

圖形識別作業三

CNN Using Matlab Toolbox

學號：0416302

姓名：戴宏周

一、一維Convolution

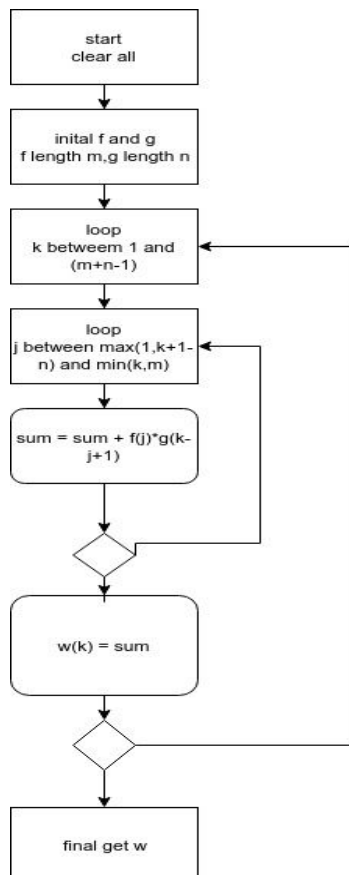
1. 敘述

進行下列運算：

$m = \text{length}(f)$ and $n = \text{length}(g)$

$$w(k) = \sum_j f(j)g(k-j+1)$$

2. flowchart



3. Toolbox 解法

使用conv()函式進行運算。

4. 結果

由於不混合時序呈現，且Toolbox與自己寫的結果相同為：

0.0100 0.0200 0.0300 0.0400 0.0300 0.0200 0.0100

二、二維convolution

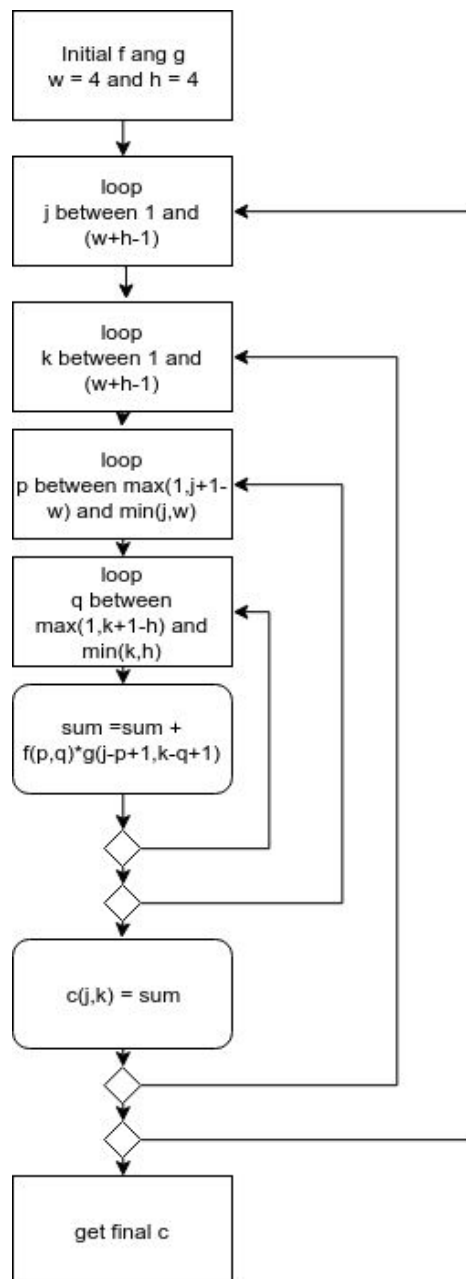
1. 敘述

進行下列運算：

$$C(j, k) = \sum_p \sum_q f(p, q)g(j - p + 1, k - q + 1)$$

p與q為所有 $f(p, q)$ 與 $g(j - p + 1, k - q + 1)$ 之合法值位置。

2. flowchart



3. ToolBox 解法

使用conv2函式

4. 結果

Toolbox與自行編寫兩者結果相同如下：

0.0100	0.0200	0.0300	0.0400	0.0300	0.0200	0.0100
0.0200	0.0400	0.0600	0.0800	0.0600	0.0400	0.0200
0.0300	0.0600	0.0900	0.1200	0.0900	0.0600	0.0300
0.0400	0.0800	0.1200	0.1600	0.1200	0.0800	0.0400
0.0300	0.0600	0.0900	0.1200	0.0900	0.0600	0.0300
0.0200	0.0400	0.0600	0.0800	0.0600	0.0400	0.0200
0.0100	0.0200	0.0300	0.0400	0.0300	0.0200	0.0100

