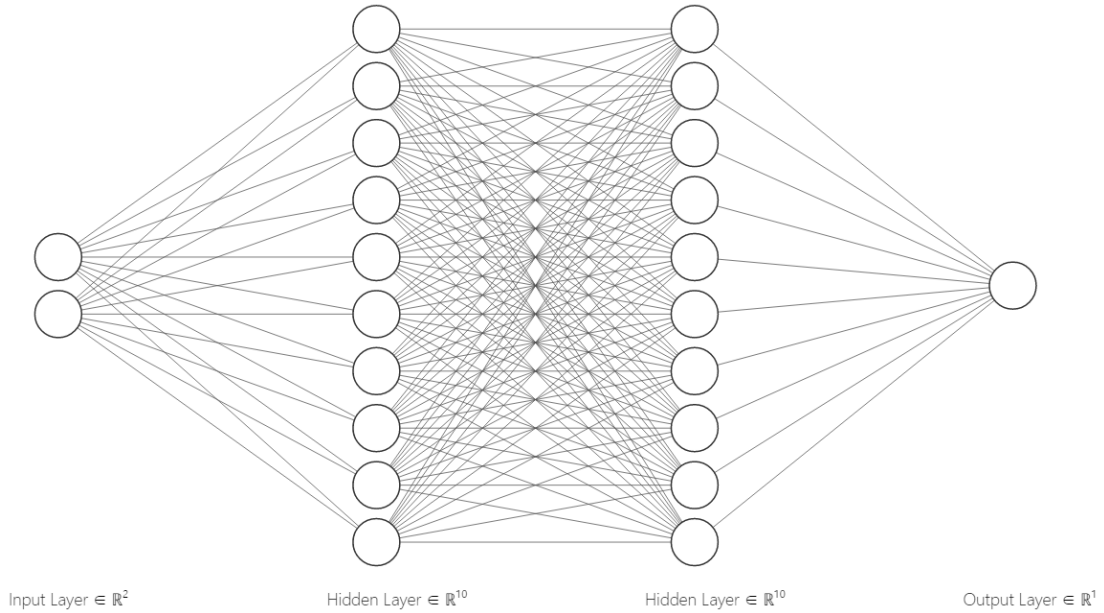
The structured fully connected neural network shows below, input layer contains 2 nodes, the first hidden layer contains 10 nodes, the second hidden layer contains 10 nodes, and the last output layer contains one node.



In this case the shape of b1, w1 between input and first hidden layer are (1,10), (10,2), and shape of b2, w2 between first and second hidden layer are (1,10),(10,10), respectively. And shape of b3, w3 between second and output hidden layer are (1,1) and (1,10), respectively.

Forward propagation:

$z_1 = xw_1^T + b_1$

$y_1 = sigmoid(z_1)$

$z_2 = y_1 w_2^T + b_2$

$y_2 = sigmoid(z_2)$

$z_3 = y_2 w_3^T + b_3$

$y_3 = z_3$

Loss Function:

$loss = (y_3 - y)^2$

Backward Propagation:

$$\frac{\partial loss}{\partial w_3} = \frac{\partial loss}{\partial y_3}\frac{\partial y_3}{\partial w_3} = 2(y_3 - y) \cdot y_2$$

$$\frac{\partial loss}{\partial b_3} = \frac{\partial loss}{\partial y_3}\frac{\partial y_3}{\partial b_3} = 2(y_3 - y)$$

$$\frac{\partial loss}{\partial w_2} = \frac{\partial loss}{\partial y_3}\frac{\partial y_3}{\partial y_2}\frac{\partial y_2}{\partial z_2}\frac{\partial z_2}{\partial w_2} = 2(y_3 - y) \cdot w_3 \cdot sigmoid(z_2)(1 - sigmoid(z_2)) \cdot y_1$$

$$\frac{\partial loss}{\partial b_2} = \frac{\partial loss}{\partial y_3}\frac{\partial y_3}{\partial y_2}\frac{\partial y_2}{\partial z_2}\frac{\partial z_2}{\partial b_2} = 2(y_3 - y) \cdot w_3 \cdot sigmoid(z_2)(1 - sigmoid(z_2))$$

$$\frac{\partial loss}{\partial w_1} = \frac{\partial loss}{\partial y_3}\frac{\partial y_3}{\partial y_2}\frac{\partial y_2}{\partial z_2}\frac{\partial z_2}{\partial y_1}\frac{\partial y_1}{\partial z_1}\frac{\partial z_1}{\partial w_1}$$

