GAN(Generative Adversarial Network) 적대적 신경망

명지대학교
MYONGJI UNIVERSITY

GAN (Generative Adversarial Networks)

- 돈을 벌기 위해 가짜 와인을 파는 사악한 고객들도 있습니다. 이 경우 가게 주인은 가짜 와인과 진품 와인을 구분

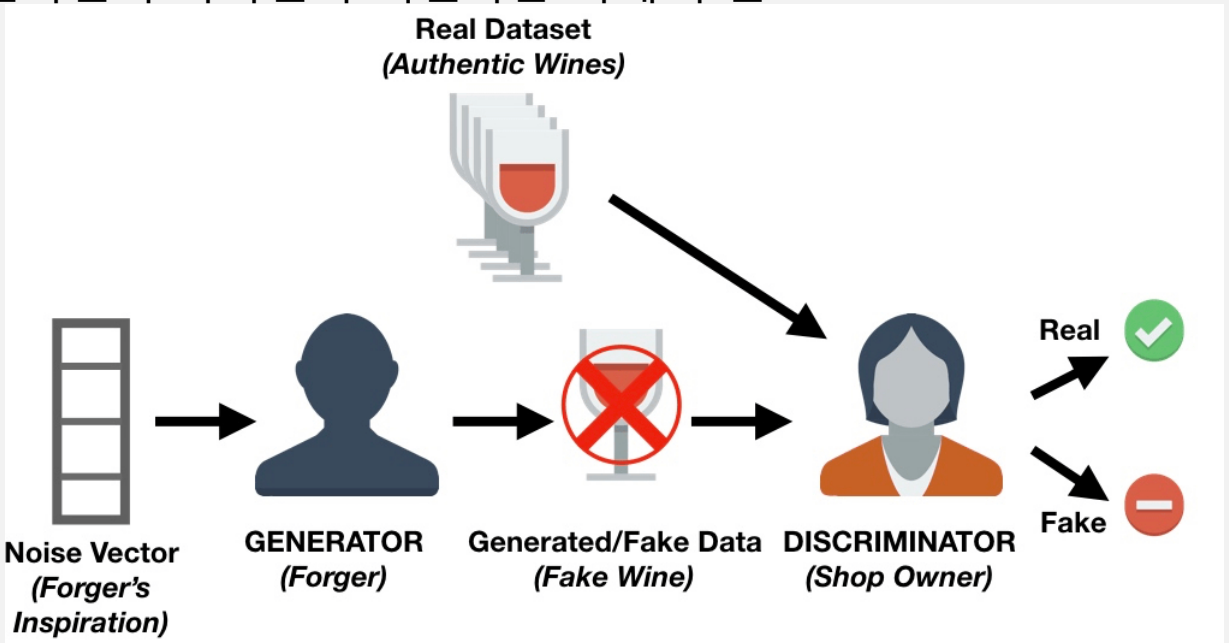
- 돈을 벌기 위해 가짜 와인을 팔기
  - 가게 주인은 가짜 와인과 진품 와인을 구분
  - 가짜 와인 위조범은 진품 와인처럼 만들기 위해서 다른 기술을 계속 시도

GAN (Generative Adversarial Networks)

