Deep Generative Models

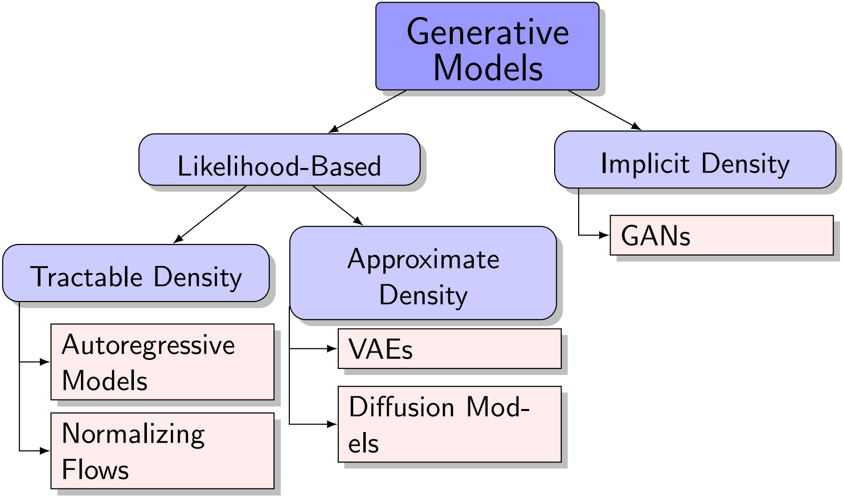
Lecture 1

Roman Isachenko

Moscow Institute of Physics and Technology
Yandex School of Data Analysis

2025, Autumn

Generative Models Zoo



Outline

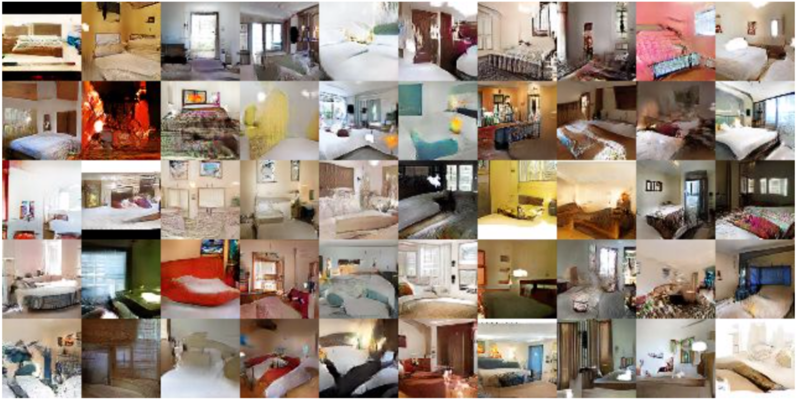
Outline

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

VAE – The First Scalable Approach for Image Generation

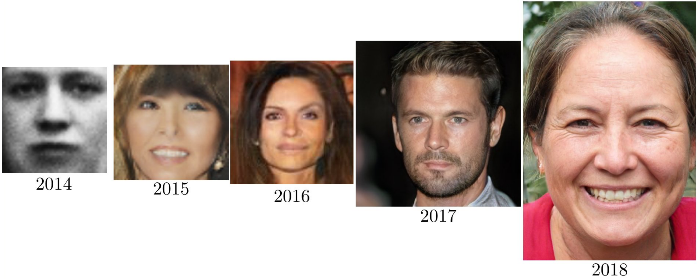
Kingma D.P., Welling M. Auto-Encoding Variational Bayes, 2013

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 Face Generation



Karras T., Laine S., Aila T. A Style-Based Generator Architecture for Generative Adversarial Networks, 2018

