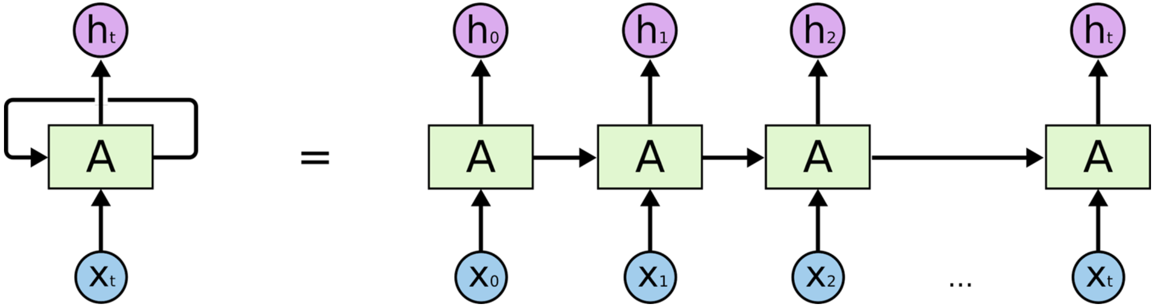
RNN
(Recurrent Neural Network)

RNN (Recurrent Neural Network)

- 순환 인공 신경망 (RNN)은 인공 신경망의 한 종류
- 유닛간의 연결이 순환적 구조
- 이러한 구조는 시계열 특징을 모델링 할 수 있음



https://en.wikipedia.org/wiki/Recurrent_neural_network

RNN 활용 (1)

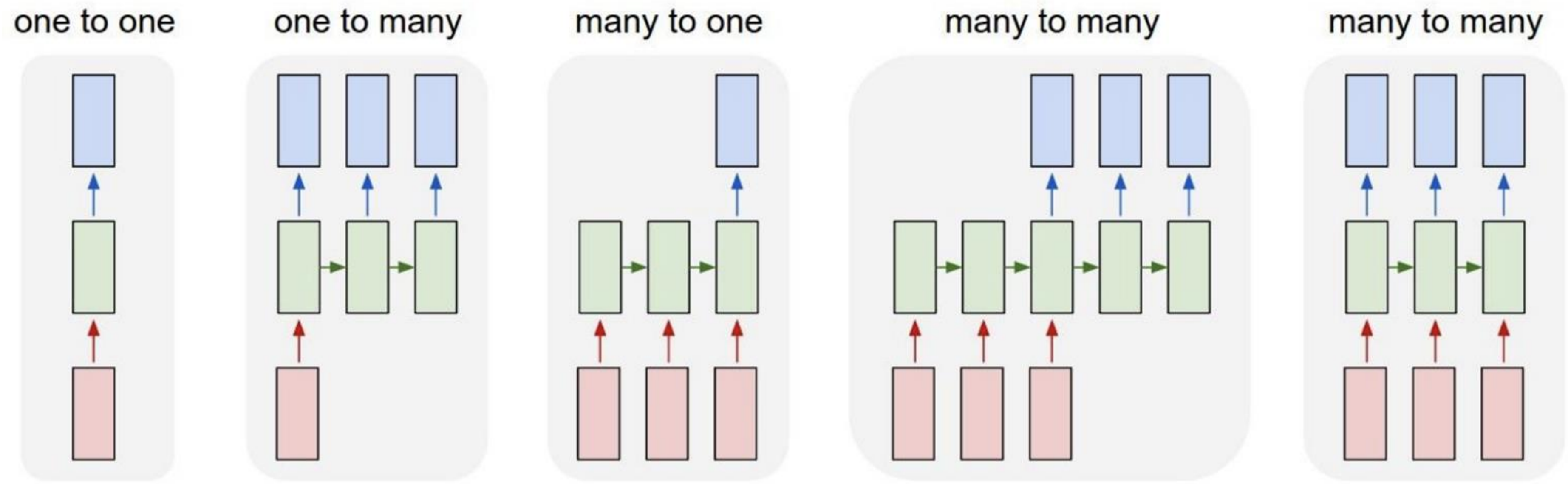
주식과 같은 연속적인(sequential) 시계열(time series) 데이터에 적합

- 랭귀지 모델링(Mikolov et al., 2010, 2011; Sutskever et al., 2011)
- 기계번역(Liu et al., 2014; Auli et al., 2013; Sutskever et al., 2014)
- 음성인식(Robinson et al., 1996; Graves et al., 2013; Graves and Jaitly, 2014; Sak et al., 2014)
- 이미지 캡셔닝(Karpathy and Fei-Fei, 2015)

RNN 활용 (2)

