Lecture 15: Generative Adversarial Networks - II

COMP 5801H/4900A: Generative AI and LLMs

2026-03-03

Sriram Subramanian
Assistant Professor & Canada Research Chair, Carleton University
Faculty Affiliate, Vector Institute for Artificial Intelligence
Faculty Affiliate, Schwartz Reisman Institute for Technology and Society

Carleton University

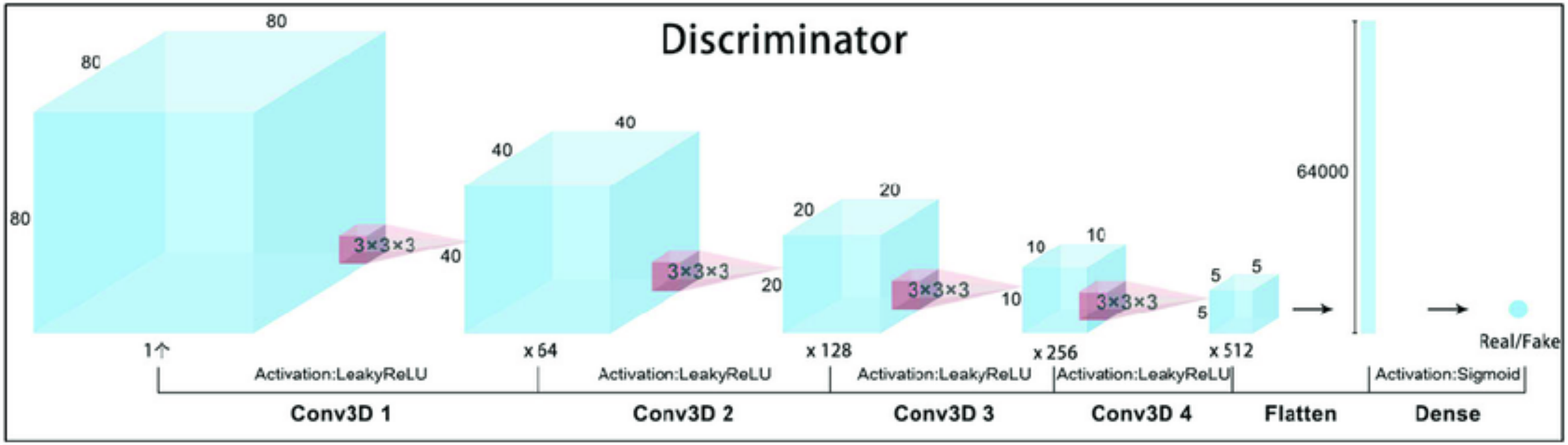
VECTOR INSTITUTE

SCHWARTZ REISMAN INSTITUTE FOR TECHNOLOGY AND SOCIETY

From MLPs to CNNs: Introducing DCGAN

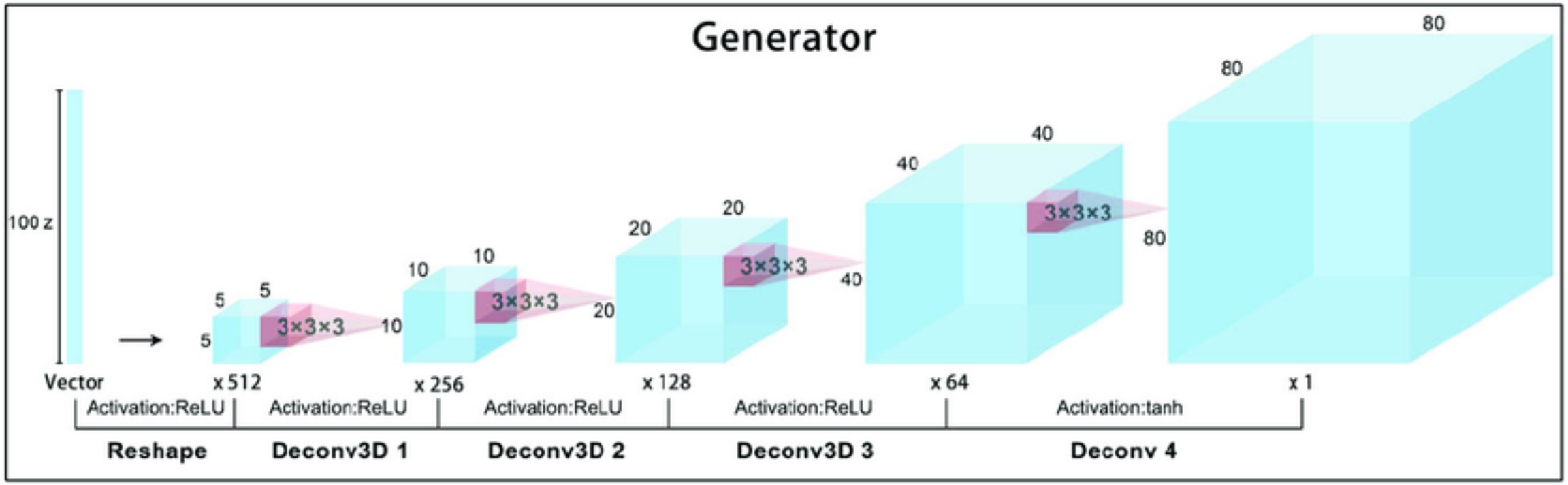
- **Why move away from MLPs?**
  - The original 2014 GAN used **Multi-Layer Perceptrons** (Fully Connected layers). This had major limitations:
    - **Spatial Ignorance**: MLPs treat nearby pixels the same as distant pixels, losing the "geometry" of images.
    - **Parameter Explosion**: Scaling MLPs to high-resolution images requires a massive number of weights, making them nearly impossible to train.
    - **Instability**: MLPs are more prone to the "Mode Collapse" we just discussed.

Discriminator (D) - Architecture



Credit: https://www.researchgate.net/figure/Architecture-of-Generator-and-Discriminator-of-3D-Deep-Convolutional-GAN_fig3_346894735

Generator - Architecture (G)



Credit: https://www.researchgate.net/figure/Architecture-of-Generator-and-Discriminator-of-3D-Deep-Convolutional-GAN_fig3_346894735

DCGAN "Recipe" for Stability

- In 2015, Radford et al. introduced **Deep Convolutional GANs**, providing a set of architectural constraints that made GANs finally work reliably:
  - **Replace Pooling with Strided Convolutions**:
    - Uses **Strided Convolutions** in the Discriminator and **Transposed Convolutions** in the Generator.
    - It allows the network to learn its own downsampling and upsampling logic. This preserves spatial information that "hard-coded" Max Pooling would simply throw away.
  - **Eliminate Fully Connected Layers**:
    - Uses Global Average Pooling instead of dense layers at the end of the networks to reduce parameters and improve stability.
  - **Batch Normalization**:
    - Apply BN to both the Generator and the Discriminator.
    - It stabilizes learning by normalizing the input to each layer to have zero mean and unit variance.
    - Do not apply BN to the Generator output layer or the Discriminator input layer (this prevents sample oscillation and model instability).
  - **Specific Activations:**
    - **Generator**: Use **ReLU** for all layers (except the output, which uses **Tanh** to scale pixels between -1 and 1).
    - **Discriminator**: Uses **Leaky ReLU** for all layers.

Latent Space Arithmetic

- Vector Arithmetic (Face Math) - Just like word embeddings in NLP (e.g., King - Man + Woman = Queen), we can perform high-level algebra on the input noise vectors of a GAN.
- Steps:
  - Take the average z vector for "**Man with Glasses.**"
  - Subtract the average z vector for "**Man without Glasses.**" (This isolates the "Glasses" feature).
  - Add the average z vector for "**Woman without Glasses.**”
  - **Result**: The Generator produces an image of a "**Woman with Glasses.**"

Why does this work?

