RNN History with Python code

RNN, LSTM, GRU

고민수

RNN

RNN Model_Summary

Num of words : 10

Embedding vector length : 6

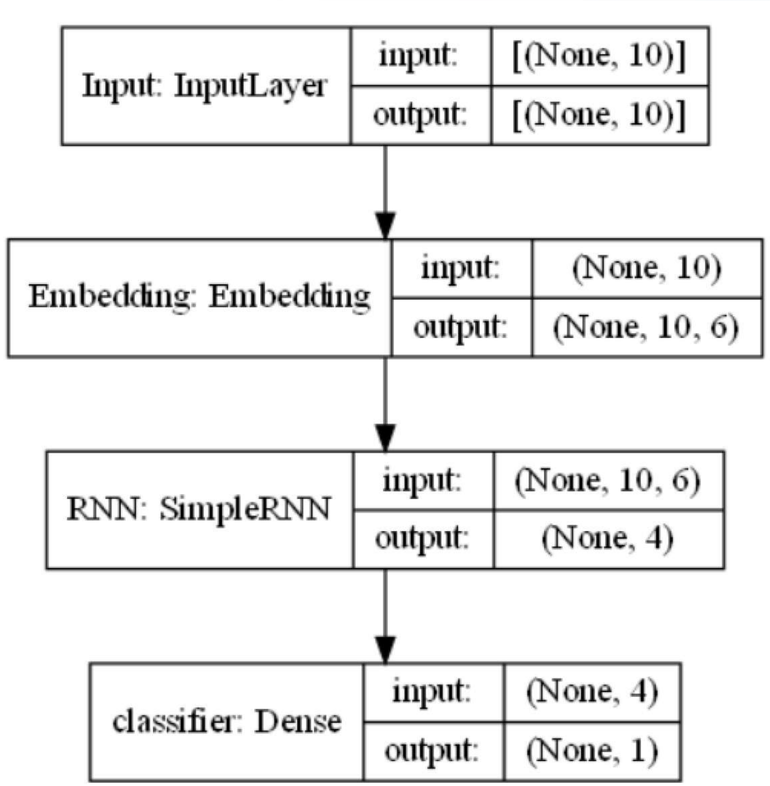
Num of RNN layers : 4

Classify : 0 / 1

Model: "RNN"

Layer (type)	Output Shape	Param #
Embedding (Embedding)	(None, 10, 6)	60000
RNN (SimpleRNN)	(None, 4)	44
classifier (Dense)	(None, 1)	5

Total params: 60,049
Trainable params: 60,049
Non-trainable params: 0



RNN

Input

```
1  timesteps = 10 # 단어의 개수 256 -> 10
2  input_dim = 6 # 임베딩 아웃풋 32 -> 6
3  hidden_units = 4 # RNN 노드 개수 32 -> 4
4  # 입력에 해당되는 2D 텐서
5  inputs = np.random.random((timesteps, input_dim))
6  inputs.shape
```

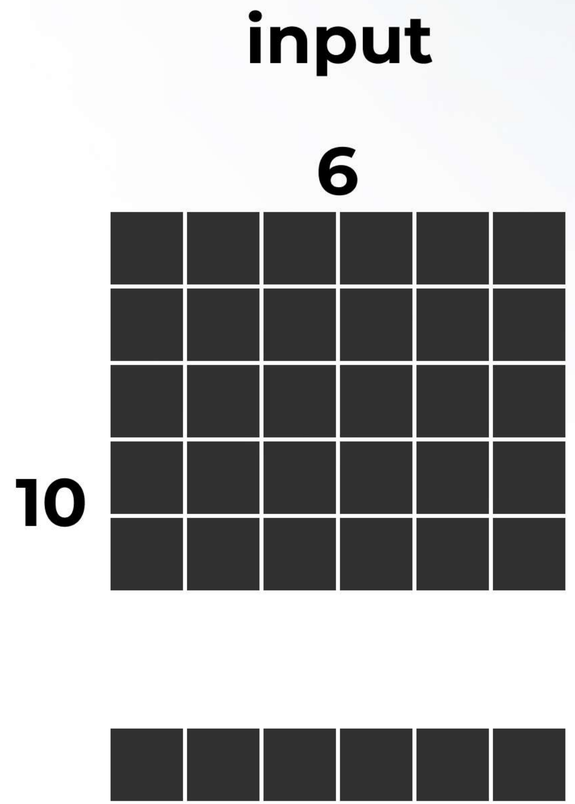
executed in 6ms, finished 13:33:25 2022-07-15

```
(10, 6)
```

```
1  # 10개의 (32,)
2  inputs[0], inputs[9]
```

executed in 14ms, finished 13:36:24 2022-07-15

```
(array([0.75989621, 0.68042876, 0.3065464 , 0.56762763, 0.61315184,
        0.46367387]),
 array([0.04417823, 0.7084128 , 0.51543125, 0.77244701, 0.60342653,
        0.89025227]))
```



RNN

RNN layers initalize

```
1  # 초기 은닉 상태는 0(벡터)로 초기화
2  hidden_state_t = np.zeros((hidden_units,))
3  hidden_state_t.shape
```

executed in 7ms, finished 13:37:25 2022-07-15

(4,)

```
1  print('초기 은닉 상태 :',hidden_state_t)
```

executed in 8ms, finished 13:37:25 2022-07-15

초기 은닉 상태 : [0. 0. 0. 0.]

