TBSI 清华-伯克利深圳学院 Tsinghua-Berkeley Shenzhen Institute

Generative Adversarial Nets

Group 3

Yihan Zhang, Guangyuan Ma, Yuou Chen (Arrange in reporting order)

23rd Apr. 2024

Overview

- a generative model G that captures the data distribution
- a discriminative model D that estimates the probability that a sample came from the training data rather than G.

- maximize the probability of D making a mistake

Previous work – VAE

- Discriminative model **vs** the encoder
  - Input is the same
  - Output of Discriminative model is the probability of authenticity
  - Output of the encoder is latent variables

- Generative model **vs** recognition model (decoder)
  - Input of Generative model is random noise
  - Input of recognition model is latent variables
  - Output is the same

Previous work: pm

(predictability minimization)

- Training criterion
  - In GANs, the competition is the sole training criterion
  - In PM, the competition between the networks is not the primary training criterion but rather a regularization technique
- Nature of Competition
- Learning Process Specification
  - PM – optimization problem
  - GAN – minimax game

PROS

- No inference needed during learning
- Deep graphical models
  - Directed – Bayesian networks (Belief network)
  - Undirected – Markov random fields

- GANs do not require explicit inference during training
- Train generator and the discriminator simultaneously in a minimax game

PROS

- Markov chains are never needed

- No need for explicit probability density estimation
- Potential for better generalization
- Parallel and efficient sampling

CONS

- no explicit representation of generative distribution

- D must be synchronized well with G during training
  - Helvetica scenario – a font where all characters look very similar
  - G is trained too much without updating D →
  - maps too many different values of the random noise vector (z) to the same output sample (x) →
  - not able to produce a diverse set of samples that can accurately model the real data distribution

Algorithm 1

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

Ideal Discriminator: $D(\boldsymbol{x})=1,\ D(G(\boldsymbol{z}))=0$

Nonideal Discriminator: $D(\boldsymbol{x}),\ D(G(\boldsymbol{z}))\in[0,1]$

$\max_D$ is to find the optimal discriminator between given $G(\boldsymbol{z})$ and $\boldsymbol{x}$:

$D(\boldsymbol{x})\uparrow \ \&\ \to 1, \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})]\uparrow$

$D(G(\boldsymbol{z}))\downarrow \ \&\ \to 0,\ (1-D(G(\boldsymbol{z}))\uparrow \ \&\ \to 1,\ \mathbb{E}_{\boldsymbol{z}\sim p_z(\boldsymbol{z})}[\log(1-D(G(\boldsymbol{z})))]\uparrow \ \&\ \to 0$

$\min_G$ is to make $G(\boldsymbol{z})$ most similar to $\boldsymbol{x}$ for given $D(x)$:

$D(G(\boldsymbol{z}))\uparrow \ \&\ \to 1,\ (1-D(G(\boldsymbol{z}))\uparrow \ \&\ \to 0,\ \mathbb{E}_{\boldsymbol{z}\sim p_z(\boldsymbol{z})}[\log(1-D(G(\boldsymbol{z})))]\downarrow \ \&\ \to -\infty$

Algorithm 1

Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k, is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

for number of training iterations **do**

  for k steps **do**

  - Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
  - Sample minibatch of m examples $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$.
  - Update the discriminator by ascending its stochastic gradient:

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

  end for

  - Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
  - Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$$

end for

The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

Algorithm 1

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

Ideal Discriminator: $D(\boldsymbol{x})=1,\ D(G(\boldsymbol{z}))=0$

Nonideal Discriminator: $D(\boldsymbol{x}),\ D(G(\boldsymbol{z}))\in[0,1]$

$\max_D$ is to find the optimal discriminator between given $G(\boldsymbol{z})$ and $\boldsymbol{x}$:

Any theoretical property?

