

# Neural Networks

## HW3 Computer Project2 CNN

授課老師

王振興

Due date

12/16

學號

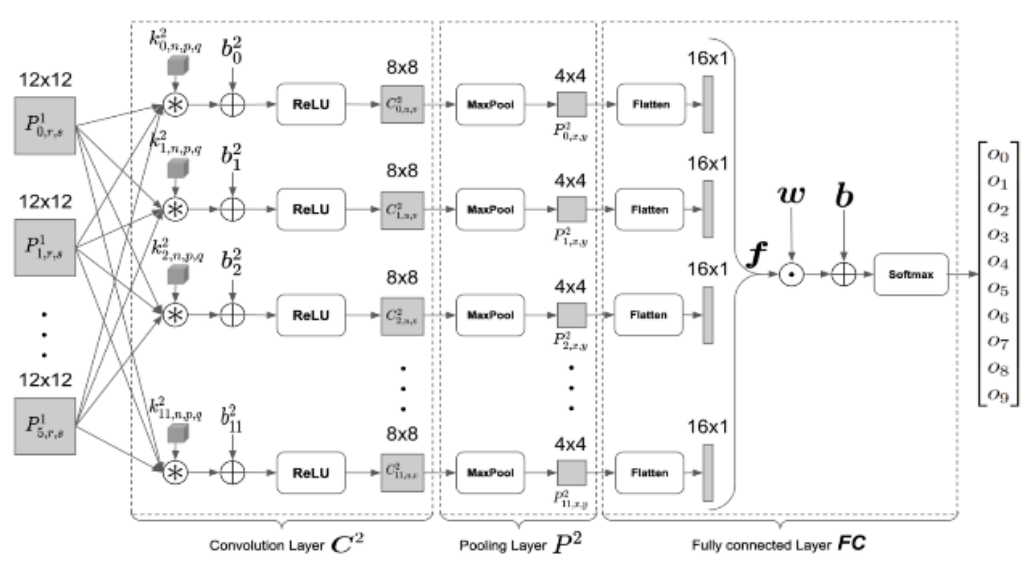
N26094883

姓名

杜冠勳

系所

電機所

derivative process	
handcraft	CNN
Model	RBF
CNN	<p>參考論文: REF.A gentle explanation of Backpropagation in Convolutional Neural Network Son Nguyen February 27, 2020</p>
	Gradient & BP 推導
	 <p>首先先從 softmax 層往回推</p> <div style="background-color: yellow; padding: 10px; margin: 10px 0;"> <math display="block">  \begin{aligned}  L &amp;= L(O_0, \dots, O_9), \\  &amp;= L(O_{label}) \\  &amp;= -\ln(O_{label}), \text{ label} \in \{0, \dots, 9\}  \end{aligned}  </math> </div> <p>因為 是採用 softmax 故使用 crossentropy 當作 loss 依據</p> <div style="background-color: yellow; padding: 10px; margin: 10px 0;"> <math display="block">  \begin{aligned}  f &amp;= \text{flatten}(P^2) \\  S_i &amp;= \sum_{j=0}^{191} w_{ij} f_j + b_i, \\  O_i &amp;= \text{softmax}(S_i) = \frac{e^{S_i}}{\sum_{k=0}^9 e^{S_k}}, \\  i &amp;= 0, \dots, 9  \end{aligned}  </math> </div> <p>藉由上式第二行可推</p> $  \frac{\partial S_k}{\partial b_i} = \begin{cases} 1 & \text{if } k = i \\ 0 & \text{if } k \neq i \end{cases}  $ <p>而因為</p>

$$\frac{\partial L}{\partial b_i} = \sum_{k=0}^9 \frac{\partial L}{\partial S_k} \frac{\partial S_k}{\partial b_i}, \quad i = 0, \dots, 9$$

可得

$$\frac{\partial L}{\partial b_i} = \frac{\partial L}{\partial S_i}, \quad i = 0, \dots, 9$$

利用 chain rule

$$\begin{aligned} \frac{\partial L}{\partial S_i} &= \frac{\partial L(O_{label})}{\partial O_{label}} \frac{\partial O_{label}}{\partial S_i} \\ \frac{\partial L(O_{label})}{\partial O_{label}} &= \frac{\partial(-\ln(O_{label}))}{\partial O_{label}} = -\frac{1}{O_{label}} \end{aligned}$$