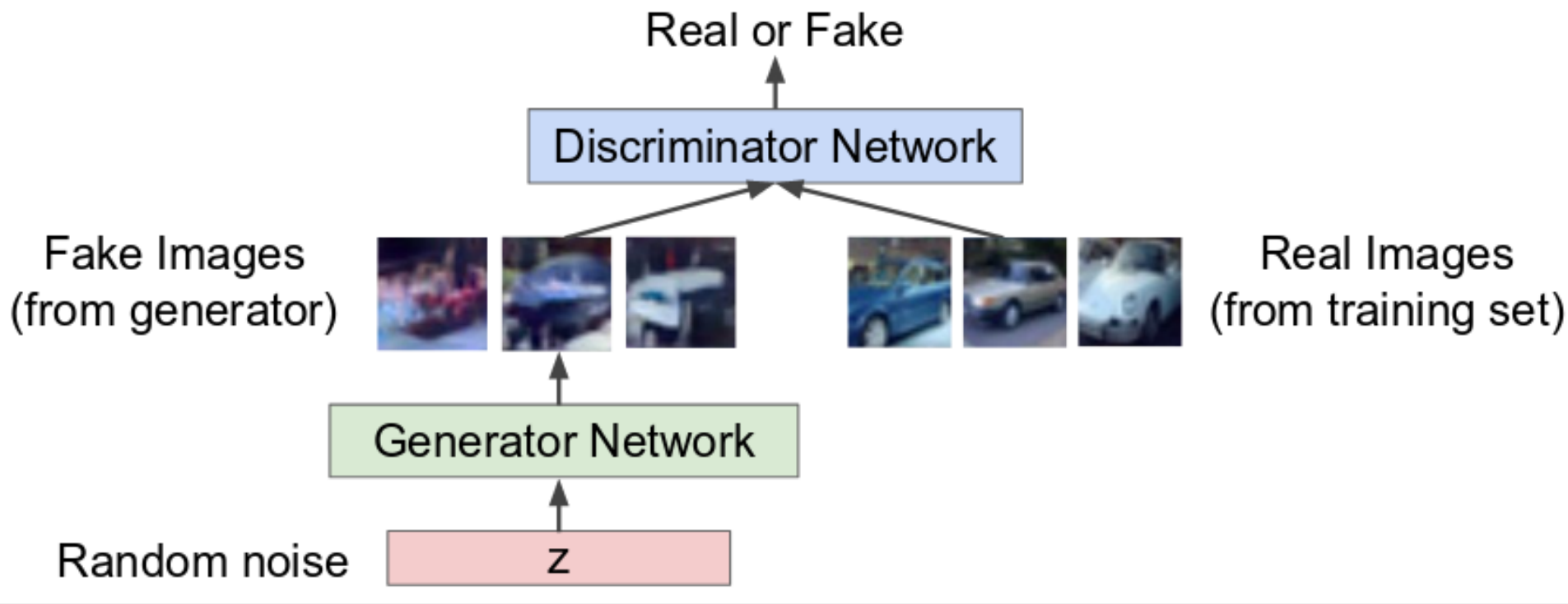
위조범과 위조를 구분해내려는 자의 경쟁이 GAN의 기본 아이디어

GANs의 주요 구성요소

- Generator(생성자) : 가짜 와인 위조범
- Discriminator(식별자) : 와인이 진짜인지 가짜인지 식별하는 가게 주인



GAN (Generative Adversarial Networks)



생성자 (Generator)

가상의 이미지를 만들어 내는 공장

DCGAN(Deep Convolutional GAN)은 생성자가 가짜 이미지 생성시 CNN 이용

- 차이 :
  - 1. 옵티마이저를 사용하는 최적화 과정이나 컴파일 과정이 없음
    - 이유 : 판별과 학습이 이곳 생성자에서 일어나는 게 아니기 때문에
  - 2. 일부 매개 변수를 삭제하는 풀링 과정이 없고 대신 패딩 과정이 포함
    - 이유 : 입력 크기와 출력 크기를 똑같이 맞추기
- 꼭 필수적으로 필요한 옵션 :
  - 배치 정규화 (평균0 표준편차1이 되도록 재배치)
    - 다음 층으로 입력될 값을 일정하게 재배치 하는 역할
    - Keras의 BatchNormalization() 함수 이용
  - 활성화 함수로 ReLU 함수 판별자로 넘기 주기 직전에는 tanh( )함수

생성자 (Generator)

임의로 정한 노드수

이미지 최초의 크기

100 차원 크기의 랜덤 벡터

```
#생성자 모델을 만듭니다.
generator = Sequential()
generator.add(Dense(128*7*7, input_dim=100, activation=LeakyReLU(0.2)))
generator.add(BatchNormalization())
generator.add(Reshape((7, 7, 128)))
generator.add(UpSampling2D())
generator.add(Conv2D(64, kernel_size=5, padding='same'))
generator.add(BatchNormalization())
generator.add(Activation(LeakyReLU(0.2)))
generator.add(UpSampling2D())
generator.add(Conv2D(1, kernel_size=5, padding='same', activation='tanh'))
```

