PyTorch

Machine Learning & Deep Learning
(Barcha uchun)

<09> Softmax

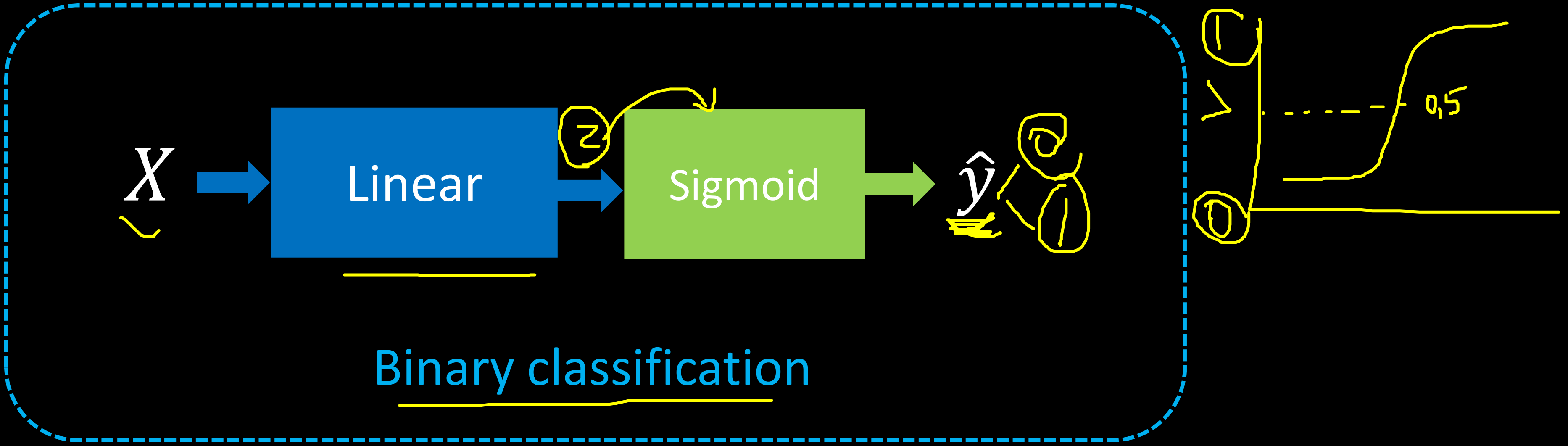
Mansurbek Abdullaev

https://uzbek.gitbook.io/ai/

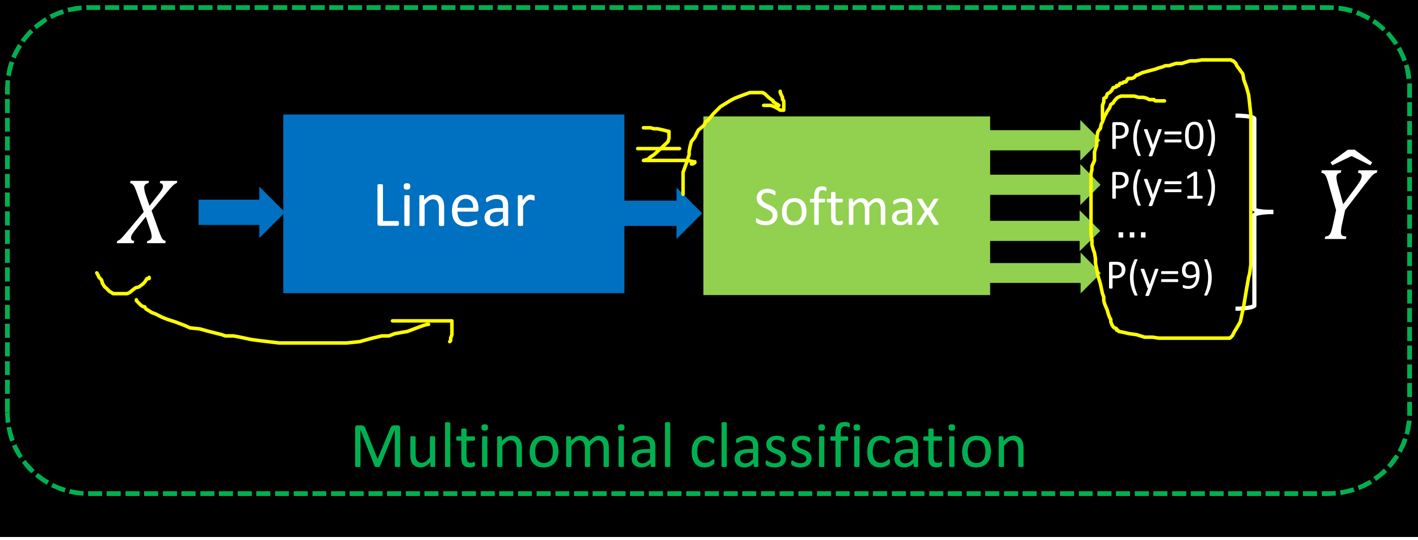
mansurbek.comchemai@gmail.com

@MansurbekUST

Ko’p sonli chiqish (output)