三、CNN Toolbox實驗

1. 敘述

使用Deep learning toolbox進行運算，使先將MNIST資料讀入並將label轉為categorical型態。接下來建立layer如下：

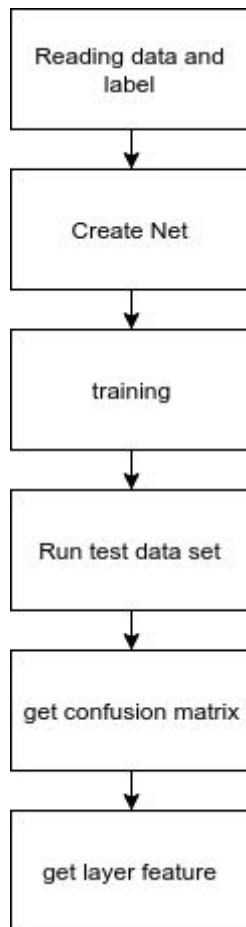


此為常見兩層Convoolution架構，輸入層為28X28大小的圖片，第一層convloution使用的filter size為3，並且建立8組filter，之後進行對該層minibatch進行batch normalization，之後使用ReLU作為Activation function，最後使用maxPooling layer提取2X2區塊中最大值。下一組Convolution架構也相同，不過filter數量倍增至16組。經過兩組Convolution, batch normalization, relu, max pooling後，先連接一層fully connected layer，並將輸出設為10對應到10種class，之後透過softmax層將每個輸出映射到(0,1)區間，且所有輸出總和為1。最後透過classification layer將結果輸出。

網路建構完成透過trainNetwork函式進行運算。使用BatchSize為128, MaxEpochs為10，使用Stochastic Gradient Descent with Momentum(sgdm)最為優化方式(solver)。

