Deep Generative Models

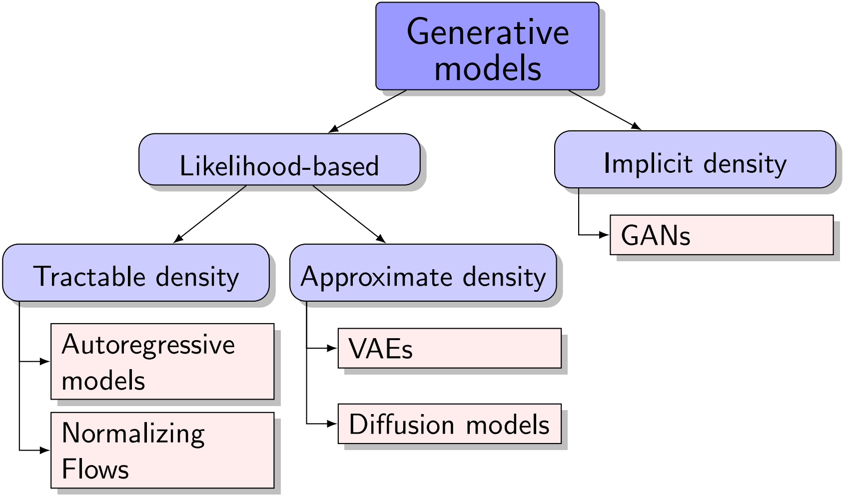
Lecture 1

Roman Isachenko

AI Masters

2024, Spring

Generative models zoo



Outline

1. Generative models overview
2. Problem statement
3. Divergence minimization framework
4. Autoregressive modelling

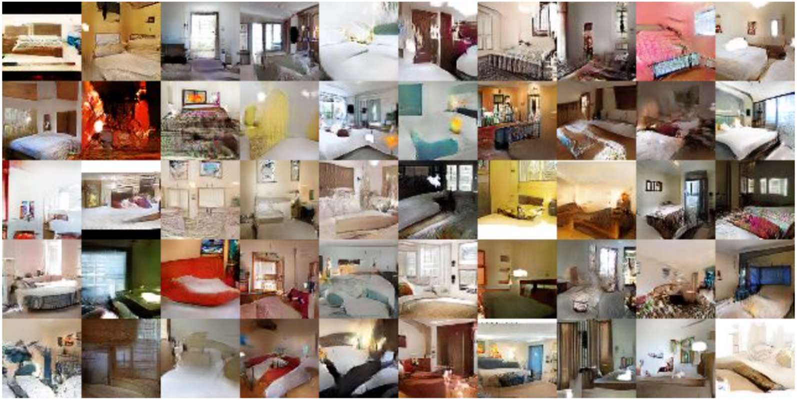
Outline

1. Generative models overview
2. Problem statement
3. Divergence minimization framework
4. Autoregressive modelling

VAE – first scalable approach for image generation

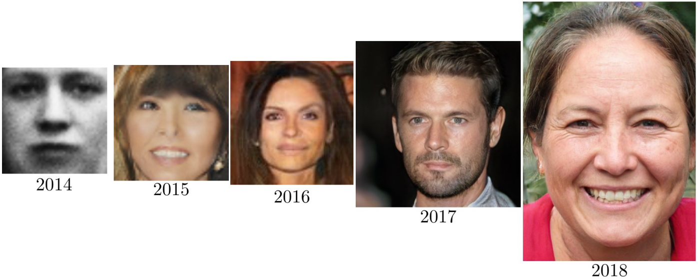
Kingma D. P., Welling M. *Auto-encoding variational bayes*, 2013

DCGAN – first convolutional GAN for image generation



Radford A., Metz L., Chintala S. Unsupervised representation learning with deep convolutional generative adversarial networks, 2015

