Deep Generative Models

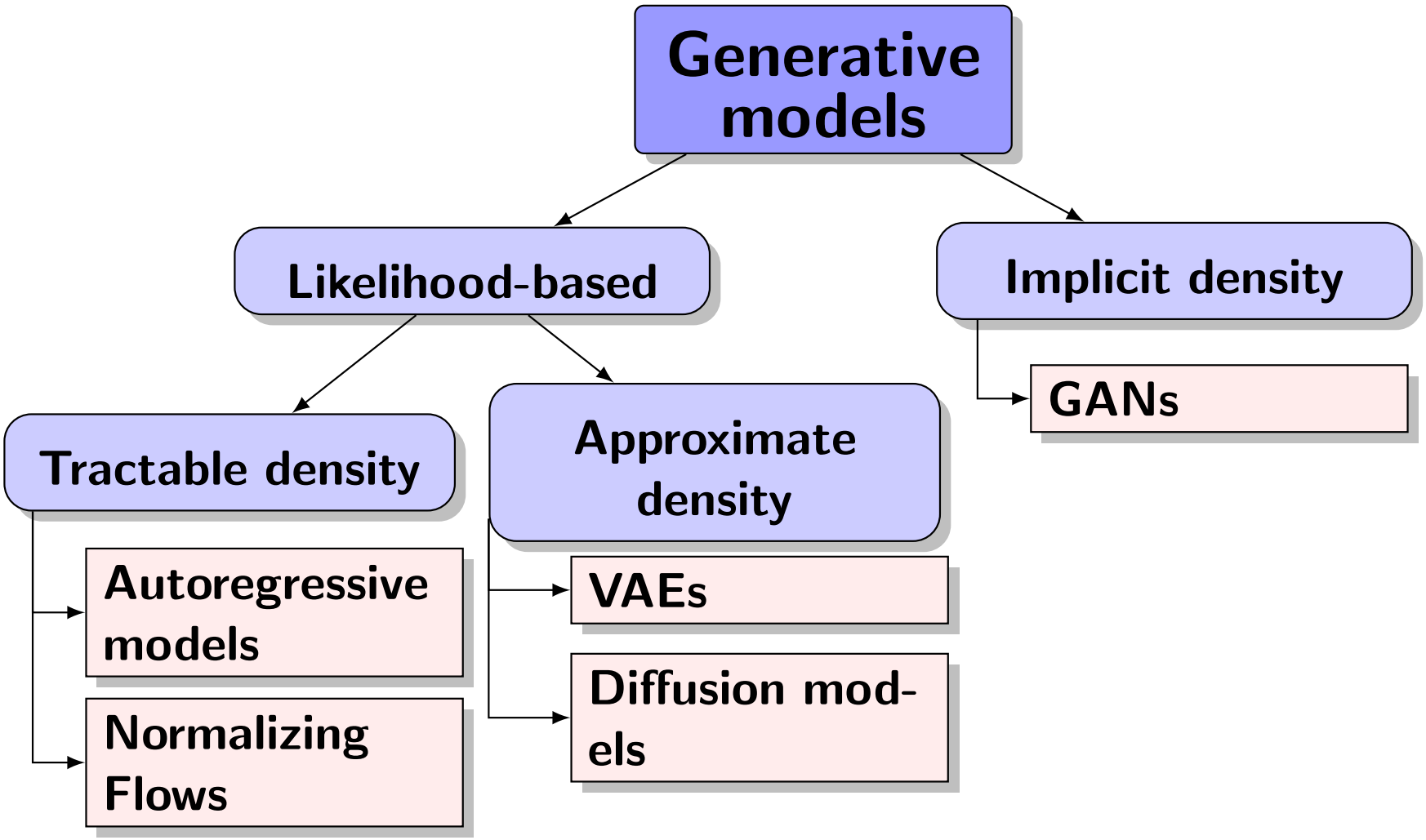
Lecture 1

Roman Isachenko

AI Masters

2025, Spring

A Zoo of Generative Models



Outline

1. **Overview of Generative Models**
2. **Course Tricks**
3. **Problem Statement**
4. **Divergence Minimization Framework**
5. **Autoregressive Modeling**