訓練完成後輸出結果，並繪製成confusion matrix，並將各層feature結果抽取出來。使用activations函式得出各層抽取結果。

2. flow chart

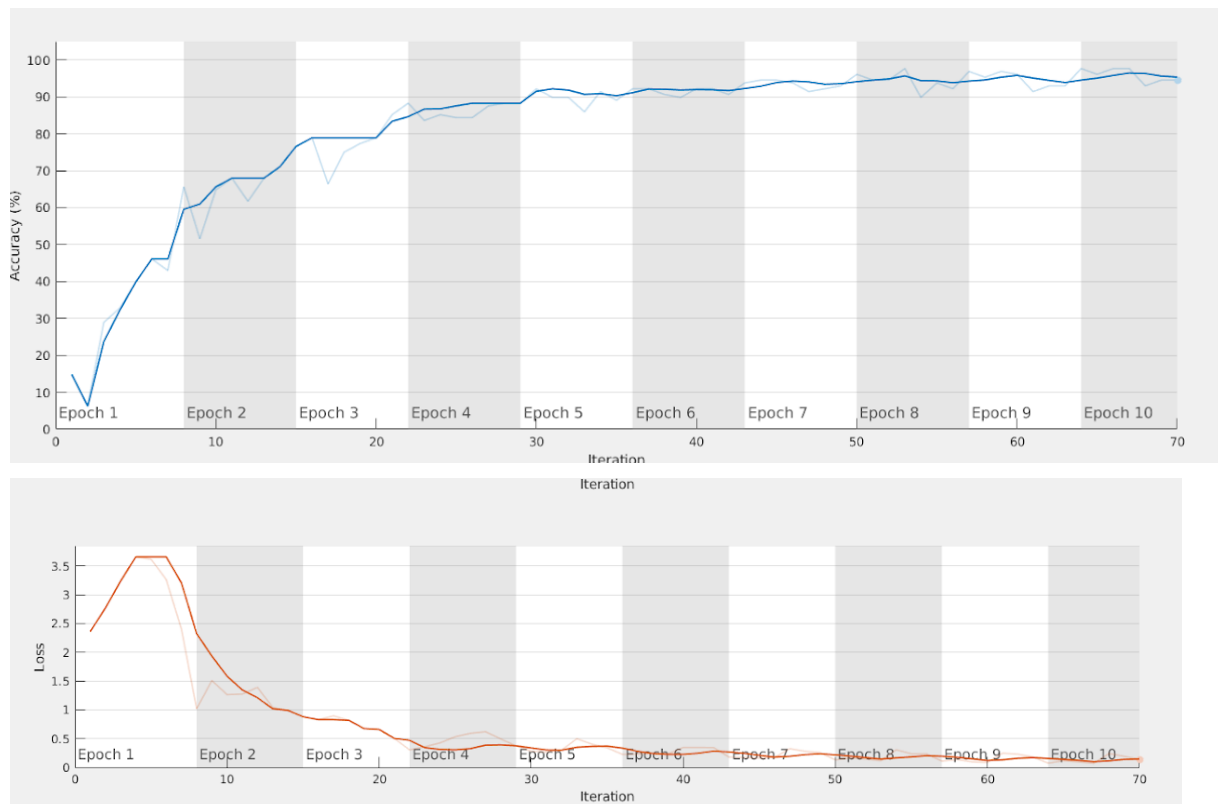


3. 結果


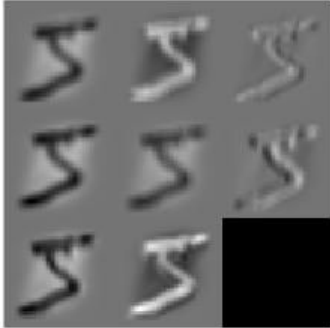
A. 選取 1000 個 examples 做 training, 1000 個做 testing。

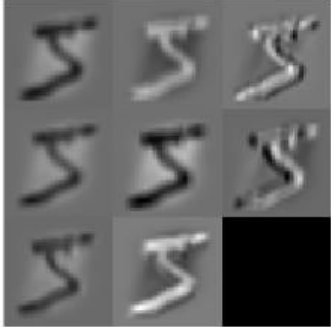


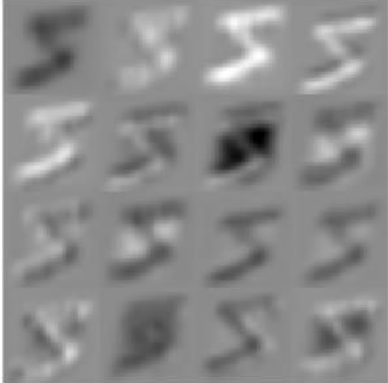
1. CPU time, error vs iteration(epochs)

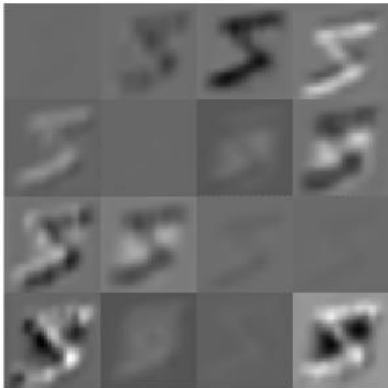
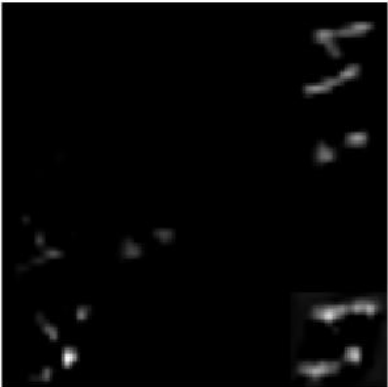
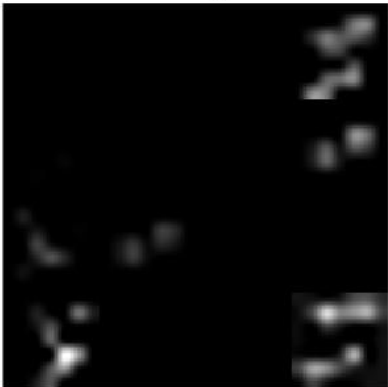
總共經過7秒，共10個epochs，使用單核心cpu，learning rate為0.1

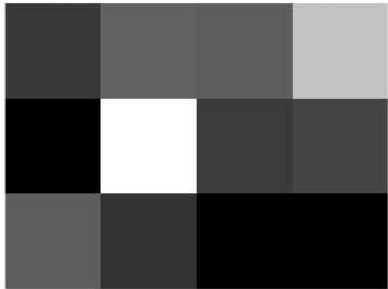
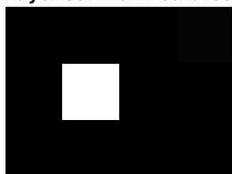
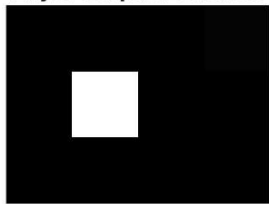


