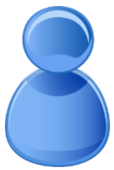
Recurrent Neural Network (RNN)

Example Application

- Slot Filling



I would like to arrive Taipei on November 2nd.

ticket booking system

Slot

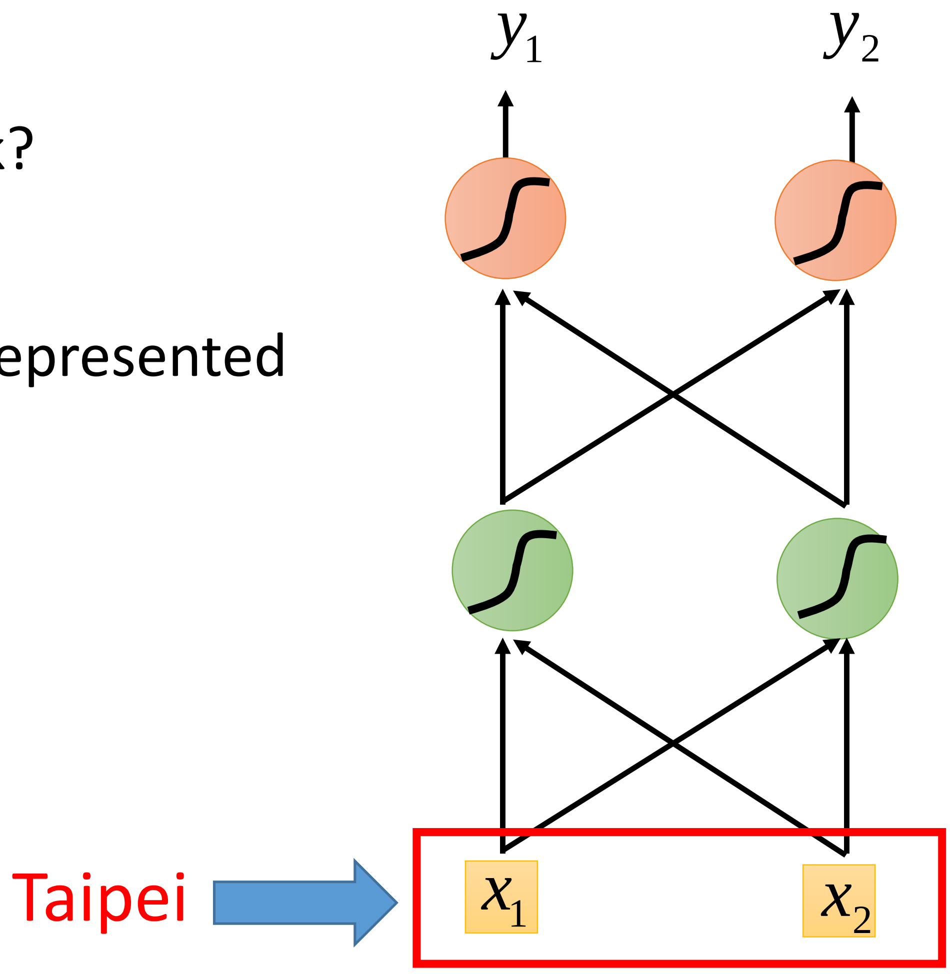
Destination:	Taipei
time of arrival:	November 2nd

Example Application

Solving slot filling by Feedforward network?

Input: a word

(Each word is represented as a vector)



1-of-N encoding

How to represent each word as a vector?

1-of-N Encoding lexicon = {apple, bag, cat, dog, elephant}

The vector is lexicon size.

Each dimension corresponds to a word in the lexicon

The dimension for the word is 1, and others are 0

apple = [1 0 0 0 0]

bag = [0 1 0 0 0]

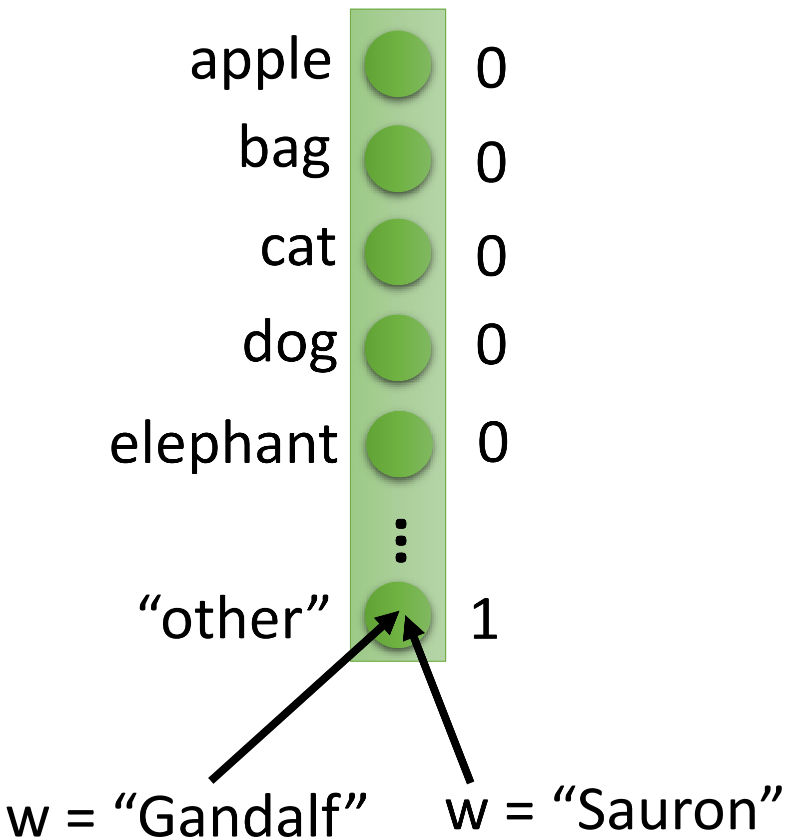
cat = [0 0 1 0 0]

dog = [0 0 0 1 0]

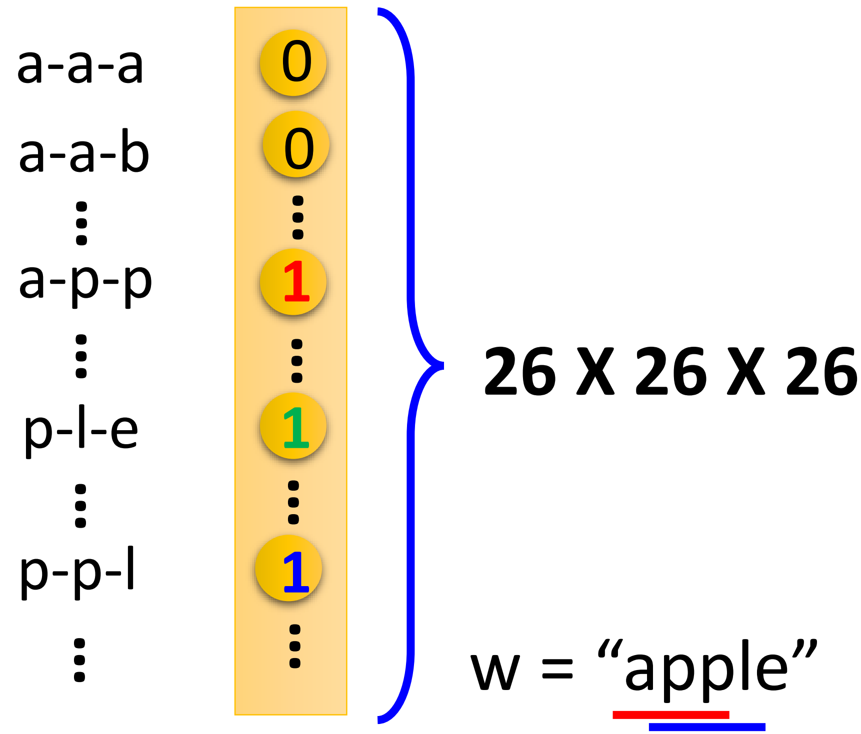
elephant = [0 0 0 0 1]

Beyond 1-of-N encoding

Dimension for "Other"



Word hashing



Example Application

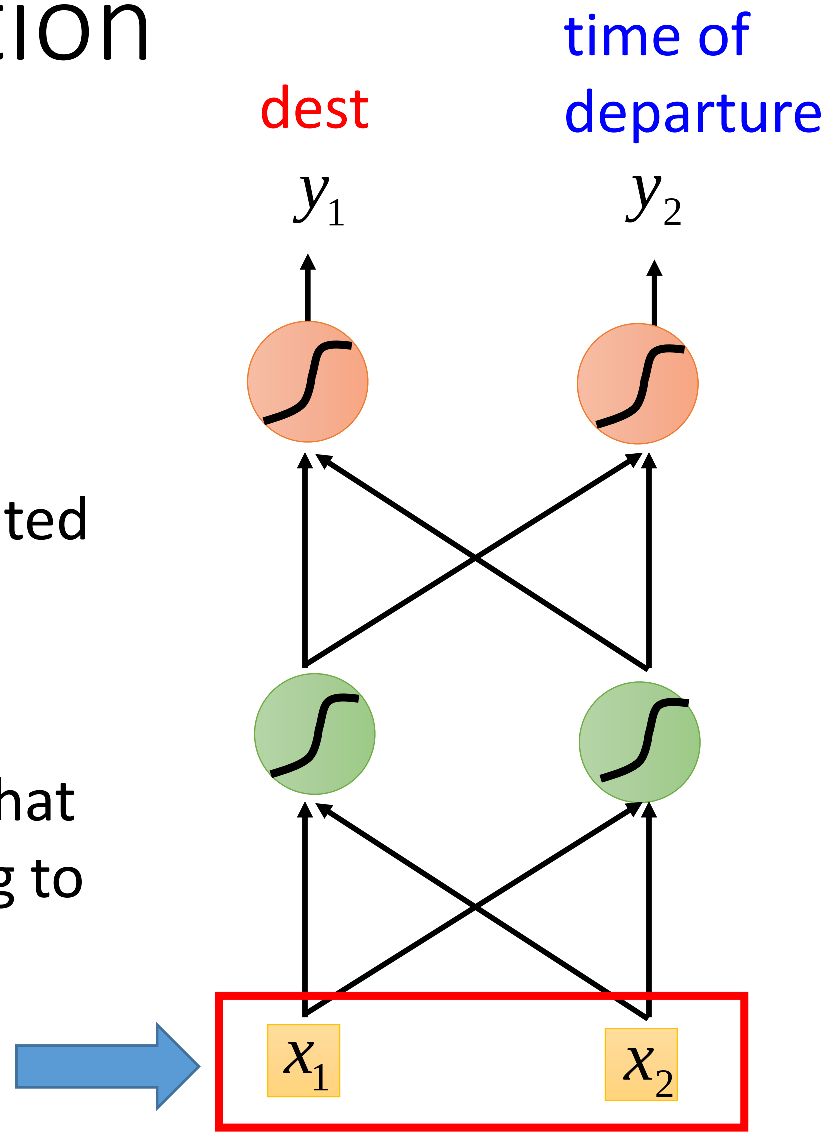
Solving slot filling by Feedforward network?

Input: a word

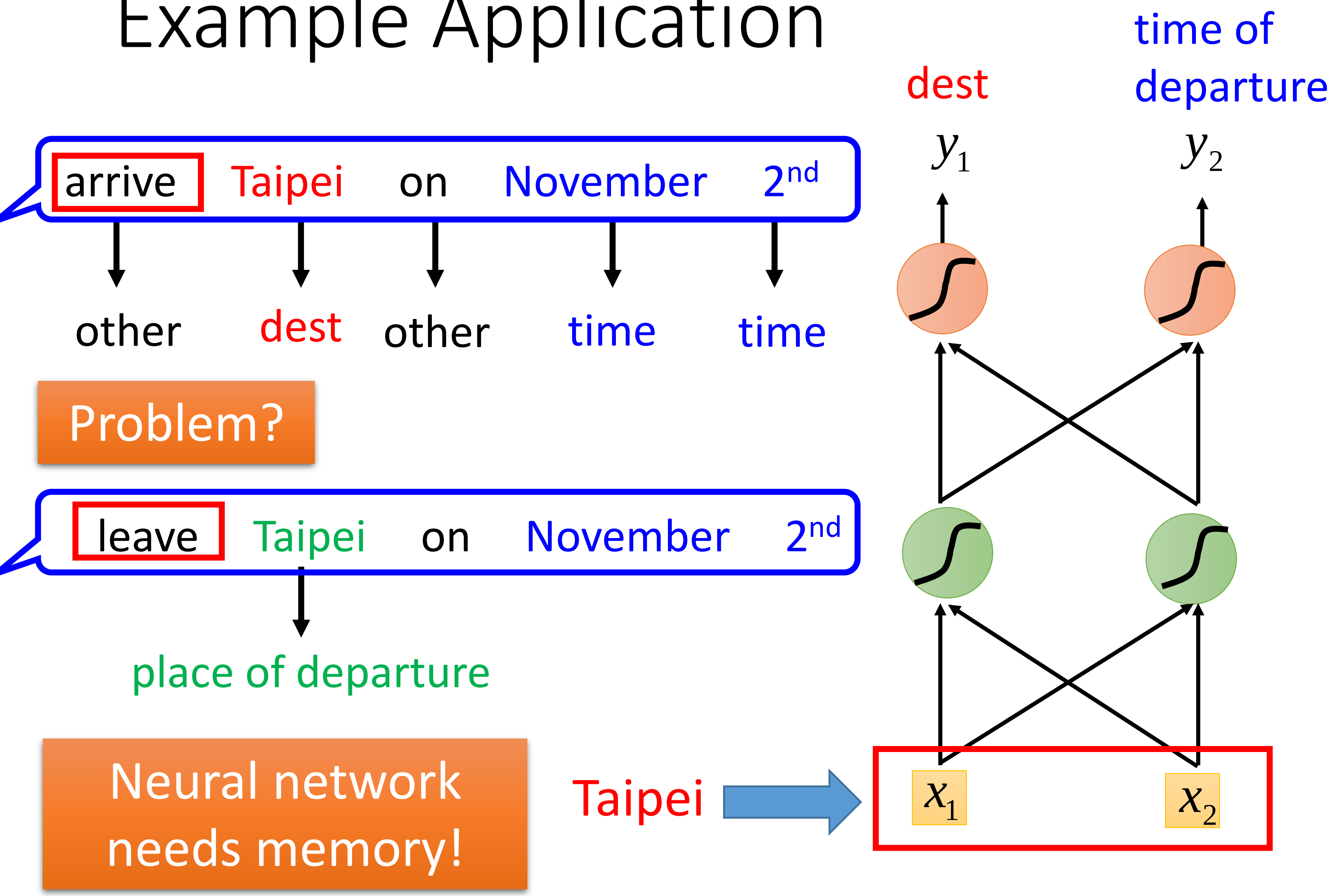
(Each word is represented as a vector)

Output:

Probability distribution that the input word belonging to the slots

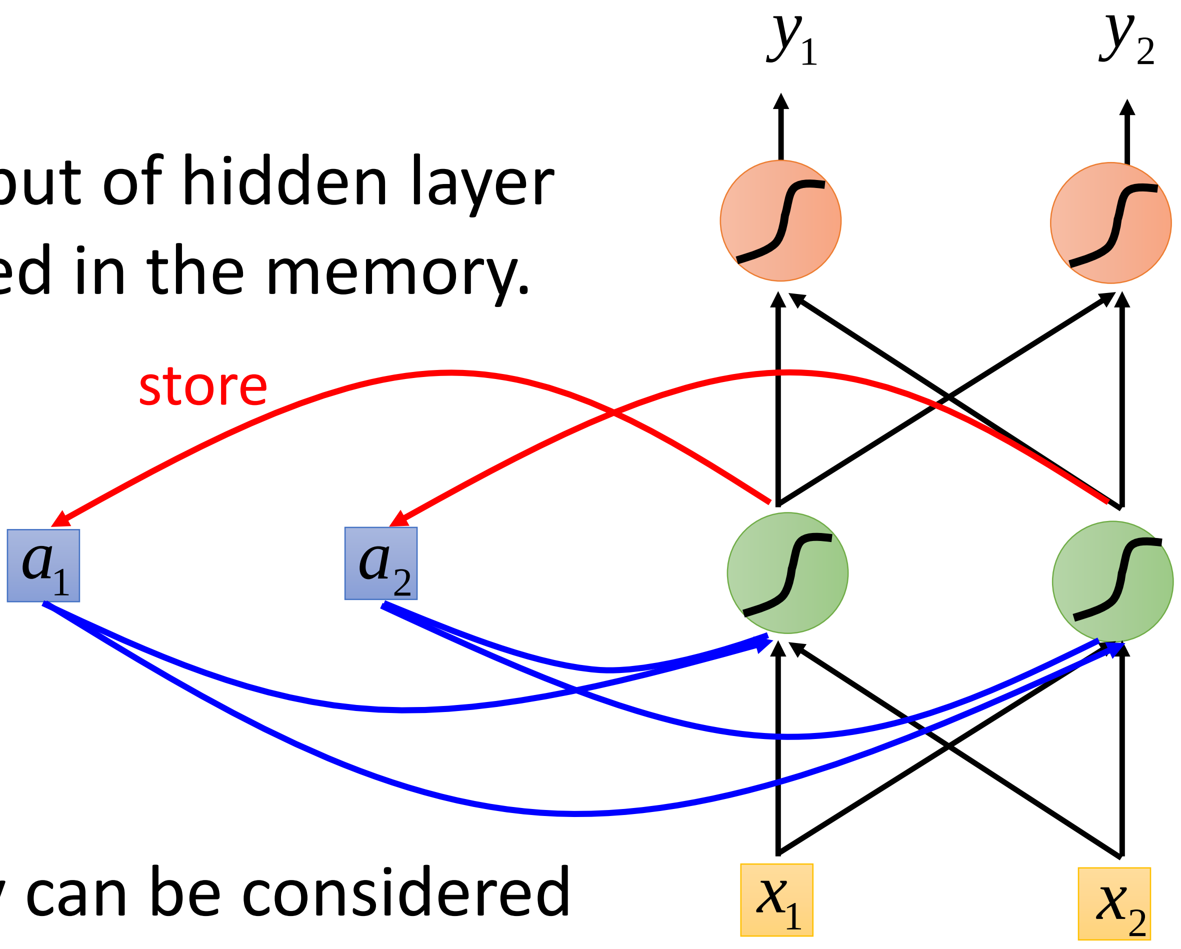


Example Application



Recurrent Neural Network (RNN)

The output of hidden layer are stored in the memory.

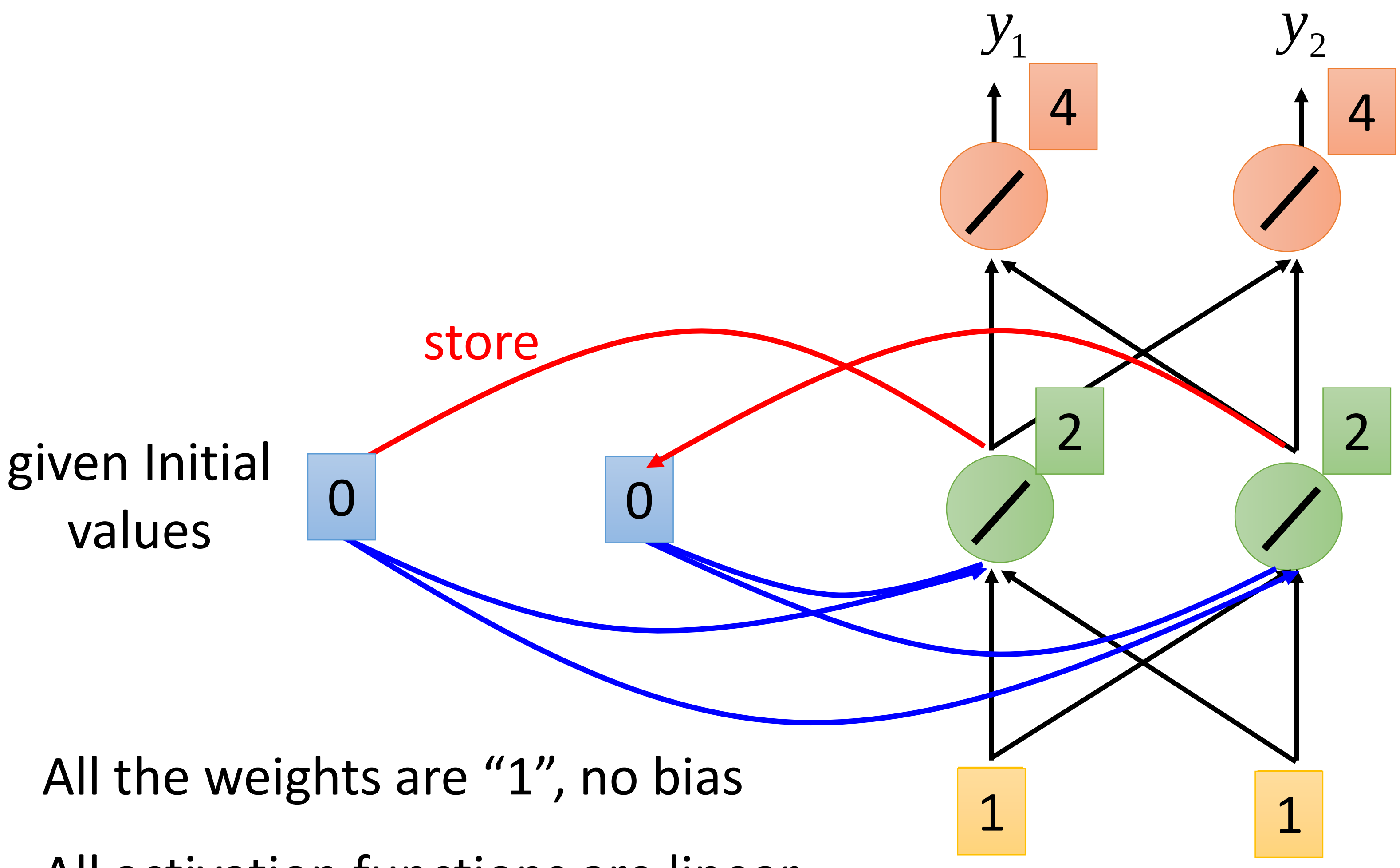


Memory can be considered as another input.

Example

Input sequence: $\begin{bmatrix}1\\1\end{bmatrix}\begin{bmatrix}1\\1\end{bmatrix}\begin{bmatrix}2\\2\end{bmatrix}$ ……

output sequence: $\begin{bmatrix}4\\4\end{bmatrix}$



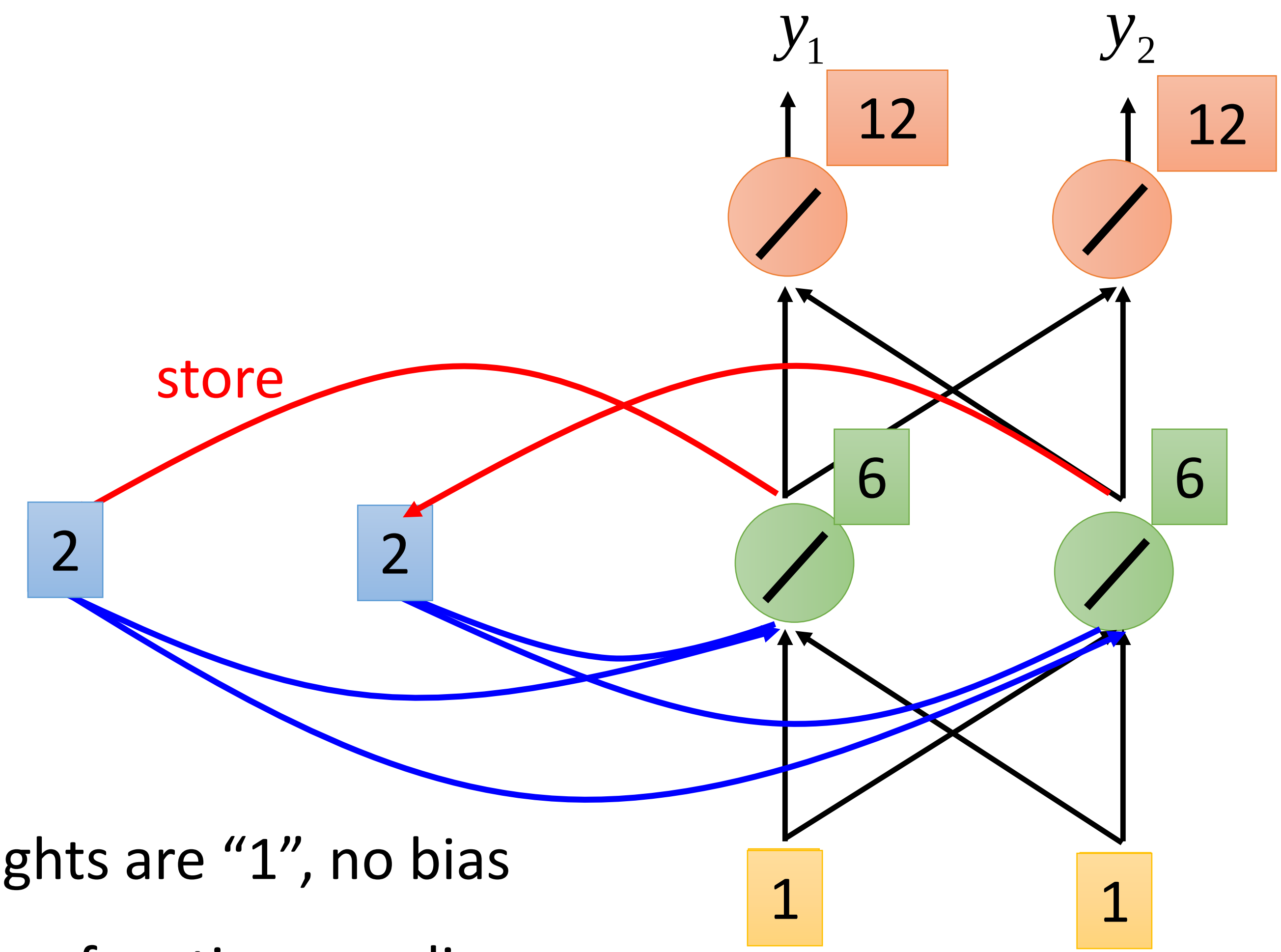
All the weights are "1", no bias

All activation functions are linear

Example

Input sequence: $\begin{bmatrix}1\\1\end{bmatrix} \begin{bmatrix}1\\1\end{bmatrix} \begin{bmatrix}2\\2\end{bmatrix}$ ……

output sequence: $\begin{bmatrix}4\\4\end{bmatrix} \begin{bmatrix}12\\12\end{bmatrix}$



All the weights are "1", no bias

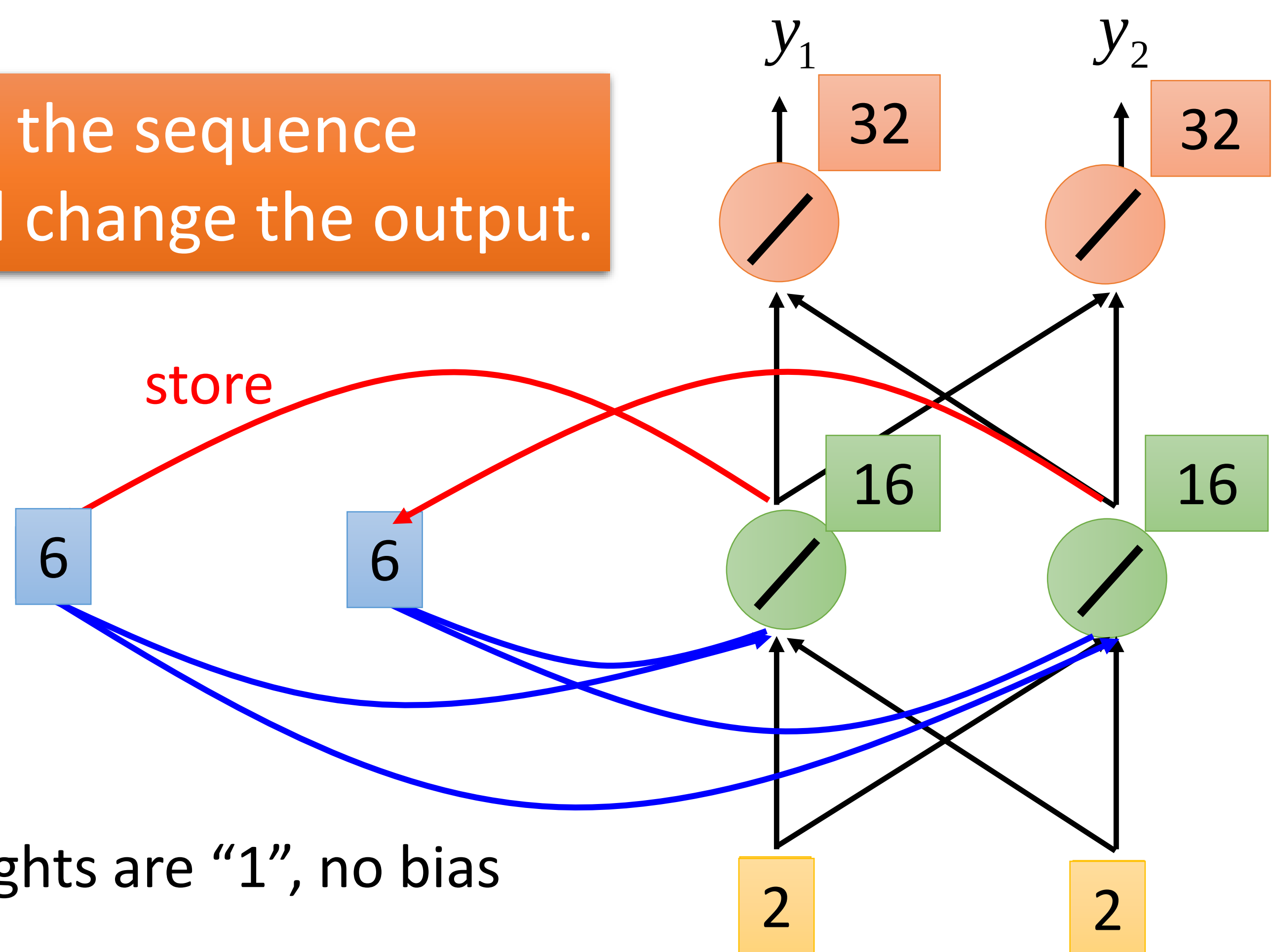
All activation functions are linear

Example

Input sequence: $\begin{bmatrix}1\\1\end{bmatrix}\begin{bmatrix}1\\1\end{bmatrix}\begin{bmatrix}2\\2\end{bmatrix}$ ……

output sequence: $\begin{bmatrix}4\\4\end{bmatrix}\begin{bmatrix}12\\12\end{bmatrix}\begin{bmatrix}32\\32\end{bmatrix}$

Changing the sequence order will change the output.

