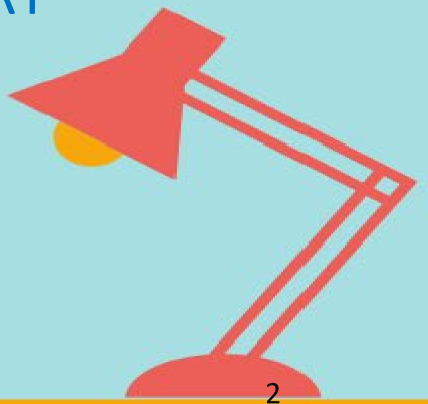
深度學習Deep Learning (10) 112-1

朱學亭老師

課程大綱

- W1-課程介紹/Introduction
- W2-Python/Colab and TensorFlow
- W3-Numpy/Pandas and PyTorch
- W4-Sklearn and 機器學習
- W5-神經網路, TensorFlow, PyTorch
- W6-載客熱點預測
- W7-自動光學檢查(AOI)-1
- W8-自動光學檢查(AOI)-2
- W9-Midterm presentation
- W10- PyTorch & **RNN**
- W11-GAN
- W12-Yolo
- W13-NLP1-Word2Vec
- W14-NLP2-Seq2Seq,Atention
- W15-NLP3-Transformer,BERT
- W16-AICUP 1
- W17-AICUP 2
- W18-Final presentation

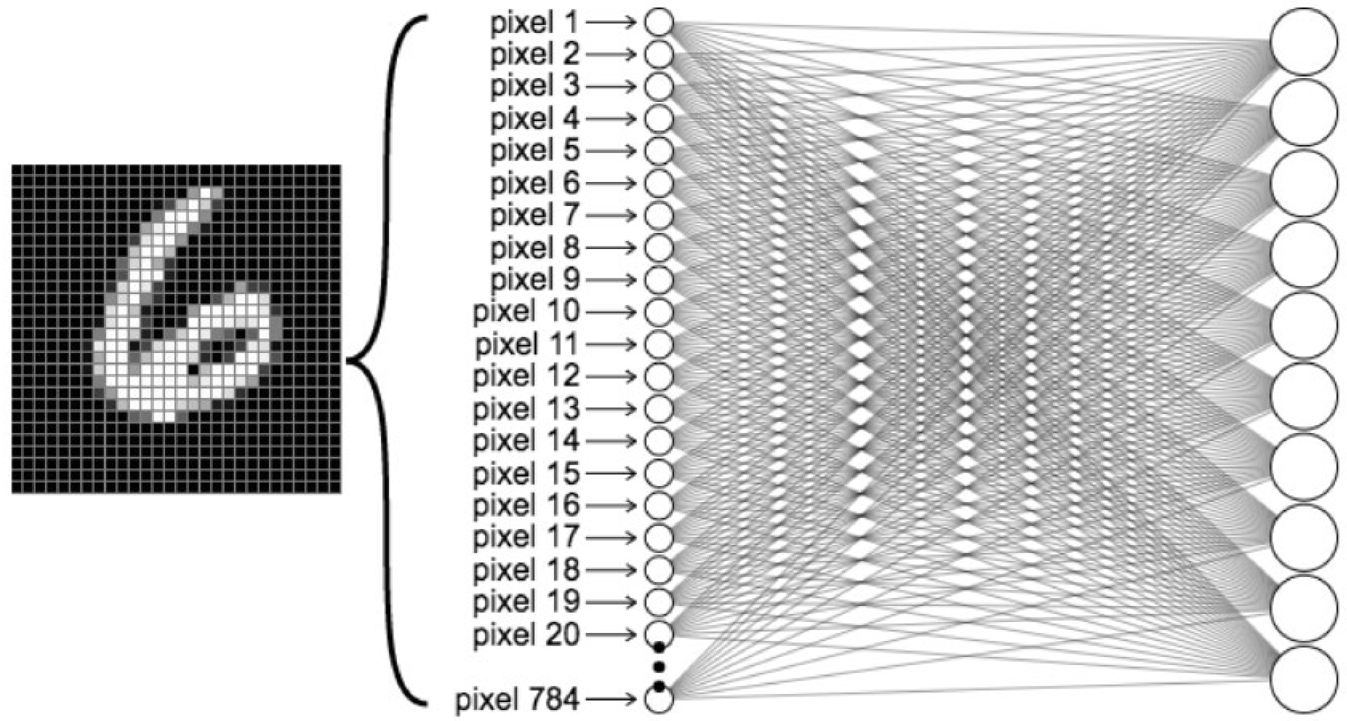


大綱

- Topic 1: CNN, RNN
- Topic 2: RNN

TOPIC 1: CNN

Feed Forward Neural Networks



NN with input and output layer, hidden layer is hidden :)

