Generative Adversarial Networks

Kate Farrahi

ECS Southampton

May 1, 2020

Lecture Outline

- Paper 1: Goodfellow, Ian, et al. "Generative adversarial nets." Advances in neural information processing systems. 2014.
- Paper 2: Radford, Alec, Luke Metz, and Soumith Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks." arXiv preprint arXiv:1511.06434. 2015.
- Paper 3: Arjovsky, Martin, Soumith Chintala, and Leon Bottou. "Wasserstein generative adversarial networks." International Conference on Machine Learning. 2017.

Generative Adversarial Networks (GANs)

- New method of training deep generative models

Generative Adversarial Networks (GANs)

- New method of training deep generative models
- Idea: put a generator and a discriminator against each other

Generative Adversarial Networks (GANs)

- New method of training deep generative models
- Idea: put a generator and a discriminator against each other
- Generator tries to draw samples from P(X)

Generative Adversarial Networks (GANs)

- New method of training deep generative models
- Idea: put a generator and a discriminator against each other
- Generator tries to draw samples from P(X)
- Discriminator tries to tell if sample came from the generator (fake) or the real world

Generative Adversarial Networks (GANs)

- New method of training deep generative models
- Idea: put a generator and a discriminator against each other
- Generator tries to draw samples from P(X)
- Discriminator tries to tell if sample came from the generator (fake) or the real world
- Both discriminator and generator are deep networks (differentiable functions)

Generative Adversarial Networks (GANs)

- New method of training deep generative models
- Idea: put a generator and a discriminator against each other
- Generator tries to draw samples from P(X)
- Discriminator tries to tell if sample came from the generator (fake) or the real world
- Both discriminator and generator are deep networks (differentiable functions)
- LeCun quote "GANs, the most interesting idea in the last ten years in machine learning"

Generative Adversarial Networks (GANs)

- New method of training deep generative models
- Idea: put a generator and a discriminator against each other
- Generator tries to draw samples from P(X)
- Discriminator tries to tell if sample came from the generator (fake) or the real world
- Both discriminator and generator are deep networks (differentiable functions)
- LeCun quote "GANs, the most interesting idea in the last ten years in machine learning"

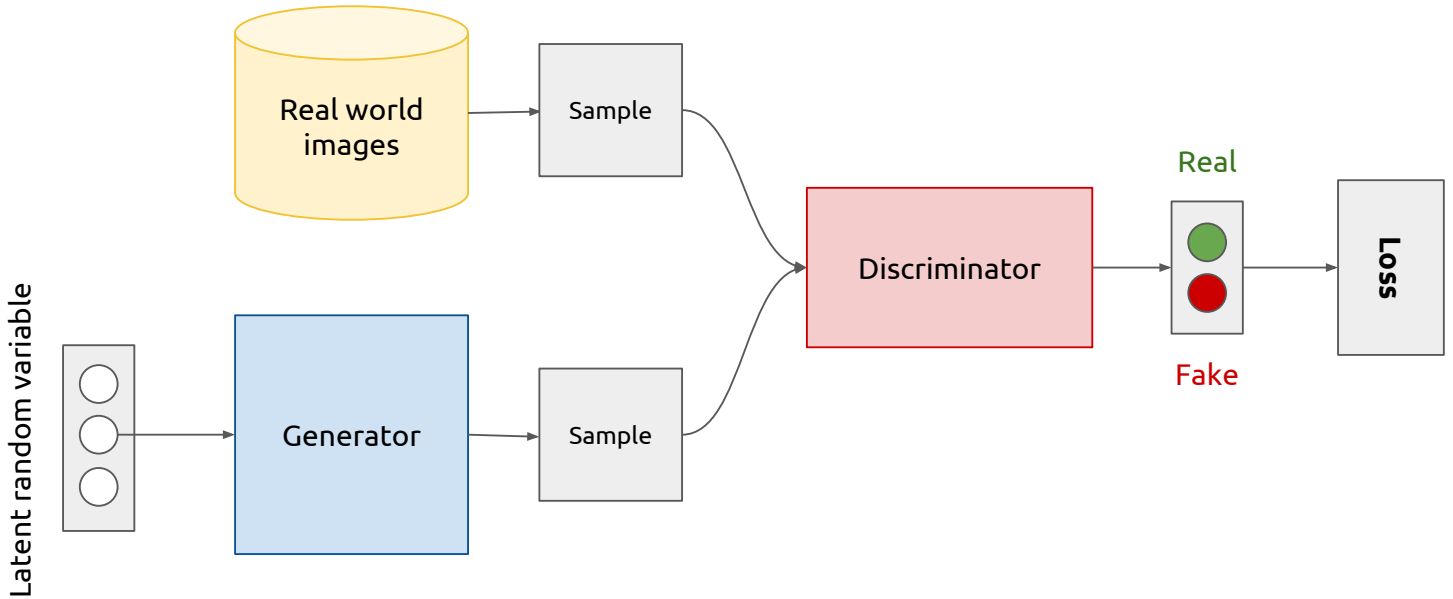
Adversarial Approach vs. Adversarial Examples

The approach of GANs is called adversarial since the two networks have antagonistic objectives.

