Generative Adverserial Networks

Vinay Setty
vinay.j.setty@uis.no

University of Stavanger

Information Access & Artificial Intelligence

Department of Electrical Engineering and Computer Science
University of Stavanger

January 16, 2026

Overview

Introduction

VAE to GANs

GAN Training

GAN Variations

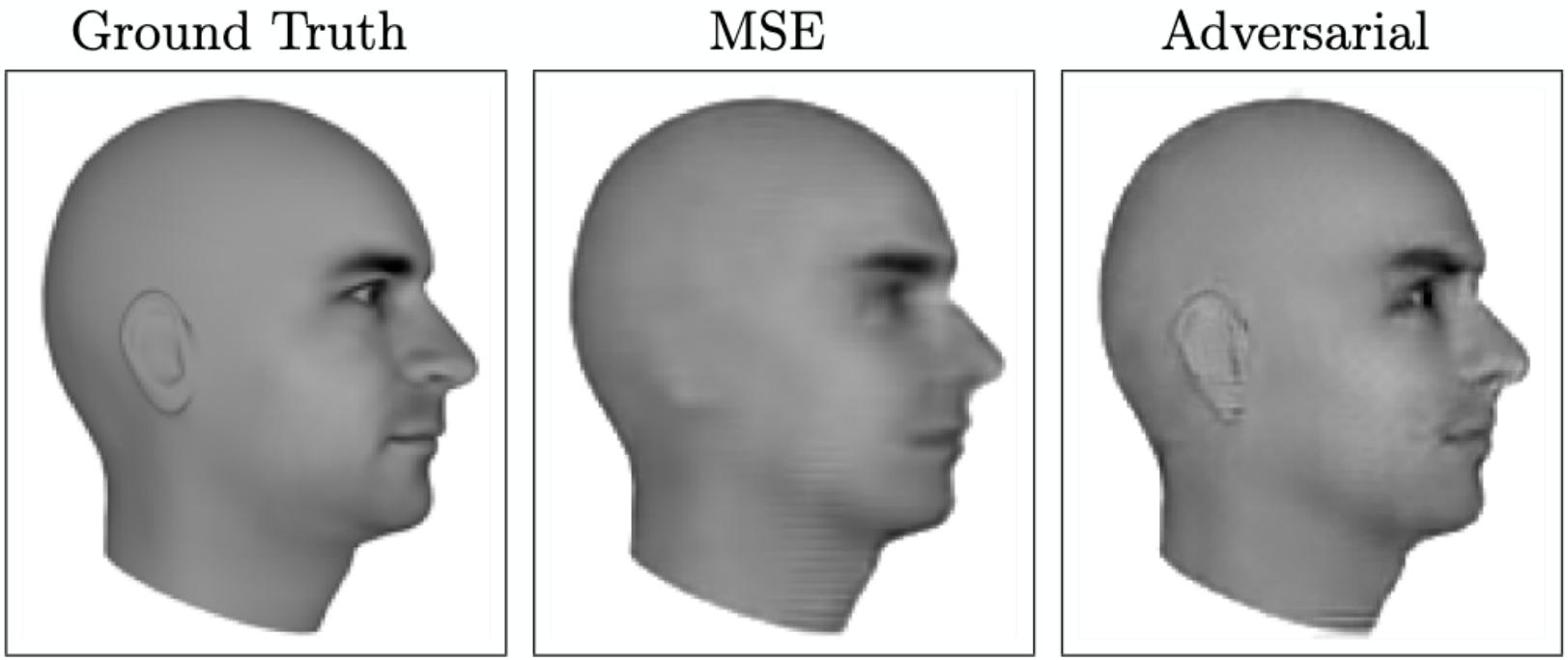
GAN intuition

Why study GANs?

- Excellent test of our ability to use high-dimensional, complicated probability distributions
- Simulate possible futures for planning or simulated RL
- Missing data: Semi-supervised learning
- Multi-modal outputs
- Realistic generation tasks