이미지의 가로 세로의 크기를 2배씩 늘려주기

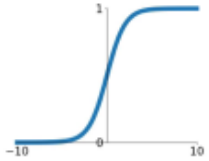
한번더 컨벌루션 과정을 거친 후 판별자로 값을 넘길 준비를 마치기

Activation Function

Activation Functions

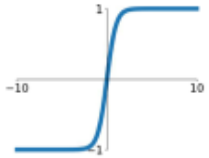
Sigmoid

$$\sigma(x) = \frac{1}{1+e^{-x}}$$



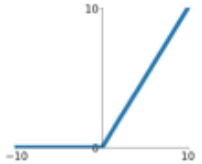
tanh

$$\tanh(x)$$



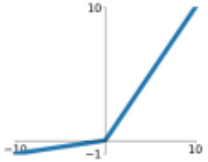
ReLU

$$\max(0, x)$$



Leaky ReLU

$$\max(0.1x, x)$$

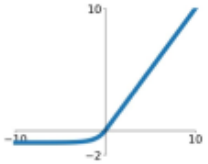


Maxout

$$\max(w_1^T x + b_1, w_2^T x + b_2)$$

ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



판별자 (Discriminator)

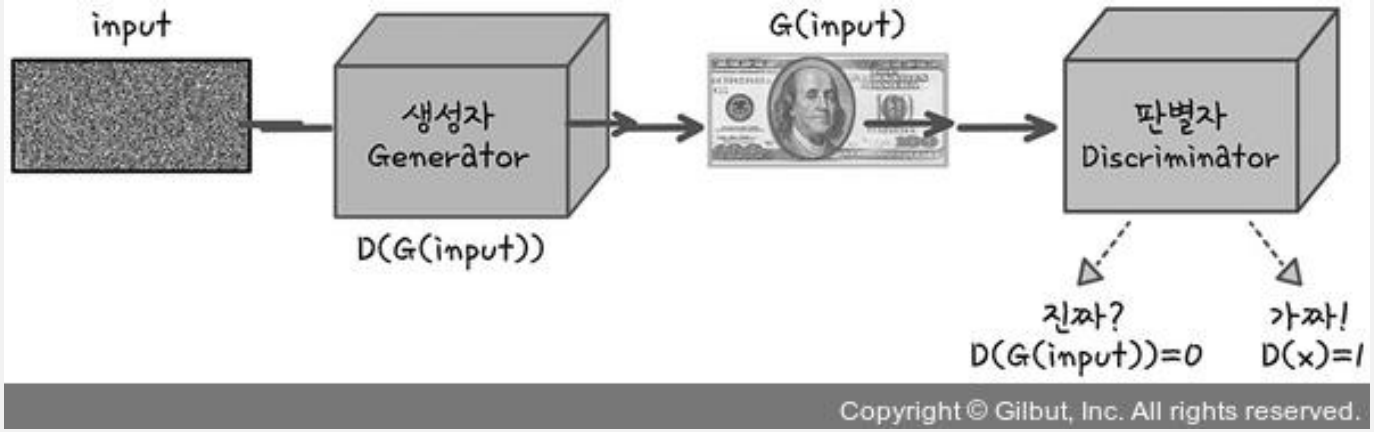
- **컨벌루션 신경망이 구별하는데 최적화된 알고리즘 이기때문에 그대로 사용 가능**
- **진짜 인지 가짜인지 둘 중 하나를 결정하는 문제**

```
#판별자 모델을 만듭니다.
discriminator = Sequential()
discriminator.add(Conv2D(64, kernel_size=5, strides=2, input_shape=(28,28,1), padding="same"))
discriminator.add(Activation(LeakyReLU(0.2)))
discriminator.add(Dropout(0.3))
discriminator.add(Conv2D(128, kernel_size=5, strides=2, padding="same"))
discriminator.add(Activation(LeakyReLU(0.2)))
discriminator.add(Dropout(0.3))
discriminator.add(Flatten())
discriminator.add(Dense(1, activation='sigmoid'))
discriminator.compile(loss='binary_crossentropy', optimizer='adam')
discriminator.trainable = False
```

