Lecture 1

CME 296: Diffusion & Large Vision Models

Afshine Amidi & Shervine Amidi

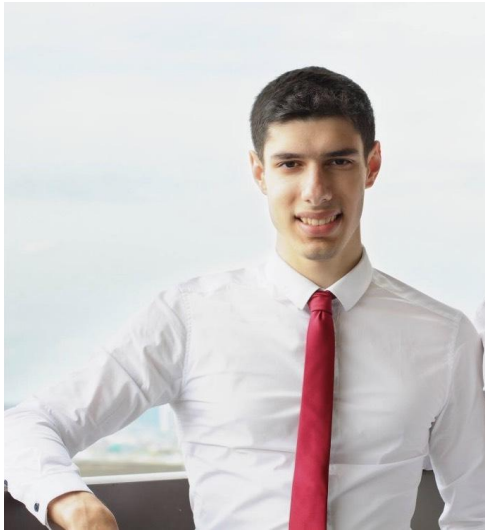
Stanford University

ICME

Teaching staff

Afshine and **Shervine**

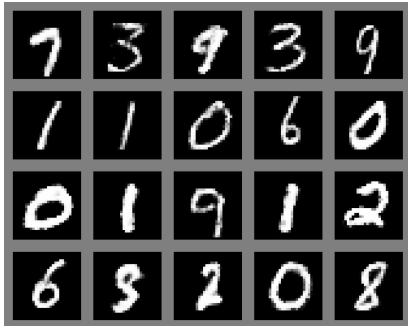
Teaching staff



Afshine
Centrale Paris ('16), MIT ('17)
Uber, Google, Netflix

Shervine
Centrale Paris ('16), Stanford ('19)
Uber, Google, Netflix

Motivation behind CME 296



Images from Generative Adversarial Networks, Goodfellow et al., 2014.

Motivation behind CME 296

Images generated by ChatGPT, March 2026.

Motivation behind CME 296

?

CME 296 overview

Goals.

1. Understand **paradigm** behind how we generate images
2. Learn **how** image generation models are **trained** & **used**

CME 296 overview

Goals.

1. Understand **paradigm** behind how we generate images
2. Learn **how** image generation models are **trained** & **used**

Audience. Interested in state-of-the-art image generation models

1. Career goal
2. Personal project
3. Curiosity

Prerequisites

- **Linear algebra**

  Vectors, matrices, gradient, divergence

Prerequisites

- **Linear algebra**

  Vectors, matrices, gradient, divergence

- **Probability theory**

  Bayes' rule, conditional/marginal probability, expectation, covariance, Gaussian distribution

Prerequisites

- **Linear algebra**

  Vectors, matrices, gradient, divergence

- **Probability theory**

  Bayes' rule, conditional/marginal probability, expectation, covariance, Gaussian distribution

- **Differential equations**

  Ordinary/stochastic differential equations, numerical solvers

Prerequisites

- **Linear algebra**

  Vectors, matrices, gradient, divergence

- **Probability theory**

  Bayes' rule, conditional/marginal probability, expectation, covariance, Gaussian distribution

- **Differential equations**

  Ordinary/stochastic differential equations, numerical solvers

- **Basics of machine learning**

  Loss function, training, inference, neural network

Logistics

Date & time.

- Fridays from 3:30pm to 5:20pm
- Thornton 110

Logistics

Date & time.

- Fridays from 3:30pm to 5:20pm
- Thornton 110

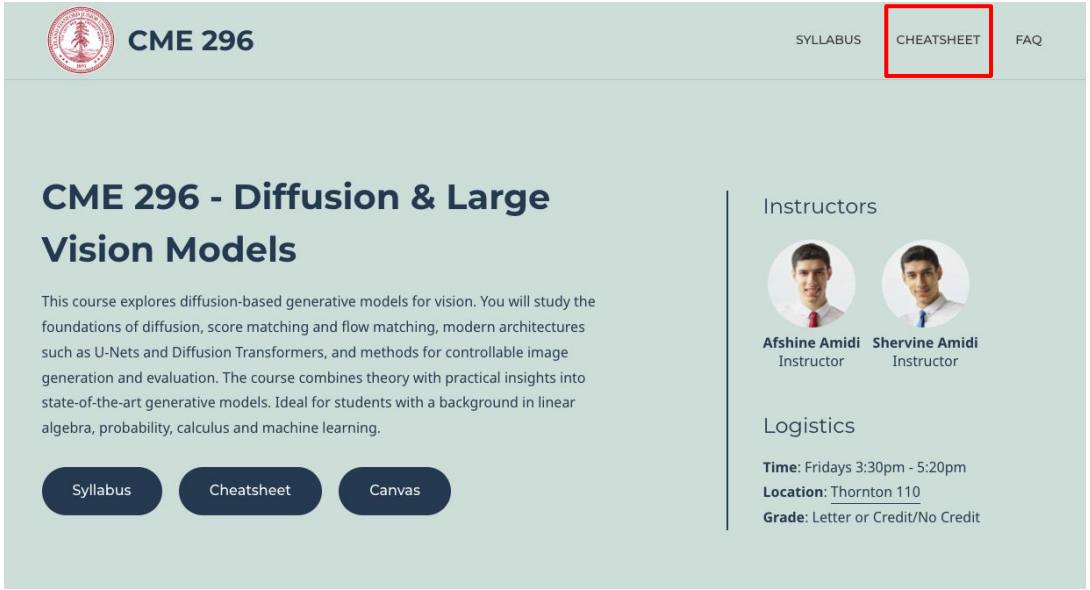
Details about the class.

- 2 units, Letter or Credit/No credit
- Lectures are recorded
- 2 exams (example for the LLM class)
  - Midterm (50% grade), scheduled on Friday May 1st
  - Final exam (50% grade), scheduled on Monday June 8th

Material

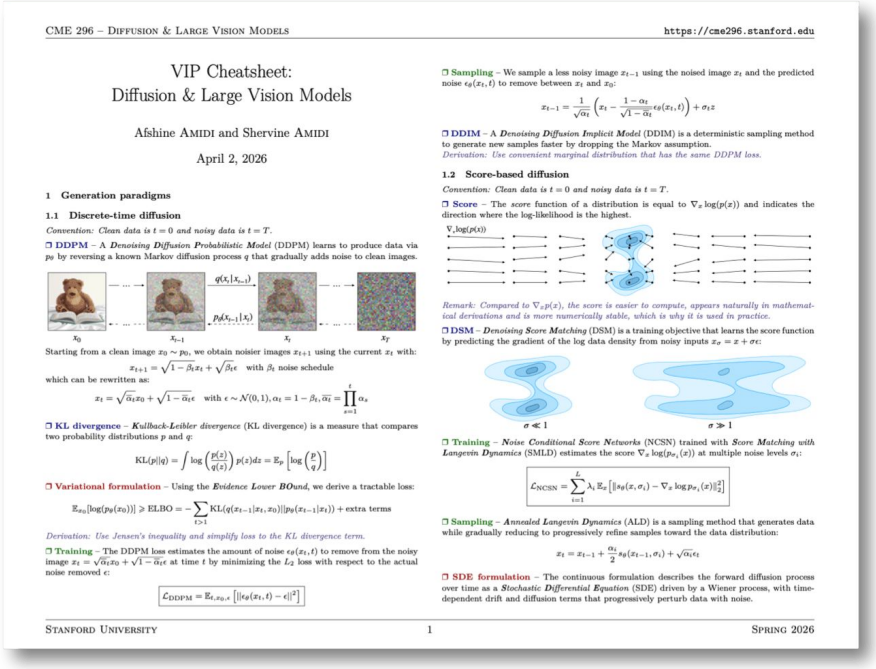
Class website. cme296.stanford.edu

- Contains syllabus & logistics
- Slides and recordings will be posted there



Material

VIP Cheatsheet



CME 296 – Diffusion & Large Vision Models — https://cme296.stanford.edu

VIP Cheatsheet:
Diffusion & Large Vision Models

Afshine Amidi and Shervine Amidi

April 2, 2026

1 Generation paradigms

1.1 Discrete-time diffusion

Convention: Clean data is $t = 0$ and noisy data is $t = T$.

❒ DDPM – A *Denoising Diffusion Probabilistic Model* (DDPM) learns to produce data via p_θ by reversing a known Markov diffusion process q that gradually adds noise to clean images.

$q(x_t|x_{t-1})$ $p_\theta(x_{t-1}|x_t)$

x_0 x_{t-1} x_t x_T

Starting from a clean image $x_0 \sim p_0$, we obtain noisier images x_{t+1} using the current x_t with:

$$x_{t+1} = \sqrt{1-\beta_t}x_t + \sqrt{\beta_t}\epsilon \quad \text{with } \beta_t \text{ noise schedule}$$

which can be rewritten as:

$$x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon \quad \text{with } \epsilon \sim \mathcal{N}(0,1), \alpha_t = 1-\beta_t, \bar{\alpha}_t = \prod_{s=1}^{t}\alpha_s$$

❒ KL divergence – *Kullback-Leibler divergence* (KL divergence) is a measure that compares two probability distributions p and q:

$$\mathrm{KL}(p||q) = \int \log\left(\frac{p(z)}{q(z)}\right)p(z)dz = \mathbb{E}_p\left[\log\left(\frac{p}{q}\right)\right]$$

❒ Variational formulation – Using the *Evidence Lower BOund*, we derive a tractable loss:

$$\mathbb{E}_{x_0}[\log(p_\theta(x_0))] \geqslant \mathrm{ELBO} = -\sum_{t>1}\mathrm{KL}(q(x_{t-1}|x_t,x_0)||p_\theta(x_{t-1}|x_t)) + \text{extra terms}$$

Derivation: Use Jensen's inequality and simplify loss to the KL divergence term.

❒ Training – The DDPM loss estimates the amount of noise $\epsilon_\theta(x_t,t)$ to remove from the noisy image $x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon$ at time t by minimizing the L_2 loss with respect to the actual noise removed ϵ:

$$\boxed{\mathcal{L}_{\mathrm{DDPM}} = \mathbb{E}_{t,x_0,\epsilon}\left[||\epsilon_\theta(x_t,t)-\epsilon||^2\right]}$$

❒ Sampling – We sample a less noisy image x_{t-1} using the noised image x_t and the predicted noise $\epsilon_\theta(x_t,t)$ to remove between x_t and x_0:

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t,t)\right) + \sigma_t z$$

❒ DDIM – A *Denoising Diffusion Implicit Model* (DDIM) is a deterministic sampling method to generate new samples faster by dropping the Markov assumption.

Derivation: Use convenient marginal distribution that has the same DDPM loss.

1.2 Score-based diffusion

Convention: Clean data is $t = 0$ and noisy data is $t = T$.

❒ Score – The score function of a distribution is equal to $\nabla_x \log(p(x))$ and indicates the direction where the log-likelihood is the highest.

$\nabla_x \log(p(x))$

Remark: Compared to $\nabla_x p(x)$, the score is easier to compute, appears naturally in mathematical derivations and is more numerically stable, which is why it is used in practice.

❒ DSM – *Denoising Score Matching* (DSM) is a training objective that learns the score function by predicting the gradient of the log data density from noisy inputs $x_\sigma = x + \sigma\epsilon$:

$\sigma \ll 1$ $\sigma \gg 1$

❒ Training – *Noise Conditional Score Networks* (NCSN) trained with *Score Matching with Langevin Dynamics* (SMLD) estimates the score $\nabla_x \log(p_{\sigma_i}(x))$ at multiple noise levels σ_i:

$$\boxed{\mathcal{L}_{\mathrm{NCSN}} = \sum_{i=1}^{L}\lambda_i\,\mathbb{E}_x\left[||s_\theta(x,\sigma_i) - \nabla_x \log p_{\sigma_i}(x)||_2^2\right]}$$

❒ Sampling – *Annealed Langevin Dynamics* (ALD) is a sampling method that generates data while gradually reducing to progressively refine samples toward the data distribution:

$$x_t = x_{t-1} + \frac{\alpha_t}{2}s_\theta(x_{t-1},\sigma_t) + \sqrt{\alpha_t}\epsilon_t$$

❒ SDE formulation – The continuous formulation describes the forward diffusion process over time as a *Stochastic Differential Equation* (SDE) driven by a Wiener process, with time-dependent drift and diffusion terms that progressively perturb data with noise.

Stanford University 1 Spring 2026

Link to PDF: https://github.com/afshinea/stanford-cme-296-diffusion-large-vision-models

Class communications

Canvas.