而因為

$$O_{label} = \frac{e^{S_{label}}}{\sum_{k=0}^9 e^{S_k}}$$

將 T 用代數帶換

$$\begin{aligned} T &= \sum_{k=0}^9 e^{S_k}, \\ O_{label} &= \frac{e^{S_{label}}}{T} = e^{S_{label}} T^{-1} \end{aligned}$$

若是 label hits 的 case 令

$$e^{S_{label}} = T - C_1,$$

C 為一常數

可得

$$\frac{\partial O_{label}}{\partial S_{label}} = \frac{\partial((T - C_1)T^{-1})}{\partial S_{label}}$$

$$= C_1 T^{-2} \frac{\partial T}{\partial S_{label}}$$

$$\begin{aligned} \frac{\partial O_{label}}{\partial S_{label}} &= C_1 T^{-2} e^{S_{label}} \\ &= (C_1 T^{-1})(e^{S_{label}} T^{-1}) \\ &= (T - e^{S_{label}}) T^{-1} (e^{S_{label}} T^{-1}) \\ &= (1 - e^{S_{label}} T^{-1})(e^{S_{label}} T^{-1}) \\ &= (1 - O_{label}) O_{label} \end{aligned}$$

而當 label 不相同時 亦是此方式得到其 gradient  
Softmax 階段 小結

$$\frac{\partial O_{label}}{\partial S_i} = \begin{cases} (1 - O_{label}) O_{label} & \text{if } i = label \\ -O_{label} O_i & \text{if } i \neq label \end{cases}, i = 0, \dots, 9$$

$$\frac{\partial L}{\partial S_i} = \begin{cases} O_{label} - 1 & \text{if } i = label \\ O_i & \text{if } i \neq label \end{cases}, i = 0, \dots, 9$$

$$\frac{\partial L}{\partial b_i} = \begin{cases} O_{label} - 1 & \text{if } i = label \\ O_i & \text{if } i \neq label \end{cases}, i = 0, \dots, 9$$

接下來再往回推 要更新 FC layer 的 weight

$$\frac{\partial L}{\partial w_{ij}} = \sum_{k=0}^9 \frac{\partial L}{\partial S_k} \frac{\partial S_k}{\partial w_{ij}}, i = 0, \dots, 9, j = 0, \dots, 191$$

$$\frac{\partial S_k}{\partial w_{ij}} = \frac{\partial(\sum_{t=0}^{191} w_{kt} f_t + b_k)}{\partial w_{ij}} = \begin{cases} f_j & \text{if } k = i \\ 0 & \text{if } k \neq i \end{cases}$$

$$\frac{\partial L}{\partial w_{ij}} = \frac{\partial L}{\partial S_i} f_j, \quad i = 0, \dots, 9, \quad j = 0, \dots, 191$$

回推到 maxpooling & convolution 層 要更新  
該 bias

$$\frac{\partial L}{\partial b_m^2} = \sum_{u=0}^7 \sum_{v=0}^7 \frac{\partial L}{\partial S_{muv}^2} \frac{\partial S_{muv}^2}{\partial b_m^2}$$

$$\frac{\partial L}{\partial S_{muv}^2} = \frac{\partial L}{\partial C_{muv}^2} \frac{\partial C_{muv}^2}{\partial S_{muv}^2}$$

Maxpooling 前 為 convolution 3D output shape 為 (8 8 12)

故 bias 更新數量應為 12

若今天更新回去的路徑上 是 maxpooling 最大值的元素則

$$\frac{\partial L}{\partial C_{muv}^2} = \frac{\partial L}{\partial C_{mu_{max}v_{max}}^2} = \frac{\partial L}{\partial P_{mxy}^2}$$

令  $f_k = P_{mxy}$

$$\frac{\partial L}{\partial C_{mu_{max}v_{max}}^2} = \frac{\partial L}{\partial P_{mxy}^2} = \frac{\partial L}{\partial f_k}$$

By chain rule 從 output 端口回推

$$\frac{\partial L}{\partial f_k} = \sum_{i=0}^9 \frac{\partial L}{\partial S_i} \frac{\partial S_i}{\partial f_k}$$

$$\frac{\partial L}{\partial f_k} = \sum_{i=0}^9 \frac{\partial L}{\partial S_i} w_{ik}$$

