순환신경망 : RNN
(Recurrent Neural Network)

순환신경망

- 순차적(sequence)인 데이터를 입력 받아 결과값을 도출하는 데 사용하는 딥러닝 모델.
- 대표적으로 자연어 처리에 상당히 많이 사용.
- 이전 입력 값들(단어들)이 현재 입력 값(단어)의 출력 값(품사)에 영향을 줌.
- ex)

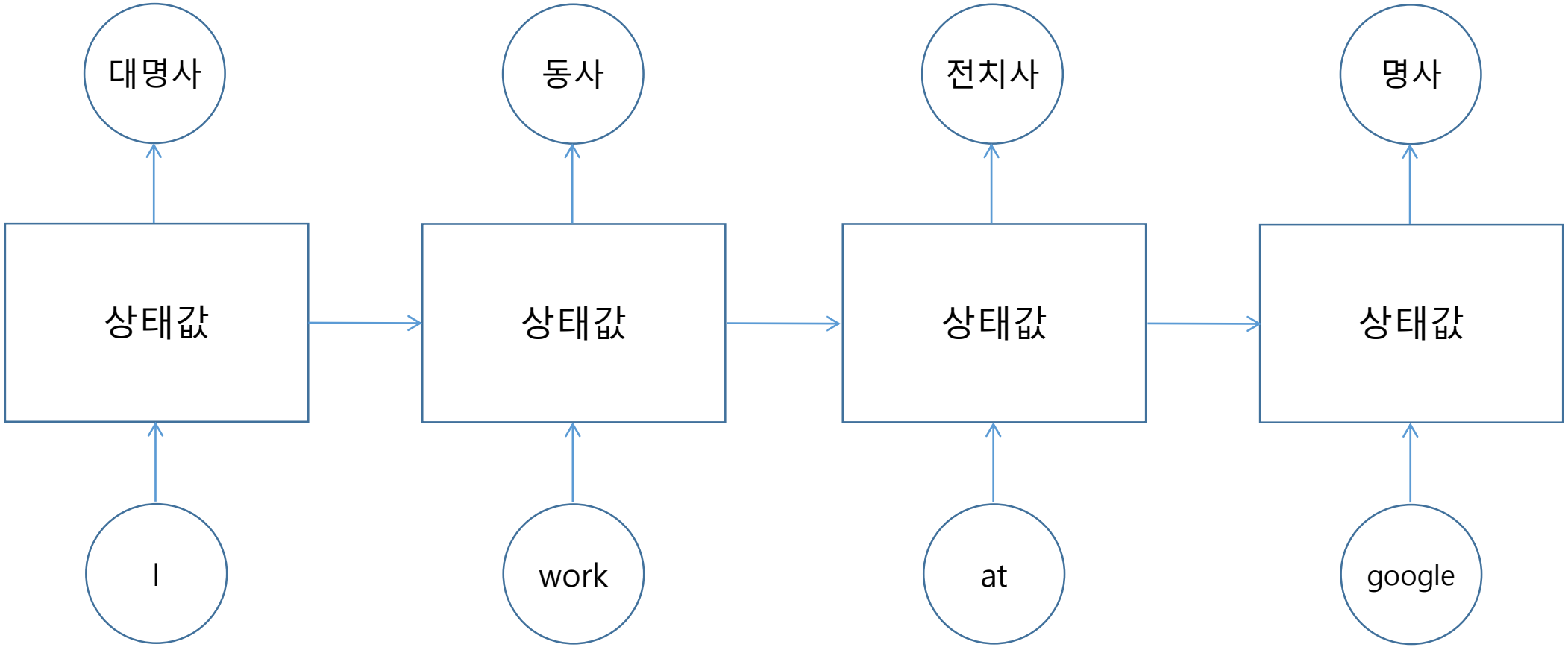
I work at google.

대명사 동사 전치사 명사

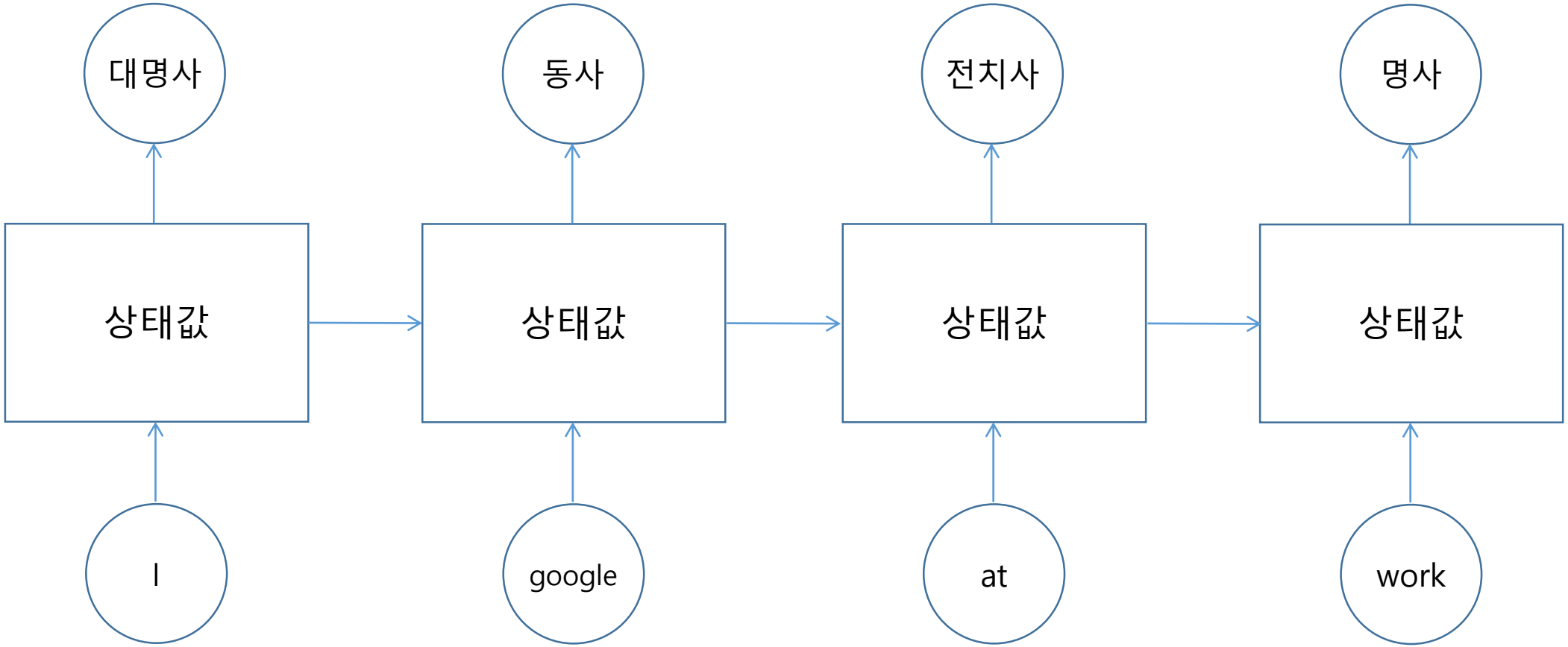
I google at work.

