

## 情報工学演習 II 岡田先生の課題

### 1 XOR

Listing 1 に作成したソースコードを示す。また図 1, 図 2, 図 3, 図 4, 図 5, 図 6 に学習率を 0.001, 0.01, 0.1, 1, 10, 100 にしたときの loss の変化及び勾配の最大値を示す。

Listing 1: XOR

```

1 import numpy as np
2 import matplotlib.pyplot as plt
3 import torch
4 import torch.nn as nn
5
6 class MLP(nn.Module):
7     def __init__(self, input_size,
8                   hidden_size, output_size):
9         super(MLP, self).__init__()
10
11        self.l1 = nn.Linear(input_size,
12                             hidden_size)
13        self.l2 = nn.Linear(hidden_size,
14                             output_size)
15
16    def forward(self, x):
17        h = torch.sigmoid(self.l1(x))
18        o = self.l2(h)
19        return o
20
21 x = [
22     [0., 0.],
23     [0., 1.],
24     [1., 0.],
25     [1., 1.]
26 ]
27
28 x = torch.tensor(x)
29 y = [
30     [0.],
31     [1.],
32     [1.],
33     [0.]
34 ]
35 y = torch.tensor(y)
36
37 Ir_all = [0.001, 0.01, 0.1, 1, 10, 100]
38 max_epoch = 30000
39
40 optimizer = torch.optim.SGD(model.
41                               parameters(), lr=Ir)
42 criterion = nn.MSELoss()
43 params = list(model.parameters())
44
45 history = []
46 history_l1 = []
47 history_l2 = []
48
49 for epoch in range(max_epoch):
50     pred = model(x)
51     error = criterion(y, pred)
52     history.append(error.item())
53     model.zero_grad()
54     error.backward()
55     optimizer.step()
56
57     history_l1.append(torch.max(params[0].
58                                grad).data)
59     history_l2.append(torch.max(params[2].
60                                grad).data)
61
62 history = np.array(history, dtype=np.
63                    float32)
64 history_l1 = np.array(history_l1, dtype=
65                       np.float32)
66 history_l2 = np.array(history_l2, dtype=
67                       np.float32)
68 epochs = np.arange(1, max_epoch+1)
69
70 fig, axes = plt.subplots(nrows=1, ncols
71                          =3, figsize=(12, 4))
72 axes[0].plot(epochs, history)
73 axes[0].set_title('Ir=%f, loss' % (Ir
74 ))
75 axes[0].set_xlabel("epochs")
76 axes[0].set_ylabel("loss")
77 axes[1].plot(epochs, history_l1)
78 axes[1].set_title('max_grad(l1)')
79 axes[1].set_xlabel("epochs")
80 axes[1].set_ylabel("max_grad")
81 axes[2].plot(epochs, history_l2)
82 axes[2].set_title('max_grad(l2)')
83 axes[2].set_xlabel("epochs")
84 axes[2].set_ylabel("max_grad")
85 plt.show()