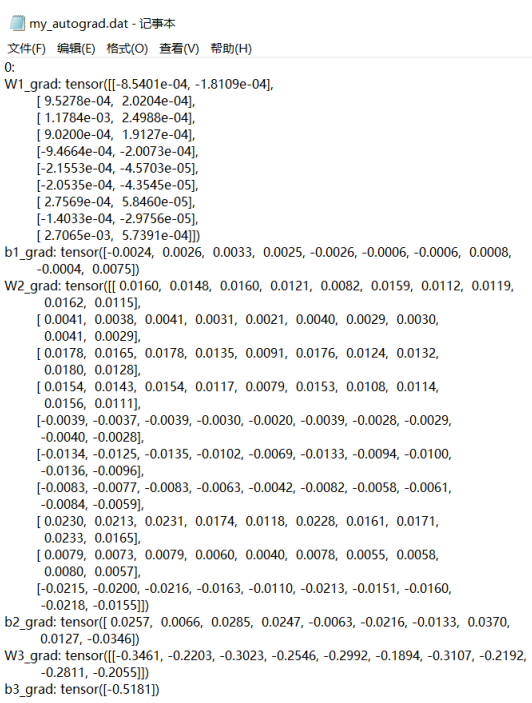
$$= 2(y_3 - y) \cdot w_3 \cdot sigmoid(z_2)(1 - sigmoid(z_2)) \cdot w_2 \cdot sigmoid(z_1)(1 - sigmoid(z_1)) \cdot x$$

$$\frac{\partial loss}{\partial b_1} = \frac{\partial loss}{\partial y_3}\frac{\partial y_3}{\partial y_2}\frac{\partial y_2}{\partial z_2}\frac{\partial z_2}{\partial y_1}\frac{\partial y_1}{\partial z_1}\frac{\partial z_1}{\partial b_1} =$$

$$= 2(y_3 - y) \cdot w_3 \cdot sigmoid(z_2)(1 - sigmoid(z_2)) \cdot w_2 \cdot sigmoid(z_1)(1 - sigmoid(z_1))$$

After this backward propagation, I use 100 samples and batch-size as 1 to generate the grad for w1,b1,w2,b2,w3 and b3. The file can be generated by torch.autograd() and by my self-implemented-backward-propagation code. Please see the all numbers in "my_autograd.dat" and "torch_autograd.dat" files.

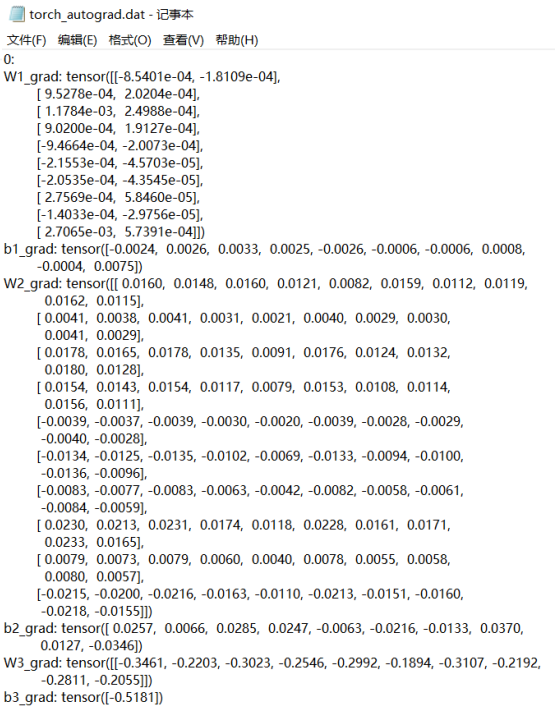
Herein is the first sample of both files, the number of them are totally same.