RNN

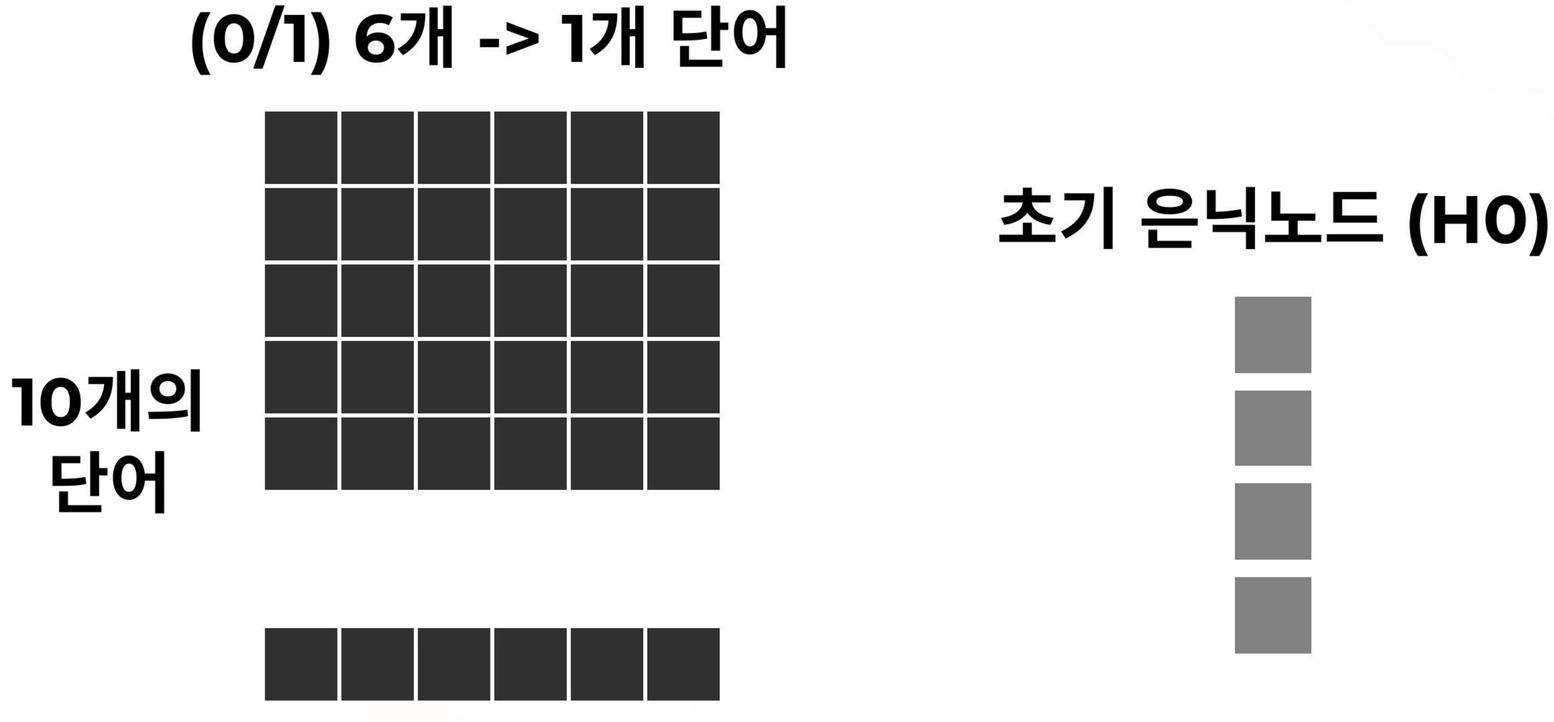
```
1 Wx = np.random.random((hidden_units, input_dim))  # (4, 6)크기의 2D 텐서 생성. 입력에 대한 가중치.
2 Wh = np.random.random((hidden_units, hidden_units)) # (4, 4)크기의 2D 텐서 생성. 은닉 상태에 대한 가중치.
3 b = np.random.random((hidden_units,)) # (4,)크기의 1D 텐서 생성. 이 값은 편향(bias).
```

executed in 6ms, finished 13:33:38 2022-07-15

```
1 print('가중치 Wx의 크기(shape) :',np.shape(Wx))
2 print('가중치 Wh의 크기(shape) :',np.shape(Wh))
3 print('편향의 크기(shape) :',np.shape(b))
```

executed in 15ms, finished 13:33:39 2022-07-15

```
가중치 Wx의 크기(shape) : (4, 6)
가중치 Wh의 크기(shape) : (4, 4)
편향의 크기(shape) : (4,)
```

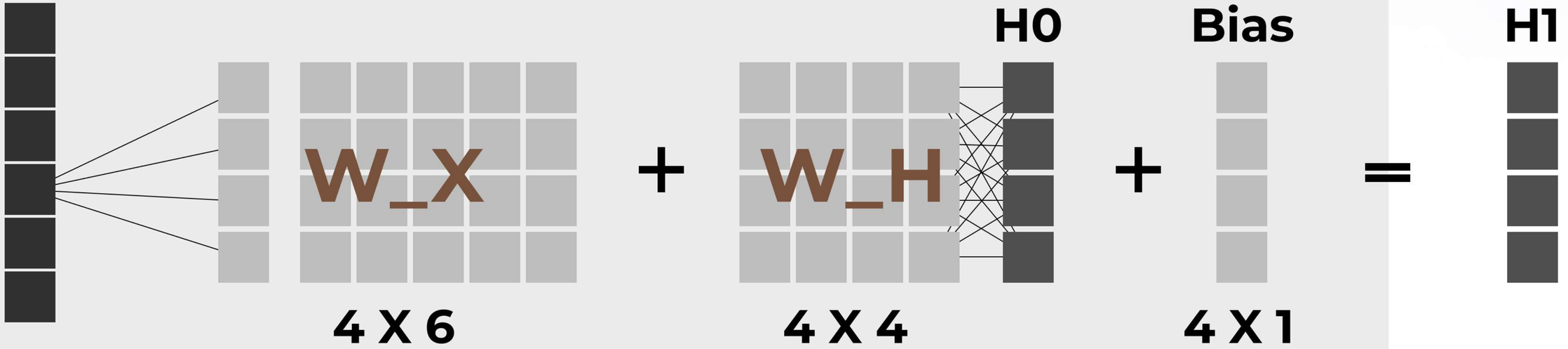


RNN

```
# Wx * Xt + Wh * Ht-1 + b(bias)
output_t = np.tanh(np.dot(Wx,input_t) + np.dot(Wh,hidden_state_t) + b)
```

