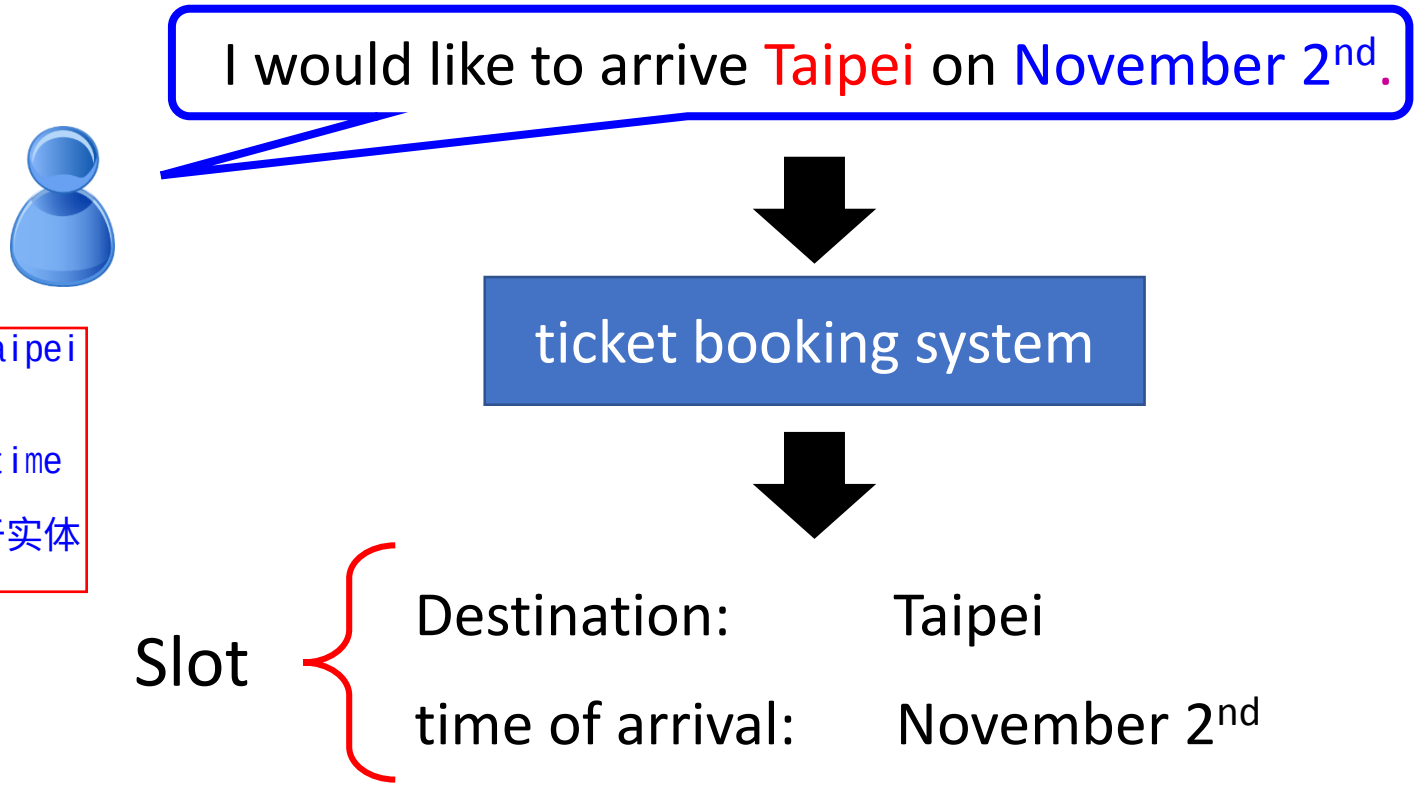
Recurrent Neural Network (RNN)

Example Application

- Slot Filling



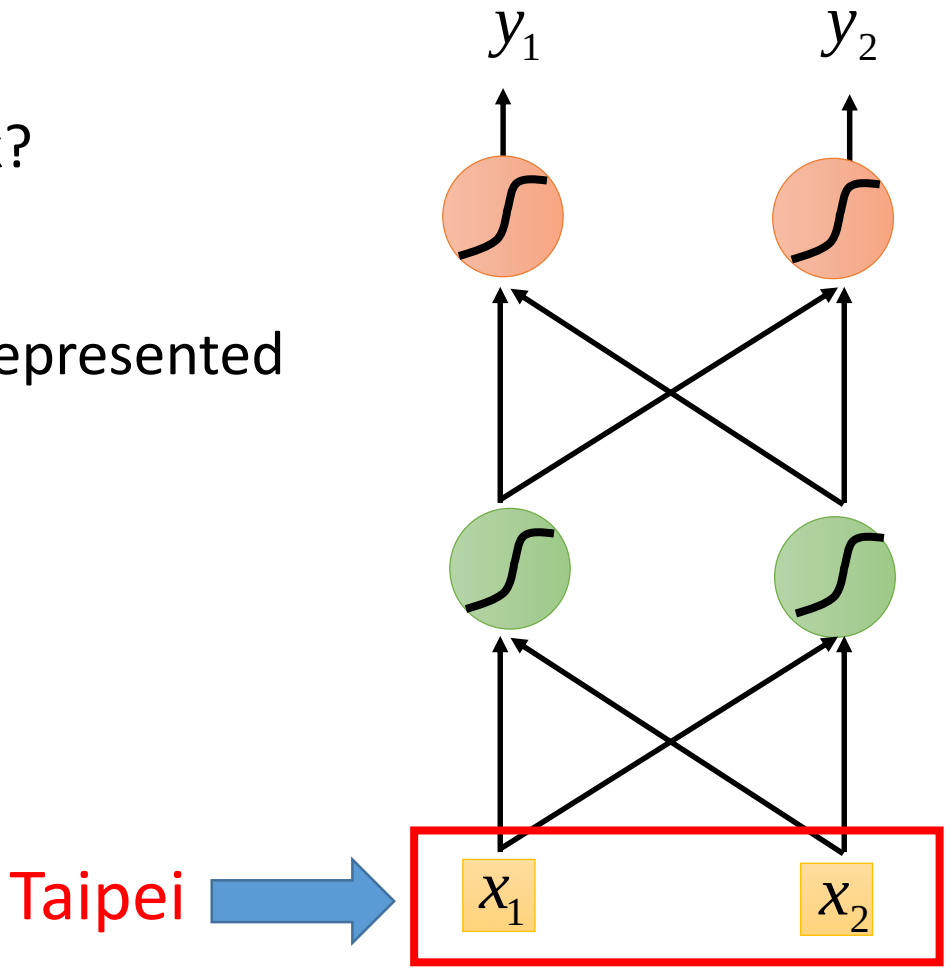
1. 系统应该知道Taipei属于Destination。
2. 系统应该知道November 2nd属于time of arrival。
3. 这个任务类似于实体识别

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]

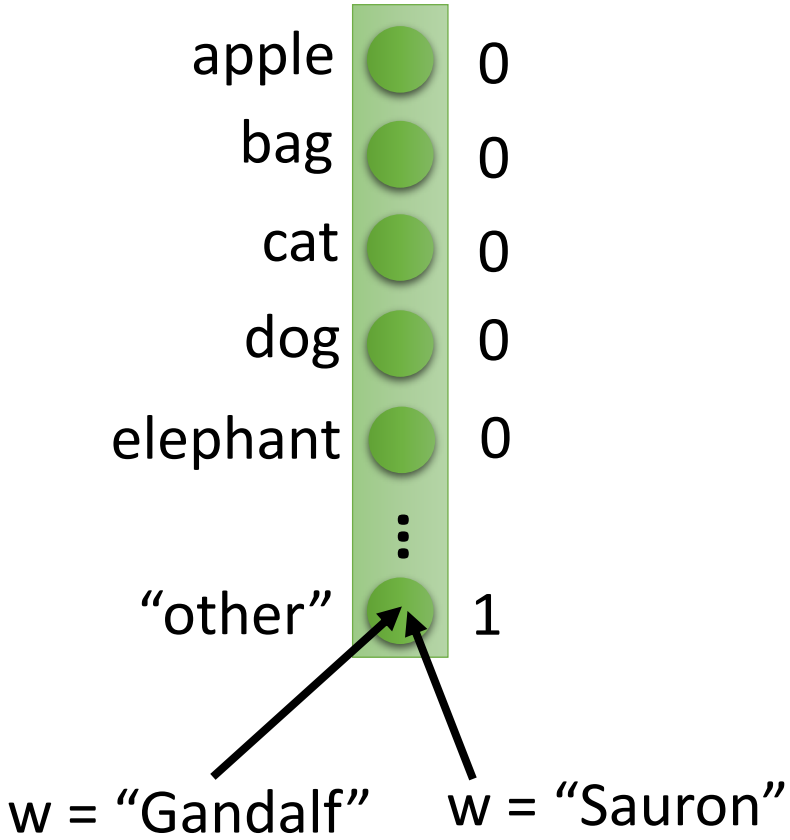
cat = [0 0 1 0 0]

dog = [0 0 0 1 0]

elephant = [0 0 0 0 1]

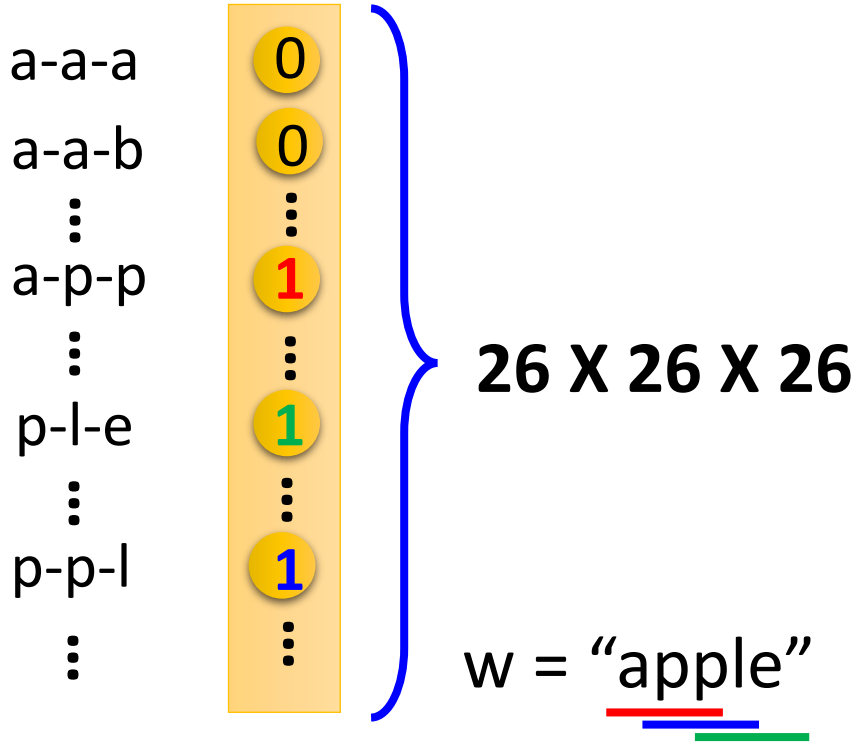
Beyond 1-of-N encoding

Dimension for "Other"



添加未登录词维度

Word hashing



用字母3gram构建词典，降低未登录词出现的可能性。

Example Application

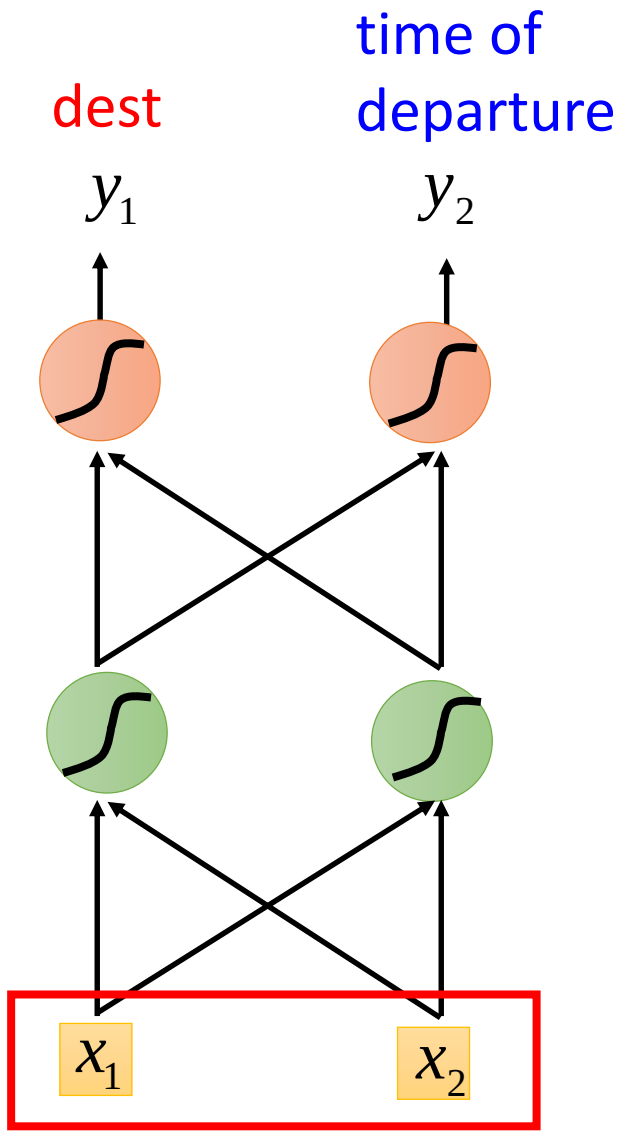
Solving slot filling by Feedforward network?