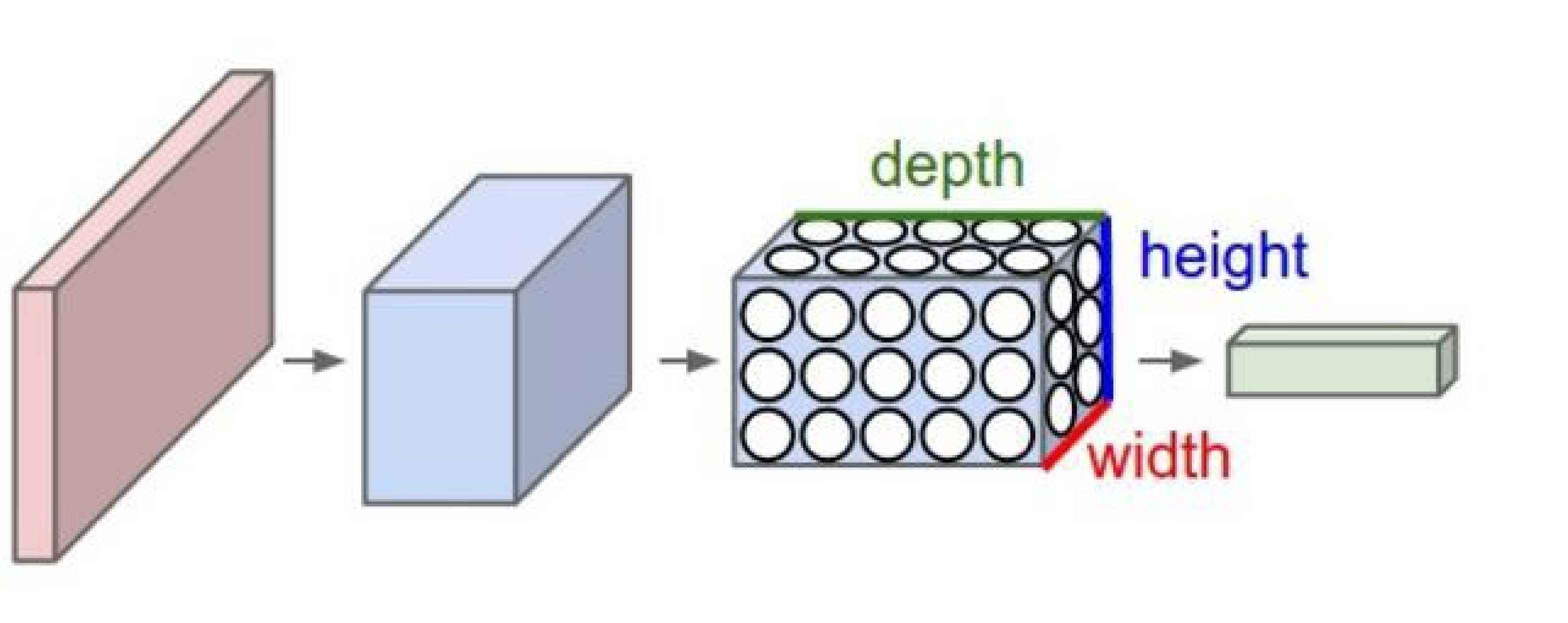
```
model = keras.models.Sequential([
  keras.layers.Flatten(input_shape=(28, 28)),
  keras.layers.Dense(128, activation='relu'),
  keras.layers.Dropout(0.2),
  keras.layers.Dense(10, activation='softmax')
])

model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])

model.fit(x_train, y_train, epochs=5)

model.evaluate(x_test, y_test)
```