2. layer feature (輸入圖片label 為5)

layer	圖片
layer 1:input layer	Layer input Features 
layer2: convolution conv_1 (8個filter)	Layer conv₁ Features 

<p>layer3: batch normalization norm_1 (8個channel, 8組圖)</p>	<p>Layer norm₁ Features</p> 
<p>layer4: relu relu_1 (8個channel, 8組圖)</p>	<p>Layer relu₁ Features</p> 
<p>layer5: max pooling Mpool_1 (8個channel, 8組圖)</p>	<p>Layer Mpool₁ Features</p> 
<p>layer6: convolution conv_2 (16個filter)</p>	<p>Layer conv₂ Features</p> 

<p>layer 7: batch normalization norm_2 (16個channel, 16組圖)</p>	<p>Layer norm₂ Features</p> 
<p>layer8: relu relu_2 (16個channel, 16組圖)</p>	<p>Layer relu₂ Features</p> 
<p>layer 9: max pooling Mpool_2 (16個channel, 16組圖)</p>	<p>Layer Mpool₂ Features</p> 

layer10: fully connected layer fc (僅10個格子，最後 兩個為黑為多餘)	<p style="text-align: center;">Layer fc Features</p> 
layer11: softmax (可見編號5的格子 為白色)	<p style="text-align: center;">Layer softmax Features</p> 
layer12: output (可見編號5的格子 為白色)	<p style="text-align: center;">Layer output Features</p> 