- Announcements
- Class discussions via Ed

Any other general inquiries / questions, email us:

- cme296-spr2526-staff@lists.stanford.edu
- afshine@stanford.edu, shervine@stanford.edu

Example of a slide in this class

Explanation of a CME 296 concept

Example of a slide in this class

Explanation of a CME 296 concept

Source & suggested reading, if interested

Some paper

Important note on conventions

⚠ Field is full of different notations / conventions ⚠

$t \rightarrow [0, 1]$ or $[0, T]$

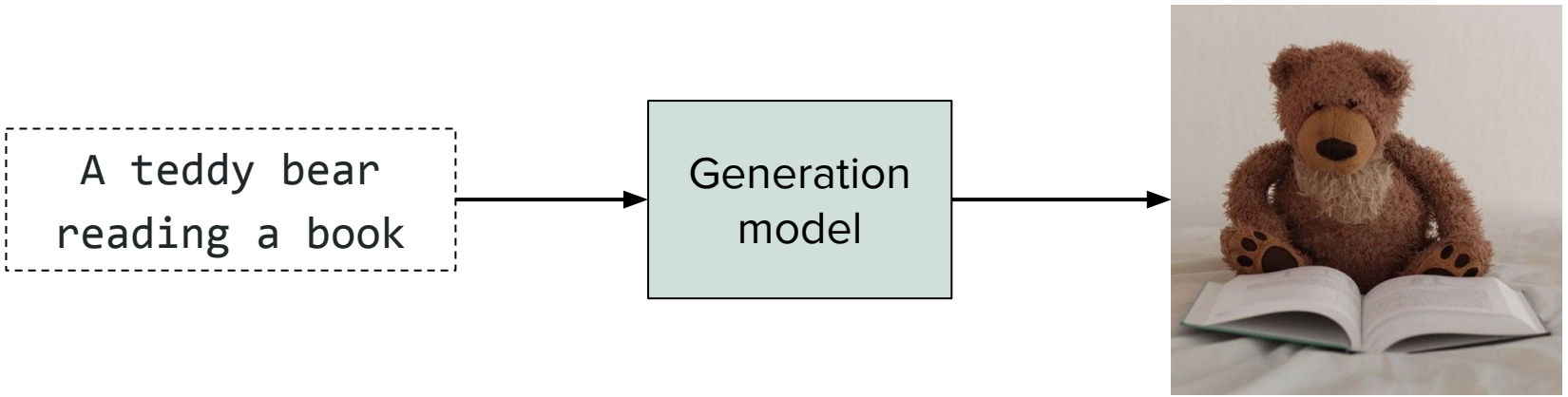
$\epsilon \rightarrow z$, x_0, x_1, ...

etc.

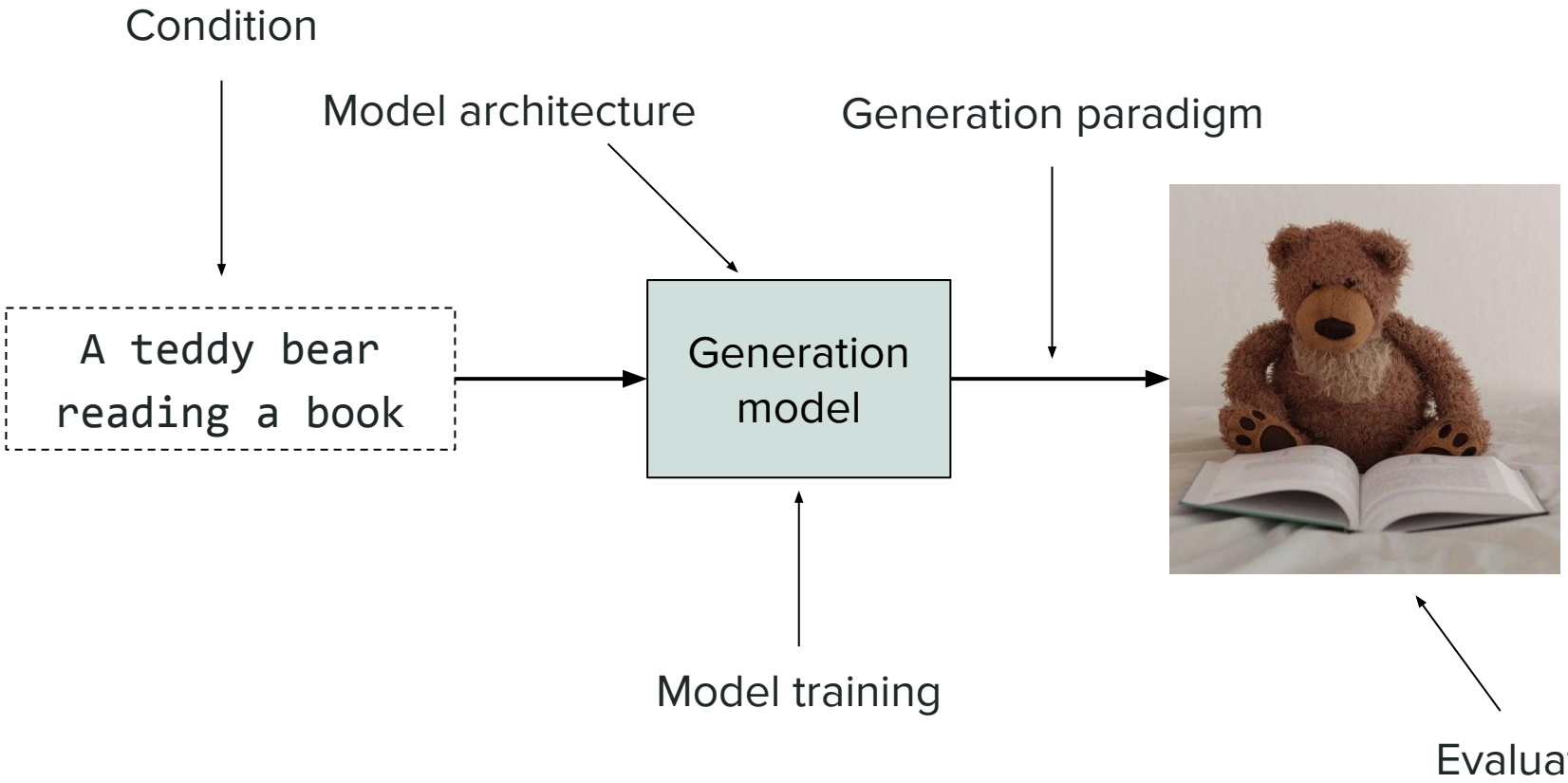
CME 296 will (try to):

1. Rely on the **most commonly-used** notations
2. **Stay consistent** throughout the class when that makes sense
3. **Call out** unintuitive change in notations

