딥러닝의 통계적이해

10강. 오토인코더와 GAN(2)

1. 비지도 학습(생성모형(1))
2. 인자분석
3. GAN

서울대학교 김용대 교수

오늘의 학습목표

1. 인자 분석이 무엇인지 이해할 수 있다.
2. GAN 방법론에 대해 이해할 수 있다.

1. 비지도 학습법(지난시간)

비지도 학습법

- 기계 학습의 일종.
- 데이터가 어떤 특징을 가지고 있는지를 알아내는 것이 목표.
- 확률 모형 기반 X
  - 주성분 분석, 군집 분석 등
- 확률 모형 기반 O
  - 인자 분석, 밀도 추정 등

심층 비지도 학습법

- 심층 신경망 모형에 기초한 비지도 학습 방법론을 총칭
- 확률 모형 기반 X
  - Auto Encoder, Deep clustering 등
- 확률 모형 기반 O
  - GAN, VAE 등

비지도 학습법

◆ 본 강연에서는 인자 분석과 GAN을 다룰 예정.

확률 모형 기반 여부 \ 변환 함수	선형 함수	심층 신경망 함수
X	주성분 분석	오토 인코더
O	인자 분석	GAN

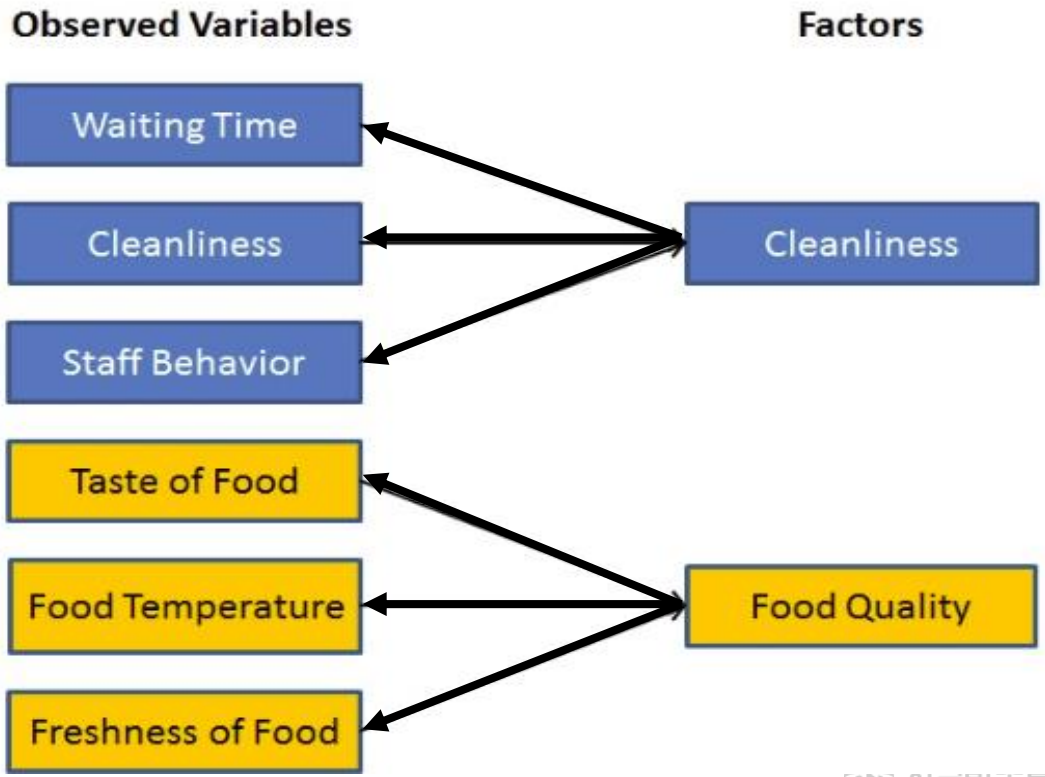
2. 인자 분석

인자분석

- Factor Analysis
- "변수들의 실현값에 영향을 주는 잠재인자 (latent factor) 들이 존재하지 않을까?"
- 변수들을 설명하는 낮은 차원의 공통 인자가 있다고 가정.

인자분석

- 변수들과 공통 인자들 사이의 관계를 파악하고 이를 통해 변수들간의 공분산 구조를 파악하는 것을 목표로 함.



인자 분석 vs. 주성분 분석

- 주성분 분석 : 차원 축소가 목표
- 인자 분석 : 소수의 관찰 불가능한 인자와 관찰 가능한 변수들 사이의 근본적인 구조를 찾는 것이 목표.

직교 인자 모형

- $\boldsymbol{X} = (X_1, \ldots, X_p)$: 평균이 μ, 공분산이 Σ 인 (관찰 가능한) 랜덤 벡터.
- $\boldsymbol{F} = (F_1, \ldots, F_q)$: 공통 인자 (common factors)
  - 주로 $q \leq p$ 를 가정.
- $\boldsymbol{\epsilon} = (\epsilon_1, \ldots, \epsilon_p)$: 오차항

직교 인자 모형

- 모형:

$$X_1 - \mu_1 = l_{11}F_1 + \cdots + l_{1q}F_q + \epsilon_1$$

$$\cdots$$

$$X_p - \mu_p = l_{p1}F_1 + \cdots + l_{pq}F_q + \epsilon_p$$

- 행렬 표현:

$$\boldsymbol{X}_{p\times 1} - \boldsymbol{\mu}_{p\times 1} = \boldsymbol{L}_{p\times q}\boldsymbol{F}_{q\times 1} + \boldsymbol{\epsilon}_{p\times 1}$$