Image Captioning
image -> sequence of words

Machine Translation
seq of words -> seq of words



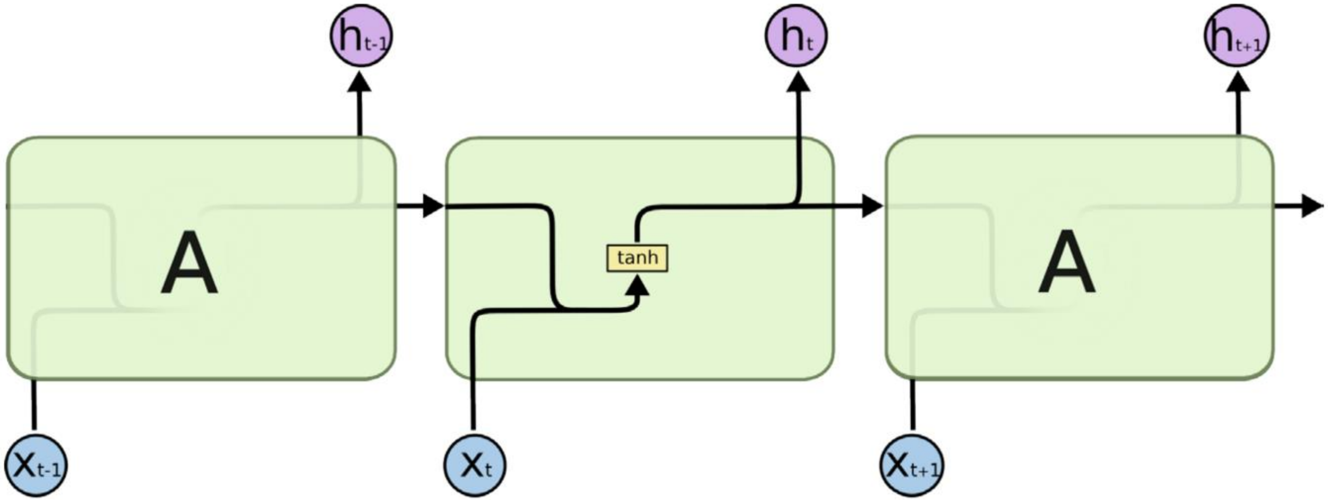
Vanilla Neural Networks

Sentiment Classification
sequence of words -> sentiment

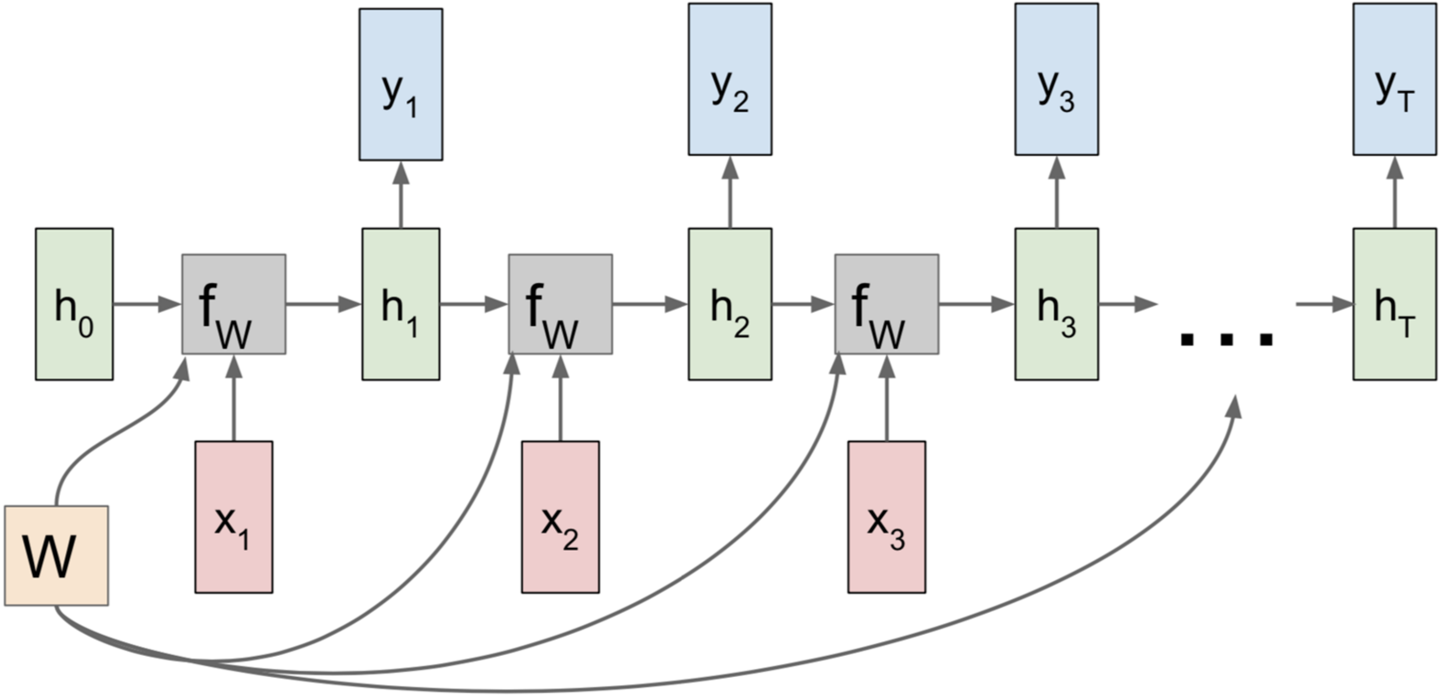
Video classification on frame level

http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture10.pdf

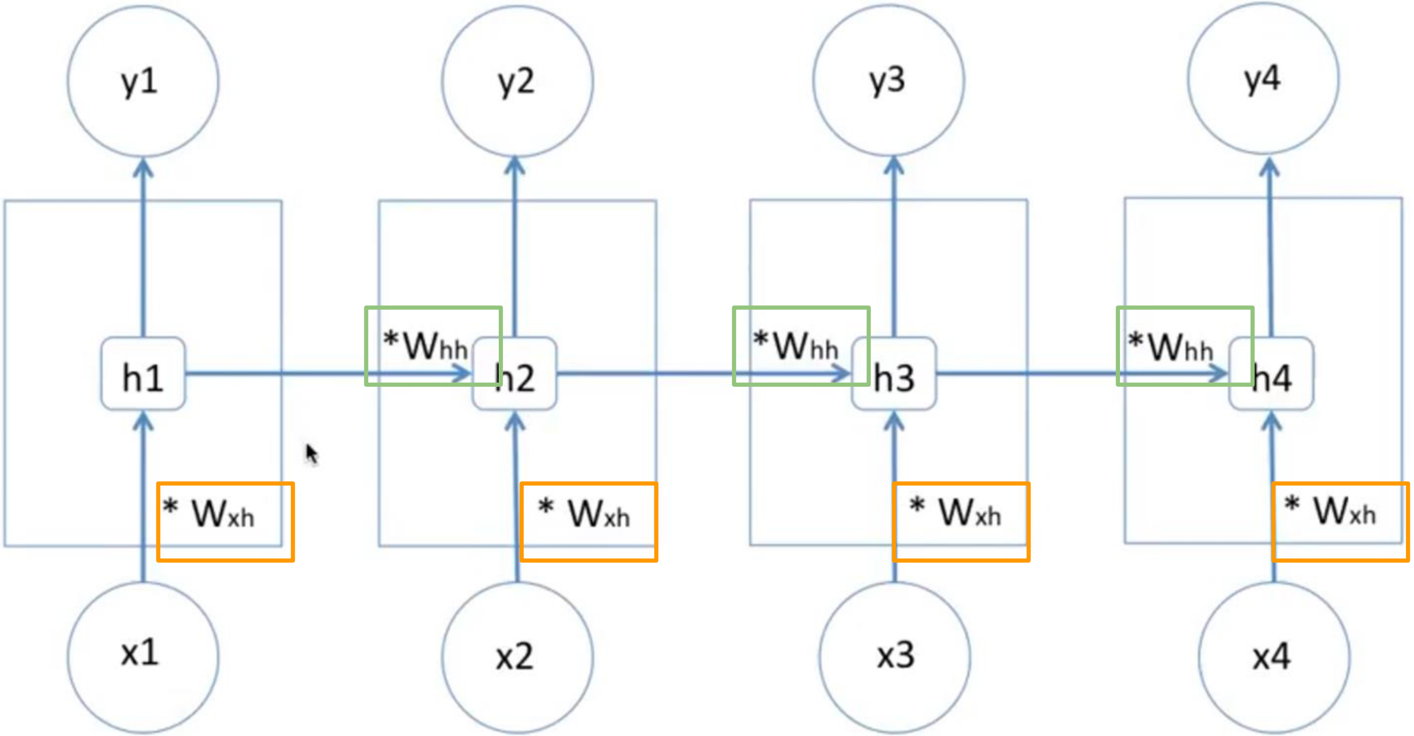
RNN (Recurrent Neural Network)



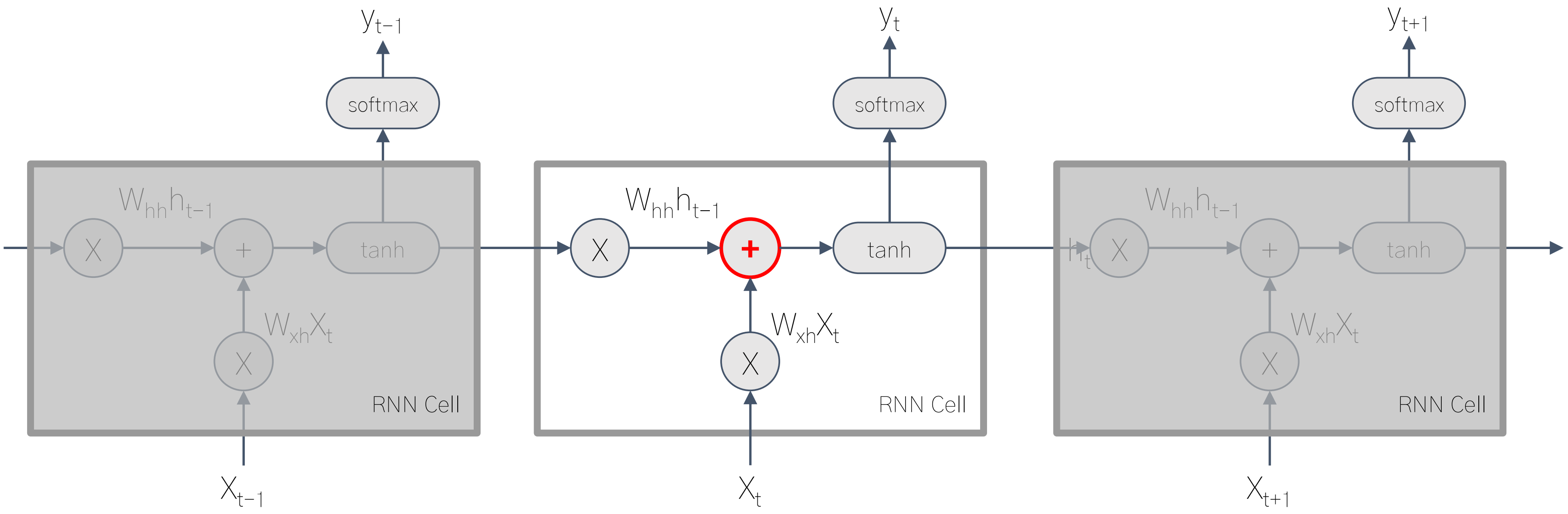
RNN (Recurrent Neural Network)



RNN (Recurrent Neural Network)



RNN (Recurrent Neural Network)



https://github.com/pangolulu/rnn-from-scratch

RNN (Recurrent Neural Network)

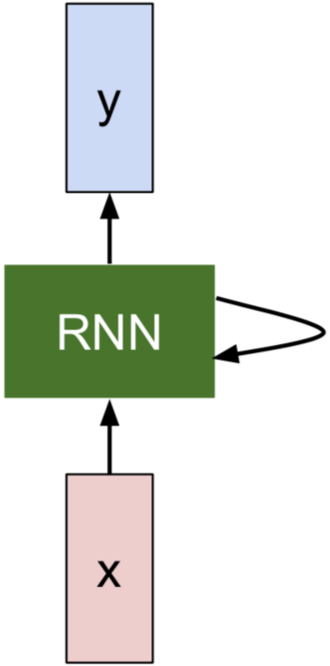
$$\boxed{h_t} = \boxed{f_W}(\boxed{h_{t-1}}, \boxed{x_t})$$

new state

some function
with parameters W

old state

input vector at
some time step

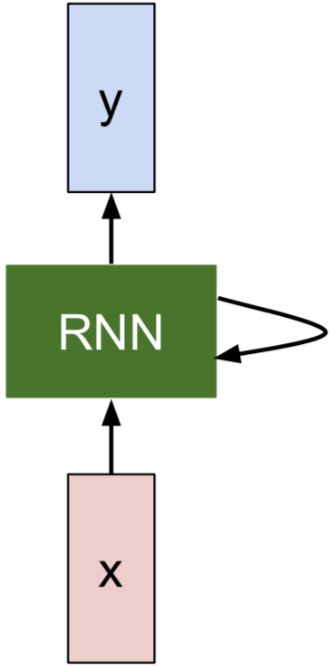


RNN (Recurrent Neural Network)