Convolutional layer



Convolution

These 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

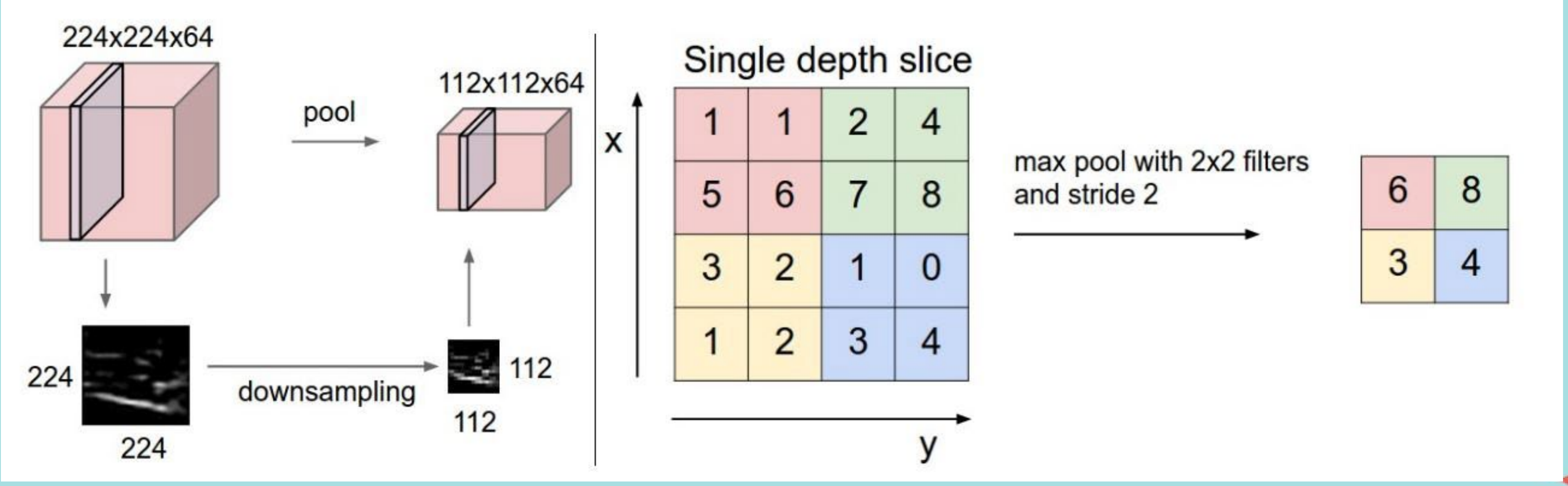
-1	1	-1
-1	1	-1
-1	1	-1

Filter 2

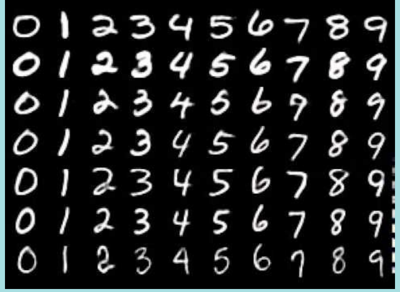
⋮ ⋮

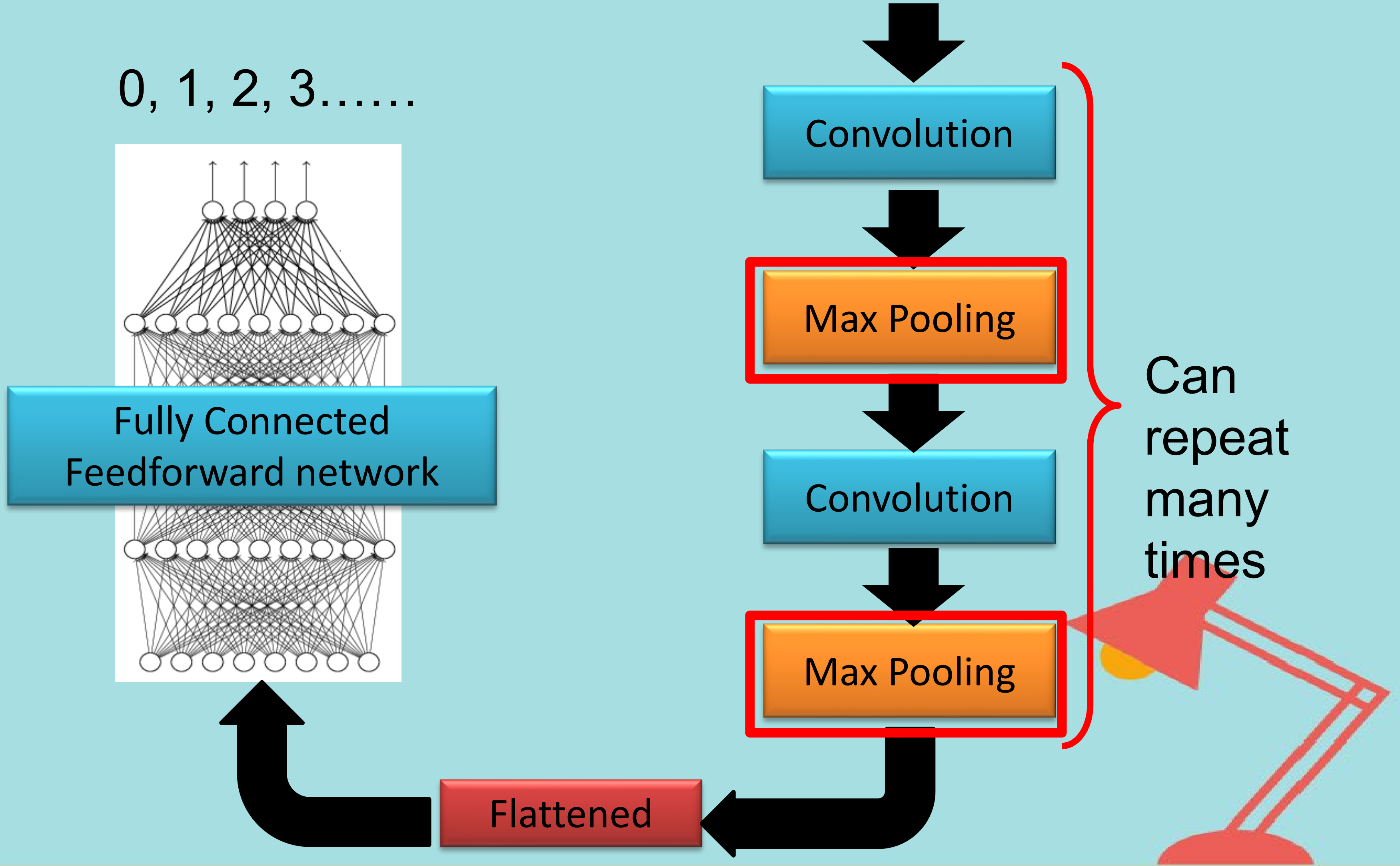
Each filter detects a small pattern (3 x 3).