Input: a word

(Each word is represented as a vector)

Output:

Probability distribution that the input word belonging to the slots



这里的类别只有两个，实际使用中还应该加入“其他”这个类别。

同一个词语“Taipei”分别属于类别“dest”和“place of departure”，无法利用语前后关系的模型是无法区分的。这一点应该就是序列型神经网络优于传统文本分类模型的地方。

Example Application

arrive Taipei on November 2nd

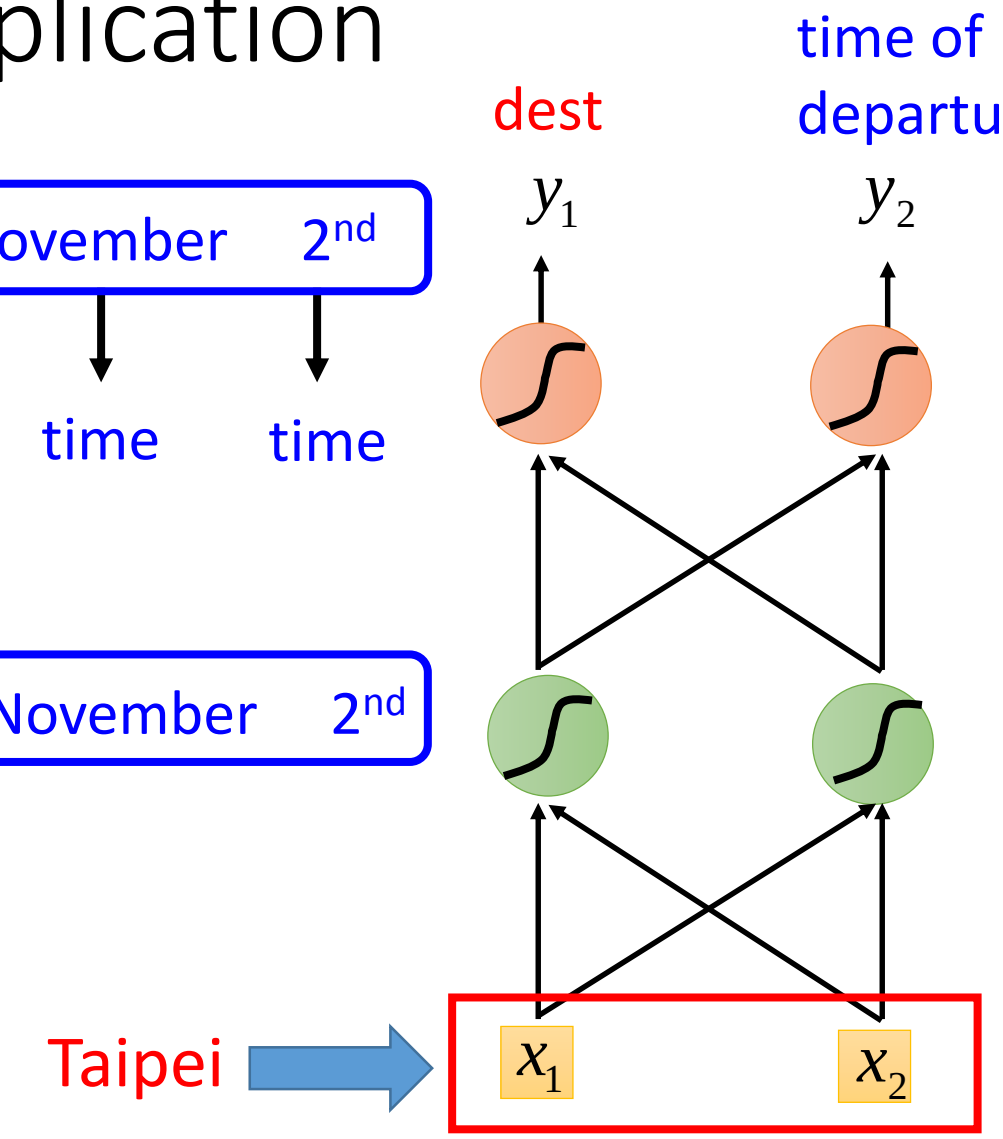
other dest other time time

Problem?

leave Taipei on November 2nd

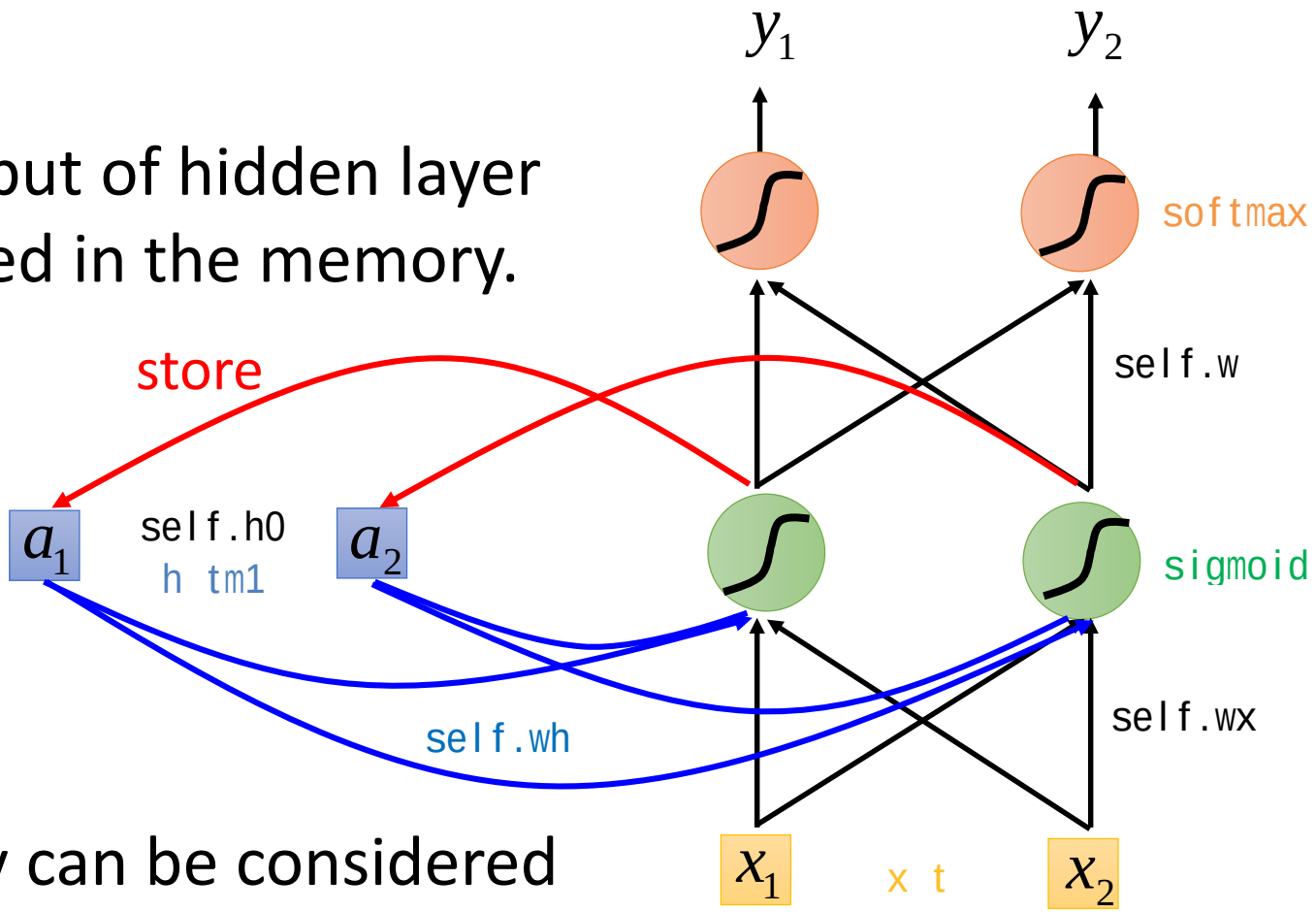
place of departure

Neural network needs memory!



Recurrent Neural Network (RNN)

The output of hidden layer are stored in the memory.

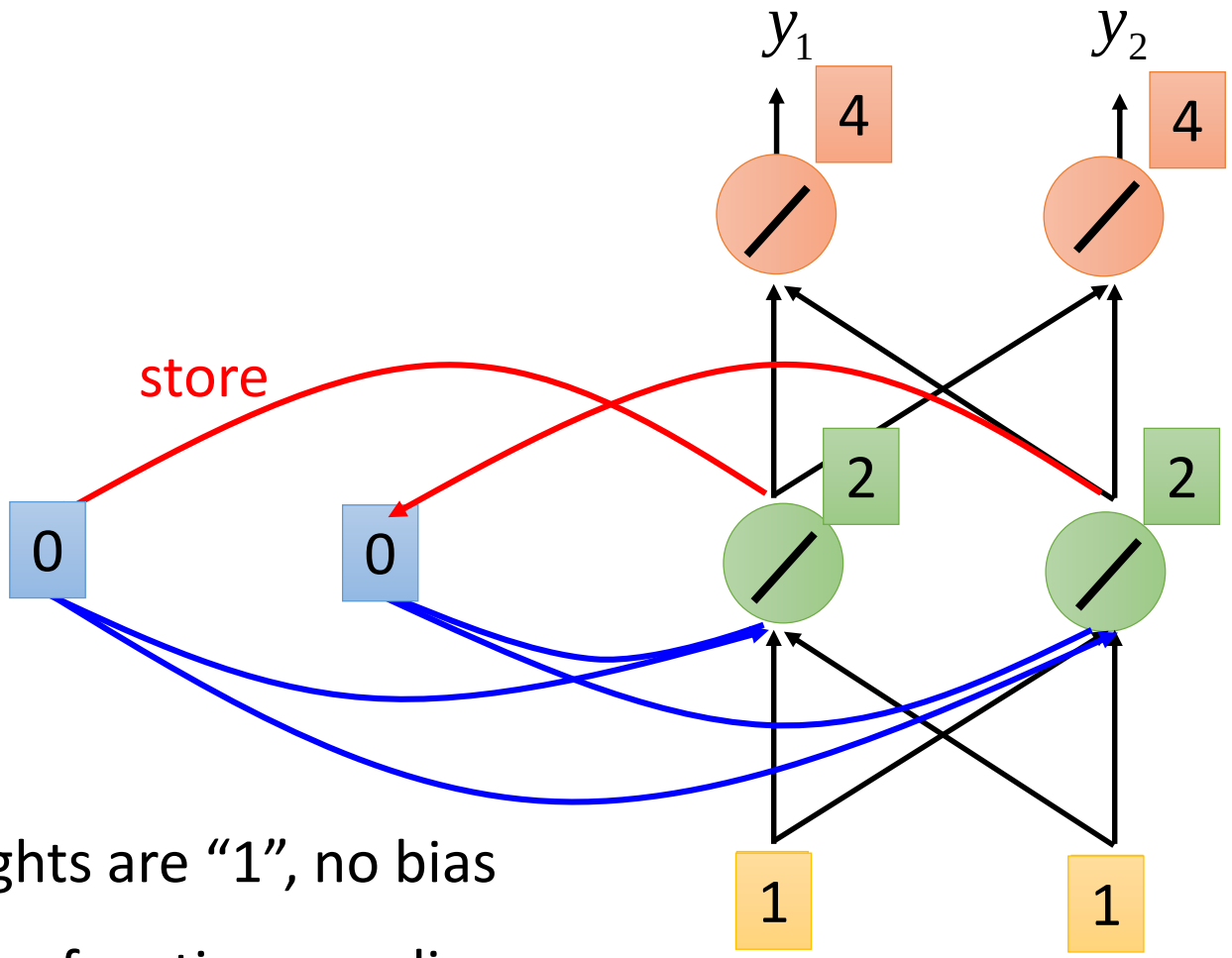


Memory can be considered as another input.

