Generative Adversarial Nets

핵심 요약

- 적대적으로 동작하는 두개의 네트워크를 사용해 새로운 데이터를 생성할 수 있는 GAN(Generative Adversarial Nets) 구조를 제안합니다.
- 생성자(Generator) 와 감별자(Discriminator) 모두 마르코프 체인 등의 구조없이 back-propagation 으로 학습이 가능한 인공신경망 구조를 사용합니다.
- 이후 등장하는 수많은 GAN 기반 모델의 기원이 되는 논문입니다.

Introduction & Related Works

분류 문제에 제한되어 사용되던 딥러닝 모델의 용도를 새로운 데이터를 생성하는 문제에도 적용할 수 있는 적대적 생성 신경망(Generative Adversarial Nets)을 최초로 제시한 논문입니다.

GAN은 아래와 같은 목표를 가진, 적대적인 두 모델을 학습합니다.

- 감별자(Discriminator) 모델
  - 데이터가 원본 데이터셋에서 온것인지, 생성자가 만든 것인지를 판별
  - 예시) 경찰이 지폐가 위조되었는지를 판별
- 생성자(Generator) 모델
  - 감별자가 구분할 수 없는 가짜 데이터를 생성
  - 예시) 위폐범이 경찰이 구분할 수 없는 위조 지폐를 제작함

논문은 해당 방법이 특별한 모델이나 학습 방법을 필요로 하지 않는 방법이라고 하며, MLP(multi-layer perception) 구조를 사용해 학습한 결과를 소개합니다.

Adversarial nets

적대적 신경망의 가장 직관적인 예시로 MLP 모델을 사용한 경우를 가정하여 설명합니다. 이 때 사용하는 표기법은 다음과 같습니다.

- $x \sim p_{data}$: 실제 데이터로부터 뽑은 샘플
- p_g : 생성자가 생성하는 데이터의 분포
- $p_z(z)$: 데이터를 생성하기 위해 사용하는 입력 노이즈 분포
- $G(z;\theta_g)$: 생성자 모델
- θ_g : 생성자 모델 파라미터
- $D(x;\theta_d)$: 감별자 모델
- θ_D : 감별자 모델 파라미터

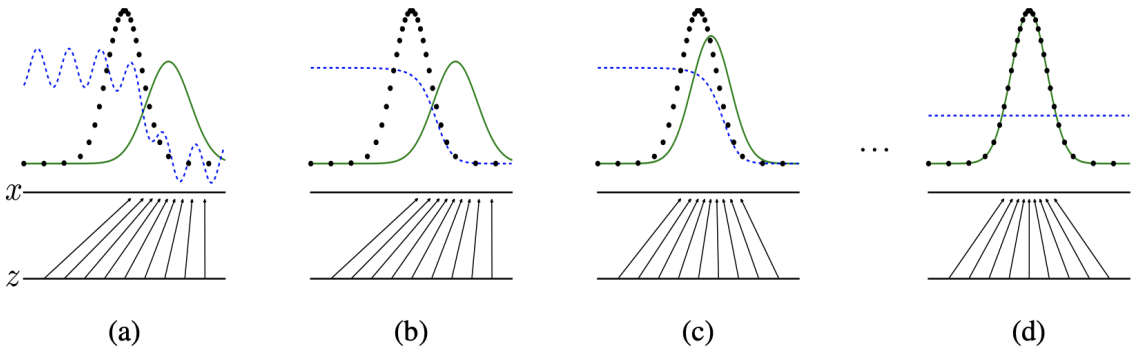
D 는 실제 데이터와 생성된 데이터에 정확히 구분할 수 있는 확률을 최대화 하려고 합니다. G 는 D가 실제 데이터로 착각할 만한 데이터를 생성하는 것을 목적으로 $\log(1 - D(G(z))$ 를 최소화 하려고 합니다.

따라서 아래와 같이 가치함수 $V(G, D)$ 가 주어졌을 때, G 는 최소화 / D는 최대화를 목적으로 경쟁합니다.

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

실제 계산에서 V 를 최대로 하는 D 를 구할 때 많은 계산이 필요하고, 데이터셋이 제한된 상황에서 과적합이 발생할 수도 있습니다. 따라서 실제 훈련에서는 D 를 k 번만 학습하고 G 를 학습합니다.

또한, 학습 초기에는 G가 생성하는 데이터의 품질이 낮으므로 D가 판별을 하기 쉬워, $\log(1-D(G(z)))$ 항이 소실될수 있습니다. 따라서, $\log D(G(z))$ 를 최대화 하는 문제로 변환하여 초기에 학습이 잘 이뤄질 수 있도록 합니다.



