# Generative Adversarial Nets

Changzhi Sun

East China Normal University

*changzhisun@stu.ecnu.edu.cn*

December 15, 2016

# Outline

# Introduction

1. deep learning successes in discriminative models
    - map a high-dimensional, rich sensory input to a class label
    - backpropagation, dropout, piecewise linear units
    - well-behaved gradient
2. deep generative models (less of an impact)
    - approximating many intractable probabilistic computations
    - difficulty of leveraging the benefits of piecewise linear units
3. adversarial nets can sidesteps these difficulties

# Introduction

1. a generative model G
    - captures the data distribution
2. a discriminative model D
    - estimates the probability that a sample came from the training data rather than G
3. train objective
    - G: maximize the probability of D making a mistake
    - D: classification objective
4. minimax two-player game
5. unique solution exists (G $= P_{data}$, D $= \frac{1}{2}$)
6. G and D are multilayer perceptrons
    - without Markov chains
    - without unrolled approximate inference network
    - either training or generation of samples

# Related Work

1. parametric specification of a probability distribution function
   - deep Boltzmann machine
   - intractable likelihood functions (numerous approximations to the likelihood gradient)
2. generative machines
   - do not explicitly represent the likelihood
   - generate samples from the desired distribution
   - Generative stochastic networks
   - extends generative machine by eliminating the Markov chains
3. using a discriminative criterion to train a generative model
   - intractable for deep generative models
   - ratios of probabilities can't be approximated using variational approximations that lower bound the probability
   - Noise-contrastive estimation (NCE)
     - informal competition mechanism

# Related Work

1. parametric specification of a probability distribution function
2. generative machines
3. using a discriminative criterion to train a generative model
4. predictability minimization (two neural networks)
    - training criterion is different
    - the nature of the competition is different
    - the specification of the learning process is different
        - optimization problem and minimax game
        - terminates at a saddle point
5. adversarial examples

# Adversarial Nets

## Def

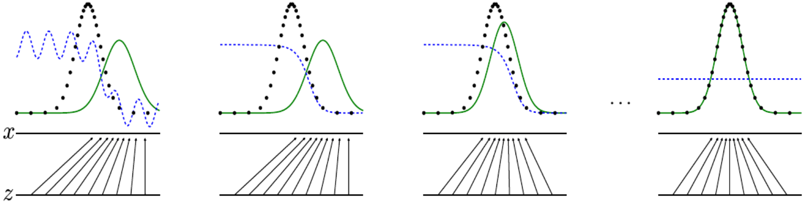
- $p_g$: generators distribution (over data $\mathbf{x}$)
- $p_{\mathbf{z}}(\mathbf{z})$: a prior on input noise variables
- $G(\mathbf{z}; \theta_g)$: mapping to data space
- $D(\mathbf{x}; \theta_d)$: the prob $\mathbf{x}$ came from the data rather than $p_g$

## Train Objective

1. D: maximize the prob of samples
2. G: minimize $log(1 - D(G(\mathbf{z})))$
3. value function

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

# Adversarial Nets




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

# Adversarial Nets

---

**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 $\{\boldsymbol{z}^{(1)}, \dots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
  - Sample minibatch of $m$ examples $\{\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(m)}\}$ 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 $\{\boldsymbol{z}^{(1)}, \dots, \boldsymbol{z}^{(m)}\}$ 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.

---

1. Given enough capacity and training time
2. Global Optimality of $p_g = p_{\mathbf{data}}$
3. Convergence of Algorithm 1

# Global Optimality of $p_g = p_{\text{data}}$

**Proposition 1.** *For $G$ fixed, the optimal discriminator $D$ is*

$$D_G^*(\boldsymbol{x}) = \frac{p_{data}(\boldsymbol{x})}{p_{data}(\boldsymbol{x}) + p_g(\boldsymbol{x})} \tag{2}$$

# Global Optimality of $p_g = p_{\text{data}}$

*Proof.* The training criterion for the discriminator D, given any generator $G$, is to maximize the quantity $V(G, D)$

$$
\begin{aligned}
V(G,D) &= \int_{\boldsymbol{x}} p_{\text{data}}(\boldsymbol{x}) \log(D(\boldsymbol{x}))dx + \int_{z} p_{\boldsymbol{z}}(\boldsymbol{z}) \log(1 - D(g(\boldsymbol{z})))dz \\
&= \int_{\boldsymbol{x}} p_{\text{data}}(\boldsymbol{x}) \log(D(\boldsymbol{x})) + p_g(\boldsymbol{x}) \log(1 - D(\boldsymbol{x}))dx \qquad (3)
\end{aligned}
$$

