

Generative Adversarial Networks - GANs

Prof. Carlos Alberto de Araújo Padilha

17 de Setembro, 2018

Sumário

1 Introdução

2 Generative Adversarial Networks - GANs

- Standard GAN
- Deep Convolutional Generative Adversarial Network - DCGAN

Sumário

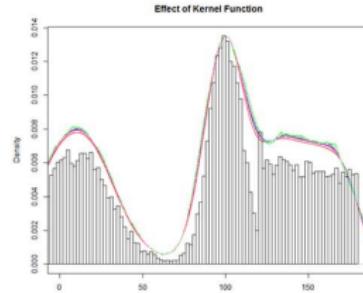
1 Introdução

2 Generative Adversarial Networks - GANs

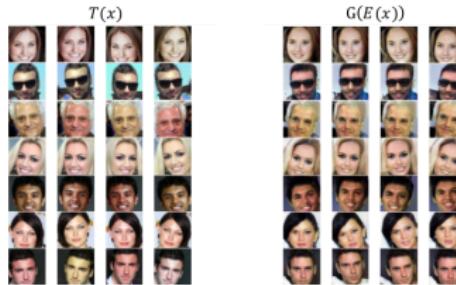
- Standard GAN
- Deep Convolutional Generative Adversarial Network - DCGAN

Introdução

- Modelagem generativa
 - Estimação da densidade de probabilidade



- Geração de amostras

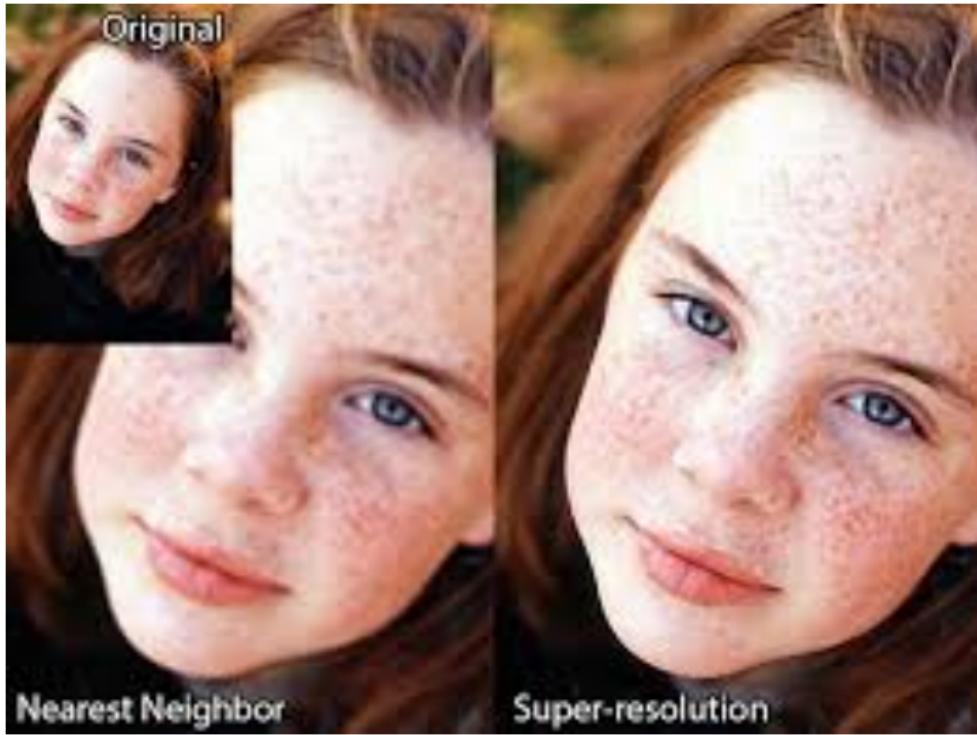


Introdução

- Podemos testar nossa habilidade de aprender distribuições complexas com alta dimensionalidade.
- Simular padrões futuros para problemas de aprendizagem por reforço.
- Gerar dados em problemas onde as amostras são excassas e custosas de seguir - **aprendizagem semi-supervisionada**.
- Geração de imagens realísticas - Introspective Adversarial Networks (Youtube video).

