國立雲林科技大學
電子工程系
Department of Electronic Engineering
教育部補助AI應用領域系列課程-
人工智慧計算晶片設計和應用人才培育

AGC Edge-AI教育訓練

第二周 CNN & LeNet

國立雲林科技大學
王斯弘 助理教授 / 前瞻學位學士學程

2021, Fall Semester

Outline

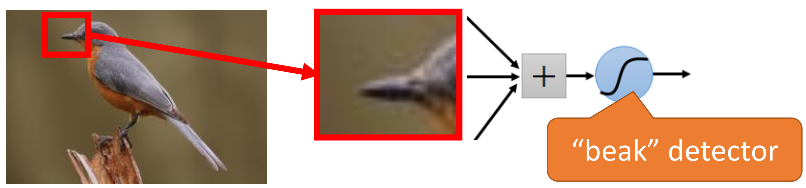
- **CNN**
- LeNet

Why CNN for Image

- Some patterns are much smaller than the whole image

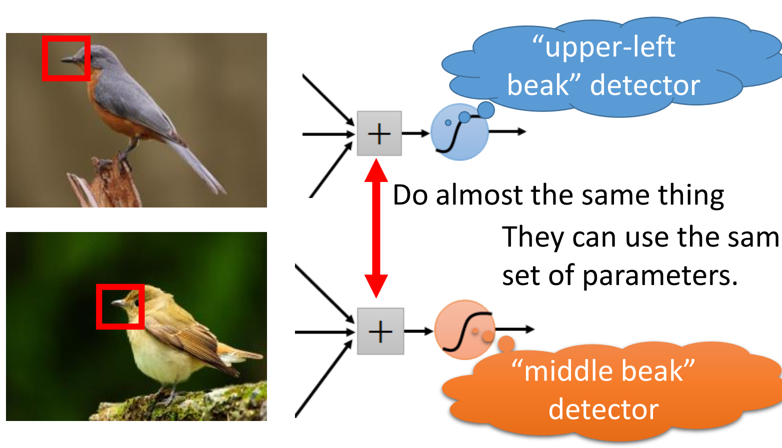
A neuron does not have to see the whole image to discover the pattern.

Connecting to small region with less parameters



Why CNN for Image

- The same patterns appear in different regions.



Why CNN for Image

- Subsampling the pixels will not change the object

bird

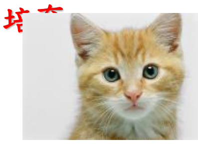
subsampling

bird

We can subsample the pixels to make image smaller

→ Less parameters for the network to process the image

The whole CNN



Convolution

Max Pooling

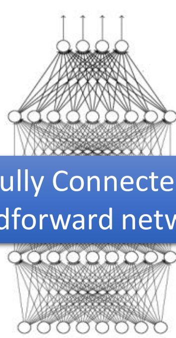
Convolution

Max Pooling

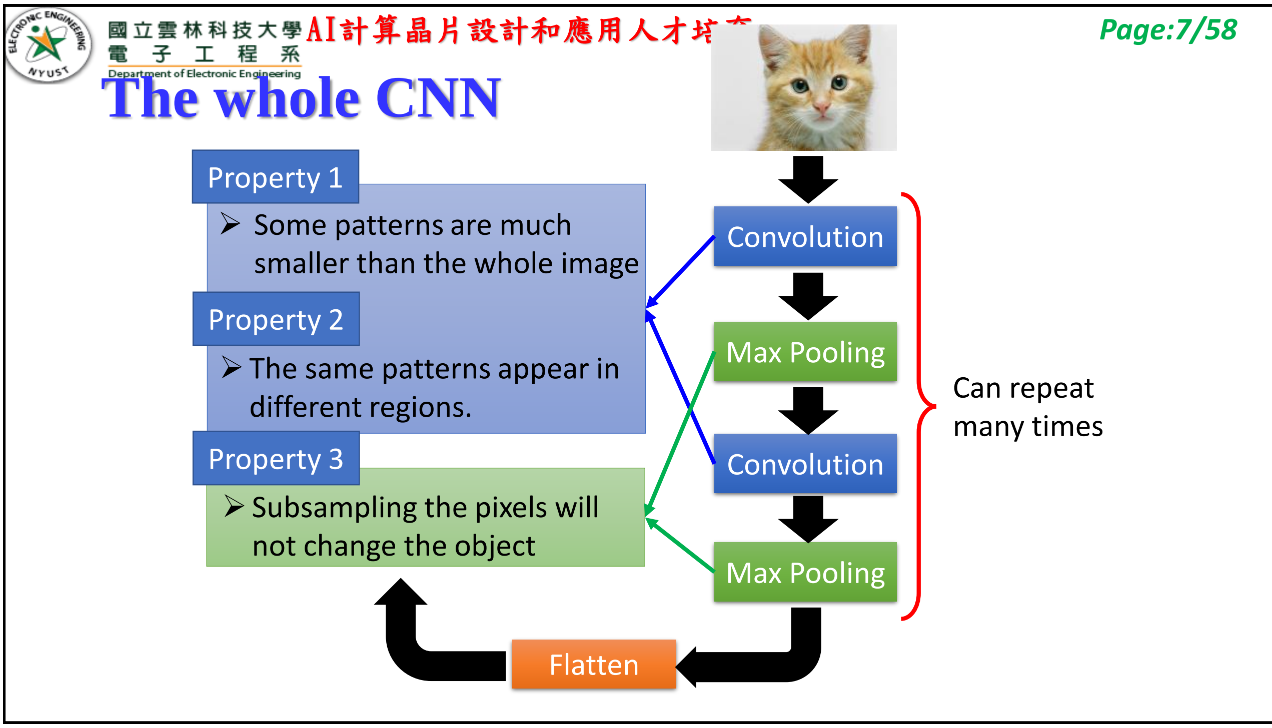
Can repeat many times

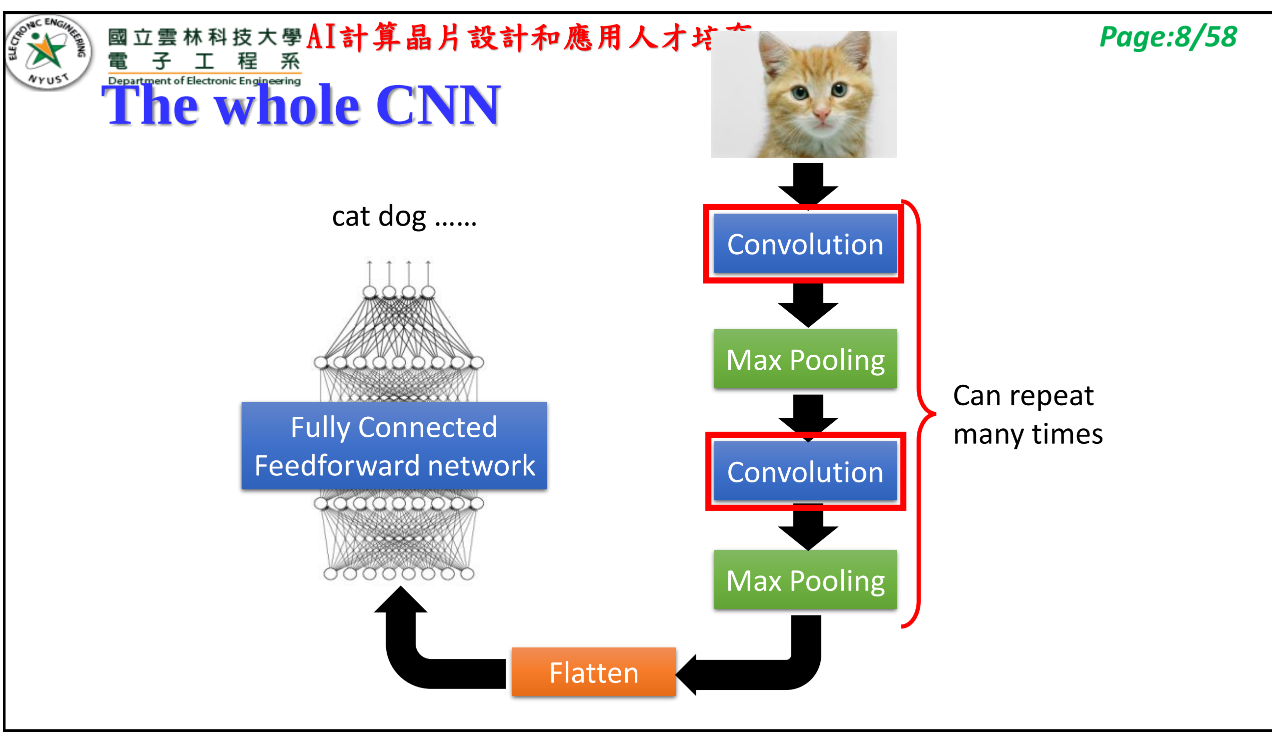
Flatten

Fully Connected Feedforward network



cat dog ......


國立雲林科技大學
電子工程系
Department of Electronic Engineering
AI計算晶片設計和應用人才培
Page:7/58
The whole CNN
Property 1
Some patterns are much smaller than the whole image
Property 2
The same patterns appear in different regions.
Property 3
Subsampling the pixels will not change the object
Convolution
Max Pooling
Convolution
Max Pooling
Can repeat many times
Flatten


國立雲林科技大學
電子工程系
Department of Electronic Engineering
AI計算晶片設計和應用人才培
Page:8/58
The whole CNN
cat dog ......
Fully Connected Feedforward network
Convolution
Max Pooling
Convolution
Max Pooling
Can repeat many times
Flatten

CNN – Convolution

Those are the network parameters to be learned.

1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

6 x 6 image

1	-1	-1
-1	1	-1
-1	-1	1

Filter 1 Matrix

-1	1	-1
-1	1	-1
-1	1	-1

Filter 2 Matrix

⋮

Property 1 Each filter detects a small pattern (3 x 3).

