Machine Vision

Ali Rida SAHILI

Generative Adversarial Networks (GANs)

Definition

Generative Adversarial Networks

Generative Models

We try to learn the underlying the distribution from which the dataset comes from.

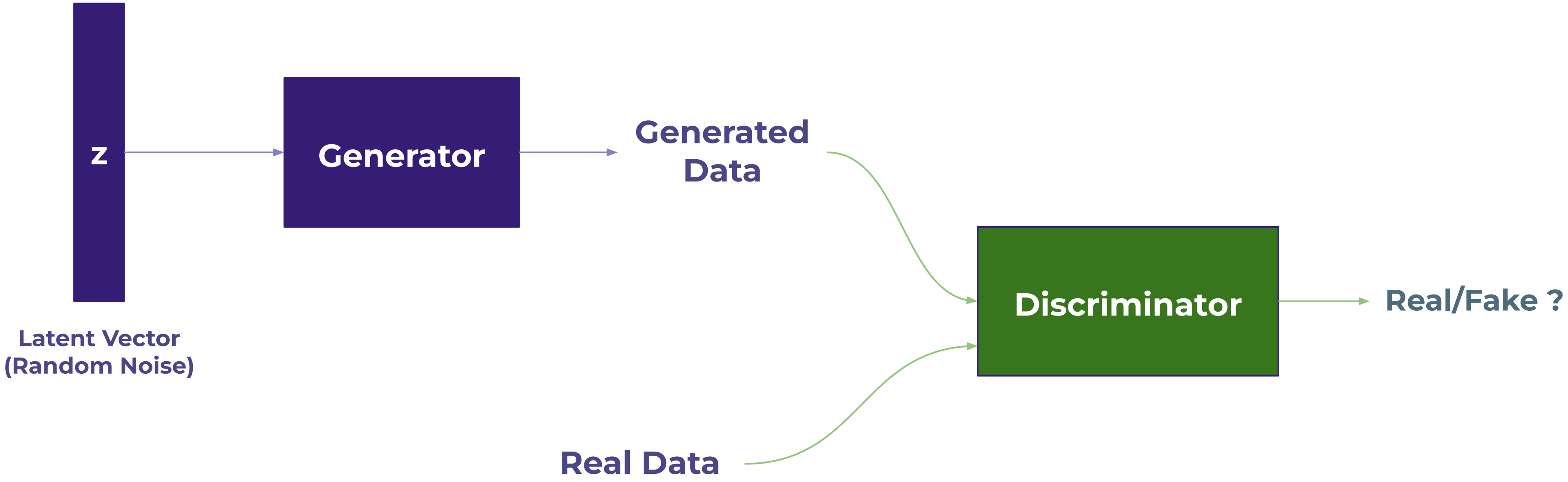
Adversarial Training

GANS are made up of two competing networks (adversaries) that are trying beat each other.