학습 과정의 모식도입니다. 파란 점선은 감별자 D의 분포, 검은 점은 원본 데이터 분포, 초록 실선은 생성자 G의 분포를 나타냅니다.

(a) 와 같이 학습이 완료되기 전의 상태에서 시작합니다.

(b) 와 같이 D 를 업데이트 할 때, 최적의 D 는 $D_G^*(x) = \frac{p_{data}(x)}{p_{data}(x)+p_g(x)}$ 로 수렴합니다.

(c) G를 업데이트하면, D 를 교란할 수 있도록 G 가 생성하는 분포가 실제 데이터 분포에 가까워집니다.

(d) 학습 과정을 반복하면 생성자는 데이터 분포와 일치하는 데이터를 생성($p_g = p_{data}$) 하며, 감별자는 어떠한 샘플도 구분할 수 없게 됩니다. ($D(x) = \frac{1}{2}$)

Theoretical Results

적대적 신경망 문제에서 생성자가 원본 데이터와 유사한 분포의 데이터를 생성할 수 있다는 증명을 제시합니다. 또한, 실제 적대적 신경망을 학습하기 위해 설계한 아래 알고리즘 또한 같은 결과에 수렴한다는 증명을 제시합니다.

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.

Global Optimality of $p_g = p_{data}$

먼저 임의의 G 가 주어졌을 때 최적의 D 를 계산하는 과정을 보입니다.

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

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

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

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

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. □

최적의 D 를 이용하여 Equation 1 을 G에 관한 수식으로 표현할 수 있습니다.

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_{data} (with $y = 1$) or from p_g (with $y = 0$). The minimax game in Eq. 1 can now be reformulated as:

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

이 때 새롭게 정리한 가치함수가, G가 생성하는 데이터의 분포가 실제 분포를 따르는 경우에만 최소화된다는 것을 다음과 같이 증명합니다.

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:

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

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 only when 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 generating process. □

Convergence of Algorithm 1

G 와 D 가 p_g 충분한 표현력을 갖고 있을 때, 제시한 알고리즘이 $p_g = p_{data}$ 로 수렴함을 아래와 같이 증명합니다.

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 α, 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. □

실제로 MLP 를 사용한 G 로는 모든 형태의 p_g 를 표현할 수 없으므로 이론적인 최고 성능을 보장하기 어렵습니다. 논문은 그럼에도 불구하고 GAN이 실제 훈련결과에서 좋은 성능을 보임을 제시합니다.

Experiments

실험에 사용한 조건은 다음과 같습니다.

- Dataset : MNIST, Toronto Face Database(TFD), CIFAR-10 사용
- Generator : ReLU/sigmoid 활성함수를 혼합하여 사용
- Discriminator : maxout 활성함수를 사용
- D를 학습시킬 때만 Dropout을 사용
- G에서 데이터를 생성하는 경우에만 noise를 input 으로 사용

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. 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 σ chosen using the validation set of each fold. On TFD, σ 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.

GAN 은 데이터 분포 자체를 구하기 위한 tractable likelihood 를 가정하지 않습니다. 이러한 모델을 평가하기 위해 기존에 제안된 방법은 다음과 같습니다.

- Generator 에서 생성한 데이터를 Gaussian Parzen window 에 fitting
- fitting 한 분포가 주어졌을 때 log-likelihood 를 계산
- Validation set 으로 교차 검증을 수행해 표준 편차를 계산

논문은 해당 방법의 분산이 크고 높은 차원의 데이터에서 잘 작동하지 않지만, GAN 이 기존 모델에 비해 상대적으로 좋은 결과를 보이고 있음을 제시합니다.