- **Linearity**: The DCGAN guidelines (Strided Convolutions and Batch Norm) force the Generator to map the random noise to features in a relatively linear way.
- **Semantic Manifolds**: The network learns that "wearing glasses" is a distinct visual concept that can be turned on or off by moving in a specific direction within the 100-dimensional z space.
- **Evidence: Latent Space Interpolation Properties**
  - If you take two random noise vectors, z_1 (e.g., a blue car) and z_2 (e.g., a red car), and slowly transition between them:
    - The GAN doesn't just "fade" one image into the other like a transparency slider.
    - It morphs the features. You will see the car's colour shift through purple, or the shape of the headlights smoothly transform.

Why this matters for Researchers?

- **Proof of Generalization:**
  - This proves the GAN isn't just "memorizing" the training data.
  - If it were just memorizing, adding vectors would result in a corrupted, noisy image.
- **Control:**
  - It opened the door to **Controllable Generation.**
  - This lead to modern tools where you can use sliders to change a character's age, hair colour, or expression.

Conditional GANs (cGAN)

- **The Problem:**
  - Standard GANs are "unsupervised."
  - You give them noise z, and you get a random image. You have no way to tell the model, "I want a picture of a cat, not a dog."
- **The Solution:**
  - In a **Conditional GAN (cGAN)**, we provide an extra piece of information y (like a class label, text, or even another image).
  - This is provided to both the Generator and the Discriminator.

How it works?

- **We feed the label y into both players in the game:**
  - **The Generator**: Now takes two inputs: $G(z, y)$. It learns to map the noise to a specific "mode" of the data based on the label.
  - **The Discriminator**: Now takes two inputs: $D(x, y)$. Its job is no longer just "Is this image real?" but "Is this image real **AND** does it match the label y?"
- **The Modified Objective Function**
  - The "Min-Max" game remains the same, but every term is now conditioned on y:

$$\min_G \max_D V(D, G) = \mathbb{E}_{x \sim p_{data}(x)}[\log D(x \mid y)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z \mid y)))]$$

  - If the Generator makes a perfect, realistic image of a "6" when you asked for a "7," the Discriminator will still reject it.
  - This forces the Generator to respect the label.

Applications of cGANs

- **Text-to-Image**: y is a sentence (e.g., "A blue bird with a yellow beak").
- **Image-to-Image** (Pix2Pix): y is an entire image (e.g., a black-and-white photo, where G generates the colour version).
- **Domain Transfer**: y is a map, and G generates the satellite view.

WGAN (Wasserstein GAN) - Problem with JSD

- **The Problem**: Standard GANs use Jensen-Shannon Divergence (JSD). JSD is upper bounded by $log2$. This can result in "Vanishing Gradient" that leads to training collapse.
- The Definition of JSD (with $M = \frac{1}{2}(P + Q)$)

$$JSD(P \parallel Q) = \frac{1}{2}KL(P \parallel M) + \frac{1}{2}KL(Q \parallel M)$$

- Imagine P (the real data) and Q (the fake data) are perfectly separated. This means for any point x:
  - If $P(x) > 0$, then $Q(x) = 0$.
  - If $Q(x) > 0$, then $P(x) = 0$.
- Now, let's look at the average distribution $M(x) = \frac{P(x) + Q(x)}{2}$:
  - Where $P(x)$ exists, $M(x) = \frac{P(x)}{2}$.
  - Where $Q(x)$ exists, $M(x) = \frac{Q(x)}{2}$.
- Recall that $KL(P \parallel M) = \sum P(x) \log \frac{P(x)}{M(x)}$.
- In the regions where $P(x)$ exists:
- $$\frac{P(x)}{M(x)} = \frac{P(x)}{P(x)/2} = 2$$