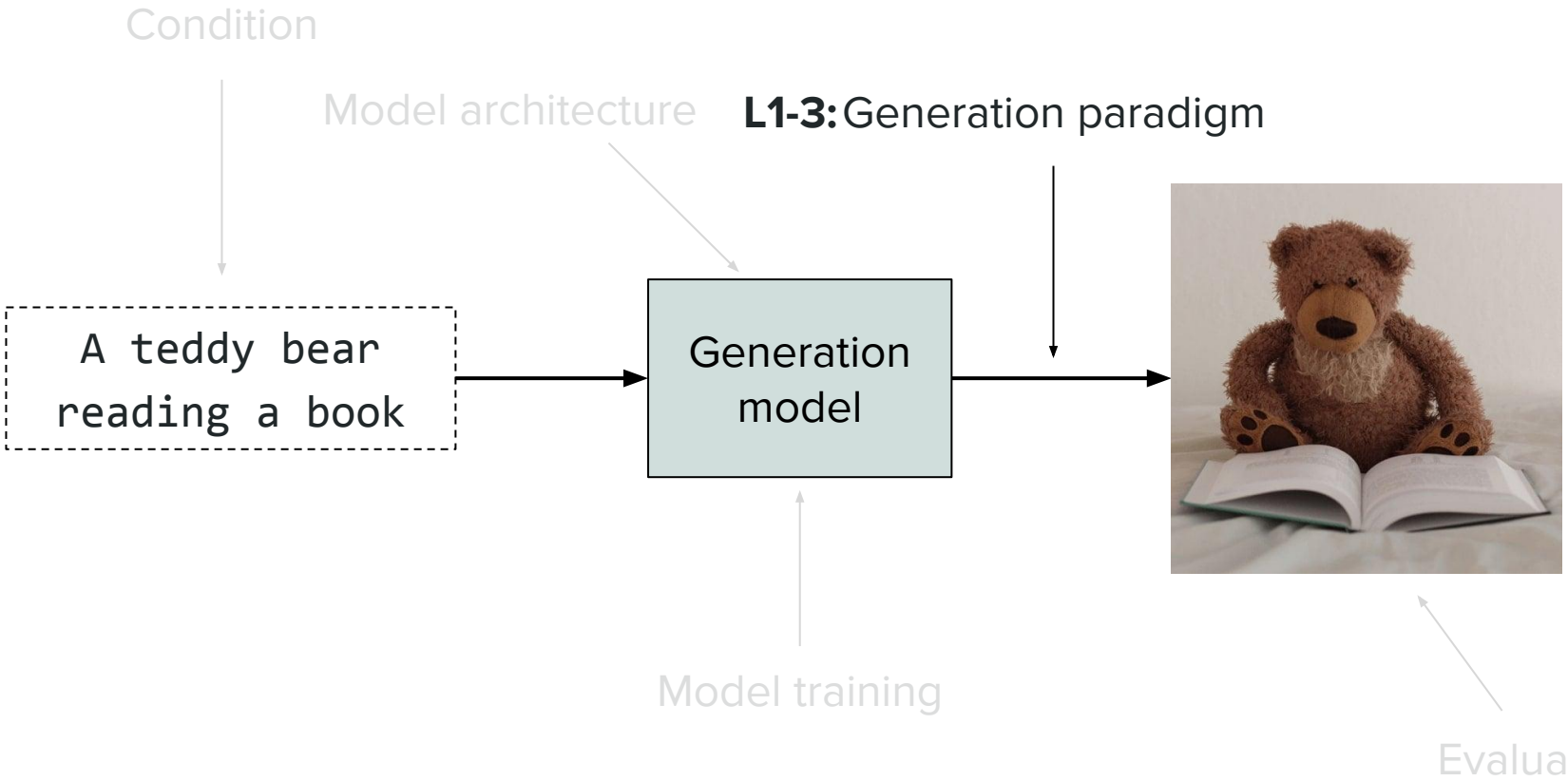
Outline of this class



Outline of this class



Outline of this class



Outline of this class

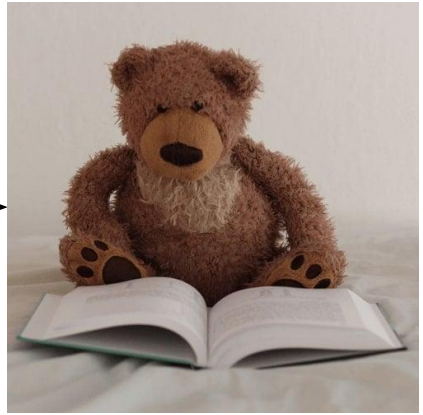
L4: Condition

Model architecture

L1-3: Generation paradigm

A teddy bear reading a book

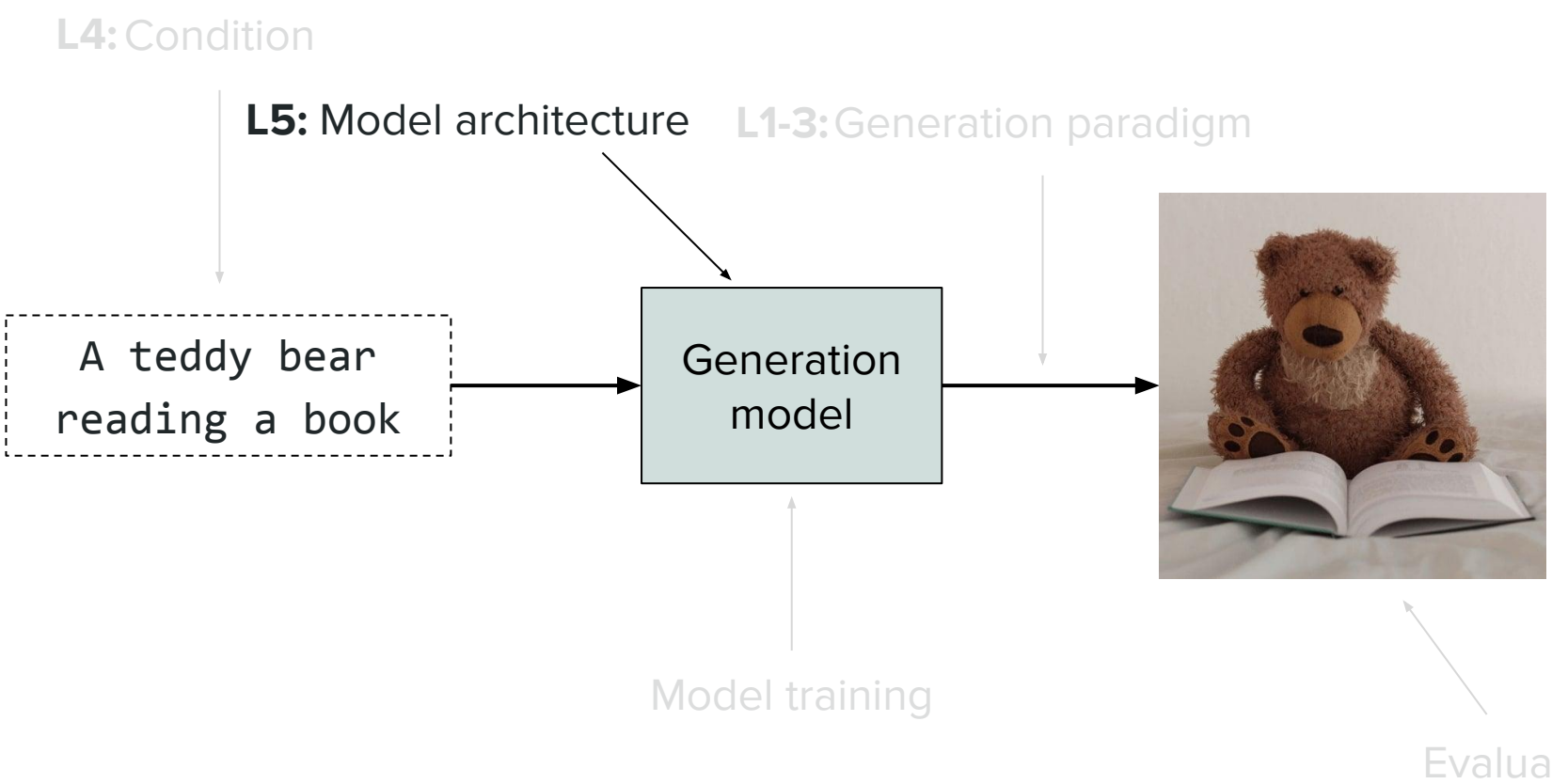
Generation model



Model training

Evaluation

Outline of this class



Outline of this class

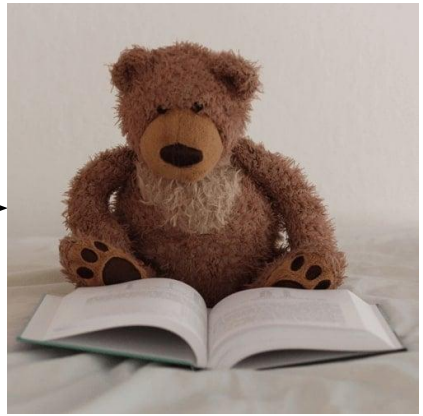
L4: Condition

L5: Model architecture

L1-3: Generation paradigm

A teddy bear reading a book

Generation model



L6: Model training

Evaluation

Outline of this class

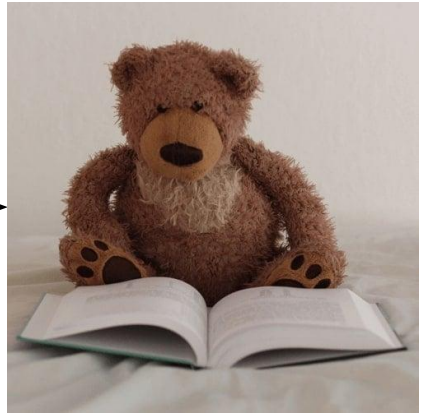
L4: Condition

L5: Model architecture

L1-3: Generation paradigm

A teddy bear reading a book

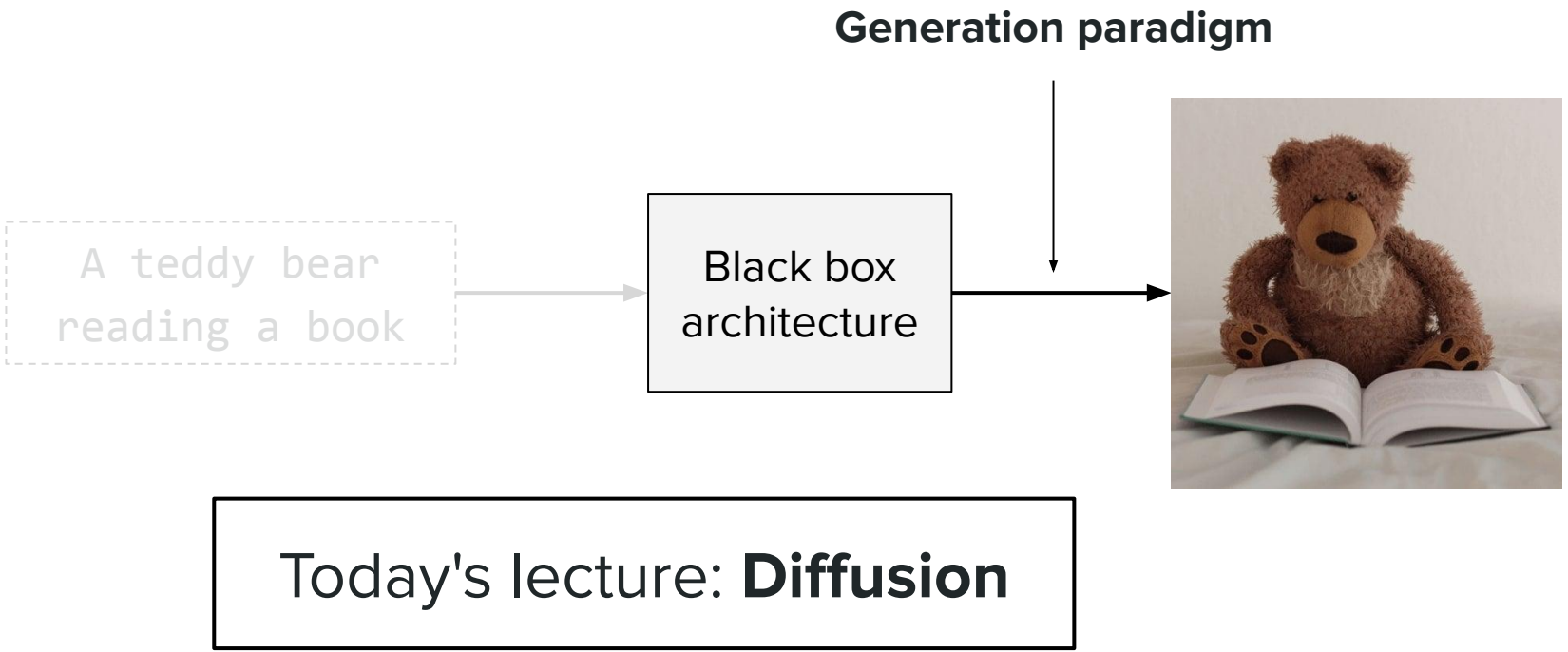
Generation model



L6: Model training

L7: Evaluation

First three lectures: generation paradigms



Today's lecture: **Diffusion**

Diffusion & Large Vision Models

Motivation

Forward / backwards process

Variational formulation

Training

Inference

Faster sampling

Problem formulation

Suppose we are given a **sample** of observations from p_{data}

Sample

New generation

Objective. Generate new images from a distribution of images p_{data}

Problem formulation

Suppose we are given a **sample** of observations from p_{data}

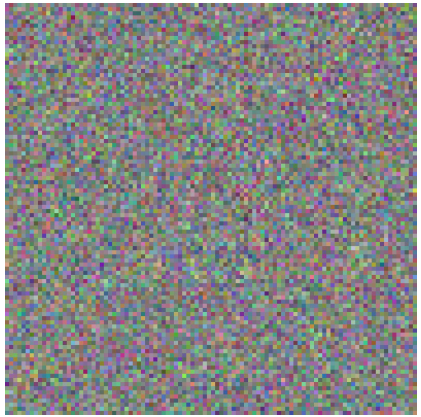
Sample

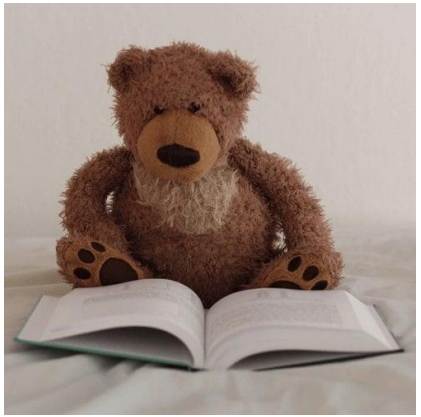
New generation

Objective. **Generate** new images from a distribution of images p_{data}

Unconditioned ← Focus of the 3 first lectures

Strategy: generate from noise





Why start from noise?

- Noise is **easy** to sample
- Introduces **randomness**
- Gaussian distribution has **nice properties**

Analogy: building a sculpture

The sculpture is already complete within the marble block, before I start my work. It is already there, I just have to chisel away the superfluous material.

–Michelangelo

Diffusion & Large Vision Models

Motivation

Forward / backwards process

Variational formulation

Training

Inference

Faster sampling

Intuition behind diffusion

1. Add **noise** (**forward process**)

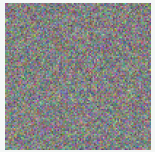
$x_{\text{image}} = x_0$

x_{t-1}

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

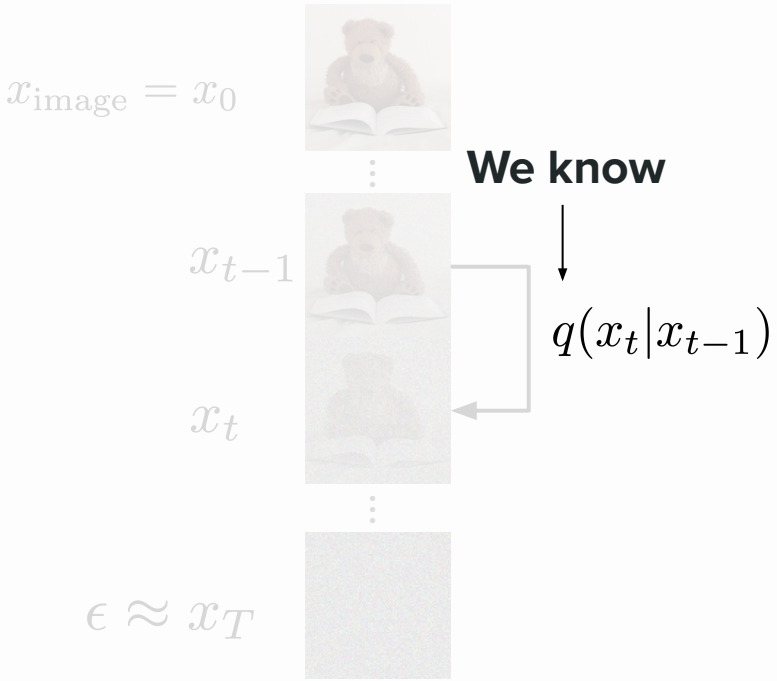
x_t

$\epsilon \approx x_T$



Intuition behind diffusion

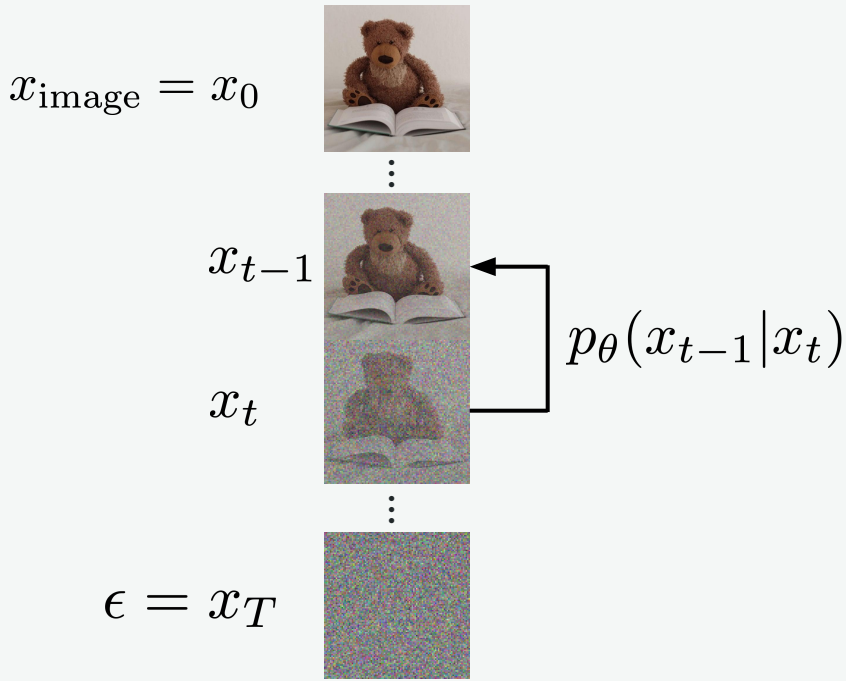
1. Add **noise** (forward process)