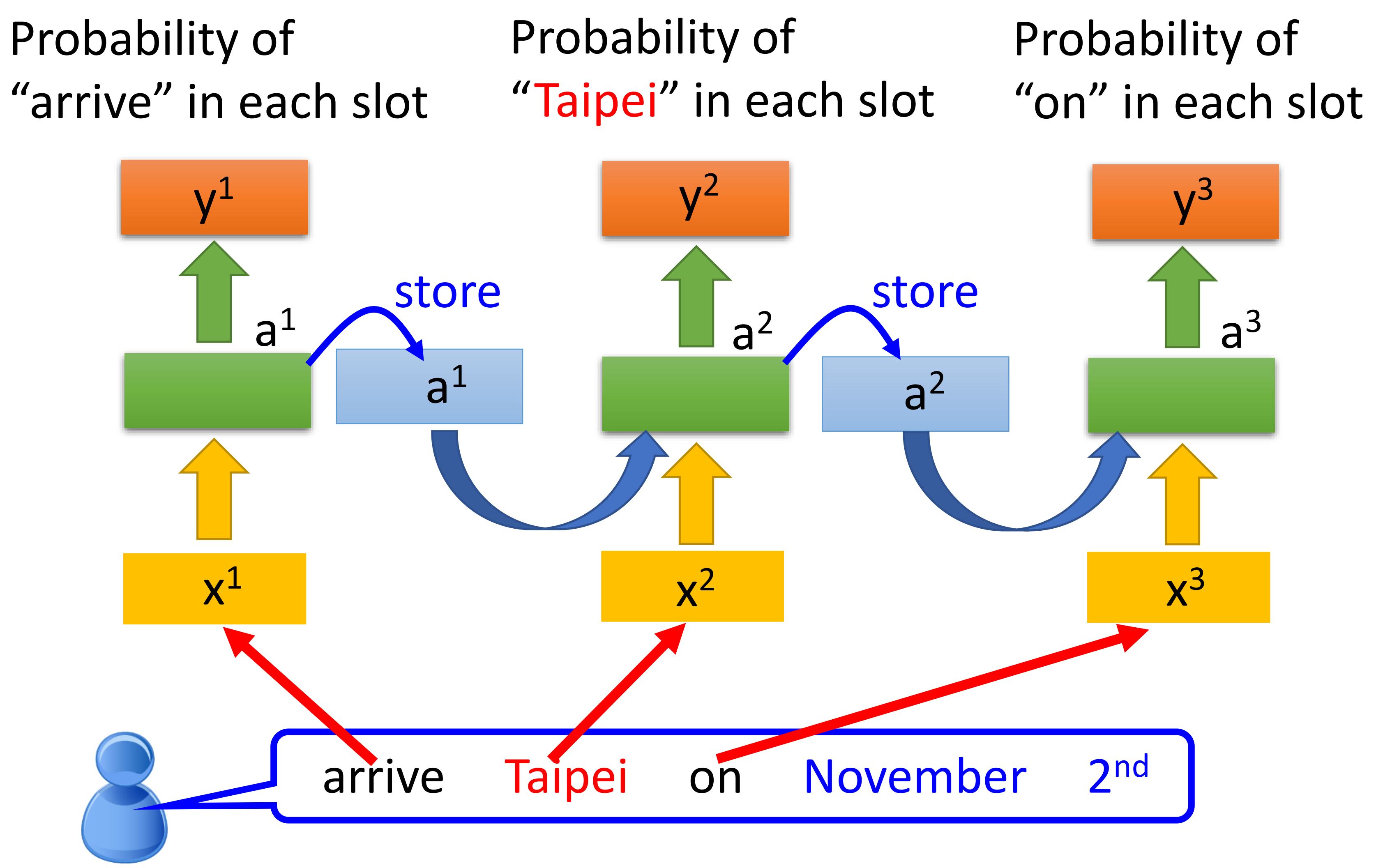


All the weights are "1", no bias

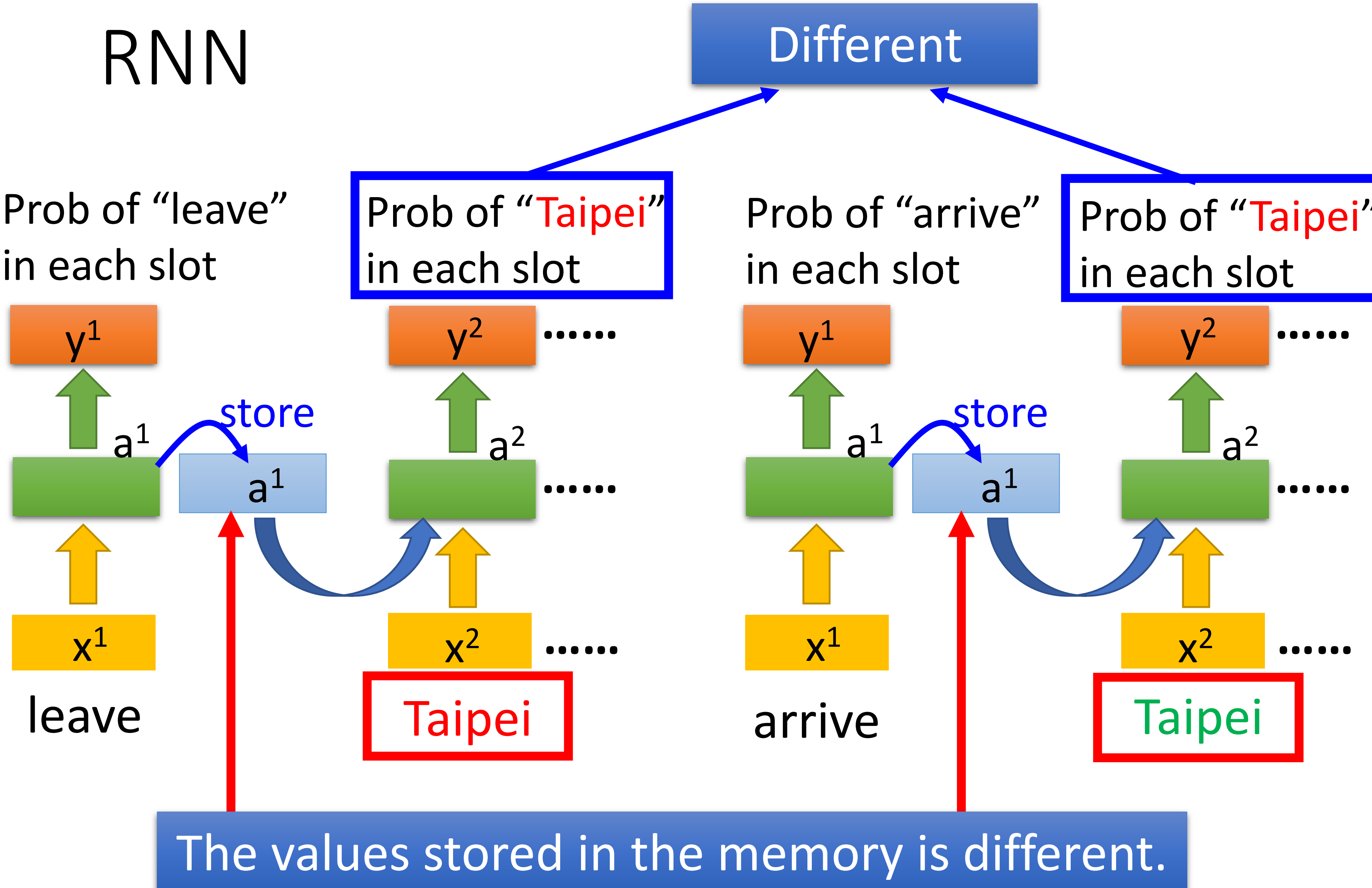
All activation functions are linear

RNN

The same network is used again and again.

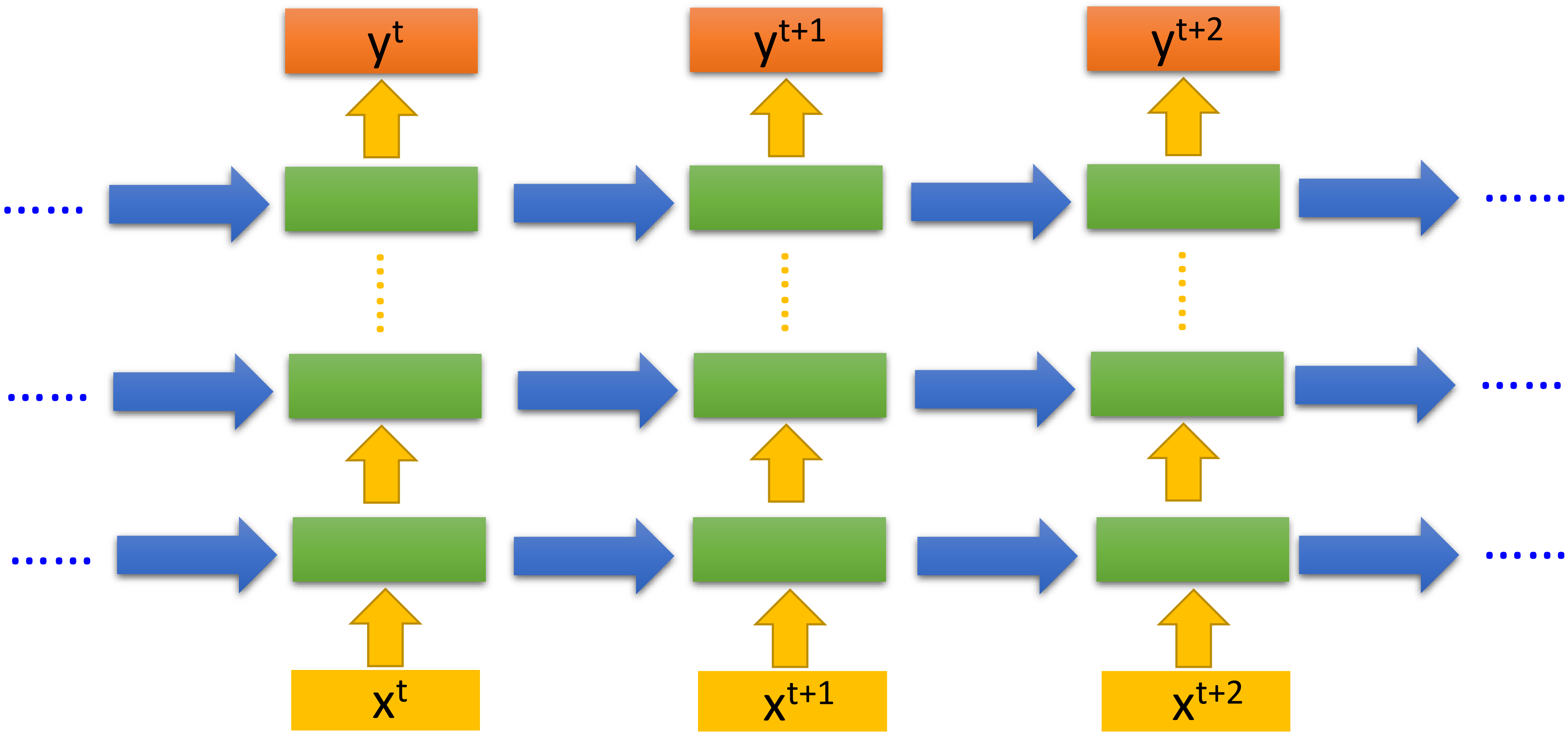


RNN

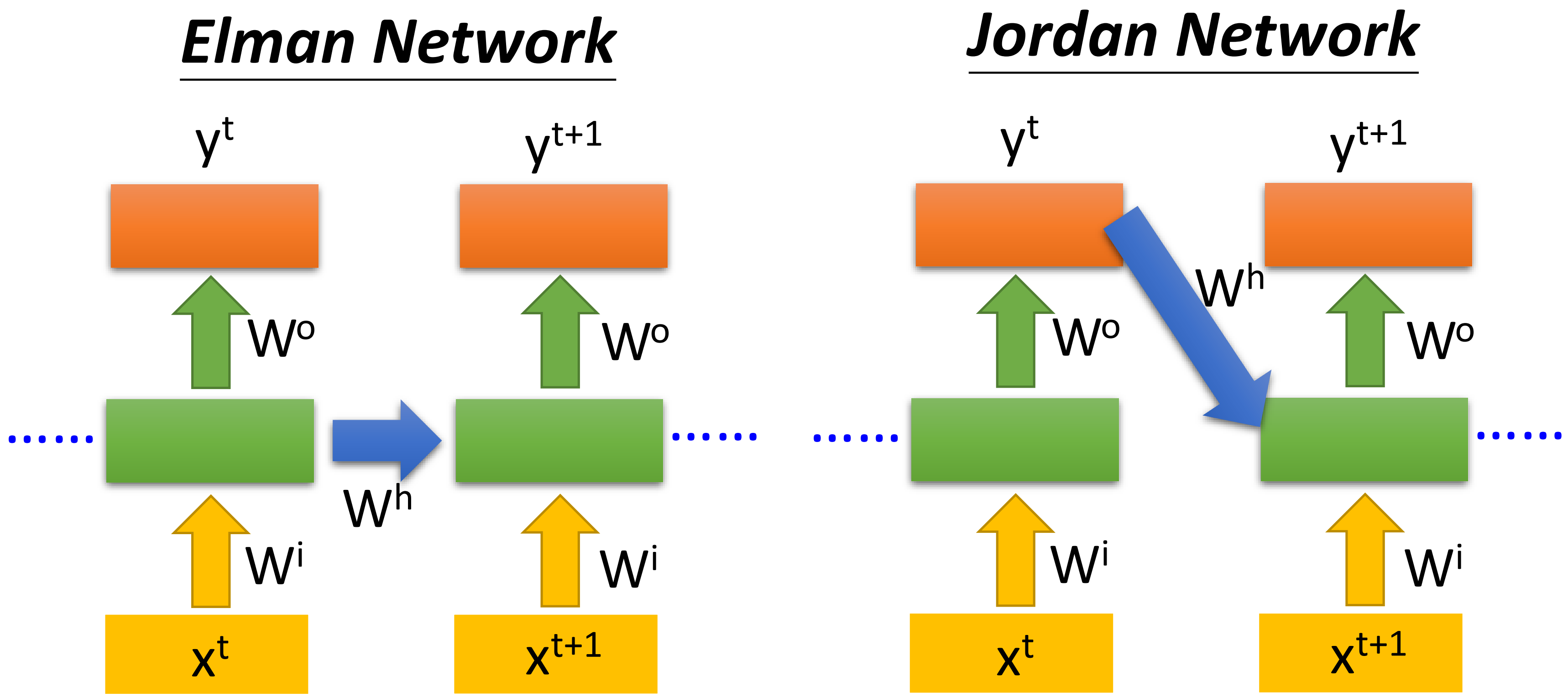


Different
Prob of "leave" in each slot
Prob of "Taipei" in each slot
Prob of "arrive" in each slot
Prob of "Taipei" in each slot
y^1
y^2
a^1
store
a^1
a^2
x^1
x^2
leave
Taipei
arrive
Taipei
The values stored in the memory is different.

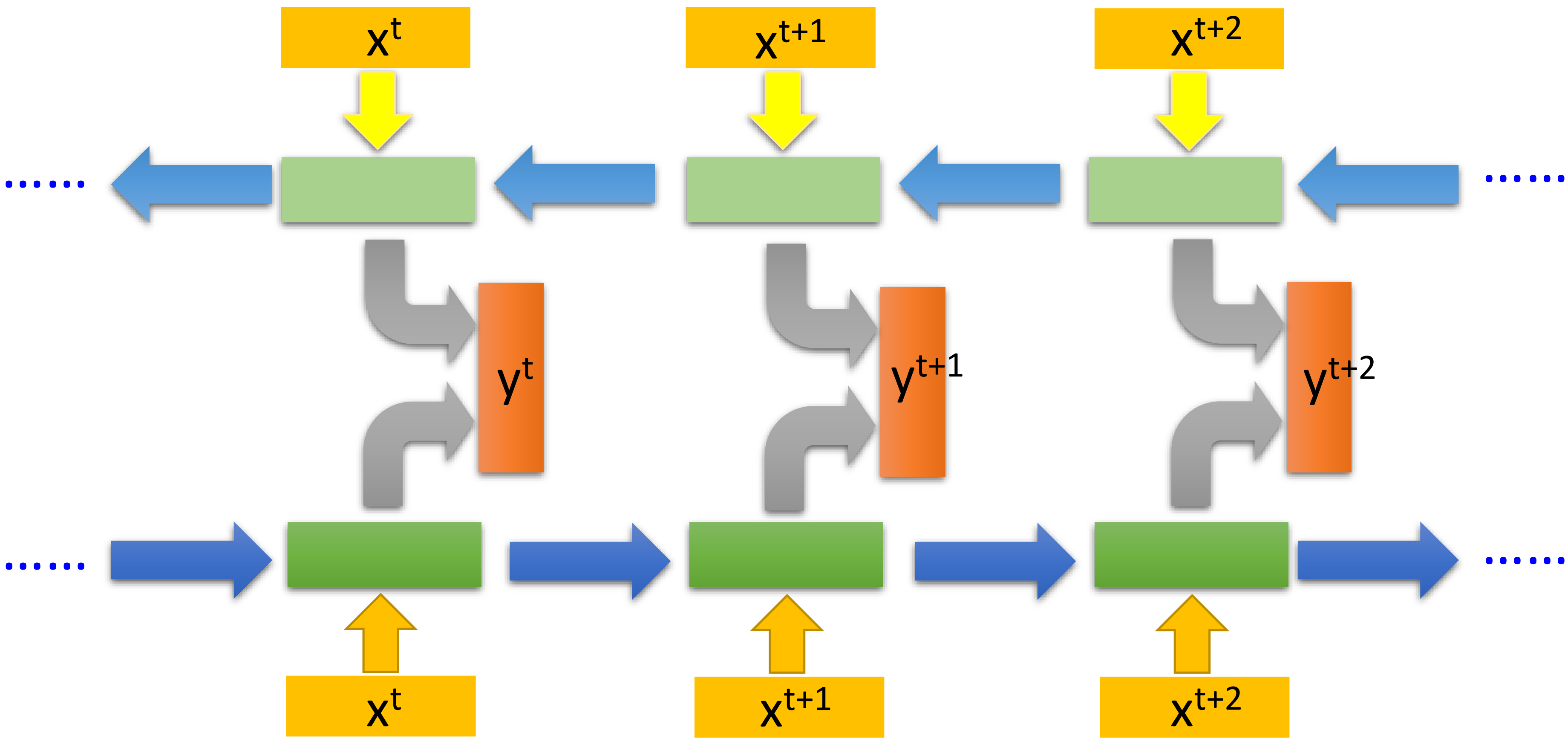
Of course it can be deep …



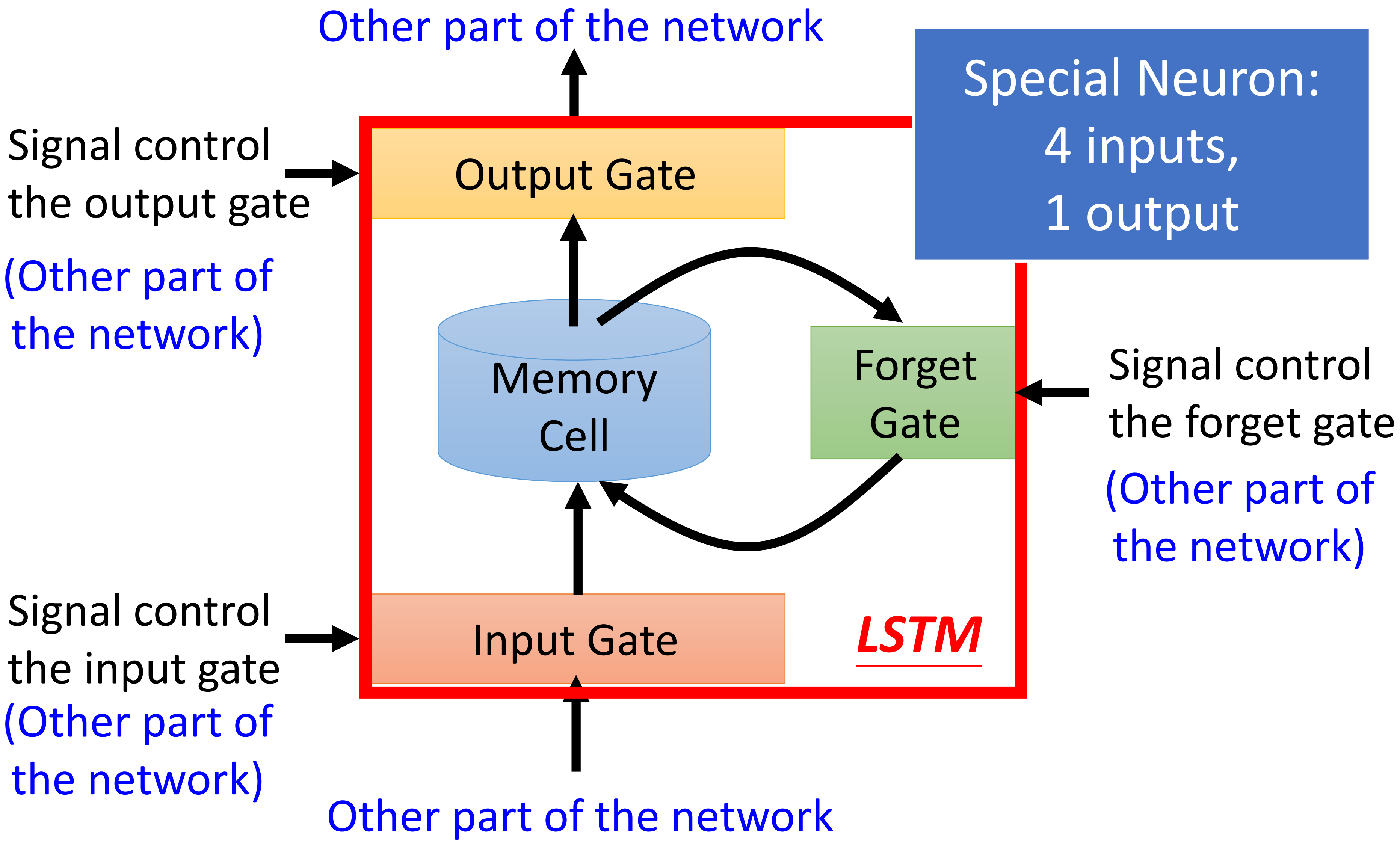
Elman Network & Jordan Network

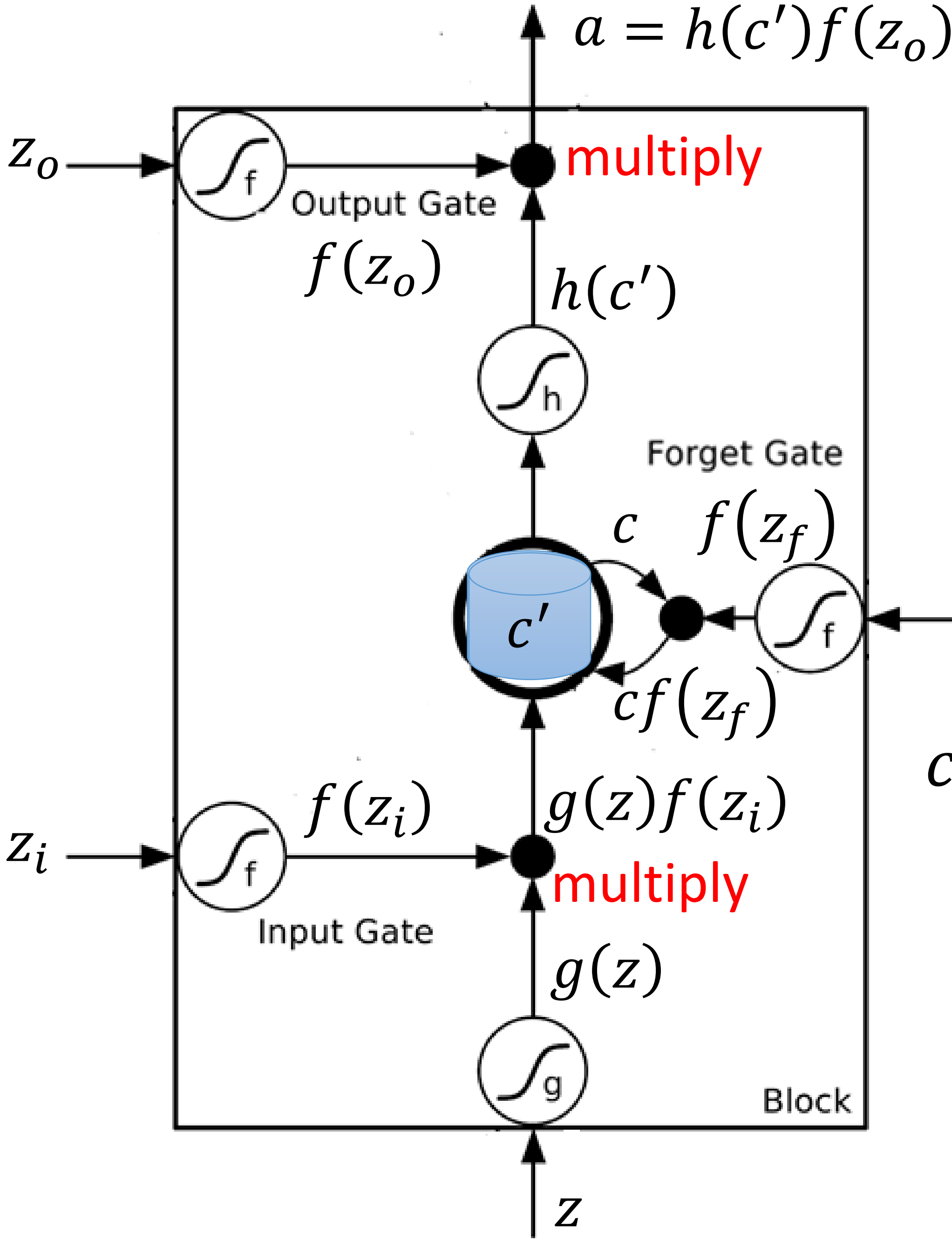


Bidirectional RNN



Long Short-term Memory (LSTM)