CNN – Convolution

1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

stride=1

1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

6 x 6 image

3 -1

國立雲林科技大學 電子工程系 Department of Electronic Engineering AI計算晶片設計和應用人才培育

Page:11/58

CNN – Convolution

1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

If stride=2

1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

6 x 6 image

3	-3

We set stride=1 below

國立雲林科技大學 電子工程系 Department of Electronic Engineering AI計算晶片設計和應用人才培育

Page:12/58

CNN – Convolution

1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

stride=1

1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

6 x 6 image

3	-1	-3	-1
-3	1	0	-3
-3	-3	0	1
3	-2	-2	-1

Property 2

國立雲林科技大學 電子工程系 Department of Electronic Engineering AI計算晶片設計和應用人才培育

CNN – Convolution

-1	1	-1
-1	1	-1
-1	1	-1

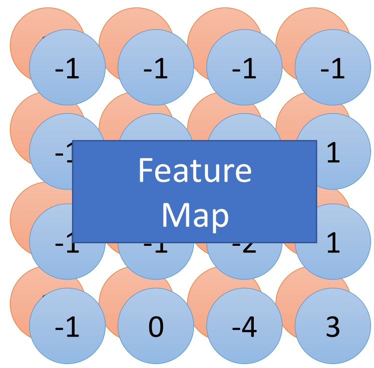
Filter 2

stride=1

1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

6 x 6 image

Do the same process for every filter



4 x 4 image

國立雲林科技大學 電子工程系 Department of Electronic Engineering AI計算晶片設計和應用人才培育

CNN – Colorful image

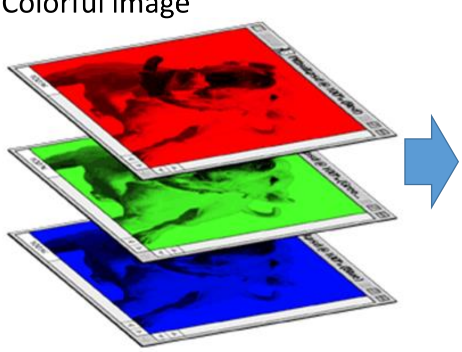
1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

