Week 9

السنة الخامسة – هندسة المعلوماتية / الذكاء الصنعي

مقرر التعلم التلقائي

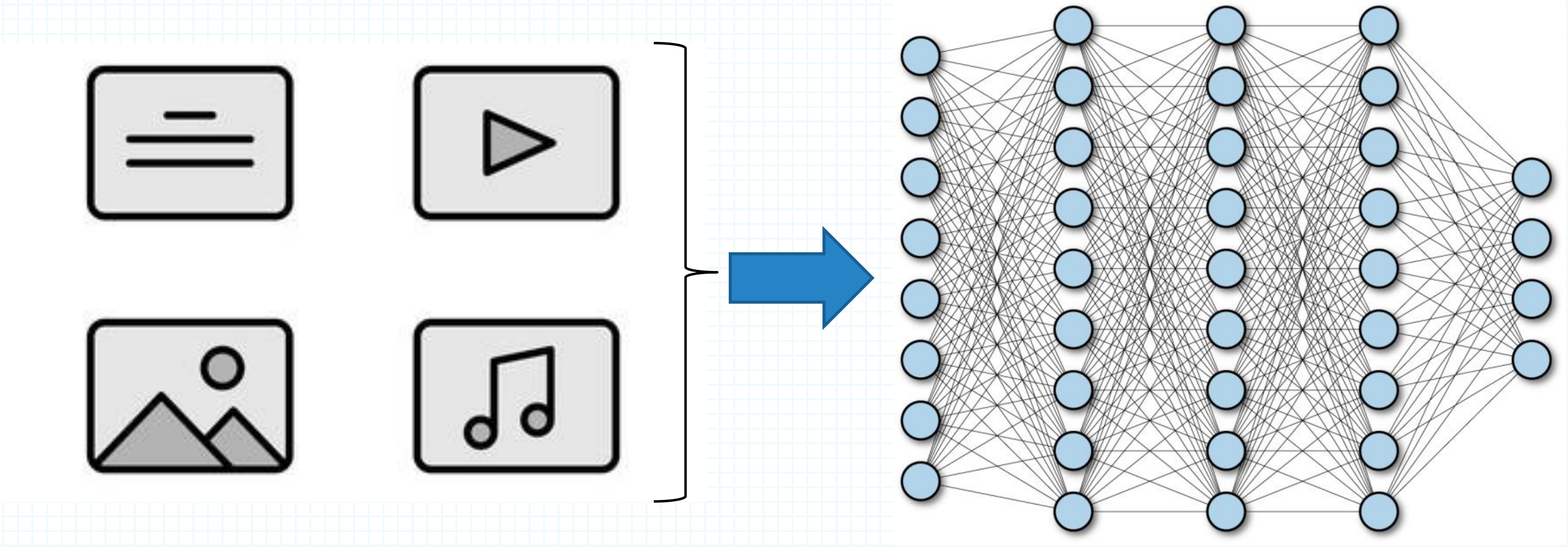
Well Known DL Architecture
CNN, RNN, LSTM

د. رياض سنبل

Access Course Materials

Large networks

- What kind of neural networks can be used for large or variable length input vectors (e.g., time series)?
- Common networks:
  - CNN, RNN, etc



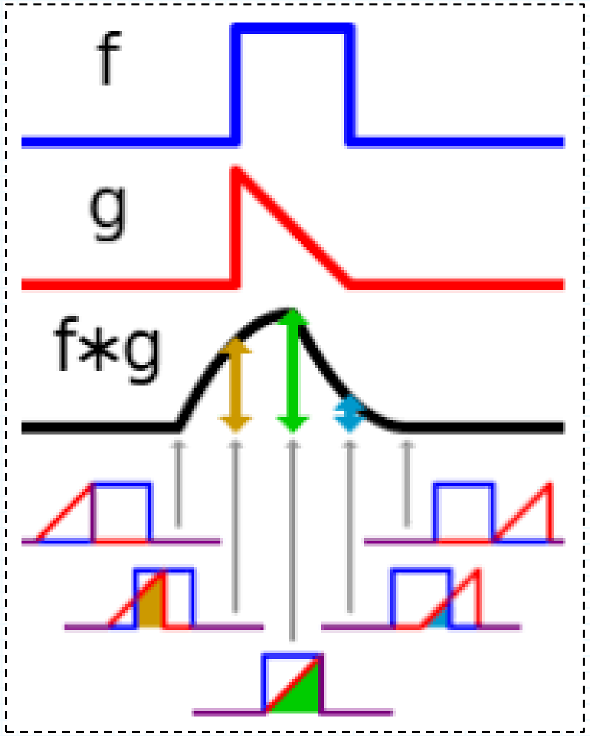
الشبكة العصبونية التلافيفية
Convolutional Neural Network (CNN)

Convolution

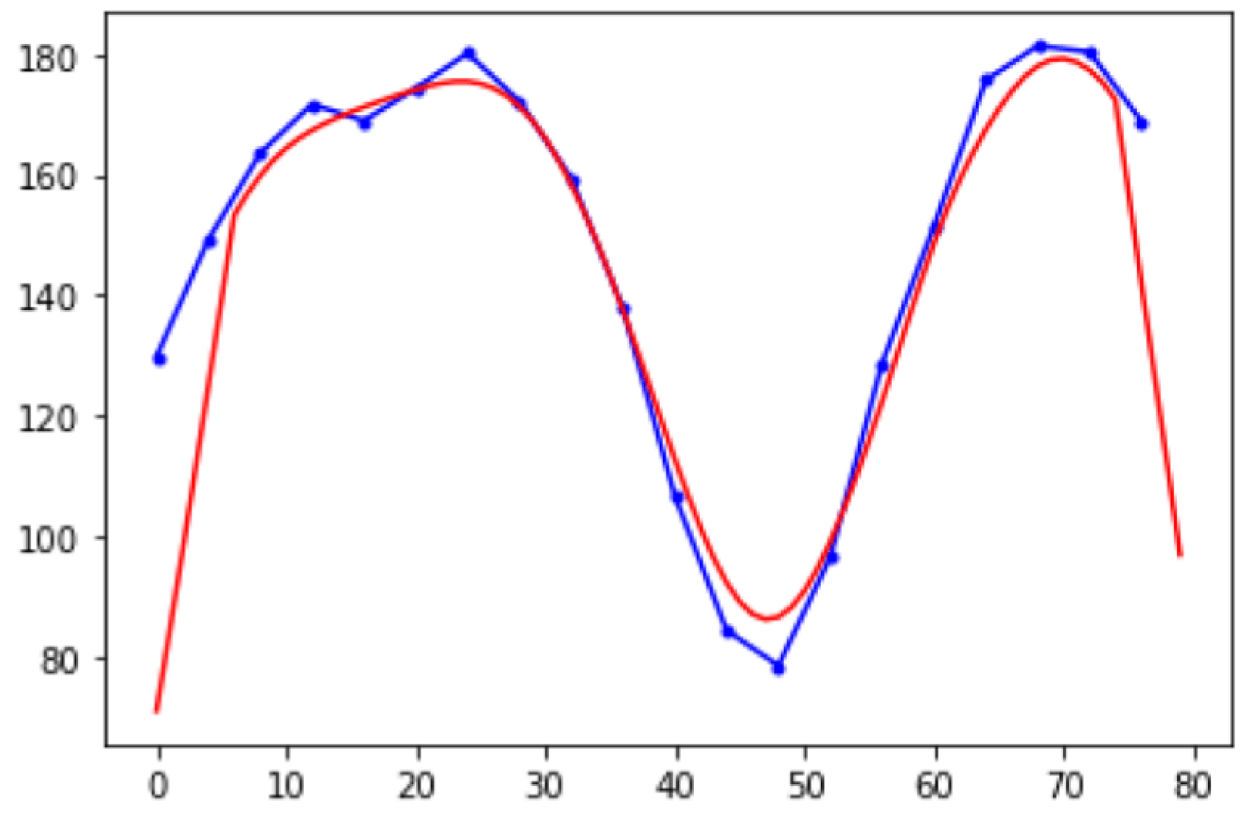
- **Convolution:** mathematical operation on two functions $x()$ and $w()$ that produces a third function $y()$ that can be viewed as a modified version of one of the original functions $x()$

$$(f * g)(t) \stackrel{\text{def}}{=} \int_{-\infty}^{\infty} f(\tau)g(t-\tau)\, d\tau$$

To convolve a kernel with an input signal: flip the signal, move to the desired time, and accumulate every interaction with the kernel



Example Smoothing



Discrete convolution

- Discrete convolution

$$y(i) = \sum_{t=-\infty}^{\infty} x(t)w(i-t)$$

- Multidimensional convolution

$$y(i,j) = \sum_{t_1=-\infty}^{\infty} \sum_{t_2=-\infty}^{\infty} x(t_1,t_2)w(i-t_1,j-t_2)$$

Example: Edge Detection

- Detect vertical edges in a grey scale image.

$$y(i,j) = x(i,j) - x(i-1,j)$$

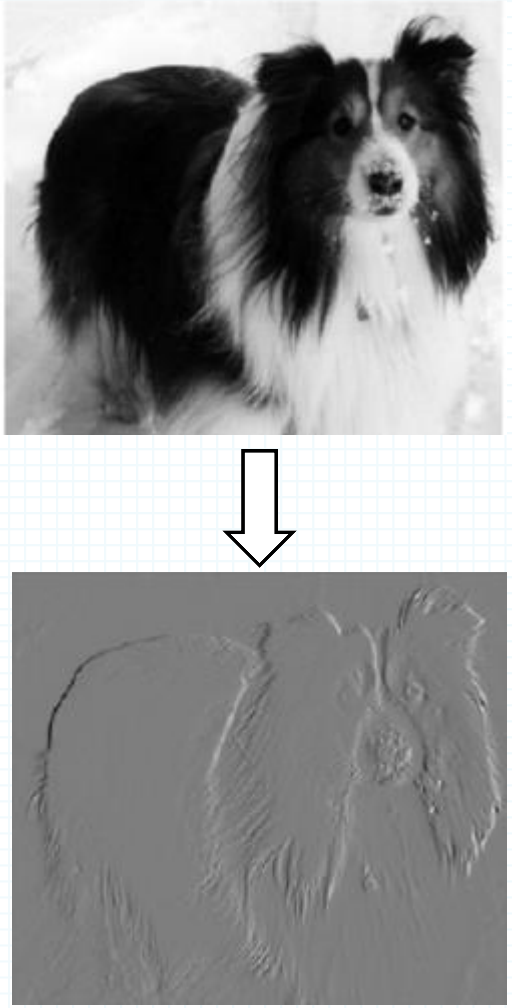
- This subtracts the pixel value **above** from the current pixel, capturing vertical changes (i.e., vertical edge)

$$w(i-t_1, j-t_2) = \begin{cases} 1 & t_1 = i, t_2 = j \\ -1 & t_1 = i-1, t_2 = j \\ 0 & \text{otherwise} \end{cases}$$

i.e. $\begin{bmatrix} -1 \\ 1 \end{bmatrix}$

- Hence:

$$y(i,j) = \sum_{t_1=-\infty}^{\infty} \sum_{t_2=-\infty}^{\infty} x(t_1,t_2) w(i-t_1, j-t_2)$$



Convolutions for feature extraction

- In neural networks:
  - A **convolution** denotes the linear combination of a **subset of units** based on a **specific pattern of weights.**

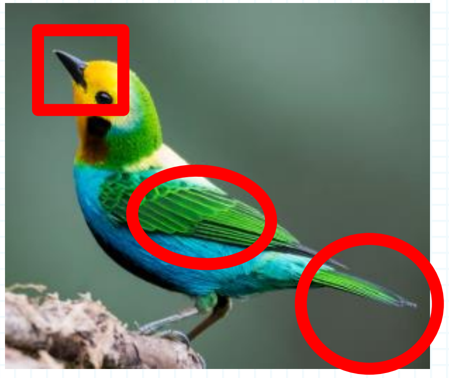
$$a_j = \sum_i w_{ji} z_i$$

  - Convolutions are often combined with an activation function to produce a feature

$$z_j = h(a_j) = h\left(\sum_i w_{ji} z_i\right)$$

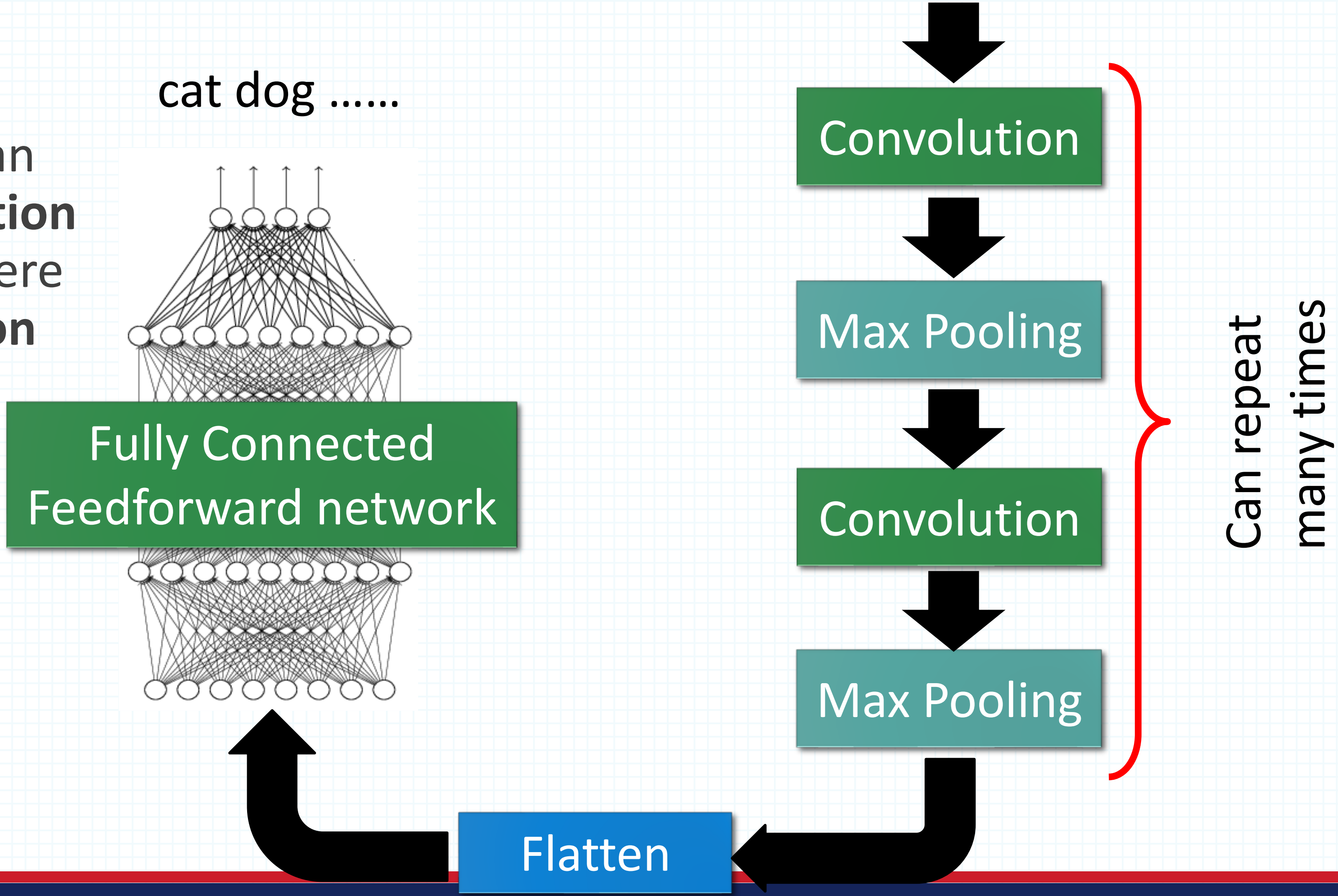
Convolutions for feature extraction

The same patterns appear in different regions.



Convolution Neural Network

- A **convolutional neural network** refers to any network that includes an **alternation of convolution and pooling layers**, where **some of the convolution weights are shared.**
- Architecture:



cat dog ......

