Neural Network with Memory

Thanks to the slides from Prof. Hung-yi Lee in NCTU.

Memory is important

Input:
2 dimensions

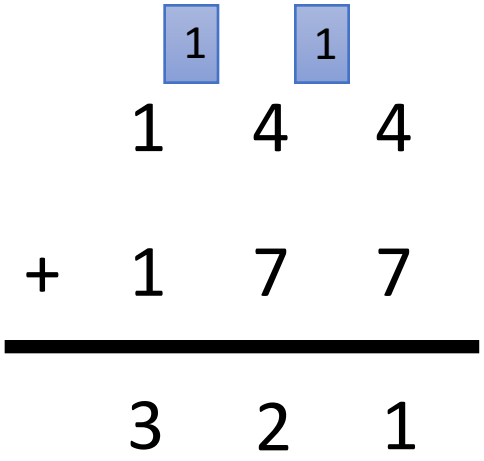
x^1 x^2 x^3

4	4	1
7	7	1

Output:
1 dimension

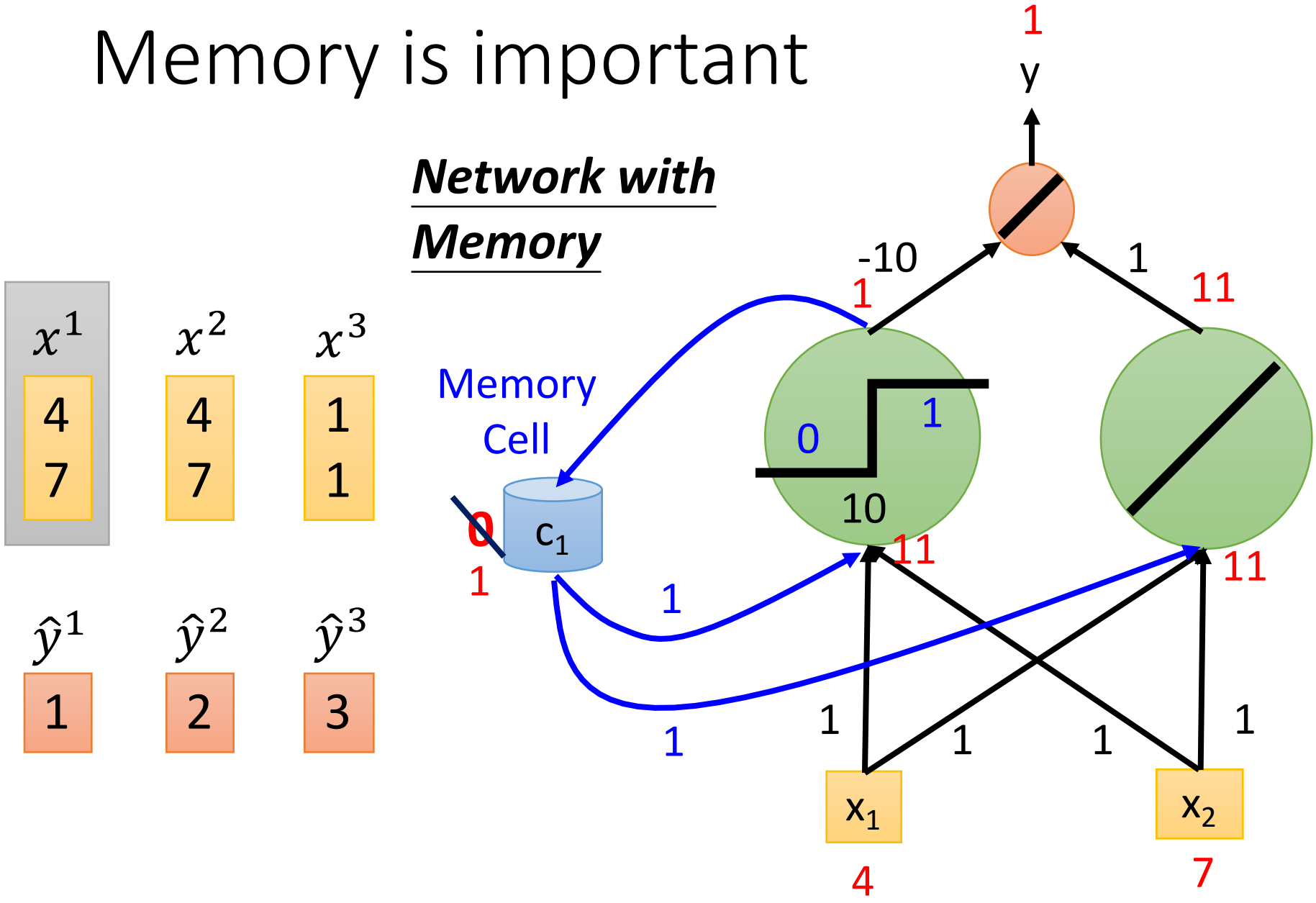
$\hat{y}^1$ $\hat{y}^2$ $\hat{y}^3$

1	2	3

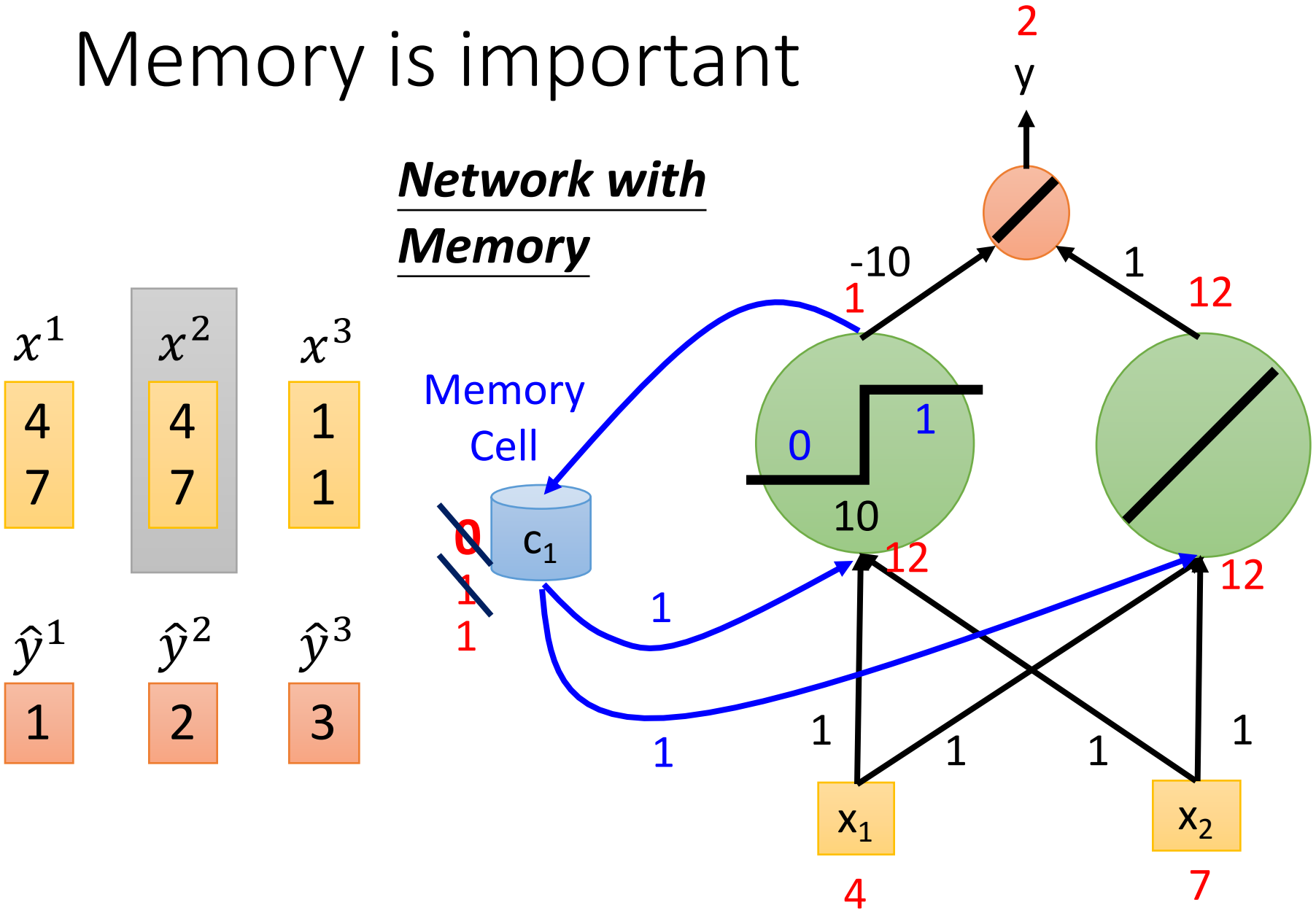


Network needs memory
to achieve this

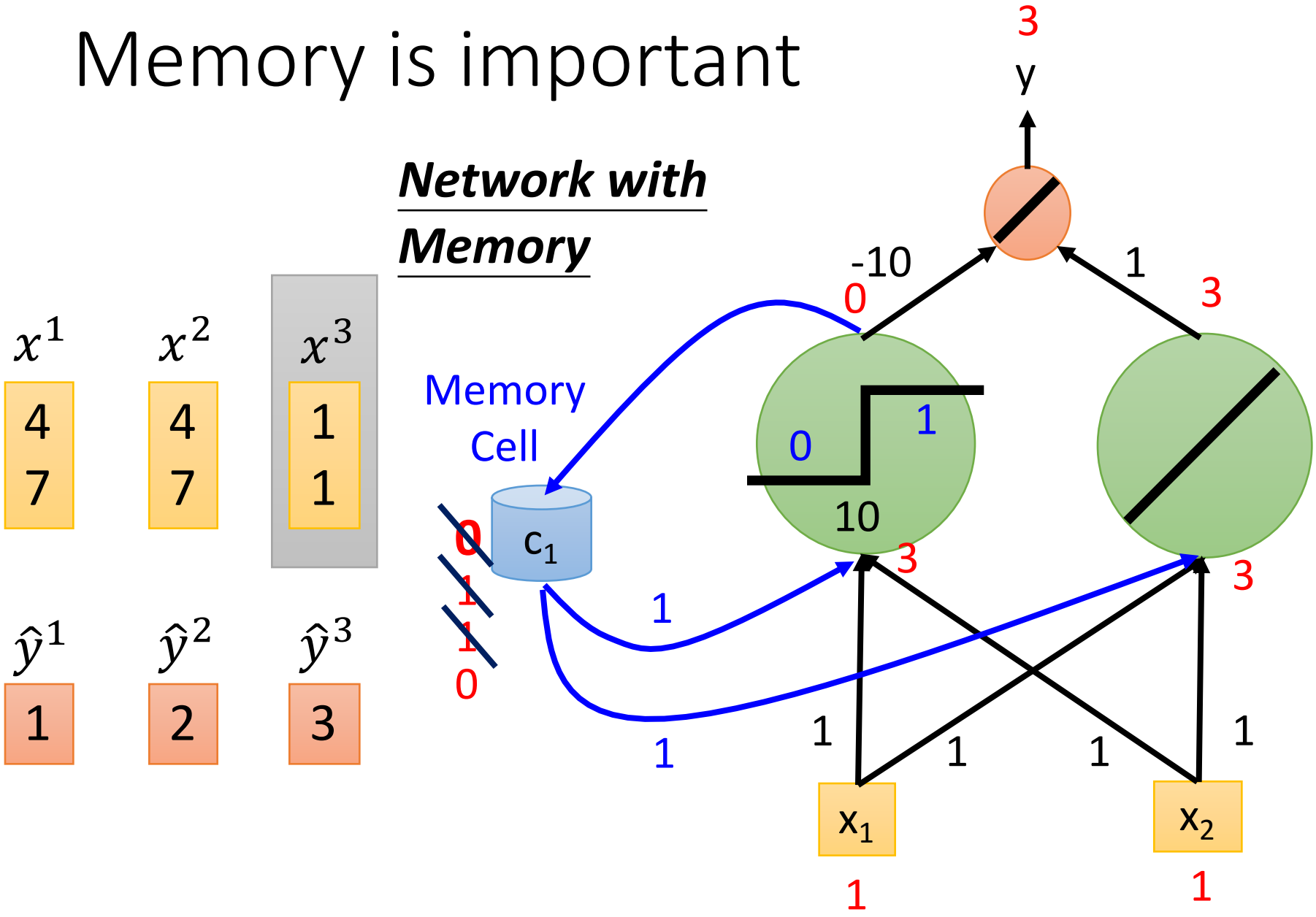
Memory is important



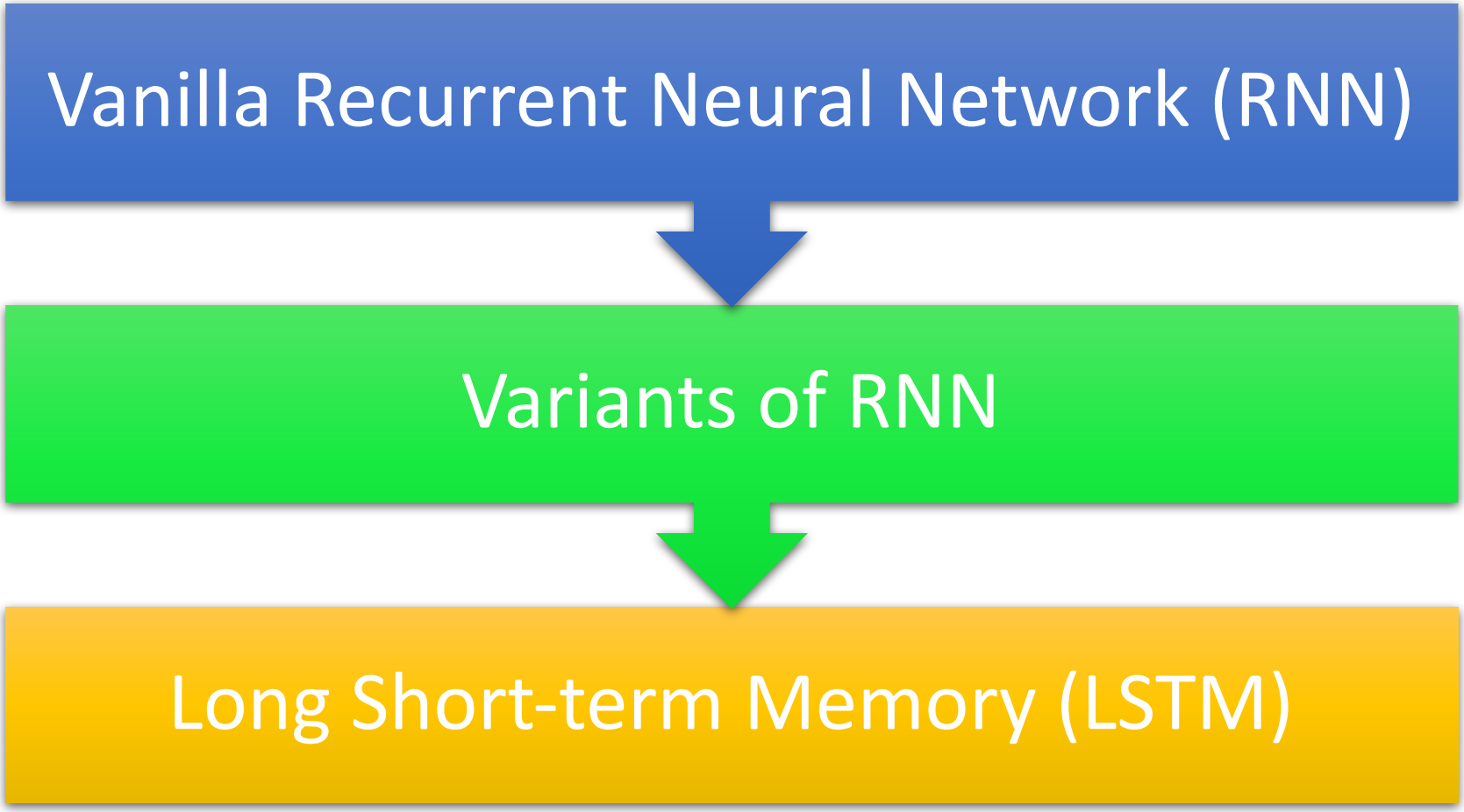
Memory is important



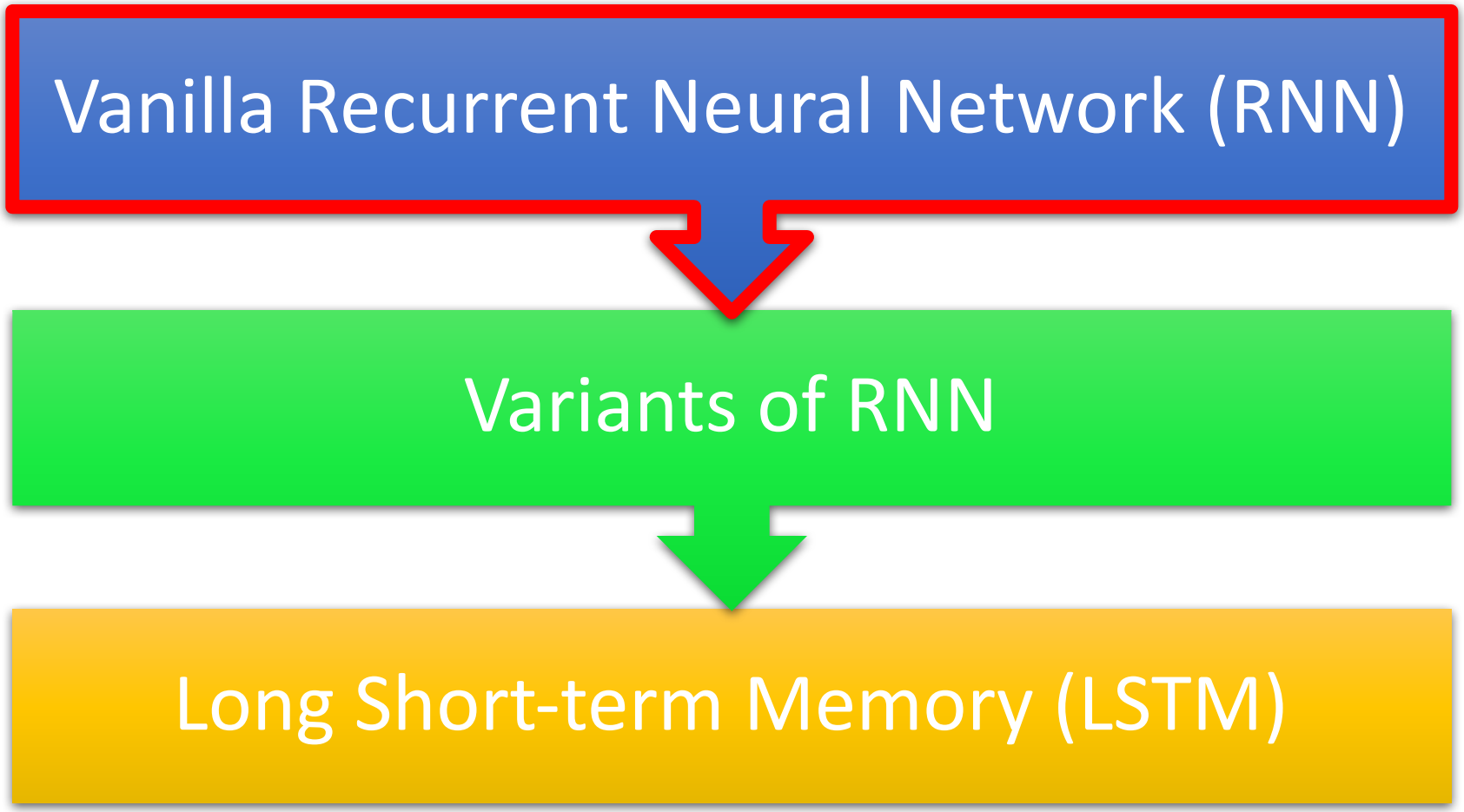
Memory is important



Outline

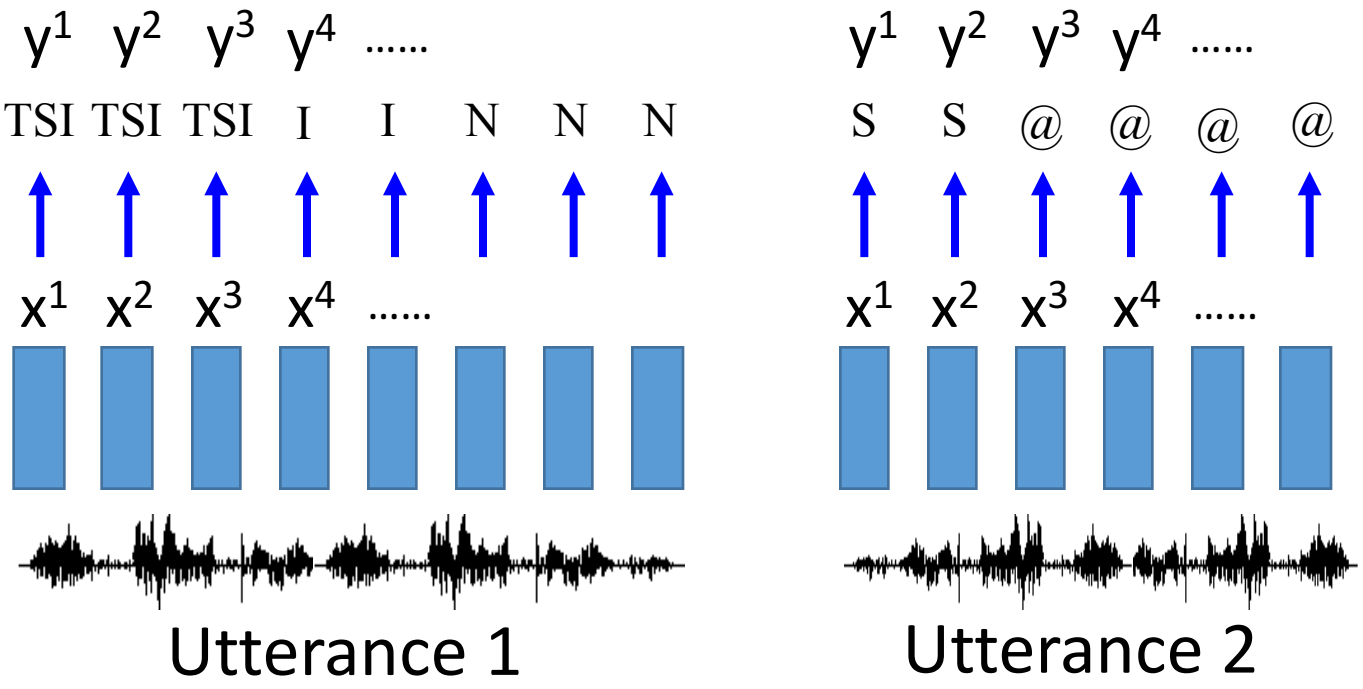