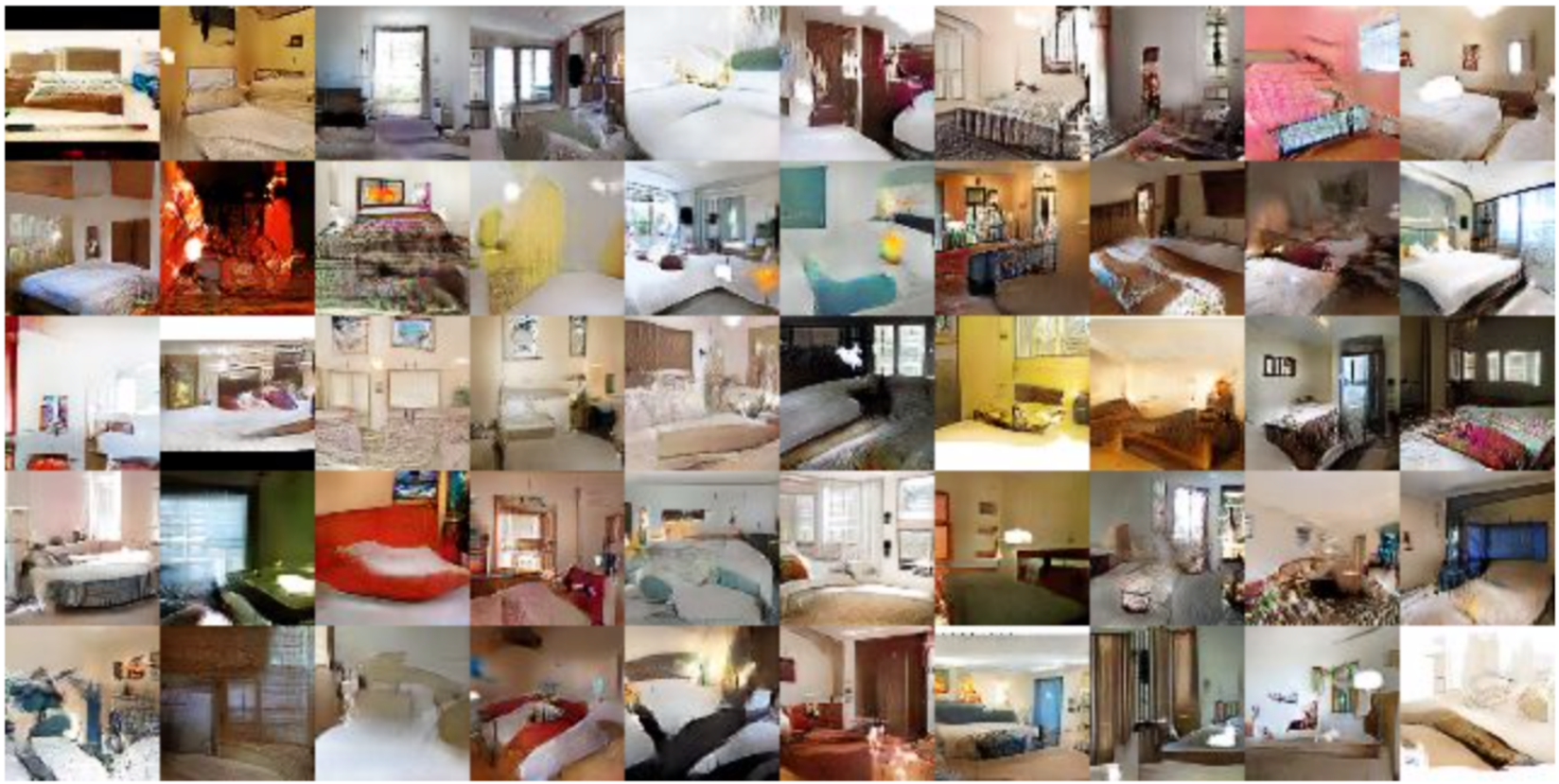
Outline

1. **Overview of Generative Models**
2. **Course Tricks**
3. **Problem Statement**
4. **Divergence Minimization Framework**
5. **Autoregressive Modeling**