Neural Networks

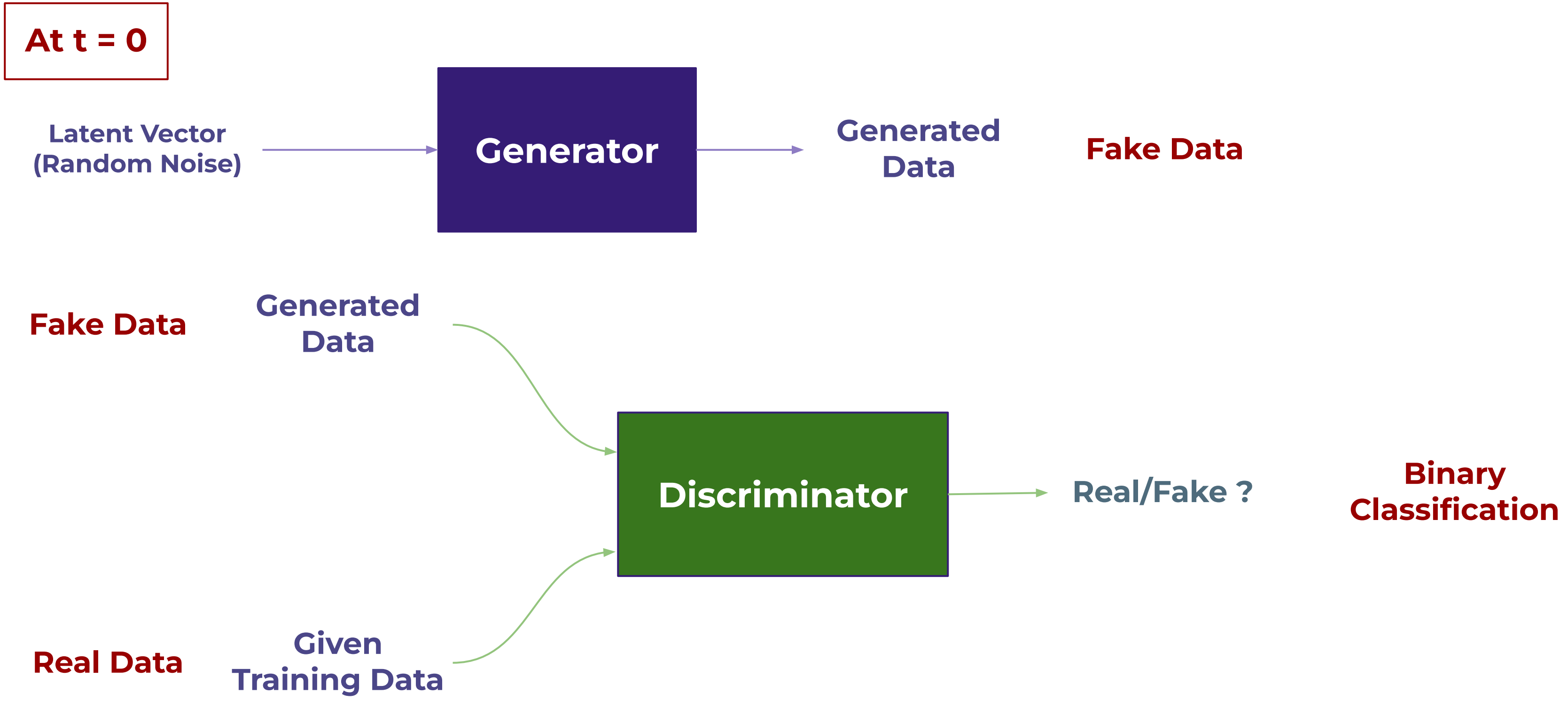
Generative Adversarial Networks (GANs)

Definition



Generative Adversarial Networks (GANs)