$D(\boldsymbol{x})\uparrow \ \&\ \to 1,\ \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})]\uparrow$

$D(G(\boldsymbol{z}))\downarrow \ \&\ \to 0,\ (1-D(G(\boldsymbol{z}))\uparrow \ \&\ \to 1,\ \mathbb{E}_{\boldsymbol{z}\sim p_z(\boldsymbol{z})}[\log(1-D(G(\boldsymbol{z})))]\uparrow \ \&\ \to 0$

$\min_G$ is to make $G(\boldsymbol{z})$ most similar to $\boldsymbol{x}$ for given $D(x)$:

$D(G(\boldsymbol{z}))\uparrow \ \&\ \to 1,\ (1-D(G(\boldsymbol{z}))\uparrow \ \&\ \to 0,\ \mathbb{E}_{\boldsymbol{z}\sim p_z(\boldsymbol{z})}[\log(1-D(G(\boldsymbol{z})))]\downarrow \ \&\ \to -\infty$

Optimal Discriminator

Proposition 1 For given $G(\boldsymbol{z})$, the optimal discriminator $D(x)$ is:

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

Proof:

$$\begin{aligned} V(D,G) &= \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_z(\boldsymbol{z})}[\log(1-D(G(\boldsymbol{z})))] \\ &= \int_{\boldsymbol{x}} p_{\text{data}}(\boldsymbol{x})\log(D(\boldsymbol{x})) + p_g(\boldsymbol{x})\log(1-D(\boldsymbol{x}))\,d\boldsymbol{x} \\ \frac{\partial V(D,G)}{\partial D} &= \int_{\boldsymbol{x}} \frac{p_{\text{data}}(\boldsymbol{x})}{D\ln a} - \frac{p_g(\boldsymbol{x})}{(1-D)\ln a}\,d\boldsymbol{x} \end{aligned}$$

$$\max_D V(D,G) \implies \frac{\partial V(D,G)}{\partial D} = 0 \implies D = \frac{p_{data}(\boldsymbol{x})}{p_{data}(\boldsymbol{x}) + p_g(\boldsymbol{x})} = D_G^*(\boldsymbol{x})$$

Algorithm 1

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

Ideal Discriminator: $D(\boldsymbol{x})=1,\ D(G(\boldsymbol{z}))=0$

Nonideal Discriminator: $D(\boldsymbol{x}),\ D(G(\boldsymbol{z}))\in[0,1]$

$\max_D$ is to find the optimal discriminator between given $G(\boldsymbol{z})$ and $\boldsymbol{x}$:

$D(\boldsymbol{x})\uparrow \ \&\ \to 1, \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})]\uparrow$

$D(G(\boldsymbol{z}))\downarrow \ \&\ \to 0,\ (1-D(G(\boldsymbol{z}))\uparrow \ \&\ \to 1,\ \mathbb{E}_{\boldsymbol{z}\sim p_z(\boldsymbol{z})}[\log(1-D(G(\boldsymbol{z})))]\uparrow \ \&\ \to 0$

$\min_G$ is to make $G(\boldsymbol{z})$ most similar to $\boldsymbol{x}$ for given $D(x)$:

Any theoretical property?

$D(G(\boldsymbol{z}))\uparrow \ \&\ \to 1,\ (1-D(G(\boldsymbol{z}))\uparrow \ \&\ \to 0,\ \mathbb{E}_{\boldsymbol{z}\sim p_z(\boldsymbol{z})}[\log(1-D(G(\boldsymbol{z})))]\downarrow \ \&\ \to -\infty$

Optimal Generator

Proposition 2 For given optimal discriminator $D_G^*(x)$, the optimal generator is:

$$p_g(\boldsymbol{x}) = p_{data}(\boldsymbol{x})$$

Theorem 1 $\min_G \max_D V(D,G) = -\log 4$

Optimal achieves $\Leftrightarrow$ $p_g(\boldsymbol{x}) = p_{data}(\boldsymbol{x})$ and $D_G^*(\boldsymbol{x}) = \dfrac{p_{data}(\boldsymbol{x})}{p_{data}(\boldsymbol{x}) + p_g(\boldsymbol{x})}$