Vanilla Recurrent Neural Network (RNN)
Variants of RNN
Long Short-term Memory (LSTM)

Outline


Vanilla Recurrent Neural Network (RNN)
Variants of RNN
Long Short-term Memory (LSTM)

Application

- (Simplified) Speech Recognition



We use DNN. All the frames are considered independently.

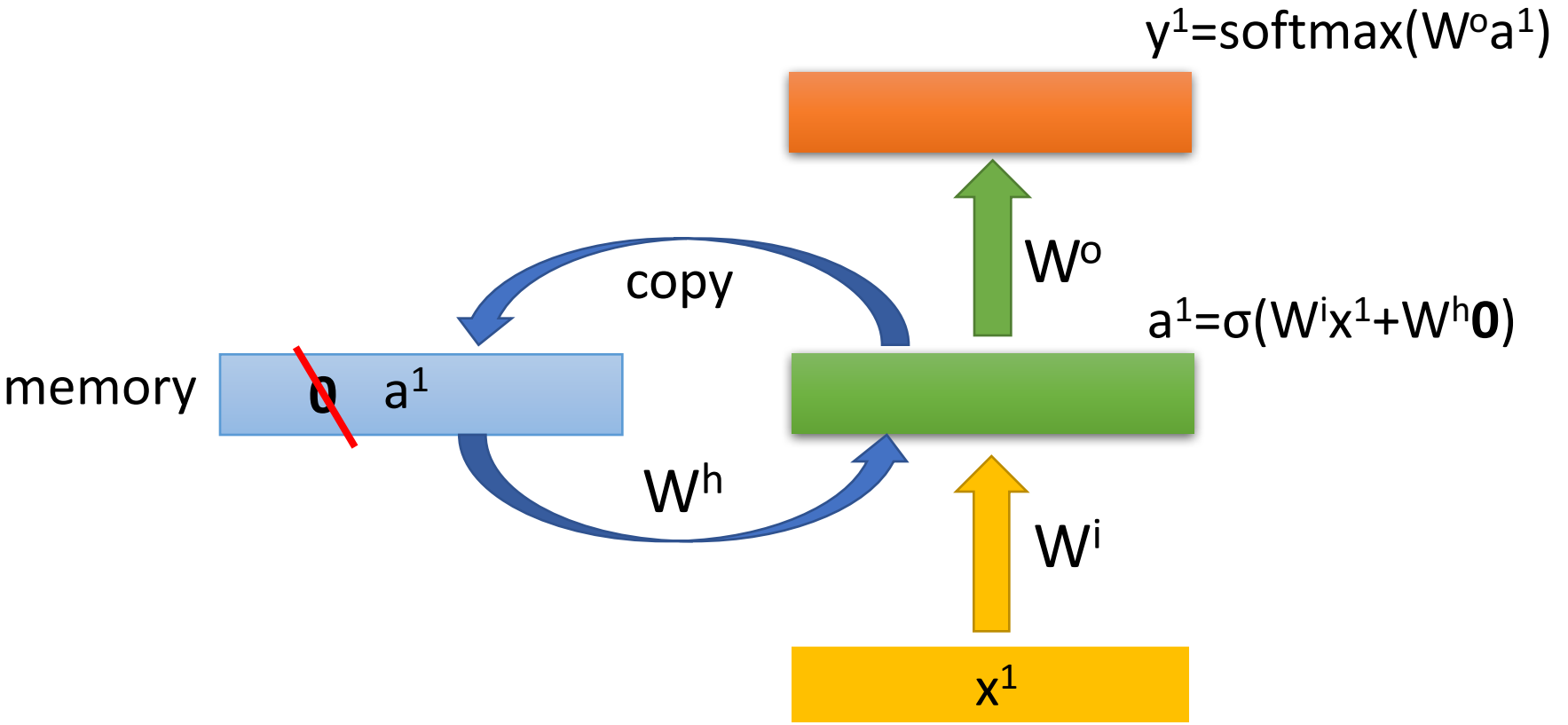
RNN

The order cannot change.

RNN input: x^1 x^2 x^3 ..... x^N

Input of RNN is one utterance

y^1



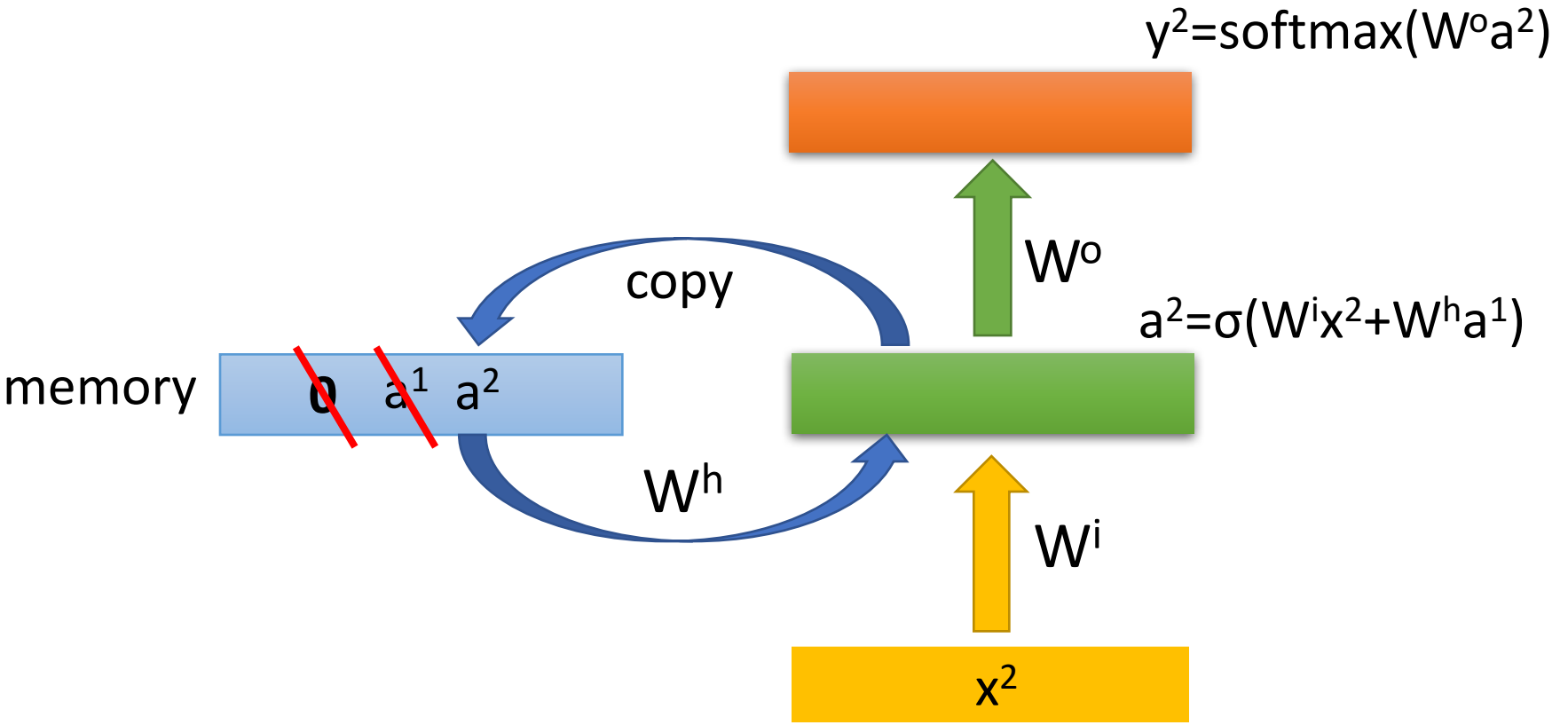
RNN

The order cannot change.

RNN input: x^1 x^2 x^3 ..... x^N

Input of RNN is one utterance

y^1 y^2



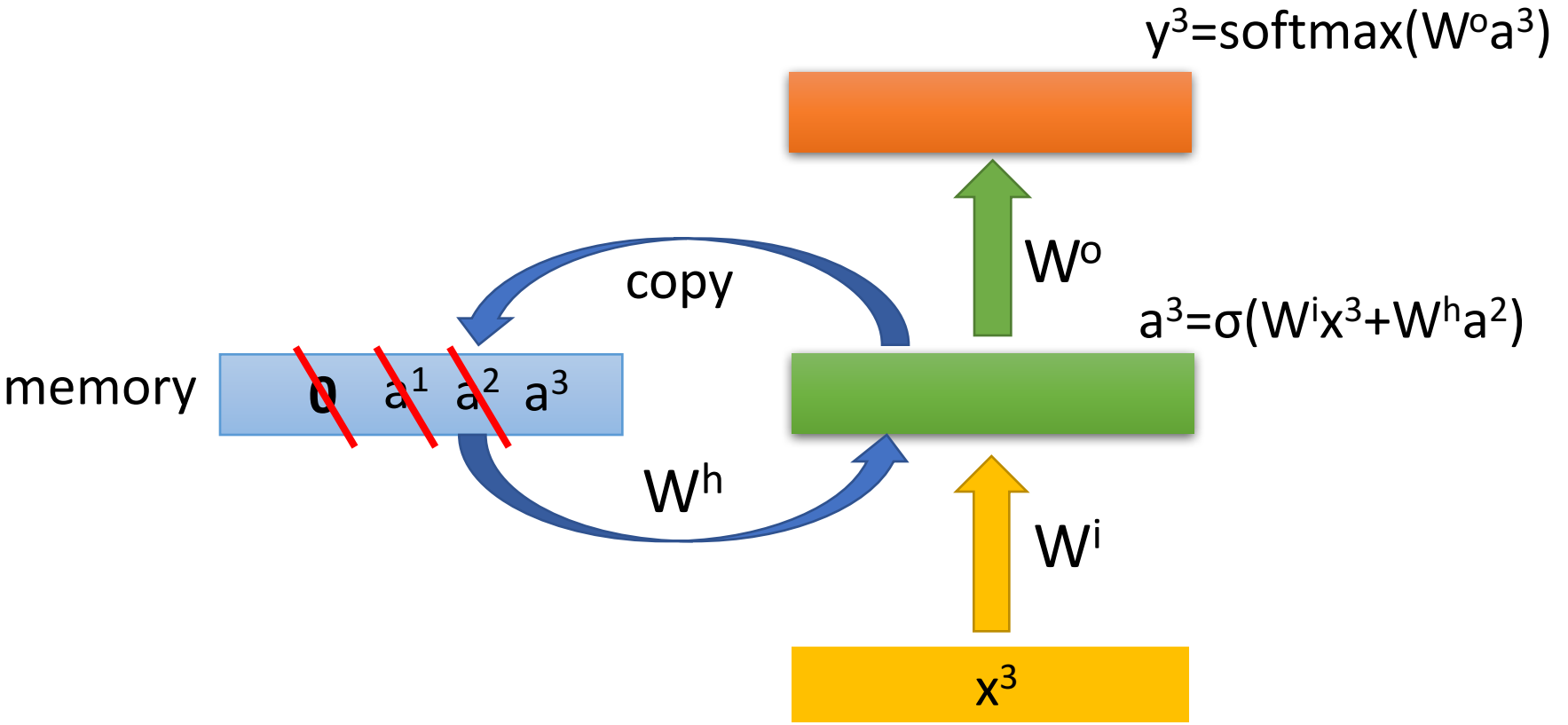
RNN

The order cannot change.

