CUAI 딥러닝 논문 구현 스터디 CV 2팀

2024.11.26

발표자 : 양지훈

스터디원 소개 및 만남 인증

스터디원 1 : 양지훈
전자전기공학부

스터디원 2 : 임현오
응용통계학과

스터디원 2 : 정현석
전자전기공학부

논문 선택

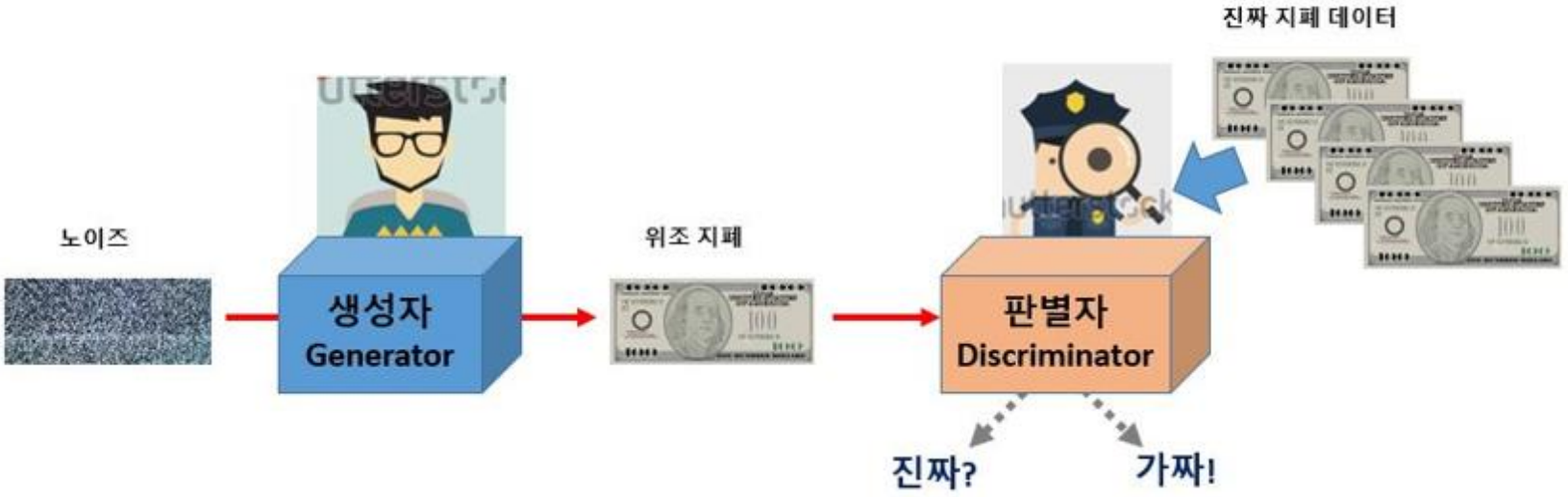
Generative Adversarial Nets

Ian J. Goodfellow, Jean Pouget-Abadie[*], Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair[†], Aaron Courville, Yoshua Bengio[‡]
Département d'informatique et de recherche opérationnelle
Université de Montréal
Montréal, QC H3C 3J7

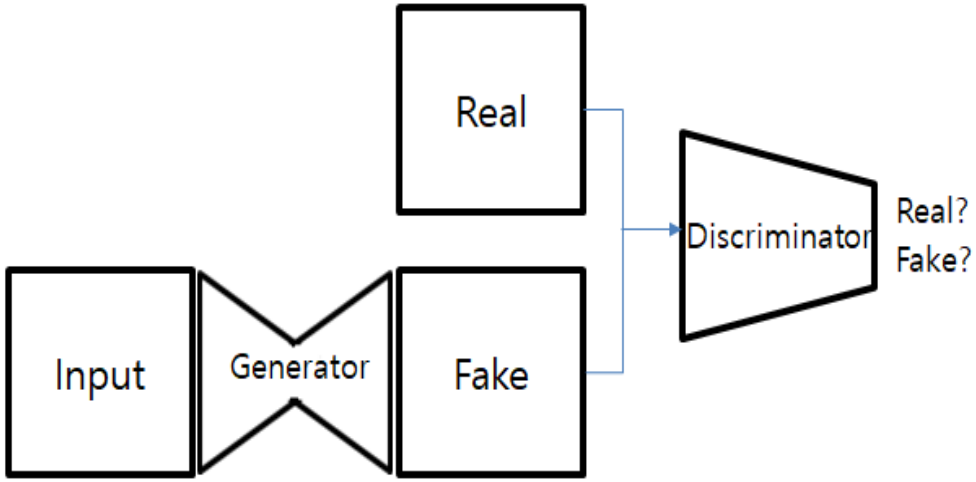
Abstract

We propose a new framework for estimating generative models via an adversarial process, in which we simultaneously train two models: a generative model G that captures the data distribution, and a discriminative model D that estimates the probability that a sample came from the training data rather than G. The training procedure for G is to maximize the probability of D making a mistake. This framework corresponds to a minimax two-player game. In the space of arbitrary functions G and D, a unique solution exists, with G recovering the training data distribution and D equal to $\frac{1}{2}$ everywhere. In the case where G and D are defined by multilayer perceptrons, the entire system can be trained with backpropagation. There is no need for any Markov chains or unrolled approximate inference networks during either training or generation of samples. Experiments demonstrate the potential of the framework through qualitative and quantitative evaluation of the generated samples.