```

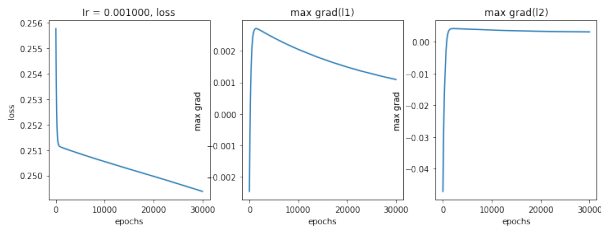


図 1: 学習率 0.001 のとき

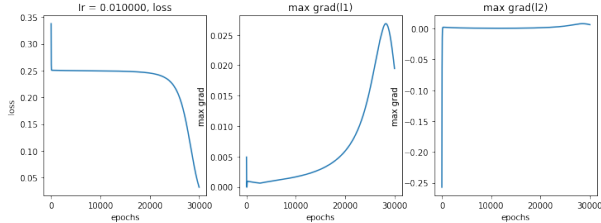


図 2: 学習率 0.01 のとき

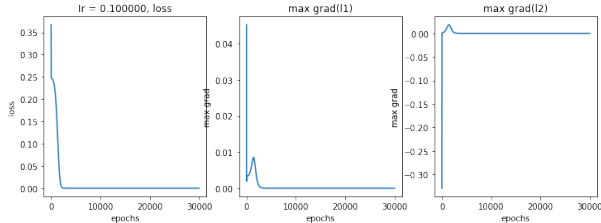


図 3: 学習率 0.1 のとき

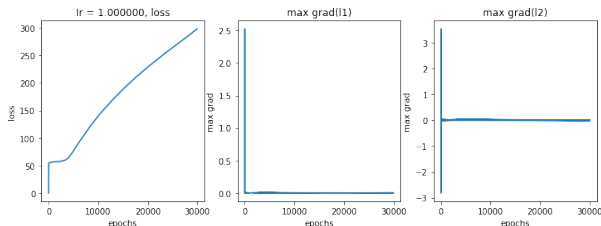


図 4: 学習率 1 のとき

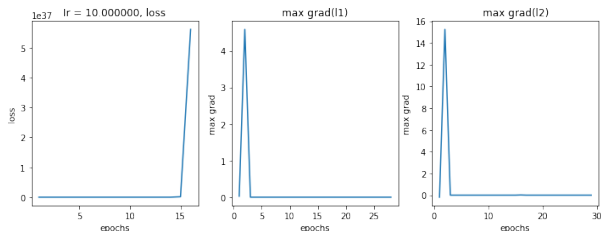


図 5: 学習率 10 のとき

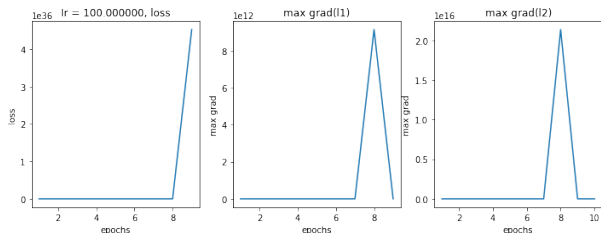


図 6: 学習率 100 のとき

## 2 パリティビット

Listing 2 に作成したソースコードを示す．また図 7 に loss の変化を示す．

Listing 2: パリティビット

```
1 import numpy as np
2 import matplotlib.pyplot as plt
3 import torch
4 import torch.nn as nn
5
6 class MLP(nn.Module):
7     def __init__(self, input_size,
8                   hidden_size, output_size):
9         super(MLP, self).__init__()
10
11         self.l1 = nn.Linear(input_size,
12                               hidden_size)
13         self.l2 = nn.Linear(hidden_size,
14                               output_size)
15
16     def forward(self, x):
17         h = torch.sigmoid(self.l1(x))
18         o = self.l2(h)
19         return o
20
21 x = torch.tensor([[0, 0, 0], [0, 0, 1],
22                   [0, 1, 0], [0, 1, 1], [1, 0, 0], [1,
23                   0, 1], [1, 1, 0], [1, 1, 1]], dtype
24                   =torch.float)
25 y = torch.tensor([0, 1, 1, 0, 1, 0, 0,
26                   1])
27
28 model = MLP(3, 3, 2)
29 optimizer = torch.optim.SGD(model.
30                               parameters(), lr=1.0e-1)
31 criterion = nn.CrossEntropyLoss()
32
33 history = []
34 max_epoch = 100000
35
36 for epoch in range(max_epoch):
37     pred = model(x)
38     error = criterion(pred, y)
39     history.append(error.item())
40     model.zero_grad()
41     error.backward()
42     optimizer.step()
43
44 history = np.array(history, dtype=np.
45                     float32)
46 epochs = np.arange(1, max_epoch+1)
47
48 plt.plot(epochs, history)
```

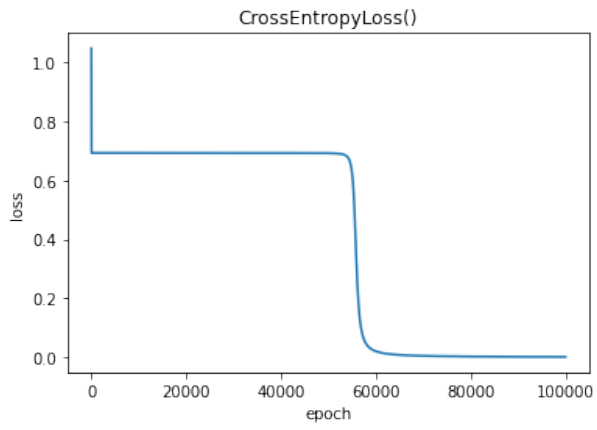


図 7: 交差エントロピーを誤差関数としたときの loss の推移

```
40 plt.title("CrossEntropyLoss()")
41 plt.xlabel("epoch")
42 plt.ylabel("loss")
43 plt.show()
```

---