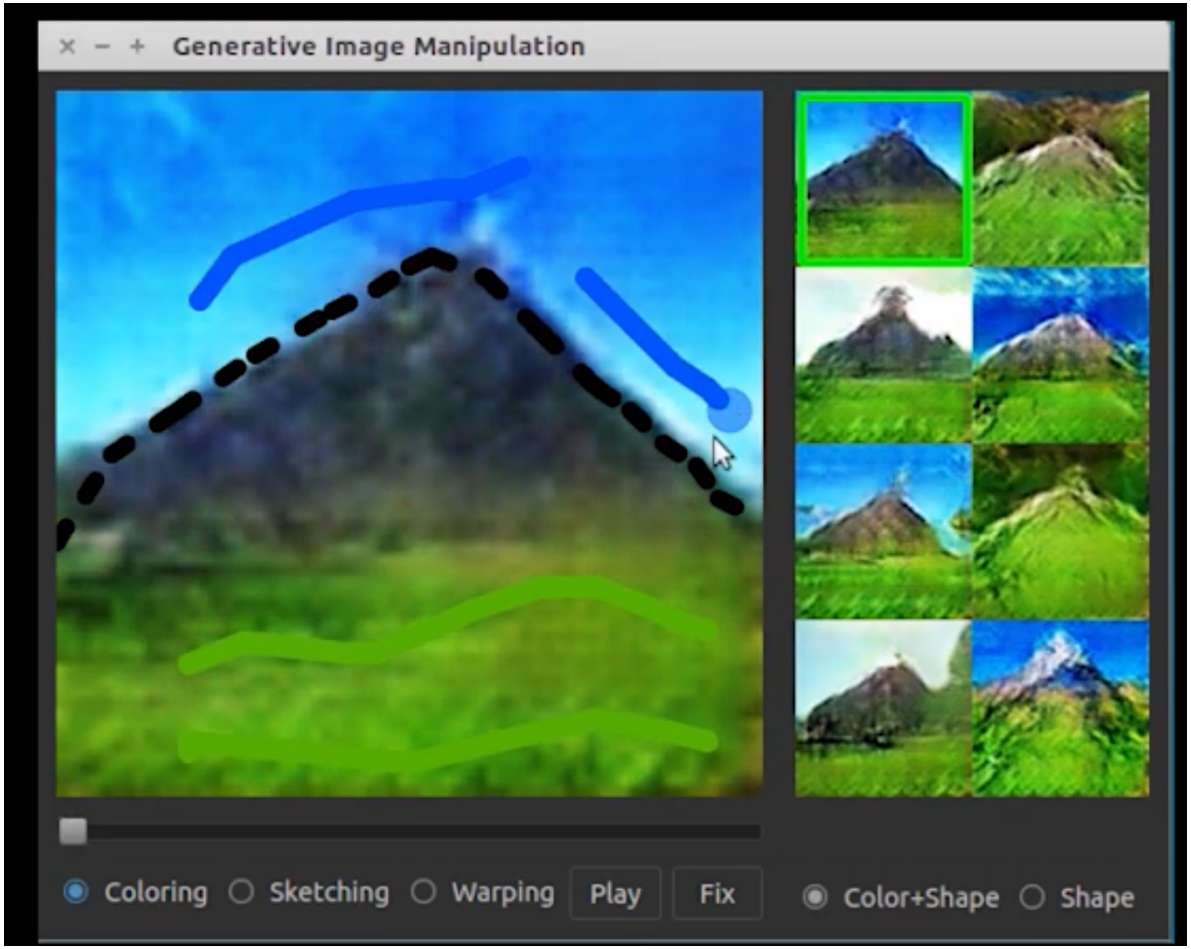
Why study GANs? (cont.)

Sharper edges and features when predicting next frame.



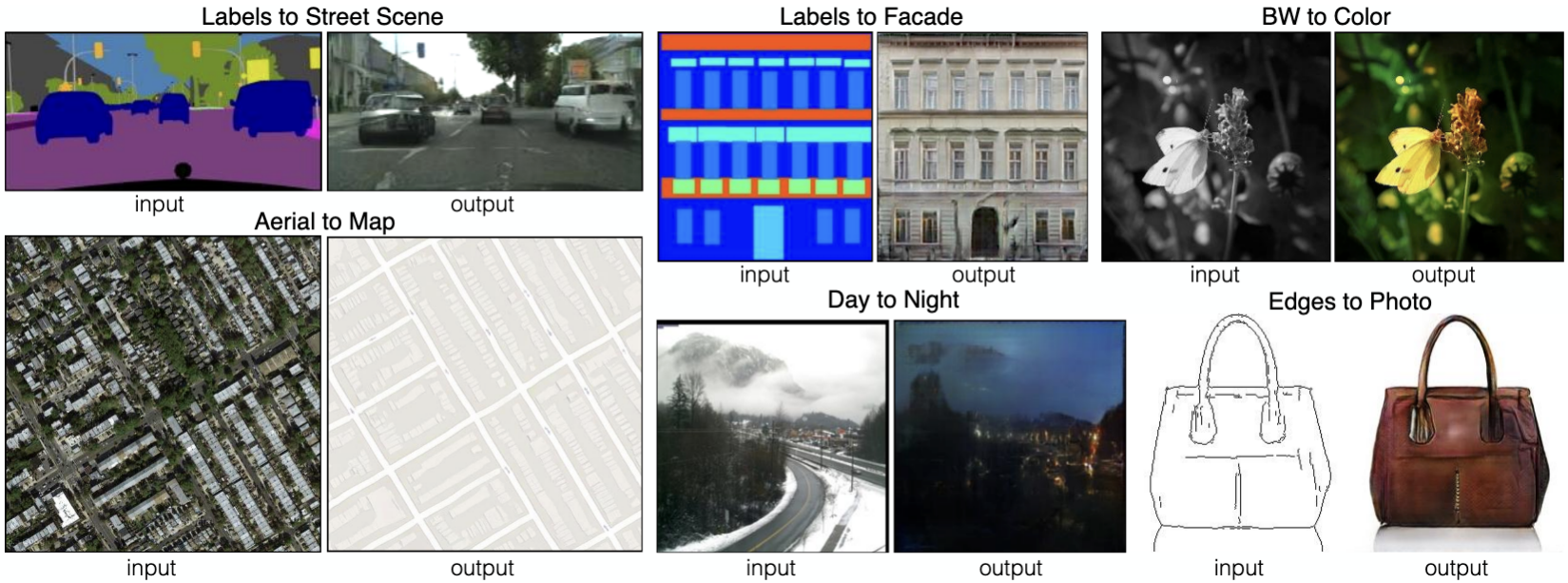
Why study GANs? (cont.)