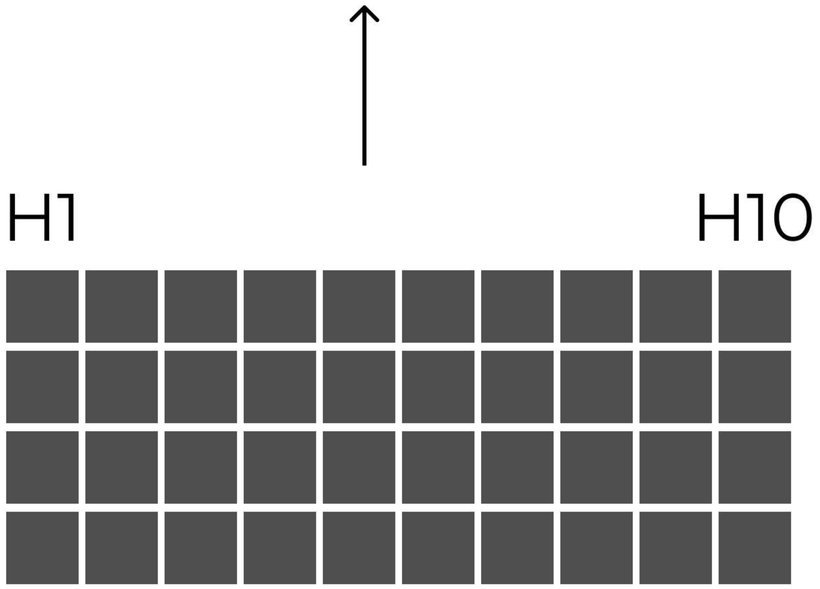
6개로 이루어진 텐서

Tanh



RNN

Many To Many

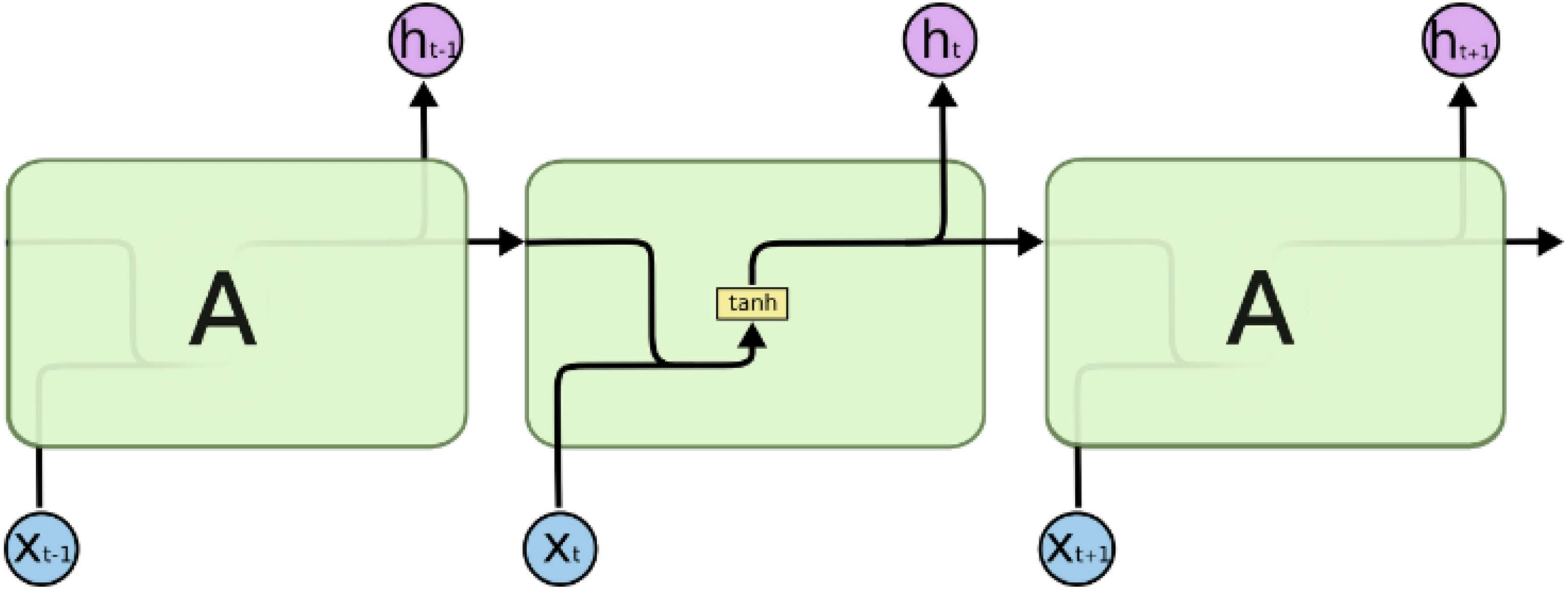


4 X 10

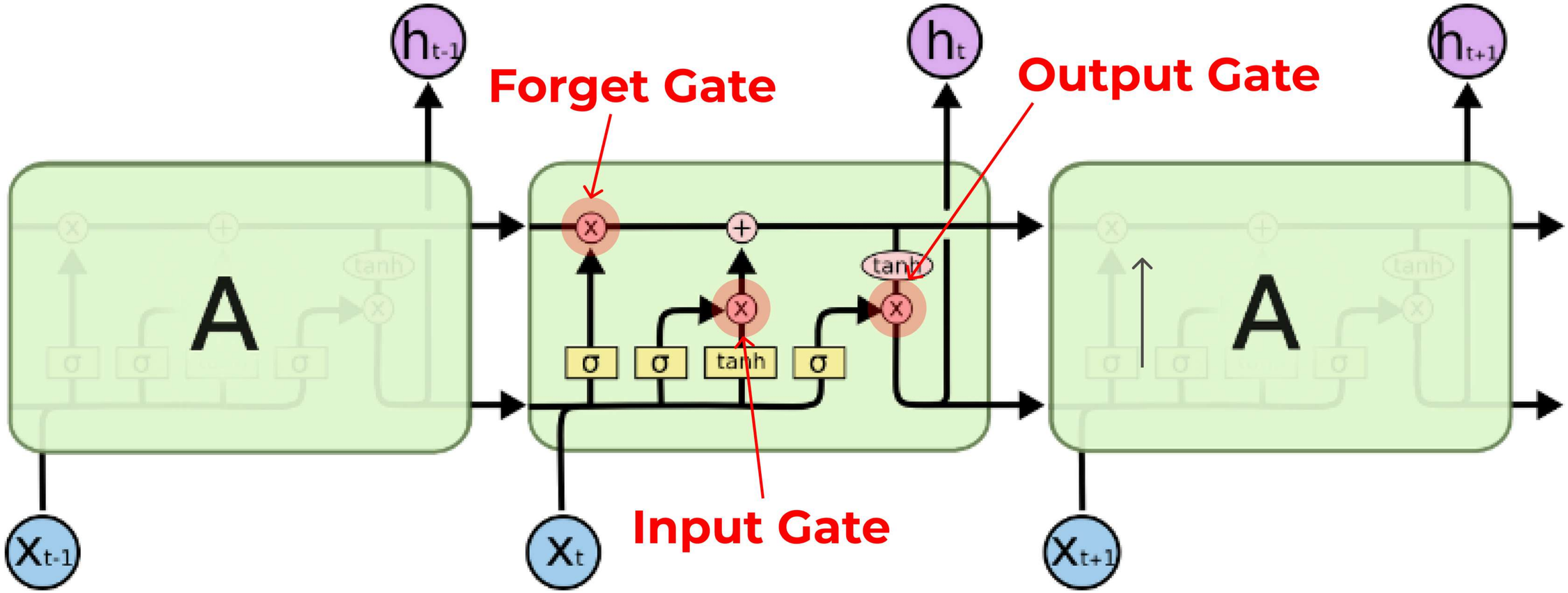
Many To One

4 X 1

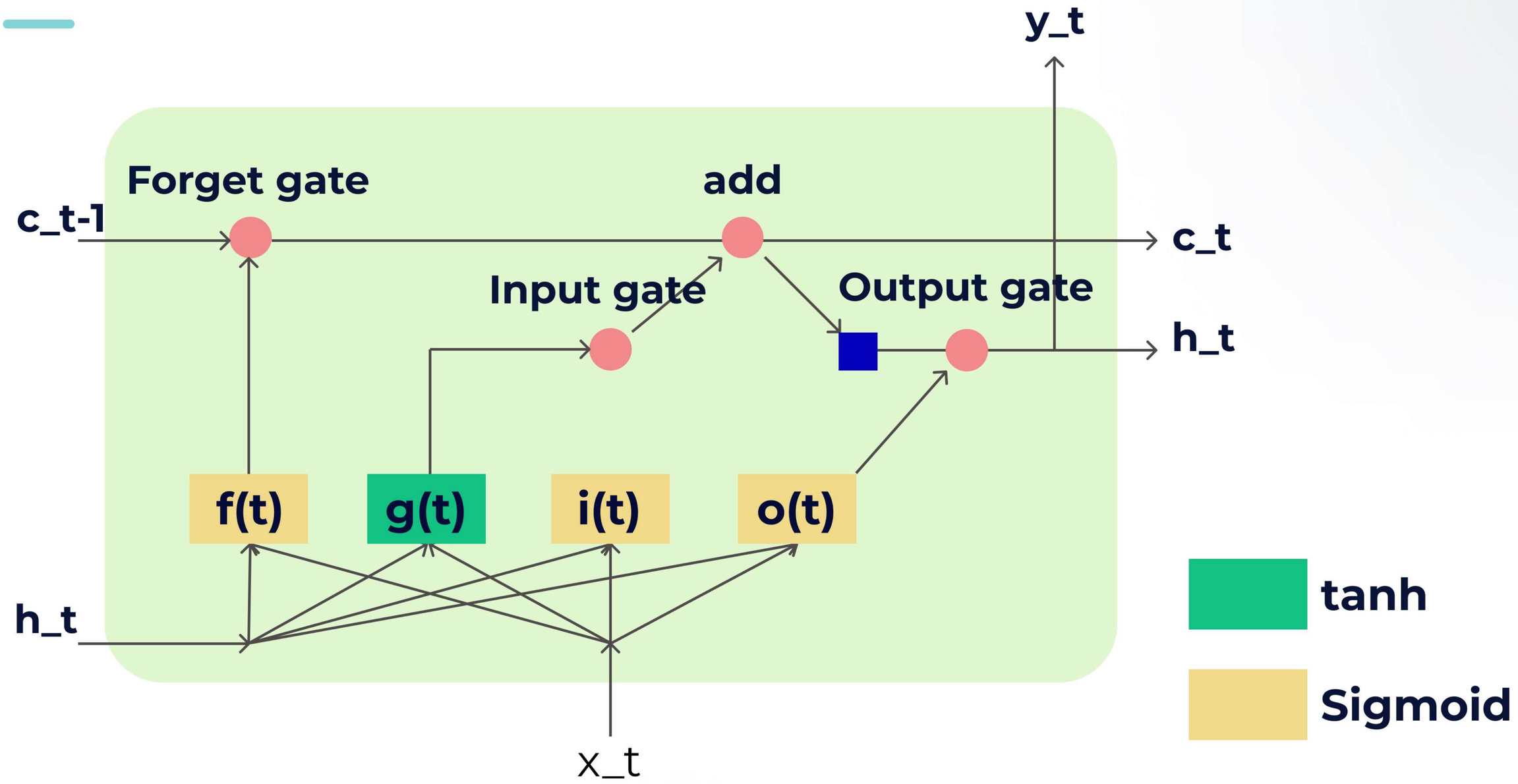
RNN



LSTM