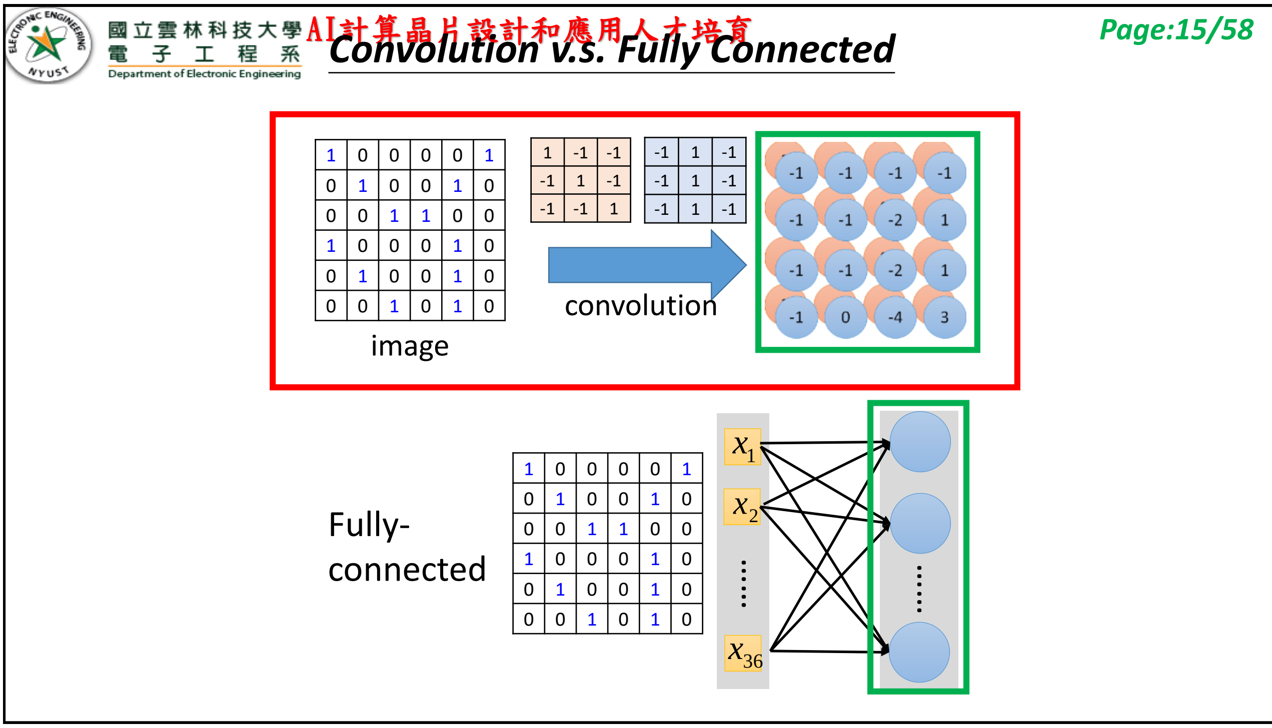
-1	1	-1
-1	1	-1
-1	1	-1

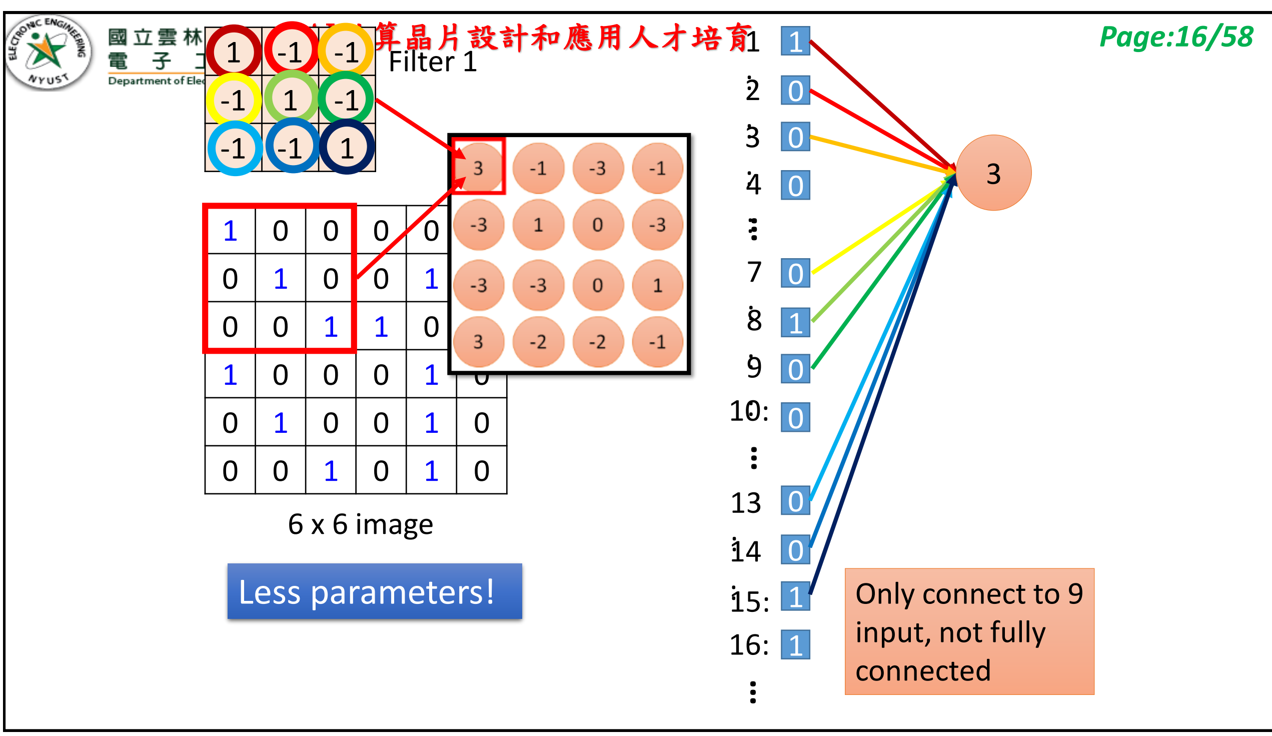
Filter 2

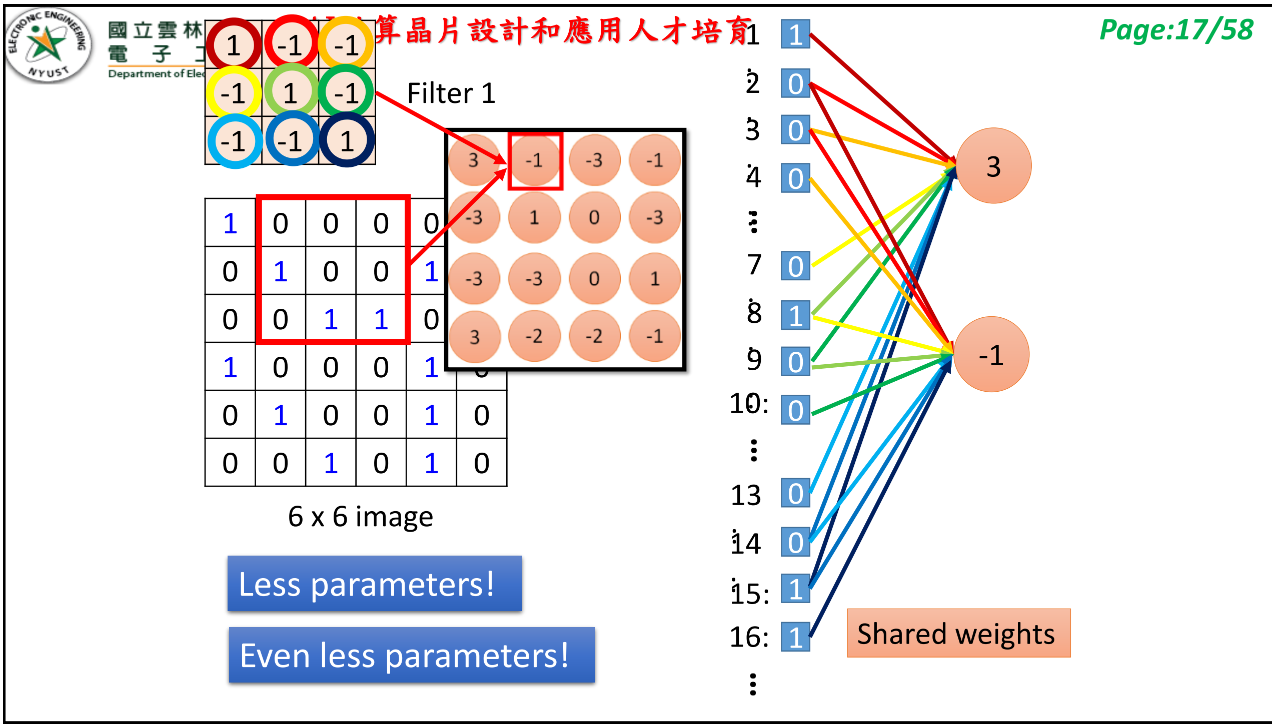
Colorful image

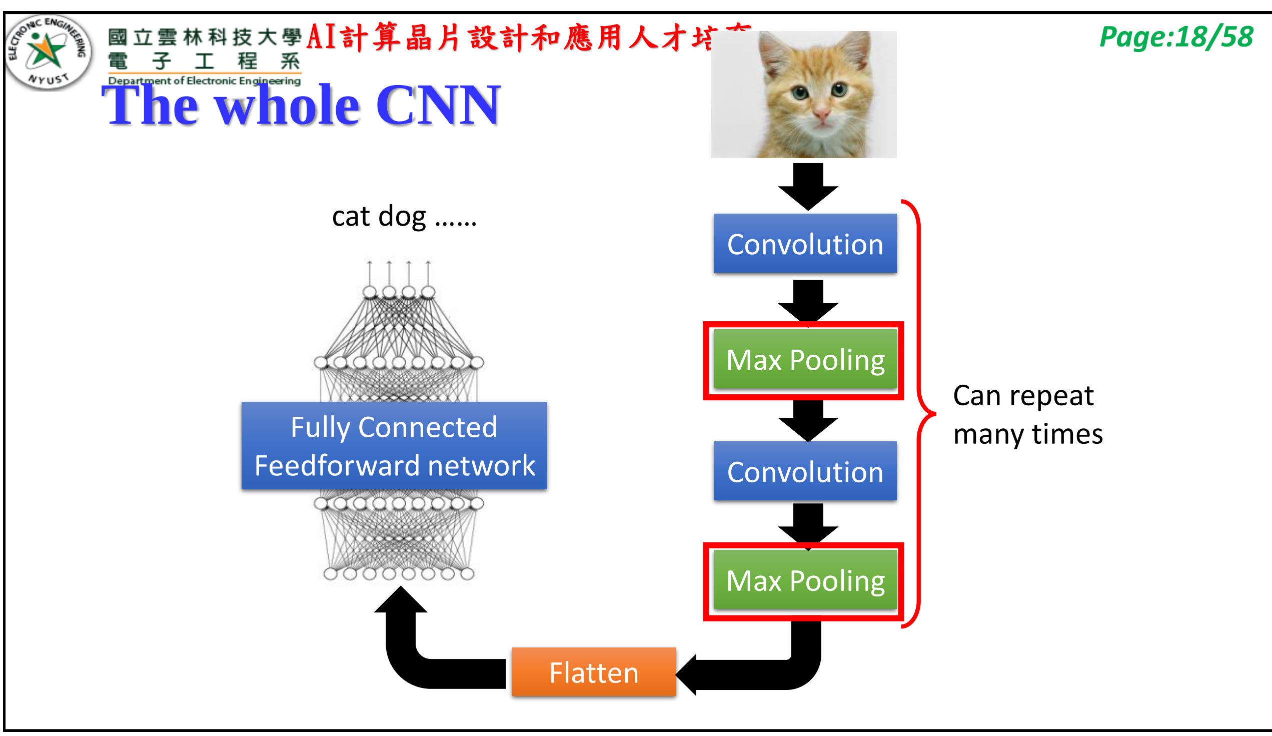


1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0


國立雲林科技大學 電子工程系 Department of Electronic Engineering
AI計算晶片設計和應用人才培育
Convolution v.s. Fully Connected
Page:15/58
image
convolution
Fully-connected
x_1
x_2
x_{36}


AI計算晶片設計和應用人才培育
Page:16/58
Filter 1
6 x 6 image
Less parameters!
Only connect to 9 input, not fully connected


國立雲林
電 子 工
算晶片設計和應用人才培育
Page:17/58
Filter 1
6 x 6 image
Less parameters!
Even less parameters!
Shared weights


國立雲林科技大學 AI計算晶片設計和應用人才培
電 子 工 程 系
Page:18/58
The whole CNN
cat dog ......
Fully Connected
Feedforward network
Convolution
Max Pooling
Convolution
Max Pooling
Can repeat
many times
Flatten