Fully Connected Feedforward network

Convolution

Max Pooling

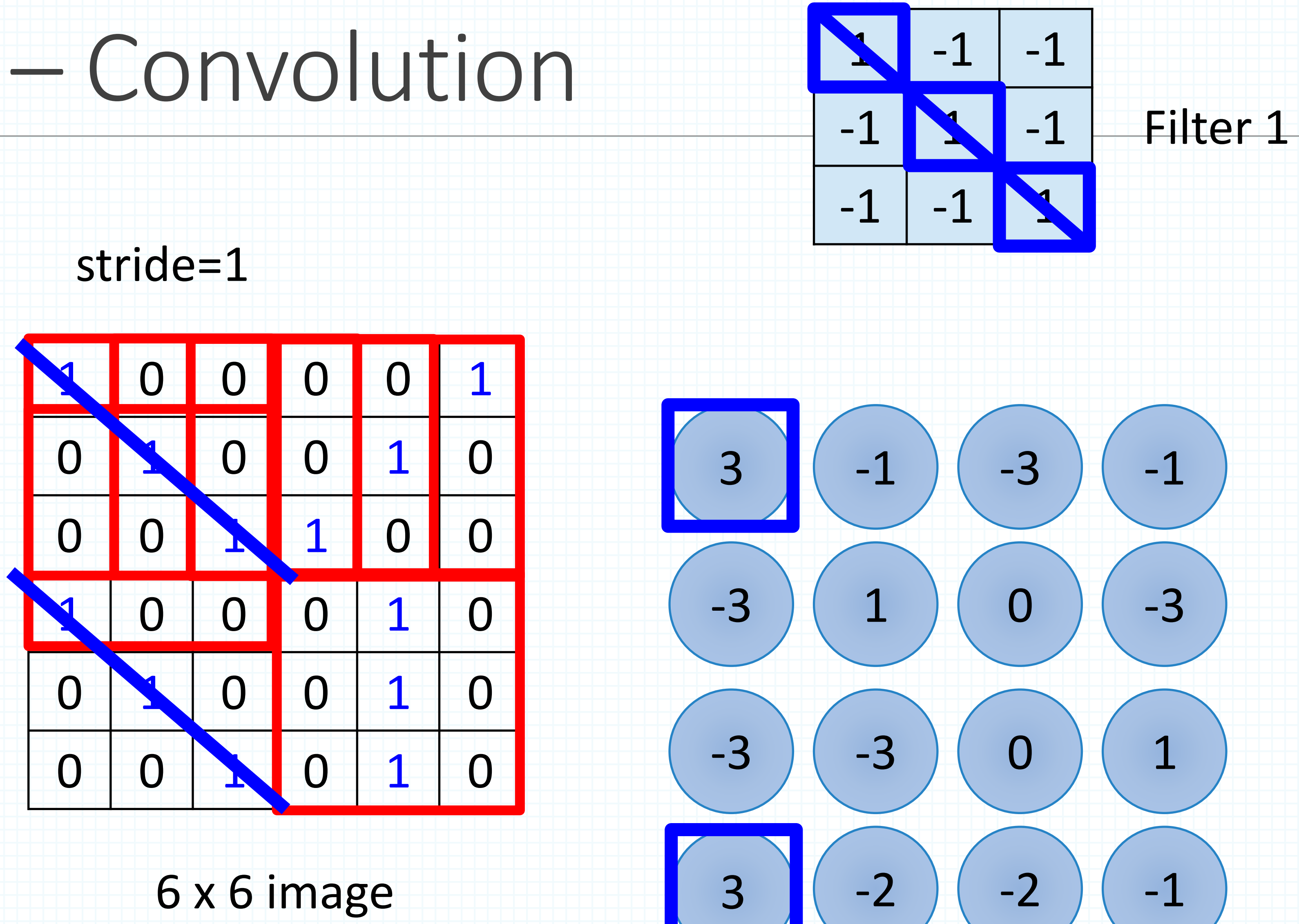
Convolution

Max Pooling

Can repeat many times

Flatten

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

CNN – Convolution

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

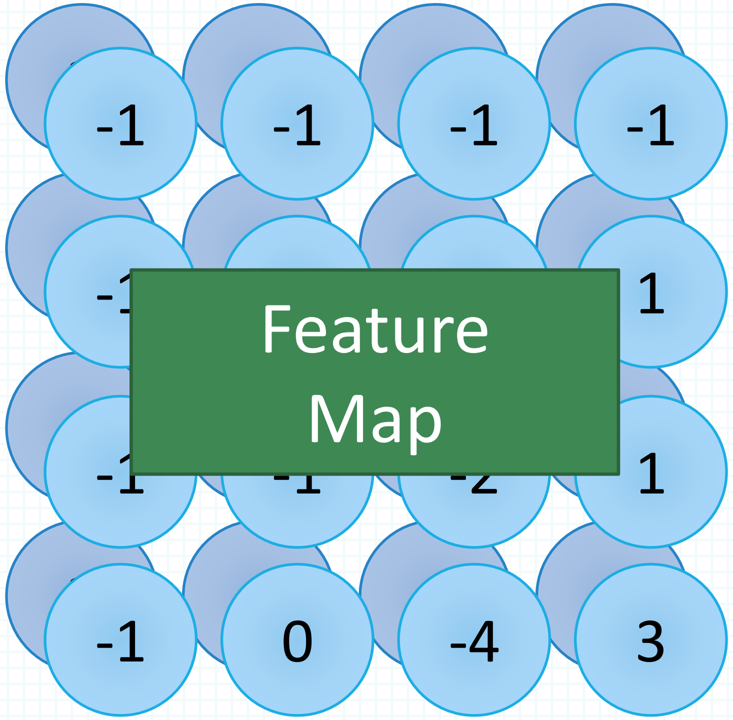
Filter 2

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

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

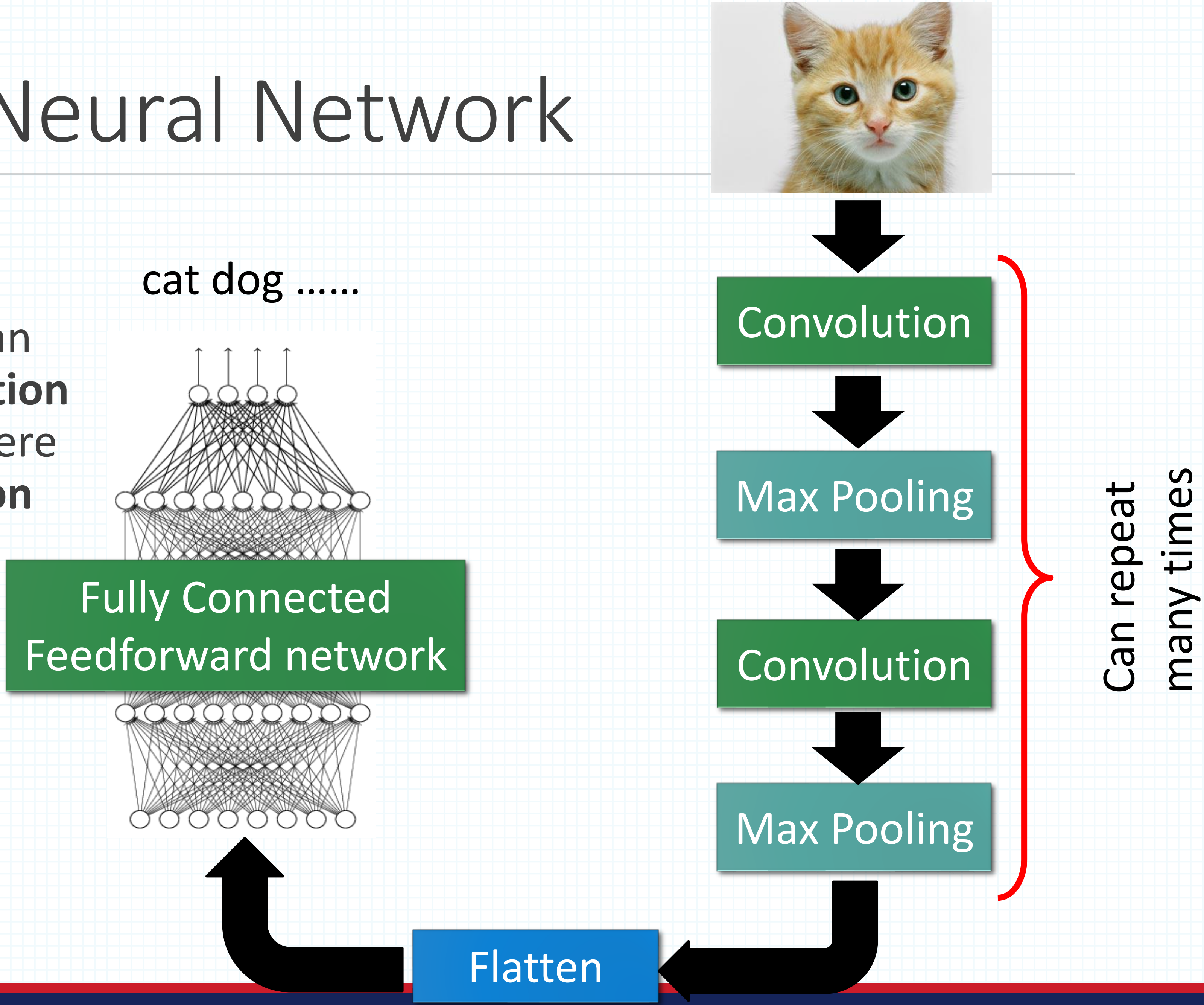
Matrix

⋮

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

Convolution Neural Network

- A **convolutional neural network** refers to any network that includes an **alternation of convolution and pooling layers**, where **some of the convolution weights are shared.**
- Architecture:



cat dog ……

Fully Connected Feedforward network

Convolution

Max Pooling

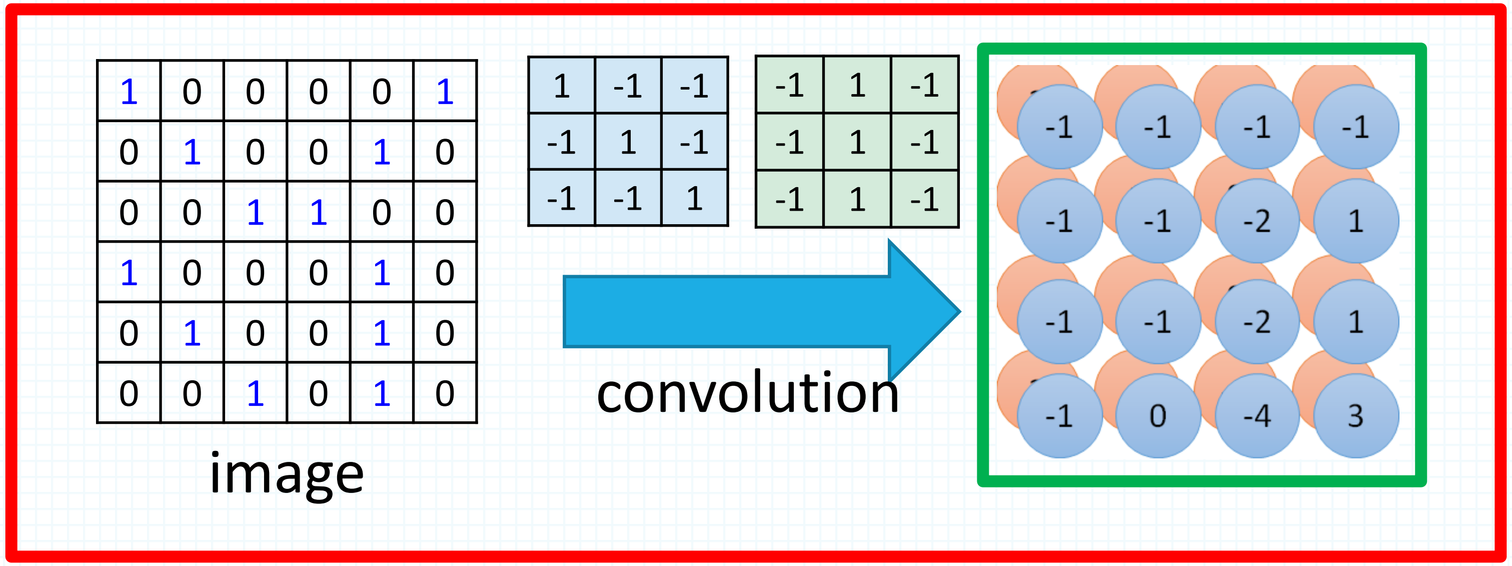
Convolution

Max Pooling

Can repeat many times

Flatten

Convolution v.s. Fully Connected



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

image

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

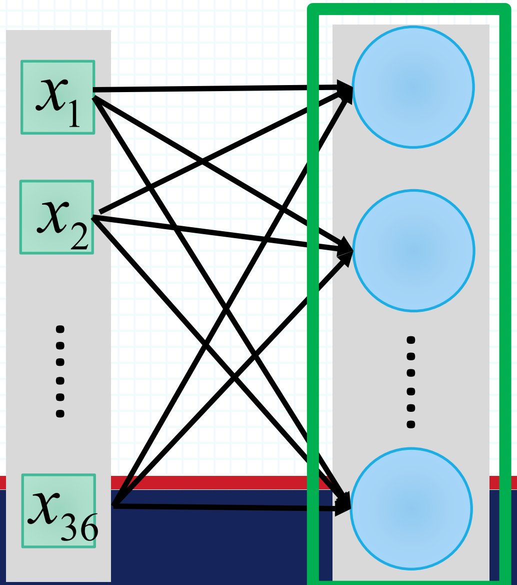
-1	1	-1
-1	1	-1
-1	1	-1

convolution

-1	-1	-1	-1
-1	-1	-2	1
-1	-1	-2	1
-1	0	-4	3

Fully-connected

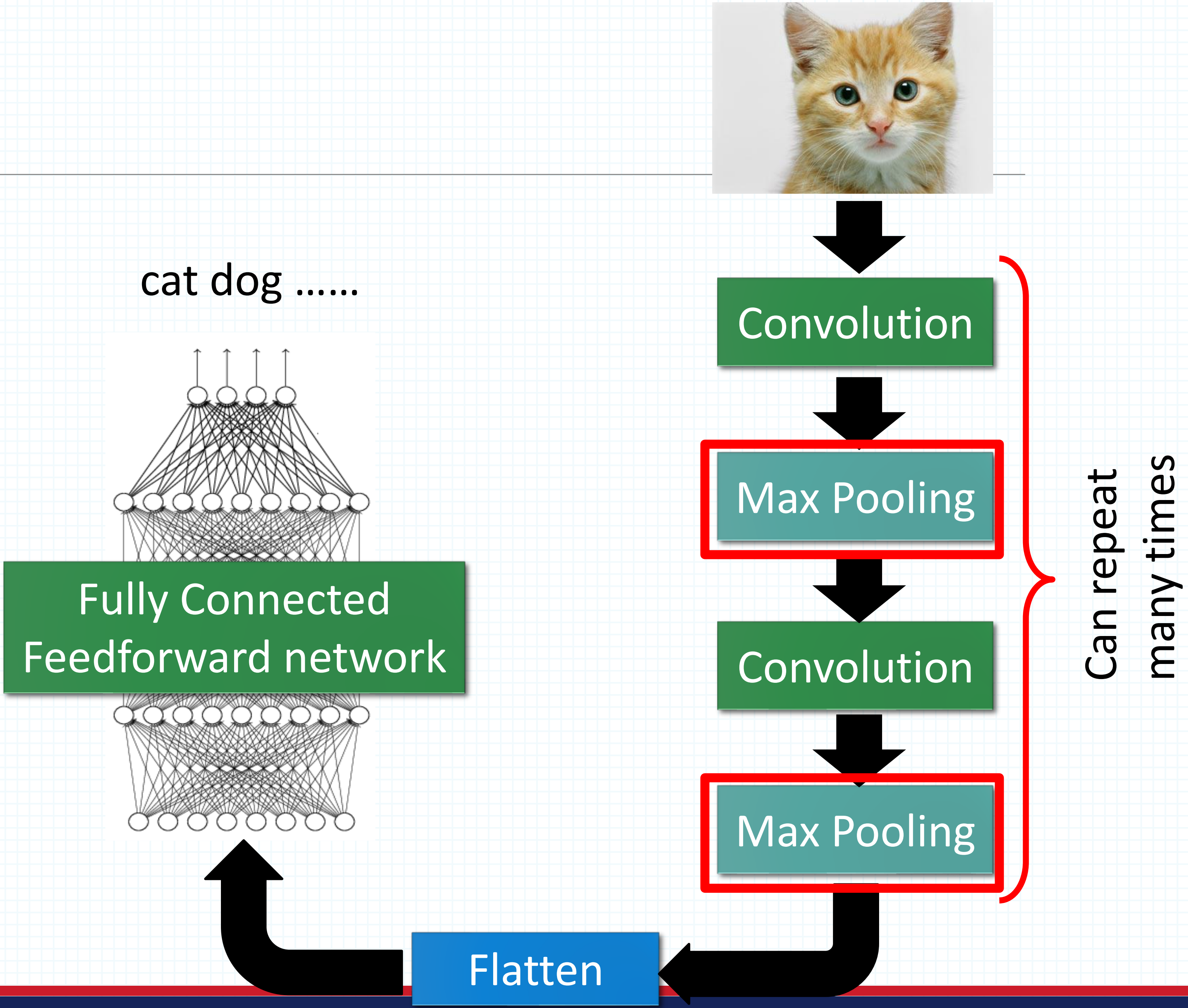
1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0



- Sparse interactions: Fewer connections
- Parameter sharing: Fewer weights
- Handle inputs of varying length