대명사 동사 전치사 명사

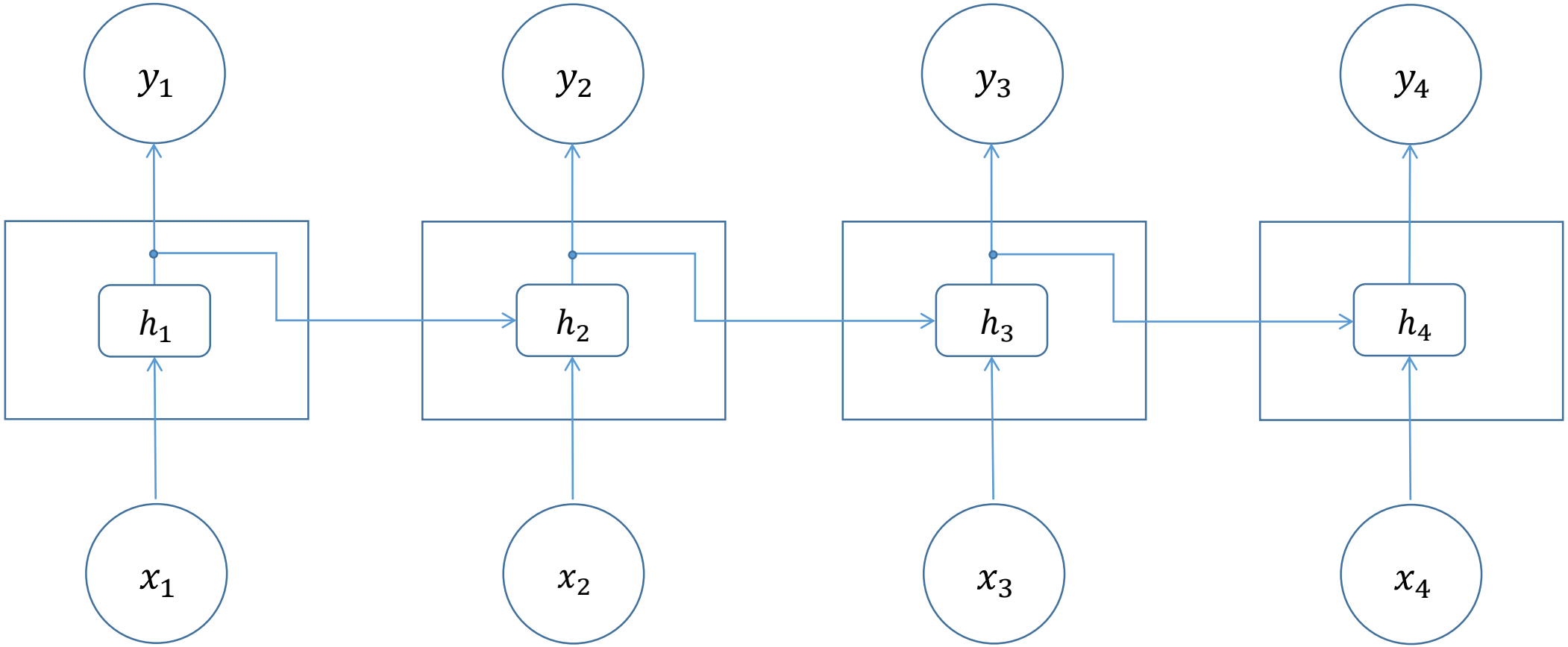
RNN의 “I work at google”의 품사 구분



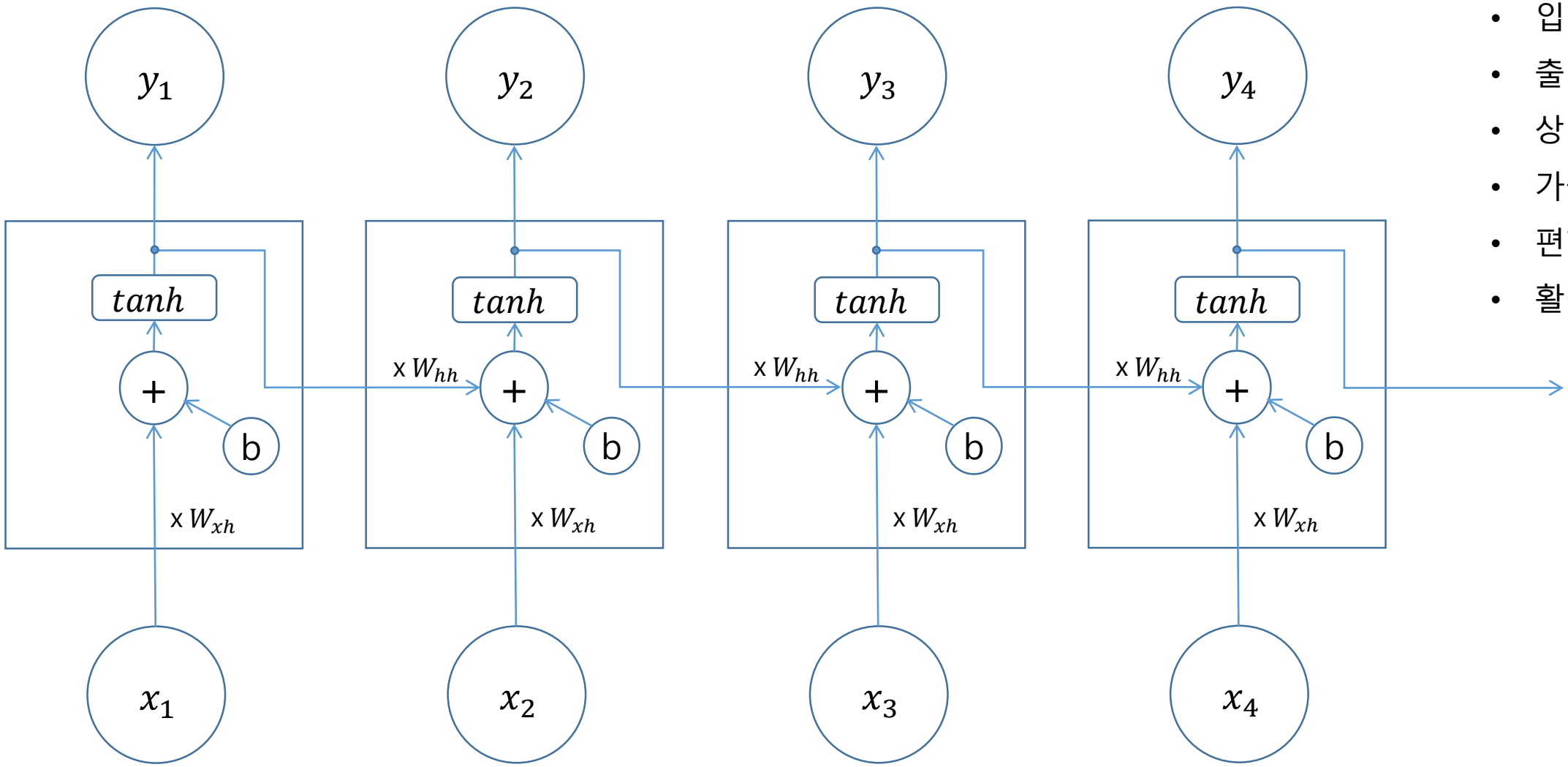
RNN의 “I google at work”의 품사 구분



RNN의 일반적인 구조

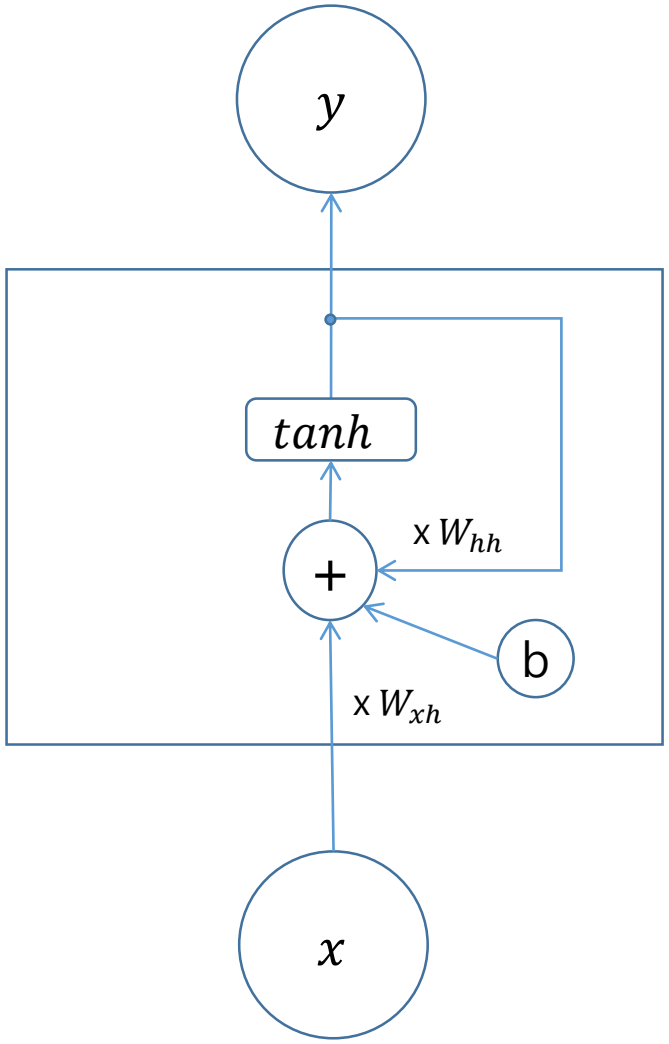