今 convolution 為 1D array 則

$$dL_f = w^T \cdot dL_S$$

今 convolution 為 3D array 則

$$dLP2 = dL_f \cdot \text{reshape}(P^2, \text{shape})$$

若今天 要對 pooling 不是最大值的 neuron 更新 可以發現 其不需要更新 因為其梯度造成的影響甚小

## 小結

$$\frac{\partial L}{\partial C_{muv}^2} = \begin{cases} \frac{\partial L}{\partial P_{mxy}^2}, & (x = \text{floor}(u/2), y = \text{floor}(v/2)) \text{ if } C_{muv}^2 \text{ is the maximum element} \\ & \text{out of 4 elements.} \\ 0 & \text{otherwise} \end{cases}$$

## RBFN

```
def call(self, inputs):
    diff = K.expand_dims(inputs) - self.mu
    l2 = K.sum(K.pow(diff, 2), axis=1)
    res = K.exp(-1 * self.gamma * l2)
    # res = 1/K.sqrt(1 + self.gamma * l2)
    return res
```

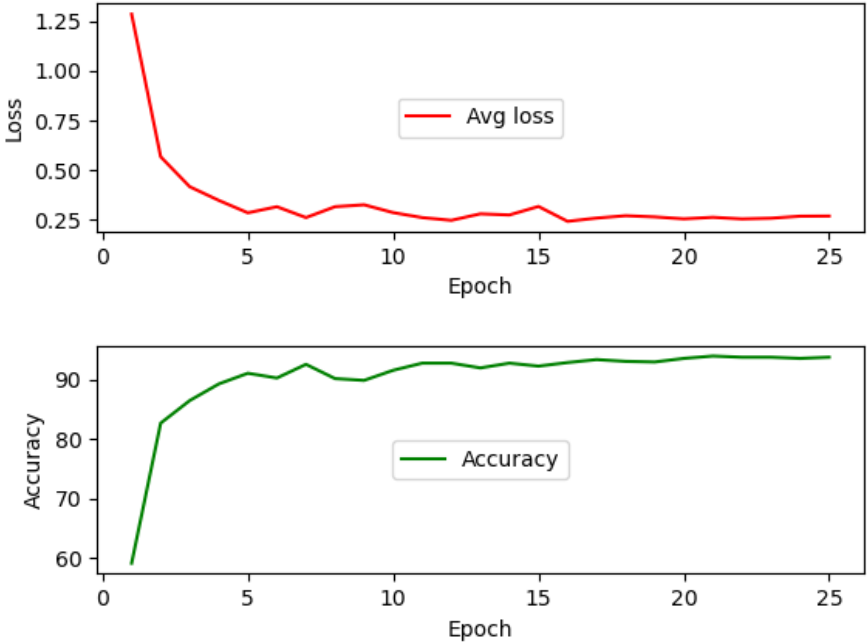
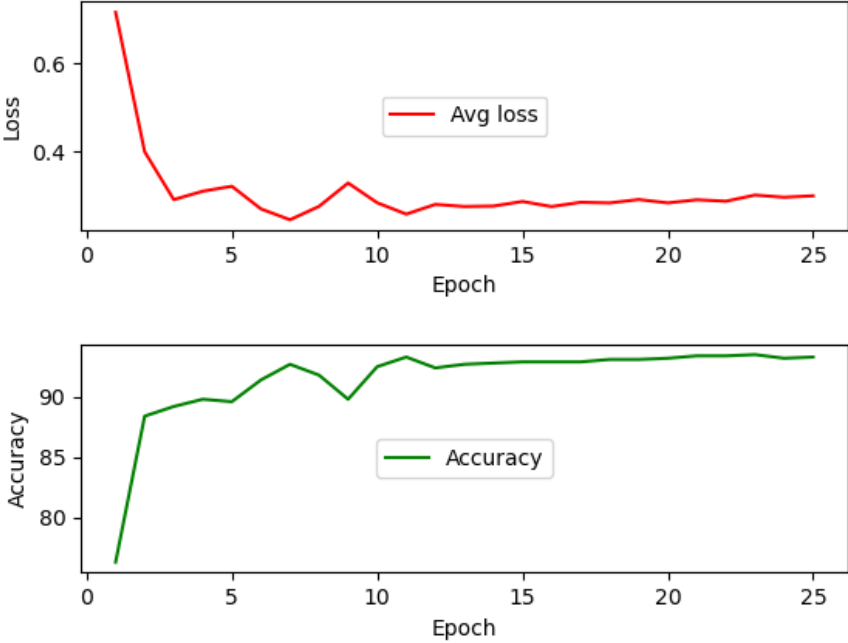
$$y(\mathbf{x}) = \sum_{i=1}^N w_i \varphi(\|\mathbf{x} - \mathbf{x}_i\|),$$