Optimal Generator

$$
\begin{aligned}
\textit{Proof}: \min_G \max_D V(D,G) &= \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_z}[\log(1-D(G(\boldsymbol{z})))] \\
&= \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}}[\log D_G^*(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_z}[\log(1-D_G^*(G(\boldsymbol{z})))] \\
&= \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}}[\log D_G^*(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{x}\sim p_g}[\log(1-D_G^*(\boldsymbol{x}))] \\
&= \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}}\left[\log \frac{p_{\text{data}}(\boldsymbol{x})}{P_{\text{data}}(\boldsymbol{x})+p_g(\boldsymbol{x})}\right] + \mathbb{E}_{\boldsymbol{x}\sim p_g}\left[\log \frac{p_g(\boldsymbol{x})}{p_{\text{data}}(\boldsymbol{x})+p_g(\boldsymbol{x})}\right] \\
&= -\log(4) + KL\left(p_{\text{data}} \,\|\, \frac{p_{\text{data}}+p_g}{2}\right) + KL\left(p_g \,\|\, \frac{p_{\text{data}}+p_g}{2}\right) \\
&= -\log(4) + 2\cdot JSD(p_{\text{data}} \,\|\, p_g)
\end{aligned}
$$

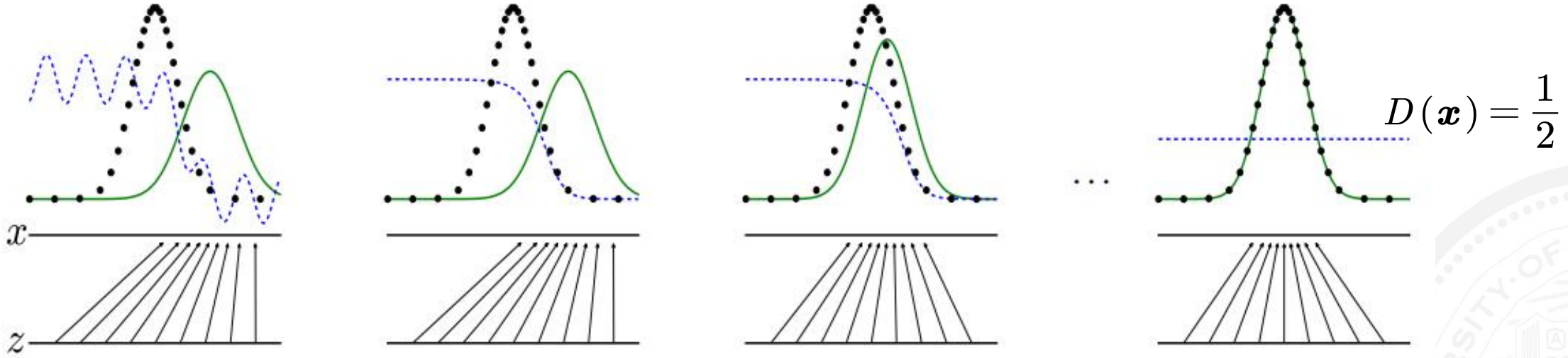
$$
KL(a\,\|\,b)\geqslant 0,\ JSD(a\,\|\,b)\geqslant 0,\ \text{zero holds when } a=b
$$

$$
\Longrightarrow p_g(\boldsymbol{x}) = p_{data}(\boldsymbol{x}),\ D_G^*(\boldsymbol{x}) = 1/2
$$

Intuitive explanation

$$p_g(\boldsymbol{x}) = p_{data}(\boldsymbol{x})$$

$$D(\boldsymbol{x}) = \frac{1}{2}$$



(a) (b) (c) (d)

An case near convergence

p_g close to p_{data}

D close to optimal classifier

D updated by

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

G updated by gradient

Optimal

$D(\boldsymbol{x})$, blue, dashed line $p_g(\boldsymbol{x})$, green, solidline $p_{data}(\boldsymbol{x})$, black, dotted line

Early stage problem

When G is poor, $D(G(z)) \to 0$, $\log(1 - D(G(z)))$ saturates.