3. 150個測試圖形



4. 10X10 confusion Matrix

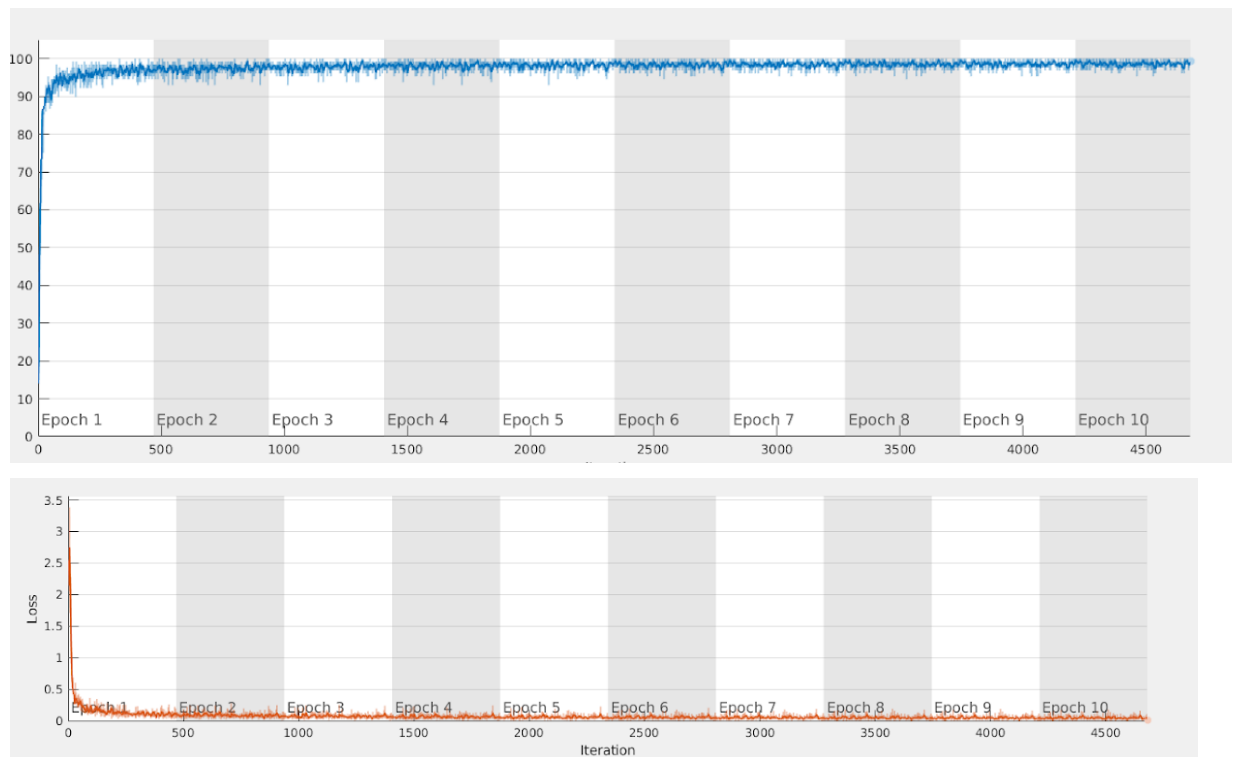
Confusion Matrix										
Output Class	1	2	3	4	5	6	7	8	9	10
	78 7.8%	0 0.0%	0 0.0%	0 0.0%	1 0.1%	1 0.1%	5 0.5%	0 0.0%	5 0.5%	0 0.0%
	0 0.0%	123 12.3%	1 0.1%	1 0.1%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	1 0.1%	0 0.0%
	1 0.1%	0 0.0%	106 10.6%	3 0.3%	0 0.0%	0 0.0%	4 0.4%	7 0.7%	4 0.4%	0 0.0%
	1 0.1%	0 0.0%	0 0.0%	84 8.4%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	1 0.1%	2 0.2%
	0 0.0%	0 0.0%	0 0.0%	0 0.0%	98 9.8%	1 0.1%	4 0.4%	0 0.0%	2 0.2%	0 0.0%
	2 0.2%	1 0.1%	0 0.0%	12 1.2%	0 0.0%	77 7.7%	4 0.4%	0 0.0%	0 0.0%	0 0.0%
	0 0.0%	0 0.0%	2 0.2%	0 0.0%	1 0.1%	1 0.1%	70 7.0%	0 0.0%	0 0.0%	0 0.0%
	0 0.0%	1 0.1%	4 0.4%	3 0.3%	2 0.2%	5 0.5%	0 0.0%	89 8.9%	4 0.4%	8 0.8%
	2 0.2%	1 0.1%	3 0.3%	3 0.3%	0 0.0%	1 0.1%	0 0.0%	0 0.0%	70 7.0%	0 0.0%
	1 0.1%	0 0.0%	0 0.0%	1 0.1%	8 0.8%	1 0.1%	0 0.0%	3 0.3%	2 0.2%	84 8.4%
										10
										91.8% 8.2%
										97.6% 2.4%
										91.4% 8.6%
										78.5% 21.5%
										89.1% 10.9%
										88.5% 11.5%
										80.5% 19.5%
										89.9% 10.1%
										78.7% 21.3%
										89.4% 10.6%
										87.9% 12.1%
Target Class										

B.全部的 60000 個 examples 做 training,10000 個做 testing。


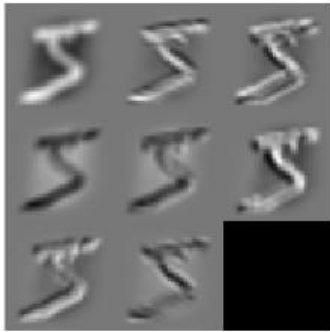
1. CPU time, error vs iteration(epochs)

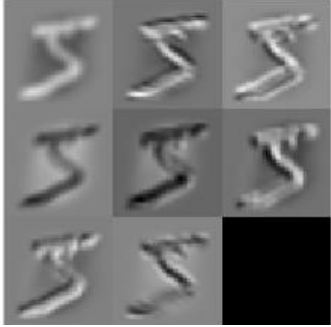


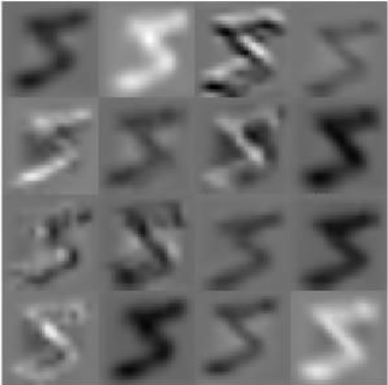
總共經過3分42秒，共10個epochs，共4680個iteration，使用單核心cpu，learning