Problem with JSD

- So, the KL Divergence becomes:

$$KL(P \parallel M) = \sum P(x)\log(2)$$

- Since $\sum P(x) = 1$, this simplifies to exactly $\log(2)$.
- The same logic applies to $KL(Q \parallel M)$, which also becomes $\log(2)$.
- Now, substitute those back into the JSD formula:

$$JSD(P \parallel Q) = \frac{1}{2}(\log 2) + \frac{1}{2}(\log 2) = \log 2 \approx 0.693$$

WGAN - Earth Mover's Distance

- WGAN replaces the **Discriminator** (which outputs a probability 0 to 1) with a **Critic** (which outputs a real number score).
  - **Standard GAN**: Is this image Real or Fake?
  - **WGAN** Critic: How real is this image?
- **The New Objective**:

$$W(P_r, P_g) = \sup_{\|f\|_L \leq 1} \mathbb{E}_{x \sim P_r}[f(x)] - \mathbb{E}_{x \sim P_g}[f(x)]$$

- The Critic tries to maximize the gap between the scores of real images and fake images.

The 1-Lipschitz Constraint

- For the math to work, the Critic’s function must be "smooth" (it can't have infinitely steep slopes). This is called the 1-Lipschitz constraint.
- **Original WGAN**: Used Weight Clipping (forcing weights to stay between -0.01 and 0.01).
- **Improved WGAN (WGAN-GP)**: Uses a **Gradient Penalty**, which is much more stable and is the modern standard.
  - We add a "penalty term" directly to the Critic's loss function. This term punishes the Critic if the magnitude of its gradient $\|\nabla f(x)\|$ is not close to 1.
  - The Loss Formula:

$$L = \underbrace{\mathbb{E}[f(x_{fake})] - \mathbb{E}[f(x_{real})]}_{\text{Wasserstein Loss}} + \underbrace{\lambda \mathbb{E}_{\hat{x}}[(\|\nabla_{\hat{x}} f(\hat{x})\|_2 - 1)^2]}_{\text{Gradient Penalty}}$$

  - $\hat{x}$ (Interpolated Points): We don't just check the gradient at real or fake images. We pick random points along the lines connecting real and fake images and check the gradient there.
  - λ (Lambda): A hyperparameter (usually 10) that determines how strictly we enforce the penalty.
  - For more details refer to Gulrajani et al. (2017) “Improved Training of Wasserstein GANs”

Why WGAN is a Game Changer?

- **Correlated Loss**: In standard GANs, the loss looks like noise. In WGAN, the Critic's loss actually correlates with image quality. As the loss goes down, the images look better.
- **No More Mode Collapse**: Because the gradient never vanishes, the Generator always has a "path" toward the real data, even if it starts far away.
- **Architecture Freedom**: WGANs are much less sensitive to the "DCGAN rules." You can use different layers and activations, and it will likely still converge.