Pooling layer



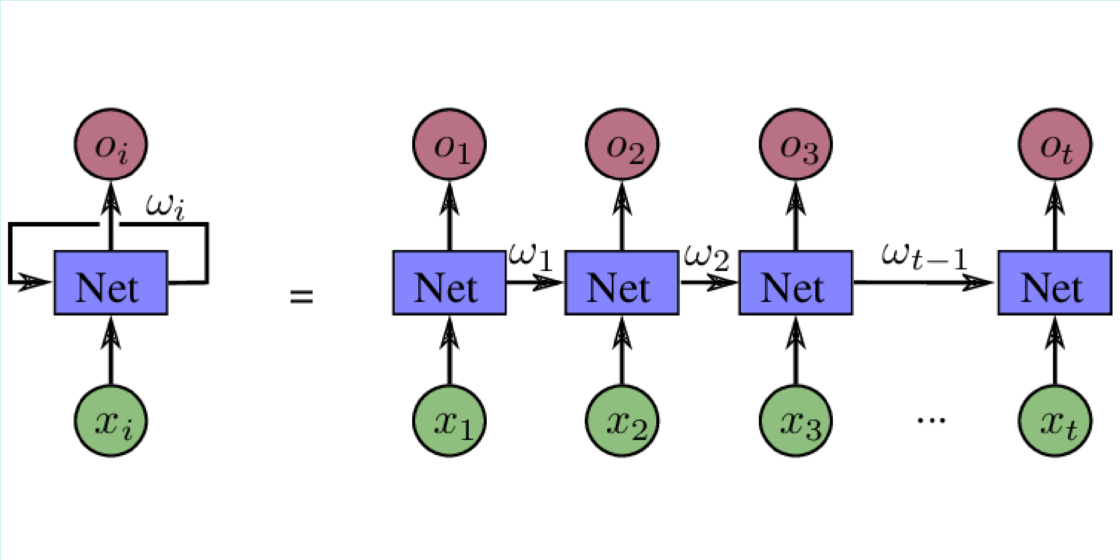
CNN-
Convolutional neural network

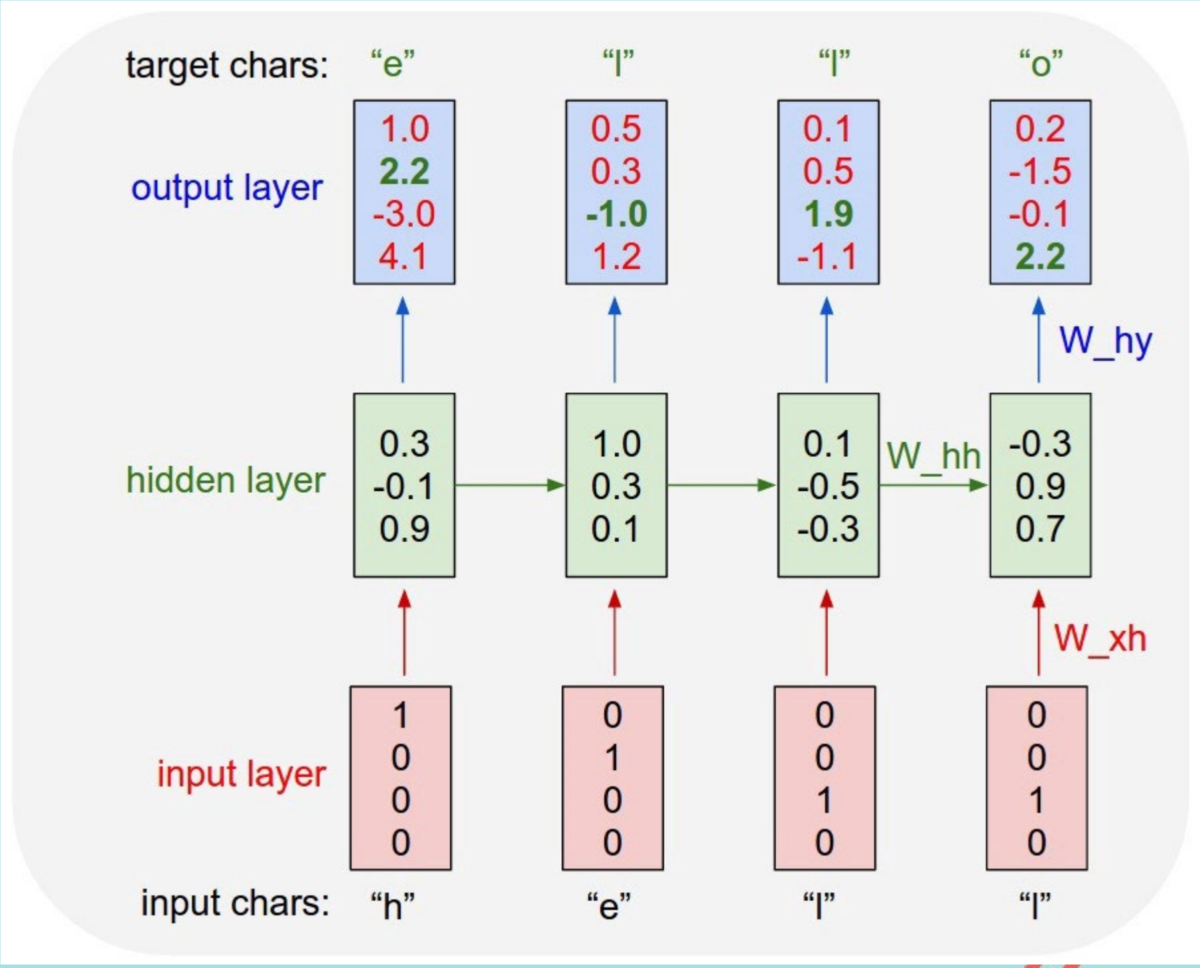




TOPIC 2: RNN

Recurrent Neural Network (RNN)
遞歸神經網路

