Generative Adversarial Networks(GANs)

Anuj Bansal (2018201096)
Nilabja Bhattacharya (2018201036)

Why Generative Models?

- **In discriminative models**
  - Given an image X, predict a label Y
  - Estimates P(Y|X)

- **Discriminative models have several key limitations**
  - Can't model P(X), i.e. the probability of seeing a certain image
  - Thus, can't sample from P(X), i.e. can't generate new images

- **Generative models (in general) cope with all of above**
  - Can model P(X)
  - Can generate new images

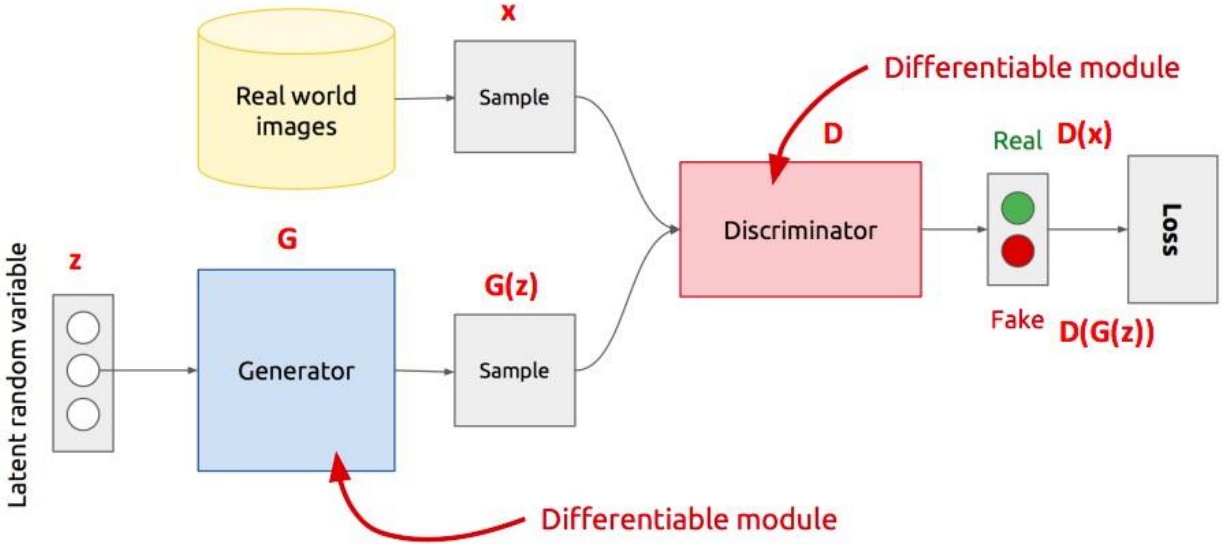
Magic of GAN

Which one is Computer generated?