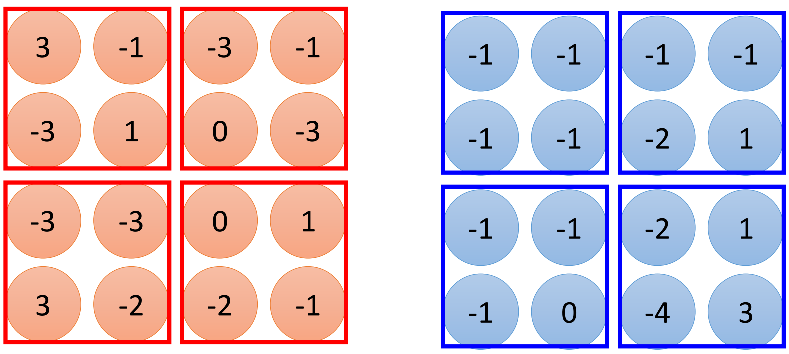
CNN – Max Pooling

Filter 1

1	-1	-1
-1	1	-1
-1	-1	1

Filter 2

-1	1	-1
-1	1	-1
-1	1	-1

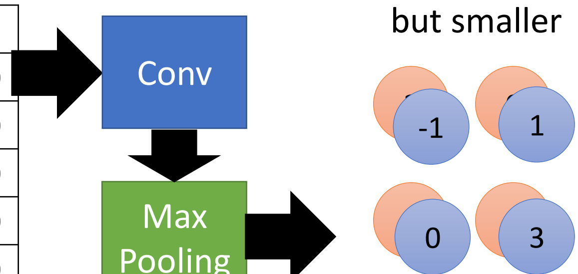


CNN – Max Pooling

1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

6 x 6 image

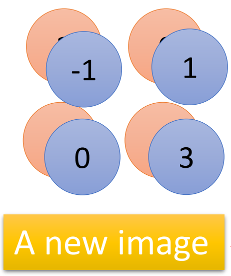
New image but smaller

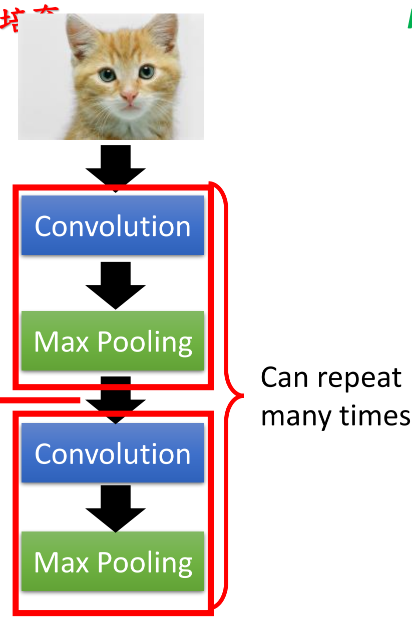


2 x 2 image

Each filter is a channel

The whole CNN

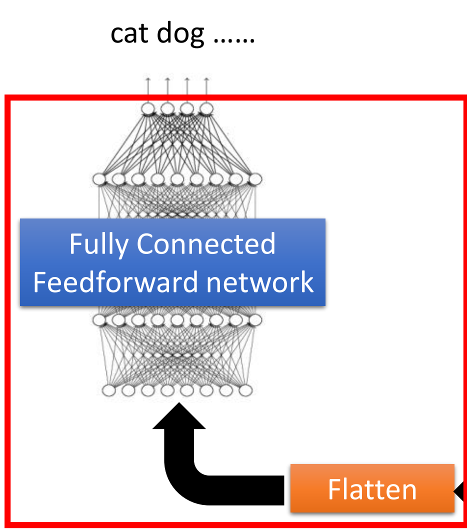


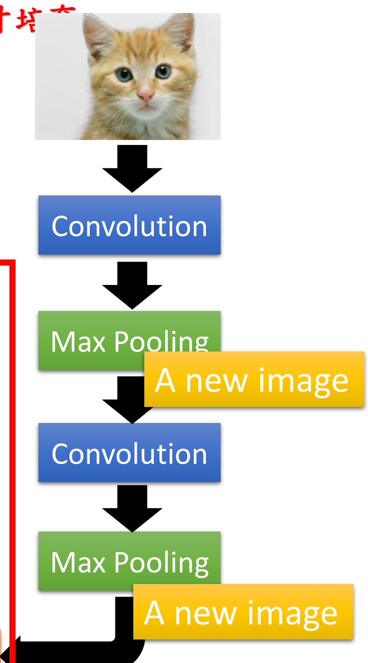


Smaller than the original image

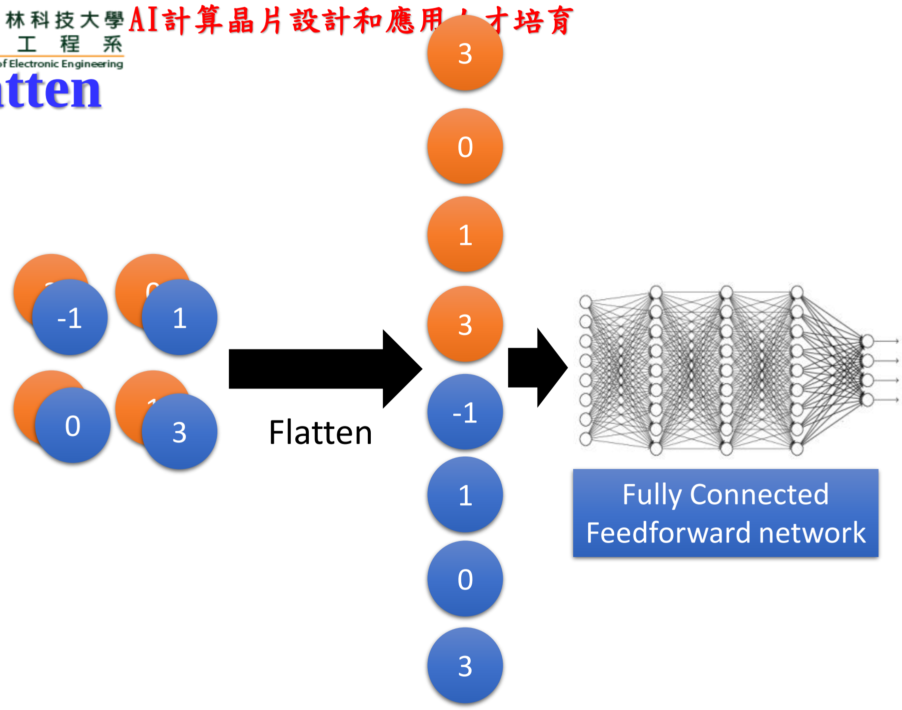
The number of the channel is the number of filters

The whole CNN





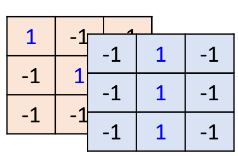
Flatten



CNN in Keras

Only modified the ***network structure*** and ***input format (vector -> 3-D tensor)***

```
model2.add( Convolution2D( 25,3,3,
            input_shape=(28,28,1)) )
```

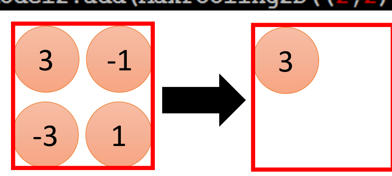


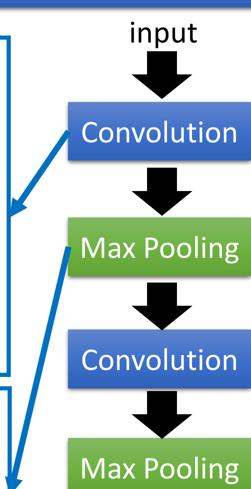
...... There are 25 3x3 filters.

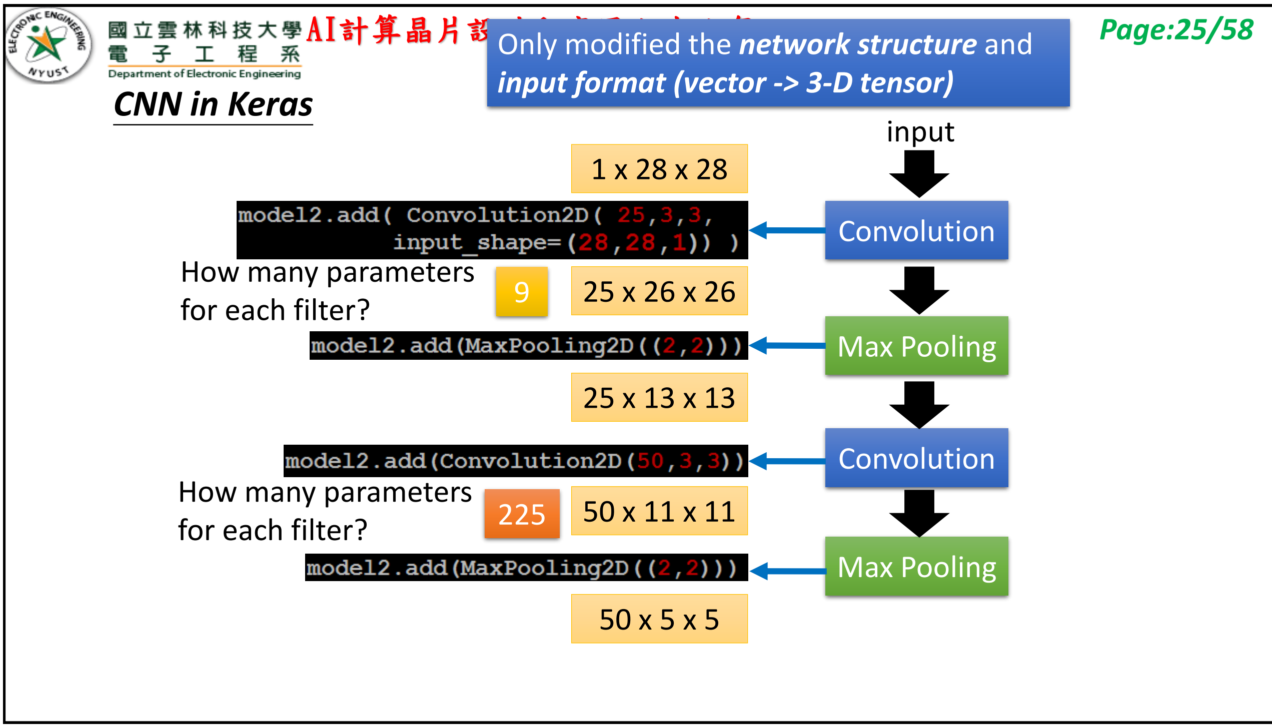
Input_shape = (28 , 28 , 1)

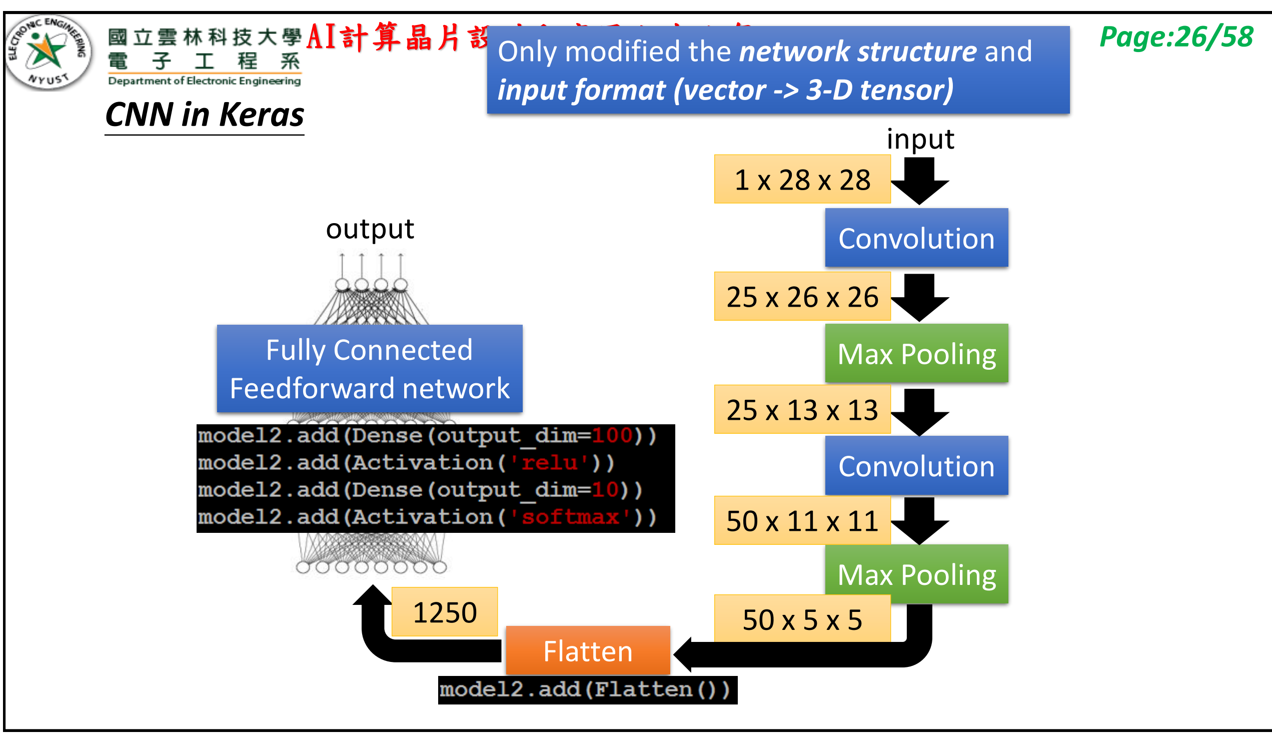
28 x 28 pixels   1: black/white, 3: RGB

```
model2.add(MaxPooling2D((2,2)))
```






國立雲林科技大學
電子工程系
Department of Electronic Engineering
AI計算晶片
Page:25/58
CNN in Keras
Only modified the network structure and input format (vector -> 3-D tensor)
input
1 x 28 x 28
model2.add(Convolution2D(25,3,3,
input_shape=(28,28,1)))
Convolution
How many parameters for each filter?
9
25 x 26 x 26
model2.add(MaxPooling2D((2,2)))
Max Pooling
25 x 13 x 13
model2.add(Convolution2D(50,3,3))
Convolution
How many parameters for each filter?
225
50 x 11 x 11
model2.add(MaxPooling2D((2,2)))
Max Pooling
50 x 5 x 5