VAE – The First Scalable Approach for Image Generation

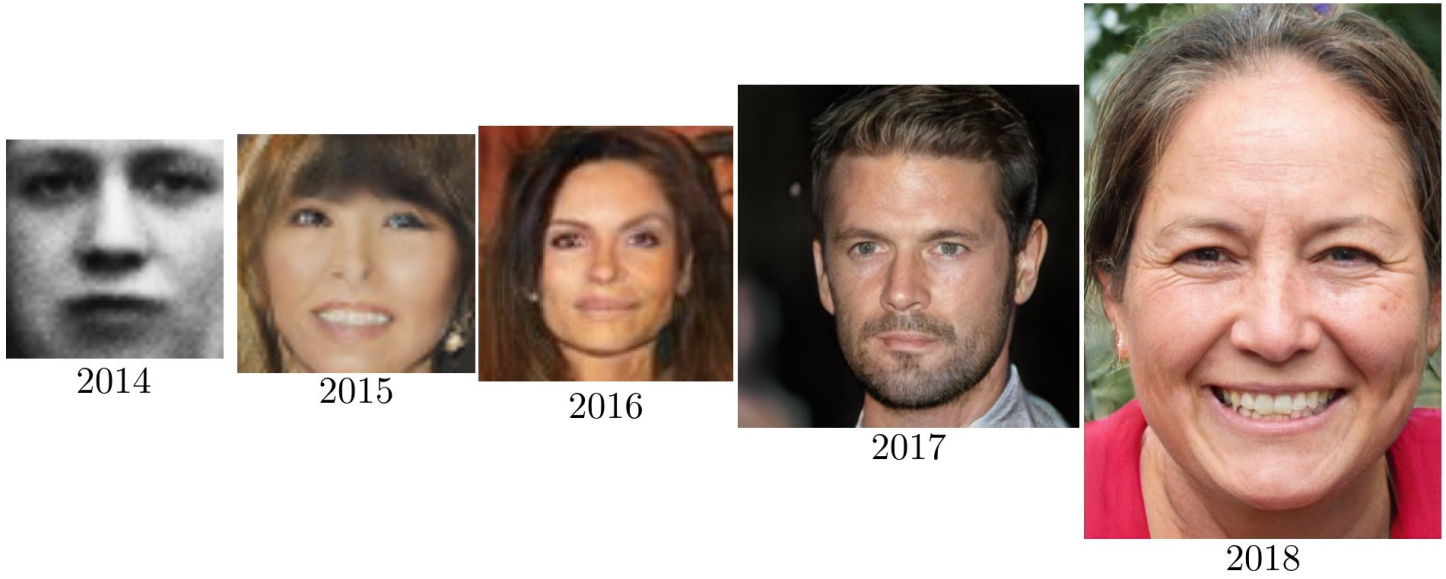
Kingma D. P., Welling M. Auto-encoding variational bayes, 2013

DCGAN – The 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 Facial Generation