Adversarial training

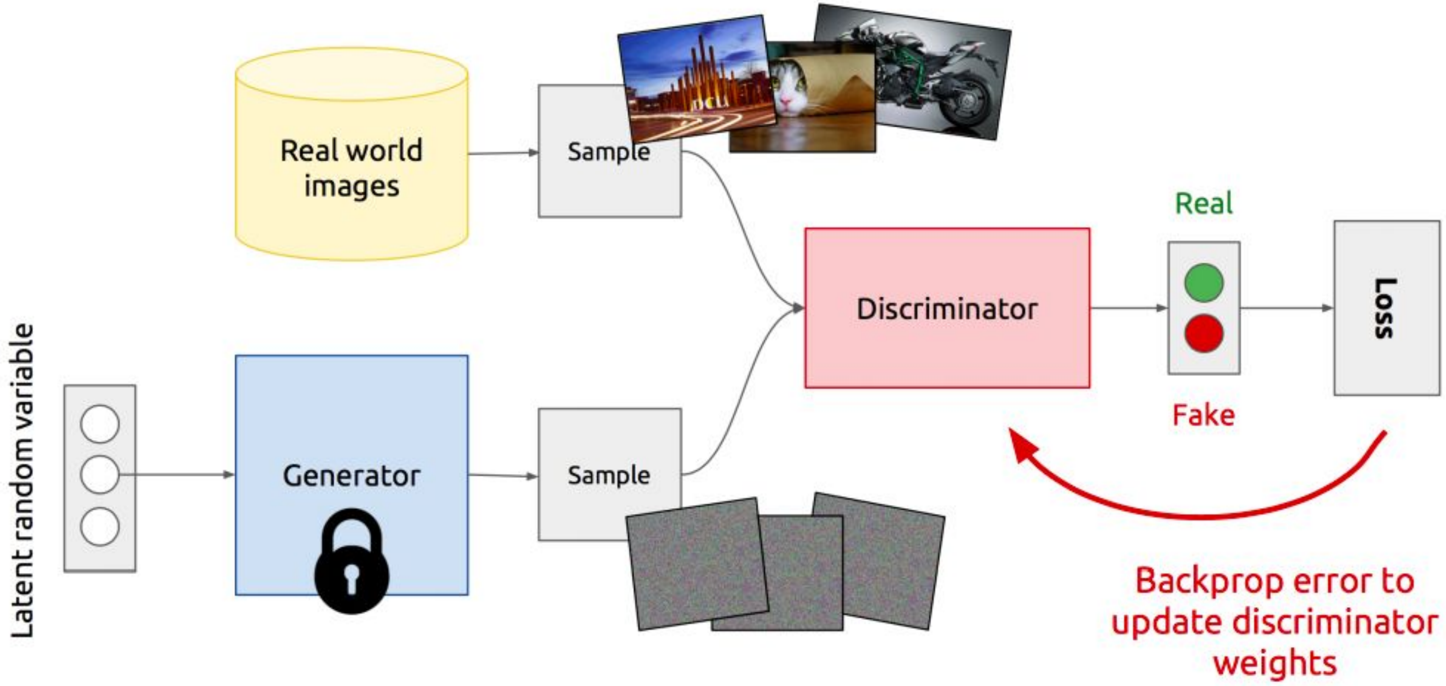
- **In discriminative models**
  - We can generate adversarial samples to fool a discriminative model
  - We can use those adversarial samples to make models robust
  - We then require more effort to generate adversarial samples
  - Repeat this and we get better discriminative model
- **GANs extend that idea to generative models**
  - Generator: generate fake samples, tries to fool the Discriminator
  - Discriminator: tries to distinguish between real and fake samples
  - Train them against each other
  - Repeat this and we get better Generator and Discriminator