Generative image manipulation.



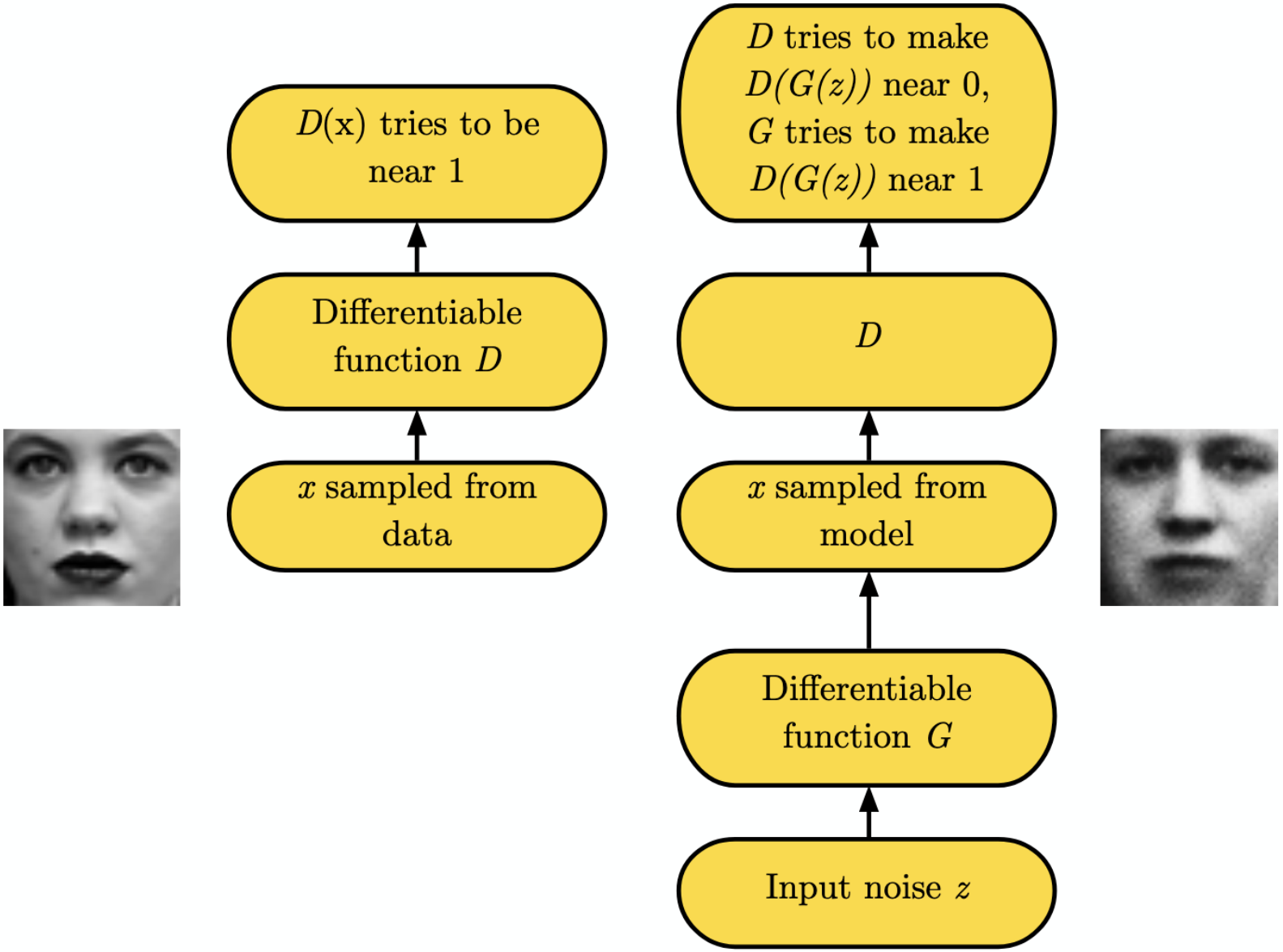
https://youtu.be/9c4z6YsBGQ0?t=209

Image Translation



Source: P. Isola et al. (2017). "Image-to-image translation with conditional adversarial networks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1125–1134

GAN architecture



Latent Variable Models and the Likelihood Bottleneck

Latent variable models define

$$p_\theta(x) = \int p_\theta(x \mid z) p(z)\, dz$$

Training requires marginalization over z, which is intractable in general.