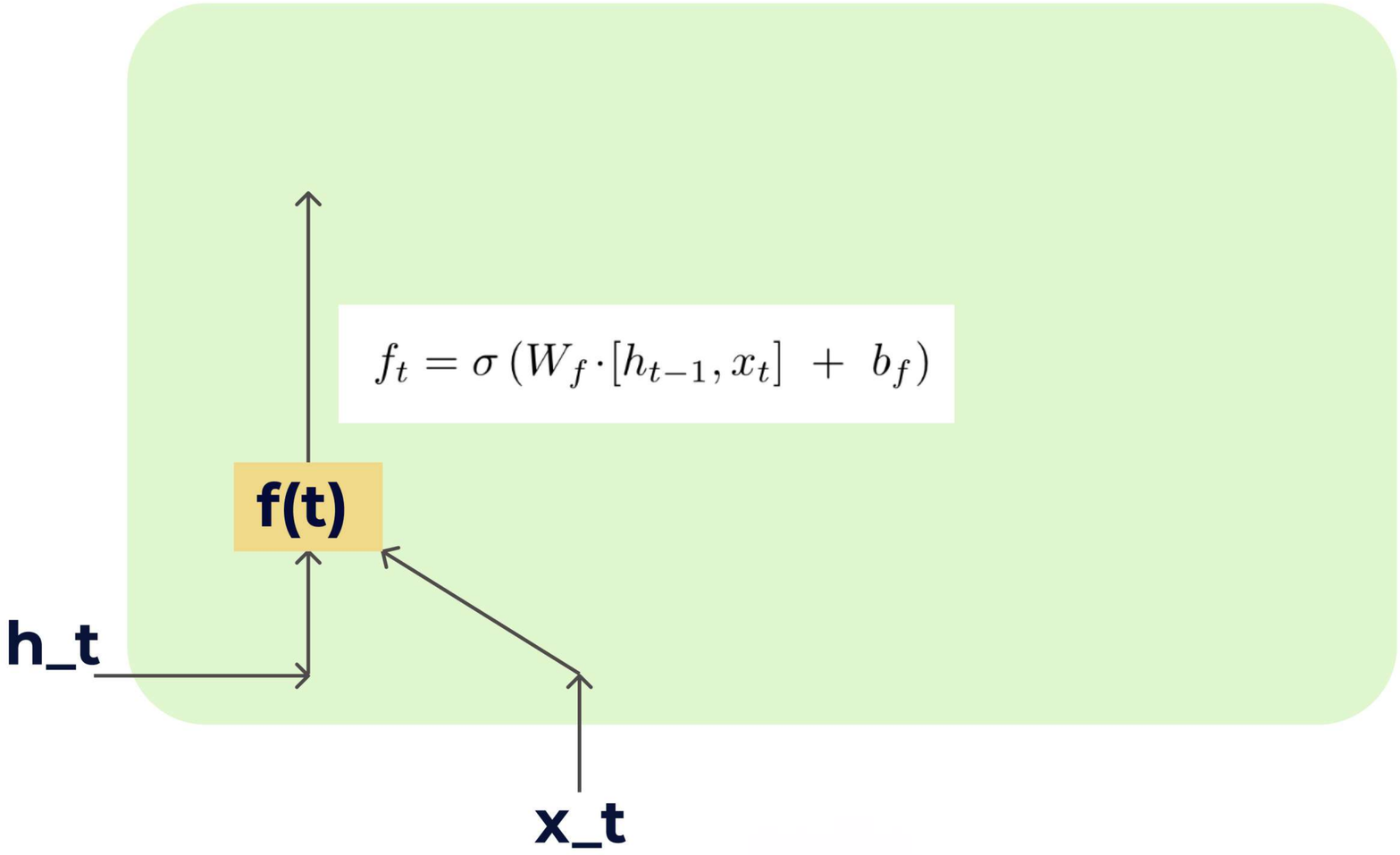

Forget Gate
Output Gate
Input Gate
A
A
tanh
σ
σ
tanh
σ
tanh
h_{t-1}
h_t
h_{t+1}
X_{t-1}
X_t
X_{t+1}

LSTM


y_t
Forget gate
add
c_t-1
c_t
Input gate
Output gate
h_t
f(t)
g(t)
i(t)
o(t)
h_t
x_t
tanh
Sigmoid

