Generative Adversarial Nets

Ian J. Goodfellow et al.

Speaker : Semin Choi

July 10, 2015

Introduction

- The most striking successes in deep learning have involved discriminative models.
- Deep generative models
  - the difficulty of approximating many intractable probabilistic computations.
  - the difficulty of leveraging the benefits of piecewise linear units in the generative context.
- We propose a new generative model estimation procedure that sidesteps these difficulties.

Introduction

- *Adversarial nets* framework
  - A discriminative model learns to determine whether a sample is from the model distribution or the data distribution.
  - The generative model can be thought of as analogous to a team of counterfeiters, trying to produce fake currency and use it without detection.
  - The discriminative model is analogous to the police, trying to detect the counterfeit currency.
- *Adversarial nets*
  - The generative model generates samples by passing random noise through a multilayer perceptron.
  - The discriminative model is also a multilayer perceptron.

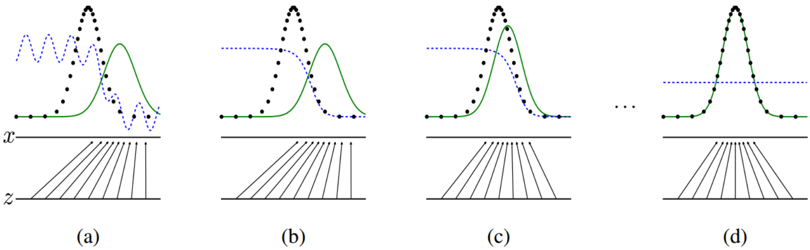
Adversarial Nets

- $p_{\mathbf{z}}(\mathbf{z})$: the distribution of input noise variables
- $G(\mathbf{z}; \theta_g)$: a mapping to data space
  - G is a differentiable function represented by a multilayer perceptron with parameters θ_g.
- $p_g(\mathbf{x})$: the generator's distribution
- $D(\mathbf{x}, \theta_d)$: the probability that $\mathbf{x}$ came from the data rather than p_g.
- We train D to maximize the probability of assigning the correct label to both training examples and samples from G.
- We simultaneously train G to minimize $\log(1 - D(G(\mathbf{z})))$.

Adversarial Nets

- In other words, D and G play the following two-player minimax game with value function $V(G, D)$
- $\text{argmin}_G \max_D V(G, D)$ where

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



Adversarial Nets

Minibatch stochastic gradient descent training of generative adversarial nets

for number of training iteration **do**

for k steps **do**

Sample minibatch of m noise samples $\{z^{(1)}, ..., z^{(m)}\}$ from $p_{\mathbf{z}}(\mathbf{z})$.

Sample minibatch of m examples $\{x^{(1)}, ..., x^{(m)}\}$ from $p_{\text{data}}(\mathbf{x})$.

Update the discriminator by ascending its stochastic gradient:

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

end for

Sample minibatch of m noise samples $\{z^{(1)}, ..., z^{(m)}\}$ from $p_{\mathbf{z}}(\mathbf{z})$.

Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log(1 - D(G(x^{(i)})))$$

end for

Theoretical Results

Proposition 1.

For G fixed, the optimal discriminator D is

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

Theorem 1.

The global minimum of the virtual training criterion $C(G) = \max_D V(G, D)$ is achieved if and only if $p_g = p_{\text{data}}$.

- In practice, adversarial nets represent a limited family of p_g distributions via the function $G(\mathbf{z}; \theta_g)$, and we optimize θ_g rather than p_g itself.

Experiments

Parzen window-based log-likelihood estimates

Model	MNIST	TFD
DBN [3]	138 ± 2	1909 ± 66
Stacked CAE [3]	121 ± 1.6	$\mathbf{2110 \pm 50}$
Deep GSN [6]	214 ± 1.1	1890 ± 29
Adversarial nets	$\mathbf{225 \pm 2}$	$\mathbf{2057 \pm 26}$

Experiments

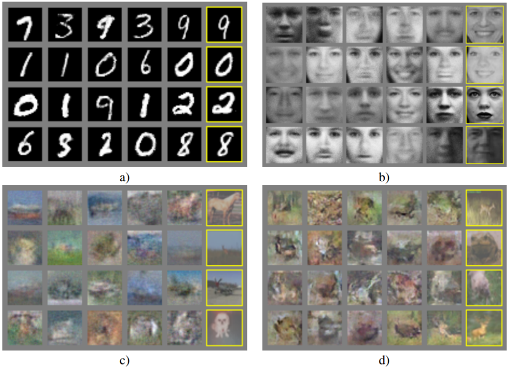


Figure 3: Digits obtained by linearly interpolating between coordinates in z space of the full model.