Example

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

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



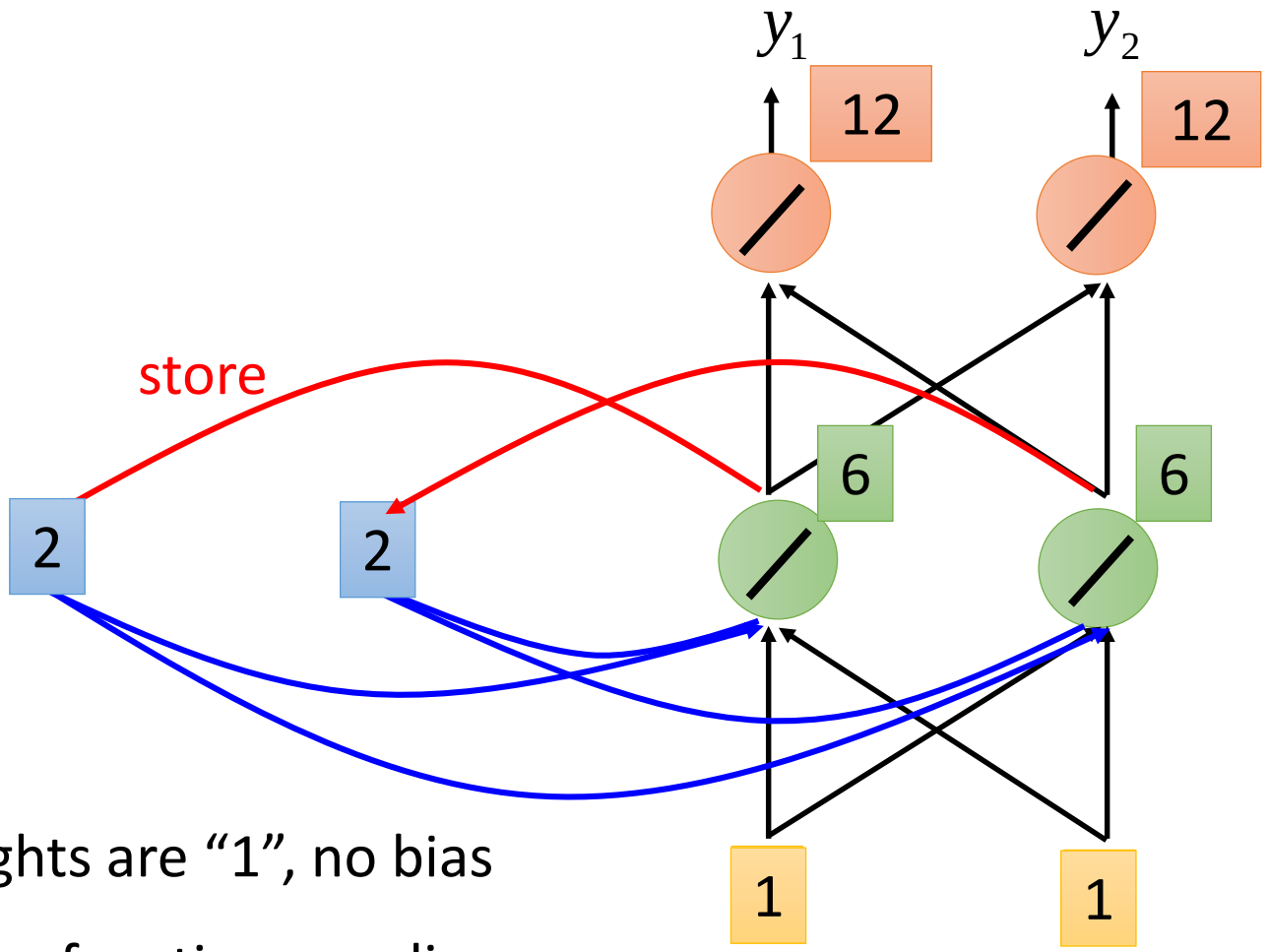
All the weights are “1”, no bias

All activation functions are linear

Example

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

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

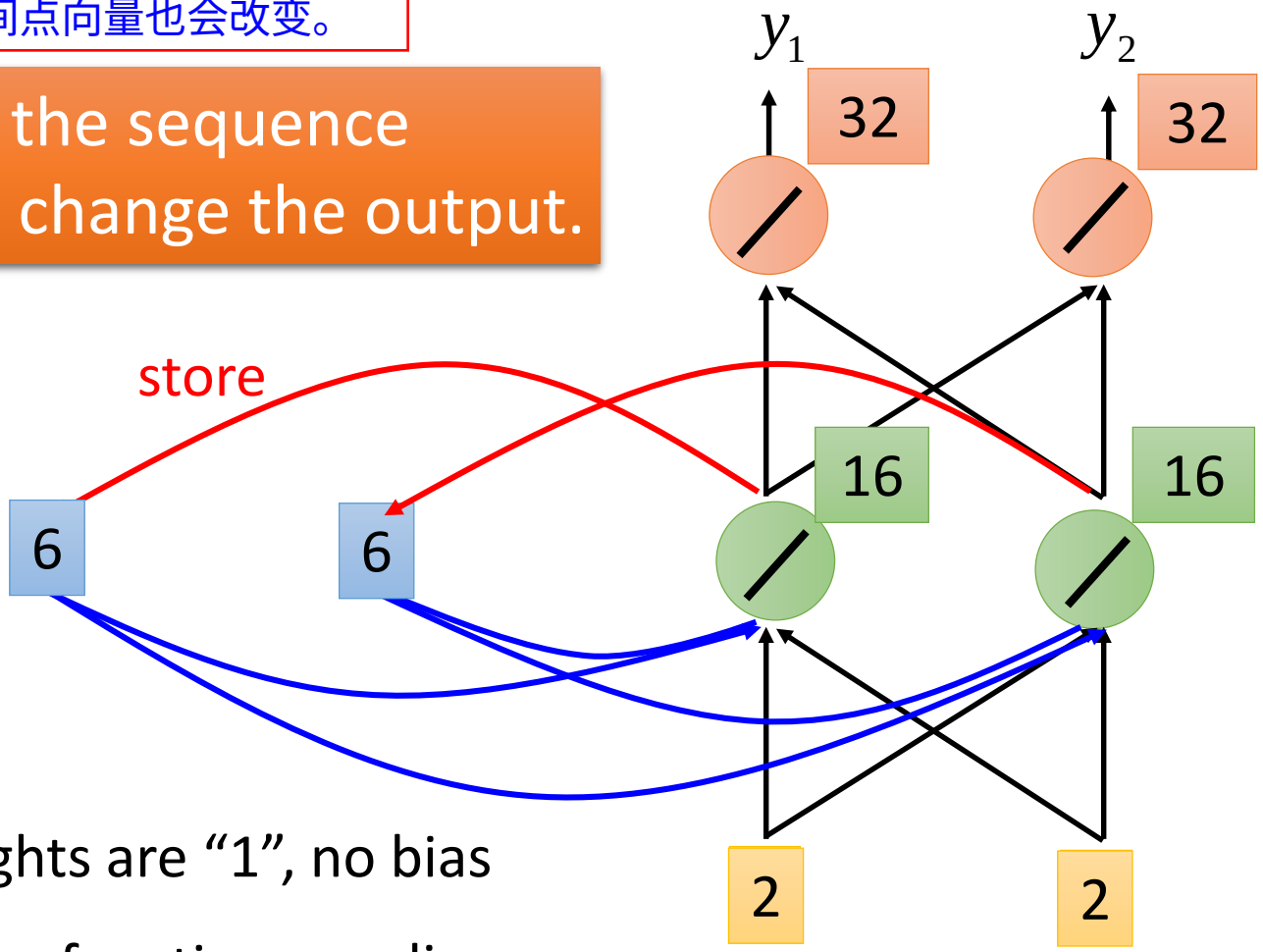
Example

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

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

改变Input sequence各时间点向量的顺序，output sequence各时间点向量也会改变。

Changing the sequence order will change the output.



All the weights are “1”, no bias

All activation functions are linear

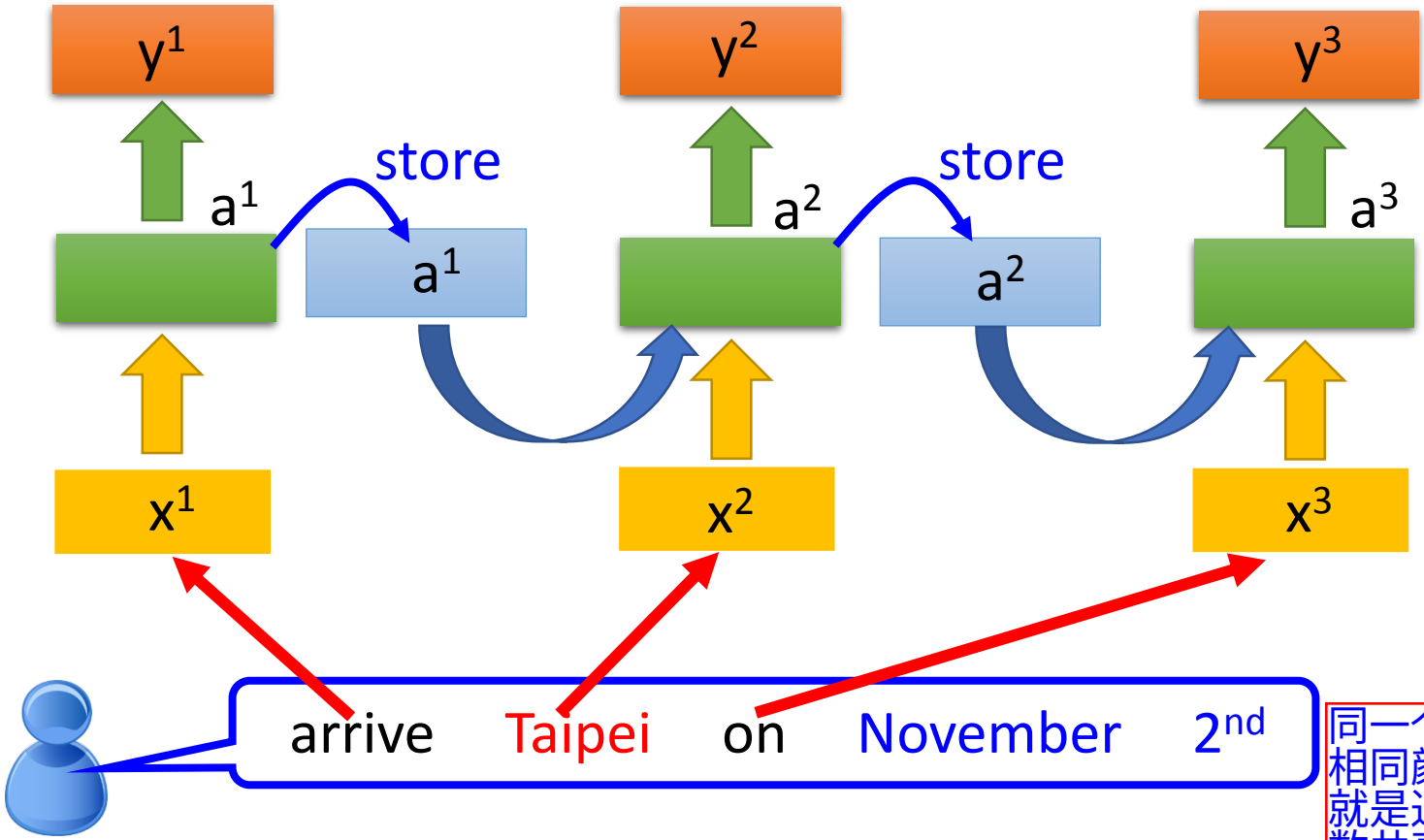
RNN

The same network is used again and again.

Probability of "arrive" in each slot

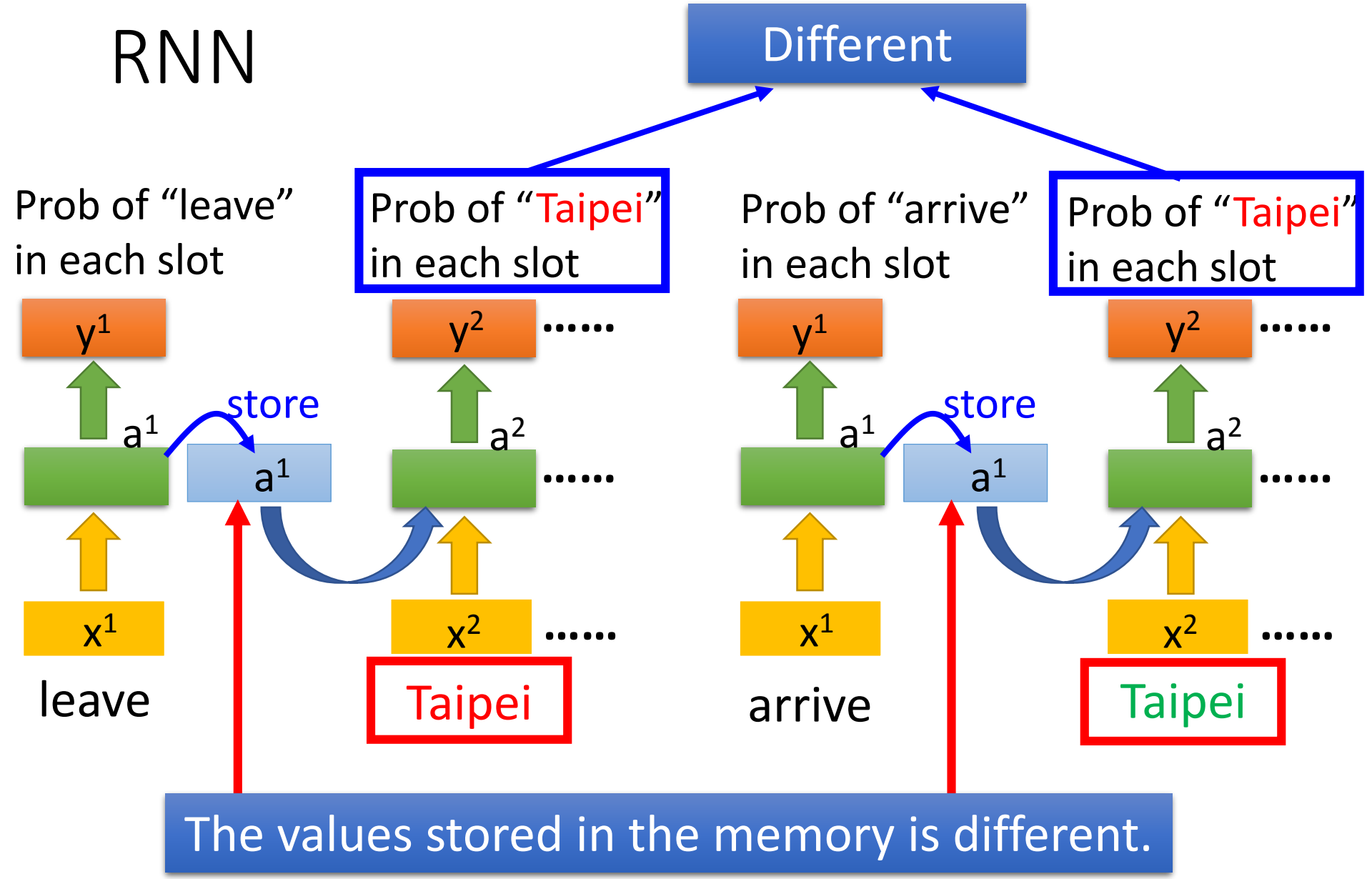
Probability of "Taipei" in each slot

Probability of "on" in each slot



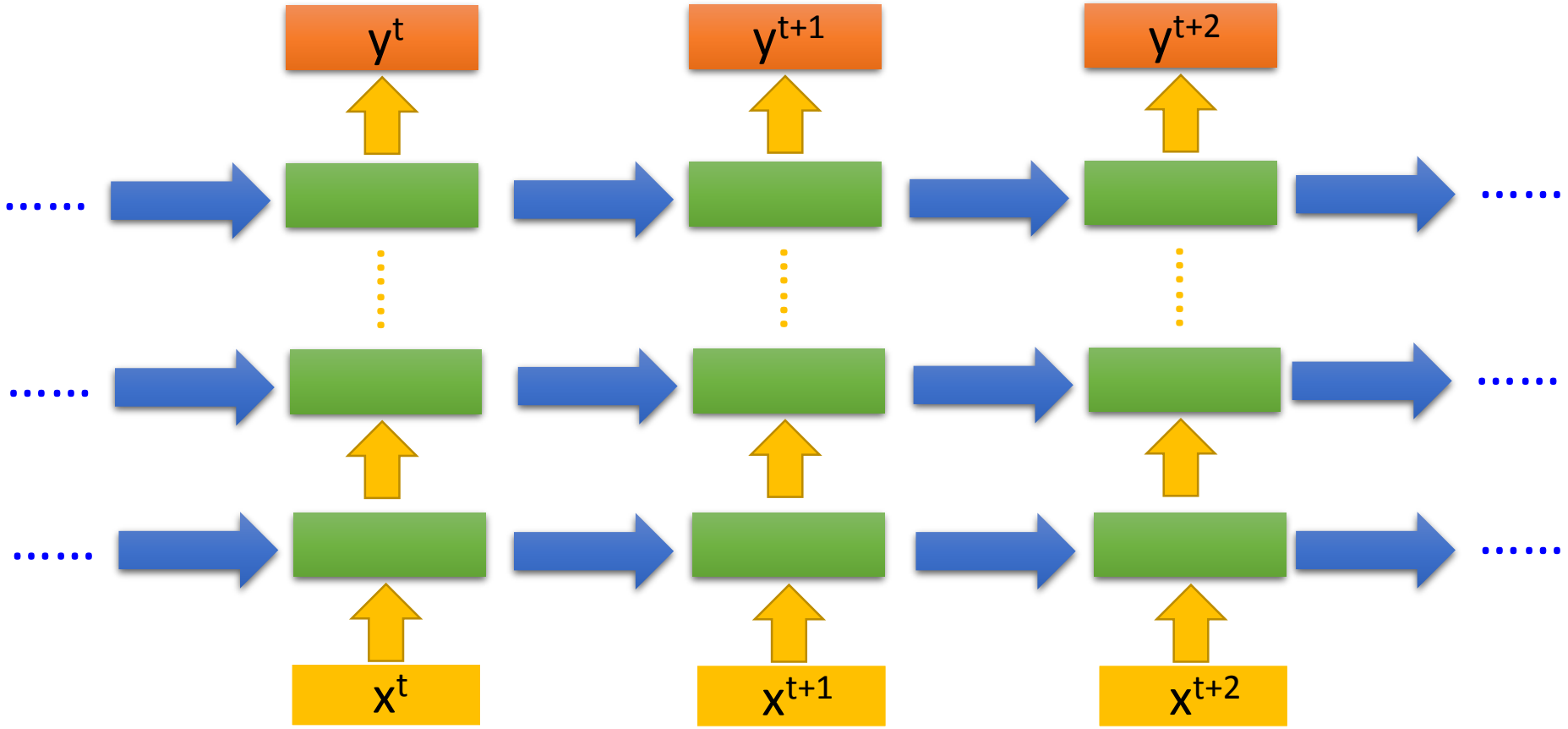
同一个weight用相同颜色表示，就是通常说的参数共享。

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
y1
y2
.....
a1
a2
store
x1
x2
leave
Taipei
arrive
Taipei
The values stored in the memory is different.