Monte Carlo approximation:

$$\log p_\theta(x) \approx \log \frac{1}{S} \sum_{s=1}^{S} p_\theta(x \mid z_s) = \mathrm{LogSumExp}_s\big(\log p_\theta(x \mid z_s)\big) - \log S$$

Issues:

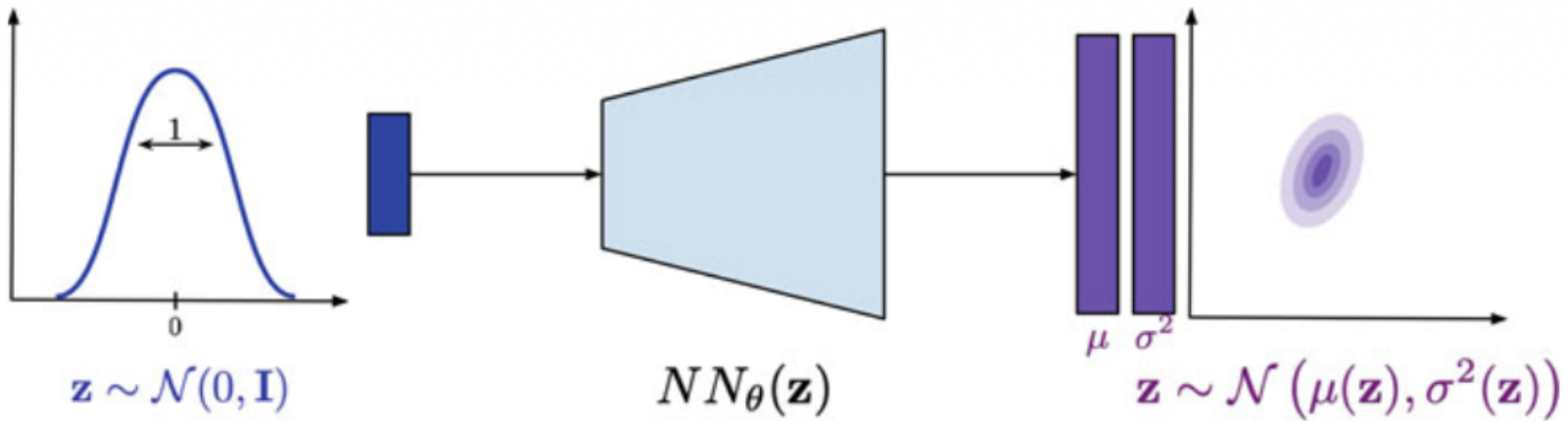
- High variance estimators
- Poor scaling in high dimensions
- Strong dependence on prescribed likelihoods

Why VAE produces blurry images

- Not asymptotically consistent unless q is perfect.
- Samples tend to have lower quality.

Upper row is VAE and lower row is GAN.

Density Networks



Density Networks and Their Limitations

Density networks are explicit probabilistic generative models that require all distributions to be specified analytically.

Assumptions

- The prior distribution $p(z)$ is predefined, typically a standard Gaussian.
- The conditional likelihood $p(x \mid z)$ is specified in advance, commonly Gaussian or a mixture of Gaussians.

Advantages

- Training objective is an approximated log-likelihood.
- Optimization is performed using gradient-based methods and automatic differentiation.
- Conditional likelihoods can be parameterized with deep neural networks.

Limitations

Limitations of VAEs and Prescribed Likelihoods

VAEs optimize a variational lower bound:

$$\log p_\theta(x) \geq \mathbb{E}_{q(z|x)}[\log p_\theta(x \mid z)] - \mathrm{KL}(q(z \mid x) \,\|\, p(z))$$

Practical limitations:

- Likelihood choice dominates perceptual quality
- Gaussian decoders induce over-smoothing
- KL term biases toward mode covering

Likelihood-based learning corresponds to

$$\min_\theta \mathrm{KL}(p_{\text{data}}(x) \,\|\, p_\theta(x))$$

a pointwise, local divergence.

From Prescribed to Implicit Models

Key idea: abandon explicit likelihoods.

Define an implicit conditional:

$$p_\theta(x \mid z) = \delta(x - G_\theta(z))$$

Marginal becomes

$$p_\theta(x) = \int \delta(x - G_\theta(z)) p(z)\, dz$$

Properties:

- Sampling is trivial
- Density evaluation is impossible
- KL divergence is undefined

Need a new way to compare distributions without $p_\theta(x)$.

Adversarial Learning Principle

Introduce a discriminator $D_\alpha(x) \in (0, 1)$.

Binary classification objective:

$$\mathcal{L}(\alpha, \beta) = \mathbb{E}_{x \sim p_{\text{data}}}[\log D_\alpha(x)] + \mathbb{E}_{z \sim p(z)}[\log(1 - D_\alpha(G_\beta(z)))]$$

Optimization:

$$\min_\beta \max_\alpha \mathcal{L}(\alpha, \beta)$$

The loss replaces explicit density comparison with a learned discrepancy.