Training Phase

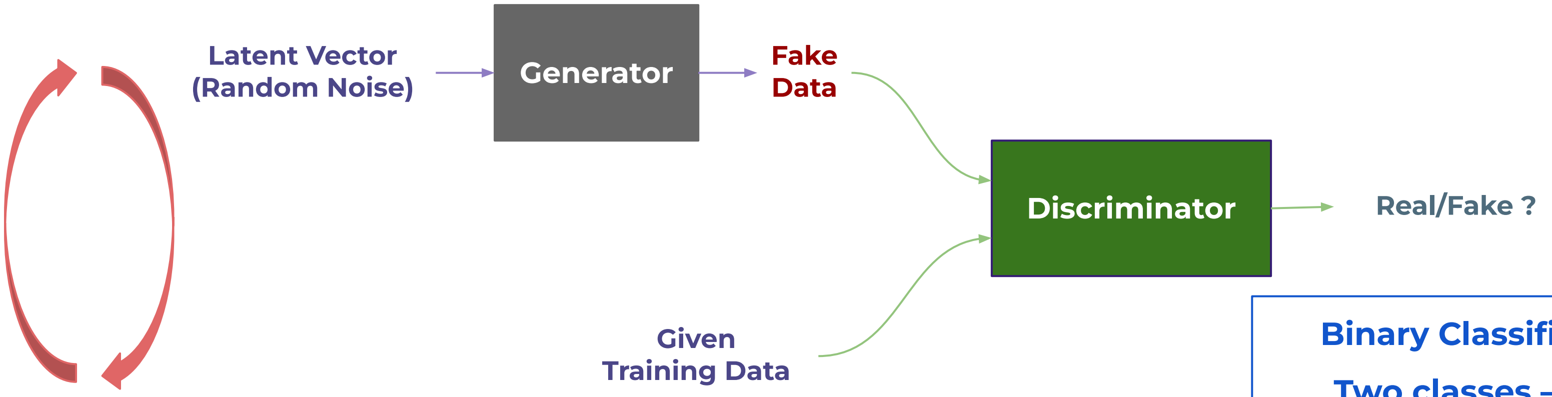


Generative Adversarial Networks (GANs)

Training Phase

Step 1: Train Discriminator

Use the current ability of the Generator



Binary Classification Problem

Two classes – Fake and Real

Step 2: Train Generator

Beating the Discriminator

Generative Adversarial Networks (GANs)

Training Phase

Fake = 0 / Real = 1

$$V(D,G) = \boxed{\mathbb{E}_{x\sim p(x)}[\log D(x)]} + \boxed{\mathbb{E}_{z\sim q(z)}[\log(1 - D(G(z)))]}$$

Step 1: Train Discriminator

Maximizing the classification over the real Data

Maximizing the classification over the Fake Data Generated by G

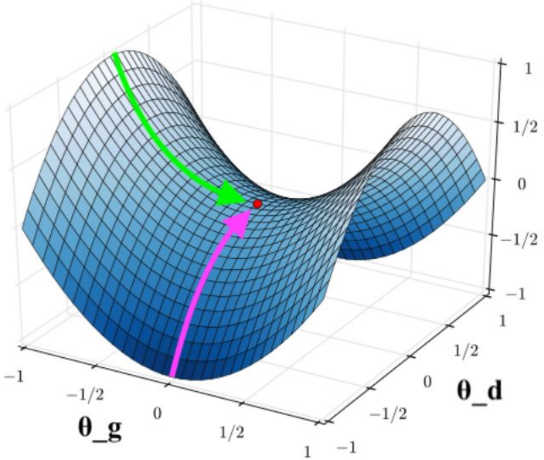
Step 2: Train Generator

–

Generating new Data minimizing the ability of D to classify fake data

$$\min_G \max_D V(D,G)$$

Saddle Point Optimization



Generative Adversarial Networks (GANs)

Training Phase

Fake = 0 / Real = 1

$$V(D, G) = \mathbb{E}_{x \sim p(x)}[\log D(x)] + \mathbb{E}_{z \sim q(z)}\left[\log\left(1 - D(G(z))\right)\right]$$

Fake = 1 / Real = 0

$$V(D, G) = \mathbb{E}_{x \sim p(x)}\left[\log\left(1 - D(x)\right)\right] + \mathbb{E}_{z \sim q(z)}\left[\log\left(D(G(z))\right)\right]$$

Generative Adversarial Networks (GANs)

Training Procedure

for i = 1 . . . N **do**

for k = 1 . . . K **do**

- Sample noise samples $\{\mathbf{z}^1, \ldots, \mathbf{z}^m\} \sim p_z(\mathbf{z})$
- Sample examples $\{\mathbf{x}^1, \ldots, \mathbf{x}^m\}$ from $p_{data}(\mathbf{x})$.
- Update the discriminator $D_{\theta d}$:

$$\theta_d = \theta_d - \alpha_d \nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\mathbf{x}^i\right) + \log\left(1 - D\left(G\left(\mathbf{z}^i\right)\right)\right)\right].$$

end for

- Sample noise samples $\{\mathbf{z}^1, \ldots, \mathbf{z}^m\} \sim p_z(\mathbf{z})$.
- Update the generator $G_{\theta g}$:

$$\theta_g = \theta_g - \alpha_g \nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\mathbf{z}^i\right)\right)\right).$$