$$h_t = f_W(h_{t-1}, x_t)$$

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

$$y_t = W_{hy}h_t$$



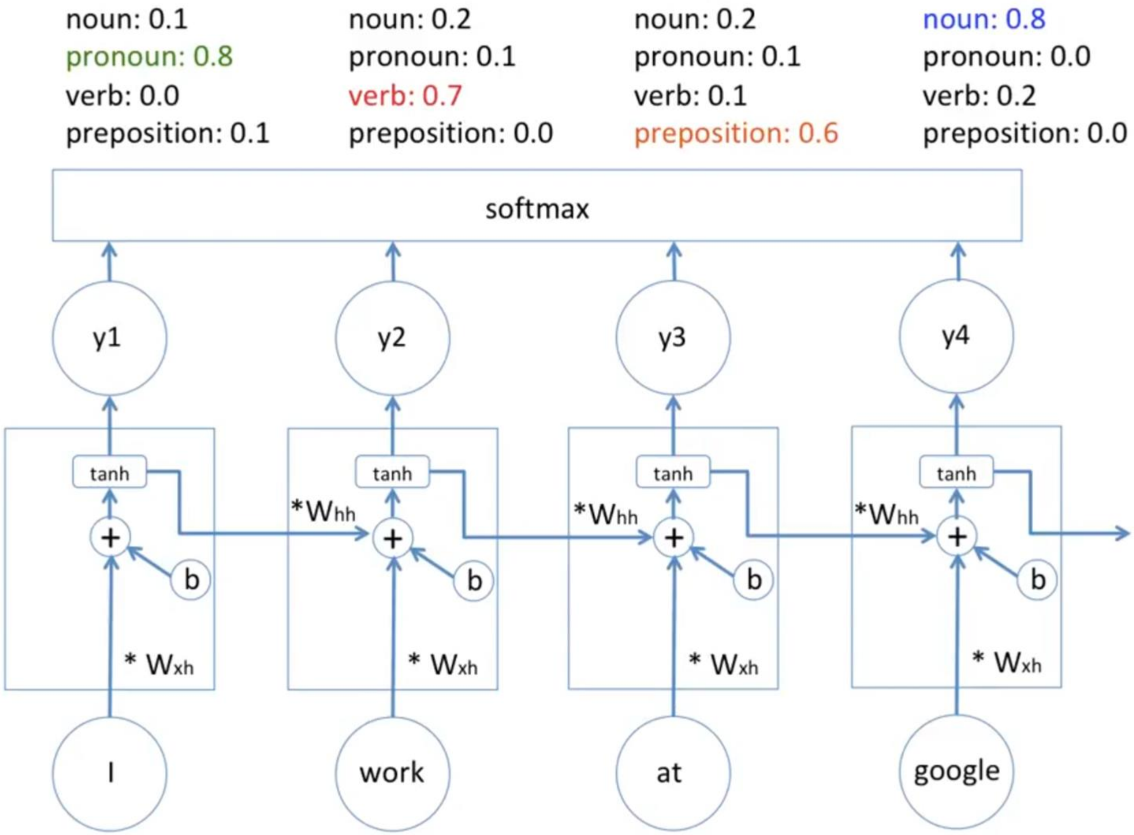
RNN 예시

- 품사 예측

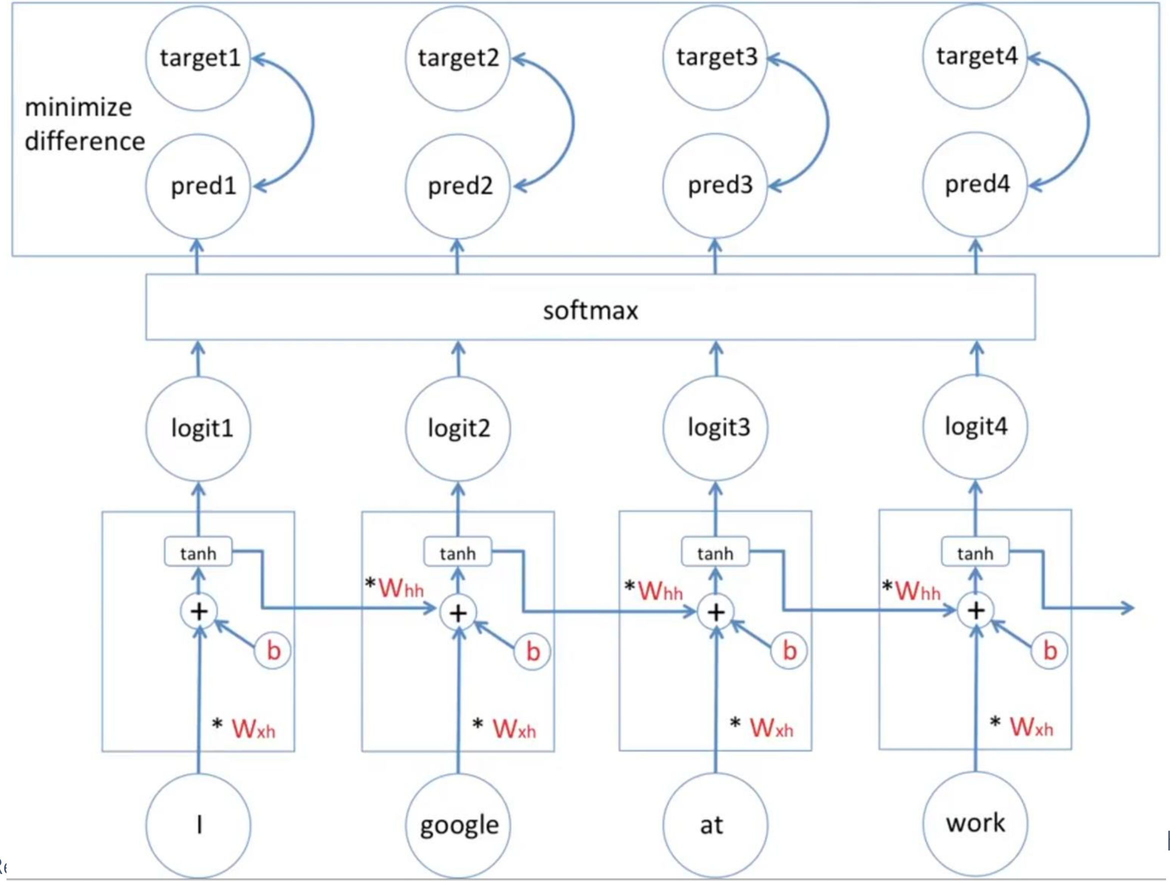
I work at google

pronoun verb preposition noun

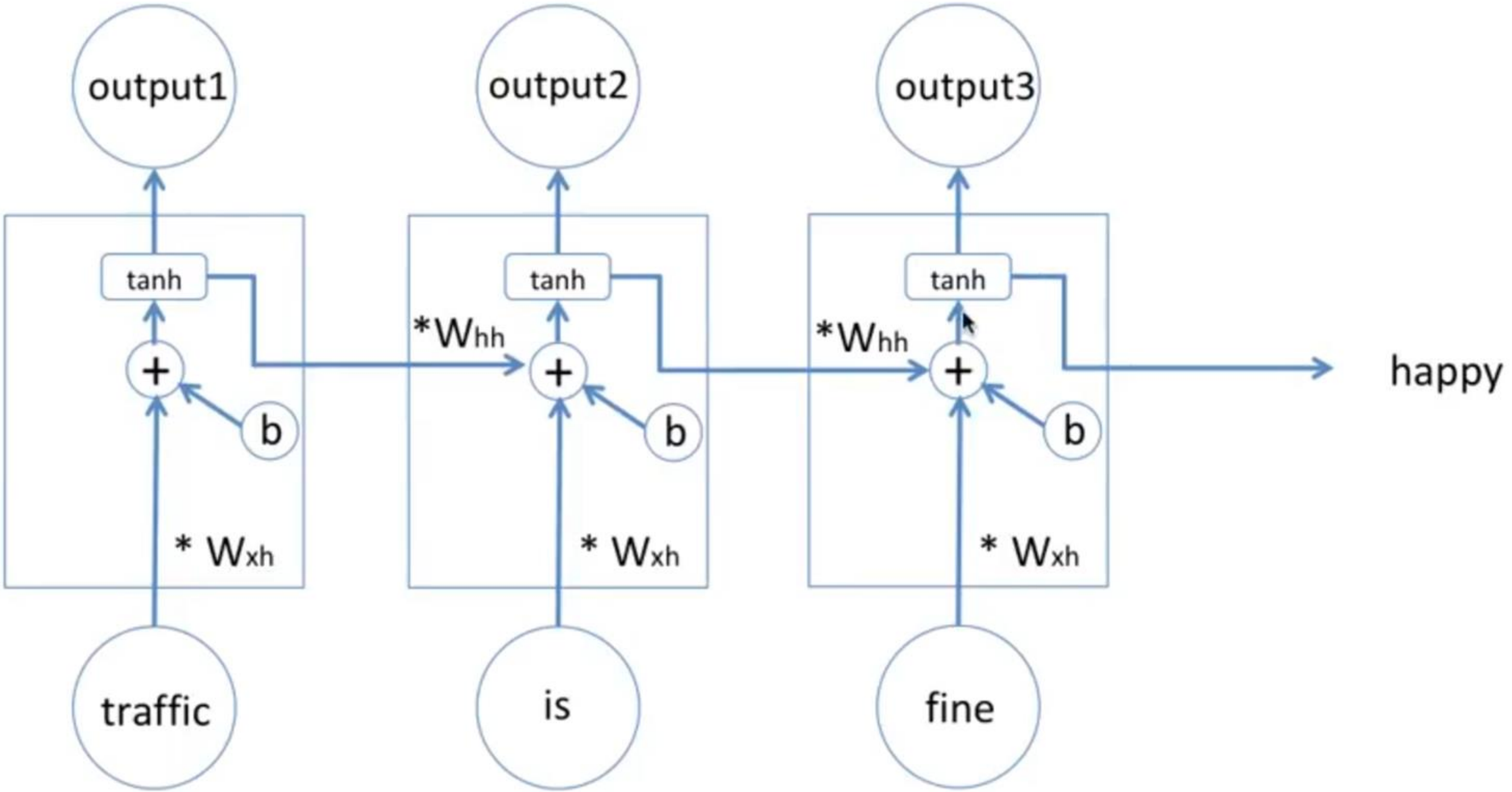
RNN 예시


noun: 0.1
pronoun: 0.8
verb: 0.0
preposition: 0.1
noun: 0.2
pronoun: 0.1
verb: 0.7
preposition: 0.0
noun: 0.2
pronoun: 0.1
verb: 0.1
preposition: 0.6
noun: 0.8
pronoun: 0.0
verb: 0.2
preposition: 0.0
softmax
y1
y2
y3
y4
tanh
*Whh
b
* Wxh
I
work
at
google

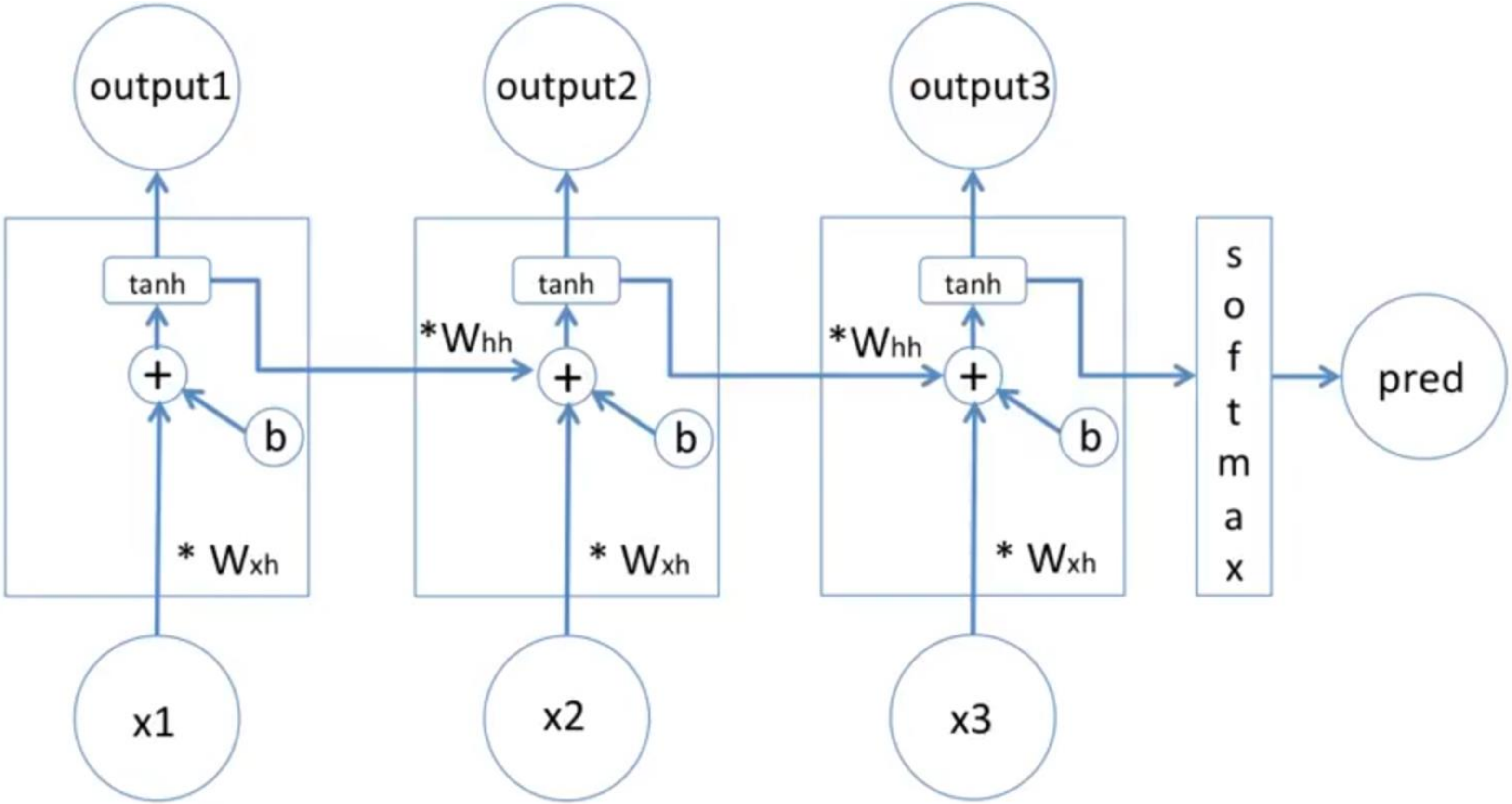
RNN 예시


minimize difference
target1
target2
target3
target4
pred1
pred2
pred3
pred4
softmax
logit1
logit2
logit3
logit4
tanh
tanh
tanh
tanh
*Whh
*Whh
*Whh
b
b
b
b
* Wxh
* Wxh
* Wxh
* Wxh
I
google
at
work

RNN 예시 2


output1
output2
output3
tanh
tanh
tanh
*Whh
*Whh
+
+
+
b
b
b
* Wxh
* Wxh
* Wxh
happy
traffic
is
fine