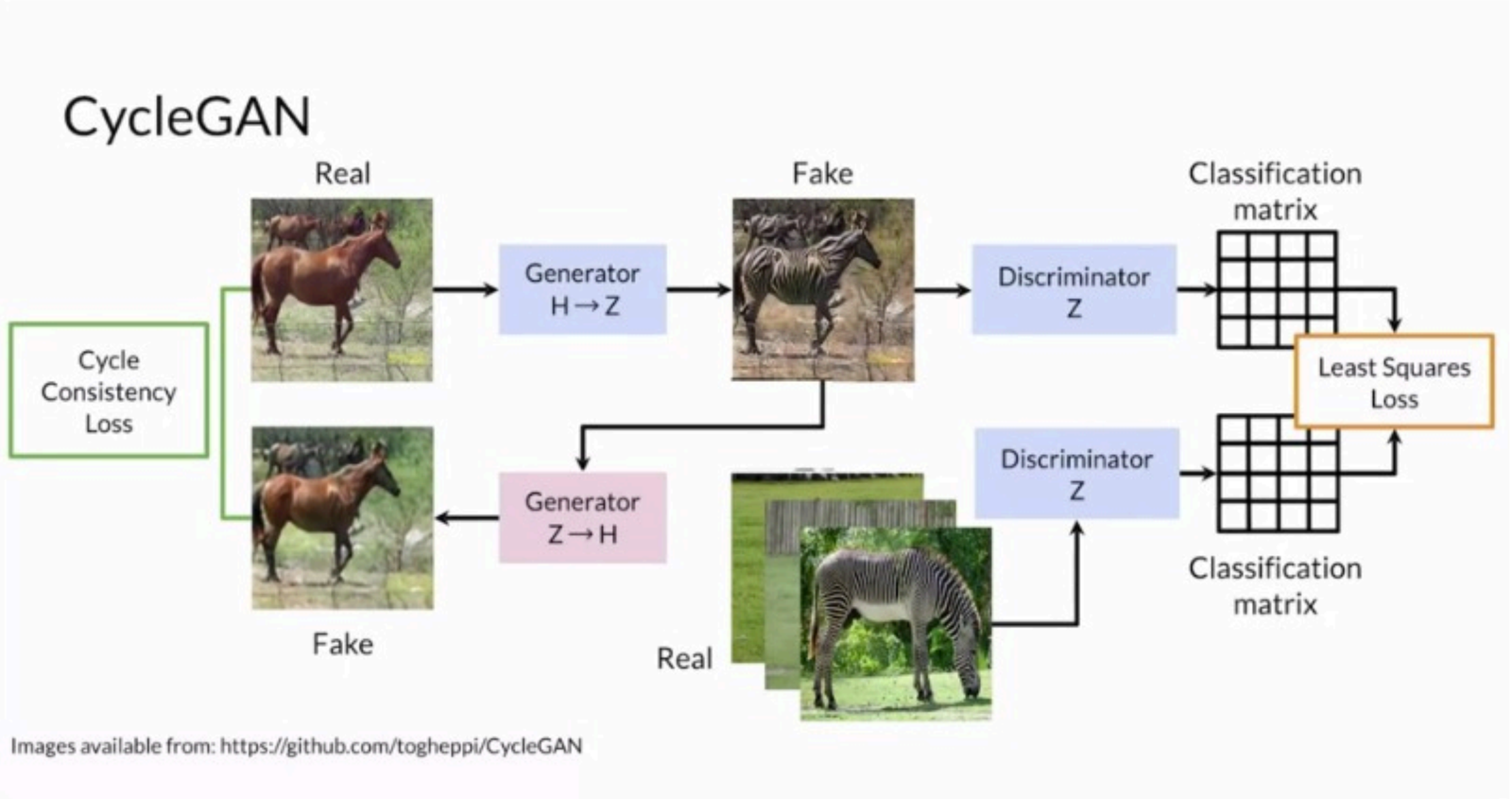
CycleGAN (Image-to-Image translation without paired data)

- **The Problem**: Traditional Image-to-Image models (like Pix2Pix) require paired data. You need a photo of a specific cat and a corresponding "label" image of that exact same cat in a different style. In the real world, we rarely have these perfect pairs (e.g., we don't have a photo of the same animal as both a horse and a zebra).
- **The Solution**: CycleGAN introduces Cycle Consistency Loss. It allows us to learn the mapping between Domain A (Horses) and Domain B (Zebras) using two separate piles of unrelated images.

Cycle GAN - Architecture

- **CycleGAN uses two Generators and two Discriminators:**
  - **Generator** G: Tries to turn a Horse into a Zebra.
  - **Generator** F: Tries to turn a Zebra back into a Horse.
  - **Discriminator** D_A **&** D_B: Check if the generated animals look like real members of their respective species.
- **Cycle Consistency: The "Translation" Rule**
- How do we make sure the network doesn't just replace a horse with a random zebra? We use the **Cycle Loss**:
  - **The Logic**: If you translate a sentence from English to French, and then back to English, you should get the original sentence back.
  - **In GAN terms**: $F(G(Horse)) \approx Horse$.
  - This forces the Generator to preserve the "content" (the pose, the background, the shape) while only changing the "style" (the stripes).