- l_{ij} , $\boldsymbol{L}$: 인자 적재 (factor loadings), 인자 적재 행렬 (matrix of factor loadings)

직교 인자 모형

- 가정:
  - $E(\boldsymbol{F}) = 0$
  - $Cov(\boldsymbol{F}) = I$ (직교 인자 모형)
  - $E(\boldsymbol{\epsilon}) = 0$
  - $Cov(\boldsymbol{\epsilon}) = \Psi = diag[\psi_1, \ldots, \psi_p]$
  - $Cov(\boldsymbol{\epsilon}, \boldsymbol{F}) = 0$

직교 인자 모형

- 관찰 가능한 변수 $\boldsymbol{X}$ 의 공분산 행렬 Σ 는 총 $p(p+1)/2$ 개의 모수를 가지는 반면,
- q 개의 인자를 가지는 인자 모형의 경우 $pq+p$ 개의 모수를 가짐.
- $q \ll p$ 인 경우 인자 모형을 이용하여 모수의 수를 상당히 줄일 수 있음.

모수 추정 방법

- $x_1, \ldots, x_n \in \mathbb{R}^p$: 관찰한 n 개의 데이터.
- 모형

$$\boldsymbol{X} - \boldsymbol{\mu} = \boldsymbol{LF} + \boldsymbol{\epsilon}$$

- 추정 해야 할 모수 : $\boldsymbol{L}, \boldsymbol{\Psi}$

모수 추정 방법

- 대표적인 모수 추정의 세 가지 방법

  - 주성분 방법 (principal component method)
  - 주인자 방법 (principal factor method)
  - 최대 가능도 방법 (maximum likelihood method)

최대 가능도 방법

- 확률에 기초하여 모수를 추정. -> 적합도 검정이 가능함.
- 추가 가정

$$F \sim N(0_q, I_q)\,,\, \epsilon \sim N(0_p\,, \boldsymbol{\Psi})$$

- 관찰 가능한 변수 x 의 확률 밀도 함수 :

$$p(\mathrm{x}; \boldsymbol{L}, \boldsymbol{\Psi}) = \int p(f) \cdot p(\mathrm{x}|f; \boldsymbol{L}, \boldsymbol{\Psi})\, df$$

최대 가능도 방법

- $x_1, \ldots, x_n \in \mathbb{R}^p$: 관찰한 n 개의 데이터.
- 로그 가능도 함수를 최대로 하는 모수 $\boldsymbol{L}, \boldsymbol{\Psi}$ 를 추정:

$$\hat{\boldsymbol{L}}, \hat{\boldsymbol{\Psi}} = \underset{\boldsymbol{L},\boldsymbol{\Psi}}{\operatorname{argmax}} \sum_{i=1}^{n} \log P(x_i; \boldsymbol{L}, \boldsymbol{\Psi})$$

- 기대화-최대화 (Expectation-Maximization) 알고리즘을 사용.

EM 알고리즘

- 관측되지 않는 잠재 변수에 의존하는 확률 모델에서 최대 가능도를 갖는 모수의 추정값을 찾는 알고리즘.

- 기대값 단계 (E-step) 와 최대화 단계 (M-step) 으로 나뉘어짐.
  - 기대값 단계 : 로그 가능도의 기대값을 계산.
  - 최대화 단계 : 기대값을 최대화하는 모수를 추정.

- 기대값 단계와 최대화 단계를 반복적으로 수행하여 모수의 추정값을 구함.

EM 알고리즘을 이용한 모수 추정

- $\boldsymbol{L}^{(1)}, \boldsymbol{\Psi}^{(1)}$: EM 알고리즘을 시작하기 위한 초기값

  → 초기값은 임의로 정해준다.

- $\boldsymbol{L}^{(m)}, \boldsymbol{\Psi}^{(m)}$: EM 알고리즘의 m 번째 단계에서의 해

EM 알고리즘을 이용한 모수 추정

1. 기대값 단계

$$Q\left(\boldsymbol{L},\boldsymbol{\Psi}\middle|\boldsymbol{L}^{(m)},\boldsymbol{\Psi}^{(m)},\mathrm{x}\right) := \int \log p(\mathrm{x},f;\boldsymbol{L},\boldsymbol{\Psi})\cdot p\left(f\middle|\mathrm{x};\boldsymbol{L}^{(m)},\boldsymbol{\Psi}^{(m)}\right)df$$

다음의 값을 계산 :

$$Q\left(\boldsymbol{L},\boldsymbol{\Psi}\middle|\boldsymbol{L}^{(m)},\boldsymbol{\Psi}^{(m)}\right) := \sum_{i=1}^{n} Q\left(\boldsymbol{L},\boldsymbol{\Psi}\middle|\boldsymbol{L}^{(m)},\boldsymbol{\Psi}^{(m)},\mathrm{x}_i\right)$$

EM 알고리즘을 이용한 모수 추정

2. 최대화 단계

기대값 단계에서 구한 Q 함수를 최대화 하는 $\boldsymbol{L}, \boldsymbol{\Psi}$ 를 찾고,

이를 $\boldsymbol{L}^{(m+1)}, \boldsymbol{\Psi}^{(m+1)}$ 라 함.

$$\boldsymbol{L}^{(m+1)}, \boldsymbol{\Psi}^{(m+1)} = \underset{\boldsymbol{L},\boldsymbol{\Psi}}{\operatorname{argmax}}\, Q\left(\boldsymbol{L}, \boldsymbol{\Psi} \middle| \boldsymbol{L}^{(m)}, \boldsymbol{\Psi}^{(m)}\right)$$

- 기대값 단계와 최대화 단계를 수렴할 때까지 반복적으로 수행.

