Recurrent Neural Networks & Long Short-Term Memory models

한연지, 임영수

AI Lab
Yeon-Jee Han, Yeong-Su Lim

한양대학교
HANYANG UNIVERSITY

Presentation Flow

- RNN 기본동작
- Stacked RNN
- LSTM 기본동작
- Tutorial1 - Character-level Model
- CRNN
- Tutorial2 - Image Captioning 시연동영상

RNN

01

02

03

04

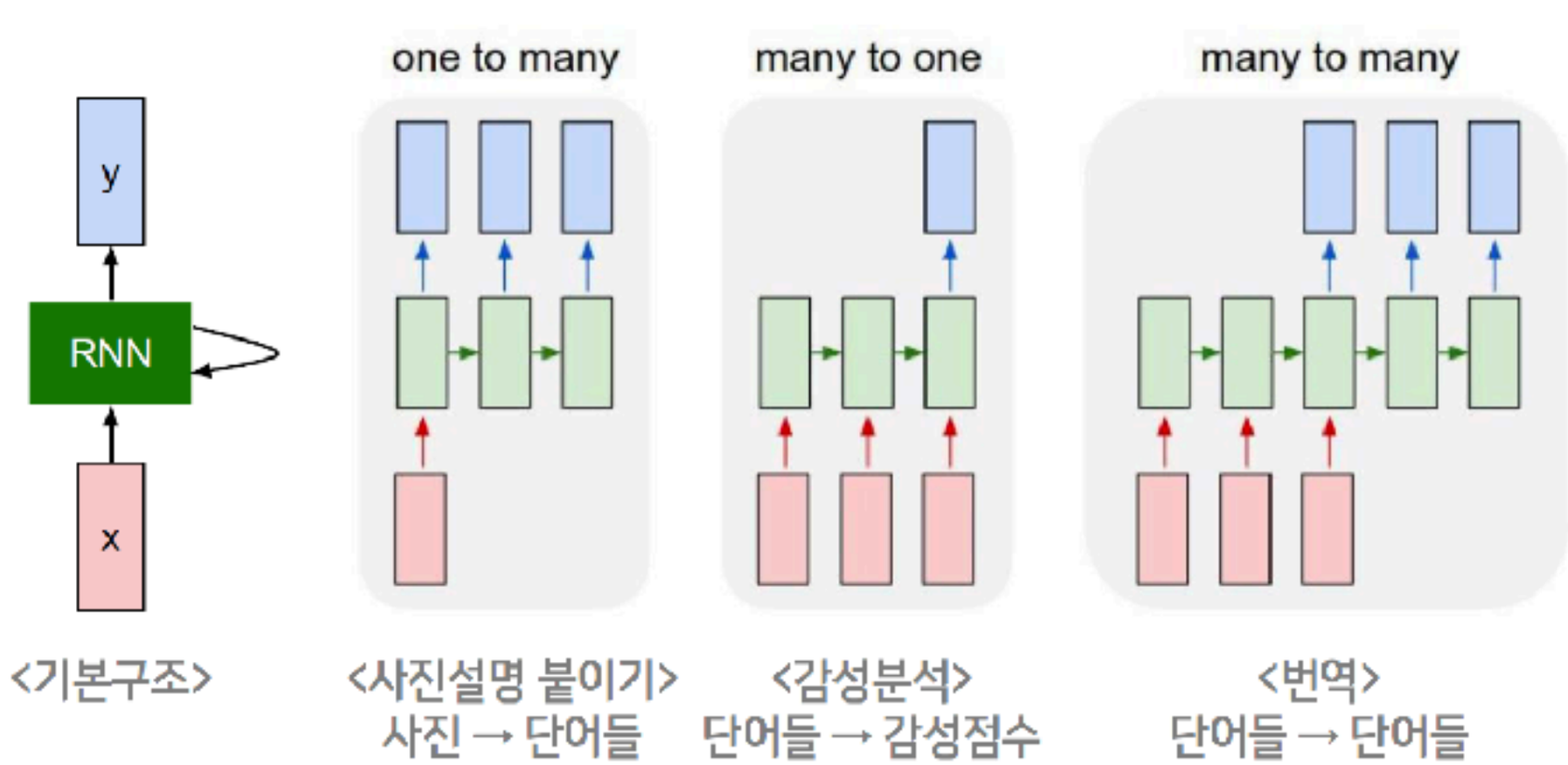
05

❖ Recurrent Neural Networks(순환신경망)