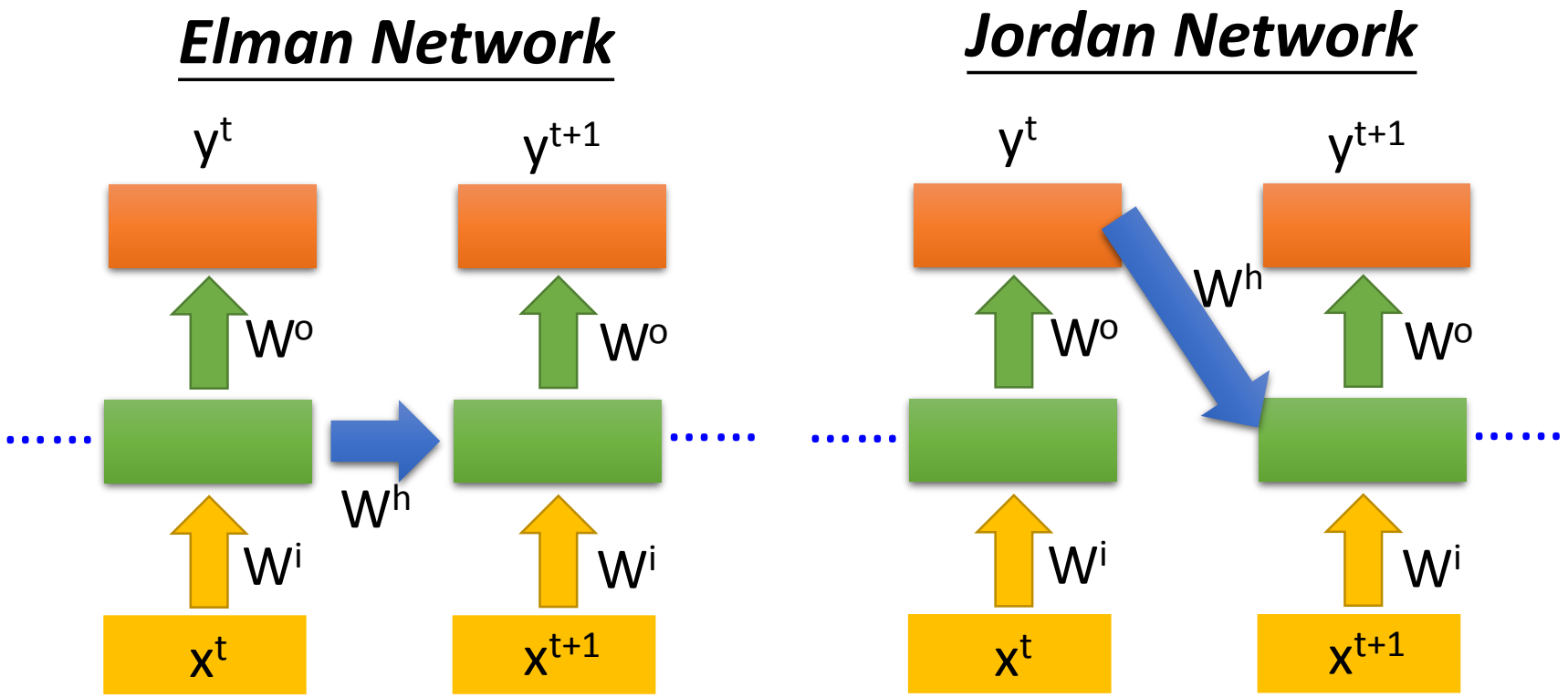
隐藏层可以不止一层

Of course it can be deep …



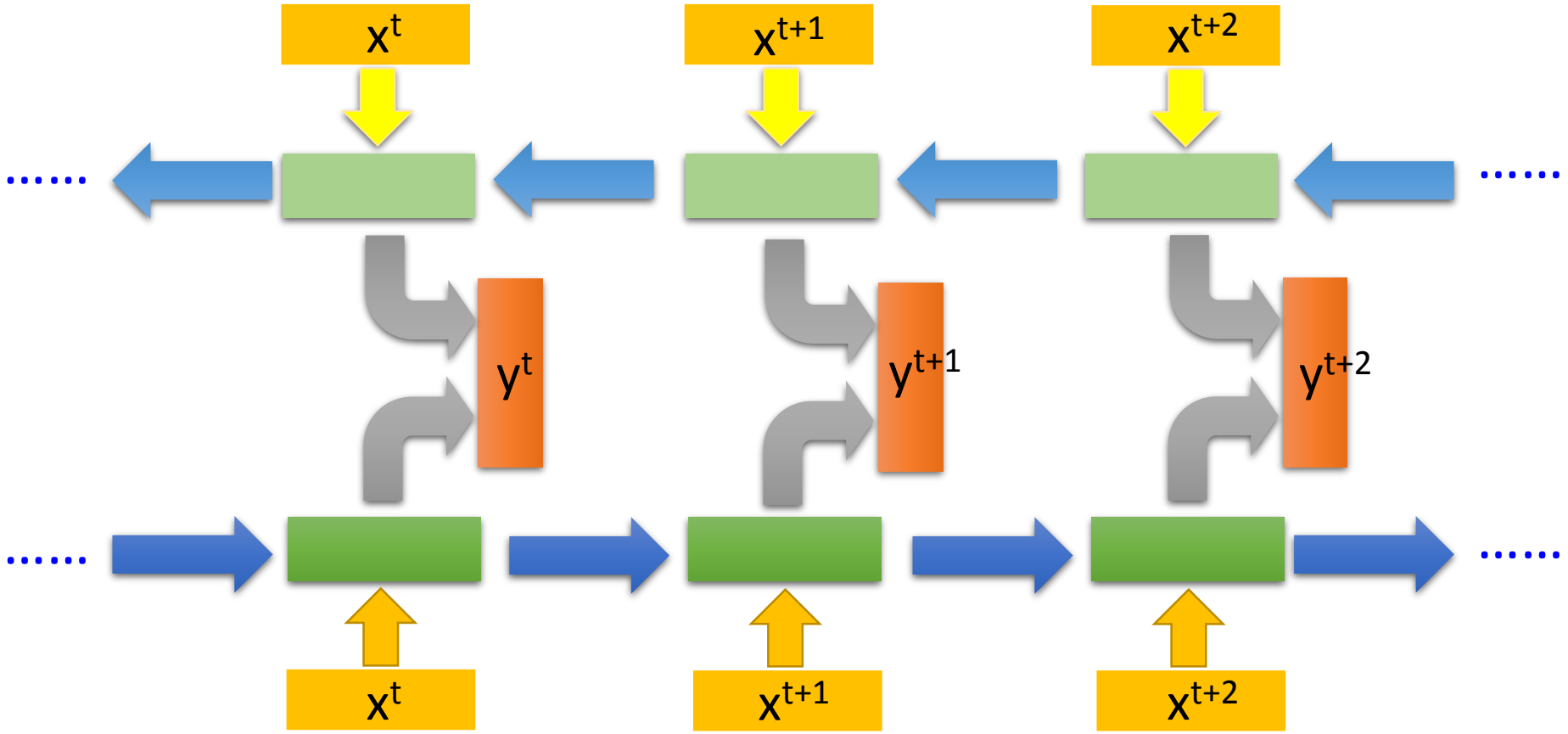
Jordan Network由于直接将前1时刻的输出与当前时刻的输入一起放入隐藏层，传说效果会好于E-RNN。我们人干预的位置也就在每一时刻的输出，或者序列结束时刻的输出。

Elman Network & Jordan Network

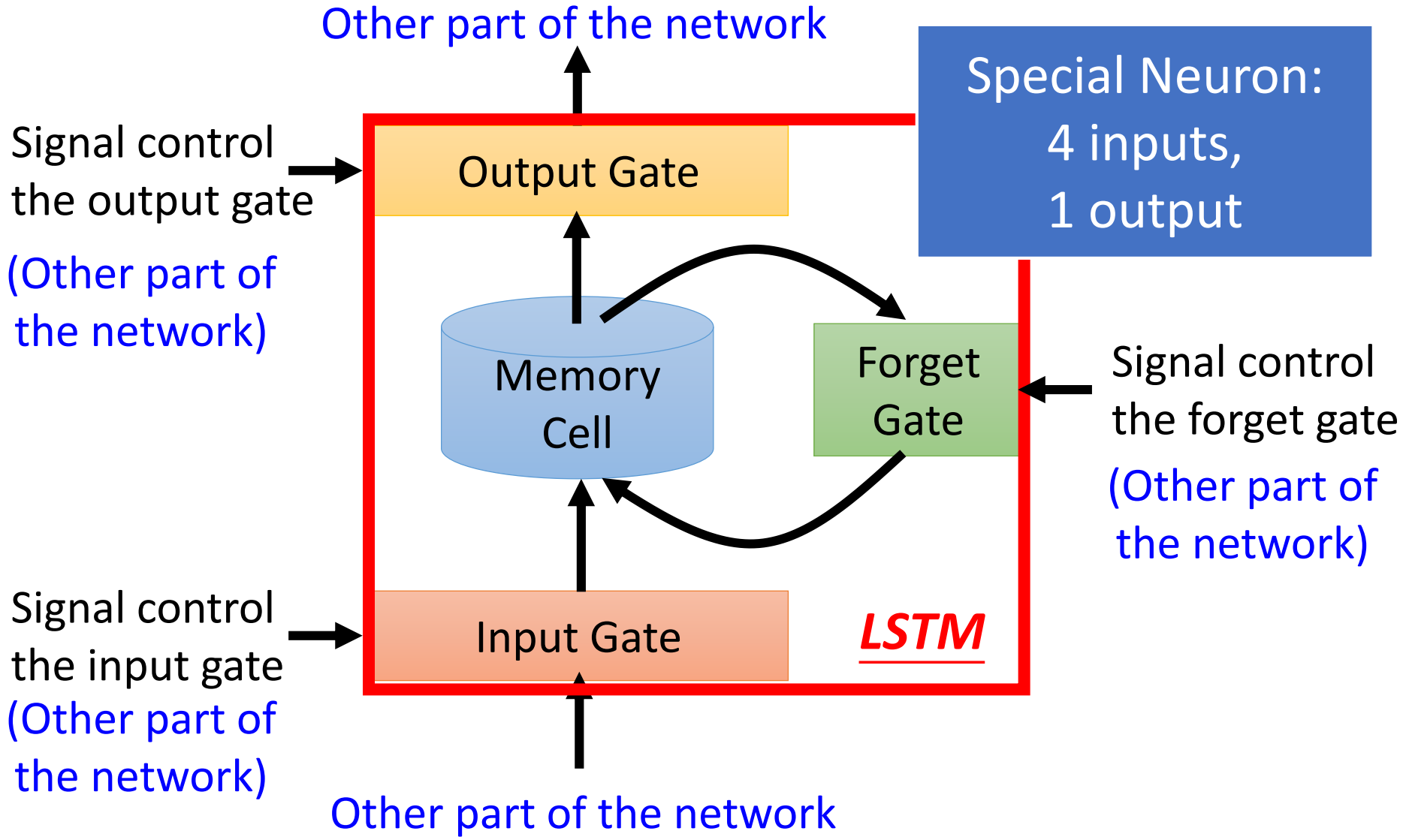


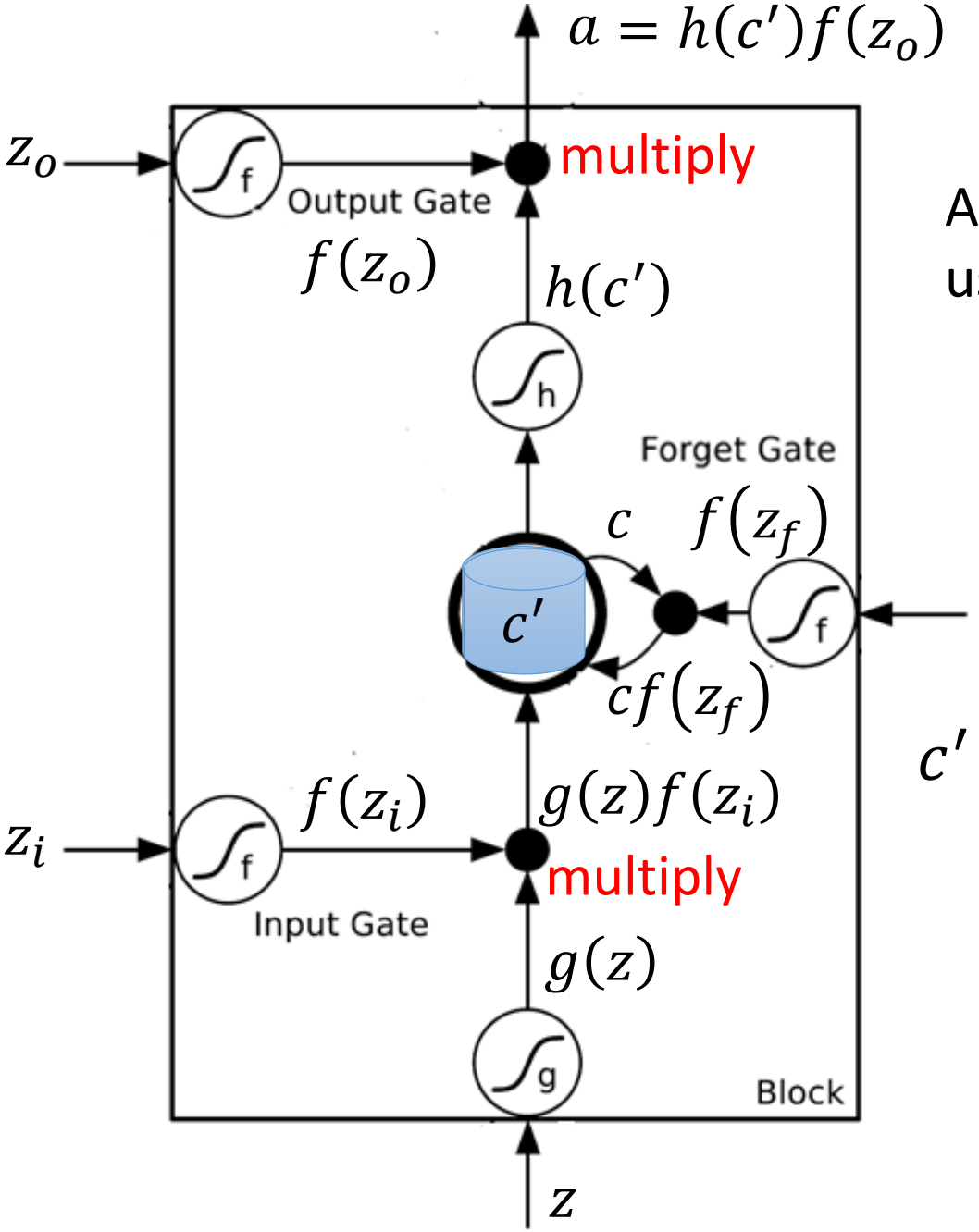
正向训练一遍，反向训练一遍，将同一时刻正反向隐层连接到输出层做预测。

Bidirectional RNN



Long Short-term Memory (LSTM)





