generative adversarial nets(GAN)

Ian J. Goodfellow , Jean Pouget-Abadie , Mehdi Mirza, Bing Xu et al.

Departement d'informatique et de recherche op ´ erationnelle

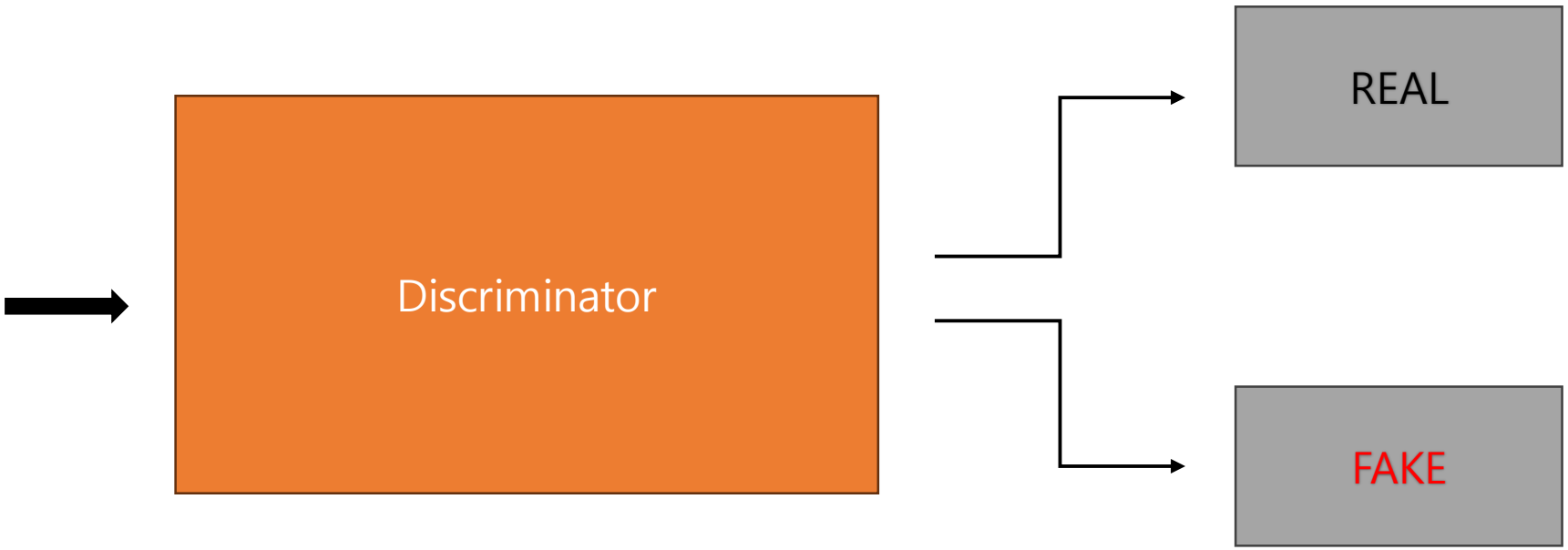
´ Universite de Montr ´ eal

Advances in Neural Information Processing Systems 27 (NIPS 2014)

GAN(Generative adversarial networks)

- GAN은 기본적으로 두개의 다른 신경망 (Generator와 Discriminator) 간의 적대적인 관계로 대립 (Adversarial)하며 서로의 성능을 점차 개선해 나가는 것
  - 생성 모델 G : 데이터의 분포를 학습
  - 판별 모델 D : 이미지를 실제 (학습을 진행할 때 사용하는 Training Data) 또는 가짜 (임의로 만든 Generated Data) 인지 분류하는 모델

