

꼼꼼한 딥러닝 논문 리뷰와 코드 실습

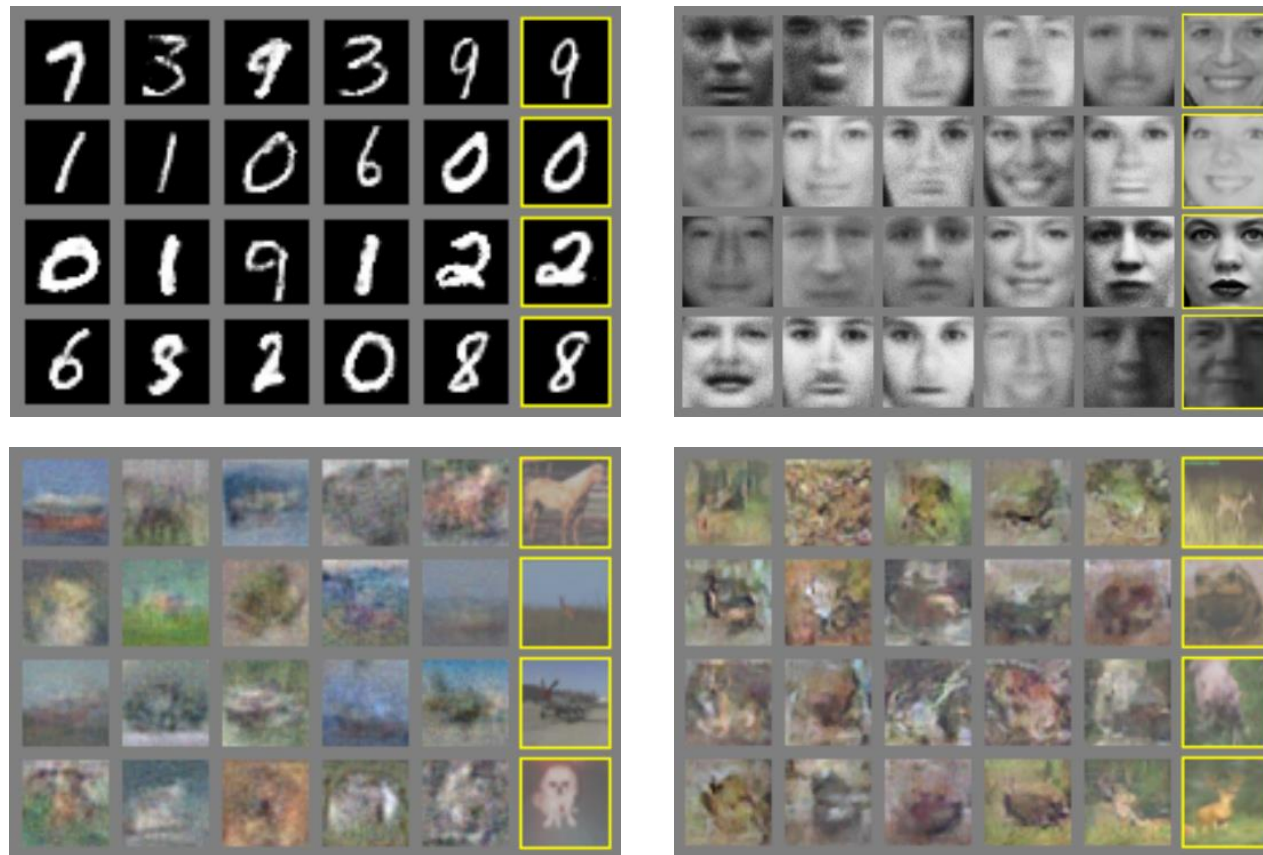
Deep Learning Paper Review and Code Practice

나동빈(dongbinna@postech.ac.kr)

Pohang University of Science and Technology

생성 모델 소개

- 컴퓨터는 어떻게 존재하지 않는 그럴싸한 이미지를 만들어 낼 수 있을까요?