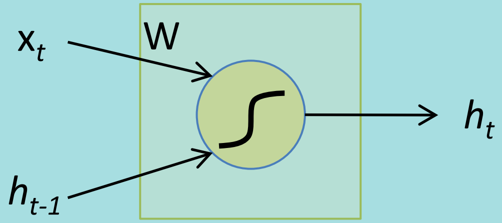




Types of RNN

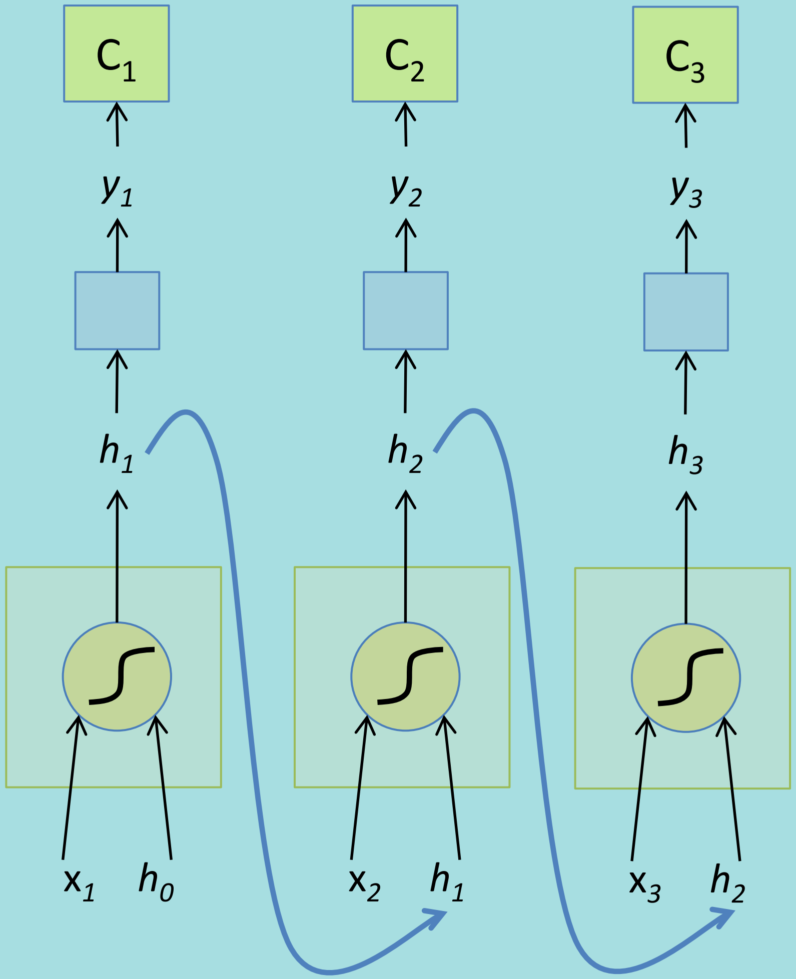
- Vanilla RNN
- LSTM- Long-Short Term Memory
- GRU-Gated Recurrent Unit
- Bi-directional RNNs

