

Deep Generative Models

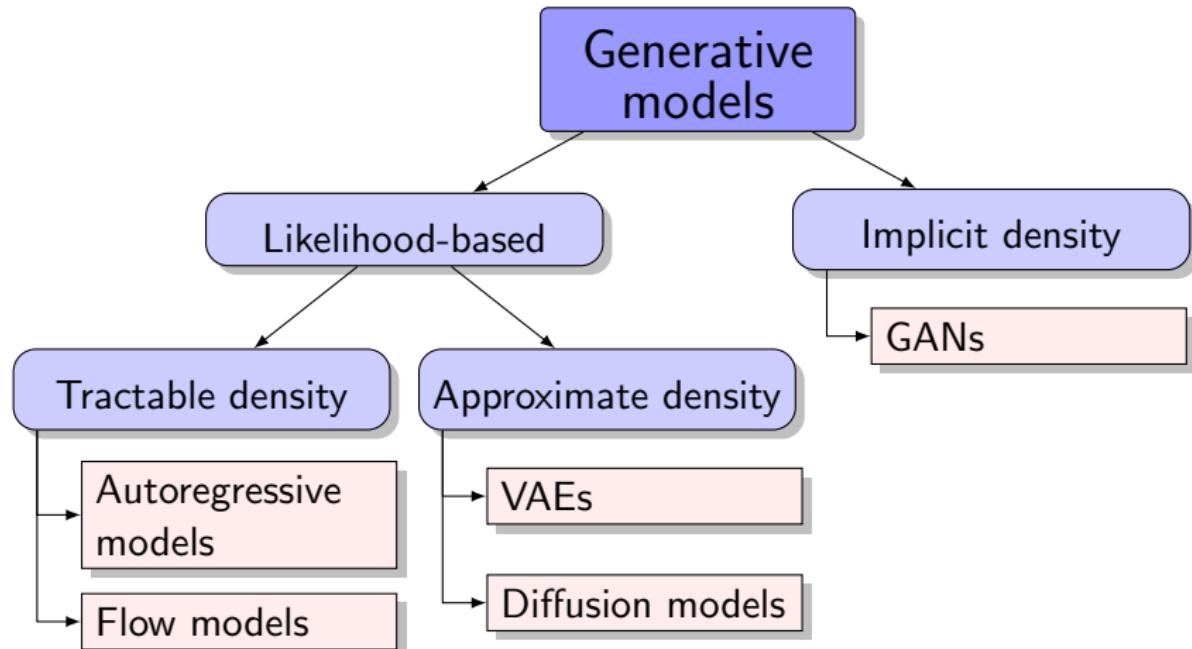
Lecture 1

Roman Isachenko

Moscow Institute of Physics and Technology

Autumn, 2021

Generative models zoo

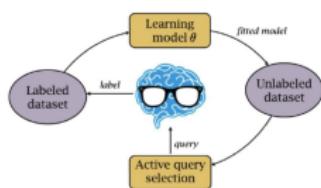


Applications

" i want to talk to you . "
" i want to be with you . "
" i do n't want to be with you . "
i do n't want to be with you .
she did n't want to be with him .

he was silent for a long moment .
he was silent for a moment .
it was quiet for a moment .
it was dark and cold .
there was a pause .
it was my turn .

Text analysis



Active Learning

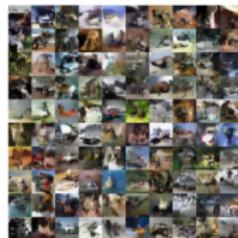
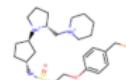
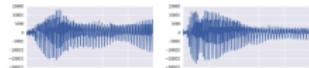


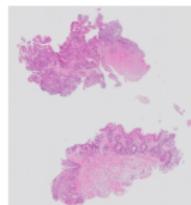
Image analysis



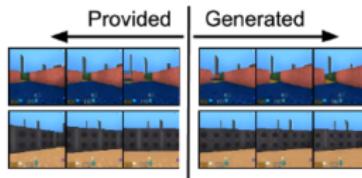
Graph analysis



Audio analysis



Medical data



Reinforcement Learning

and more...