인자 분석 코드

```
from keras.datasets import mnist
from sklearn.decomposition import FactorAnalysis as FA
```

MNIST 모듈 및 요인 분석 함수 불러오기

```
(X_train, y_train), (X_test, y_test) = mnist.load_data()
X_train = X_train.reshape(X_train.shape[0],28*28)
X_test = X_test.reshape(X_test.shape[0],28*28)
```

MNIST 자료를 불러오고 형태 변환
(28,28)->(28*28)

```
fa = FA(n_components = 10, random_state = 77)
fa.fit(X_train)
```

훈련 자료를 이용해 잠재 요인 찾기
(n_components는 찾을 잠재 요인의 개수)

```
FactorAnalysis(copy=True, iterated_power=3, max_iter=1000, n_components=10,
        noise_variance_init=None, random_state=77, svd_method='randomized',
        tol=0.01)
```

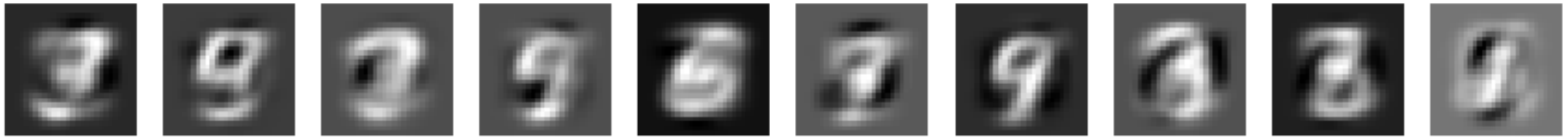
인자 분석 코드

랜덤하게 공통 인자 벡터를 생성한 뒤
그것과 인자적재를 바탕으로
원래와 비슷한 꼴의 자료를 생성

```
np.random.seed(15)
loading = np.transpose(fa.components_)
generated = np.transpose(np.dot(loading,np.random.normal(size=(10,10))))
mu = np.mean(X_train,0)

import matplotlib.pyplot as plt
plt.figure(figsize=(15,3))

for i in range(10):
    plt.subplot(1, 10, i+1)
    plt.axis('off')
    plt.imshow((mu+generated[i]).reshape(28,28), cmap='gray')
plt.show()
```



3. GAN 방법론

Deep Generative Model

- 심층 생성 모형
- 이미지 자료, 오디오 자료 등 고차원 자료에 많이 사용.
- 인자 모형과 마찬가지로, 고차원 자료를 설명하는 관측 불가능한 저차원의 잠재 벡터가 있다고 가정.

인자 모형 vs. 심층 생성 모형

- 인자 벡터와 관찰 벡터 사이의 관계
- 인자 모형 : 선형 함수 관계
- 심층 생성 모형 : 심층 신경망 함수 관계

Deep Generative Model

- $X \in \mathbb{R}^p$: 관찰 가능한 랜덤 벡터
- $Z \in \mathbb{R}^q$: 관찰 불가능한 잠재 벡터 ($q < p$ 를 가정)
- 오차항이 있는 모형
  - Variational Auto Encoder
- 오차항이 없는 모형
  - Generative Adversarial Network

Deep Generative Model

1. 오차항이 있는 모형

$$Z \sim N(0, I),$$

$$X|Z \sim N(G(Z;\theta), \sigma^2 I),$$

여기서, $G(\cdot;\theta)$ 는 θ 를 모수로 갖는 심층 신경망.

- 일반적인 확률 밀도 함수가 존재하므로
  최대 가능도 추정 방법으로 모수를 추정할 수 있음.

Deep Generative Model

2. 오차항이 없는 모형

$$Z \sim N(0, I),$$

$$X|Z = G(Z; \theta),$$

여기서, $G(\cdot; \theta)$ 는 θ 를 모수로 갖는 심층 신경망.

- 일반적인 확률 밀도 함수가 존재하지 않으므로,
  최대 가능도 추정 방법으로 모수를 추정할 수 없음.

Deep Generative Model

- 본 강연에서는 오차항이 없는 모형에 대해서만 다룸.
- 대표적인 추정 방법 : Generative Adversarial Network (GAN)

GAN의 의미

- Generative
  - 어떤 목적을 가진 방법론인지.
  - 실제 데이터의 분포를 최대한 따라하는 생성 모형을 학습.

GAN 의 의미

- Adversarial
  - 생성 모형을 어떻게 학습할 것인지.
  - 두 개의 모형을 적대적 (adversarial) 으로 경쟁시키며 발전시킴.