Language Modeling at Scale

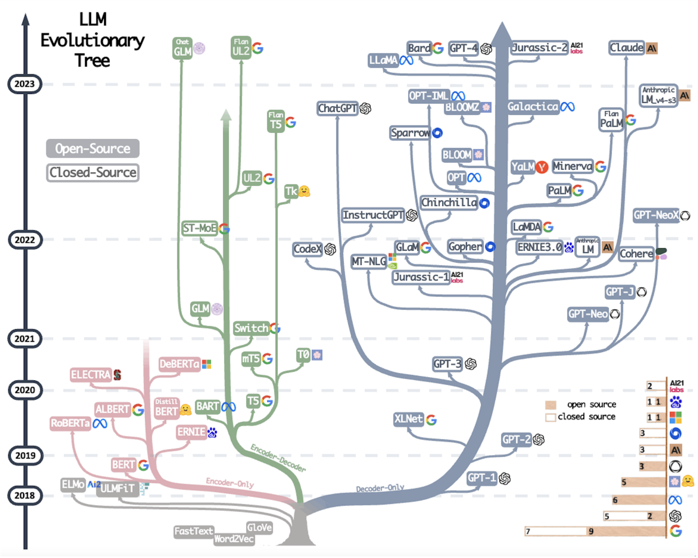


Image credit: https://blog.biocomm.ai/2023/05/14/open-source-proliferation-llm-evolutionary-tree/

Denoising Diffusion Probabilistic Model

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

Course Tricks I

Log-Derivative Trick

Given a differentiable function $f : \mathbb{R}^m \to \mathbb{R}$,

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

Jensen's Inequality

If $\mathbf{x} \in \mathbb{R}^m$ is a continuous random variable with density $p(\mathbf{x})$ and $f : \mathbb{R}^m \to \mathbb{R}$ is convex, 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 density $p(\mathbf{x})$, and $\mathbf{f} : \mathbb{R}^m \to \mathbb{R}^d$ be any vector-valued 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 II

Change of Variables Theorem (CoV)

Suppose $\mathbf{x}$ is a continuous random variable with density $p(\mathbf{x})$, and $\mathbf{f} : \mathbb{R}^m \to \mathbb{R}^m$ is differentiable and **invertible**. 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 monotonically increasing.

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

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

Course Tricks III

Law of the Unconscious Statistician (LOTUS)

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

Any deterministic variable $\mathbf{x}_0$ can be interpreted as a random variable with density $p(\mathbf{x}) = \delta(\mathbf{x} - \mathbf{x}_0)$.

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

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

Outline

Problem Statement

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

Objective

Our aim is to learn a distribution $\pi(\mathbf{x})$ that allows us to:

- Evaluate $\pi(\mathbf{x})$ on novel data (answering "How likely is an object $\mathbf{x}$?") — **density estimation**;
- Generate new samples from $\pi(\mathbf{x})$ (sample $\mathbf{x} \sim \pi(\mathbf{x})$) — **generation**.

Challenge

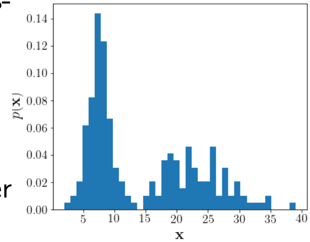
The data is high-dimensional and complex. For example, image datasets live in $\mathbb{R}^{\text{width}\times\text{height}\times\text{channels}}$. The curse of dimensionality makes accurately estimating $\pi(\mathbf{x})$ infeasible.

Histogram as a Generative Model

Assume $x \sim \mathrm{Categorical}(\boldsymbol{\pi})$. The histogram model is fully characterized by

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

Curse of dimensionality: The number of bins rises exponentially.



MNIST example: 28×28 grayscale images, with each image $\mathbf{x} = (x_1, \ldots, x_{784})$, $x_i \in \{0, 1\}$:

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

A complete histogram would require $2^{28\times28} - 1$ parameters for $\pi(\mathbf{x})$.

Question: How many parameters are required in these cases?

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

Problem Statement: Conditional Models

Conditional Models

In practice, we're typically interested in learning conditional models $\pi(\mathbf{x}|\mathbf{y})$.

