Deep Generative Models

Lecture 2

Roman Isachenko

Ozon Masters

Spring, 2022

Recap of previous lecture

We are given i.i.d. samples $\{\mathbf{x}_i\}_{i=1}^n \in \mathcal{X}$ (e.g. $\mathcal{X} = \mathbb{R}^m$) from unknown distribution $\pi(\mathbf{x})$.

Goal

We would like to learn a distribution $\pi(\mathbf{x})$ for

- evaluating $\pi(\mathbf{x})$ for new samples (how likely to get object $\mathbf{x}$?);
- sampling from $\pi(\mathbf{x})$ (to get new objects $\mathbf{x} \sim \pi(\mathbf{x})$).

Instead of searching true $\pi(\mathbf{x})$ over all probability distributions, learn function approximation $p(\mathbf{x}|\boldsymbol{\theta}) \approx \pi(\mathbf{x})$.

Divergence

- $D(\pi||p) \geq 0$ for all $\pi, p \in \mathcal{S}$;
- $D(\pi||p) = 0$ if and only if $\pi \equiv p$.

General divergence minimization task

$$\min_{\boldsymbol{\theta}} D(\pi||p).$$

Recap of previous lecture

Forward KL

$$KL(\pi || p) = \int \pi(\mathbf{x}) \log \frac{\pi(\mathbf{x})}{p(\mathbf{x}|\boldsymbol{\theta})} d\mathbf{x} \rightarrow \min_{\boldsymbol{\theta}}$$

Reverse KL

$$KL(p || \pi) = \int p(\mathbf{x}|\boldsymbol{\theta}) \log \frac{p(\mathbf{x}|\boldsymbol{\theta})}{\pi(\mathbf{x})} d\mathbf{x} \rightarrow \min_{\boldsymbol{\theta}}$$

Maximum likelihood estimation (MLE)

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

Maximum likelihood estimation is equivalent to minimization of the Monte-Carlo estimate of forward KL.

Recap of previous lecture

Likelihood as product of conditionals

Let $\mathbf{x} = (x_1, \dots, x_m)$, $\mathbf{x}_{1:j} = (x_1, \dots, x_j)$. Then

$$p(\mathbf{x}|\boldsymbol{\theta}) = \prod_{j=1}^{m} p(x_j|\mathbf{x}_{1:j-1}, \boldsymbol{\theta}); \quad \log p(\mathbf{x}|\boldsymbol{\theta}) = \sum_{j=1}^{m} \log p(x_j|\mathbf{x}_{1:j-1}, \boldsymbol{\theta}).$$

MLE problem for autoregressive model

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} p(\mathbf{X}|\boldsymbol{\theta}) = \arg\max_{\boldsymbol{\theta}} \sum_{i=1}^{n} \sum_{j=1}^{m} \log p(x_{ij}|\mathbf{x}_{i,1:j-1}\boldsymbol{\theta}).$$

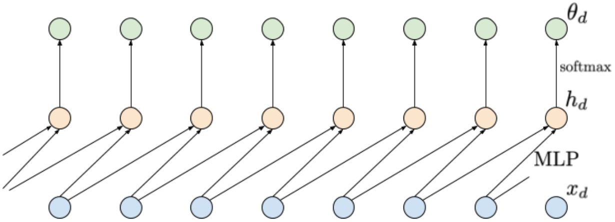
Sampling

$$\hat{x}_1 \sim p(x_1|\boldsymbol{\theta}), \quad \hat{x}_2 \sim p(x_2|\hat{x}_1, \boldsymbol{\theta}), \dots, \quad \hat{x}_m \sim p(x_m|\hat{\mathbf{x}}_{1:m-1}, \boldsymbol{\theta})$$

New generated object is $\hat{\mathbf{x}} = (\hat{x}_1, \hat{x}_2, \dots, \hat{x}_m)$.

