Diffusion Model

Lei Li

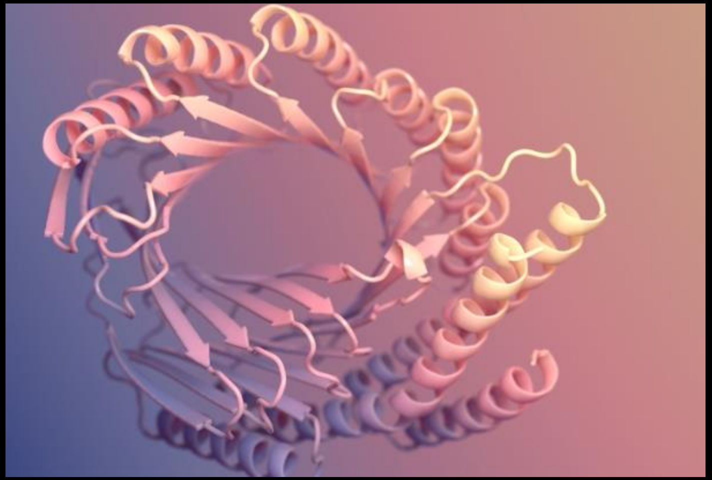
Carnegie Mellon University
Language Technologies Institute

Image and Video Generation

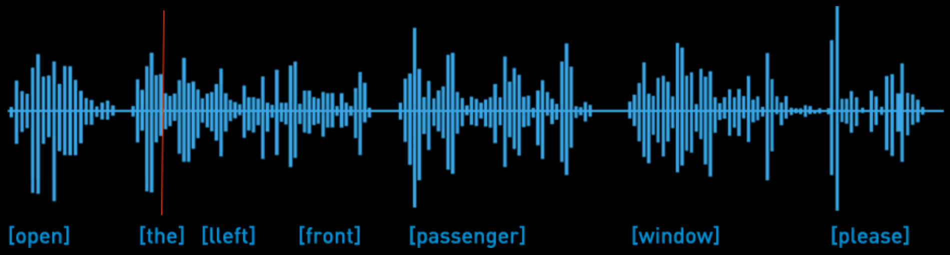
generate an image of a colorful deer riding on the walking into the sky at cmu campus

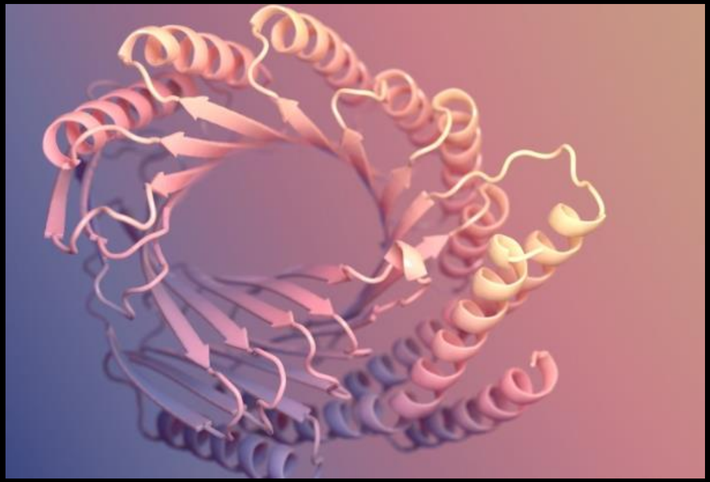
Diffusion Models are state-of-the-art models for (Continuous) Data Generation

- Time series data
- Audio/speech
- 3D Objects (coordinates)
  - Molecule structures