StyleGAN – high quality generation of faces

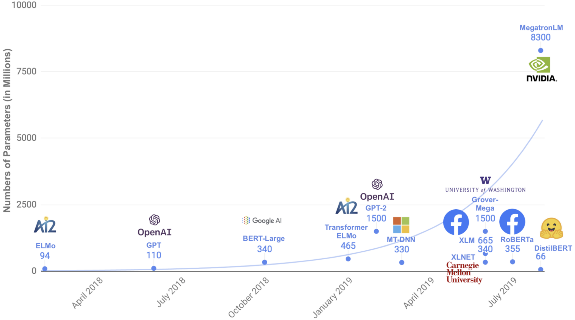


Karras T., Laine S., Aila T. A style-based generator architecture for generative adversarial networks, 2018

VQ-VAE-2 – high quality generation without GANs

Razavi A., Oord A., Vinyals O. Generating Diverse High-Fidelity Images with VQ-VAE-2, 2019

Language modelling at scale



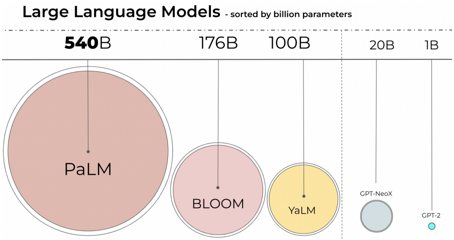


image credit: http://jalammar.github.io/illustrated-gpt2
image credit: https://huggingface.co/blog/hf-bitsandbytes-integration

Language modelling at scale

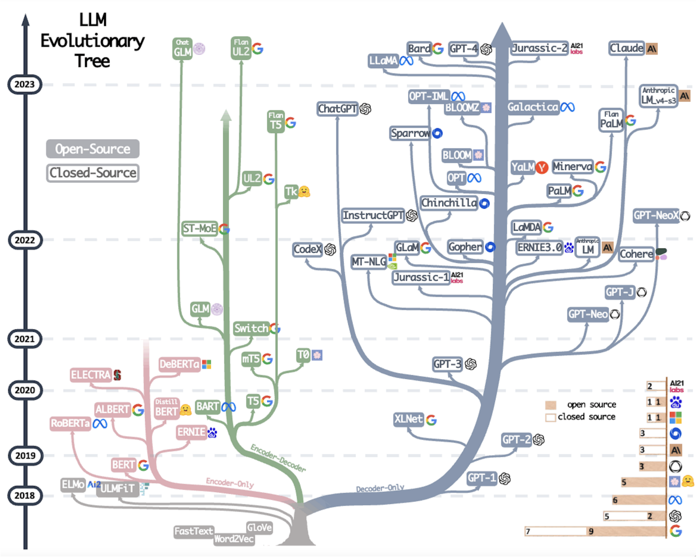


image credit:
https://blog.biocomm.ai/2023/05/14/open-source-proliferation-llm-evolutionary-tree/

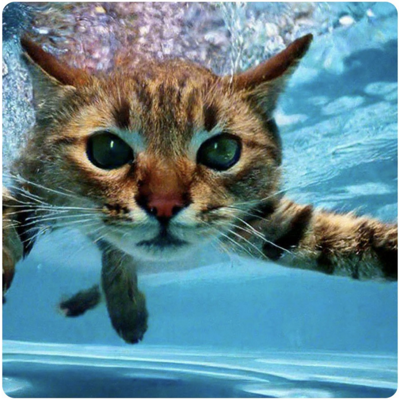
DDPM - diffusion model

Dhariwal P., Nichol A. Diffusion Models Beat GANs on Image Synthesis, 2021

Stable Diffusion - awesome text-to-image results

Rombach R., et al. High-Resolution Image Synthesis with Latent Diffusion Models, 2021
https://github.com/CompVis/stable-diffusion
LAION-5B dataset

Shedevrum - Yandex generative model



Кот ныряет в бассейн, как ребенок на обложке альбома Nevermind, реалистично

рука человека с пятью пальцами, ни четыремя, ни шестью, а с 5 (пять) пальцами

Outline

1. Generative models overview
2. Problem statement
3. Divergence minimization framework
4. Autoregressive modelling

Course tricks 1

Let $\mathbf{x} \in \mathbb{R}^m$ be a random variable with a density $p_x(\mathbf{x})$.

