Image Generation with VAEs, GANs, and Diffusion Models

Applied Deep Learning — Chapter 22

Yin Yang

Foundation Books Project

Roadmap

From Segmentation to Generation

Generative Modeling Vocabulary

Autoencoders and VAEs

Generative Adversarial Networks

Diffusion Models

Latent Diffusion and Text-to-Image

Fine-Tuning Diffusion Models

Evaluating Generated Images

Practical Workflow

Summary

From Segmentation to Generation

The Direction of Attention Reverses

Earlier chapters: image in, structured answer out (class, boxes, masks).

Image generation: image in for training, but the question becomes

- produce a new image,
- reconstruct an existing one, or
- edit an image under a condition (label, prompt, mask).

The output is a candidate image, so the sampling recipe and evaluation evidence matter as much as the network name.

Image Tensors and Dataset Scale

Running example: 32×32 RGB color images.

$$x \in \mathbb{R}^{32\times 32\times 3}, \quad \text{one image} = 32 \cdot 32 \cdot 3 = 3072 \text{ scalars.}$$

A 50,000-image dataset:

$$50{,}000 \cdot 3072 = 153{,}600{,}000 \text{ scalar pixel values.}$$

"Small" image data quickly becomes large enough that direct memorization is the wrong goal — but overparameterized models can still memorize, so evidence of held-out behavior, diversity, and useful patterns is required.

Unconditional vs Conditional

Unconditional Training set $\mathcal{D} = \{x_i\}_{i=1}^n$. Model samples images with no explicit input condition.

Conditional Training set $\mathcal{D} = \{(c_i, x_i)\}_{i=1}^n$. c_i is a class label, caption, source image, or segmentation mask.

A segmentation mask can itself become a generation condition: segmentation asks "which pixels are road?" — generation asks "can a plausible image be produced under this layout?"

Prediction task vs sampling task.

Three Forms of Evidence

1. Fixed examples make reconstruction comparable across epochs.
2. Fixed seeds or prompts make samples comparable across model settings.
3. Failure cases prevent a report from becoming a gallery of best samples.

A small number of beautiful outputs can hide weak coverage of the training distribution.

Generative Modeling Vocabulary

Distributions, Not Single Outputs

$p_{\text{data}}(x)$: the real but unknown distribution over training images.

$p_\theta(x)$: a learned model distribution with parameters θ.

Goal: samples from p_θ are useful approximations to samples from p_{data}.

Memorization vs generation, in 2D. Training points near $(-2, 0)$ and $(2, 0)$:

- generator returning $(-2, 0)$ alone: half of the data ignored;
- generator returning points near both clusters with variation: represents the data distribution.

Latent Variables, Priors, Likelihood

Latent $z \in \mathbb{R}^d$: unobserved hidden factors of variation. **Prior** $p(z)$: where z is sampled from before decoding. Common choice: $\mathcal{N}(0, I)$.

Sample $z \sim \mathcal{N}(0, I_2)$:

- $(0.2, -0.5)$: ordinary, near the prior's high-density region.
- $(5, 5)$: very unlikely, five σ from the mean.

Likelihood $p_\theta(x)$ or $p_\theta(x \mid c)$. Not every useful generator gives a tractable likelihood; this drives differences between VAEs, GANs, and diffusion models in evaluation, debugging, and speed.

Fidelity and Diversity

Fidelity How realistic or domain-plausible generated images look.

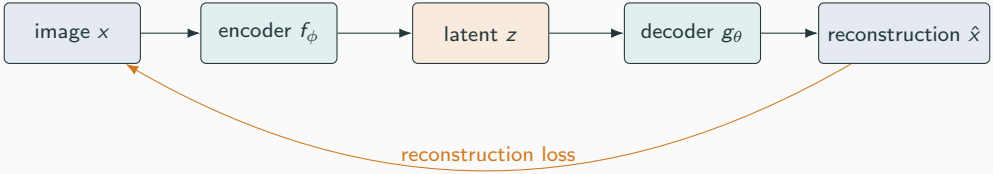
Diversity How well generated images cover useful variation in the target domain.

A generator with high fidelity and low diversity may produce sharp images that all look nearly the same.