Applications: Image generation (VAE)



Applications: Image generation (DCGAN)



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

Applications: Face generation (StyleGAN)



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

Applications: Face generation (VQ-VAE-2)



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

Applications: Language modelling

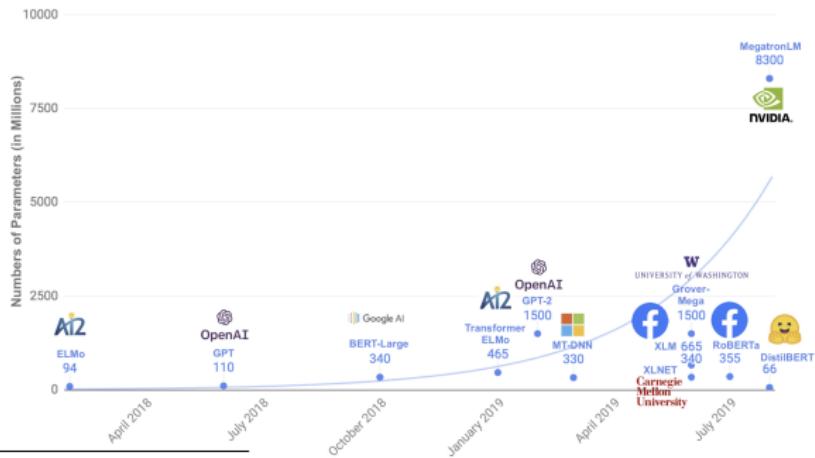
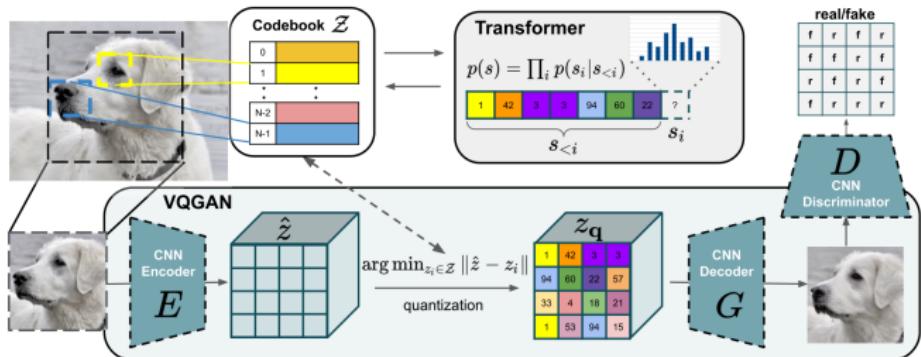


image credit: <http://jalammar.github.io/illustrated-gpt2>

Sanh V. et al. *DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter, 2019.*

Applications: Image generation, new era

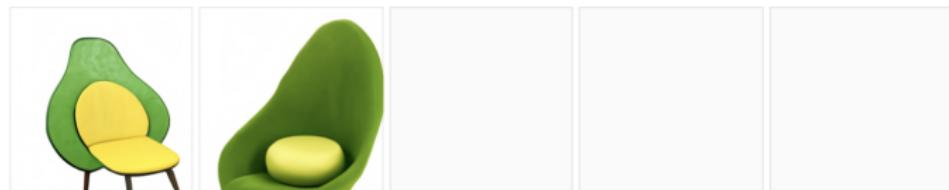


Esser P., Rombach R., Ommer B. *Taming Transformers for High-Resolution Image Synthesis*, 2020

Applications: Cross-modal image-text models

TEXT PROMPT an armchair in the shape of an avocado....