개념적 이해



논문 리뷰

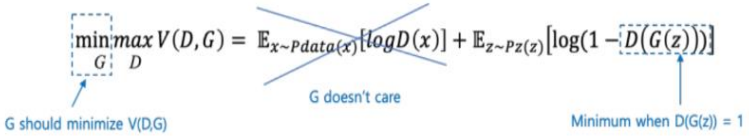


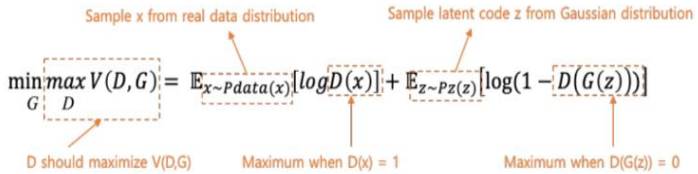
▪ GAN은 생성자(Generator)와 판별자(Discriminator), 두 개의 신경망이 상호 경쟁적으로 학습하는 생성 모델이다

▪ 생성자(Generator)는 실제 데이터와 유사한 가짜 데이터를 생성하며, 판별자(Discriminator)는 실제 데이터와 가짜 데이터를 구별하는 역할을 한다

논문 리뷰

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim Pdata(x)}[logD(x)] + \mathbb{E}_{z\sim Pz(z)}[\log(1 - D(G(z)))]$$





- GAN의 수식 V(D,G)**에서 Generator의 목적은 V(D,G)를 최소화하는 것이며, Discriminator의 목적은 V(D,G)를 최대화하는 것**이다

- x는 실제 데이터이며, G(z)는 noise z를 바탕으로 G가 생성한 가짜 데이터이다

- D의 목적은 가짜 데이터에 대해 0을 출력해야 하고, 실제 데이터에 대해 1을 출력해야 한다 [D(G(z))=0 & D(x)=1]

- 따라서 **logD(x) 및 log(1-D(G(z))가 최대가 되는 것이며 V(D,G)를 최대화한다**

- G의 목적은 D가 가짜 데이터에 대해 1을 출력해야 한다 [D(G(z))=1]

- 따라서 log(1-D(G(z))가 음의 무한대로 발산해 V(D,G)를 최소화한다

논문 구현

Generator

```
def make_generator_model():
    model = tf.keras.Sequential()
    model.add(layers.Dense(7*7*256, use_bias=False, input_shape=(100,)))
    model.add(layers.BatchNormalization())
    model.add(layers.LeakyReLU())

    model.add(layers.Reshape((7, 7, 256)))
    assert model.output_shape == (None, 7, 7, 256) # 주목: 배치사이즈로 None이 주어집니다.

    model.add(layers.Conv2DTranspose(128, (5, 5), strides=(1, 1), padding='same', use_bias=False))
    assert model.output_shape == (None, 7, 7, 128)
    model.add(layers.BatchNormalization())
    model.add(layers.LeakyReLU())

    model.add(layers.Conv2DTranspose(64, (5, 5), strides=(2, 2), padding='same', use_bias=False))
    assert model.output_shape == (None, 14, 14, 64)
    model.add(layers.BatchNormalization())
    model.add(layers.LeakyReLU())

    model.add(layers.Conv2DTranspose(1, (5, 5), strides=(2, 2), padding='same', use_bias=False, act
    assert model.output_shape == (None, 28, 28, 1)

    return model
```

Discriminator

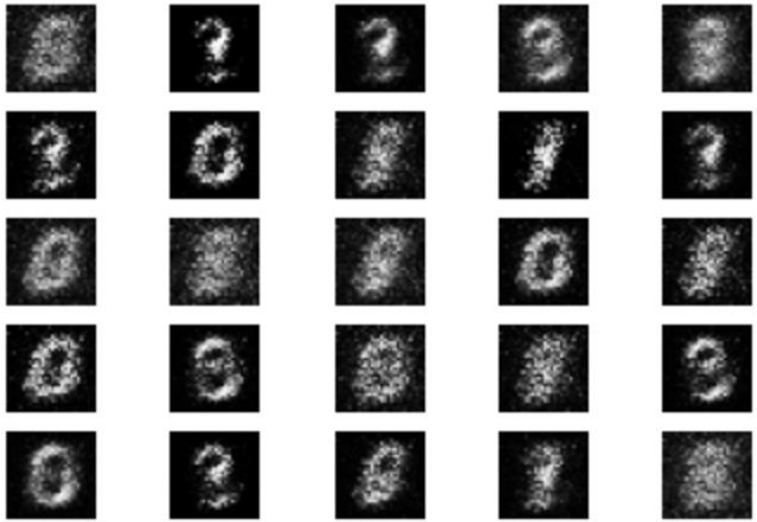
```
def make_discriminator_model():
    model = tf.keras.Sequential()
    model.add(layers.Conv2D(64, (5, 5), strides=(2, 2), padding='same',
                                     input_shape=[28, 28, 1]))
    model.add(layers.LeakyReLU())
    model.add(layers.Dropout(0.3))

    model.add(layers.Conv2D(128, (5, 5), strides=(2, 2), padding='same'))
    model.add(layers.LeakyReLU())
    model.add(layers.Dropout(0.3))

    model.add(layers.Flatten())
    model.add(layers.Dense(1))

    return model
```

논문 구현



1000epoch

감사합니다!

모두 포근한 연말 보내시길 바래요:)