Intuition behind diffusion

1. Add **noise** (**forward process**)

2. Learn to **denoise** it (**reverse process**)



Intuition behind diffusion

1. Add **noise** (**forward process**)

2. Learn to **denoise** it (**reverse process**)

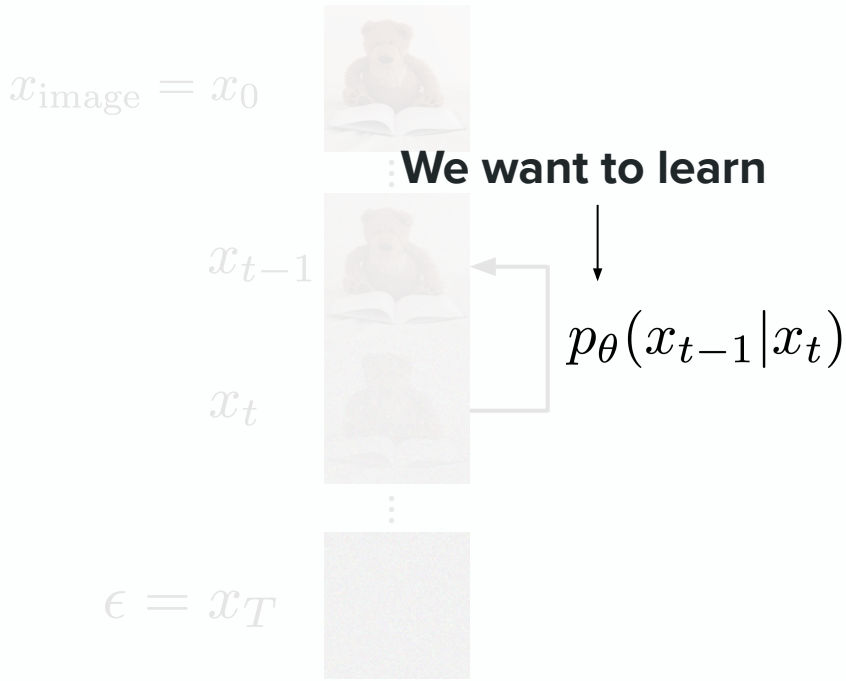
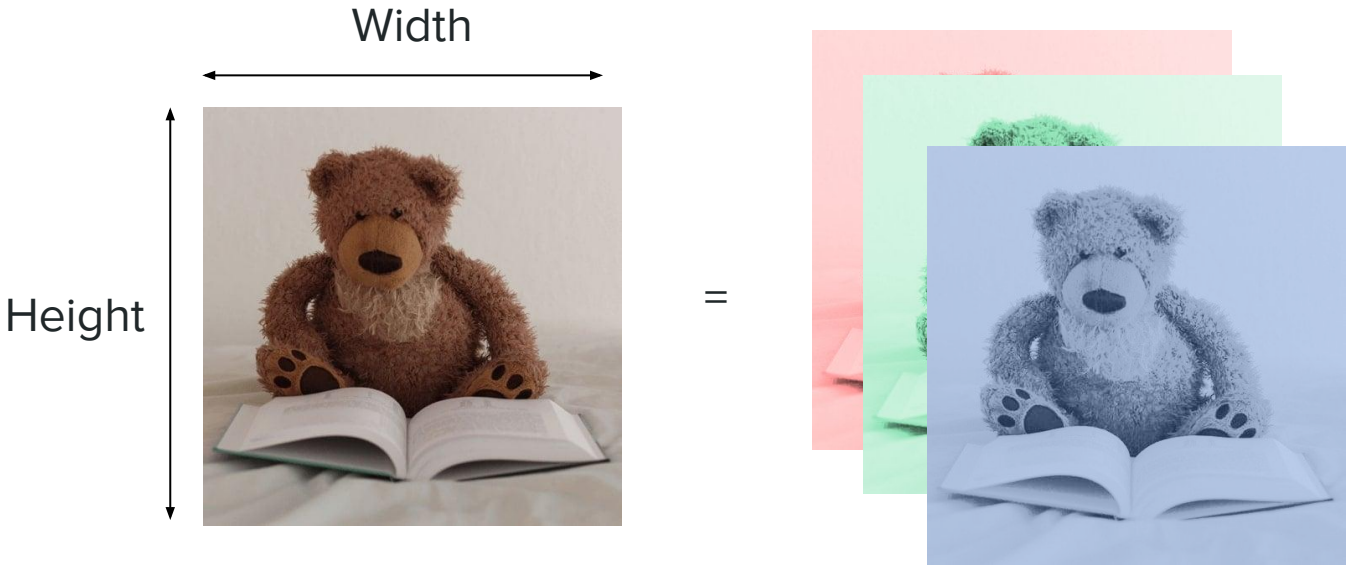


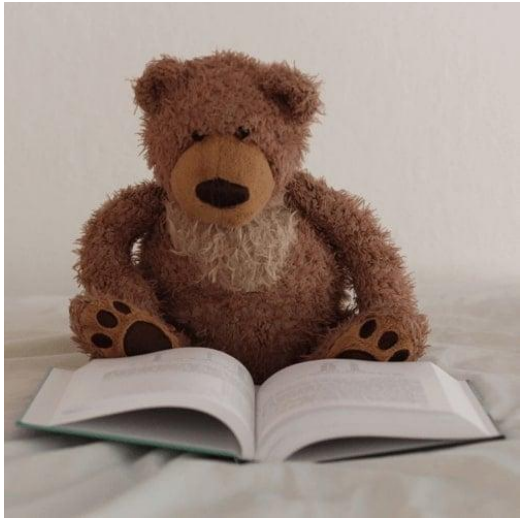
Image representation



R G B

pixel = (0 to 255, 0 to 255, 0 to 255)

Image representation



$$\longleftrightarrow \quad x_{\text{image}} = \begin{pmatrix} x^{(1)} \\ x^{(2)} \\ \vdots \\ x^{(n)} \end{pmatrix}$$

Probability distribution in 1-D space

$$x \sim \mathcal{N}(\mu, \sigma^2)$$

Scalar

Expectation

Variance

Probability distribution in high-dim space

$$x \sim \mathcal{N}(\mu, \Sigma)$$

Vector Expectation **Covariance**

$$\begin{pmatrix} x^{(1)} \\ x^{(2)} \\ \vdots \\ x^{(n)} \end{pmatrix} \quad \begin{pmatrix} \mu^{(1)} \\ \mu^{(2)} \\ \vdots \\ \mu^{(n)} \end{pmatrix} \quad \begin{pmatrix} \Sigma^{(1,1)} & \Sigma^{(1,2)} & \dots & \Sigma^{(1,n)} \\ \Sigma^{(2,1)} & \Sigma^{(2,2)} & \dots & \Sigma^{(2,n)} \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma^{(n,1)} & \Sigma^{(n,2)} & \dots & \Sigma^{(n,n)} \end{pmatrix}$$

Meaning of probability distribution in high-dim space

$$x \sim \mathcal{N}(\mu, \Sigma)$$

$$x^{(i)} \sim \mathcal{N}(\mu^{(i)}, \Sigma^{(i,i)}) \quad \text{and} \quad \mathrm{Cov}(x^{(i)}, x^{(j)}) = \Sigma^{(i,j)}$$

Good news. We will mainly (if not only) be dealing with **isotropic** Gaussians:

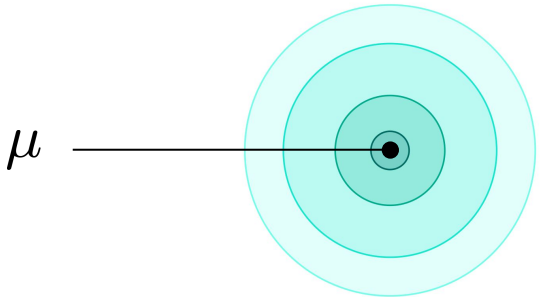
$$\Sigma = \sigma^2 I = \begin{pmatrix} \sigma^2 & 0 & \cdots & 0 \\ 0 & \sigma^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma^2 \end{pmatrix}$$

Meaning of probability distribution in high-dim space

$$x \sim \mathcal{N}(\mu, \Sigma)$$

$$x^{(i)} \sim \mathcal{N}(\mu^{(i)}, \Sigma^{(i,i)}) \quad \text{and} \quad \mathrm{Cov}(x^{(i)}, x^{(j)}) = \Sigma^{(i,j)}$$

Good news. We will mainly (if not only) be dealing with **isotropic** Gaussians:



same variance
in all directions

Meaning of probability distribution in high-dim space

$$x \sim \mathcal{N}(\mu, \Sigma)$$

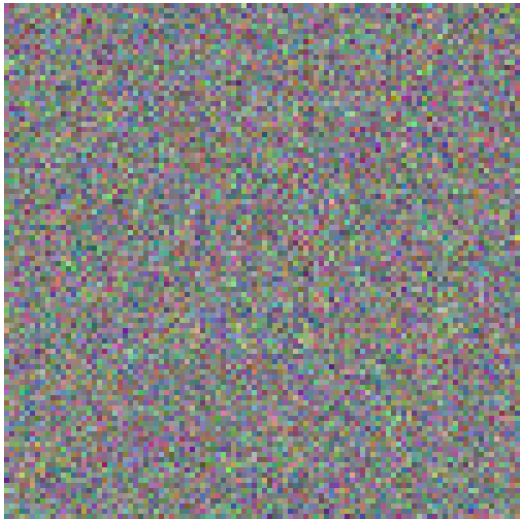
$$x^{(i)} \sim \mathcal{N}(\mu^{(i)}, \Sigma^{(i,i)}) \quad \text{and} \quad \mathrm{Cov}(x^{(i)}, x^{(j)}) = \Sigma^{(i,j)}$$

Good news. We will mainly (if not only) be dealing with **isotropic** Gaussians:

$$p(x) = \frac{1}{(2\pi\sigma^2)^{d/2}} \exp\left(-\frac{\|x-\mu\|^2}{2\sigma^2}\right)$$

probability density
is known and "easy"

Noise representation

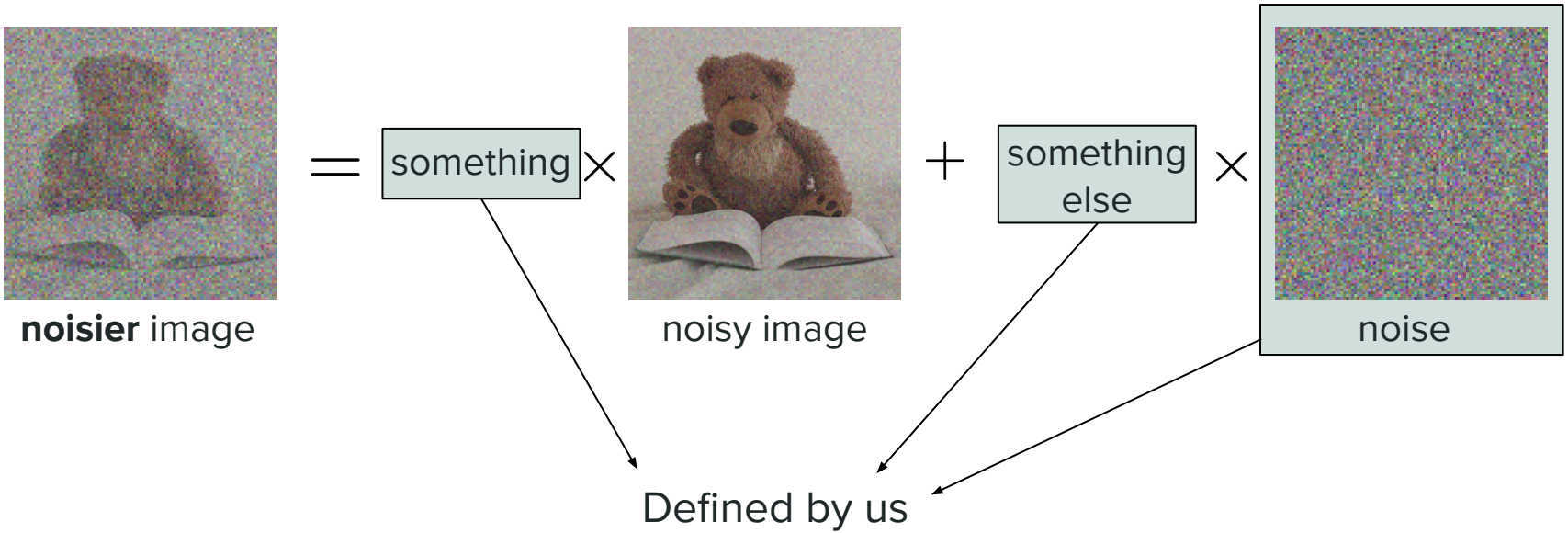


$$\longleftrightarrow \quad \epsilon = \begin{pmatrix} \epsilon^{(1)} \\ \epsilon^{(2)} \\ \vdots \\ \epsilon^{(n)} \end{pmatrix} \sim \mathcal{N}(0, I)$$