Cycle GAN - Architecture



Credit: https://www.haikutechcenter.com/2020/11/cyclegan-gan-architecture-for-learning.html

Why it’s a Breakthrough

- **No Labels Required**: You just need a folder of "Horse photos" and a folder of "Zebra photos." They don't need to match.
- **Identity Loss**: It prevents the model from changing colours unnecessarily (e.g., it ensures a green field stays green even if the horse becomes a zebra).
- **Creative Freedom**: This technology powers filters that turn "Day to Night," "Summer to Winter," or "Photos into Monet Paintings."

StyleGAN

- **The Revolution**: Created by NVIDIA (2018–present), StyleGAN moved away from the traditional "noise-to-image" approach. Instead of feeding noise directly into the first layer, it uses a **Mapping Network** to control the "style" of the image at every single resolution level.

- **The Mapping Network** ($z \rightarrow w$)

  - In standard GANs, noise z is often "tangled" (e.g., changing hair colour might accidentally change face shape).

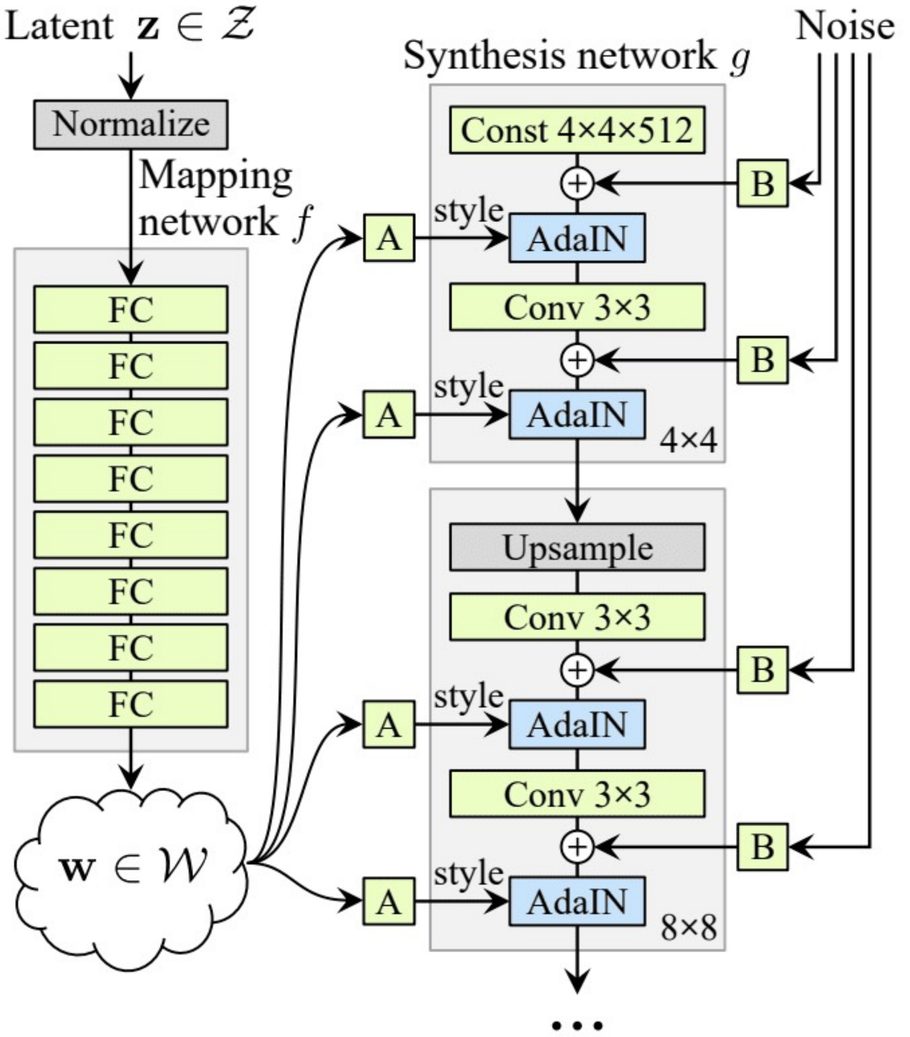
  - StyleGAN feeds z into an 8-layer MLP to create an **Intermediate Latent Space** (w).