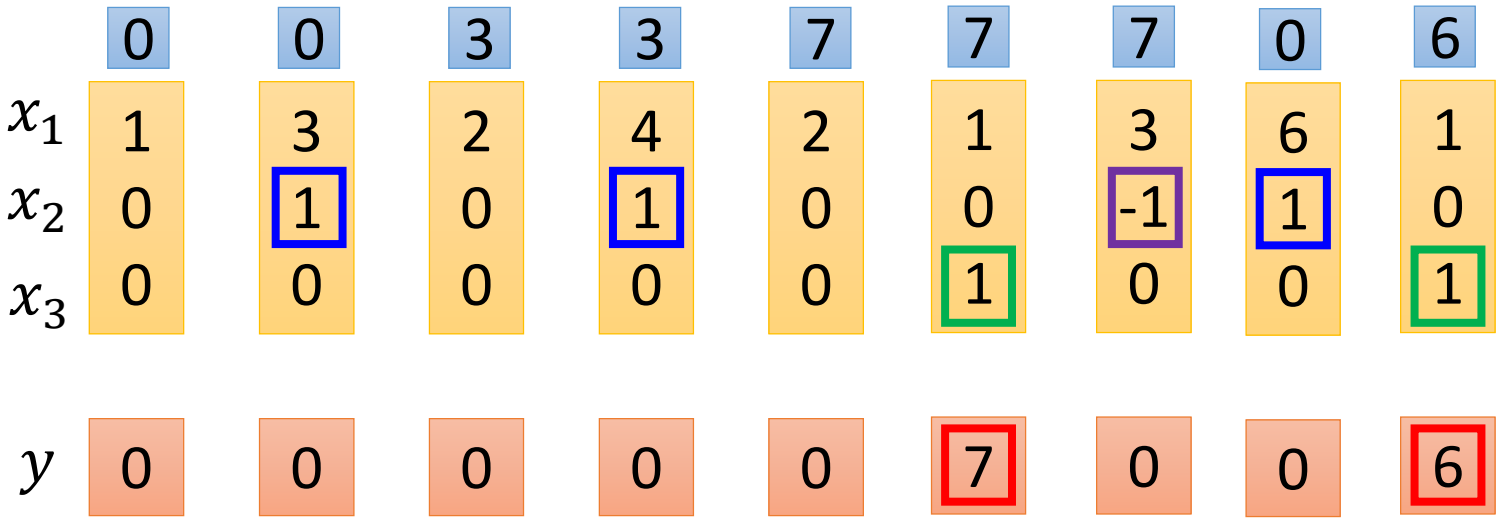
Activation function f is usually a sigmoid function

Between 0 and 1

Mimic open and close gate

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

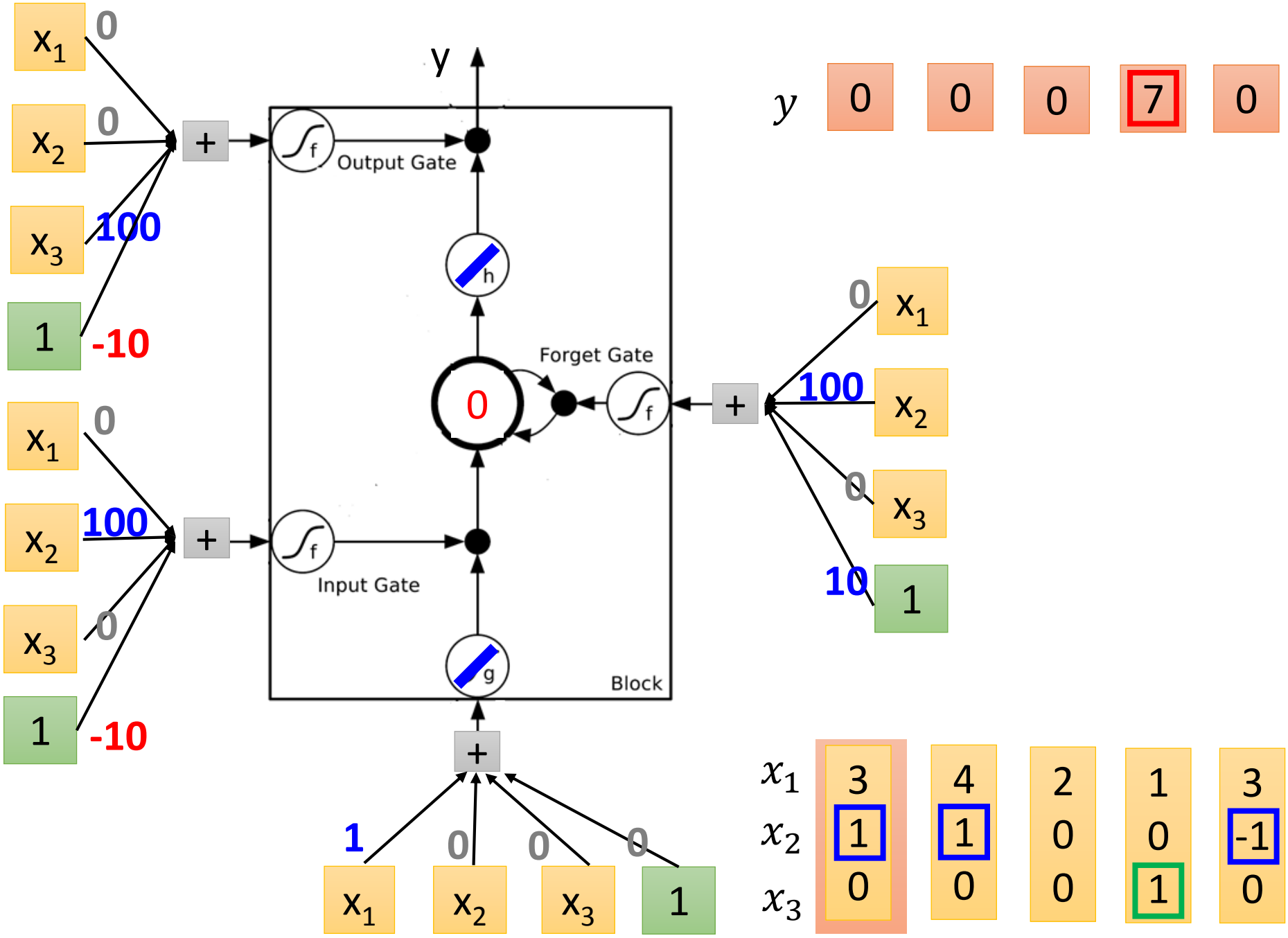
LSTM - Example



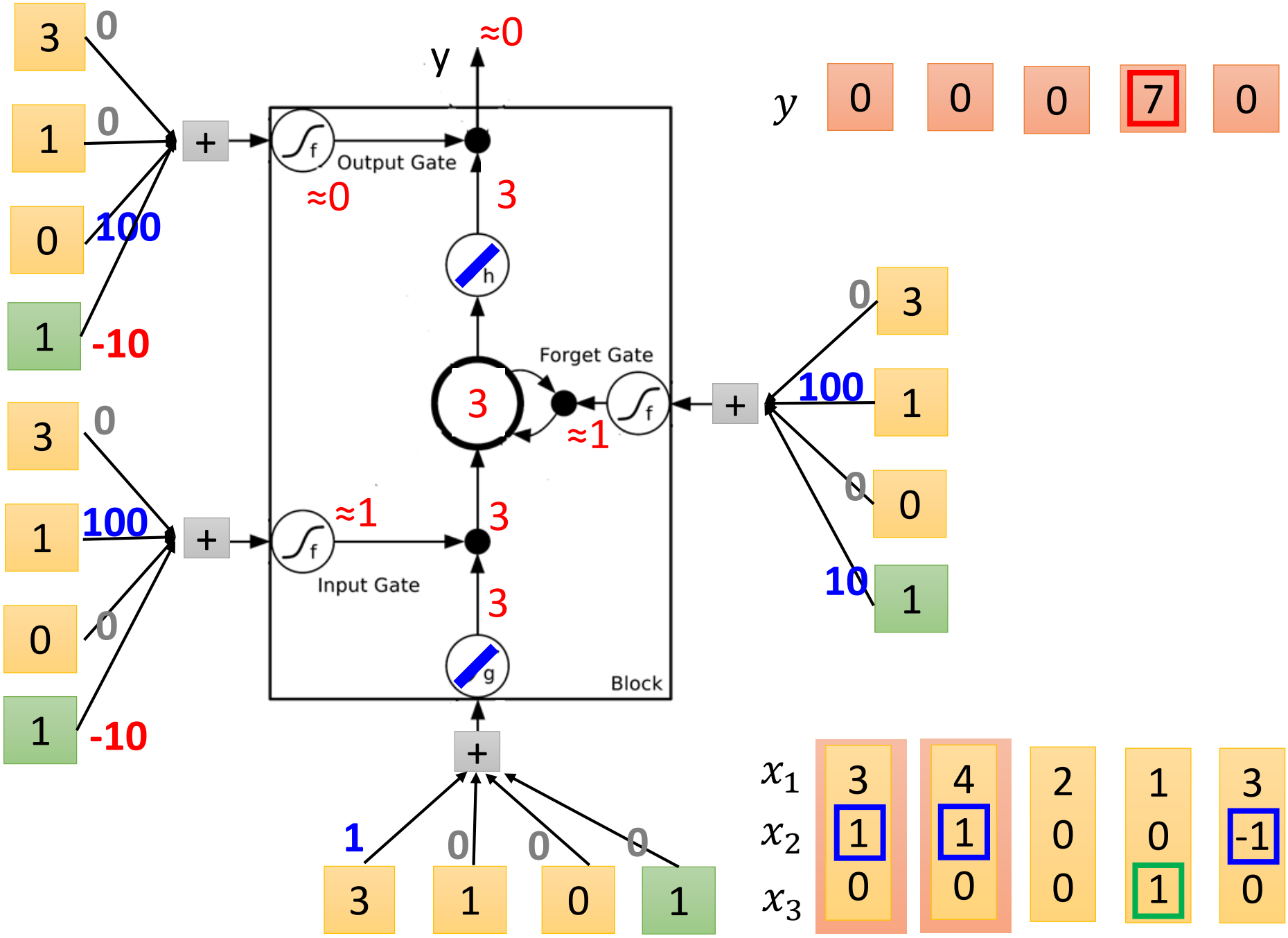
When x_2 = 1, add the numbers of x_1 into the memory

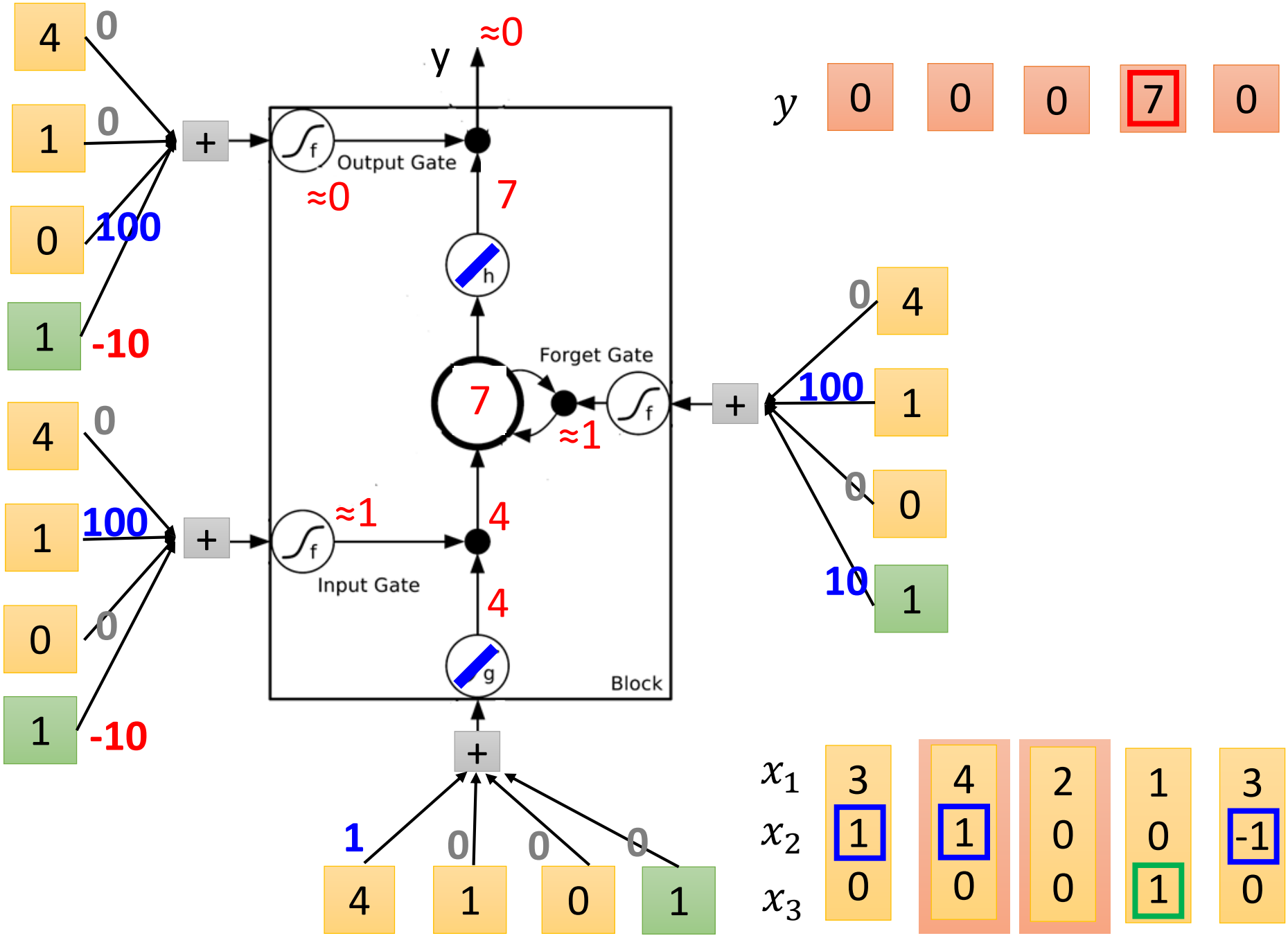
When x_2 = -1, reset the memory

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

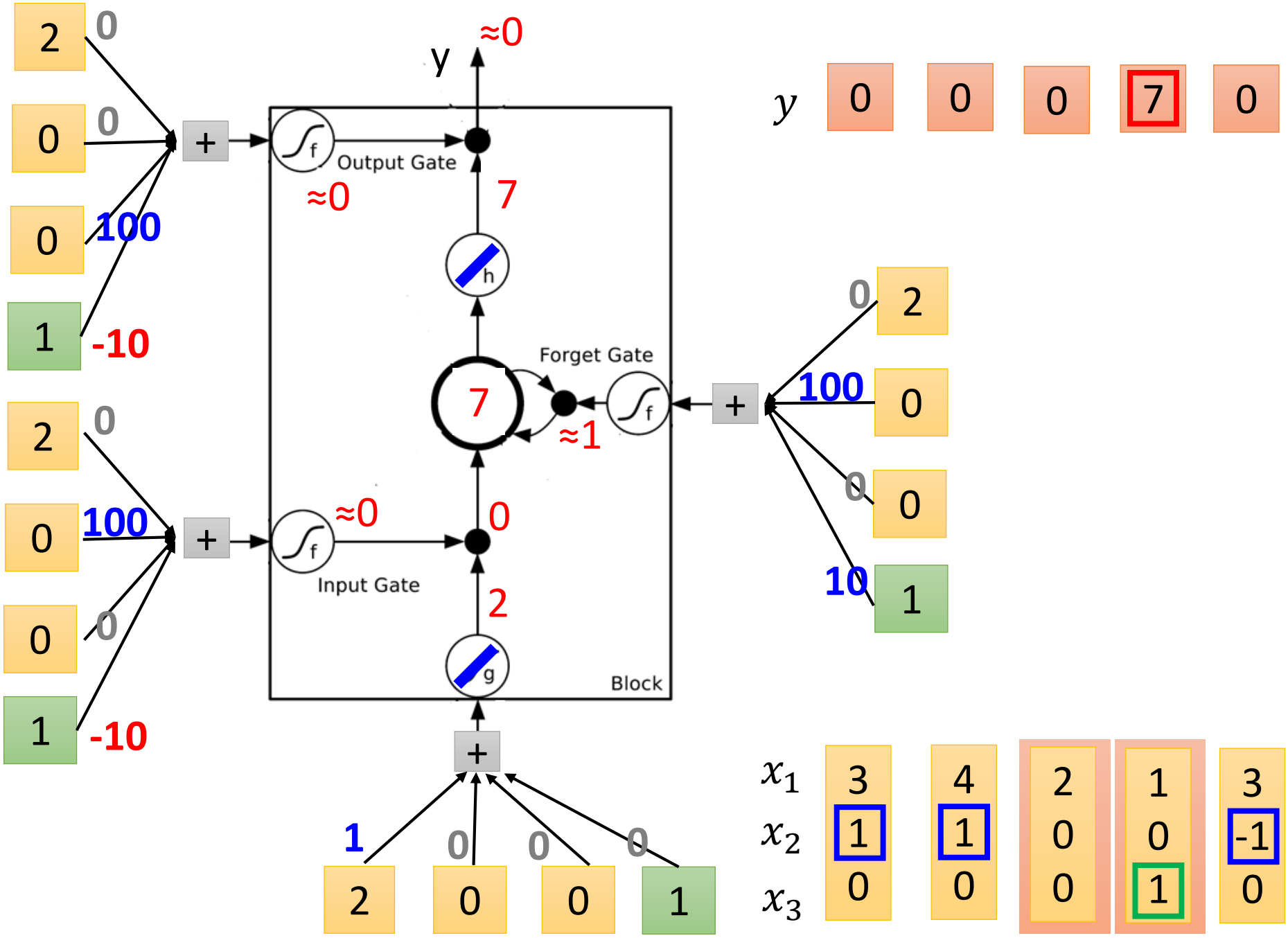


y
Output Gate
Forget Gate
Input Gate
Block
0
x1
x2
x3
1
0
0
100
-10
0
100
0
-10
0
100
0
10
1
0
0
0
1
y
0
0
0
7
0
x1
x2
x3
3
1
0
4
1
0
2
0
0
1
0
1
3
-1
0

