Generative Adversarial Nets

Author: Goodfellow et al.

created by paper2slides

Date: 2014-06-10

https://arxiv.org/abs/1406.2661

Executive Summary

- The paper introduces Generative Adversarial Networks (GANs), a framework for estimating generative models via an adversarial process.
- Two models are trained simultaneously: a generative model G and a discriminative model D.
- G aims to generate data indistinguishable from real data, while D attempts to distinguish between real and generated data.
- The framework corresponds to a minimax two-player game.

Introduction

- Deep learning has seen success primarily in discriminative models.
- Generative models face difficulties due to intractable probabilistic computations.
- The proposed adversarial nets framework sidesteps these difficulties.
- GANs train via backpropagation and avoid Markov chains or approximate inference during training and sample generation.

Adversarial Modeling Framework

- Generative model G maps noise variables $\boldsymbol{z}$ to data space via a neural network.
- Discriminative model D outputs the probability that a sample came from the real data rather than G.
- Training involves a minimax game:

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

- D is trained to maximize classification accuracy, while G is trained to fool D.

Theoretical Results

- Global Optimality: The global minimum of the training criterion is reached if and only if $p_g = p_{\text{data}}$.
- Proposition: For any fixed G, the optimal discriminator D is:

$$D_G^*(\boldsymbol{x}) = \frac{p_{\text{data}}(\boldsymbol{x})}{p_{\text{data}}(\boldsymbol{x}) + p_g(\boldsymbol{x})}$$

- G converges to p_{data} if D is optimal at each step and G is updated gradually.

Algorithm for Training GANs

- Training involves iterative updates:
  - Update D for k steps to maximize $\log D(\boldsymbol{x}) + \log(1 - D(G(\boldsymbol{z})))$.
  - Update G to minimize $\log(1 - D(G(\boldsymbol{z})))$.
- Usually, $k = 1$ to alternate updates efficiently.
- Practical implementation avoids reliance on Markov chains or approximate inference.

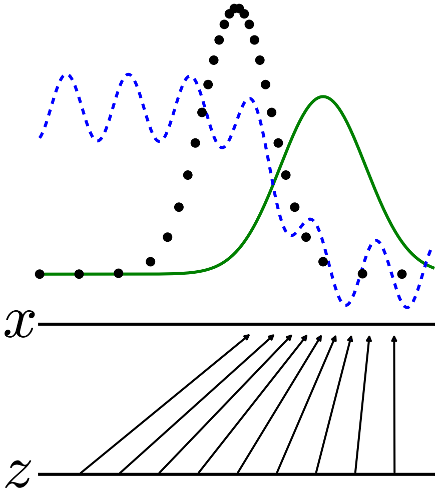

Figure: Training involves G generating data to fool D and D distinguishing real versus generated data.

Experimental Results

- Experiments conducted on MNIST, TFD, and CIFAR-10 datasets.
- Use of rectifier linear units and maxout units in G and D respectively.
- Dropout used in training D for regularization.

Quantitative Results

Model	MNIST	TFD
DBN	138 ± 2	1909 ± 66
Stacked CAE	121 ± 1.6	$\mathbf{2110 \pm 50}$
Deep GSN	214 ± 1.1	1890 ± 29
Adversarial nets	$\mathbf{225 \pm 2}$	$\mathbf{2057 \pm 26}$

Table: Parzen window-based log-likelihood estimates showing competitive results of GANs.