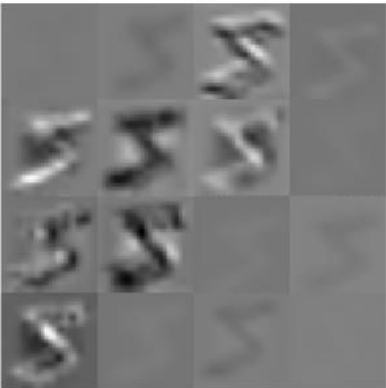

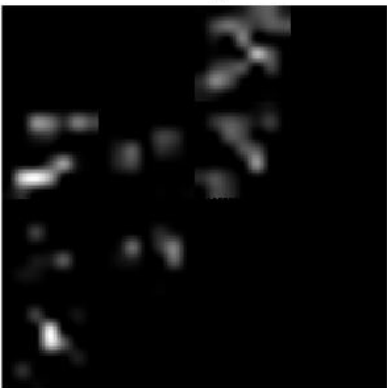
rate為0.1

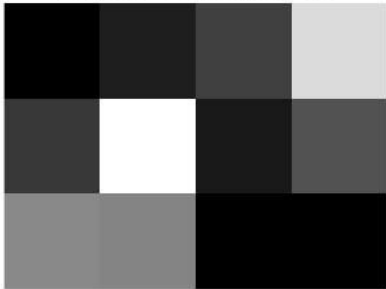
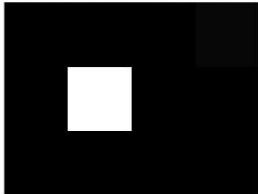
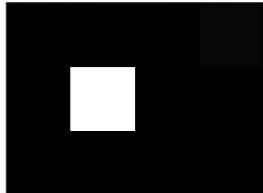


2. layer feature (輸入圖片label 為5)

layer	圖片
layer 1:input layer	Layer input Features 
layer2: convolution conv_1 (8個filter, 產生8組 feature map)	Layer conv₁ Features 

<p>layer3: batch normalization norm_1 (8個channel, 8組圖)</p>	<p>Layer norm₁ Features</p> 
<p>layer4: relu relu_1 (8個channel, 8組圖)</p>	<p>Layer relu₁ Features</p> 
<p>layer5: max pooling Mpool_1 (8個channel, 8組圖)</p>	<p>Layer Mpool₁ Features</p> 
<p>layer6: convolution conv_2 (16個filter)</p>	<p>Layer conv₂ Features</p> 

<p>layer 7: batch normalization norm_2 (16個channel, 16組圖)</p>	<p>Layer norm₂ Features</p> 
<p>layer8: relu relu_2 (16個channel, 16組圖)</p>	<p>Layer relu₂ Features</p> 
<p>layer 9: max pooling Mpool_2 (16個channel, 16組圖)</p>	<p>Layer Mpool₂ Features</p> 

layer10: fully connected layer fc (僅10個格子，最後 兩個為黑為多餘)	<p style="text-align: center;">Layer fc Features</p> 
layer11: softmax (可見編號5的格子 為白色)	<p style="text-align: center;">Layer softmax Features</p> 
layer12: output (可見編號5的格子 為白色)	<p style="text-align: center;">Layer output Features</p> 

3. 150個測試圖形



4. 10X10 confusion Matrix

Confusion Matrix											
Output Class	1	2	3	4	5	6	7	8	9	10	
	976 9.8%	0 0.0%	2 0.0%	1 0.0%	0 0.0%	2 0.0%	7 0.1%	0 0.0%	2 0.0%	6 0.1%	98.0% 2.0%
	0 0.0%	1128 11.3%	3 0.0%	0 0.0%	0 0.0%	1 0.0%	3 0.0%	10 0.1%	1 0.0%	2 0.0%	98.3% 1.7%
	0 0.0%	2 0.0%	1011 10.1%	1 0.0%	1 0.0%	2 0.0%	2 0.0%	14 0.1%	5 0.1%	0 0.0%	97.4% 2.6%
	0 0.0%	2 0.0%	3 0.0%	1003 10.0%	0 0.0%	13 0.1%	0 0.0%	4 0.0%	1 0.0%	10 0.1%	96.8% 3.2%
	1 0.0%	2 0.0%	3 0.0%	1 0.0%	975 9.8%	1 0.0%	10 0.1%	5 0.1%	2 0.0%	12 0.1%	96.3% 3.7%
	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	867 8.7%	4 0.0%	0 0.0%	0 0.0%	3 0.0%	99.2% 0.8%
	1 0.0%	0 0.0%	0 0.0%	0 0.0%	1 0.0%	2 0.0%	928 9.3%	0 0.0%	0 0.0%	0 0.0%	99.6% 0.4%
	1 0.0%	0 0.0%	4 0.0%	4 0.0%	0 0.0%	0 0.0%	0 0.0%	984 9.8%	1 0.0%	1 0.0%	98.9% 1.1%
	1 0.0%	1 0.0%	6 0.1%	0 0.0%	2 0.0%	2 0.0%	4 0.0%	3 0.0%	961 9.6%	2 0.0%	97.9% 2.1%
	0 0.0%	0 0.0%	0 0.0%	0 0.0%	3 0.0%	2 0.0%	0 0.0%	8 0.1%	1 0.0%	973 9.7%	98.6% 1.4%
Target Class											
	99.6% 0.4%	99.4% 0.6%	98.0% 2.0%	99.3% 0.7%	99.3% 0.7%	97.2% 2.8%	96.9% 3.1%	95.7% 4.3%	98.7% 1.3%	96.4% 3.6%	98.1% 1.9%