LSTM

Forget gate



tanh

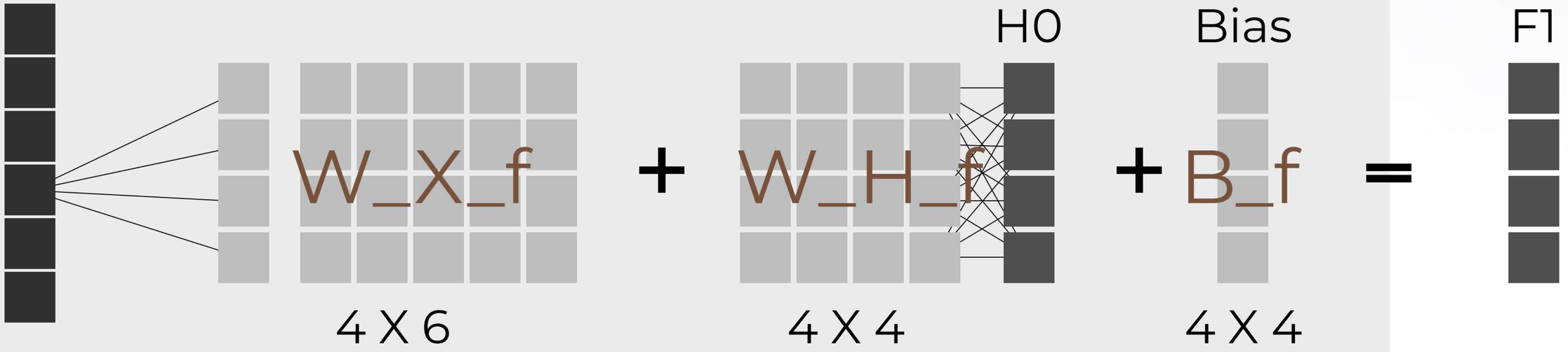
Sigmoid

LSTM

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

6개로 이루어진 텐서

Sigmoid



LSTM

Forget gate

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

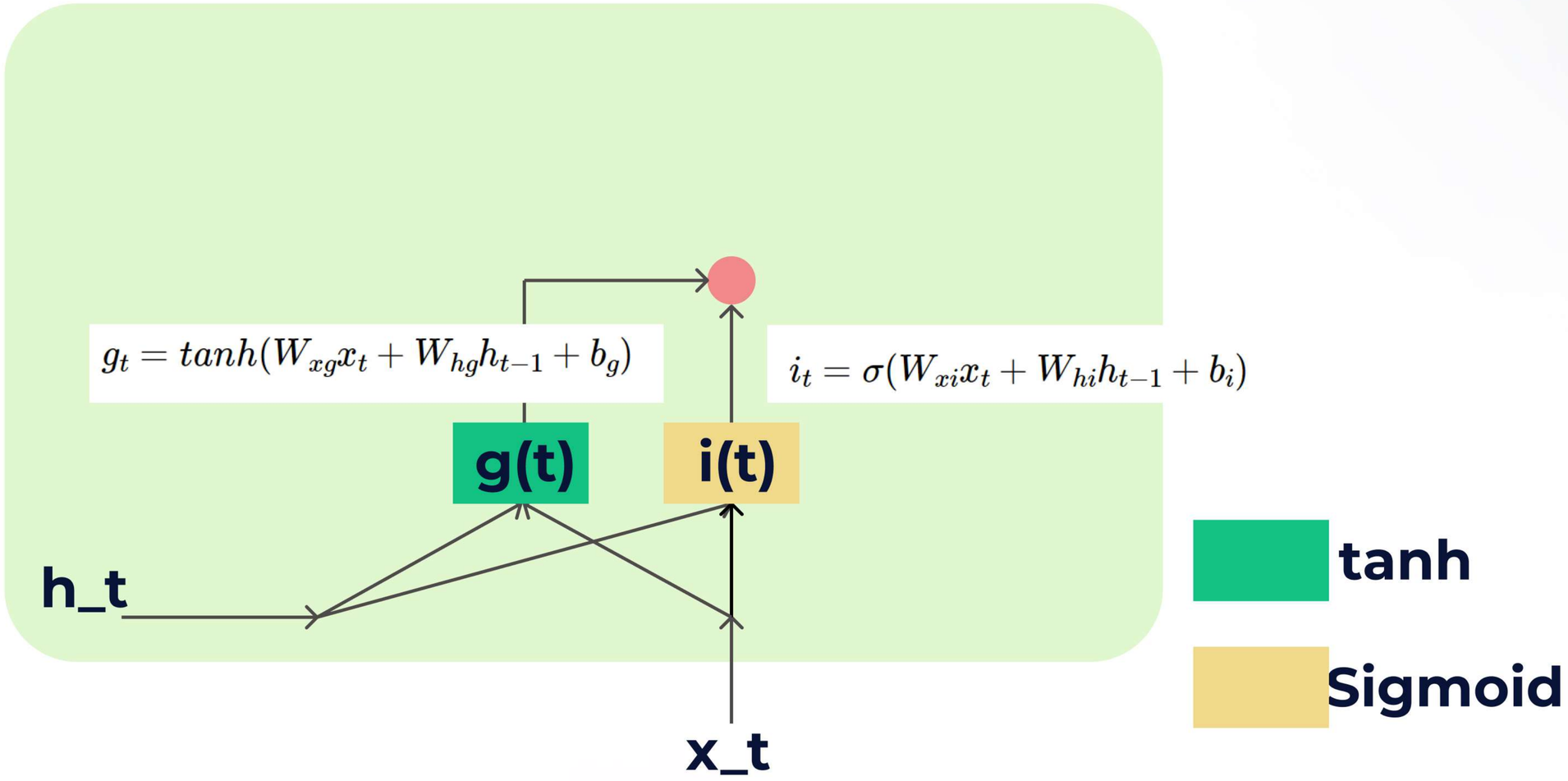