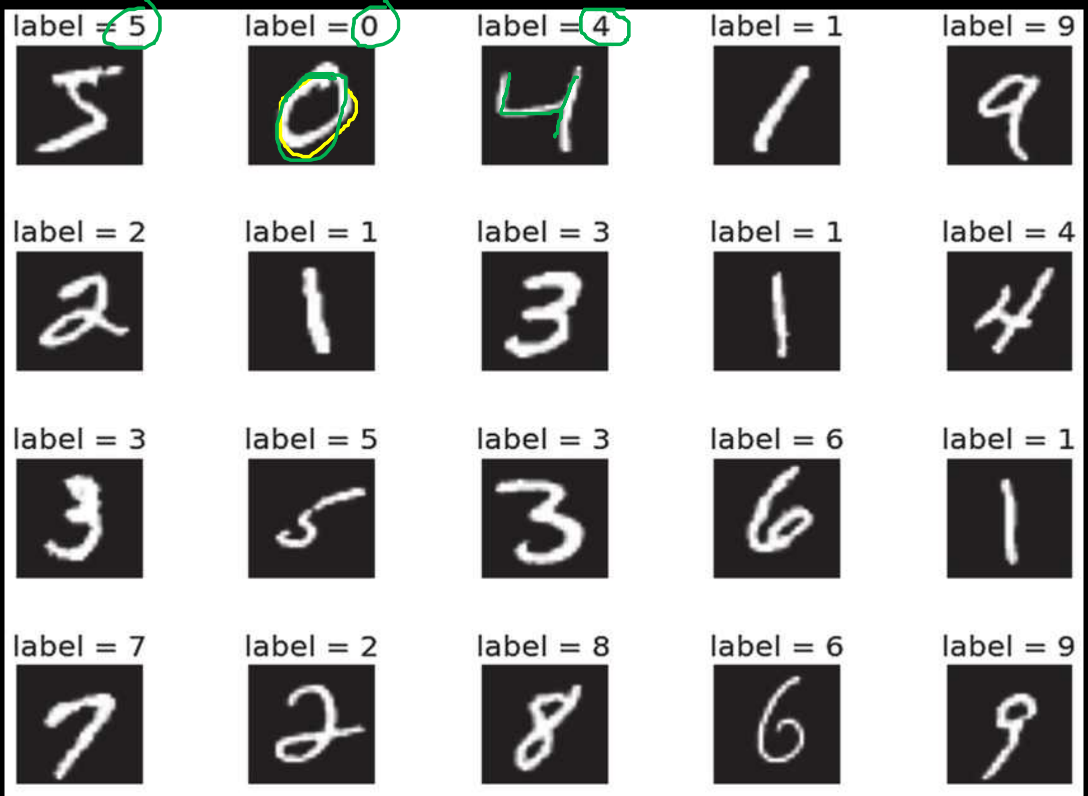
Binary classification



Multinomial classification

MNIST dataset : 10 labels

0 — 9



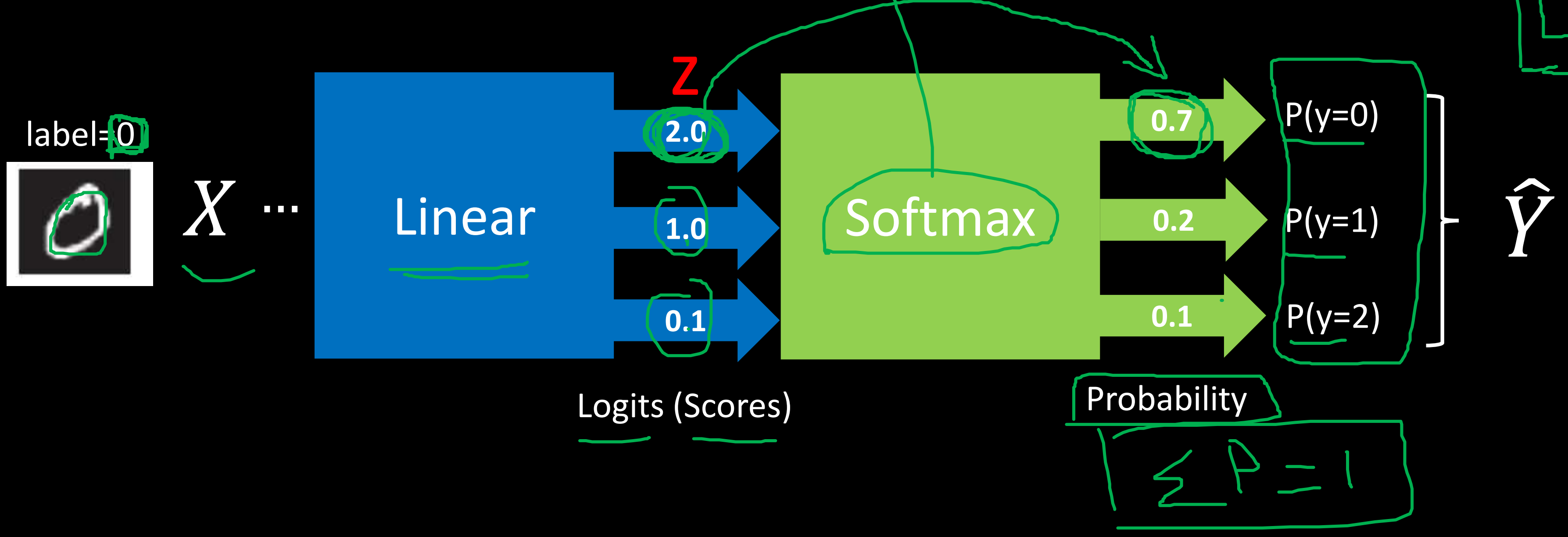
Softmax

$$\boxed{\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}}$$

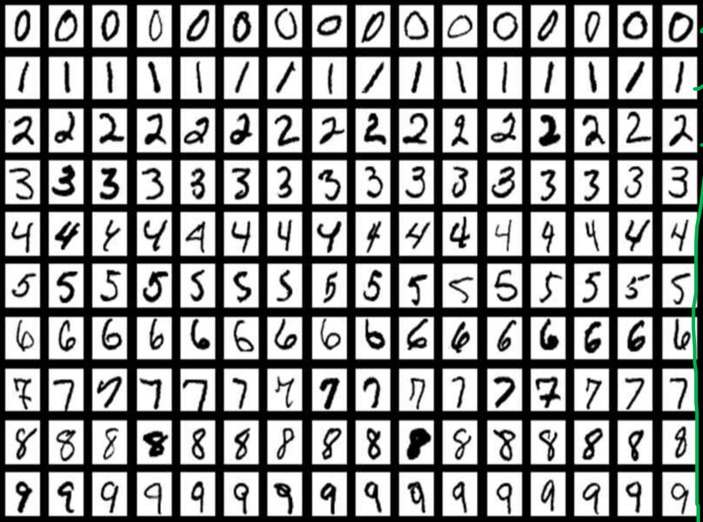
j =1, …., K uchun

$j \to 1$

$$\boxed{L(\hat{y}, y)}$$



One-hot encoding (vector)



0 1 2 3 4 5 6 - - 9

0 [1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0]