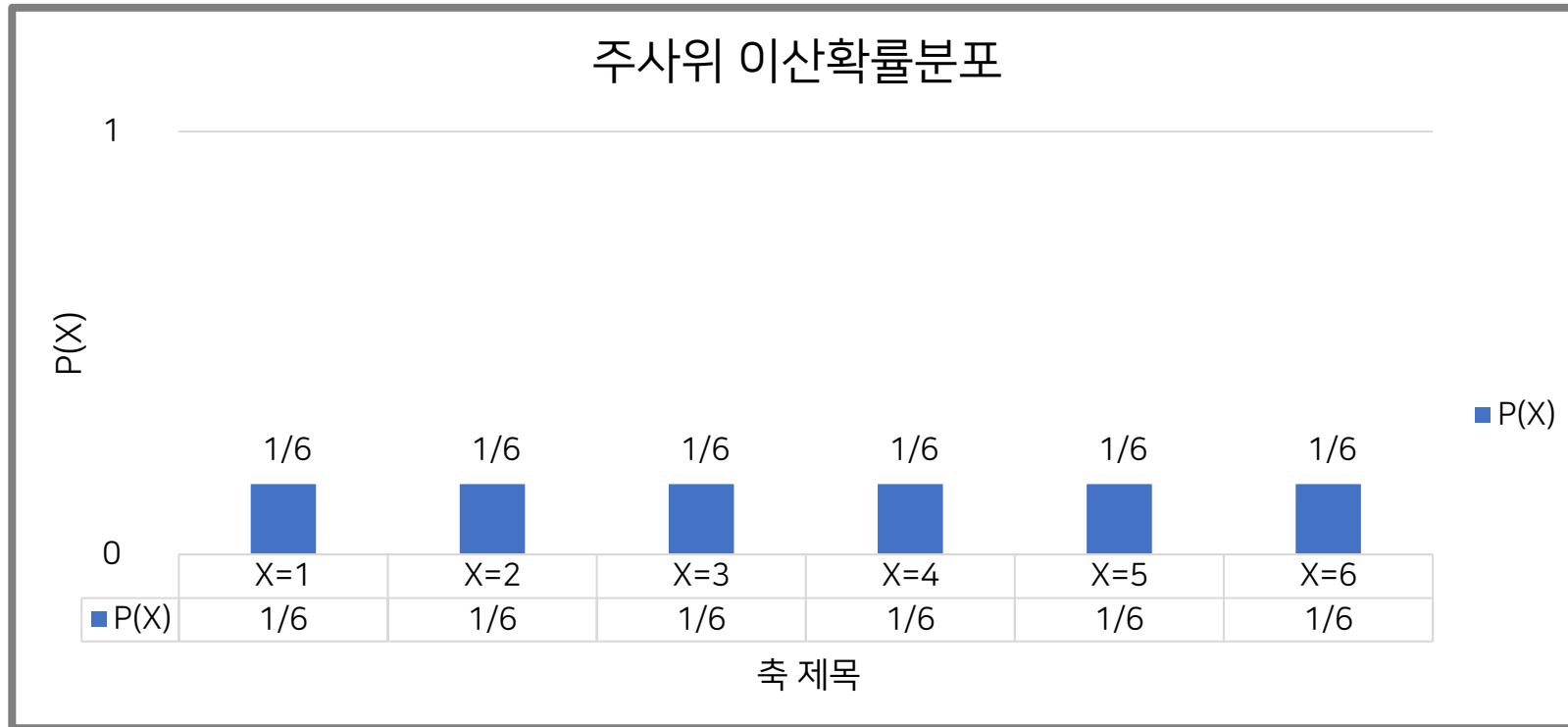
Generative Adversarial Networks (NIPS 2014)

확률분포

- 확률분포는 확률 변수가 특정한 값을 가질 확률을 나타내는 함수를 의미합니다.
- 예를 들어 주사위를 던졌을 때 나올 수 있는 수를 확률변수 X 라고 합시다.
 - 확률변수 X 는 1, 2, 3, 4, 5, 6의 값을 가질 수 있습니다.
 - $P(X = 1)$ 는 $1/6$
 - $P(X = 1) = P(X = 2) = P(X = 3) = P(X = 4) = P(X = 5) = P(X = 6)$