판별이 끝나고 나면 판별자 자신이 학습되지 않게끔 학습 기능을 꺼주기

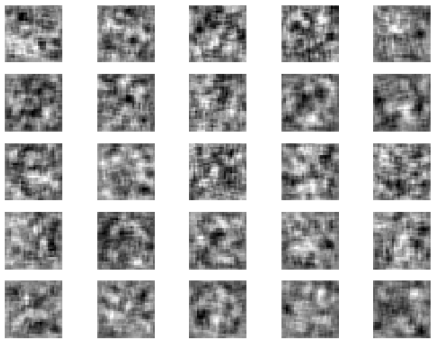
적대적 신경망 실행하기

생성자와 판별자를 연결시킨다는 것은 생성자에서 나온 출력물을 판별자에 넣어서 진위 여부를 판별하게 만든다.



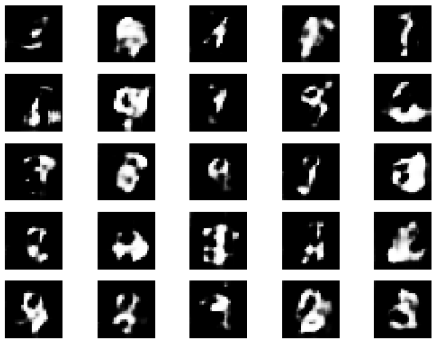

```
# 생성자와 판별자 모델을 연결시키는 gan 모델을 만듭니다.
ginput = Input(shape=(100,))
dis_output = discriminator(generator(ginput))
gan = Model(ginput, dis_output)
gan.compile(loss='binary_crossentropy', optimizer='adam')
gan.summary()
```