N 為 hidden layer 的 neuron 個數  $\mathbf{x}_i$  代表的是中心點位置  $w$  為 權重  $y$  為 該 neuron 的輸出值

淨向基函數將 input 與一定點去做距離比較

距離的計算方式有許多種 本次作業採用了高斯函數逆多二次函數

$$\exp\left[-\beta\|\mathbf{x} - \mathbf{c}_i\|^2\right].$$

Result																																																																																
Model_name	lr	PICTURE																																																																														
CNN	0.05	 <p>The figure displays two line graphs for a CNN model with a learning rate of 0.05 over 25 epochs. The top graph, titled 'Avg loss', shows a red line starting at 1.25 and decreasing to approximately 0.25. The bottom graph, titled 'Accuracy', shows a green line starting at 60% and increasing to approximately 95%.</p> <table border="1"><thead><tr><th>Epoch</th><th>Avg loss</th><th>Accuracy (%)</th></tr></thead><tbody><tr><td>1</td><td>1.25</td><td>60</td></tr><tr><td>2</td><td>0.55</td><td>83</td></tr><tr><td>3</td><td>0.40</td><td>87</td></tr><tr><td>4</td><td>0.35</td><td>89</td></tr><tr><td>5</td><td>0.30</td><td>91</td></tr><tr><td>6</td><td>0.32</td><td>90</td></tr><tr><td>7</td><td>0.28</td><td>92</td></tr><tr><td>8</td><td>0.32</td><td>90</td></tr><tr><td>9</td><td>0.33</td><td>90</td></tr><tr><td>10</td><td>0.30</td><td>91</td></tr><tr><td>11</td><td>0.28</td><td>92</td></tr><tr><td>12</td><td>0.25</td><td>93</td></tr><tr><td>13</td><td>0.28</td><td>92</td></tr><tr><td>14</td><td>0.30</td><td>93</td></tr><tr><td>15</td><td>0.32</td><td>92</td></tr><tr><td>16</td><td>0.25</td><td>93</td></tr><tr><td>17</td><td>0.28</td><td>94</td></tr><tr><td>18</td><td>0.28</td><td>94</td></tr><tr><td>19</td><td>0.28</td><td>93</td></tr><tr><td>20</td><td>0.27</td><td>94</td></tr><tr><td>21</td><td>0.27</td><td>95</td></tr><tr><td>22</td><td>0.26</td><td>95</td></tr><tr><td>23</td><td>0.26</td><td>95</td></tr><tr><td>24</td><td>0.27</td><td>95</td></tr><tr><td>25</td><td>0.27</td><td>95</td></tr></tbody></table>	Epoch	Avg loss	Accuracy (%)	1	1.25	60	2	0.55	83	3	0.40	87	4	0.35	89	5	0.30	91	6	0.32	90	7	0.28	92	8	0.32	90	9	0.33	90	10	0.30	91	11	0.28	92	12	0.25	93	13	0.28	92	14	0.30	93	15	0.32	92	16	0.25	93	17	0.28	94	18	0.28	94	19	0.28	93	20	0.27	94	21	0.27	95	22	0.26	95	23	0.26	95	24	0.27	95	25	0.27	95
Epoch	Avg loss	Accuracy (%)																																																																														
1	1.25	60																																																																														
2	0.55	83																																																																														
3	0.40	87																																																																														
4	0.35	89																																																																														
5	0.30	91																																																																														
6	0.32	90																																																																														
7	0.28	92																																																																														
8	0.32	90																																																																														
9	0.33	90																																																																														
10	0.30	91																																																																														
11	0.28	92																																																																														
12	0.25	93																																																																														
13	0.28	92																																																																														
14	0.30	93																																																																														
15	0.32	92																																																																														
16	0.25	93																																																																														
17	0.28	94																																																																														
18	0.28	94																																																																														
19	0.28	93																																																																														
20	0.27	94																																																																														
21	0.27	95																																																																														
22	0.26	95																																																																														
23	0.26	95																																																																														
24	0.27	95																																																																														
25	0.27	95																																																																														
