Network = 28 * 28 MNIST

filter = 3 * 3

Filter_bias = 0

Fordward pass

(feature * filter) + filter_bias = Feature Map → ReLU → POOLING
↓
Neural NET
↓
softmax

Backward pass

ENTROPY → ReLU' → feature map

NEURAL NET

Layers - 3

activations - ReLU & softmax

forward pass

$$\text{feature map}[i,j] = \left(\sum_{i=0}^{x}\sum_{j=0}^{y}\sum_{u=0}^{3}\sum_{v=0}^{3} \text{filter}[u,v] * \text{feature}[i+u, j+v]\right) + \text{filter\_bias}$$

$$X = \frac{N-L}{1} + 1 = \frac{28-3}{1} + 1 = 26$$

$$Y = \frac{N-L}{1} + 1 = \frac{28-3}{1} + 1 = 26$$

Feature Map = ReLU (feature map)

Pooling _ Indexes

P = np.zeros(x, y)

$$x, y = \frac{\text{f\_Length} - \text{pool\_Arr\_Length}}{2} + 1$$

$$= \frac{26-2}{2} + 1 = 13$$

indexes = np.zero((x, y), d_type = tuple)

```
for i in x
  for j in y
    i_start = i * 2 (stride)   if i = 12, i_start = 24
    j_start = j * 2 (stride)   if j = 12, j_start = 24

     window = feature_map[i_start : i_start+2, j_start : j_start+2]

     value = max (window)
     index = argmax (window)
     i_dash = index / 2 (stride)
     j_dash = index % 2

     P[i,j] = value
     indexes[i_start, j_start] = (i_start + i_dash, j_start + j_dash)
```

Neural Network

Layers = 169, 70, 10

weights = [(70, 169), (10, 70)]

biases = [(70,1), (10,1)]

L_1 = A0 = 13 * 13 = Pooled Array = (169, 1)

$$L_2 = A_1 = \sum_{i=0}^{169}\sum_{j=0}^{70} \text{ReLU}\left(w_1^{ji} \cdot A_0^{i} + b_1\right) = \text{ReLU}(z1) \quad \therefore \text{shape } (70,1)$$

$$L_3 = A_2 = \sum_{j=0}^{70}\sum_{k=0}^{10} \text{SoftMax}\left(w_2^{kj} \cdot A_1^{j} + b_2\right) = \text{Softmax}(z2) \quad \therefore \text{shape } (10,1)$$

Cross Entropy = $A_2 - Y$

BACK PROPagation

$$\frac{d\ \text{Cross Entropy}}{dz^2} \Big/ \boxed{\delta^2 = a^2 - 1} \quad \therefore \text{Shape } (10,1)$$

$$\frac{dc}{dw^2} = \frac{dc}{dz^2} \cdot \frac{dz^2}{dw^2}$$

$$= \delta^2 \cdot a^{1T} \quad \therefore \text{shape} = (10,70) \therefore w^2 = w^2 - \eta \cdot \frac{dL}{dw^2}$$

$$\frac{dc}{db^2} = \delta^2 \quad \therefore \text{Shape } (10,1) \quad \therefore b^2 = b^2 - \eta \cdot \frac{dL}{db^2}$$

$$\frac{dc}{da^1} = \frac{dc}{dz^2} \cdot \frac{dz^2}{da^1}$$

$$= \delta^2 \cdot w^2$$

$$\frac{dc}{dz^1} = \frac{dc}{da^1} \cdot \frac{da^1}{dz^1}$$

$$= \delta^2 \cdot w^2 \cdot \frac{d}{dz^1} \text{ReLU}(z^1)$$

$$\delta^1 = \delta^2 \cdot w^2 * \text{ReLU}'(z^1) \quad \therefore \text{Shape } [\overset{w^{2T}}{(70,10)} \cdot \overset{\delta^2}{(10,1)} * (70,1)] = (70,1) \quad \therefore b_1 = b_1 - \eta \cdot \frac{dc}{db_1}$$

$$\frac{dc}{dw^1} = \frac{dc}{dz^1} \cdot \frac{dz^1}{dw^1}$$

$$= \delta^1 \cdot \frac{d}{dw^1}(w^1 \cdot a^0 + b^1)$$

$$= \delta^1 \cdot a^{0T} \quad \therefore \text{shape } [(70,1) \cdot (1,169)] = (70,169). \quad \therefore w_1 = w_1 - \eta \cdot \frac{dL}{dw^1}$$

$$\frac{dc}{db^1} = \delta^1$$

$$\frac{dc}{da^0} = \frac{dc}{dz^1} \cdot \frac{dz^1}{da^0}$$

$$\delta^0 = \delta^1 \cdot w^{1T} \quad \therefore \text{shape } [\overset{w^{1T}}{(169,70)} \cdot \overset{\delta^1}{(70,1)}] \quad (169,1)$$

new_map = np.zeros((26, 26))

```
Back fill_map()
     for index in $\delta^0$
          new_map[indexes[i]] = $\delta^0$[i]
```

new_map_ReLU_Prime = new_map * ReLU'(feature_map)

// forward propagation = ReLU (feature_map)
// derivative = ReLU'(feature_map) * derivative_feature_map (new_map)

dw = filter derivative

dw = np.zeros[(3 * 3)]

```
fill_dw:
     for u in 0,3
        for v in 0,3
           for i in 0,26
             for j in 0,26
                 dw[u,v] += new_map[i,j] * Feature[i+u, j+v]
```

filter = filter - η * dw

d_filter_bias = sum(dw)

filter_bias = filter_bias - η * d_filter_bias