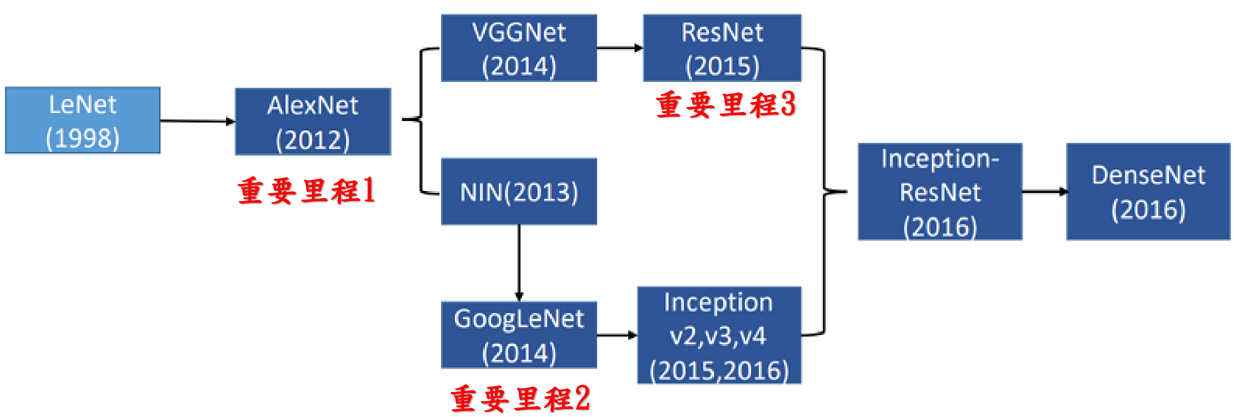

國立雲林科技大學
電子工程系
Department of Electronic Engineering
AI計算晶片
Page:26/58
CNN in Keras
Only modified the network structure and input format (vector -> 3-D tensor)
input
1 x 28 x 28
Convolution
25 x 26 x 26
Max Pooling
25 x 13 x 13
Convolution
50 x 11 x 11
Max Pooling
50 x 5 x 5
output
Fully Connected Feedforward network
model2.add(Dense(output_dim=100))
model2.add(Activation('relu'))
model2.add(Dense(output_dim=10))
model2.add(Activation('softmax'))
1250
Flatten
model2.add(Flatten())

Outline

- CNN
- **LeNet**

國立雲林科技大學 電子工程系 Department of Electronic Engineering AI計算晶片設計和應用人才培育

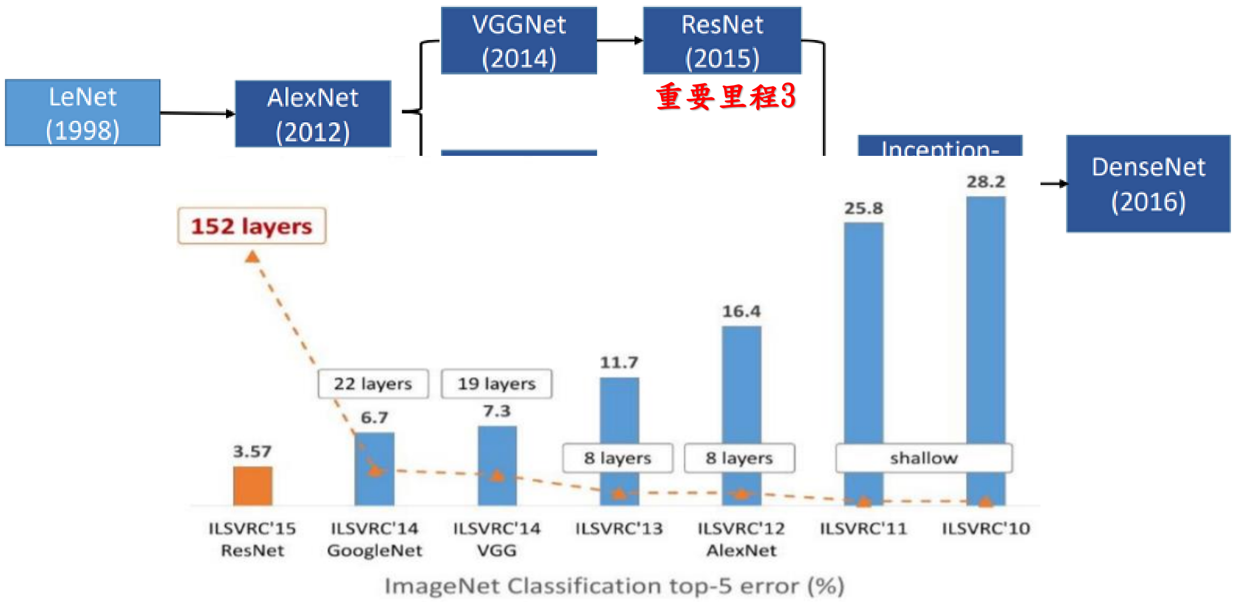
Introduction to CNN



國立雲林科技大學 電子工程系 Department of Electronic Engineering AI計算晶片設計和應用人才培育

Page:29/58

Introduction to CNN



國立雲林科技大學 電子工程系 Department of Electronic Engineering AI計算晶片設計和應用人才培育

Page:30/58

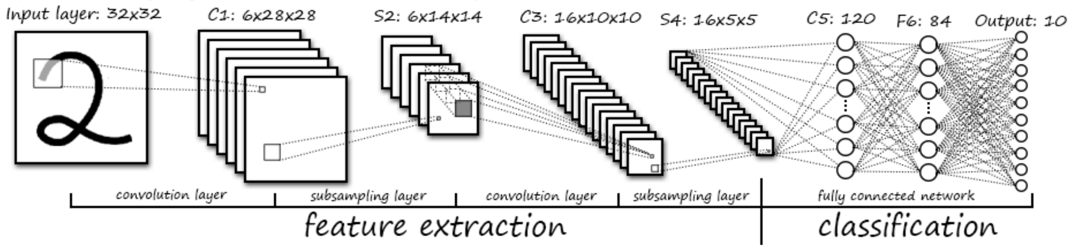
Deep Convolutional Networks

- Compared to standard feed forward neural networks with similarly-sized layers,
  - CNNs have much fewer connections and parameters
  - and so they are easier to train,
  - while their theoretically-best performance is likely to be only slightly worse.

LeNet 5

Y. LeCun, L. Bottou, Y. Bengio and P. Haffner: **Gradient-Based Learning Applied to Document Recognition**, *Proceedings of the IEEE, 86(11):2278-2324, November* ***1998***

LeNet5, LeCun1998



- **Input**:32x32 pixel image. Largest character is 20x20
  (All important info should be in the center of the receptive fields of the highest level feature detectors)
- **Cx**: Convolutional layer (C1, C3, C5)
- **Sx**: Subsample layer (S2, S4)
- **Fx**: Fully connected layer (F6)
- Black and White pixel values are normalized:
  E.g. White = -0.1, Black =1.175 (Mean of pixels = 0, Std of pixels =1)

2-Dimensional Convolution

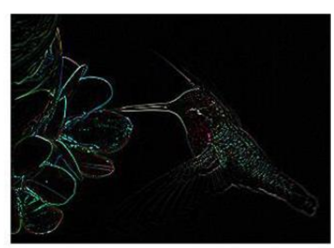
$$f[x,y] * g[x,y] = \sum_{n_1=-\infty}^{\infty} \sum_{n_2=-\infty}^{\infty} f[n_1,n_2] \cdot g[x-n_1, y-n_2]$$

Original

Filter (=kernel)

0.00	0.00	0.00	0.00	0.00
0.00	0.00	-2.00	0.00	0.00
0.00	-2.00	8.00	-2.00	0.00
0.00	0.00	-2.00	0.00	0.00
0.00	0.00	0.00	0.00	0.00

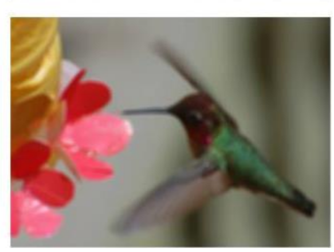
sum(filter) = 0



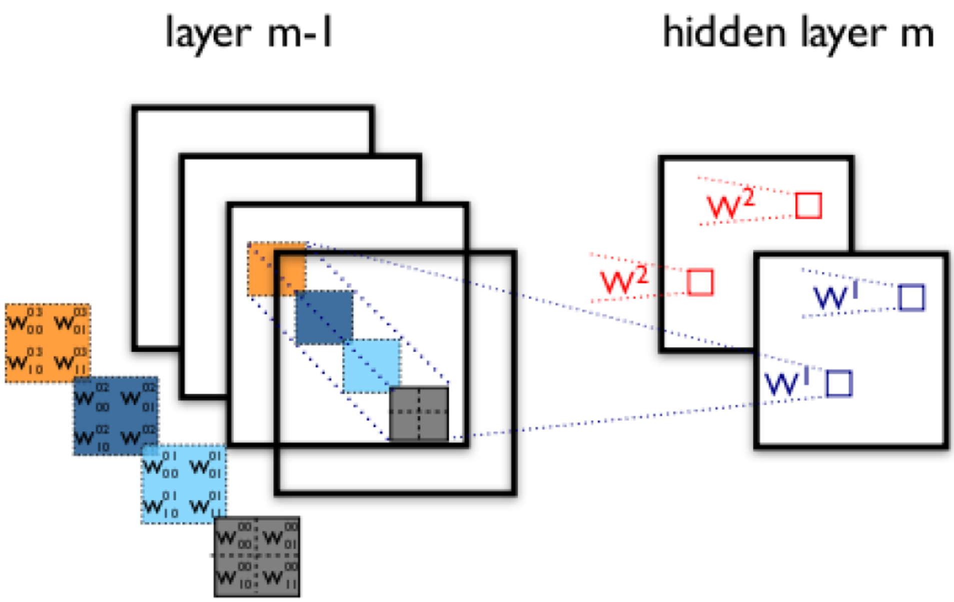
0.04	0.04	0.04	0.04	0.04
0.04	0.04	0.04	0.04	0.04
0.04	0.04	0.04	0.04	0.04
0.04	0.04	0.04	0.04	0.04
0.04	0.04	0.04	0.04	0.04