1 [0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0]

2 [0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0]

3 [0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0]

4 [0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0]

5 [0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0]

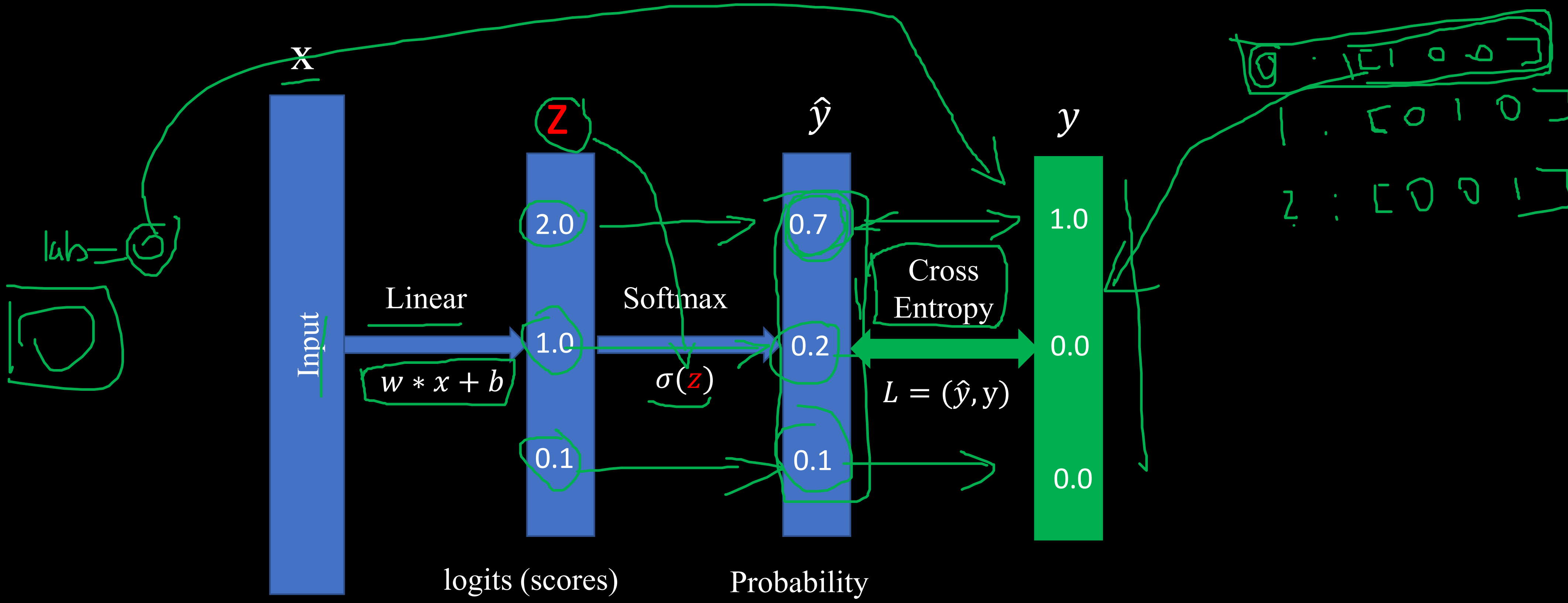
6 [0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0]

7 [0.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0]