  - w is much more "disentangled," meaning specific dimensions correspond to specific, independent features like "age" or "gender."

StyleGAN

- **Progressive Growth & Style Injection**
- The image is built from a small 4×4 resolution up to 1024×1024. At **every scale**, the model "injects" the style vector w using **AdaIN** (Adaptive Instance Normalization).
  - **Coarse Styles (4×4 to 8×8)**: Controls pose, face shape, and hair style.
  - **Middle Styles (16×16 to 32×32)**: Controls facial features (eyes, nose, mouth).
  - **Fine Styles (64×64 to 1024×1024)**: Controls micro-details (skin pores, individual hairs, colour schemes).
- **Stochastic Variation (Noise Injection)**
  - Ever notice how no two humans have the exact same placement of freckles or flyaway hairs?
  - StyleGAN injects **random noise** directly into each layer *after* the style is set.
  - This handles "stochastic" details—things that don't change the person's identity but make the image look "real" (like the exact curl of a hair or the texture of skin).

StyleGAN



Credit: Karras et al. (2019) “A Style-Based Generator Architecture for Generative Adversarial Networks”

Why it Changed the World

- "**This Person Does Not Exist**": StyleGAN achieved such high fidelity that the human eye can rarely distinguish its outputs from real photography.
- **Mixing Styles**: You can take the "Coarse Styles" of Person A (their pose) and the "Fine Styles" of Person B (their skin/hair color) to create a hybrid person.

Sharpness (GANs) vs. Coverage (VAEs)

- **VAE: The Density Estimator (Coverage)**
  - VAEs aim to maximize the **Likelihood** of the data. They try to find a smooth distribution that "covers" all possible real data points.
  - **The Result**: They are mathematically stable and excellent at capturing the entire variety of the dataset.
  - **The Flaw**: Because they use an "average" loss (like Mean Squared Error), their outputs are often blurry. They prefer to be "safely" in the middle of a distribution rather than committing to sharp edges.
- **GAN: The Game Theorist (Sharpness)**
  - GANs don't care about likelihood; they care about **deception**.
  - **The Result**: To fool the Discriminator, a GAN must produce sharp, high-contrast, realistic details. A blurry face is an immediate "Fail" in a GAN's world.
  - **The Flaw**: GANs suffer from **Mode Collapse**. They might get so good at drawing one perfect "7" that they stop bothering to learn how to draw a "1" or a "3." They sacrifice coverage for perfection.
- **The Mathematical Divergence**
  - **VAEs minimize KL Divergence:** This pushes the model to cover all "modes" of the data. If it misses a mode, the loss becomes infinite.
  - **GANs minimize JS Divergence (or Wasserstein):** This allows the model to ignore difficult parts of the data as long as the parts it does generate look real.

Summary Comparison

Feature	Variational Autoencoders (VAE)	Generative Adversarial Networks (GAN)
Primary Goal	Maximize Data Likelihood	Win the Minimax Game
Output Quality	Often Blurry	Highly Sharp & Realistic
Stability	High (Easy to train)	Low (Fickle/Sensitive)
Data Coverage	Excellent (Diverse)	Poor (Risk of Mode Collapse)
Latent Space	Structured & Continuous	Organized but potentially "gappy"

Pros and Cons of GANs

- **The Pros: Why we use them**
  - **Superior Sharpness**: Unlike VAEs, GANs do not suffer from blurriness. The "adversarial" nature forces the model to produce crisp, high-frequency details.
  - **Latent Space Power**: As seen in DCGAN and StyleGAN, the latent space (z or w) is semantically meaningful, allowing for "Vector Arithmetic" and precise feature manipulation.
  - **Fast Inference**: Once trained, a GAN Generator is a simple "feed-forward" pass. It can generate high-resolution images in milliseconds, unlike modern Diffusion models which require multiple "steps."
  - **Versatility**: They can be adapted for almost any task: text-to-image, image-to-image (CycleGAN), and even super-resolution (SRGAN).