RNN input: x^1 x^2 x^3 ..... x^N

y^1 y^2 y^3

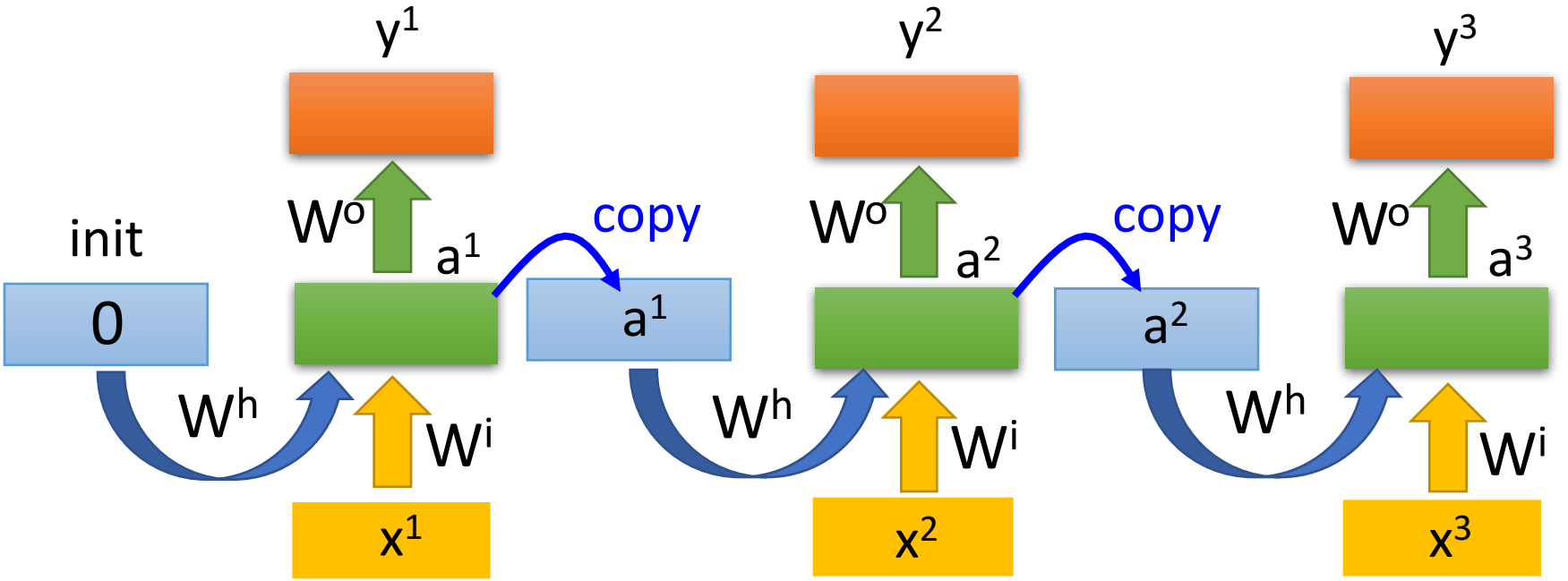
Input of RNN is one utterance



RNN

Input data: x^1 x^2 x^3 …… x^N

Input of RNN is one utterance



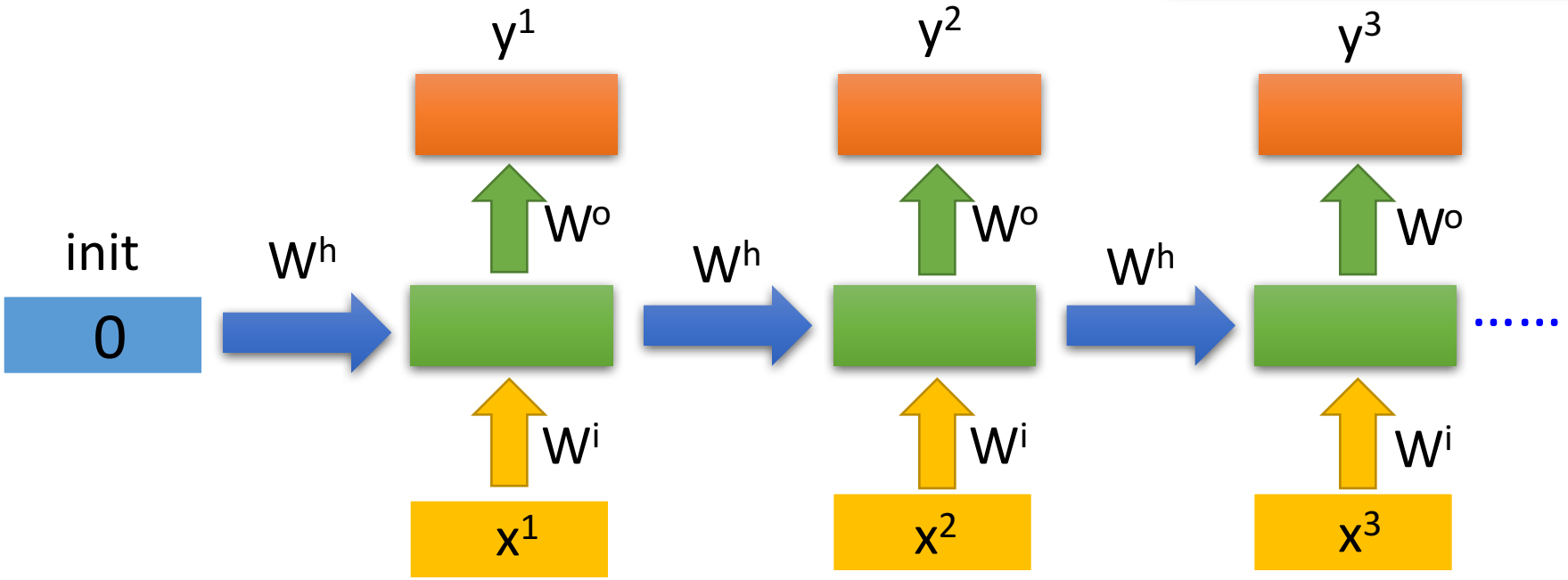
The same network is used again and again.

Output y^i depends on x^1, x^2, …… x^i

RNN

Input data: x^1 x^2 x^3 …… x^N

Input of RNN is one utterance



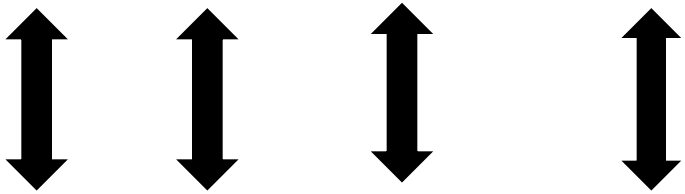
The same network is used again and again.

Output y^i depends on x^1, x^2, …… x^i

Cost

RNN input: x^1 x^2 x^3 …… x^N

RNN output: y^1 y^2 y^3 …… y^N

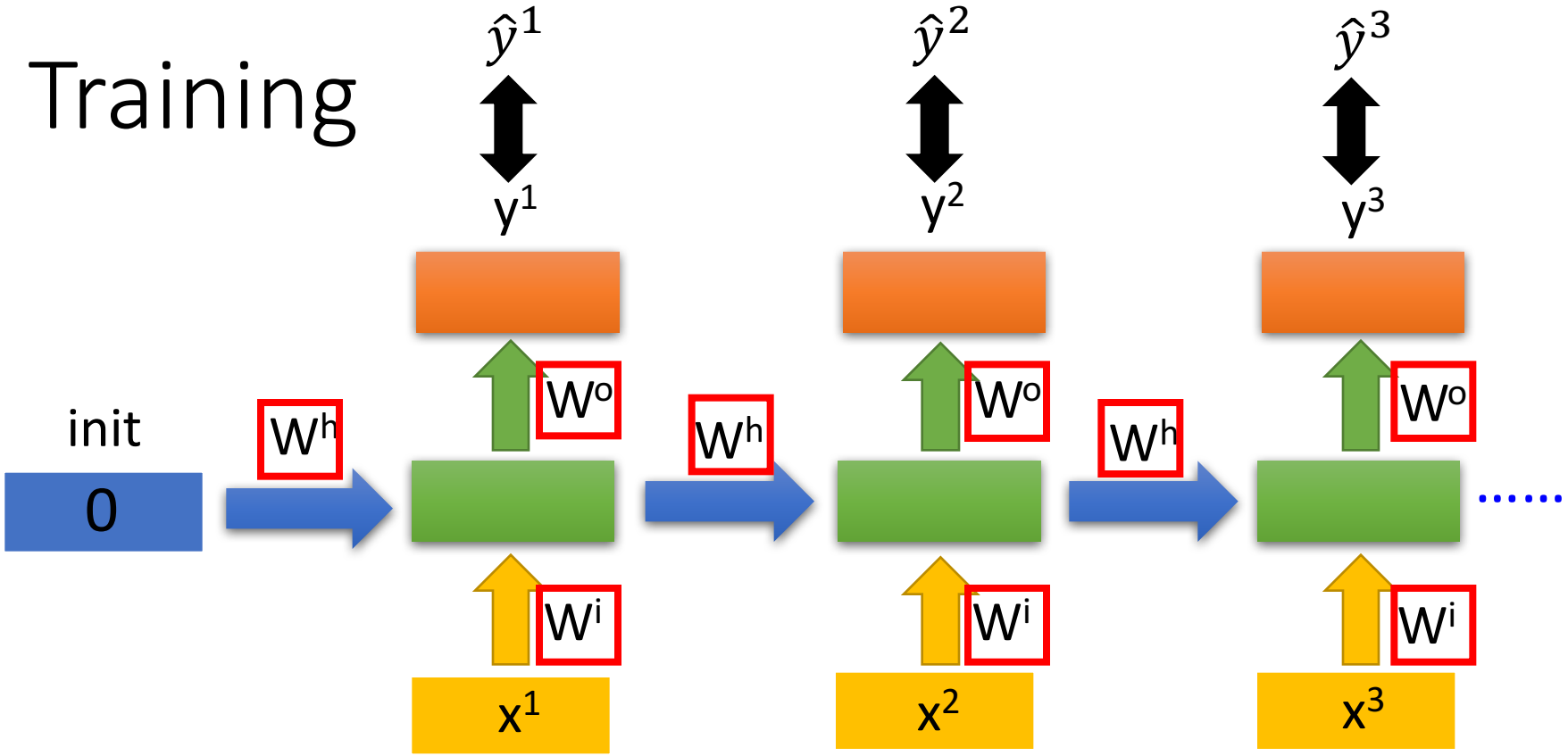


RNN output: $\hat{y}^1$ $\hat{y}^2$ $\hat{y}^3$ …… $\hat{y}^N$

$$C = \frac{1}{2}\sum_{n=1}^{N} \|y^n - \hat{y}^n\|^2$$

$$C = \frac{1}{2}\sum_{n=1}^{N} -log y^n_{r^n}$$

Training



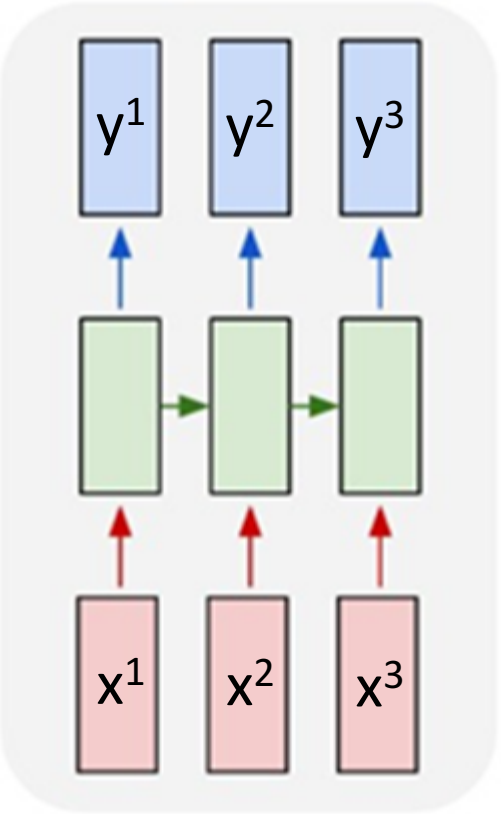
w is an element in W^h, W^i or W^o ➡ $w \leftarrow w - \eta \partial C / \partial w$

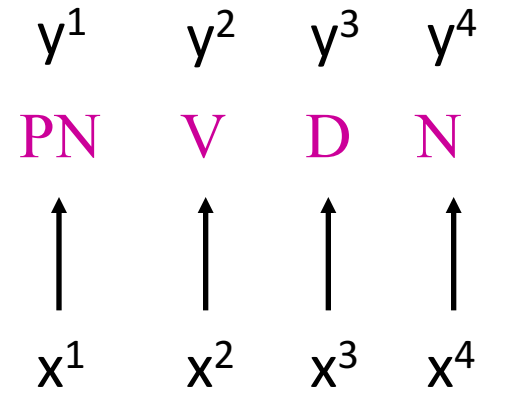
➡ Backpropagation through time (BPTT)

RNN Training is very difficult in practice.