■ 히든 노드가 **directed cycle**을 형성하는 인공신경망의 한 종류

■ 음성, 문자 등 순차적으로 등장 하는 데이터 처리에 적합

■ 다양하고 유연하게 네트워크 구성 가능



<기본구조>

<사진설명 붙이기>
사진 → 단어들

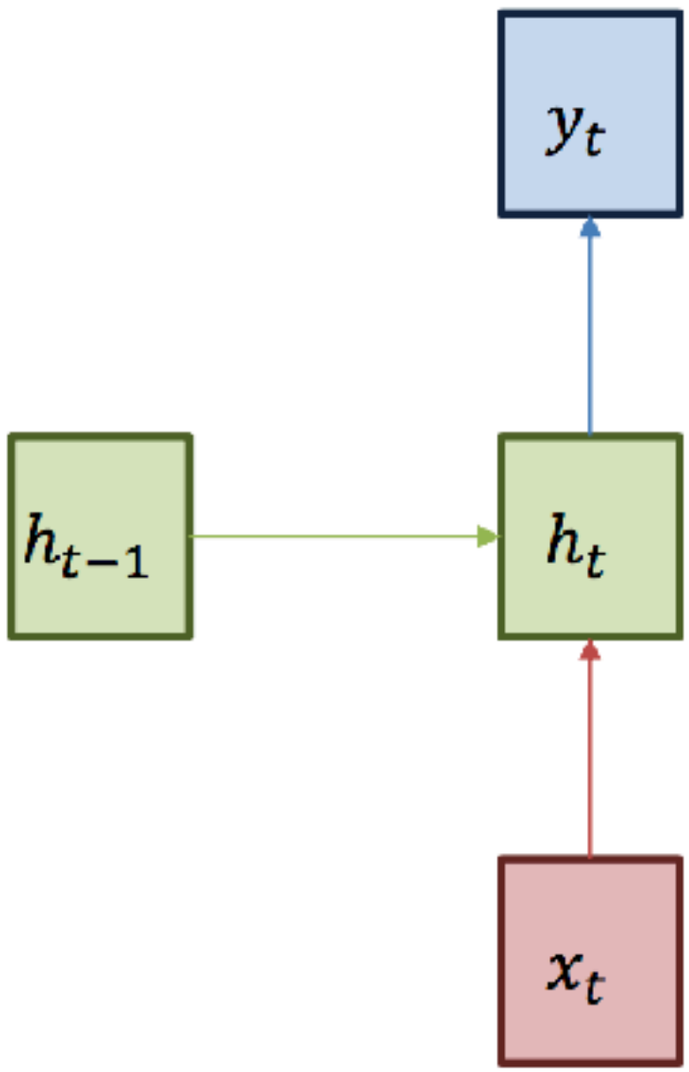
<감성분석>
단어들 → 감성점수

<번역>
단어들 → 단어들

RNN

01
02
03
04
05

❖ RNN 기본동작



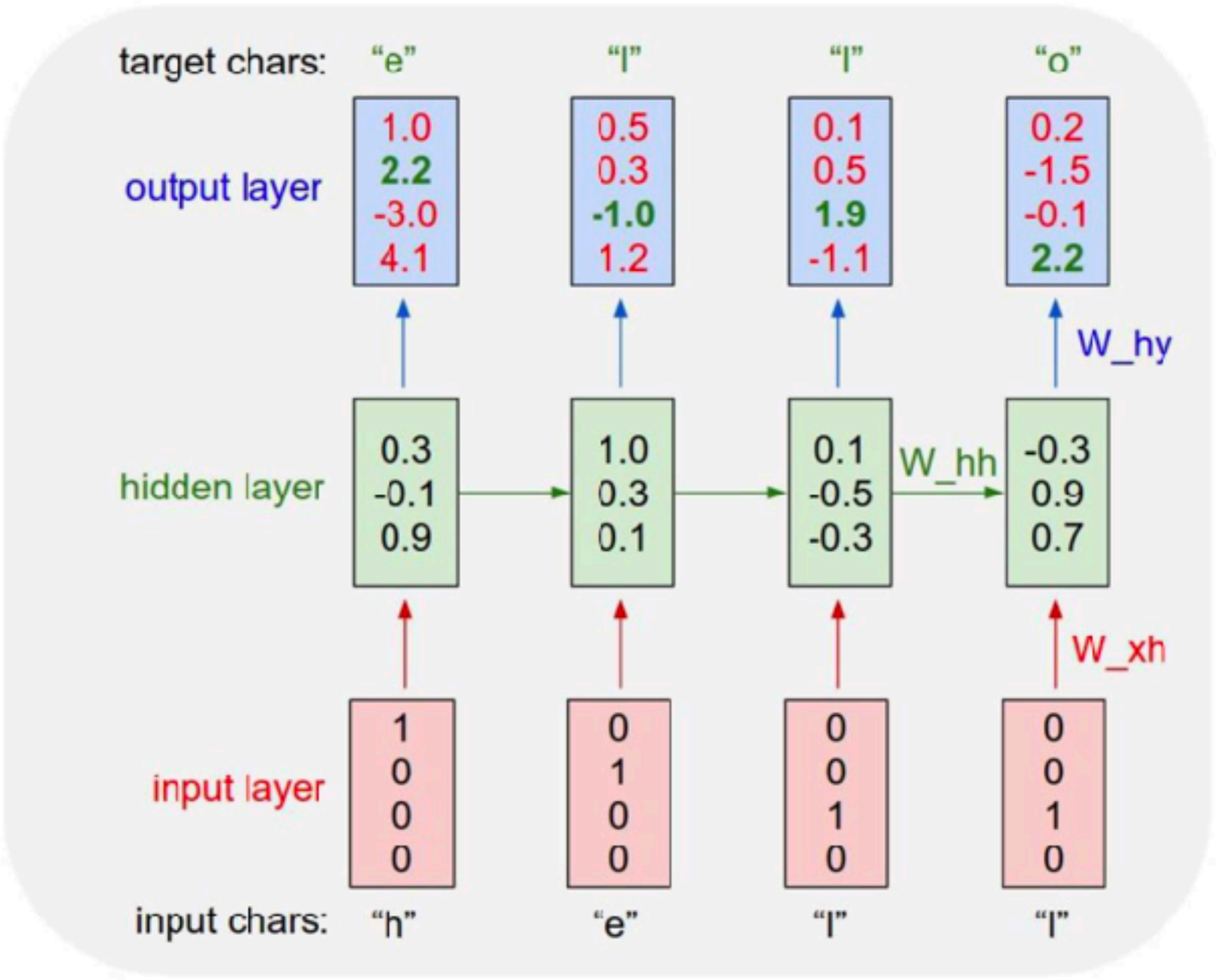
$$y_t = W_{hy} h_t + b_y$$

$$h_t = \tanh(W_{hh} h_{t-1} + W_{xh} x_t + b_h)$$

RNN

01
02
03
04
05

❖ RNN 기본동작



RNN

01

02

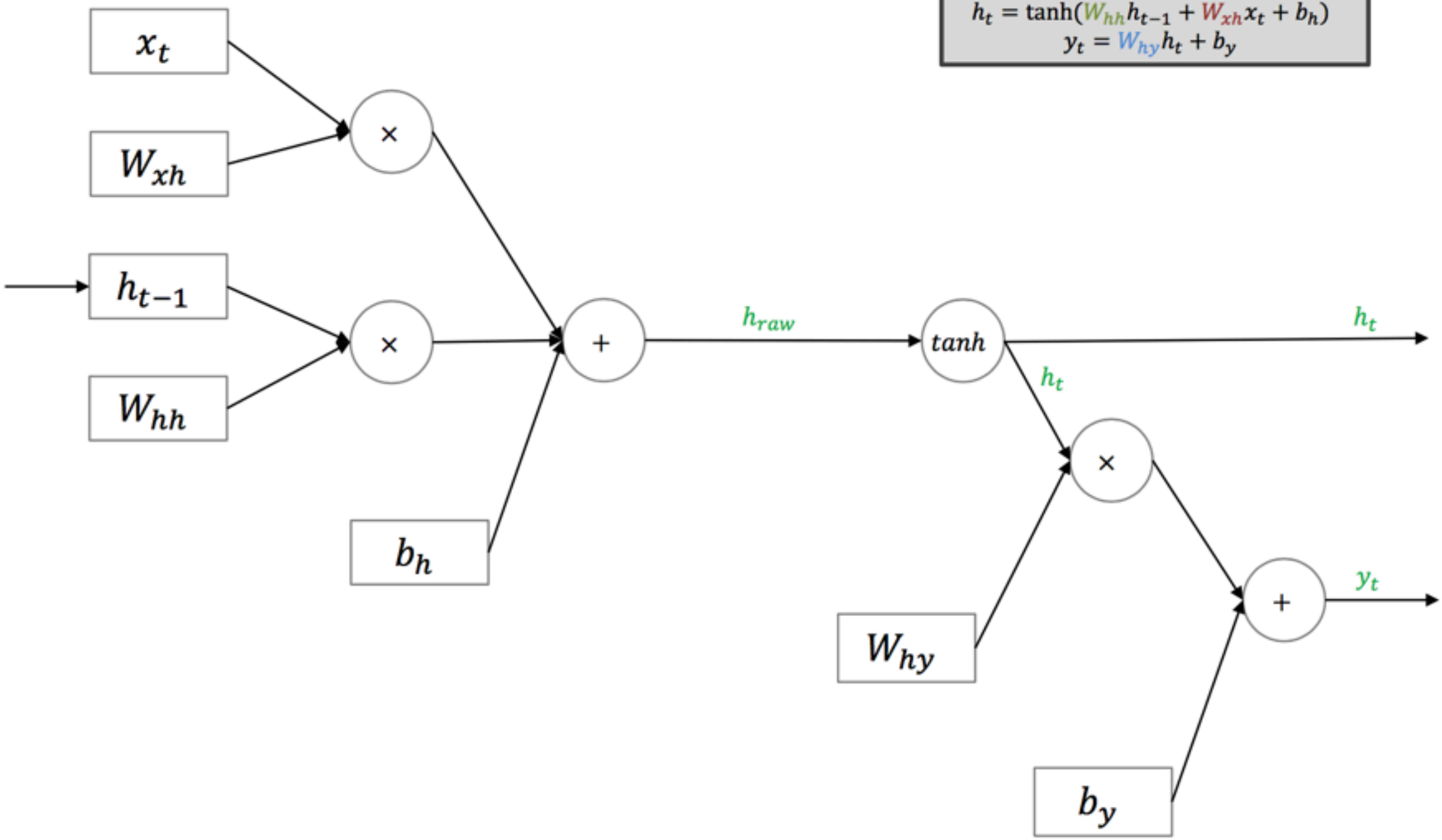
03

04

05

❖ RNN forward pass

$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t + b_h)$$
$$y_t = W_{hy}h_t + b_y$$



RNN

01

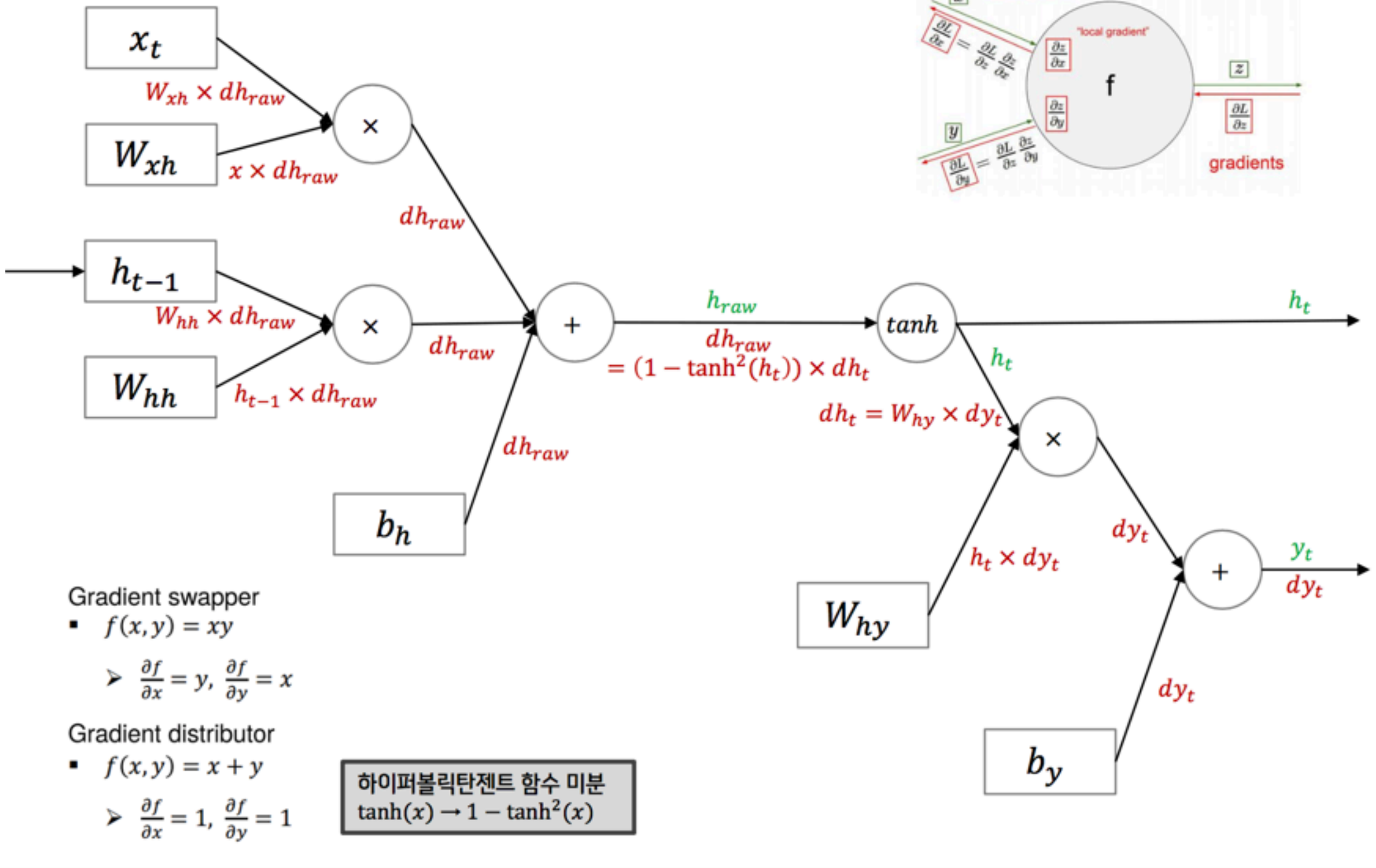
02

03

04

05

❖ RNN backward pass



Gradient swapper

- $f(x, y) = xy$
  - ➢ $\frac{\partial f}{\partial x} = y,\ \frac{\partial f}{\partial y} = x$

Gradient distributor

- $f(x, y) = x + y$
  - ➢ $\frac{\partial f}{\partial x} = 1,\ \frac{\partial f}{\partial y} = 1$

하이퍼볼릭탄젠트 함수 미분
$\tanh(x) \rightarrow 1 - \tanh^2(x)$

RNN

01

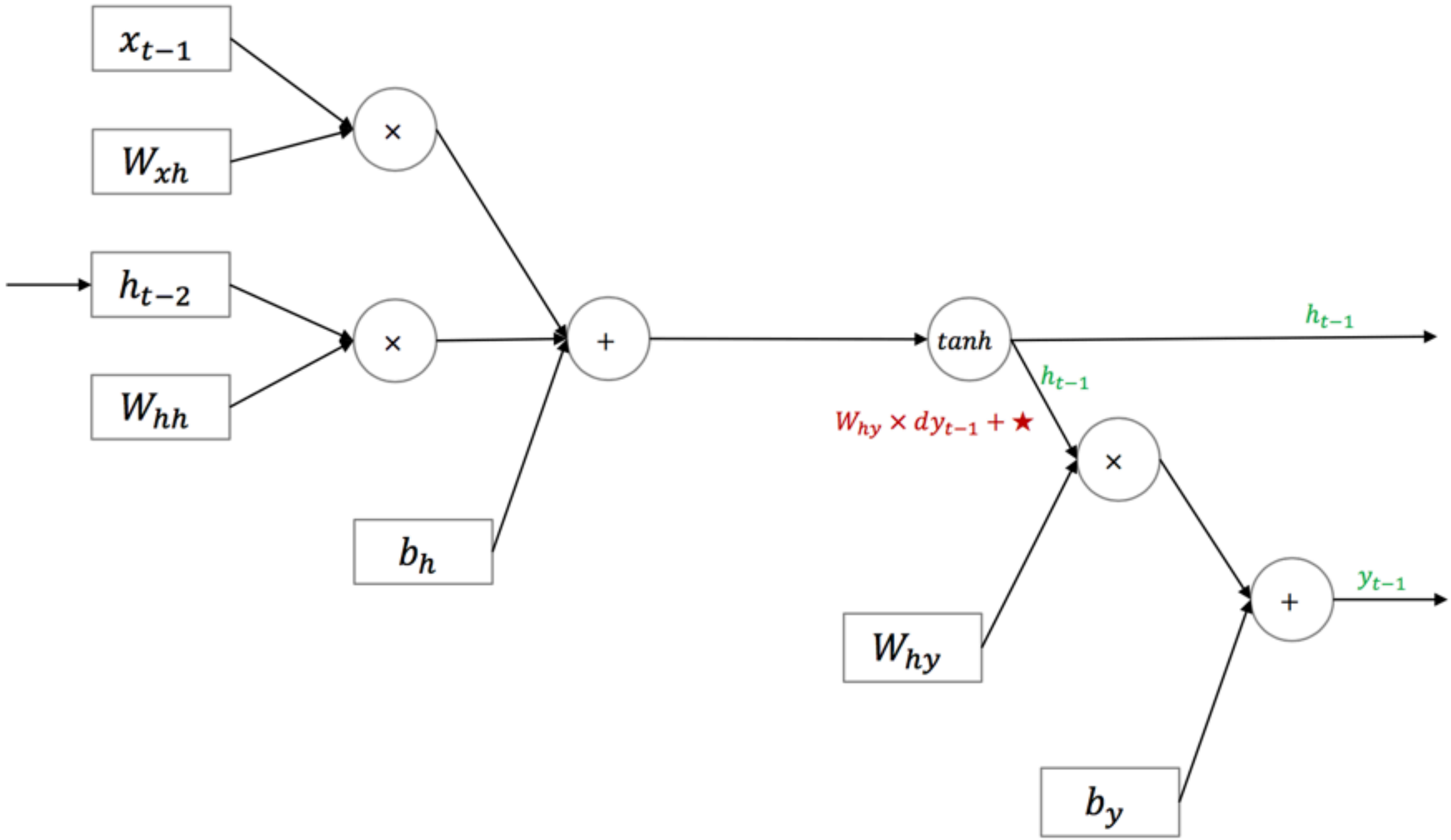
02

03

04

05

❖ RNN backward pass



RNN

01

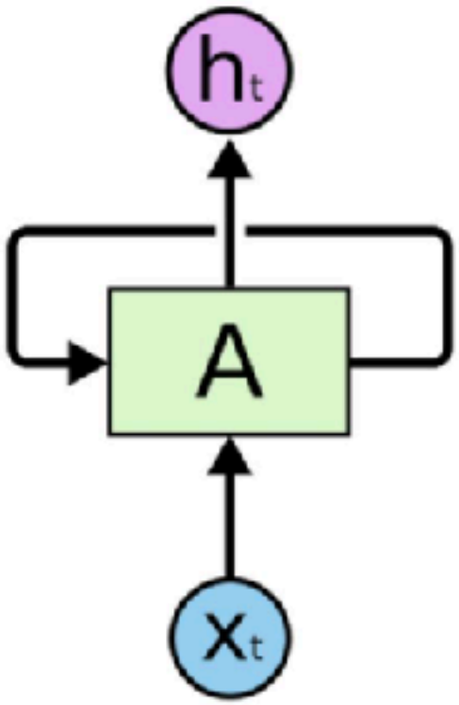
02

03

04

05

❖ RNN in TensorFlow




```
cell = tf.contrib.rnn.BasicRNNCell(num_units=hidden_size)

...
outputs, _states = tf.nn.dynamic_rnn(cell, x_data, dtype=tf.float32)
```

RNN

01

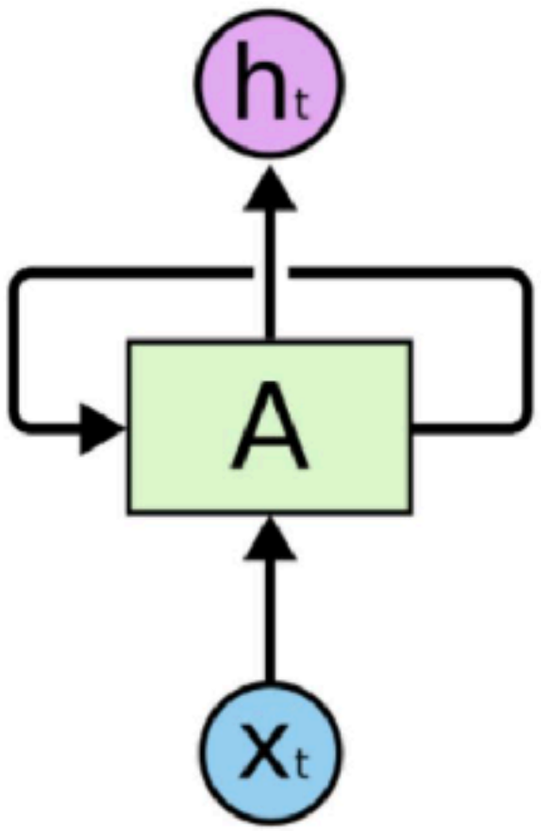
02

03

04

05

❖ One node : 4(input-dim) in 2 (hidden_size)