Predição do próximo frame de um vídeo

Reconstrução de imagens



Sumário

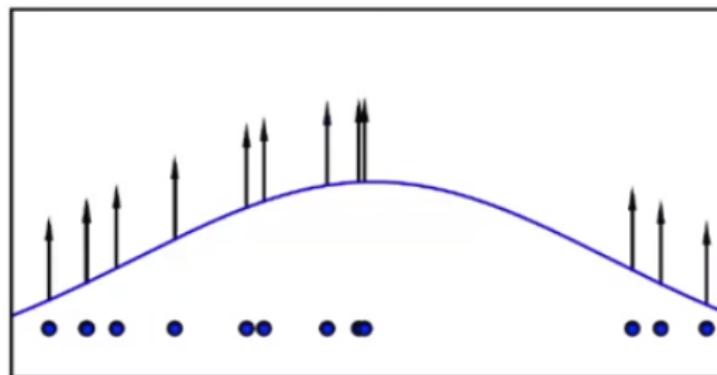
1 Introdução

2 Generative Adversarial Networks - GANs

- Standard GAN
- Deep Convolutional Generative Adversarial Network - DCGAN

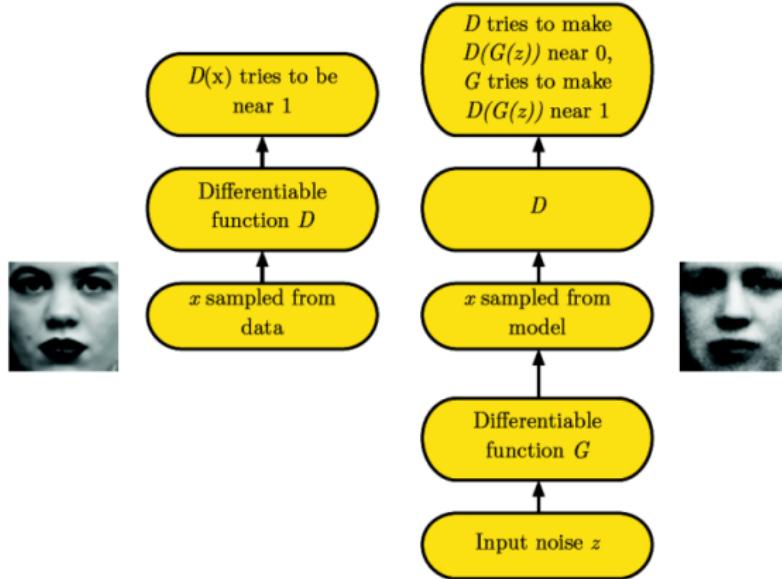
Generative Adversarial Networks - GANs

- Propostas por Ian Goodfellow e colaboradores em 2014.
- Assim como outros métodos generativos, as GANs implementam a estimativa da máxima verossimilhança.

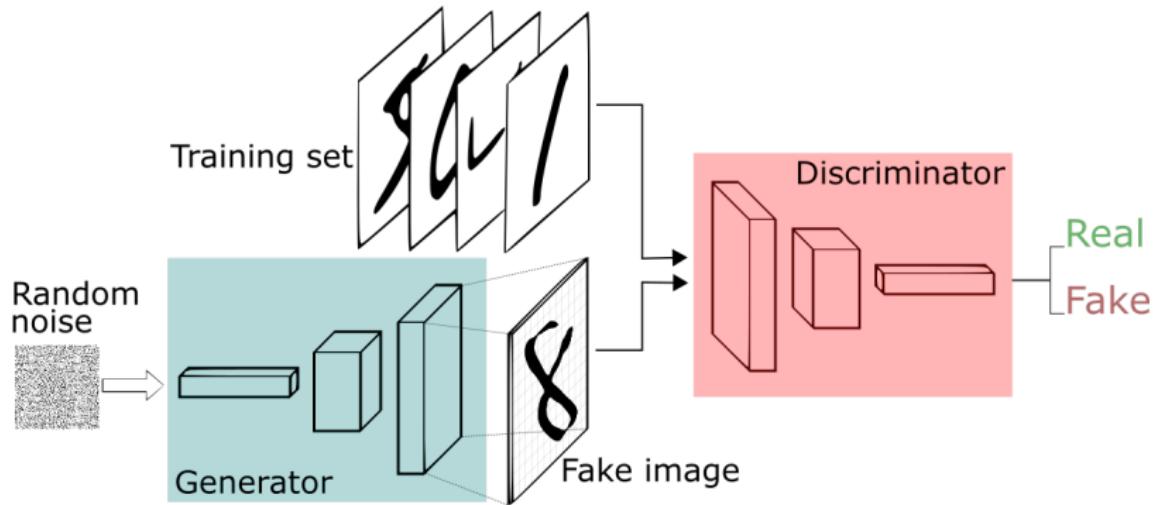


$$\theta^* = \arg \max_{\theta} \mathbb{E}_{x \sim p_{\text{data}}} \log p_{\text{model}}(x | \theta)$$