A report should contain more than one image: fixed prompts, multiple seeds, failure examples, and time and memory records.

Autoencoders and VAEs

Reconstruction as an Entry Point



$32 \times 32 \times 3$ image, 64-dim latent: encoder compresses 3072 numbers into 64; decoder reconstructs from the bottleneck.

Reconstruction Loss

Mean squared error on normalized pixels:

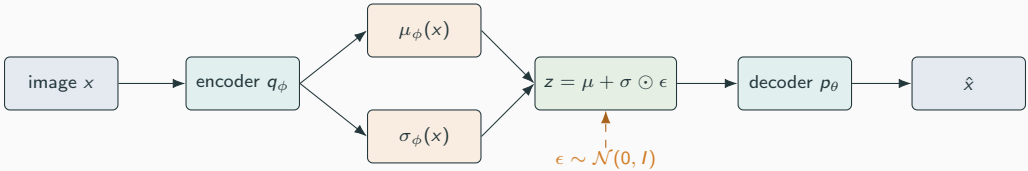
$$\mathrm{MSE}(x, \hat{x}) = \frac{1}{d} \sum_{j=1}^{d} (x_j - \hat{x}_j)^2.$$

Worked example, $x = (0, 1, 1, 0)$, $\hat{x} = (0.1, 0.8, 0.6, 0.2)$:

$$\mathrm{MSE} = \frac{0.01 + 0.04 + 0.16 + 0.04}{4} = 0.0625.$$

A plain autoencoder reconstructs well but is a poor random generator: it has only learned to decode codes from real images, not arbitrary samples from a simple prior.

The Variational Autoencoder



ELBO: $\log p_\theta(x) \geq \mathbb{E}_{q_\phi}[\log p_\theta(x \mid z)] - \mathrm{KL}(q_\phi(z \mid x) \,\|\, p(z))$. Reparameterization makes z differentiable in μ_ϕ, σ_ϕ.

The KL Term in One Dimension

$q(z \mid x) = \mathcal{N}(\mu, 1)$, $p(z) = \mathcal{N}(0, 1)$:

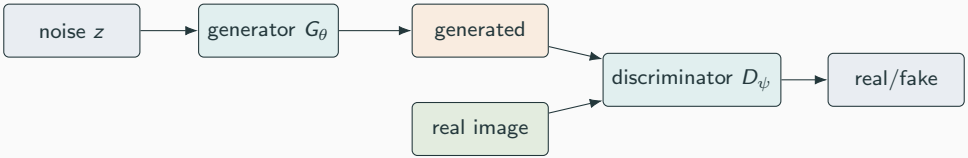
$$\mathrm{KL}(q \| p) = \frac{\mu^2}{2}.$$

- $\mu = 0 \Rightarrow \mathrm{KL} = 0$, encoder matches prior.
- $\mu = 1 \Rightarrow \mathrm{KL} = 0.5$, latent moved to help reconstruction.

KL shapes the latent space. The VAE pays reconstruction cost and prior-matching cost simultaneously — so VAE samples are often smoother and blurrier than purely reconstruction-driven autoencoders.

Generative Adversarial Networks

Generator vs Discriminator



$D_\psi(x) \in [0,1]$: probability that x is a real training image.

Minimax objective:

$$\min_\theta \max_\psi \mathbb{E}_{x\sim p_{\text{data}}}[\log D_\psi(x)] + \mathbb{E}_{z\sim p(z)}[\log(1 - D_\psi(G_\theta(z)))].$$

GAN Loss Curves Are Different

Suppose $D(x_{\text{real}}) = 0.8$, $D(G(z)) = 0.3$.

- Real term: $-\log 0.8 \approx 0.223$.
- Fake term: $-\log(1 - 0.3) = -\log 0.7 \approx 0.357$.

Generator improves, $D(G(z))$ rises to 0.6:

$$-\log(1 - 0.6) = -\log 0.4 \approx 0.916.$$

Worse for D, better evidence the generator is challenging it.

GAN loss curves cannot be read like ordinary supervised validation curves.

Mode Collapse

Generator produces too little variety. Samples may look sharp, but many are nearly identical or cover only part of the distribution.