Recap of previous lecture

Autoregressive MLP



Autoregressive RNN

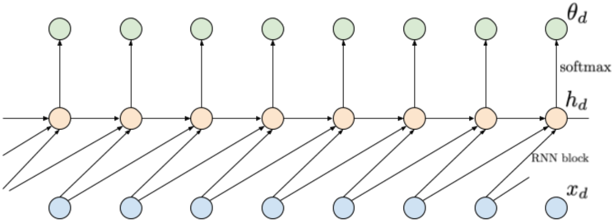


image credit: https://jmtomczak.github.io/blog/2/2_ARM.html

Outline

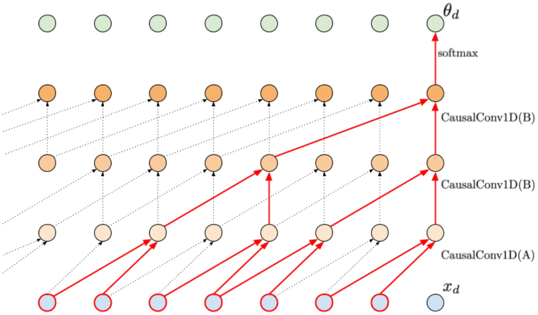
1. Autoregressive models (WaveNet, PixelCNN, PixelCNN++)
2. Bayesian framework
3. Latent variable models (LVM)

Outline

1. Autoregressive models (WaveNet, PixelCNN, PixelCNN++)
2. Bayesian framework
3. Latent variable models (LVM)

Autoregressive models

- Convolutions could be used for autoregressive models, but they have to be **causal**.
- Try to find and understand the difference between Conv A/B.



- Could learn long-range dependecies.
- Do not suffer from gradient issues.
- Easy to estimate probability for given input, but hard generation of new samples (the sequential process).

image credit: https://jmtomczak.github.io/blog/2/2_ARM.html

WaveNet

Goal

Efficient generation of raw audio waveforms with natural sounds.

Solution

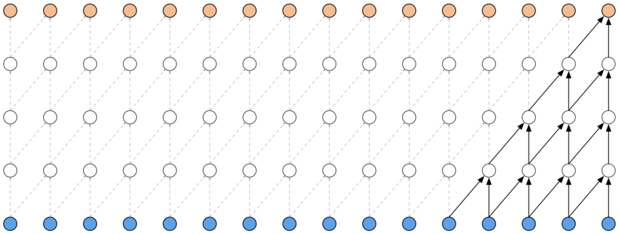
Autoregressive model

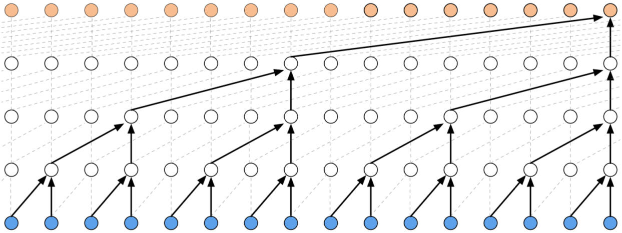
$$p(\mathbf{x}|\boldsymbol{\theta}) = \prod_{t=1}^{T} p(x_t|\mathbf{x}_{1:t-1}, \boldsymbol{\theta}).$$

- Each conditional $p(x_t|\mathbf{x}_{1:t-1}, \boldsymbol{\theta})$ models the distribution for the timestamp t.
- The model uses **causal** dilated convolutions.

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

WaveNet





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

PixelCNN

Goal

Model a distribution $\pi(\mathbf{x})$ of natural images.

Solution

Autoregressive model on 2D pixels

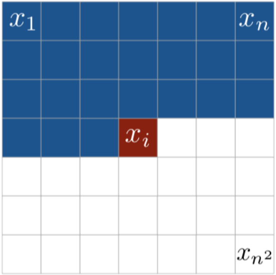
$$p(\mathbf{x}|\boldsymbol{\theta}) = \prod_{j=1}^{\text{width}\times\text{height}} p(x_j|\mathbf{x}_{1:j-1}, \boldsymbol{\theta}).$$

- We need to introduce the ordering of image pixels.
- The convolution should be **masked** to make them causal.
- The image has RGB channels, these dependencies could be addressed.