We can train G *by* $\max_G \log D(G(z))$ rather than $\min_G \log(1 - D(G(z)))$, as the previous way provides larger gradient.

Experiments

The adversarial nets are trained in a definite method.

Datasets

MNIST, the Toronto Face Database (TFD) and CIFAR-10

Generator

- Activated by a mixture of rectifier linear activations and sigmoid activations
- Noise as the input to only the bottommost layer of the generator network

Discriminator

- Activated by maxout activations
- Dropout applied in the training process

a minimax two-player game

Probability Estimation

- A Gaussian Parzen window is fitted to the generated samples.
- σ of the Gaussians is obtained by cross validation on the validation set.
- The log-likelihood is calculated and reported.

Results

Adversarial nets are compared to other existing methods.

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	$\mathbf{225 \pm 2}$	$\mathbf{2057 \pm 26}$

Table 1. Parzen window-based log-likelihood estimates.

For MNIST the mean loglikelihood of samples on test set were calculated with the standard error of the mean across examples.

For TFD the standard error across folds were calculated with a different σ chosen using the validation set of each fold.

Key ideas:

- Adversarial nets show better outcomes in general.
- This method of likelihood estimation has somewhat high variance and does not perform well in high dimensional spaces but it is the best method available.
- Advances in generative models that can sample but not estimate likelihood directly motivate further research into how to evaluate such models.

Results

Visualization of generated samples reflects the model performance.

Figure 2

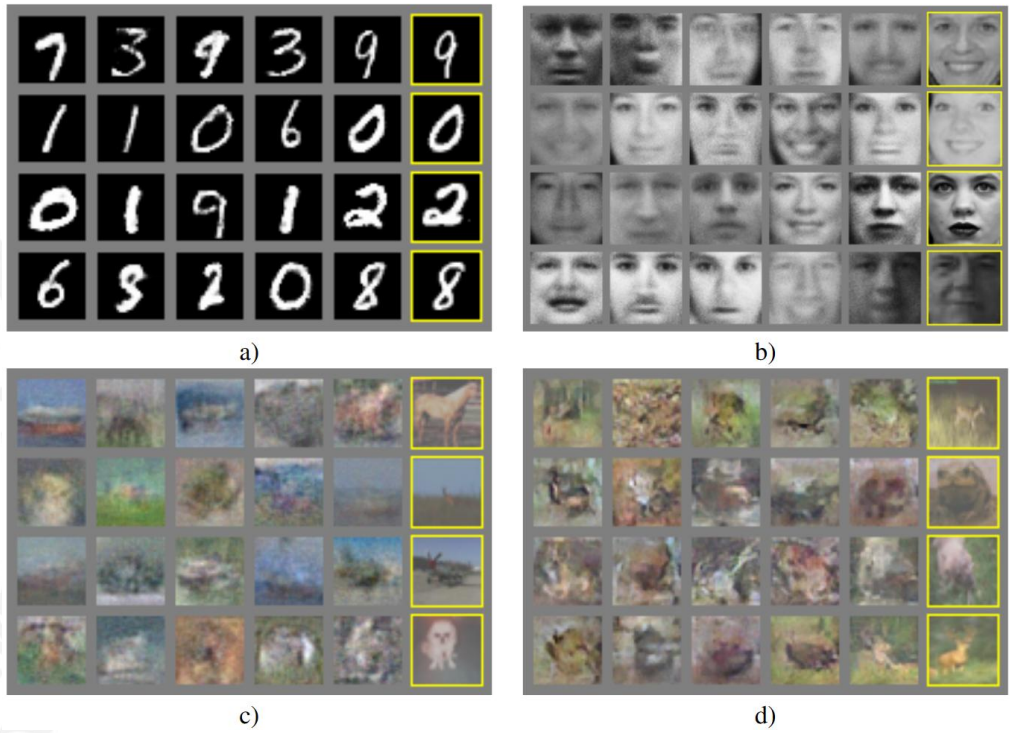


Figure 3

Figure 2. Visualization of samples from the model.