GAN’s Architecture

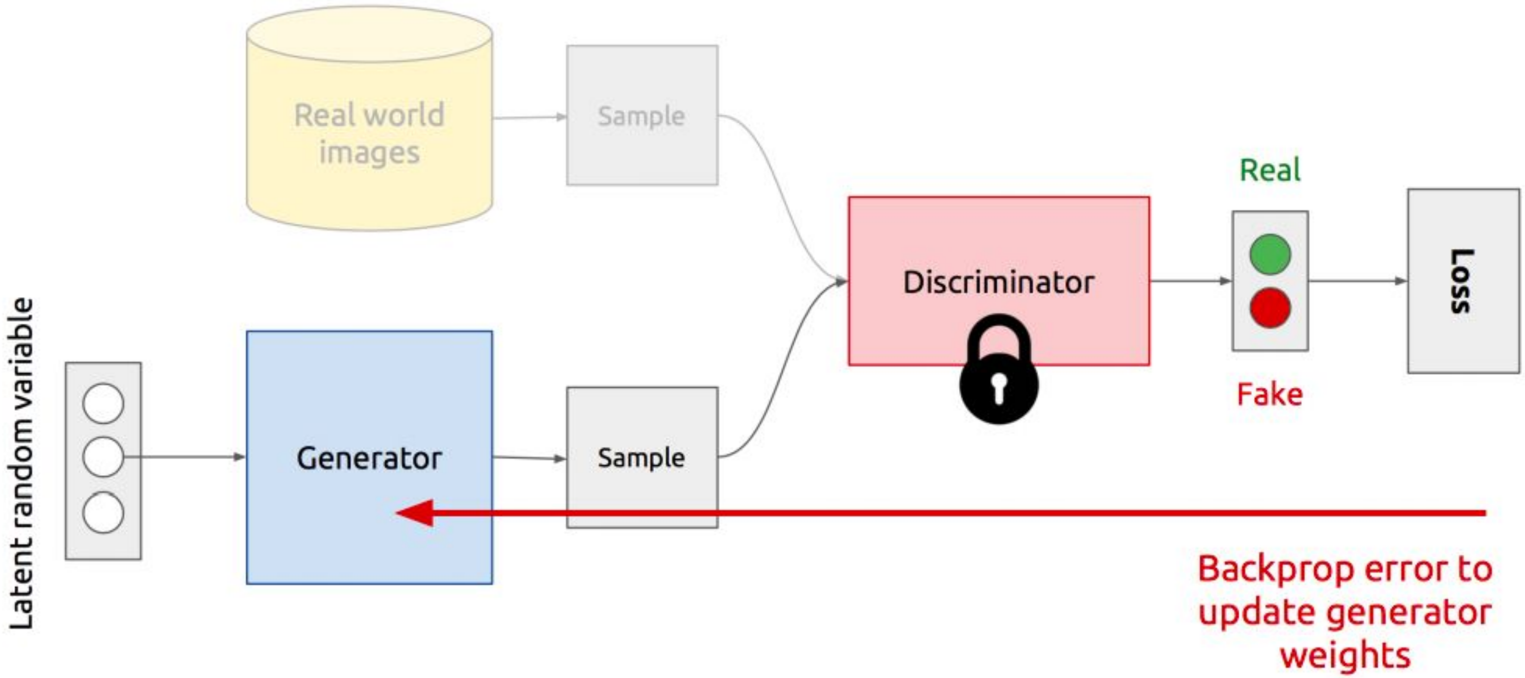


- **Z** is some random noise (Gaussian/Uniform).
- **Z** can be thought as the latent representation of the image.

Training Discriminator



Training Generator



GAN's formulation

$$\min_{G} \max_{D} V(D, G)$$

- It is formulated as a **minimax game**, where:
  - The Discriminator is trying to maximize its reward $\boldsymbol{V(D, G)}$
  - The Generator is trying to minimize Discriminator's reward (or maximize its loss)

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

- The Nash equilibrium of this particular game is achieved at:
  - $P_{data}(x) = P_{gen}(x) \;\; \forall x$
  - $\mathrm{D}(x) = \frac{1}{2} \;\; \forall x$

GAN's Algorithm

Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k, is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

for number of training iterations **do**

Discriminator updates

for k steps **do**

- Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \dots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
- Sample minibatch of m examples $\{\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$.
- Update the discriminator by ascending its stochastic gradient:

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

end for

Generator updates

- Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \dots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
- Update the generator by descending its stochastic gradient:

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

end for

The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

Advantages of GAN

- **Plenty of existing work on Deep Generative Models**
    - Boltzmann Machine
    - Deep Belief Nets
    - Variational Auto Encoders (VAE)

- **Why GANs?**
    - Sampling (or generation) is straightforward
    - Training doesn't involve Maximum Likelihood estimation
    - Robust to Overfitting since Generator never sees the training data
    - Empirically, GANs are good at capturing the modes of the distribution

Problems with GAN

- **Probability Distribution is Implicit**
  - Not straightforward to compute P(X)
  - Thus Vanilla GANs are only good for Sampling/Generation.

- **Training is Hard**
  - Non-Convergence
  - Mode-Collapse

Result of GAN

Thank you