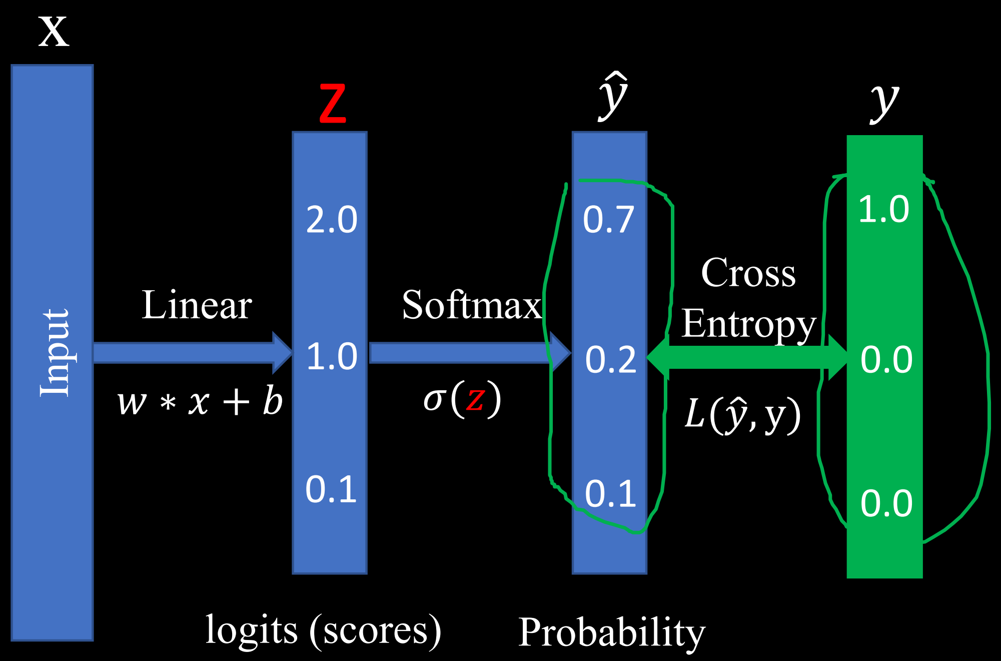
8 [0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0]

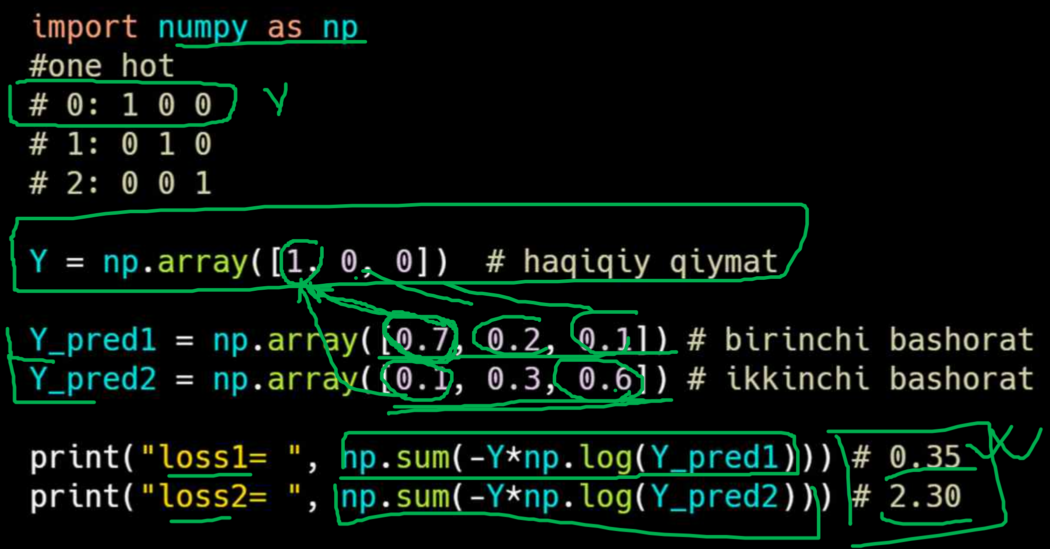
9 [0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0]

Forward pass



Loss: Cross Entropy



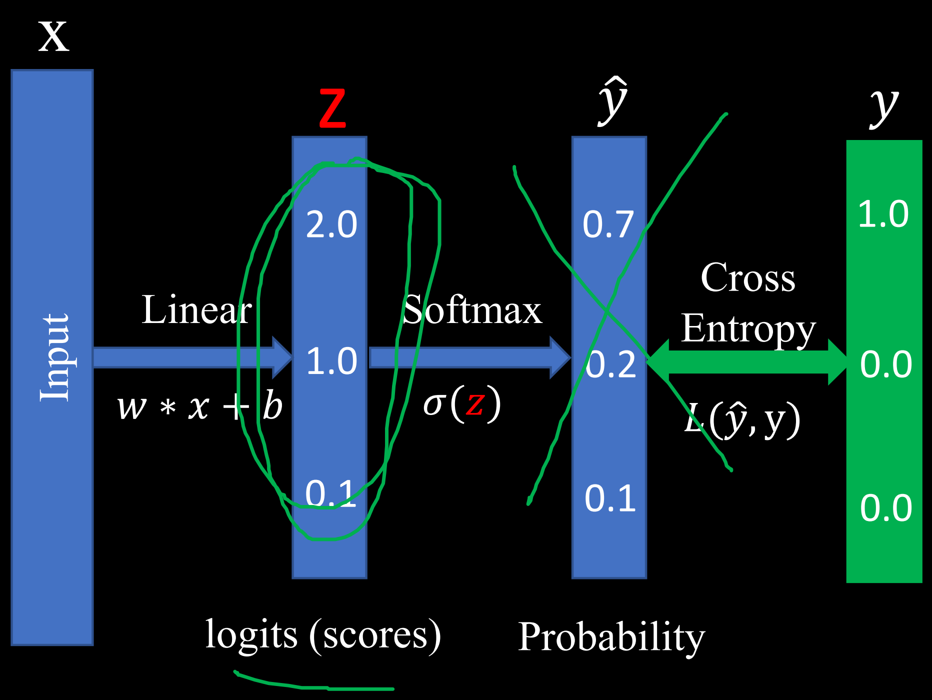


```
import numpy as np
#one hot
# 0: 1 0 0
# 1: 0 1 0
# 2: 0 0 1

Y = np.array([1, 0, 0])  # haqiqiy qiymat

Y_pred1 = np.array([0.7, 0.2, 0.1]) # birinchi bashorat
Y_pred2 = np.array([0.1, 0.3, 0.6]) # ikkinchi bashorat

print("loss1= ", np.sum(-Y*np.log(Y_pred1))) # 0.35
print("loss2= ", np.sum(-Y*np.log(Y_pred2))) # 2.30
```