Pooling

- Pooling: commutative mathematical operation that combines several units
- Examples:
  - max, sum, product, average, Euclidean norm, etc.
- Commutative property (order does not matter):
  - max a, b = max(b, a)



cat dog ……

Convolution

Max Pooling

Convolution

Max Pooling

Can repeat many times

Flatten

Fully Connected Feedforward network

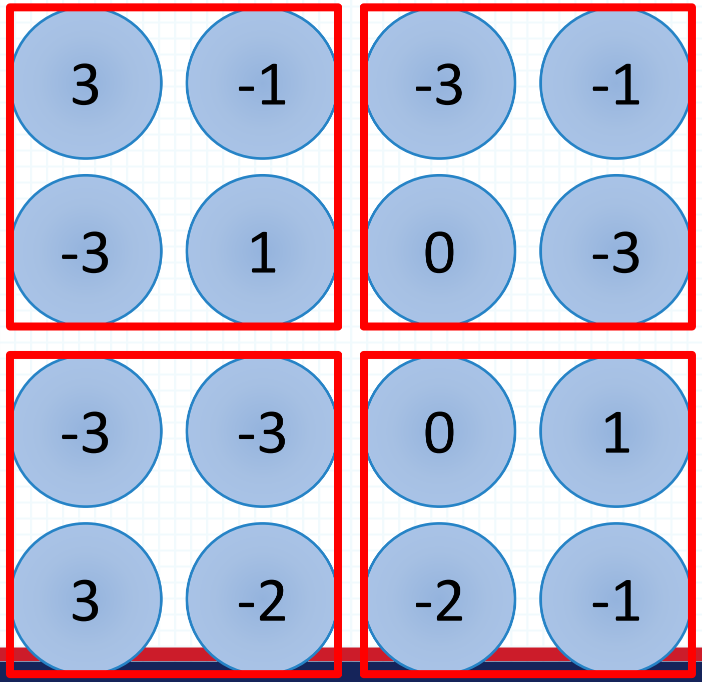
CNN – Max Pooling

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

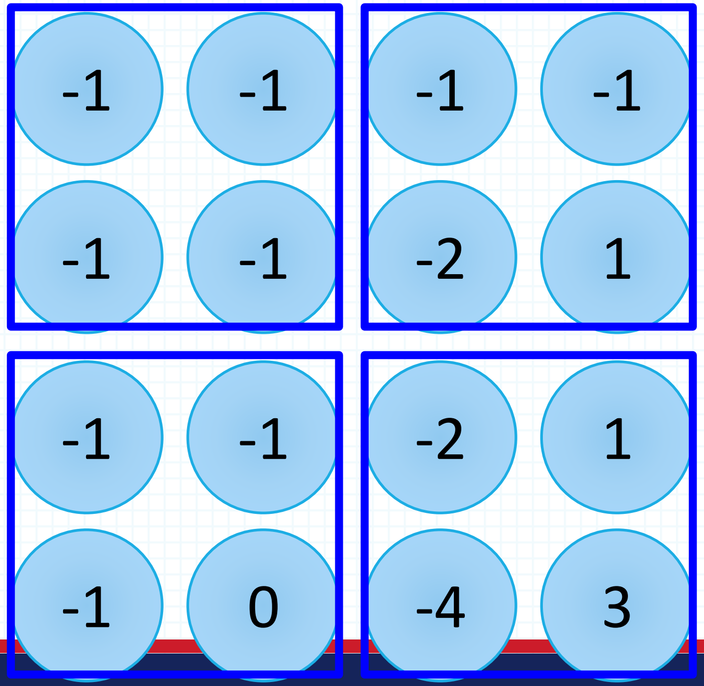
Filter 1

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

Filter 2



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



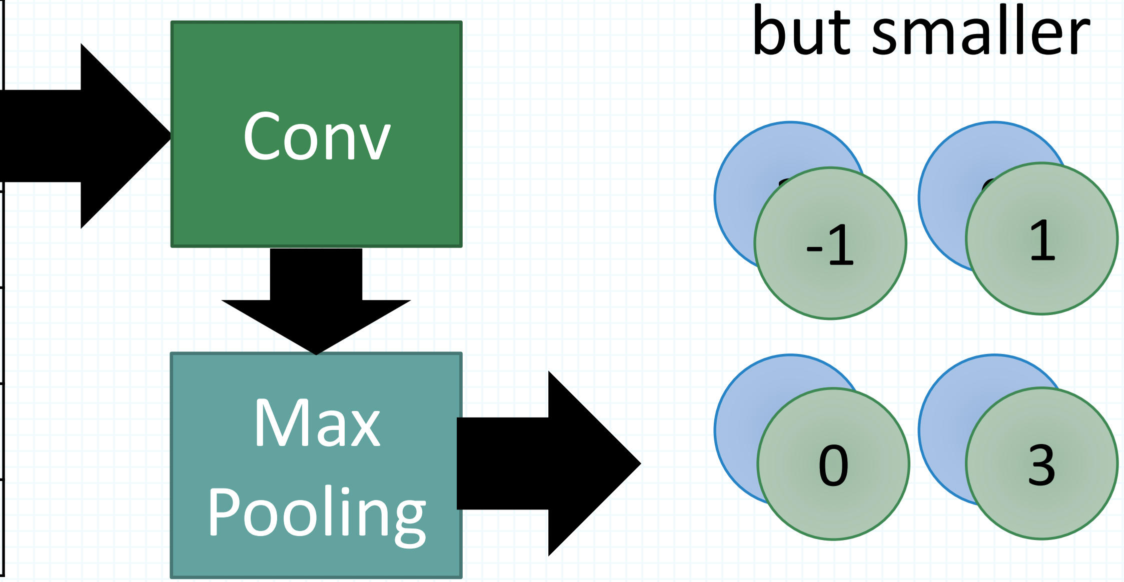
-1	-1	-1	-1
-1	-1	-2	1
-1	-1	-2	1
-1	0	-4	3

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

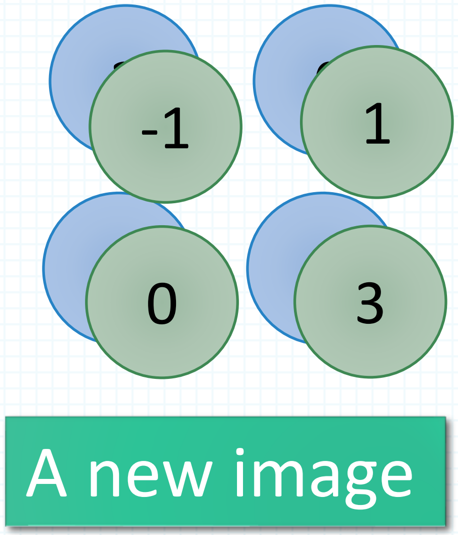
New image
but smaller



2 x 2 image

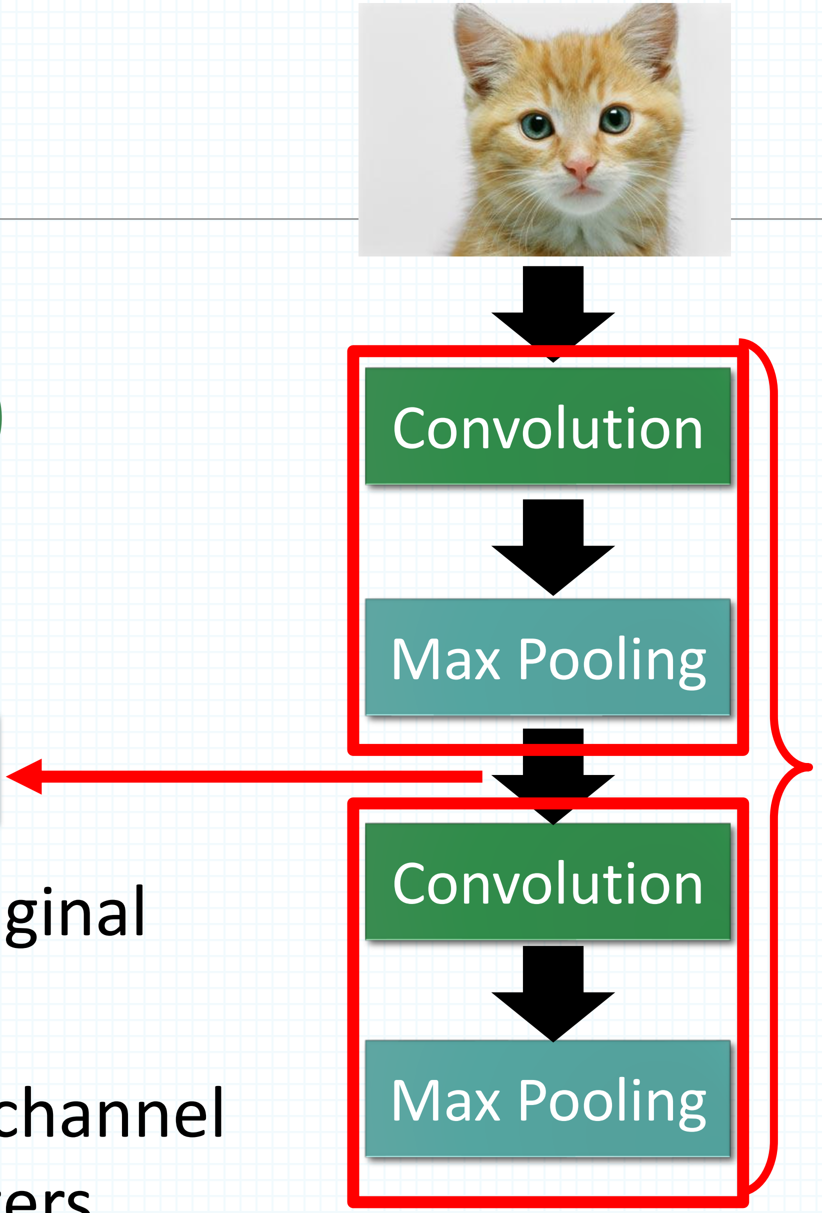
Each filter
is a channel

The whole CNN



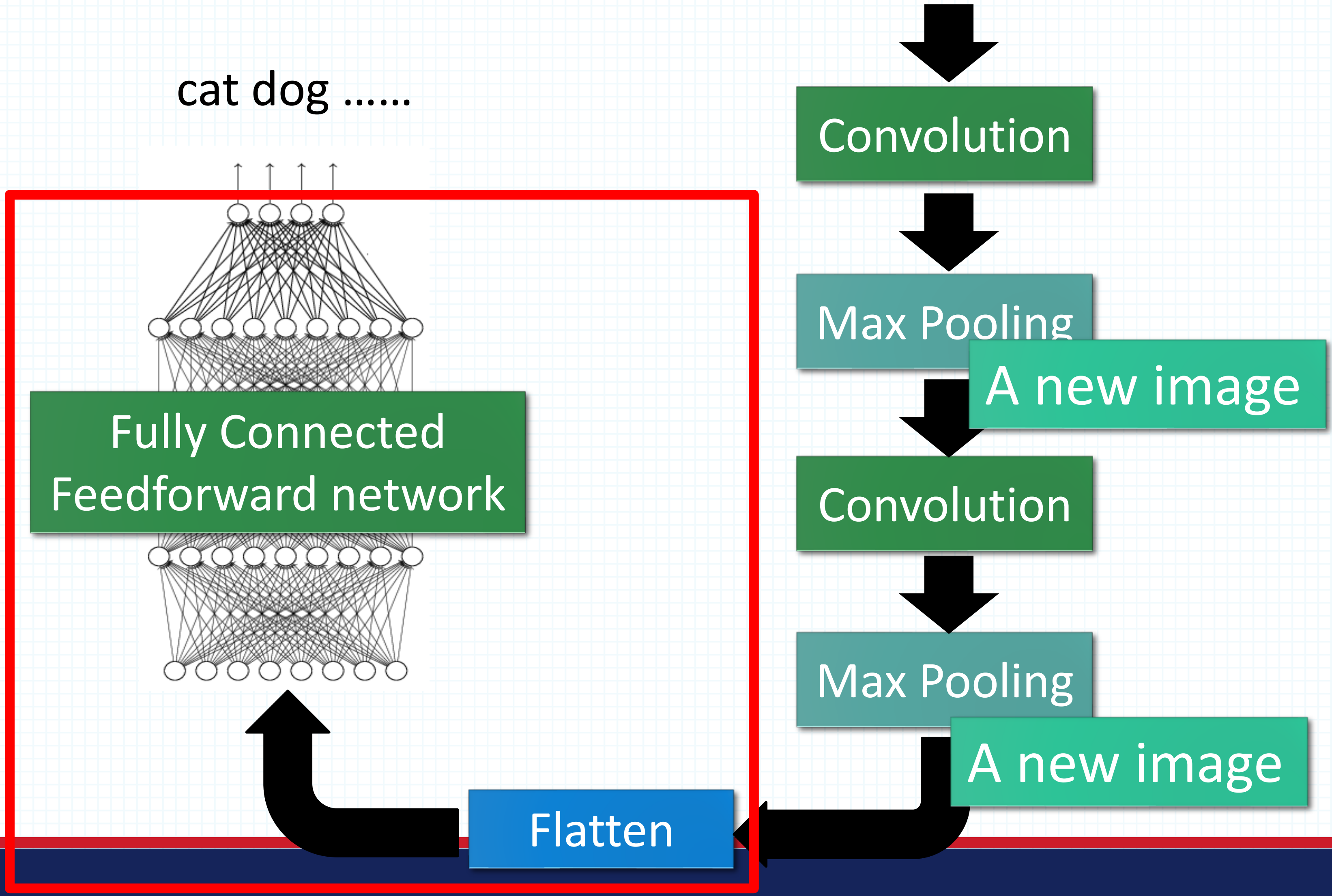
Smaller than the original image

The number of the channel is the number of filters



Can repeat many times

The whole CNN



cat dog ……

Convolution

Max Pooling

A new image

Convolution

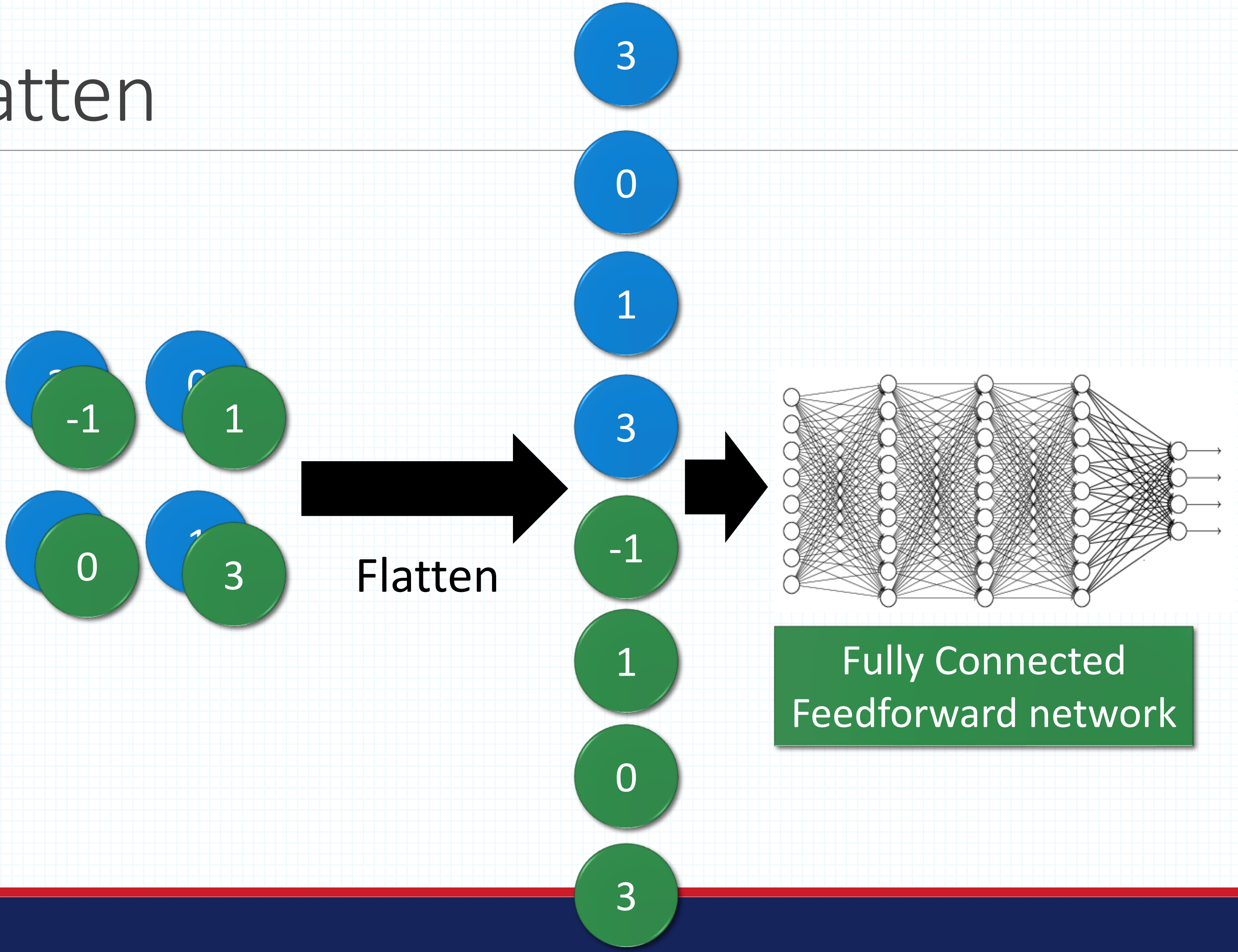
Max Pooling

A new image

Flatten

Fully Connected Feedforward network

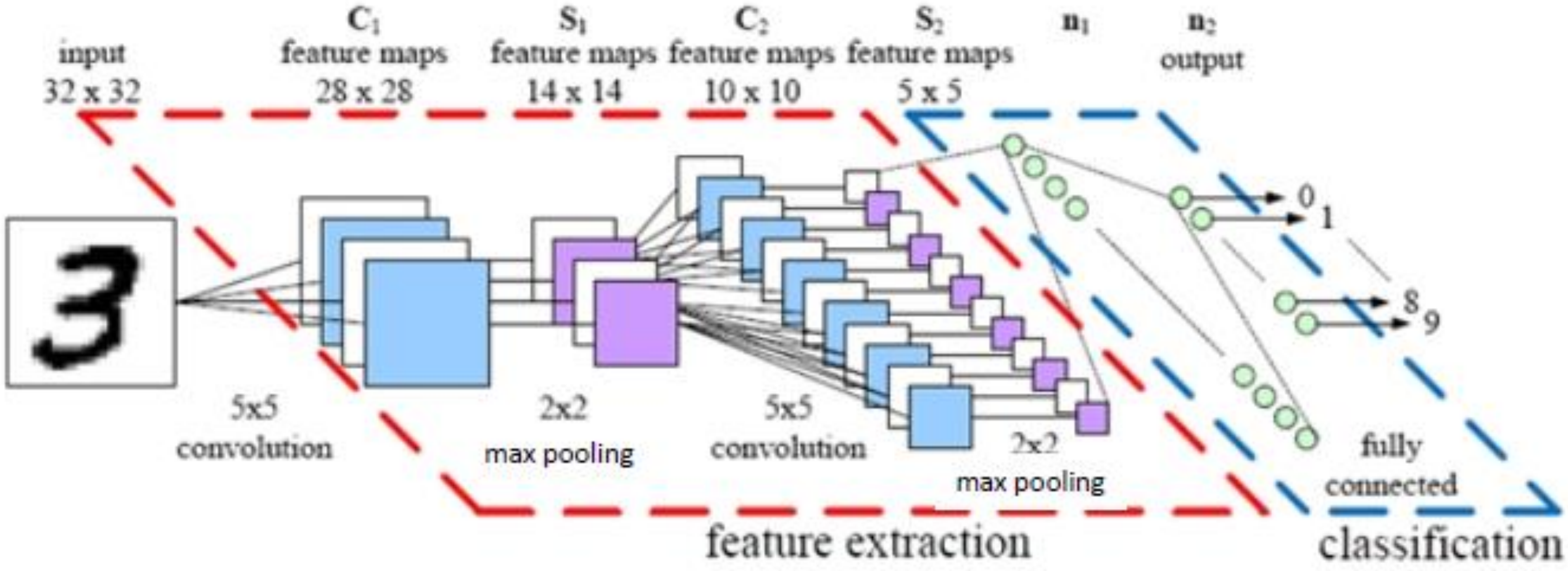
Flatten



Flatten

Fully Connected Feedforward network

Example


input
32 x 32
C1
feature maps
28 x 28
S1
feature maps
14 x 14
C2
feature maps
10 x 10
S2
feature maps
5 x 5
n1
n2
output
0
1
8
9
5x5
convolution
2x2
max pooling
5x5
convolution
2x2
max pooling
fully
connected
feature extraction
classification