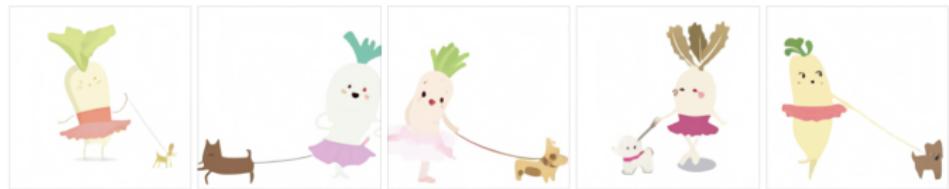
AI-GENERATED IMAGES



Edit prompt or view more images↓

TEXT PROMPT an illustration of a baby daikon radish in a tutu walking a dog

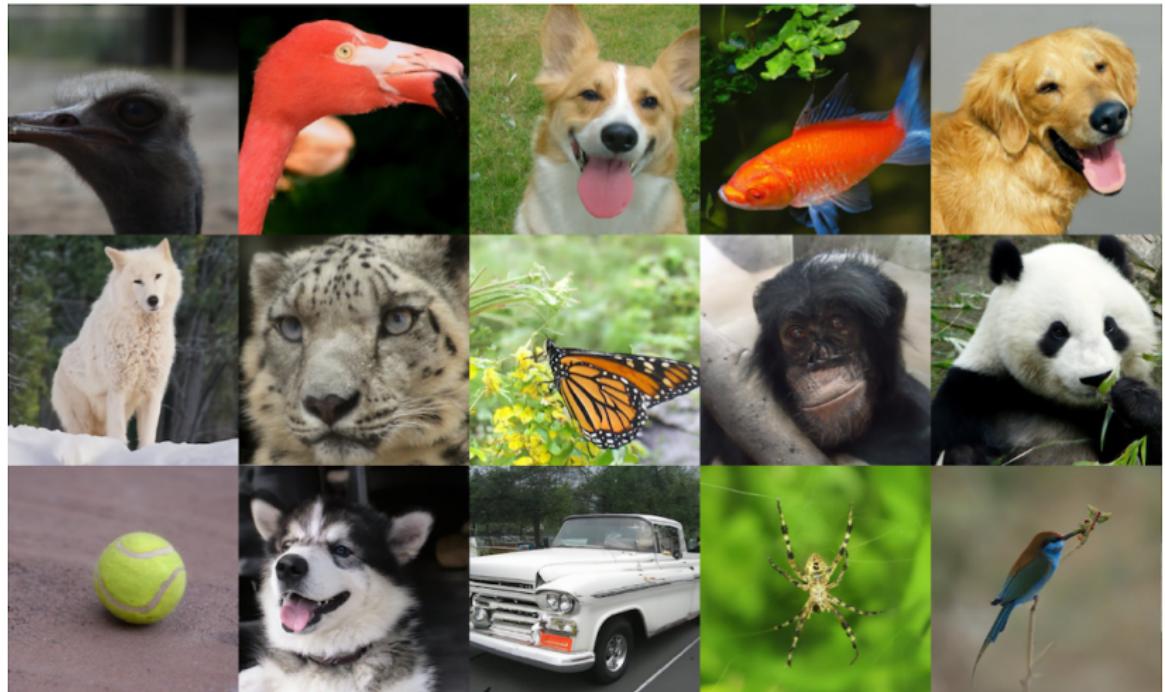
AI-GENERATED IMAGES



Edit prompt or view more images↓

image credit: <https://openai.com/blog/dall-e/>
Ramesh A. et al. Zero-shot text-to-image generation, 2021

Applications: Image generation



Problem Statement

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

Challenge

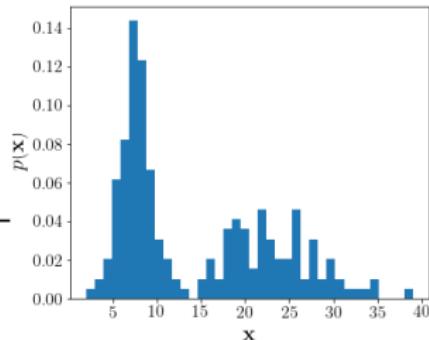
Data is complex and high-dimensional. E.g. the dataset of images lies in the space $X \subset \mathbb{R}^{\text{width} \times \text{height} \times \text{channels}}$.

Histogram as a generative model

Let $x \sim \text{Categorical}(\pi)$. 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 $x_i \in \{0, 1\}$.

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

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?

$$\pi(\mathbf{x}) = \pi(x_1) \cdot \pi(x_2) \cdot \dots \cdot \pi(x_m);$$

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

Tricks

Monte-Carlo estimation

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

Integral of expected value could be estimated using only samples from the distribution.