Law of the unconscious statistician (LOTUS)

Let $\mathbf{y} = f(\mathbf{x})$ with density $p_y(\mathbf{y})$. Then

$$\mathbb{E}_{p_y} g(\mathbf{y}) = \int p_y(\mathbf{y}) g(\mathbf{y}) d\mathbf{y} = \int p_x(\mathbf{x}) g(f(\mathbf{x})) d\mathbf{x} = \mathbb{E}_{p_x} g(f(\mathbf{x})).$$

Monte-Carlo estimation

Expected value could be estimated using only the samples:

$$\mathbb{E}_{p(\mathbf{x})} f(\mathbf{x}) = \int p(\mathbf{x}) f(\mathbf{x}) d\mathbf{x} \approx \frac{1}{n} \sum_{i=1}^{n} f(\mathbf{x}_i), \quad \text{where } \mathbf{x}_i \sim p(\mathbf{x}).$$

Jensen's Inequality

Assume $f(\cdot)$ is a convex function. Then

$$\mathbb{E}[f(\mathbf{x})] \geq f(\mathbb{E}[\mathbf{x}]).$$

Course tricks 2

Let $\mathbf{x} \in \mathbb{R}^m$ be a random variable with a density $p_x(\mathbf{x})$.

Decomposition to conditionals

$$p(\mathbf{x}) = p(x_1) \cdot p(x_2|x_1) \cdot p(x_3|x_2, x_1) \cdot \dots \cdot p(x_m|x_{m-1}, \dots, x_1).$$

Log-derivative trick

$$\nabla \log f(\mathbf{x}) = \frac{1}{f(\mathbf{x})} \cdot \nabla f(\mathbf{x}).$$

Dirac delta function

$$\delta(\mathbf{x}) = \begin{cases} +\infty, & \mathbf{x} = 0; \\ 0, & \mathbf{x} \neq 0; \end{cases} \quad \int \delta(\mathbf{x}) d\mathbf{x} = 1; \; \int f(\mathbf{x}) \delta(\mathbf{x} - \mathbf{x}_0) dx = f(\mathbf{x}_0).$$

We could treat any deterministic variable $\mathbf{x}_0$ as a random variable with density $p(\mathbf{x}) = \delta(\mathbf{x} - \mathbf{x}_0)$.

Problem statement

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 $\mathbf{x}$?) – **density evaluation**;
- sampling from $\pi(\mathbf{x})$ (to get new objects $\mathbf{x} \sim \pi(\mathbf{x})$) – **generation**.

Challenge

Data is complex and high-dimensional. E.g. the dataset of images lies in the space $\mathcal{X} \subset \mathbb{R}^{\text{width} \times \text{height} \times \text{channels}}$.

Problem statement

Conditional model

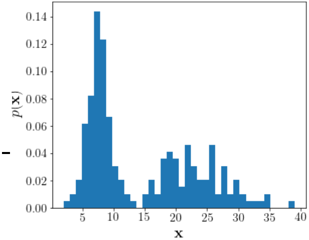
In practice the popular task is to create a conditional model $\pi(\mathbf{x}|\mathbf{y})$.

- $\mathbf{y}$ – class label, $\mathbf{x}$ – image $\Rightarrow$ image conditional model.
- $\mathbf{y}$ – text prompt, $\mathbf{x}$ – image $\Rightarrow$ text-to-image model.
- $\mathbf{y}$ – image, $\mathbf{x}$ – image $\Rightarrow$ image-to-image model.
- $\mathbf{y}$ – image, $\mathbf{x}$ – text $\Rightarrow$ text-to-image model (image captioning).
- $\mathbf{y}$ – sound, $\mathbf{x}$ – text $\Rightarrow$ speech-to-text model (automatic speech recognition).
- $\mathbf{y}$ – English text, $\mathbf{x}$ – Russian text $\Rightarrow$ sequence-to-sequence model (machine translation).
- $\mathbf{y} = \emptyset$, $\mathbf{x}$ – image $\Rightarrow$ image unconditional model.