RNN의 심층 구조



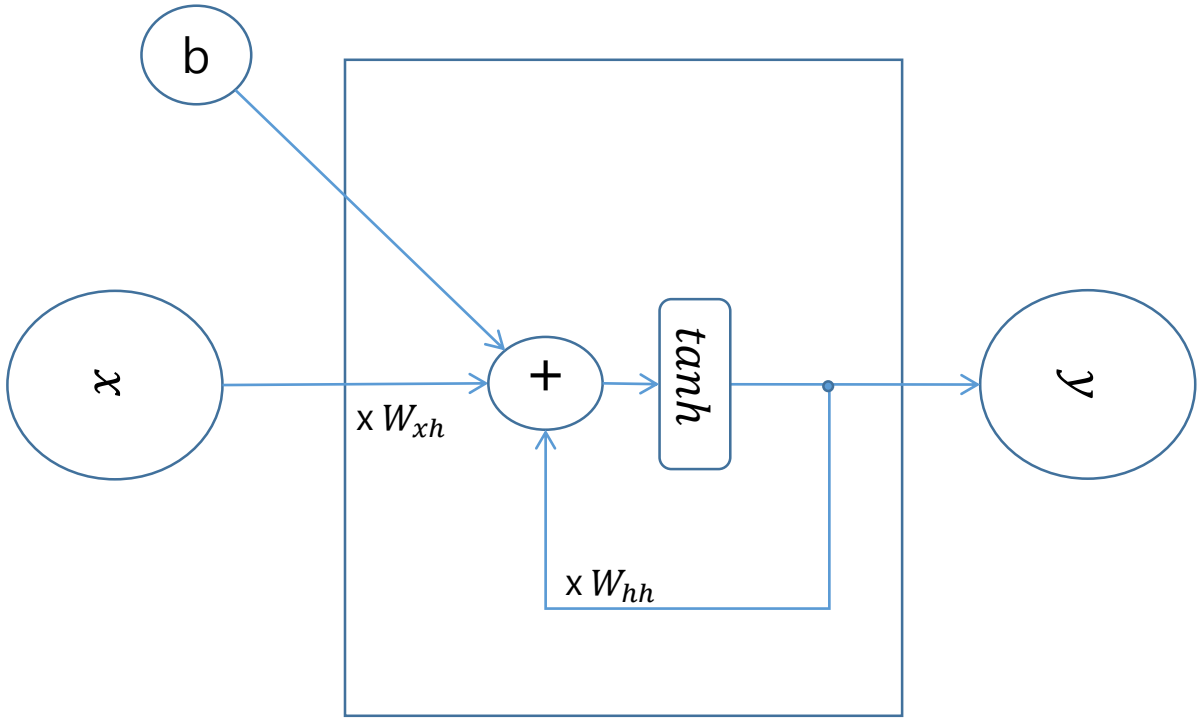
- 입력값(x)
- 출력값(y)
- 상태값(h)
- 가중치(W)
- 편향값(b)
- 활성화함수($tanh$)

RNN의 심층 구조



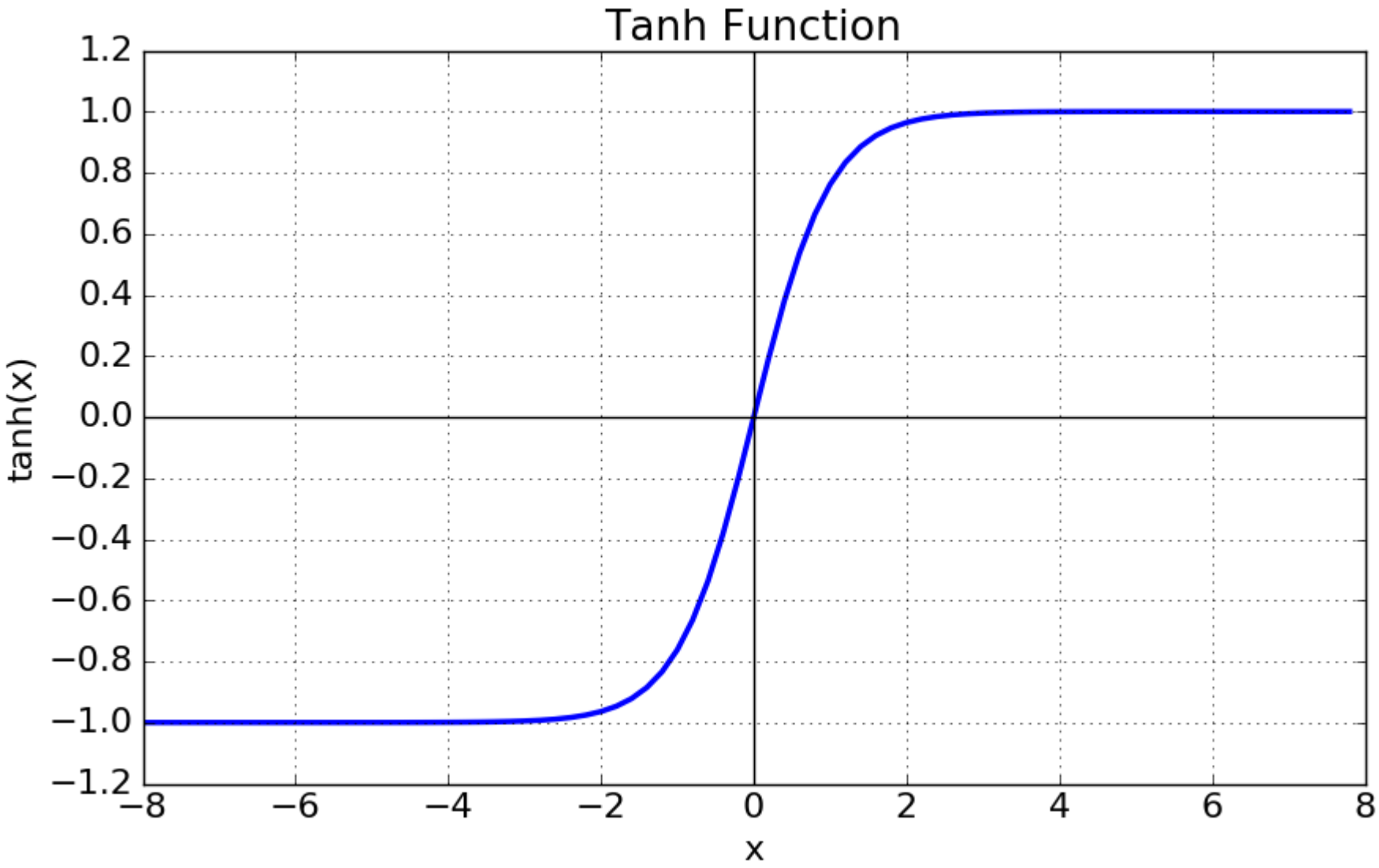
- 입력값(x)
- 출력값(y)
- 상태값(h)
- 가중치(W)
- 편향값(b)
- 활성화함수($tanh$)