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

Language Modeling at Scale

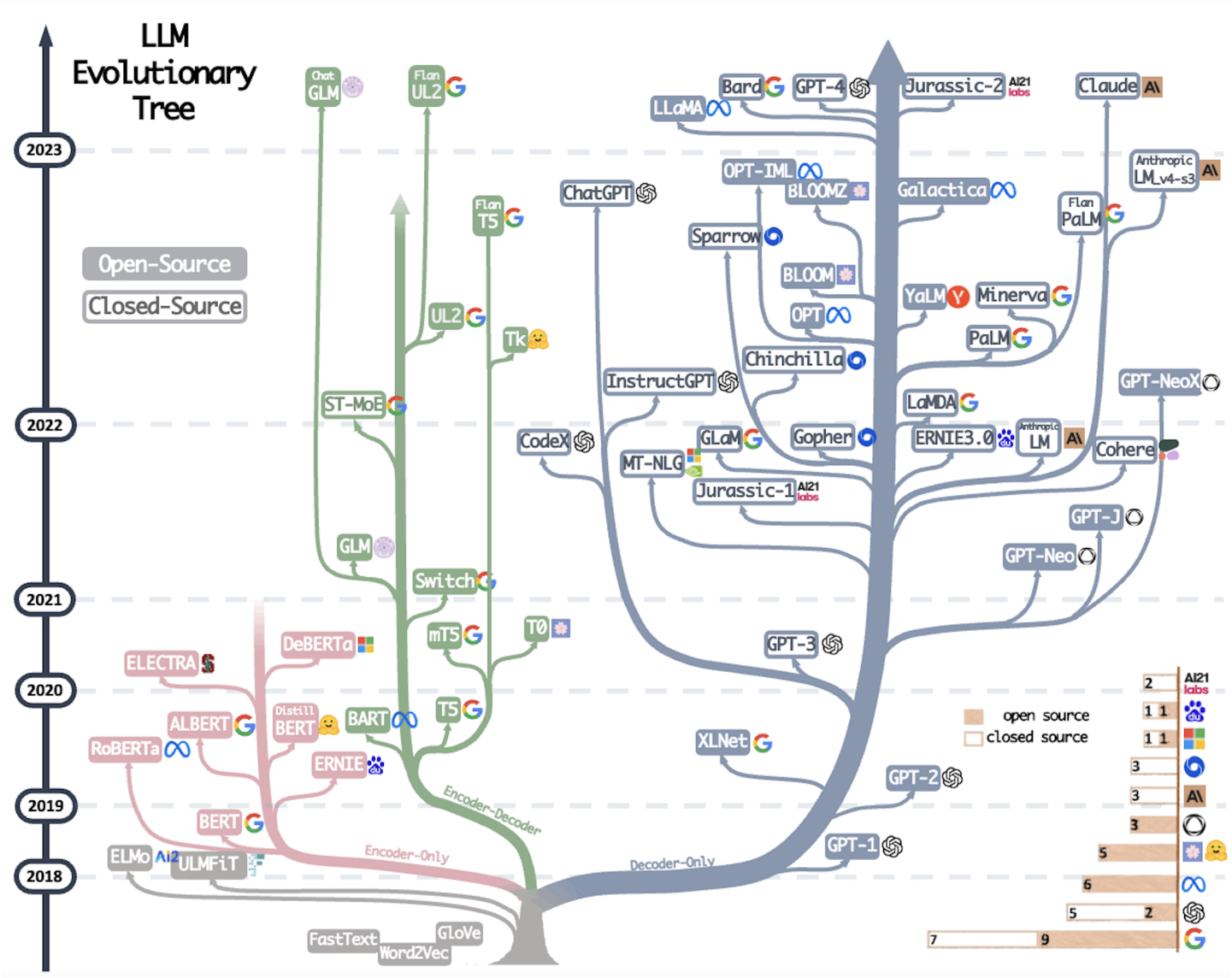


image credit:
https://blog.biocomm.ai/2023/05/14/open-source-proliferation-llm-evolutionary-tree/

Denoising Diffusion Probabilistic Models

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

Midjourney – Impressive Text-to-Image Results

image credit: https://www.midjourney.com/explore

Stable Diffusion 3 – Flow Matching

image credit: https://stability.ai/news/stable-diffusion-3

Sora – Video Generation

image credit: https://openai.com/index/sora

Outline

1. Overview of Generative Models
2. **Course Tricks**
3. Problem Statement
4. Divergence Minimization Framework
5. Autoregressive Modeling

Course Tricks (Part 1)

Log-Derivative Trick

Let $f : \mathbb{R}^m \to \mathbb{R}$ be a differentiable function

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

Jensen's Inequality

Let $\mathbf{x} \in \mathbb{R}^m$ be a continuous random variable with a density $p(\mathbf{x})$ and $f : \mathbb{R}^m \to \mathbb{R}$ be a convex function. Then

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

Monte Carlo Estimation

Let $\mathbf{x} \in \mathbb{R}^m$ be a continuous random variable with a density $p(\mathbf{x})$ and let $\mathbf{f} : \mathbb{R}^m \to \mathbb{R}^d$ be a vector function. Then

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

Course Tricks (Part 2)

Change-of-Variable Theorem (CoV)

Let $\mathbf{x}$ be a continuous random variable with a density $p(\mathbf{x})$ and $\mathbf{f} : \mathbb{R}^m \to \mathbb{R}^m$ be a differentiable, **invertible** function. If $\mathbf{y} = \mathbf{f}(\mathbf{x})$, then

$$p(\mathbf{y}) = p(\mathbf{x}) \left| \det \left(\frac{\partial \mathbf{x}}{\partial \mathbf{y}} \right) \right| = p(\mathbf{f}^{-1}(\mathbf{y})) \left| \det \left(\frac{\partial \mathbf{f}^{-1}(\mathbf{y})}{\partial \mathbf{y}} \right) \right| .$$

Proof (1D)

Assume f is a monotonically increasing function

$$F_Y(y) = P(Y \leq y) = P(x \leq f^{-1}(y)) = F_X(f^{-1}(y))$$

$$p(y) = \frac{dF_Y(y)}{dy} = \frac{dF_X(f^{-1}(y))}{dy} = \frac{dF_X(x)}{dx} \frac{df^{-1}(y)}{dy} = p(x) \frac{df^{-1}(y)}{dy}$$

Course Tricks (Part 3)

Law of the Unconscious Statistician (LOTUS)

Let $\mathbf{x} \in \mathbb{R}^m$ be a continuous random variable with a density $p(\mathbf{x})$ and let $\mathbf{f} : \mathbb{R}^m \to \mathbb{R}^m$ be a measurable function. If $\mathbf{y} = \mathbf{f}(\mathbf{x})$ then

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

Dirac Delta Function

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

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

$$\mathbb{E}_{p(\mathbf{x})}\mathbf{f}(\mathbf{x}) = \int \delta(\mathbf{x} - \mathbf{x}_0)\mathbf{f}(\mathbf{x})dx = \mathbf{f}(\mathbf{x}_0).$$

Outline

1. Overview of Generative Models

2. Course Tricks

3. **Problem Statement**

4. Divergence Minimization Framework

5. Autoregressive Modeling

Problem Statement

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

Goal

We want to learn the distribution $\pi(\mathbf{x})$ to:

- **evaluate $\pi(\mathbf{x})$ for new samples (i.e., how likely it is to obtain the object $\mathbf{x}$?) – density evaluation;**
- **generate new objects $\mathbf{x} \sim \pi(\mathbf{x})$ – generation.**

