Diffusion Models for Everyone

Kunal Kanawade

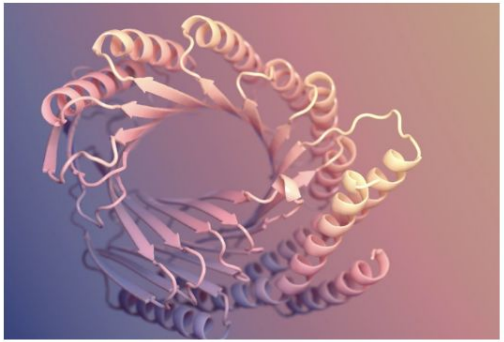
Diffusion based models in use

Stable Diffusion

DALEE

OpenAI Sora

Meta MovieGen



AlphaFold3

RFDiffusion

Diffusion based LLM

https://www.inceptionlabs.ai/blog/mercury-refreshed

Diffusion based LLM

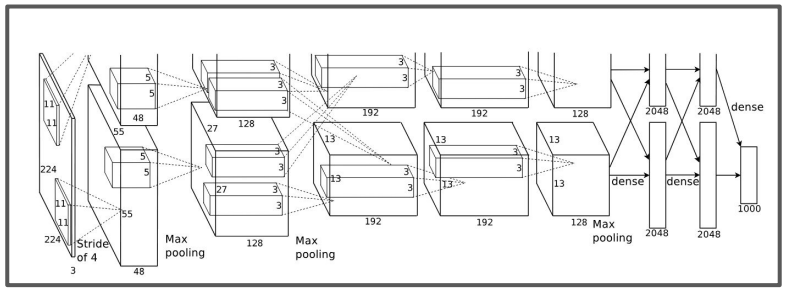
Rather than predicting tokens one at a time, dLLMs generate entire blocks of language in parallel.

The output starts as a broad, noisy sketch, then through an iterative “denoising” process rapidly converges on the best answer.

Tokens (and letters, words, and code) emerge all at once, not one at a time.

https://x.com/multimodalart/status/1894842951353671750

Classification



y

"Elephant"

Classification is possible for natural images because they have **clear patterns** and meanings that machines can **learn and recognize.**

Can we go in the reverse direction ? Yes indeed

Generation

Generation is sampling from a data distribution

$$z \sim p_{data} \Longrightarrow z =$$

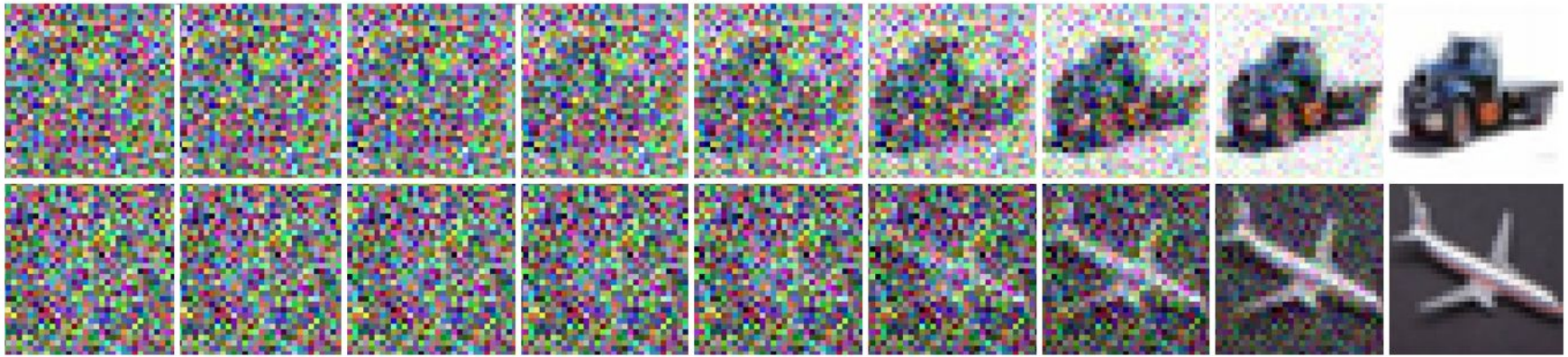
What if we want to generate an image of “Dog” or “Cat” or “Sunset”