CNN – Colorful image

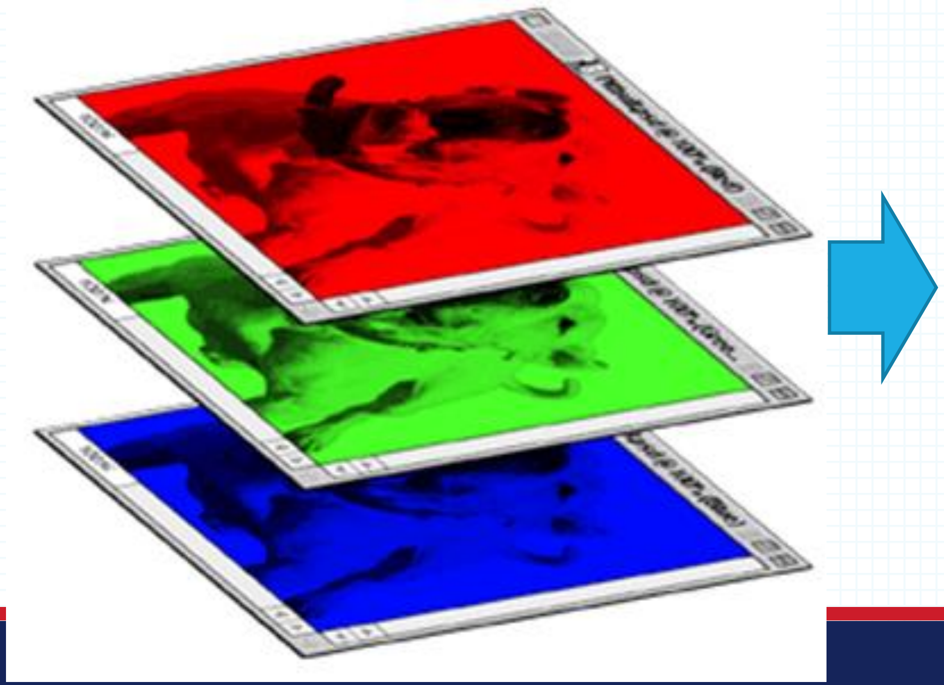
1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

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

Filter 2

Colorful image

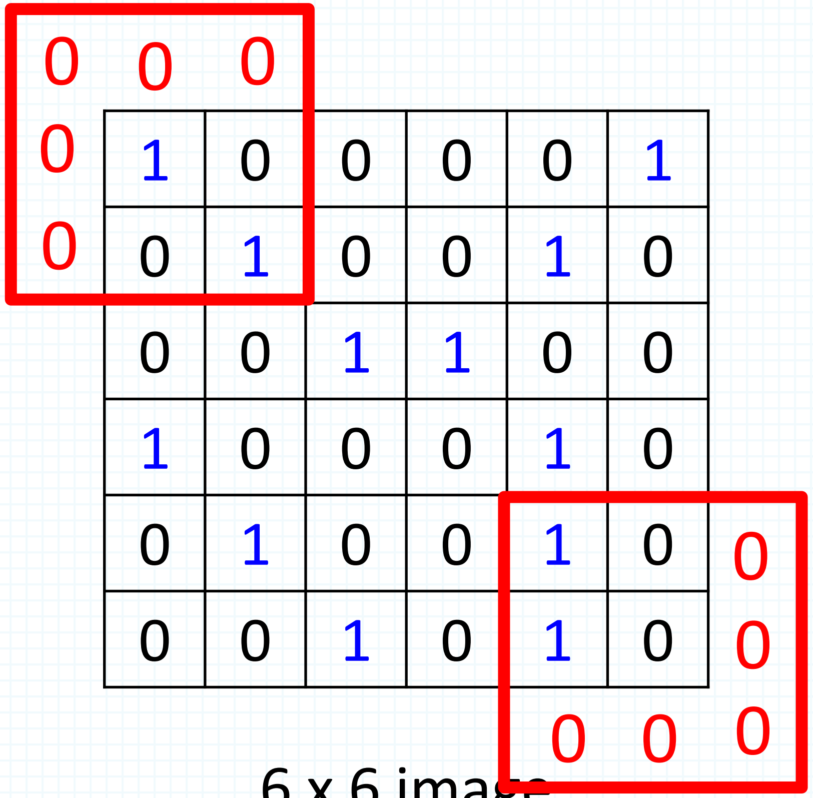


1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

CNN – Zero Padding

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

Filter 1



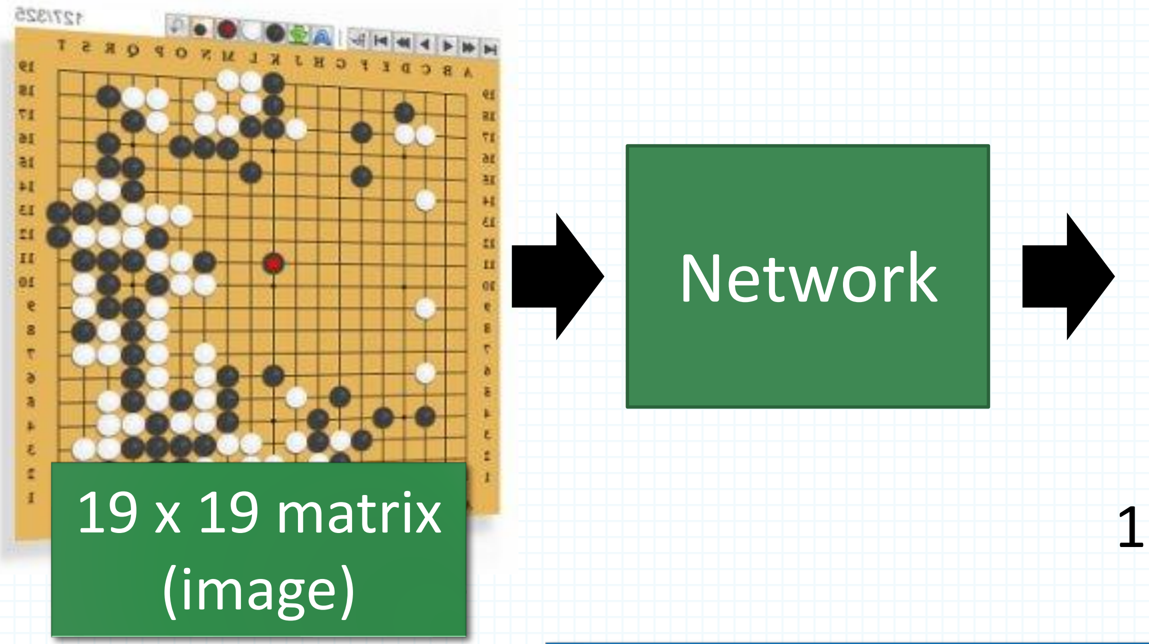
1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

6 x 6 image

You will get another 6 x 6 images in this way

Zero padding

More Application: Playing Go



19 x 19 matrix (image)

Network

Next move (19 x 19 positions)

19 x 19 vector

Black: 1

white: -1

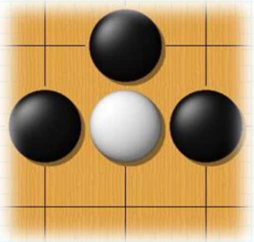
none: 0

Fully-connected feedforward network can be used

But CNN performs much better.

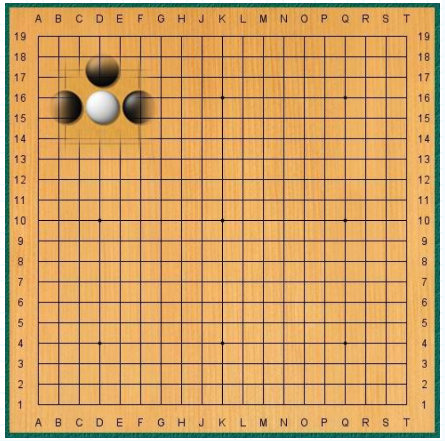
Why CNN for playing Go?

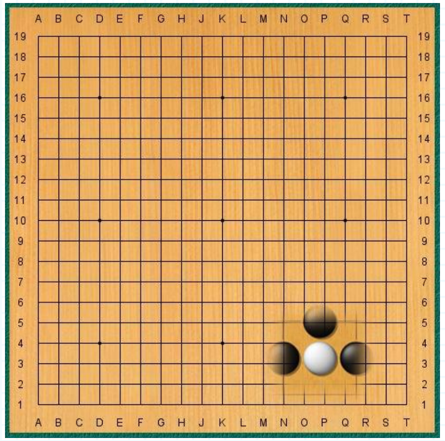
- Some patterns are much smaller than the whole image



- The same patterns appear in different regions.

Alpha Go uses 5 x 5 for first layer



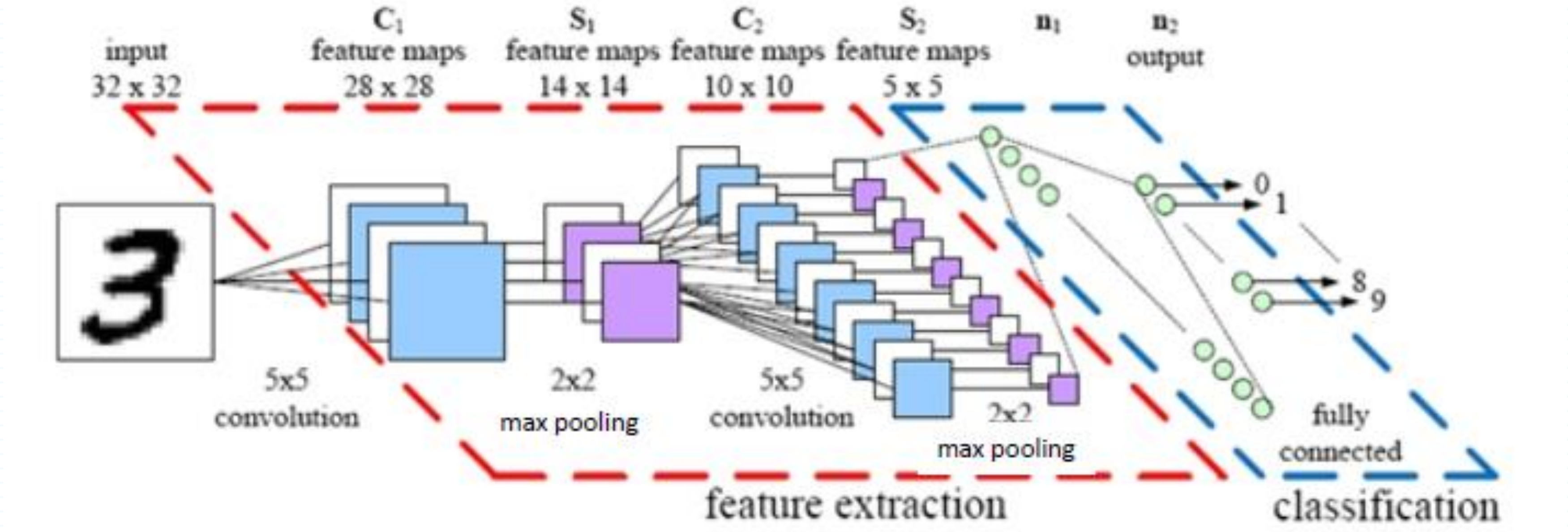


Parameters

- **# of filters**: integer indicating the # of filters applied to each window.
- **kernel size:** tuple (width, height) indicating the size of the window.
- **Stride:** tuple (horizontal, vertical) indicating the horizontal and vertical shift between each window.
- **Padding:** “valid” or ”same”. Valid indicates no input padding. Same indicates that the input is padded with a border of zeros to ensure that the output has the same size as the input.

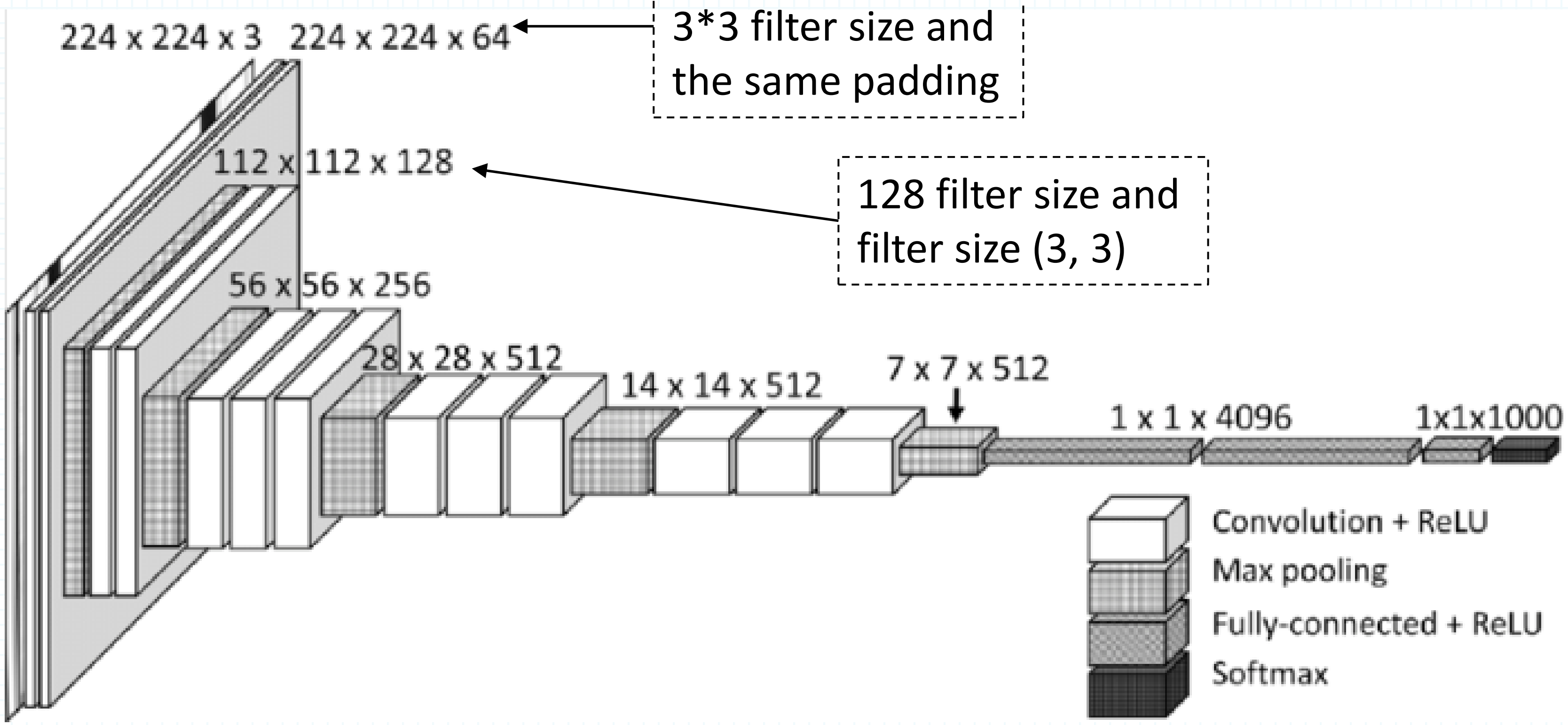
Training

- Convolutional neural networks are trained in the same way as other neural networks
  - backpropagation
- Weight sharing:
  - Combine gradients of shared weights into a single gradient