0~1 사이의 값 ⟶ 삭제된 정보의 양

F1

0에 가까울수록 많은 정보가 삭제

LSTM

Input gate



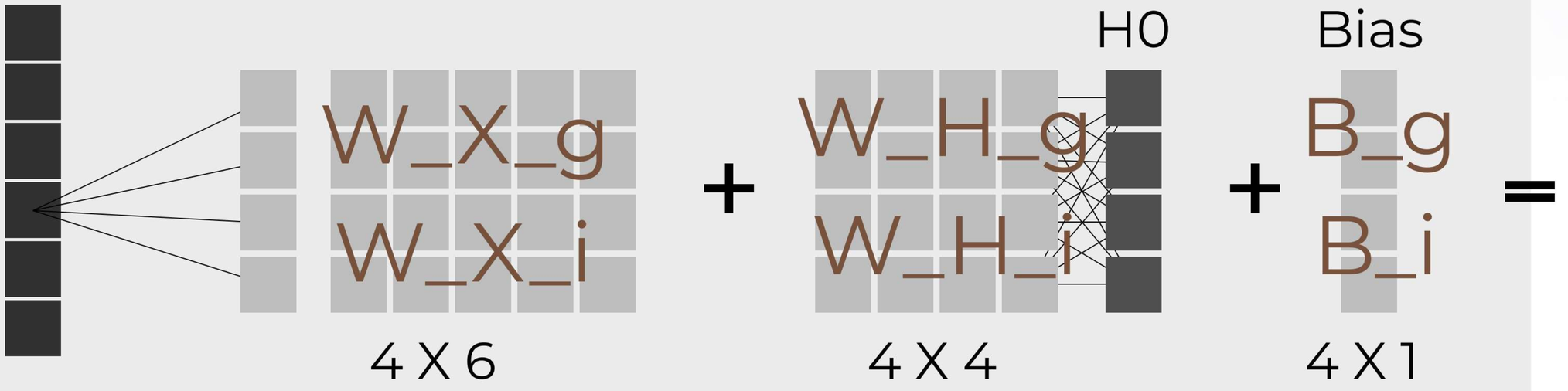
LSTM

Input gate

$$g_t = tanh(W_{xg}x_t + W_{hg}h_{t-1} + b_g)$$

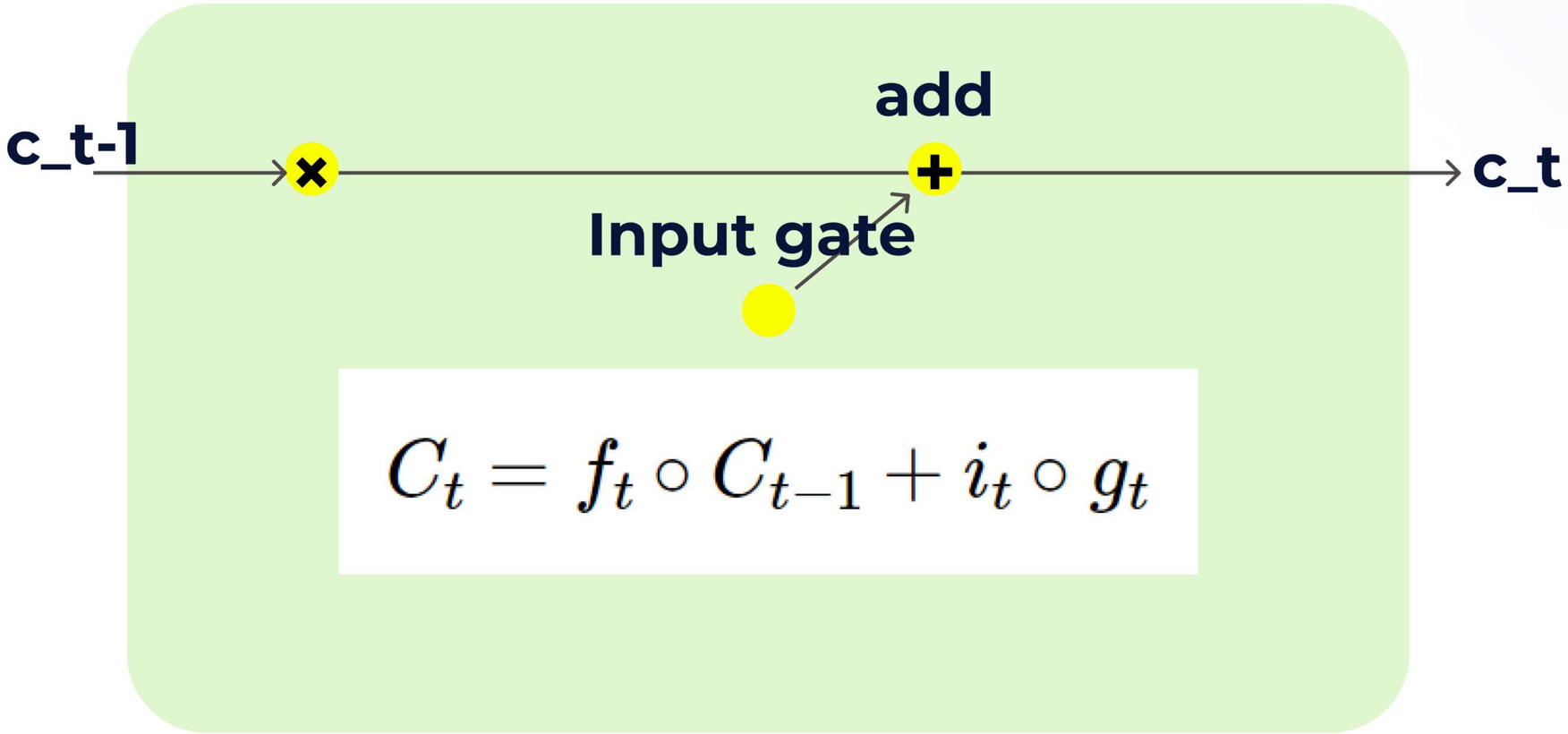
$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i)$$