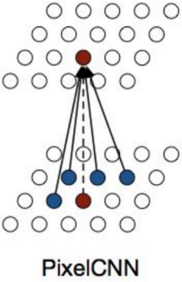
Oord A., Kalchbrenner N., Kavukcuoglu K. Pixel recurrent neural networks, 2016

PixelCNN

Raster ordering

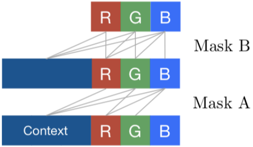


Dependencies between pixels



Masked convolution kernel

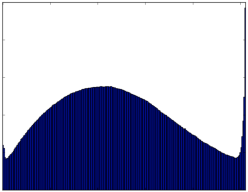
1	1	1
1	0	0
0	0	0



Oord A., Kalchbrenner N., Kavukcuoglu K. Pixel recurrent neural networks, 2016

PixelCNN++

CIFAR-10 pixel values distribution



- Standard PixelCNN outputs softmax probabilities for values $\{0, 255\}$ (256 outputs feature maps).
- Categorical distribution do not know anything about numerical relationships (220 is close to 221 and far from 15).
- If pixel value is not presented in the training dataset, it won't be predicted.
- (Look at the edges of the distributions: they have higher probability mass).

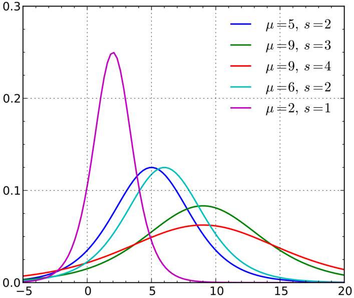
Salimans T. et al. PixelCNN++: Improving the PixelCNN with Discretized Logistic Mixture Likelihood and Other Modifications, 2017

PixelCNN++

Mixture of logistic distributions

$$p(x|\mu, s) = \frac{\exp^{-(x-\mu)/s}}{s(1+\exp^{-(x-\mu)/s})^2};$$

$$p(x|\boldsymbol{\mu}, \mathbf{s}, \boldsymbol{\pi}) = \sum_{k=1}^{K} \pi_k p(x|\mu_k, s_k);$$



To adopt probability calculation to discrete values:

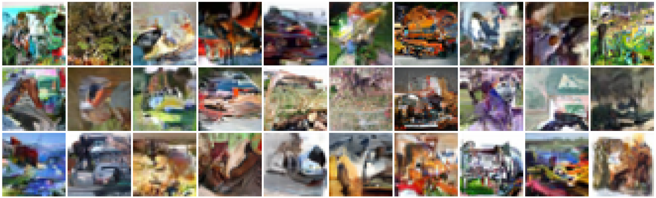
$$P_d(x|\boldsymbol{\mu}, \mathbf{s}, \boldsymbol{\pi}) = P(x+0.5|\boldsymbol{\mu}, \mathbf{s}, \boldsymbol{\pi}) - P(x-0.5|\boldsymbol{\mu}, \mathbf{s}, \boldsymbol{\pi})$$

For the edge case of 0, replace $x - 0.5$ by $-\infty$, and for 255 replace $x + 0.5$ by $+\infty$.