The Vanilla RNN Cell



$$h_t = \tanh W \begin{pmatrix} x_t \\ h_{t-1} \end{pmatrix}$$

The Vanilla RNN Forward

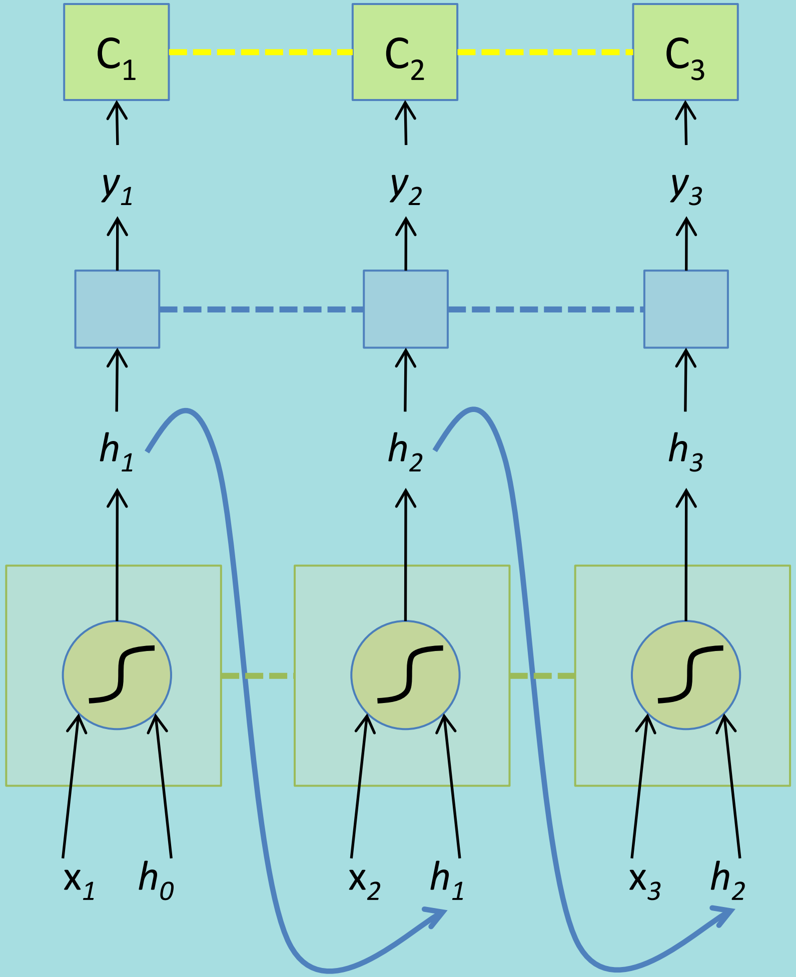


$$h_t = \tanh W \begin{pmatrix} x_t \\ h_{t-1} \end{pmatrix}$$

$$y_t = \mathrm{F}(h_t)$$

$$C_t = \mathrm{Loss}(y_t, \mathrm{GT}_t)$$

The Vanilla RNN Forward



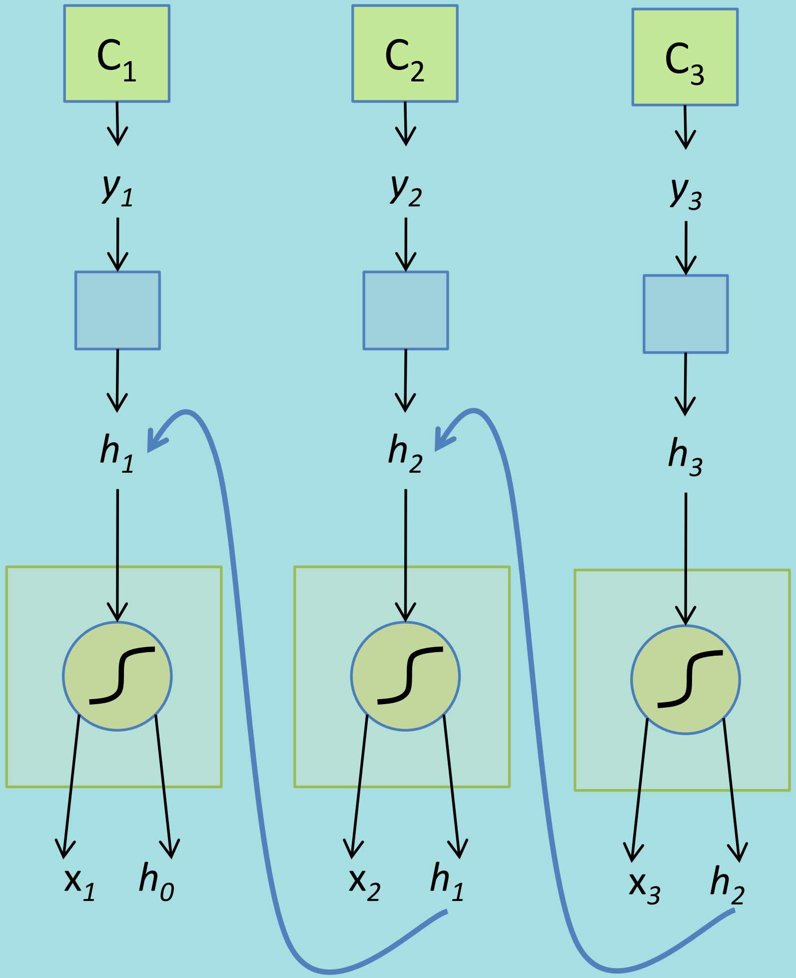
$$h_t = \tanh W \begin{pmatrix} x_t \\ h_{t-1} \end{pmatrix}$$