[[[1,0,0,0]]]
shape=(1,1,4)

```
# One hot encoding
h = [1, 0, 0, 0]
e = [0, 1, 0, 0]
l = [0, 0, 1, 0]
o = [0, 0, 0, 1]
```

RNN

01

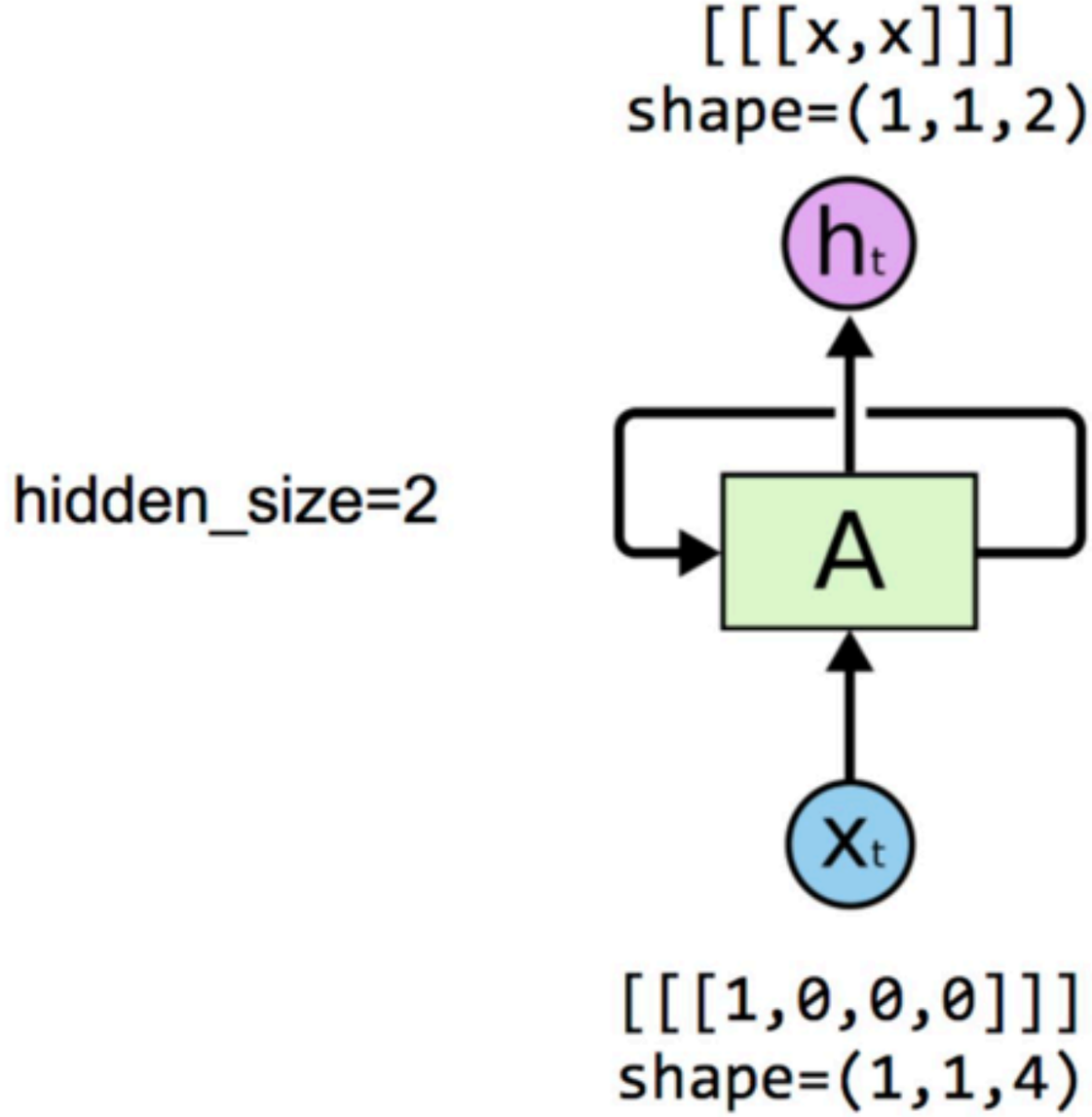
02

03

04

05

❖ One node : 4(input-dim) in 2 (hidden_size)




```
# One hot encoding
h = [1, 0, 0, 0]
e = [0, 1, 0, 0]
l = [0, 0, 1, 0]
o = [0, 0, 0, 1]
```

RNN

01

❖ Unfolding to n sequences

02

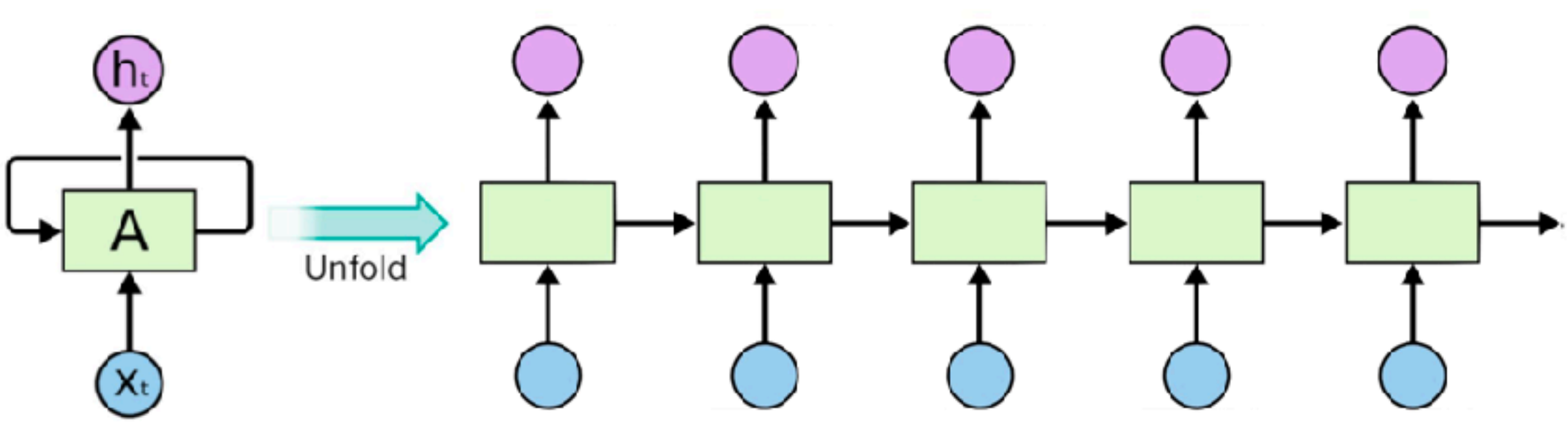
03

Hidden_size=2
sequence_length=5

04

05

shape=(1,5,2): [[[x,x], [x,x], [x,x], [x,x], [x,x]]]



shape=(1,5,4): [[[1,0,0,0], [0,1,0,0], [0,0,1,0], [0,0,1,0], [0,0,0,1]]]
h e l l o

RNN

01

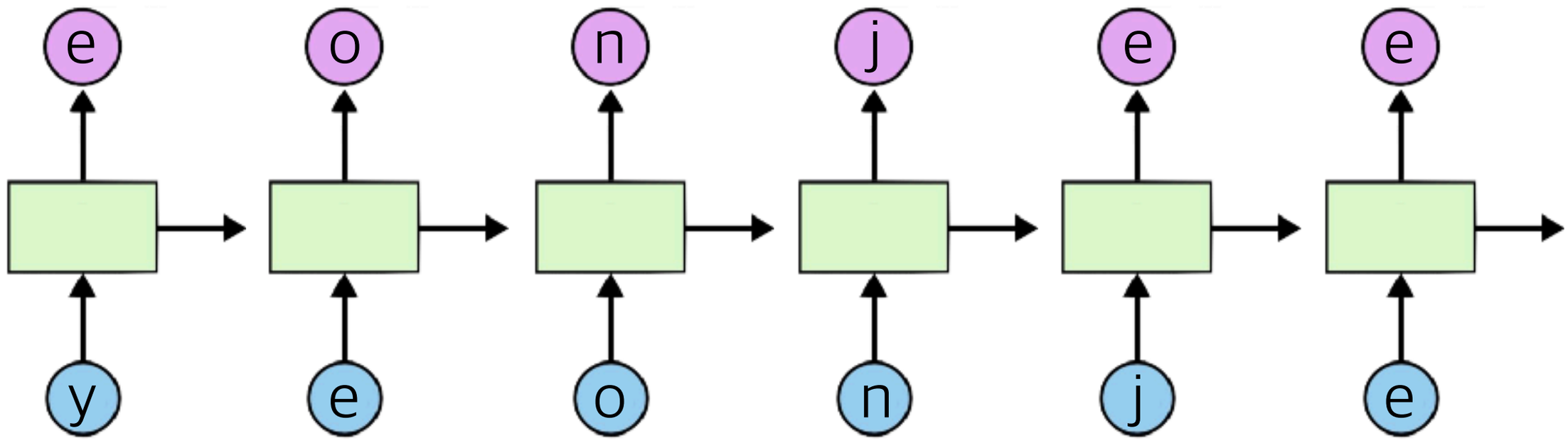
02

03

04

05

❖ RNN 'yeonjee'



RNN

01

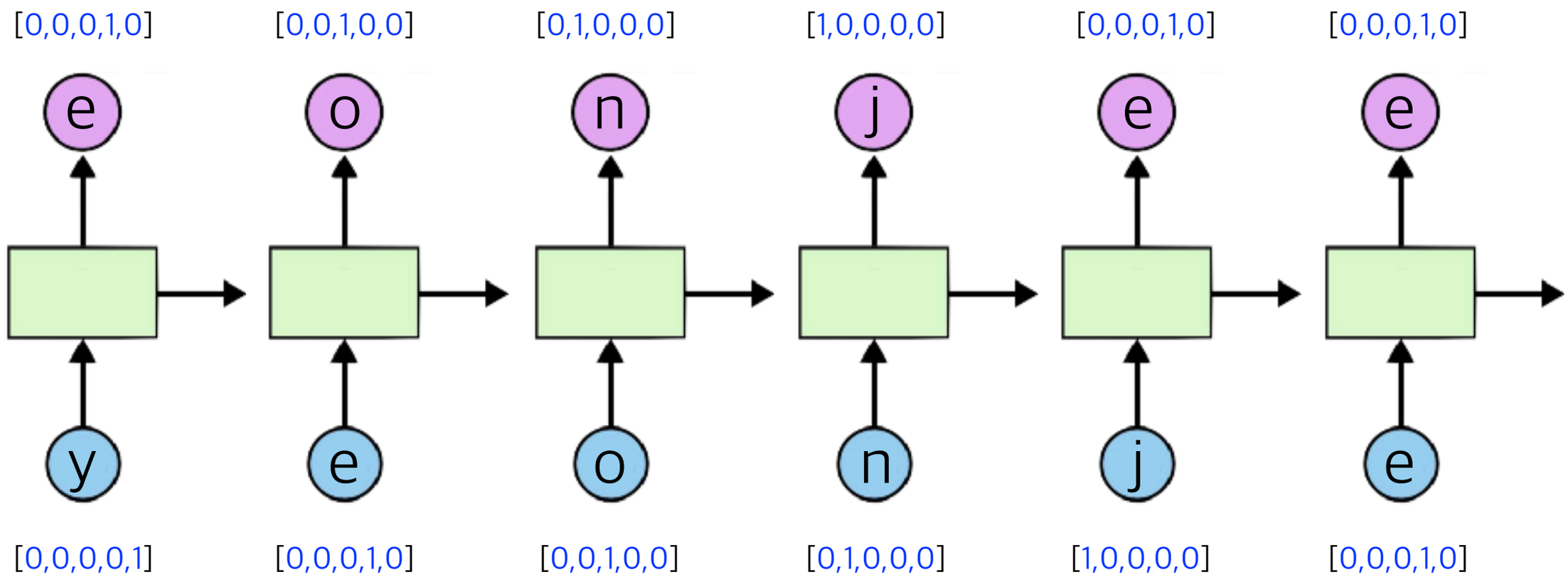
02

03

04

05

❖ RNN 'yeonjee'