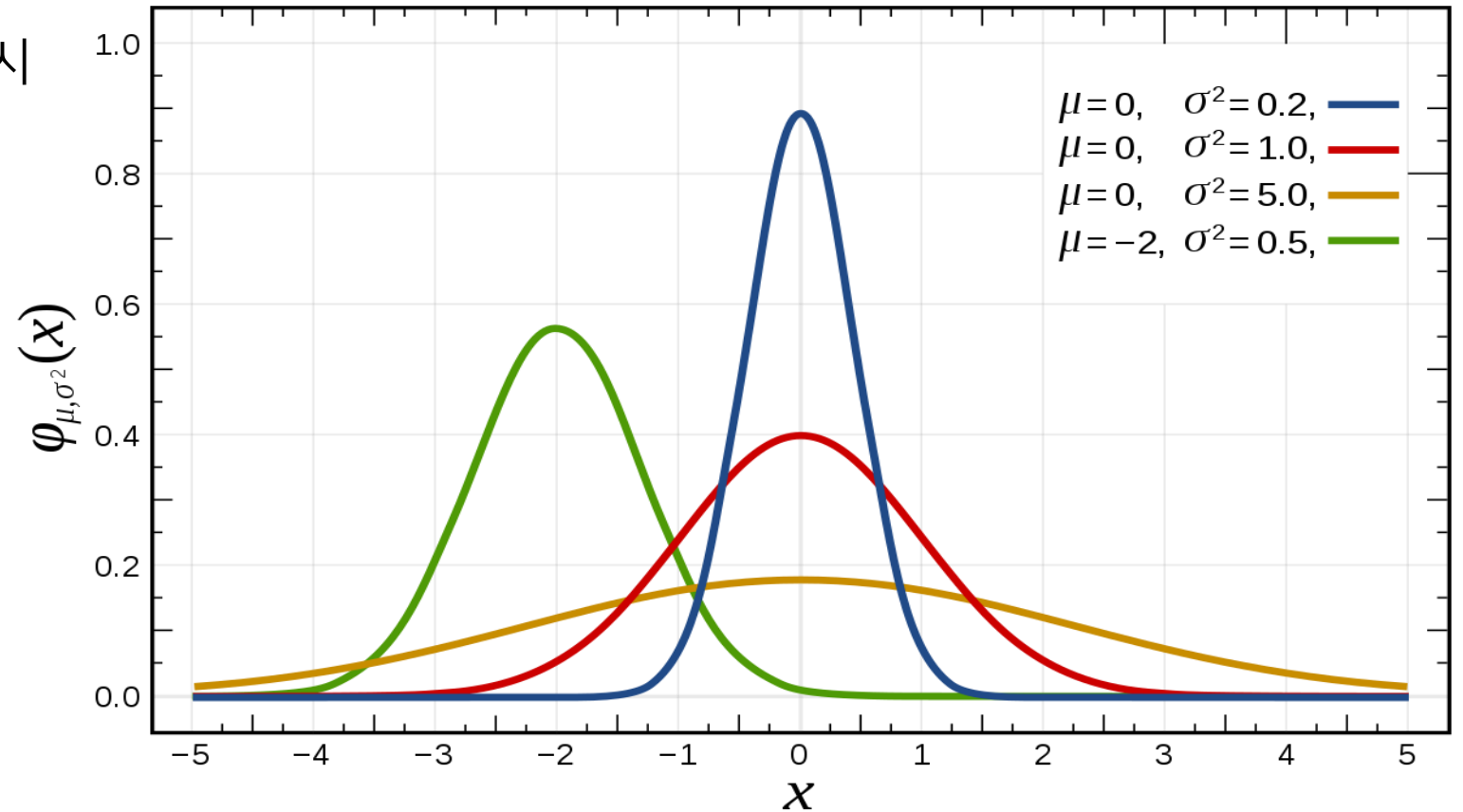
이산확률분포

- 확률변수 X 의 개수를 정확히 셀 수 있을 때 이산확률분포라 말합니다.
- 주사위 눈금 X 의 확률 분포는 다음과 같습니다.



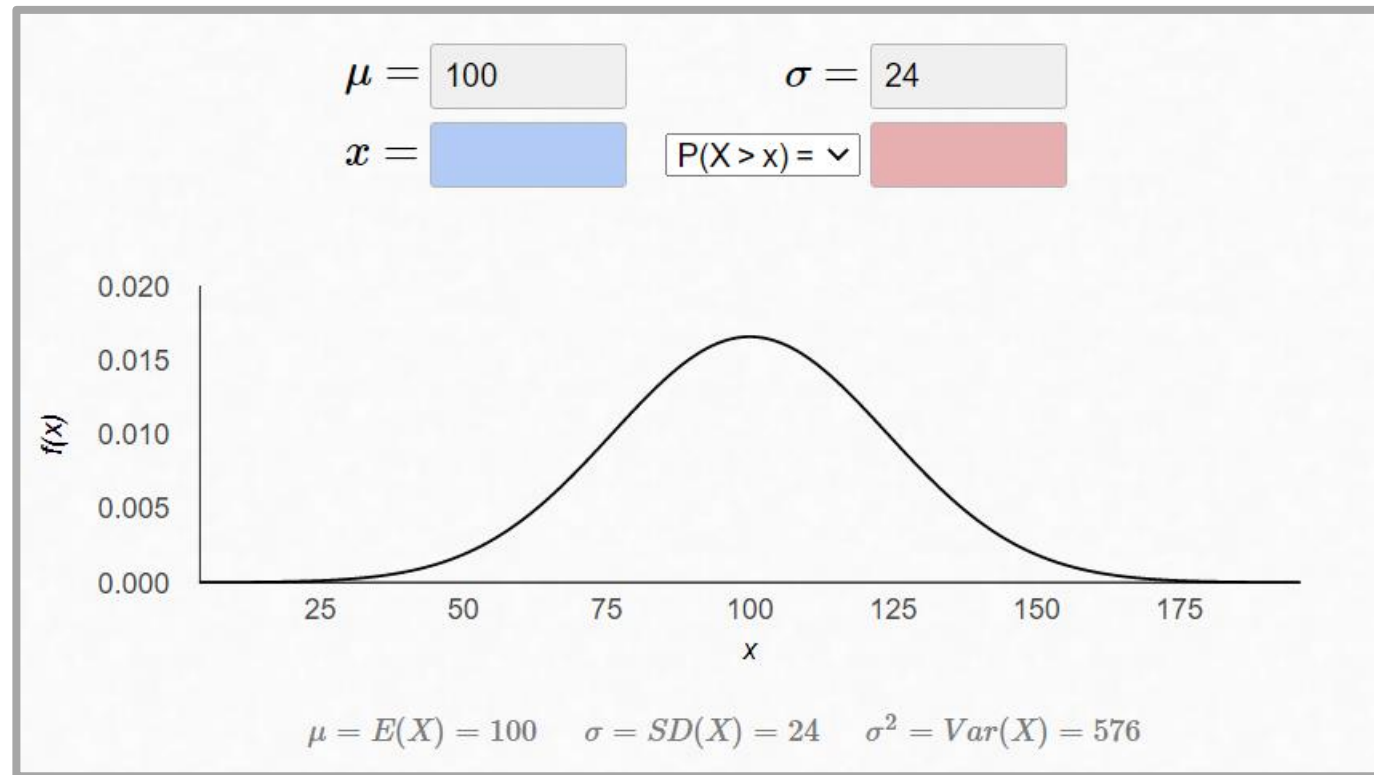
연속확률분포

- 확률변수 X 의 개수를 정확히 셀 수 없을 때 연속확률분포라 말합니다. (확률 밀도 함수를 이용해 분포를 표현)
- 연속적인 값의 예시: 키, 달리기 성적
- 정규분포(normal distribution) 예시