Activation function f is usually a sigmoid function

Between 0 and 1

Mimic open and close gate

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

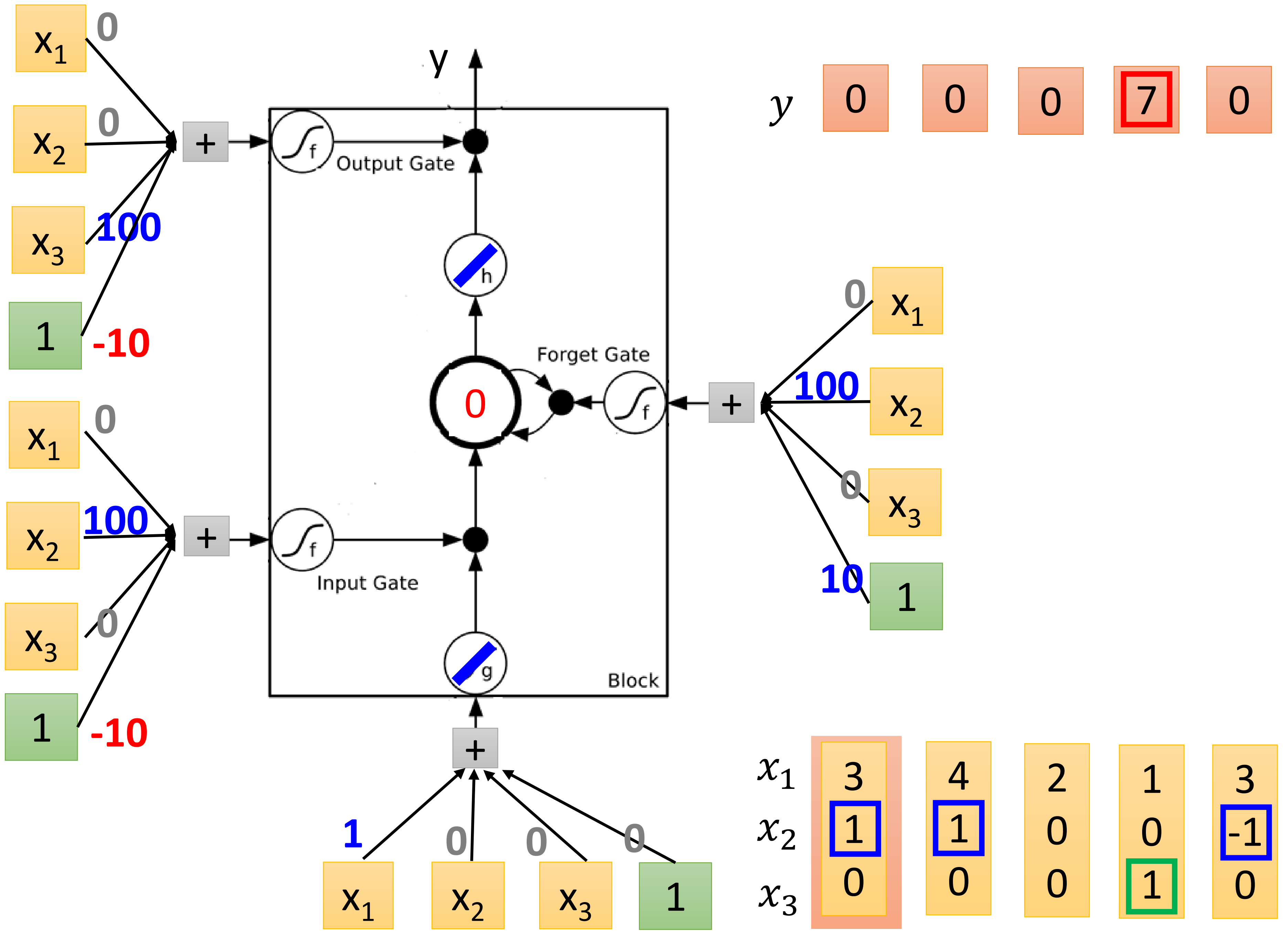
LSTM - Example

	0	0	3	3	7	7	7	0	6
x_1	1	3	2	4	2	1	3	6	1
x_2	0	$\boxed{1}$	0	$\boxed{1}$	0	0	$\boxed{-1}$	$\boxed{1}$	0
x_3	0	0	0	0	0	$\boxed{1}$	0	0	$\boxed{1}$
y	0	0	0	0	0	$\boxed{7}$	0	0	$\boxed{6}$

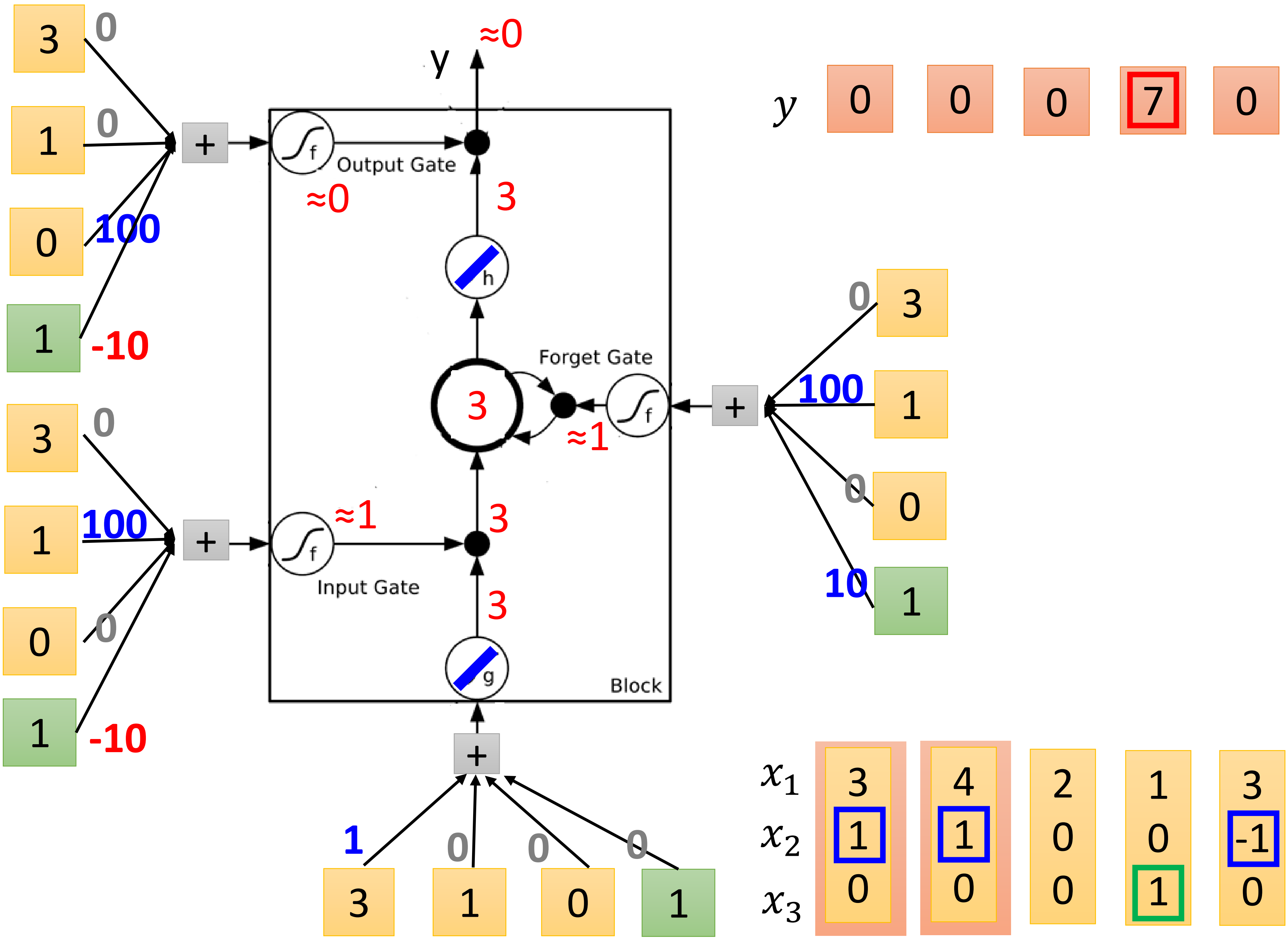
When x_2 = 1, add the numbers of x_1 into the memory

When x_2 = -1, reset the memory

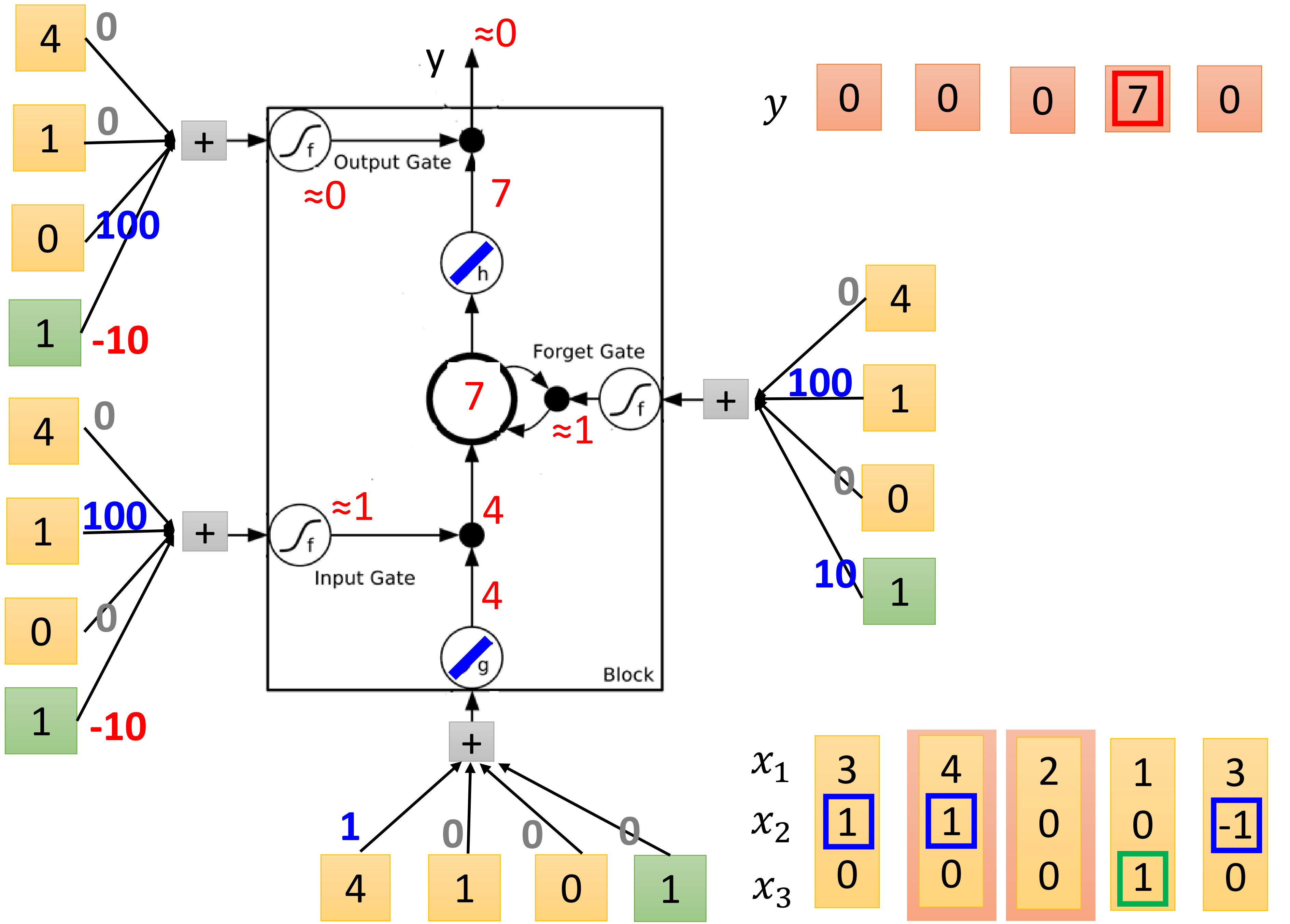
When x_3 = 1, output the number in the memory.



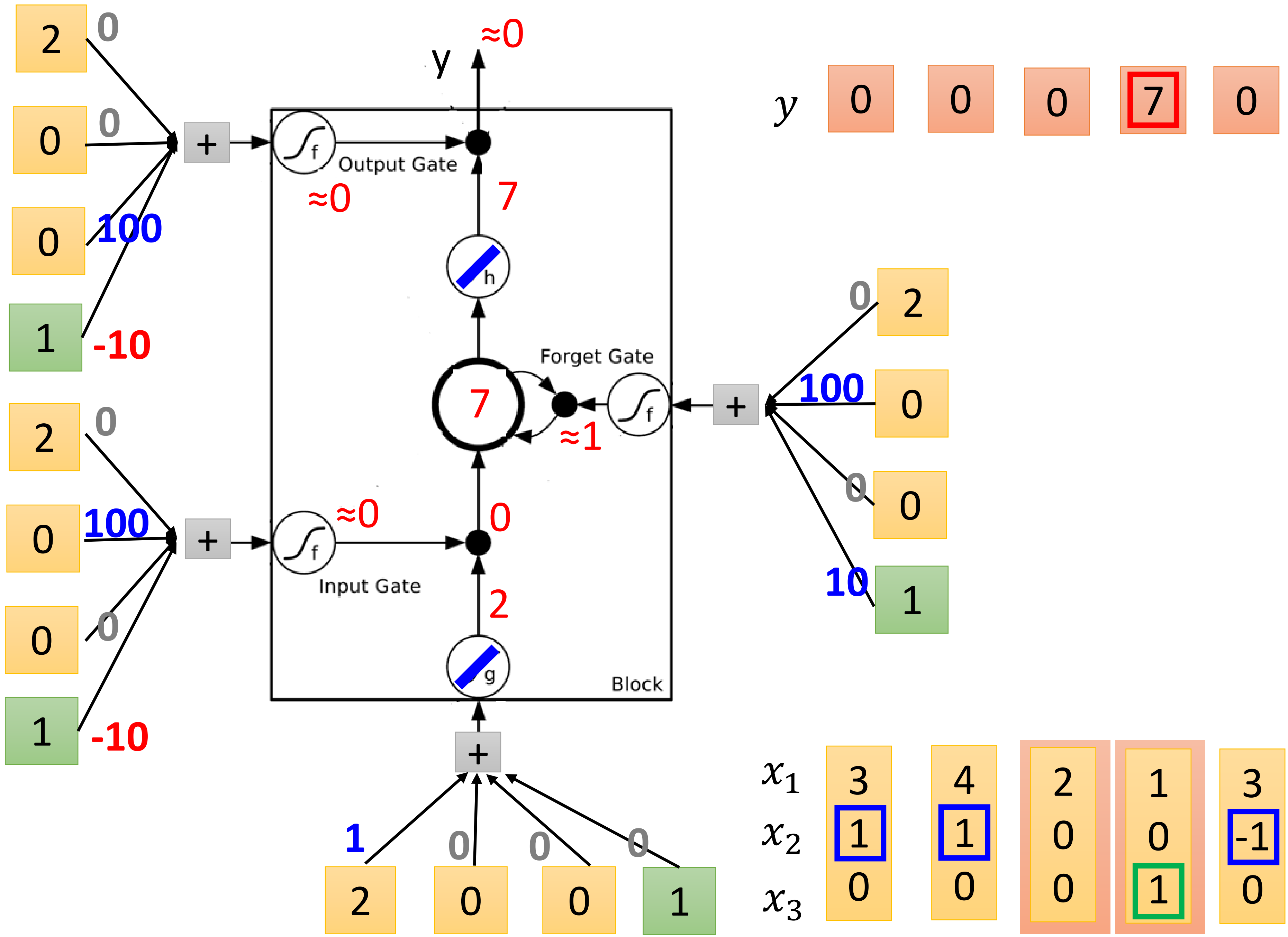
y
0 0 0 7 0
Output Gate
Forget Gate
Input Gate
Block
x1 0
x2 0
x3 100
1 -10
x1 0
x2 100
x3 0
1 -10
x1 0
x2 100
x3 0
1 10
1 0 0 0
x1 x2 x3 1
0
h
g
f
x1 3 4 2 1 3
x2 1 1 0 0 -1
x3 0 0 0 1 0



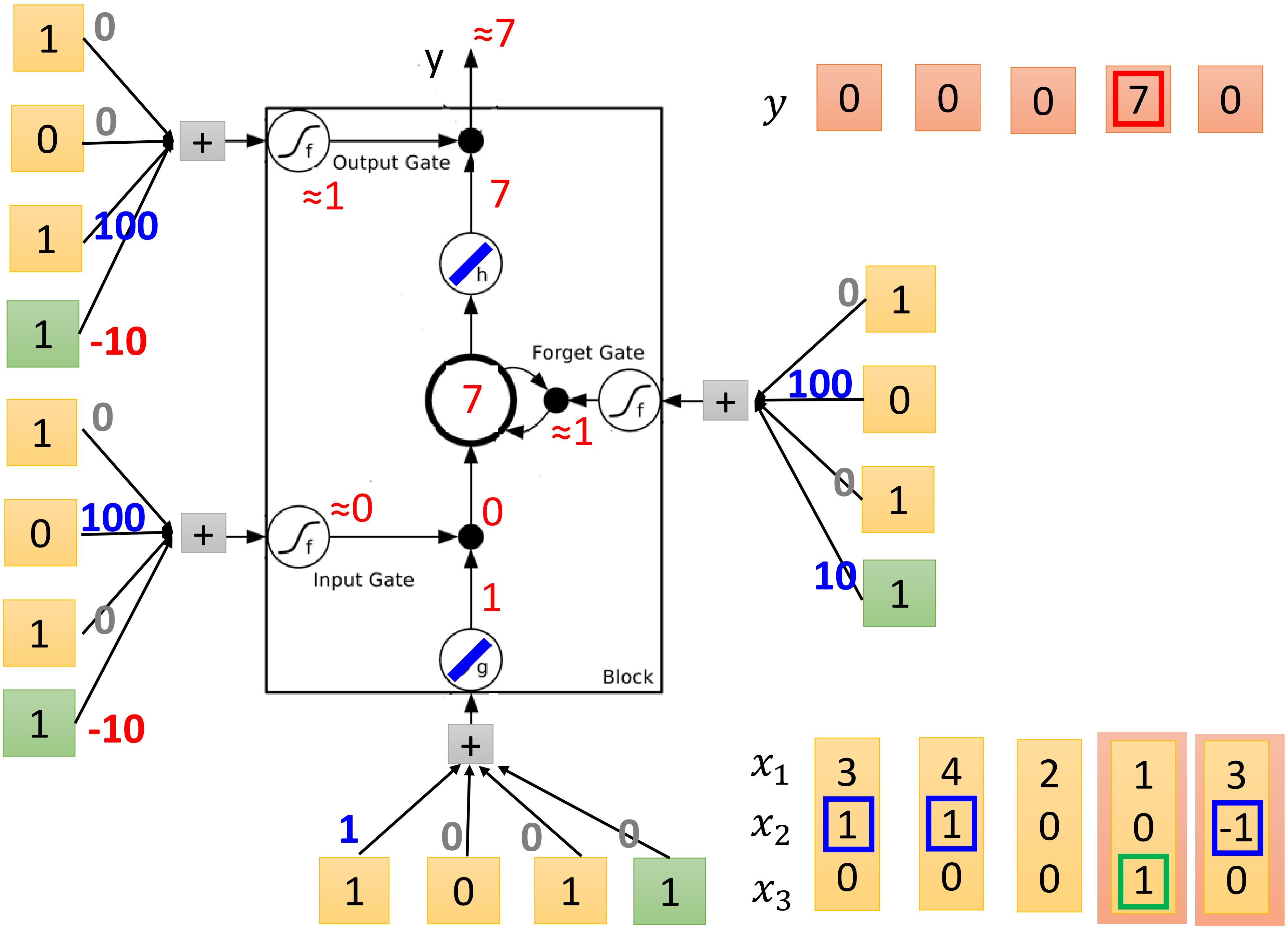
y
≈0
Output Gate
≈0
3
Forget Gate
≈1
≈1
3
3
Input Gate
Block
100
-10
100
-10
100
10
1
y
0 0 0 7 0
x_1 3 4 2 1 3
x_2 1 1 0 0 -1
x_3 0 0 0 1 0



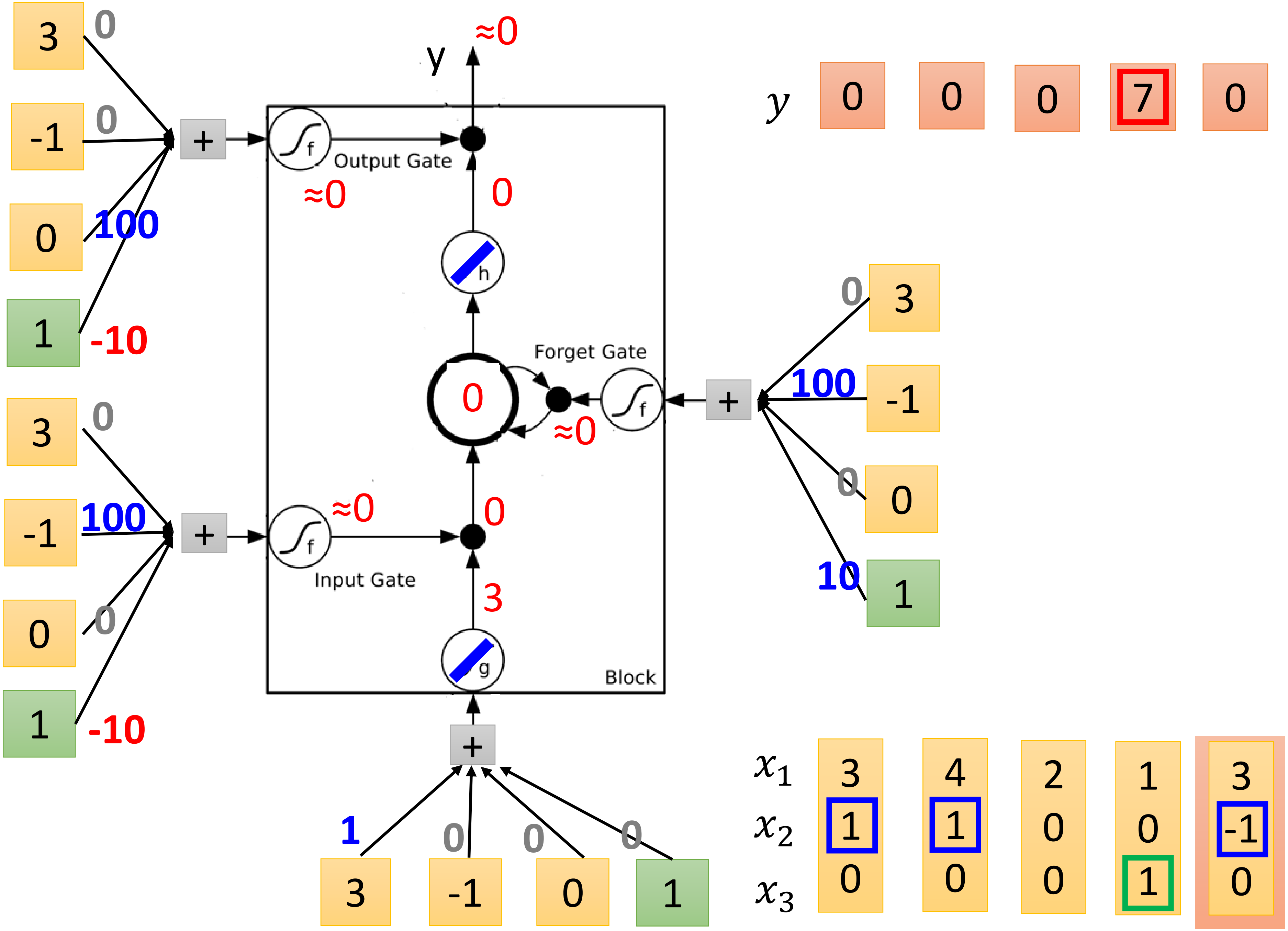
y
≈0
Output Gate
≈0
7
Forget Gate
7
≈1
≈1
4
Input Gate
4
Block
4
1
0
0
1
0
0
100
1
-10
4
0
1
100
0
0
1
-10
0
4
100
1
0
0
10
1
1
4
0
1
0
0
0
1
1
y
0
0
0
7
0
x1
x2
x3
3
1
0
4
1
0
2
0
0
1
0
1
3
-1
0



y
≈0
Output Gate
≈0
7
h
Forget Gate
7
≈1
f
Input Gate
≈0
0
2
g
Block
2 0
0 0
0 100
1 -10
2 0
0 100
0 0
1 -10
0 2
100 0
0 0
10 1
1 0 0 0
2 0 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