RNN

01

02

03

04

05

❖ Data Creation

```
idx2char = ['y', 'e', 'o', 'n', 'j']
# Teach hello: yeonje -> eonjee
x_data = [[0, 1, 2, 3, 4, 1]]   # yeonje
x_one_hot = [[[0, 0, 0, 0, 1],   # y 0
              [0, 0, 0, 1, 0],   # e 1
              [0, 0, 1, 0, 0],   # o 2
              [0, 1, 0, 0, 0],   # n 3
              [1, 0, 0, 1, 0],   # j 4
              [0, 0, 0, 1, 0]]]  # e 1

y_data = [[1, 2, 3, 4, 1, 1]]    # eonjee
```

RNN

01

02

03

04

05

❖ Feed to RNN

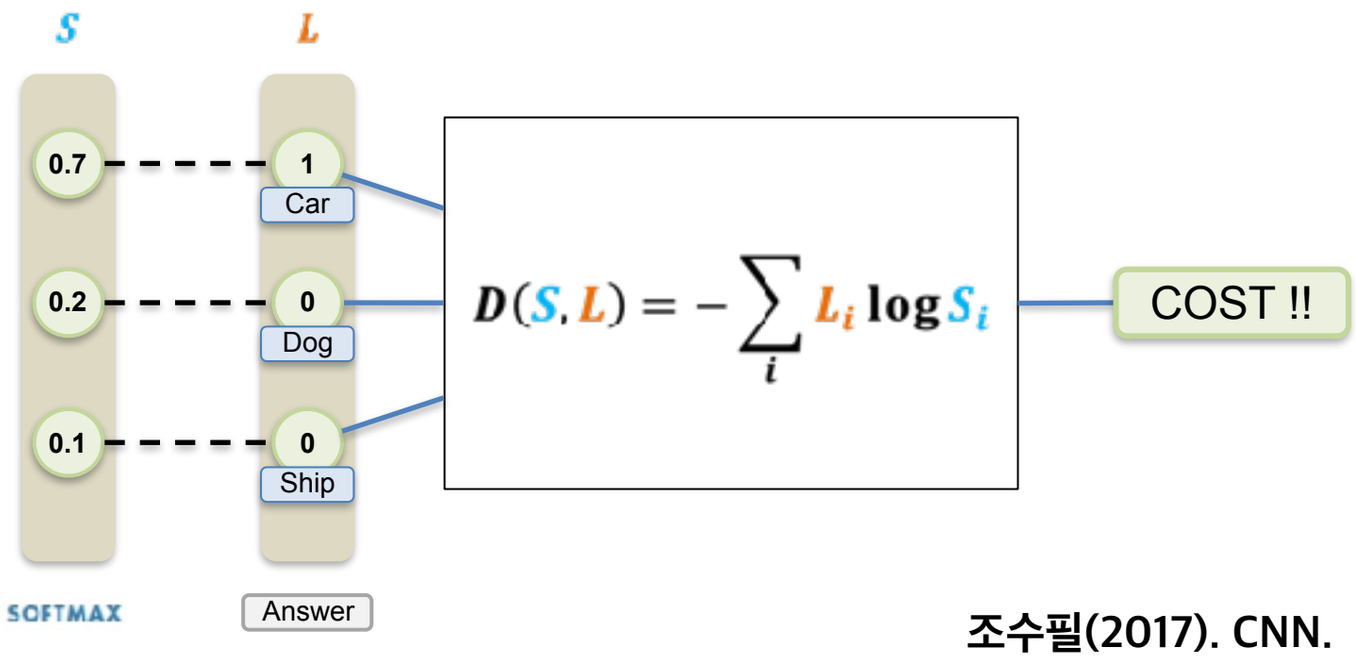
```
cell =
tf.contrib.rnn.BasicLSTMCell(num_units=hidde
n_size, state_is_tuple=True)
initial_state = cell.zero_state(batch_size,
tf.float32)
outputs, _states = tf.nn.dynamic_rnn(
   cell, X, initial_state=initial_state,
dtype=tf.float32)
# FC layer
X_for_fc = tf.reshape(outputs, [-1,
hidden_size])

# fc_w = tf.get_variable("fc_w", [hidden_size,
num_classes])
# fc_b = tf.get_variable("fc_b", [num_classes])
# outputs = tf.matmul(X_for_fc, fc_w) + fc_b
outputs = tf.contrib.layers.fully_connected(
   inputs=X_for_fc, num_outputs=num_classes,
activation_fn=None)
```

RNN

❖ Cost : sequence_loss (Cross-entropy)

```
weights = tf.ones([batch_size,
sequence_length])
sequence_loss =
tf.contrib.seq2seq.sequence_loss(
    logits=outputs, targets=Y, weights=weights)
loss = tf.reduce_mean(sequence_loss)
train =
tf.train.AdamOptimizer(learning_rate=learning_
rate).minimize(loss)
```



조수필(2017). CNN.

RNN

01

02

03

04

05

❖ Training

```
prediction = tf.argmax(outputs, axis=2)

with tf.Session() as sess:
    sess.run(tf.global_variables_initializer())
    for i in range(50):
        l, _ = sess.run([loss, train], feed_dict={X:
x_one_hot, Y: y_data})
        result = sess.run(prediction,
feed_dict={X: x_one_hot})
        print(i, "loss:", l, "prediction: ", result,
"true Y: ", y_data)
```

```
================ RESTART: C:/Users/user/Desktop/yeonjeeRNN.py ================
0 loss: 1.60616 prediction:  [[1 1 1 1 1 1]] True Y:  [[1, 2, 3, 4, 1, 1]]
        Prediction str:  eeeeee
1 loss: 1.46487 prediction:  [[1 1 1 1 1 1]] true Y:  [[1, 2, 3, 4, 1, 1]]
        Prediction str:  eeeeee
2 loss: 1.34492 prediction:  [[1 1 1 1 1 1]] true Y:  [[1, 2, 3, 4, 1, 1]]
        Prediction str:  eeeeee
3 loss: 1.22156 prediction:  [[1 1 1 1 1 1]] true Y:  [[1, 2, 3, 4, 1, 1]]
        Prediction str:  eeeeee
4 loss: 1.10087 prediction:  [[1 1 1 1 1 1]] true Y:  [[1, 2, 3, 4, 1, 1]]
        Prediction str:  eeeeee
5 loss: 0.968749 prediction:  [[1 1 3 4 1 1]] true Y:  [[1, 2, 3, 4, 1, 1]]
        Prediction str:  eenjee
6 loss: 0.841272 prediction:  [[1 1 3 4 1 1]] true Y:  [[1, 2, 3, 4, 1, 1]]
        Prediction str:  eenjee
7 loss: 0.737312 prediction:  [[1 1 3 4 1 1]] true Y:  [[1, 2, 3, 4, 1, 1]]
        Prediction str:  eenjee
8 loss: 0.639483 prediction:  [[1 1 3 4 1 1]] true Y:  [[1, 2, 3, 4, 1, 1]]
        Prediction str:  eenjee
9 loss: 0.540761 prediction:  [[1 1 3 4 1 1]] true Y:  [[1, 2, 3, 4, 1, 1]]
        Prediction str:  eenjee
10 loss: 0.446995 prediction:  [[1 1 3 4 1 1]] true Y:  [[1, 2, 3, 4, 1, 1]]
        Prediction str:  eenjee
11 loss: 0.364881 prediction:  [[1 2 3 4 1 1]] true Y:  [[1, 2, 3, 4, 1, 1]]
        Prediction str:  eenjee
12 loss: 0.297175 prediction:  [[1 2 3 4 1 1]] true Y:  [[1, 2, 3, 4, 1, 1]]
        Prediction str:  eenjee
13 loss: 0.238987 prediction:  [[1 2 3 4 1 1]] true Y:  [[1, 2, 3, 4, 1, 1]]
        Prediction str:  eenjee
14 loss: 0.187764 prediction:  [[1 2 3 4 1 1]] true Y:  [[1, 2, 3, 4, 1, 1]]
        Prediction str:  eenjee
15 loss: 0.14653 prediction:  [[1 2 3 4 1 1]] true Y:  [[1, 2, 3, 4, 1, 1]]
        Prediction str:  eenjee
16 loss: 0.115158 prediction:  [[1 2 3 4 1 1]] true Y:  [[1, 2, 3, 4, 1, 1]]
        Prediction str:  eenjee
17 loss: 0.0908733 prediction:  [[1 2 3 4 1 1]] true Y:  [[1, 2, 3, 4, 1, 1]]
        Prediction str:  eenjee
18 loss: 0.0716325 prediction:  [[1 2 3 4 1 1]] true Y:  [[1, 2, 3, 4, 1, 1]]
        Prediction str:  eenjee
19 loss: 0.0565701 prediction:  [[1 2 3 4 1 1]] true Y:  [[1, 2, 3, 4, 1, 1]]
```

Really long sentence?

```
sentence = ("if you want to build a ship, don't drum up people together to"
            "collect wood and don't assign them tasks and work, but rather"
            "teach them to long for the endless immensity of the sea.")
```

Stacked RNN

❖ Making dataset

```
char_set = list(set(sentence))
char_dic = {w: i for i, w in enumerate(char_set)}

dataX = []
dataY = []
for i in range(0, len(sentence) - seq_length):
    x_str = sentence[i:i + seq_length]
    y_str = sentence[i + 1: i + seq_length + 1]
    print(i, x_str, '->', y_str)

    x = [char_dic[c] for c in x_str]  # x str to index
    y = [char_dic[c] for c in y_str]  # y str to index

    dataX.append(x)
    dataY.append(y)
```

```
# training dataset
0 if you wan -> f you want
1 f you want ->  you want
2  you want  -> you want t
3 you want t -> ou want to
…
168  of the se -> of the sea
169 of the sea -> f the sea.
```

Stacked RNN

01

02

03

04

05

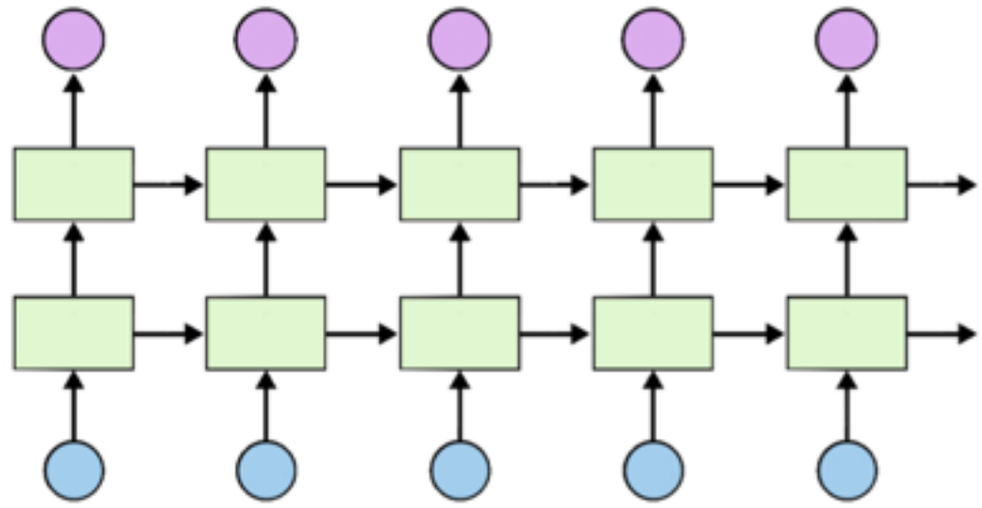
❖ Stacked RNN

```
X = tf.placeholder(tf.int32, [None, seq_length])
Y = tf.placeholder(tf.int32, [None, seq_length])

# One-hot encoding
X_one_hot = tf.one_hot(X, num_classes)
print(X_one_hot)  # check out the shape

# Make a lstm cell with hidden_size (each unit output vector size)
cell = rnn.BasicLSTMCell(hidden_size, state_is_tuple=True)
cell = rnn.MultiRNNCell([cell] * 2, state_is_tuple=True)

# outputs: unfolding size x hidden size, state = hidden size
outputs, _states = tf.nn.dynamic_rnn(cell, X_one_hot, dtype=tf.float32)
```