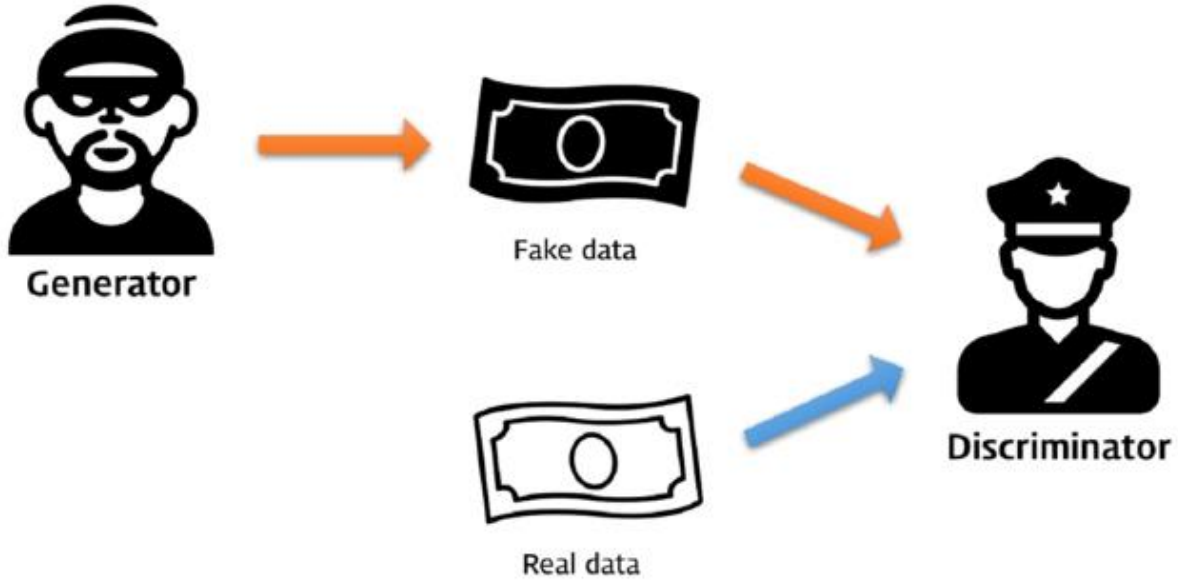
GAN 의 의미

- Adversarial (계속)
  - 생성자 : 위조 지폐범, 구분자 : 경찰
  - 생성자의 목적 : 구분자가 구분할 수 없게끔 그럴듯한 가짜 데이터를 생성.
  - 구분자의 목적 : 생성자가 만든 가짜 데이터와 진짜 데이터를 구분.

GAN의 의미

- Adversarial (계속)

  - 이 둘을 함께 학습 -> 진짜와 구분할 수 없는 가짜를 만들어내는 생성자를 얻을 수 있음.



GAN의 의미

- Network
  - 어떤 형태의 모형을 사용할 것인지.
  - 심층 인공 신경망 함수를 사용.

GAN을 구성하는 두 모형

- **생성 모형** (Generator)
  - 랜덤 노이즈에서 자료를 생성하는 심층 신경망 모형.

- **판별 모형** (Discriminator)
  - 자료가 입력값으로 들어올 때, 주어진 자료가 학습자료에서 얻어진 것인지, 혹은 생성 모형을 이용해 생성 되었는지를 판별하는 심층 신경망 모형.

