代码：

import torch

import random

X = torch.tensor([[0.0,0.0],[0.0,1.0],[1.0,0.0],[1.0,1.0]]) #输入

o = torch.tensor([0.0,1.0,1.0,0.0]) #输出层的输出期望

w11 = torch.tensor(0.11,requires\_grad=True)

w12 = torch.tensor(0.12,requires\_grad=True)

w13 = torch.tensor(0.13,requires\_grad=True)

w14 = torch.tensor(0.14,requires\_grad=True)

w21 = torch.tensor(0.21,requires\_grad=True)

w22 = torch.tensor(0.22,requires\_grad=True)

w23 = torch.tensor(0.23,requires\_grad=True)

w24 = torch.tensor(0.24,requires\_grad=True)

v11 = torch.tensor(0.3,requires\_grad=True)

v21 = torch.tensor(0.4,requires\_grad=True)

v31 = torch.tensor(0.5,requires\_grad=True)

v41 = torch.tensor(0.6,requires\_grad=True)

for k in range(1000000): #循环100万次

i = random.randint(0,3)

lr = 0.02 #学习率定为0.02

x1 = X[i][0]

x2 = X[i][1]

y1 = torch.sigmoid(w11\*x1+w21\*x2) #四个中间节点

y2 = torch.sigmoid(w12\*x1+w22\*x2)

y3 = torch.sigmoid(w13\*x1+w23\*x2)

y4 = torch.sigmoid(w14\*x1+w24\*x2)

z = torch.sigmoid(v11\*y1+v21\*y2+v31\*y3+v41\*y4) #异或输出

error = (o[i]-z)\*(o[i]-z) #误差

if k%1000==0 or k>1000000-10:

print("i=",i,"error=",error)

error.backward()

w11.data = w11.data-lr\*w11.grad

w12.data = w12.data-lr\*w12.grad

w13.data = w13.data-lr\*w13.grad

w14.data = w14.data-lr\*w14.grad

w21.data = w21.data-lr\*w21.grad

w22.data = w22.data-lr\*w22.grad

w23.data = w23.data-lr\*w23.grad

w24.data = w24.data-lr\*w24.grad

v11.data = v11.data-lr\*v11.grad

v21.data = v21.data-lr\*v21.grad

v31.data = v31.data-lr\*v31.grad

v41.data = v41.data-lr\*v41.grad

#梯度清零

w11.grad.zero\_()

w12.grad.zero\_()

w13.grad.zero\_()

w14.grad.zero\_()

w21.grad.zero\_()

w22.grad.zero\_()

w23.grad.zero\_()

w24.grad.zero\_()

v11.grad.zero\_()

v21.grad.zero\_()

v31.grad.zero\_()

v41.grad.zero\_()

'''if k%1000==0 or k>1000000-10:

print("i=",i,"error=",error)'''

一百万次循环结果：