end for

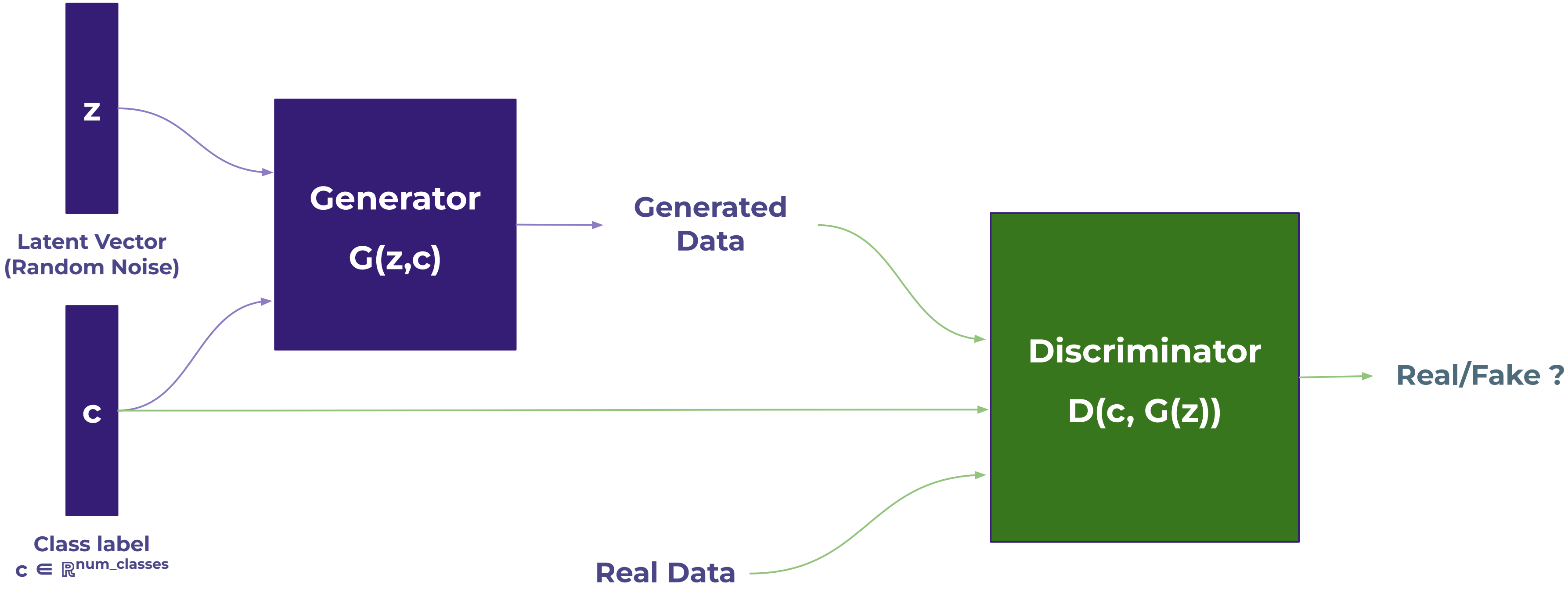
Generative Adversarial Networks (GANs)

Training Tricks

1. **Use Soft and Noisy Labels:**
   - Label Smoothing, i.e. if you have two target labels: Real=1 and Fake=0, then for each incoming sample, if it is real, then replace the label with a random number between 0.7 and 1.2, and if it is a fake sample, replace it with 0.0 and 0.3 (for example).
2. **Different Optimizers:** Use SGD for Discriminator and ADAM for Generator
3. **Schedule Learning:** try to balance training D and G.
4. **Limit Discriminator:** restrict the capacity of the discriminator.
5. **Progressive Growing:** start with low-resolution images and gradually increase resolution by adding layers.