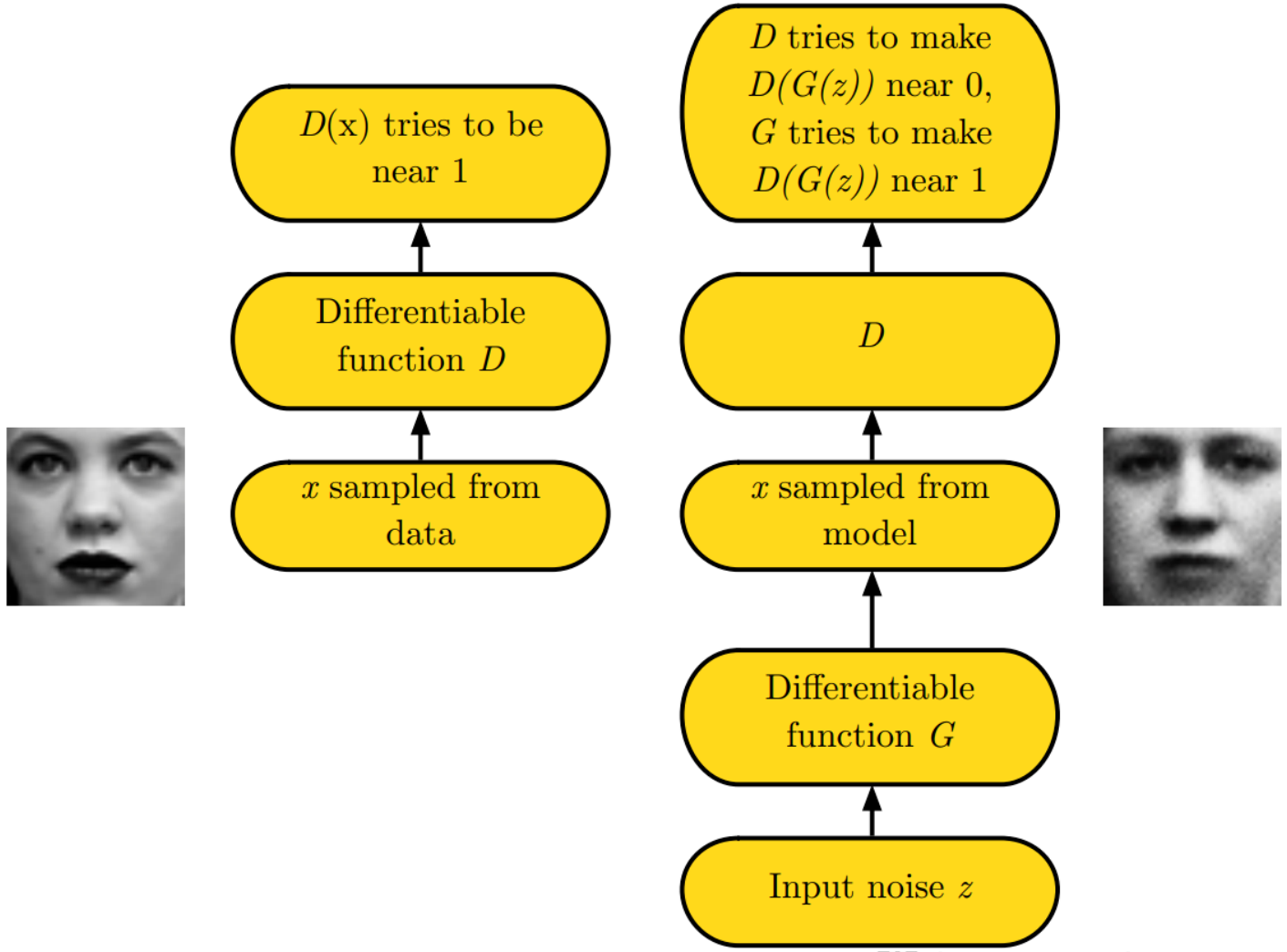
GAN의 표기법

- $Z \sim p(z)$: 랜덤 노이즈. 균일 분포 혹은 표준 정규 분포를 사용.
- $G(z;\theta)$: z 를 이용하여 자료를 생성하는 심층 신경망 모형.
  θ 는 함수에 필요한 모수.
- $D(\mathrm{x};\eta)$: x 를 입력하여 0부터 1 사이의 값을 출력하는 심층 신경망 모형.
  η 는 함수에 필요한 모수.
  - $D(\mathrm{x};\eta) > 0.5$: 학습 자료 내 이미지로 판단
  - $D(\mathrm{x};\eta) < 0.5$: 생성된 이미지로 판단.

GAN의 학습

- Two-player minimax game
- 생성 함수 G 는 판별 함수 D 가 학습 자료와 생성 자료를 구분하지 못하도록 θ 를 학습.
- 판별 함수 D 는 학습 자료와 생성 자료를 잘 구분하도록 η 를 학습.
- 두 함수가 서로 경쟁하는 게임의 형태로 학습.

GAN의 학습


D tries to make D(G(z)) near 0, G tries to make D(G(z)) near 1
D(x) tries to be near 1
D
Differentiable function D
x sampled from model
x sampled from data
Differentiable function G
Input noise z

GAN의 학습

- 목적 함수

$$\min_{\theta}\max_{\eta}\mathbb{E}_{x\sim p_{data}}[\log D(x;\eta)]+\mathbb{E}_{z\sim p(z)}[\log(1-D(G(z;\theta);\eta))]$$

- 주어진 목적 함수를 η에 대해서 최대화하고,
  θ에 대해서 최소화하는 작업을 번갈아 진행.

GAN 의 학습

- 주어진 generator 에 대해서
  discriminator 는 실제 data 와 만들어진 data 의 분류를
  최대한 잘 하는 방향으로 학습.

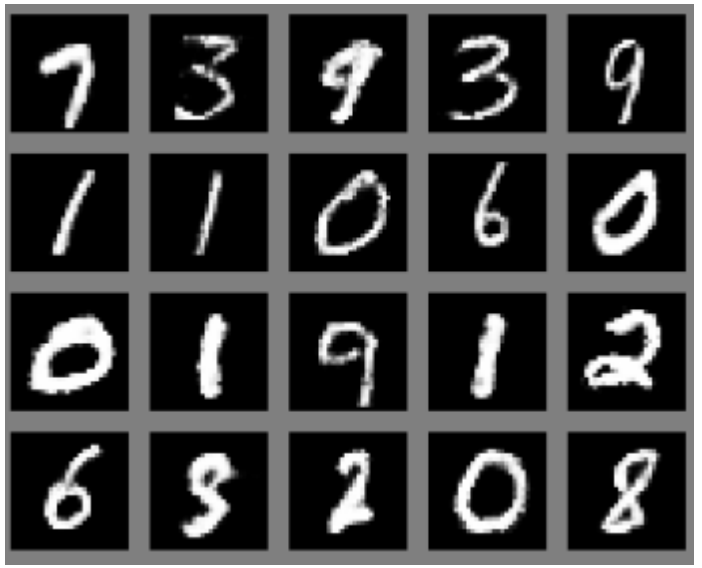
- 주어진 discriminator 에 대해서
  generator 는 discriminator 가 최대한 헷갈리도록
  실제 data 와 비슷한 input 을 생성하는 방향으로 학습.

GAN 의 학습

- 실제 학습 시
- p_{data} 를 알지 못함 -> Empirical distribution 을 사용.
- $p(z)$ 로 적분하는 것이 힘듦 -> Monte Carlo 적분 이용.