More Applications

- Input and output are vector sequences with ***the same length***



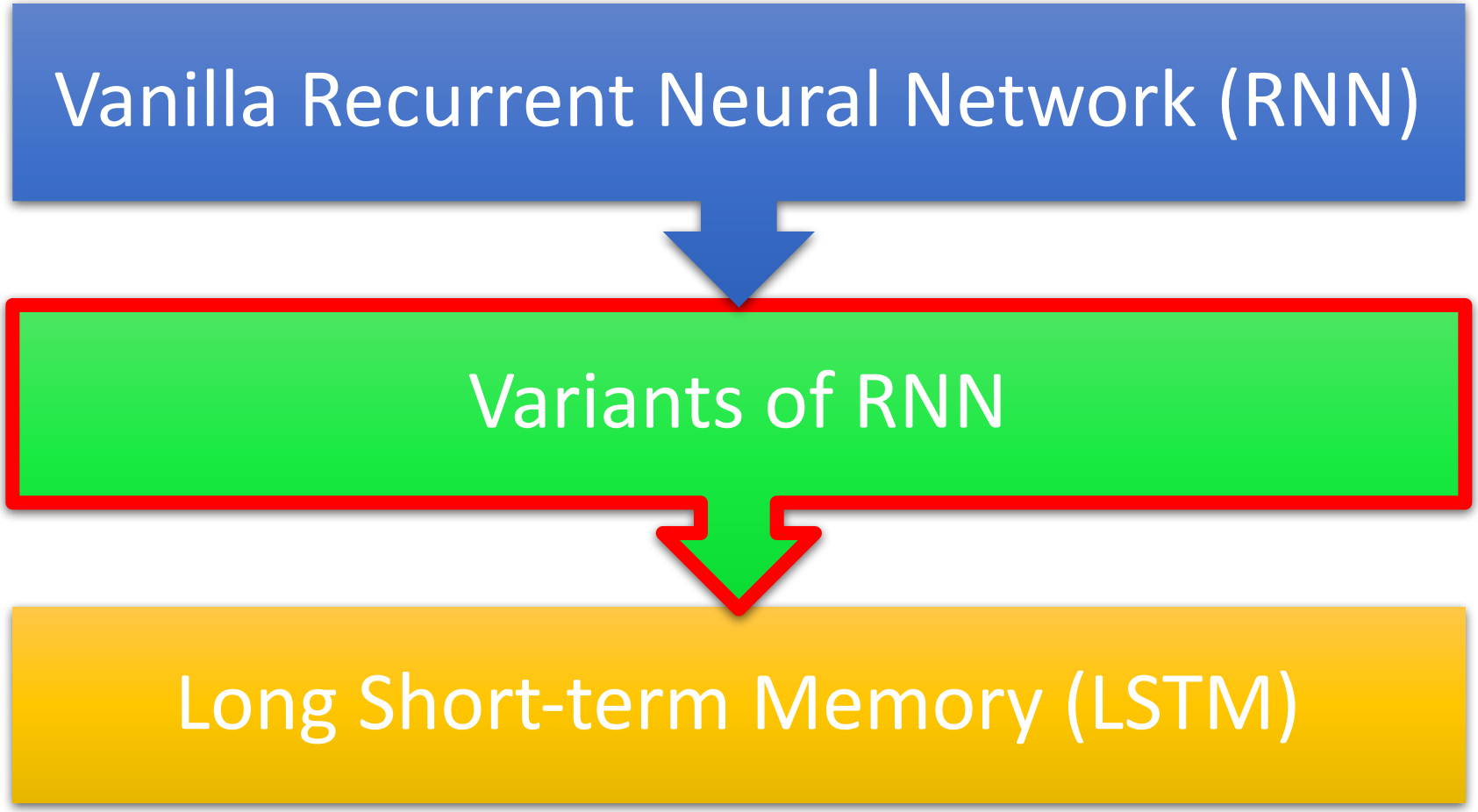


POS Tagging

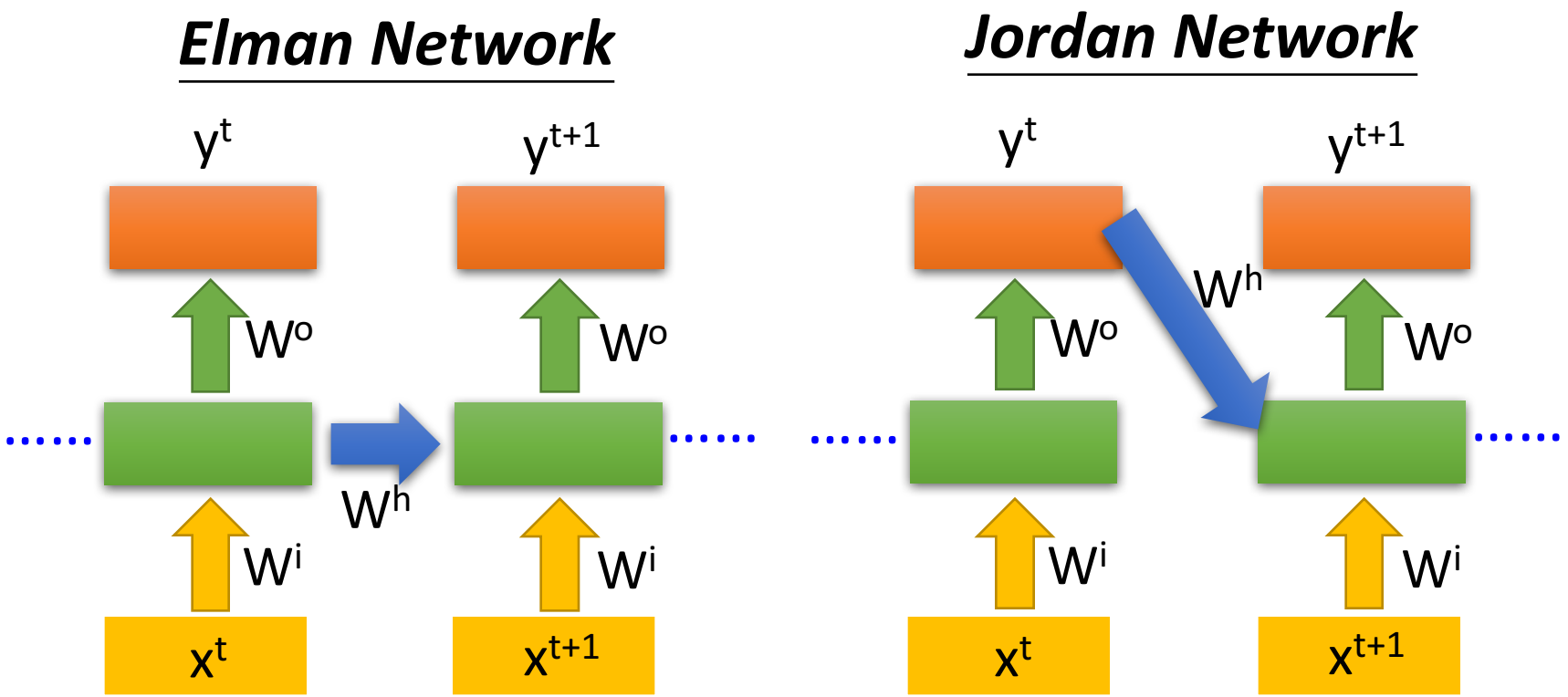
More Applications

- Name entity recognition
  - Identifying names of people, places, organizations, etc. from a sentence
  - Harry Potter is a student of Hogwarts and lived on Privet Drive.
    - people, organizations, places, not a name entity
- Information extraction
  - Extract pieces of information relevant to a specific application, e.g. flight booking
  - I would like to leave Boston on November 2nd and arrive in Taipei before 2 p.m.
    - place of departure, destination, time of departure, time of arrival, other

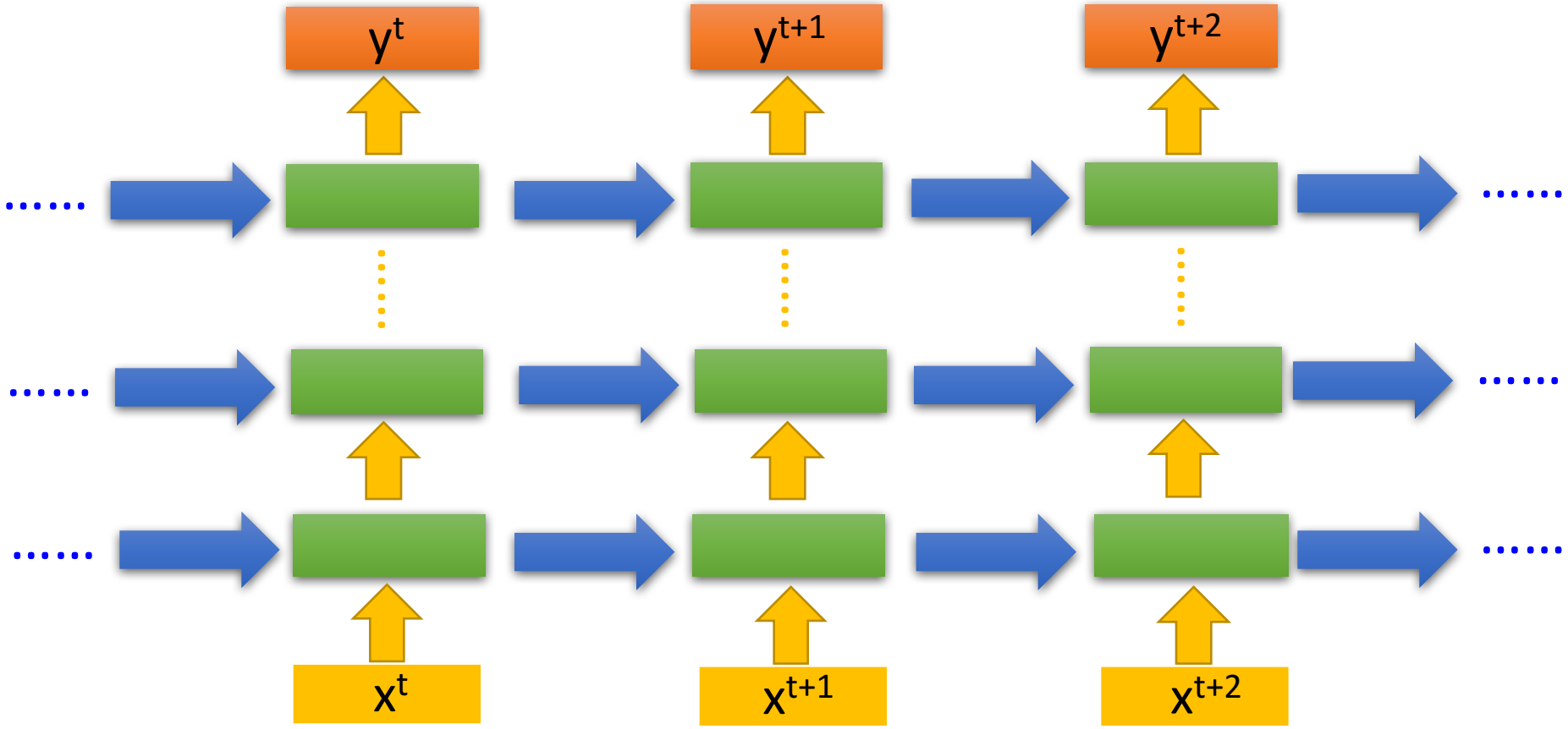
Outline


Vanilla Recurrent Neural Network (RNN)
Variants of RNN
Long Short-term Memory (LSTM)