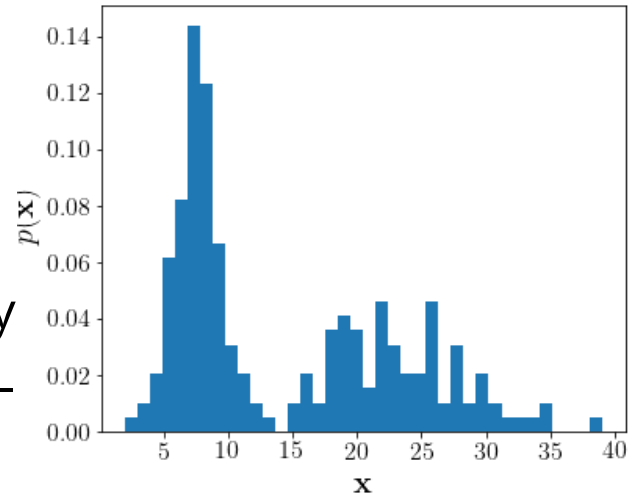
Challenge

Data is complex and high-dimensional. For example, the dataset of images exists in $\mathbb{R}^{\text{width}\times\text{height}\times\text{channels}}$. The curse of dimensionality prevents us from knowing the exact density $\pi(\mathbf{x})$.

Histogram as a Generative Model

Let $x \sim \text{Categorical}(\pi)$. **The histogram is fully determined by**

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



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

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

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

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

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

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

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

Conditional Model

Practical Scenarios

In practice, it is common to build a conditional model $\pi(\mathbf{x}|\mathbf{y})$.

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

Outline

1. Overview of Generative Models

2. Course Tricks

3. Problem Statement

4. **Divergence Minimization Framework**

5. Autoregressive Modeling

Divergences

- **Fix a probabilistic model $p(\mathbf{x}|\theta)$ – a family of parameterized distributions.**
- **Instead of searching for the true $\pi(\mathbf{x})$ over all distributions, we learn a function approximation $p(\mathbf{x}|\theta) \approx \pi(\mathbf{x})$.**

What is a Divergence?

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

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

Divergence Minimization Task

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

where $\pi(\mathbf{x})$ is the true data distribution and $p(\mathbf{x}|\theta)$ is the model distribution.

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

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

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 minimizing the Monte Carlo estimate of Forward KL.