Architecture design: VGG

- What is the preferred filter size?
- VGG (Visual Geometry Group at Oxford, 2014): stack of small filters is often preferred to single large filter
    - Fewer parameters
    - Deeper network



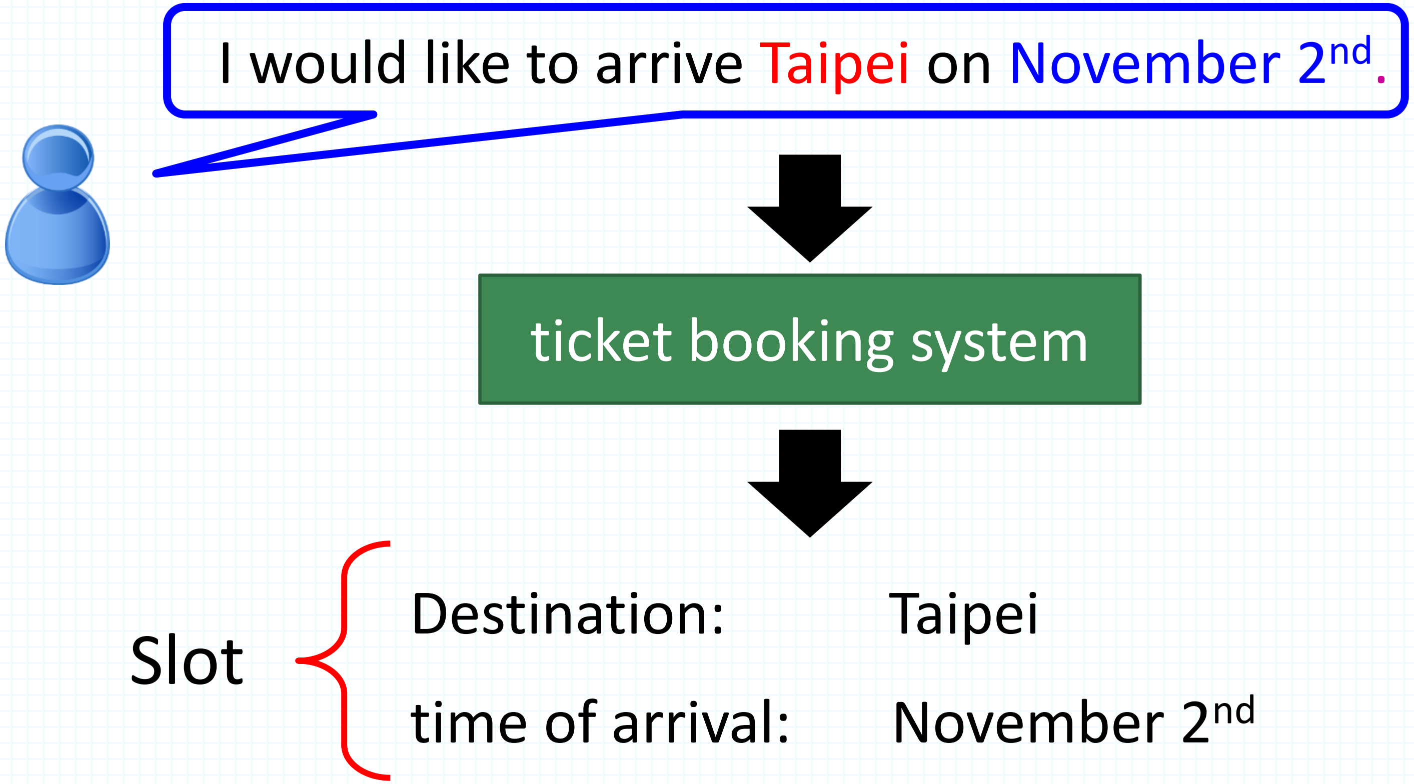
64 channels of a 3*3 filter size and the same padding

128 filter size and filter size (3, 3)

الشبكات العصبونية التكراريّة
Recurrent Neural Networks

Example Application

- Slot Filling



I would like to arrive Taipei on November 2nd.

ticket booking system

Slot

Destination: Taipei

time of arrival: November 2nd

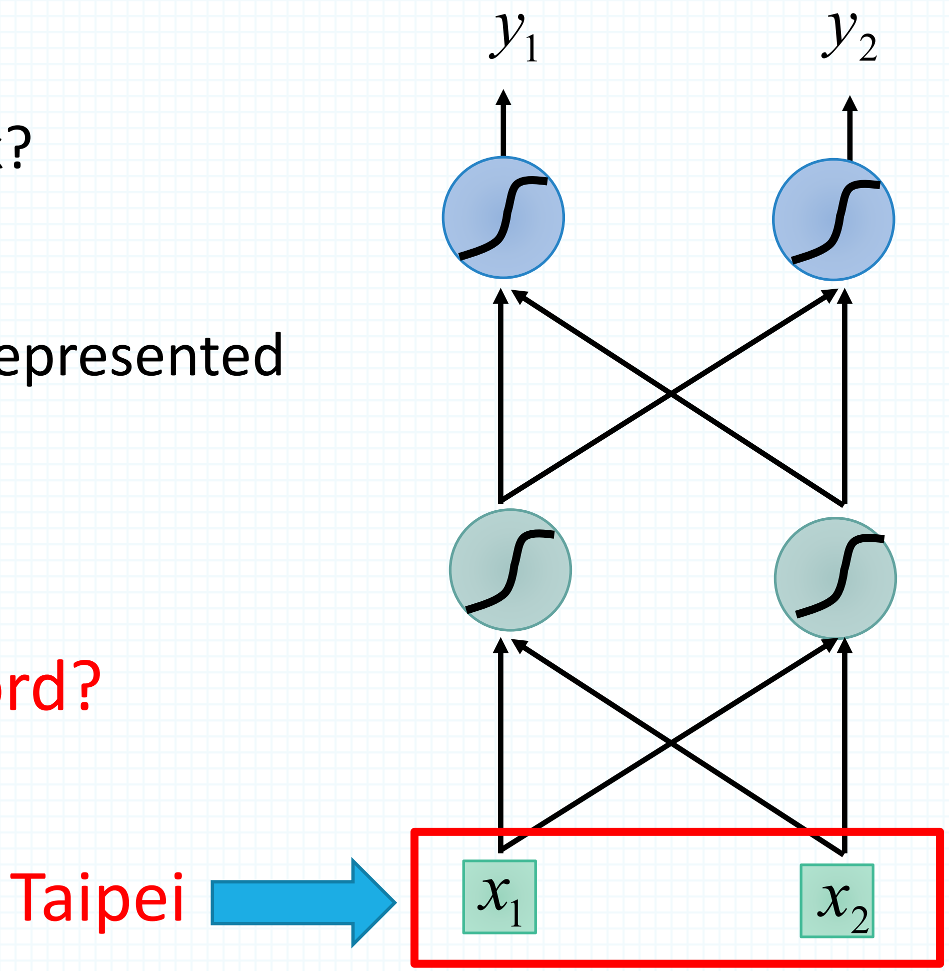
Example Application

Solving slot filling by Feedforward network?

Input: a word

(Each word is represented as a vector)

How can we represent each word?



1-of-N encoding

How to represent each word as a vector?

1-of-N Encoding lexicon = {apple, bag, cat, dog, elephant}

The vector is lexicon size.

Each dimension corresponds to a word in the lexicon

The dimension for the word is 1, and others are 0

apple = [1 0 0 0 0]

bag = [0 1 0 0 0]

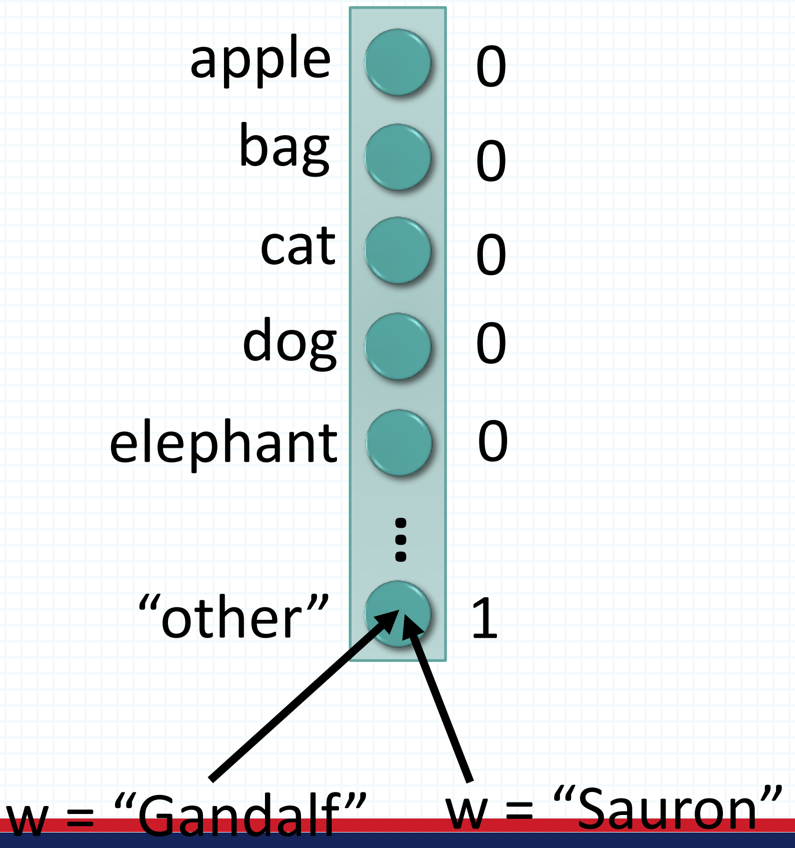
cat = [0 0 1 0 0]

dog = [0 0 0 1 0]

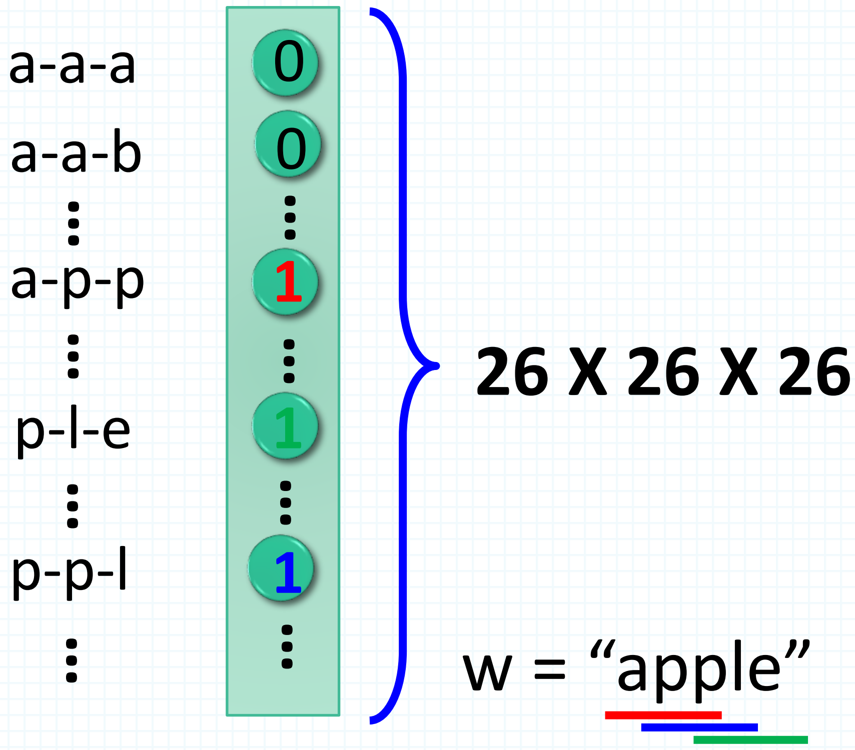
elephant = [0 0 0 0 1]

Beyond 1-of-N encoding

Dimension for "Other"



Word hashing



Example Application

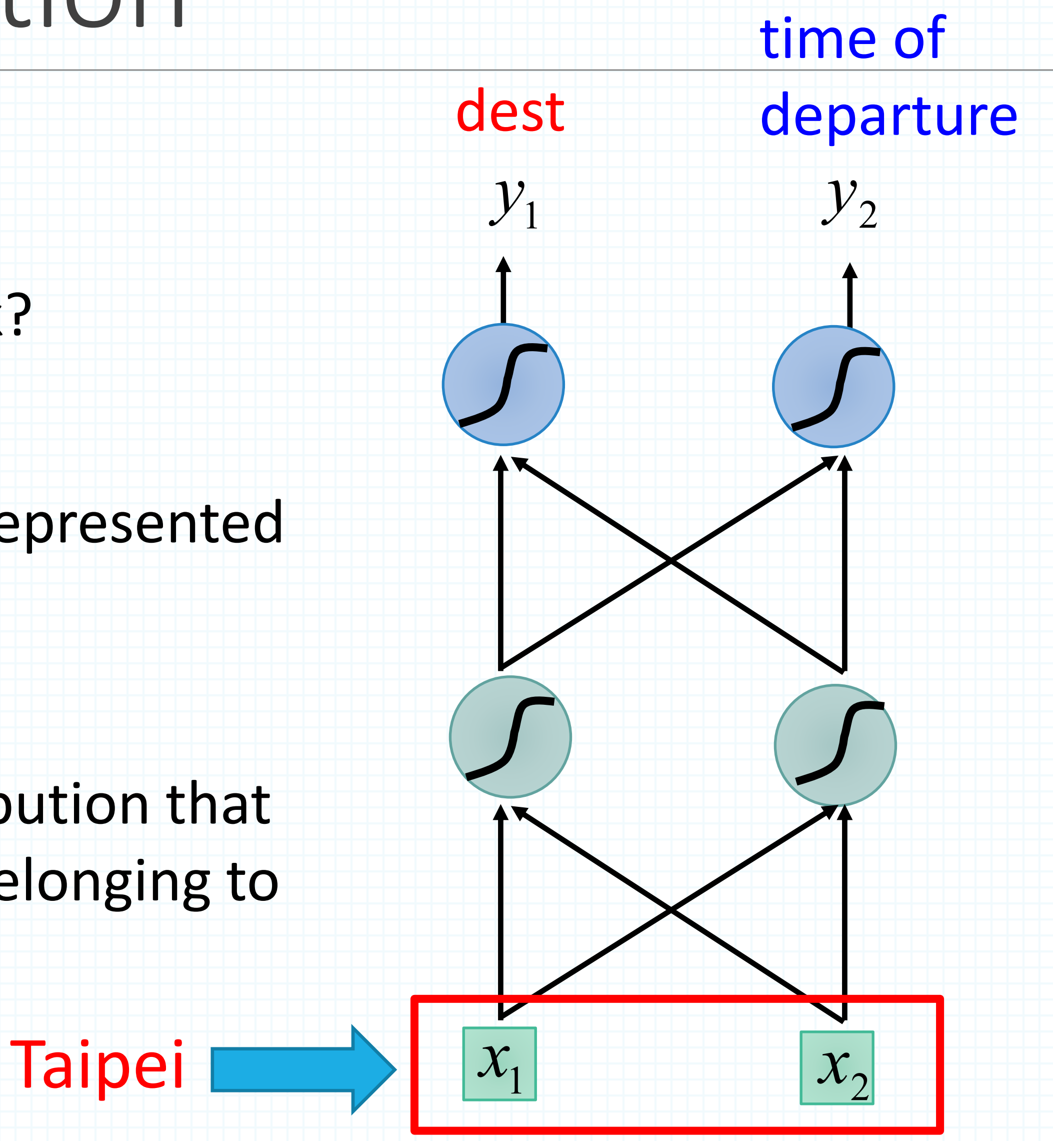
Solving slot filling by Feedforward network?