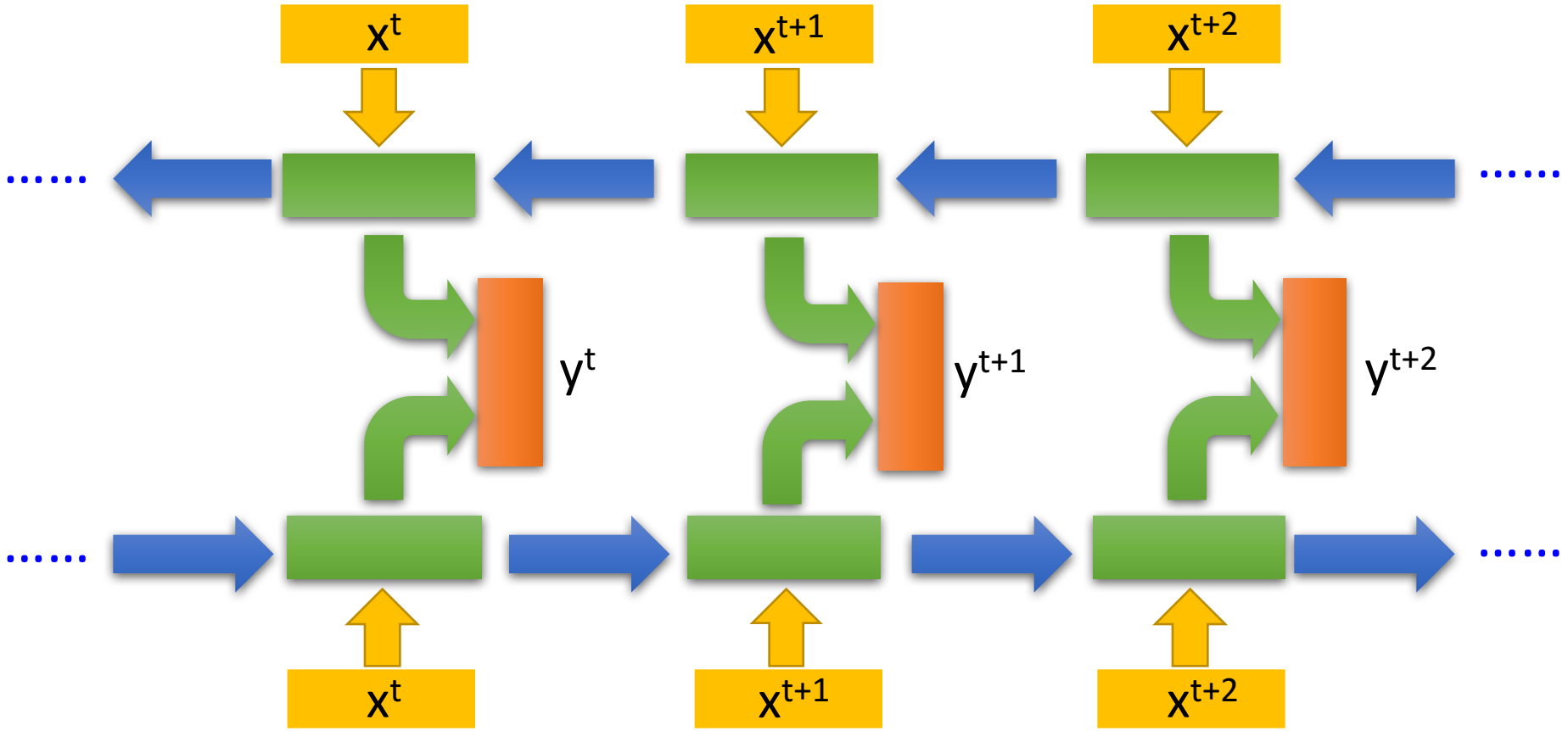
Elman Network & Jordan Network



Deep RNN



Bidirectional RNN

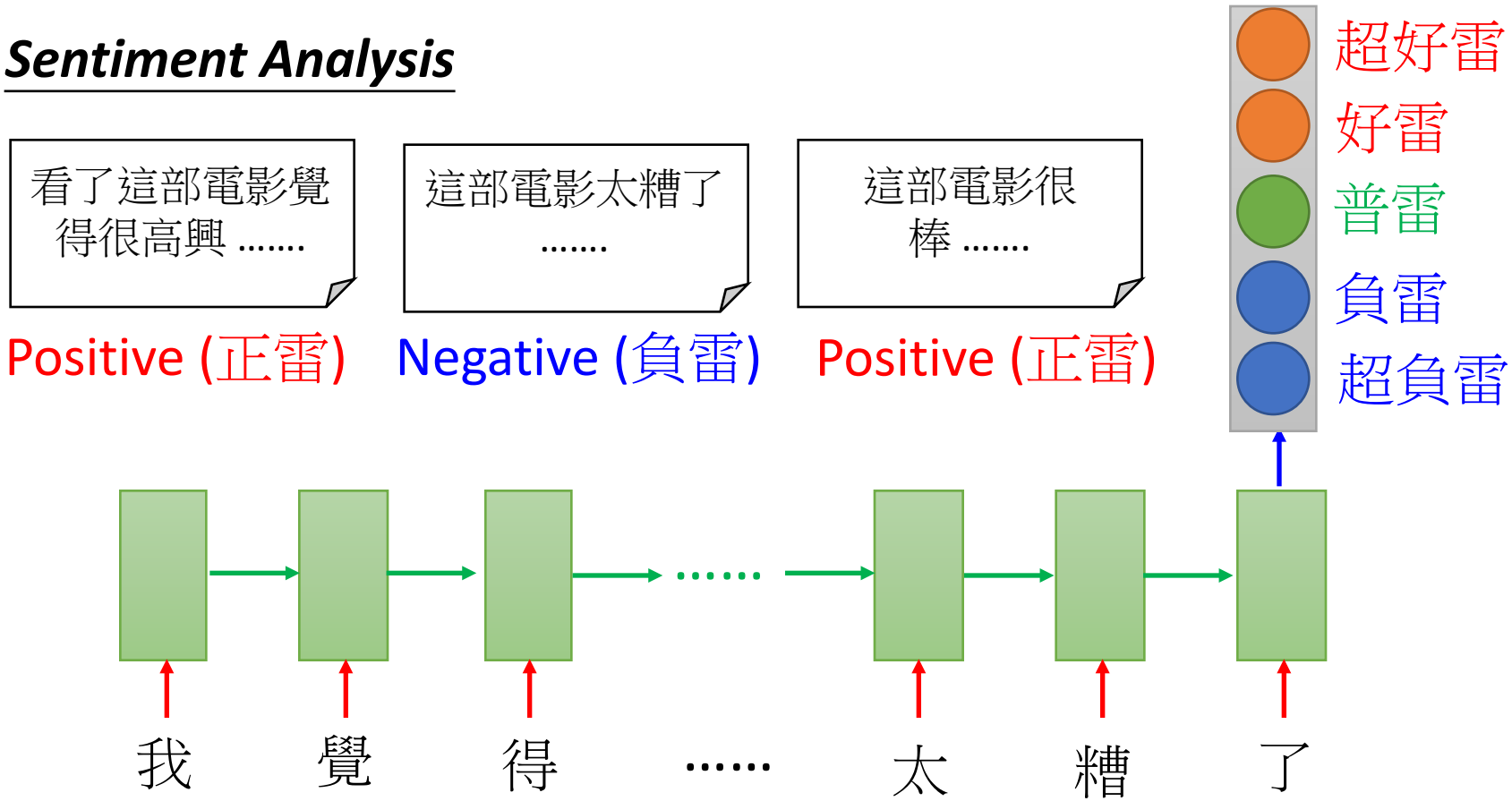


下列文句何者不是倒裝句型？

- (A)惟兄嫂是依
  (B)白雪紛紛何所似
  (C)撒鹽空中差可擬
  (D)不患人之不己知

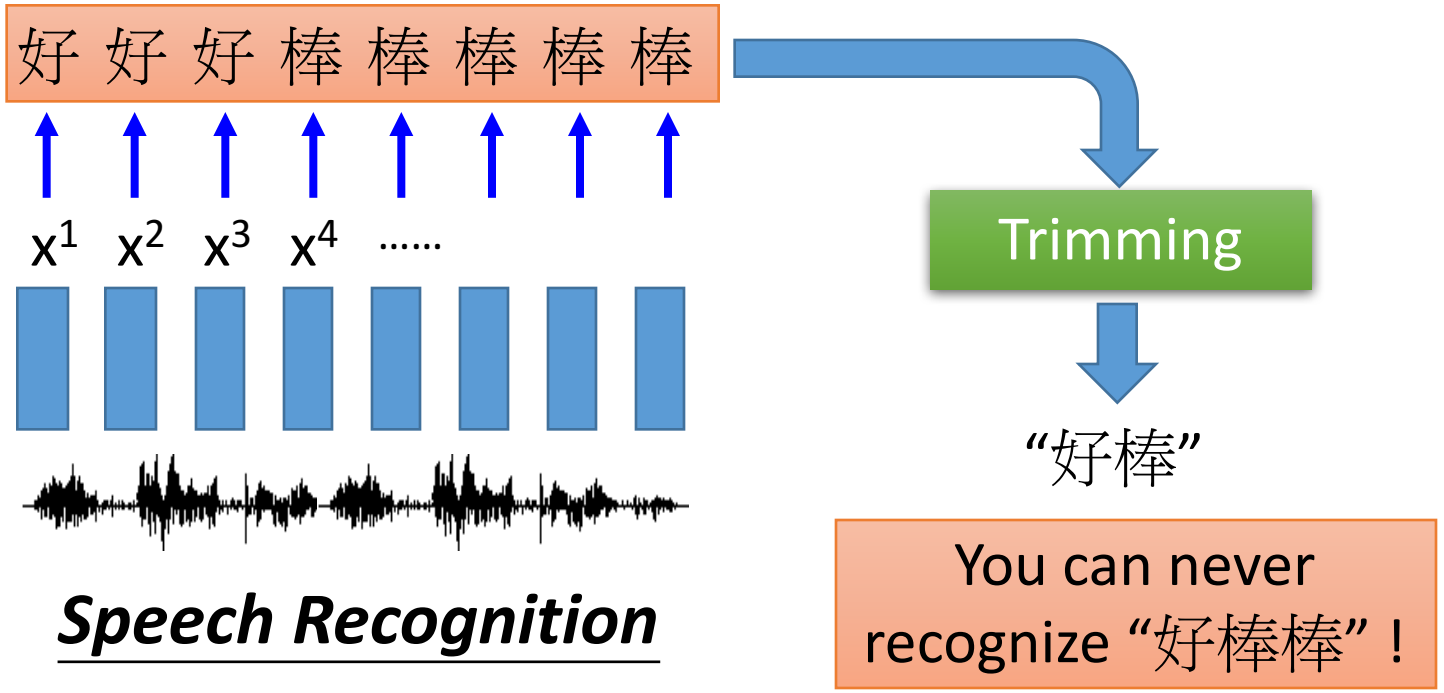
Many to one

- Input is a vector sequence, but output is only one vector



Many to Many (Output is shorter)

- Both input and output are vector sequences, ***but the output is shorter.***



Many to Many (Output is shorter)

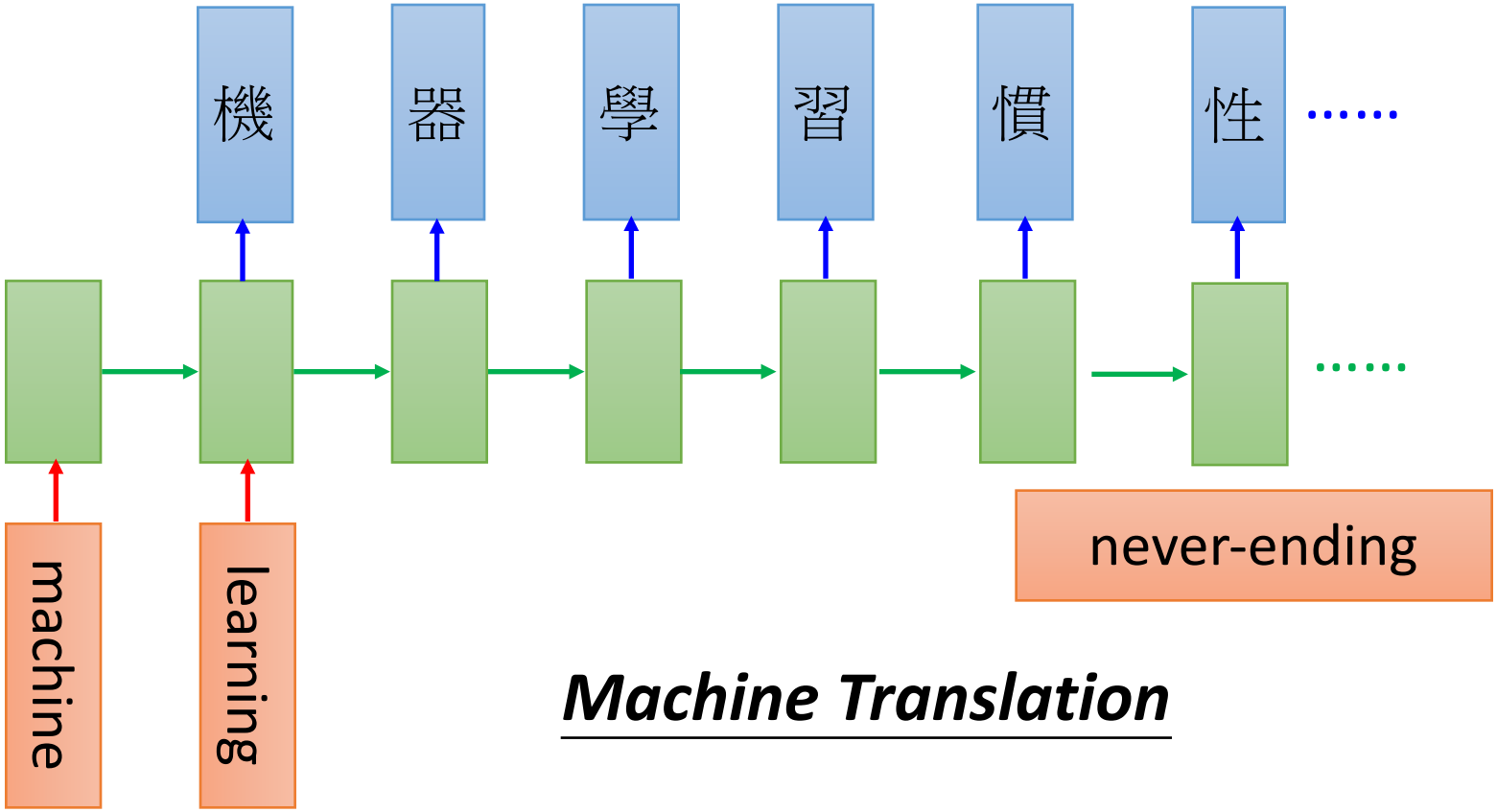
- Both input and output are vector sequences, ***but the output is shorter.***
- Connectionist Temporal Classification (CTC)
  - Add an extra symbol "ϕ" (同上)

好 ϕ ϕ 棒 ϕ ϕ ϕ ϕ → "好棒"

好 ϕ ϕ 棒 ϕ 棒 ϕ ϕ → "好棒棒"

Many to Many (No Limitation)

- Both input and output are vector sequences ***with different lengths***. → ***Sequence to sequence learning***



Machine Translation

Many to Many (No Limitation)

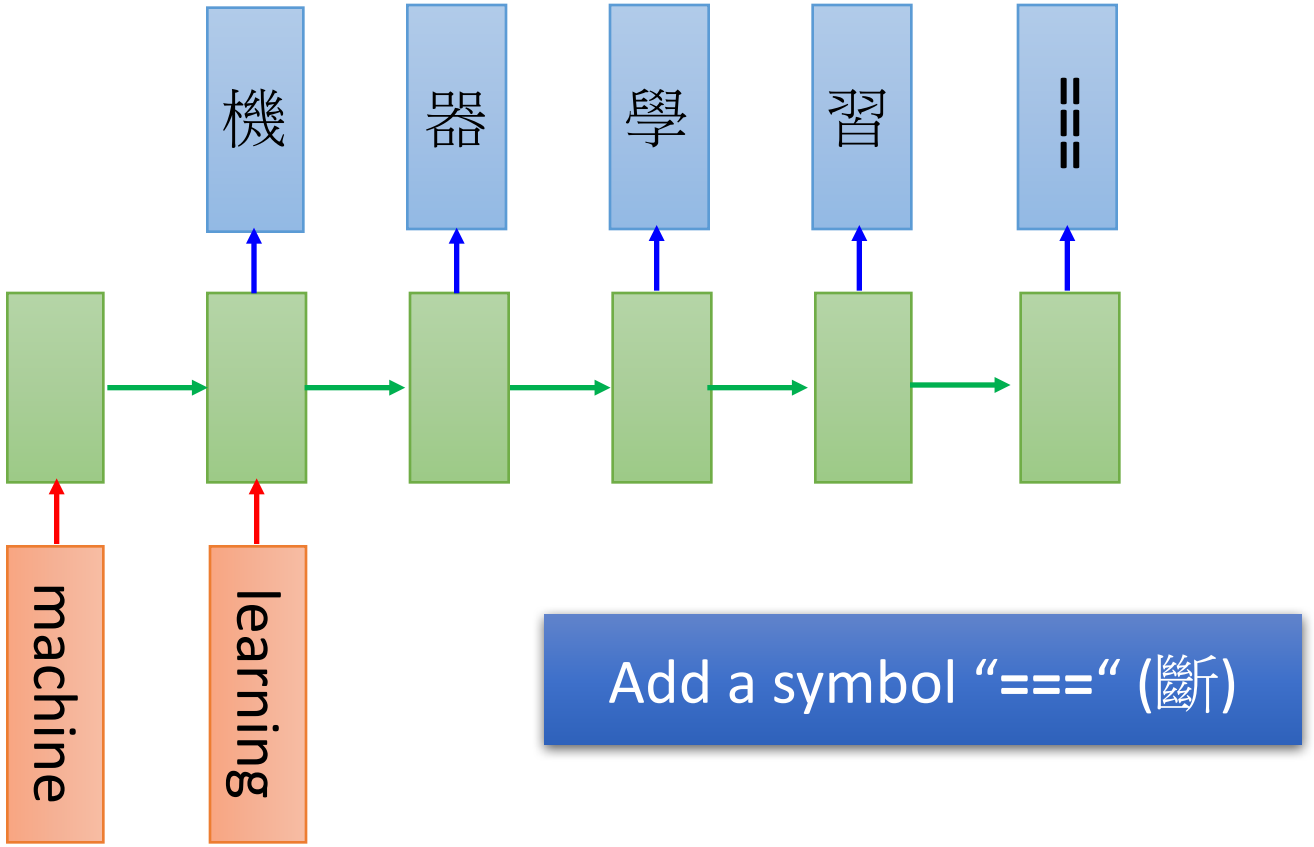
- 推文接龍
  - Ref: http://pttpedia.pixnet.net/blog/post/168133002-%E6%8E%A5%E9%BE%8D%E6%8E%A8%E6%96%87

推xxx: ptt萬歲
推dd: 歲平安
噓dddf: 全
推zzzzzzzzzz: 家就是你家
⋮

推tlkagk: ==========斷==========

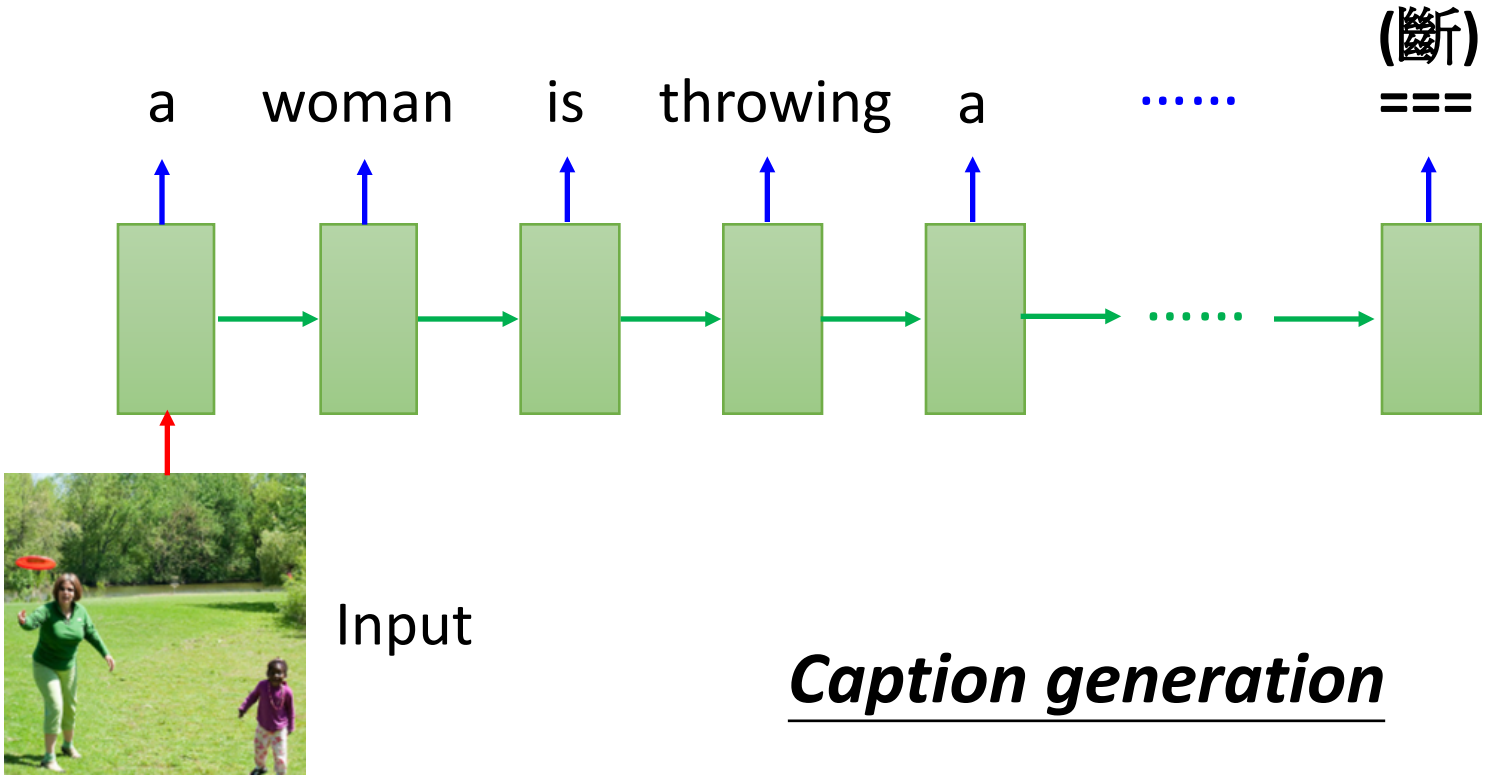
Many to Many (No Limitation)

- Both input and output are vector sequences ***with different lengths***. → ***Sequence to sequence learning***