결과물

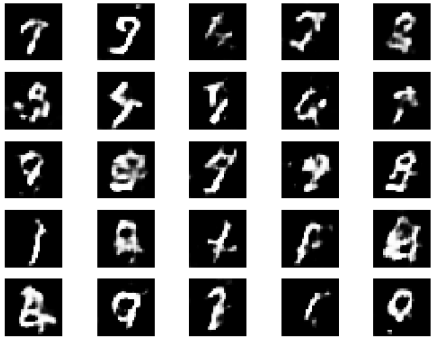


Epoch 0

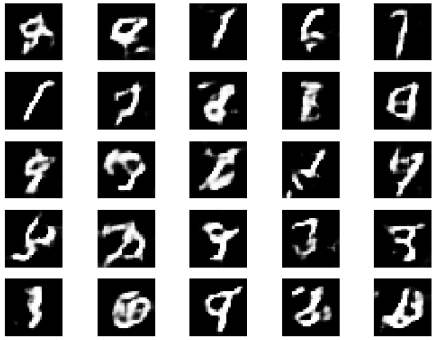
Epoch 400



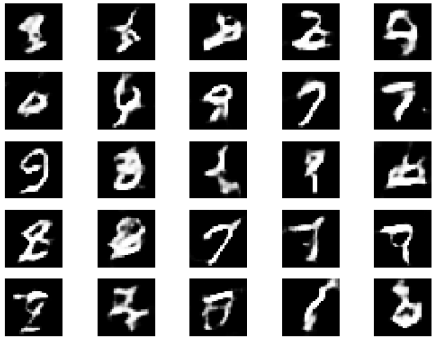
Epoch 800



Epoch 1600

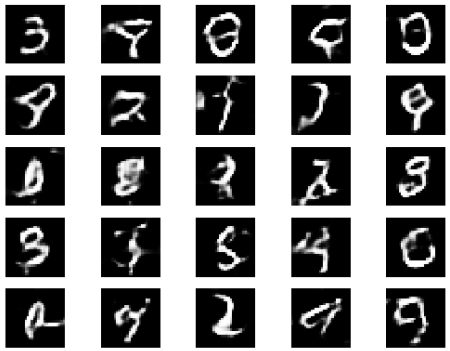


Epoch 2000

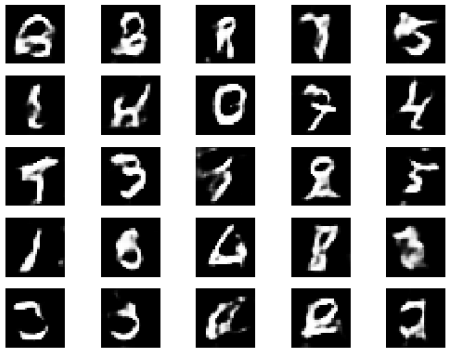


Epoch 2400

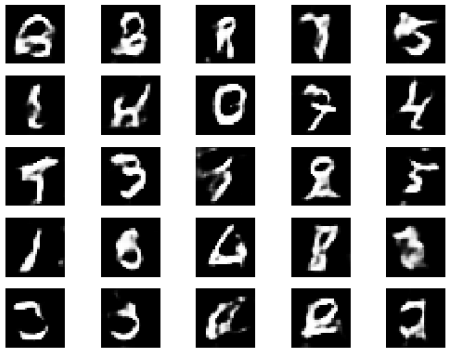
결과물



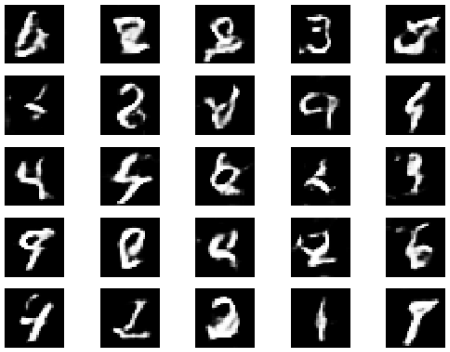
Epoch 2800



Epoch 3200



Epoch 3600



Epoch 4000

GAN 예

https://thispersondoesnotexist.com/

http://nvidia-research-mingyuliu.com/gaugan/

GAN 예

https://github.com/gtoubassi/mnist-gan

Supervised Learning 지도 학습

- Supervised Learning 지도 학습
- 대표적인 모델로 **Discriminative Model**이 있으며, 로지스틱 회귀분석, 뉴럴 네트워크 등이 해당된다. Input에 해당하는 클래스를 맞추기 위해 학습하게 된다. 예를 들어 남자냐, 여자냐를 구분하는 것이다.

Unsupervised Learning 비지도 학습

- label이 없는 데이터를 잘 학습하는 것이다. **Generative Model**에는 Naive Bayes, Gaussian discriminant analysis (GDA, 가우시안 판별 분석)이 있다.학습 데이터에서 분포를 학습한 뒤, 학습 데이터와 유사한 데이터를 만드는 게 목표이다. 학습이 진행될 수록 실제 데이터와 생성한 데이터의 분포가 유사해진다.