Input: a word

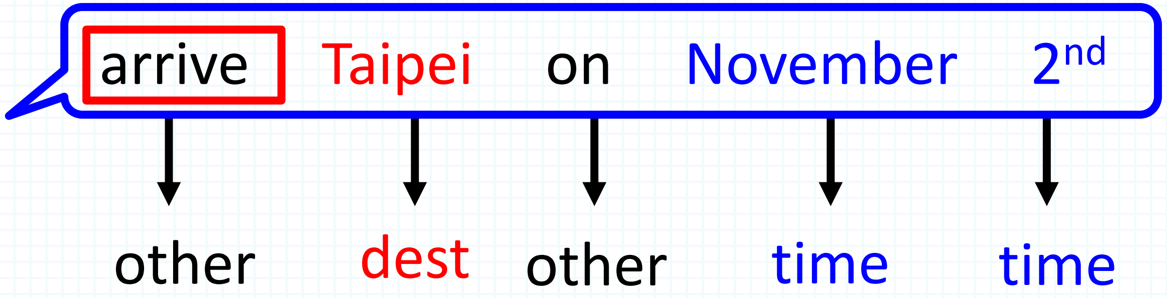
(Each word is represented as a vector)

Output:

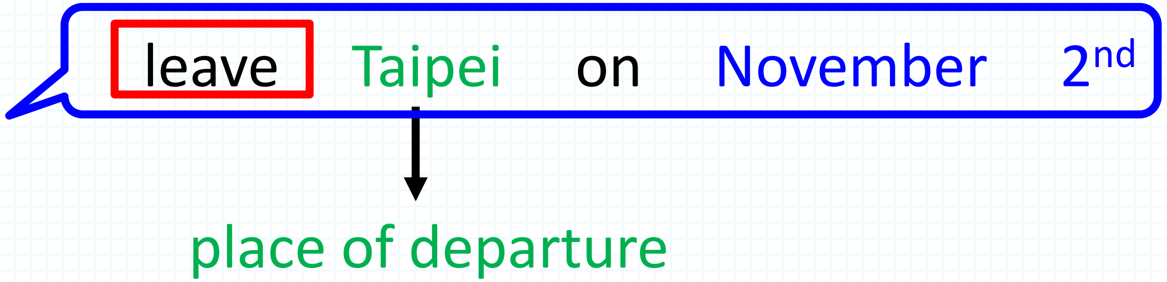
Probability distribution that the input word belonging to the slots



Example Application

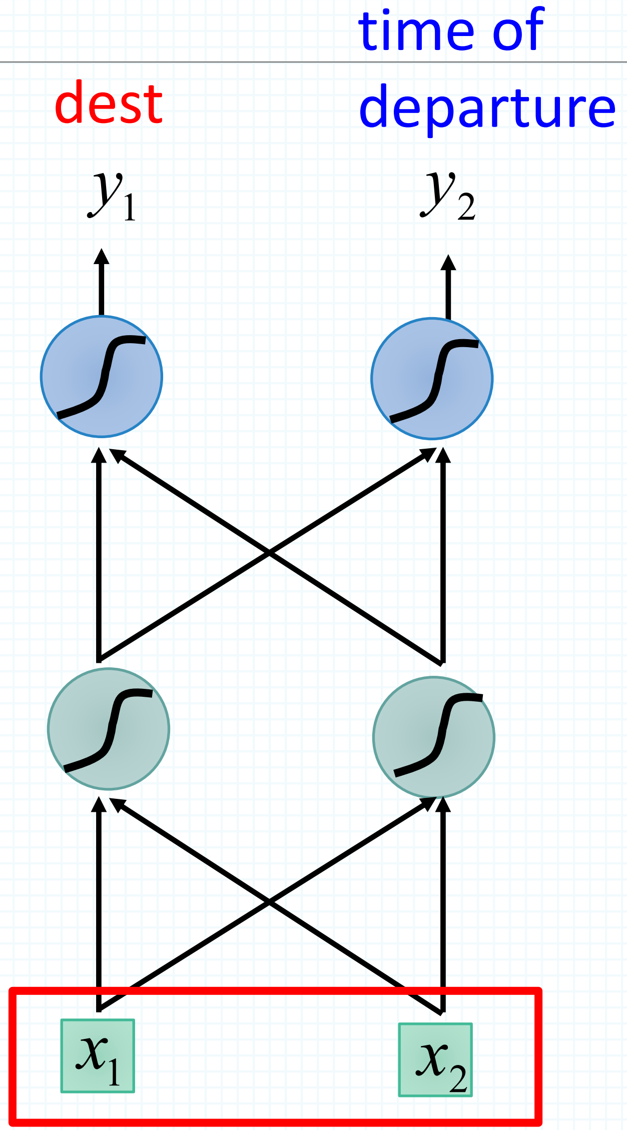


Problem?



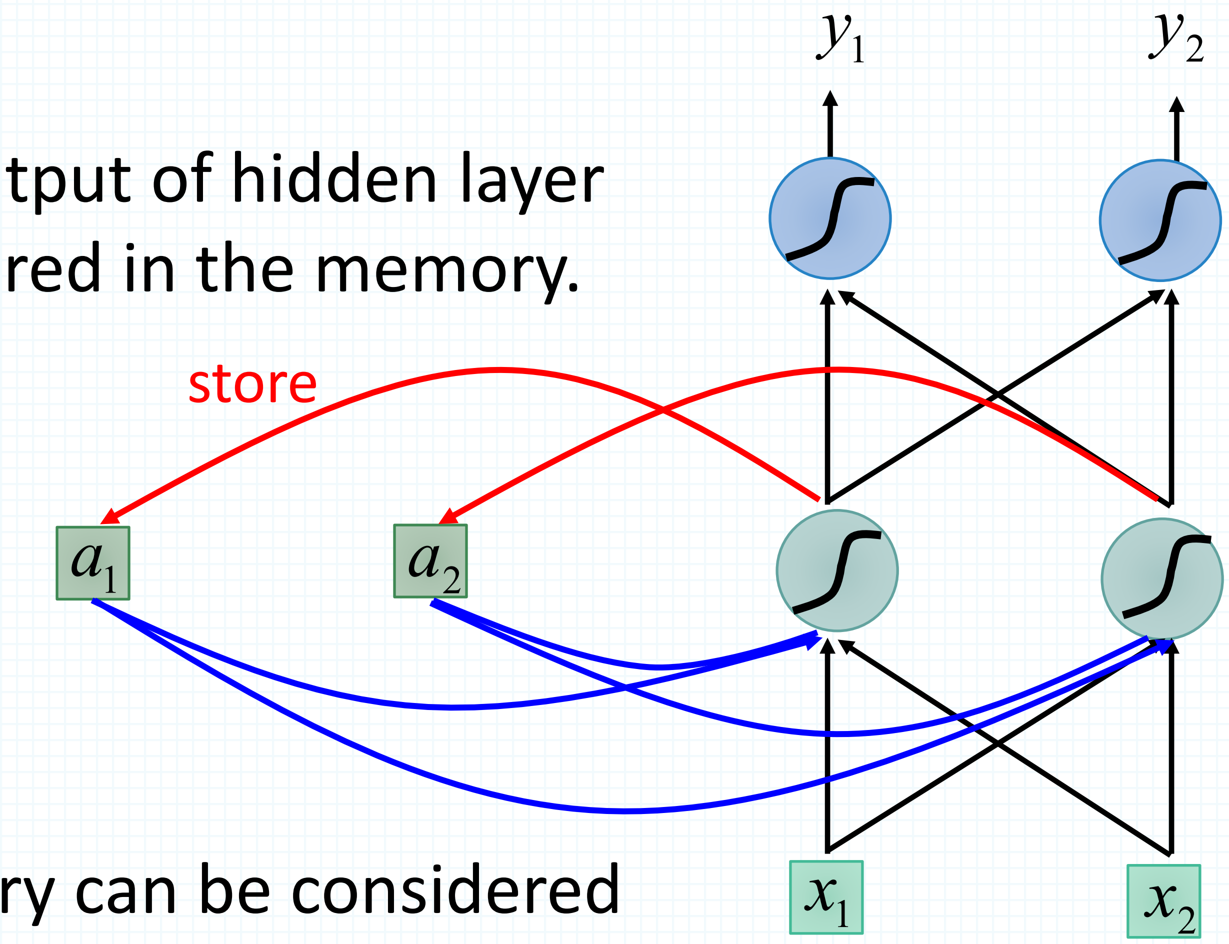
Neural network needs memory!

Taipei



Recurrent Neural Network (RNN)

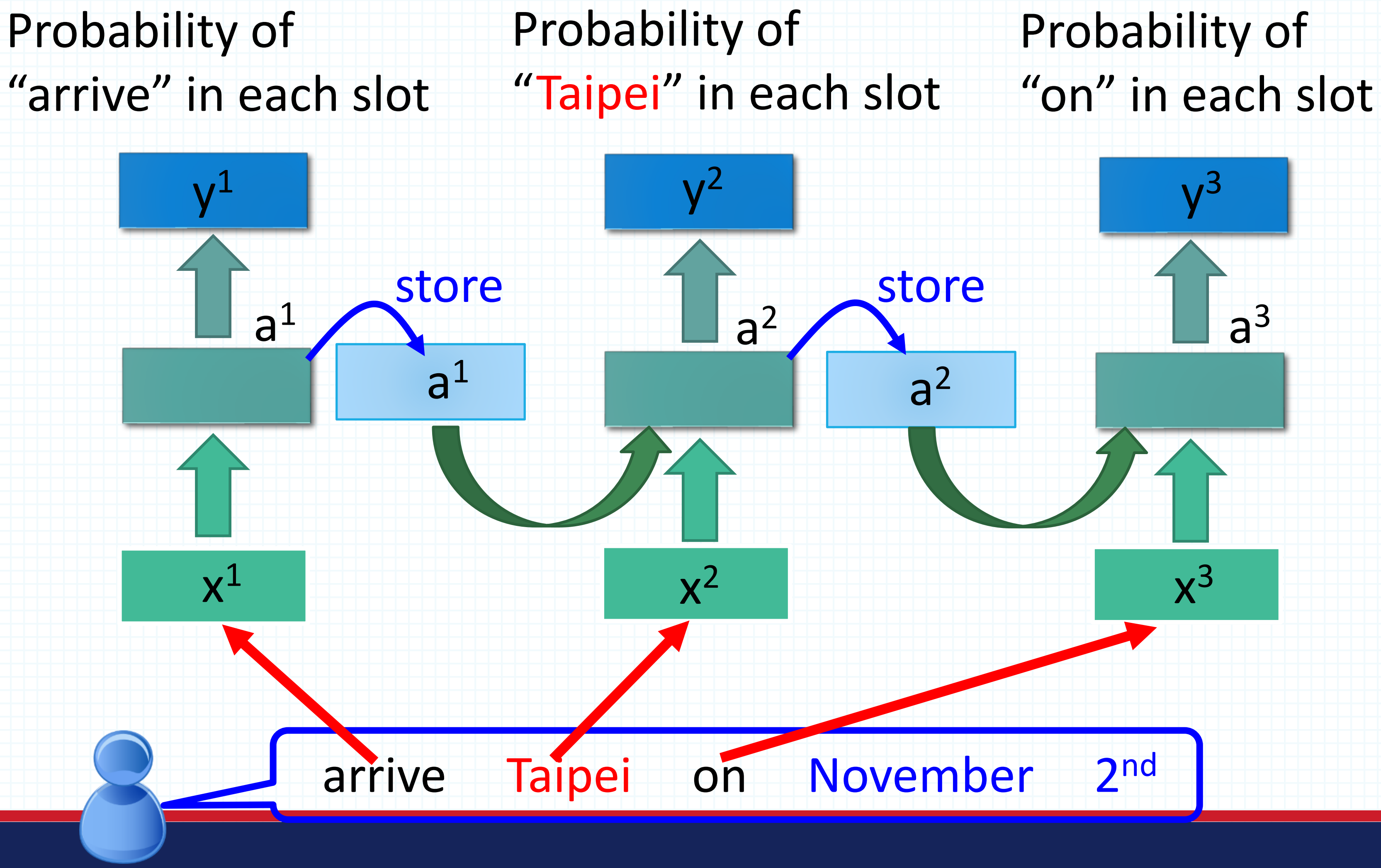
The output of hidden layer are stored in the memory.



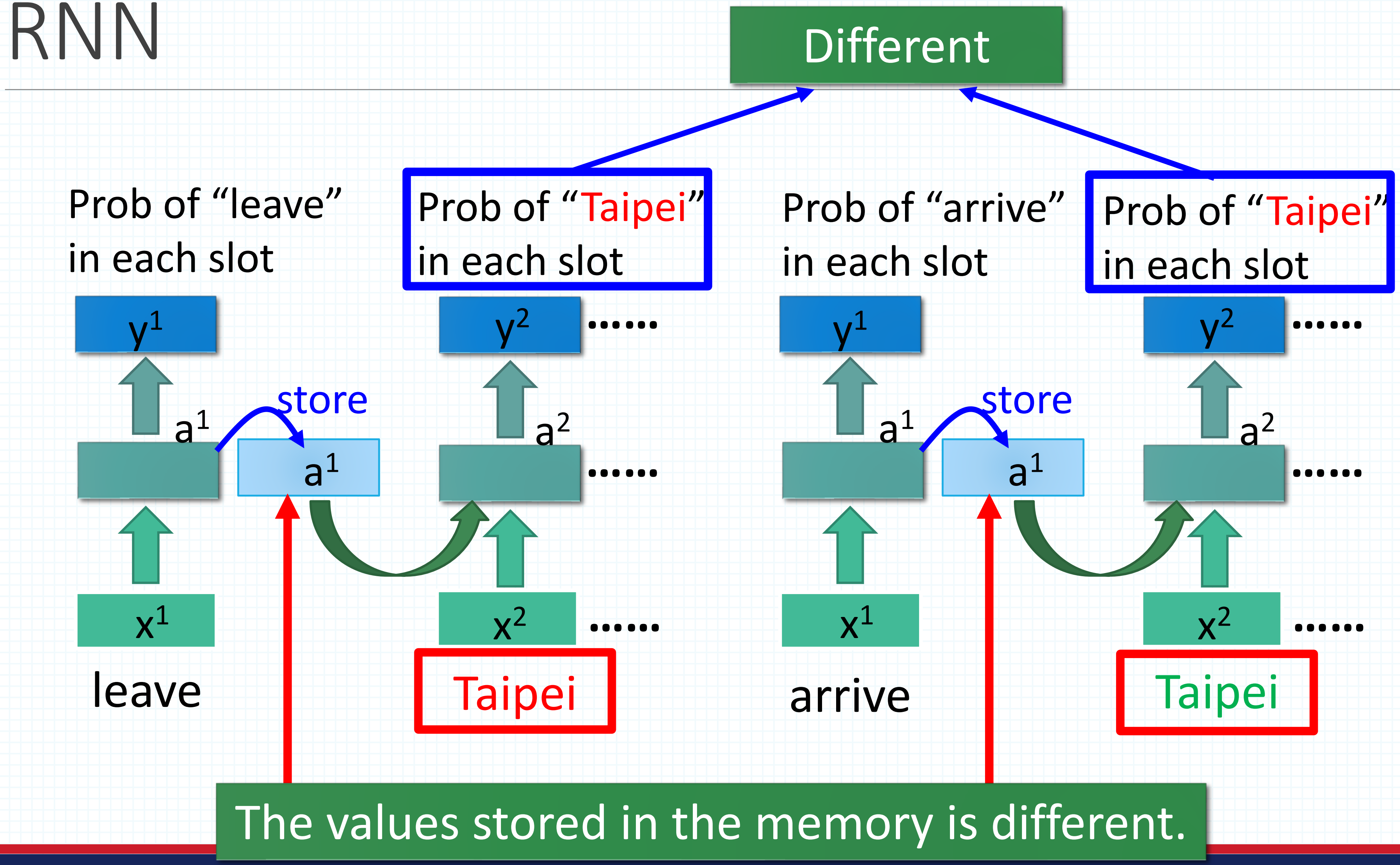
Memory can be considered as another input.

RNN

The same network is used again and again.

