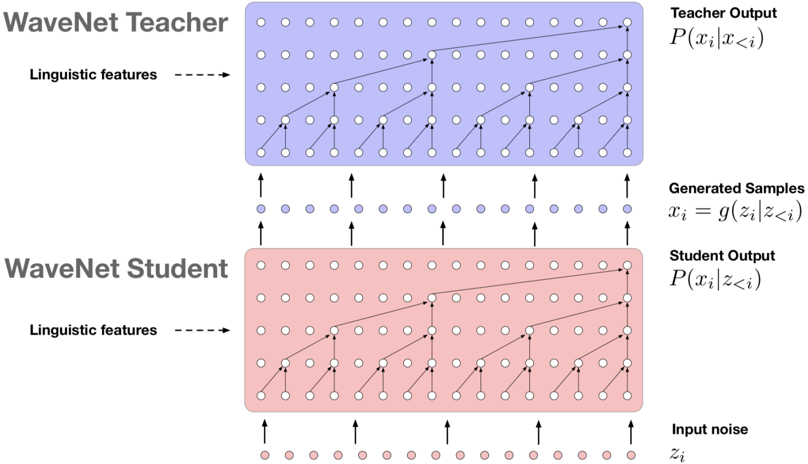
Student objective

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

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

Summary

- Autoregressive flows use autoregressive transformation to make the Jacobian triangular.
- MAF/IAF is a special case of autoregressive flows.
- The RealNVP is an effective type of flow (special case of AR flows) that uses coupling layer.