sum(filter) = 1

1/25 = 0.04

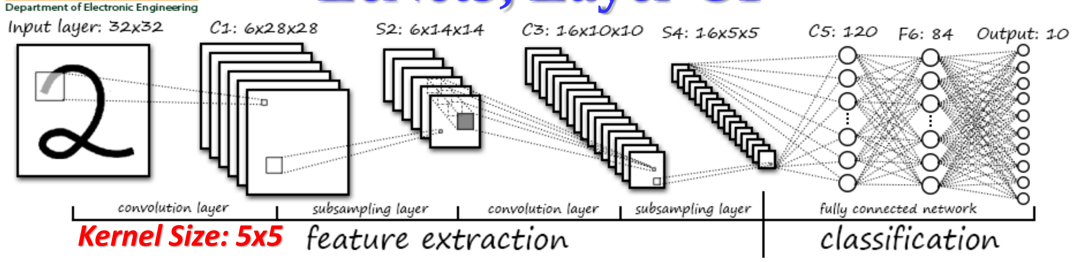


Convolutional Layer



國立雲林科技大學 電子工程系 Department of Electronic Engineering AI計算晶片設計和應用人才培育

LeNet5, Layer C1



Kernel Size: 5x5

C1: Convolutional layer with 6 feature maps of size 28x28.

Each unit of C1 has a 5x5 receptive field in the input layer. $C1^k,\ k = 1, \ldots, 6.$

- Topological structure
- Sparse connections
- Shared weights

$$C1^k{}_{ij} = \tanh([W^k * x]_{ij} + b_k).$$

$$W_k \in \mathbb{R}^{5\times5}, b_k \in \mathbb{R}, x \in \mathbb{R}^{32\times32}$$

$$k=1,\ldots,6 \quad C1^k \in \mathbb{R}^{28\times28}.$$

(5*5+1)*6 = 156 parameters to learn

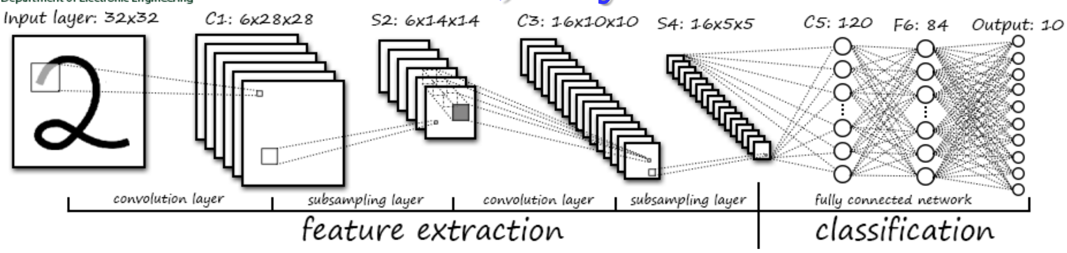
Connections: (5*5+1)*28*28*6=122,304

If it was fully connected, we had (32*32+1)*(28*28)*6 parameters = 4,821,600 connections

國立雲林科技大學 電子工程系 Department of Electronic Engineering AI計算晶片設計和應用人才培育

LeNet5, Layer S2

Page:35/58



- S2: Subsampling layer with 6 feature maps of size 14x14
- 2x2 non overlapping receptive fields in C1

$$w_1^k, w_2^k \in \mathbb{R}$$

$$S2_{ij}^k = \tanh(w_1^k \sum_{s,t=0}^{1} C1_{2i-s,2j-t}^k + w_2^k).$$

$$k=1,...,6,\ i,j = 1,\dots,14$$

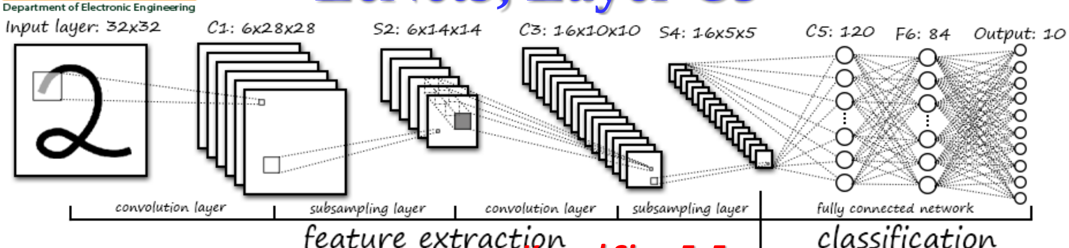
$$S2^k \in \mathbb{R}^{14\times 14}.$$

- Layer S2: 6*2=12 trainable parameters.
- Connections: 14*14*(2*2+1)*6=5880

國立雲林科技大學 電子工程系 Department of Electronic Engineering AI計算晶片設計和應用人才培育

LeNet5, Layer C3

Page:36/58



Kernel Size: 5x5

- C3: Convolutional layer with 16 feature maps of size 10x10
- Each unit in C3 is connected to several! 5x5 receptive fields at identical locations in S2

Layer C3: 1516 trainable parameters.

=(3*5*5+1)*6+(4*5*5+1)*9+(6*5*5+1)

Connections: 151600

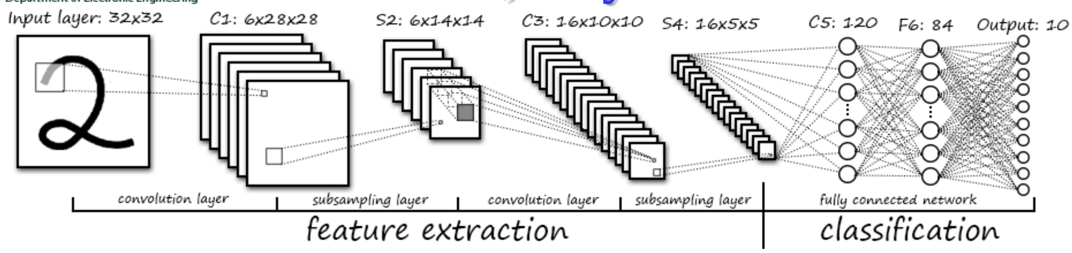
(3*5*5+1)*6*10*10+(4*5*5+1)*9*10*10+(6*5*5+1)*10*10

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	X				X	X	X			X	X	X	X		X	X
1	X	X				X	X	X			X	X	X	X		X
2	X	X	X				X	X	X			X		X	X	X
3		X	X	X			X	X	X	X			X		X	X
4			X	X	X			X	X	X	X		X	X		X
5				X	X	X			X	X	X	X		X	X	X

TABLE I

EACH COLUMN INDICATES WHICH FEATURE MAP IN S2 ARE COMBINED BY THE UNITS IN A PARTICULAR FEATURE MAP OF C3.

LeNet5, Layer S4



- S4: Subsampling layer with 16 feature maps of size 5x5
- Each unit in S4 is connected to the corresponding 2x2 receptive field at C3

$$w_1^k, w_2^k \in \mathbb{R}$$

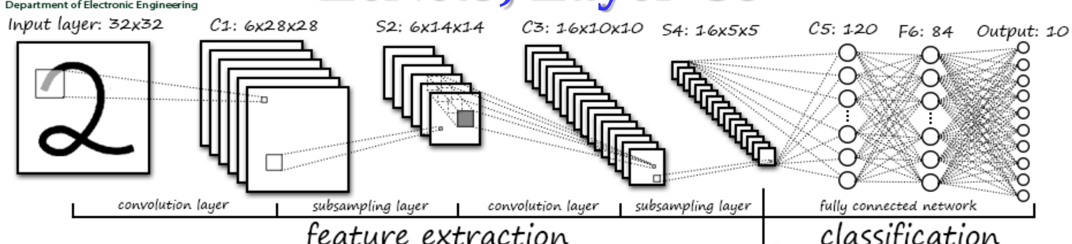
$$S4_{ij}^k = \tanh(w_1^k \sum_{s,t=0}^{1} C1_{2i-s,2j-t}^k + w_2^k).$$

$$k=1,\ldots,16,\ i,j = 1,\ldots,5$$

$$S4^k \in \mathbb{R}^{5\times 5}.$$

Layer S4: **16*2=32** trainable parameters.
Connections: 5*5*(2*2+1)*16=2000

LeNet5, Layer C5

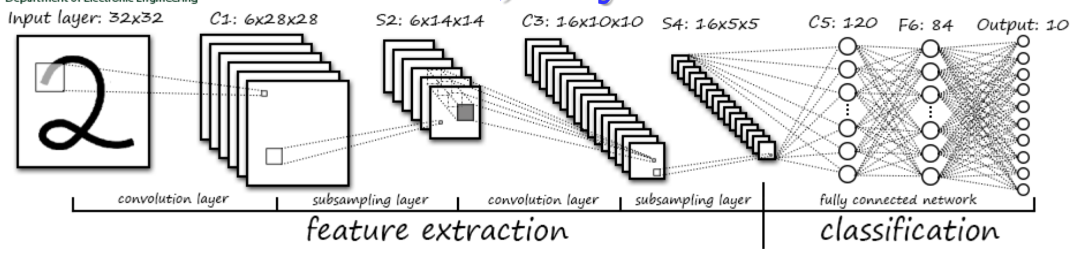


Kernel Size: 5x5

- C5: Convolutional layer with 120 feature maps of size 1x1
- Each unit in C5 is connected to all 16 5x5 receptive fields in S4

Layer C5: 120*(16*25+1) = 48120 trainable parameters and connections (Fully connected)

LeNet5, Layer C5



- **Layer F6**: 84 fully connected units. 84*(120+1)=10164 trainable parameters and connections.
- **Output layer**: 10RBF (One for each digit)

