2020.03.18

PyTorch GANs:

Youngtaek Hong, PhD

Objective function

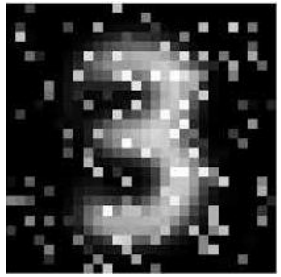
$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

$\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})]$

x

1

$\mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$



$G(z)$

0

Code review

```
import torch
import torch.nn. as nn

D = nn.Sequential(
        nn.Linear(784 ,128),
        nn.ReLU(),
        nn.Linear(128, 1),
        nn.Sigmoid())

G = nn.Sequential(
        nn.Linear(100, 128),
        nn.ReLU(),
        nn.Linear(128, 784),
        nn.Tanh()) # 생성된 값이 -1 ~ 1
```

Code review

```
criterion = nn.BCELoss() # Binary Cross Entropy Loss(h(x), y), Si

d_optimizer = torch.optim.Adam(D.parameters(), lr=0.01)
g_optimizer = torch.optim.Adam(G.parameters(), lr=0.01)
# 충돌하기에 2개의 optimizer를 설정

while True:
        # train D
        loss = criterion(D(x), 1) + criterion(D(G(z)), 0)
        loss.backward() # 모든 weight에 대해 gradient값을 계산
        d_optimizer.step()

        # train G
        loss = criterion(D(G(z)), 1)
        loss.backward()
        g_optimizer.step() # generator의 파라미터를 학습
```

Binary Cross Entropy:

$$BCE(x) = -\frac{1}{N}\sum_{i=1}^{N} y_i \log\big(h(x_i;\theta)\big) + (1-y_i)\log\big(1-h(x_i;\theta)\big).$$

```
loss = criterion(D(x), 1) + criterion(D(G(z)), 0)
```

Binary Cross Entropy (h(x), y) $-ylog\big(h(x)\big) - (1-y)\log(1-h(x))$

`criterion(D(x), 1)` $-log\big(D(x)\big)$

`criterion(D(G(z)), 0)` $-log(1-D(G(z))$

$$-[log\big(D(x)\big) + lo\ g\big(1-D(G(z)\big)]$$

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1-D(G(\boldsymbol{z})))]$$

Code review

```
criterion = nn.BCELoss() # Binary Cross Entropy Loss(h(x), y), Si

d_optimizer = torch.optim.Adam(D.parameters(), lr=0.01)
g_optimizer = torch.optim.Adam(G.parameters(), lr=0.01)
# 충돌하기에 2개의 optimizer를 설정

while True:
        # train D
        loss = criterion(D(x), 1) + criterion(D(G(z)), 0)
        loss.backward() # 모든 weight에 대해 gradient값을 계산
        d_optimizer.step()  # gradient updated by optimizer

        # train G
        loss = criterion(D(G(z)), 1)
        loss.backward()
        g_optimizer.step() # generator의 파라미터를 학습
```

Code review

```
# train G
loss = criterion(D(G(z)), 1)
loss.backward()
g_optimizer.step() # generator의 파라미터를 학습
```

Binary Cross Entropy (h(x), y) $-ylog\big(h(x)\big) - (1-y)\log(1-h(x))$

`criterion(D(G(z)), 1)` $-log(D(G(z)))$