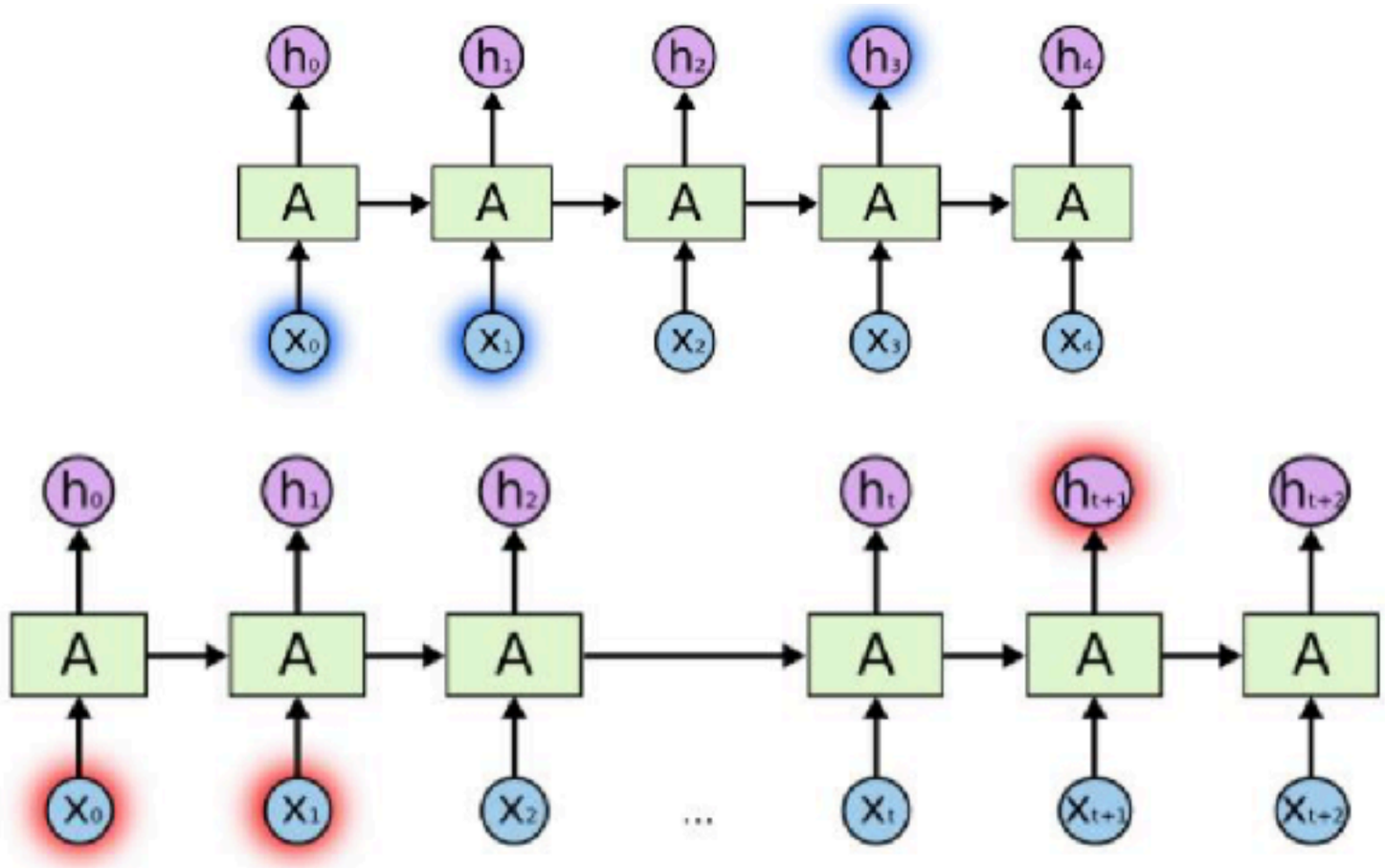
LSTM

01
02
03
04
05

❖ Long Short-Term Memory models)