https://tykimos.github.io/2018/10/10/Understanding_Generative_Adversarial_Nets/

Auto Encorder

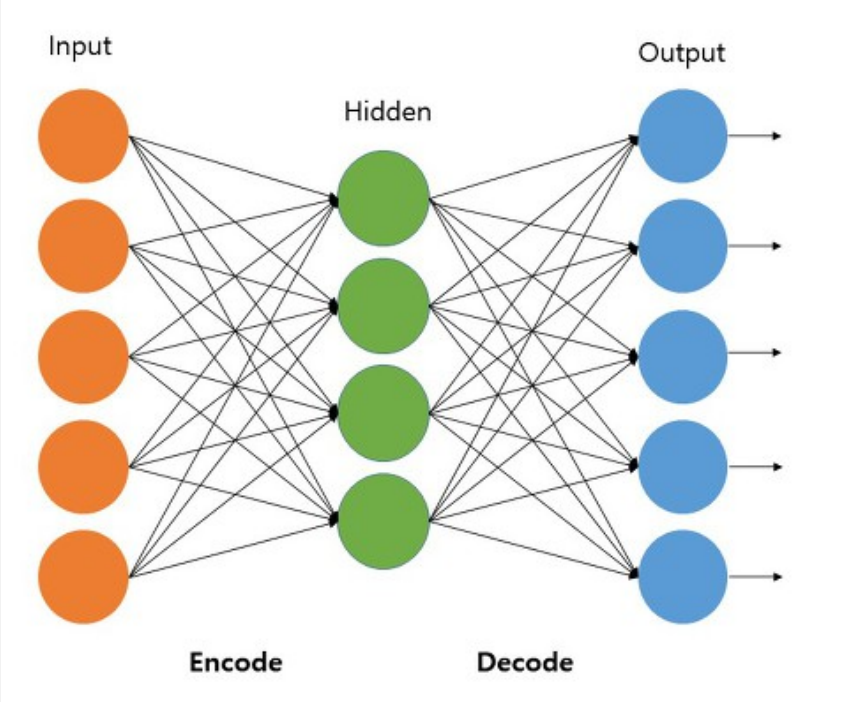
명지대학교
MYONGJI UNIVERSITY

오토인코더(Auto Encoder)

- 데이터를 효율적으로 coding하기(압축) 위한 딥러닝
- 어떤 데이터를 효율적으로 나타내기 위해서 고차원을 저차원으로 차원 축소하는 방법
- 입력층과 출력층을 같도록 구성한뒤 중간에 은닉층을 적재

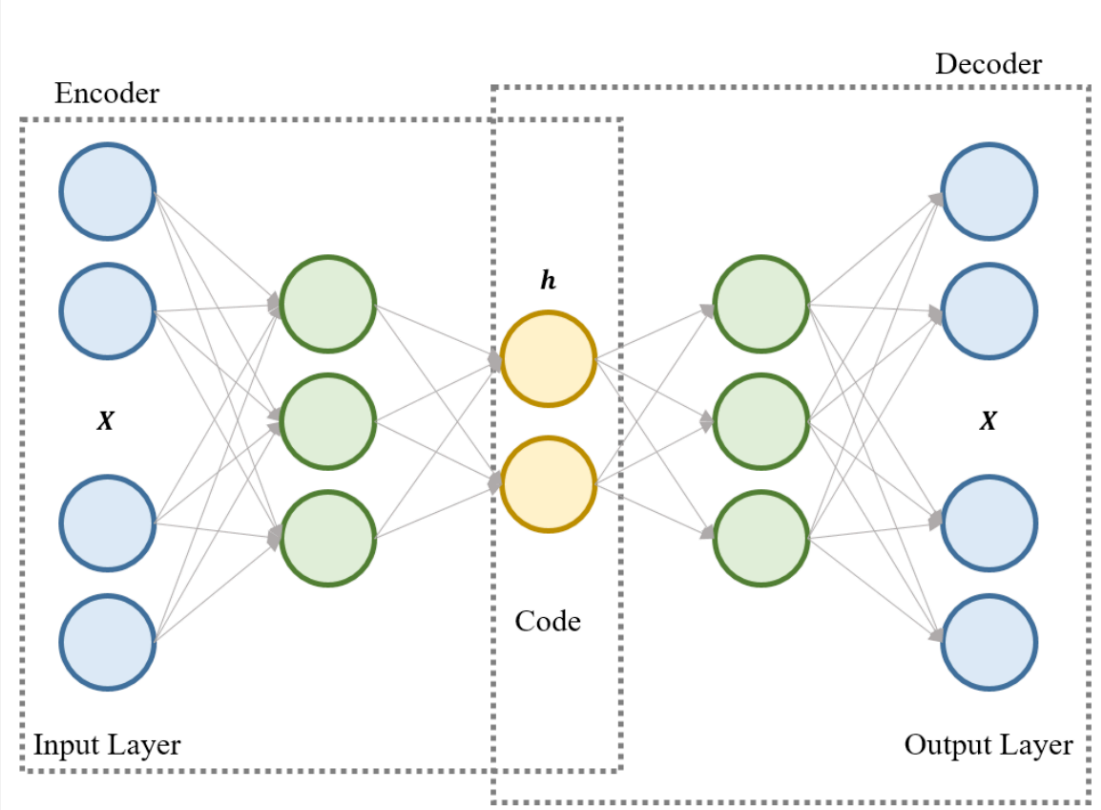
오토인코더(Auto Encoder)