Condition data distribution

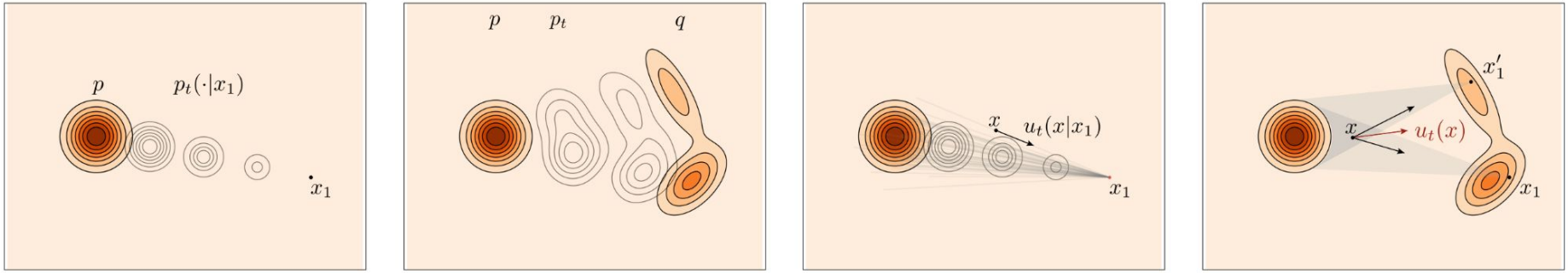
$$z \sim p_{data}(\cdot|y)$$

Conditional variable **y**
y = ’elephant’

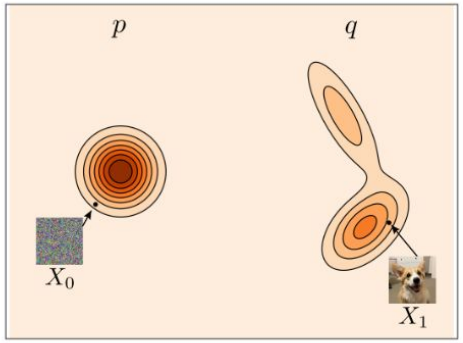
What diffusion models really do



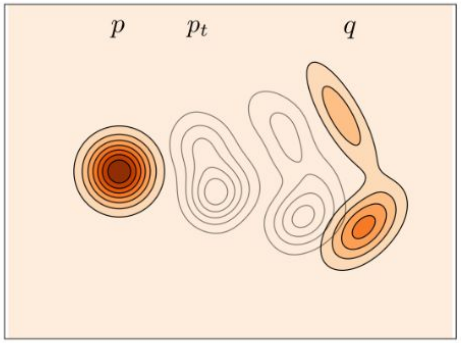
Gradual process of denoising the image in such a way that target image is generated



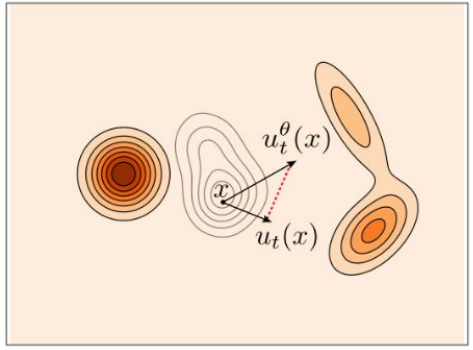
Paths for initial distribution to target distribution



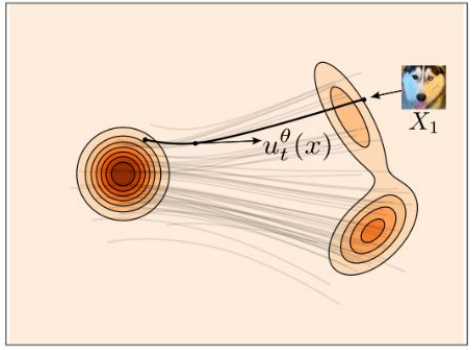
(a) Data.



(b) Path design.



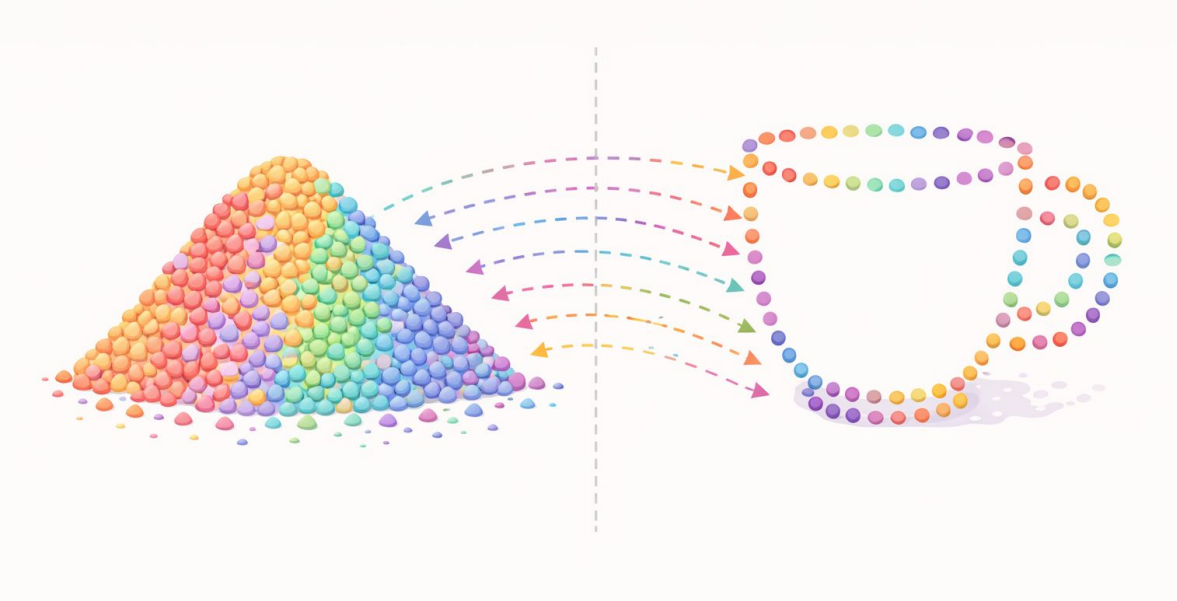
(c) Training.



