

CS 4824/ECE 4424: Generative Adversarial Networks

Acknowledgement:

Many of these slides are derived from Tom Mitchell, Pascal Poupart, Pieter Abbeel, Eric Eaton, Carlos Guestrin, William Cohen, and Andrew Moore.

Generative networks

- Neural networks are typically used for classification or regression
 - Input: data
 - Output: class or prediction
- Can we design neural networks that can generate data?
 - Input: random vector
 - Output: data

Generative networks

- Several types of generative networks
 - Boltzmann machines
 - Sigmoid belief networks
 - Variational autoencoders
 - **Generative adversarial networks**
 - Generative moment matching networks
 - Sum-product networks
 - Normalizing flows
 - ...

Generative Adversarial Networks

- Approach based on game theory
- Two networks:
 - Generator $g(\mathbf{z}; \mathbf{W}_g) \rightarrow \mathbf{x}$
 - Discriminator $d(\mathbf{x}; \mathbf{W}_d) \rightarrow \Pr(\mathbf{x} \text{ is real})$
- Objective

Generative Adversarial Networks

- Approach based on game theory
- Two networks:
 - Generator $g(\mathbf{z}; \mathbf{W}_g) \rightarrow \mathbf{x}$
 - Discriminator $d(\mathbf{x}; \mathbf{W}_d) \rightarrow \Pr(\mathbf{x} \text{ is real})$
- Objective

$$\begin{aligned} & \min_{\mathbf{W}_g} \max_{\mathbf{W}_d} \sum_n \log \Pr(\mathbf{x}_n \text{ is real}; \mathbf{W}_d) + \log \Pr(g(\mathbf{z}_n; \mathbf{W}_g) \text{ is fake}; \mathbf{W}_d) \\ & \equiv \min_{\mathbf{W}_g} \max_{\mathbf{W}_d} \sum_n \log d(\mathbf{x}_n; \mathbf{W}_d) + \log (1 - d(g(\mathbf{z}_n; \mathbf{W}_g); \mathbf{W}_d)) \end{aligned}$$

Generative Adversarial Networks

- Schematic

GAN training

- We have a min-max optimization
 - Optimize the discriminator by stochastic gradient ascent
 - Optimize the generator by stochastic gradient descent

GAN training

- Repeat until convergence
 - For k steps do
 - Sample $\mathbf{z}_1, \dots, \mathbf{z}_N$ from $\Pr(\mathbf{z})$
 - Sample $\mathbf{x}_1, \dots, \mathbf{x}_N$ from training set
 - Update discriminator by ascending its stochastic gradient

$$\nabla_{\mathbf{W}_d} \left(\frac{1}{N} \sum_{n=1}^N \left[\log d(\mathbf{x}_n; \mathbf{W}_d) + \log (1 - d(g(\mathbf{z}_n; \mathbf{W}_g); \mathbf{W}_d)) \right] \right)$$

- Sample $\mathbf{z}_1, \dots, \mathbf{z}_N$ from $\Pr(\mathbf{z})$
- Update generator by descending its stochastic gradient

$$\nabla_{\mathbf{W}_g} \left(\frac{1}{N} \sum_{n=1}^N \log (1 - d(g(\mathbf{z}_n; \mathbf{W}_g); \mathbf{W}_d)) \right)$$

GAN training

- In the limit (with sufficiently expressive networks, sufficient data and global convergence)

$$\Pr(\mathbf{x}|\mathbf{z}; \mathbf{W}_g) \rightarrow \text{true data distribution}$$

$$\Pr(\mathbf{x} \text{ is real}; \mathbf{W}_d) \rightarrow 0.5 \text{ (for real and fake data)}$$

- Problems in practice:
 - Imbalance: one network may dominate the other
 - Local convergence

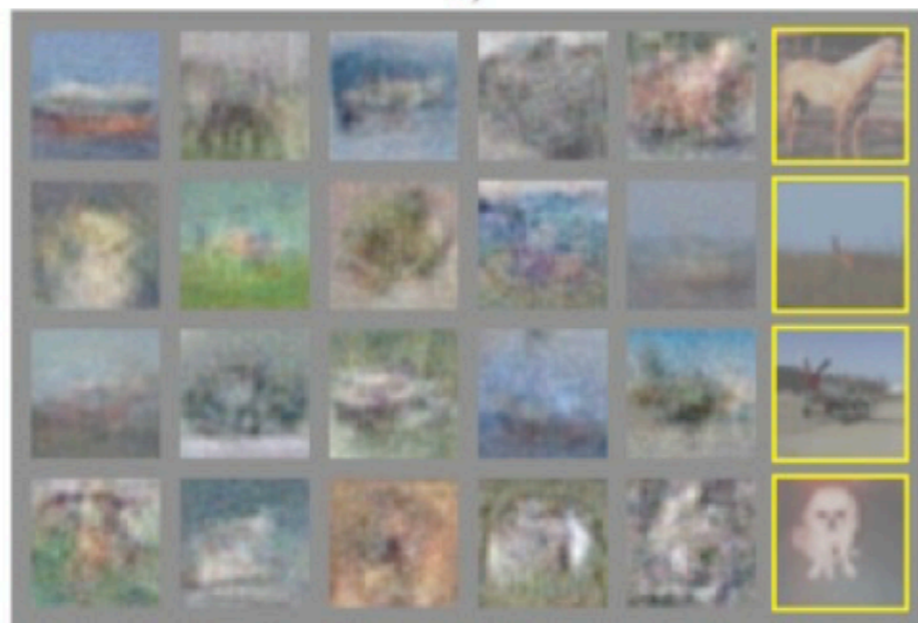
Images generated with GANs training



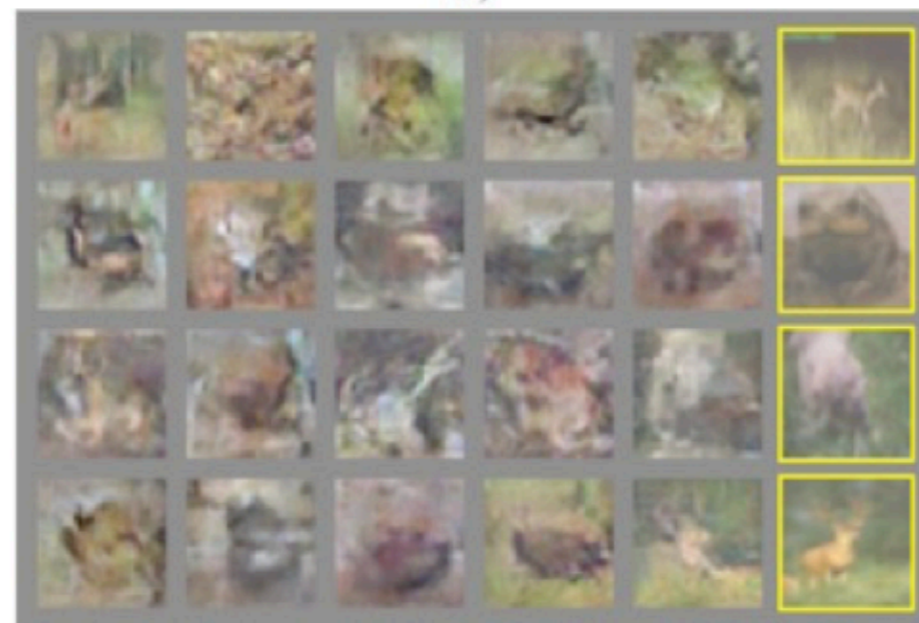
a)



b)



c)



d)

- Right columns are nearest neighbour training examples of adjacent column