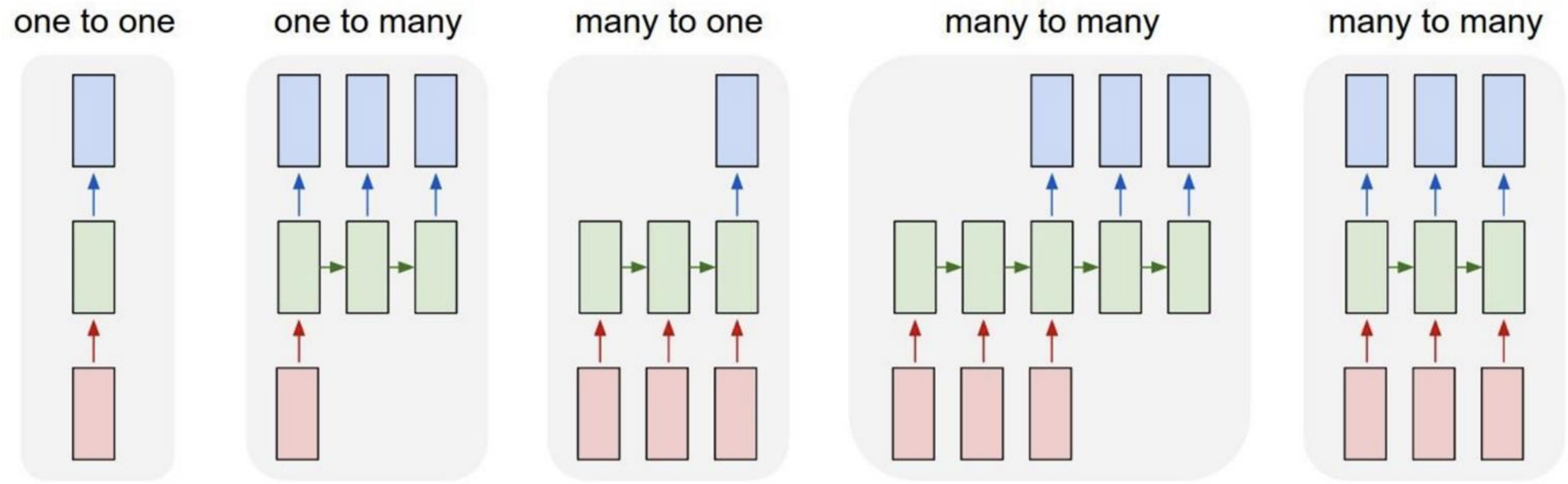
RNN 예시 2


output1
output2
output3
tanh
tanh
tanh
*Whh
*Whh
+
+
+
b
b
b
* Wxh
* Wxh
* Wxh
softmax
pred
x1
x2
x3

RNN 활용 (2)

Image Captioning
image -> sequence of words

Machine Translation
seq of words -> seq of words



Vanilla Neural Networks

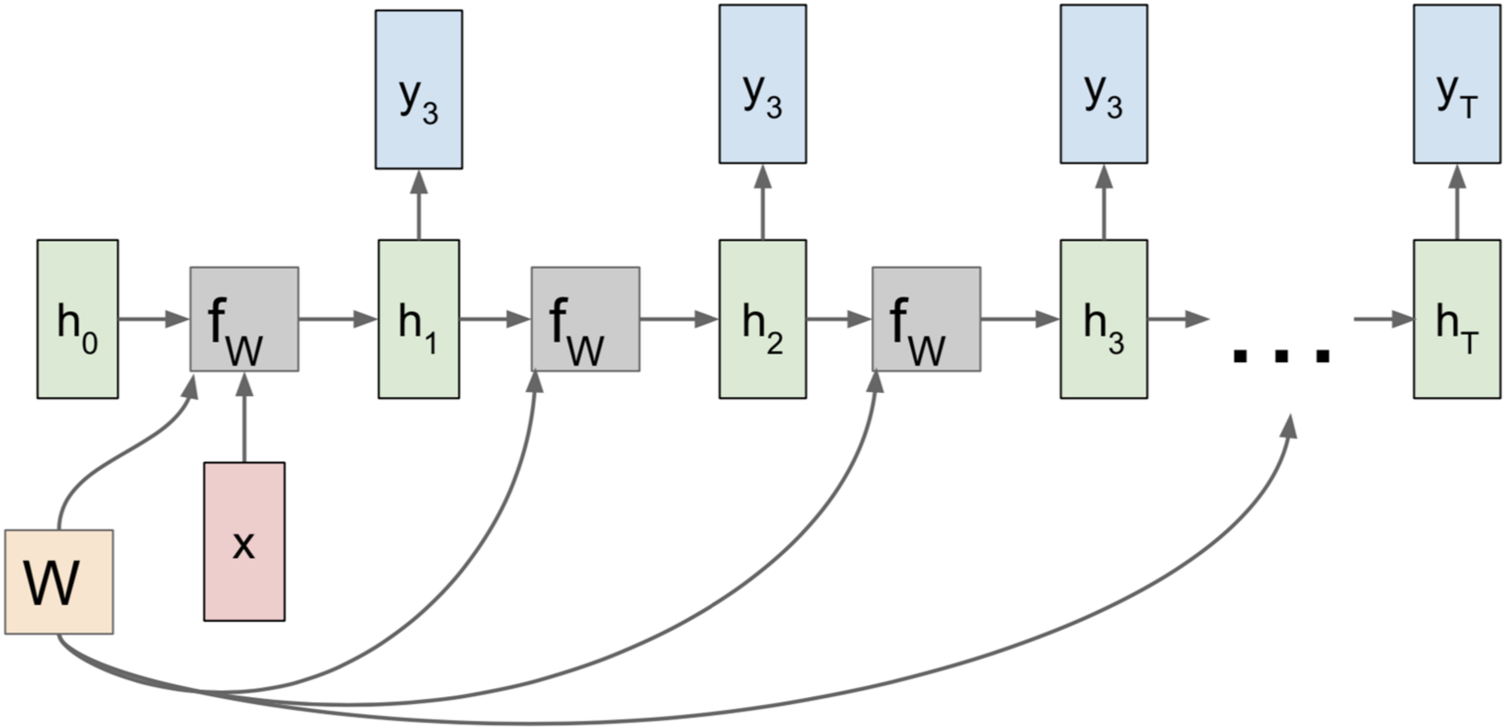
Sentiment Classification
sequence of words -> sentiment

Video classification on frame level

http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture10.pdf

RNN – One to Many
(Recurrent Neural Network)

RNN : One to Many



RNN – Many to One
(Recurrent Neural Network)

RNN : Many to One (1)

Sequence classification

eg. classify polarity of sentence

sequence : sentence, tokens : word

['This movie is good']

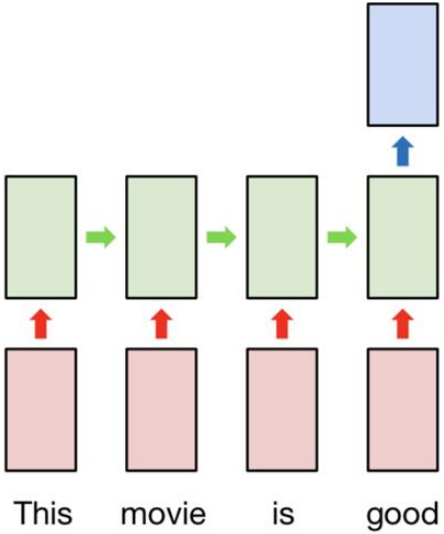
↓ Tokenization

['This', 'movie', 'is', 'good']

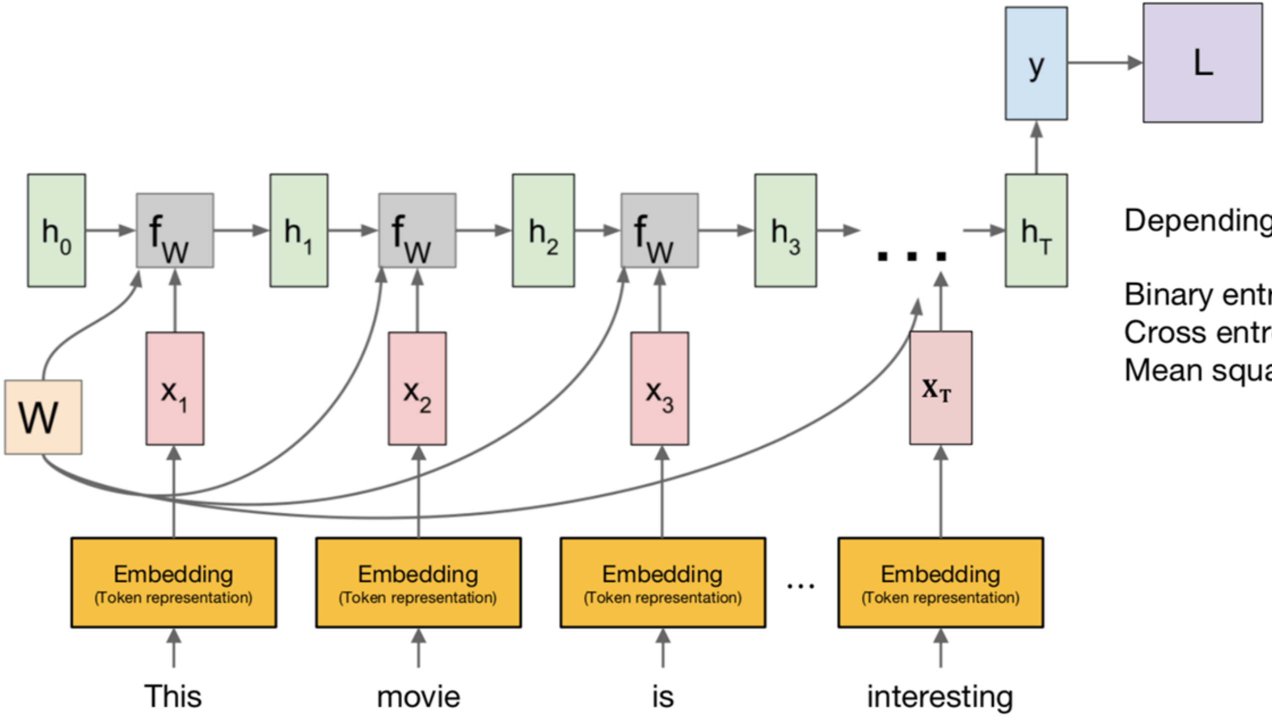
↓ Classification

Positive

Classification : Positive or negative?



RNN : Many to One (2)



Depending on the task…

Binary entropy loss (binary classifier)
Cross entropy loss (softmax classifier)
Mean squared loss (regression)

RNN – Many to One Stacking
(Recurrent Neural Network)

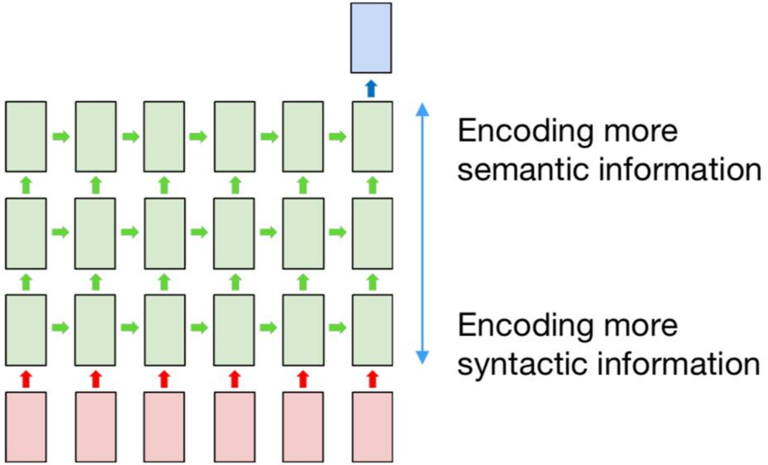
RNN : Many to One Stacking (1)

- What is “stacking”?
- many to one stacking
- Example : sentence classification
  - Preparing dataset
  - Creating and training model
  - Checking performance

RNN : Many to One Stacking (2)

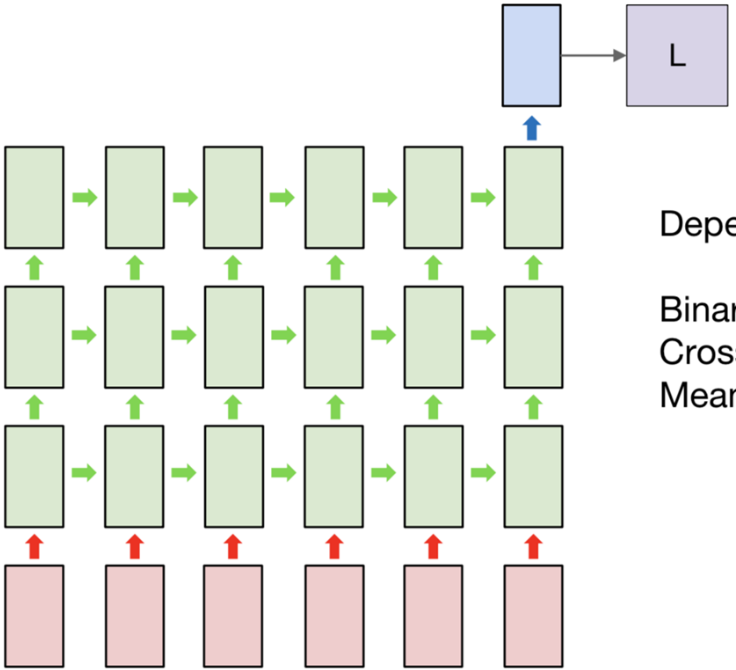
What is “stacking”?