Diagnostics:

- Fixed latent vectors to track per-vector progress.
- Many random latents to check diversity.
- Inspect both networks — overstrong D starves G; weak D gives little signal.

A pretty sample is not evidence if the next 63 samples repeat the same pattern under different seeds.

GAN Variants in Practice

DCGAN Convolutional generator and discriminator; made small image experiments practical.

Conditional GAN Adds class labels or other inputs.

pix2pix Paired image translation with adversarial loss.

CycleGAN Unpaired image translation with cycle consistency.

Here, GANs teach adversarial loss, implicit generation, and fidelity-diversity failures. For modern text-to-image, pretrained diffusion models are usually the more practical default.

Diffusion Models

The Forward Noising Process



$$x_t = \sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1-\bar{\alpha}_t}\, \epsilon, \quad \epsilon \sim \mathcal{N}(0, I).$$

Scalar example, $x_0 = 0.7$, $\bar{\alpha}_t = 0.64$, $\epsilon = -0.5$: $x_t = 0.8 \cdot 0.7 + 0.6 \cdot (-0.5) = 0.26$. Larger $t \Rightarrow$ smaller $\bar{\alpha}_t \Rightarrow$ more noise replaces signal.

Noise Prediction Objective

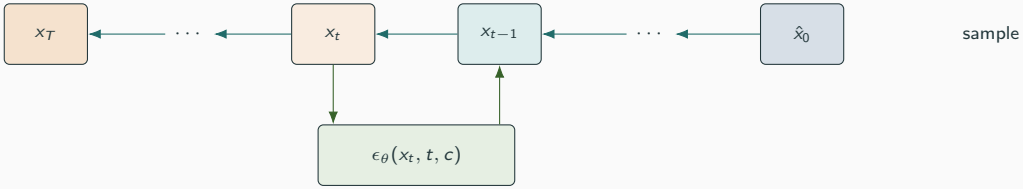
Train a network $\epsilon_\theta(x_t, t, c)$ to predict the noise:

$$\mathcal{L}_{\text{simple}}(\theta) = \mathbb{E}_{x_0,t,\epsilon}\left[\, \|\epsilon - \epsilon_\theta(x_t, t, c)\|_2^2 \,\right].$$

This is supervised learning: the training code sampled ϵ, so the target is known.

Sampling later starts from pure noise and uses the same network in reverse — many small denoising steps refine a noisy state into an image.

The Reverse (Denoising) Step



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

Sampling is iterative. Step count is part of model behavior, cost, and comparison protocol.

Cost: GAN vs Diffusion Sampling

- GAN generator: one forward pass per sample.
- Diffusion sampler: 20, 30, 50, or more denoising steps.

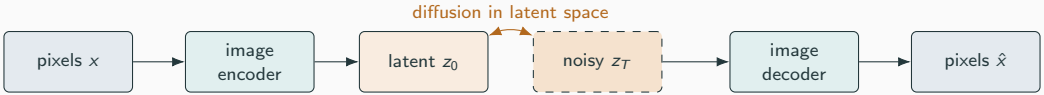
In an applied report, the number of inference steps is not cosmetic — it changes:

- time per image,
- prompt adherence,
- artifact patterns,
- diversity.

Start small: a tiny denoising model on 32×32 images shows the forward equation, the loss, and the reverse loop without pretending to be a modern text-to-image system.

Latent Diffusion and Text-to-Image

Why Latent Diffusion?



Pixel-space diffusion is expensive at high resolution. **Latent diffusion** runs every denoising step on a compressed code, then decodes once to pixels — same recipe, smaller representation, lower cost per step.

Text Conditioning via Cross-Attention

Text prompt $\rightarrow$ text encoder $\rightarrow$ embeddings.