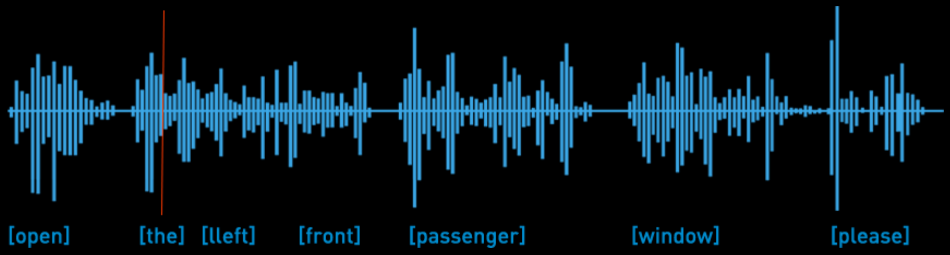


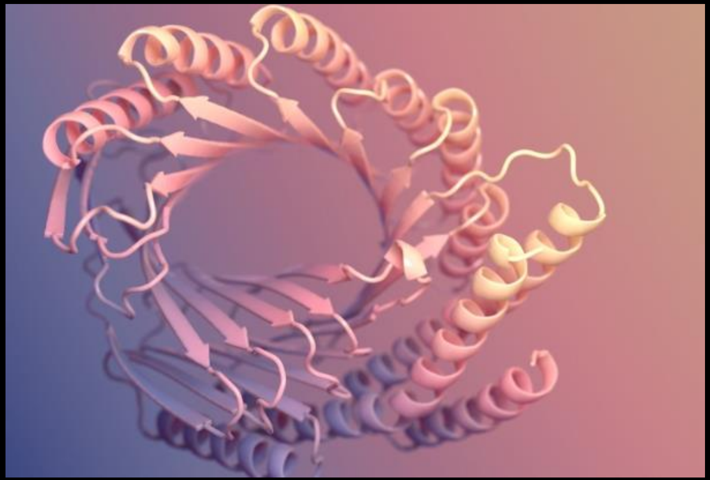
Data Representation





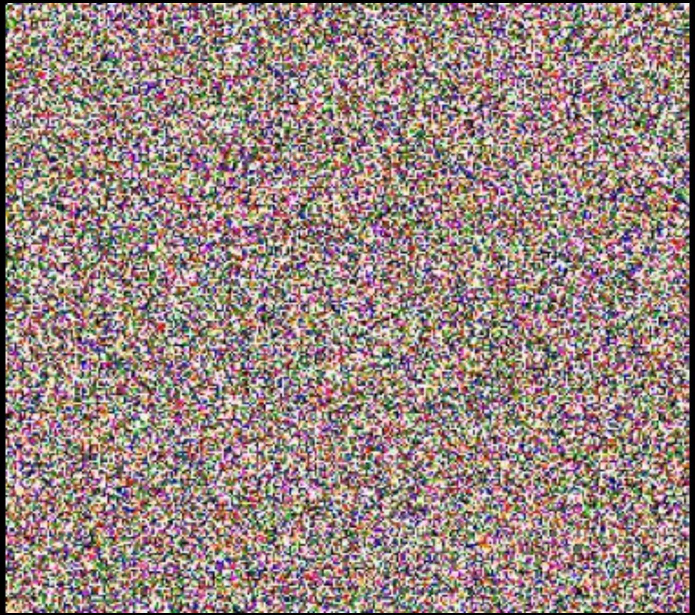
Probabilistic Model for Data



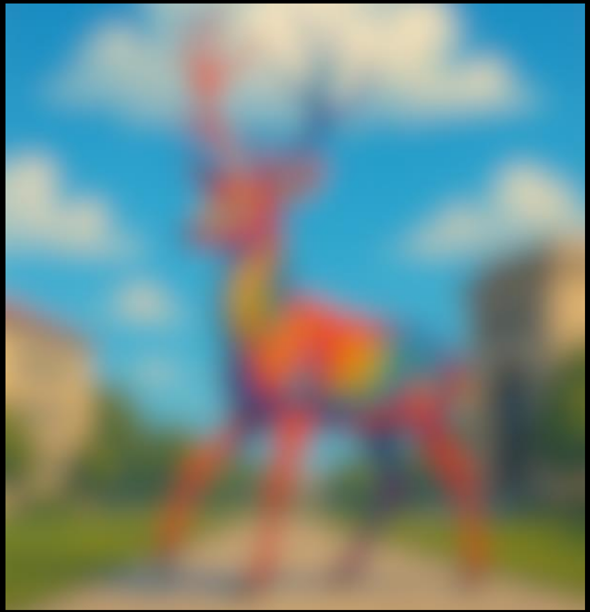


Probabilistic Generative Process

Start from initial distribution $p_T(x_T)$
e.g. Gaussian $N(0, I)$



Generate slightly improved data
$x_{t-1} \sim p_{t-1}(x_{t-1}|x_t)$



final data
$x_0 \sim p_0(x_0|x_1)$

Learning the generative model = learning the parameters for each $p_{t-1}(x_{t-1}|x_t)$