Besides, many works have shown that different layers of deep RNNs encode different types of information.



RNN : Many to One Stacking (3)

many to one stacking



Depending on the task…

Binary entropy loss (binary classifier)
Cross entropy loss (softmax classifier)
Mean squared loss (regression)

RNN – Many to Many
(Recurrent Neural Network)

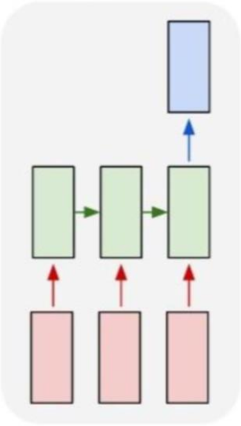
RNN : Many to Many (1)

- What is “many to many”?
- Example : part of speech tagging
  - Preparing dataset
  - Creating and training model
  - Checking performance

RNN : Many to Many (1)

What is “many to many”?

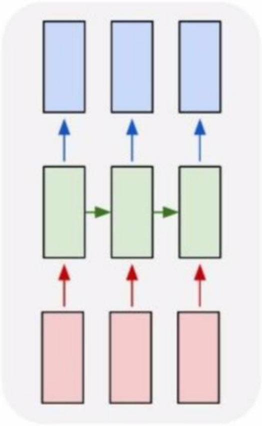
many to one



producing an output for final input it reads in.

VS

many to many



producing an output for each input it reads in.

RNN : Many to Many (2)

What is “many to many”?

Sequence tagging

eg. part of speech tagging

sequence : sentence, tokens : word

[‘tensorflow is very easy’]

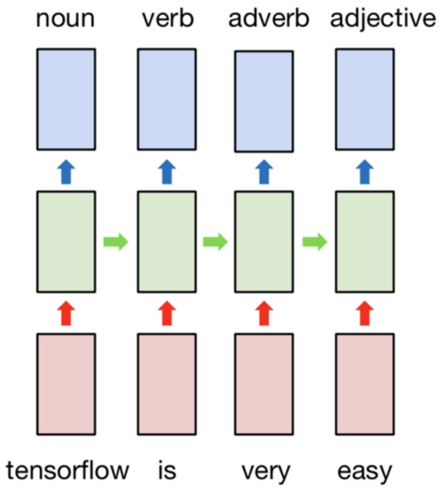
↓ Tokenization

[‘tensorflow’, ‘is’, ‘very’, ‘easy’]

↓ Tagging

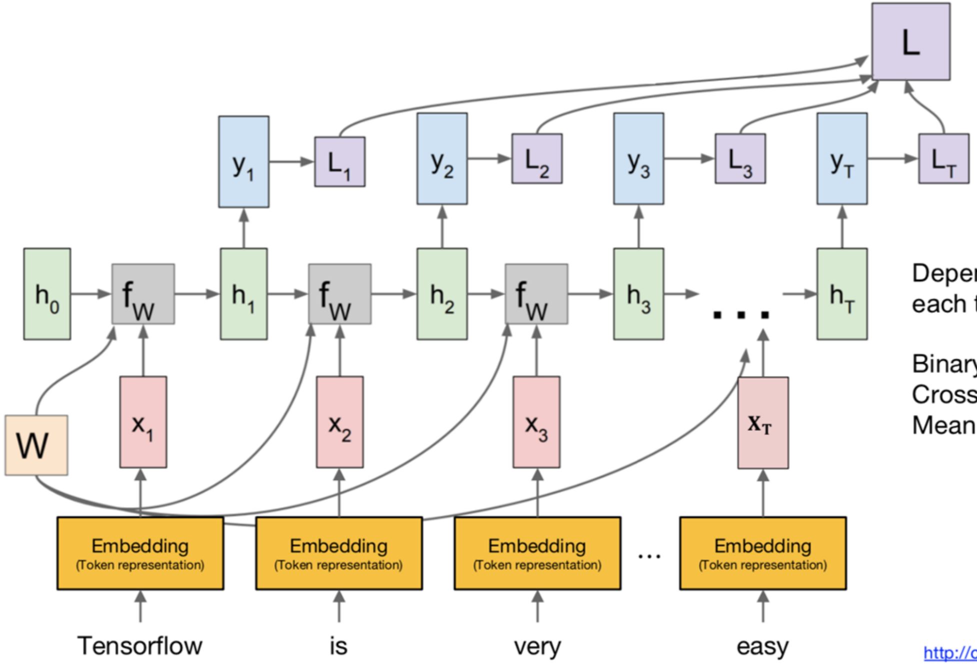
[‘noun’, ‘verb’, ‘adverb’, ‘adjective’]

classification (each time step)



RNN : Many to Many (3)

What is "many to many"?



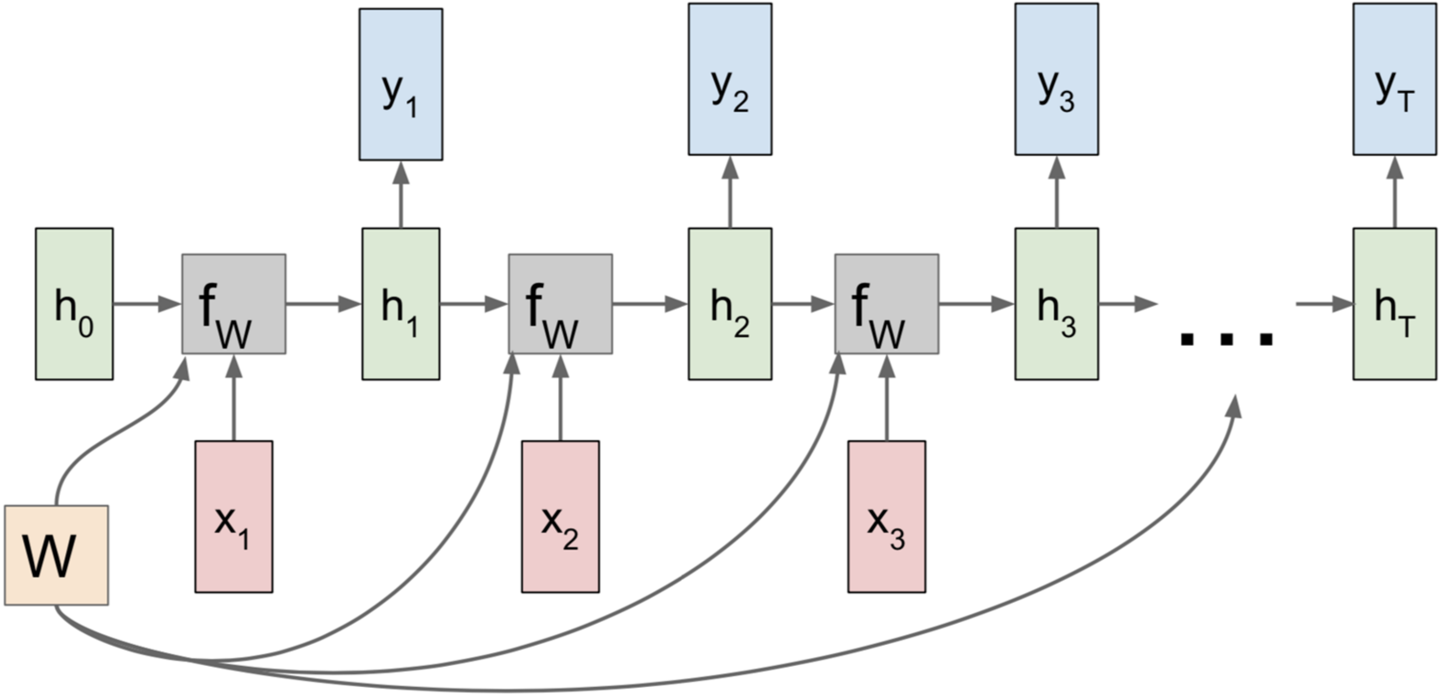
Depending on the task with regard to each time step…

Binary entropy loss (binary classifier)
Cross entropy loss (softmax classifier)
Mean squared loss (regression)

aka Sequence loss!

http://cs231n.stanford.edu/slides/2018/cs231n_2018_lecture10.pdf

RNN : Many to Many (4)



RNN – Many to Many Bidirectional

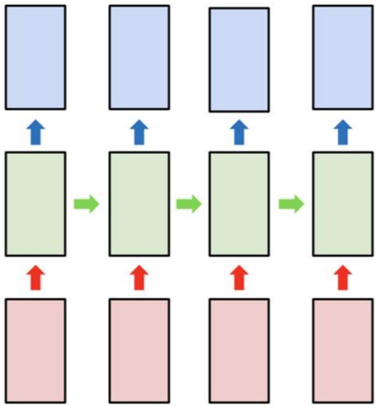
(Recurrent Neural Network)

RNN : Many to Many Bidirectional

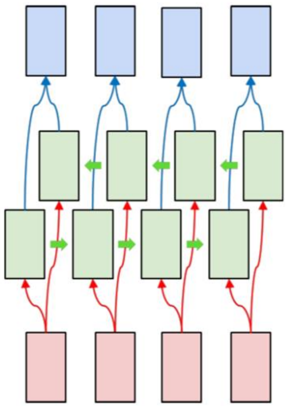
- What is “bidirectional”?
- many to many bidirectional
- Example : part of speech tagging
    - Preparing dataset
    - Creating and training model
    - Checking performance

RNN : Many to Many Bidirectional (1)

What is “bidirectional”?