Y

$$L(\hat{y}, y) = -y\log\hat{y}$$

Cross Entropy: PyTorch

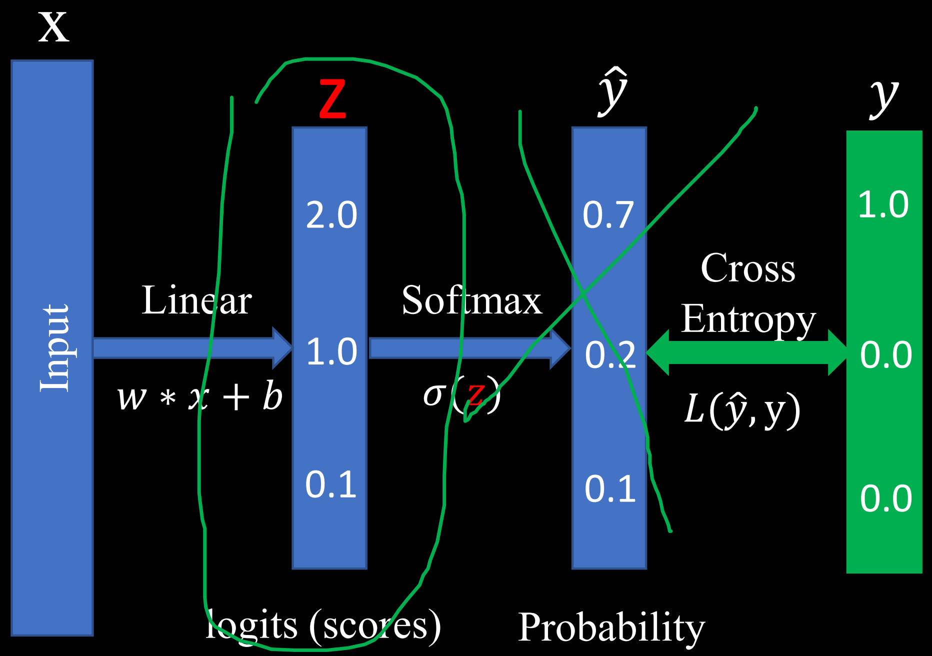


~~0 [1 0 0]~~
~~1: [0 1 0]~~
~~2: [0 0 1]]~~

```
# CrossEntropy
loss = nn.CrossEntropyLoss()

# Haqiqiy qiymat klass ko'rinishida one-hot enc emas
Y = tensor([0], requires_grad=False)

# Y_pred1 va Y_pred2 qiymatlari "logits", probability emas
Y_pred1 = tensor([[2.0, 1.0, 0.1]])
Y_pred2 = tensor([[0.5, 2.0, 0.3]])

l1 = loss(Y_pred1, Y)
l2 = loss(Y_pred2, Y)

print(f'PyTorch Loss1: {l1.item():.4f}') #0.41
print(f'PyTorch Loss2: {l2.item():.4f}') #1.84
```