In other words, $\epsilon^{(i)} \sim \mathcal{N}(0, 1)$ independent and identically distributed

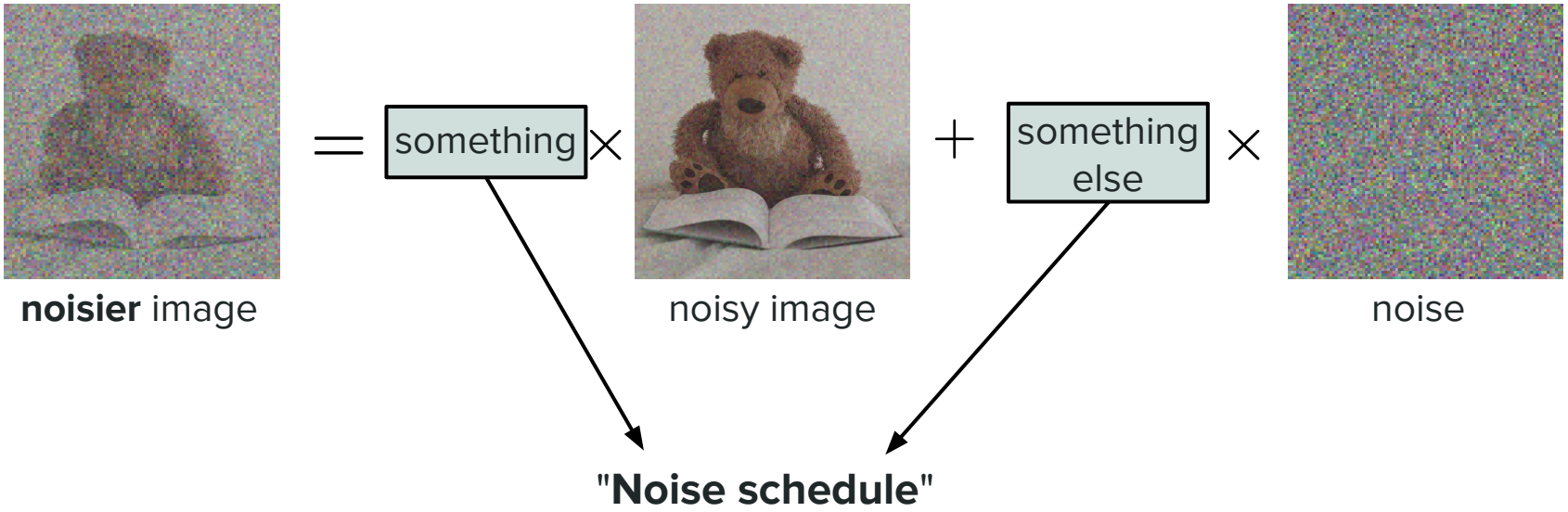
Process behind this

q is **chosen** by us and its **distribution** is **known** (reminder: it is stochastic!)



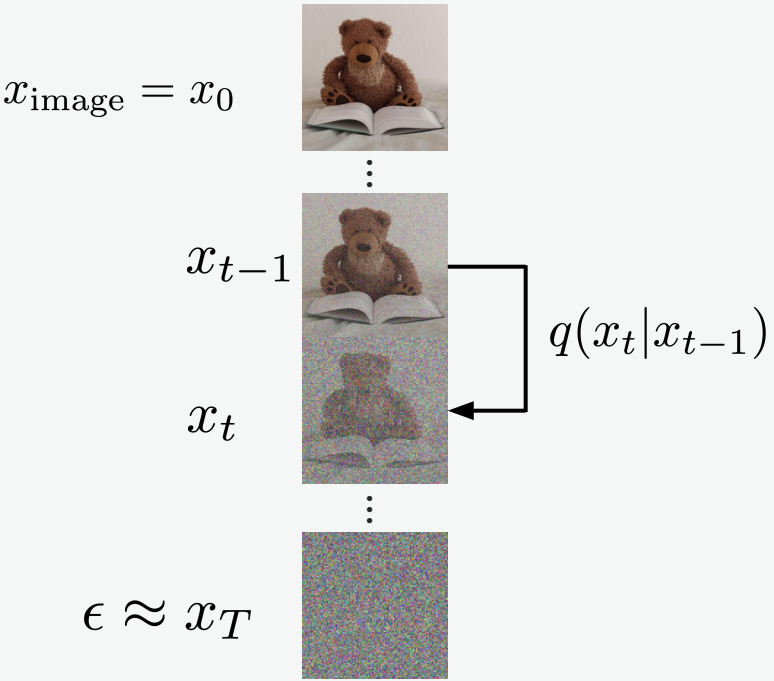
Process behind this

q is **chosen** by us and its **distribution** is **known** (reminder: it is stochastic!)



Mathematical formalism of forward process

1. Add **noise** (**forward process**)



Noisier image

Noise

$$q(x_t|x_{t-1}) = \mathcal{N}(\sqrt{1-\beta_t}x_{t-1}, \beta_t I)$$

Previous image

with $0 \leqslant \beta_1 < \beta_2 < \ldots < \beta_T \leqslant 1$

noise schedule

Mathematical formalism of forward process

$$q(x_t|x_{t-1}) = \mathcal{N}(\sqrt{1-\beta_t}x_{t-1}, \beta_t I)$$

$$\downarrow$$

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

with $\bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i$ and $\alpha_t = 1 - \beta_t$

Derivation: Variance of sum of independent Gaussians is sum of respective variances.

Diffusion & Large Vision Models

Motivation

Forward / backwards process

Variational formulation

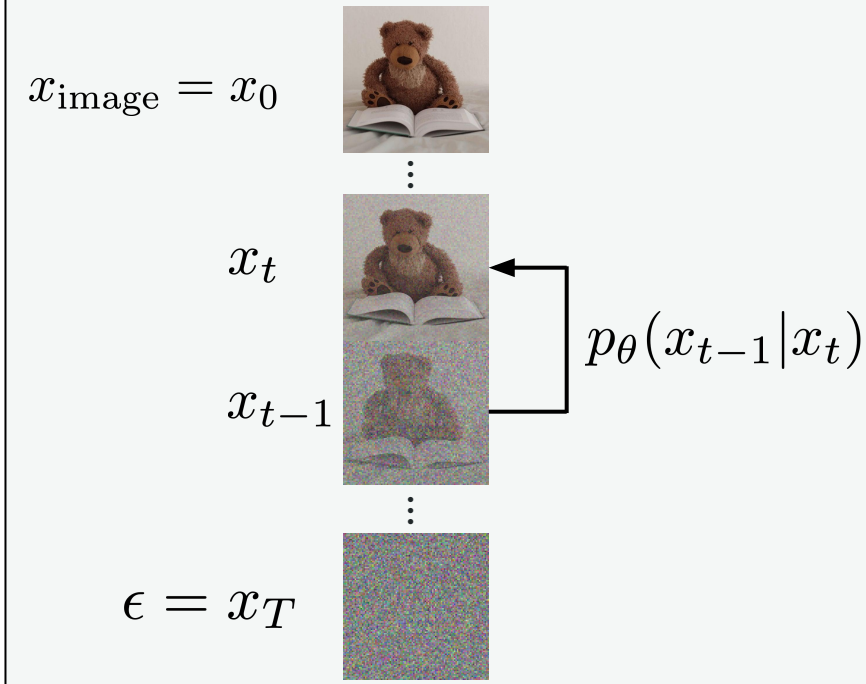
Training

Inference

Faster sampling

Learn reverse process

2. Learn to **denoise** it (**reverse process**)



Learn reverse process

Objective. We want to learn:

$$p_\theta$$

Objective

$$\max_{\theta} \quad p_\theta(x_0)$$

find parameters θ that maximize the quantity

probability given by the model of parameters θ

training images

Objective

computationally stable +

nice properties from log

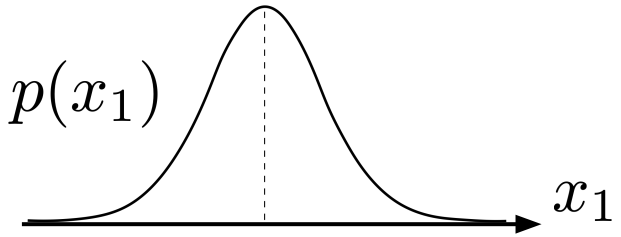
$$\max_{\theta} \log p_\theta(x_0)$$

find parameters θ that maximize the quantity

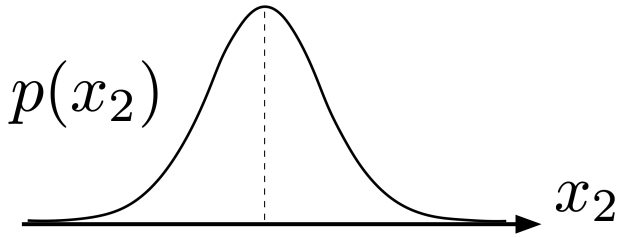
probability given by the model of parameters θ

training images

Refresher: joint probability distribution



Probability density of x_1



Probability density of x_2

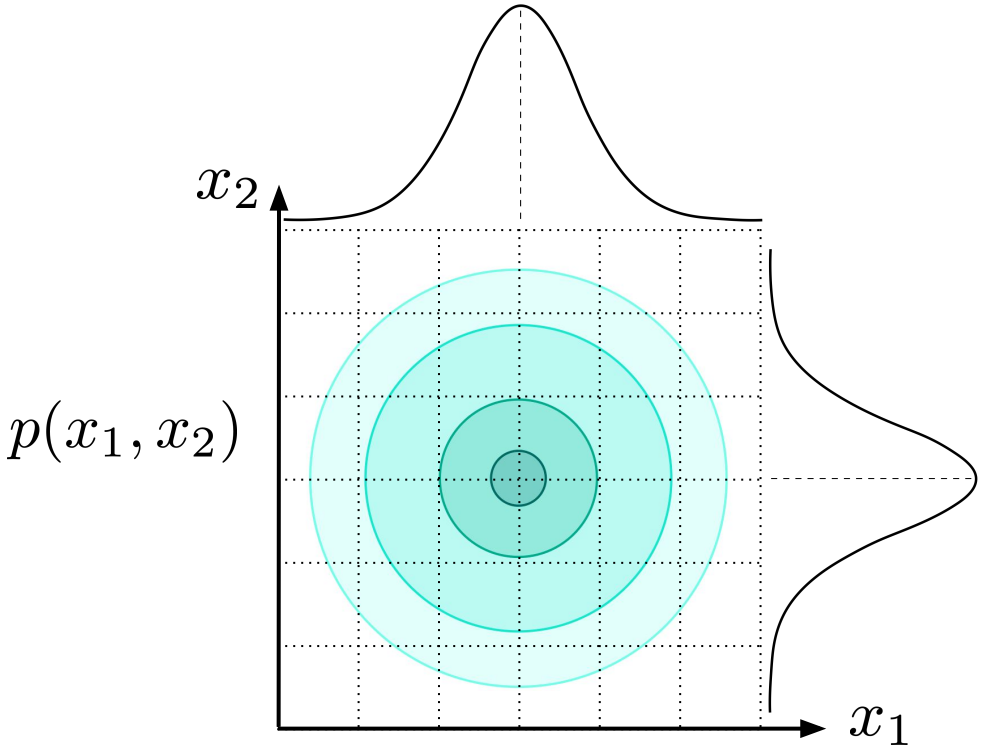
Refresher: joint probability distribution