6개로 이루어진 텐서 tanh Sigmoid



LSTM

Cell state

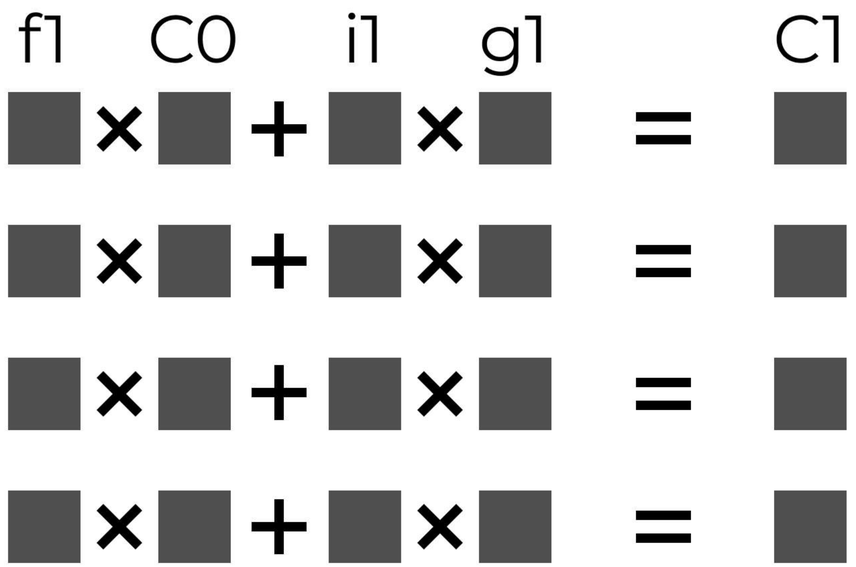


$$C_t = f_t \circ C_{t-1} + i_t \circ g_t$$

LSTM

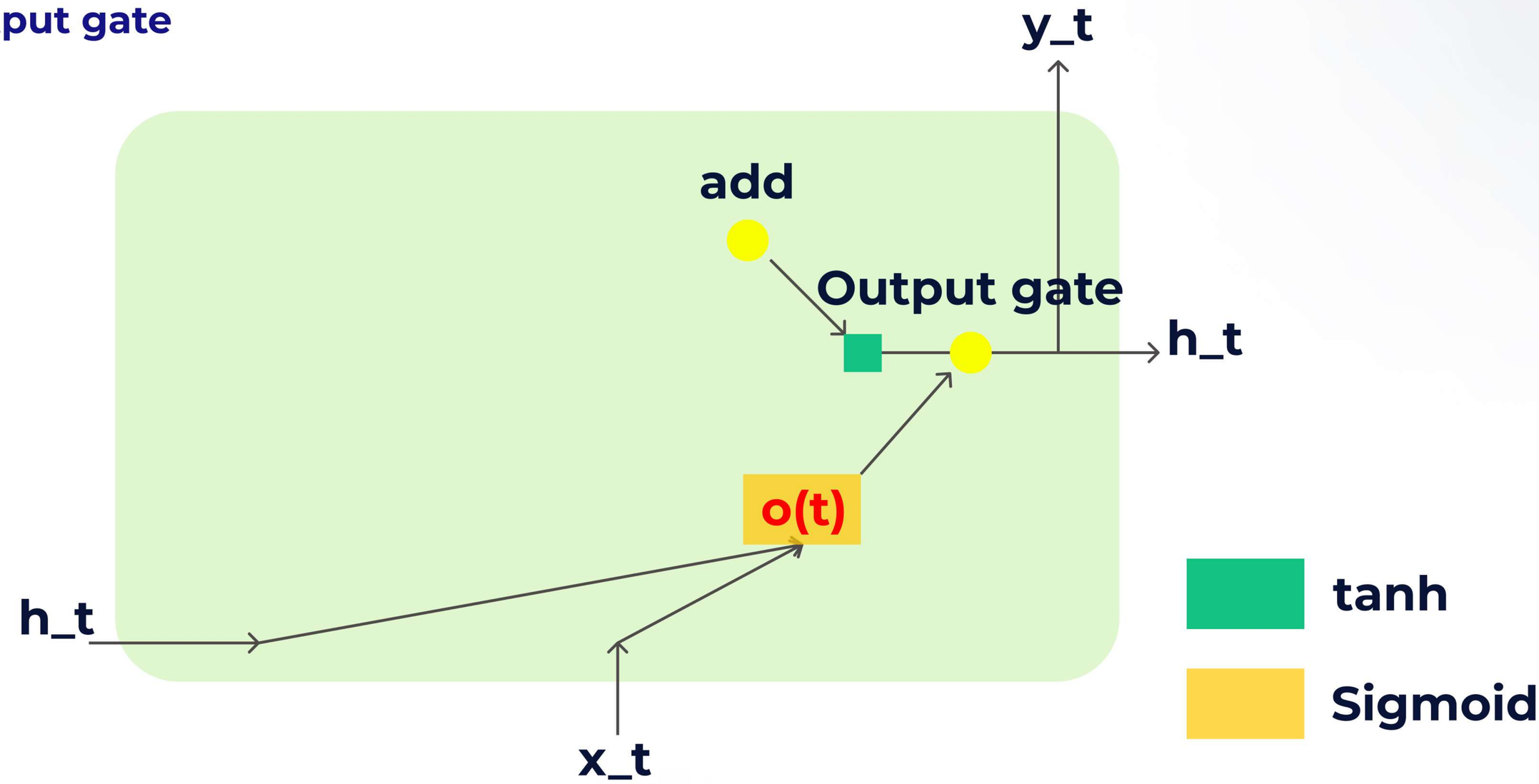
Cell state

$$C_t = f_t \circ C_{t-1} + i_t \circ g_t$$



LSTM

Output gate

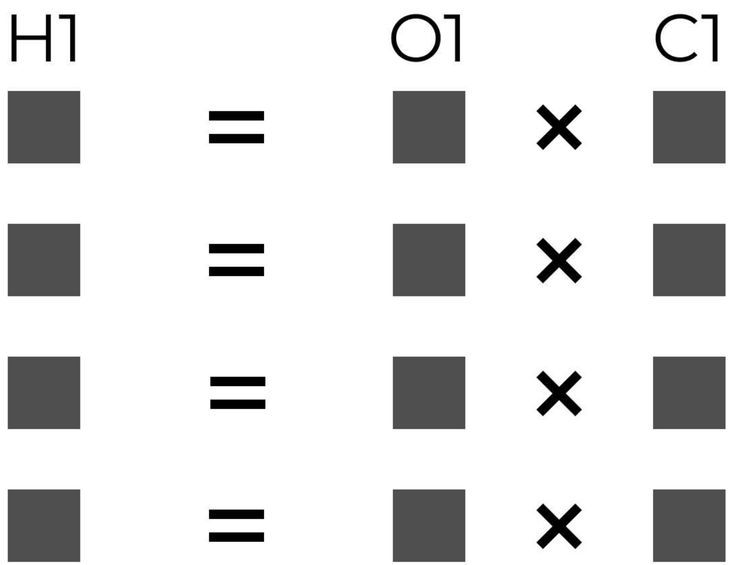


LSTM

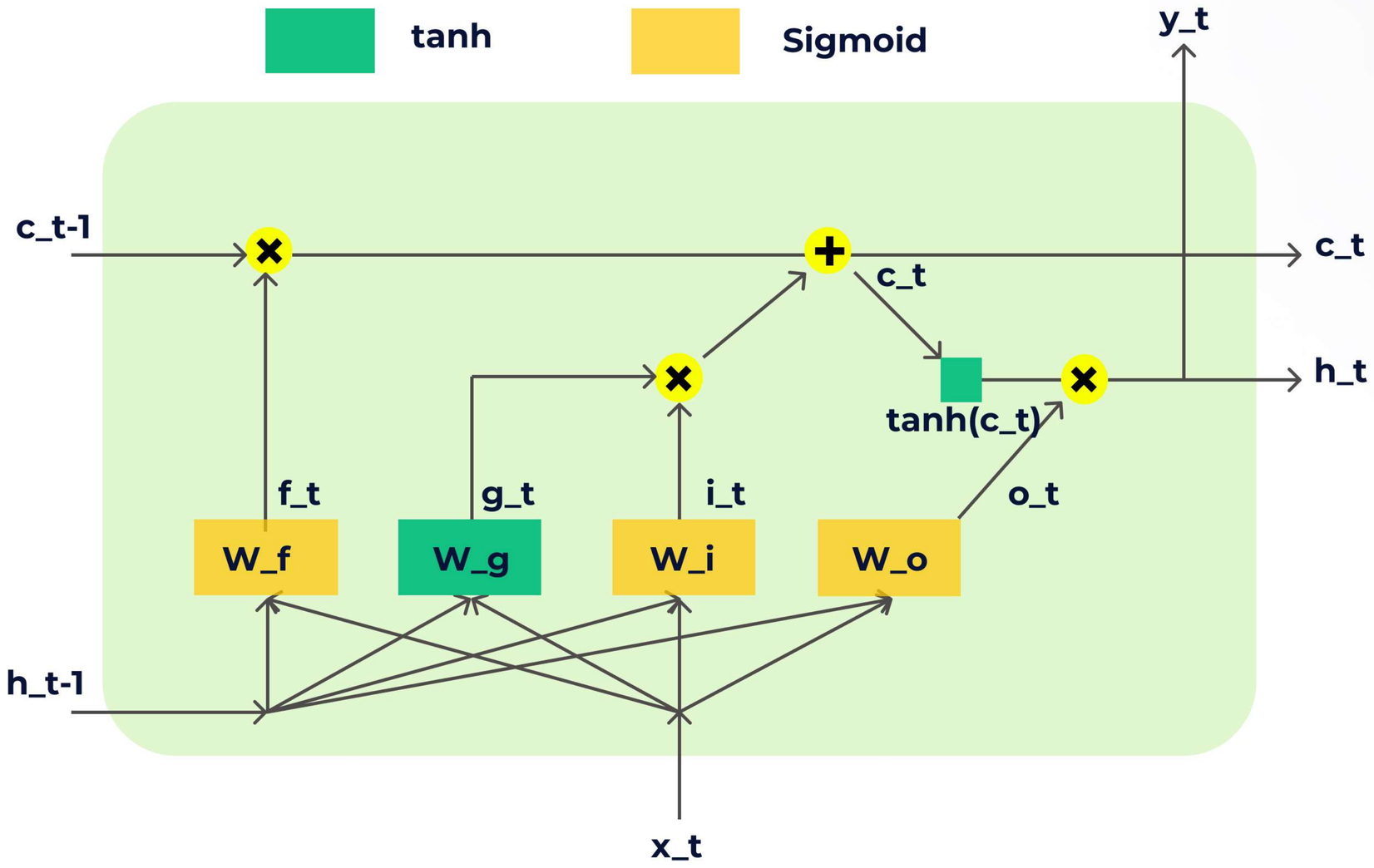
Output gate

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o)$$

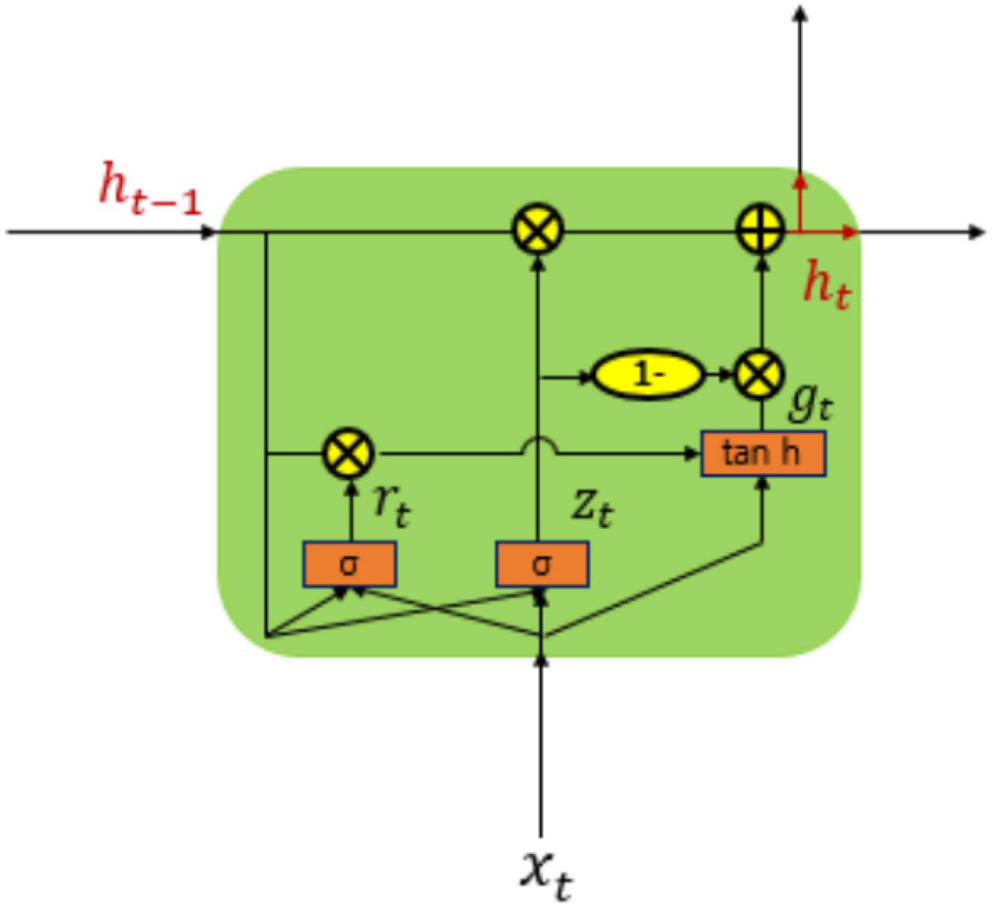