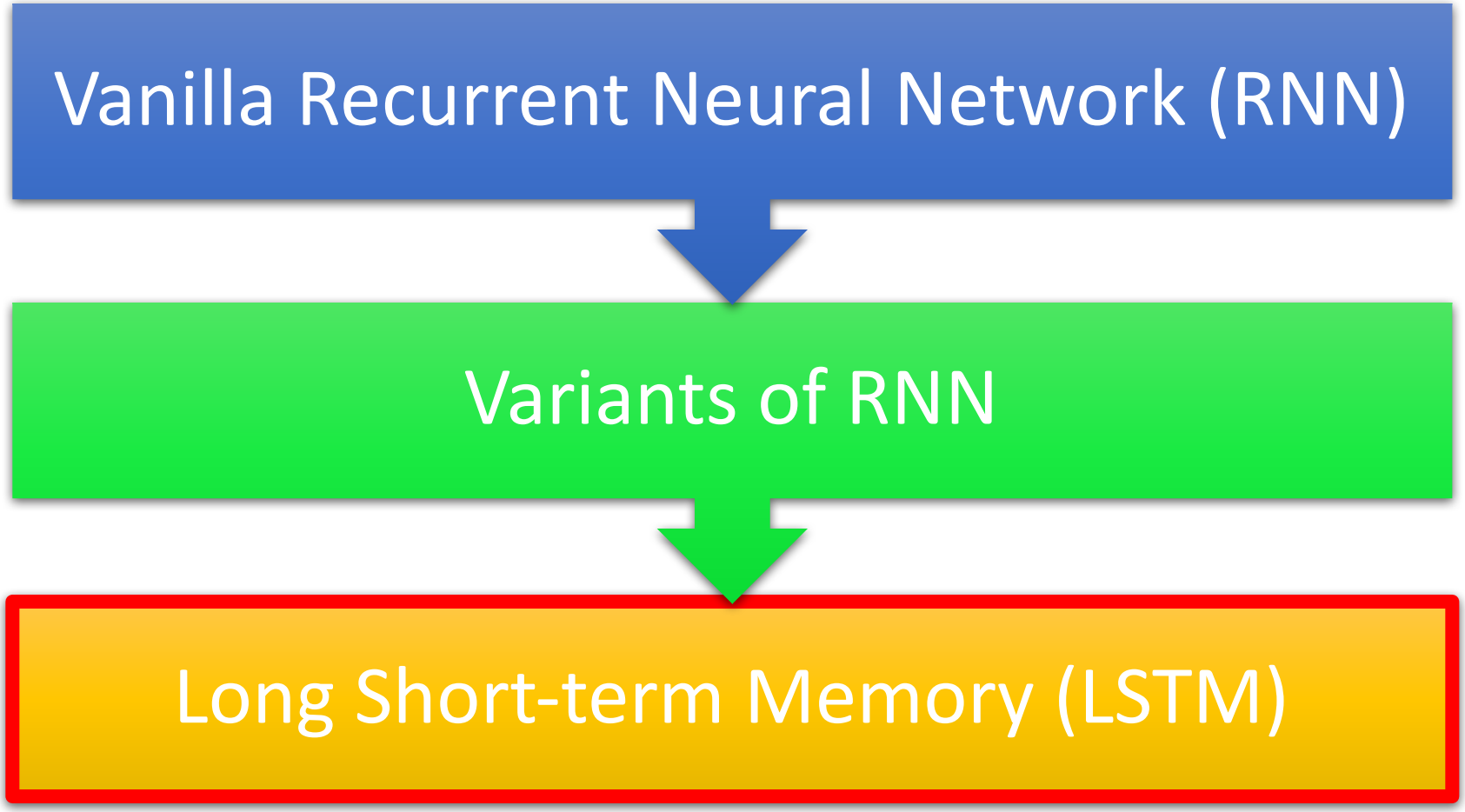
One to Many

- Input is one vector, but output is a vector sequence

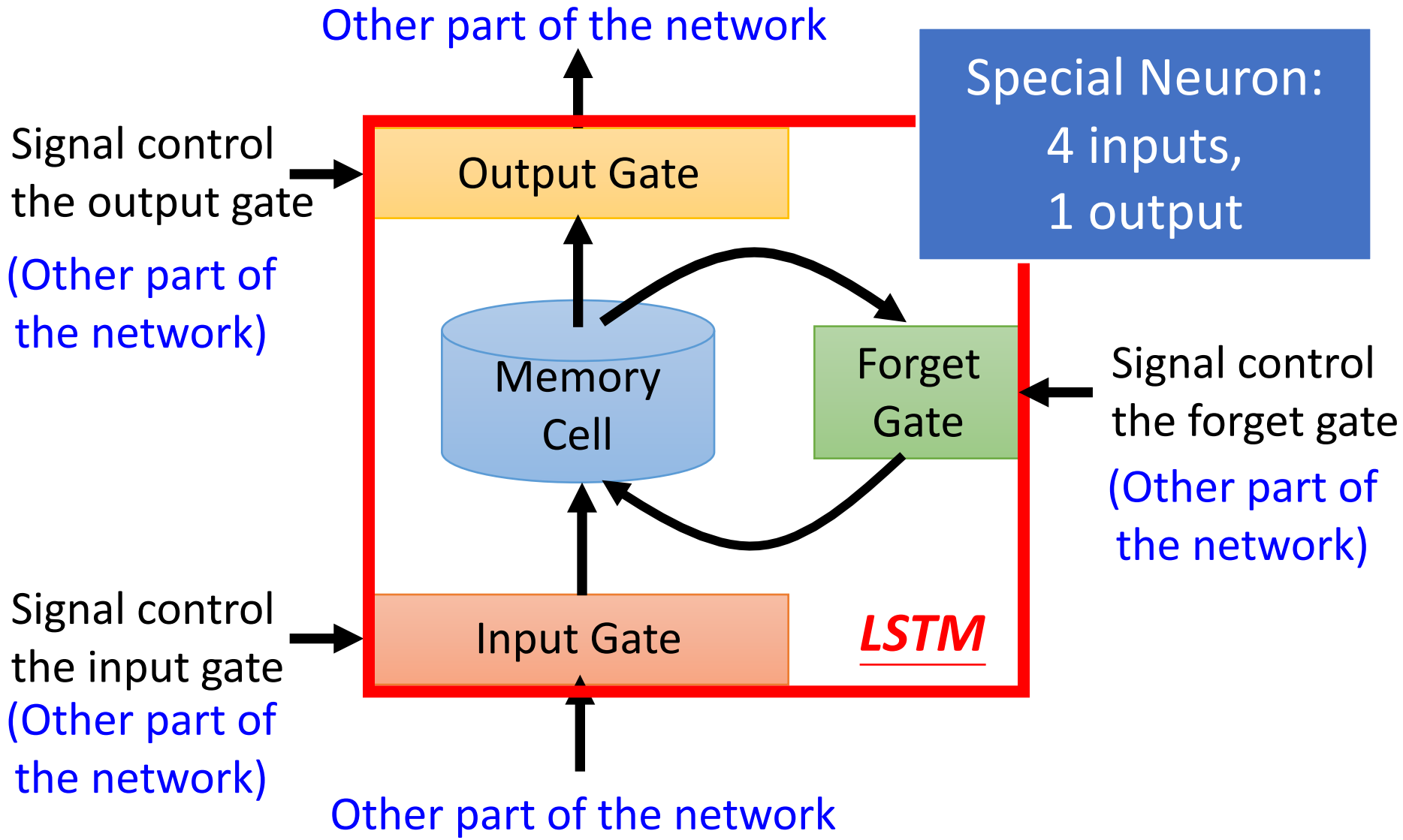


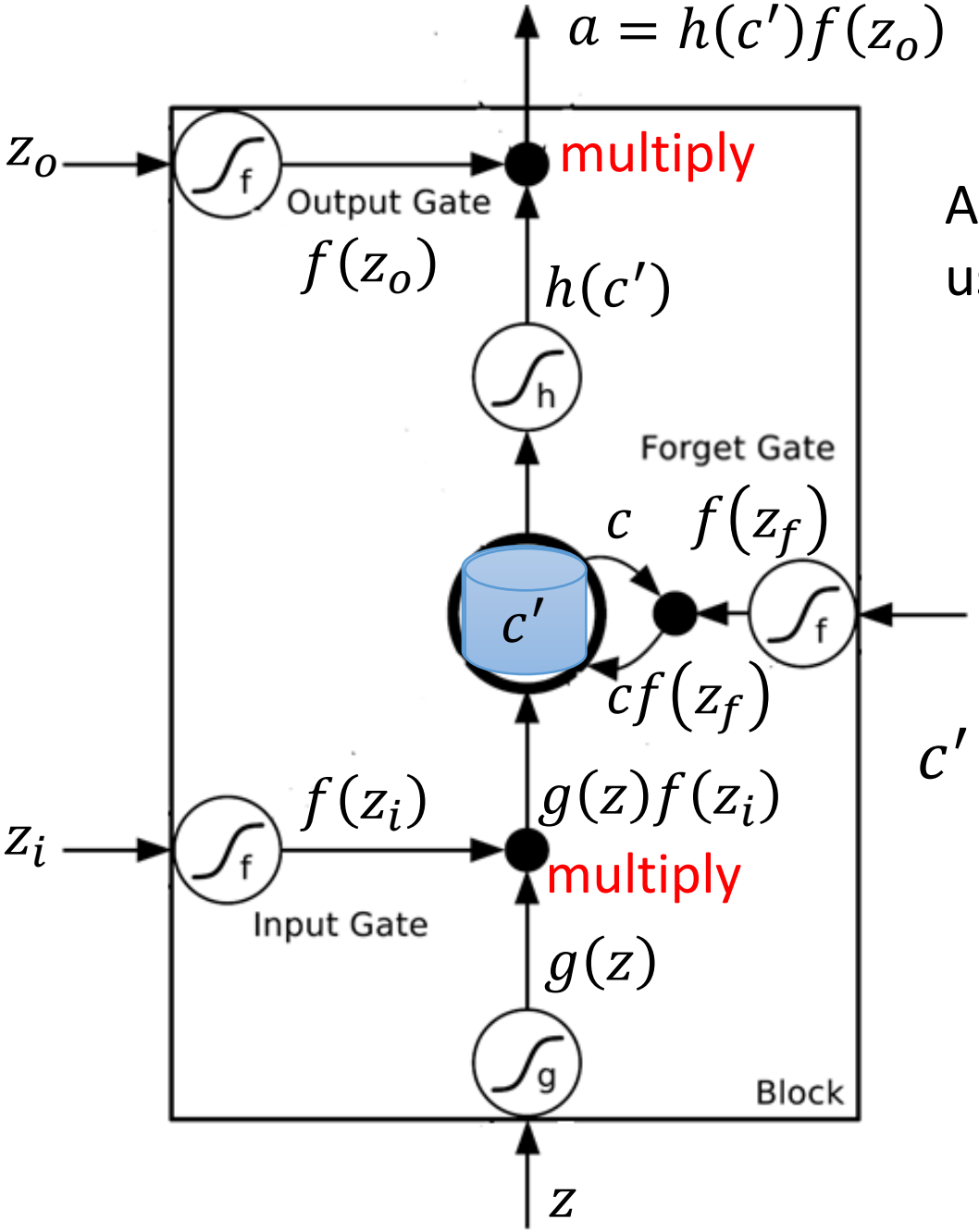
Caption generation

Outline


Vanilla Recurrent Neural Network (RNN)
Variants of RNN
Long Short-term Memory (LSTM)

Long Short-term Memory (LSTM)