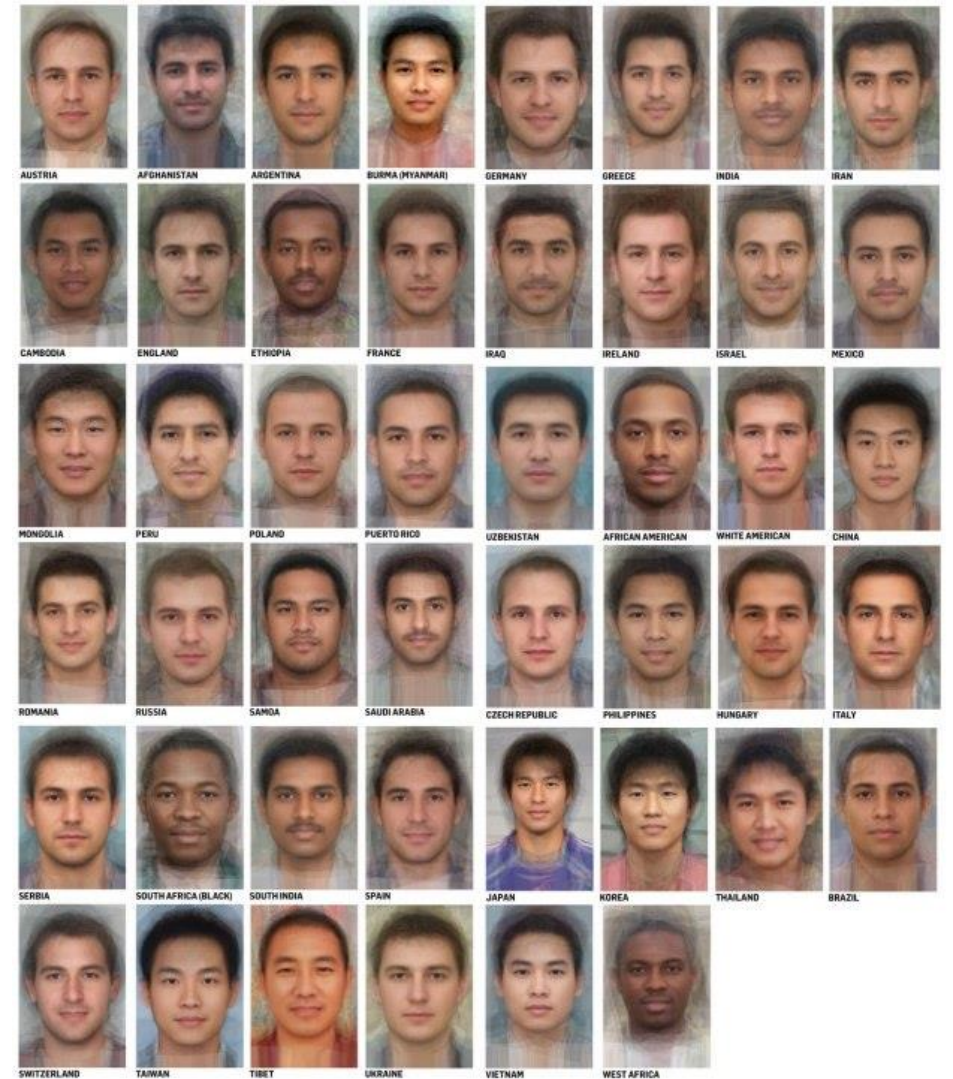
연속확률분포

- 실제 세계의 많은 데이터는 정규분포로 표현할 수 있습니다.
- IQ에 대한 정규분포 예시 (표준편차 = 24)