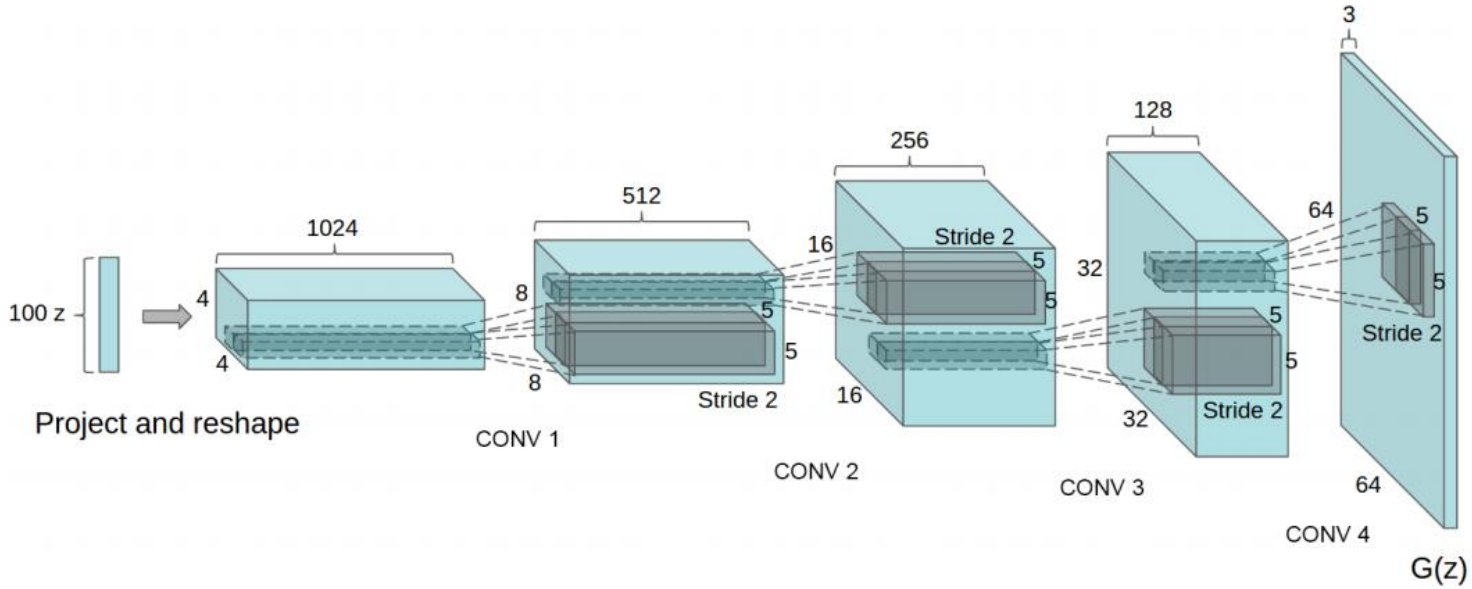
GAN 의 장단점

- 장점
  - 균일 분포 또는 표준 정규 분포로부터 손쉽게 표본을 추출하여 이미지를 생성할 수 있음.
  - 생성된 이미지가 실제 이미지처럼 선명.
- 단점
  - 학습이 불안정. (min, max를 동시에 적용.)
  - 고해상도 이미지에서 우수한 성능 기대 X.
  - Mode collapsing이 생김.
    - MNIST의 경우 몇 가지 숫자들만 생성.



GAN 의 발전

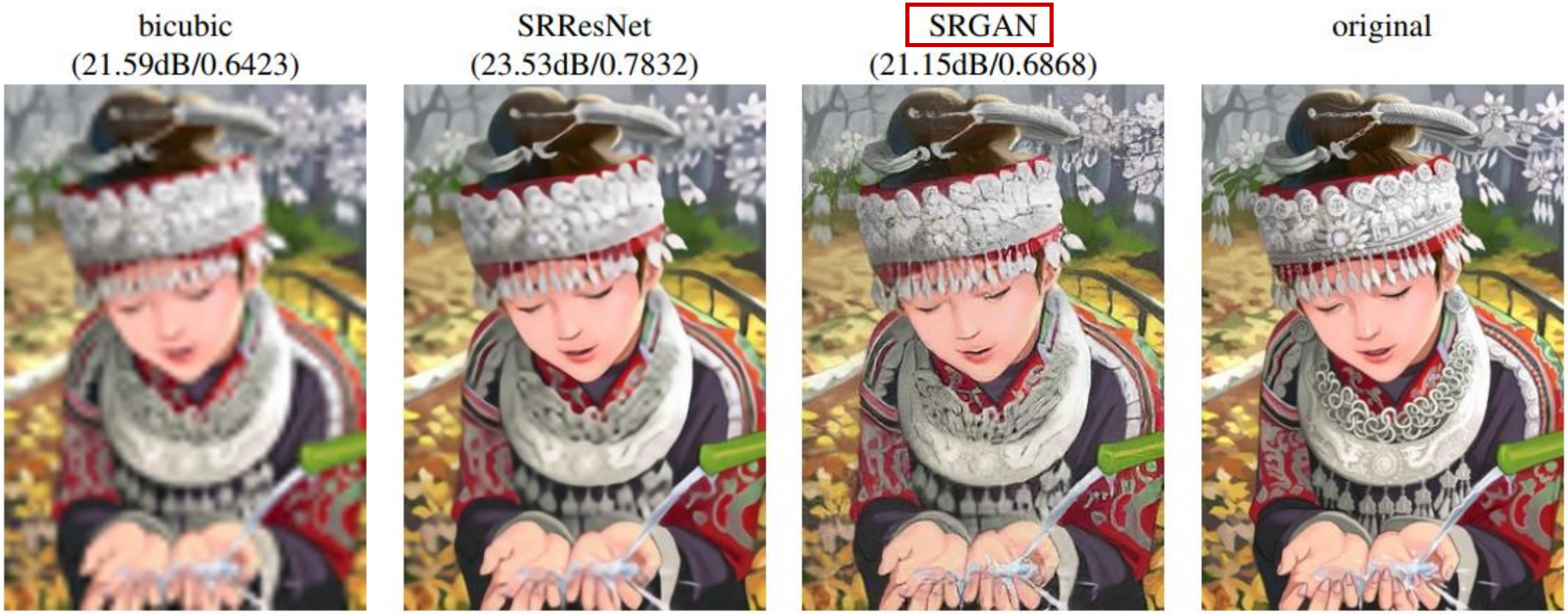
- Deep Convolutional Generative Adversarial Networks (DCGAN; Radford et al., 2016)
  - GAN 학습의 불안정성을 해결하기 위하여 생성함수와 판별함수의 표준적인 형태를 제공
  - 제공하는 CNN 구조를 통해 고해상도 이미지에서도 GAN보다 잘 작동하도록 함