$$L(\hat{y}, y) = -y \log \hat{y}$$

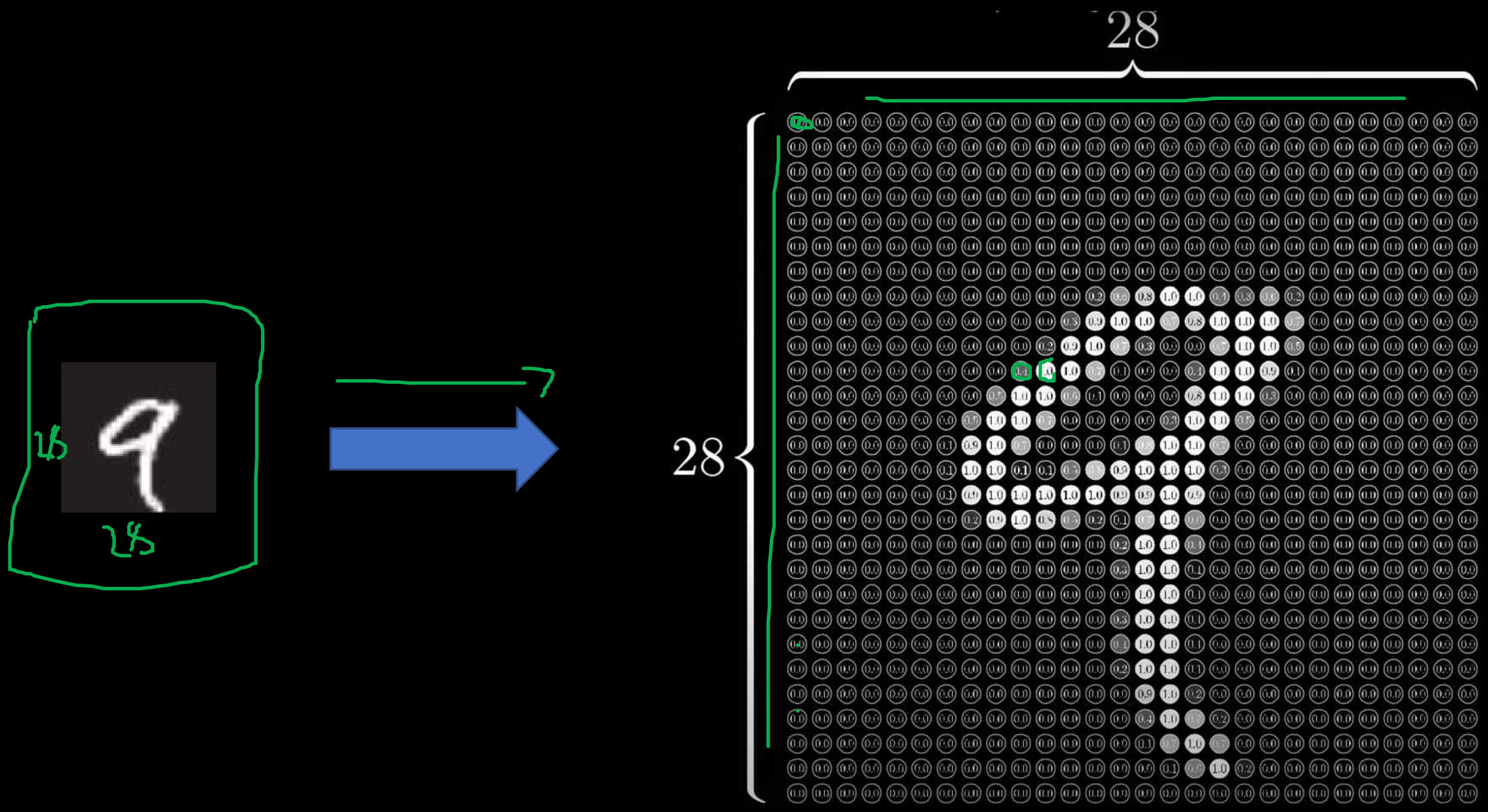
Cross Entropy: PyTorch (batch)



```
# CrossEntropy
loss = nn.CrossEntropyLoss()

# Haqiqiy qiymat klass ko'rinishida (bu yerda batch)
Y = tensor([2, 0, 1], requires_grad=False)

#Y_pred1 va Y_pred2 qiymatlari "logits", probability emas
Y_pred1 = tensor([[0.1, 0.2, 0.9],
                  [1.1, 0.1, 0.2],
                  [0.2, 2.1, 0.1]]) # batch uchun

Y_pred2 = tensor([[0.8, 0.2, 0.3],
                  [0.2, 0.3, 0.5],
                  [0.2, 0.2, 0.5]])  # batch uchun

l1 = loss(Y_pred1, Y)
l2 = loss(Y_pred2, Y)

print(f'Batch Loss1:  {l1.data:.4f}')
print(f'Batch Loss2:  {l2.data:.4f}')
```