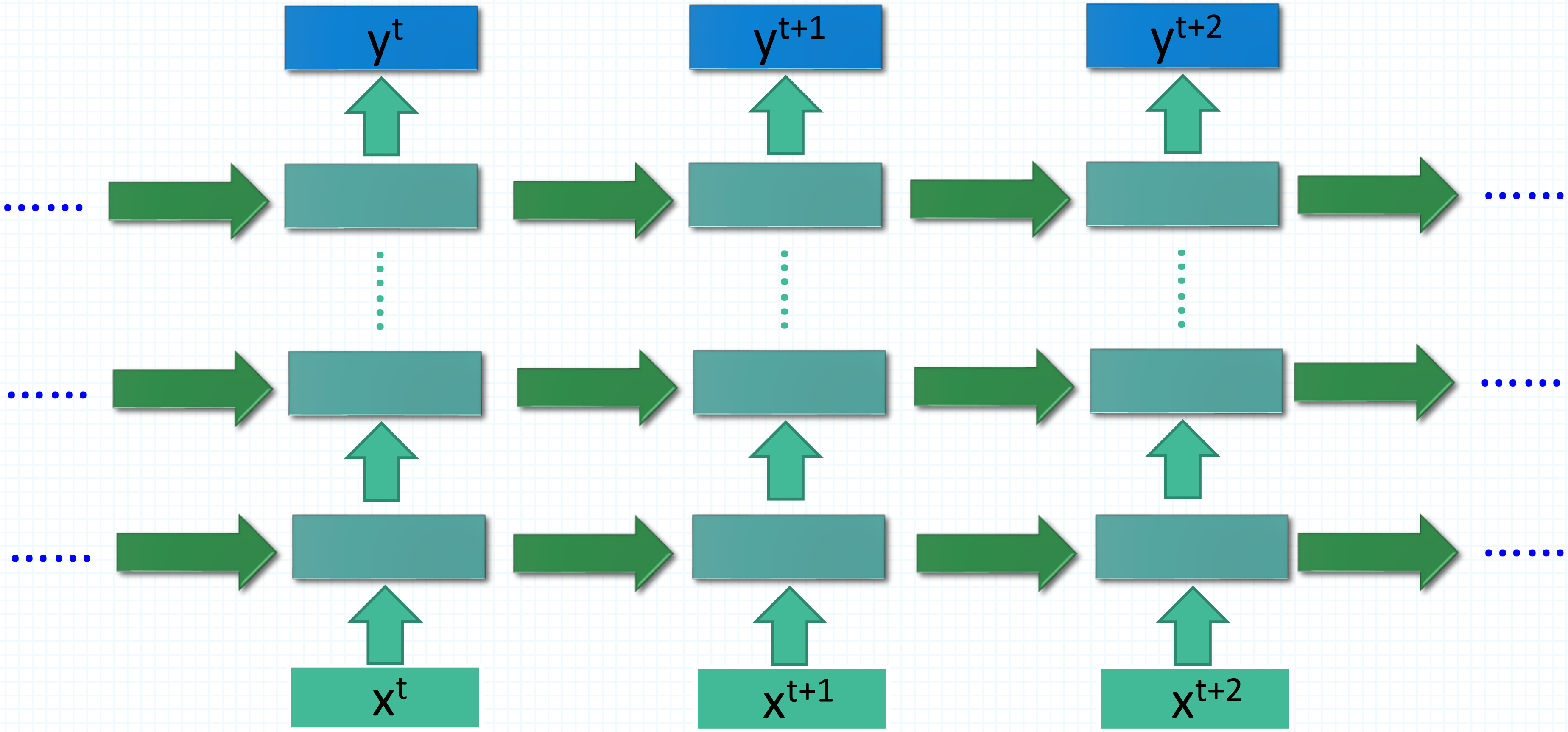


RNN



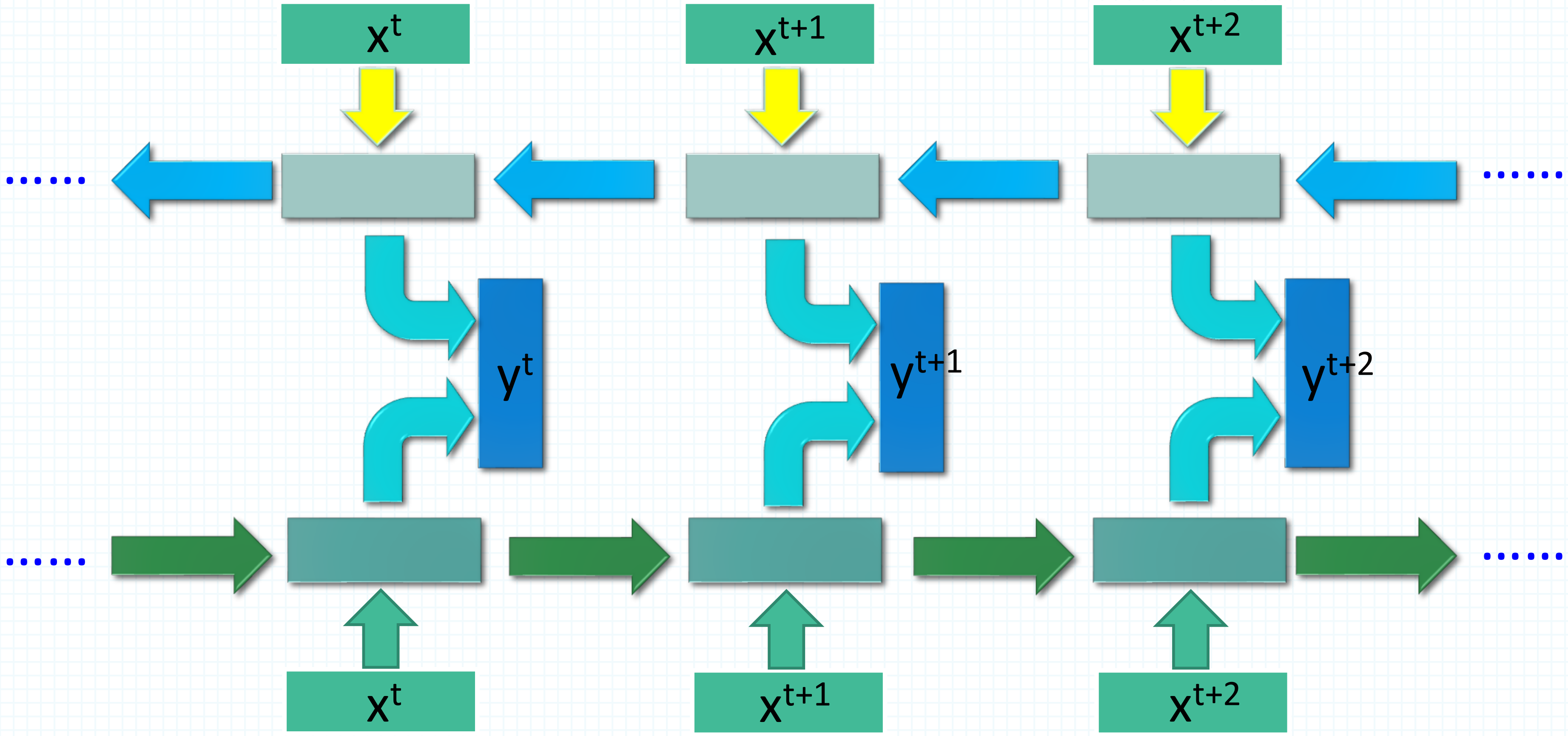
The values stored in the memory is different.

Of course it can be deep …



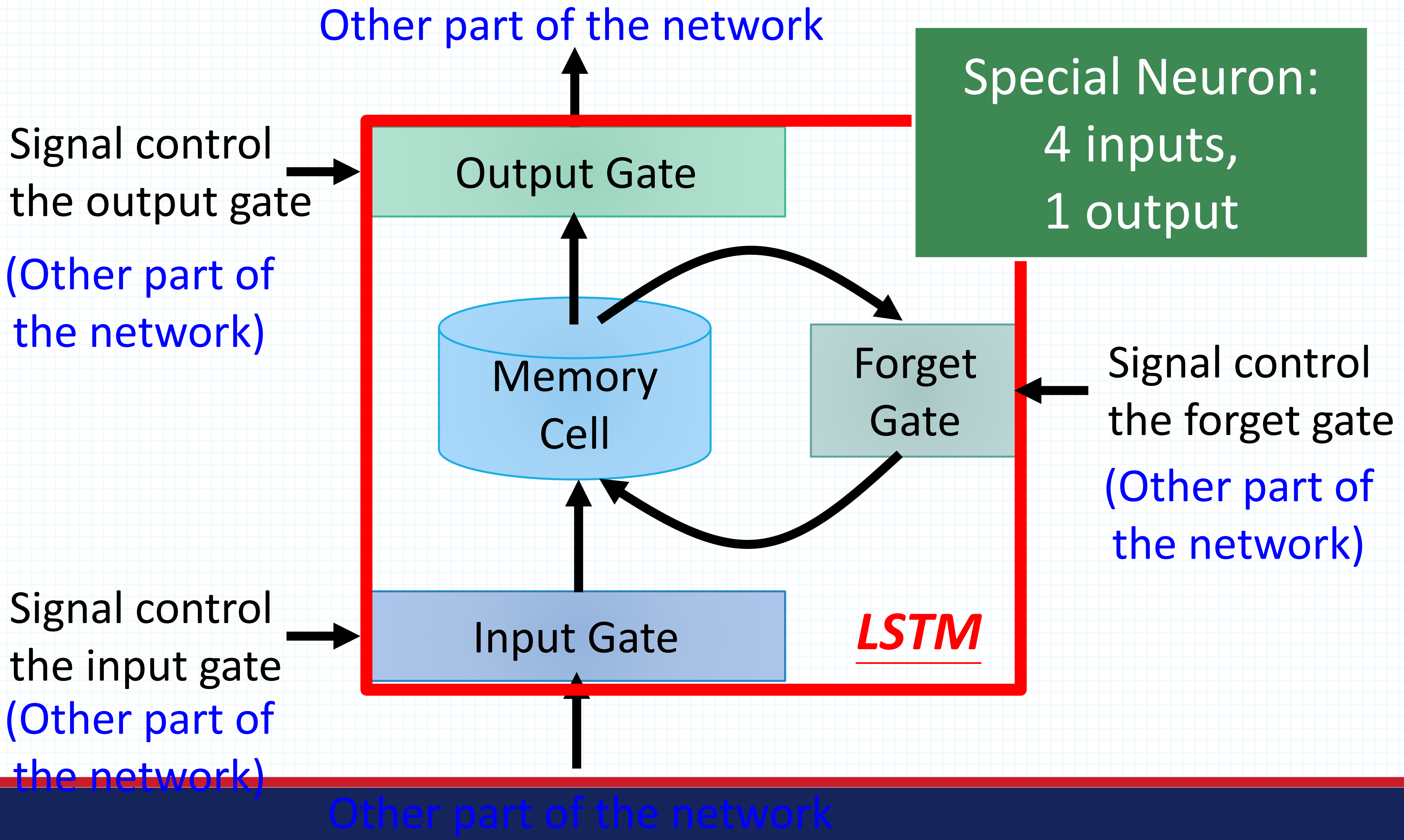
Bidirectional RNN

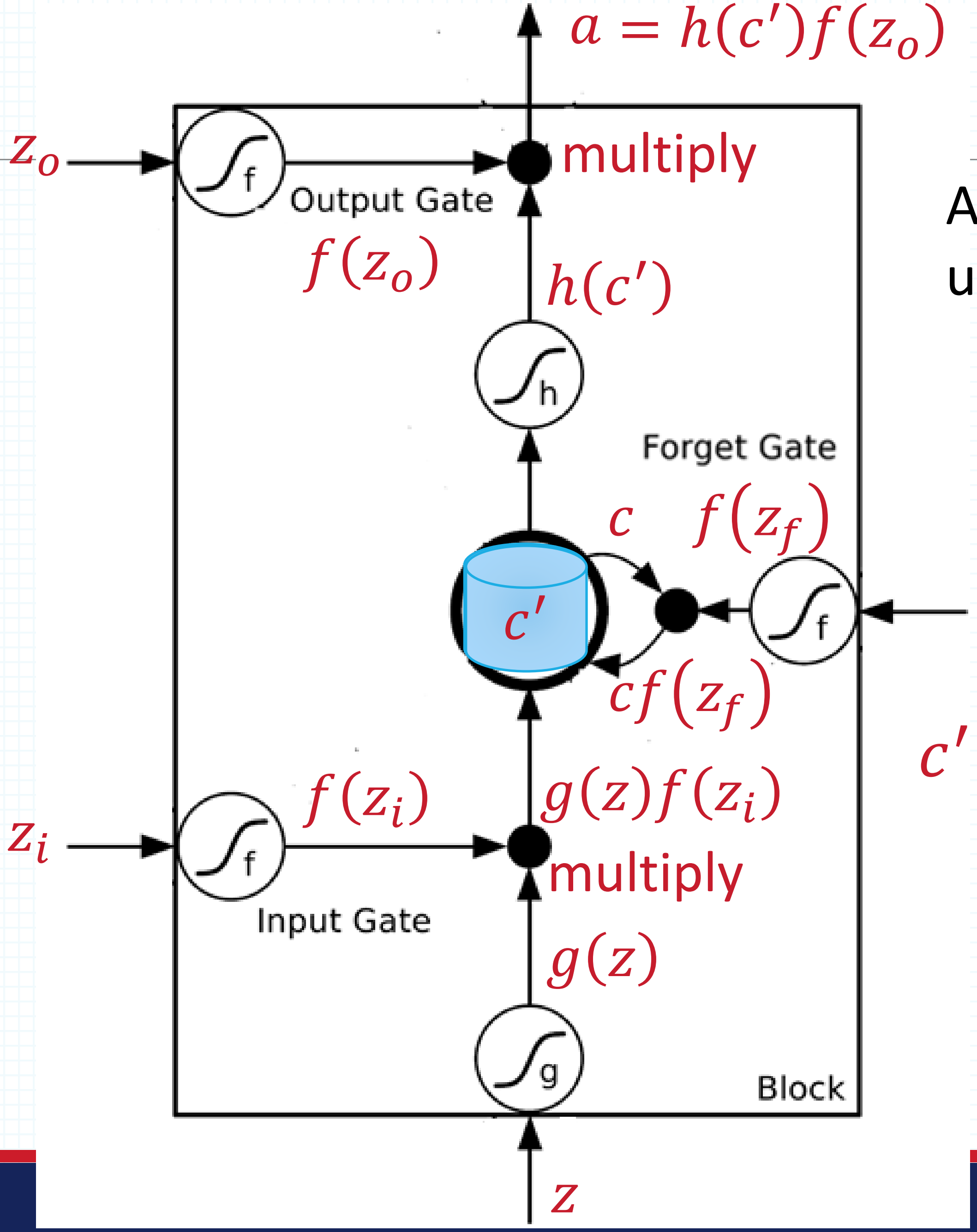
We can combine past and future evidence in separate chains



Long Short-term Memory (LSTM)

Long Short-term Memory (LSTM)


Other part of the network
Special Neuron:
4 inputs,
1 output
Signal control the output gate
(Other part of the network)
Output Gate
Memory Cell
Forget Gate
Signal control the forget gate
(Other part of the network)
Signal control the input gate
(Other part of the network)
Input Gate
LSTM
Other part of the network