(출처 : Radford et al, 2016)

GAN의 응용

- 이미지 해상도 복원 (SRGAN; Ledig et al., 2016)
  - 학습 결과



(출처 : Ledig et al., 2016)

GAN의 응용

- 이미지 변환 (pix2pix; Isola et al., 2016)

  - 학습 결과

(출처 : Isola et al., 2016)

DCGAN 예제

DCGAN

```
randomDim = 100

(X_train, y_train), (X_test, y_test) = mnist.load_data()
X_train = (X_train.astype(np.float32) - 127.5)/127.5
X_train = X_train[:, np.newaxis, :, :]

adam = Adam(lr=0.0002, beta_1=0.5)

generator = Sequential()
generator.add(Dense(128*7*7, input_dim=randomDim, kernel_initializer=initializers.RandomNormal(stddev=0.02)))
generator.add(LeakyReLU(0.2))
generator.add(Reshape((128, 7, 7)))
generator.add(UpSampling2D(size=(2, 2)))
generator.add(Conv2D(64, kernel_size=(5, 5), padding='same'))
generator.add(LeakyReLU(0.2))
generator.add(UpSampling2D(size=(2, 2)))
generator.add(Conv2D(1, kernel_size=(5, 5), padding='same', activation='tanh'))
generator.compile(loss='binary_crossentropy', optimizer=adam)
```

- 생성함수 구조 정의

```
discriminator = Sequential()
discriminator.add(Conv2D(64, kernel_size=(5, 5), strides=(2, 2), padding='same', input_shape=(1, 28, 28), kernel_initializer=initializers.RandomNormal(stddev=0.02)))
discriminator.add(LeakyReLU(0.2))
discriminator.add(Dropout(0.3))
discriminator.add(Conv2D(128, kernel_size=(5, 5), strides=(2, 2), padding='same'))
discriminator.add(LeakyReLU(0.2))
discriminator.add(Dropout(0.3))
discriminator.add(Flatten())
discriminator.add(Dense(1, activation='sigmoid'))
discriminator.compile(loss='binary_crossentropy', optimizer=adam)
```

- 판별함수 구조 정의

DCGAN 예제

DCGAN

- 학습 단계

```
ganInput = Input(shape=(randomDim,))
x = generator(ganInput)
ganOutput = discriminator(x)
gan = Model(inputs=ganInput, outputs=ganOutput)
gan.compile(loss='binary_crossentropy', optimizer=adam)

dLosses = []
gLosses = []

def train(epochs=1, batchSize=128):
    batchCount = X_train.shape[0] / batchSize
    print 'Epochs:', epochs
    print 'Batch size:', batchSize
    print 'Batches per epoch:', batchCount

    for e in xrange(1, epochs+1):
        print '-'*15, 'Epoch %d' % e, '-'*15
        for _ in tqdm(xrange(batchCount)):
            noise = np.random.normal(0, 1, size=[batchSize, randomDim])
            imageBatch = X_train[np.random.randint(0, X_train.shape[0], size=batchSize)]
            generatedImages = generator.predict(noise)
            X = np.concatenate([imageBatch, generatedImages])
            yDis = np.zeros(2*batchSize)
            yDis[:batchSize] = 0.9
            discriminator.trainable = True
            dloss = discriminator.train_on_batch(X, yDis)
            noise = np.random.normal(0, 1, size=[batchSize, randomDim])
            yGen = np.ones(batchSize)
            discriminator.trainable = False
            gloss = gan.train_on_batch(noise, yGen)
```