RNN의 심층 구조

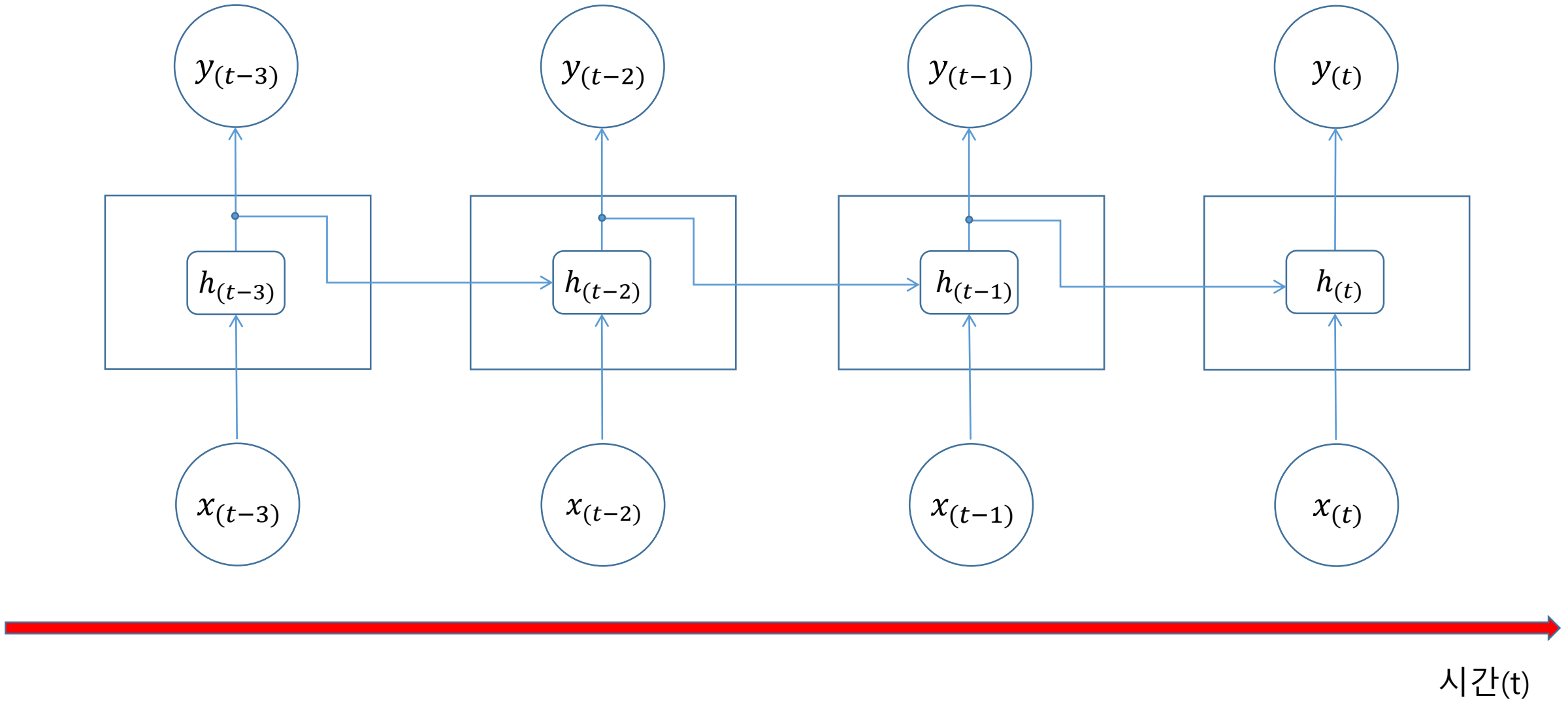


- 입력값(x)
- 출력값(y)
- 상태값(h)
- 가중치(W)
- 편향값(b)
- 활성화함수($tanh$)

RNN의 활성화 함수 - $\tanh(x)$



RNN의 일반적인 구조(time step 혹은 frame 고려)



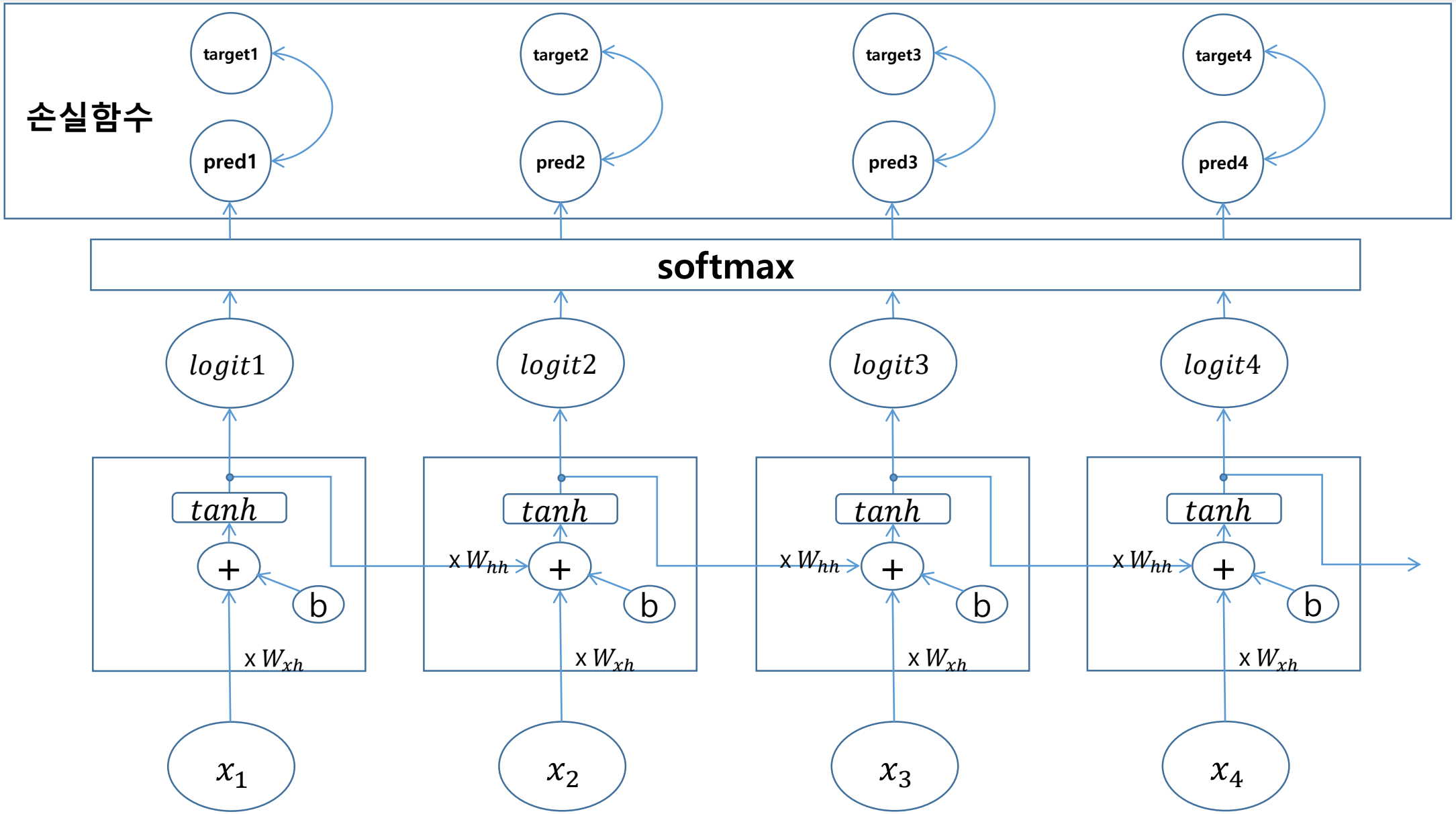
RNN의 구조 – 메모리 셀

- 메모리 셀(memory cell 혹은 간단히 cell)
  - 타임 스텝 t에서 순환 뉴런의 출력은 이전 타임 스텝의 모든 입력에 대한 함수이기 때문에 이를 일종의 메모리 형태라고 말할 수 있음.
  - **타임 스텝에 걸쳐서 어떤 상태를 보존하는 신경망의 구성 요소.**
  - 보통 RNN에서 셀이라고 말할 때는 완전 연결 신경망에서 층(layer)을 의미.