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

CIFAR-10 generated samples

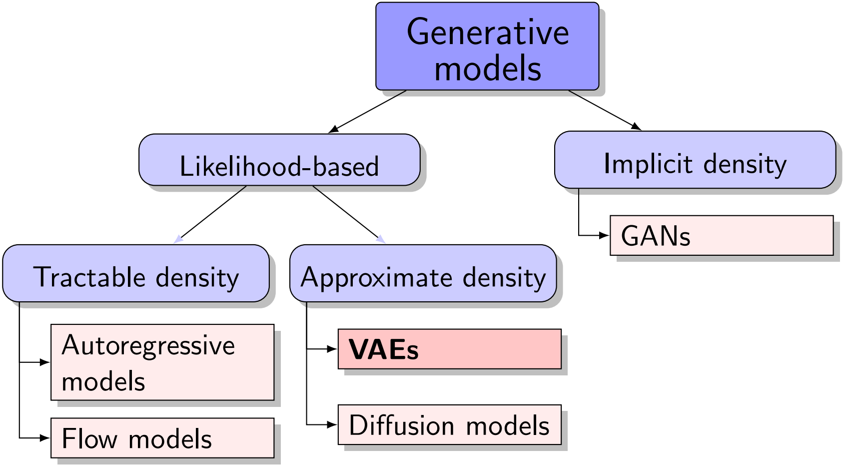
PixelCNN



PixelCNN++

Oord A., Kalchbrenner N., Kavukcuoglu K. Pixel recurrent neural networks, 2016
Salimans T. et al. PixelCNN++: Improving the PixelCNN with Discretized Logistic Mixture Likelihood and Other Modifications, 2017

Generative models zoo



Outline

Bayesian framework

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.

Meaning

We have unobserved variables $\mathbf{t}$ and some prior knowledge about them $p(\mathbf{t})$. Then, the data $\mathbf{x}$ has been observed. Posterior distribution $p(\mathbf{t}|\mathbf{x})$ summarizes the knowledge after the observations.

Bayesian framework

Let consider the case, where the unobserved variables $\mathbf{t}$ is our model parameters $\boldsymbol{\theta}$.

- $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^n$ – observed samples;
- $p(\boldsymbol{\theta})$ – prior parameters distribution (we treat model parameters $\boldsymbol{\theta}$ as random variables).

Posterior distribution

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

Bayesian inference

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

Note the difference from

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

Bayesian framework

Posterior distribution

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

Bayesian inference

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

If evidence $p(\mathbf{X})$ is intractable (due to multidimensional integration), we can't get posterior distribution and perform the precise inference.

Maximum a posteriori (MAP) estimation

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} p(\boldsymbol{\theta}|\mathbf{X}) = \arg\max_{\boldsymbol{\theta}} \big(\log p(\mathbf{X}|\boldsymbol{\theta}) + \log p(\boldsymbol{\theta})\big)$$

Bayesian framework

MAP estimation

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} p(\boldsymbol{\theta}|\mathbf{X}) = \arg\max_{\boldsymbol{\theta}} \big(\log p(\mathbf{X}|\boldsymbol{\theta}) + \log p(\boldsymbol{\theta})\big)$$

Estimated $\boldsymbol{\theta}^*$ is a deterministic variable, but we could treat it as a random variable with density $p(\boldsymbol{\theta}|\mathbf{X}) = \delta(\boldsymbol{\theta} - \boldsymbol{\theta}^*)$.

Dirac delta function

$$\delta(x) = \begin{cases} +\infty, & x = 0; \\ 0, & x \neq 0; \end{cases} \quad \int \delta(x)dx = 1; \int f(x)\delta(x-y)dx = f(y).$$

MAP inference

$$p(\mathbf{x}|\mathbf{X}) = \int p(\mathbf{x}|\boldsymbol{\theta})p(\boldsymbol{\theta}|\mathbf{X})d\boldsymbol{\theta} \approx p(\mathbf{x}|\boldsymbol{\theta}^*).$$

Outline

1. Autoregressive models (WaveNet, PixelCNN, PixelCNN++)
2. Bayesian framework
3. Latent variable models (LVM)

Latent variable models (LVM)

MLE problem

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

The distribution $p(\mathbf{x}|\boldsymbol{\theta})$ could be very complex and intractable (as well as real distribution $\pi(\mathbf{x})$).

Extended probabilistic model

Introduce latent variable $\mathbf{z}$ for each sample $\mathbf{x}$

$$p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta}) = p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta})p(\mathbf{z}); \quad \log p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta}) = \log p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta}) + \log p(\mathbf{z}).$$

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

Motivation

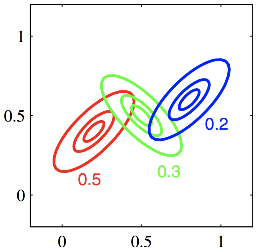
The distributions $p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta})$ and $p(\mathbf{z})$ could be quite simple.