VS

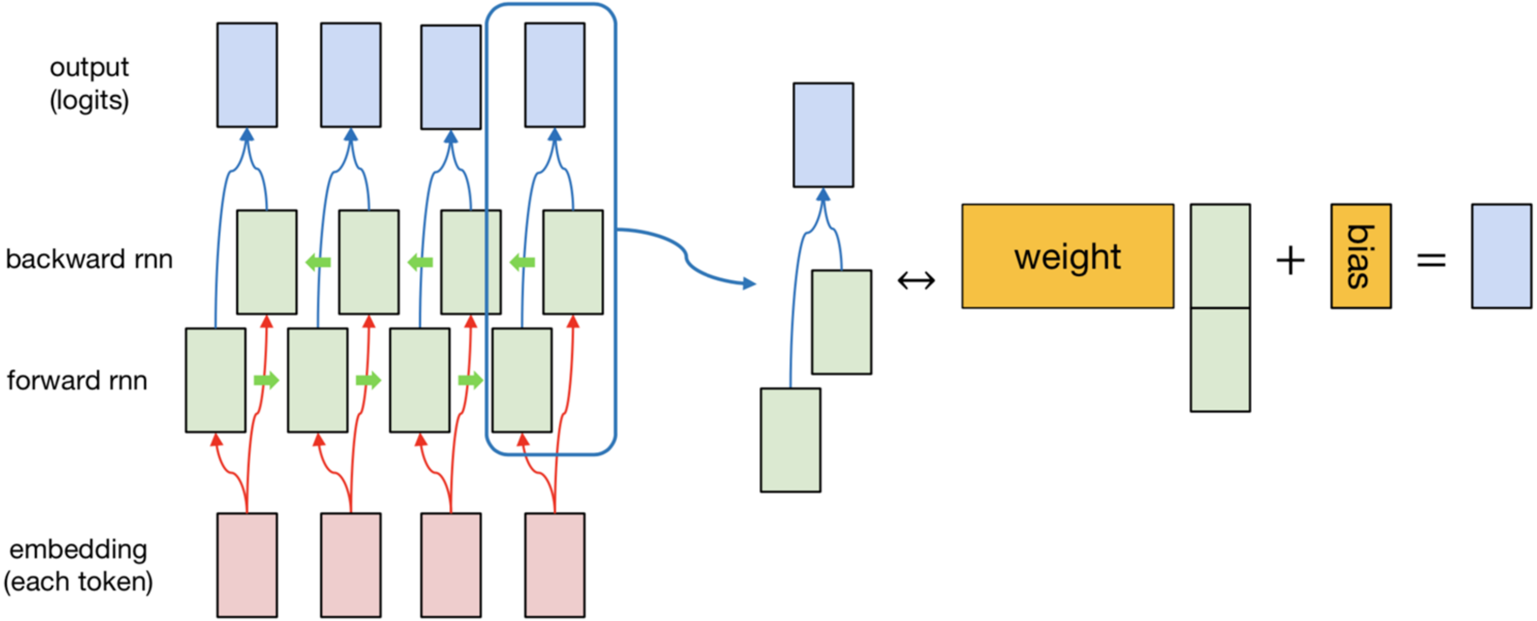


There is imbalance in the amount of information seen by the hidden states at different time steps.

There is balance in the amount of information seen by the hidden states at different time steps.

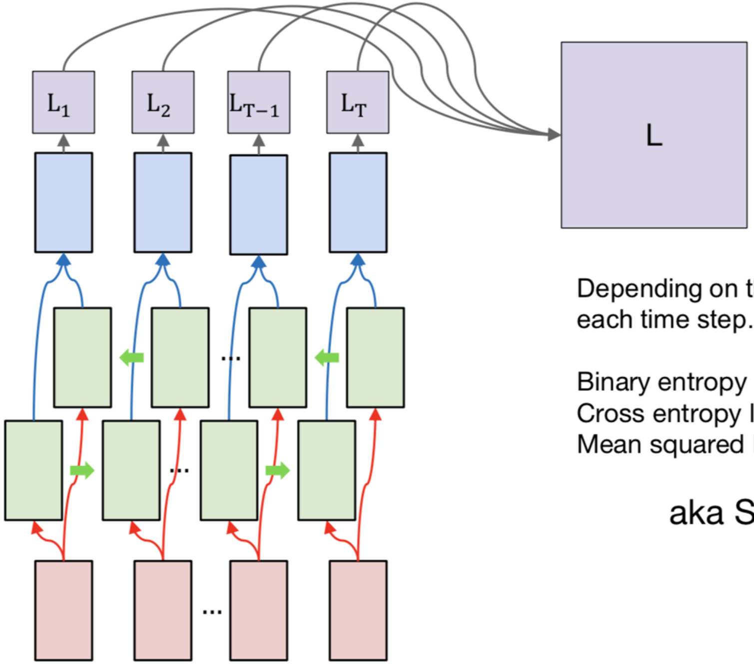
RNN : Many to Many Bidirectional (2)

What is “bidirectional”?



RNN : Many to Many Bidirectional (3)

many to many bidirectional



Depending on the task with regard to each time step…

Binary entropy loss (binary classifier)
Cross entropy loss (softmax classifier)
Mean squared loss (regression)

aka Sequence loss!

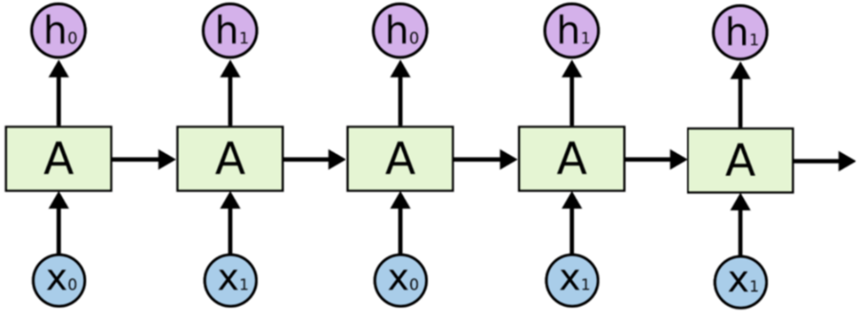
실습

hidden_size=2
sequance_length=5

Using word vector

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

```
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
```

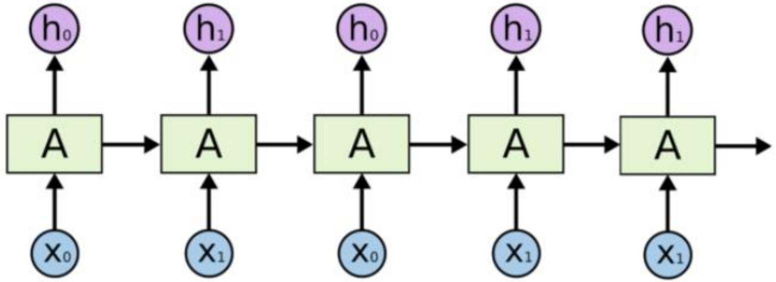
실습

hidden_size=2
sequacne_length=5
batch = 3

Using word vector

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

```
shape=(3,5,2): [[[x,x], [x,x], [x,x], [x,x], [x,x]],
                [[x,x], [x,x], [x,x], [x,x], [x,x]],
                [[x,x], [x,x], [x,x], [x,x], [x,x]]]
```

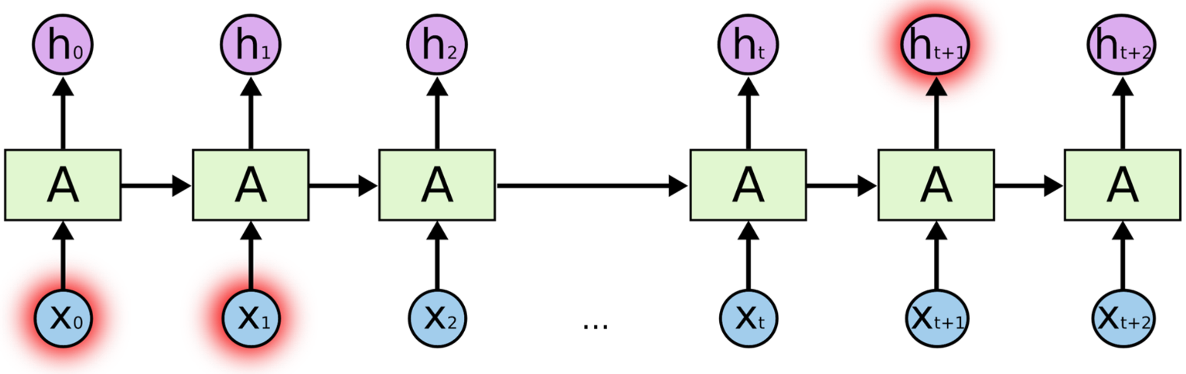



```
shape=(3,5,4): [[[1,0,0,0], [0,1,0,0], [0,0,1,0], [0,0,1,0], [0,0,0,1]], # hello
                [[0,1,0,0], [0,0,0,1], [0,0,1,0], [0,0,1,0], [0,0,1,0]]  # eolll
                [[0,0,1,0], [0,0,1,0], [0,1,0,0], [0,1,0,0], [0,0,1,0]]] # lleel
```

LSTM
(Long short-term memory)

RNN의 문제점

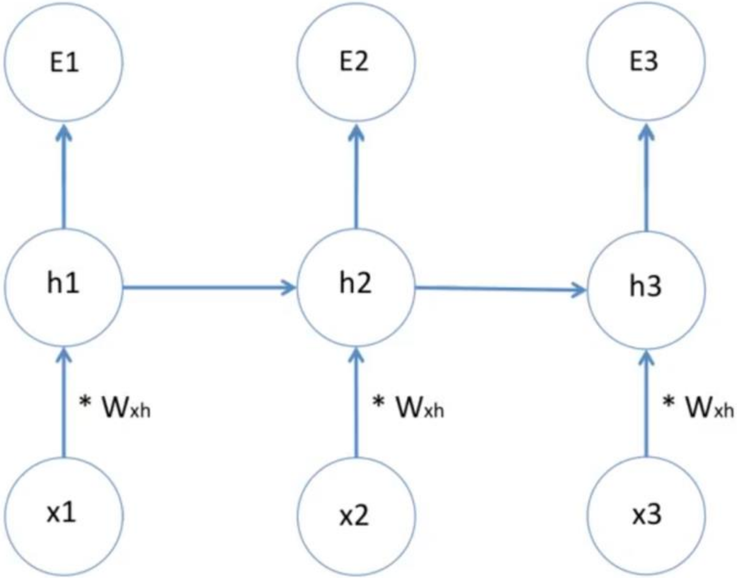
- RNN은 이전 셀의 hidden state를 전달 받음으로써 시계열 데이터 처리에 적합
- 장기의존성(Long dependency) 문제



https://colah.github.io/posts/2015-08-Understanding-LSTMs/

RNN의 문제점

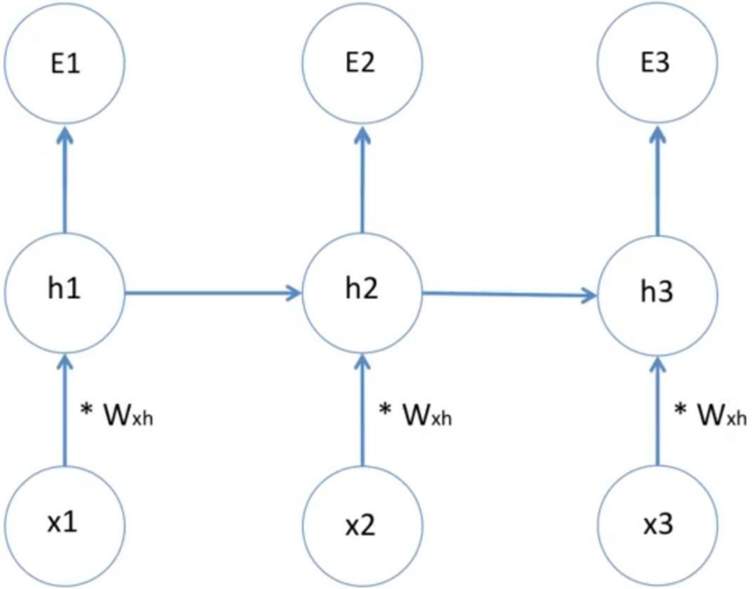
$$W = W - \text{learning\_rate} * \frac{\partial E}{\partial W}$$