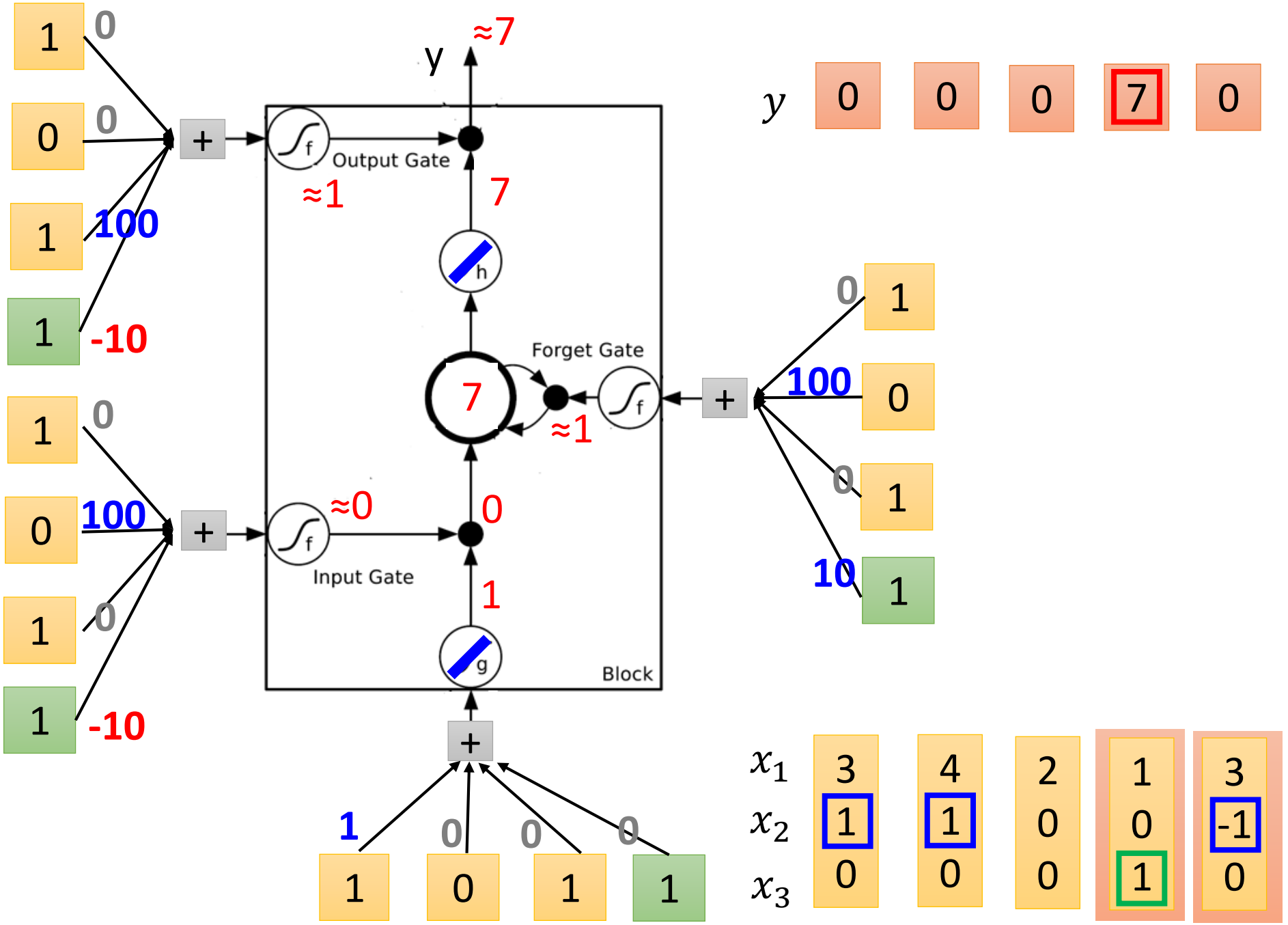

y
≈0
Output Gate
≈0
3
Forget Gate
3
≈1
Input Gate
≈1
3
3
Block
3
0
1
100
0
1
-10
3
0
1
100
0
1
-10
0
3
100
1
0
0
10
1
1
0
0
0
3
1
0
1
y 0 0 0 7 0
x_1 3 4 2 1 3
x_2 1 1 0 0 -1
x_3 0 0 0 1 0



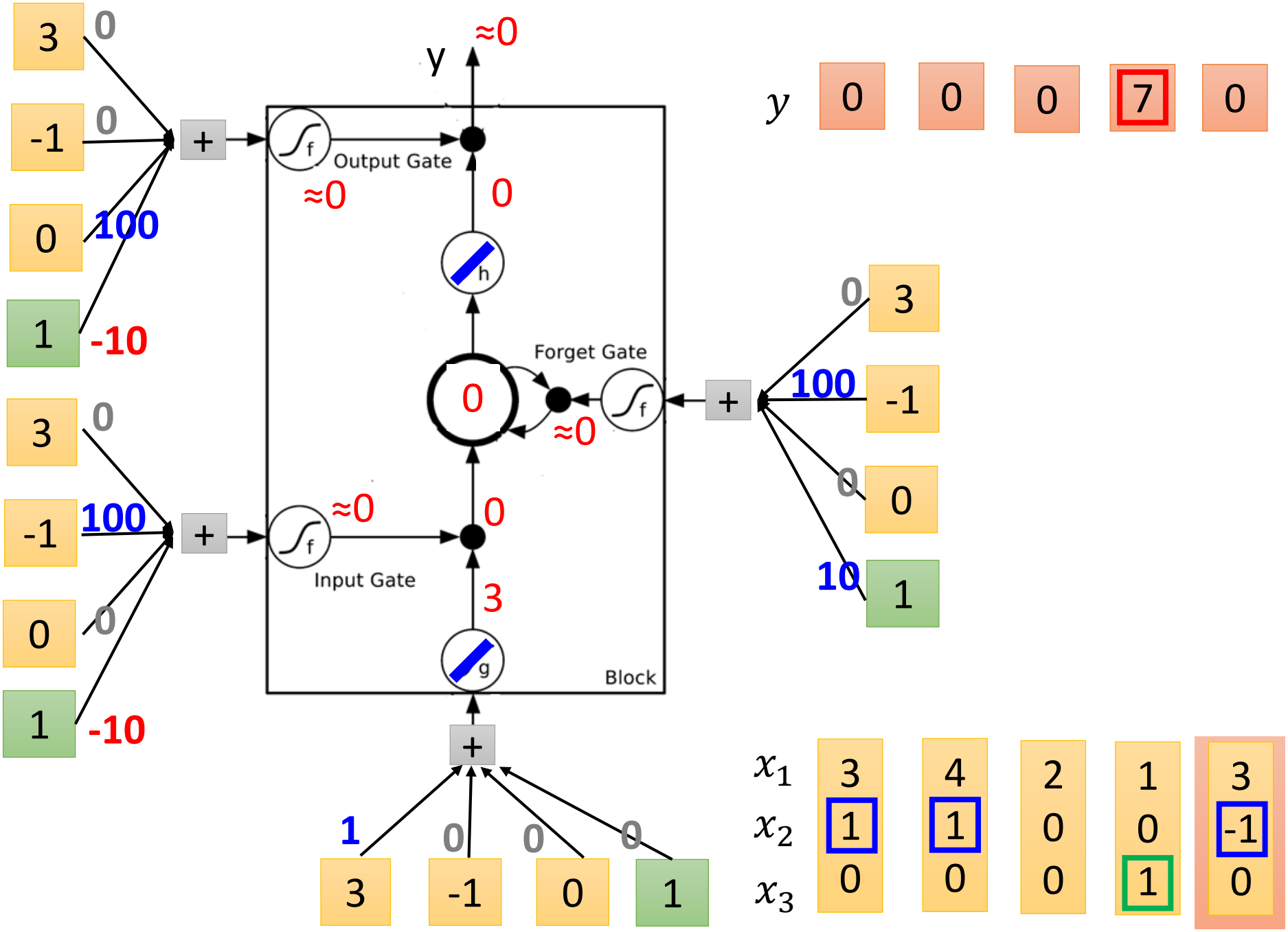
y
≈0
Output Gate
≈0
7
h
Forget Gate
7
≈1
f
Input Gate
≈1
4
4
g
Block
+
4
1
0
1
0
0
100
-10
4
1
0
1
0
100
0
-10
0
4
1
0
1
100
0
10
1
0
0
0
7
0
x_1
x_2
x_3
3
1
0
4
1
0
2
0
0
1
0
1
3
-1
0