Latent variable models (LVM)

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

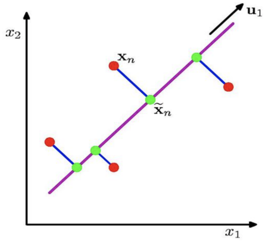
Examples

Mixture of gaussians



- $p(\mathbf{x}|z, \boldsymbol{\theta}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_z, \boldsymbol{\Sigma}_z)$
- $p(z) = \text{Categorical}(\boldsymbol{\pi})$

PCA model



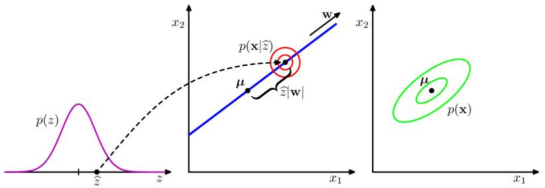
- $p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta}) = \mathcal{N}(\mathbf{x}|\mathbf{W}\mathbf{z} + \boldsymbol{\mu}, \sigma^2\mathbf{I})$
- $p(\mathbf{z}) = \mathcal{N}(\mathbf{z}|0, \mathbf{I})$

Bishop C. Pattern Recognition and Machine Learning, 2006

Latent variable models (LVM)

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

PCA projects original data **X** onto a low dimensional latent space while maximizing the variance of the projected data.



$\boldsymbol{x} = W\boldsymbol{z} + \boldsymbol{\mu} + \boldsymbol{\epsilon}$

- $p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta}) = \mathcal{N}(\mathbf{x}|\mathbf{W}\mathbf{z} + \boldsymbol{\mu}, \sigma^2\mathbf{I})$
- $p(\mathbf{z}) = \mathcal{N}(\mathbf{z}|0, \mathbf{I})$
- $p(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \mathbf{W}\mathbf{W}^T + \sigma^2\mathbf{I})$
- $p(\mathbf{z}|\mathbf{x}) = \mathcal{N}\big(\mathbf{M}^{-1}\mathbf{W}^T(\mathbf{x} - \boldsymbol{\mu}), \sigma^2\mathbf{M}\big)$, where $\mathbf{M} = \mathbf{W}\mathbf{W}^T + \sigma^2\mathbf{I}$

Maximum likelihood estimation for LVM

MLE for extended problem

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta}) = \arg\max_{\boldsymbol{\theta}} \prod_{i=1}^{n} p(\mathbf{x}_i, \mathbf{z}_i|\boldsymbol{\theta}) =$$

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

However, $\mathbf{Z}$ is unknown.

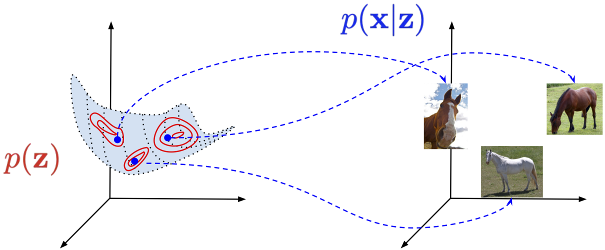
MLE for original problem

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

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

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

Naive approach



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

Challenge: to cover the space properly, the number of samples grows exponentially with respect to dimensionality of $\mathbf{z}$.

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

Summary

- WaveNet and PixelCNN models use masked causal convolutions (1D or 2D) to get autoregressive model.
- PixelCNN++ proposes to use discretized mixture of logistics for output distribution.
- Bayesian inference is a generalization of most common machine learning tasks. It allows to construct MLE, MAP and bayesian inference, to compare models complexity and many-many more cool stuff.
- LVM introduces latent representation of observed samples to make model more interpretable.