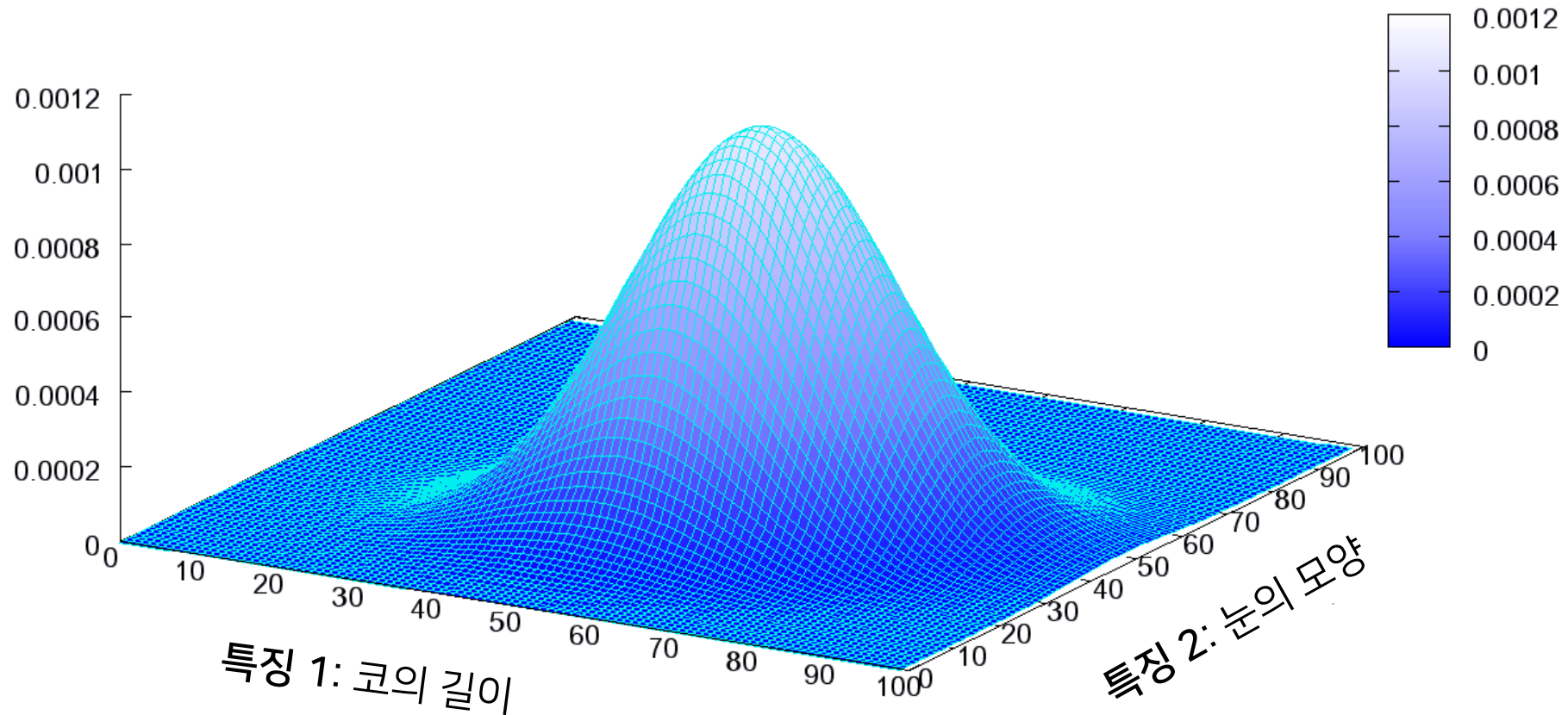
이미지 데이터에 대한 확률분포

- 이미지 데이터는 다차원 특징 공간의 한 점으로 표현됩니다.
 - 이미지의 분포를 근사하는 모델을 학습할 수 있습니다.
- 사람의 얼굴에는 통계적인 평균치가 존재할 수 있습니다.
 - 모델은 이를 수치적으로 표현할 수 있게 됩니다.



이미지 데이터에 대한 확률분포

- 이미지에서의 다양한 특징들이 각각의 확률 변수가 되는 분포를 의미합니다.
 - 다변수 확률분포(multivariate probability distribution) 예시는 다음과 같습니다.

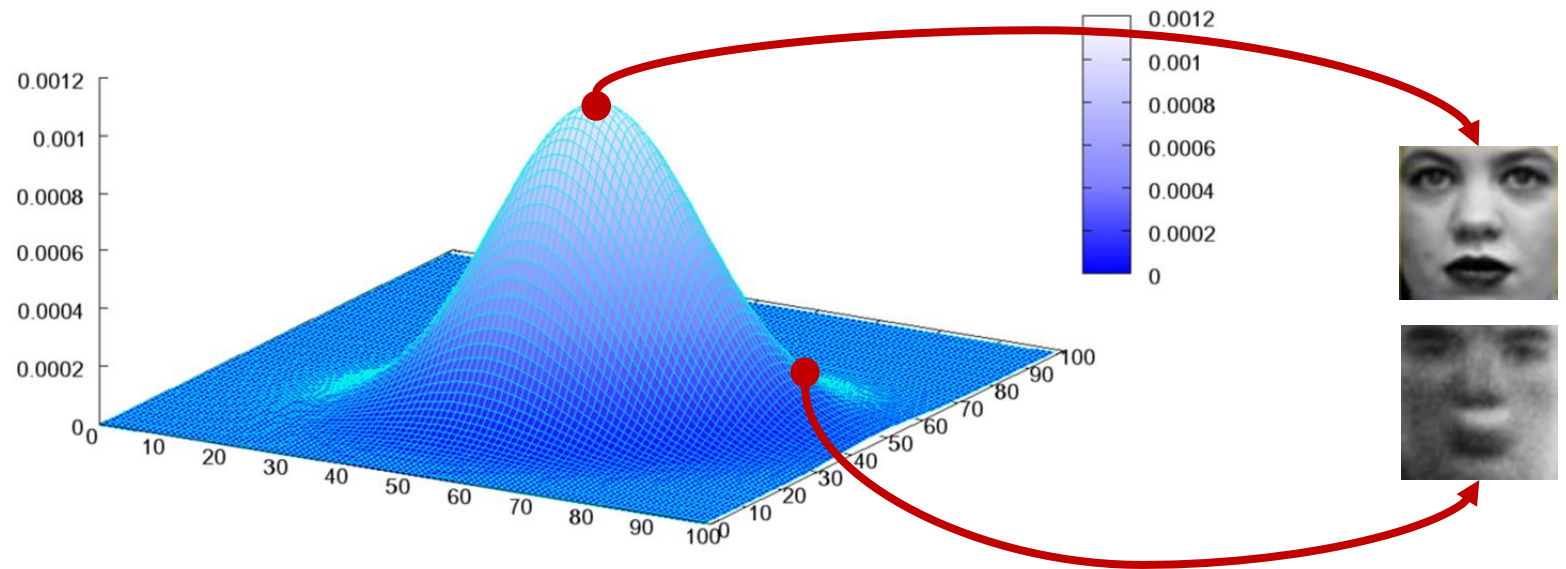
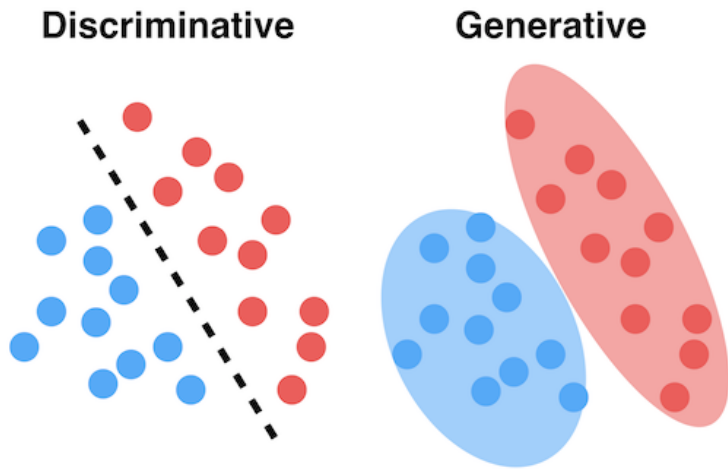