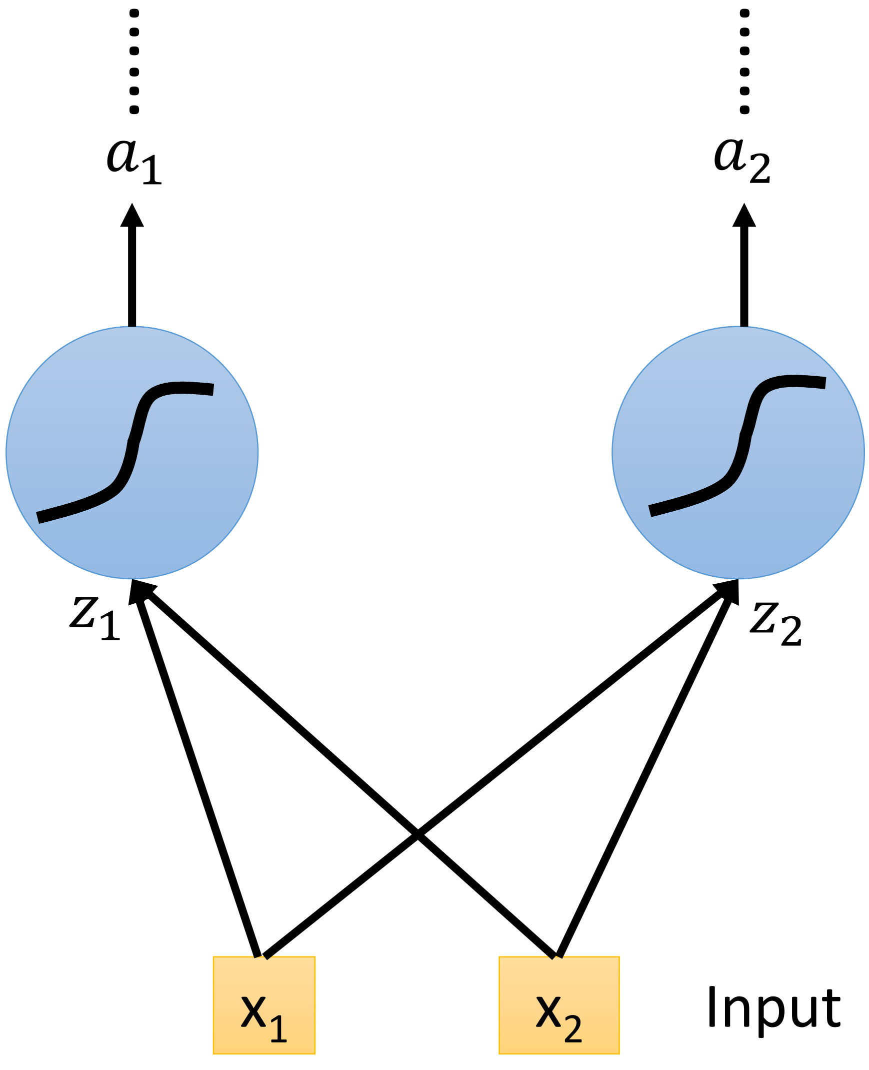
y
≈7
Output Gate
≈1
7
Forget Gate
≈1
≈0
0
1
Input Gate
Block
100
-10
10
y
0 0 0 7 0
x_1
3 4 2 1 3
x_2
1 1 0 0 -1
x_3
0 0 0 1 0

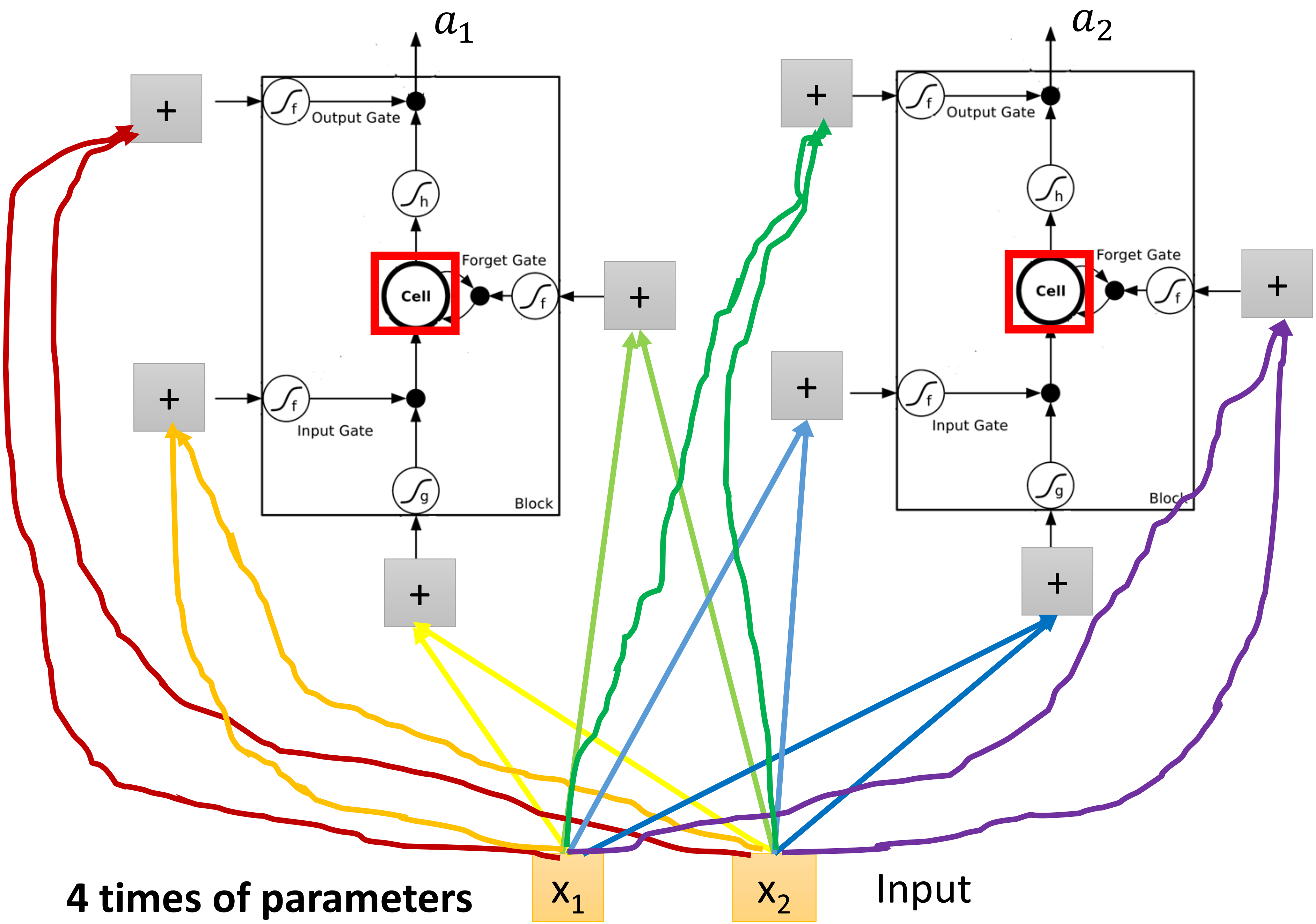


y
≈0
Output Gate
Forget Gate
Input Gate
Block
f
h
g
100
-10
10
1
y 0 0 0 7 0
x_1 3 4 2 1 3
x_2 1 1 0 0 -1
x_3 0 0 0 1 0

Original Network:

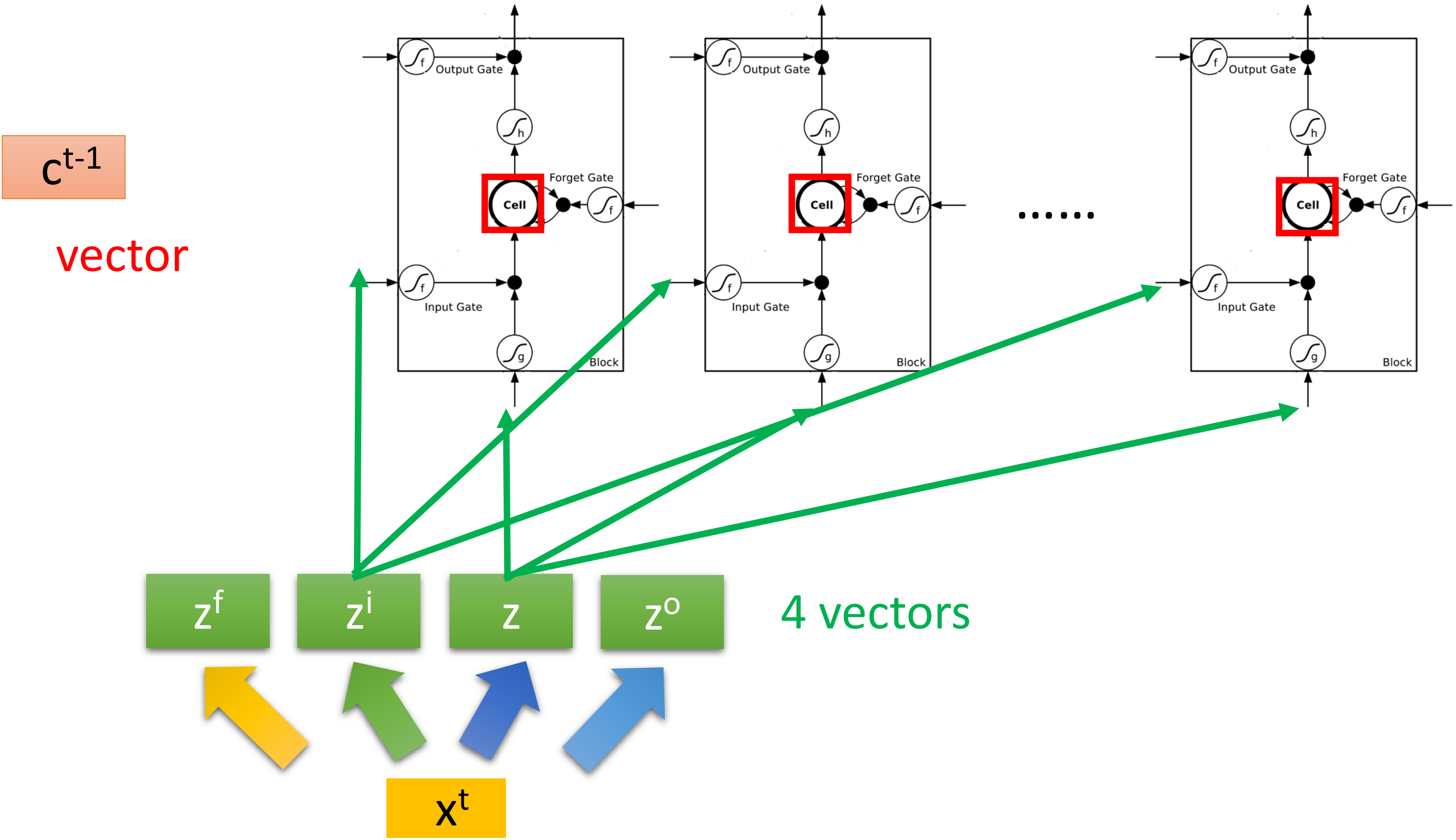
➢ Simply replace the neurons with LSTM





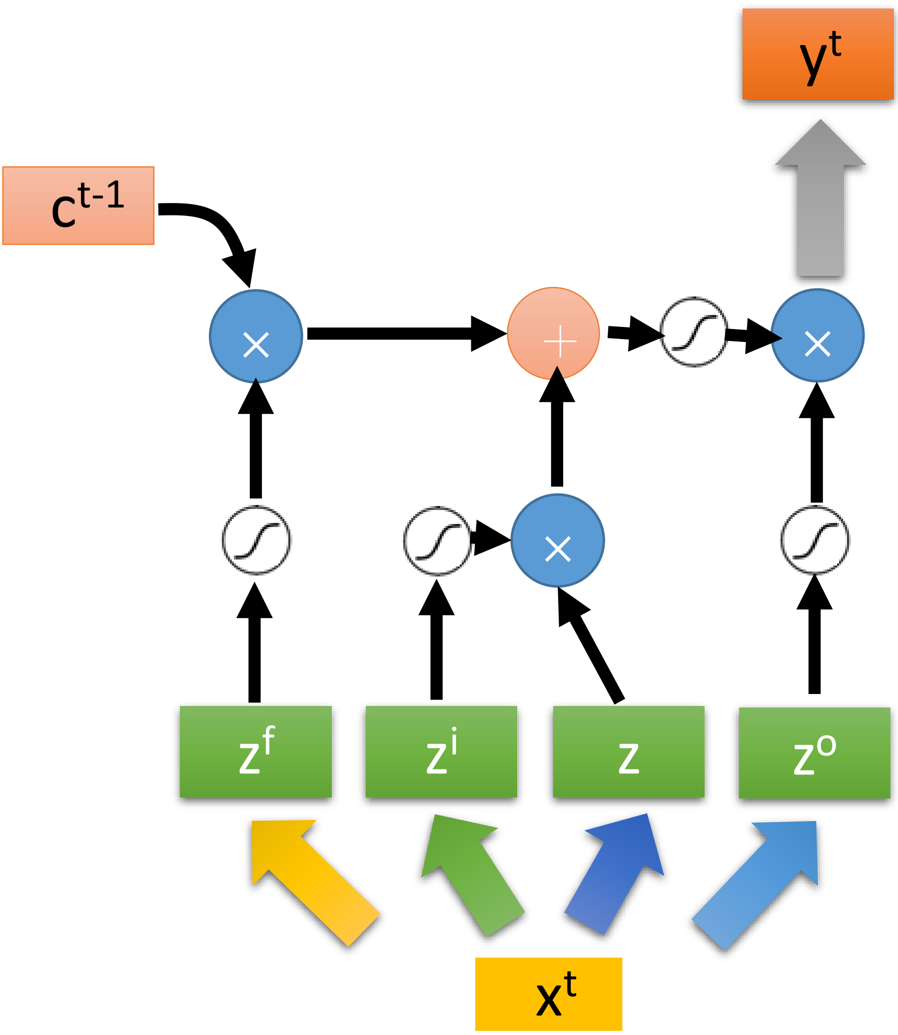
4 times of parameters

LSTM

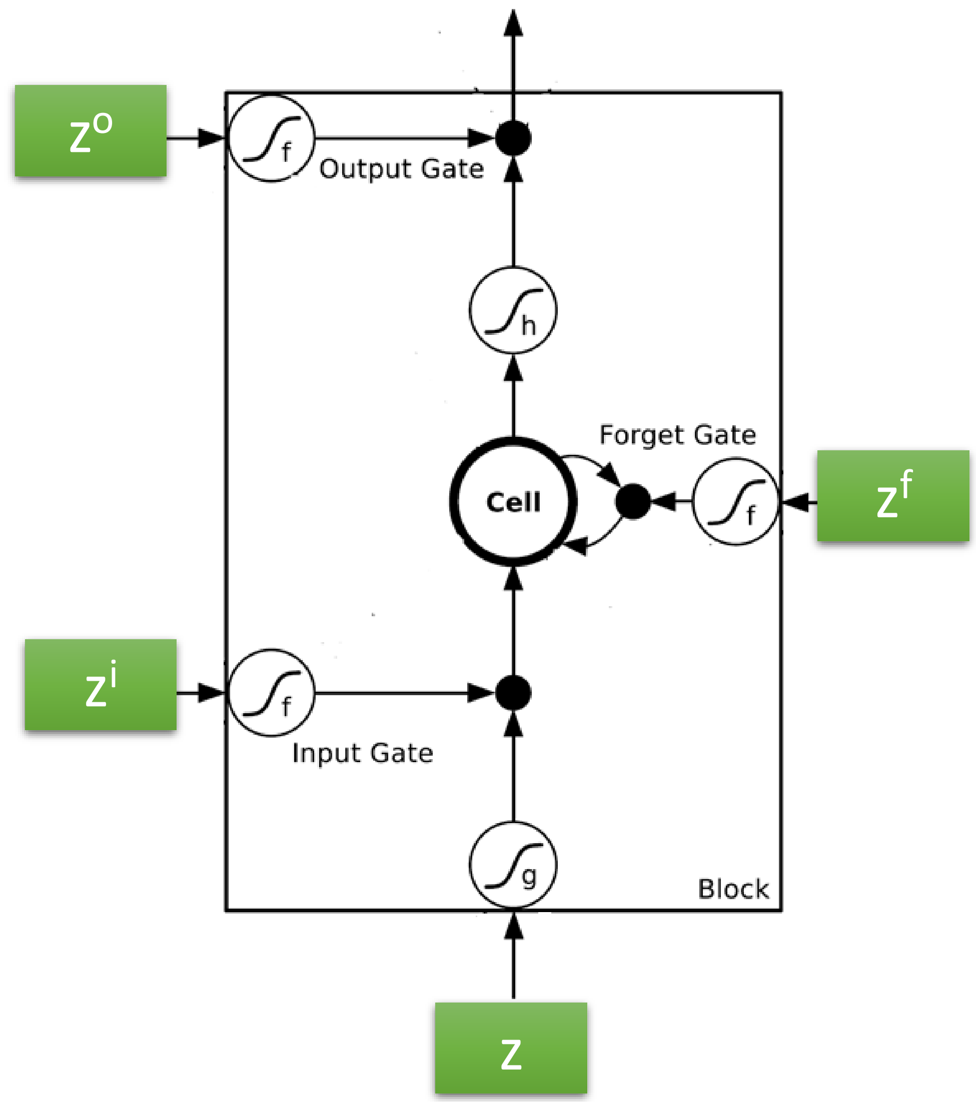


c^t-1
vector
Output Gate
Forget Gate
Cell
Input Gate
Block
......
z^f
z^i
z
z^o
4 vectors
x^t

LSTM



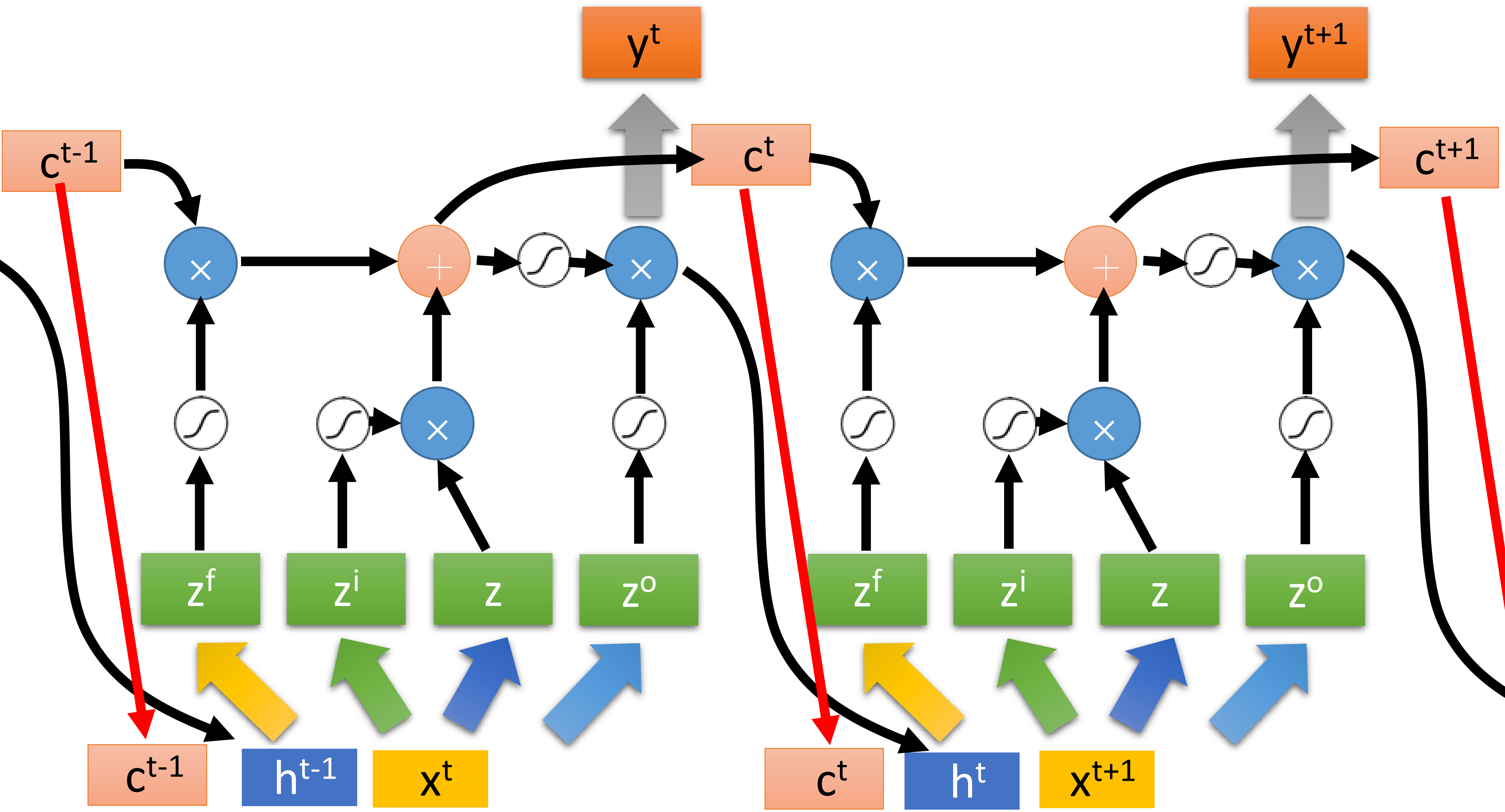
y^t
c^{t-1}
z^f
z^i
z
z^o
x^t



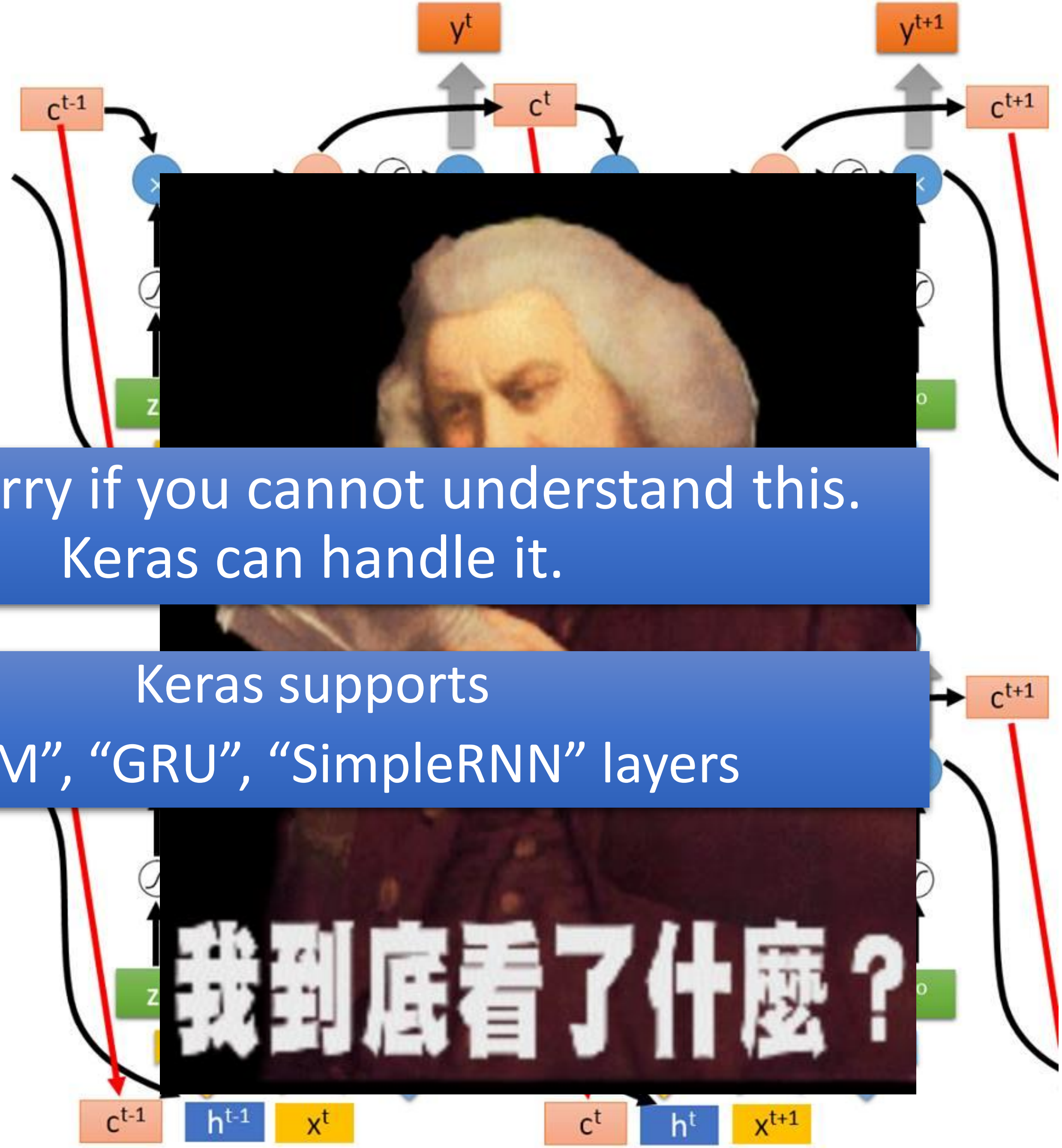
z^o
Output Gate
Forget Gate
Cell
z^f
z^i
Input Gate
Block
z

LSTM

Extension: “peephole”



Multiple-layer LSTM



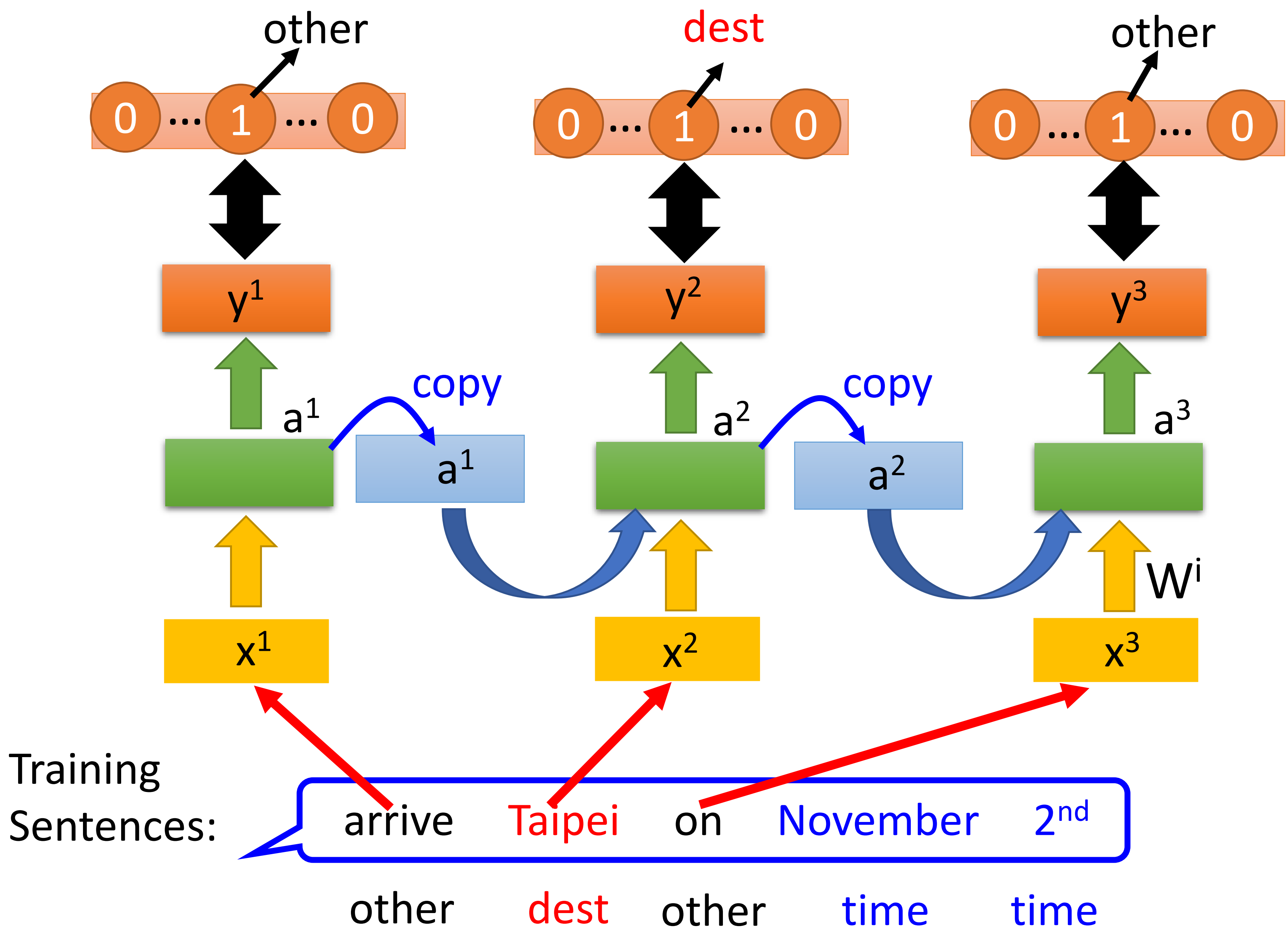
Don’t worry if you cannot understand this.
Keras can handle it.