$$L(\hat{y}, y) = -y \log \hat{y}$$

MNIST



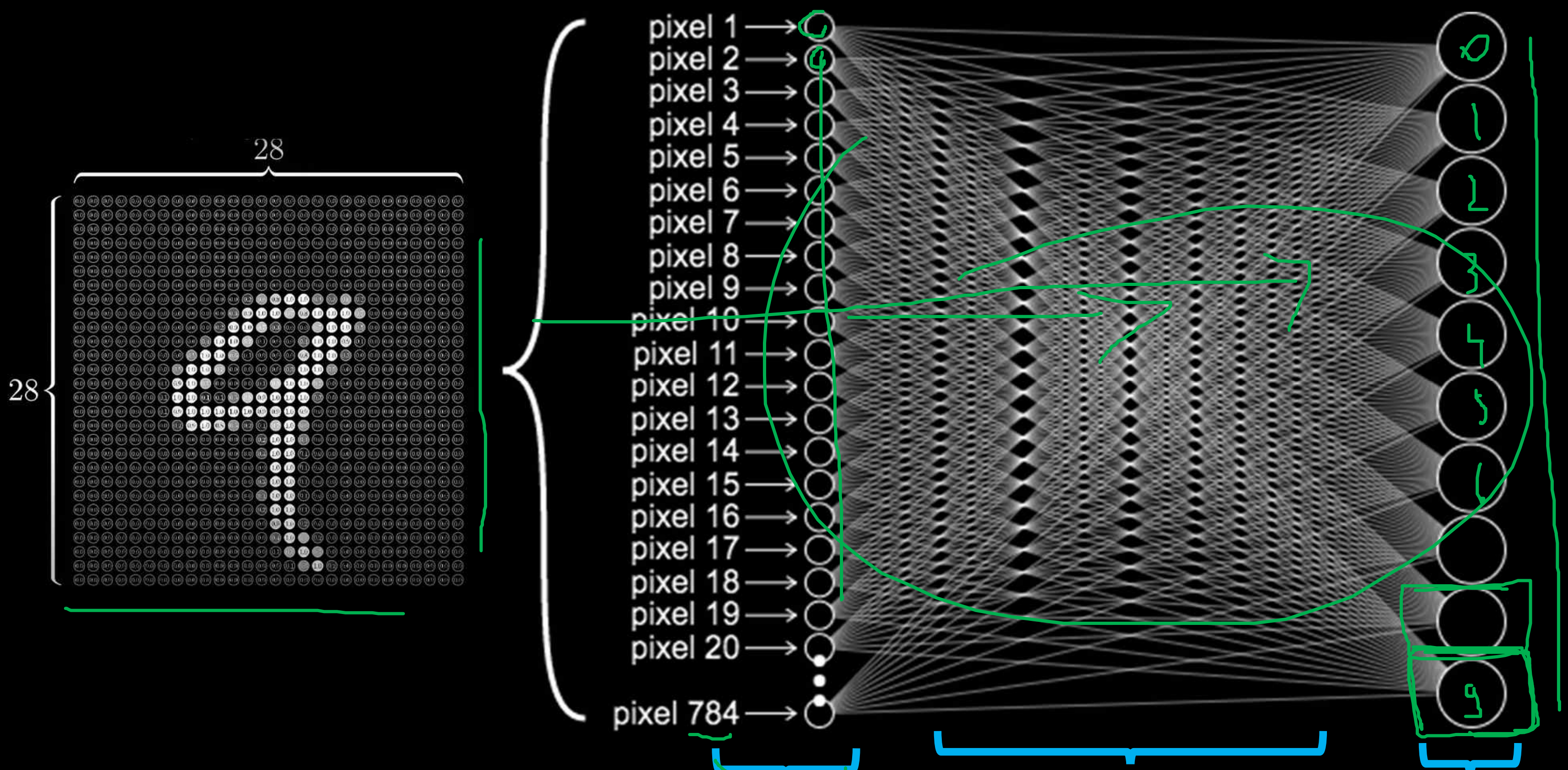
$1 \times 28 \times 28 = 784$

~~$3 \times 28 \times 28 = 784 \times 3$~~

rgb

MNIST Network

0 — 9

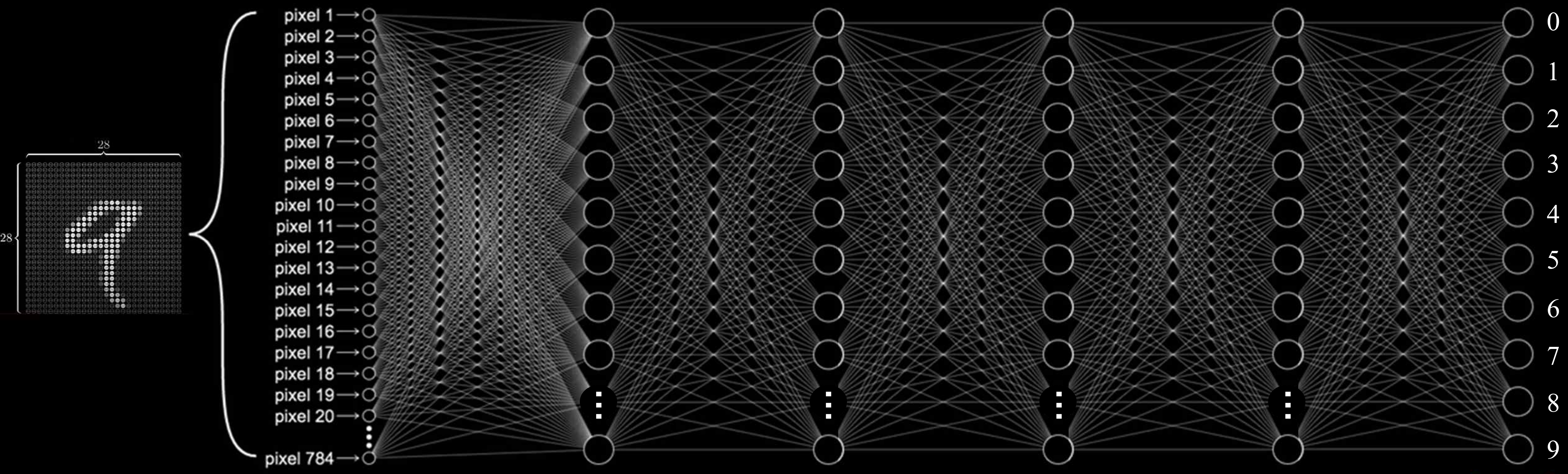


Input layer
784

Hidden layer

Output layer
10 (label/class)