■ Vanishing gradient problem을 극복하기 위한 RNN의 일종

■ Hidden state + cell state

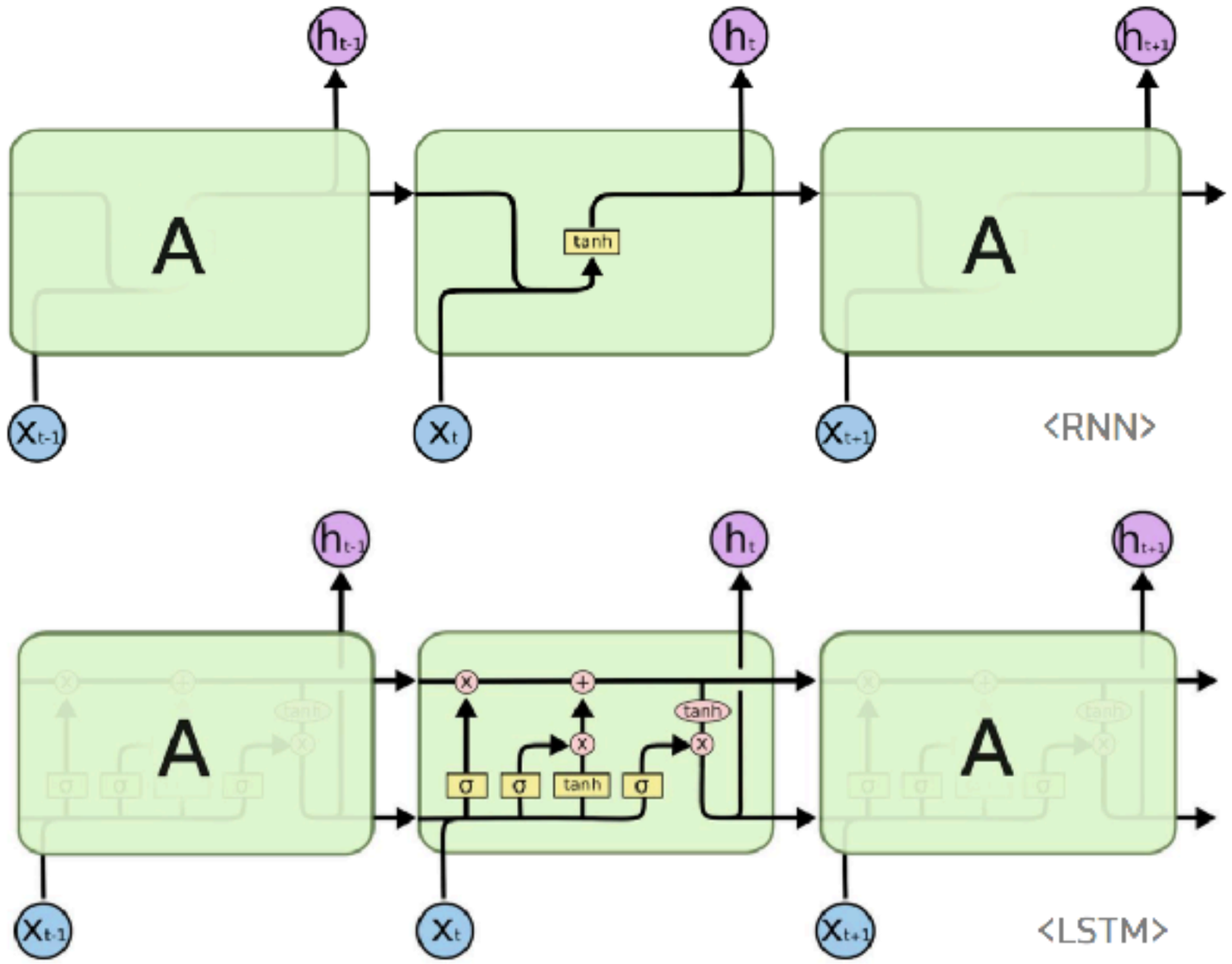


<관련 정보와 그 정보를 사용하는 지점 사이 거리가 멀 경우 RNN 학습능력 저하>

LSTM

01
02
03
04
05

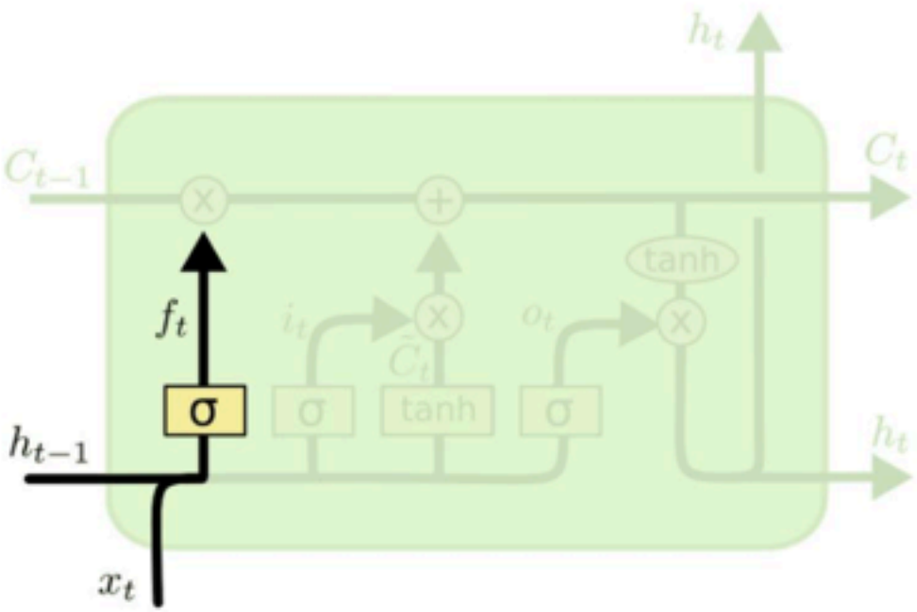
❖ LSTM 기본동작



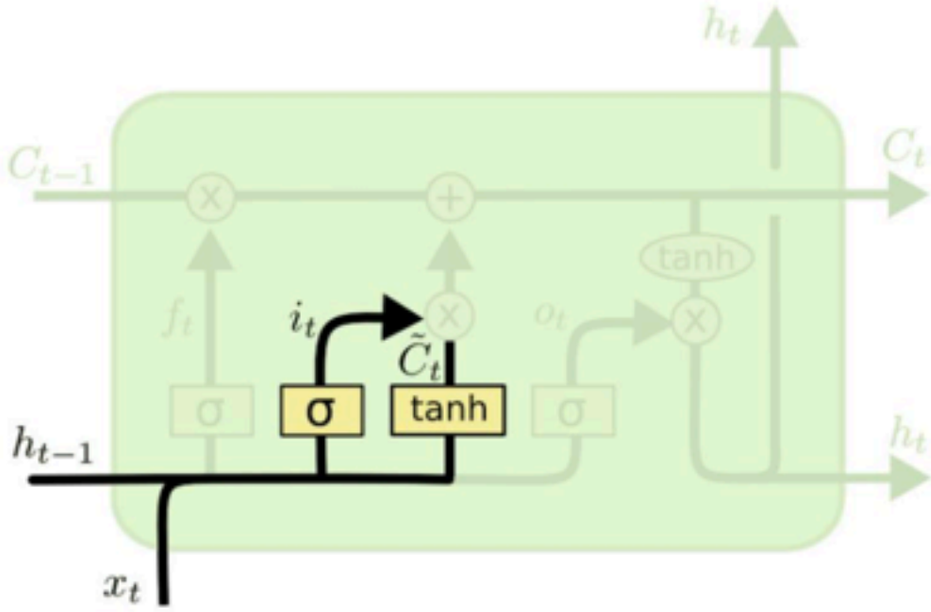
LSTM

01
02
03
04
05

❖ LSTM 기본동작



<forget gate>

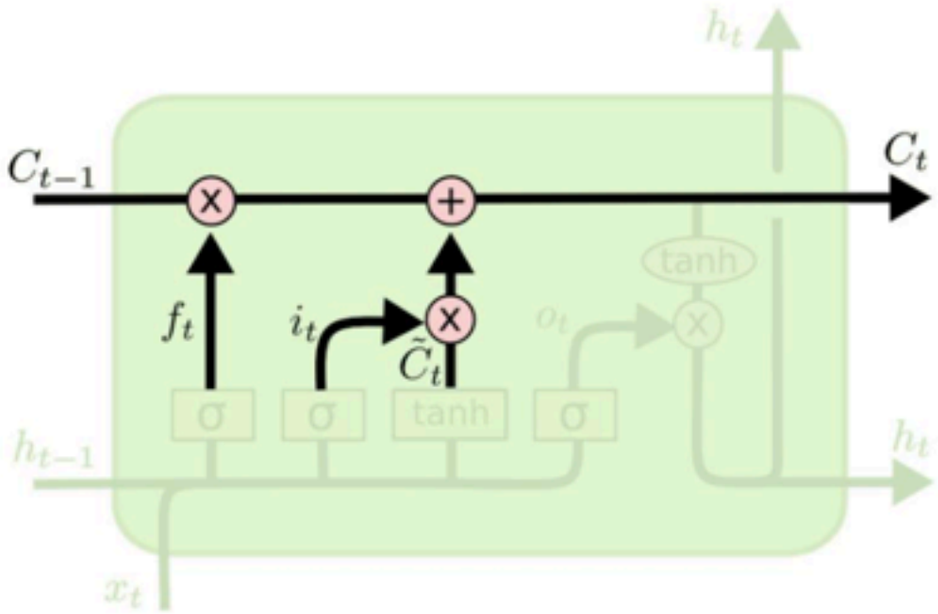


<input gate>

LSTM

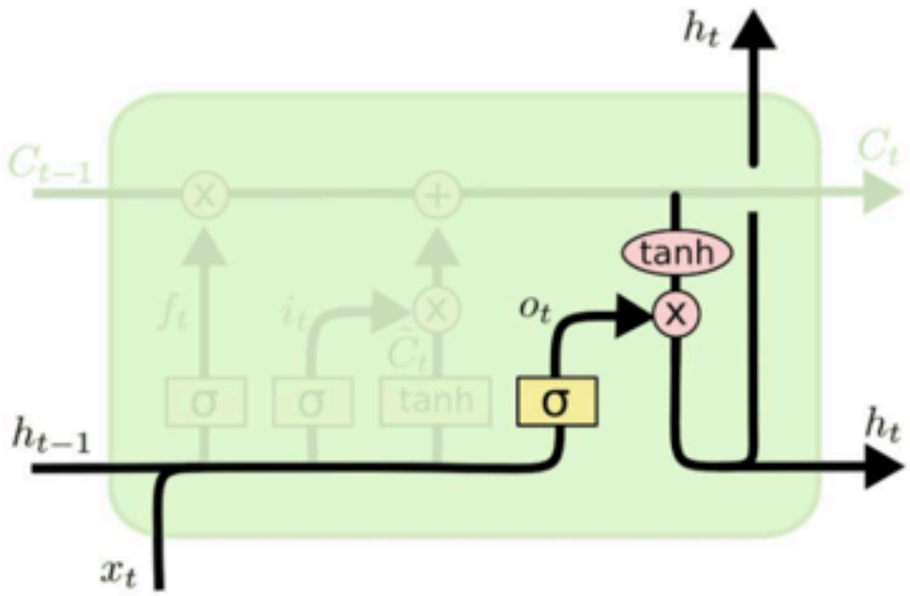
01
02
03
04
05

❖ LSTM 기본동작



<cell state update>

$$c_t = f_t \times c_{t-1} + i_t \times g_t$$