Histogram as a generative model

Let $x \sim \text{Categorical}(\boldsymbol{\pi})$. The histogram is totally defined by

$$\pi_k = \pi(x = k) = \frac{\sum_{i=1}^{n}[x_i = k]}{n}.$$



Problem: curse of dimensionality (number of bins grows exponentially).

MNIST example: 28x28 gray-scaled images, each image is $\mathbf{x} = (x_1, \dots, x_{784})$, where $x_i = \{0, 1\}$.

$$\pi(\mathbf{x}) = \pi(x_1) \cdot \pi(x_2|x_1) \cdot \dots \cdot \pi(x_m|x_{m-1}, \dots, x_1).$$

Hence, the histogram will have $2^{28 \times 28} - 1$ parameters to specify $\pi(\mathbf{x})$.

Question: How many parameters do we need in these cases?

$$\pi(\mathbf{x}) = \pi(x_1) \cdot \pi(x_2) \cdot \dots \cdot \pi(x_m);$$
$$\pi(\mathbf{x}) = \pi(x_1) \cdot \pi(x_2|x_1) \cdot \dots \cdot \pi(x_m|x_{m-1}).$$

Outline

1. Generative models overview
2. Problem statement
3. Divergence minimization framework
4. Autoregressive modelling

Divergences

Fix probabilistic model $p(\mathbf{x}|\boldsymbol{\theta})$ – the set of parameterized distributions.
Instead of searching true $\pi(\mathbf{x})$ over all probability distributions, learn function approximation $p(\mathbf{x}|\boldsymbol{\theta}) \approx \pi(\mathbf{x})$.

What is a divergence?

Let $\mathcal{S}$ be the set of all possible probability distributions. Then $D : \mathcal{S} \times \mathcal{S} \to \mathbb{R}$ is a divergence if

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

Divergence minimization task

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

where $\pi(\mathbf{x})$ is a true data distribution, $p(\mathbf{x}|\boldsymbol{\theta})$ is a model distribution.

f-divergence family

f-divergence

$$D_f(\pi||p) = \mathbb{E}_{p(\mathbf{x})} f\left(\frac{\pi(\mathbf{x})}{p(\mathbf{x})}\right) = \int p(\mathbf{x}) f\left(\frac{\pi(\mathbf{x})}{p(\mathbf{x})}\right) d\mathbf{x}.$$

Here $f : \mathbb{R}_+ \to \mathbb{R}$ is a convex, lower semicontinuous function satisfying $f(1) = 0$.

Name	$D_f(P\|Q)$	Generator $f(u)$
Kullback-Leibler	$\int p(x) \log \frac{p(x)}{q(x)} \, \mathrm{d}x$	$u \log u$
Reverse KL	$\int q(x) \log \frac{q(x)}{p(x)} \, \mathrm{d}x$	$-\log u$
Pearson χ^2	$\int \frac{(q(x)-p(x))^2}{p(x)} \, \mathrm{d}x$	$(u-1)^2$
Squared Hellinger	$\int \left(\sqrt{p(x)} - \sqrt{q(x)}\right)^2 \mathrm{d}x$	$(\sqrt{u}-1)^2$
Jensen-Shannon	$\frac{1}{2} \int p(x) \log \frac{2p(x)}{p(x)+q(x)} + q(x) \log \frac{2q(x)}{p(x)+q(x)} \, \mathrm{d}x$	$-(u+1) \log \frac{1+u}{2} + u \log u$
GAN	$\int p(x) \log \frac{2p(x)}{p(x)+q(x)} + q(x) \log \frac{2q(x)}{p(x)+q(x)} \, \mathrm{d}x - \log(4)$	$u \log u - (u+1) \log(u+1)$

Nowozin S., Cseke B., Tomioka R. f-GAN: Training Generative Neural Samplers using Variational Divergence Minimization, 2016

Forward KL vs Reverse KL

Forward KL

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

Reverse KL

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

What is the difference between these two formulations?