Activation function f is usually a sigmoid function

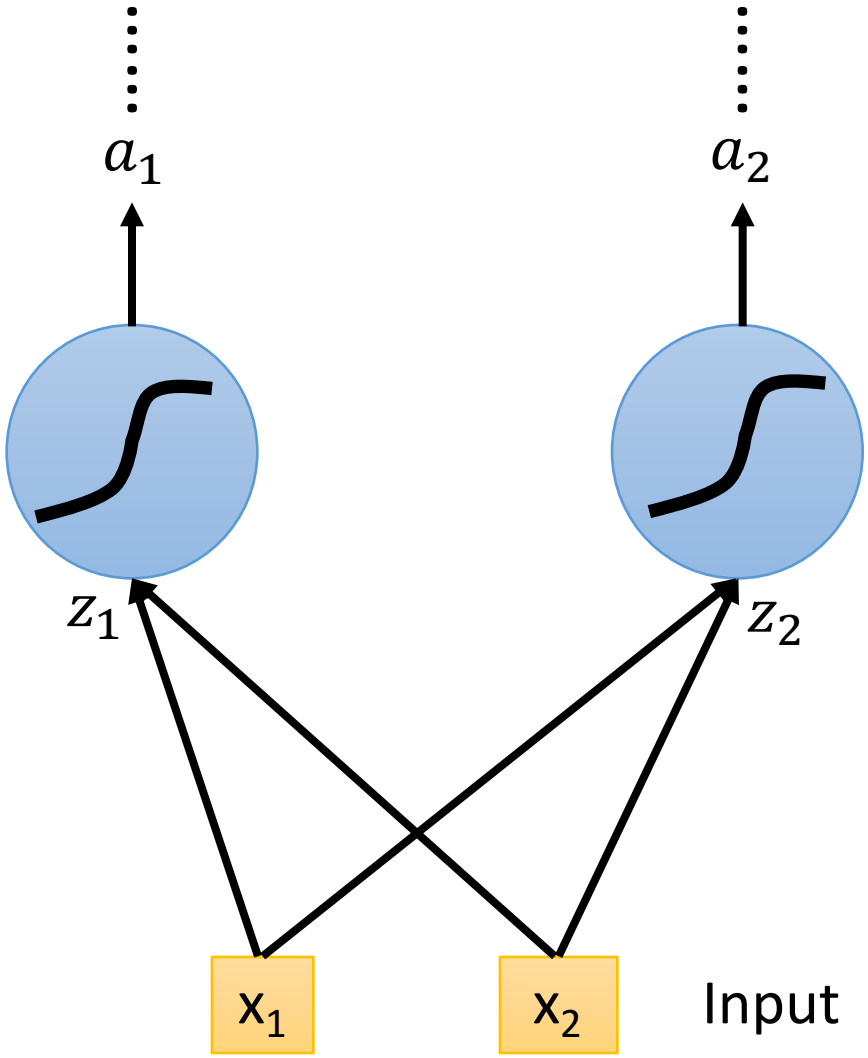
Between 0 and 1

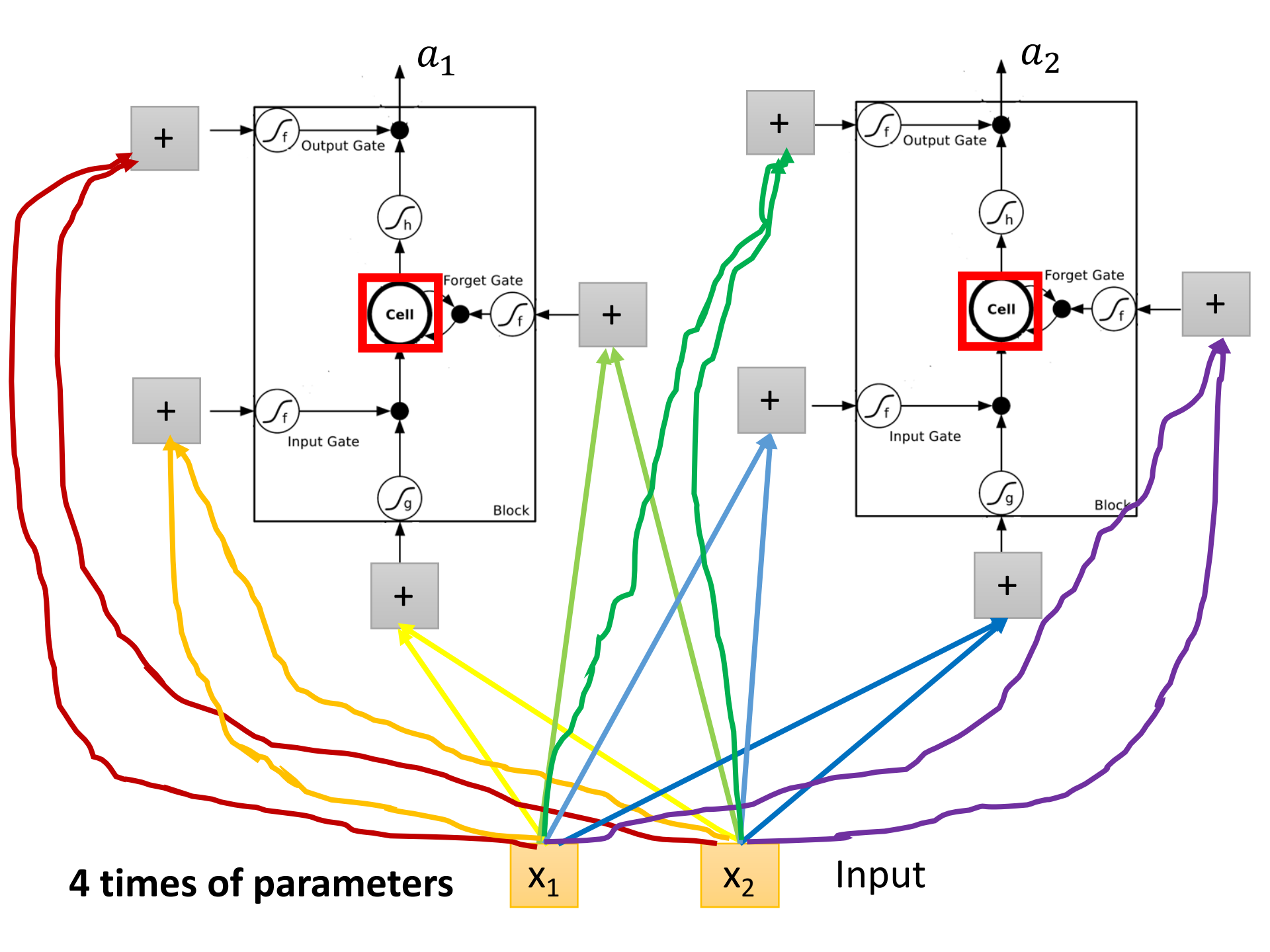
Mimic open and close gate

$$c' = g(z)f(z_i) + cf(z_f)$$

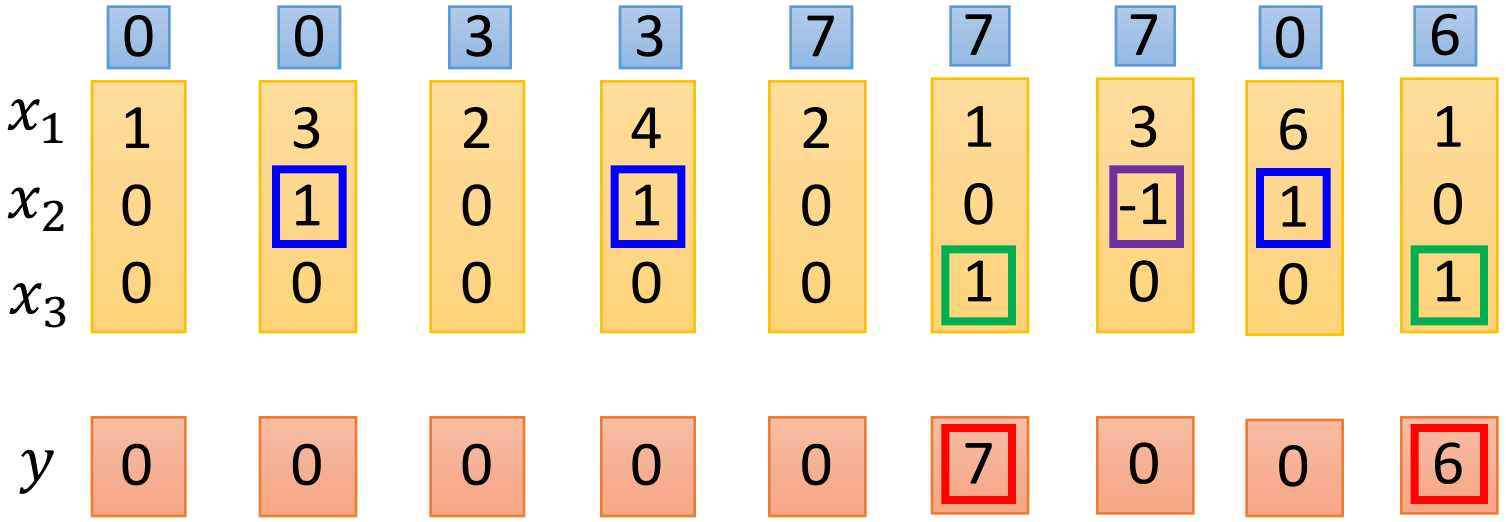
Original Network:

- Simply replace the neurons with LSTM




a_1
a_2
Output Gate
Forget Gate
Input Gate
Cell
Block
x_1
x_2
Input
4 times of parameters

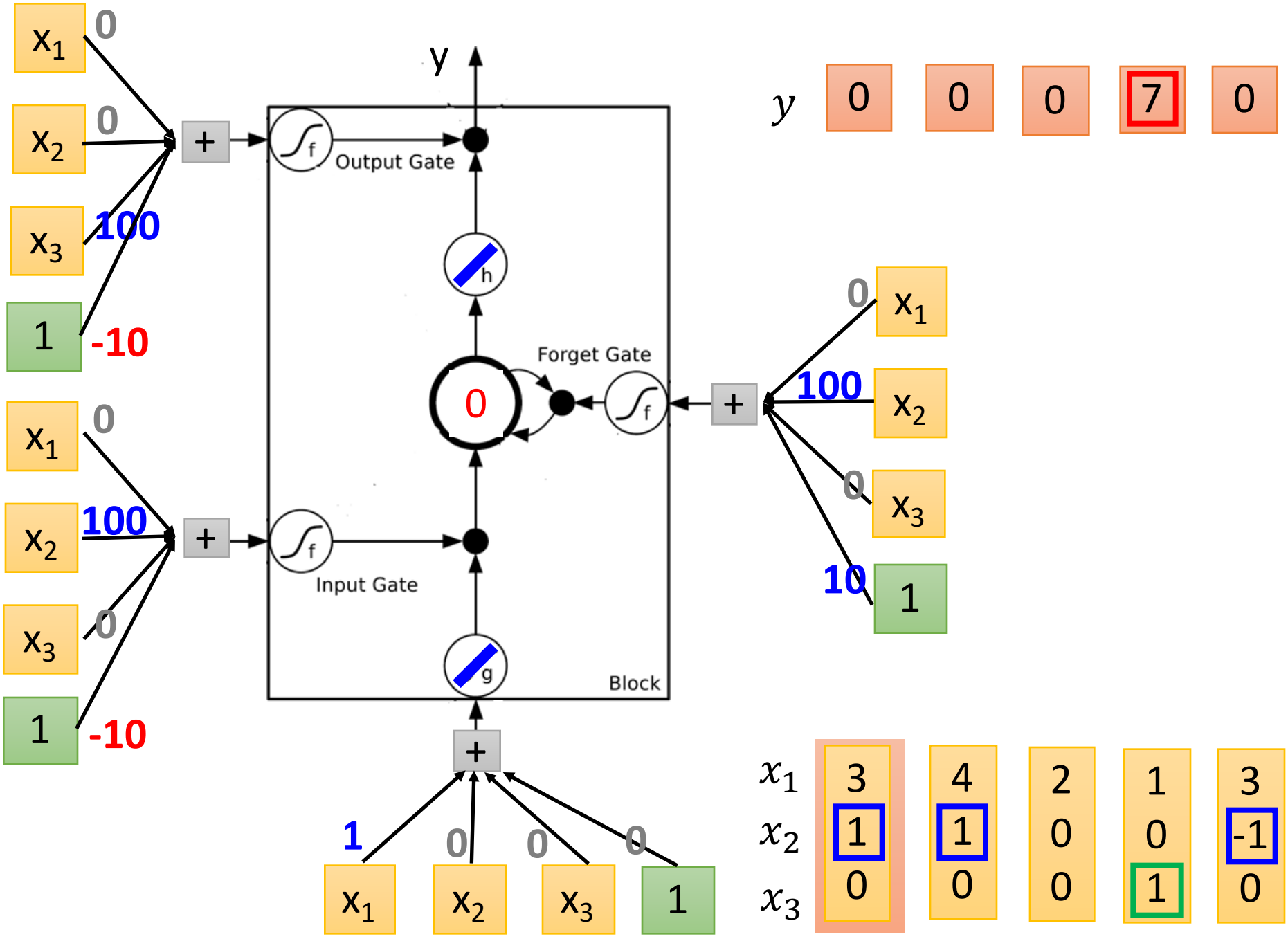
LSTM - Example

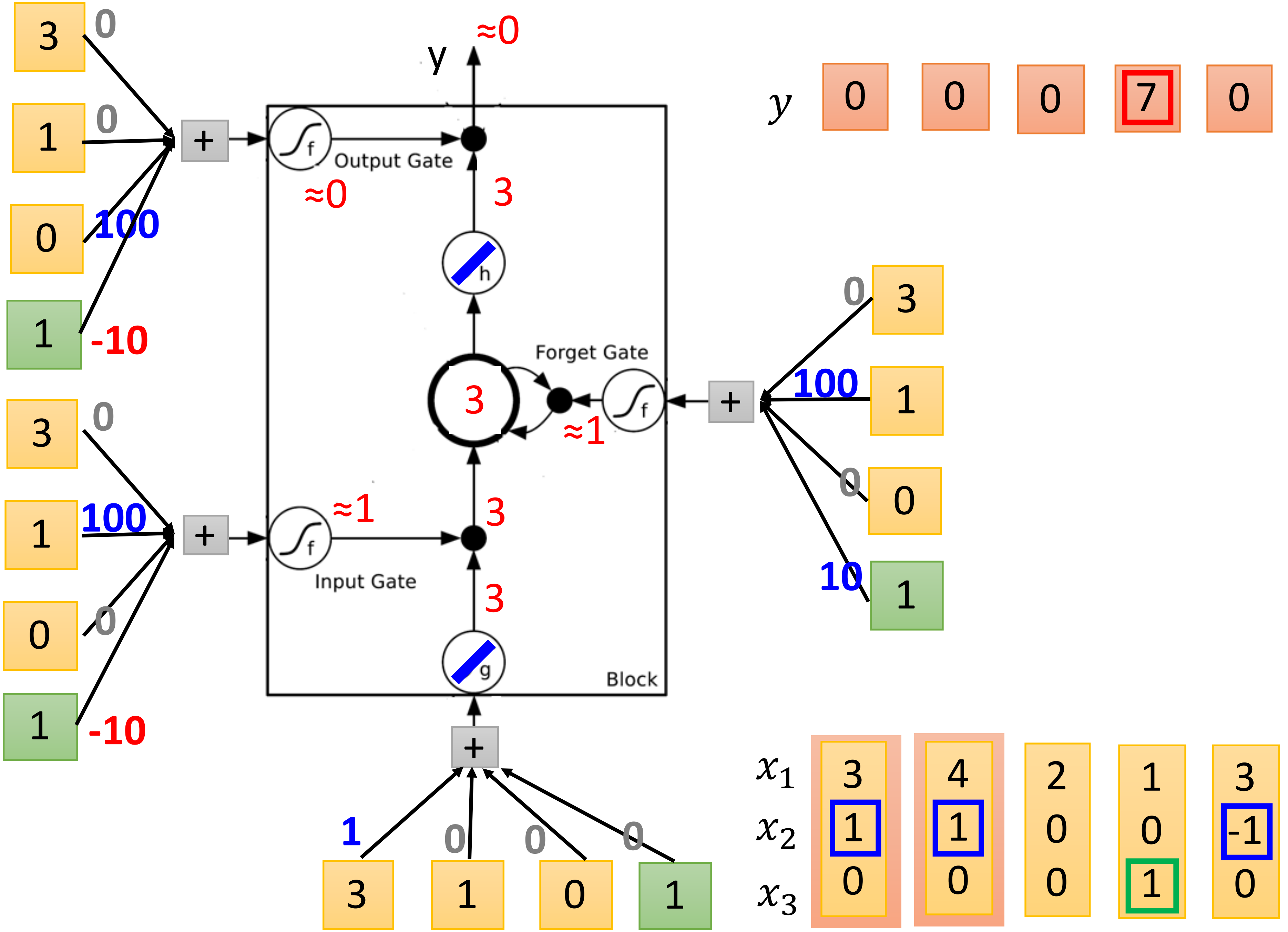


When x_2 = 1, add the numbers of x_1 into the memory

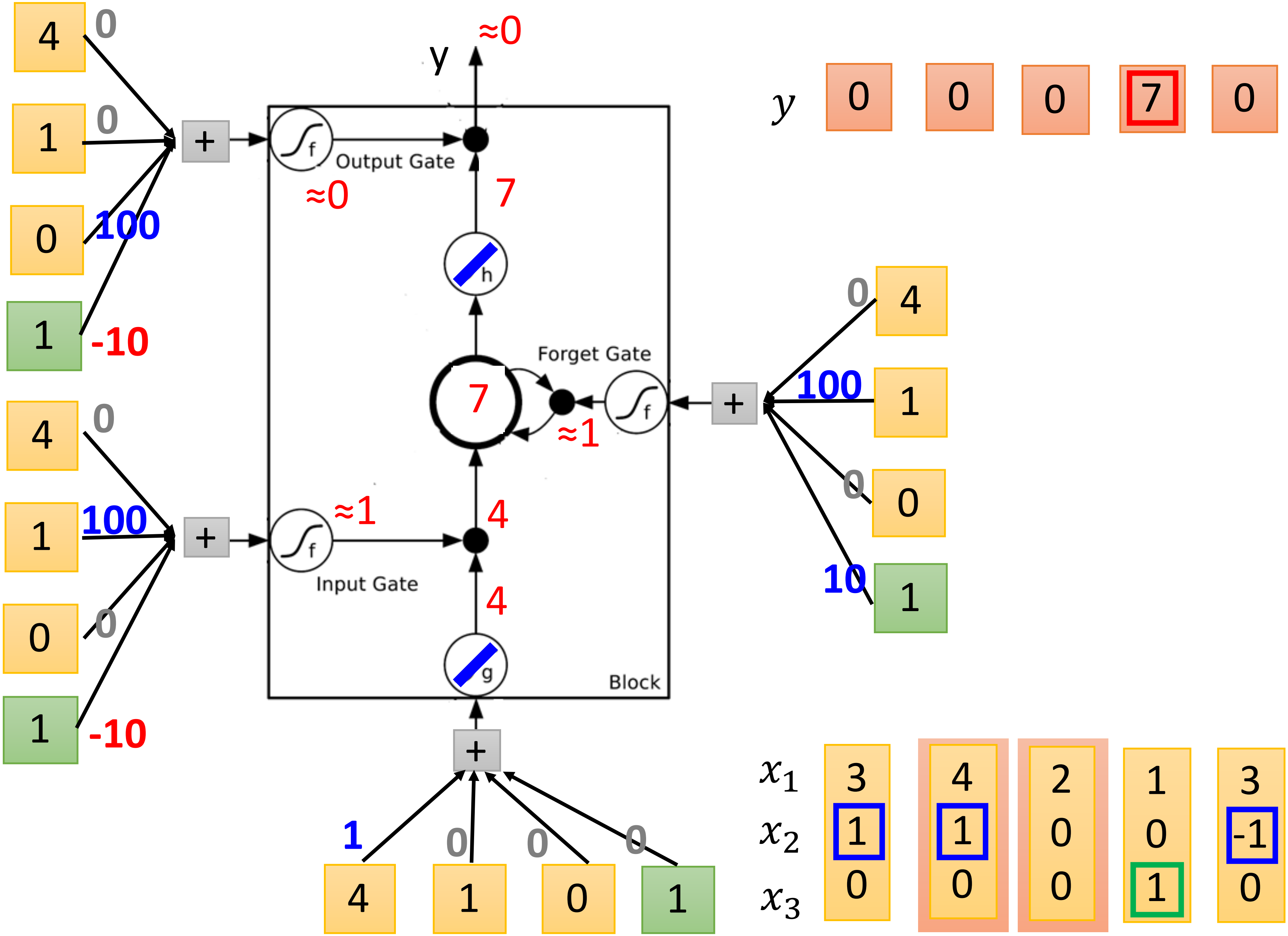
When x_2 = -1, reset the memory

When x_3 = 1, output the number in the memory.