https://colah.github.io/posts/2015-08-Understanding-LSTMs/

RNN의 문제점

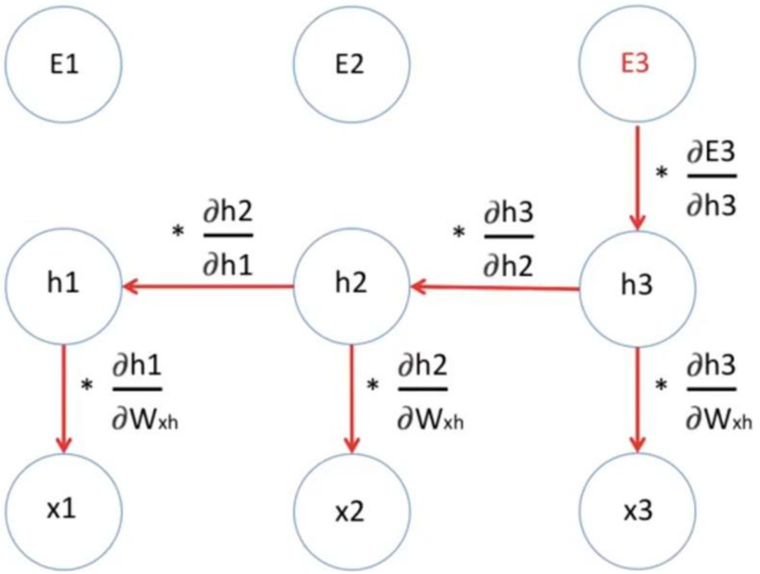
$$W = W - \text{learning\_rate} * \frac{\partial E}{\partial W} \qquad \frac{\partial E}{\partial W} = \frac{\partial E1}{\partial W} + \frac{\partial E2}{\partial W} + \frac{\partial E3}{\partial W}$$



https://colah.github.io/posts/2015-08-Under

RNN의 문제점

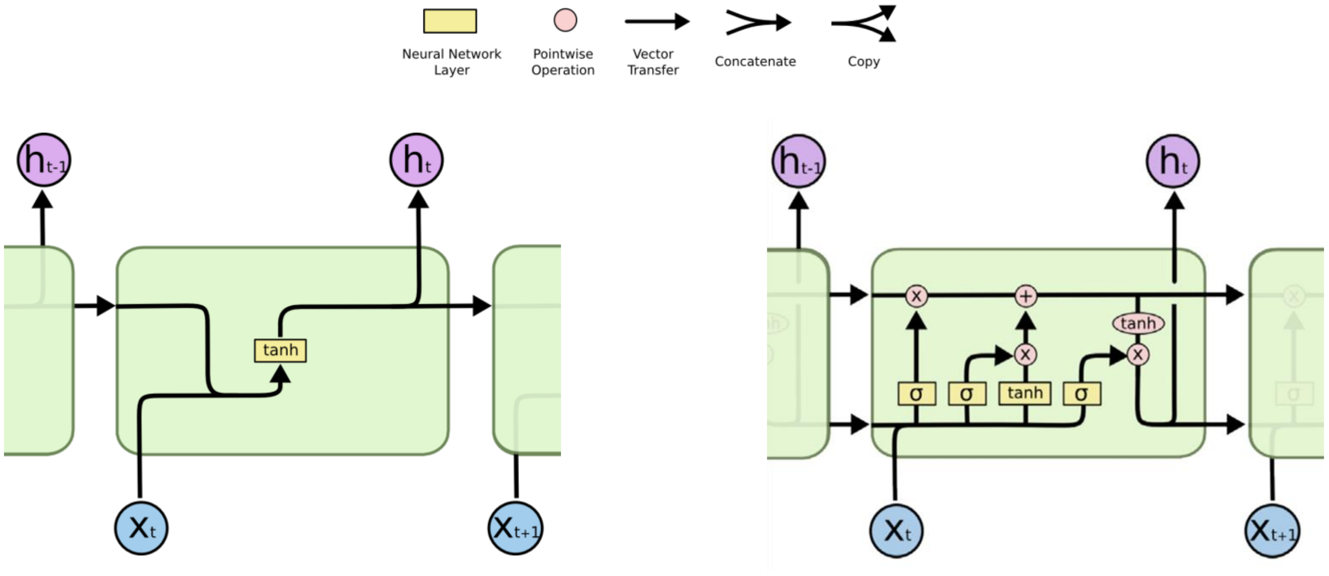
$$\frac{\partial E3}{\partial W} = \frac{\partial E3}{\partial h3} * \frac{\partial h3}{\partial W_{xh}} + \frac{\partial E3}{\partial h3} * \frac{\partial h3}{\partial h2} * \frac{\partial h2}{\partial W_{xh}} + \frac{\partial E3}{\partial h3} * \frac{\partial h3}{\partial h2} * \frac{\partial h2}{\partial h1} * \frac{\partial h1}{\partial W_{xh}}$$



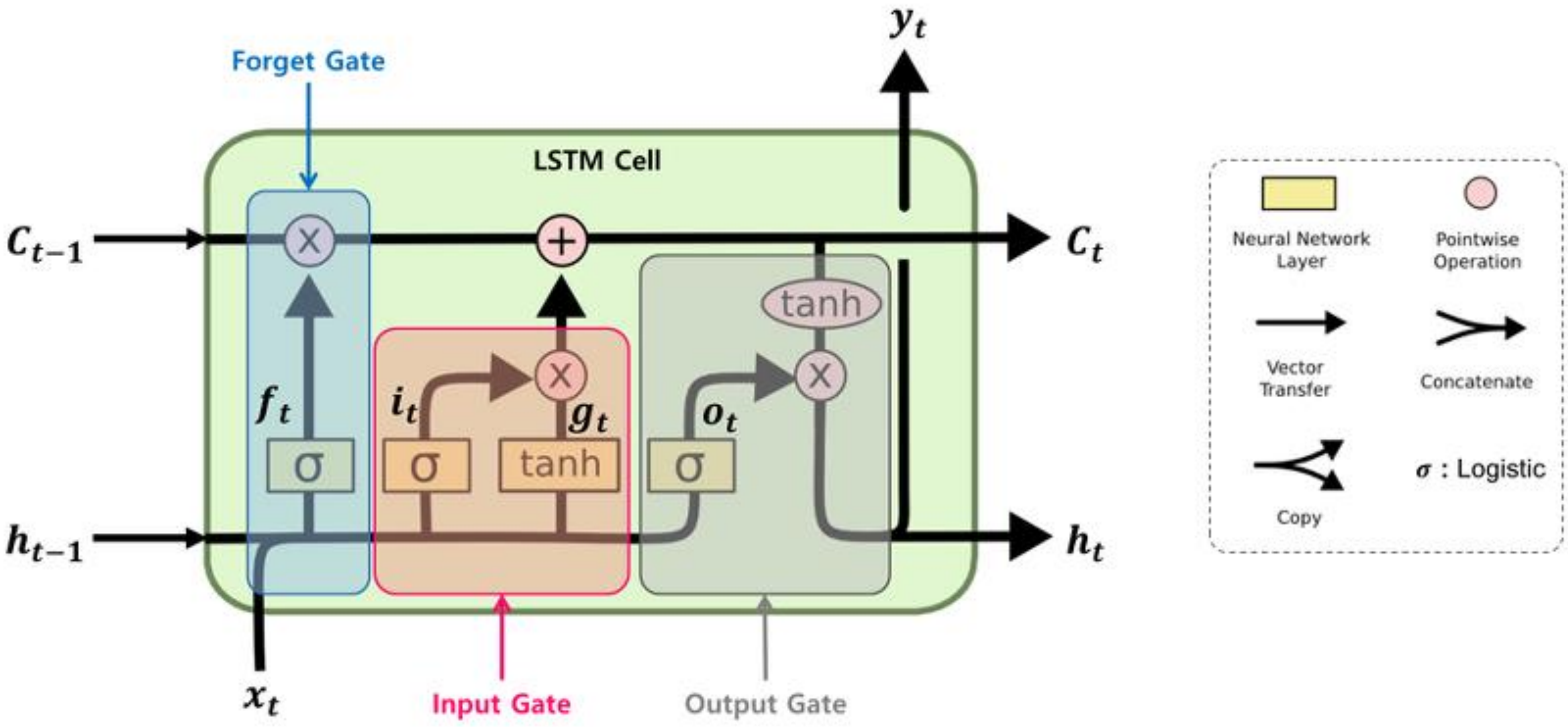
https://colah.github.io/posts/2015-08-Understanding-LSTMs/

LSTM（Long short-term memory)

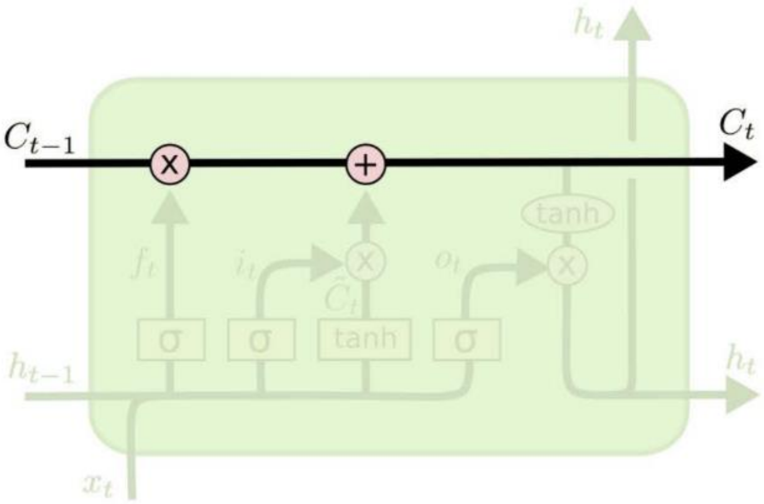
- 장기 단기 메모리 네트워크는 장기적인 종속성을 학습 할 수있는 특수한 종류의 RNN입니다



LSTM 구조 – Cell State

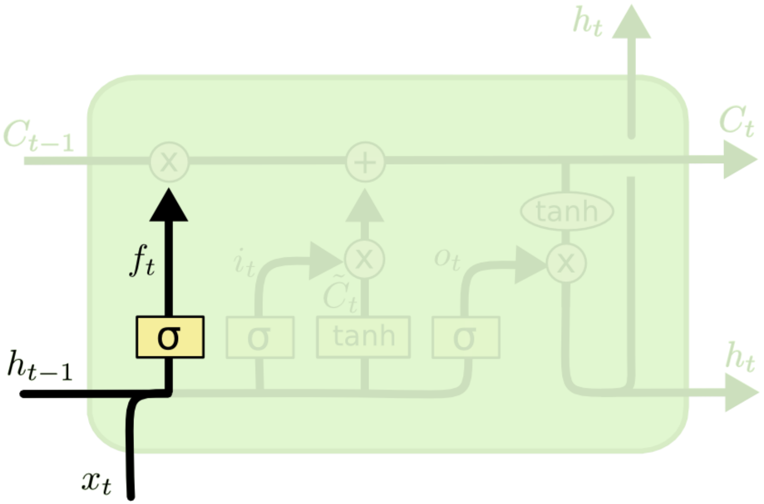


LSTM 구조 – Cell State



- LSTM의 핵심은 Cell state
- 이전 정보를 유지하기 용이
- LSTM은 Input gate를 활용하여 Cell state에 대한 정보를 잊거나 추가 할 수 있음
  - 게이트는 선택적으로 정보를 전달할 수 있는 방법
- 시그모이드 레이어는 0에서 1 사이의 숫자를 출력하여 각 구성 요소를 얼마나 많이 통과시켜야하는지 결정
  - 값이 0 : "아무것도 통과하지 않음"
    값이 1 : "모든 것을 통과"