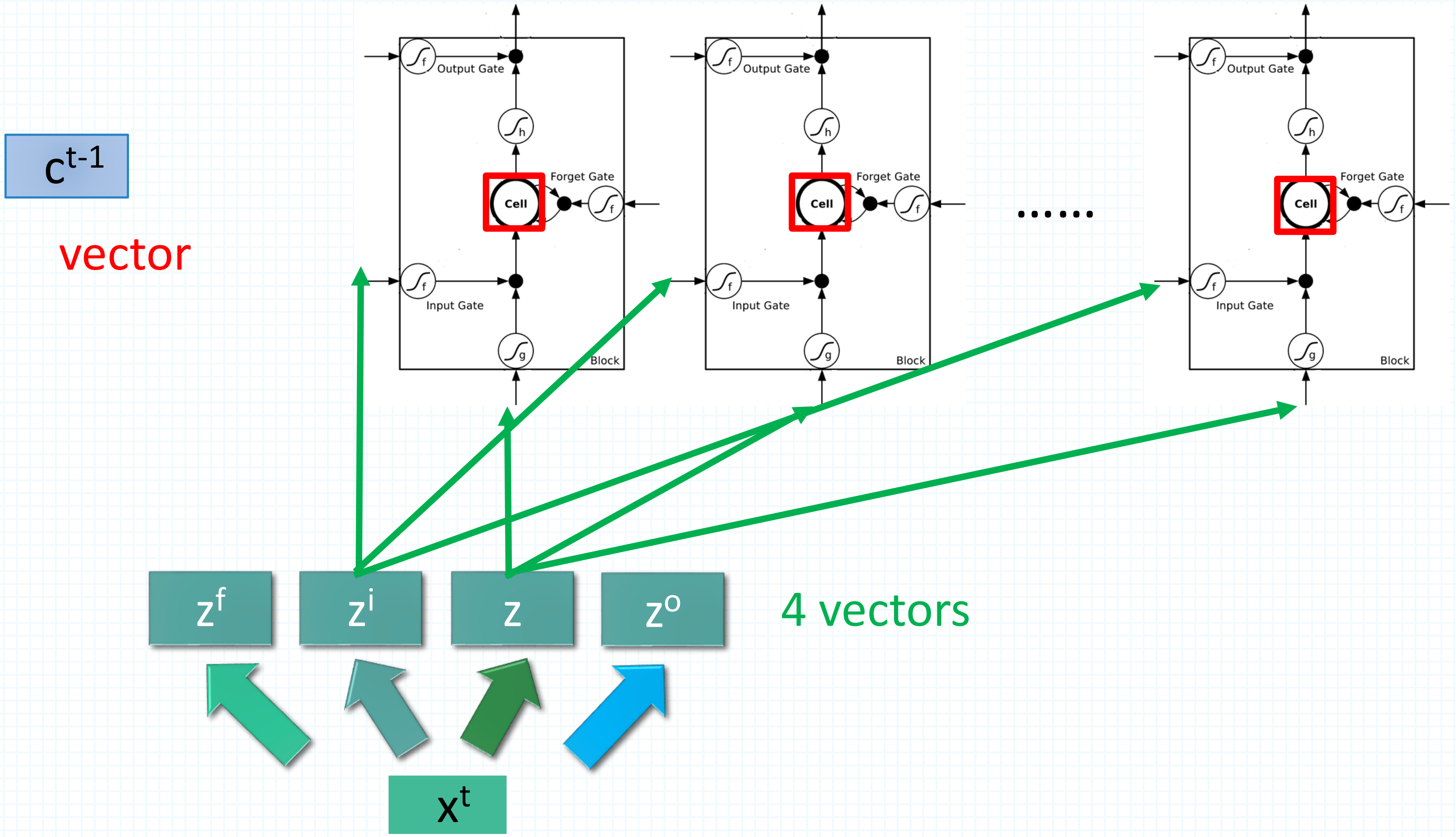


Activation function f is usually a sigmoid function

- Between 0 and 1
- Mimic open and close gate

$$c' = g(z)f(z_i) + cf(z_f)$$

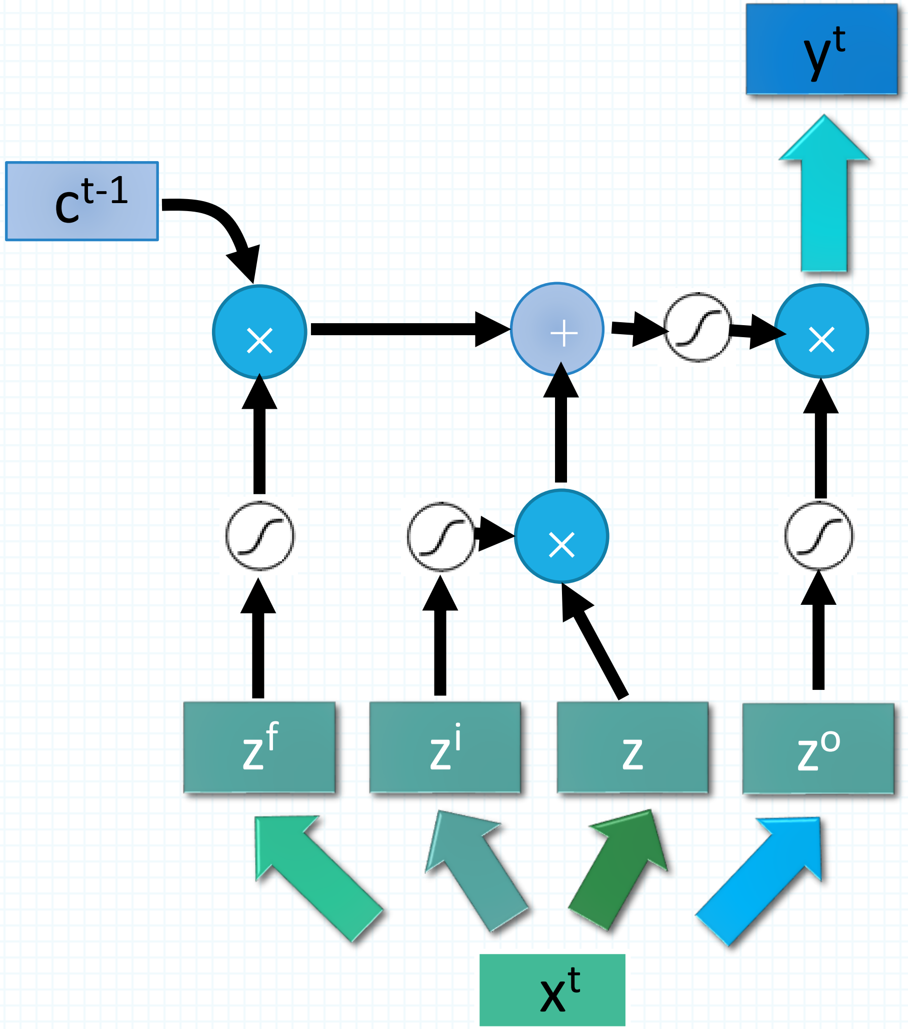
LSTM

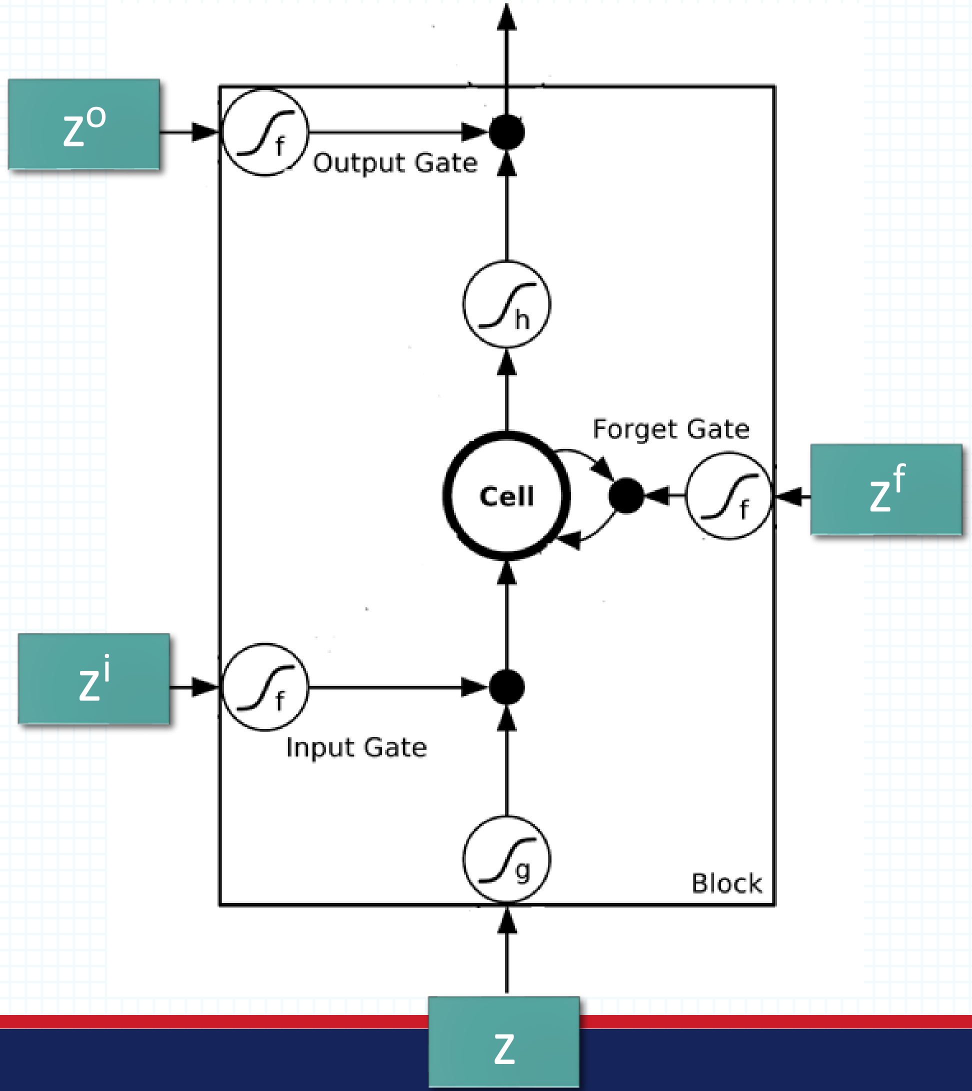


vector

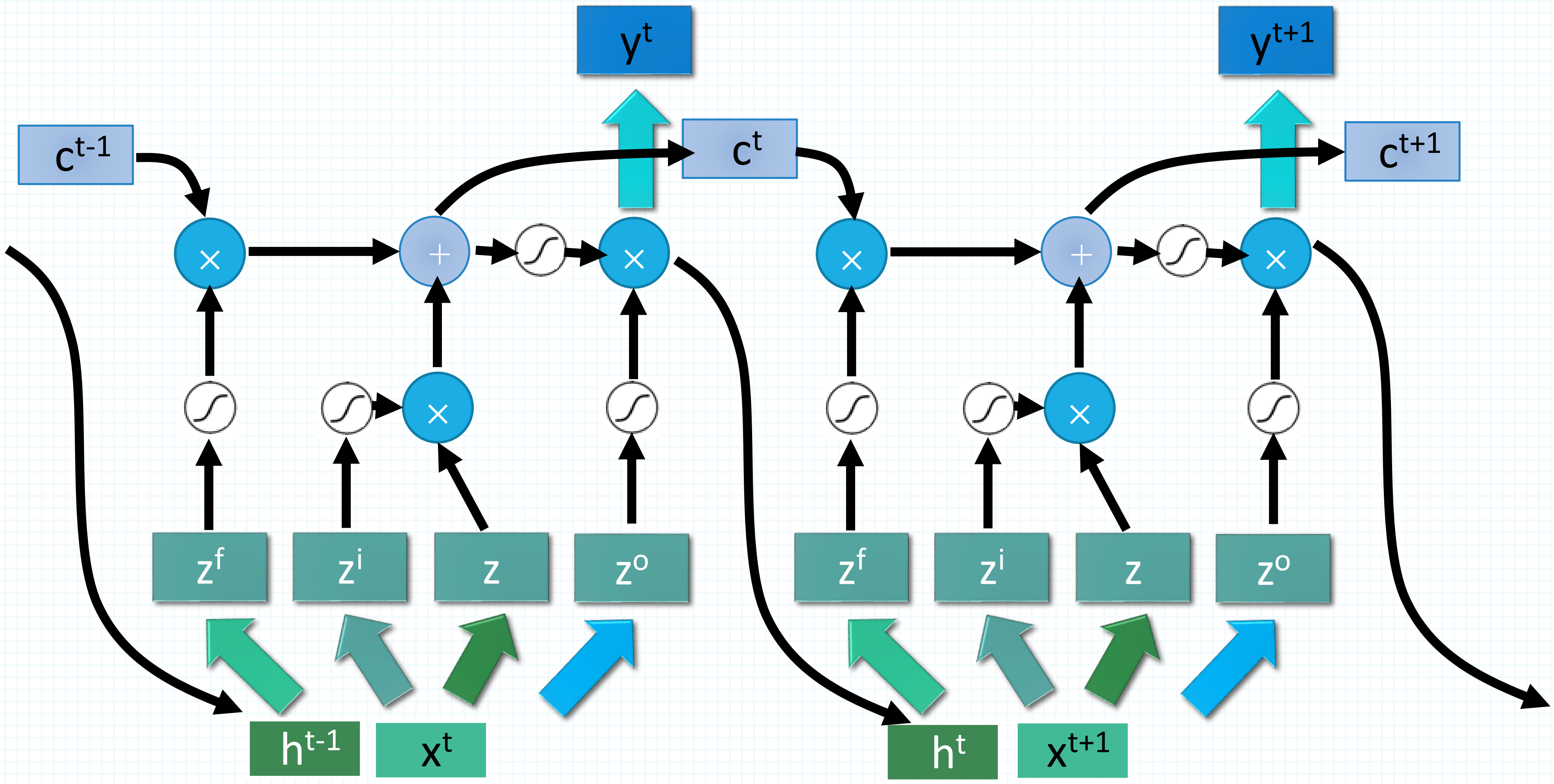
4 vectors

LSTM


y^t
c^{t-1}
×
+
×
×
z^f
z^i
z
z^o
x^t


z^o
f
Output Gate
h
Forget Gate
Cell
f
z^f
z^i
f
Input Gate
g
Block
z

LSTM



Encoder-Decoder Model (Seq2Seq Model)

Encoder-Decoder Model

- $X^{(i)}$: i^th input
- $Y^{(i)}$: i^th output
- C: context (embedding)

Usage:

- Machine translation
- Question answering
- Dialog

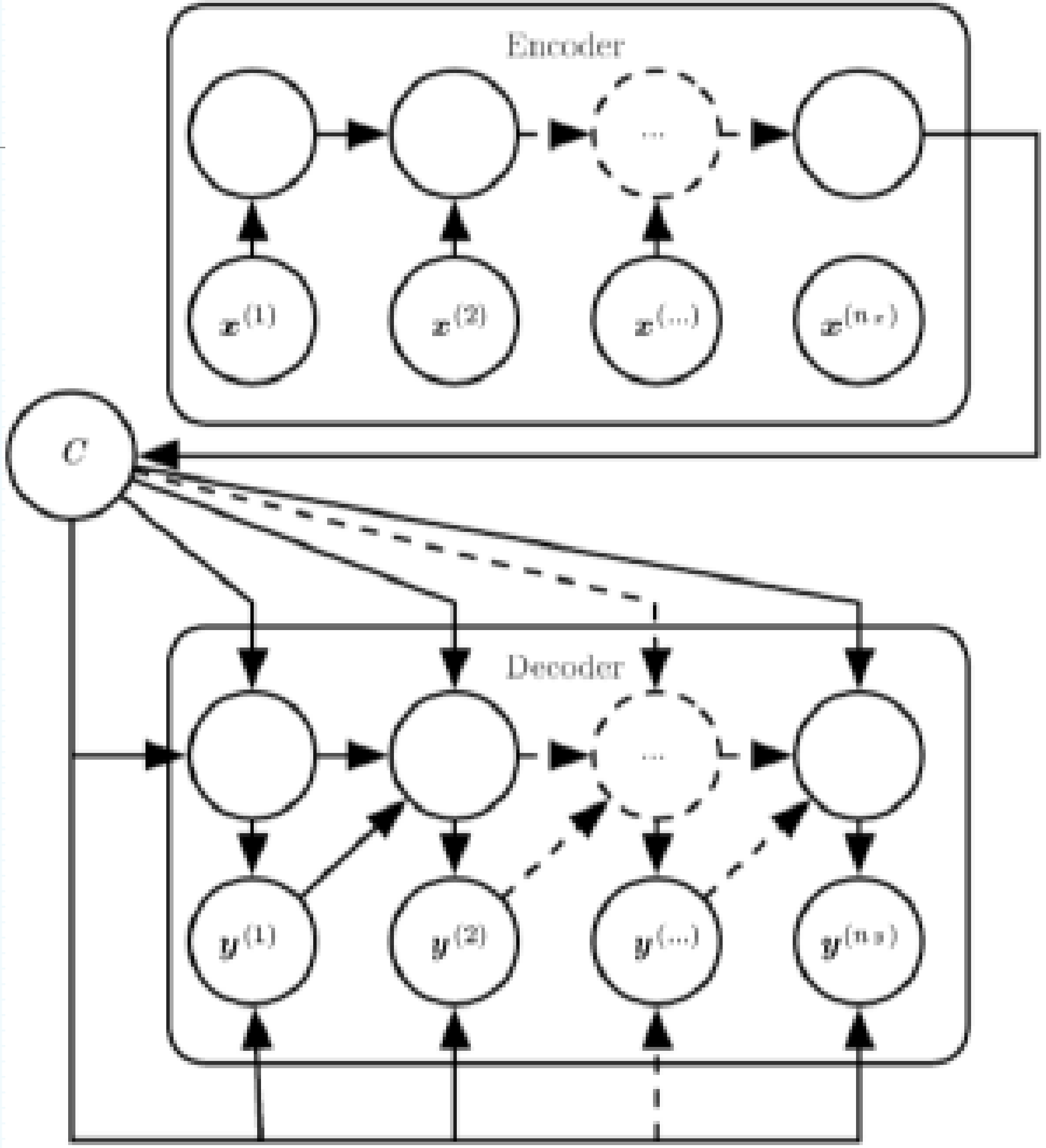
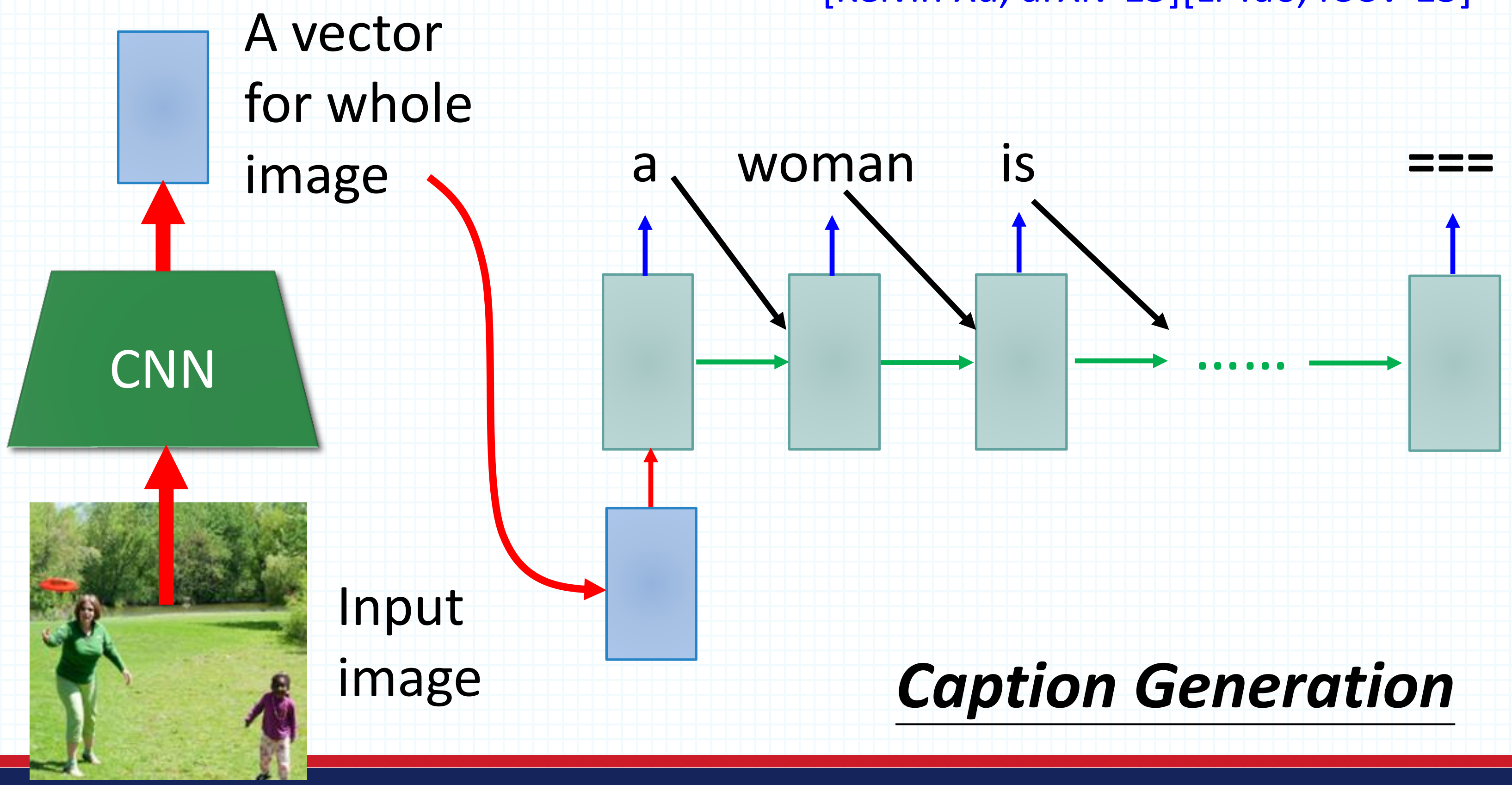


Image Caption Generation

- Input an image, but output a sequence of words

[Kelvin Xu, arXiv’15][Li Yao, ICCV’15]



Caption Generation