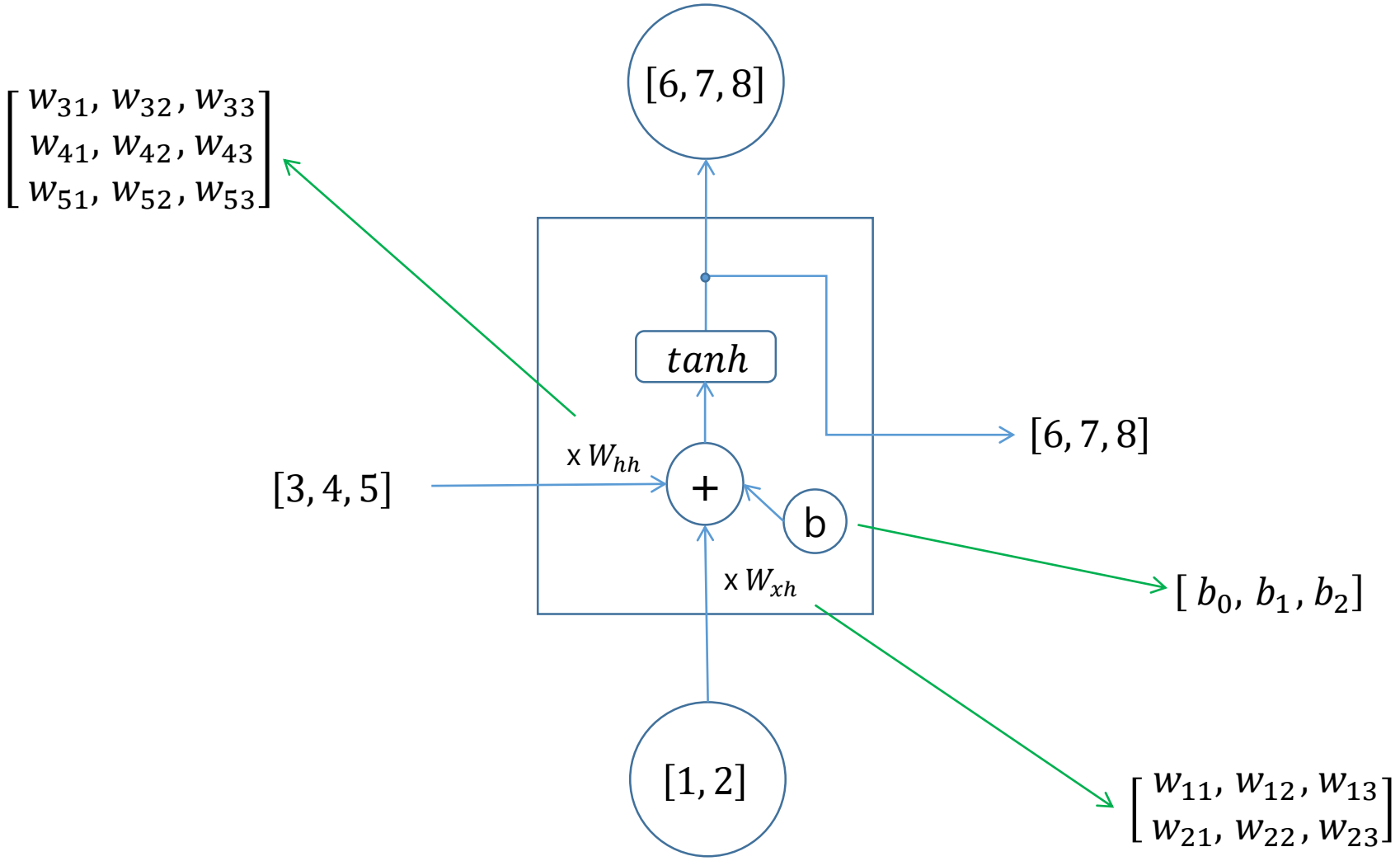
- 일반적으로 타임 스텝 t에서의 셀의 상태 h_t(h 는 hidden을 의미).

$$h_t = f\left(h_{(t-1)},\ x_{(t)}\right)$$

RNN의 최적화 과정을 통한 파라미터 조정



예) RNN 네트워크에 셀이 한 개일 경우의 가중치, 상태값, 출력값



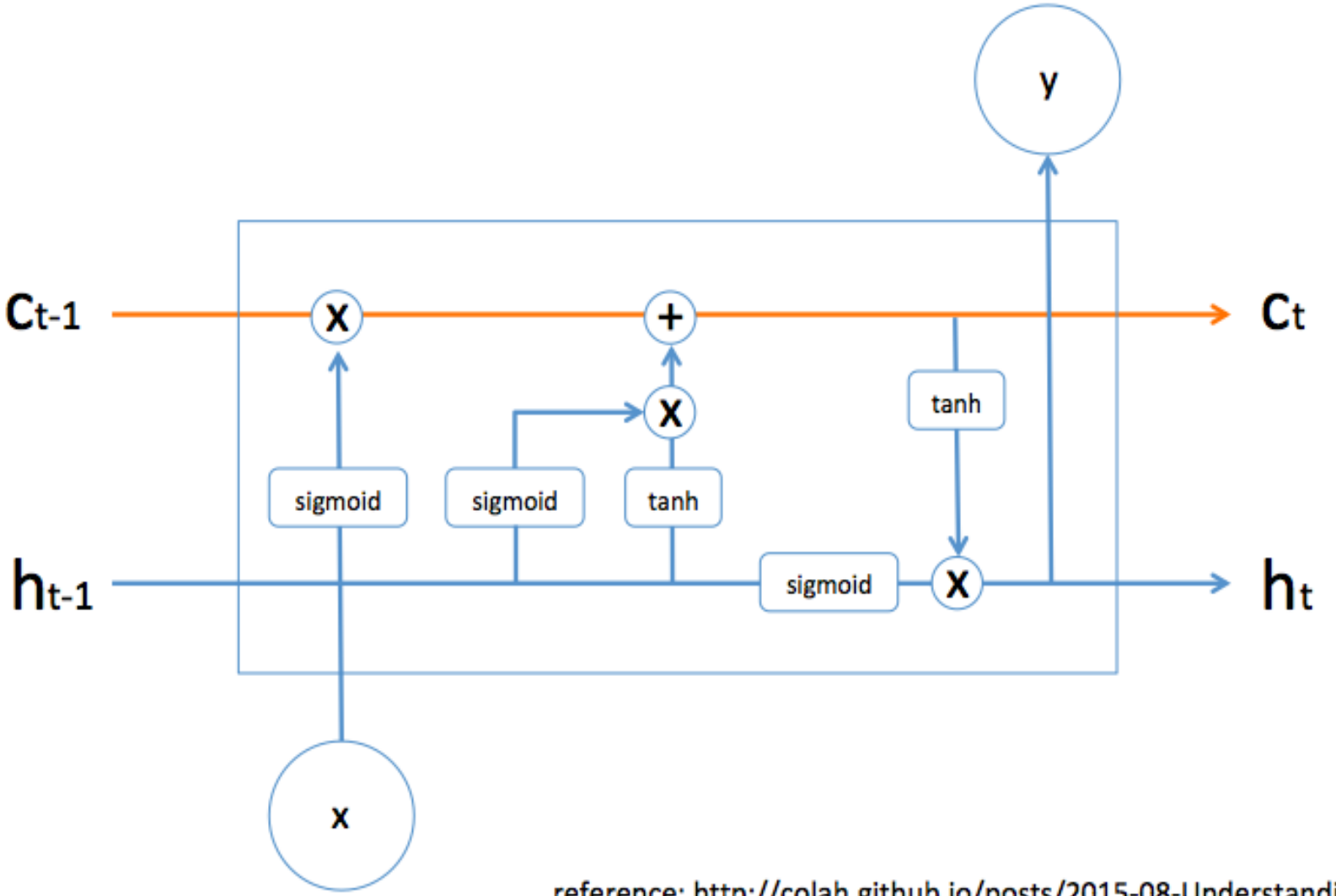
예) RNN 텐서플로우 코드 출력 이해하기

1. RNN 셀이 한개일 경우 그 출력값과 상태값이 동일합니다.
   · output values [[[-0.9314169 0.75578666 -0.6819246]]]
   · state value [[-0.9314169 0.75578666 -0.6819246]]

2. 입력값이 1x2행렬이고, RNN의 상태값이 1x3의 행렬일 경우, **W**는 총 5개의 행을 가지게 됩니다.
   · [[-0.62831575 0.38538355 0.79733914]
   · [-0.5203329 0.30046564 -0.8150209]
   · [0.39399797 0.16670114 0.4062907]
   · [-0.6391754 0.8460203 0.5266966]
   · [0.41124135 0.66347724 -0.0210759]]

3. 입력값이 1x2행렬이고, RNN의 상태값이 1x3의 행렬일 경우, 편향값은 총 3개가 필요합니다.
   · rnn/basic_rnn_cell/bias:0 [0. 0. 0.]