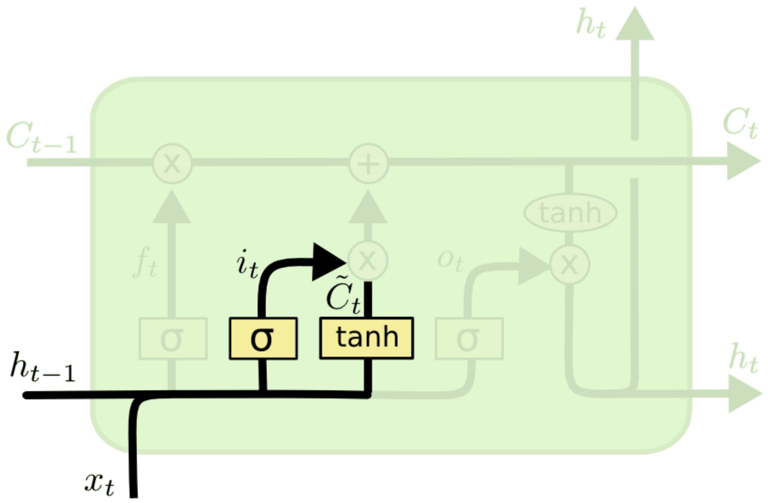
LSTM 구조 – Forget Gate



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

- Cell state의 정보를 버릴지 유지할 지 결정
- Sigmoid를 활용하여 1의 경우 유지, 0인 경우 제거를 의미

LSTM 구조 – Input Gate

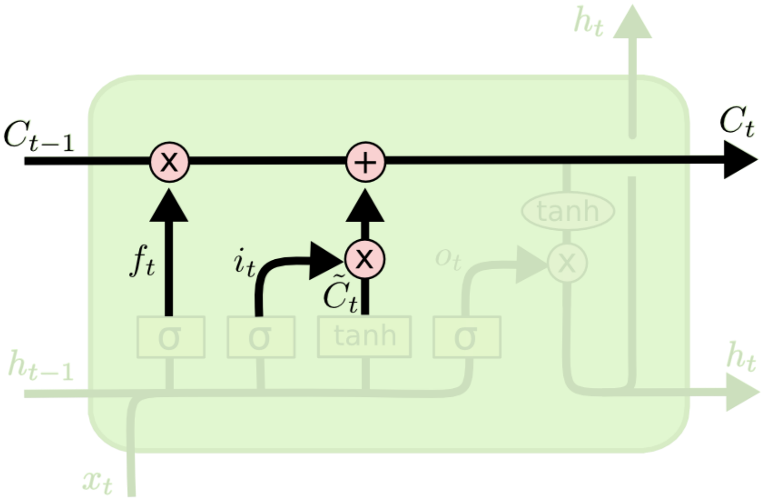


$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \ + \ b_C)$$

- Cell state에 새로운 정보를 추가할 것인지 결정
- Sigmoid : Cell state에 새로운 정보를 얼마나 추가할 것인지 결정
- Tanh : Cell state에 추가할 새로운 정보

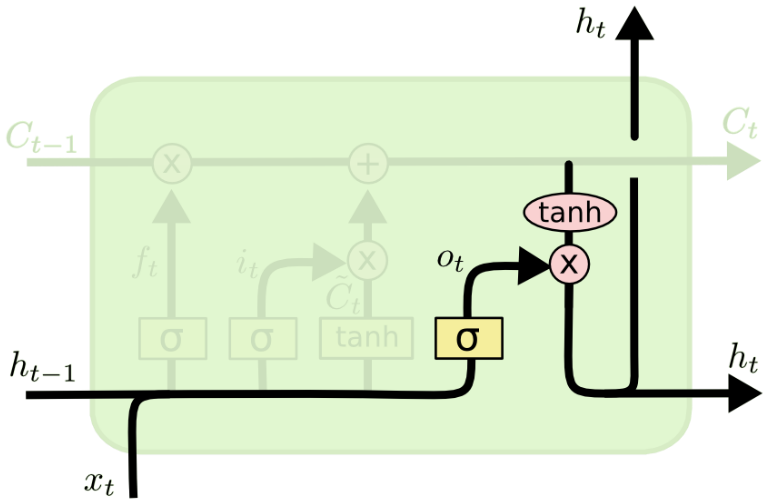
LSTM – Cell State



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

- Forget gate : 이전 Cell state를 얼마나 유지할지 적용
- Input gate : Cell state에 새로운 정보를 얼마나 추가할지 적용

LSTM – Output gate



Cell state 를 얼마나 정보로 출력할지 sigmoid로 결정

$$o_t = \sigma\left(W_o \left[h_{t-1}, x_t\right] + b_o\right)$$

$$h_t = o_t * \tanh\left(C_t\right)$$

https://github.com/tensorflow/nmt

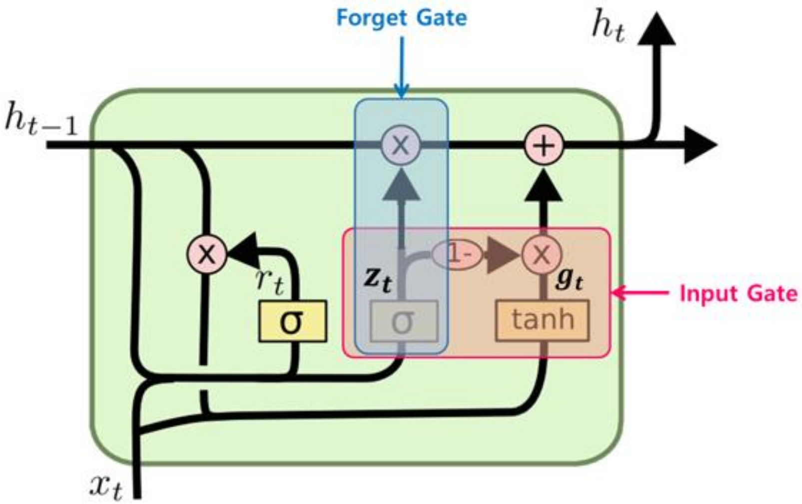
GRU
(Gated Recurrent Unit)

BERT

GRU (Gated Recurrent Unit)

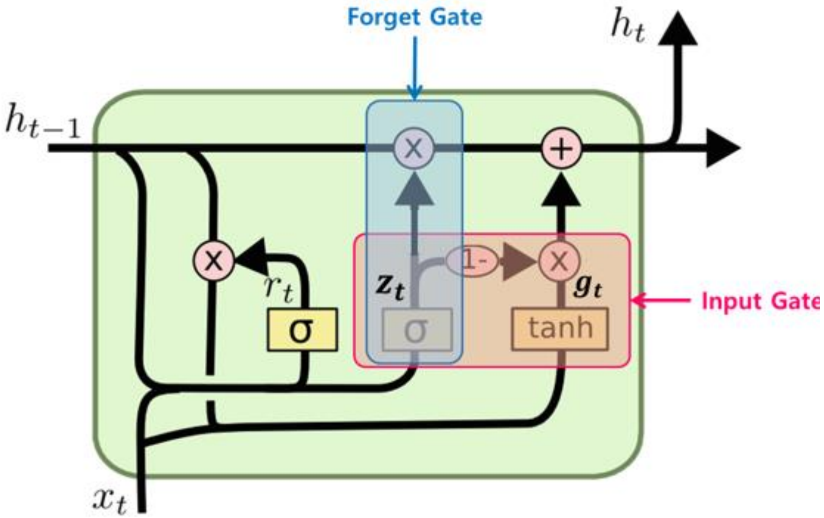
GRU(Gated Recurrent Unit) 셀은 2014년에 K. Cho(조경현) 등에 의해 제안된 LSTM 셀의 간소화된 버전

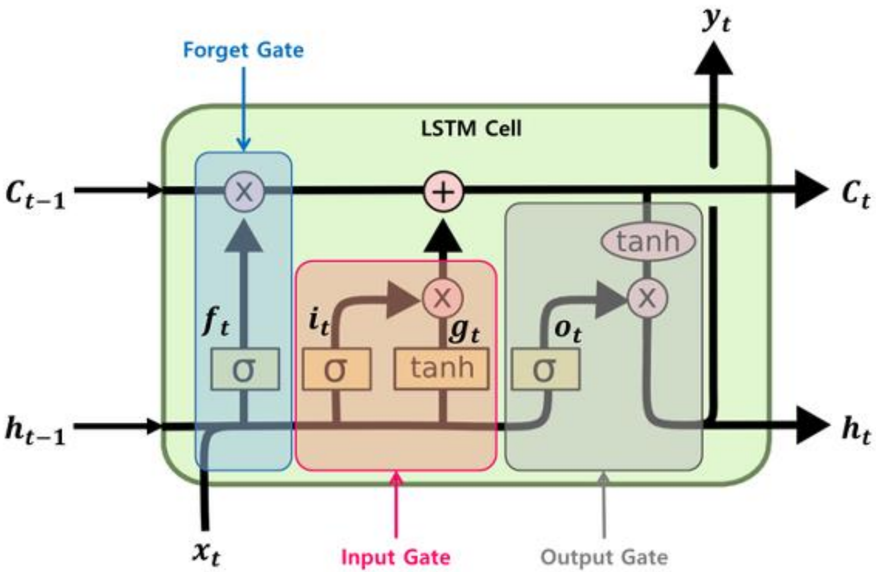
https://arxiv.org/pdf/1406.1078v3.pdf



GRU (Gated Recurrent Unit)

LSTM과 비교하면 유사함을 확인할수 있음





$$
\begin{aligned}
\mathbf{r}_t &= \sigma\left(\mathbf{W}_{xr}^T \cdot \mathbf{x}_t + \mathbf{W}_{hr}^T \cdot \mathbf{h}_{t-1} + \mathbf{b}_r\right) \\
\mathbf{z}_t &= \sigma\left(\mathbf{W}_{xz}^T \cdot \mathbf{x}_t + \mathbf{W}_{hz}^T \cdot \mathbf{h}_{t-1} + \mathbf{b}_z\right) \\
\mathbf{g}_t &= \tanh\left(\mathbf{W}_{xg}^T \cdot \mathbf{x}_t + \mathbf{W}_{hg}^T \cdot (\mathbf{r}_t \otimes \mathbf{h}_{t-1}) + \mathbf{b}_g\right) \\
\mathbf{h}_t &= \mathbf{z}_t \otimes \mathbf{h}_{t-1} + (1 - \mathbf{z}_t) \otimes \mathbf{g}_t
\end{aligned}
$$

RNN vs LSTM vs GRU