Keras supports
“LSTM”, “GRU”, “SimpleRNN” layers

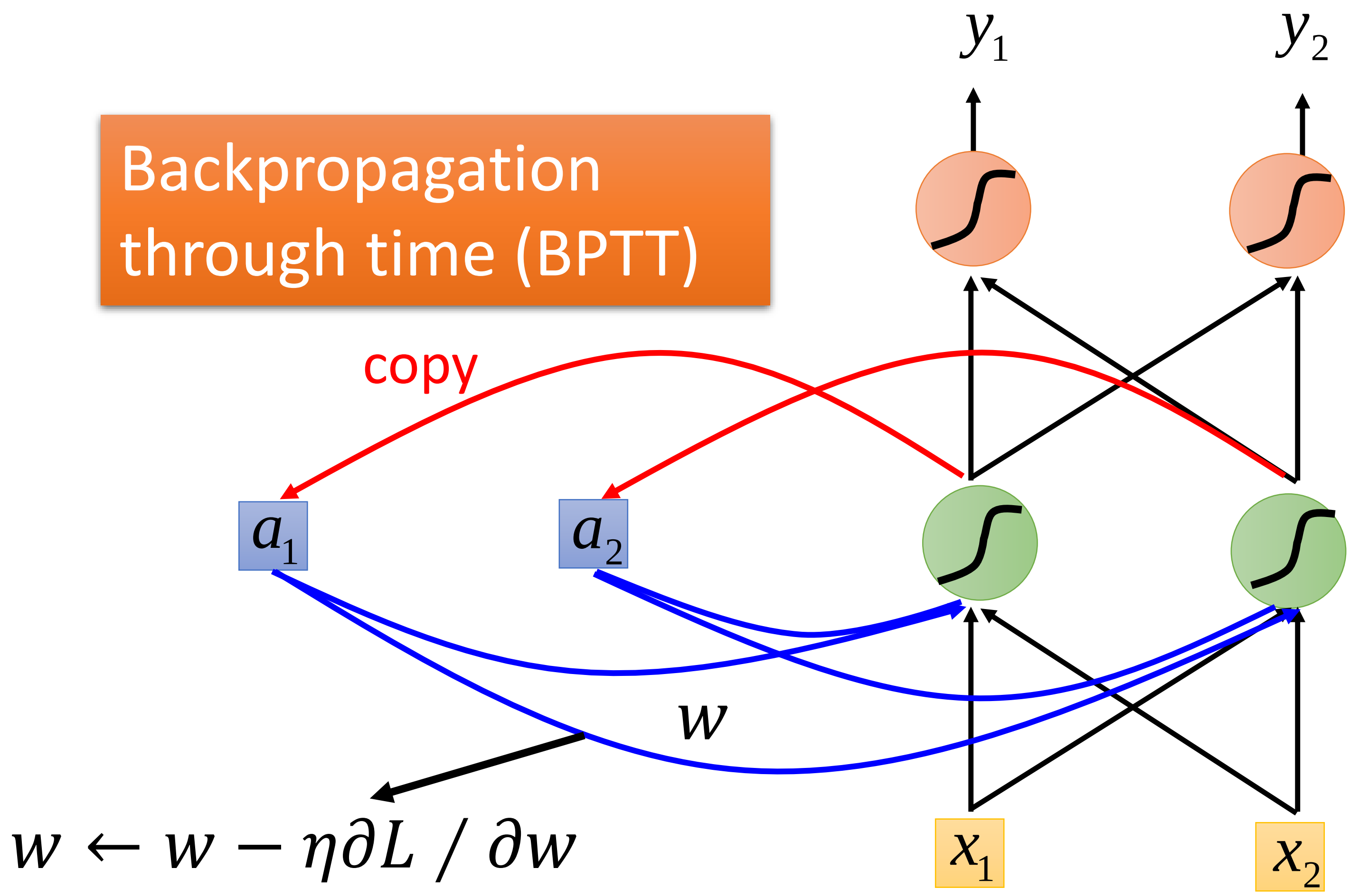
This is quite standard now.

https://img.komicolle.org/2015-09-20/src/14426967627131.gif

Learning Target



Learning

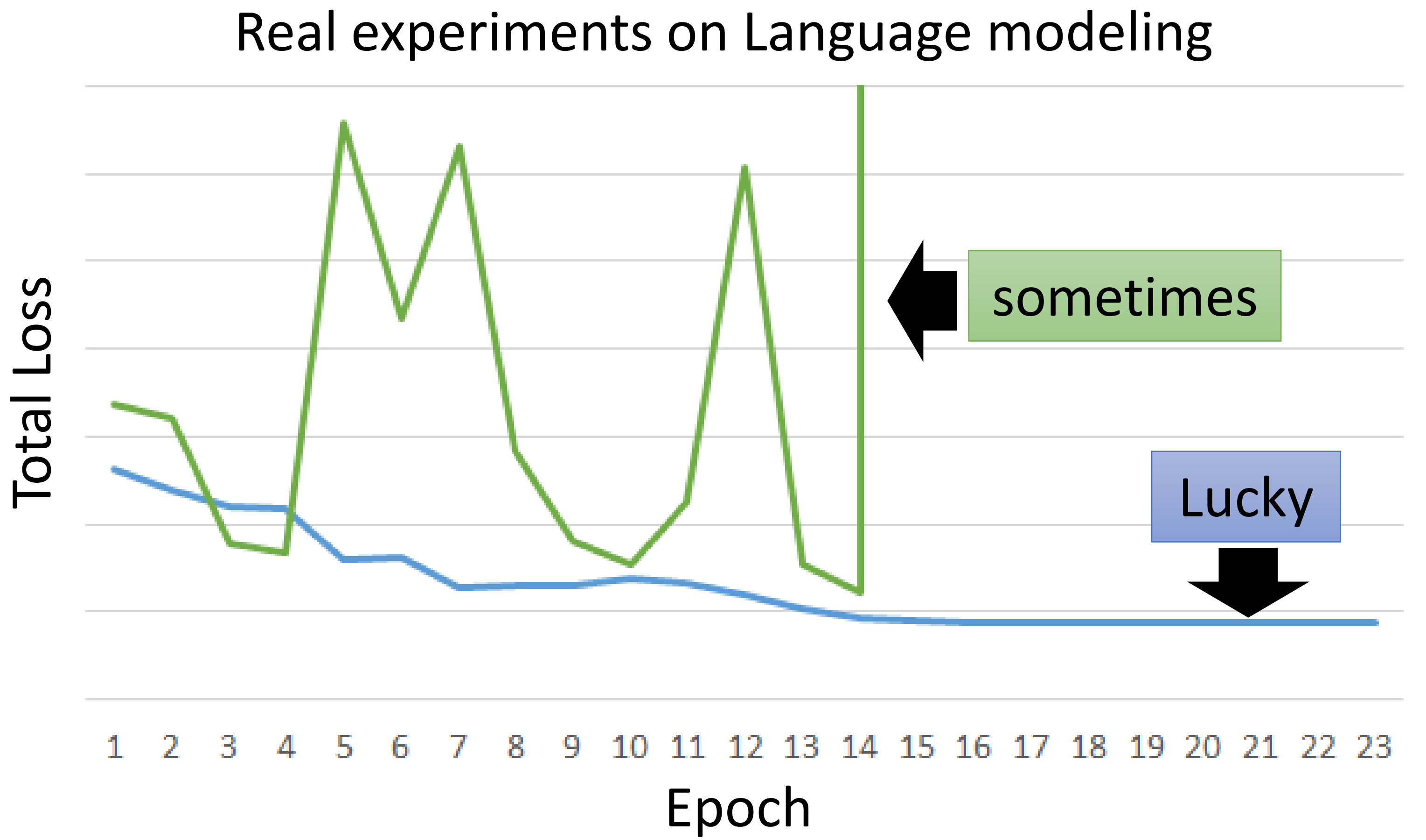


Backpropagation through time (BPTT)
copy
y_1
y_2
a_1
a_2
w
x_1
x_2
$w \leftarrow w - \eta \partial L / \partial w$

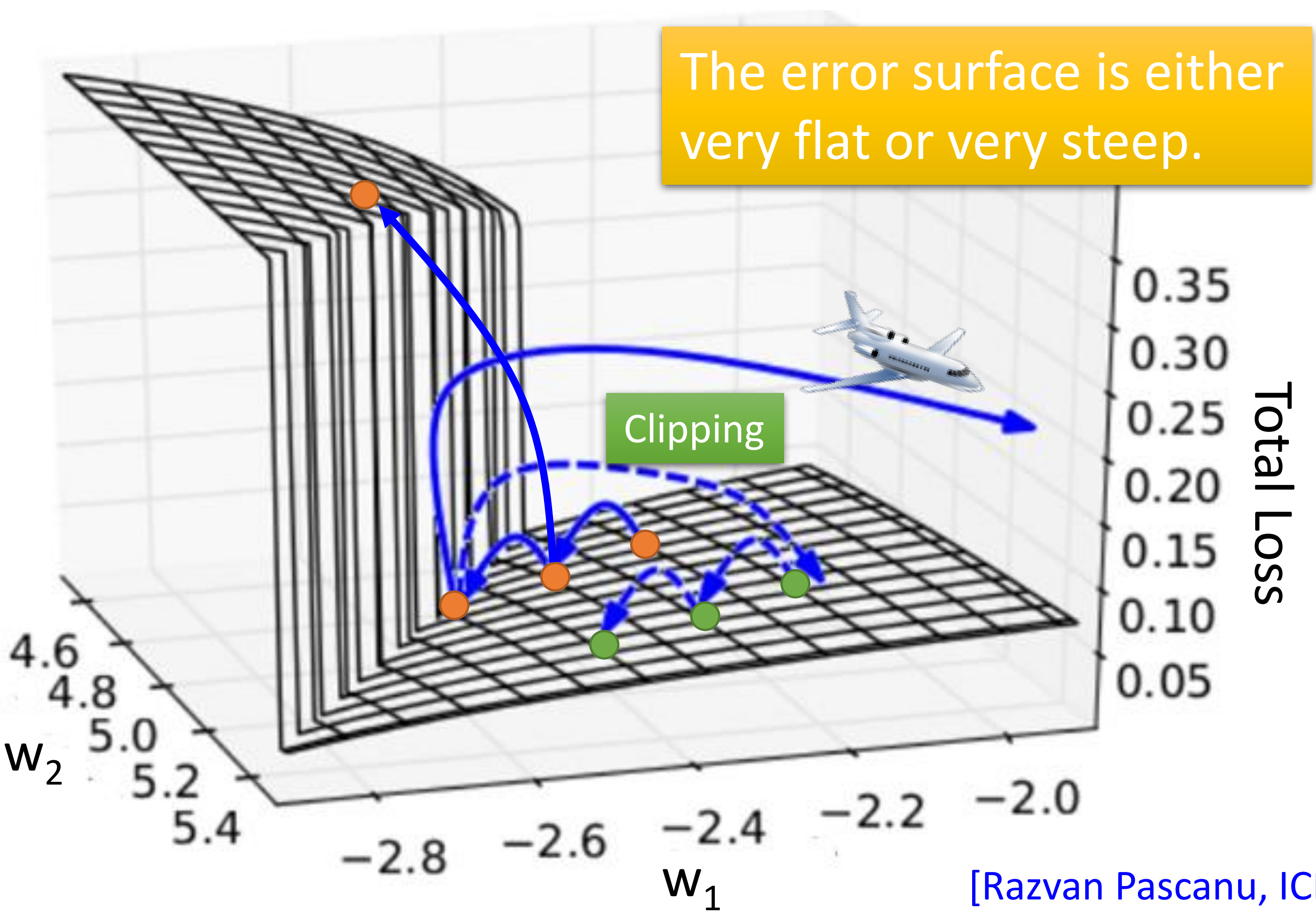
感謝 曾柏翔 同學
提供實驗結果

Unfortunately ……

- RNN-based network is not always easy to learn



The error surface is rough.



[Razvan Pascanu, ICML'13]

Why?

$w = 1 \Rightarrow y^{1000} = 1$

$w = 1.01 \Rightarrow y^{1000} \approx 20000$

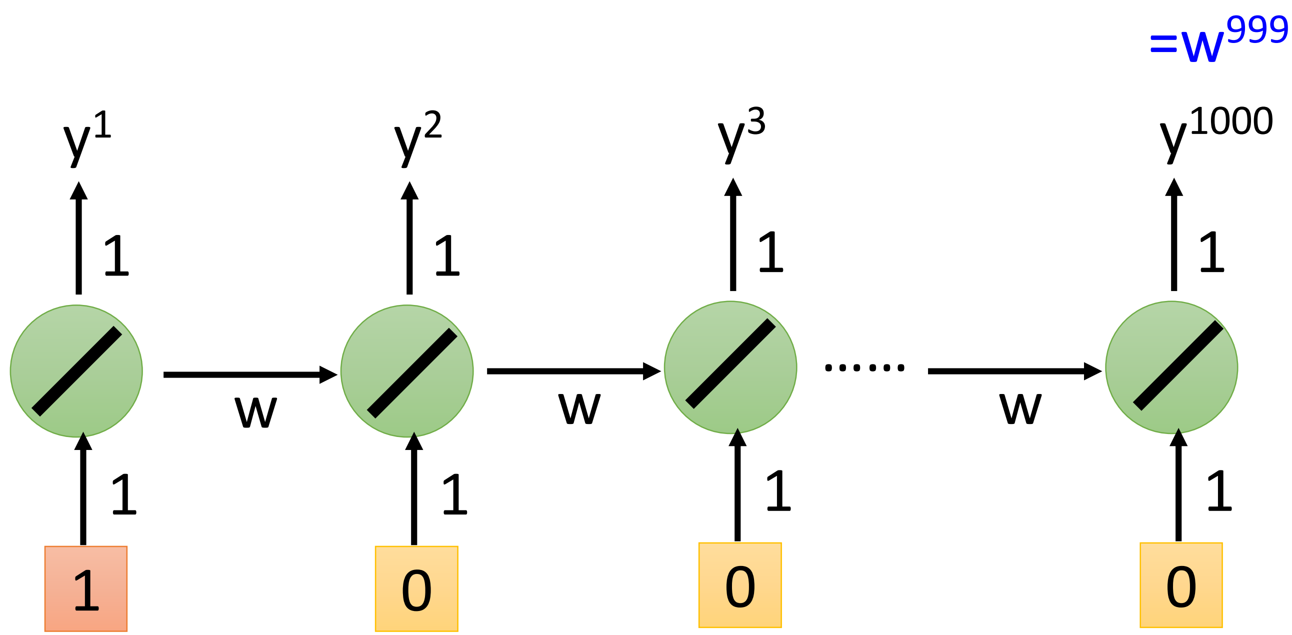
Large $\partial L / \partial w$ → Small Learning rate?

$w = 0.99 \Rightarrow y^{1000} \approx 0$

$w = 0.01 \Rightarrow y^{1000} \approx 0$

small $\partial L / \partial w$ → Large Learning rate?

Toy Example



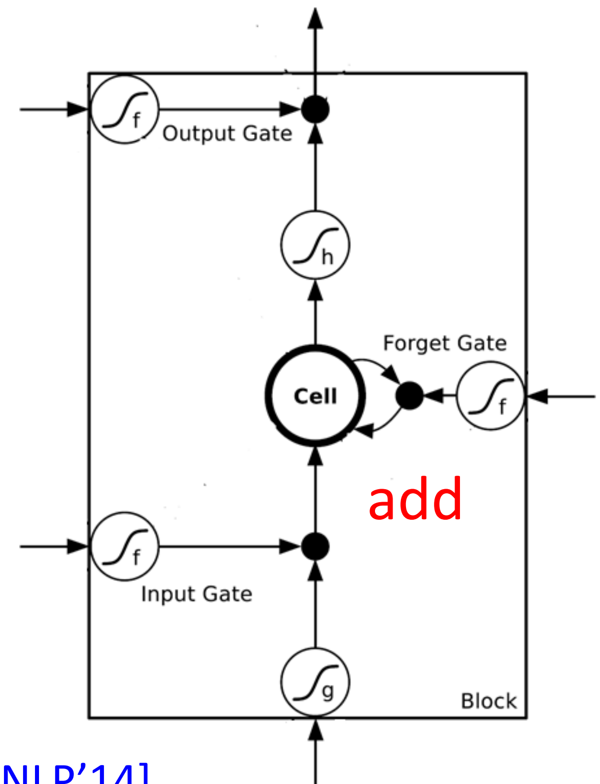
Helpful Techniques

- Long Short-term Memory (LSTM)
  - Can deal with gradient vanishing (not gradient explode)

➢ Memory and input are ***added***

➢ The influence never disappears unless forget gate is closed

No Gradient vanishing
(If forget gate is opened.)

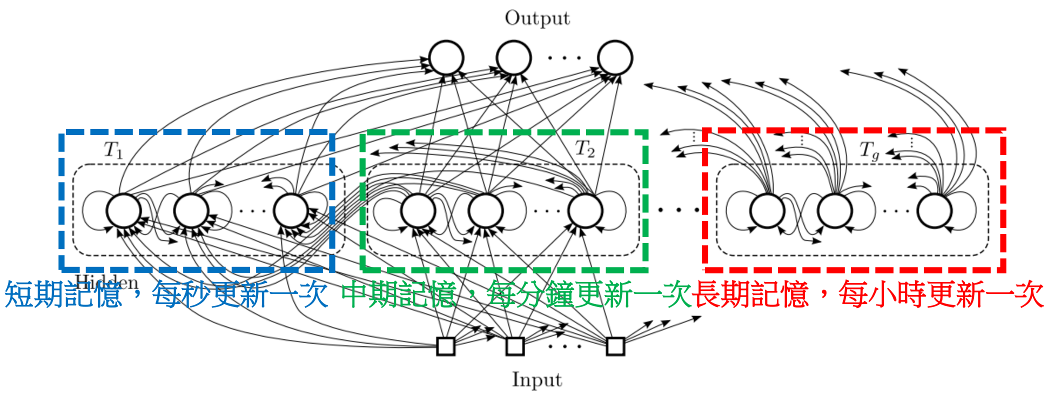


Gated Recurrent Unit (GRU): simpler than LSTM

[Cho, EMNLP'14]

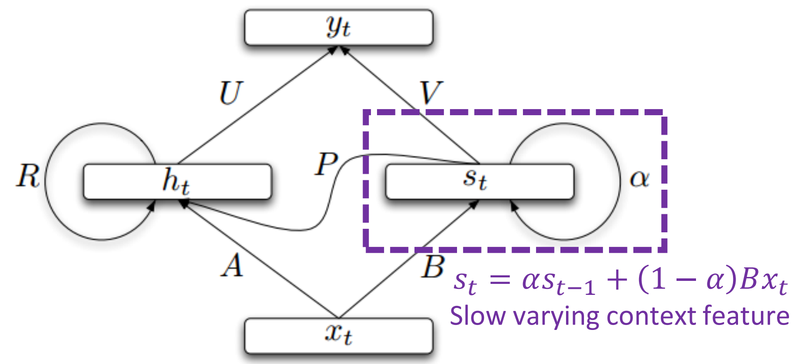
Helpful Techniques

Clockwise RNN



[Jan Koutnik, JMLR’14]

Structurally Constrained Recurrent Network (SCRN)



[Tomas Mikolov, ICLR’15]

Vanilla RNN Initialized with Identity matrix + ReLU activation function [Quoc V. Le, arXiv’15]

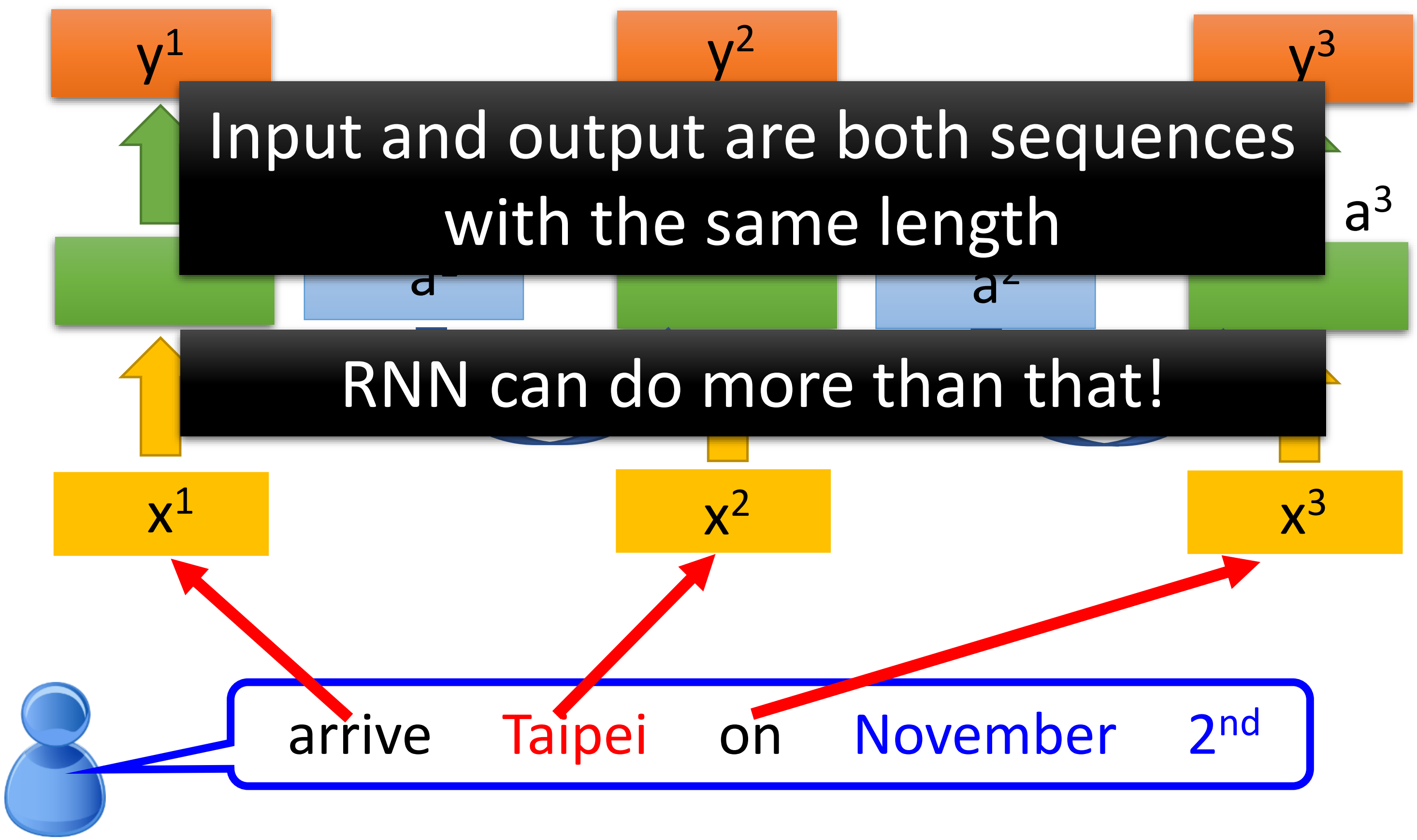
- ➢ Outperform or be comparable with LSTM in 4 different tasks

More Applications ……

Probability of “arrive” in each slot

Probability of “Taipei” in each slot

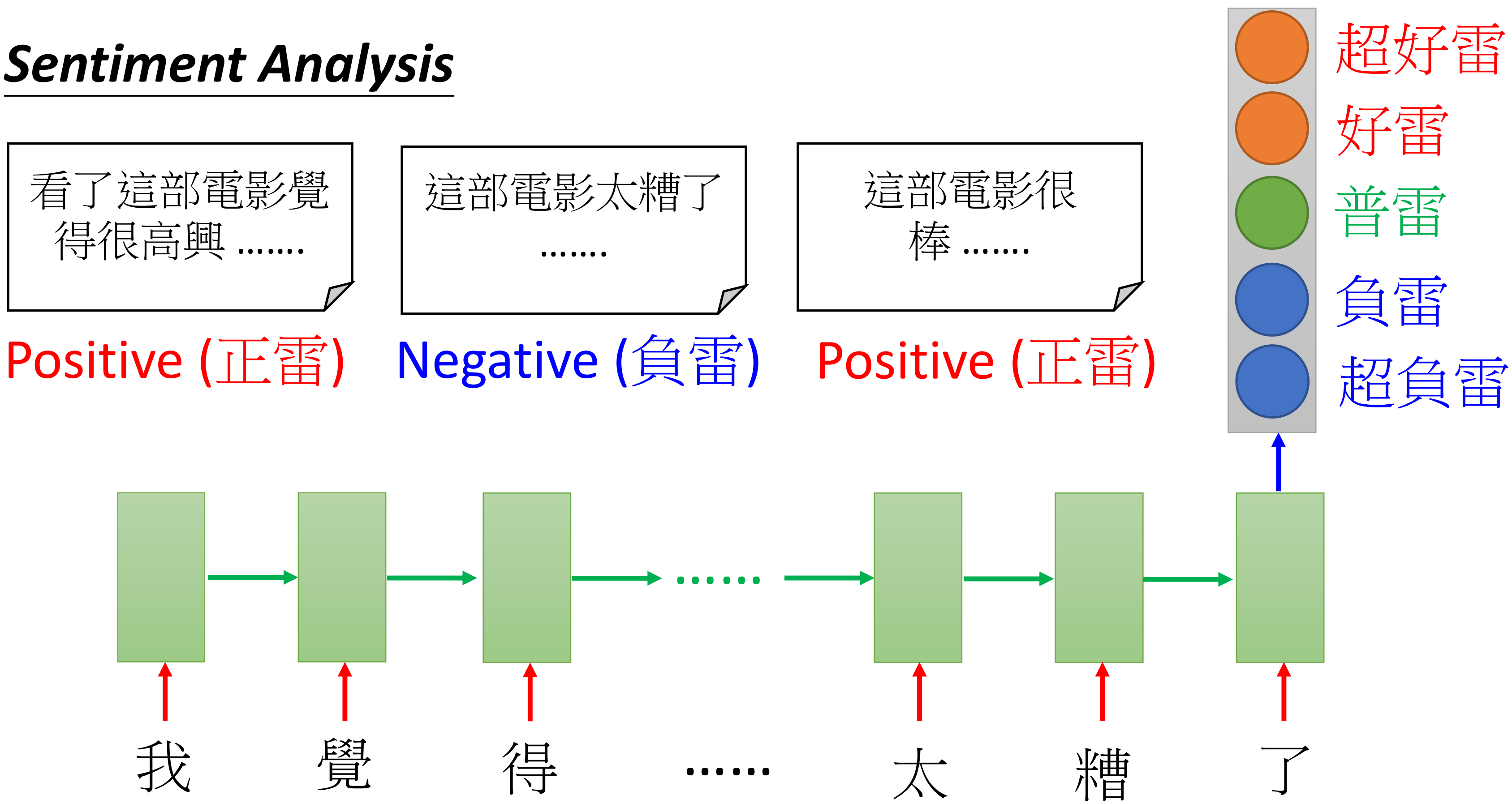
Probability of “on” in each slot



Many to one

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

Sentiment Analysis

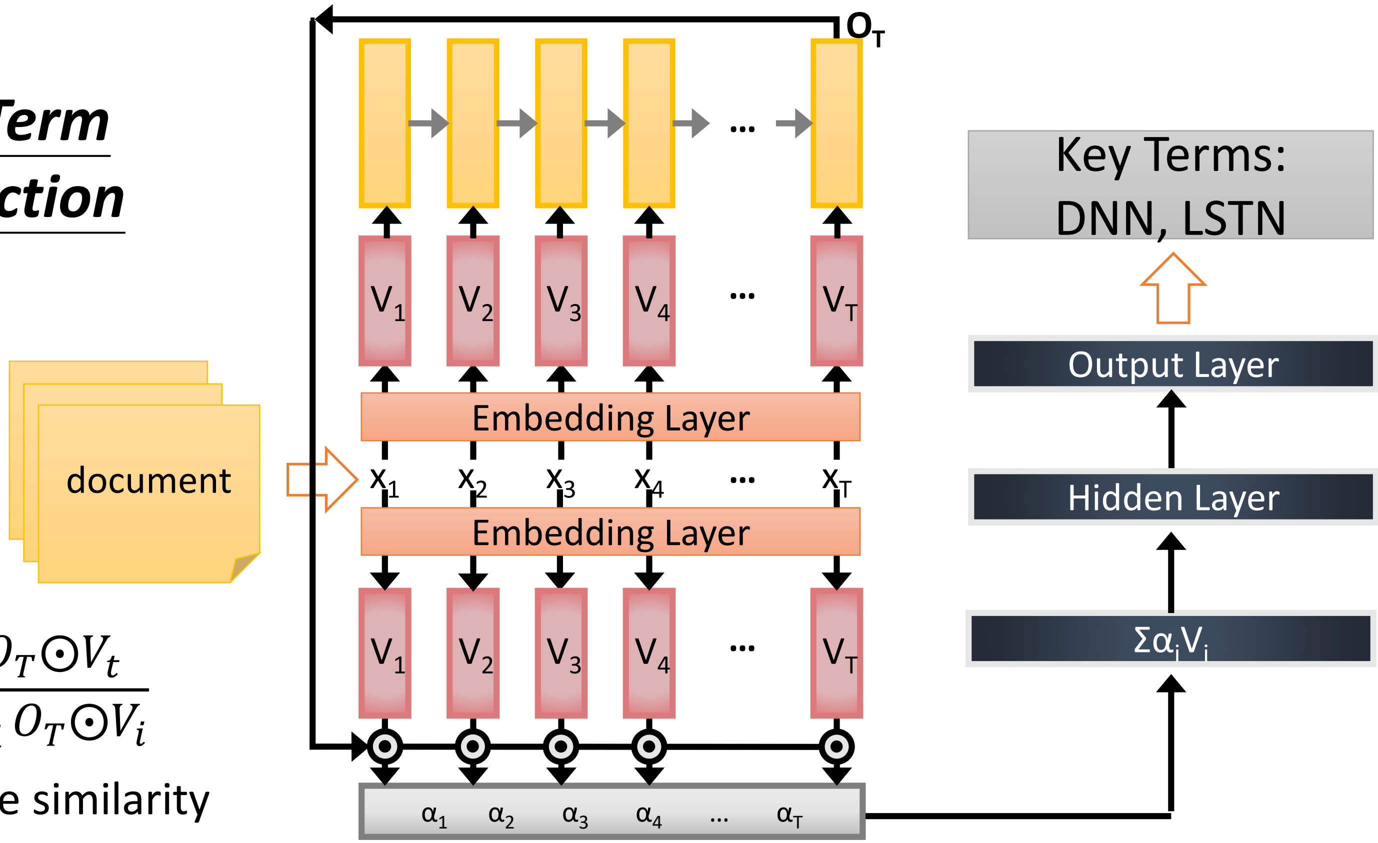


Many to one

[Shen & Lee, *Interspeech* 16]