For any $(a, b) \in \mathbb{R}^2 \setminus \{0, 0\}$, the function $y \to a\log(y) + b\log(1-y)$ achieves its maximum in $[0, 1]$ at $\frac{a}{a+b}$. The discriminator does not need to be defined outside of $Supp(p_{\text{data}}) \cup Supp(p_g)$, concluding the proof. $\square$

Note that the training objective for $D$ can be interpreted as maximizing the log-likelihood for estimating the conditional probability $P(Y = y|\boldsymbol{x})$, where $Y$ indicates whether $\boldsymbol{x}$ comes from $p_{\text{data}}$ (with $y = 1$) or from $p_g$ (with $y = 0$). The minimax game in Eq. 1 can now be reformulated as:

$$
\begin{aligned}
C(G) =& \max_D V(G, D) \\
=& \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}}[\log D^*_G(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}}[\log(1 - D^*_G(G(\boldsymbol{z})))] \qquad (4) \\
=& \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}}[\log D^*_G(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{x}\sim p_g}[\log(1 - D^*_G(\boldsymbol{x}))] \\
=& \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}}\left[\log \frac{p_{\text{data}}(\boldsymbol{x})}{P_{\text{data}}(\boldsymbol{x}) + p_g(\boldsymbol{x})}\right] + \mathbb{E}_{\boldsymbol{x}\sim p_g}\left[\log \frac{p_g(\boldsymbol{x})}{p_{\text{data}}(\boldsymbol{x}) + p_g(\boldsymbol{x})}\right]
\end{aligned}
$$

# Global Optimality of $p_g = p_{\text{data}}$

**Theorem 1.** *The global minimum of the virtual training criterion $C(G)$ is achieved if and only if $p_g = p_{data}$. At that point, $C(G)$ achieves the value $-\log 4$.*

*Proof.* For $p_g = p_{\text{data}}$, $D^*_G(\boldsymbol{x}) = \frac{1}{2}$, (consider Eq. 2). Hence, by inspecting Eq. 4 at $D^*_G(\boldsymbol{x}) = \frac{1}{2}$, we find $C(G) = \log \frac{1}{2} + \log \frac{1}{2} = -\log 4$. To see that this is the best possible value of $C(G)$, reached only for $p_g = p_{\text{data}}$, observe that

$$\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}}[-\log 2] + \mathbb{E}_{\boldsymbol{x}\sim p_g}[-\log 2] = -\log 4$$

and that by subtracting this expression from $C(G) = V(D^*_G, G)$, we obtain:

$$C(G) = -\log(4) + KL\left(p_{\text{data}} \left\| \frac{p_{\text{data}} + p_g}{2}\right.\right) + KL\left(p_g \left\| \frac{p_{\text{data}} + p_g}{2}\right.\right) \tag{5}$$

where KL is the Kullback–Leibler divergence. We recognize in the previous expression the Jensen–Shannon divergence between the model's distribution and the data generating process:

$$C(G) = -\log(4) + 2 \cdot JSD\left(p_{\text{data}} \| p_g\right) \tag{6}$$

Since the Jensen–Shannon divergence between two distributions is always non-negative, and zero iff they are equal, we have shown that $C^* = -\log(4)$ is the global minimum of $C(G)$ and that the only solution is $p_g = p_{\text{data}}$, i.e., the generative model perfectly replicating the data distribution. □

**Proposition 2.** *If G and D have enough capacity, and at each step of Algorithm 1, the discriminator is allowed to reach its optimum given G, and $p_g$ is updated so as to improve the criterion*

$$\mathbb{E}_{\boldsymbol{x}\sim p_{data}}[\log D_G^*(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{x}\sim p_g}[\log(1 - D_G^*(\boldsymbol{x}))]$$

