Problem3

20191128 Jian Park

In [1]:

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

In [2]:

```
class DoorLock(nn.Module):
  def __init__(self):
    super().__init__()
    self.f = nn.Sequential(
        nn.Linear(100, 1),
        nn.Sigmoid()
    )
    for p in self.f.parameters():
      p.requires_grad = False

  def forward(self, x):
    y = self.f(x)
    if(y > 0.9):
      print('Opened!')
    return y

class DoorHack(nn.Module):
  def __init__(self, locker):
    super().__init__()
    self.g = nn.Sequential(
        nn.Linear(100, 100),
    )
    self.locker = locker

  def forward(self, z):
    y = self.locker(self.g(z))
    return y
```

In [3]:

```
num_trials = 50 # we optimaize 50 trials
learning_rate = 0.1
locker = DoorLock()
hacker = DoorHack(locker)
```

In [4]:

```
import numpy as np
z_set = torch.rand(100) # randum value for train
print(z_set)
```

```
tensor([0.1849, 0.0155, 0.2598, 0.4631, 0.6136, 0.0316, 0.9940, 0.5674, 0.5168,
        0.2586, 0.1798, 0.6873, 0.4229, 0.2203, 0.9068, 0.9419, 0.4312, 0.6140,
        0.6433, 0.4919, 0.8637, 0.2005, 0.2231, 0.2714, 0.1817, 0.0746, 0.3486,
        0.5800, 0.9966, 0.7219, 0.7009, 0.6515, 0.7491, 0.7076, 0.6409, 0.3726,
        0.2764, 0.4258, 0.5679, 0.2294, 0.3987, 0.1648, 0.6334, 0.1657, 0.7631,
        0.1219, 0.3511, 0.3317, 0.4526, 0.0187, 0.4938, 0.7053, 0.8781, 0.3218,
        0.5012, 0.6477, 0.4078, 0.9835, 0.8517, 0.9423, 0.4030, 0.3403, 0.1234,
        0.6322, 0.3136, 0.8275, 0.7039, 0.2016, 0.6047, 0.7514, 0.0438, 0.3994,
        0.3078, 0.2686, 0.0955, 0.1693, 0.9494, 0.6799, 0.9937, 0.1803, 0.9707,
        0.1986, 0.6117, 0.6396, 0.1543, 0.2847, 0.9968, 0.4339, 0.7773, 0.2123,
        0.2468, 0.8082, 0.0682, 0.3441, 0.5562, 0.5121, 0.6385, 0.0946, 0.1995,
        0.7381])
```

In [5]:

In [5]:

```
loss_func = torch.nn.MSELoss() # use MSE loss
optimizer = torch.optim.Adam([z_set], learning_rate) # use Adam optimizer, and we update
z values (input)
losses = []
```