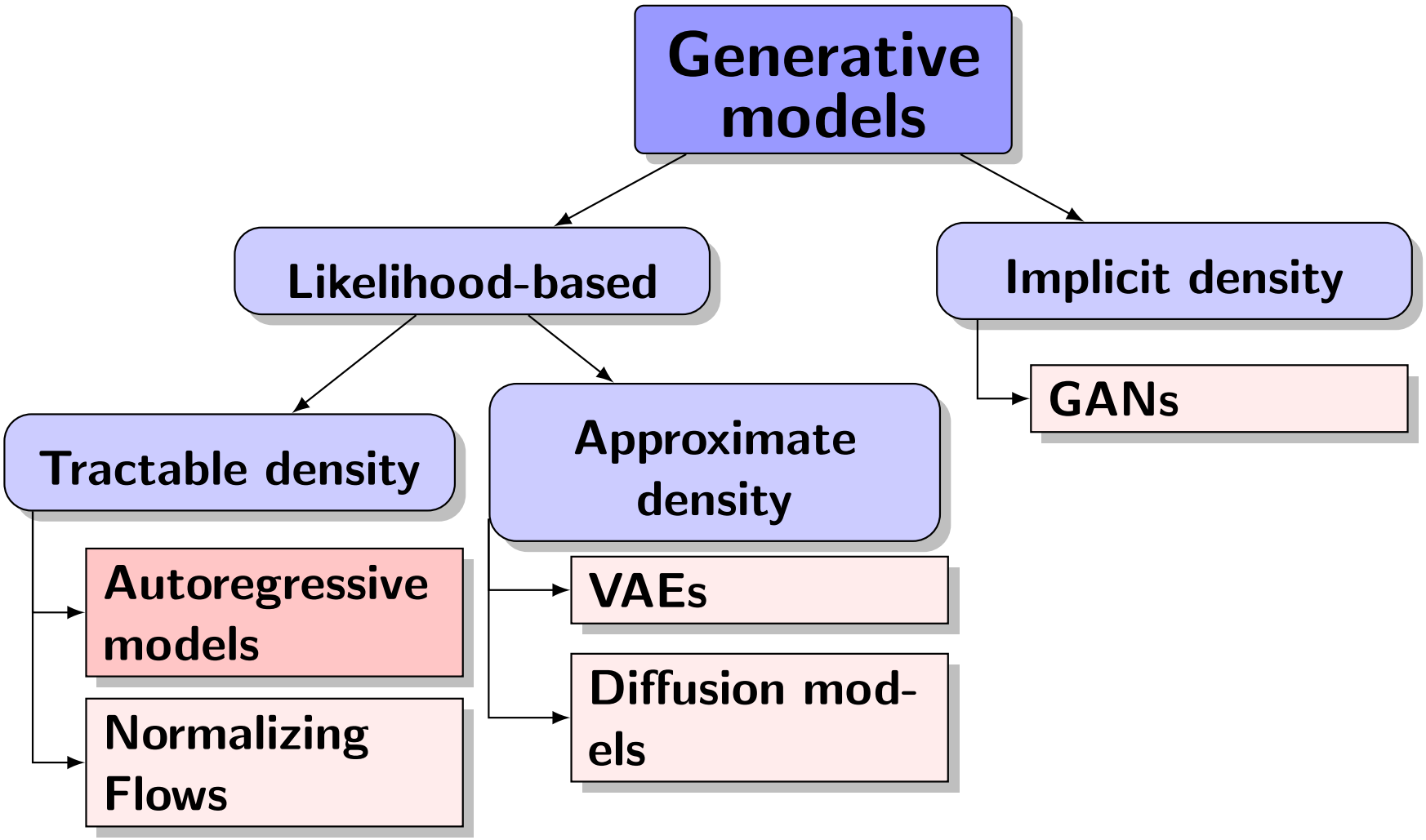
Reverse KL

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

Outline

1. Overview of Generative Models
2. Course Tricks
3. Problem Statement
4. Divergence Minimization Framework
5. **Autoregressive Modeling**

Generative Models Zoo (Revisited)



Autoregressive Modeling

MLE Problem

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

- **We aim to solve the problem using gradient-based optimization.**
- **We must be able to compute** $\log p(\mathbf{x}|\theta)$ **and its gradient** $\frac{\partial \log p(\mathbf{x}|\theta)}{\partial \theta}$ **efficiently.**

Likelihood as Product of Conditionals

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

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

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

Autoregressive Models

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

- **Sampling is sequential:**
  - **sample** $\hat{x}_1 \sim p(x_1|\theta)$;
  - **sample** $\hat{x}_2 \sim p(x_2|\hat{x}_1, \theta)$;
  - $\ldots$
  - **sample** $\hat{x}_m \sim p(x_m|\hat{\mathbf{x}}_{1:m-1}, \theta)$;
  - **the newly generated object is** $\hat{\mathbf{x}} = (\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_m)$.
- **Each conditional** $p(x_j|\mathbf{x}_{1:j-1}, \theta)$ **can be modeled by a neural network.**
- **Modeling all conditionals separately is infeasible. Shared parameters** θ **help avoid this.**

Autoregressive Models: MLP

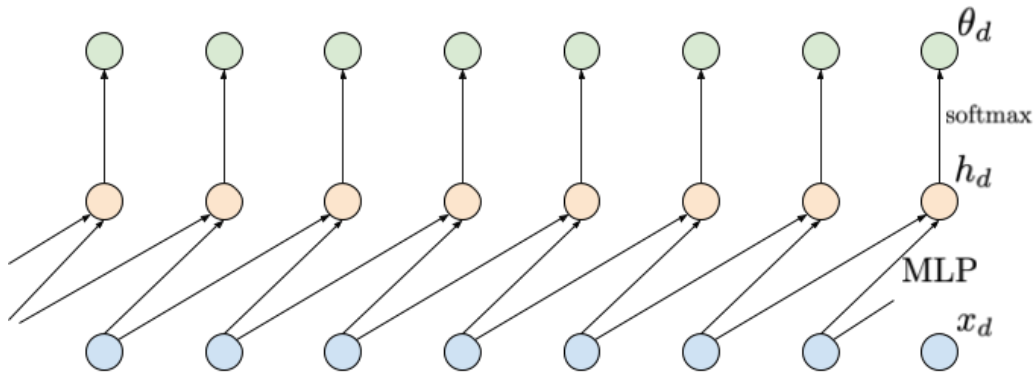
For large j, the conditional $p(x_j|\mathbf{x}_{1:j-1}, \theta)$ may be expensive. Moreover, the history $\mathbf{x}_{1:j-1}$ has variable length.

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

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



We can also model continuous distributions instead of discrete ones.