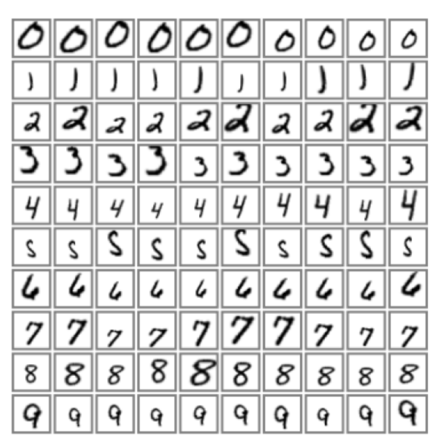
$$y_i = \sum_{j=1}^{84}(x_j - w_{ij})^2,\ i = 1, \ldots, 10.$$

From F6

84=7x12, stylized image.
84 parameters, 84*10 connections
Weight update: Backpropagation

MINIST Dataset

60,000 original datasets
Test error: 0.95%



540,000 artificial distortions
+ 60,000 original
Test error: 0.8%

國立雲林科技大學 電 子 工 程 系 Department of Electronic Engineering AI計算晶片設計和應用人才培育

Page:41/58

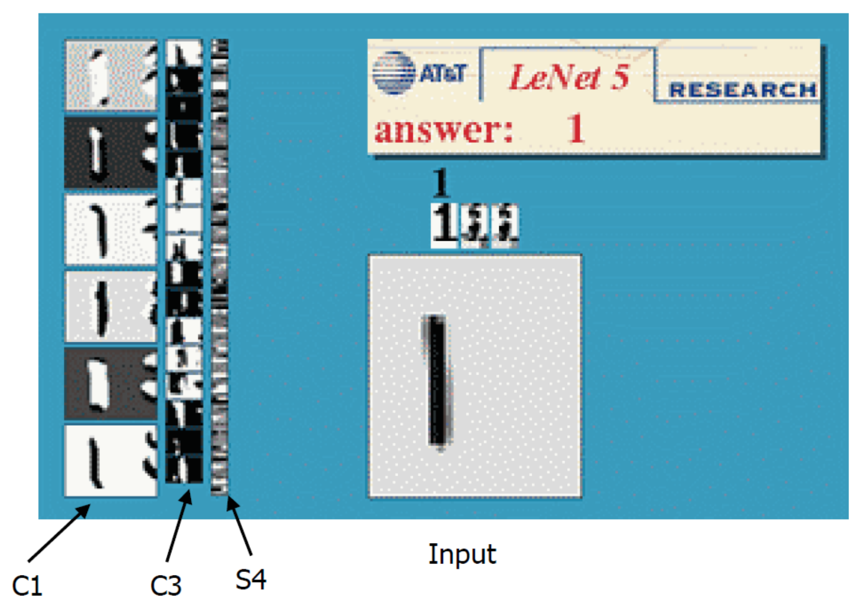
Misclassified examples

- True label -> Predicted label

國立雲林科技大學 電 子 工 程 系 Department of Electronic Engineering AI計算晶片設計和應用人才培育

Page:42/58

LeNet5 in Action



國立雲林科技大學 電子工程系 Department of Electronic Engineering AI計算晶片設計和應用人才培育

LeNet5, Shift invariance

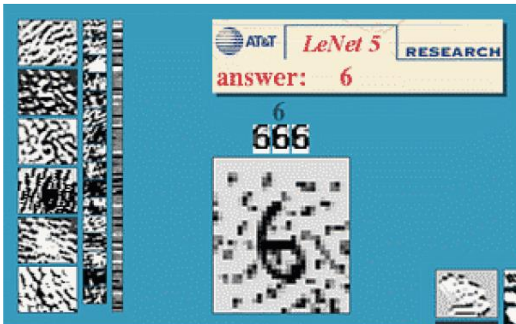
Page:43/58

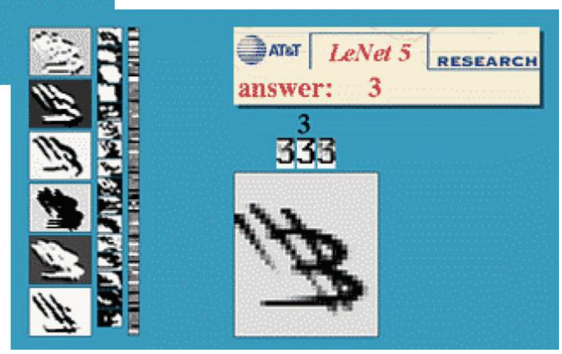
國立雲林科技大學 電子工程系 Department of Electronic Engineering AI計算晶片設計和應用人才培育

LeNet5, Rotation invariance

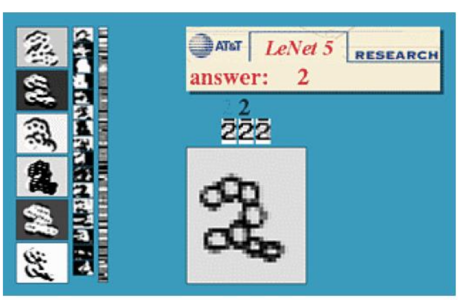
Page:44/58

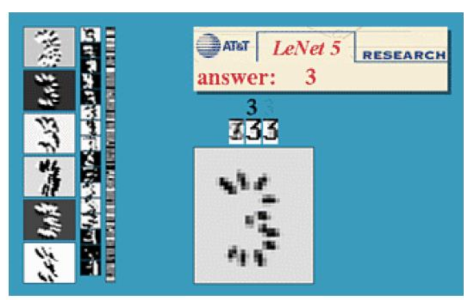
LeNet5, Nosie resistance

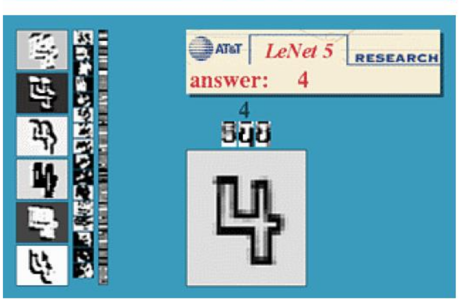


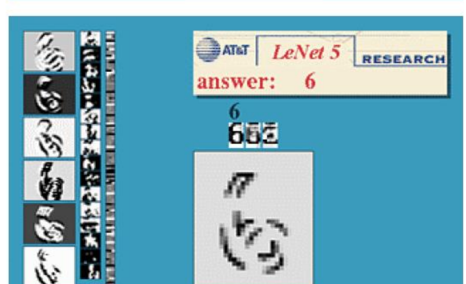


LeNet5, Unusual Patterns

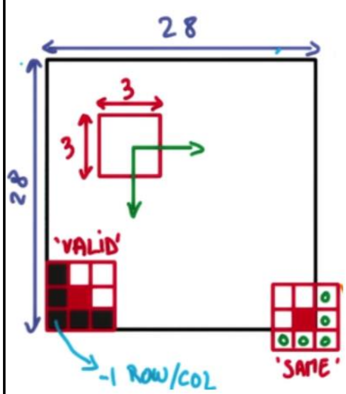






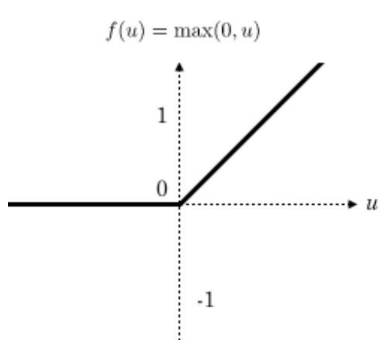


Platform: Tensorflow



- **tf.nn.conv2d**(input, filter, strides, padding, use_cudnn_on_gpu=None, name=None)
- **input**: 輸入影像,格式為[batch，長，寬，通道數]
- **filter**:卷積核，其格式為[長，寬，輸入通道數，輸出通道數]
- **strides** :步長，一般情況下的格式為[1，長上步長，寬上步長，1]
- **padding**: 是卷積核在邊緣處的處理方法，可以取'VALID'或者'SAME'

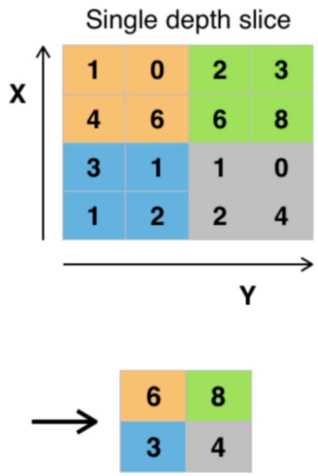
Platform: Tensorflow



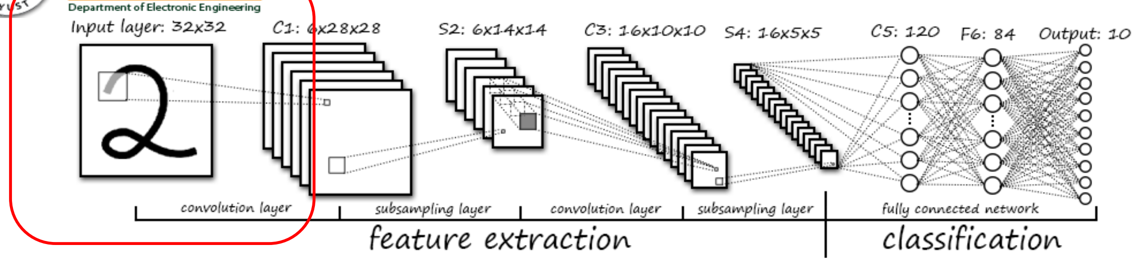
- tf.nn.relu(input_data)
- Example:

  b = tf.nn.relu(a)