In [6]:

```
z_set.requires_grad = True # for update z values

for i in range(num_trials):
  optimizer.zero_grad()
  output = hacker(z_set)
  print(output)
  loss = loss_func(output.to(torch.float32), torch.tensor([1]).to(torch.float32)) # calc
ulate loss func with ture value(1 is near value with 0.9)
  loss.backward()
  optimizer.step()
  losses.append(loss.item())
print(losses)
```

```
tensor([0.5353], grad_fn=<SigmoidBackward0>)
tensor([0.5963], grad_fn=<SigmoidBackward0>)
tensor([0.6540], grad_fn=<SigmoidBackward0>)
tensor([0.7066], grad_fn=<SigmoidBackward0>)
tensor([0.7530], grad_fn=<SigmoidBackward0>)
tensor([0.7927], grad_fn=<SigmoidBackward0>)
tensor([0.8259], grad_fn=<SigmoidBackward0>)
tensor([0.8532], grad_fn=<SigmoidBackward0>)
tensor([0.8754], grad_fn=<SigmoidBackward0>)
tensor([0.8934], grad_fn=<SigmoidBackward0>)
Opened!
tensor([0.9081], grad_fn=<SigmoidBackward0>)
Opened!
tensor([0.9200], grad_fn=<SigmoidBackward0>)
Opened!
tensor([0.9297], grad_fn=<SigmoidBackward0>)
Opened!
tensor([0.9377], grad_fn=<SigmoidBackward0>)
Opened!
tensor([0.9443], grad_fn=<SigmoidBackward0>)
Opened!
tensor([0.9497], grad_fn=<SigmoidBackward0>)
Opened!
tensor([0.9543], grad_fn=<SigmoidBackward0>)
Opened!
tensor([0.9582], grad_fn=<SigmoidBackward0>)
Opened!
tensor([0.9615], grad_fn=<SigmoidBackward0>)
Opened!
tensor([0.9643], grad_fn=<SigmoidBackward0>)
Opened!
tensor([0.9667], grad_fn=<SigmoidBackward0>)
Opened!
tensor([0.9687], grad_fn=<SigmoidBackward0>)
Opened!
tensor([0.9705], grad_fn=<SigmoidBackward0>)
Opened!
tensor([0.9721], grad_fn=<SigmoidBackward0>)
Opened!
tensor([0.9735], grad_fn=<SigmoidBackward0>)
Opened!
tensor([0.9747], grad_fn=<SigmoidBackward0>)
Opened!
tensor([0.9757], grad_fn=<SigmoidBackward0>)
Opened!
tensor([0.9766], grad_fn=<SigmoidBackward0>)
Opened!
tensor([0.9775], grad_fn=<SigmoidBackward0>)
Opened!
tensor([0.9782], grad_fn=<SigmoidBackward0>)
Opened!
```

```
tensor([0.9789], grad_fn=<SigmoidBackward0>)
Opened!
tensor([0.9795], grad_fn=<SigmoidBackward0>)
Opened!
tensor([0.9800], grad_fn=<SigmoidBackward0>)
Opened!
tensor([0.9805], grad_fn=<SigmoidBackward0>)
Opened!
tensor([0.9809], grad_fn=<SigmoidBackward0>)
Opened!
tensor([0.9813], grad_fn=<SigmoidBackward0>)
Opened!
tensor([0.9816], grad_fn=<SigmoidBackward0>)
Opened!
tensor([0.9820], grad_fn=<SigmoidBackward0>)
Opened!
tensor([0.9823], grad_fn=<SigmoidBackward0>)
Opened!
tensor([0.9825], grad_fn=<SigmoidBackward0>)
Opened!
tensor([0.9828], grad_fn=<SigmoidBackward0>)
Opened!
tensor([0.9830], grad_fn=<SigmoidBackward0>)
Opened!
tensor([0.9832], grad_fn=<SigmoidBackward0>)
Opened!
tensor([0.9834], grad_fn=<SigmoidBackward0>)
Opened!
tensor([0.9835], grad_fn=<SigmoidBackward0>)
Opened!
tensor([0.9837], grad_fn=<SigmoidBackward0>)
Opened!
tensor([0.9838], grad_fn=<SigmoidBackward0>)
Opened!
tensor([0.9840], grad_fn=<SigmoidBackward0>)
Opened!
tensor([0.9841], grad_fn=<SigmoidBackward0>)
Opened!
tensor([0.9842], grad_fn=<SigmoidBackward0>)
[0.21596169471740723, 0.16298003494739532, 0.1197163388133049, 0.08606893569231033, 0.061
008989810943604, 0.042981695383787155, 0.030325893312692642, 0.021563975140452385, 0.0155
26346862316132, 0.011355145834386349, 0.008450527675449848, 0.006404699757695198, 0.00494
4032058119774, 0.003885720856487751, 0.003107299562543630 6, 0.00252608023583889, 0.002085
7355557382107, 0.0017474113265052438, 0.0014839953510090709, 0.0012763222912326455, 0.001
1106639867648482, 0.0009770506294444203, 0.0008681908948346972, 0.0007786417263559997, 0.
0007043326622806 49, 0.000642155937384814, 0.0005897418595850468, 0.0005452391924336553, 0
.0005072070634923875, 0.0004745092301163822 4, 0.0004462387005332857 4, 0.00042166386265307
665, 0.0004001971392426 64, 0.0003813579678535461 4, 0.000364760315278545, 0.00035007143742
40488, 0.0003370246267877519, 0.0003253943286836147 3, 0.0003149914555251598 4, 0.000305653
05496566, 0.0002972480142489075 7, 0.000289655610686168 1, 0.0002827845455612987 3, 0.000276
5388635452837, 0.0002708512183744460 3, 0.000265655748080462 2, 0.000260897504631429 9, 0.00
0256531551713123 9, 0.0002525092277210206, 0.0002487949968781322]
```