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

Key Term Extraction



$$\alpha_t = \frac{O_T \odot V_t}{\sum_i O_T \odot V_i}$$

$\odot$: cosine similarity

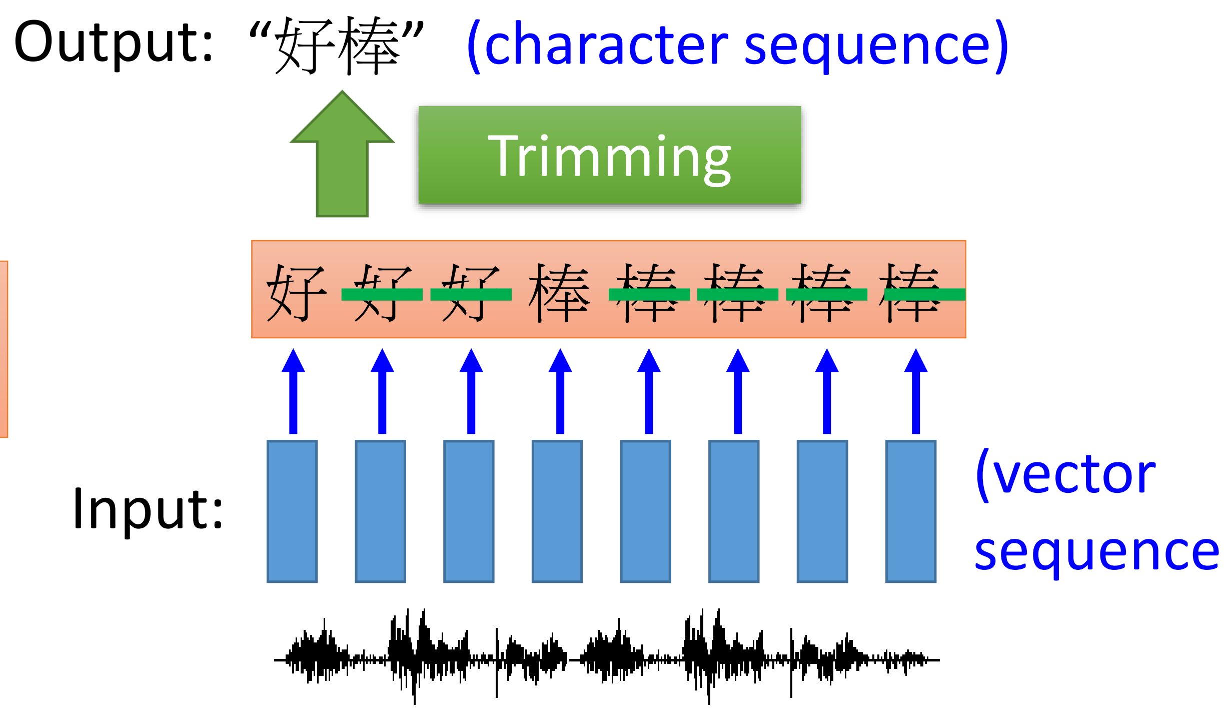
Sheng-syun Shen, Hung-Yi Lee, "Neural Attention Models for Sequence Classification: Analysis and Application to Key Term Extraction and Dialogue Act Detection", the 17th Annual Conference of the International Speech Communication Association (INTERSPEECH'16), San Francisco, Sept. 2016

Many to Many (Output is shorter)

- Both input and output are both sequences, ***but the output is shorter.***
  - E.g. ***Speech Recognition***

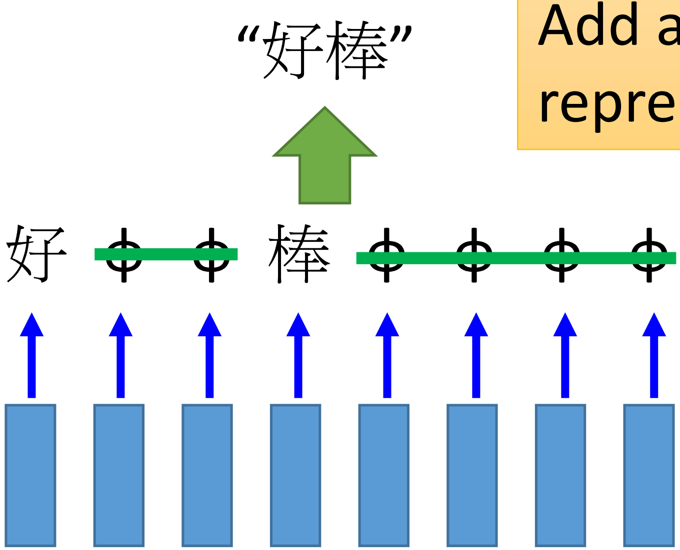
Problem?

Why can’t it be “好棒棒”

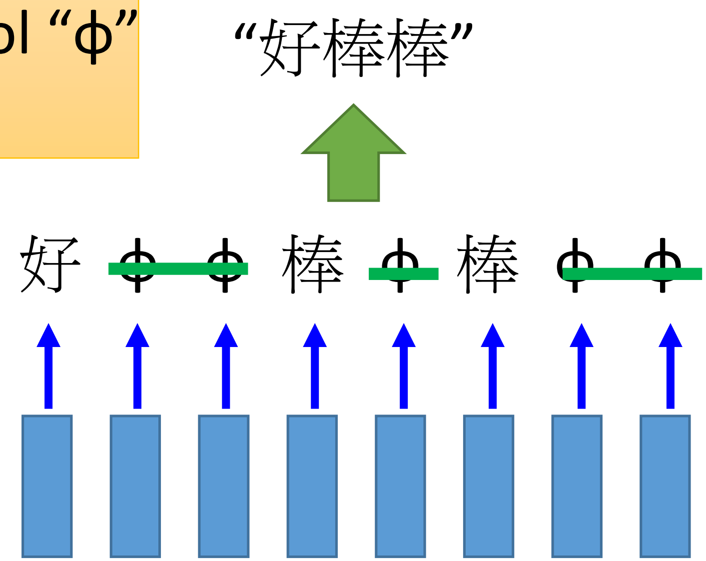


Many to Many (Output is shorter)

- Both input and output are both sequences, ***but the output is shorter.***
- Connectionist Temporal Classification (CTC) [Alex Graves, ICML’06][Alex Graves, ICML’14][Haşim Sak, Interspeech’15][Jie Li, Interspeech’15][Andrew Senior, ASRU’15]



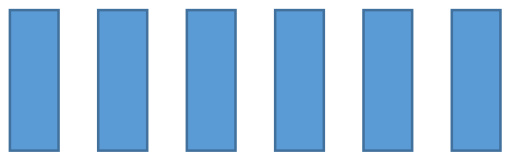
Add an extra symbol “φ” representing “null”



Many to Many (Output is shorter)

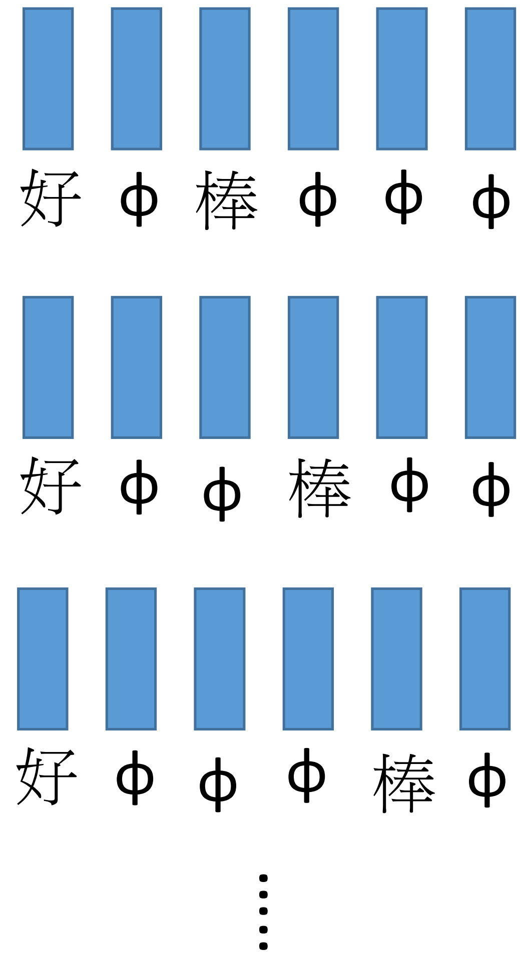
- CTC: Training

Acoustic Features:



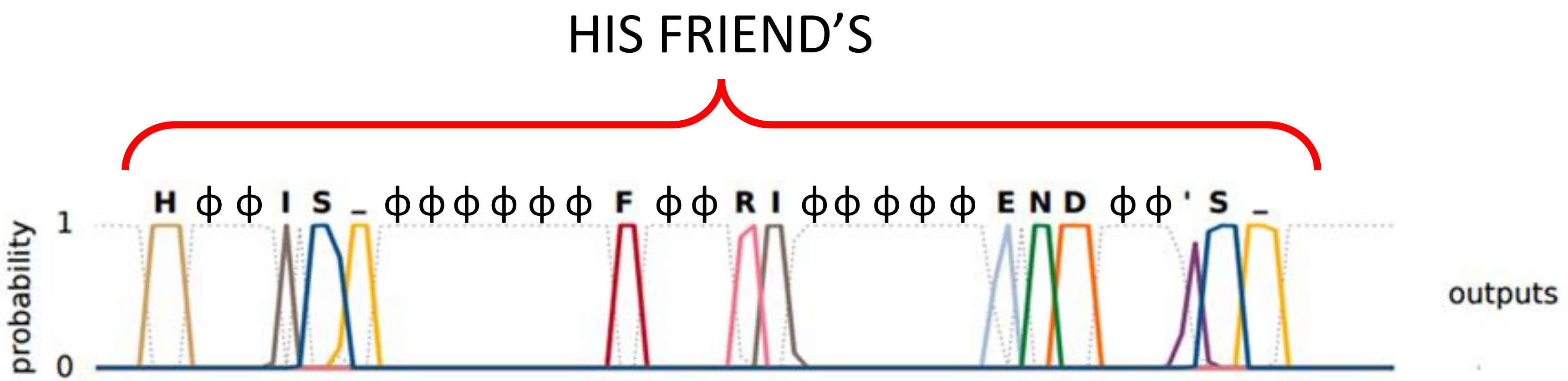
Label: 好 棒

All possible alignments are considered as correct.



Many to Many (Output is shorter)

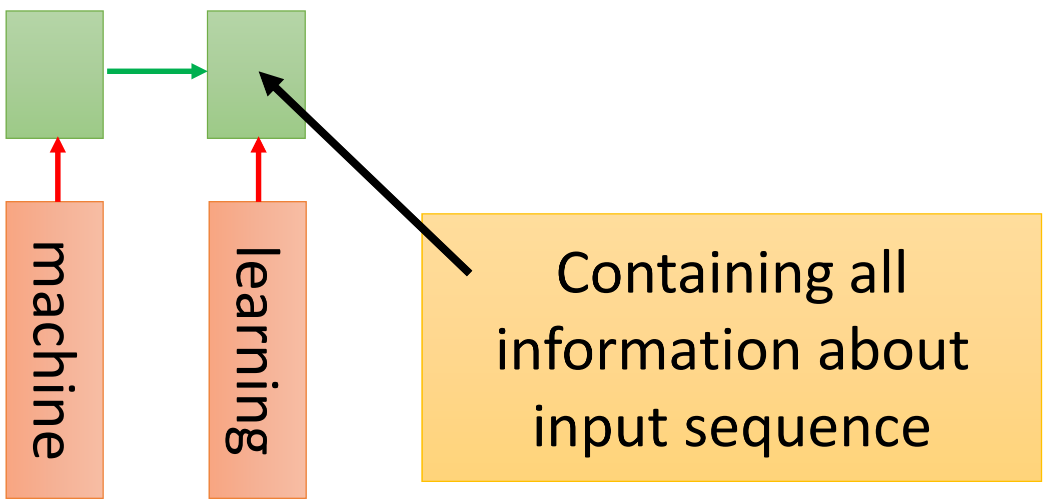
- CTC: example



Graves, Alex, and Navdeep Jaitly. "Towards end-to-end speech recognition with recurrent neural networks." *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*. 2014.

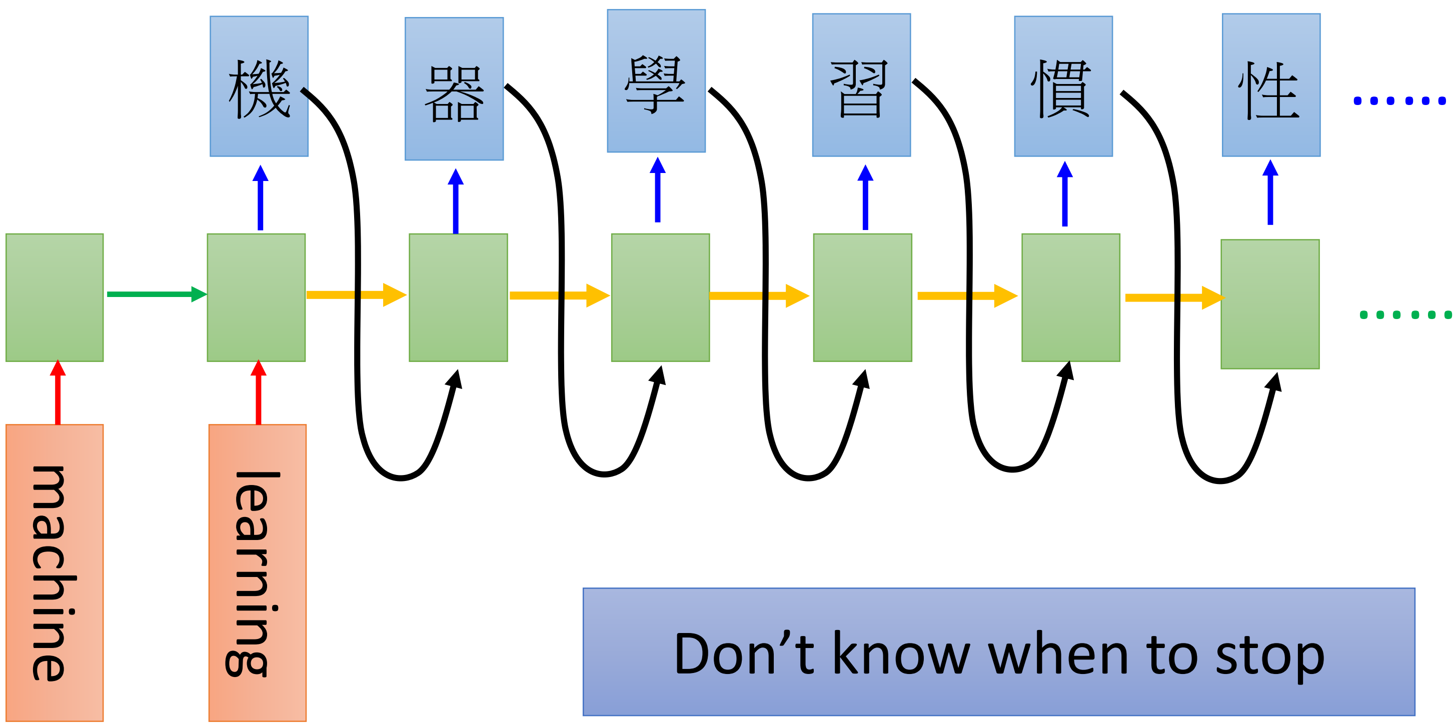
Many to Many (No Limitation)

- Both input and output are both sequences ***with different lengths***. → ***Sequence to sequence learning***
  - E.g. ***Machine Translation*** (machine learning→機器學習)

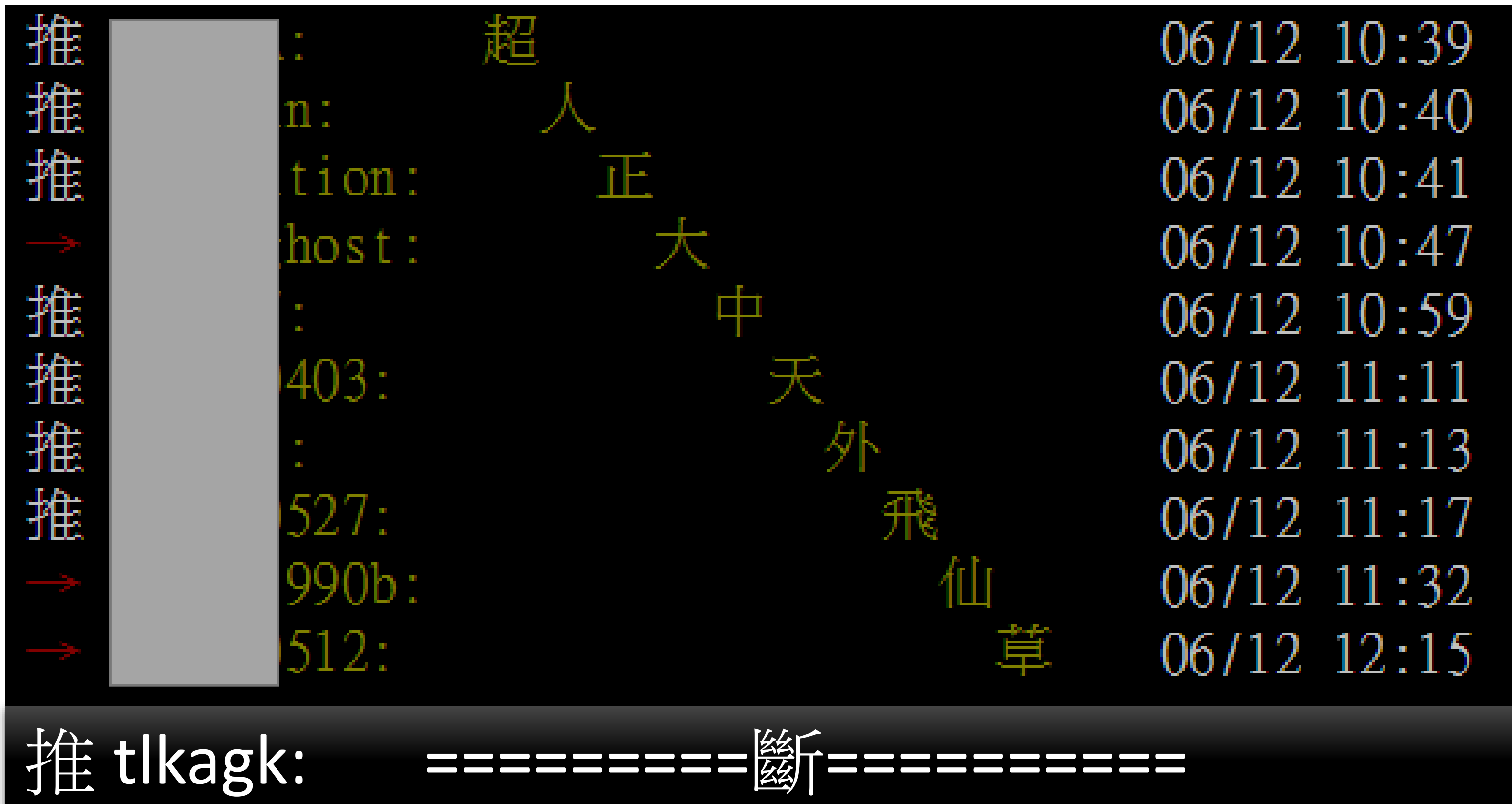


Many to Many (No Limitation)

- Both input and output are both sequences ***with different lengths***. → ***Sequence to sequence learning***
  - E.g. ***Machine Translation*** (machine learning→機器學習)



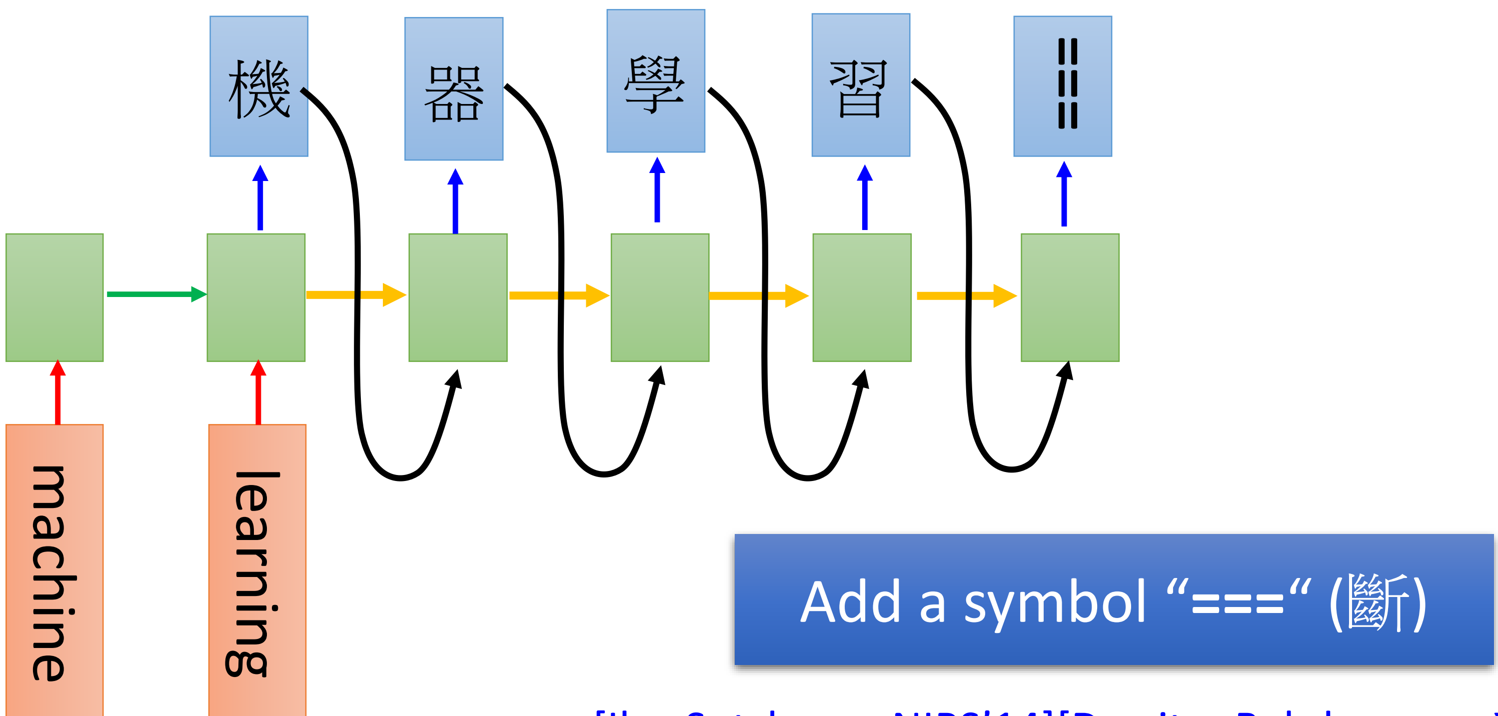
Many to Many (No Limitation)



接龍推文是ptt在推文中的一種趣味玩法，與推齊有些類似但又有所不同，是指在推文中接續上一樓的字句，而推出連續的意思。該類玩法確切起源已不可知(鄉民百科)

Many to Many (No Limitation)

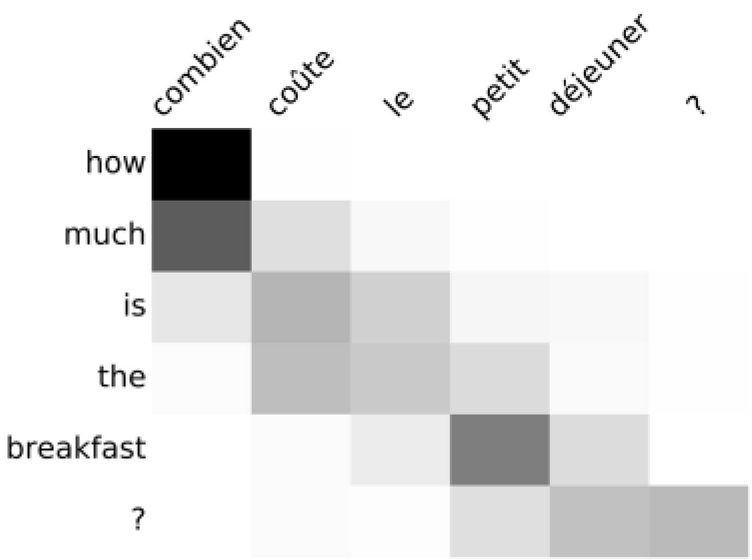
- Both input and output are both sequences ***with different lengths***. → ***Sequence to sequence learning***
  - E.g. ***Machine Translation*** (machine learning→機器學習)