Law of the unconscious statistician (LOTUS)

Let X be a random variable and let $Y = g(X)$. Then

$$\mathbb{E}_{p_Y} Y = \mathbb{E}_{p_X} g(X) = \int g(\mathbf{x}) p(\mathbf{x}) d\mathbf{x}.$$

The name LOTUS came from a purported tendency to use the identity without realizing that it must be treated as the result of a rigorously proved theorem, not merely a definition.

Divergences

Fix probabilistic model $p(\mathbf{x}|\theta)$ – the set of parameterized distributions.

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

What is a divergence?

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

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

General divergence minimization task

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

where $\pi(\mathbf{x})$ is a true data distribution, $p(\mathbf{x}|\theta)$ 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 $f : \mathbb{R}_+ \rightarrow \mathbb{R}$ is a convex, lower semicontinuous function satisfying $f(1) = 0$.

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

Forward KL vs Reverse KL

Forward KL

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

Reverse KL

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

What is the difference between these two formulations?

Maximum likelihood estimation (MLE)

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

Forward KL vs Reverse KL

Forward KL

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

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

Reverse KL

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

Autoregressive model

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 $\mathbf{x} = (x_1, \dots, x_m)$, $\mathbf{x}_{1:i} = (x_1, \dots, x_i)$. Then

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

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

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

- ▶ 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 $p(x_i|\mathbf{x}_{1:i-1}, \boldsymbol{\theta})$ 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

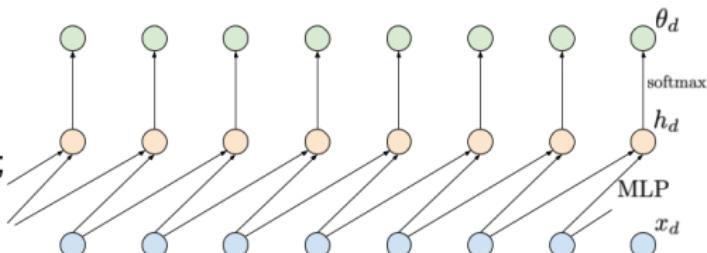
For large i the conditional distribution $p(x_i | \mathbf{x}_{1:i-1}, \theta)$ could be infeasible. Moreover, the history $\mathbf{x}_{1:i-1}$ has non-fixed length.

Markov assumption

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

Example

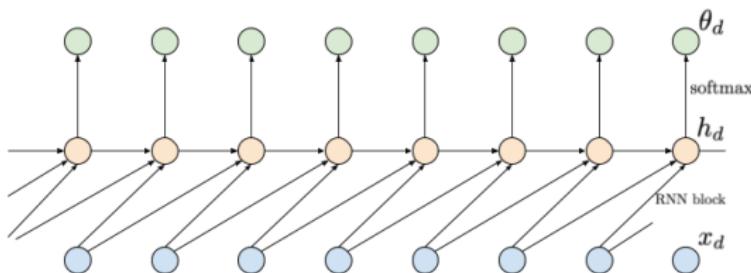
- ▶ $d = 2;$
- ▶ $x_i \in \{0, 255\};$
- ▶ $\mathbf{h}_i = \text{MLP}_\theta(x_{i-1}, x_{i-2});$
- ▶ $\pi_i = \text{softmax}(\mathbf{h}_i);$
- ▶ $p(x_i | x_{i-1}, x_{i-2}, \theta) = \text{Categorical}(\pi_i).$



Autoregressive models

- ▶ Previous model has **limited** memory d . It is insufficient for many modalities (e.g. for images and text).
- ▶ Recurrent NN fixes this problem and potentially could learn long-range dependencies:

$$p(x_i | \mathbf{x}_{1:i-1}, \theta) = p(x_i | \mathbf{h}_i, \theta), \quad \mathbf{h}_i = \text{RNN}(\mathbf{x}_{i-1}, \mathbf{h}_{i-1})$$



- ▶ Sequential computation of all conditionals $p(x_i | \mathbf{x}_{1:i-1}, \theta)$, hence, the training is slow.
- ▶ RNN suffers from vanishing and exploding gradients.

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 conditionals.
- ▶ Sampling from autoregressive models is trivial, but sequential
 - ▶ sample $x_1 \sim p(x_1)$;
 - ▶ sample $x_2 \sim p(x_2|x_1)$;
 - ▶
- ▶ Density estimation:

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

- ▶ Autoregressive models work on both continuous and discrete data.