*then $p_g$ converges to $p_{data}$*

*Proof.* Consider $V(G, D) = U(p_g, D)$ as a function of $p_g$ as done in the above criterion. Note that $U(p_g, D)$ is convex in $p_g$. The subderivatives of a supremum of convex functions include the derivative of the function at the point where the maximum is attained. In other words, if $f(x) = \sup_{\alpha\in\mathcal{A}} f_\alpha(x)$ and $f_\alpha(x)$ is convex in $x$ for every $\alpha$, then $\partial f_\beta(x) \in \partial f$ if $\beta = \arg\sup_{\alpha\in\mathcal{A}} f_\alpha(x)$. This is equivalent to computing a gradient descent update for $p_g$ at the optimal $D$ given the corresponding $G$. $\sup_D U(p_g, D)$ is convex in $p_g$ with a unique global optima as proven in Thm 1, therefore with sufficiently small updates of $p_g$, $p_g$ converges to $p_x$, concluding the proof. $\square$

# Experiments

## Setup

- datasets: MNIST, Toronto Face Database, CIFAR-10
- activation: a mixture of rectifier linear and sigmoid
- dropout on D

## Evaulation

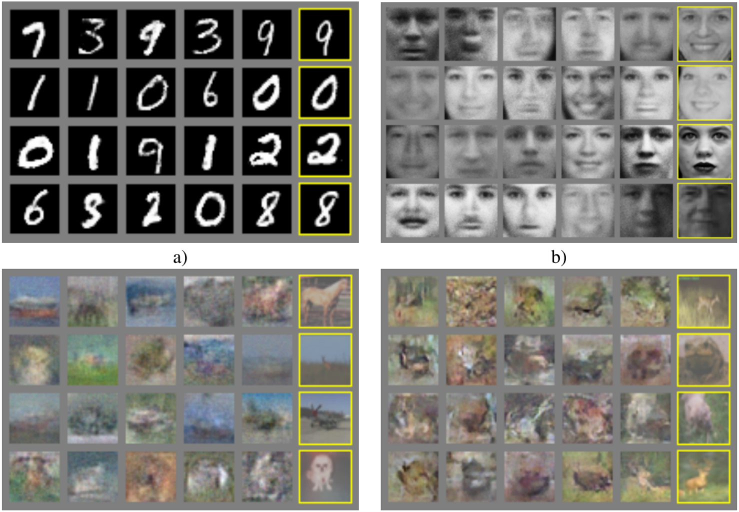
- Parzen window-based log-likelihood estimates
  - the exact likelihood is not tractable
  - somewhat high variance and does not perform well in high dimensional spaces
- no better than existing methods, competitive
- highlight the potential of the adversarial framework

# Experiments

| Model | MNIST | TFD |
|---|---|---|
| DBN [3] | $138 \pm 2$ | $1909 \pm 66$ |
| Stacked CAE [3] | $121 \pm 1.6$ | $\mathbf{2110 \pm 50}$ |
| Deep GSN [5] | $214 \pm 1.1$ | $1890 \pm 29$ |
| Adversarial nets | $\mathbf{225 \pm 2}$ | $\mathbf{2057 \pm 26}$ |

Table 1: Parzen window-based log-likelihood estimates. The reported numbers on MNIST are the mean log-likelihood of samples on test set, with the standard error of the mean computed across examples. On TFD, we computed the standard error across folds of the dataset, with a different $\sigma$ chosen using the validation set of each fold. On TFD, $\sigma$ was cross validated on each fold and mean log-likelihood on each fold were computed. For MNIST we compare against other models of the real-valued (rather than binary) version of dataset.

# Experiments



a) b) c) d)

# Experiments

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

# Advantages and Disadvantages

## Disadvantages

- no explicit representation of $p_g(\mathbf{x})$
- D must be synchronized well with G during training
  - avoid "the Helvetica scenario"

## Advantages

- Markov chains are never needed
- no inference during learning
- a wide variety of functions can be incorporated into the model
- gain some statistical advantage
- represent very sharp, even degenerate distributions

# Summary

| | 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 | Models need to be designed to work with the desired inference scheme — some inference schemes support similar model families as GANs | 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.

# Conclusions and Future Work

1. conditional generative model $p(\mathbf{x}|\mathbf{c})$
2. learned approximate inference
3. approximately model all conditionals $p(\mathbf{x}_S|\mathbf{x}_{-S}$
4. semi-supervised learning
5. efficiency improvements
6. these research directions could prove useful

# Thanks Q&A