Platform: Tensorflow

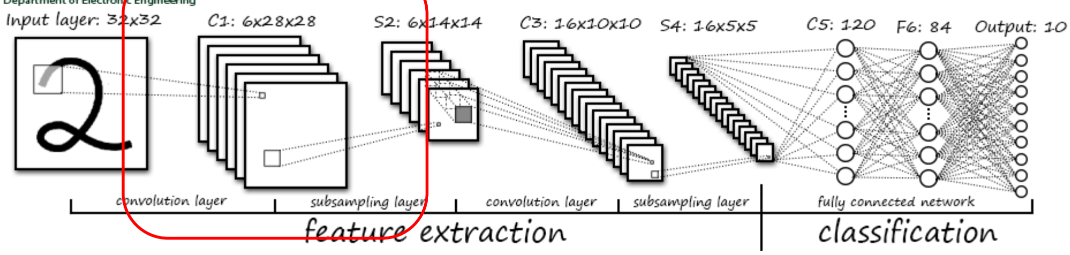


- **tf.nn.max_pool**(value, ksize, strides, padding, name=None)
- **value**：池化的輸入(input feature map)，格式[batch, height, width, channels]
- **ksize**：池化窗口的大小，格式是[1, height, width, 1]
- **strides**：類似卷積，格式是[1, stride,stride, 1]
- **padding**：類似卷積，可以取'VALID'或者'SAME'



reshape imput as [number of examples (m), weight, height, channel]

X_ = tf.reshape(X, [-1, 28, 28, 1]) # num_channel = 1 (gray image)



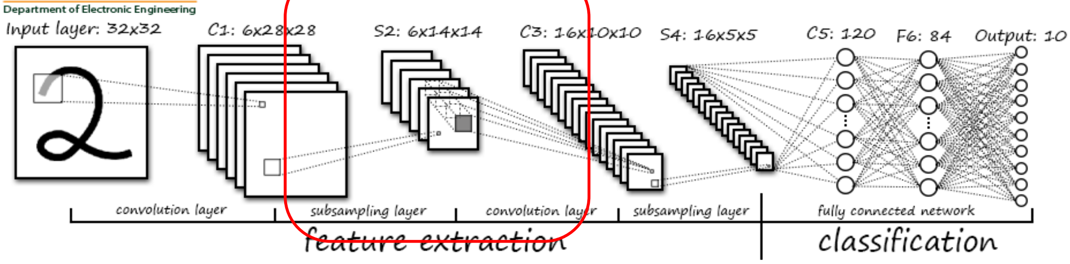
CONV1 (f = 5*5*1, n_f = 6, s = 1, p = 'same')

W_conv1 = weight_variable(shape = [5, 5, 1, 6])

b_conv1 = bias_variable(shape = [6])

shape of A_conv1 ~ [m,28,28,6]

A_conv1 = tf.nn.relu(tf.nn.conv2d(X_, W_conv1, strides = [1, 1, 1, 1], padding = 'SAME') + b_conv1)



MAXPOOL2 (f = 2*2*1, s = 2, p = 'same')

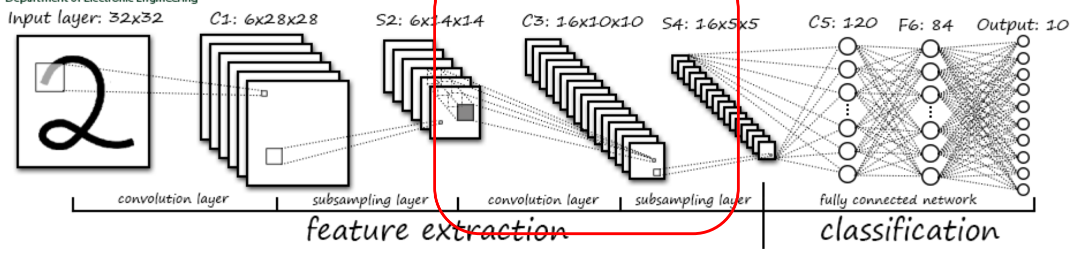
shape of A_pool1 ~ [m,14,14,6]

A_pool1 = tf.nn.max_pool(A_conv1, ksize = [1, 2, 2, 1], strides=[1, 2, 2, 1], padding = 'SAME')

國立雲林科技大學 電子工程系 Department of Electronic Engineering

AI計算晶片設計和應用人才培育

Page:53/58



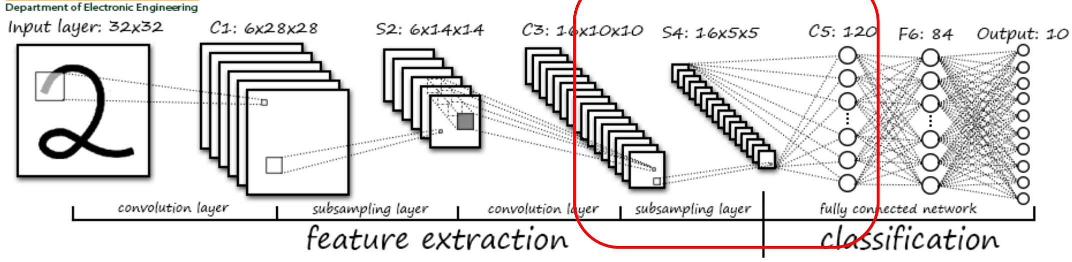
CONV3 (f = 5*5*1, n_f = 16, s = 1, p = 'same')

```
W_conv2 = weight_variable(shape = [5, 5, 6, 16])
b_conv2 = bias_variable(shape = [16])
# shape of A_conv2 ~ [m,10,10,16]
A_conv2 = tf.nn.relu(tf.nn.conv2d(A_pool1, W_conv2, strides = [1, 1, 1, 1], padding = 'VALID') + b_conv2)
```

國立雲林科技大學 電子工程系 Department of Electronic Engineering

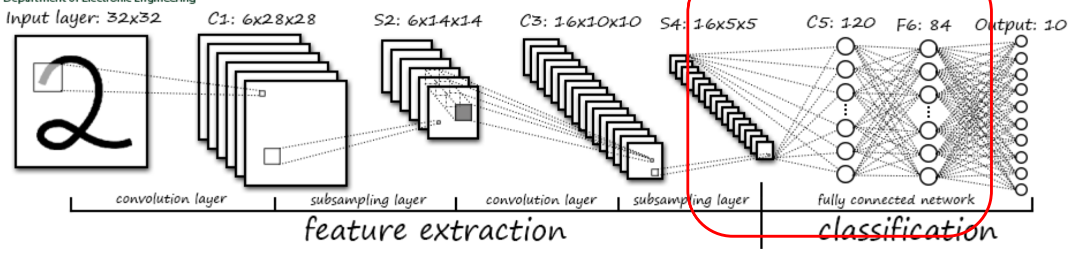
AI計算晶片設計和應用人才培育

Page:54/58

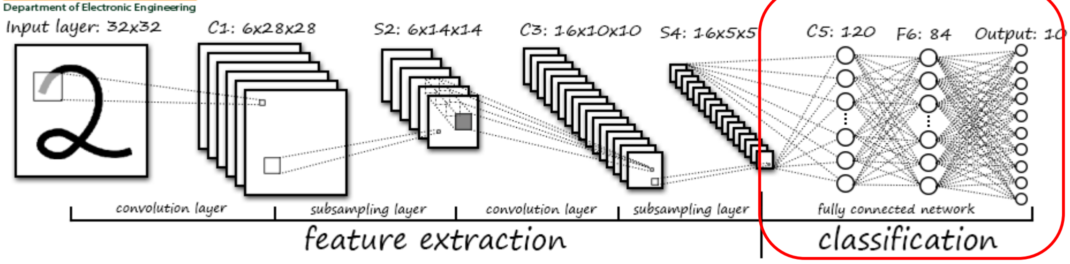


MAXPOOL4 (f = 2*2*1, s = 2, p = 'same')

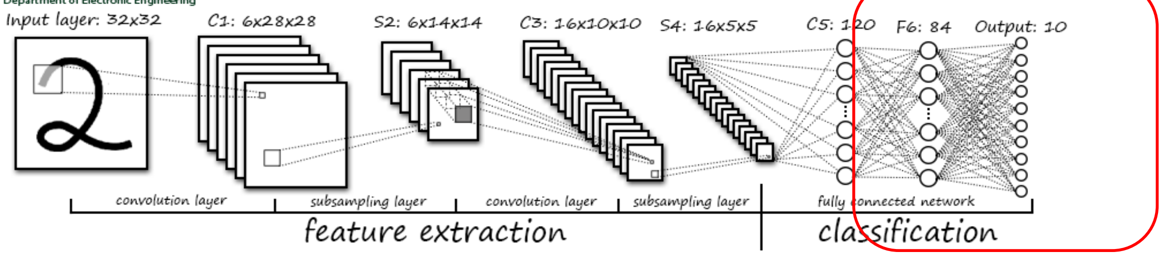
```
# shape of A_pool2~ [m,5,5,16]
A_pool2 = tf.nn.max_pool(A_conv2, ksize = [1, 2, 2, 1], strides=[1, 2, 2, 1], padding = 'SAME')
```




```
### FC5 (n = 120)
  # flatten the volumn to vector
  A_pool2_flat = tf.reshape(A_pool2, [-1, 5*5*16])
  W_fc3 = weight_variable([5*5*16, 120])
  b_fc3 = bias_variable([120])
  # shape of A_fc3 ~ [m,120]
  A_fc3 = tf.nn.relu(tf.matmul(A_pool2_flat, W_fc3) + b_fc3)
```


```
### FC6 (n = 84)
  W_fc4 = weight_variable([120, 84])
  b_fc4 = bias_variable([84])
  # shape of A_fc4 ~ [m, 84]
  A_fc4 = tf.nn.relu(tf.matmul(A_fc3, W_fc4) + b_fc4)
```




```
# Softmax (n = 10)
  W_l = weight_variable([84, 10])
  b_l = bias_variable([10])
  # shape of A_l ~ [m,10]
  A_l=tf.nn.softmax(tf.matmul(A_fc4, W_l) + b_l)
```


Reference

- 台灣大學、電機系、李宏毅, ” Machine Learning”
- CS231n: Convolutional Neural Networks for Visual Recognition, Stanford
- CS230: Deep Learning, Stanford