$$y_t = \mathrm{F}(h_t)$$

$$C_t = \mathrm{Loss}(y_t, \mathrm{GT}_t)$$

------ indicates shared weights

The Vanilla RNN Backward



$$h_t = \tanh W \begin{pmatrix} x_t \\ h_{t-1} \end{pmatrix}$$

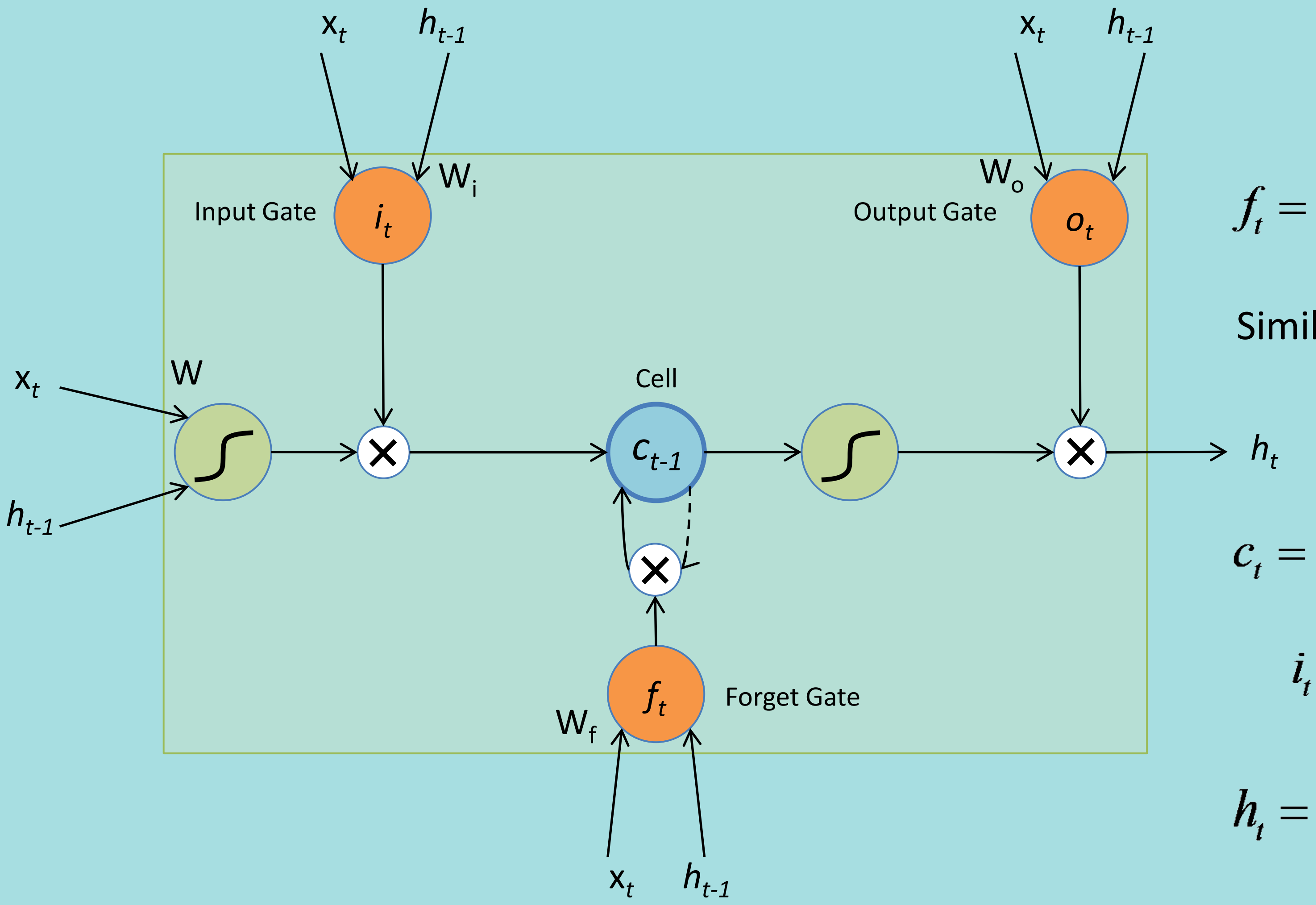
$$y_t = \mathrm{F}(h_t)$$

$$C_t = \mathrm{Loss}(y_t, \mathrm{GT}_t)$$

$$\frac{\partial C_t}{\partial h_1} = \left(\frac{\partial C_t}{\partial y_t}\right)\left(\frac{\partial y_t}{\partial h_1}\right)$$

$$= \left(\frac{\partial C_t}{\partial y_t}\right)\left(\frac{\partial y_t}{\partial h_t}\right)\left(\frac{\partial h_t}{\partial h_{t-1}}\right)\cdots\left(\frac{\partial h_2}{\partial h_1}\right)$$