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



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

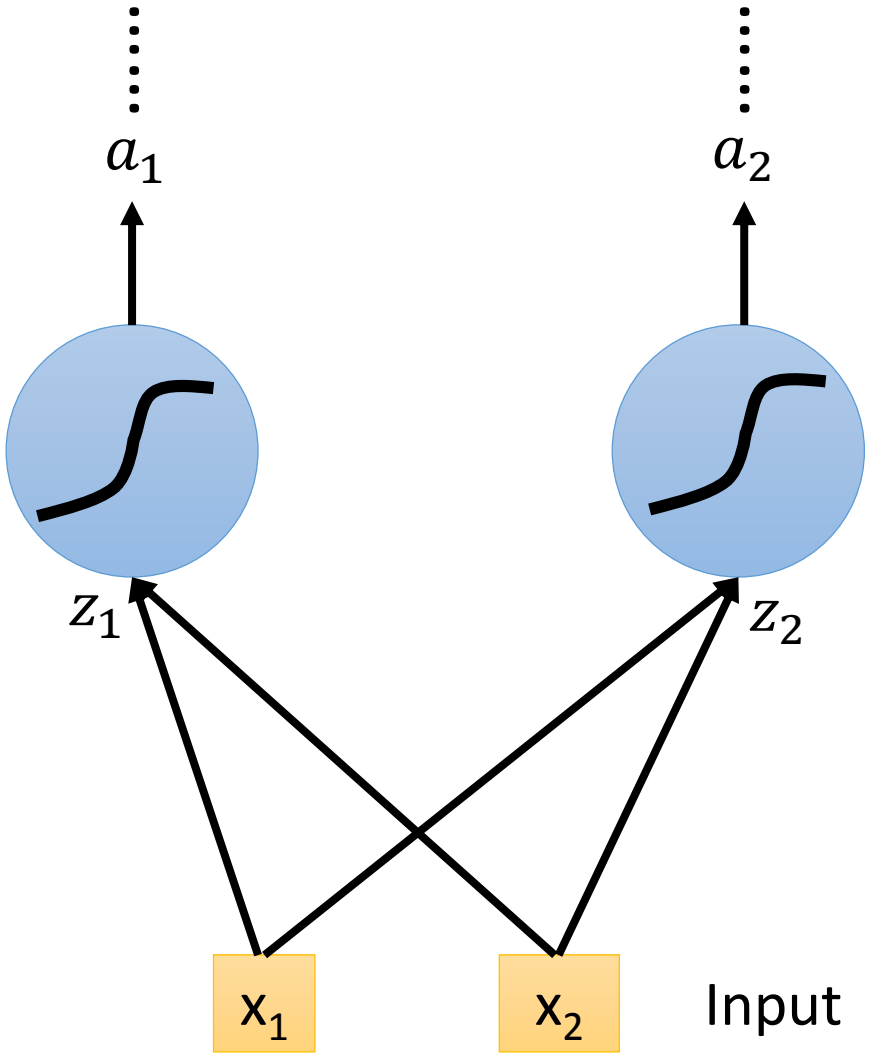


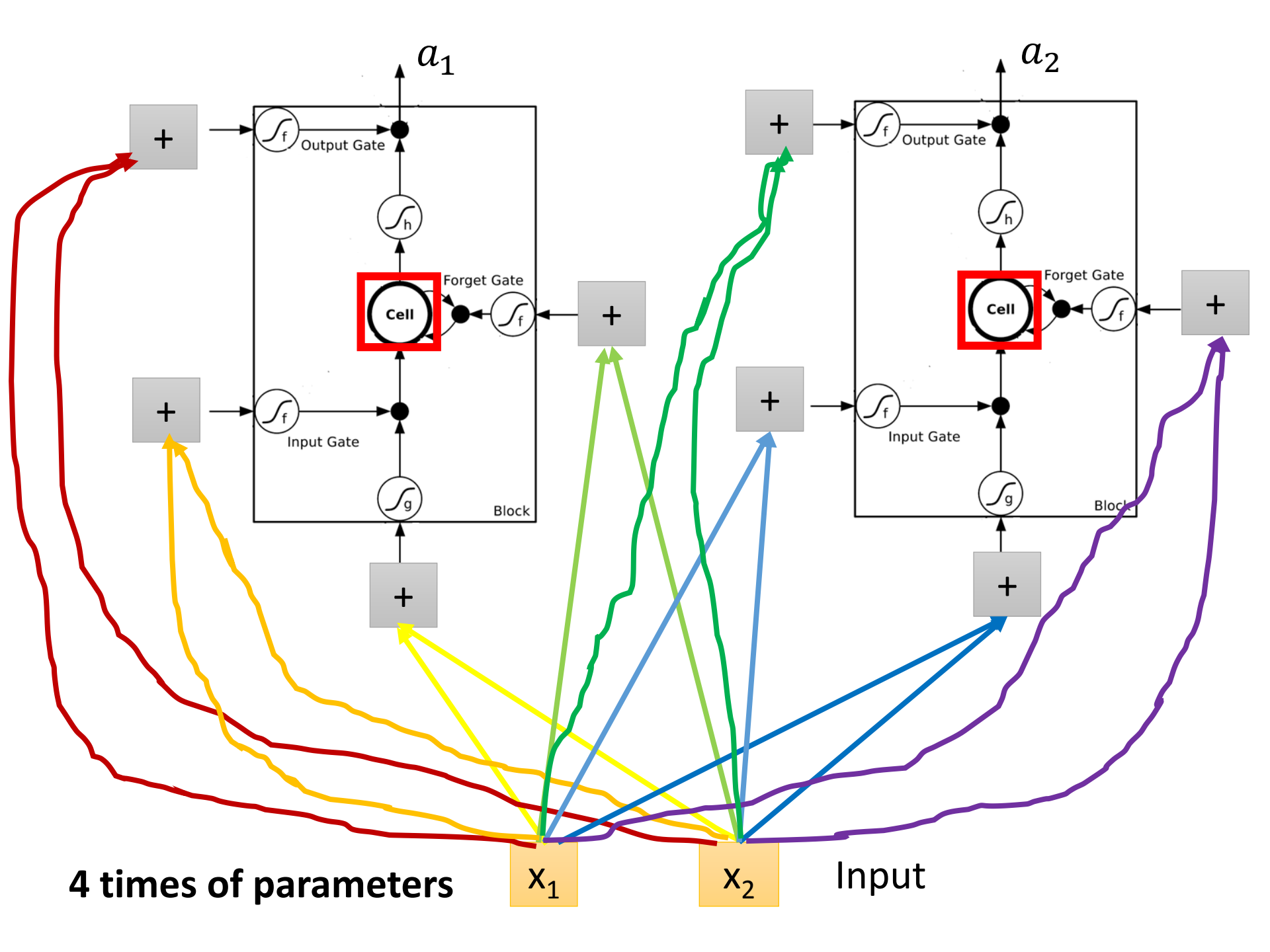
y	0	0	0	7	0

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

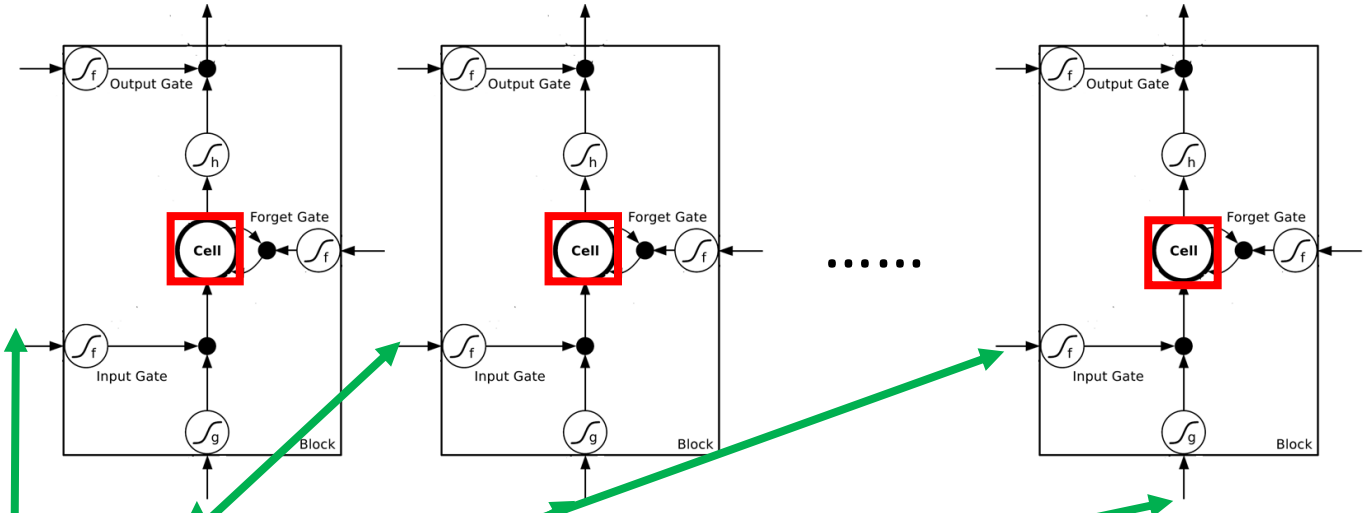



a_1
a_2
Output Gate
Forget Gate
Input Gate
Cell
Block
x_1
x_2
Input
4 times of parameters