(d) Sampling.

Monge - Kantorovich Problem

Defined by Monge and later by Kantorovich, the mass transportation problem aims at finding a transport plan from a source distribution to a target distribution so that it minimizes a global cost.



Monge - Kantorovich Problem

1.1 Monge Problem for Discrete Points

Matching Problem Given a cost matrix $(\mathrm{C}_{i,j})_{i\in[\![n]\!],j\in[\![m]\!]}$ and assuming $n = m$, the optimal assignment problem aims to find a bijection σ within the set $\mathrm{Perm}(n)$ of permutations of n elements that solves

$$\min_{\sigma\in\mathrm{Perm}(n)} \frac{1}{n}\sum_{i=1}^{n} \mathrm{C}_{i,\sigma(i)}. \tag{1}$$

One could naively evaluate the cost function above using all permutations in the set $\mathrm{Perm}(n)$. However, this set has size $n!$, which becomes enormous even for small values of n. In general, the optimal σ is not unique.

Monge- Kantorovich Problem - “best” way ?

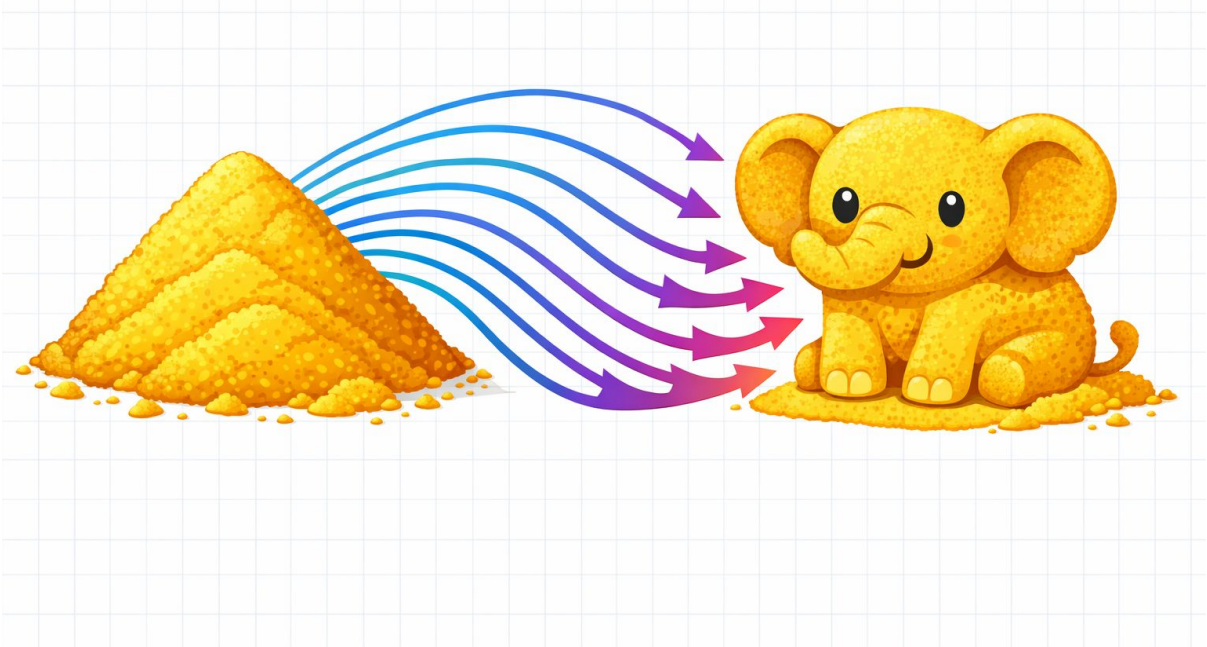
$$X_t = tX_1 + (1-t)X_0$$

OT interpolation equation gives us the optimal transport map from X0 to X1 that reduces the cost of transportation

Completely parallelizable

Training involves sampling from t = [0,1) and

Monge- Kantorovich Problem - “best” way ?