LSTM(Long Short-Term Memory)



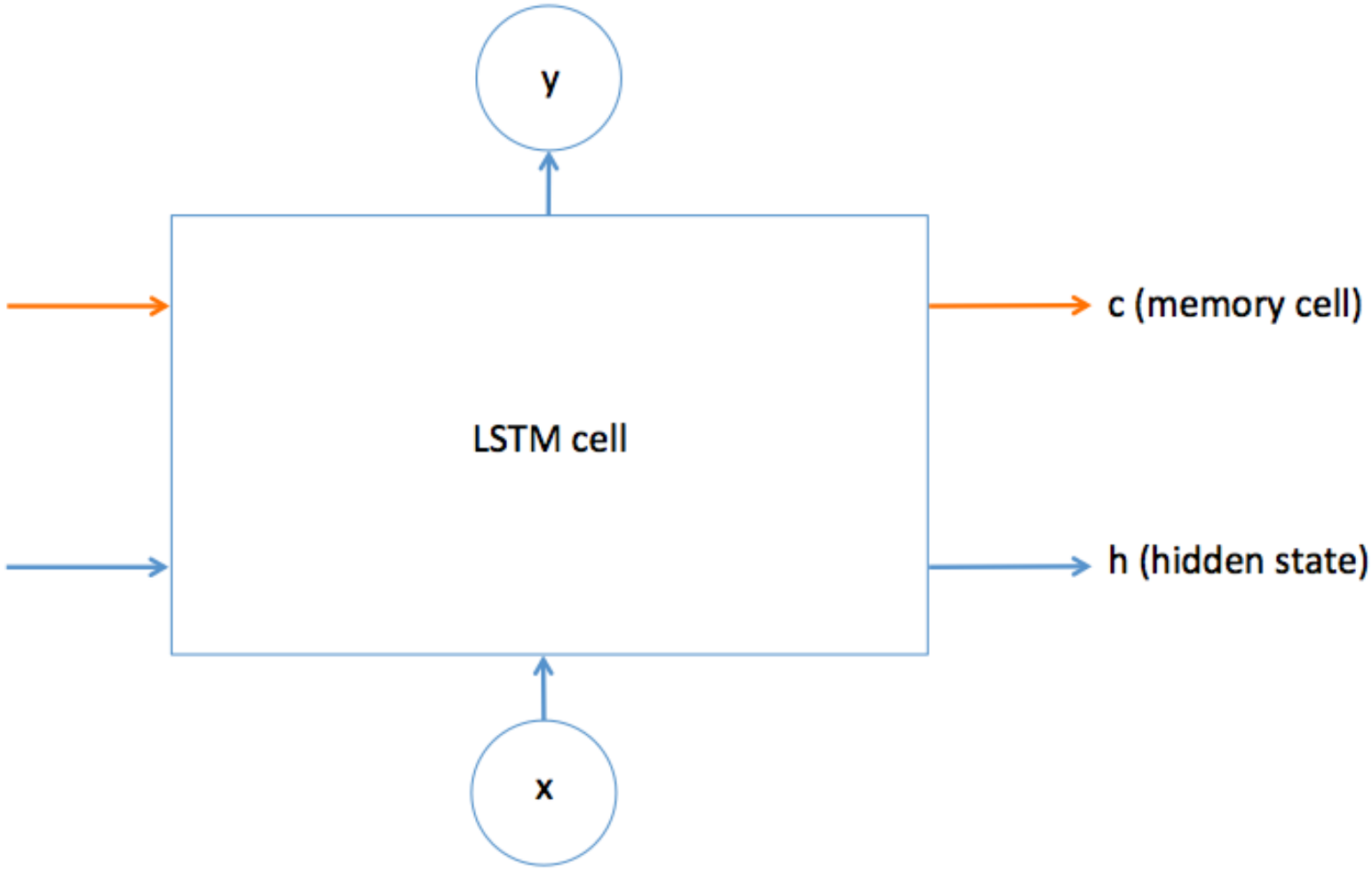
reference: http://colah.github.io/posts/2015-08-Understanding-LSTMs/

LSTM(Long Short-Term Memory)

- gradient vanishing 또는 gradient exploding과 같은 기존 RNN의 단점을 극복하고자 만들어진 조금 더 진화된 RNN 셀.
- LSTM 셀 내부를 살펴보면 기존 RNN의 단점을 극복하기 위해 이전 정보를 지우거나 기억하기 위한 로직과 현재 정보를 기억하기 위한 로직이 구현되어 있음.
- 다음 그림에서 보면 기존 RNN에서 보지 못했던 주황색 선과 조금 더 많아진 활성화 함수와 수학 기호들을 볼 수 있음.
- 주황색 선은 메모리 셀이라고 부름.
- 주황색 선상의 곱하기 기호에서, 0부터 1까지의 값인 시그모이드 출력값이 곱해지게 되어, 메모리 셀의 기존 정보를 어느 정도까지 기억할 지 결정하게 됨.
- 주황색 선상의 더하기 기호는 새로운 정보를 메모리 셀의 기존 정보에 더하는 로직.
- 그리고 **h**t 선상의 곱하기 기호에서 메모리 셀의 정보와 현재 정보가 함께 계산되어 상태값을 출력하는 것을 확인할 수 있음.

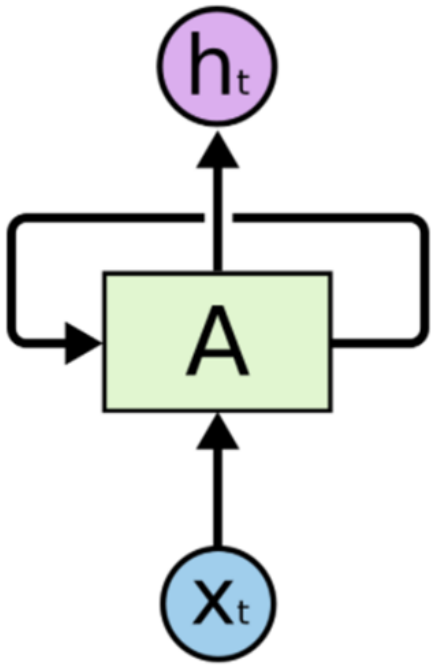
LSTM(Long Short-Term Memory)

- 텐서플로우 사용 시, 이미 LSTM은 구현되어 있기 때문에 직접 위 그림을 구현할 필요는 없음.
- 텐서플로우 LSTM을 사용 시 아래 그림만 잘 이해하셔도, 사용에 큰 무리가 없음.



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

hidden_size=2

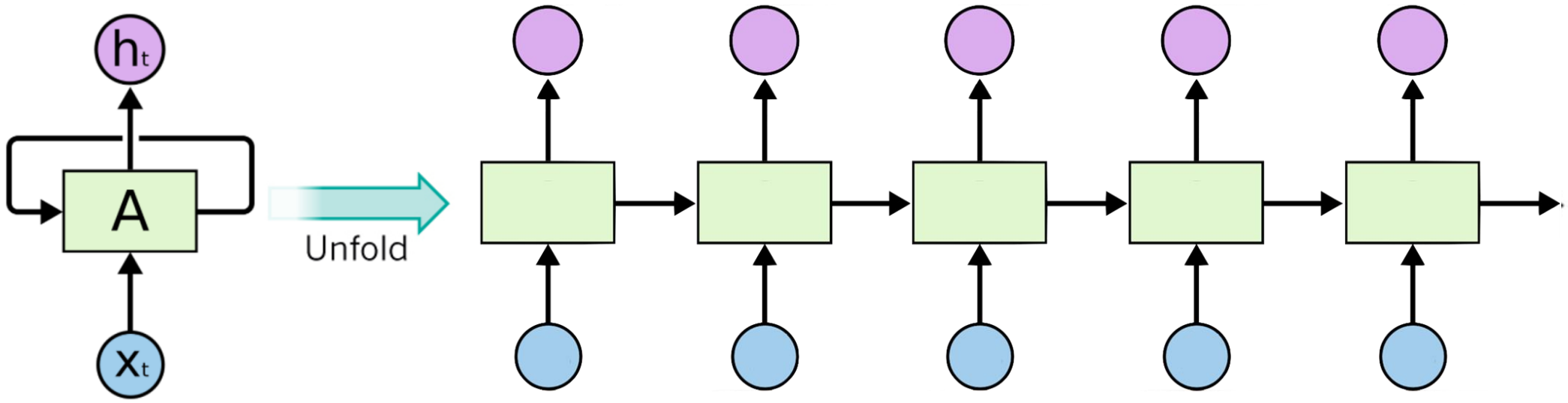



```
# One hot encoding
h = [1, 0, 0, 0]
e = [0, 1, 0, 0]
l = [0, 0, 1, 0]
o = [0, 0, 0, 1]
```