GANs as Implicit Generative Models

GAN components:

- Generator $G_\beta : z \mapsto x$
- Discriminator $D_\alpha : x \mapsto [0, 1]$

At optimum:

$$p_G(x) = p_{\text{data}}(x)$$

Advantages:

- No likelihood specification
- High-fidelity samples
- Global distribution matching

Trade-offs:

- No tractable $p(x)$
- Training instability
- Mode collapse

Generator Network

Diagram placeholder: $z \rightarrow G \rightarrow x$.

$$x = G(z; \theta^{(G)}).$$

- Must be differentiable
- No invertibility requirement
- Trainable for any size of z
- Can make x conditionally Gaussian given z but need not do so

Training Procedure

- Use SGD-like algorithm (e.g., Adam) on two minibatches:
  - Minibatch of training examples
  - Minibatch of generated samples
- Optional: run k steps of one player for every step of the other

Minimax Game

- Equilibrium is a saddle point of discriminator loss
- Resembles Jensen–Shannon divergence
- Generator minimizes log-probability of discriminator being correct

$$J(D) = -\frac{1}{2}\mathbb{E}_{x\sim p_{\text{data}}} \log D(x) - \frac{1}{2}\mathbb{E}_z \log\big(1 - D(G(z))\big), \qquad J(G) = -J(D).$$

Non-Saturating Game

$$J(D) = -\frac{1}{2}\mathbb{E}_{x\sim p_{\text{data}}} \log D(x) - \frac{1}{2}\mathbb{E}_z \log\big(1 - D(G(z))\big),$$

$$J(G) = -\frac{1}{2}\mathbb{E}_z \log D(G(z)).$$

- Heuristic: gives stronger gradients early in training

GAN Training Procedure

Generator and discriminator architectures are simple and comparable to VAE components. The core distinction lies in the training dynamics.

Discriminator training:

- Build a dataset of real images $x \sim p_{\text{data}}$ and fake images $G(z)$
- Binary labels: real $= 1$, fake $= 0$
- Optimize binary cross-entropy:

$$\mathcal{L}_D = -\mathbb{E}_{x \sim p_{\text{data}}}[\log D(x)] - \mathbb{E}_{z \sim p(z)}[\log(1 - D(G(z)))]$$

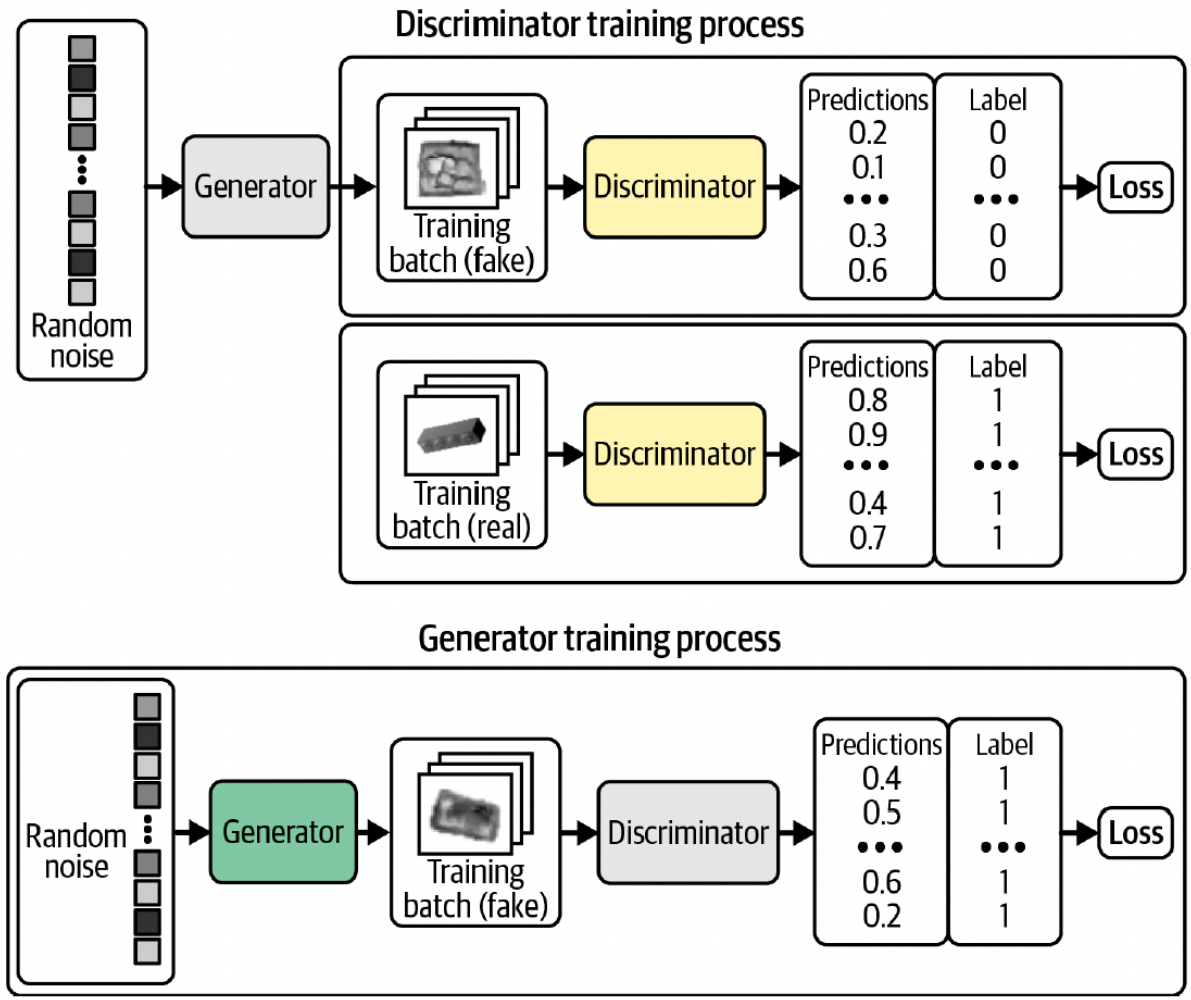
Generator training:

- Score generated samples using the discriminator
- Optimize binary cross-entropy against target label 1:

$$\mathcal{L}_G = -\mathbb{E}_{z \sim p(z)}[\log D(G(z))]$$

GAN Training Procedure

Training alternates updates, freezing one network while optimizing the other.



GAN Training Failure Modes

GAN training is unstable due to coupled optimization of generator and discriminator.

Discriminator overpowering generator:

- Near-perfect separation of real and fake samples
- Generator gradients vanish
- Training stalls completely

Generator overpowering discriminator:

- Mode collapse with low sample diversity
- Many latent inputs map to the same output
- Gradients collapse, recovery becomes unlikely

Monitoring difficulty:

- Generator loss does not correlate with sample quality

Practical Remedies and Stabilization

Weakening an overpowered discriminator:

- Increase dropout
- Reduce learning rate or model capacity
- Add noise to labels or randomly flip labels

Strengthening a weak discriminator:

- Increase capacity or update frequency
- Reduce learning rates of both networks
- Increase batch size

Remaining challenges:

- High sensitivity to hyperparameters
- No reliable training diagnostics

Motivation for modified objectives such as WGAN-GP.

From GAN Loss to Wasserstein Loss

Standard GAN (binary cross-entropy):

Discriminator minimizes

$$\mathcal{L}_D = -\mathbb{E}_{x\sim p_{\text{data}}}[\log D(x)] - \mathbb{E}_{z\sim p(z)}[\log(1 - D(G(z)))]$$

Generator minimizes

$$\mathcal{L}_G = -\mathbb{E}_{z\sim p(z)}[\log D(G(z))]$$

Wasserstein GAN: Arjovsky et al. 2017

Key changes:

- Labels $y \in \{+1, -1\}$ instead of $\{1, 0\}$
- Remove sigmoid from discriminator output
- Discriminator becomes a *critic* with $D(x) \in \mathbb{R}$

Wasserstein loss:

$$\mathcal{L}_{\text{WGAN}} = \mathbb{E}_{x\sim p_{\text{data}}}[D(x)] - \mathbb{E}_{z\sim p(z)}[D(G(z))]$$

WGAN Critic and Generator Objectives

In Wasserstein GANs, the discriminator is replaced by a *critic* that assigns real-valued scores.

Critic objective:

$$\min_D \; -\Big(\mathbb{E}_{x\sim p_{\text{data}}}[D(x)] - \mathbb{E}_{z\sim p(z)}[D(G(z))]\Big)$$

The critic maximizes the score difference between real and generated samples.

Generator objective:

$$\min_G \; -\mathbb{E}_{z\sim p(z)}[D(G(z))]$$

The generator is trained to produce samples that receive high critic scores.

No sigmoid output, no probability interpretation, only relative ordering of samples.

The Lipschitz Constraint

For the Wasserstein objective to be valid, the critic must be **1-Lipschitz continuous**.

Definition:

$$|D(x_1) - D(x_2)| \leq |x_1 - x_2| \quad \forall x_1, x_2$$

Interpretation:

- The critic output cannot change too rapidly
- Enforces smooth, meaningful scores

Without this constraint, the Wasserstein distance estimate breaks down.