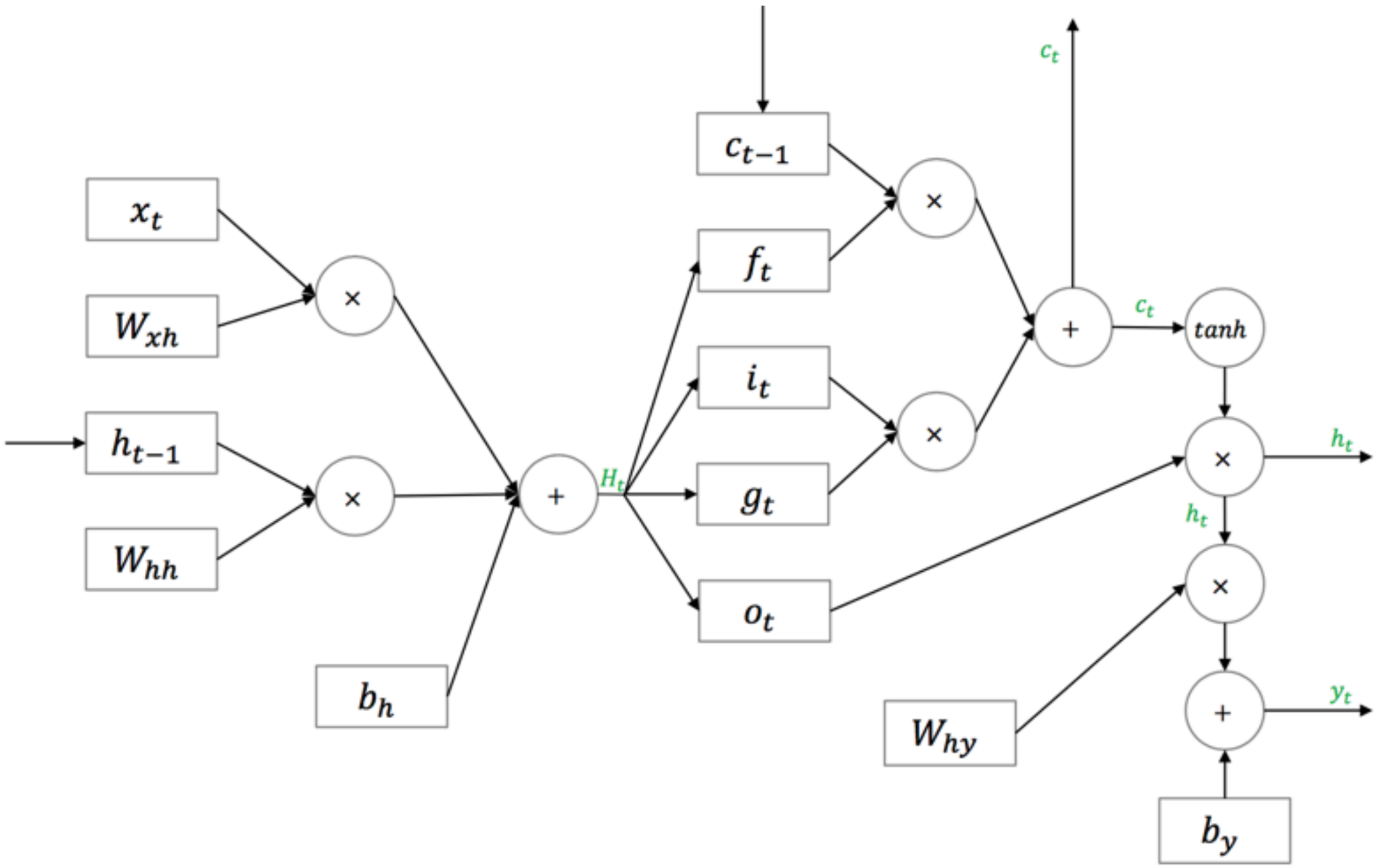
<hidden state update>

$$h_t = o_t \times \tanh(c_t)$$

LSTM

01
02
03
04
05

❖ LSTM forward pass



LSTM

01

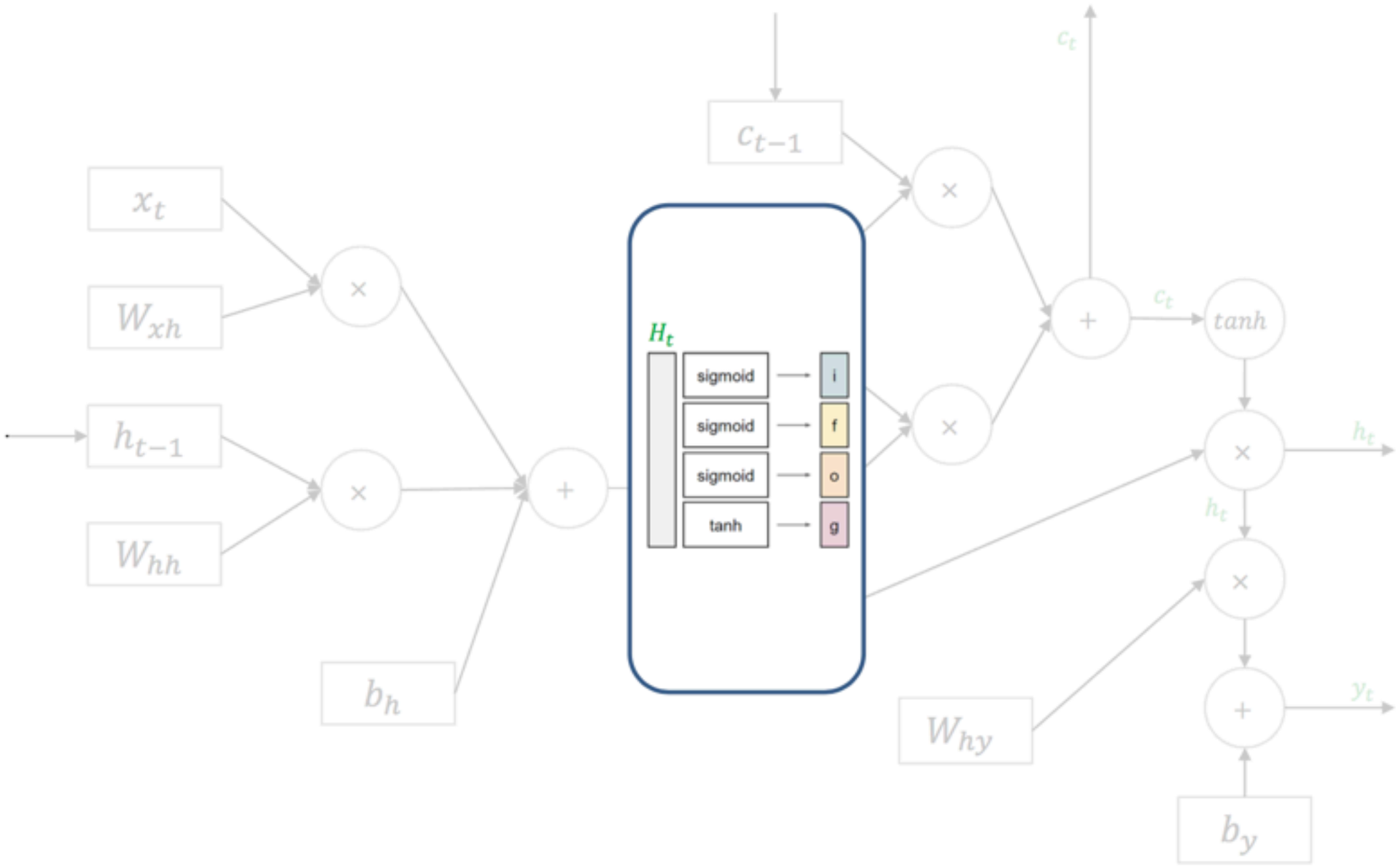
02

03

04

05

❖ LSTM forward pass



LSTM

01

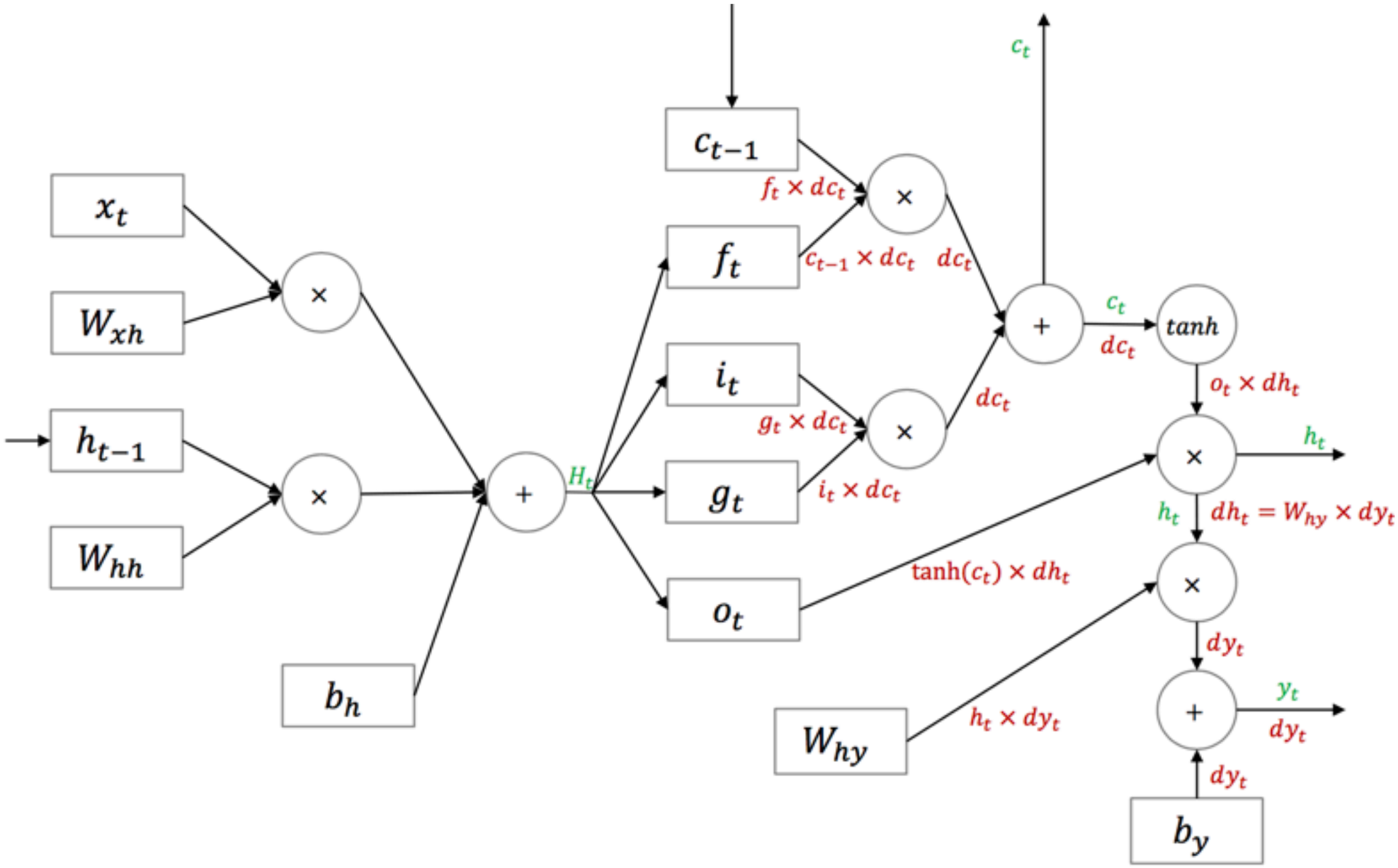
02

03

04

05

❖ LSTM backward pass



LSTM

01

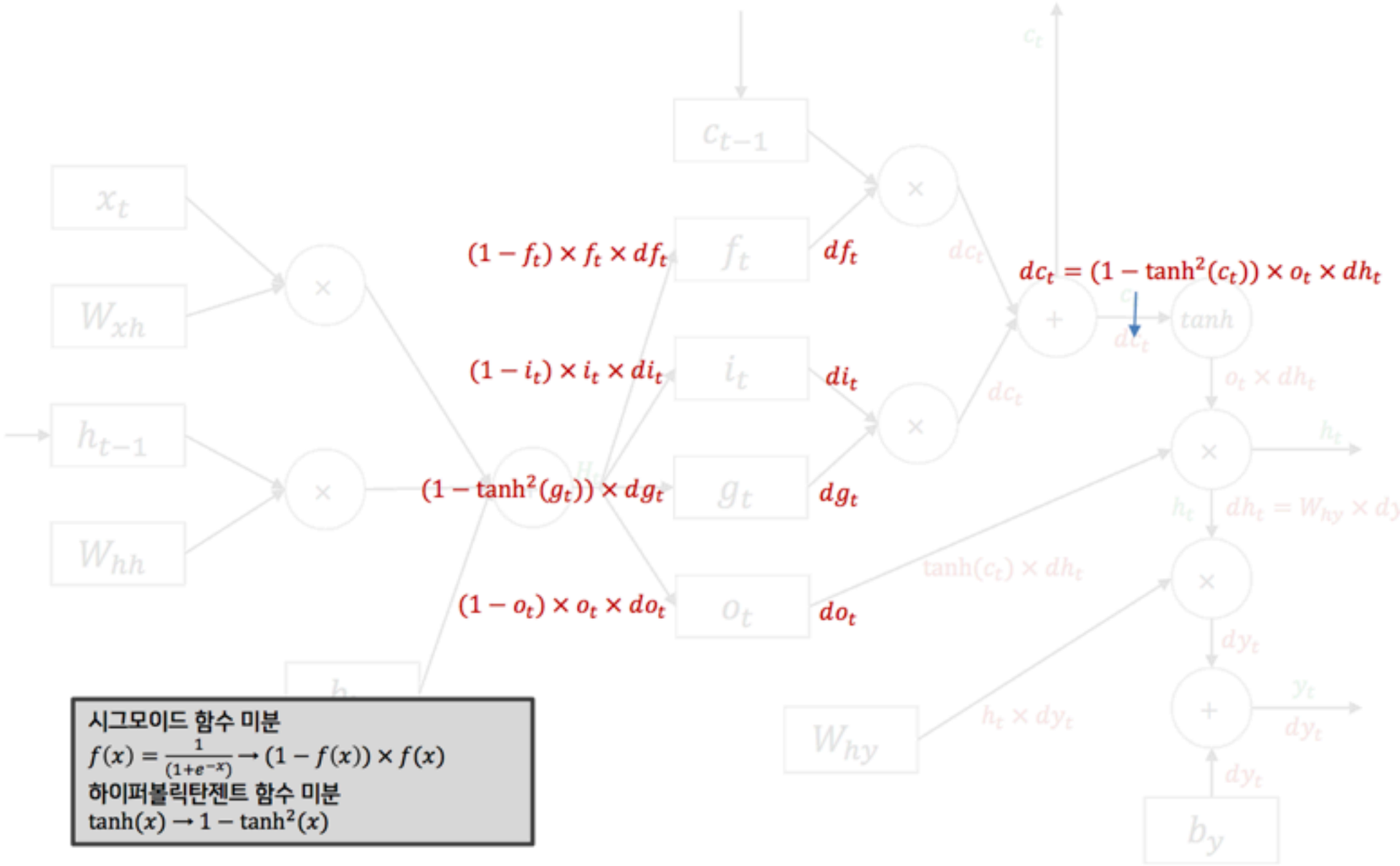
02

03

04

05

❖ LSTM backward pass



시그모이드 함수 미분
$f(x) = \frac{1}{(1+e^{-x})} \rightarrow (1-f(x)) \times f(x)$
하이퍼볼릭탄젠트 함수 미분
$\tanh(x) \rightarrow 1-\tanh^2(x)$