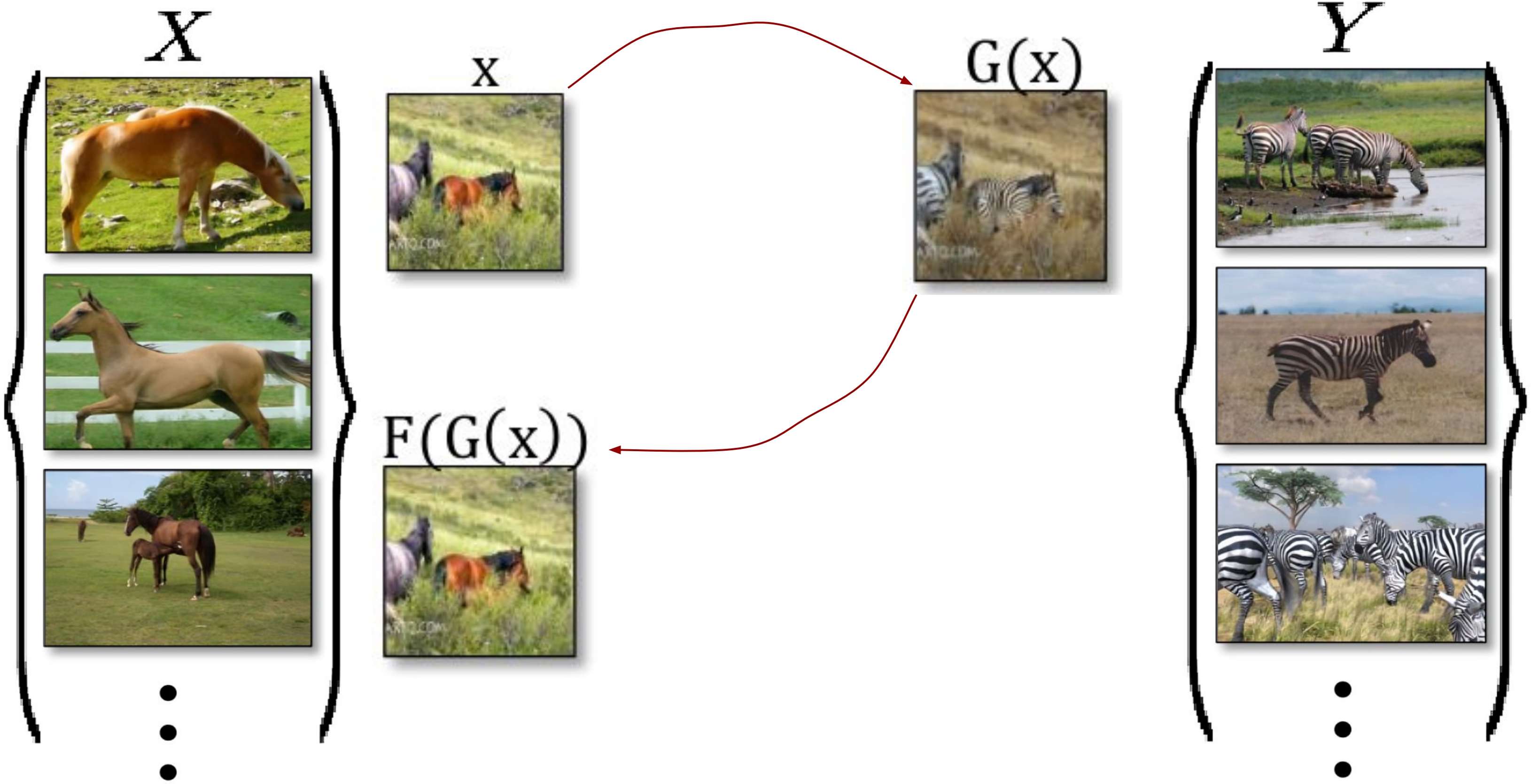
Conditional GAN (cGAN)