Joint probability distribution.

$$p(x_1, x_2) = p(x_1) \times p(x_2|x_1)$$

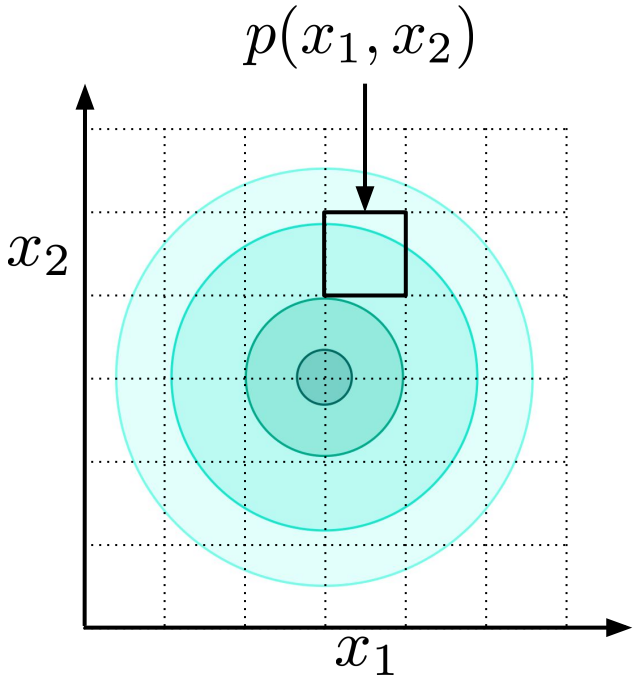
Conditional probability: if you know x_1 then what is the probability of x_2

Refresher: joint probability distribution



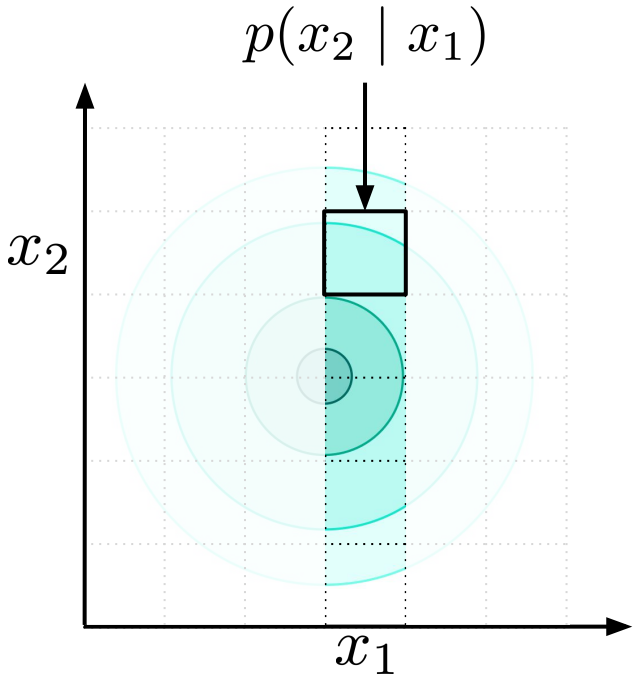
Joint probability $p(x_1, x_2)$ is the probability that we have x_1 **and** x_2

Refresher: joint probability distribution



Joint probability $p(x_1, x_2)$ is the probability that we have x_1 **and** x_2

Refresher: joint probability distribution



Conditional probability $p(x_2 \mid x_1)$

is the probability that we have x_2

knowing that we have x_1

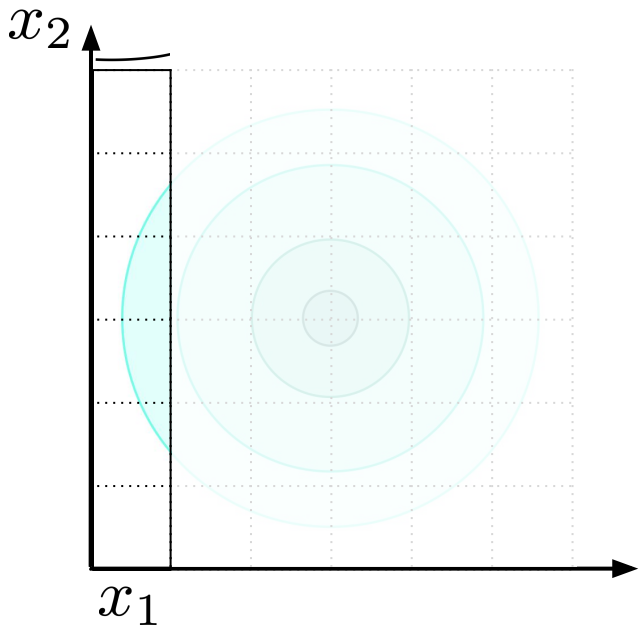
Refresher: joint probability distribution

Marginalization.

$$p(x_1) = \int p(x_1, x_2) dx_2$$

Sum across all x_2

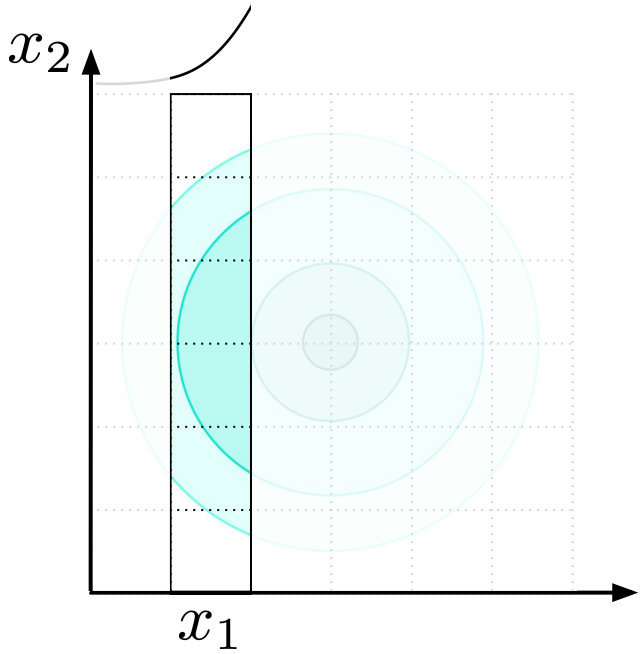
Refresher: joint probability distribution



$$p(x_1) = \int p(x_1, x_2) dx_2$$

Sum across all x_2

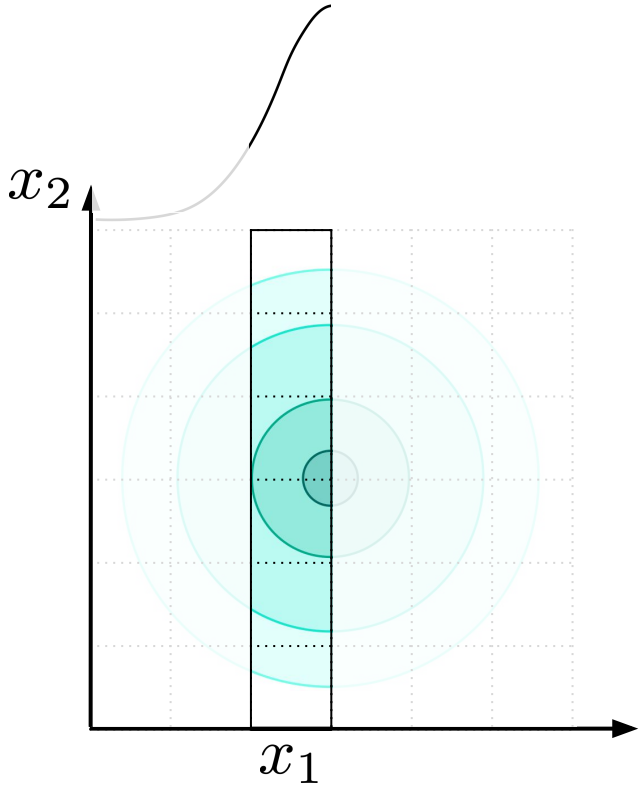
Refresher: joint probability distribution



$$p(x_1) = \int p(x_1, x_2) dx_2$$

Sum across all x_2

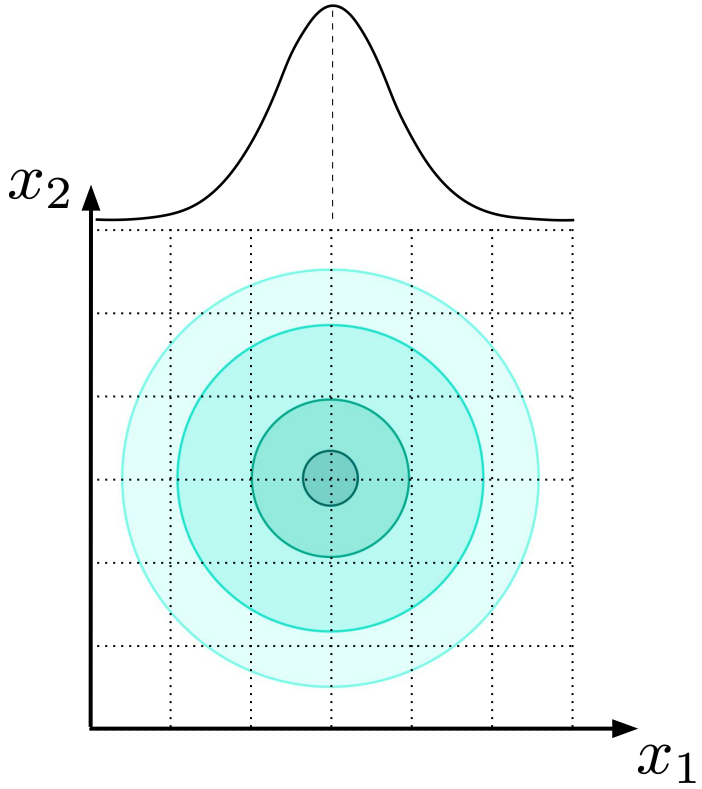
Refresher: joint probability distribution



$$p(x_1) = \int p(x_1, x_2) dx_2$$

Sum across all x_2

Refresher: joint probability distribution



$$p(x_1) = \int p(x_1, x_2) dx_2$$

Sum across all x_2

Refresher: joint probability distribution

Joint probability distribution.

$$p(x_1, x_2, ..., x_T) = p(x_1) \times p(x_2|x_1) \times ... \times p(x_T|x_1, ..., x_{T-1})$$

Marginalization.

$$p(x_1) = \int p(x_1, x_2, ..., x_T) dx_2 ... dx_T$$

Refresher: joint probability distribution

Joint probability distribution.

$$p(x_1, x_2, ..., x_T) = p(x_1) \times p(x_2|x_1) \times ... \times p(x_T|x_1, ..., x_{T-1})$$

$$p(x_1, x_2, ..., x_T) = p(x_{1:T})$$

Marginalization.

$$p(x_1) = \int p(x_1, x_2, ..., x_T) dx_2 ... dx_T$$

Refresher: joint probability distribution

Joint probability distribution (Simplified notation).

$$p(x_{1:T}) = p(x_1) \prod_{t=2}^{T} p(x_t | x_{1:t-1})$$

Marginalization (Simplified notation).

$$p(x_1) = \int p(x_{1:T}) dx_{2:T}$$

Problem

$$\log p_\theta(x_0) = \ ?$$

Derivation of a tractable loss

Strategy.