Adversarial Nets Framework



Framework das GANs



Treinamento

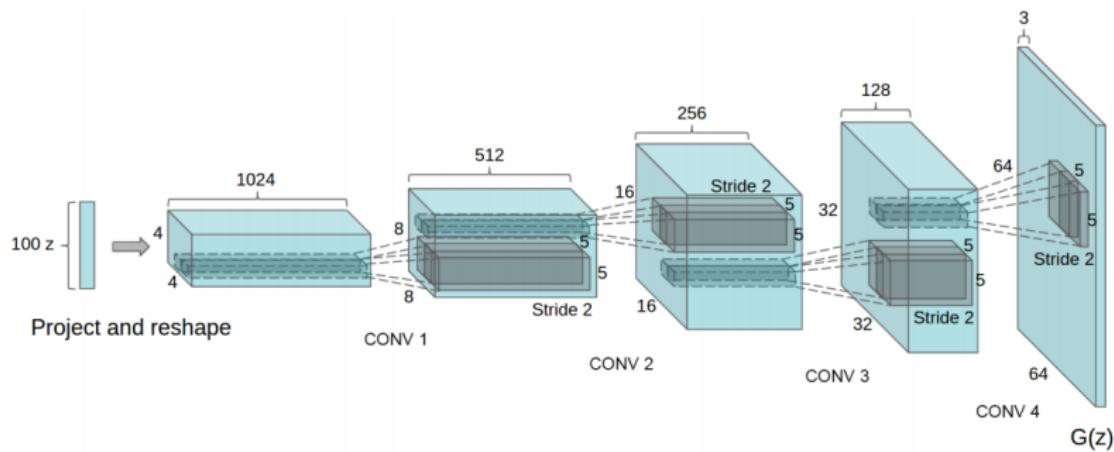
- Usar qualquer algoritmo baseado em SGD (*Stochastic Gradient Descent*) em dois batches de amostras simultaneamente:
 - Um batch de amostras de treinamento
 - Um batch de amostras geradas pelo generator
- Otimização

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

Variantes das GANs

- AC-GAN - <https://arxiv.org/abs/1610.09585>
- CC-GAN - <https://arxiv.org/abs/1611.06430>
- DCGAN - <https://arxiv.org/abs/1511.06434>
- Pix2Pix - <https://arxiv.org/abs/1611.07004>
- WGAN - <https://arxiv.org/abs/1701.07875>
- ...

Deep Convolutional Generative Adversarial Network - DCGAN



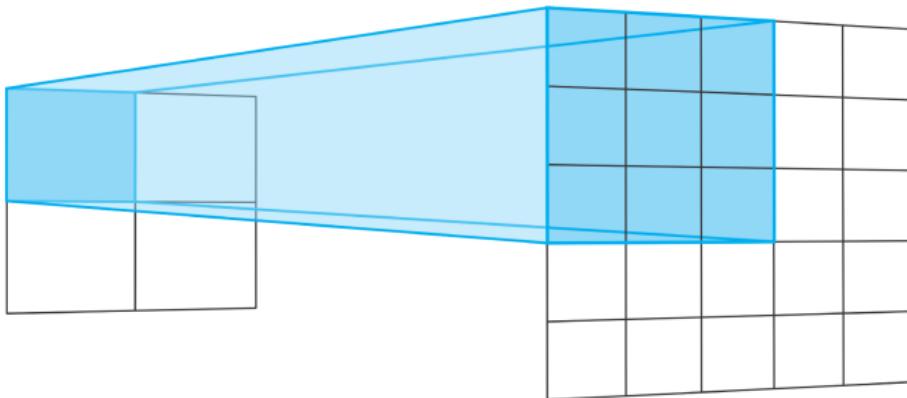
Deep Convolutional Generative Adversarial Network - DCGAN

- As DCGANs foram propostas em 2016 e adicionaram às GANs algumas técnicas de Deep Learning como forma de estabilizar o treinamento desse tipo de rede:
 - Usar apenas camadas convolutivas: no lugar de usar camadas de pooling, os autores enfatizaram o uso de camada convolutivas "strided", ou seja, aumentando e diminuindo as dimensões espaciais dos features maps.
 - Batch normalization (BN): normaliza os feature maps para ter média zero e variância unitária em todas as camadas.