It is difficult to directly construct a series of probability distributions

(Forward) Noising Process

- adding Gaussian noise at each time step t $\mathrm{x_t} \sim q\ (x_t | x_{t-1}) = N(\sqrt{\alpha_t} x_{t-1}, \beta_t I)$, $\alpha_t = 1 - \beta_t$

equivalently $\mathrm{x_t} = \sqrt{\alpha_t} x_{t-1} + \sqrt{\beta_t} \epsilon, \epsilon \sim N(0, I)$

t=0 t=1 t=T

Consequence: with sufficient large T, the result will be Gaussian noise

(Reverse) Generation/Denoising Process

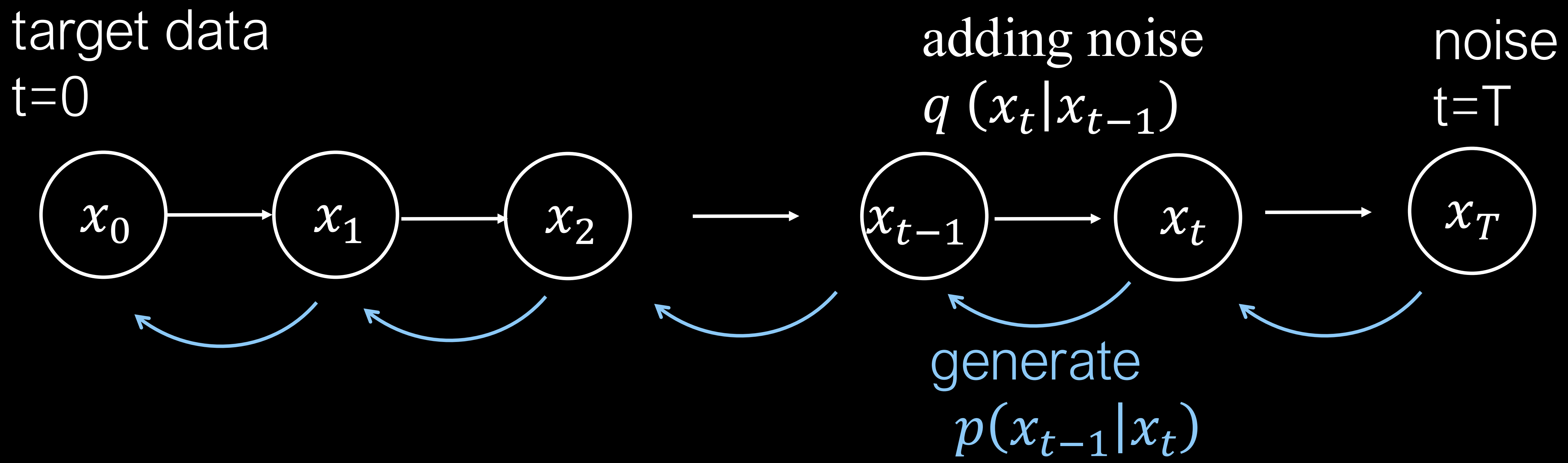
adding noise $q\ (x_t|x_{t-1})$

t=0 t=1 t=T

generate $p\ (x_{t-1}|x_t)$

Learning problem: how to find parameters of p to match the sequence of noised images?

Diffusion Model



Learning problem: how to find parameters of p to match the sequence of noised images?