Enforcing the Lipschitz Constraint

Weight clipping (original WGAN):

- Clip critic weights to a fixed range, e.g. $[-0.01, 0.01]$
- Simple but severely limits critic capacity

Gradient penalty (WGAN-GP):

$$\lambda \, \mathbb{E}_{\hat{x}} \left(\|\nabla_{\hat{x}} D(\hat{x})\|_2 - 1 \right)^2$$

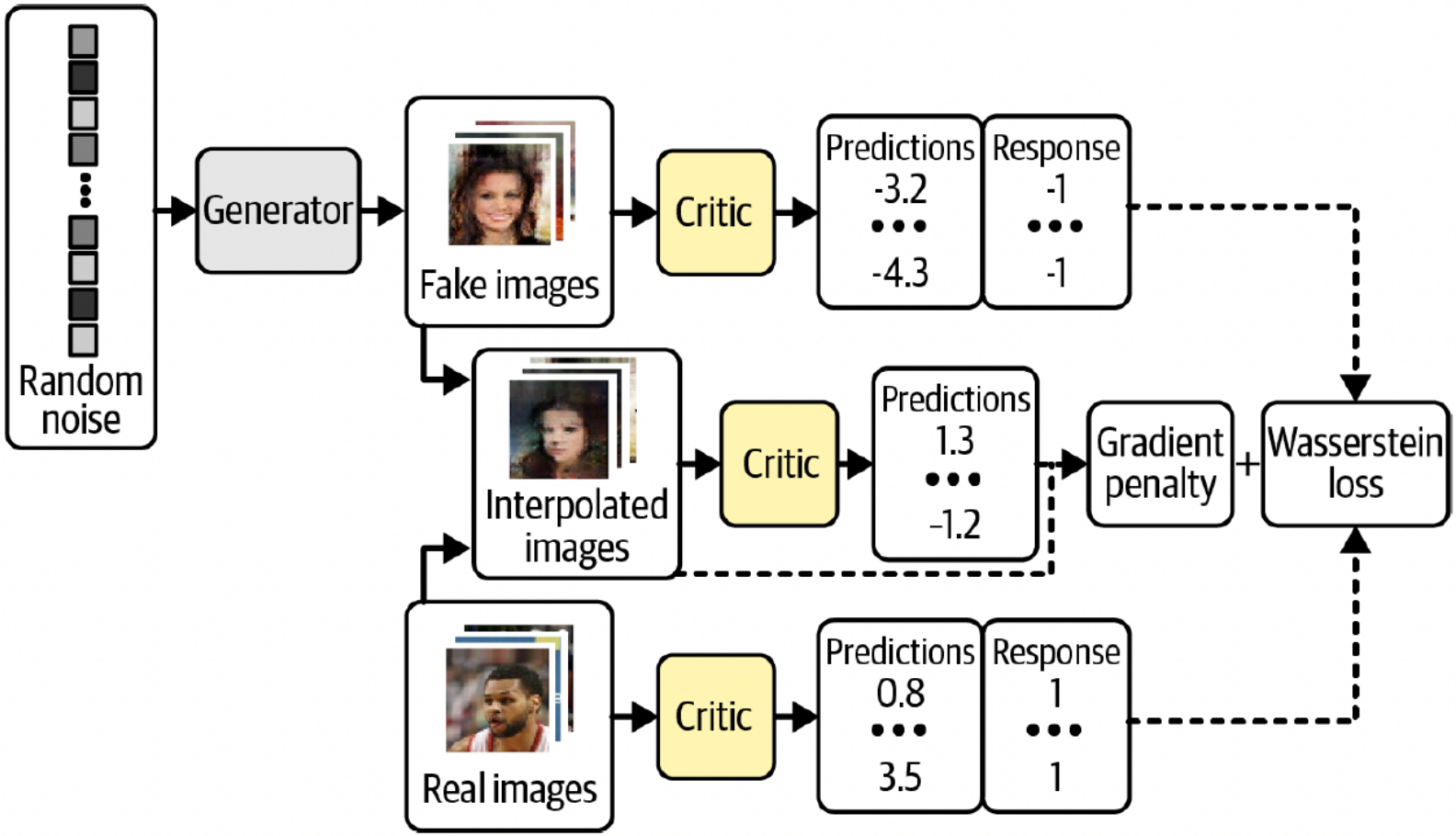
where

$$\hat{x} = \epsilon x + (1 - \epsilon) G(z), \quad \epsilon \sim \mathcal{U}(0, 1)$$

Benefits:

- Directly enforces Lipschitz condition
- Preserves critic expressiveness
- Significantly improves training stability

Gradient Penalty Loss for WGAN



Conditional GAN Architecture

Conditional GANs extend GANs by conditioning generation on label information y.

Generator input:

- Latent noise vector $z \sim p(z)$
- One-hot encoded label y

$$x = G(z, y)$$

Critic input:

- Image x (real or generated)
- Label y encoded as additional channels

$$D(x, y) \rightarrow \text{realness score}$$

Both networks explicitly depend on the conditioning variable.