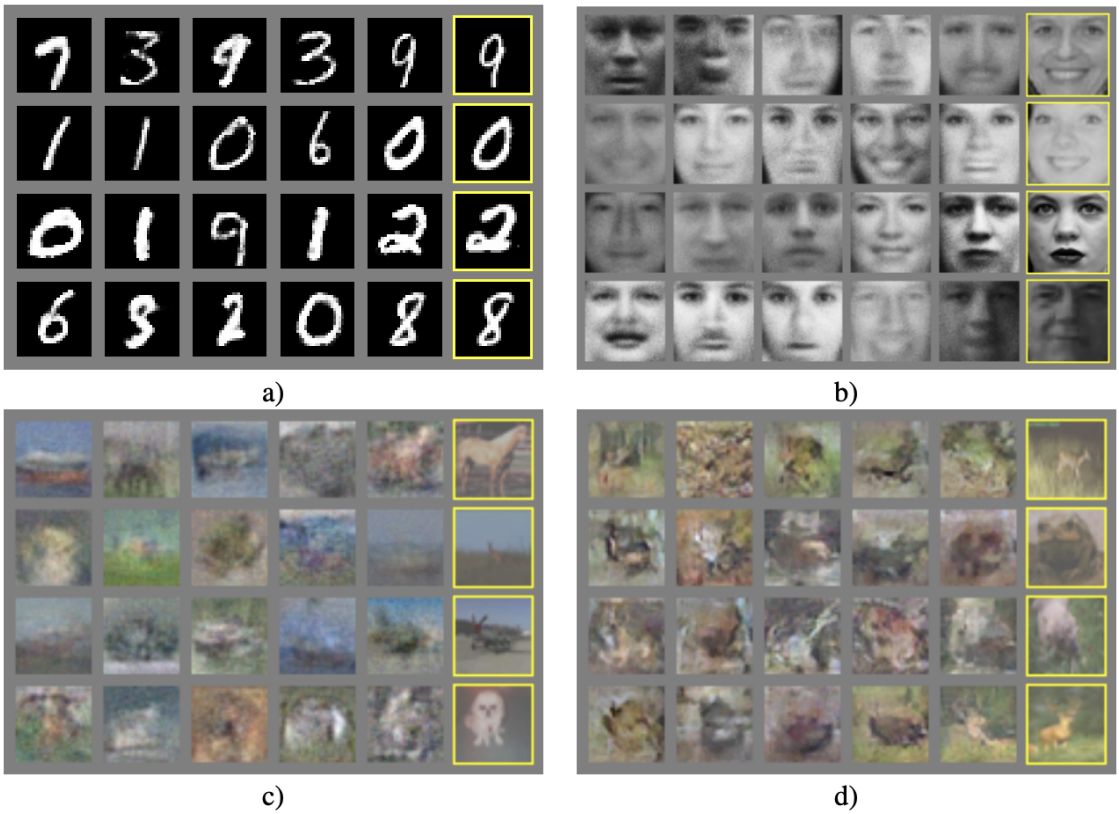


Figure 2: Visualization of samples from the model. Rightmost column shows the nearest training example of the neighboring sample, in order to demonstrate that the model has not memorized the training set. Samples are fair random draws, not cherry-picked. Unlike most other visualizations of deep generative models, these images show actual samples from the model distributions, not conditional means given samples of hidden units. Moreover, these samples are uncorrelated because the sampling process does not depend on Markov chain mixing. a) MNIST b) TFD c) CIFAR-10 (fully connected model) d) CIFAR-10 (convolutional discriminator and "deconvolutional" generator)

다음으로 GAN 모델로 생성한 데이터를 제시합니다. 가장 우측에는 원본 데이터셋 중 생성된 데이터에 가장 가까운 데이터를 배치하였습니다.

논문은 해당 모델이 기존의 생성 모델보다 낫다고 주장하기는 어렵지만, 비슷한 성과와 응용 가능성을 보여줄 수 있다는 의견을 제시합니다.

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

또한 위 그림과 같이, Generator 의 Input noise 를 점진적으로 변형시킬 때, 점점 interploation 되어가는 생성 데이터를 확인할 수 있습니다.

Advantages and disadvantages

GAN 의 단점을 아래와 같이 정리합니다.

- Generator 가 생성하는 데이터의 분포가 명시적으로 존재하지 않습니다.
- Generator 와 Discriminator 의 균형이 깨지는 경우 학습이 원활이 이루어지지 않습니다.

또한, GAN 의 장점을 아래와 같이 정리합니다.

- 마르코프 체인 같은 구조 없이 역전파 만으로도 학습이 가능합니다.
- Generator 의 분포로 특별한 모델을 가정하지 않습니다.
- 더욱 복잡한 데이터 분포를 모사할 수 있어 선명한 데이터를 생성할 수 있습니다.

Conclusions and future work

GAN 프레임워크를 확장하고 개선할 수 있는 다양한 방법을 제시합니다.

1. 주어진 조건에 따라 데이터를 생성하는 모델로 발전 가능
2. x가 주어졌을 때 z를 예측하는 보조 네트워크를 학습한다면 생성자의 데이터 분포를 예측할 수 있음
3. parameters를 공유하는 conditionals model를 학습함으로써 다른 conditionals models을 근사적으로 모델링할 수 있음
4. Semi-supervised learning에도 활용 가능 : 데이터가 제한된 경우 Discriminator 를 활용하여 classifier의 성능을 향상시킬 수 있음
5. 효율성 개선: G와 D를 균형있게 학습할 수 있는 방법이나 새로운 z 분포를 제시하여 학습 속도 개선 가능