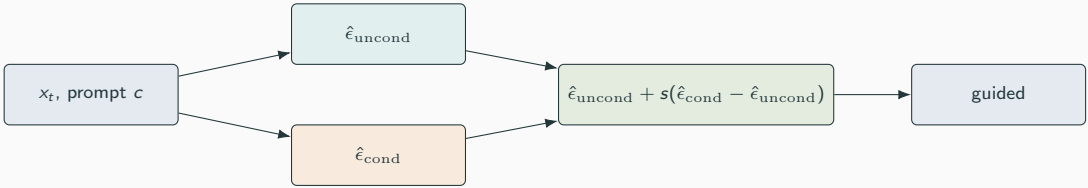
Latent-image features query text-encoder features:

$$Q = H_z W_Q, \quad K = H_t W_K, \quad V = H_t W_V,$$

$$A = \mathrm{softmax}\left(\frac{QK^\top}{\sqrt{d_h}}\right), \quad \mathrm{CrossAttn}(H_z, H_t) = AV.$$

At sampling time, different prompts steer the denoising trajectory. This is not language understanding — training pairs made the denoiser sensitive to text embeddings.

Classifier-Free Guidance



Scalar example: uncond $= 0.2$, cond $= 0.5$, $s = 3 \Rightarrow 0.2 + 3(0.3) = 1.1$. Guided prediction has moved *beyond* the conditional. Larger s strengthens prompt adherence but can reduce diversity — guidance scale is an experimental knob, not a quality dial.

Comparing Two Text-to-Image Runs

Same prompt, different runs — many things can change the output:

- random seed,
- guidance scale,
- scheduler and step count,
- model checkpoint or revision,
- image size.

A comparison must fix these settings or record exactly how they differ.

Otherwise the report cannot tell whether an improvement came from fine-tuning, a changed prompt, a different seed, or a slower sampler.

Fine-Tuning Diffusion Models

Why Fine-Tune at All?

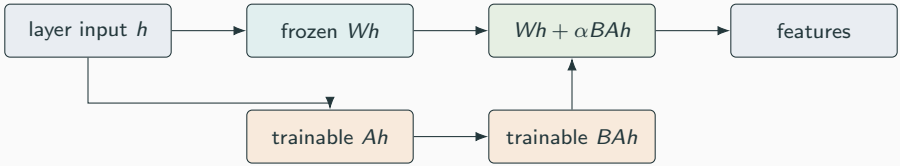
Training a high-quality diffusion model from scratch needs large datasets, large compute, and substantial engineering.

Fine-tuning starts from a model that already contains broad visual knowledge and adapts a smaller part for the new dataset.

Targets: a product concept, a simple style, a domain-specific object, or a controlled licensed collection.

Data quality matters first. Duplicates, inconsistent crops, and vague captions corrupt the result before the optimizer matters.

LoRA Adapters in Diffusion



Rank-8 adapter on a 768×768 projection: $8 \cdot 768 + 768 \cdot 8 = 12{,}288$ params $\approx 2.08\%$ of the original.

The Adapter Map

Practical questions for diffusion fine-tuning:

- **Where** are adapters placed — cross-attention projections, text-conditioning paths, selected denoising blocks?
- **What is frozen?**
- Do validation prompts show useful adaptation, or memorization?

Personalization variants:

DreamBooth adapt to a small image set under a special token.

Textual inversion learn a new embedding rather than weight updates.

Before-and-After Discipline

Baseline run. Pretrained model on a fixed validation prompt set, fixed seeds, fixed sampler settings. Answers: what does the model already know?

Fine-tuning run. Recorded recipe: model id, dataset source, image resolution, caption policy, optimizer, learning rate, batch size, gradient accumulation, precision, steps, LoRA rank, checkpoint interval, hardware.

Comparison. Re-run the same validation prompts and seeds.

Only after a baseline is the before-and-after comparison meaningful.

Evaluating Generated Images

Reconstruction Metrics Are Not Enough

When a target image exists (autoencoder validation), MSE works:

$$\mathrm{MSE}(x, \hat{x}) = \frac{1}{d} \sum_{j=1}^{d} (x_j - \hat{x}_j)^2.$$

For open-ended generation, the prompt "a red cup on a desk" has many valid images. Pixelwise agreement with one target is the wrong standard.

Evaluation must instead ask whether the image:

- follows the prompt;
- is plausible;
- supports useful variation across many samples;
- is not a copy of training data.

Fréchet Inception Distance

Compare distributions of real and generated images in a feature space:

$$\mathrm{FID} = \|\mu_r - \mu_g\|_2^2 + \mathrm{Tr}\Big(\Sigma_r + \Sigma_g - 2(\Sigma_r\Sigma_g)^{1/2}\Big).$$

Lower FID indicates closer feature-distribution match.