Unfolding to n sequences

```
Hidden_size=2
sequence_length=5
```

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

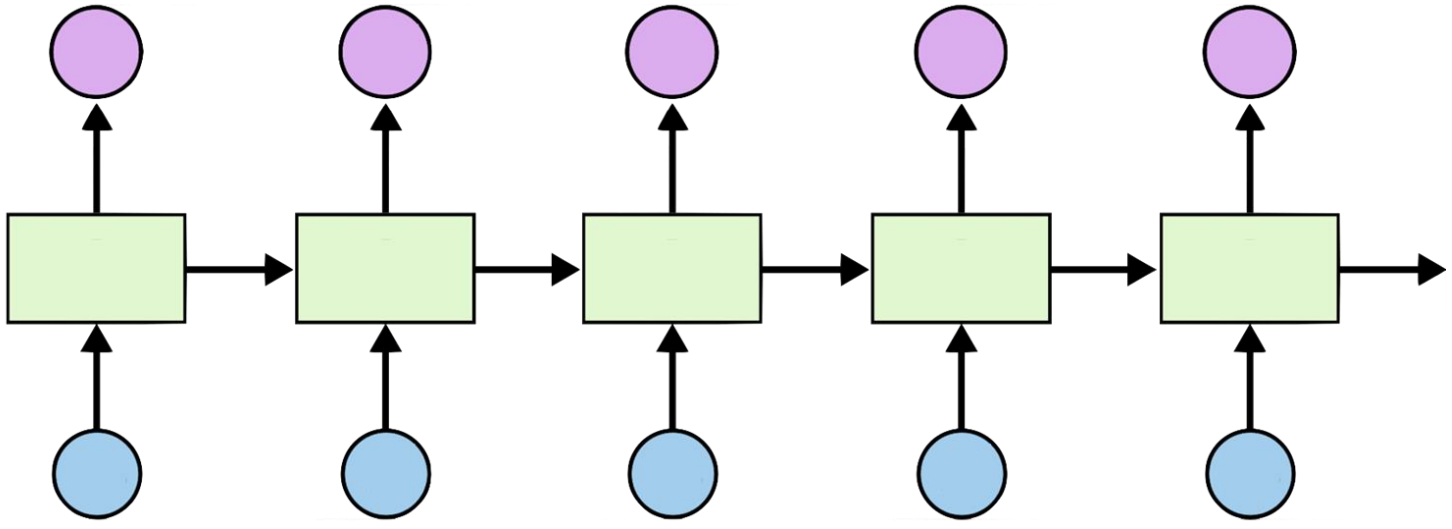



```
shape=(1,5,4): [[[1,0,0,0], [0,1,0,0], [0,0,1,0], [0,0,1,0], [0,0,0,1]]]
                      h         e         l         l         o
```

Batching input

```
Hidden_size=2
sequence_length=5
batch_size=3
```

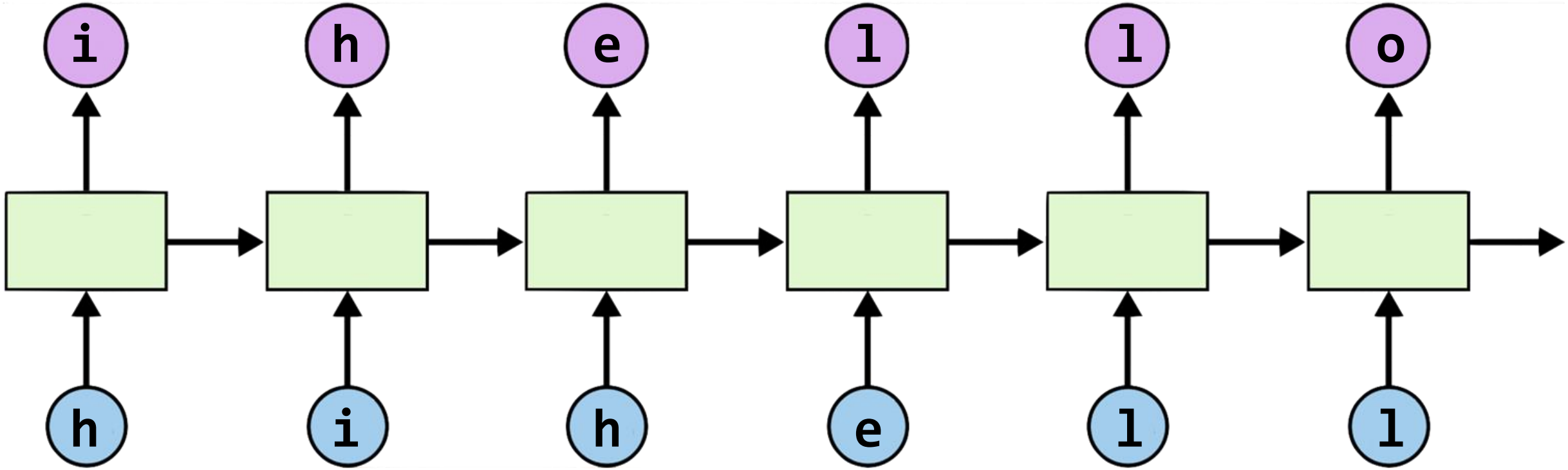
```
shape=(3,5,2): [[[x,x], [x,x], [x,x], [x,x], [x,x]],
                [[x,x], [x,x], [x,x], [x,x], [x,x]],
                [[x,x], [x,x], [x,x], [x,x], [x,x]]]
```



```
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
```

Hi Hello RNN

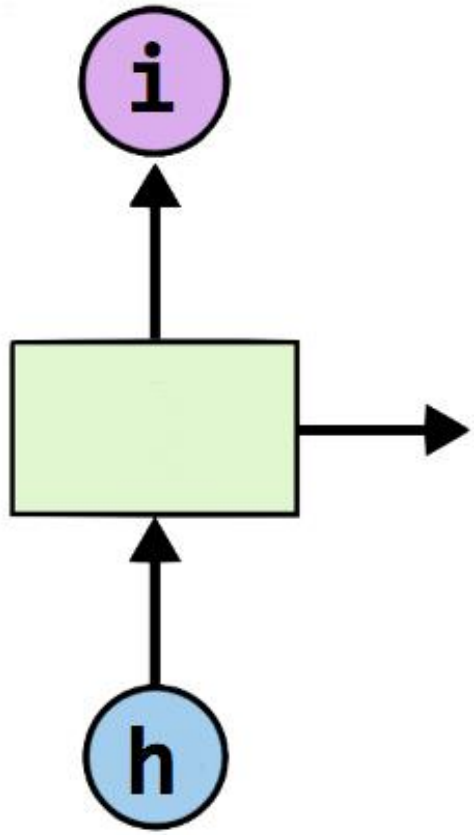
Teach RNN ‘hihello’