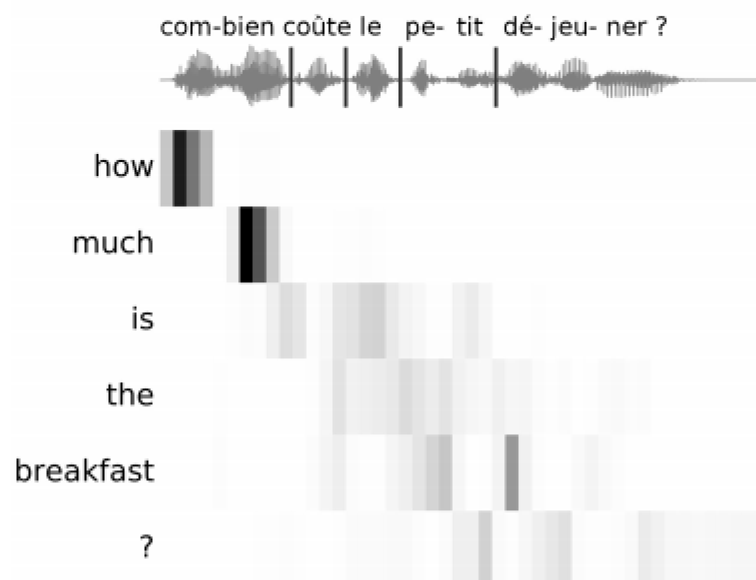
[Ilya Sutskever, NIPS'14][Dzmitry Bahdanau, arXiv'15] https://arxiv.org/pdf/1612.01744v1.pdf

Many to Many (No Limitation)

- Both input and output are both sequences ***with different lengths***. → ***Sequence to sequence learning***
  - E.g. ***Machine Translation*** (machine learning→機器學習)



(a) Machine translation alignment

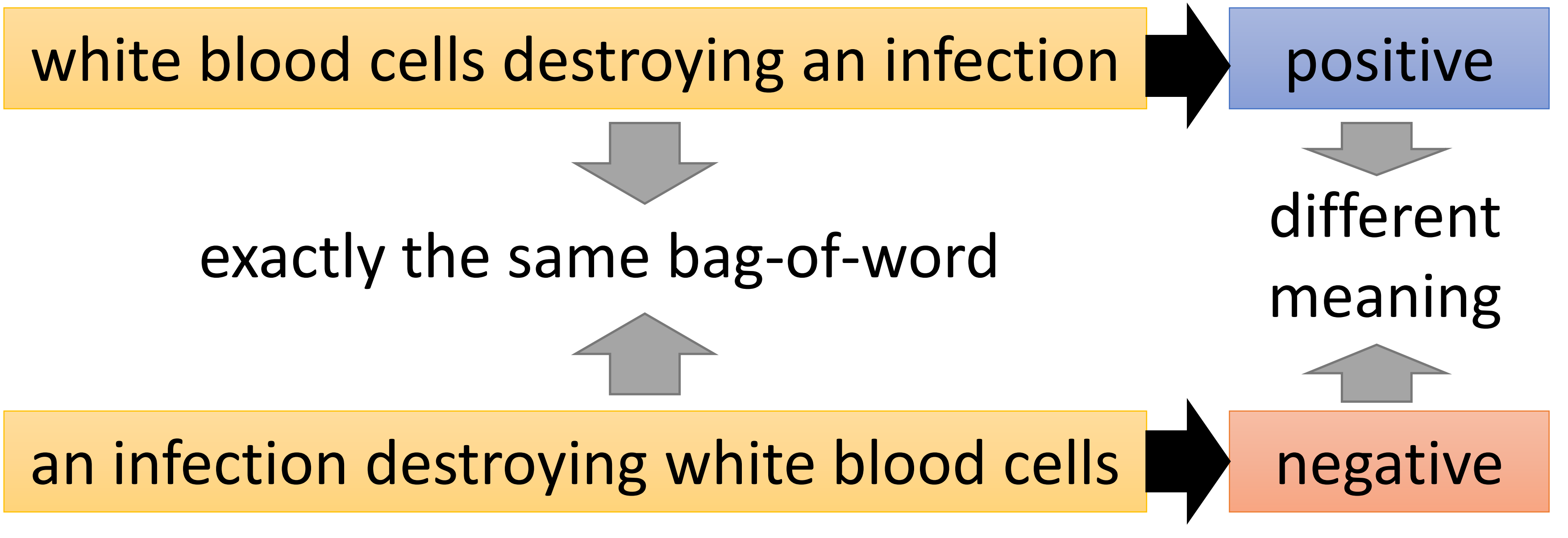


(b) Speech translation alignment

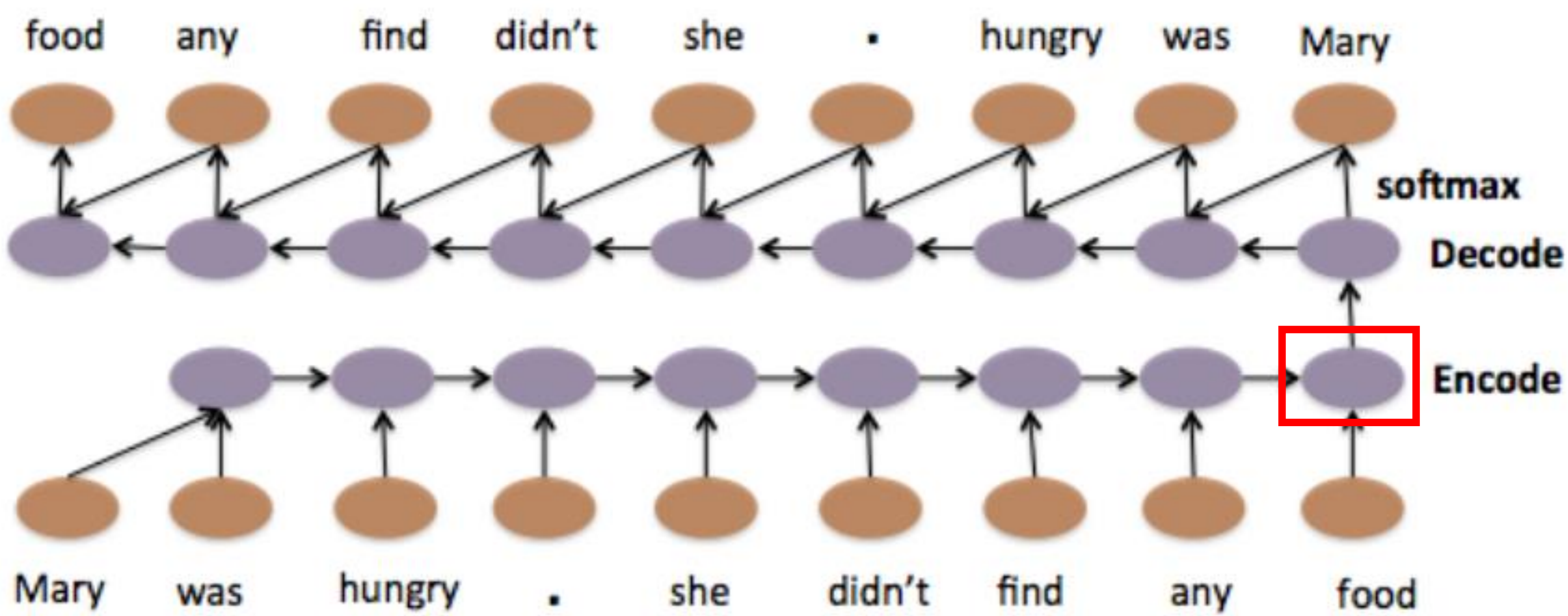
Figure 1: Alignments performed by the attention model during training

Sequence-to-sequence Auto-encoder - Text

- To understand the meaning of a word sequence, the order of the words can not be ignored.

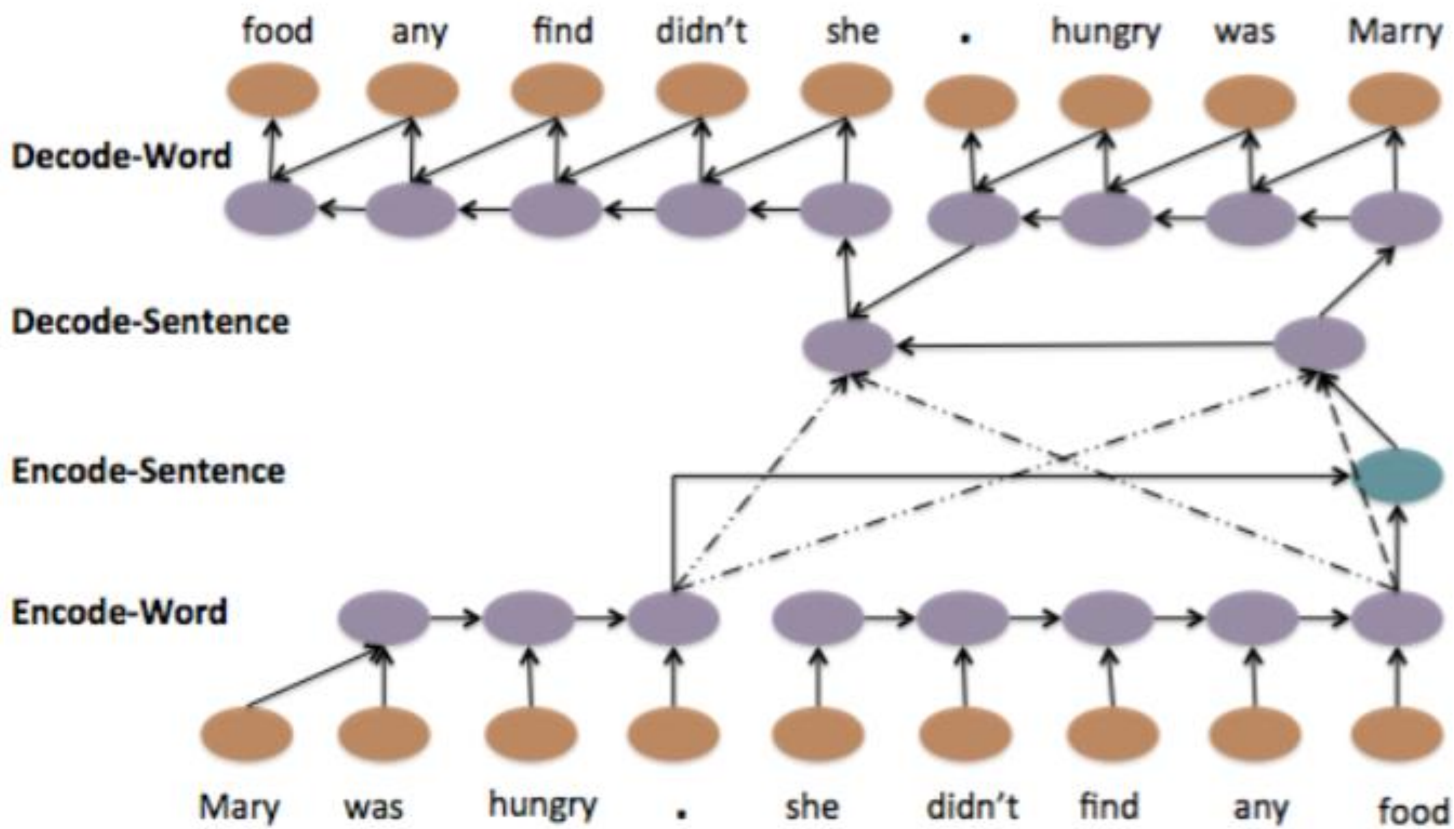


Sequence-to-sequence Auto-encoder - Text



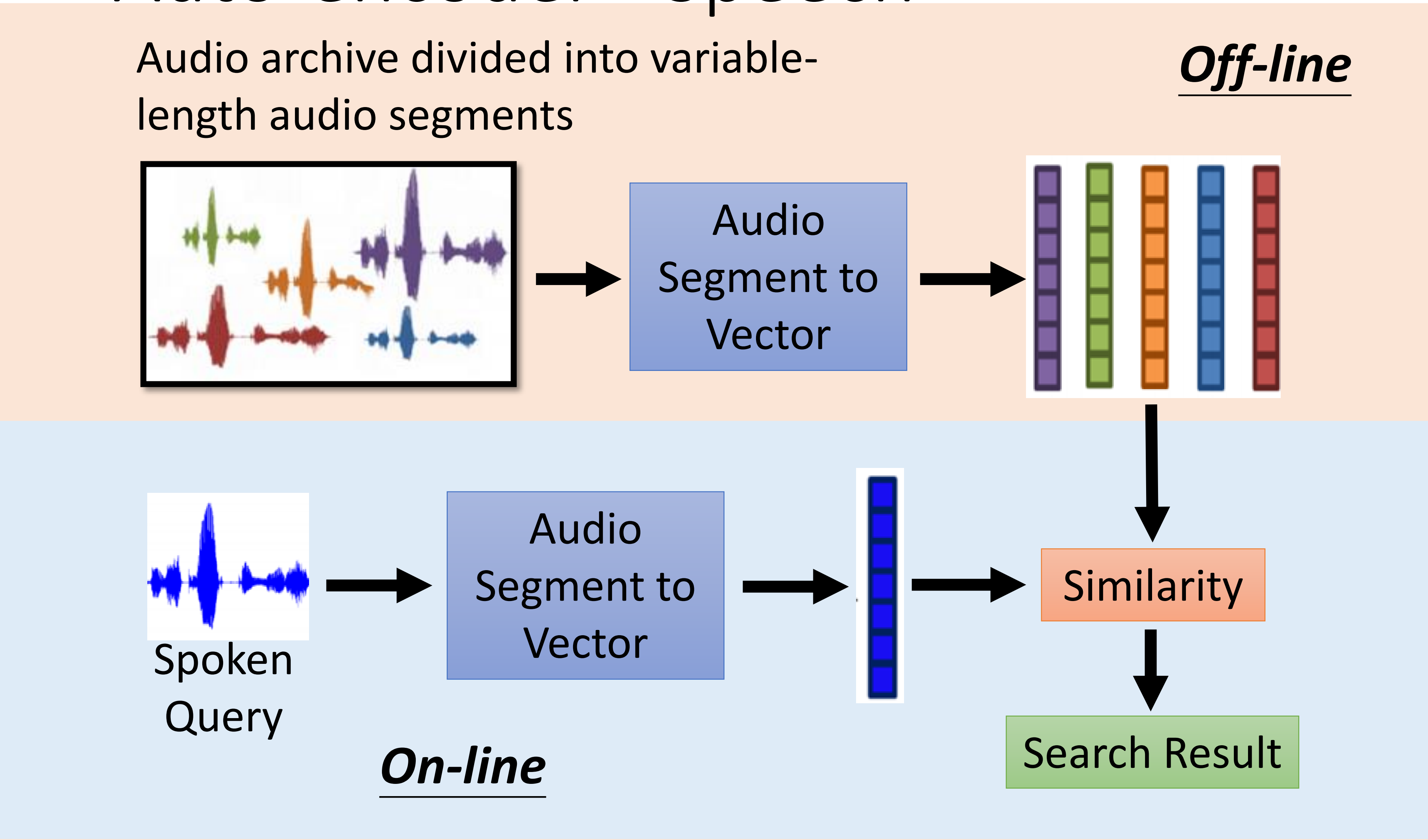
Li, Jiwei, Minh-Thang Luong, and Dan Jurafsky. "A hierarchical neural autoencoder for paragraphs and documents." *arXiv preprint arXiv:1506.01057*(2015).

Sequence-to-sequence Auto-encoder - Text

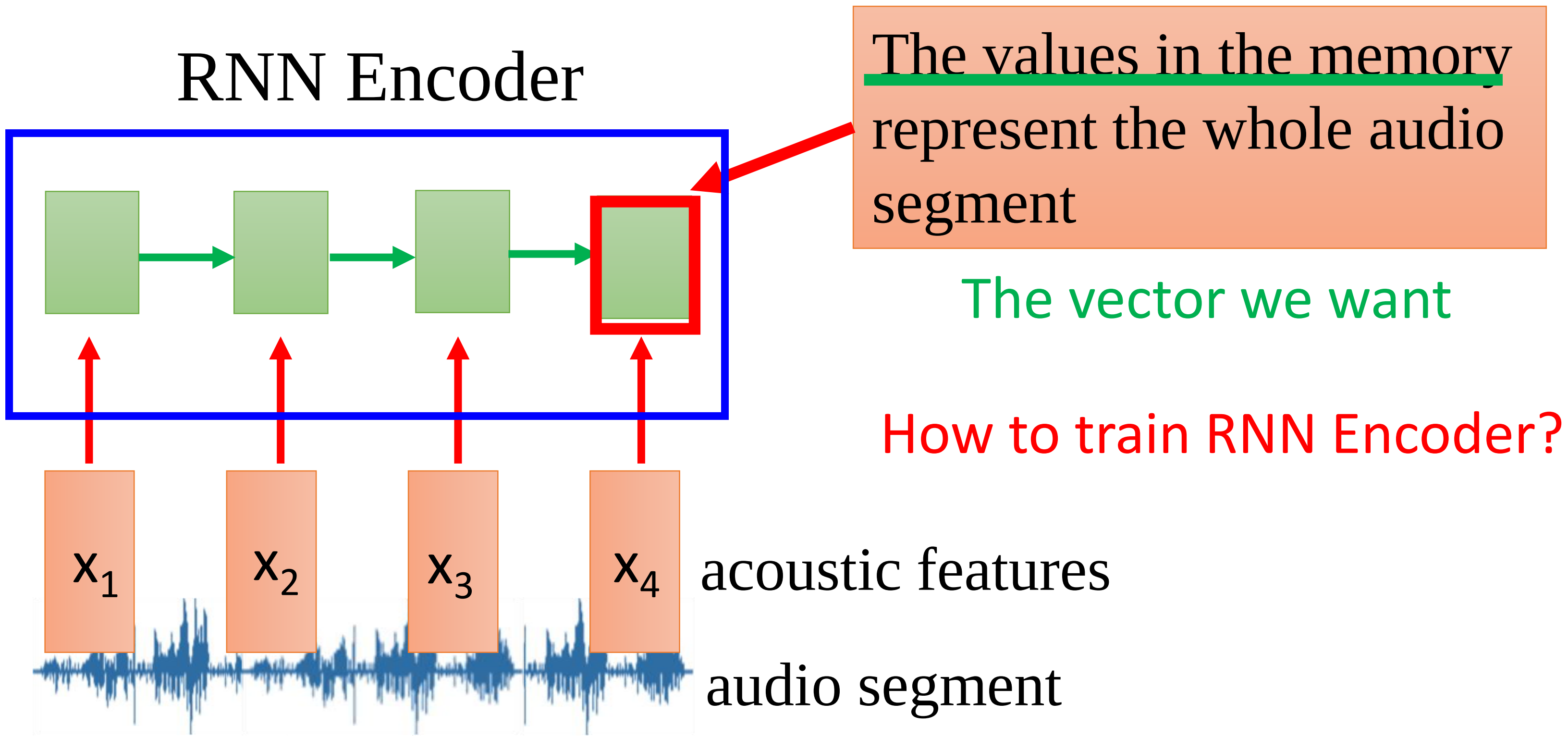


Li, Jiwei, Minh-Thang Luong, and Dan Jurafsky. "A hierarchical neural autoencoder for paragraphs and documents." *arXiv preprint arXiv:1506.01057*(2015).

Sequence-to-sequence Auto-encoder - Speech

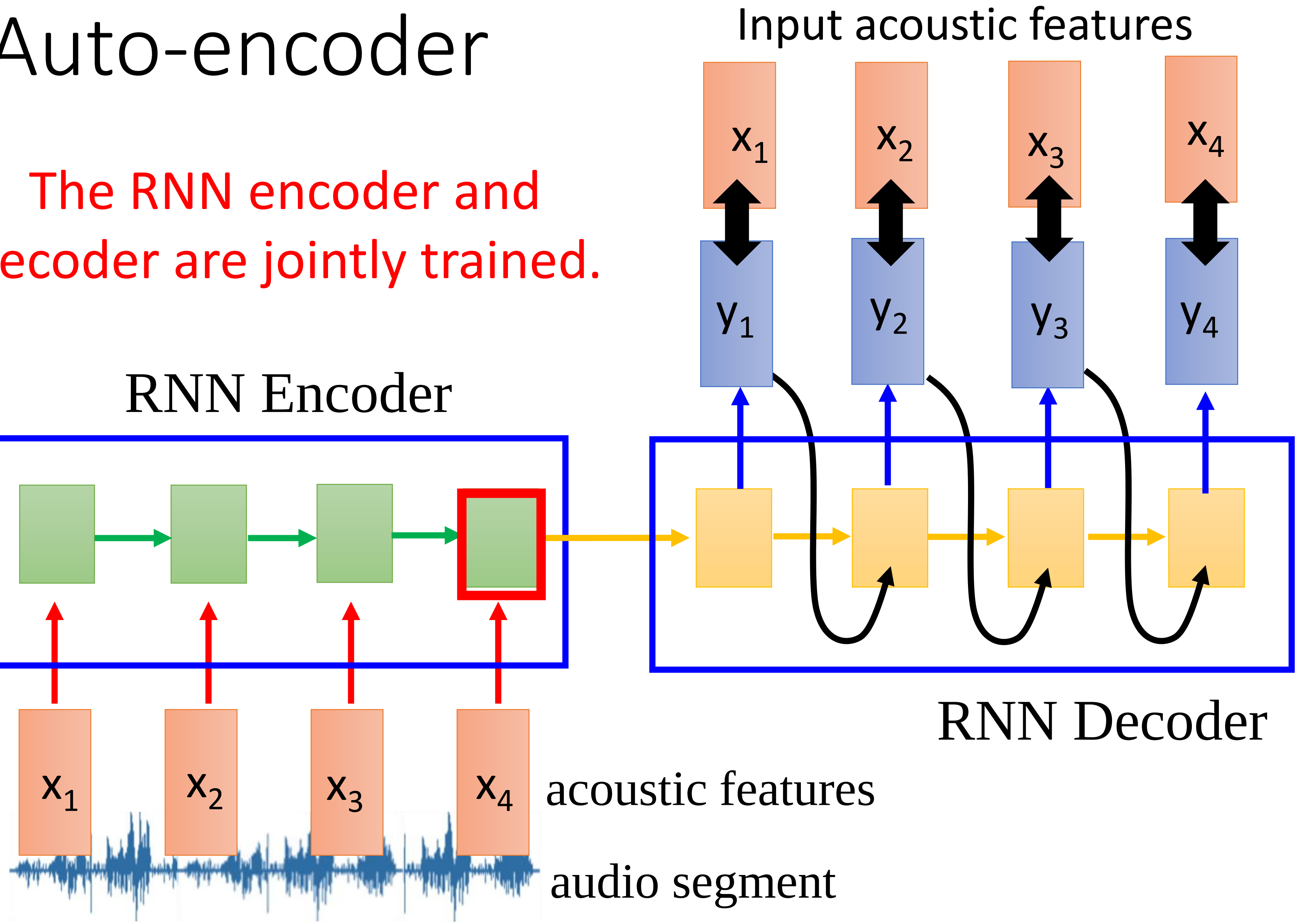


Sequence-to-sequence Auto-encoder - Speech



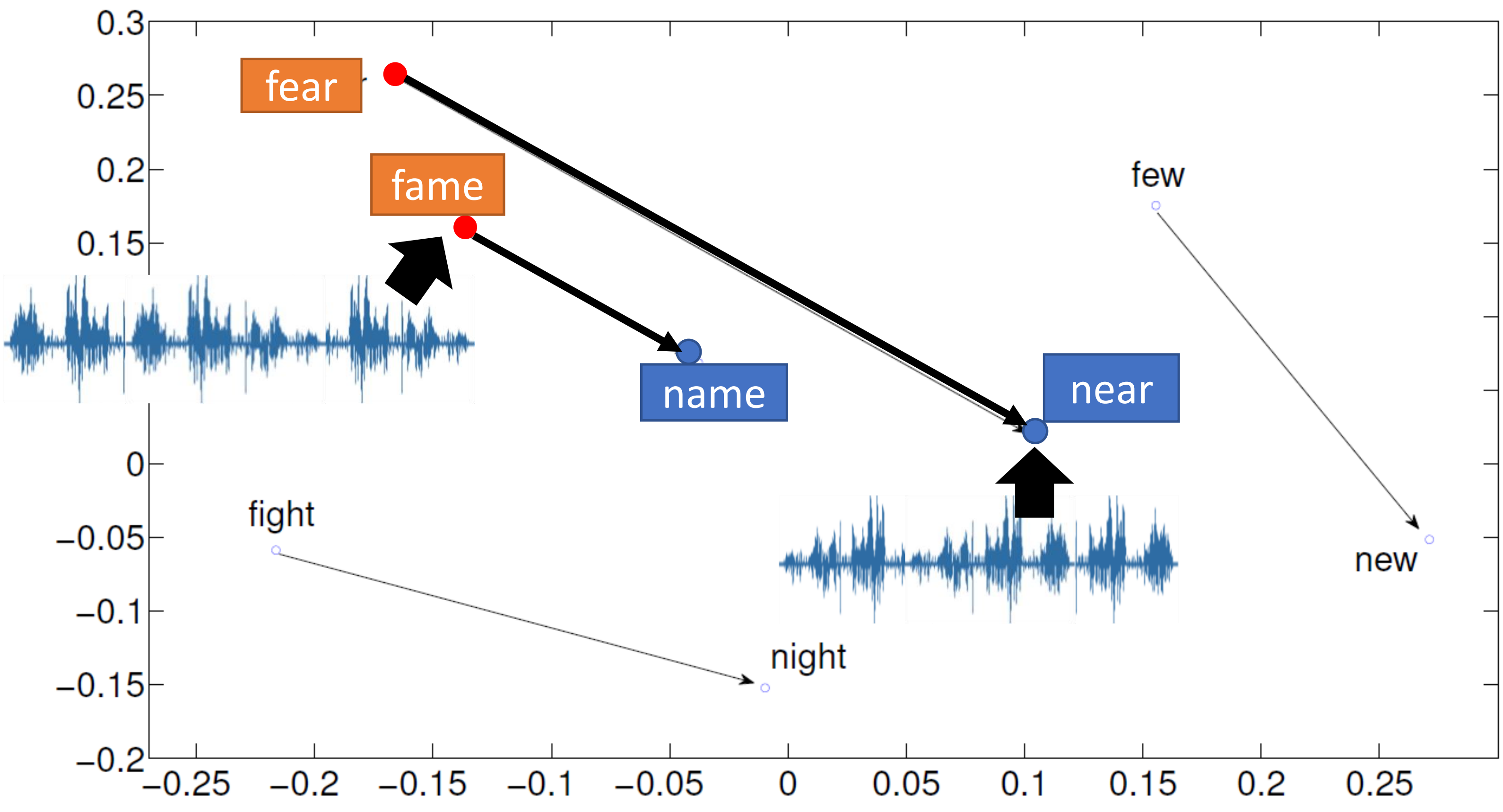
Sequence-to-sequence Auto-encoder

The RNN encoder and decoder are jointly trained.