Rightmost column shows the nearest training example of the neighboring sample. Samples are fair random draws, not cherry-picked.

(a) MNIST (b) TFD (c) CIFAR-10 (fully connected model) (d) CIFAR-10 (convolutional discriminator and “deconvolutional” generator).

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

Key ideas:

- Unlike most other visualizations of deep generative models, they are actual samples from the model distributions, not conditional means given samples of hidden units.
- These samples are uncorrelated because the sampling process does not depend on Markov chain mixing.
- The results are at least competitive with the better generative models in the literature and highlight the potential of the adversarial framework.

Advantages and disadvantages

Adversarial nets come with advantages and disadvantages.

Advantages

1. Computational:

- ✓ Unnecessary Markov chains
- ✓ Only backprop for obtaining gradients
- ✓ Unnecessary inference
- ✓ Various functions permitted

2. Statistical:

- ✓ Not being updated directly with data examples, but only with gradients flowing through the discriminator
- ✓ Capable of representing very sharp, even degenerate distributions

Disadvantages

1. No explicit representation of $p_g(x)$

2. *D* must be synchronized well with *G* during training

In particular

G* must not trained too much without updating *D, in order to avoid "the Helvetica scenario" in which *G* collapses too many values of z to the same value of x to have enough diversity to model p_{data}, much as the negative chains of a Boltzmann machine must be kept up to date between learning steps.

Comparison

This new framework exhibits advantages and disadvantages relative to previous modeling frameworks.

	Deep directed graphical models	Deep undirected graphical models	Generative autoencoders	Adversarial models
Training	Inference needed during training.	Inference needed during training. MCMC needed to approximate partition function gradient.	Enforced tradeoff between mixing and power of reconstruction generation	Synchronizing the discriminator with the generator. Helvetica.
Inference	Learned approximate inference	Variational inference	MCMC-based inference	Learned approximate inference
Sampling	No difficulties	Requires Markov chain	Requires Markov chain	No difficulties
Evaluating $p(x)$	Intractable, may be approximated with AIS	Intractable, may be approximated with AIS	Not explicitly represented, may be approximated with Parzen density estimation	Not explicitly represented, may be approximated with Parzen density estimation
Model design	Nearly all models incur extreme difficulty	Careful design needed to ensure multiple properties	Any differentiable function is theoretically permitted	Any differentiable function is theoretically permitted

Table 2. Challenges in generative modeling.

A summary of the difficulties encountered by different approaches to deep generative modeling for each of the major operations involving a model.

Key ideas:

- Adversarial nets are superior to other existing methods in inference, sampling and model design.
- Adversarial nets still exhibit drawbacks in the training process, in terms of synchronization and Helvetica.

Conclusions and future work

Adversarial nets admit many straightforward extensions.

Conclusions

A new framework for **estimating generative models via an adversarial process** is proposed with two simultaneously trained models: **a generative model *G*** and **a discriminative model *D***. This framework corresponds to a **minimax two-player game**. There is no need for any Markov chains or unrolled approximate inference networks during either training or generation of samples.

Future work

1. Conditional generative model

$p(x|c)$ can be obtained by adding c as input to both G and D.

2. Learned approximate inference

An auxiliary network can be trained to predict z given x with a fixed well-trained generator net.

3. Approximate model for all conditions

A family of conditional models can be trained with shared parameters.

4. Semi-supervised learning

Features from the discriminator or inference net could improve performance of classifiers when limited labeled data is available.

5. Efficiency improvements

Training could be accelerated greatly by divising better methods for coordinating G and D or determining better distributions to sample z.

Thanks for listening!

Group 3

23rd Apr. 2024