Definition



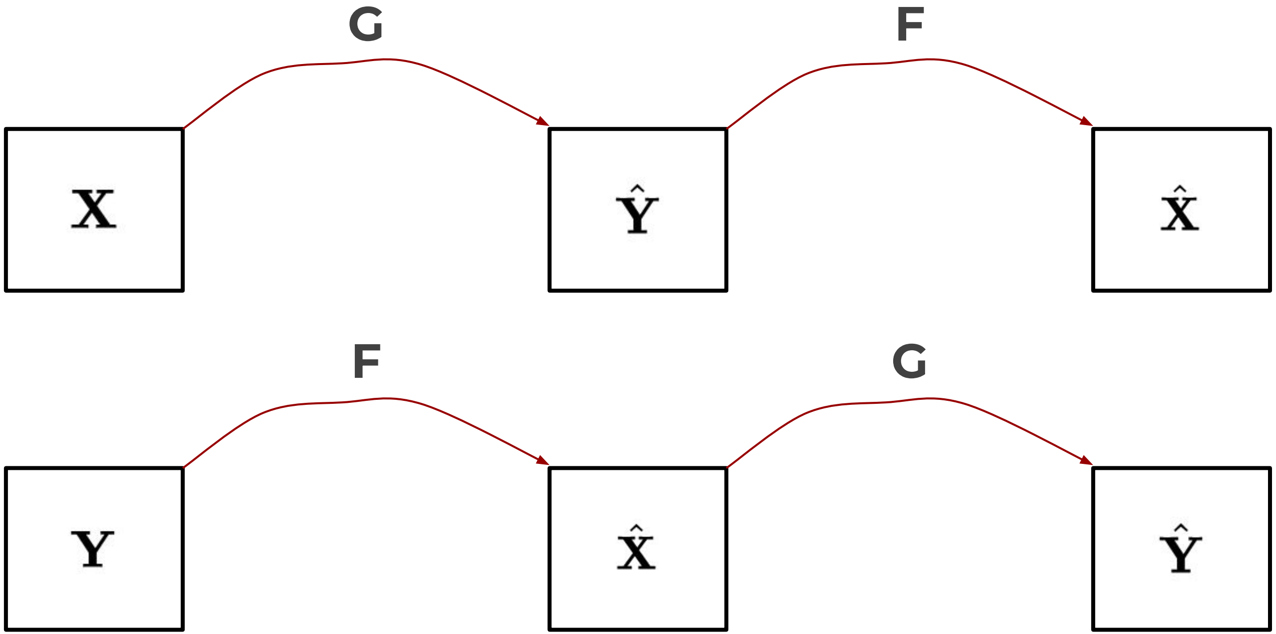
Neural Transfer Style

Definition – CycleGAN



Neural Transfer Style

CycleGAN



Reconstruction Error

$$\left\| F\big(G(x)\big) - x \right\|_1$$

$$\left\| G\big(F(y)\big) - y \right\|_1$$

Neural Transfer Style

CycleGAN

$$\begin{aligned}\mathcal{L}_{\text{GAN}}(G, D_Y, X, Y) &= \mathbb{E}_{y\sim p_{\text{data}}(y)}[\log D_Y(y)] \\ &+ \mathbb{E}_{x\sim p_{\text{data}}(x)}[\log(1 - D_Y(G(x))],\end{aligned}$$

Domain X   Domain Y

Total Loss

$$\begin{aligned}\mathcal{L}(G, F, D_X, D_Y) =&\mathcal{L}_{\text{GAN}}(G, D_Y, X, Y) \\ &+ \mathcal{L}_{\text{GAN}}(F, D_X, Y, X) \\ &+ \lambda\mathcal{L}_{\text{cyc}}(G, F),\end{aligned}$$

Cycle-Consistency Loss

$$\begin{aligned}\mathcal{L}_{\text{cyc}}(G, F) &= \mathbb{E}_{x\sim p_{\text{data}}(x)}[\|F(G(x)) - x\|_1] \\ &+ \mathbb{E}_{y\sim p_{\text{data}}(y)}[\|G(F(y)) - y\|_1].\end{aligned}$$

Evaluation Modalities

- Summary of the project:
  - Implement a **CNN classifier** for the cows-horses dataset.
  - Improve classifier accuracy by doing **GAN-based data augmentation**.
  - Do **unpaired neural style transfer** between images of the two classes (cow, horse).

For all 3 steps, you need to clearly explain and justify your code/method.

- What we expect from you:
  - **Working notebook**, with clearly explained sections!
  - A **pdf report** which contains the code, figures, and your interpretations.
  - An **individual work**.
  - An **oral presentation** of you work.
- Deadline: **TBA** (end of April/beginning of May)