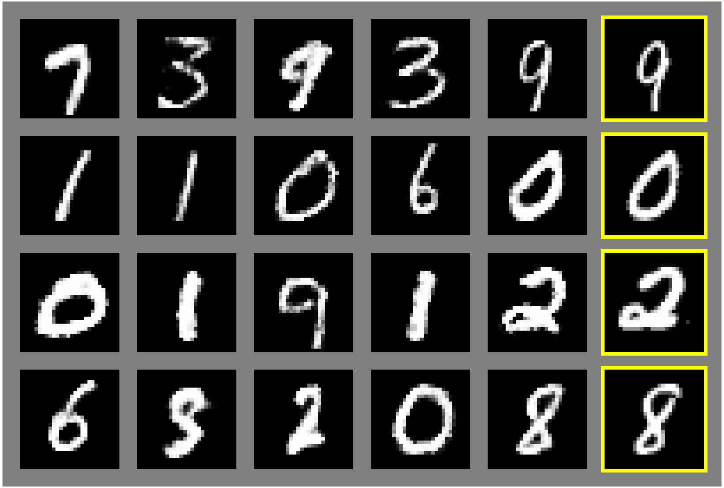

Figure: Samples from GAN on MNIST dataset.

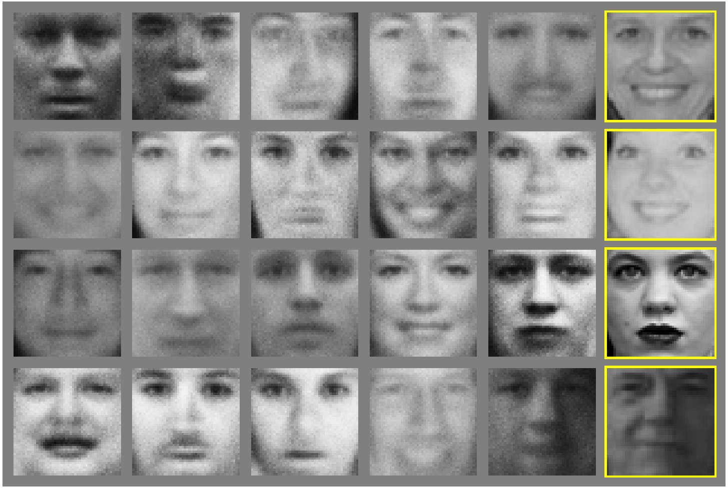

Figure: Samples from GAN on TFD dataset.

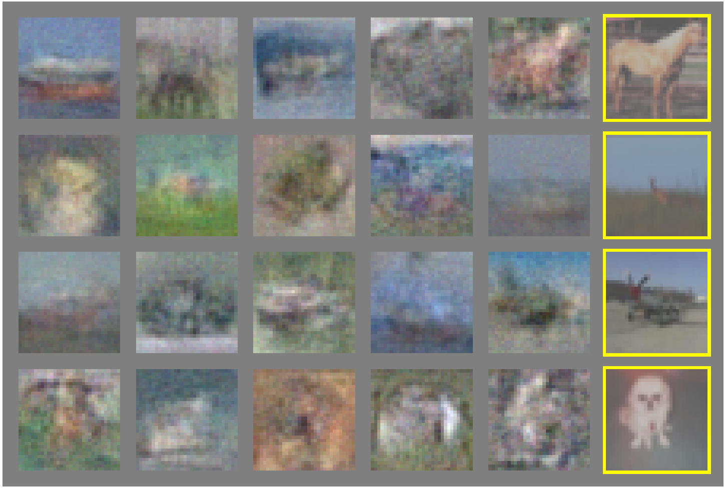

Figure: Generated samples on CIFAR-10 using fully connected model.

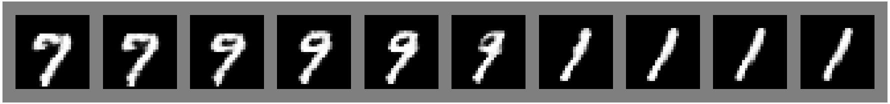

Figure: Interpolations between generated digits by linearly interpolating in the latent space.

Advantages and Disadvantages

- Advantages:
  - No Markov chains required.
  - Backpropagation sufficient for training.
  - No approximate inference needed.
- Disadvantages:
  - No explicit density estimation.
  - Requires careful synchronization between G and D.

Conclusions and Future Work

- GANs provide a new way to train generative models through adversarial learning.
- Potential extensions include conditional adversarial nets, semi-supervised learning, and improved training stability.
- Future work can explore more effective ways to synchronize G and D and other model architectures.