- 이미지 생성

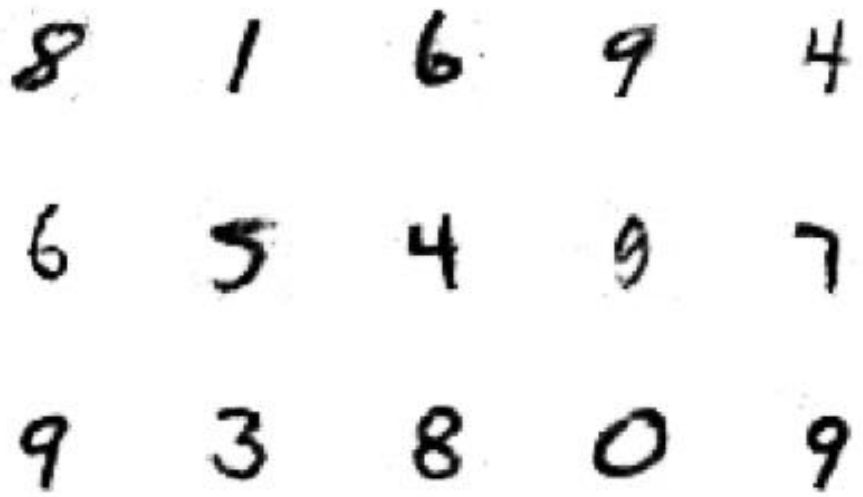
```
def plotGeneratedImages(epoch, examples=100, dim=(10, 10), figsize=(10, 10)):
    noise = np.random.normal(0, 1, size=[examples, randomDim])
    generatedImages = generator.predict(noise)

    plt.figure(figsize=figsize)
    for i in range(generatedImages.shape[0]):
        plt.subplot(dim[0], dim[1], i+1)
        plt.imshow(generatedImages[i, 0], interpolation='nearest', cmap='gray_r')
        plt.axis('off')
    plt.tight_layout()
    plt.savefig('images/dcgan_generated_image_epoch_%d.png' % epoch)
```

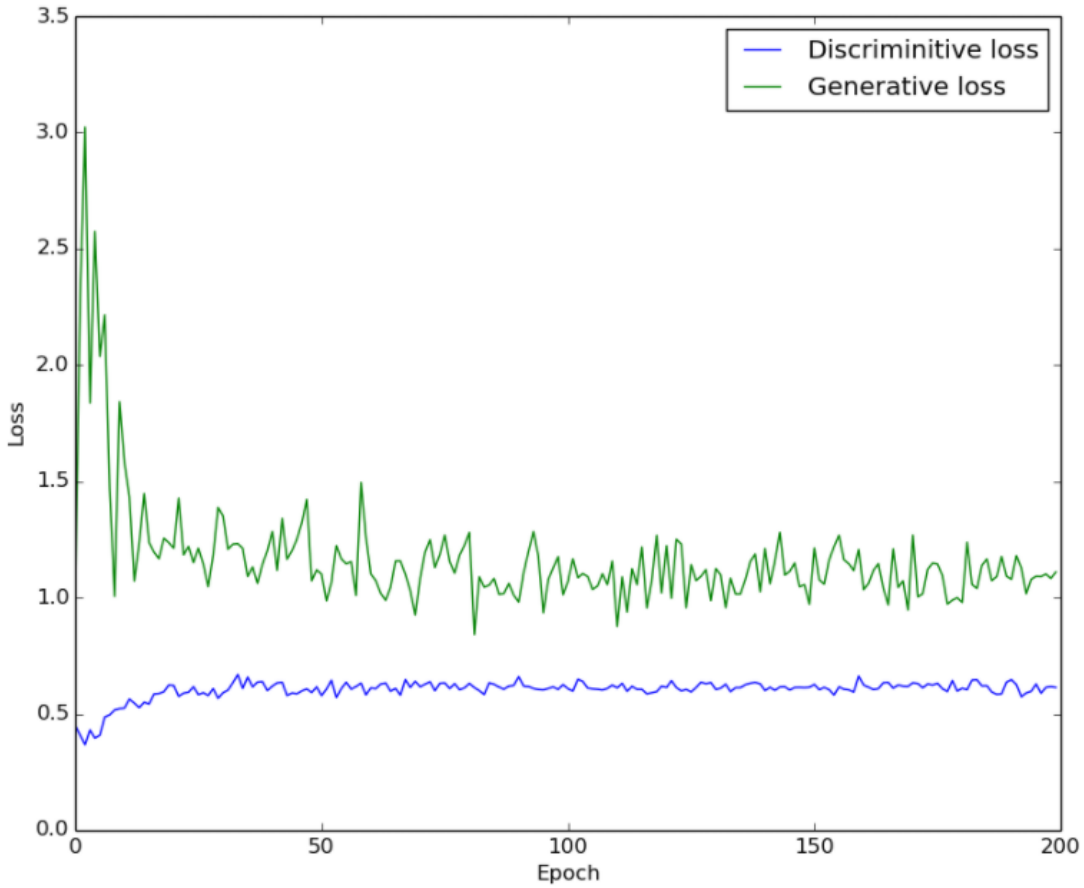
DCGAN 예제

DCGAN

- DCGAN 표본 추출 결과



- DCGAN 학습 과정 loss 변화



학습정리

- ✓ 인자 분석은 잠재 인자와 관측 변수 간의 선형 관계를 가정한 모형이며, EM 알고리즘을 통해 모수를 추정할 수 있다.
- ✓ 심층 생성 모형은 잠재 인자와 관측 변수 간의 관계를 심층 인공망 함수로 가정한 모형이다.
- ✓ 심층 생성 모형의 대표적인 추정 방법으로는 GAN 이 있으며, GAN은 다양한 분야에 응용될 수 있다.

딥러닝의 통계적이해

다음시간안내

11강. 순환신경망 (1)