생성 모델 (Generative Models)

- 생성 모델은 실존하지 않지만 있을 법한 이미지를 생성할 수 있는 모델을 의미합니다.

Generative Model (produce) → An image that does not exist but is likely to exist

- A statistical model of the joint probability distribution
- An architecture to generate new data instances

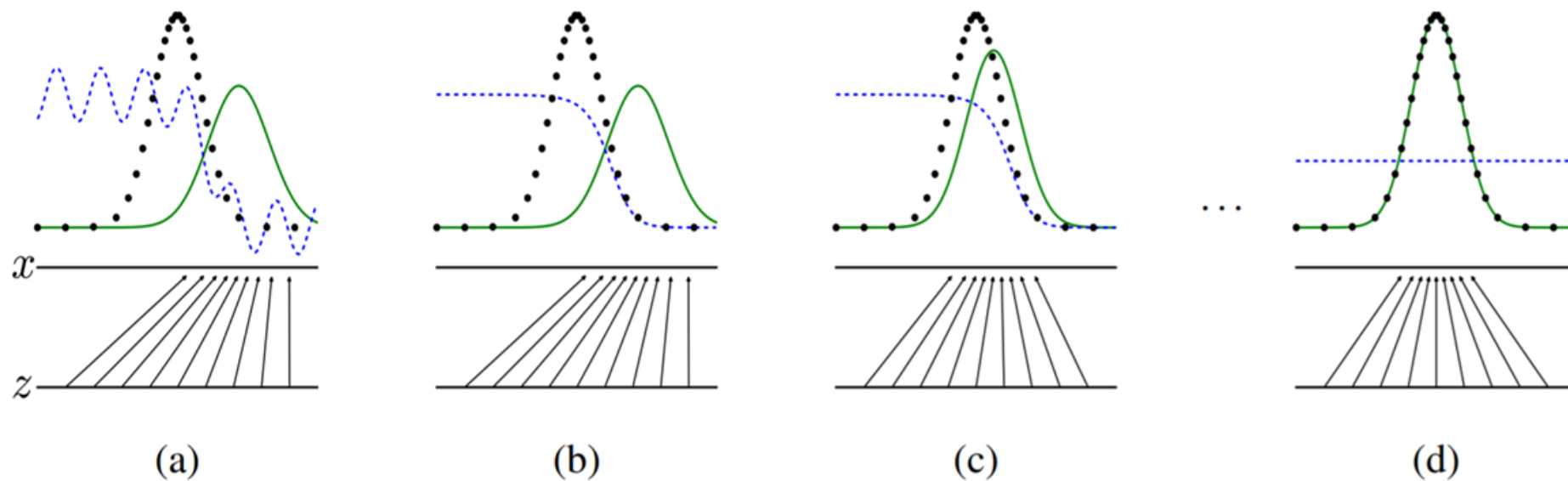


생성 모델 (Generative Model)의 목표

- 이미지 데이터의 분포를 근사하는 모델 G를 만드는 것이 생성 모델의 목표입니다.
- 모델 G가 잘 동작한다는 의미는 원래 이미지들의 분포를 잘 모델링할 수 있다는 것을 의미합니다.
 - 2014년에 제안된 Generative Adversarial Networks (GAN)이 대표적입니다.
 - GAN으로부터 매우 다양한 논문들이 파생되었습니다.

생성 모델 (Generative Model)의 목표

- 모델 G는 원래 데이터(이미지)의 분포를 근사할 수 있도록 학습됩니다.