DCGAN - Generator

- O generator recebe como entrada um vetor aleatório z gerado com uma distribuição normal.
- A arquitetura da rede do generator é composta por 4 camadas de convoluções transpostas, tendo entre cada uma delas uma camada de BN seguida de uma camada de ativação ReLU.
- A entrada recebida pelo generator é profunda mas estreita. A cada passagem por uma das camadas de convolução, as features vão se tornando menos profundas e mais amplas.

Exemplo de Convolução Transposta

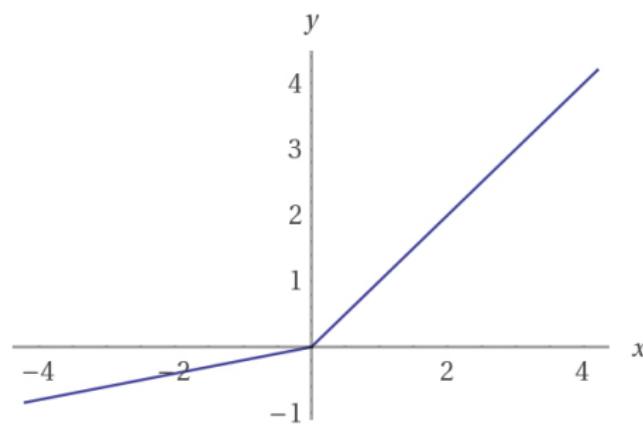
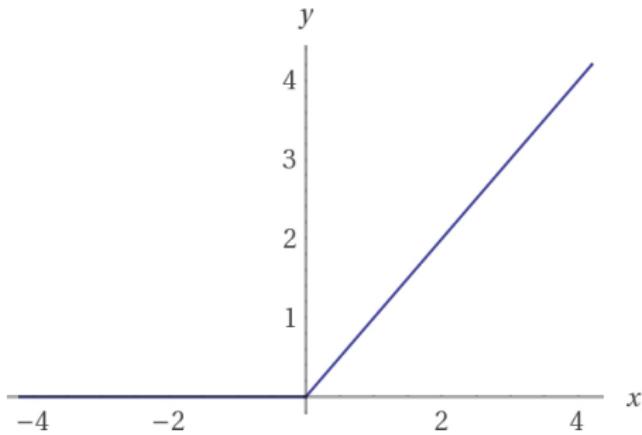


DCGAN - Generator

- Todas as convoluções transpostas usam filtros 5×5 e stride 2.
- Assim, as features terão suas profundidades reduzidas de 512 até 3, representando os canais RGB.
- A função de ativação da última camada é a tangente hiperbólica, escalando os valores das features entre -1 e 1.
- Para acompanhar essa escolha, o vetor z também precisa ser escalado nessa faixa.
- A largura e altura das imagens produzidas irão variar dependendo do dataset (para o CIFAR seria 32×32).

DCGAN - Discriminador

- A arquitetura do discriminador também conta com 4 camadas convolutivas com BN, mas com funções de ativação do tipo leaky ReLU.
- A leaky ReLU ajuda os gradientes a fluirem melhor através da rede, pois não truncam valores negativos em zero.
- $\text{LeakyReLU} = \max(\alpha * x, x)$



DCGAN - Discriminador

- O discriminador recebe uma imagem de tamanho $H \times W \times 3$ e a cada passagem por uma convolução com stride 2, as features vão reduzindo em largura e altura pela metade, enquanto que o número de filtro dobra.
- Na última camada, usamos uma função sigmoid para que o modelo faça a classificação binária das imagens (real ou falsa).
- Assim, durante o treinamento, quando discriminador classifica as imagens do generator como falsas, a informação de gradiente irá ajustar o generator para que ele gere imagens mais fidedignas as do dataset.

Vamos praticar!