my_autograd.dat - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```
0:
W1_grad: tensor([[-8.5401e-04, -1.8109e-04],
        [ 9.5278e-04,  2.0204e-04],
        [ 1.1784e-03,  2.4988e-04],
        [ 9.0200e-04,  1.9127e-04],
        [-9.4664e-04, -2.0073e-04],
        [-2.1553e-04, -4.5703e-05],
        [-2.0535e-04, -4.3545e-05],
        [ 2.7569e-04,  5.8460e-05],
        [-1.4033e-04, -2.9756e-05],
        [ 2.7065e-03,  5.7391e-04]])
b1_grad: tensor([-0.0024,  0.0026,  0.0033,  0.0025, -0.0026, -0.0006, -0.0006,  0.0008,
        -0.0004,  0.0075])
W2_grad: tensor([[ 0.0160,  0.0148,  0.0160,  0.0121,  0.0082,  0.0159,  0.0112,  0.0119,
          0.0162,  0.0115],
        [ 0.0041,  0.0038,  0.0041,  0.0031,  0.0021,  0.0040,  0.0029,  0.0030,
          0.0041,  0.0029],
        [ 0.0178,  0.0165,  0.0178,  0.0135,  0.0091,  0.0176,  0.0124,  0.0132,
          0.0180,  0.0128],
        [ 0.0154,  0.0143,  0.0154,  0.0117,  0.0079,  0.0153,  0.0108,  0.0114,
          0.0156,  0.0111],
        [-0.0039, -0.0037, -0.0039, -0.0030, -0.0020, -0.0039, -0.0028, -0.0029,
         -0.0040, -0.0028],
        [-0.0134, -0.0125, -0.0135, -0.0102, -0.0069, -0.0133, -0.0094, -0.0100,
         -0.0136, -0.0096],
        [-0.0083, -0.0077, -0.0083, -0.0063, -0.0042, -0.0082, -0.0058, -0.0061,
         -0.0084, -0.0059],
        [ 0.0230,  0.0213,  0.0231,  0.0174,  0.0118,  0.0228,  0.0161,  0.0171,
          0.0233,  0.0165],
        [ 0.0079,  0.0073,  0.0079,  0.0060,  0.0040,  0.0078,  0.0055,  0.0058,
          0.0080,  0.0057],
        [-0.0215, -0.0200, -0.0216, -0.0163, -0.0110, -0.0213, -0.0151, -0.0160,
         -0.0218, -0.0155]])
b2_grad: tensor([ 0.0257,  0.0066,  0.0285,  0.0247, -0.0063, -0.0216, -0.0133,  0.0370,
          0.0127, -0.0346])
W3_grad: tensor([[-0.3461, -0.2203, -0.3023, -0.2546, -0.2992, -0.1894, -0.3107, -0.2192,
         -0.2811, -0.2055]])
b3_grad: tensor([-0.5181])
```



torch_autograd.dat - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```
0:
W1_grad: tensor([[-8.5401e-04, -1.8109e-04],
        [ 9.5278e-04,  2.0204e-04],
        [ 1.1784e-03,  2.4988e-04],
        [ 9.0200e-04,  1.9127e-04],
        [-9.4664e-04, -2.0073e-04],
        [-2.1553e-04, -4.5703e-05],
        [-2.0535e-04, -4.3545e-05],
        [ 2.7569e-04,  5.8460e-05],
        [-1.4033e-04, -2.9756e-05],
        [ 2.7065e-03,  5.7391e-04]])
b1_grad: tensor([-0.0024,  0.0026,  0.0033,  0.0025, -0.0026, -0.0006, -0.0006,  0.0008,
        -0.0004,  0.0075])
W2_grad: tensor([[ 0.0160,  0.0148,  0.0160,  0.0121,  0.0082,  0.0159,  0.0112,  0.0119,
          0.0162,  0.0115],
        [ 0.0041,  0.0038,  0.0041,  0.0031,  0.0021,  0.0040,  0.0029,  0.0030,
          0.0041,  0.0029],
        [ 0.0178,  0.0165,  0.0178,  0.0135,  0.0091,  0.0176,  0.0124,  0.0132,
          0.0180,  0.0128],
        [ 0.0154,  0.0143,  0.0154,  0.0117,  0.0079,  0.0153,  0.0108,  0.0114,
          0.0156,  0.0111],
        [-0.0039, -0.0037, -0.0039, -0.0030, -0.0020, -0.0039, -0.0028, -0.0029,
         -0.0040, -0.0028],
        [-0.0134, -0.0125, -0.0135, -0.0102, -0.0069, -0.0133, -0.0094, -0.0100,
         -0.0136, -0.0096],
        [-0.0083, -0.0077, -0.0083, -0.0063, -0.0042, -0.0082, -0.0058, -0.0061,
         -0.0084, -0.0059],
        [ 0.0230,  0.0213,  0.0231,  0.0174,  0.0118,  0.0228,  0.0161,  0.0171,
          0.0233,  0.0165],
        [ 0.0079,  0.0073,  0.0079,  0.0060,  0.0040,  0.0078,  0.0055,  0.0058,
          0.0080,  0.0057],
        [-0.0215, -0.0200, -0.0216, -0.0163, -0.0110, -0.0213, -0.0151, -0.0160,
         -0.0218, -0.0155]])
b2_grad: tensor([ 0.0257,  0.0066,  0.0285,  0.0247, -0.0063, -0.0216, -0.0133,  0.0370,
          0.0127, -0.0346])
W3_grad: tensor([[-0.3461, -0.2203, -0.3023, -0.2546, -0.2992, -0.1894, -0.3107, -0.2192,
         -0.2811, -0.2055]])
b3_grad: tensor([-0.5181])
```