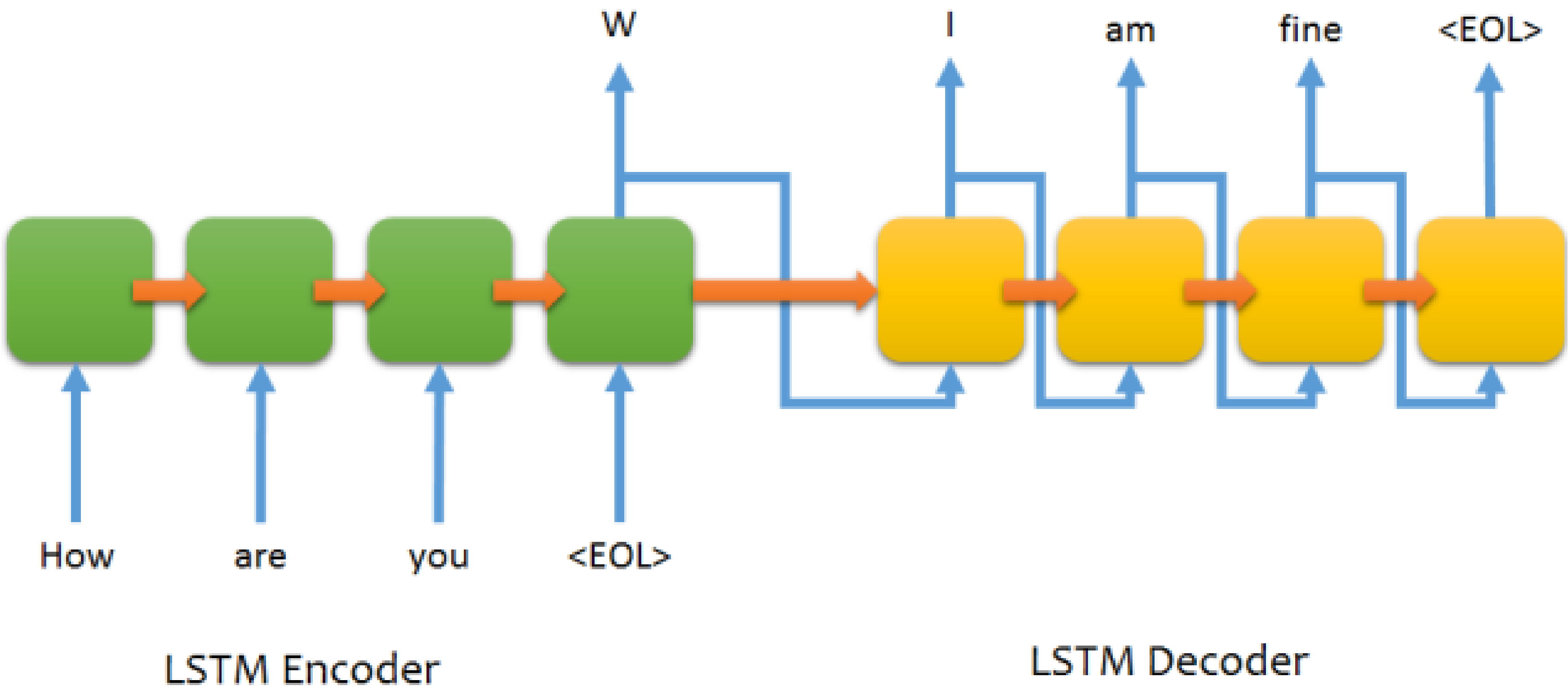


Sequence-to-sequence Auto-encoder - Speech

- Visualizing embedding vectors of the words

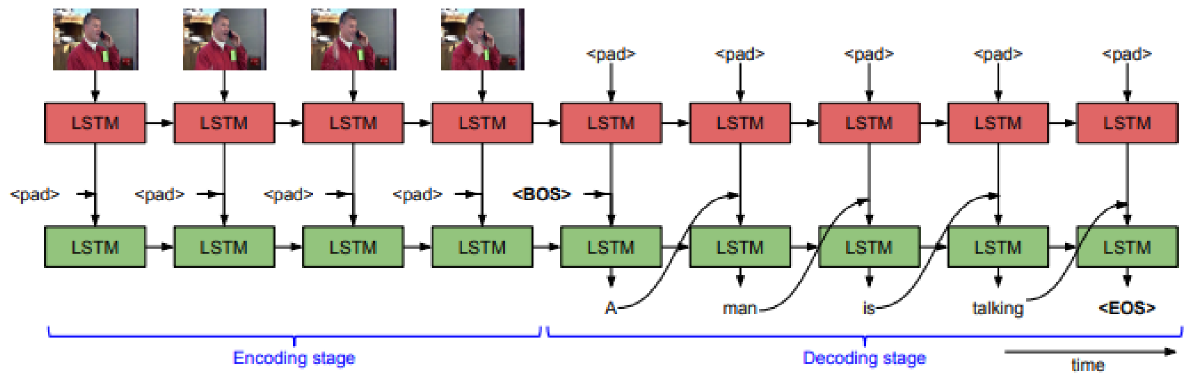


Demo: Chat-bot



電視影集 (~40,000 sentences)、美國總統大選辯論

Video Caption Generation

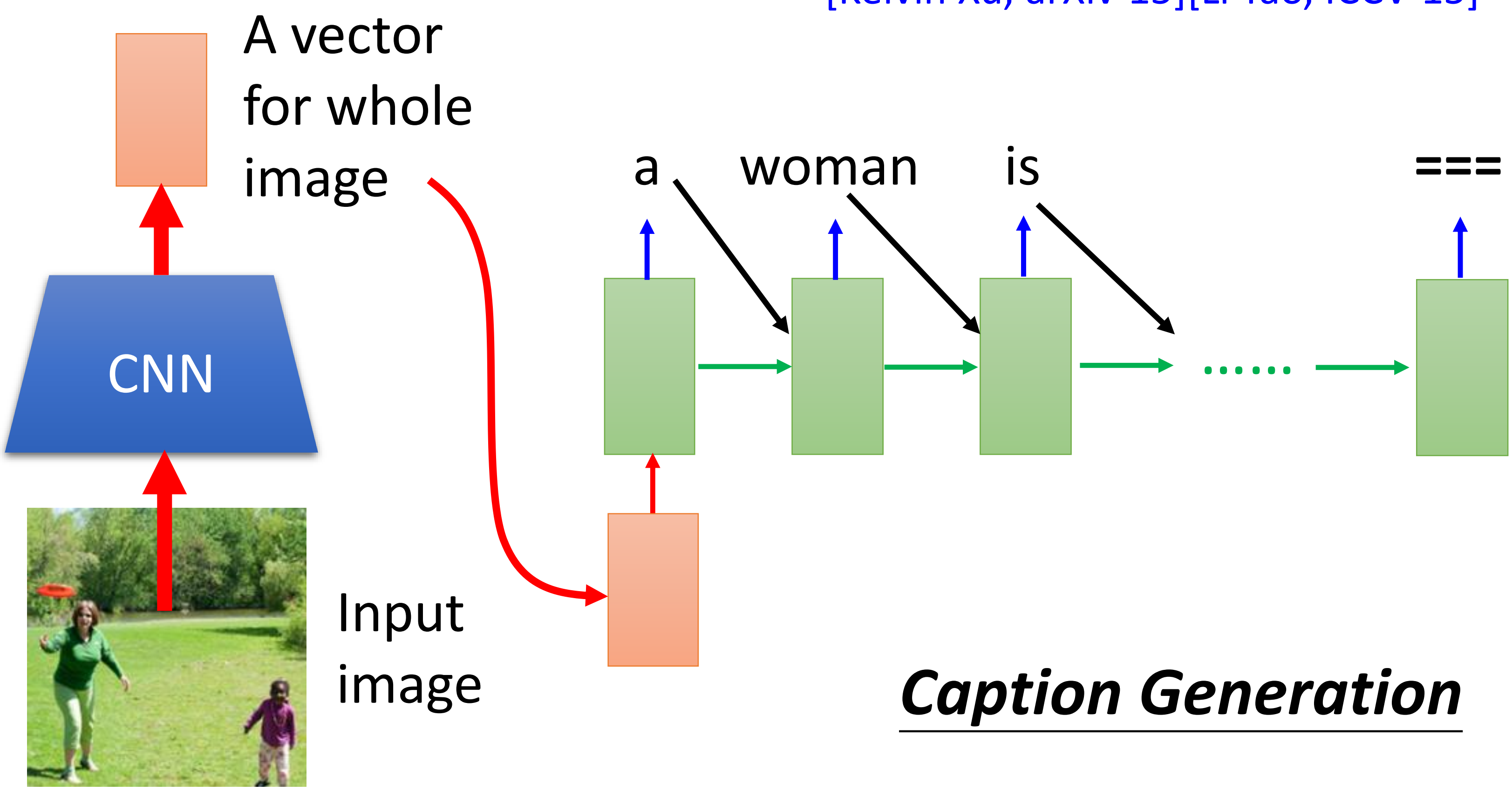


Subhashini Venugopalan, Marcus Rohrbach, Jeffrey Donahue, Raymond Mooney, Trevor Darrell, and Kate Saenko. 2015. Sequence to Sequence -- Video to Text. In Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV) (ICCV '15). IEEE Computer Society, Washington, DC, USA, 4534-4542.

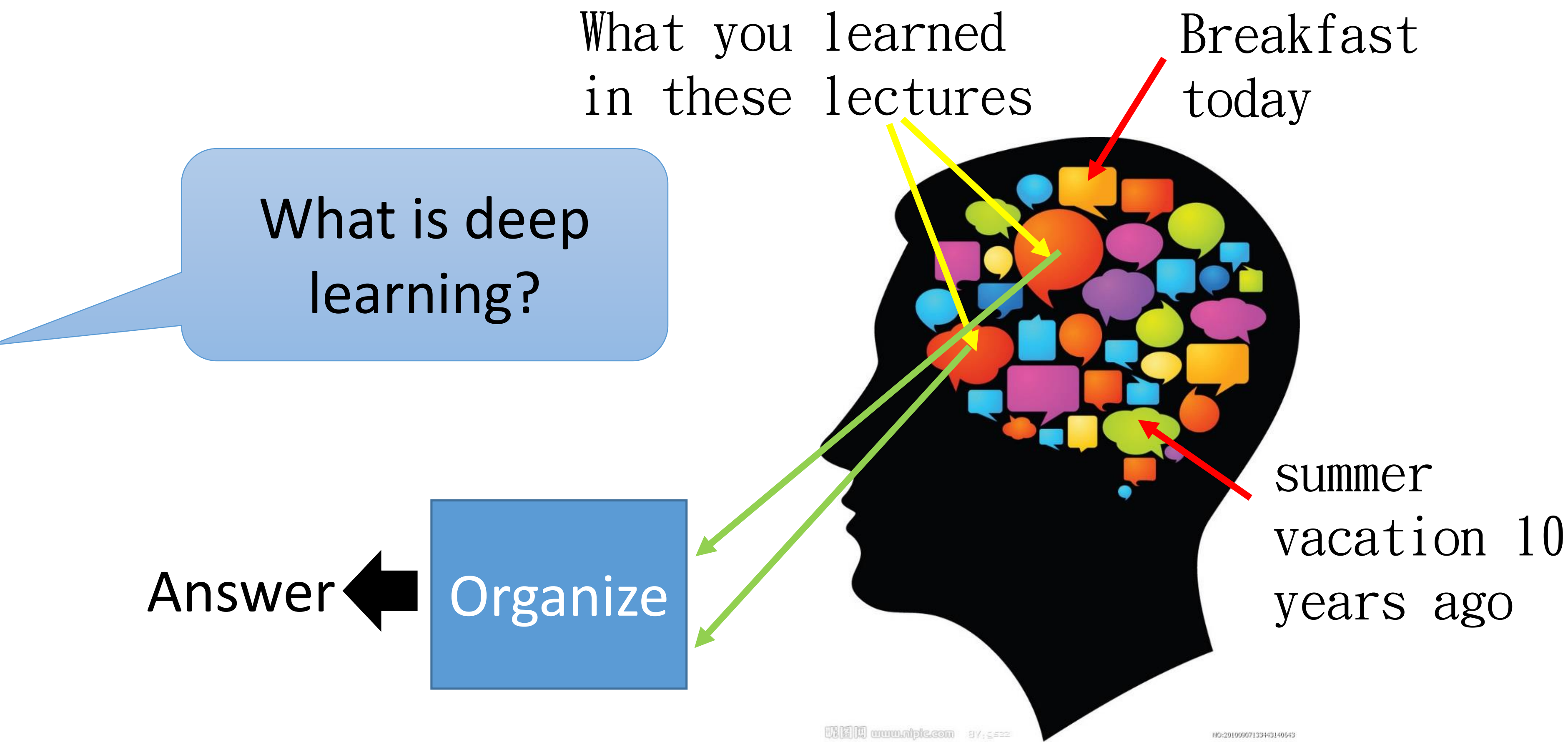
Demo: Image Caption Generation

- Input an image, but output a sequence of words

[Kelvin Xu, arXiv'15][Li Yao, ICCV'15]

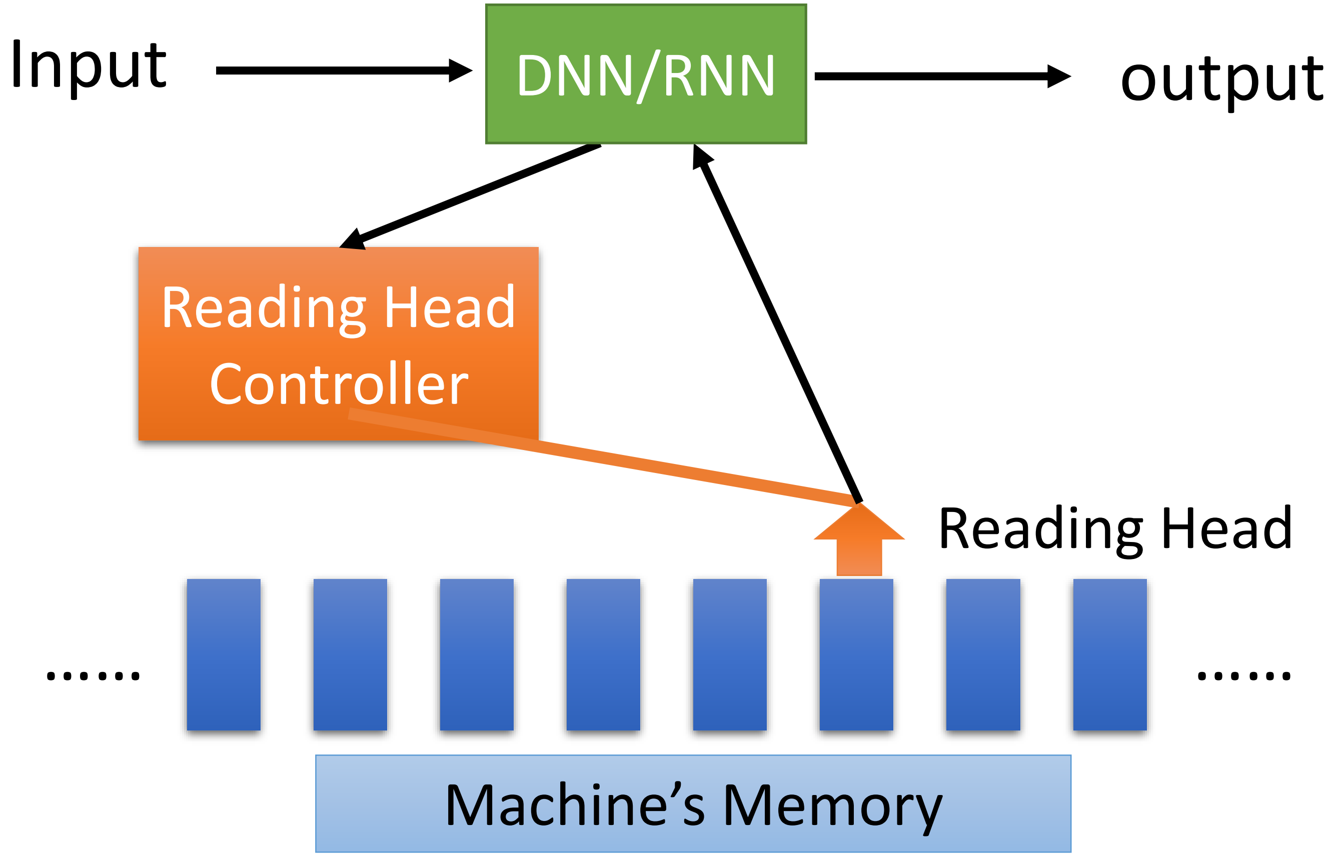


Attention-based Model



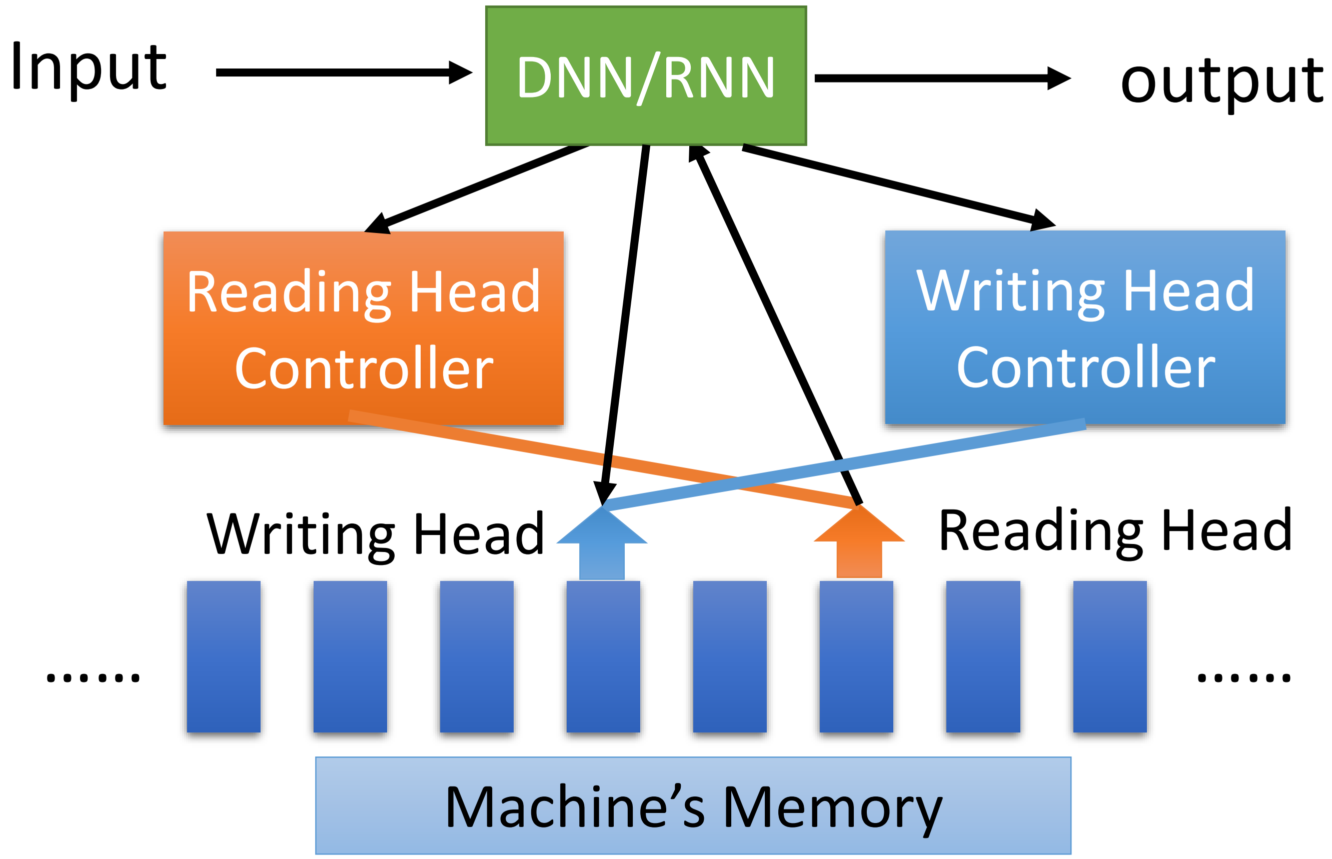
http://henrylo1605.blogspot.tw/2015/05/blog-post_56.html

Attention-based Model



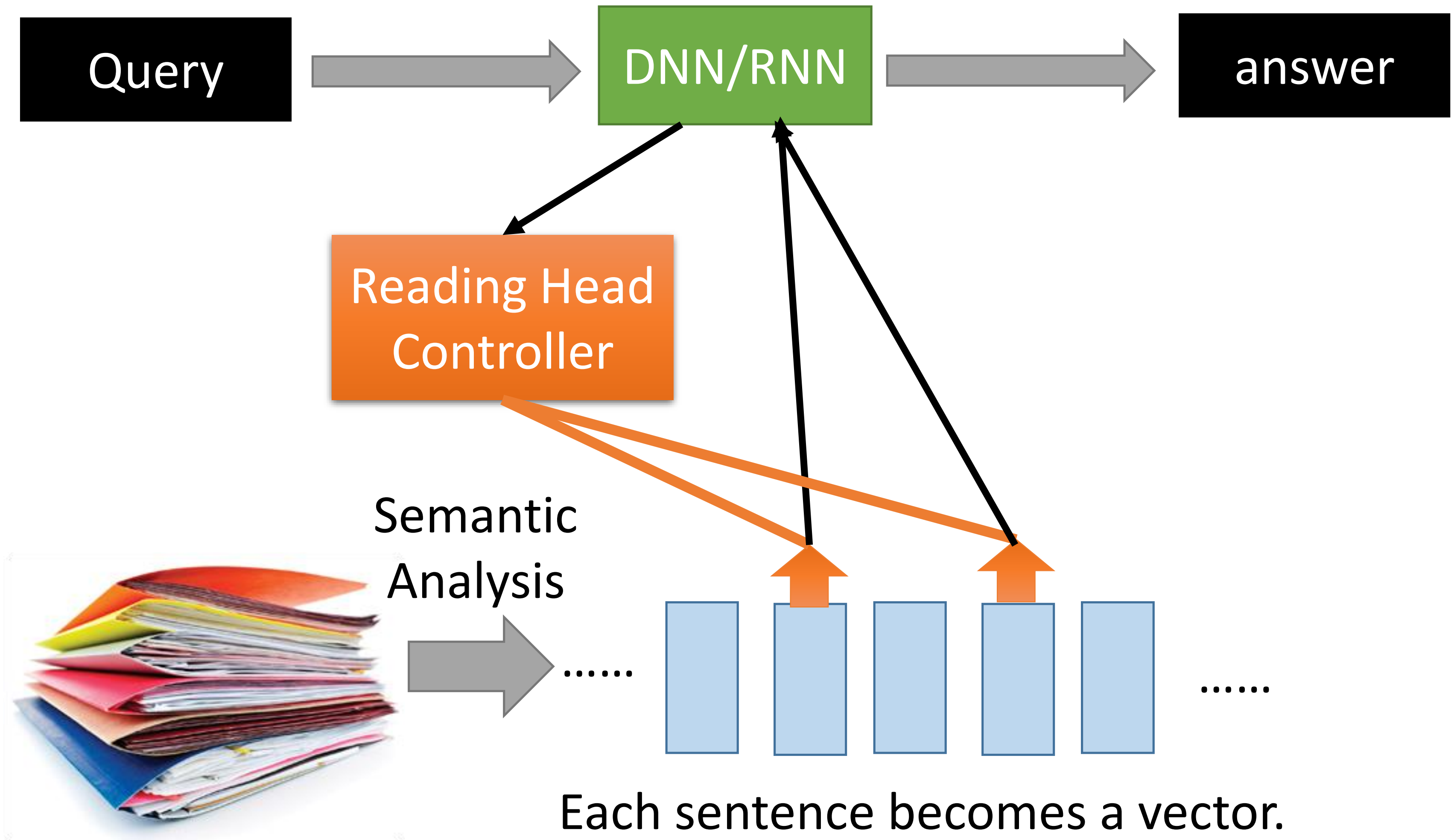
Ref:
http://speech.ee.ntu.edu.tw/~tlkagk/courses/MLDS_2015_2/Lecture/Attain%20(v3).ecm.mp4/index.html

Attention-based Model v2

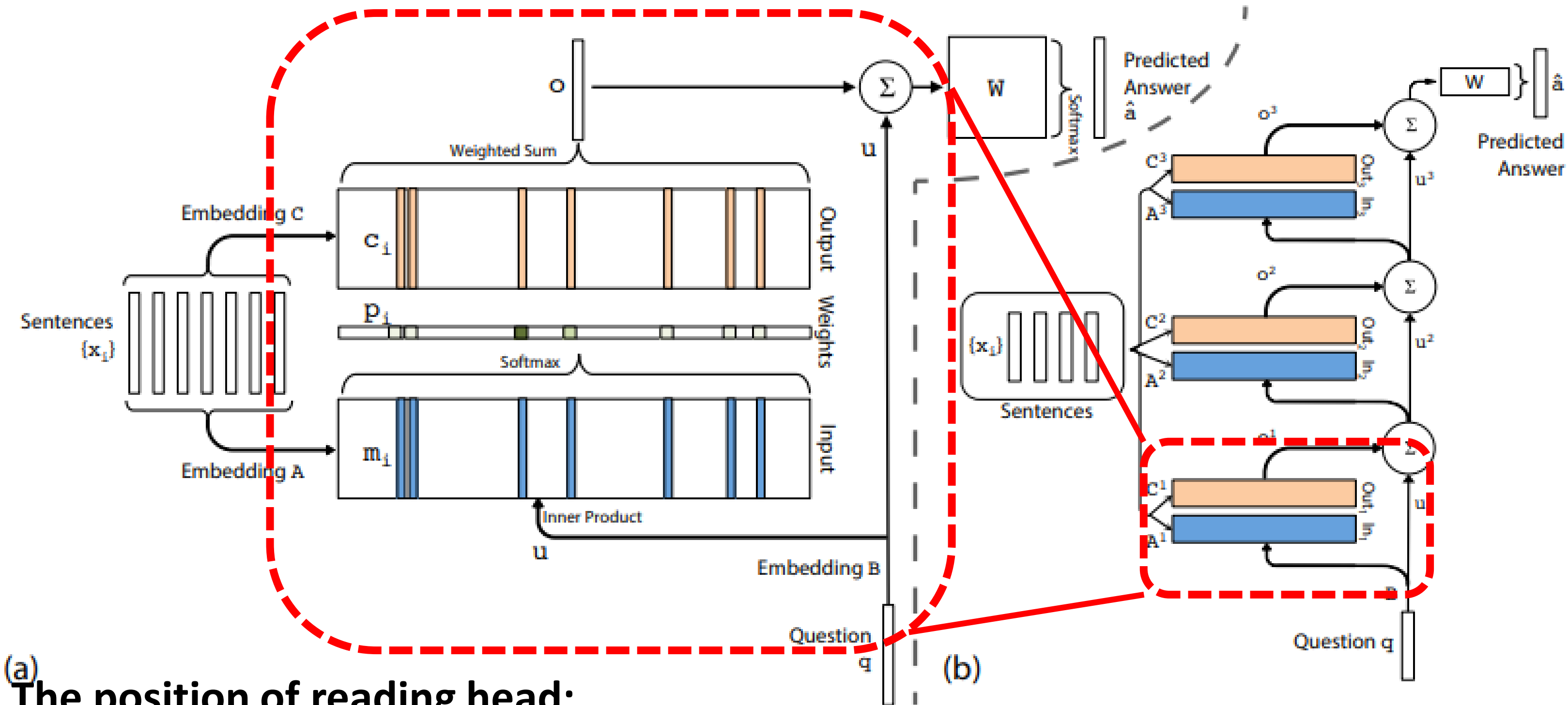


Neural Turing Machine

Reading Comprehension



Reading Comprehension



The position of reading head:

Story (16: basic induction)	Support	Hop 1	Hop 2	Hop 3
Brian is a frog.	yes	0.00	0.98	0.00
Lily is gray.		0.07	0.00	0.00
Brian is yellow.	yes	0.07	0.00	1.00
Julius is green.		0.06	0.00	0.00
Greg is a frog.	yes	0.76	0.02	0.00
What color is Greg? Answer: yellow Prediction: yellow				

End-To-End Memory Networks. S. Sukhbaatar, A. Szlam, J. Weston, R. Fergus. NIPS, 2015.

Keras example: https://github.com/fchollet/keras/blob/master/examples/babi_memnn.py

Visual Question Answering

What is the mustache made of?

AI System

bananas

source: http://visualqa.org/

Visual Question Answering