1. Derive **lower bound** for maximum (log-)likelihood estimation
2. **Expand terms** of lower bound
3. Show lower bound is **tractable**
4. Deduce **final loss function**

Step 1: Derivation of a lower bound

1 Derive **lower bound** for maximum (log-)likelihood estimation

$$\mathbb{E}_{x_0 \sim q(x_0)} \left[\log p_\theta(x_0) \right] \geqslant \mathbb{E}_{x_{0:T} \sim q(x_{0:T})} \left[\log \frac{p_\theta(x_{0:T})}{q(x_{1:T}|x_0)} \right]$$

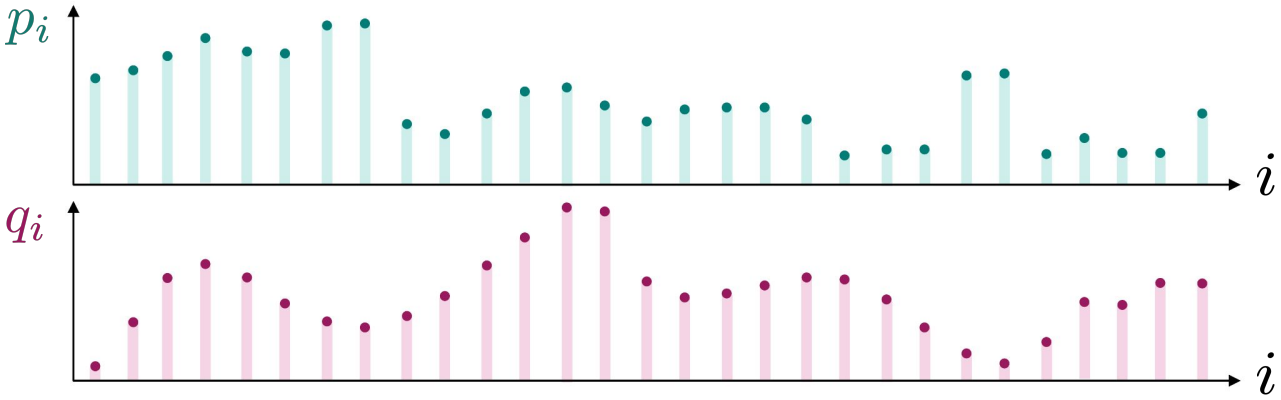
ELBO = **E**vidence **L**ower **BO**und

Derivation: Use Jensen's inequality on a convenient variational distribution

Full derivation in Denoising Diffusion Probabilistic Models, Ho et al., 2020.

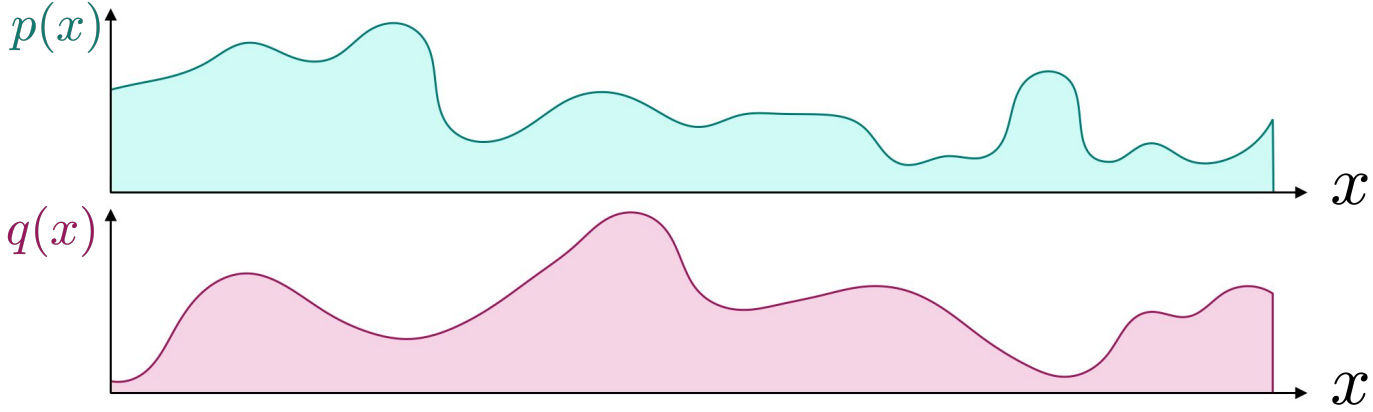
Refresher: KL divergence (discrete)

$$\mathrm{KL}(P\|Q) = \sum_{i=1}^{n} p_i \log\left(\frac{p_i}{q_i}\right)$$



Refresher: KL divergence (continuous)

$$\mathrm{KL}(P\|Q) = \int \log\left(\frac{p(x)}{q(x)}\right) p(x)\, dx = \mathbb{E}_{x\sim p}\left[\log\left(\frac{p(x)}{q(x)}\right)\right]$$



Step 2: Expand terms of the lower bound

2 **Expand terms** of lower bound

$$\text{ELBO} = -\sum_{t=2}^{T} \text{KL}\big(q(x_{t-1} \mid x_t, x_0) \,\|\, p_\theta(x_{t-1} \mid x_t)\big) + \text{some other terms}$$

Can we compute this?

We want to learn θ

Derivation: Use properties of the log function and rearrange terms

Full derivation in Denoising Diffusion Probabilistic Models, Ho et al., 2020.

Step 3: Show lower bound is tractable

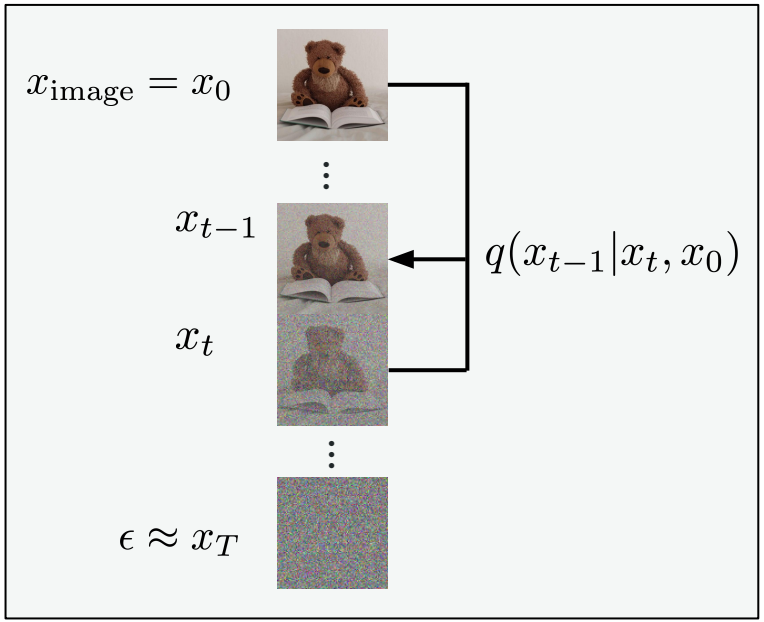
3. Show lower bound is **tractable**

   a. Show $q(x_{t-1} \mid x_t, x_0)$ is tractable

$$\text{ELBO} = -\sum_{t=2}^{T} \text{KL}\big(q(x_{t-1} \mid x_t, x_0) \,\|\, p_\theta(x_{t-1} \mid x_t)\big) + \text{some other terms}$$

Step 3: Show lower bound is tractable

3 a Show $q(x_{t-1} \mid x_t, x_0)$ is tractable



noisy image clean image

$$q(x_{t-1}|x_t, x_0)$$

less noisy image

Refresher: Bayes' rule

Likelihood

Posterior $p(A|B) = \frac{p(B|A)p(A)}{p(B)}$ Prior

Marginal likelihood, aka evidence

Step 3: Show lower bound is tractable

3 a Show $q(x_{t-1} \mid x_t, x_0)$ is tractable

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

sampled noise used to go from clean to noisy image

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

noisy image

Derivation: Compute density with Bayes' rule via $q(. \mid x_0)$ + Markov property

Full derivation in Denoising Diffusion Probabilistic Models, Ho et al., 2020.

Step 3: Show lower bound is tractable

3 Show lower bound is **tractable**

a Show $q(x_{t-1} \mid x_t, x_0)$ is tractable

b Show $p_\theta(x_{t-1} \mid x_t)$ is tractable

$$\text{ELBO} = -\sum_{t=2}^{T} \text{KL}\big(q(x_{t-1} \mid x_t, x_0) \,\|\, p_\theta(x_{t-1} \mid x_t)\big) + \text{some other terms}$$

Step 3: Show lower bound is tractable

3 b Show $p_\theta(x_{t-1} \mid x_t)$ is tractable

We assume that p_θ is Gaussian:

$$p_\theta(x_{t-1} \mid x_t) = \mathcal{N}\big(\mu_\theta(x_t, t), \Sigma_\theta(x_t, t)\big)$$

Justification: If $q(x_t|x_{t-1})$ Gaussian, then $q(x_{t-1}|x_t)$ is approx. Gaussian

Full derivation in Denoising Diffusion Probabilistic Models, Ho et al., 2020.

Step 4: Deduce final loss function

4 Deduce **loss function**

$$\mathcal{L}_{\text{DDPM}} = \mathbb{E}_{t,x_0,\epsilon}\left[\left\|\epsilon_\theta\left(\sqrt{\bar{\alpha}_t}\,x_0 + \sqrt{1-\bar{\alpha}_t}\,\epsilon,\, t\right) - \epsilon\right\|^2\right]$$

$t \sim \mathcal{U}\{1, T\}$ $x_0 \sim q_0(x_0)$ $\epsilon \sim \mathcal{N}(0, I)$

Derivation: Compute KL divergence between two Gaussians

Full derivation in Denoising Diffusion Probabilistic Models, Ho et al., 2020.

Summary of what we have done

Strategy.

1. Derive **lower bound** for maximum (log-)likelihood estimation ← **ELBO!**
2. **Expand terms** of lower bound ← **Surfaced KL divergence**
3. Show lower bound is **tractable** ← **Bayes' rule + Gaussian properties**
4. Deduce **final loss function** ← **Incredibly simple noise prediction!**

Diffusion & Large Vision Models

Motivation

Forward / backwards process

Variational formulation

Training

Inference

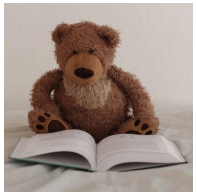
Faster sampling

Training recipe

1. Sample:

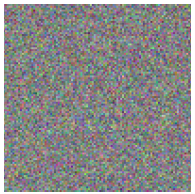
clean image

$x_0 \sim q_0(x_0)$



noise

$\epsilon \sim \mathcal{N}(0, I)$



time step

$t \sim \mathcal{U}\{1, T\}$

noised image

$$x_t = \sqrt{\overline{\alpha}_t} x_0 + \sqrt{1 - \overline{\alpha}_t}\epsilon$$

Training recipe

1. Sample:

clean image

$x_0 \sim q_0(x_0)$

noise

$\epsilon \sim \mathcal{N}(0, I)$

time step

$t \sim \mathcal{U}\{1, T\}$

noised image

$$x_t = \sqrt{\overline{\alpha}_t} x_0 + \sqrt{1 - \overline{\alpha}_t} \epsilon$$

2. Use x_t and t to **predict** ϵ via $\epsilon_\theta(x_t, t)$

Training recipe

1. Sample:

clean image $x_0 \sim q_0(x_0)$

noise $\epsilon \sim \mathcal{N}(0, I)$

time step $t \sim \mathcal{U}\{1, T\}$

noised image $x_t = \sqrt{\overline{\alpha}_t} x_0 + \sqrt{1 - \overline{\alpha}_t}\epsilon$

2. Use x_t and t to **predict** ϵ via $\epsilon_\theta(x_t, t)$

3. Compute **loss** $\mathcal{L} = ||\epsilon_\theta(x_t, t) - \epsilon||^2$ and **backpropagate** through ϵ_θ

Diffusion & Large Vision Models

Motivation

Forward / backwards process

Variational formulation

Training

Inference

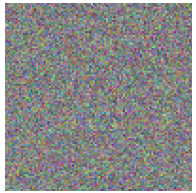
Faster sampling

Inference recipe

1. Sample:

noise

$$x_T \sim \mathcal{N}(0, I)$$



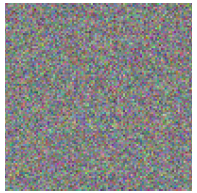
2. Perform **iterative update** from x_t to x_{t-1}

Inference recipe

1. Sample:

noise

$$x_T \sim \mathcal{N}(0, I)$$



2. Perform **iterative update** from x_t to x_{t-1}

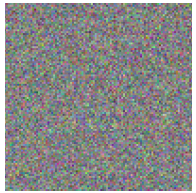
x_t

Inference recipe

1. Sample:

noise

$x_T \sim \mathcal{N}(0, I)$



2. Perform **iterative update** from x_t to x_{t-1}

x_t

$\epsilon_\theta(x_t, t)$

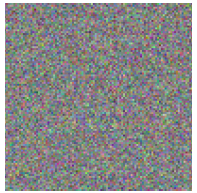
Predicted noise
to remove

Inference recipe

1. Sample:

noise

$$x_T \sim \mathcal{N}(0, I)$$



2. Perform **iterative update** from x_t to x_{t-1}

$$x_t - \frac{1-\alpha_t}{\sqrt{1-\overline{\alpha}_t}}\epsilon_\theta(x_t, t)$$

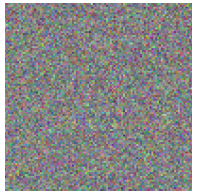
Predicted noise
to remove

Inference recipe

1. Sample:

noise

$$x_T \sim \mathcal{N}(0, I)$$



2. Perform **iterative update** from x_t to x_{t-1}

$$\frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{1-\alpha_t}{\sqrt{1-\overline{\alpha}_t}}\epsilon_\theta(x_t, t)\right) + \sigma_t z$$