LSTM

01

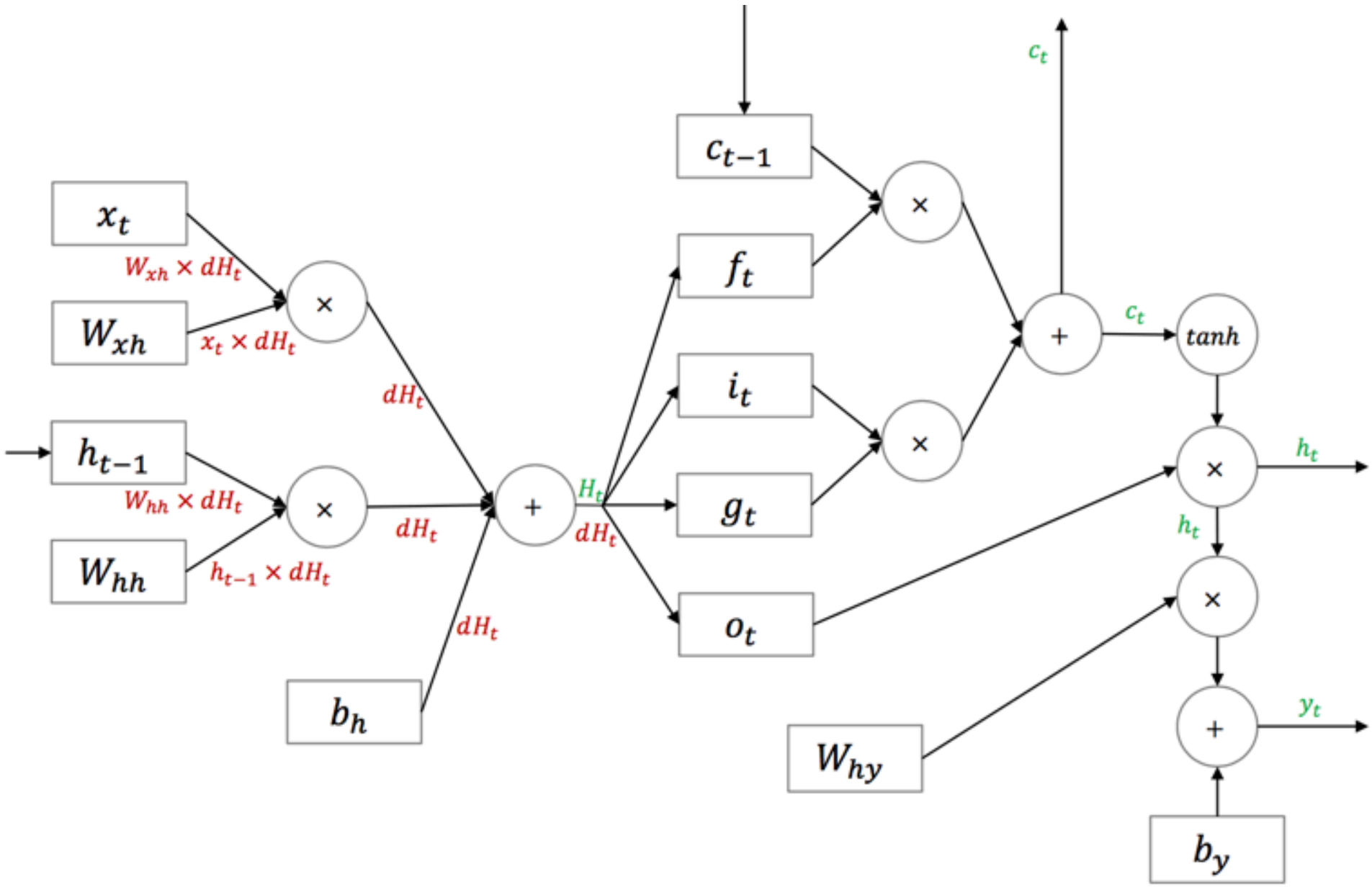
02

03

04

05

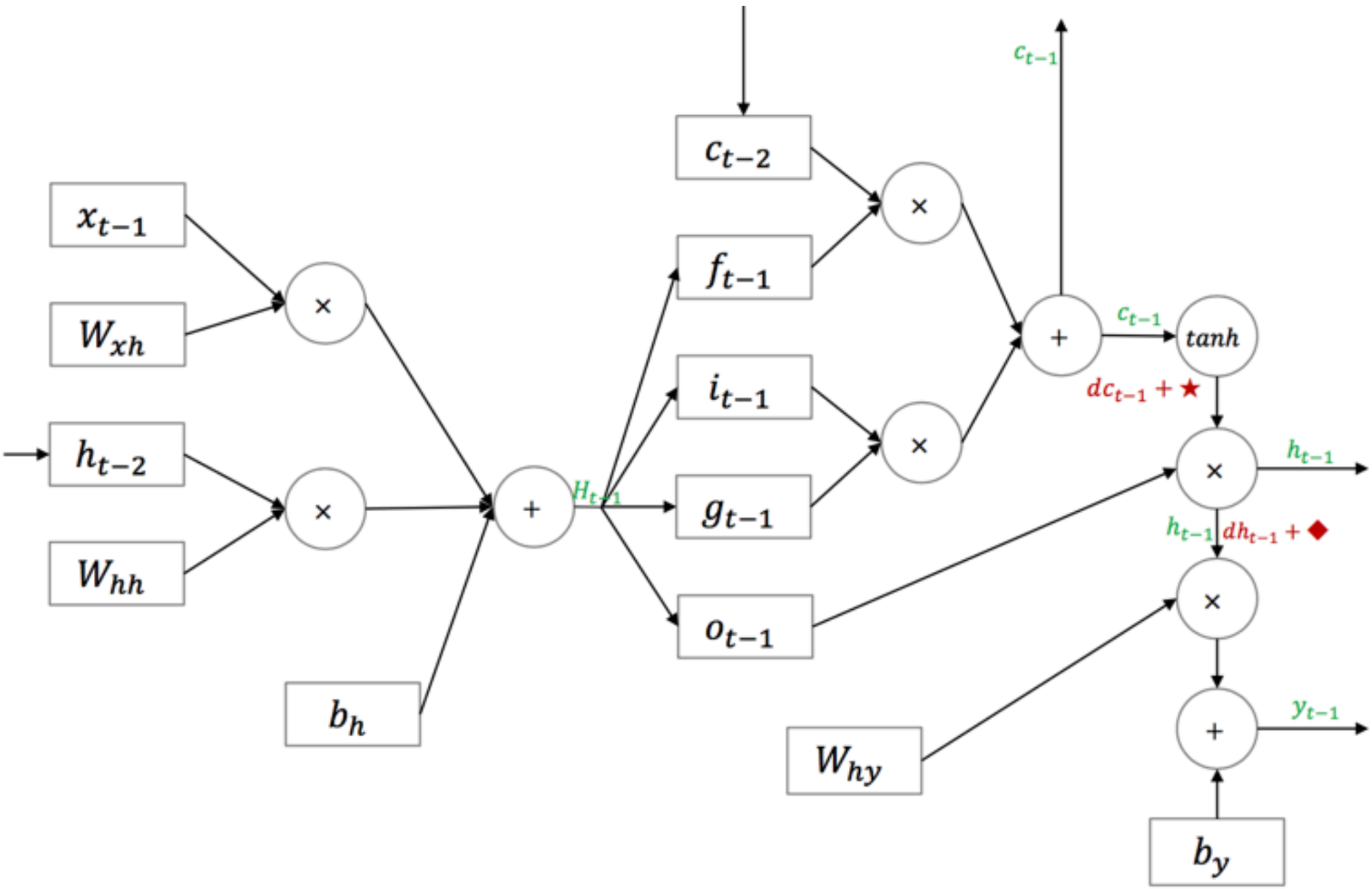
❖ LSTM backward pass



LSTM

01
02
03
04
05

❖ LSTM backward pass



RCNN

01

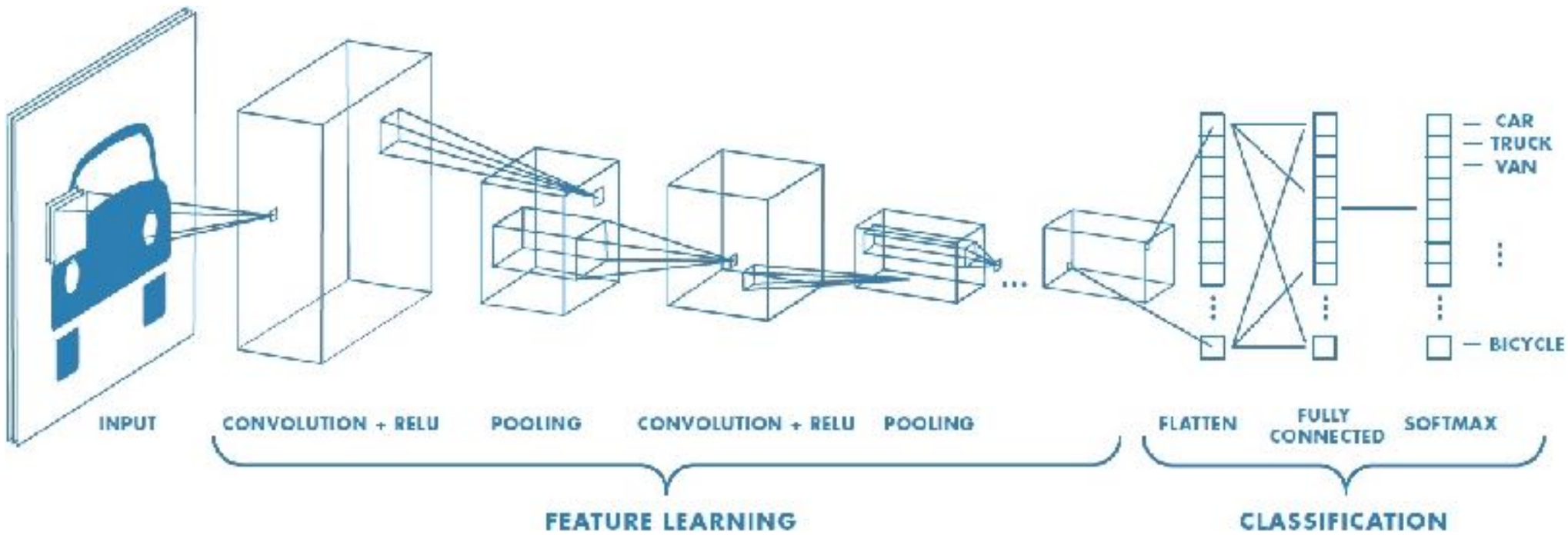
02

03

04

05

❖ RNN + CNN



조수필(2017). CNN.

CRNN

01

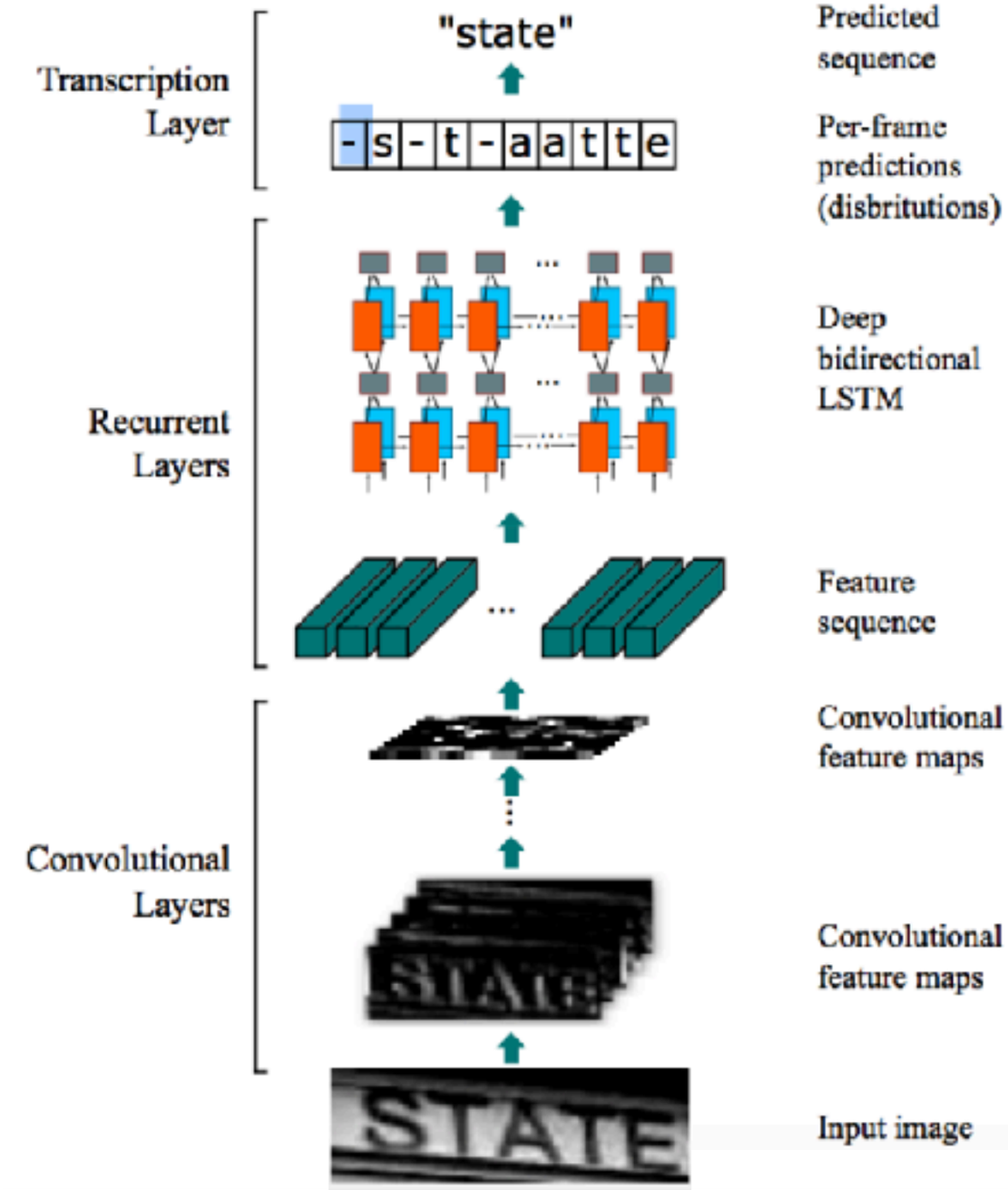
02

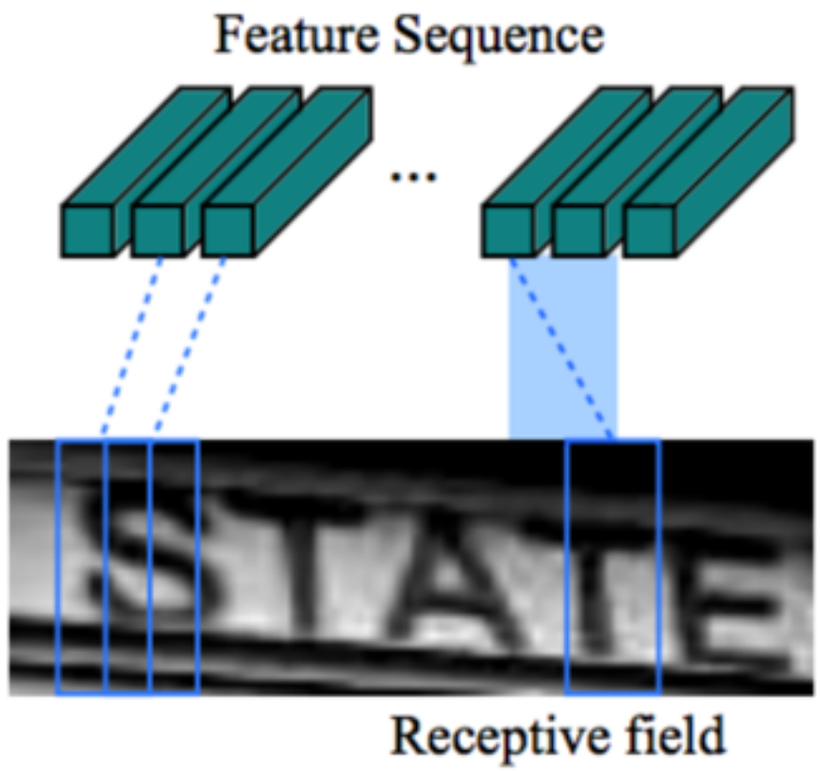
03

04

05

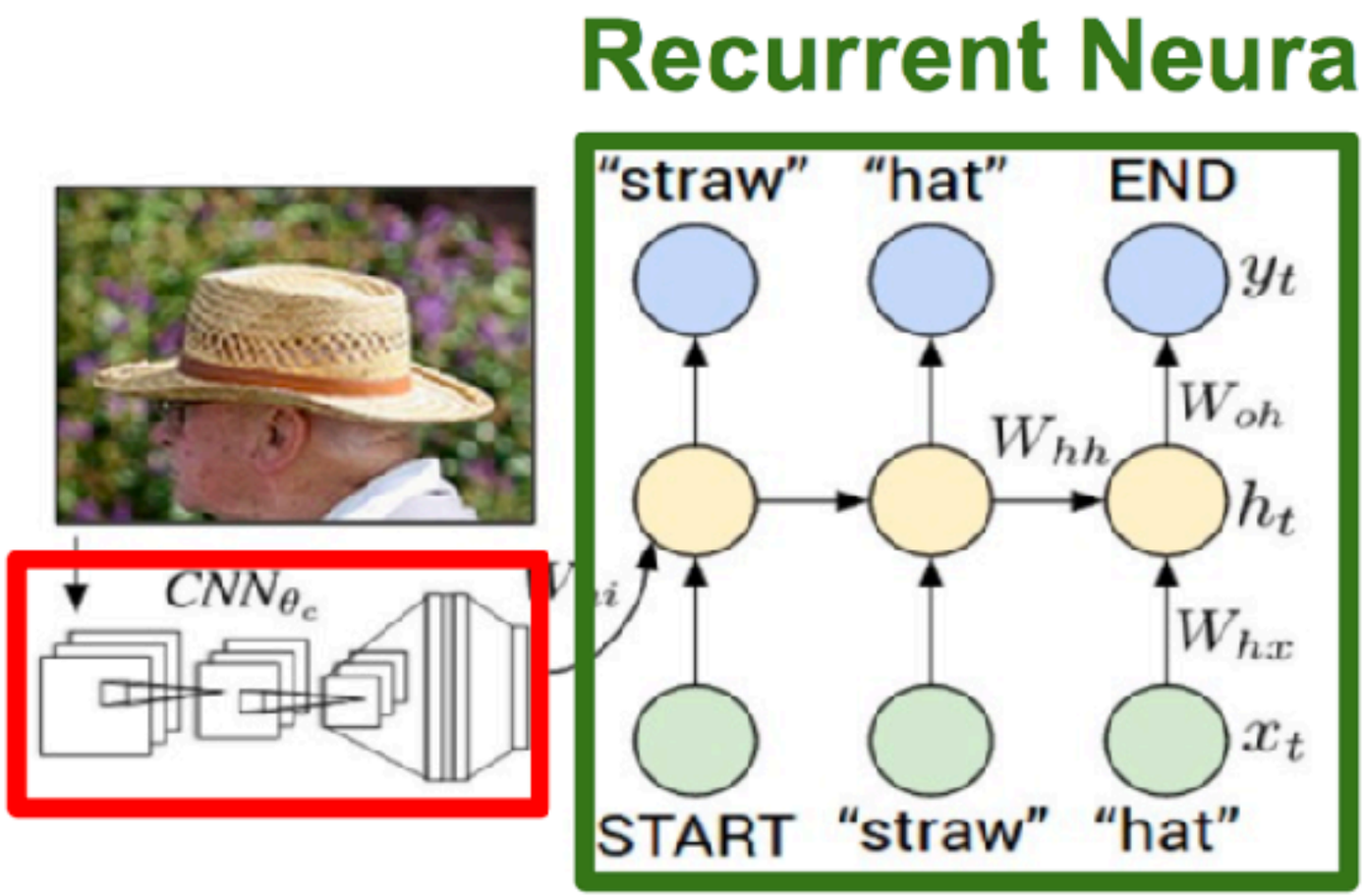
❖ CNN + RNN





01
02
03
04
05

❖ Image Captioning



01
02
03
04
05

❖ Image Captioning