- 은닉층을 기준으로 좌우 대칭이 되도록
- 데이터의 차원 축소 및 특징 추출이 목적인 경우 입력층의 노드 수보다 작게
- 인코더를 이용하면 X를 어떤 코드값 h로 특징 추출 가능 이때 h의 크기를 X보다 작게 구성하면 차원 축소 가능
- 인코더를 이용하여 차원 축소 및 특징 추출



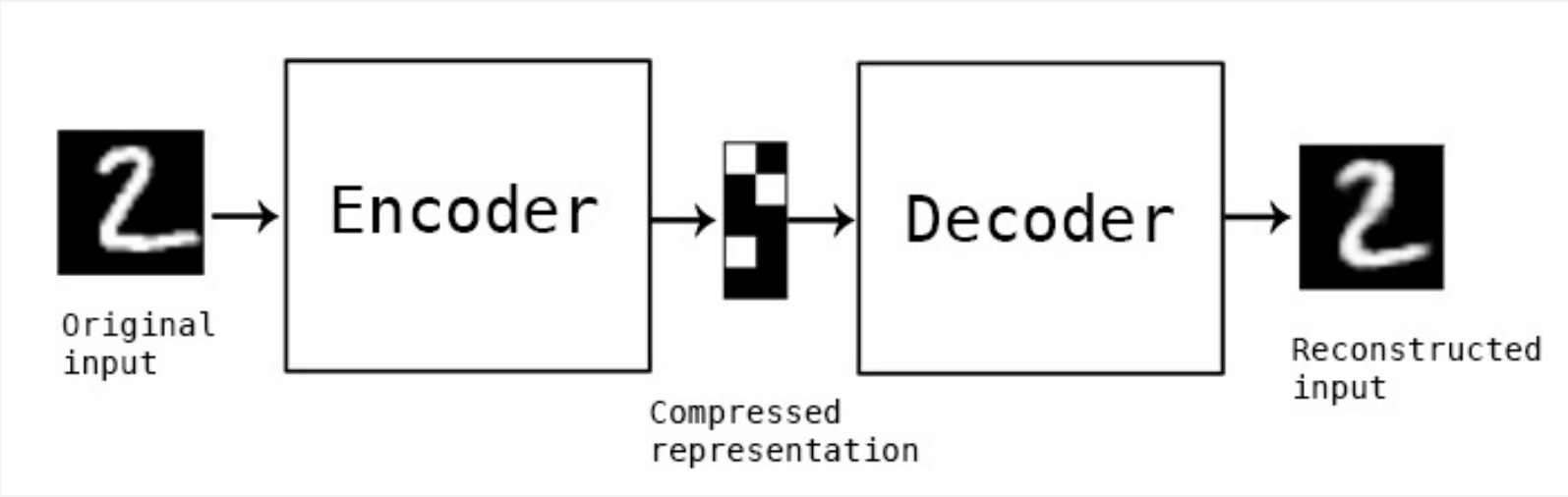
Stacked AutoEncorder

- **Single Autoencorder는 단층의 Encorder와 단층의 Decorder로 구성**
- **여러 층을 적재하여 stacked autoencorder**



오토인코더(Auto Encoder)

- 신경망 기반 비지도학습 기법으로 차원축소, 이미지 압축, 이미지 노이즈 제거, 이미지 생성 등에 사용되는 신경망 구조
- 오토인코더의 가장 큰 특징은 입력을 다시 라벨