One-dimensional version with $\mu_r = 1.0$, $\sigma_r = 2.0$, $\mu_g = 1.5$, $\sigma_g = 1.0$:

$$(1.0 - 1.5)^2 + 4 + 1 - 2 \cdot 2 \cdot 1 = 0.25 + 4 + 1 - 4 = 1.25.$$

FID inherits assumptions: feature network, sample size, and the match between feature space and the actual domain.

Memorization

When the dataset is small and training is long, fine-tuned diffusion can reproduce training examples too closely.

Mitigations:

- Inspect near-duplicates between training images and samples.
- Use validation prompts that ask for the concept in new contexts.
- Save sample grids over time — a checkpoint that improves at step 400 may memorize at step 1200.

If the model only works under one memorized caption, the run has not produced a robust adapted generator.

An Evidence Record

Reference run: tiny prompt-conditioned DDPM on 32×32 procedural shapes, rank-8 LoRA fine-tune.

Field	Value
Training time	191.1 s full reference; 149.8 s before rank-ablations
Peak memory	405 MB PyTorch allocation
Inference / image	0.0079 s for 50 denoising steps
Fixed val prompts	8 held prompts including the target concept
Failure grid	Always included in the report

Trust a plain grid with repeated seeds and named failures over a polished single image whose prompt, seed, and checkpoint are undocumented.

Practical Workflow

Conservative First Choices

Decision	Conservative choice	What to inspect
Image resolution	fits comfortably in memory	need more detail later?
Batch size	largest stable, or use accumulation	OOM, throughput, stability
Learning rate	documented script default	smooth change vs degradation
Training steps	short with intermediate samples	first useful, first overfit
Guidance / steps	fixed validation setting	time and prompt adherence

Confounded Ablations

Bad comparison: rank 4 at 400 steps vs rank 8 at 800 steps. Conclusion "rank 8 is better" confounds rank with step count.

Better: fix dataset, prompts, resolution, learning rate, steps; vary one factor.

- Rank 4 vs rank 8 with steps fixed.
- 400 vs 800 steps with rank fixed.

One controlled ablation is more useful than many unrecorded changes.

The Result Claim Should Be Modest

Under the fixed prompt set and seed policy, did the adapted model:

- represent the target concept more consistently,
- preserve useful variation,
- avoid inspected near-duplicates,
- fit the reported time and memory budget?

If the evidence is weaker, the conclusion should be weaker.

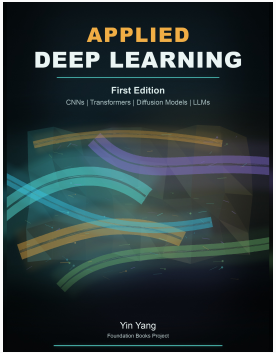
Applied image generation is not about the best image; it is about the best defensible recipe for the stated purpose.

Summary

Summary

- **Image generation models a distribution,** not a single output. Fidelity and diversity must be reported together.
- **Autoencoders and VAEs** introduce latents through reconstruction; KL shapes the latent space at the cost of blurriness.
- **GANs** train generator vs discriminator; sharp samples can hide mode collapse.
- **Diffusion** models reverse a noising process; sampling is iterative, and step count is a real cost.
- **Latent diffusion + cross-attention** make text-to-image practical; classifier-free guidance is a tradeoff knob.
- **Pretrained + LoRA** is the usual applied default.
- **Evaluation** needs fixed prompts, seeds, baselines, failure cases, and recorded cost.

Next: automatic speech recognition — waveforms and time-indexed features replace image tensors and sample grids, but the evidence discipline carries forward.



Applied Deep Learning

Chapter overview slides for the book by Yin Yang.

Book page	`foundation-books.github.io/applied-deep-learning`
Code	`github.com/foundation-books/applied-deep-learning-code`
Print	`amazon.com/dp/B0GZF7QRSH`
Kindle	`amazon.com/dp/B0GZF9221X`
License	CC BY-NC-ND 4.0

These free overview slides may be shared non-commercially with attribution and without modifications. Full explanations, exercises, and companion-code context are in the book and linked repository.