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

Outline

Divergences

- Let us fix a probabilistic model $p(\mathbf{x}|\boldsymbol{\theta})$—a parametric family of distributions.
- Instead of searching among all possible distributions for the true $\pi(\mathbf{x})$, we seek a functional approximation $p(\mathbf{x}|\boldsymbol{\theta}) \approx \pi(\mathbf{x})$.

What is a Divergence?

Let $\mathcal{P}$ be the set of all probability distributions. A mapping $D : \mathcal{P} \times \mathcal{P} \to \mathbb{R}$ is called 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 Problem

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

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

Forward KL vs Reverse KL (Kullback-Leibler Divergence)

Forward KL

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

Reverse KL

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

What's the practical distinction between these two objectives?

Maximum Likelihood Estimation (MLE)

Let $\{\mathbf{x}_i\}_{i=1}^n$ be i.i.d. observed 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: MLE as Forward KL

Forward KL

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

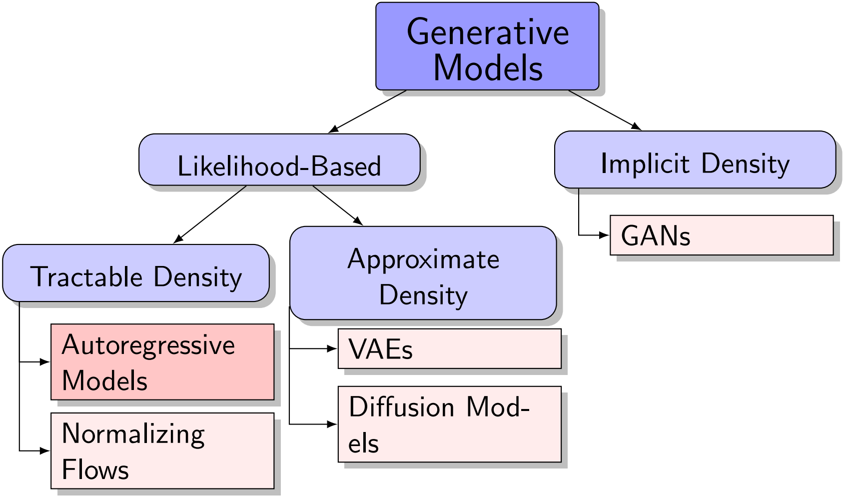
Maximum likelihood estimation is thus equivalent to minimizing a Monte Carlo estimate of the forward KL divergence.

Reverse KL

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

Outline

Generative Models Zoo



Autoregressive Modeling

MLE Problem

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

- This maximization is typically solved via gradient-based optimization.
- Thus, efficient computation of both $\log p(\mathbf{x}|\boldsymbol{\theta})$ and its gradient $\frac{\partial \log p(\mathbf{x}|\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}$ is crucial.

Likelihood as a Product of Conditionals

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

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

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

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 performed sequentially:
  - Sample $\hat{x}_1 \sim p(x_1|\boldsymbol{\theta})$;
  - Sample $\hat{x}_2 \sim p(x_2|\hat{x}_1, \boldsymbol{\theta})$;
  - ...
  - Sample $\hat{x}_m \sim p(x_m|\hat{\mathbf{x}}_{1:m-1}, \boldsymbol{\theta})$;
  - The generated sample is $\hat{\mathbf{x}} = (\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_m)$.
- Each conditional $p(x_j|\mathbf{x}_{1:j-1}, \boldsymbol{\theta})$ can be modeled using a neural network.
- Modeling all conditionals separately isn't feasible. To address this, we share parameters across all conditionals.

Autoregressive Models: MLP

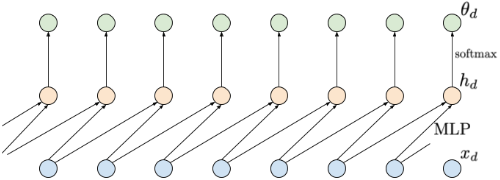
For large j, the conditional $p(x_j|\mathbf{x}_{1:j-1}, \boldsymbol{\theta})$ becomes intractable as the history $\mathbf{x}_{1:j-1}$ grows variable-length.