One-hot encoding

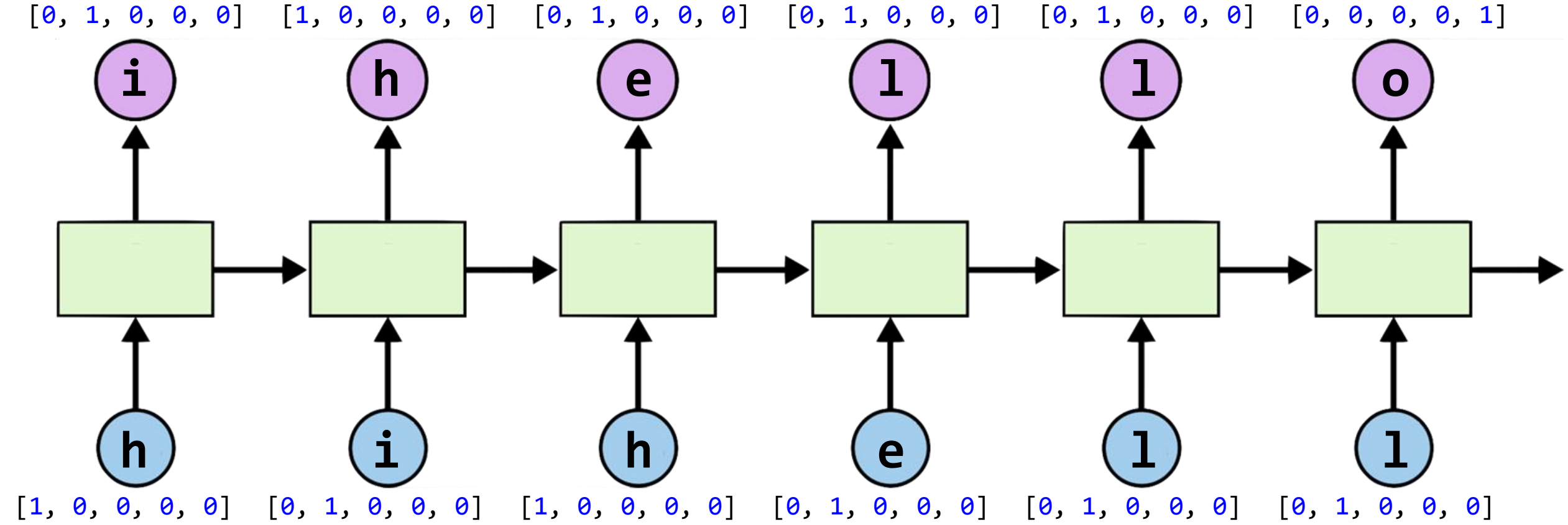
- text: ‘hihello’
- unique chars (vocabulary, voc):
  h, i, e, l, o
- voc index:
  h:0, i:1, e:2, l:3, o:4

```
[1, 0, 0, 0, 0],   # h 0
[0, 1, 0, 0, 0],   # i 1
[0, 0, 1, 0, 0],   # e 2
[0, 0, 0, 1, 0],   # l 3
[0, 0, 0, 0, 1],   # o 4
```



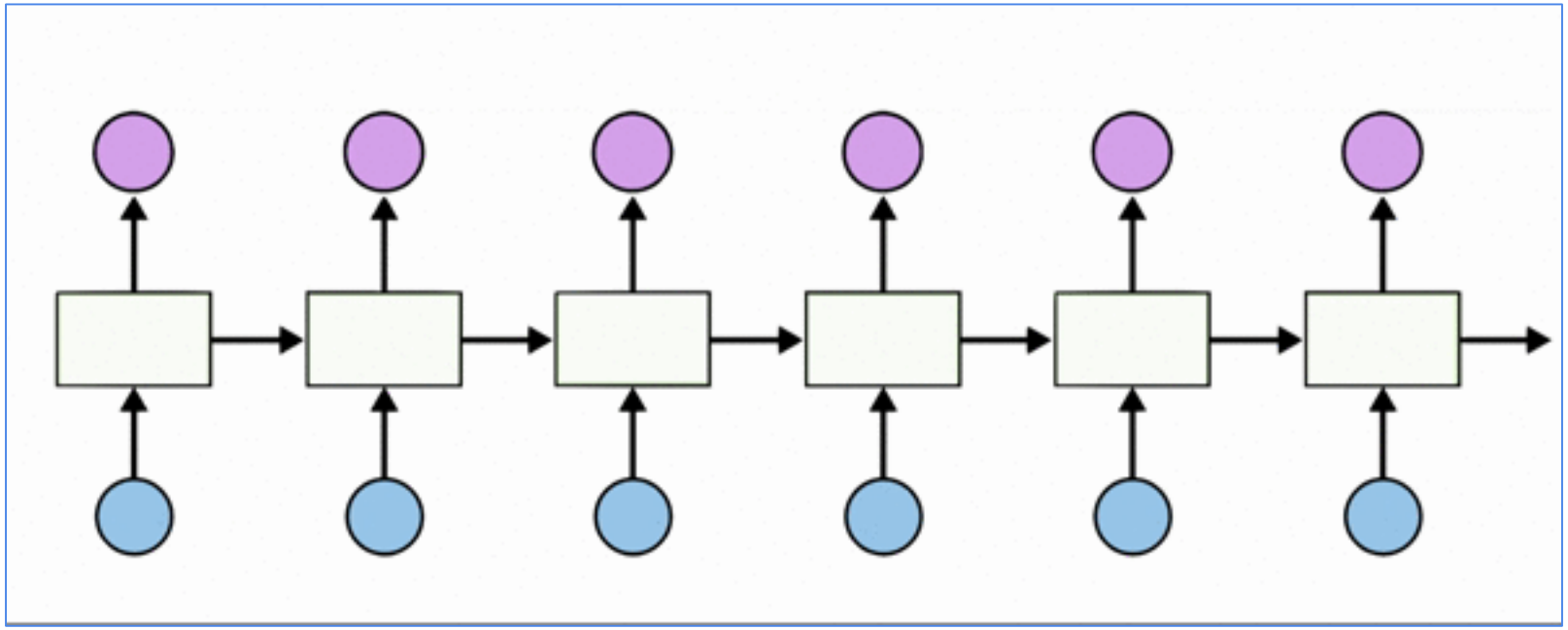
Teach RNN ‘hihello’

```
[1, 0, 0, 0, 0],   # h 0
[0, 1, 0, 0, 0],   # i 1
[0, 0, 1, 0, 0],   # e 2
[0, 0, 0, 1, 0],   # l 3
[0, 0, 0, 0, 1],   # o 4
```



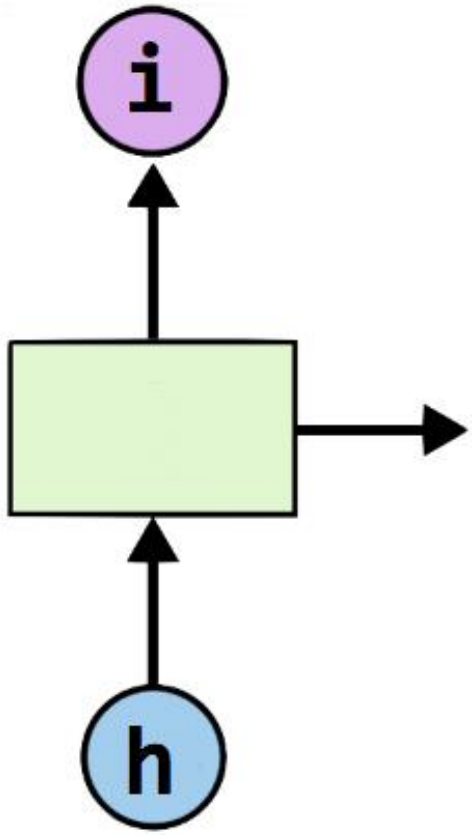
Teach RNN ‘hihello’

```
[1, 0, 0, 0, 0],   # h 0
[0, 1, 0, 0, 0],   # i 1
[0, 0, 1, 0, 0],   # e 2
[0, 0, 0, 1, 0],   # l 3
[0, 0, 0, 0, 1],   # o 4
```



RNN parameters

```
hidden_size = 5         # output from the LSTM
input_dim = 5           # one-hot size
batch_size = 1          # one sentence
sequence_length = 6     # |ihello| == 6
```



Data creation

```
idx2char = ['h', 'i', 'e', 'l', 'o'] # h=0, i=1, e=2, l=3, o=4
#x_data = [[0, 1, 0, 2, 3, 3]]         # hihell
x_one_hot = [[[1, 0, 0, 0, 0],        # h 0
              [0, 1, 0, 0, 0],        # i 1
              [1, 0, 0, 0, 0],        # h 0
              [0, 0, 1, 0, 0],        # e 2
              [0, 0, 0, 1, 0],        # l 3
              [0, 0, 0, 1, 0]]]       # l 3

y_data = [[1, 0, 2, 3, 3, 4]]         # ihello
```

텐서플로우로 지문을 읽고 주제를 분류하는 모델 구현하기

➢