Predicted noise to remove

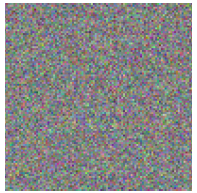
Sampled from a Normal distribution

Inference recipe

1. Sample:

noise

$$x_T \sim \mathcal{N}(0, I)$$



2. Perform **iterative update** from x_t to x_{t-1}

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{1-\alpha_t}{\sqrt{1-\overline{\alpha}_t}}\epsilon_\theta(x_t, t)\right) + \sigma_t z$$

Predicted noise to remove

Sampled from a Normal distribution

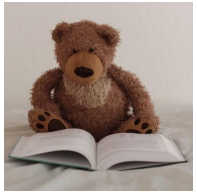
Inference recipe

1. Sample: noise $x_T \sim \mathcal{N}(0, I)$

2. Perform **iterative update** from x_t to x_{t-1}

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{1 - \alpha_t}{\sqrt{1 - \overline{\alpha}_t}} \epsilon_\theta(x_t, t) \right) + \sigma_t z$$

3. Obtain **final** image x_0



Landmark paper that is a must-read

DDPM = **D**enoising **D**iffusion **P**robabilistic **M**odels

Denoising Diffusion Probabilistic Models

Jonathan Ho
UC Berkeley
jonathanho@berkeley.edu

Ajay Jain
UC Berkeley
ajayj@berkeley.edu

Pieter Abbeel
UC Berkeley
pabbeel@cs.berkeley.edu

Abstract

We present high quality image synthesis results using diffusion probabilistic models, a class of latent variable models inspired by considerations from nonequilibrium thermodynamics. Our best results are obtained by training on a weighted variational bound designed according to a novel connection between diffusion probabilistic models and denoising score matching with Langevin dynamics, and our models naturally admit a progressive lossy decompression scheme that can be interpreted as a generalization of autoregressive decoding. On the unconditional CIFAR10 dataset, we obtain an Inception score of 9.46 and a state-of-the-art FID score of 3.17. On 256x256 LSUN, we obtain sample quality similar to ProgressiveGAN. Our implementation is available at https://github.com/hojonathanho/diffusion.

Denoising Diffusion Probabilistic Models, Ho et al., 2020.

Diffusion & Large Vision Models

Motivation

Forward / backwards process

Variational formulation

Training

Inference

Faster sampling

Limitations

Generation process $p_\theta(x_{t-1} \mid x_t)$ $T \gg 1$

hojonathanho/diffusion, official GitHub repository of the DDPM paper, 2020.

Limitations

Generation process $p_\theta(x_{t-1} \mid x_t)$ $T \gg 1$

Orders of magnitude

- $O(T)$ more expensive than VAE/GAN
- $T = 1000 \rightarrow$ possibly spend ~minutes per sample

hojonathanho/diffusion, official GitHub repository of the DDPM paper, 2020.

Limitations

Generation process $p_\theta(x_{t-1} \mid x_t)$ $T \gg 1$

Orders of magnitude

- $O(T)$ more expensive than VAE/GAN
- $T = 1000 \rightarrow$ possibly spend ~minutes per sample

In practice, DDPM is too slow at inference time!

Mitigation attempts

Starting point. Recall iterative generative formula between two steps:

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t, t)\right) + \sigma_t z$$

Predicted noise to remove

Sampled from a Normal distribution

Mitigation attempts

Attempt. What if we derive a formula by induction?

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t, t)\right) + \sigma_t z$$

$$x_{t-2} = \frac{1}{\sqrt{\alpha_{t-1}}}\left(x_{t-1} - \frac{1-\alpha_{t-1}}{\sqrt{1-\bar{\alpha}_{t-1}}}\epsilon_\theta(x_{t-1}, t-1)\right) + \sigma_{t-1} z$$

$$\vdots$$

$$x_{\tau_{i-1}} = Ax_{\tau_i} + \sum_{s=\tau_{i-1}+1}^{\tau_i} B_s \epsilon_\theta(x_s, s) + C_s z_s$$

Mitigation attempts

Attempt. What if we derive a formula by induction?

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t, t)\right) + \sigma_t z$$

$$x_{t-2} = \frac{1}{\sqrt{\alpha_{t-1}}}\left(x_{t-1} - \frac{1-\alpha_{t-1}}{\sqrt{1-\bar{\alpha}_{t-1}}}\epsilon_\theta(x_{t-1}, t-1)\right) + \sigma_{t-1} z$$

$$\vdots$$

$$x_{\tau_{i-1}} = Ax_{\tau_i} + \sum_{s=\tau_{i-1}+1}^{\tau_i} B_s \epsilon_\theta(x_s, s) + C_s z_s$$

Problem: Still need to evaluate function several times. No progress.

Mitigation attempts

Attempt 2. What if we directly skip steps?

$$\tau_0 = 0 < \tau_1 < \ldots < \tau_{S-1} < \tau_S = T$$

$$x_{\tau_{i-1}} = \sqrt{\frac{\bar{\alpha}_{\tau_{i-1}}}{\bar{\alpha}_{\tau_i}}} \left(x_{\tau_i} - \frac{1 - \bar{\alpha}_{\tau_i} / \bar{\alpha}_{\tau_{i-1}}}{\sqrt{1 - \bar{\alpha}_{\tau_i}}} \epsilon_\theta(x_{\tau_i}, \tau_i) \right) + \sigma_{\tau_i} z$$

Denoising Diffusion Implicit Models, Song et al., 2020.

Mitigation attempts

Attempt 2. What if we directly skip steps?

$$\tau_0 = 0 < \tau_1 < \ldots < \tau_{S-1} < \tau_S = T$$

$$x_{\tau_{i-1}} = \sqrt{\frac{\bar{\alpha}_{\tau_{i-1}}}{\bar{\alpha}_{\tau_i}}} \left(x_{\tau_i} - \frac{1 - \bar{\alpha}_{\tau_i} / \bar{\alpha}_{\tau_{i-1}}}{\sqrt{1 - \bar{\alpha}_{\tau_i}}} \epsilon_\theta(x_{\tau_i}, \tau_i) \right) + \boxed{\sigma_{\tau_i} z}$$

Problem: mix large jumps + stochasticity = bad quality

Problem reformulation

Follow-up idea. How to remove stochasticity while making these larger jumps?

Key property. Loss function only relies on marginals $q(x_t \mid x_0)$

Problem statement. Find q such that:

- marginal is the same as in DDPM
- generation process can be made deterministic

Problem reformulation

$$q_\sigma(x_t \mid x_0) = q(x_t \mid x_0)$$

$$q_\sigma(x_{t-1} \mid x_t, x_0)$$

$$p_\theta(x_{t-1} \mid x_t)$$ can be made deterministic

$$q_\sigma(x_{t-1} \mid x_t, x_0) = \mathcal{N}\left(\sqrt{\bar{\alpha}_{t-1}}\, x_0 + \sqrt{1 - \bar{\alpha}_{t-1} - \sigma_t^2}\, \frac{x_t - \sqrt{\bar{\alpha}_t}\, x_0}{\sqrt{1 - \bar{\alpha}_t}},\ \sigma_t^2 I\right)$$

Problem reformulation

$$q_\sigma(x_{t-1} \mid x_t, x_0)$$

$$q_\sigma(x_t \mid x_0) = q(x_t \mid x_0)$$

Justification: proof by induction

$$p_\theta(x_{t-1} \mid x_t)$$ can be made deterministic

$$q_\sigma(x_{t-1} \mid x_t, x_0) = \mathcal{N}\left(\sqrt{\bar{\alpha}_{t-1}}\, x_0 + \sqrt{1 - \bar{\alpha}_{t-1} - \sigma_t^2}\, \frac{x_t - \sqrt{\bar{\alpha}_t}\, x_0}{\sqrt{1 - \bar{\alpha}_t}},\ \sigma_t^2 I\right)$$

Problem reformulation

$q_\sigma(x_{t-1} \mid x_t, x_0)$

$q_\sigma(x_t \mid x_0) = q(x_t \mid x_0)$

Justification: proof by induction

$p_\theta(x_{t-1} \mid x_t)$ can be made deterministic

$$p_\theta(x_{t-1} \mid x_t) \triangleq q_{\sigma=0}(x_{t-1} \mid x_t, \hat{x}_0(x_t))$$

$$q_\sigma(x_{t-1} \mid x_t, x_0) = \mathcal{N}\left(\sqrt{\bar{\alpha}_{t-1}}\, x_0 + \sqrt{1 - \bar{\alpha}_{t-1} - \sigma_t^2}\, \frac{x_t - \sqrt{\bar{\alpha}_t}\, x_0}{\sqrt{1 - \bar{\alpha}_t}}, \sigma_t^2 I\right)$$

Why is DDIM called that way?

DDIM = **D**enoising **D**iffusion **I**mplicit **M**odels

$$\sigma_t = 0$$

$$\hat{x}_0 = \frac{x_t - \sqrt{1-\bar{\alpha}_t}\,\epsilon_\theta(x_t, t)}{\sqrt{\bar{\alpha}_t}}$$

"our prediction of the clean image from time t"

Derivation: use the marginal distribution formula

Why is DDIM called that way?

DDIM = **D**enoising **D**iffusion **I**mplicit **M**odels

$$\sigma_t = 0$$

$$\hat{x}_0 = \frac{x_t - \sqrt{1-\bar{\alpha}_t}\,\epsilon_\theta(x_t, t)}{\sqrt{\bar{\alpha}_t}}$$

$$x_{t-1} = \sqrt{\bar{\alpha}_{t-1}}\,\hat{x}_0 + \sqrt{1-\bar{\alpha}_{t-1}}\,\epsilon_\theta(x_t, t) \quad + \quad \varnothing$$

"Our prediction of the clean image from time t"

"...that we noise to time t-1..."

... with no stochasticity!

Why is the DDIM formulation useful?

Update rule

$$x_{t-1} = \sqrt{\bar{\alpha}_{t-1}}\,\hat{x}_0(t) + \sqrt{1-\bar{\alpha}_{t-1}}\,\epsilon_\theta(x_t, t)$$

Why is the DDIM formulation useful?

Update rule: accelerated!

$$x_{\tau_{i-1}} = \sqrt{\bar{\alpha}_{\tau_{i-1}}}\,\hat{x}_0(\tau_i) + \sqrt{1-\bar{\alpha}_{\tau_{i-1}}}\,\epsilon_\theta(x_{\tau_i}, \tau_i)$$

x_{τ_0} $x_{\tau_{i-1}}$ x_{τ_i} x_{τ_S}

$$\tau_0 = 0 < \tau_1 < \ldots < \tau_{S-1} < \tau_S = T$$

$S \ll T$ with $\frac{T}{S} \triangleq$ "speedup"

Applications

Speedup

- Typically 10x-100x
- Need to trade off speed with quality

ermongroup/ddim, official GitHub repository of the DDIM paper, 2020.

Applications

Speedup

- Typically 10x-100x
- Need to trade off speed with quality

Quality (based on CIFAR10 experiments)

Speed-up	1x	10x	20x	50x	100x
FID impact	baseline	+3%	+16%	+70%	+330%

Denoising Diffusion Implicit Models, Song et al., 2020.

Applications

Speedup

- Typically 10x-100x
- Need to trade off speed with quality

Quality (based on CIFAR10 experiments)

Speed-up	1x	10x	20x	50x	100x
FID impact	baseline	+3%	+16%	+70%	+330%

Discussion

- Skipping steps with inter-steps stochasticity → much worse quality
- Schedule for $\{\tau_i\}$ can be chosen as a hyperparameter

Denoising Diffusion Implicit Models, Song et al., 2020.

DDIM conclusion

Recipe

1. Choose "convenient" modeling assumptions compatible with DDPM optimization
2. Remove inter-step stochasticity = DDIM
3. Skip steps!

$$\boxed{p_\theta(x_{t-1} \mid x_t) \longrightarrow p_\theta(x_{\tau_i - 1} \mid x_{\tau_i})}$$

Thank you for your attention!