Markov Assumption

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

Example

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



Can we also model continuous-valued data, not just the discrete case?

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

Autoregressive Models: LLM

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

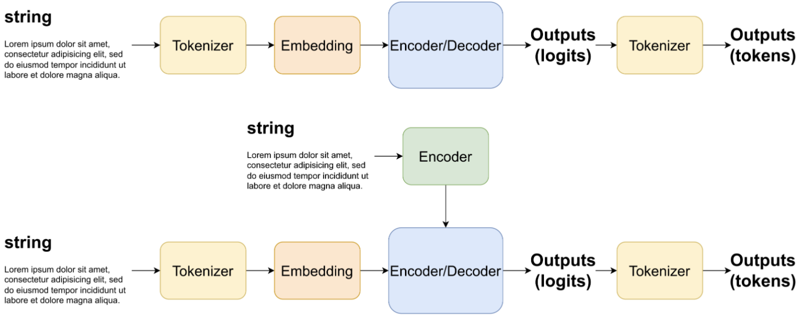


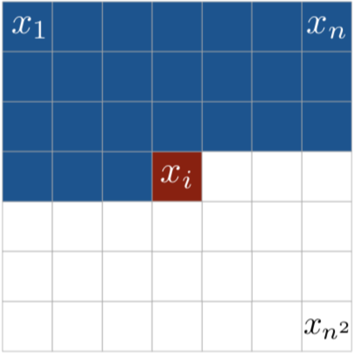
Image credit: https://jmtomczak.github.io/blog/20/20_llms.html

Autoregressive Models for Images

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

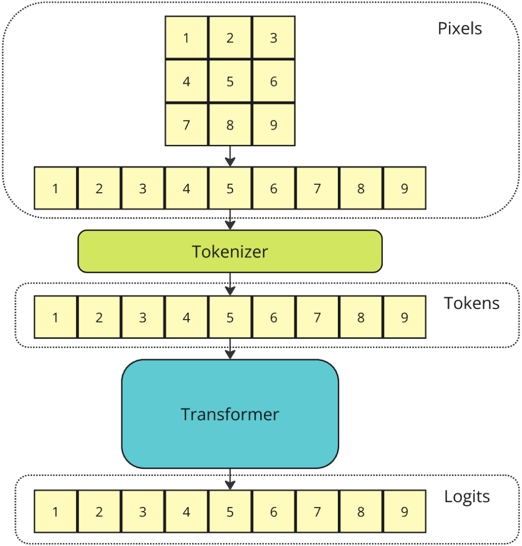
$$p(\mathbf{x}|\boldsymbol{\theta}) = \prod_{j=1}^{\text{width}\times\text{height}} p(x_j|\mathbf{x}_{1:j-1}, \boldsymbol{\theta})$$

- A pixel ordering must be selected; the raster scan is a standard choice.
- RGB channel dependencies can be modeled explicitly as well.



Oord A., Kalchbrenner N., Kavukcuoglu K. Pixel Recurrent Neural Networks, 2016

Autoregressive Models: ImageGPT



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

Summary

- Our target is to approximate the data distribution both for density estimation and for generation.
- The divergence minimization framework offers a principled way to learn distributions that match the data.
- Minimizing the forward KL divergence is equivalent to maximum likelihood estimation.
- Autoregressive models decompose the joint distribution as a product of conditionals.
- Autoregressive sampling is simple, but inherently sequential.
- Joint density evaluation multiplies all conditional probabilities $p(x_j|\mathbf{x}_{1:j-1}, \boldsymbol{\theta})$.
- ImageGPT applies a transformer architecture to sequences of raster-ordered image pixels.