CNN	0.01	 <p>The figure displays two line graphs for a CNN model with a learning rate of 0.01 over 25 epochs. The top graph, titled 'Avg loss', shows a red line starting at 0.7 and decreasing to approximately 0.3. The bottom graph, titled 'Accuracy', shows a green line starting at 75% and increasing to approximately 95%.</p> <table border="1"><thead><tr><th>Epoch</th><th>Avg loss</th><th>Accuracy (%)</th></tr></thead><tbody><tr><td>1</td><td>0.70</td><td>75</td></tr><tr><td>2</td><td>0.40</td><td>88</td></tr><tr><td>3</td><td>0.30</td><td>89</td></tr><tr><td>4</td><td>0.32</td><td>90</td></tr><tr><td>5</td><td>0.33</td><td>89</td></tr><tr><td>6</td><td>0.30</td><td>90</td></tr><tr><td>7</td><td>0.28</td><td>92</td></tr><tr><td>8</td><td>0.30</td><td>91</td></tr><tr><td>9</td><td>0.34</td><td>89</td></tr><tr><td>10</td><td>0.30</td><td>92</td></tr><tr><td>11</td><td>0.28</td><td>94</td></tr><tr><td>12</td><td>0.28</td><td>93</td></tr><tr><td>13</td><td>0.28</td><td>94</td></tr><tr><td>14</td><td>0.28</td><td>94</td></tr><tr><td>15</td><td>0.28</td><td>94</td></tr><tr><td>16</td><td>0.28</td><td>94</td></tr><tr><td>17</td><td>0.28</td><td>94</td></tr><tr><td>18</td><td>0.28</td><td>94</td></tr><tr><td>19</td><td>0.28</td><td>94</td></tr><tr><td>20</td><td>0.28</td><td>94</td></tr><tr><td>21</td><td>0.28</td><td>94</td></tr><tr><td>22</td><td>0.28</td><td>94</td></tr><tr><td>23</td><td>0.28</td><td>94</td></tr><tr><td>24</td><td>0.28</td><td>94</td></tr><tr><td>25</td><td>0.28</td><td>94</td></tr></tbody></table>	Epoch	Avg loss	Accuracy (%)	1	0.70	75	2	0.40	88	3	0.30	89	4	0.32	90	5	0.33	89	6	0.30	90	7	0.28	92	8	0.30	91	9	0.34	89	10	0.30	92	11	0.28	94	12	0.28	93	13	0.28	94	14	0.28	94	15	0.28	94	16	0.28	94	17	0.28	94	18	0.28	94	19	0.28	94	20	0.28	94	21	0.28	94	22	0.28	94	23	0.28	94	24	0.28	94	25	0.28	94
Epoch	Avg loss	Accuracy (%)																																																																														
1	0.70	75																																																																														
2	0.40	88																																																																														
3	0.30	89																																																																														
4	0.32	90																																																																														
5	0.33	89																																																																														
6	0.30	90																																																																														
7	0.28	92																																																																														
8	0.30	91																																																																														
9	0.34	89																																																																														
10	0.30	92																																																																														
11	0.28	94																																																																														
12	0.28	93																																																																														
13	0.28	94																																																																														
14	0.28	94																																																																														
15	0.28	94																																																																														
16	0.28	94																																																																														
17	0.28	94																																																																														
18	0.28	94																																																																														
19	0.28	94																																																																														
20	0.28	94																																																																														
21	0.28	94																																																																														
22	0.28	94																																																																														
23	0.28	94																																																																														
24	0.28	94																																																																														
25	0.28	94																																																																														

CNN+RBF	0.05	<div data-bbox="558 123 965 414"> <p>RBF accuracy</p> </div> <div data-bbox="981 123 1388 414"> <p>RBF loss</p> </div>
CNN+RBF	0.005	<div data-bbox="558 459 965 750"> <p>RBF accuracy</p> </div> <div data-bbox="981 459 1388 750"> <p>RBF loss</p> </div>
RBF	0.05	<div data-bbox="558 795 965 1086"> <p>RBF accuracy</p> </div> <div data-bbox="981 795 1388 1086"> <p>RBF loss</p> </div>



- 從結果上來看純粹用 Dense + RBF 的網路 因為沒有經過特徵篩選 故會相較於 CNN based RBF 網路在相同的 learning rate 下較慢 提升 accuracy
- 選用 CNN 取代 dense 除了獲取到較好的特徵外 也減少了 weight 的數量 對硬體上設計也是較好的
- 相同的 learning rate 下 CNN 相較於 CNN +RBF 可以接受較大的 learning rate 來找到更好的 performance
- 同樣的網路架構 learning rate 調過大可能導致無法收斂 CNN +RBF 需用較小的 learning rate tune 出 model 但也不排除是因為網路較為複雜的緣故。