$$h_t = o_t \circ tanh(c_t)$$



LSTM

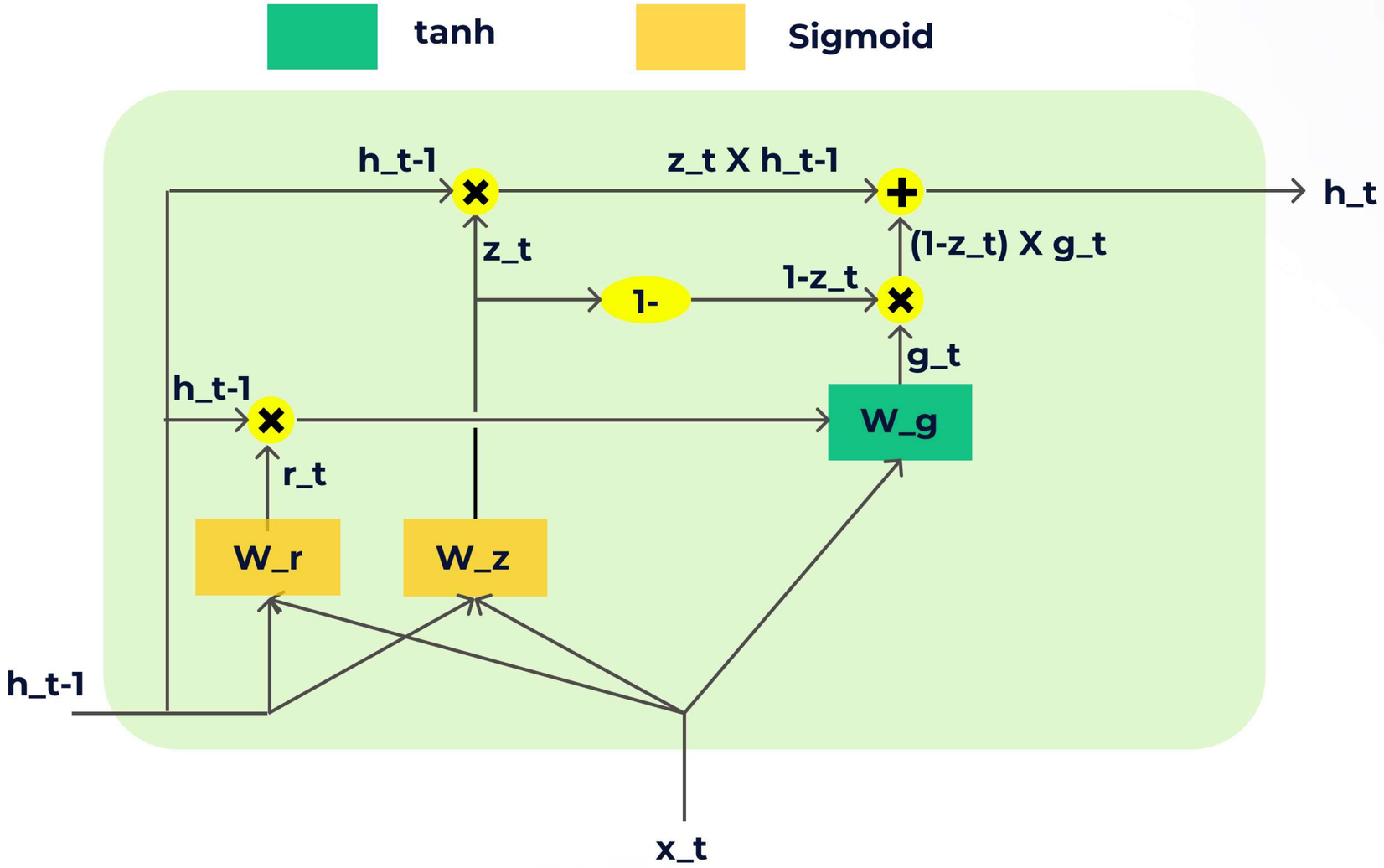


GRU



$$r_t = \sigma(W_{xr}x_t + W_{hr}h_{t-1} + b_r)$$
$$z_t = \sigma(W_{xz}x_t + W_{hz}h_{t-1} + b_z)$$
$$g_t = tanh(W_{hg}(r_t \circ h_{t-1}) + W_{xg}x_t + b_g)$$
$$h_t = (1 - z_t) \circ g_t + z_t \circ h_{t-1}$$

GRU


tanh
Sigmoid
h_t-1
z_t X h_t-1
h_t
z_t
(1-z_t) X g_t
1-
1-z_t
g_t
h_t-1
W_g
r_t
W_r
W_z
h_t-1
x_t

Thank you

RNN History with Python code

RNN, LSTM, GRU

고민수