The Popular LSTM Cell



$$f_t = \sigma\left(W_f \begin{pmatrix} x_t \\ h_{t-1} \end{pmatrix} + b_f\right)$$

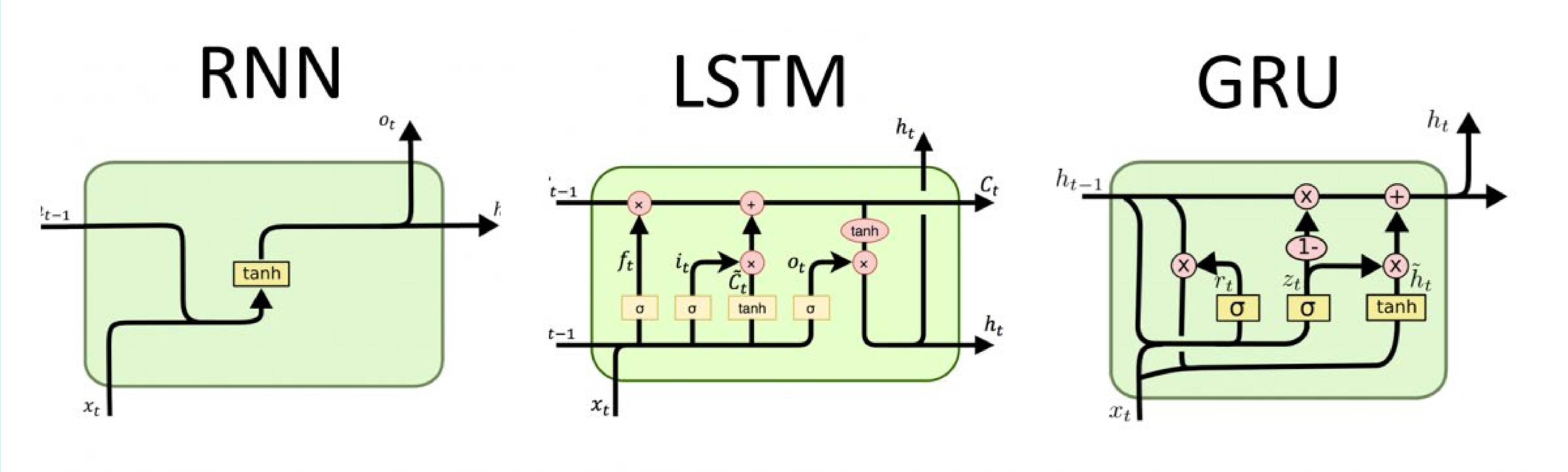
Similarly for i_t, o_t

$$c_t = f_t \otimes c_{t-1} + i_t \otimes \tanh W \begin{pmatrix} x_t \\ h_{t-1} \end{pmatrix}$$

$$h_t = o_t \otimes \tanh c_t$$

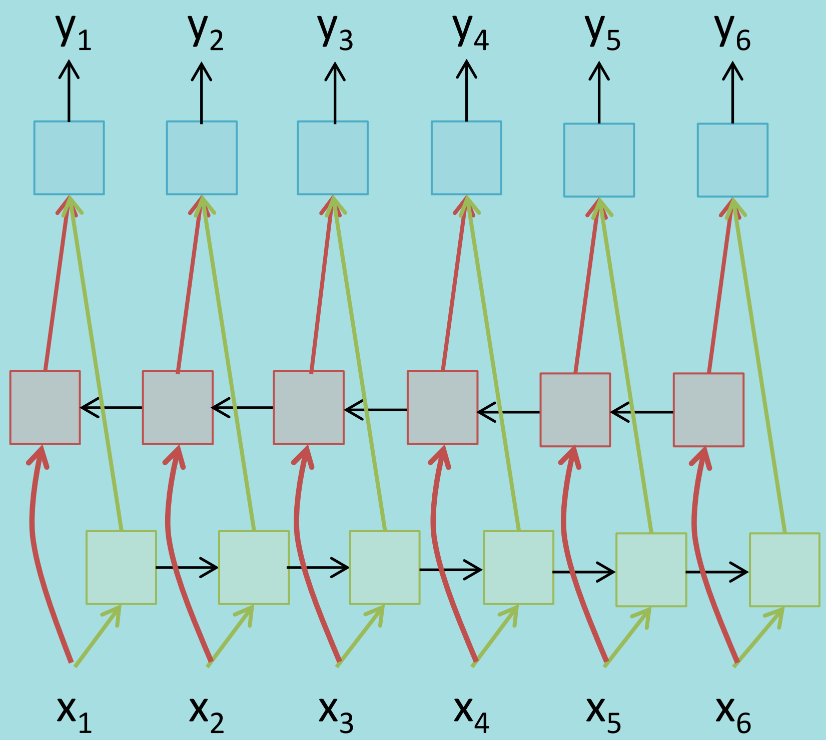
* Dashed line indicates time-lag

GRU RNN



Bi-directional RNNs

- RNNs can process the input sequence in forward and in the reverse direction



- Popular in speech recognition

Thanks! Q&A