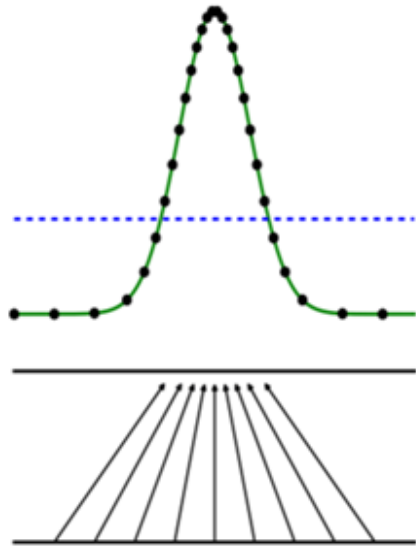
■ 원본 데이터의 분포
■ 생성 모델의 분포

시간이 지나면서 생성 모델 G가 원본 데이터의 분포를 학습

생성 모델 (Generative Model)의 목표

- 모델 G의 학습이 잘 되었다면 원본 데이터의 분포를 근사할 수 있습니다.
 - 학습이 잘 되었다면 통계적으로 평균적인 특징을 가지는 데이터를 쉽게 생성할 수 있습니다.

있을 법한 이미지 생성



Generative Adversarial Networks (GAN)

- 생성자(generator)와 판별자(discriminator) 두 개의 네트워크를 활용한 생성 모델입니다.
- 다음의 목적 함수(objective function)를 통해 생성자는 이미지 분포를 학습할 수 있습니다.

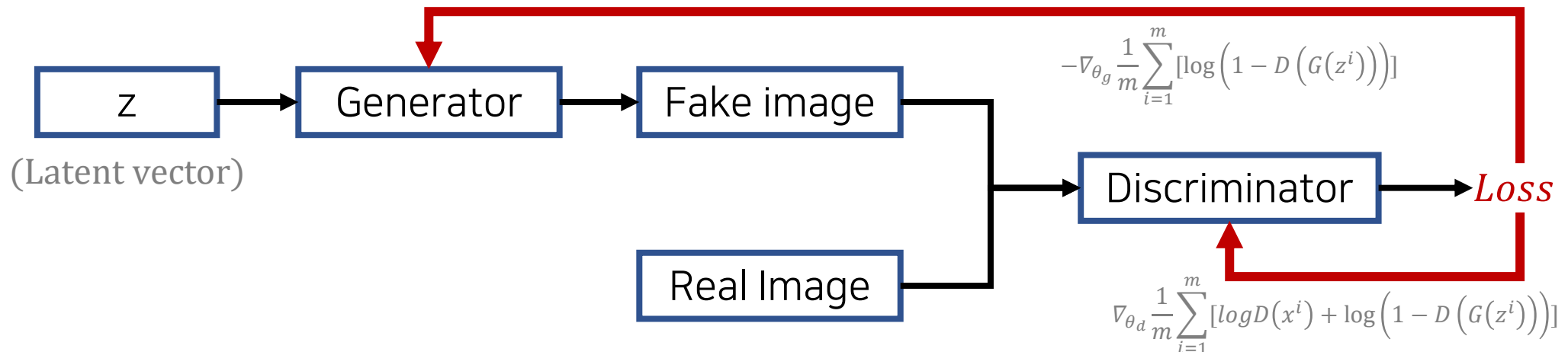
$$\min_G \max_D V(D, G) = E_{x \sim p_{data}(x)} [\log D(x)] + E_{z \sim p_z(z)} [\log(1 - D(G(z)))]$$

Generator

$G(z)$: new data instance

Discriminator

$D(x)$ = Probability: a sample came from the real distribution (Real: 1 ~ Fake: 0)



GAN에서의 기댓값 계산 방법

- 프로그램상에서 기댓값(expected value)을 계산하는 가장 간단한 방법은?
 - 단순히 모든 데이터를 하나씩 확인하여 식에 대입한 뒤에 평균을 계산하면 됩니다.
- $E_{x \sim p_{data}(x)}[\log D(x)]$
 - 원본 데이터 분포(data distribution)에서의 샘플 x 를 뽑아 $\log D(x)$ 의 기댓값 계산
- $E_{z \sim p_z(z)}[\log(1 - D(G(z)))]$
 - 노이즈 분포에서의 샘플 z 를 뽑아 $\log(1 - D(G(z)))$ 의 기댓값 계산

기댓값 공식

- 기댓값은 모든 사건에 대해 확률을 곱하면서 더하여 계산할 수 있습니다.
- 이산확률변수에 대한 기댓값은 다음의 공식을 통해 계산할 수 있습니다.

$$E[X] = \sum_i x_i \cdot f(x_i)$$

- 연속확률변수에 대한 기댓값은 다음의 공식을 통해 계산할 수 있습니다.