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

Autoregressive Models: LLM

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

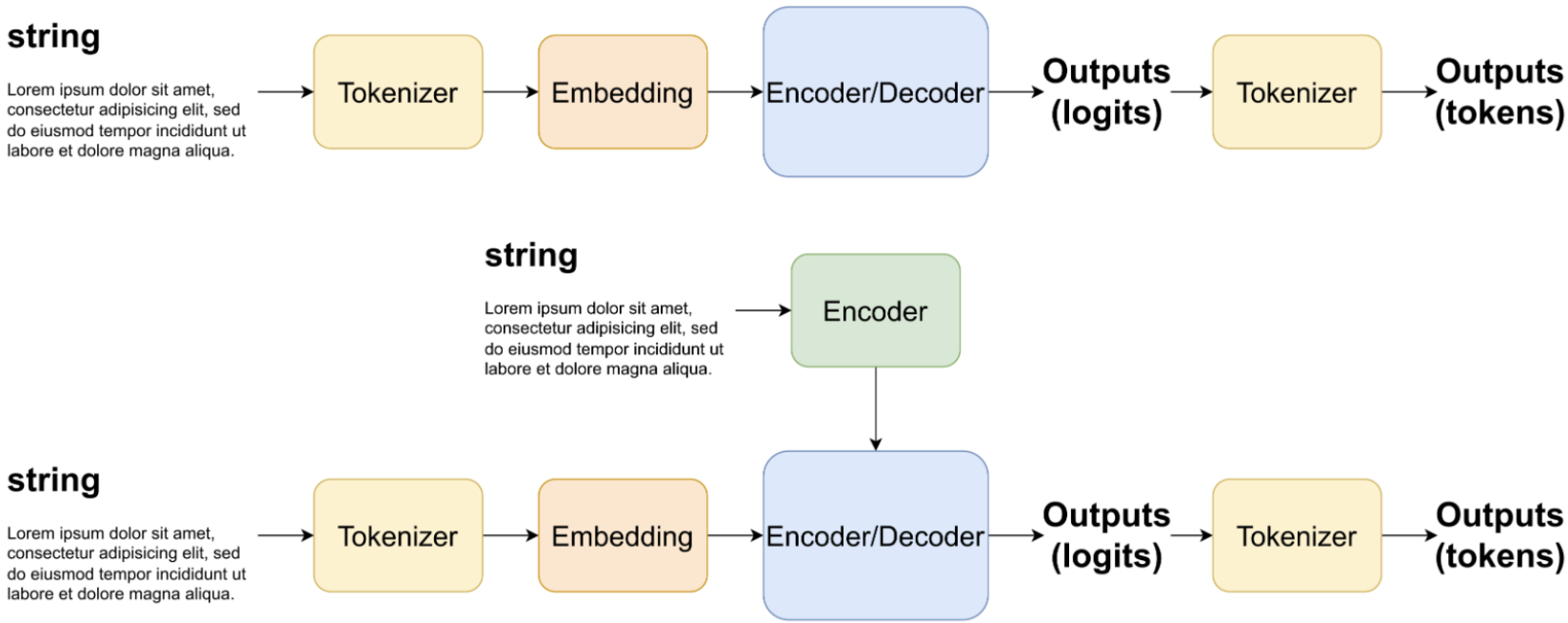


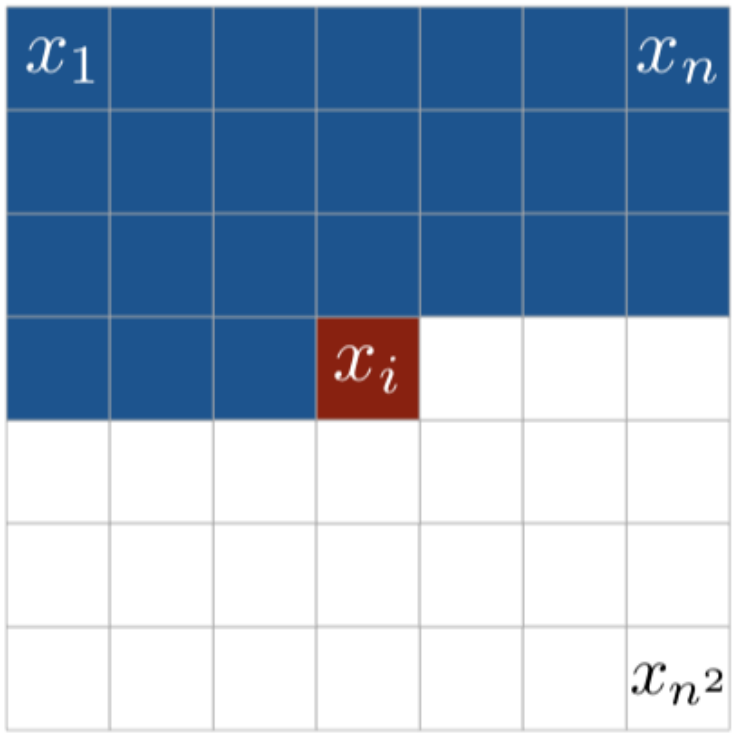
image credit: https://jmtomczak.github.io/blog/20/20_llms.html

Autoregressive Models for Images

How do we model the distribution $\pi(\mathbf{x})$ of natural images?