Discriminator

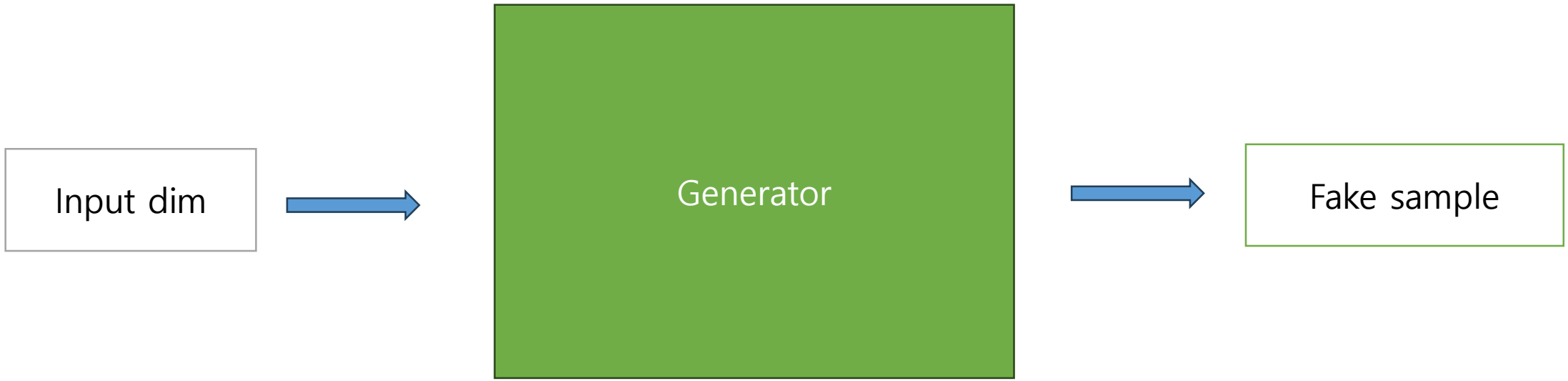


Discriminator

- 신경망의 구조로 Linear -활성화 함수로 간단하게 구성되어 있음
- Fake와 real로 분류하기 위해 이진분류를 위한 sigmoid 함수를 사용

```
class Discriminator(nn.Module):
    def __init__(self, in_features):
        super().__init__()
        self.disc = nn.Sequential(s
            nn.Linear(in_features, 128),
            nn.LeakyReLU(0,1),
            nn.Linear(128, 1),
            nn.Sigmoid(),
        )
    def forward(self ,x ):
        return self.disc(x)
```

Generator



Generator

- 입력 값으로 벡터 z_dim과 출력 값인 img_dim을 받는다
- Linear – LeakyReLU – Linear – Tanh을 구성
- Tanh() 함수를 쓰면 출력값을 -1~1 사이로 맞출 수 있음

```
class Generator(nn.Module):
    def __init__(self, z_dim, img_dim):
        super().__init__()
        self.gen = nn.Sequential(
            nn.Linear(z_dim,256),
            nn.LeakyReLU(0,1),
            nn.Linear(256, img_dim),
            nn.Tanh(),
        )
    def forward(self ,x):
        return self.gen(x)
```

Hyperparameters

- 입력 벡터인 z_dim는 64로 설정
- Batch_size는 32
- Epoch은 50으로 설정

```
device = "cuda" if
torch.cuda.is_available() else "cpu"
lr = 3e-4
z_dim = 64 # 128, 256
image_dim = 28 * 28 * 1 # 784
batch_size = 32
num_epochs = 50
```

Latent vector

- 이미지는 28 * 28 -> 784개의 값으로 나열이 가능
- 생성모델은 픽셀 값이 임의의 값을 취했을 때의 동시확률 분포를 구하는 것과 같음
- 784차원의 모든 값의 분포에 대해서 확률을 구하는 것 확률 밀도 분포를 알게 되면 그확률 밀도를 기본으로 데이터를 인공적으로 생성할 수 있다.
- z_dim이라는 차원을 latent vector라고 하고 이것의 차원을 늘려서 사진으로 만든다 하면
- 잠재 변수 z로 부터 이미지를 만들고 이를 학습하면서 실제 이미지의 확률 분포를 가중치로써 학습한다

prepare

- Discriminator는 image의 차원을 input으로 받음
- Generator는 input으로 latent벡터의 차원을 입력받고 ouput으로 image의 차원을 받음
- Optimizer는 adam을 사용
- 손실함수로는 torch의 binary cross entropy를 사용(0,1로 분류)

Training

- 실제와 생성된 사진을 disc의 input으로 넣어서 학습 시킴 이때의 손실함수의 값은 서로의 손실의 평균으로 내고 학습시킨다
- Dic는 학습 데이터셋과 G에서 만든 데이터에 correct label을 할당할 확률을 최대 값으로 만들게 학습
- Gen은 Dic에서 correct label을 제대로 할당할 수 없을 정도로 학습 데이터 셋과 유사한 데이터를 만든 는것이 목표

Result

Deep Convolution Gan (DCGAN)