$$E[X] = \int x \cdot f(x) dx$$

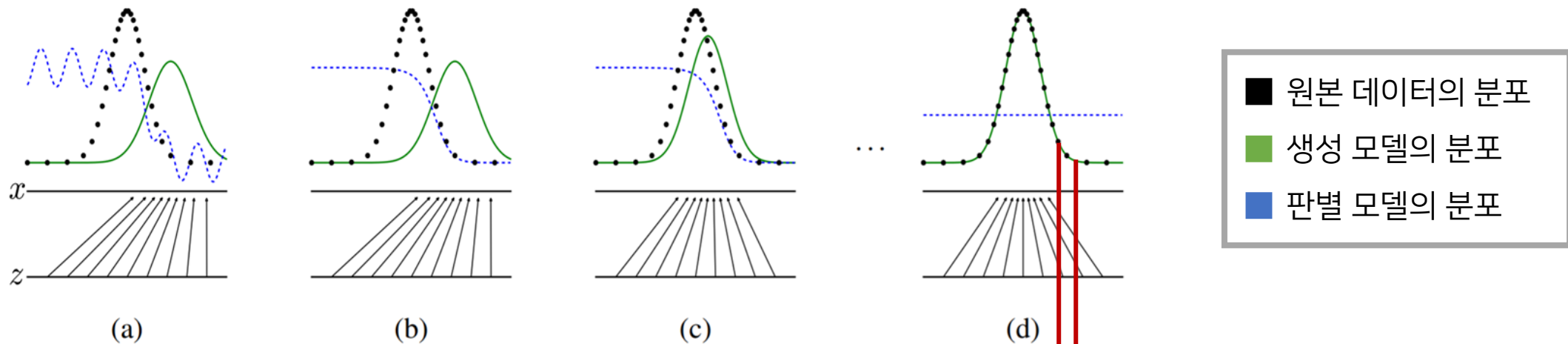
x : 사건

$f(x)$: 확률 분포 함수

GAN의 수렴 과정

- 공식의 목표(Goal of Formulation)

- $P_g \rightarrow P_{data}, D(G(z)) \rightarrow 1/2$ ($G(z)$ is not distinguishable by D)



- How can the formulation lead P_g converge to P_{data} ?

key of proof

original data instance

new data instance

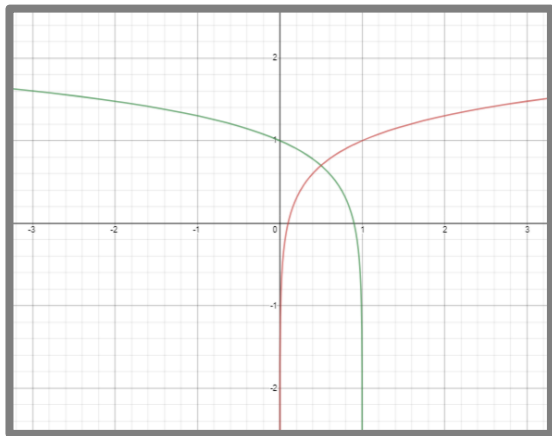
증명: Global Optimality ①

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

Proof: For G fixed,

$$\begin{aligned} V(G, D) &= E_{x \sim p_{data}(x)}[\log D(x)] + E_{z \sim p_z(z)}[\log(1 - D(G(z)))] \\ &= \int_x p_{data}(x) \log(D(x)) dx + \int_z p_z(z) \log(1 - D(g(z))) dz \\ &= \int_x p_{data}(x) \log(D(x)) + p_g(x) \log(1 - D(x)) dx \end{aligned}$$

$E[X] = \int_{-\infty}^{\infty} xf(x)dx$



function $y \rightarrow a \log(y) + b \log(1 - y)$ achieves its maximum in $[0, 1]$ at $\frac{a}{a + b}$

same as *optimal control*: $\frac{\delta V(G, D)}{\delta D} [D^*(x)] = 0$

증명: Global Optimality ②

Proposition: Global optimum point is $p_g = p_{data}$

Proof:

$$\begin{aligned}
 C(G) &= \max_D V(G, D) = E_{x \sim p_{data}(x)} [\log D^*(x)] + E_{z \sim p_z(z)} [\log(1 - D^*(G(z)))] \\
 &= E_{x \sim p_{data}(x)} \left[\log \frac{p_{data}(x)}{p_{data}(x) + p_g(x)} \right] + E_{x \sim p_g(x)} \left[\log \frac{p_g(x)}{p_{data}(x) + p_g(x)} \right] \\
 &= E_{x \sim p_{data}(x)} \left[\log \frac{2 * p_{data}(x)}{p_{data}(x) + p_g(x)} \right] + E_{x \sim p_g(x)} \left[\log \frac{2 * p_g(x)}{p_{data}(x) + p_g(x)} \right] - \log(4) \\
 &= KL(p_{data} || \frac{p_{data}(x) + p_g(x)}{2}) + KL(p_g || \frac{p_{data}(x) + p_g(x)}{2}) - \log(4) \\
 &= 2 * JSD(p_{data} || p_g) - \log(4)
 \end{aligned}$$

removed when $p_g = p_{data}$

$D_G^*(x) = \frac{p_{data}(x)}{p_{data}(x) + p_g(x)}$
 $KL(p_{data} || p_g) = \int_{-\infty}^{\infty} p_{data}(x) \log \left(\frac{p_{data}(x)}{p_g(x)} \right) dx$
 $JSD(p || q) = \frac{1}{2} KL(p || \frac{p+q}{2}) + \frac{1}{2} KL(q || \frac{p+q}{2})$

for the number of training iterations do

for k steps do

Sample minibatch of m noise samples $\{z^{(1)}, \dots, z^{(m)}\}$ from noise prior $p_g(z)$.

Sample minibatch of m examples $\{x^{(1)}, \dots, x^{(m)}\}$ from data generating distribution $p_{data}(x)$.

Update the discriminator by ascending its stochastic gradient:

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

end for

Sample minibatch of m noise samples $\{z^{(1)}, \dots, z^{(m)}\}$ from noise prior $p_g(z)$.

Update the generator by descending its stochastic gradient:

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

end for

The gradient-based updates can use any standard gradient-based learning rule. They used momentum.

Visualization of Experiment



- Not cherry-picked
- **Not memorized** the training set
- Competitive with the better generative models
- Images represent sharp

Fig. Visualization of samples from the model



Fig. Digits obtained by linearly interpolating between coordinates in z space of the model