Maximum likelihood estimation (MLE)

Let $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^n$ be the set of the given i.i.d. samples.

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

Forward KL vs Reverse KL

Forward KL

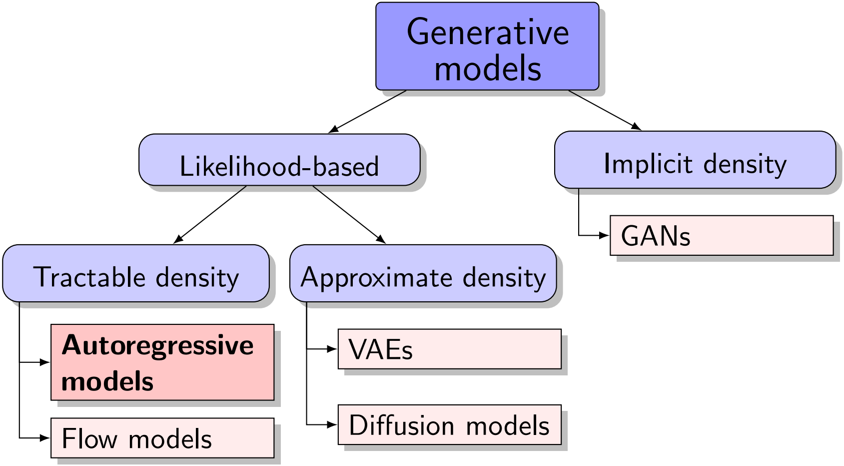
$$
\begin{aligned}
KL(\pi || p) &= \int \pi(\mathbf{x}) \log \frac{\pi(\mathbf{x})}{p(\mathbf{x}|\boldsymbol{\theta})} d\mathbf{x} \\
&= \int \pi(\mathbf{x}) \log \pi(\mathbf{x}) d\mathbf{x} - \int \pi(\mathbf{x}) \log p(\mathbf{x}|\boldsymbol{\theta}) d\mathbf{x} \\
&= -\mathbb{E}_{\pi(\mathbf{x})} \log p(\mathbf{x}|\boldsymbol{\theta}) + \text{const} \\
&\approx -\frac{1}{n} \sum_{i=1}^{n} \log p(\mathbf{x}_i|\boldsymbol{\theta}) + \text{const} \rightarrow \min_{\boldsymbol{\theta}}.
\end{aligned}
$$

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

Reverse KL

$$
\begin{aligned}
KL(p || \pi) &= \int p(\mathbf{x}|\boldsymbol{\theta}) \log \frac{p(\mathbf{x}|\boldsymbol{\theta})}{\pi(\mathbf{x})} d\mathbf{x} \\
&= \mathbb{E}_{p(\mathbf{x}|\boldsymbol{\theta})} \left[\log p(\mathbf{x}|\boldsymbol{\theta}) - \log \pi(\mathbf{x})\right] \rightarrow \min_{\boldsymbol{\theta}}
\end{aligned}
$$

Generative models zoo



Outline

Autoregressive modelling

MLE problem

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

- We would like to solve the problem using gradient-based optimization.
- We have to efficiently compute $\log p(\mathbf{x}|\boldsymbol{\theta})$ and $\frac{\partial \log p(\mathbf{x}|\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}$.

Likelihood as product of conditionals

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

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

Example: $p(x_1, x_2, x_3) = p(x_2) \cdot p(x_1|x_2) \cdot p(x_3|x_1, x_2)$.