MNIST Network



Input layer
784

Hidden 1:
520

Hidden 2:
320

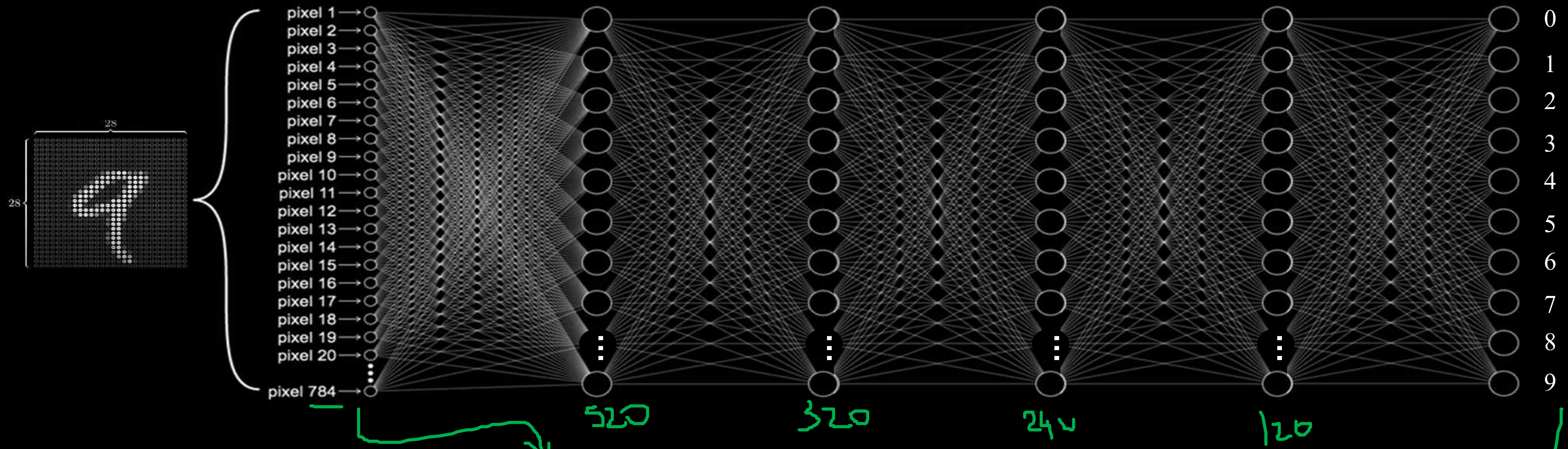
Hidden 3:
240

Hidden 4:
120

Output layer
10 (label/class)

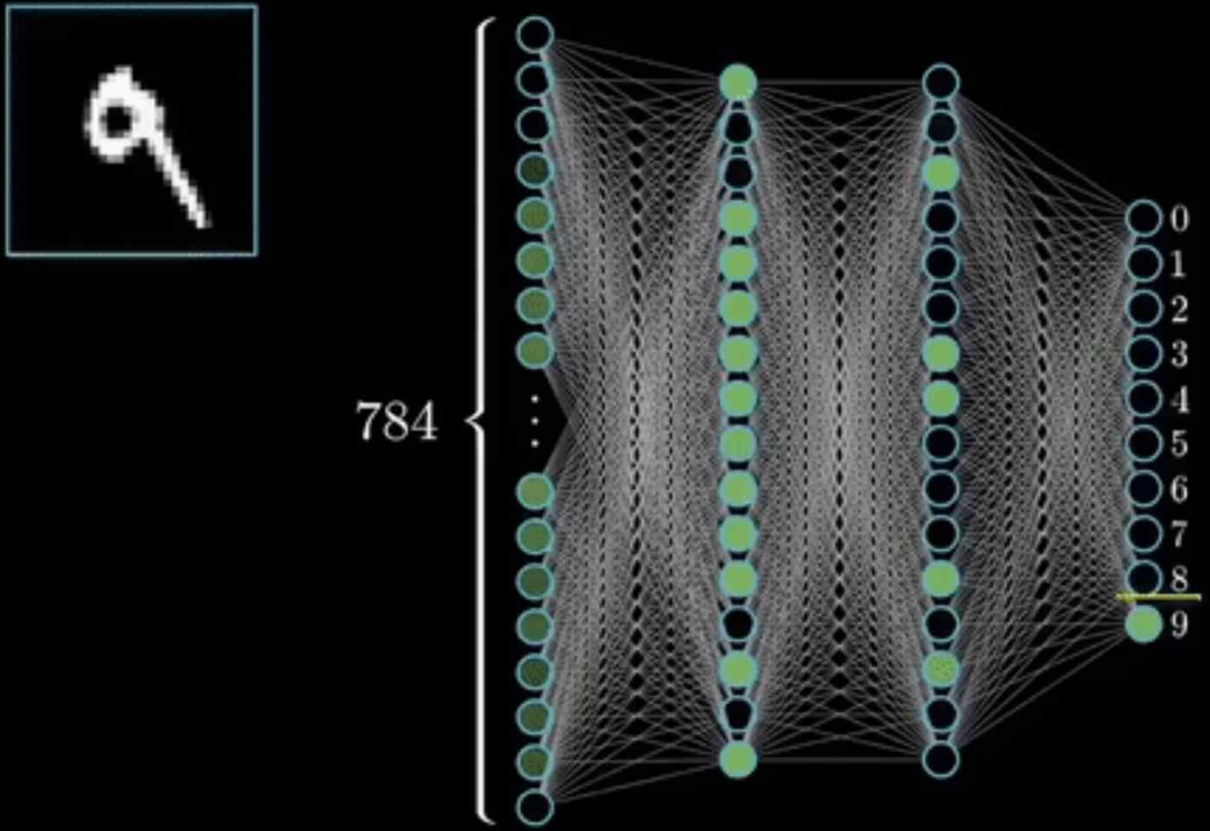
Hyper parameters

MNIST Network



```
self.l1 = nn.Linear(784, 520)
      self.l2 = nn.Linear(520, 320)
           self.l3 = nn.Linear(320, 240)
                self.l4 = nn.Linear(240, 120)
                     self.l5 = nn.Linear(120, 10)
```

MNIST


784
0
1
2
3
4
5
6
7
8
9