How Conditioning Is Enforced

Generator conditioning:

- Label y appended to latent vector z
- Encourages generation consistent with y

Critic conditioning:

- Label y broadcast to image shape
- Added as extra input channels

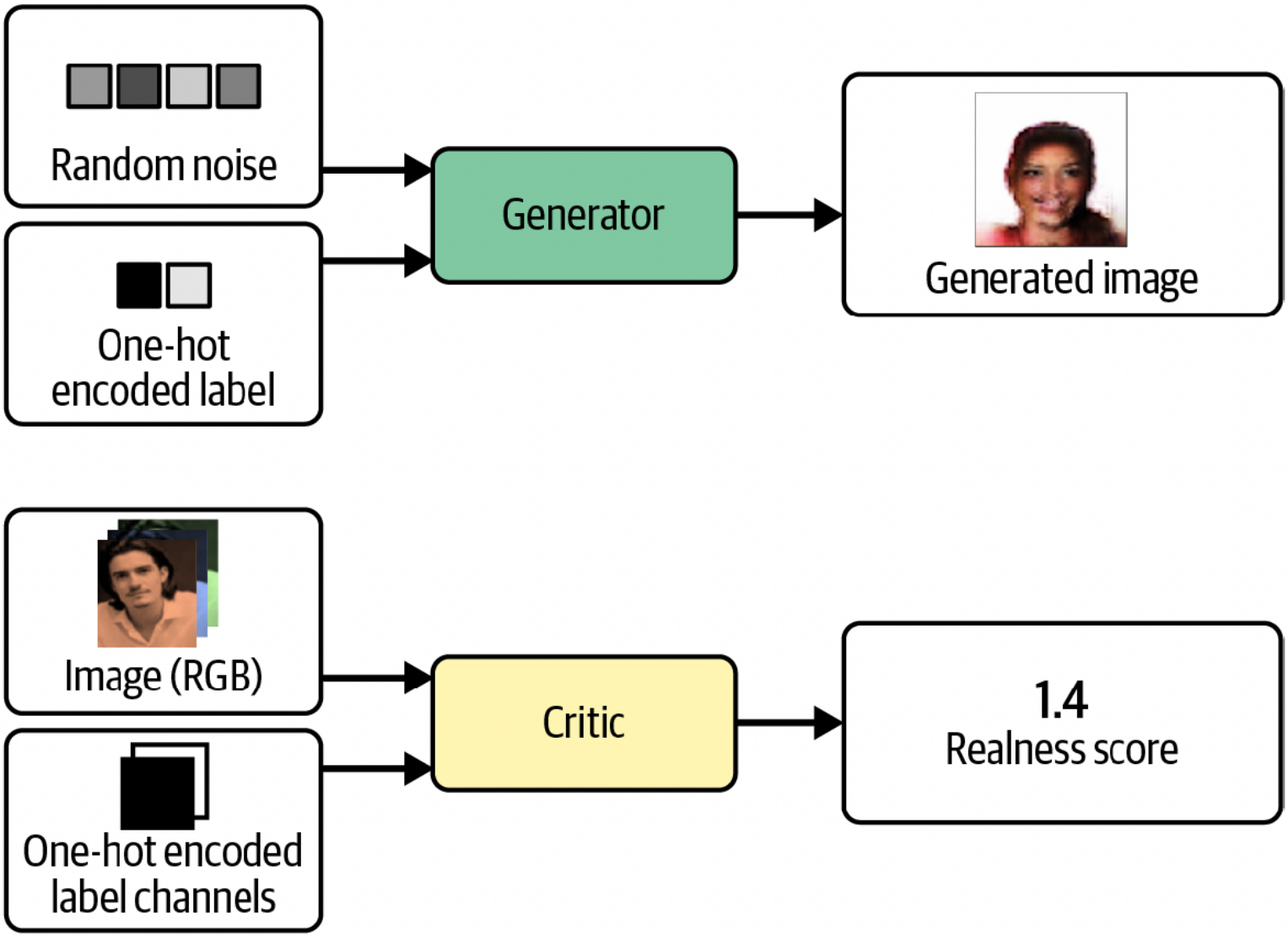
Training objective:

- Generator must fool the critic *and* match the label
- Critic checks both realism and label agreement

Result:

- Controlled generation
- Improved semantic consistency

Conditional GAN - CGAN



Conditional GAN Architecture

Conditional GANs extend GANs by conditioning generation on label information y. Mirza et al. 2014

Generator input:

- Latent noise vector $z \sim p(z)$
- One-hot encoded label y

$$x = G(z, y)$$

Critic input:

- Image x (real or generated)
- Label y encoded as additional channels

$$D(x, y) \rightarrow \text{realness score}$$

Both networks explicitly depend on the conditioning variable.

How Conditioning Is Enforced

Generator conditioning:

- Label y appended to latent vector z
- Encourages generation consistent with y

Critic conditioning:

- Label y broadcast to image shape
- Added as extra input channels

Training objective:

- Generator must fool the critic *and* match the label
- Critic checks both realism and label agreement

Result:

- Controlled generation
- Improved semantic consistency

Bibliography

Arjovsky, M. et al. (2017). *Wasserstein GAN*.

Isola, P. et al. (2017). "Image-to-image translation with conditional adversarial networks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1125–1134.

Mirza, M. et al. (2014). "Conditional generative adversarial nets". In: *arXiv preprint arXiv:1411.1784*.