Autoregressive models

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

- Sampling is sequential:
  - sample $\hat{x}_1 \sim p(x_1|\boldsymbol{\theta})$;
  - sample $\hat{x}_2 \sim p(x_2|\hat{x}_1, \boldsymbol{\theta})$;
  - ...
  - sample $\hat{x}_m \sim p(x_m|\hat{\mathbf{x}}_{1:m-1}, \boldsymbol{\theta})$;
  - new generated object is $\hat{\mathbf{x}} = (\hat{x}_1, \hat{x}_2, \dots, \hat{x}_m)$.
- Each conditional $p(x_j|\mathbf{x}_{1:j-1}, \boldsymbol{\theta})$ could be modelled by neural network.
- Modelling all conditional distributions separately is infeasible and we would obtain separate models. To extend to high dimensions we could share parameters $\boldsymbol{\theta}$ across conditionals.

Autoregressive models: MLP

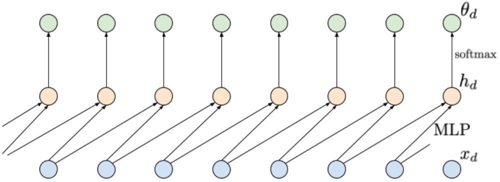
For large j the conditional distribution $p(x_j|\mathbf{x}_{1:j-1}, \boldsymbol{\theta})$ could be infeasible. Moreover, the history $\mathbf{x}_{1:j-1}$ has non-fixed length.

Markov assumption

$$p(x_j|\mathbf{x}_{1:j-1}, \boldsymbol{\theta}) = p(x_j|\mathbf{x}_{j-d:j-1}, \boldsymbol{\theta}), \quad d \text{ is a fixed model parameter.}$$

Example

- $d = 2$;
- $x_j \in \{0, 255\}$;
- $\mathbf{h}_j = \text{MLP}_{\boldsymbol{\theta}}(x_{j-1}, x_{j-2})$;
- $\boldsymbol{\pi}_j = \text{softmax}(\mathbf{h}_j)$;
- $p(x_j|x_{j-1}, x_{j-2}, \boldsymbol{\theta}) = \text{Categorical}(\boldsymbol{\pi}_j)$.



Is it possible to model continuous distributions instead of discrete one?

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

Autoregressive models: PixelCNN

Goal

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

Solution

Autoregressive model on 2D pixels

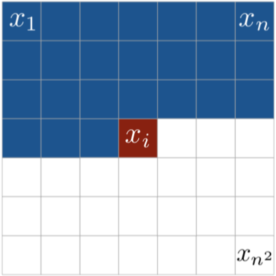
$$p(\mathbf{x}|\boldsymbol{\theta}) = \prod_{j=1}^{\text{width} \times \text{height}} p(x_j|\mathbf{x}_{1:j-1}, \boldsymbol{\theta}).$$

- 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.

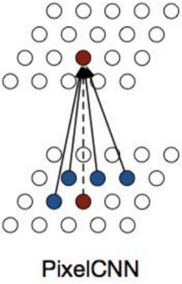
Oord A., Kalchbrenner N., Kavukcuoglu K. Pixel recurrent neural networks, 2016

Autoregressive models: PixelCNN

Raster ordering

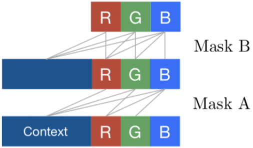


Dependencies between pixels



Mask for the convolution kernel

1	1	1
1	0	0
0	0	0



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

Summary

- We are trying to approximate the distribution of samples for density estimation and generation of new samples.
- To fit model distribution to the real data distribution one could use divergence minimization framework.
- Minimization of forward KL is equivalent to the MLE problem.
- Autoregressive models decompose the distribution to the sequence of the conditionals.
- Sampling from the autoregressive models is trivial, but sequential!
- To estimate density you need to multiply all conditionals $p(x_j|\mathbf{x}_{1:j-1}, \boldsymbol{\theta})$.
- PixelCNN model use masked causal convolutions (1D or 2D) to get autoregressive model.