Why such training is allowed

Many probabilistic models are defined only up to a normalization constant.

To turn this into a proper probability distribution, you need to divide by the normalization constant (also called the partition function).

$$p_\theta(x) = \frac{1}{Z(\theta)}\, q(x), \qquad Z(\theta) = \int q(x)\, dx$$

$$\log p_\theta(x) = \log q_\theta(x) - \log Z(\theta)$$

Score Function

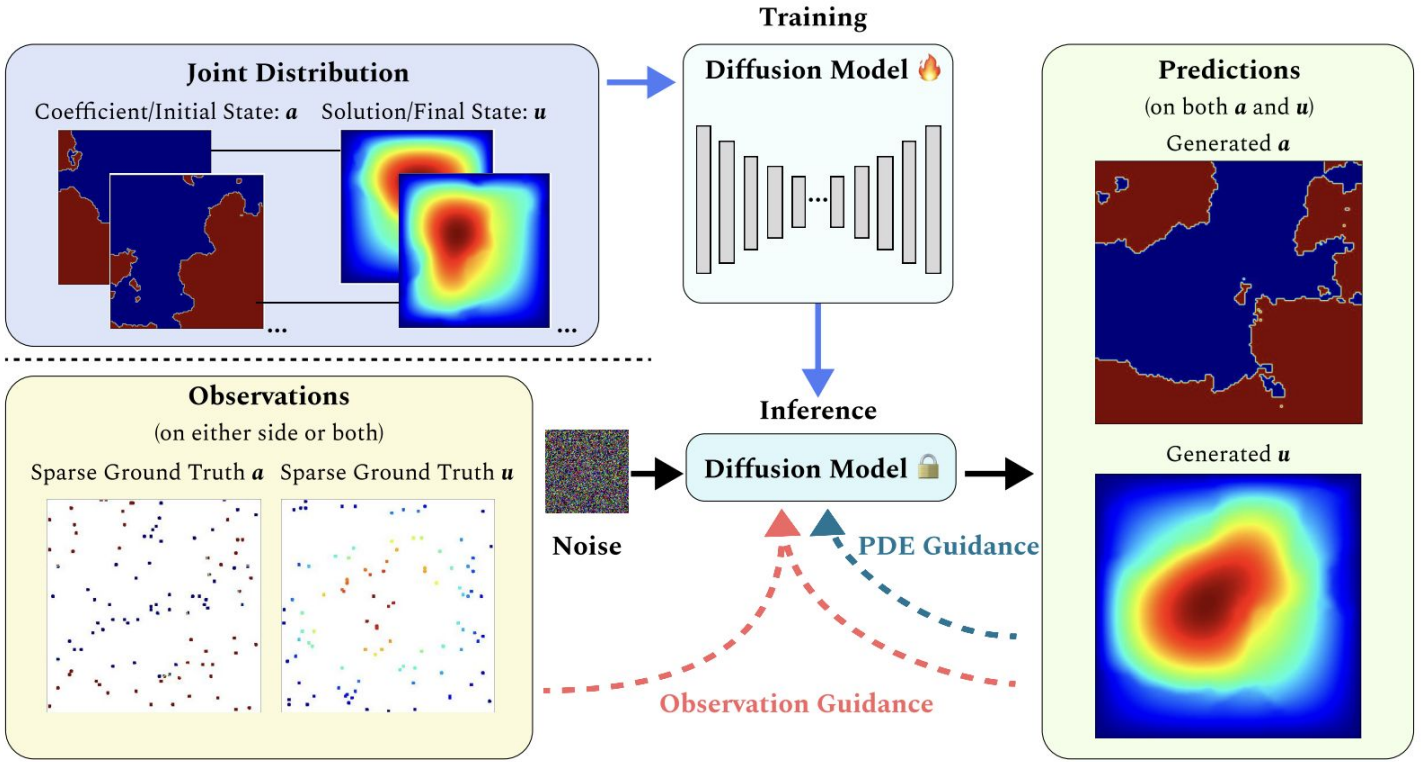
$$\log p_\theta(x) = \log q_\theta(x) - \log Z(\theta)$$

$$\nabla_x \log p_\theta(x) = \nabla_x \log q(x) \quad \text{since} \quad \nabla_x \log Z(\theta) = 0$$

This gradient is called the **Score function**.

So instead of matching actual probabilities, Hyvärinen 2005 proposed to match the scores the gradients of log-probability with respect to x.

Inverse Problem



Thank You !!