4. 討論

a. feature變換過程

在網路中feature會先過convolution取得邊界，之後透過ReLU強化結果，可以發現ReLU feature的圖形對比度較高。而Max pooling在其中扮演的角色比較像sub sample，不過是保留訊號最強的部份，可以發現Max pooling後的圖像解析度看似較差。第二次convolution後原本圖形的形狀變得更加模糊，且人類也變得較難以閱讀，不過可以從較強的訊號處發現，以5為例，她似乎判斷依據是5最上面那一橫豎以及5最後下方的轉折來判斷。最後可以發現原本fully connected layer的值比較散亂，經過softmax後變成只有以一個class的訊號比較突出。

b. 不同數量的train data有影響

可以從error曲線與iteration的關係，可以發現在同一數量epochs，資料量多的曲線就快到90%以上，但是資料集少的則到了第十個epoch才接近80%。當然由於資料量少，所以一個epochs中的iteration比較少。

c. 網路架構

這次使用兩層convolution的組合，每層不同數量的filter，產生更多的feature map，之後用batch normalization和relu強化結果，並使用max pooling進行處理，以便適應圖形位移。經過兩組convolution和後處理後，整個網路在適應不同字跡的數字上有了更高的彈性。可以從feature圖形變化看出上述過程。有趣的是，在第二層convolution結束後，並做完batch

normalization後，有多個feature map的圖片幾乎沒有圖形(白點)。即便通過ReLU後也一樣，可能有特徵在這次處理後消失了。

四、程式碼

所有程式均上傳e3。

1. PA_self.m: 1d convolution自己寫的版本。
2. PA_toolbox.m: 1d convolution使用conv。
3. PB_self.m: 2d convolution自己寫的版本。
4. PB_self.m: 2d convolution使用conv2。
5. PC_100.m: c小題MNIST資料集，選取 1000 個 examples 做 training,1000 個做 testing。
6. PC_full.m: c小題MNIST資料集，全部的 60000 個 examples 做 training,10000 個做 testing。
7. Get_MNIST.m: MNIST資料載入函式。
8. Get_MNISTLABEL.m: MNIST label資料載入程式。

五、參考

1d convolution:<https://www.mathworks.com/help/matlab/ref/conv.html>

2d convolution:<https://www.mathworks.com/help/matlab/ref/conv2.html>

convolution2dLayer:<https://www.mathworks.com/help/deeplearning/ref/nnet.cnn.layer.convolution2dlayer.html>

batchNormalizationLayer:<https://www.mathworks.com/help/deeplearning/ref/nnet.cnn.layer.batchnormalizationlayer.html>

reluLayer:<https://www.mathworks.com/help/deeplearning/ref/nnet.cnn.layer.relu.html>

maxPooling2sLayer:<https://www.mathworks.com/help/deeplearning/ref/nnet.cnn.layer.maxpooling2dlayer.html>

fullyConnectedLayer:<https://www.mathworks.com/help/deeplearning/ref/nnet.cnn.layer.fullyconnectedlayer.html>

softmaxLayer:<https://www.mathworks.com/help/deeplearning/ref/nnet.cnn.layer.softmaxlayer.html>

classificationLayer:<https://www.mathworks.com/help/deeplearning/ref/classificationlayer.html>

Gradient-based learning applied to document recognition, Y. Lecun ; L. Bottou ; Y. Bengio ; P. Haffner, 1998:<https://ieeexplore.ieee.org/document/726791>