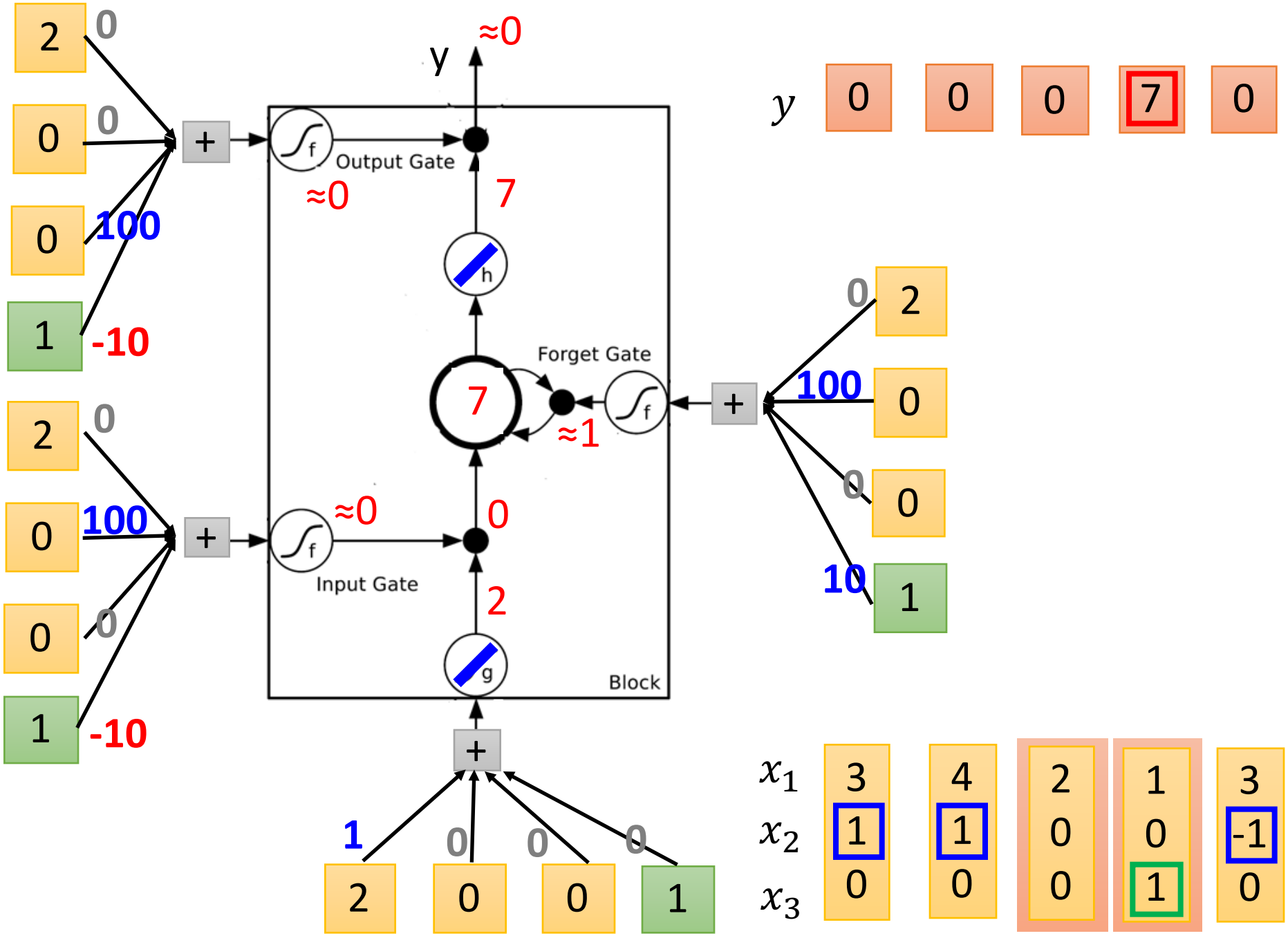

y
Output Gate
Forget Gate
Input Gate
Block
0
100
-10
1
x1
x2
x3
+
f
h
g
y 0 0 0 7 0
x1 3 4 2 1 3
x2 1 1 0 0 -1
x3 0 0 0 1 0

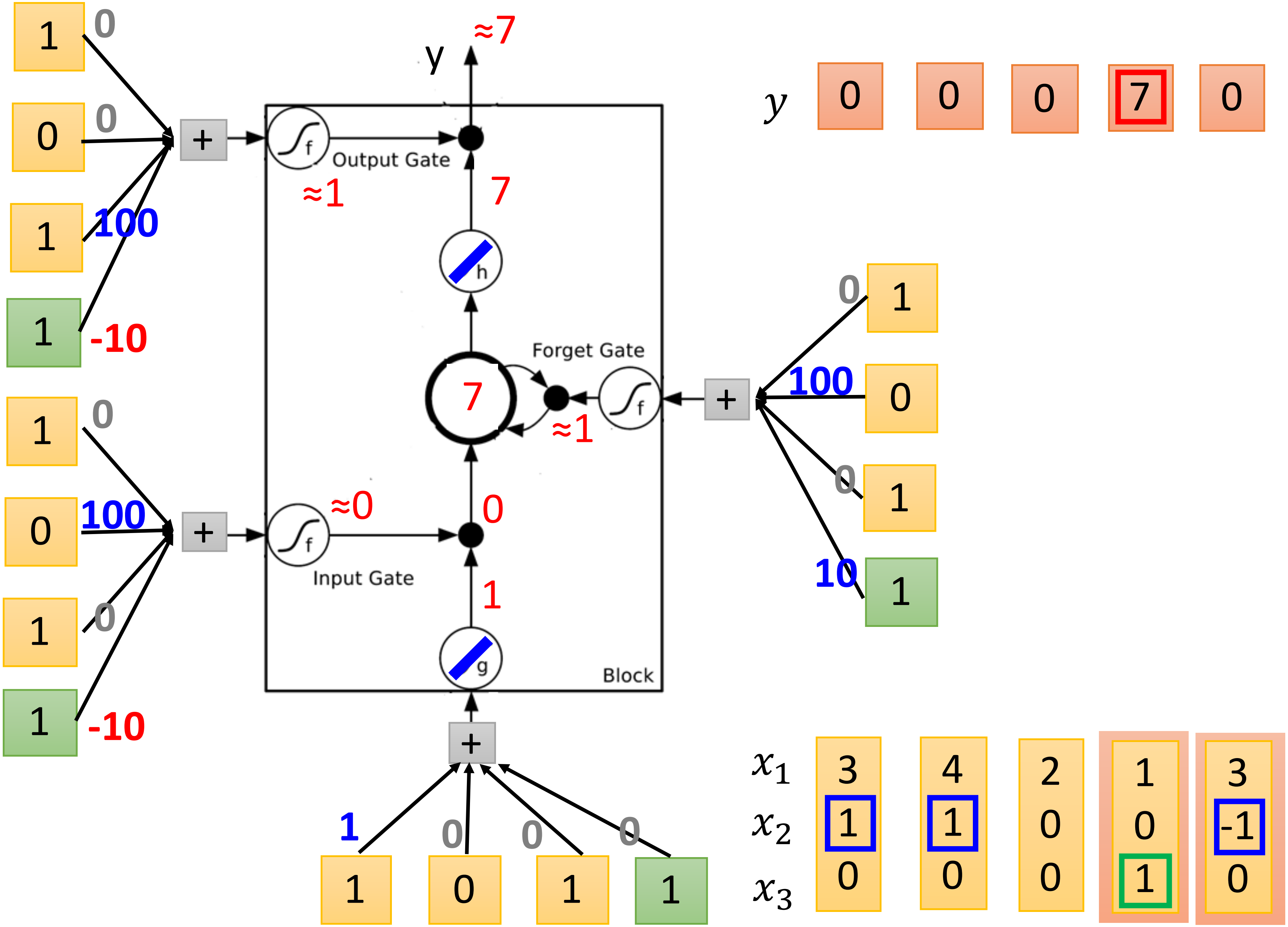


y
≈0
Output Gate
≈0
3
Forget Gate
3
≈1
Input Gate
≈1
3
3
Block
+
+
+
+
3 0
1 0
0 100
1 -10
3 0
1 100
0 0
1 -10
0 3
100 1
0 0
10 1
1 0 0 0
3 1 0 1
y 0 0 0 7 0
x_1 3 4 2 1 3
x_2 1 1 0 0 -1
x_3 0 0 0 1 0

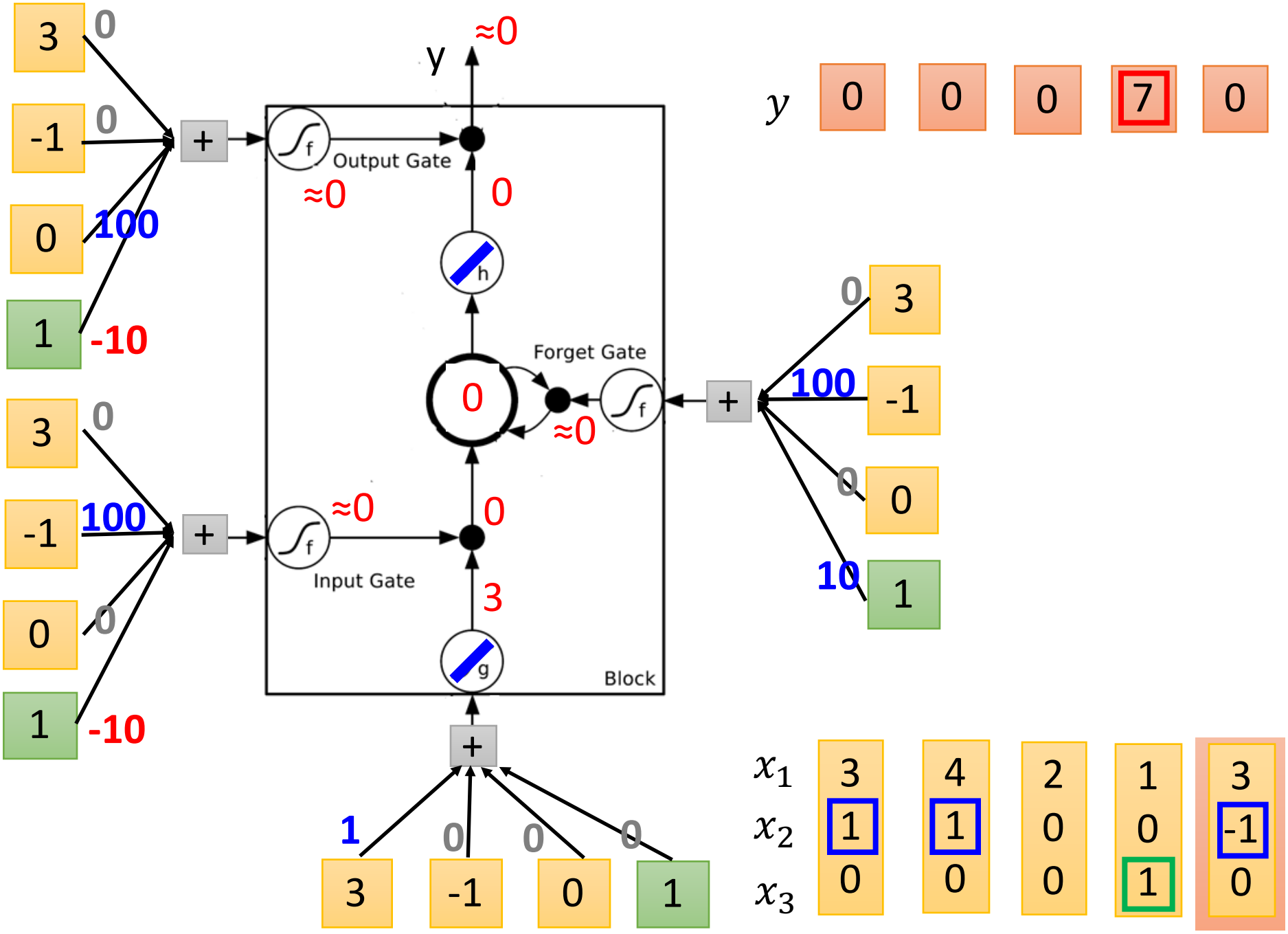


Output Gate
Forget Gate
Input Gate
Block
y
≈0
7
4
≈1
100
-10
10
y 0 0 0 7 0
x_1 3 4 2 1 3
x_2 1 1 0 0 -1
x_3 0 0 0 1 0


y
≈0
Output Gate
≈0
7
Forget Gate
7
≈1
Input Gate
≈0
0
2
Block
+
+
+
+
f
f
f
h
g
2
0
0
100
1
-10
2
0
100
0
0
-10
0
2
0
100
0
0
10
1
1
0
0
0
2
0
0
1
y
0
0
0
7
0
x_1
x_2
x_3
3
1
0
4
1
0
2
0
0
1
0
1
3
-1
0

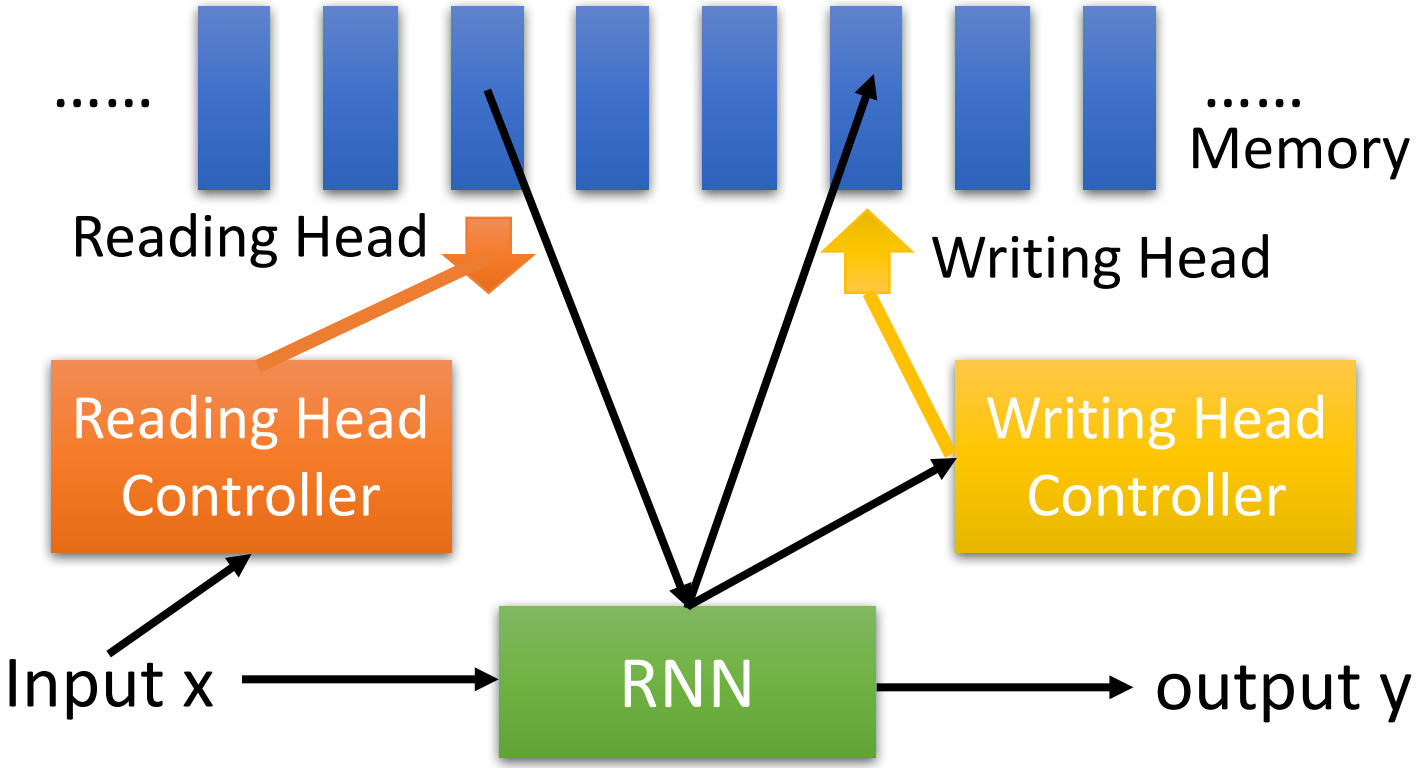


y
≈7
Output Gate
≈1
7
Forget Gate
≈1
≈0
0
Input Gate
1
Block
100
-10
10
x1
x2
x3


3
-1
0
1
0
0
100
-10
+
y
≈0
f
Output Gate
≈0
0
h
Forget Gate
0
≈0
f
+
0
3
100
-1
0
0
10
1
3
-1
0
1
100
0
-10
+
≈0
f
Input Gate
0
3
g
Block
+
1
0
0
0
3
-1
0
1
y
0
0
0
7
0
x_1
x_2
x_3
3
1
0
4
1
0
2
0
0
1
0
1
3
-1
0

What is the next wave?

- Attention-based Model



Recommended Reading List

- The Unreasonable Effectiveness of Recurrent Neural Networks
  - http://karpathy.github.io/2015/05/21/rnn-effectiveness/
- Understanding LSTM Networks
  - http://colah.github.io/posts/2015-08-Understanding-LSTMs/
- Attention Is All You Need
  - https://arxiv.org/abs/1706.03762