Denoising Diffusion Probabilistic Models (DDPM)

- Reverse Process (Generation), aka denoising

$$\boxed{\begin{gathered} p_\theta = \prod_{t=1}^{T} p_\theta(x_{t-1}|x_t) \\ p_\theta(x_{t-1}|x_t) = N\big(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t)\big) \end{gathered}}$$

predicted by NN

- Forward Process (Diffusion), aka noising

$$\boxed{\begin{gathered} q(x_{1:T}|x_0) = \prod_{t=1}^{T} q(x_t|x_{t-1}) \\ q(x_t|x_{t-1}) = N\big(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I\big) \end{gathered}}$$

noising schedule: β_t in [0, 1], and follows a predefined schedule

DDPM Training

- For each data sample x_0 (e.g. image), random pick step t
  - sample a noise from standard Gaussian N(0,I) => ϵ
  - noise image at step t: $x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t}\epsilon$, $\bar{\alpha}_t = \prod_{s=1}^{t}(1 - \beta_s)$
  - estimate noise using Neural network: $\hat{\epsilon}_\theta = \mathrm{NN}_{\mathbf{forward}}(x_t, t)$
  - compute MSE loss: $L = ||\epsilon - \hat{\epsilon}_\theta||^2$
  - compute gradient via backward through NN
  - update model parameters using optimization alg (e.g. Adam)

DDPM training is essentially predicting the noise based on image

DDPM Inference

- randomly sample from Gaussian(0, I) ➔ x_T
- For step t = T to 1
  - estimate noise using Neural network: $\hat{\epsilon}_\theta = \mathrm{NN}_{\mathbf{forward}}(x_t, t)$
  - generate noise from N(0, I) ➔ η
  - estimate data mean: $\tilde{\mu}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\hat{\epsilon}_\theta\right)$
  - update data: $x_{t-1} = \tilde{\mu}_{t-1} + \sqrt{\beta_t}\eta$

DDPM inference is essentially iteratively removing predicted noise

Intuition and Theory

- Training objective:

$$L = -\log p_\theta(x_0)$$
$$\leq -\mathrm{E}\left[\log \frac{p_\theta(x_{0:T})}{q(x_{1:T}|x_0)}\right] = L_{elbo}$$

elbo loss

$$L_{elbo} = \sum_{t=1}^{T} \mathrm{KL}\big(q(x_{t-1}|x_t, x_0) \parallel p_\theta(x_{t-1}|x_t)\big) + \mathrm{KL}\big(q(x_T|x_0) \parallel p_\theta(x_T)\big) - \mathrm{E}\log p_\theta(x_0|x_1)$$

Hint: Jensen's inequality, posterior for linear Gaussian distribution

$$q(x_t|x_0) = N\left(x_t; \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)I\right)$$

$$\bar{\alpha}_t = \prod_{s=1}^{t}(1-\beta_s)$$

$$q(x_{t-1}|x_t, x_0) = N\left(x_{t-1}; \tilde{\mu}_t, \tilde{\beta}_t I\right)$$

$$\tilde{\mu}_t = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon\right)$$

$$\tilde{\beta}_t = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}$$

What about $p_\theta(x_{t-1}|x_t)$?

just set

$$p_\theta(x_{t-1}|x_t) = N(\mu_\theta(x_t, t), \sigma_t^2 I)$$

$$\mathrm{KL}\big(q(x_{t-1}|x_t, x_0) \parallel p_\theta(x_{t-1}|x_t)\big)$$

becomes

$$\mathrm{KL}\left(N\big(x_{t-1}; \tilde{\mu}_t, \tilde{\beta}_t I\big) \parallel N(\mu_\theta(x_t, t), \sigma_t^2 I)\right)$$

let $\sigma_t^2 = \tilde{\beta}_t$, and let

$$\mu_\theta(x_t, t) = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\hat{\epsilon}_\theta(x_t, t)\right)$$

$$\min \|\tilde{\mu}_t - \mu_\theta(x_t, t)\|^2$$

is equivalent to

$$\min \frac{\beta_t^2}{\alpha_t(1-\bar{\alpha}_t)}\|\epsilon - \hat{\epsilon}_\theta(x_t, t)\|^2$$

$\boxed{\text{Essentially, DDPM is using NN to predict noise given noised data}}$

Code Example (see colab)

Summary

- Diffusion Model
  - is a probabilistic generative model by iteratively removing noise
  - construct a sequence of corrupted data by applying Gaussian noises
  - use a neural network to estimate the noise given the corrupted data (without knowing the original clean data)
  - once estimate the noise, use posterior estimation to remove noise
  - can use any suitable NN (e.g. UNet)
  - much better than VAE, but much slower