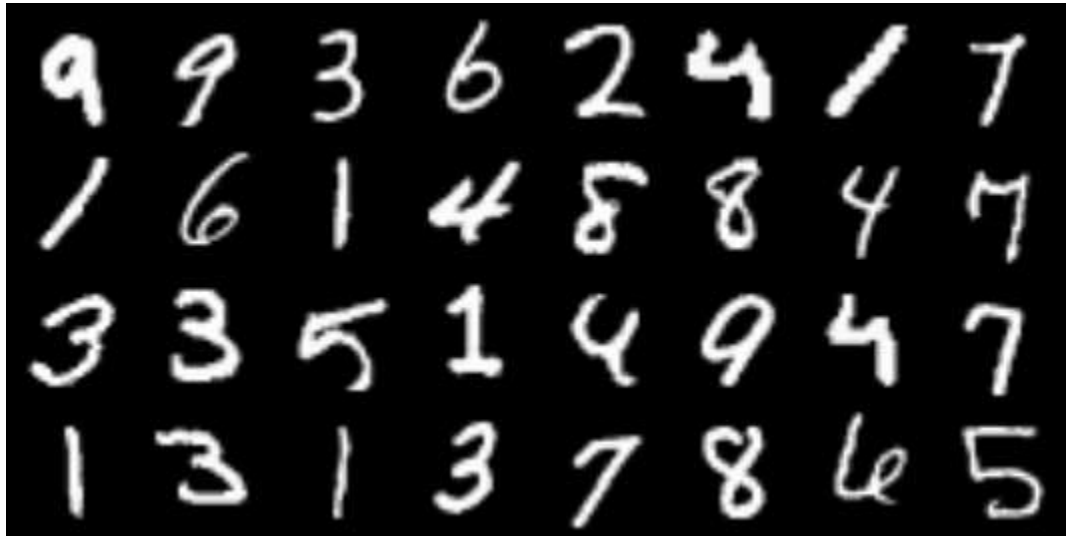
- GAN을 개선시키고 Convolution을 적용한 모델

- Contribute
  - 거의 대부분인 상황에서 안정적으로 학습이 가능하다
  - 학습이 된 판별기가 이미지 분류에서 다른 비지도 알고리즘들과 비교 했을 때 대등한 성능을 보인다
  - DCGAN이 학습한 filter들을 visualize하고 특정 filter가 특정 object를 생성하는 역할을 한다
  - DCGAN이 벡터 산술 연산이 가능한 성질을 갖는다. Semantic quality를 갖는다

Model Architecture

- 1. Max Polling To Strided Convoluton
  - Convolution 과정안에 polling 기법을 넣어서 사용 G에서는 Spatial Down sampling과정을 학습 , D에서는 Spatial up Sampling이 가능해짐
- 2. Eliminate Fully-Connected Layers
  - Noise vector z를 넣는 첫번째 layer와 마지막 softmax layer를 제외하고는 FC layer를 제외
- 3. Batch Normalization 추가
  - 모든 layer에는 추가하지 않고, G의 output layer와 D의 input layer에는 추가하지 않음

Experiment

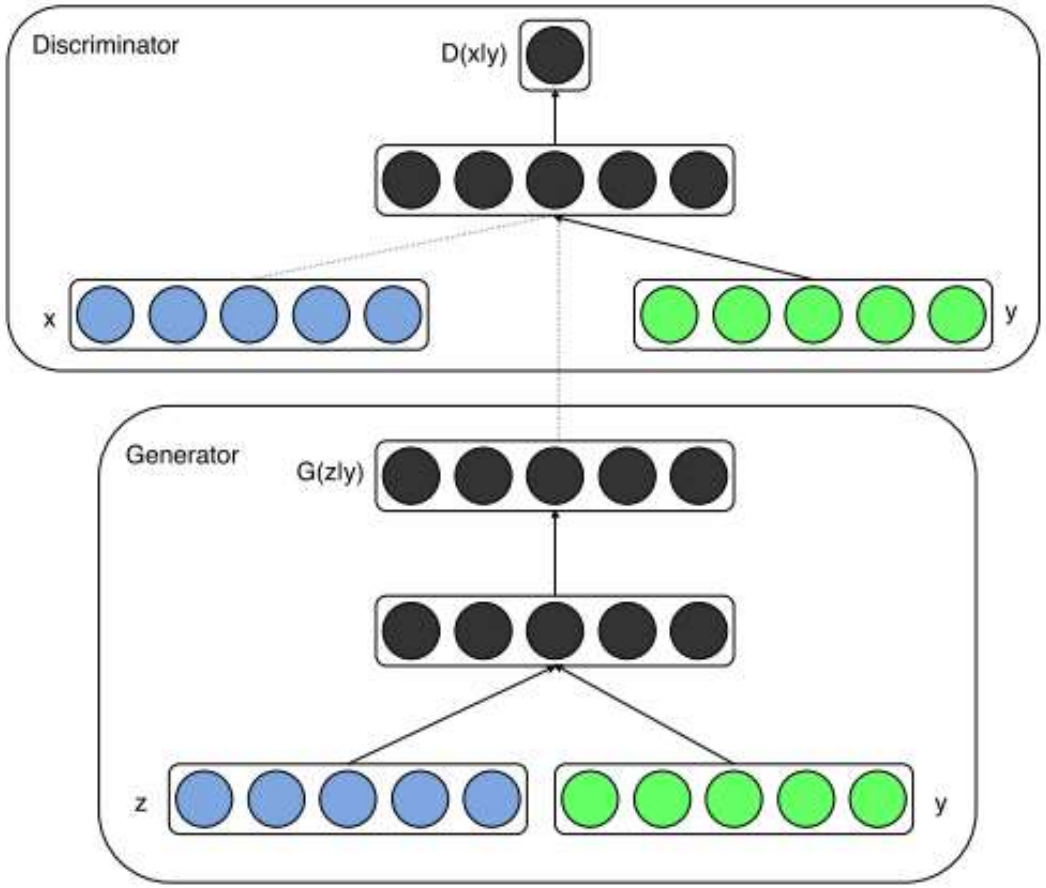


Conditional Gan (CGAN)

- 기존 GAN에 조건부 데이터를 입력하여 성능을 향상시킨 모델
  - 이전의 GAN은 생성되는 데이터를 조절할 수 없다는 단점이 있었음
  - Conditional Gan에서는 y라는 추가정보를 함께 넣어 조건부 모델로 확장 시켜준다.

Model Architecture

- 기존의 generator에 noise와 discriminator에 img만 입력해주는 GAN과 다르게 label을 추가해줌



Experiment