This is not to be confused with adversarial examples in machine learning.

See these two papers for more details:
`https://arxiv.org/pdf/1412.6572.pdf`
`https://arxiv.org/pdf/1312.6199.pdf`

Generative adversarial networks (conceptual)



[1]

[1] Credit: Xavier Giro-i-Nieto

More Formally

- The **generator**

$$\mathbf{G} : \mathbb{R}^D \rightarrow X \tag{1}$$

  is trained so that it gets a random input and produces a sample following the data distribution as output (ideally).

- The **discriminator**

$$\mathbf{D} : X \rightarrow [0, 1] \tag{2}$$

  gets a sample as input and predicts if it is real or fake.

More Practically

- Training a standard GAN is difficult and often results in two undesirable behaviors
  1. Oscillations without convergence. Contrary to standard loss minimization, we have no guarantee that the loss will actually decrease.
  2. The **mode collapse** problem, when the generator models very well a small sub-population, concentrating on a few modes.
- Additionally, performance is hard to assess and often boils down to heuristic observations.

Deep Convolutional Generative Adversarial Networks (DCGANs)

- Motivates the use of GANS to learn reusable feature representations from large unlabeled datasets.

Deep Convolutional Generative Adversarial Networks (DCGANs)

- Motivates the use of GANS to learn reusable feature representations from large unlabeled datasets.
- GANs known to be unstable to train, often resulting in generators that produce "nonsensical outputs".

Deep Convolutional Generative Adversarial Networks (DCGANs)

- Motivates the use of GANS to learn reusable feature representations from large unlabeled datasets.
- GANs known to be unstable to train, often resulting in generators that produce "nonsensical outputs".
- Extensive model exploration to identify a family of architectures that result in **stable** training across a range of datasets and allowed for training higher resolution and deeper models.

Architecture Guildelines for Stable DCGAN

- Replace pooling layers with strided convolutions in the discriminator and fractional-strided convolutions in the generator.

Architecture Guildelines for Stable DCGAN

- Replace pooling layers with strided convolutions in the discriminator and fractional-strided convolutions in the generator. This will allow the network to learn its own spatial downsampling.

Architecture Guildelines for Stable DCGAN

- Replace pooling layers with strided convolutions in the discriminator and fractional-strided convolutions in the generator. This will allow the network to learn its own spatial downsampling.
- Use batchnorm in both the generator and the discriminator.

Architecture Guildelines for Stable DCGAN

- Replace pooling layers with strided convolutions in the discriminator and fractional-strided convolutions in the generator. This will allow the network to learn its own spatial downsampling.
- Use batchnorm in both the generator and the discriminator. This helps deal with training problems due to poor initialisation and helps the gradient flow.

Architecture Guildelines for Stable DCGAN

- Replace pooling layers with strided convolutions in the discriminator and fractional-strided convolutions in the generator. This will allow the network to learn its own spatial downsampling.
- Use batchnorm in both the generator and the discriminator. This helps deal with training problems due to poor initialisation and helps the gradient flow.
- Eliminate fully connected hidden layers for deeper architectures.

Architecture Guildelines for Stable DCGAN

- Replace pooling layers with strided convolutions in the discriminator and fractional-strided convolutions in the generator. This will allow the network to learn its own spatial downsampling.
- Use batchnorm in both the generator and the discriminator. This helps deal with training problems due to poor initialisation and helps the gradient flow.
- Eliminate fully connected hidden layers for deeper architectures.
- Use ReLU activation in the generator for all layers except for the output, which uses Tanh.

Architecture Guildelines for Stable DCGAN

- Replace pooling layers with strided convolutions in the discriminator and fractional-strided convolutions in the generator. This will allow the network to learn its own spatial downsampling.
- Use batchnorm in both the generator and the discriminator. This helps deal with training problems due to poor initialisation and helps the gradient flow.
- Eliminate fully connected hidden layers for deeper architectures.
- Use ReLU activation in the generator for all layers except for the output, which uses Tanh.
- Use LeakyReLU activation in the discriminator for all layers.