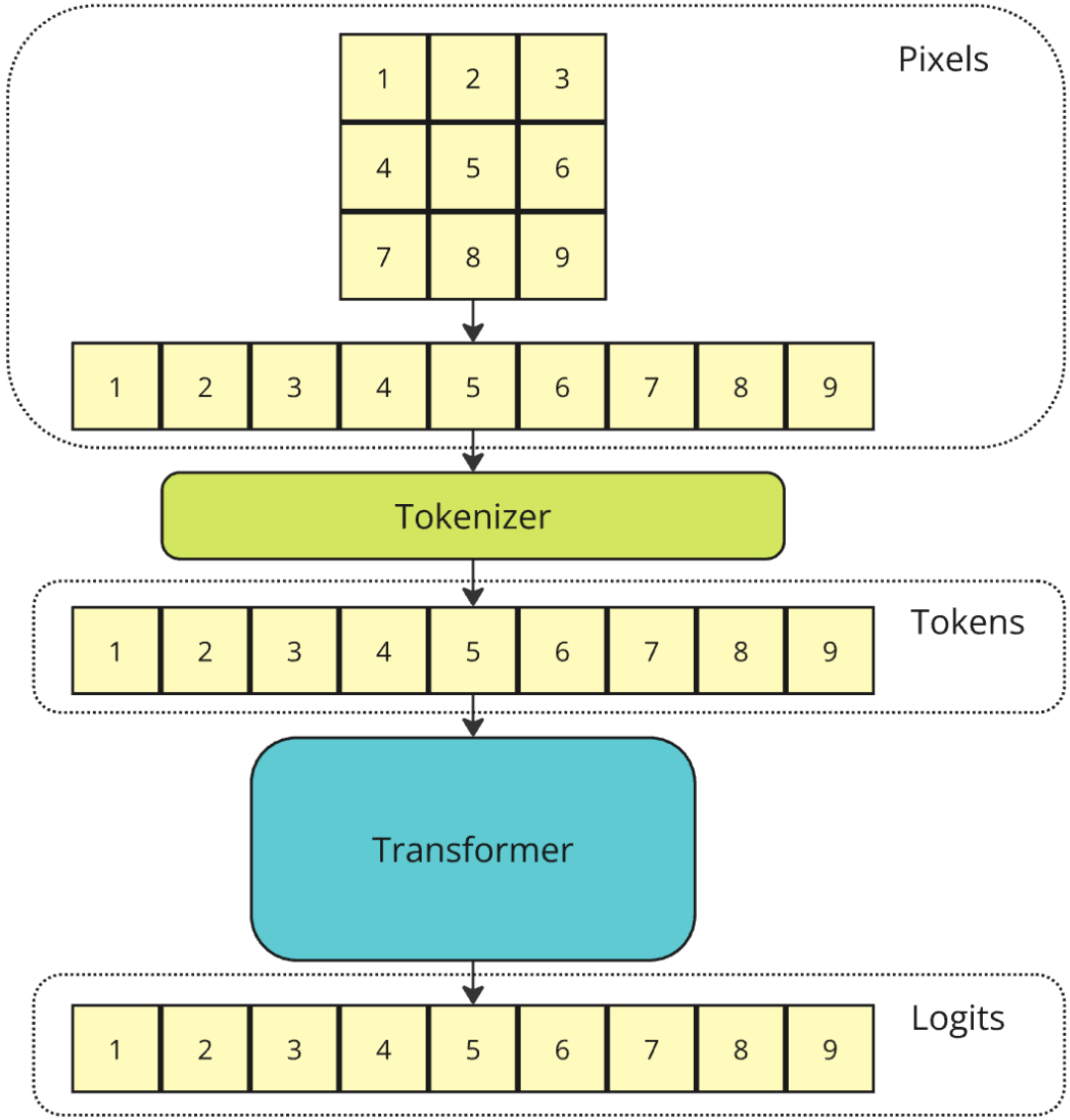
$$p(\mathbf{x}|\theta) = \prod_{j=1}^{\text{width}\times\text{height}} p(x_j|\mathbf{x}_{1:j-1}, \theta).$$

- **We must specify an order of the image pixels. Raster ordering is a common approach.**
- **We can also incorporate RGB channel dependencies.**



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

Autoregressive Models: ImageGPT



Chen M. et al. Generative Pretraining from Pixels, 2020

Summary

- We aim to approximate the data distribution for both density estimation and sample generation.
- Fitting a model distribution to the true data distribution can be posed as divergence minimization.
- Minimizing the forward KL is equivalent to maximizing the likelihood (MLE).
- Autoregressive models decompose the distribution into sequential conditionals.
- Sampling from autoregressive models is straightforward, but it must proceed sequentially.
- To compute the density, we multiply all conditionals $p(x_j|\mathbf{x}_{1:j-1}, \theta)$.
- ImageGPT flattens the image (raster ordering) so that the Transformer model can process it.