Pros and Cons of GANs

- **The Cons: The "Dark Side" of GANs**
  - **Training Instability**: Because you are training two networks against each other, the system is a **non-convex game**. Instead of finding a "minimum," you are looking for a **Nash Equilibrium**, which is incredibly hard to hit.
  - **Mode Collapse**: The Generator may find a single output that "tricks" the Discriminator and then stop producing anything else. You get 1,000 identical, perfect images of a cat, but zero dogs.
  - **Hyperparameter Sensitivity**: A GAN might work perfectly with a learning rate of 0.0002, but completely "explode" (diverge) at 0.0003.
  - **Evaluation Difficulties**: There is no "objective" loss function. You cannot look at the loss curve to know if your images look good; you often have to rely on human inspection or metrics like **FID (Fréchet Inception Distance)**.

Modern Applications

- **Super-Resolution (SRGAN)**
  - GANs can "hallucinate" missing details in low-resolution images.
    - **How it works:** Instead of simple interpolation (which makes images blurry), the Discriminator punishes "smooth" edges, forcing the Generator to create realistic textures like skin pores or fabric weaves.
    - **Use Case:** Upscaling old films, satellite imagery, and forensic analysis.
- **Deepfakes & Digital Humans**
  - This is the most famous (and controversial) application, popularized by models like **StyleGAN** and **First Order Motion Models**.
    - **Face Swapping**: Replacing one person's face with another while maintaining the original lighting and expression.
    - **Puppetry**: Using one person's movements to "drive" a static image of someone else.
    - **Impact**: Changing the film industry (de-aging actors) but raising significant ethical concerns regarding consent and misinformation.
- **Medical Imaging & Data Augmentation**
  - Medical data is often expensive to collect and protected by privacy laws.
    - **Synthetic Data**: GANs can generate "fake" MRI or CT scans that are medically accurate but don't belong to a real patient.
    - **Cross-Modality**: Turning a CT scan into an MRI-style image (Image-to-Image translation).
    - **Benefit**: Training other AI models to detect tumours when real-world examples are limited.

Ethical Implications

- **The Risks of Synthetic Media**
  - **Misinformation & Deepfakes**: GANs can be used to create non-consensual imagery or fabricate political events. This leads to the "Liar's Dividend", where real evidence is dismissed as "just an AI fake."
  - **Bias Reinforcement**: If a GAN is trained on biased data, it will amplify those biases. For example, early face generators often struggled with diverse skin tones or reinforced gender stereotypes.
  - **Identity Theft**: Using voice and face synthesis to bypass biometric security or conduct high-level phishing attacks.
- **The Defence: Detection & Watermarking**
  - As generation gets better, detection must evolve. Current strategies include:
    - **Artifact Analysis**: Looking for "checkerboard" patterns or inconsistencies in blinking and heart rate (biological signals) that GANs often miss.
    - **Frequency Domain Analysis**: Using Fourier transforms to find high-frequency "noise" left behind by the upsampling process in DCGAN-style architectures.
    - **Digital Provenance (C2PA)**: Embedding cryptographic metadata at the moment of capture to prove an image is a "real" photograph.
- **Responsible AI Frameworks**
  - **Transparency**: Clearly labeling AI-generated content (e.g., "Generated by AI").
  - **Consent**: Restricting the use of person-specific data for training without explicit permission.
  - **Open Source Responsibility**: The debate over whether high-fidelity weights (like StyleGAN) should be released to the public or kept behind "safety filters."