MNIST를 이용한 오토인코더

```
from tensorflow.keras.datasets import mnist
from tensorflow.keras.models import Sequential, Model
from tensorflow.keras.layers import Input, Dense, Conv2D, MaxPooling2D, UpSampling2D, Flatten, Res
hape
import matplotlib.pyplot as plt
import numpy as np

#MNIST데이터 셋을 불러옵니다.

(X_train, _), (X_test, _) = mnist.load_data()
X_train = X_train.reshape(X_train.shape[0], 28, 28, 1).astype('float32') / 255
X_test = X_test.reshape(X_test.shape[0], 28, 28, 1).astype('float32') / 255
```

MNIST를 이용한 오토인코더

```
#생성자 모델을 만듭니다.
autoencoder = Sequential()

# 인코딩 부분입니다.
autoencoder.add(Conv2D(16, kernel_size=3, padding='same', input_shape=(28,28,1), activation='relu'))
autoencoder.add(MaxPooling2D(pool_size=2, padding='same'))
autoencoder.add(Conv2D(8, kernel_size=3, activation='relu', padding='same'))
autoencoder.add(MaxPooling2D(pool_size=2, padding='same'))
autoencoder.add(Conv2D(8, kernel_size=3, strides=2, padding='same', activation='relu'))

# 디코딩 부분이 이어집니다.
autoencoder.add(Conv2D(8, kernel_size=3, padding='same', activation='relu'))
autoencoder.add(UpSampling2D())
autoencoder.add(Conv2D(8, kernel_size=3, padding='same', activation='relu'))
autoencoder.add(UpSampling2D())
autoencoder.add(Conv2D(16, kernel_size=3, activation='relu'))
autoencoder.add(UpSampling2D())
autoencoder.add(Conv2D(1, kernel_size=3, padding='same', activation='sigmoid'))
```

MNIST를 이용한 오토인코더

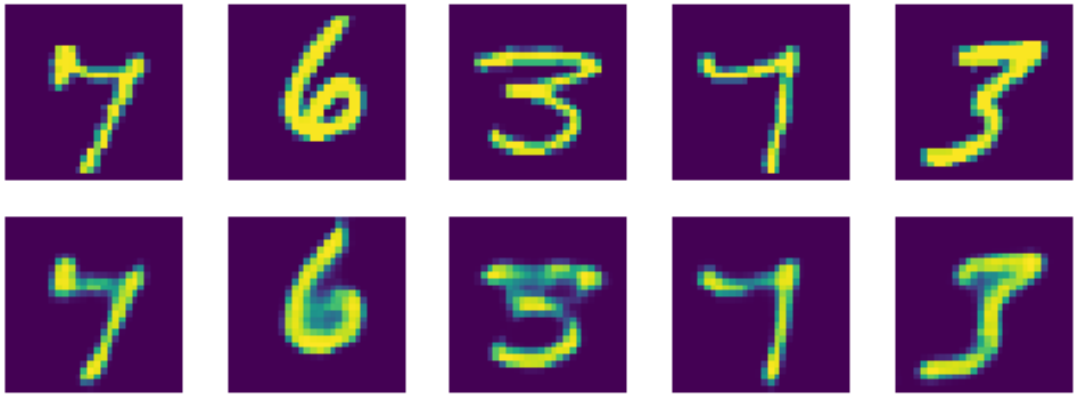
```
# 전체 구조를 확인해 봅니다.
autoencoder.summary()

# 컴파일 및 학습을 하는 부분입니다.
autoencoder.compile(optimizer='adam', loss='binary_crossentropy')
autoencoder.fit(X_train, X_train, epochs=50, batch_size=128, validation_data=(X_test, X_test))

#학습된 결과를 출력하는 부분입니다.
random_test = np.random.randint(X_test.shape[0], size=5)  #테스트할 이미지를 랜덤하게 불러옵니다.
ae_imgs = autoencoder.predict(X_test)  #앞서 만든 오토인코더 모델에 집어 넣습니다.
```

MNIST를 이용한 오토인코더

epochs=10



테스트할 이미지

오토 인코더에 의해 생성된 이미지 이미지

MNIST를 이용한 오토인코더

epochs=50

테스트할 이미지

오토 인코더에 의해 생성된 이미지 이미지