LSTM

c^{t-1}

vector

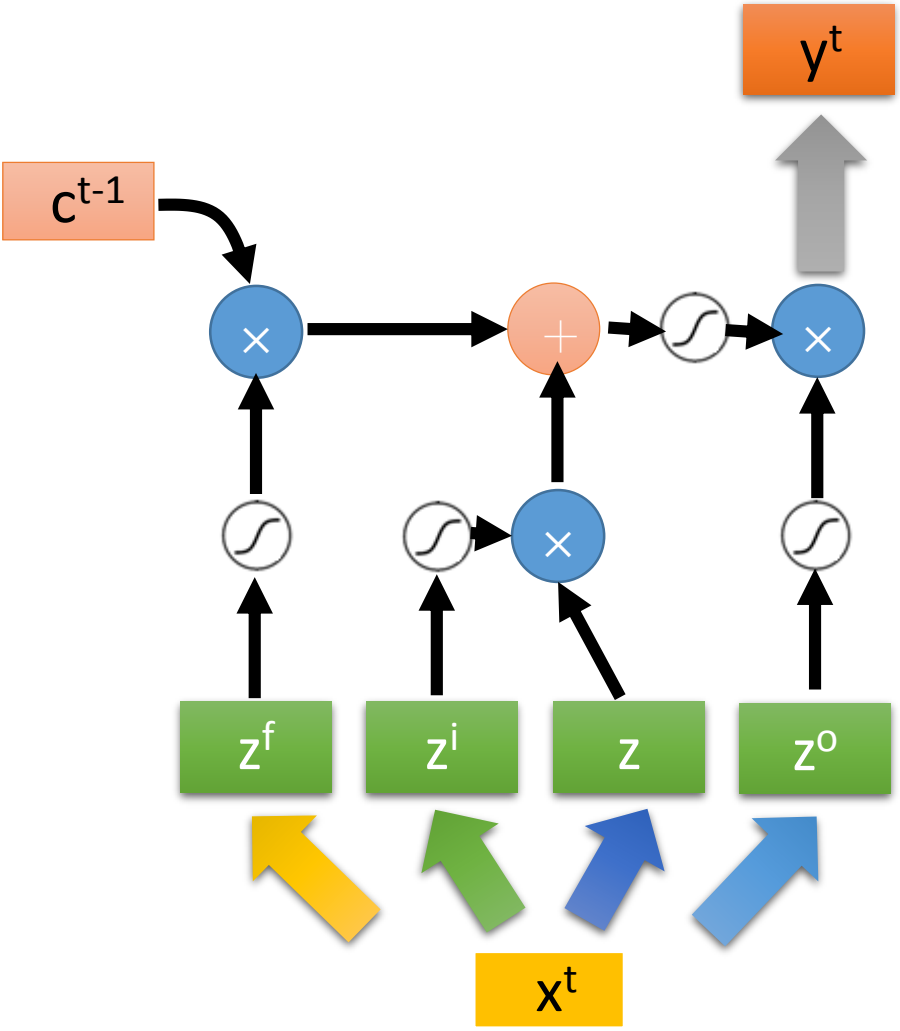


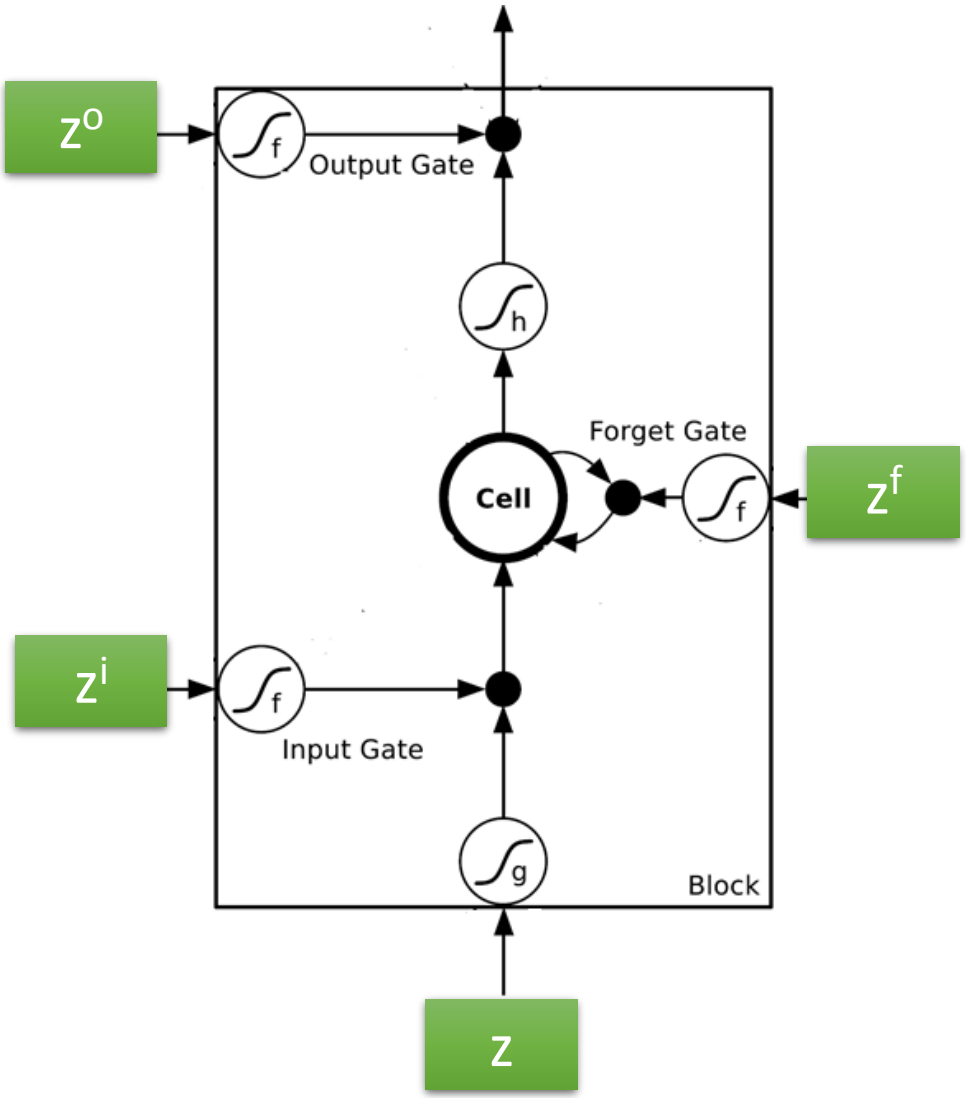
z^f z^i z z^o

4 vectors

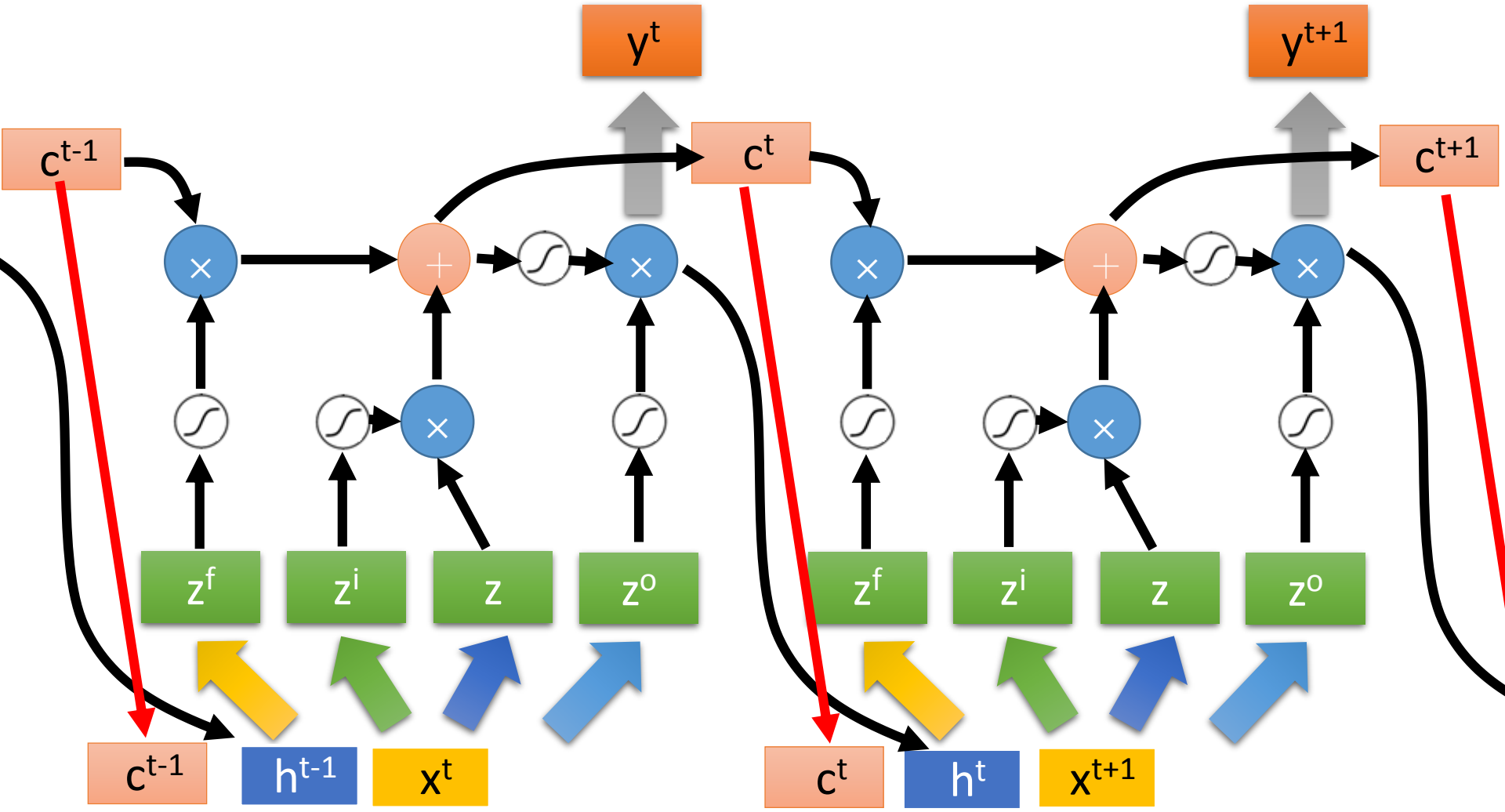
x^t

LSTM

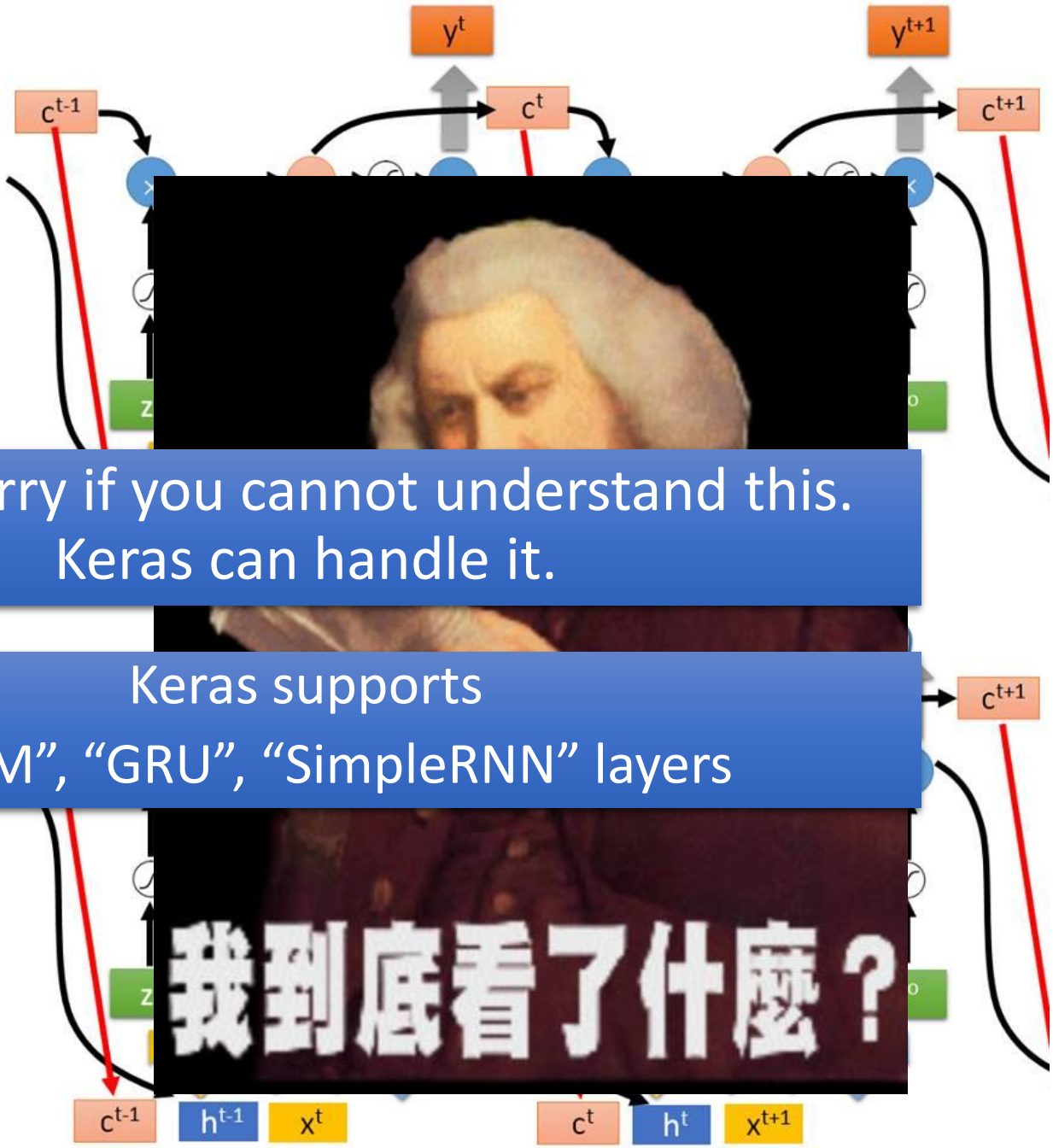

y^t
c^{t-1}
z^f
z^i
z
z^o
x^t


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

LSTM


Extension: “peephole”
y^t
y^{t+1}
c^{t-1}
c^t
c^{t+1}
z^f
z^i
z
z^o
c^{t-1}
h^{t-1}
x^t
c^t
h^t
x^{t+1}

Multiple-layer LSTM



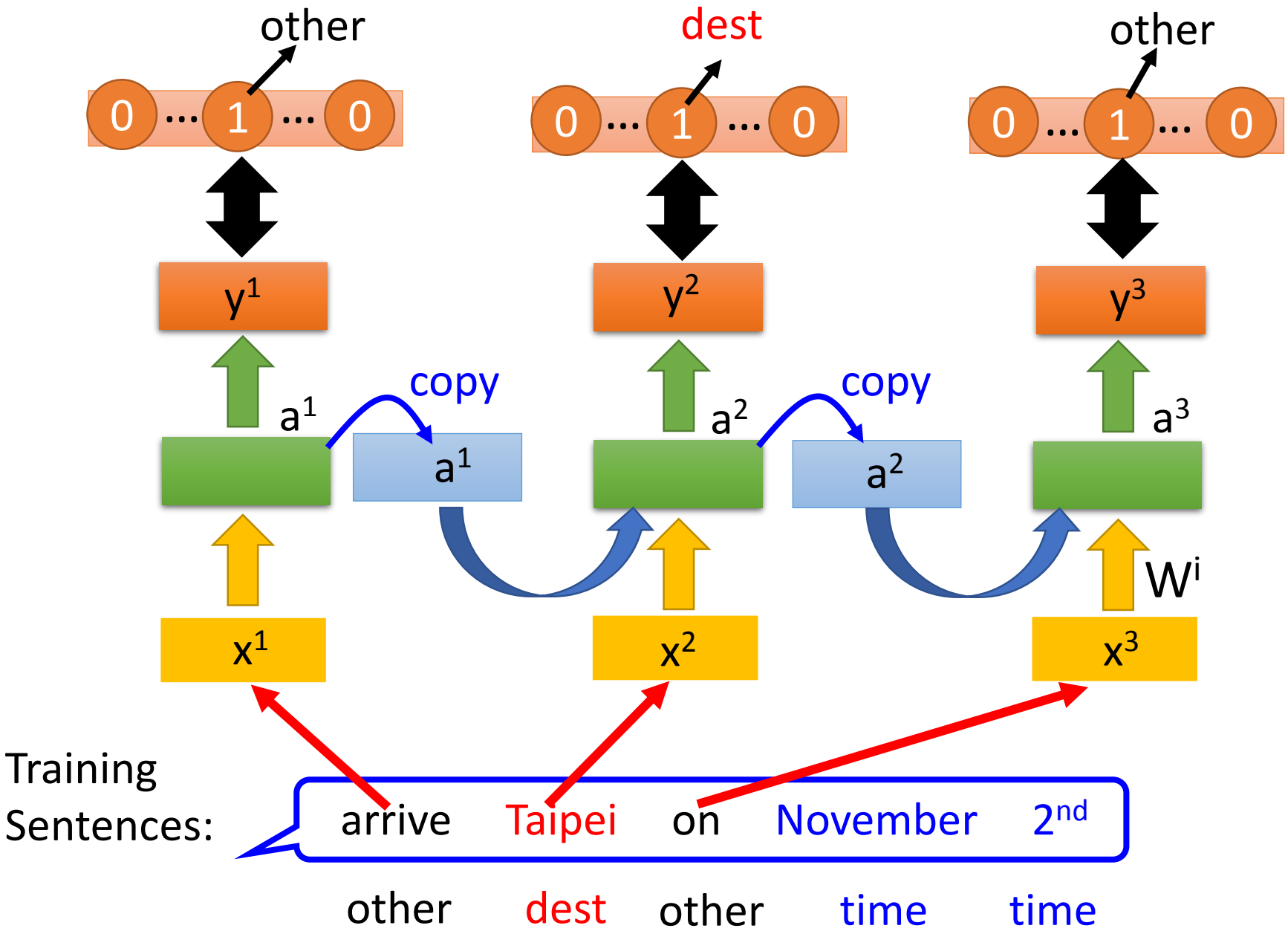
Don’t worry if you cannot understand this.
Keras can handle it.

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

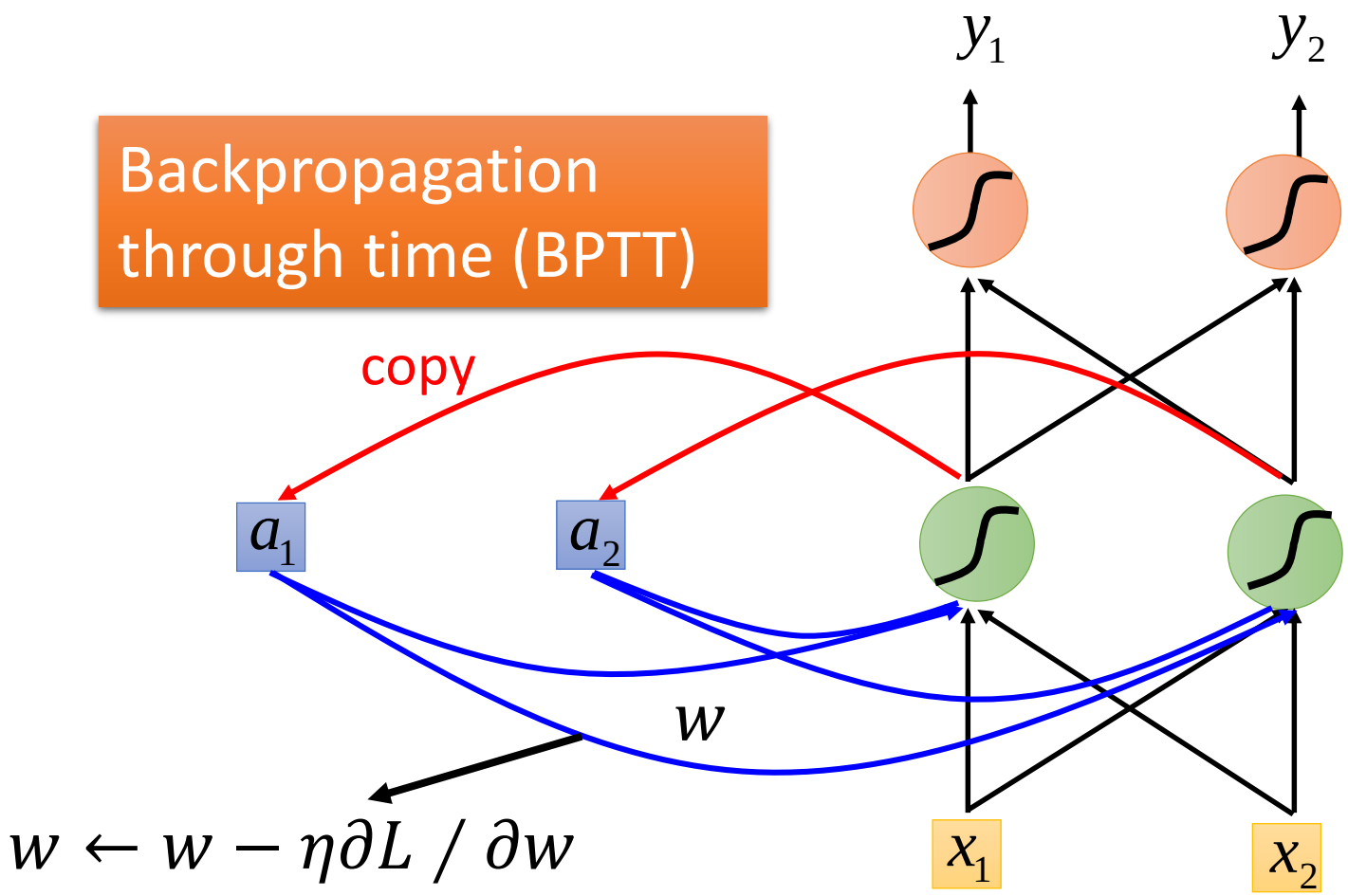
This is quite standard now.

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

Learning Target



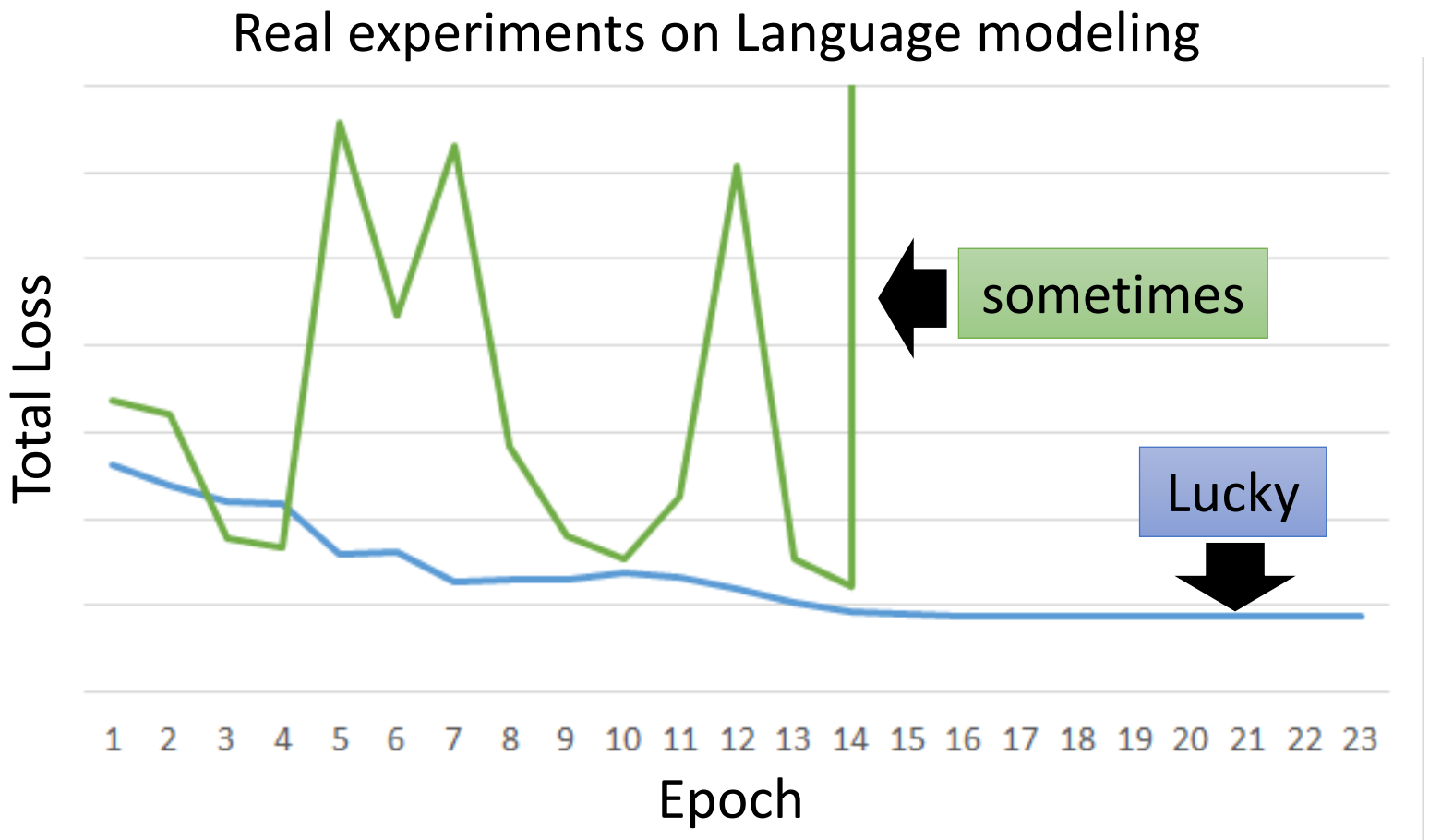
Learning


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

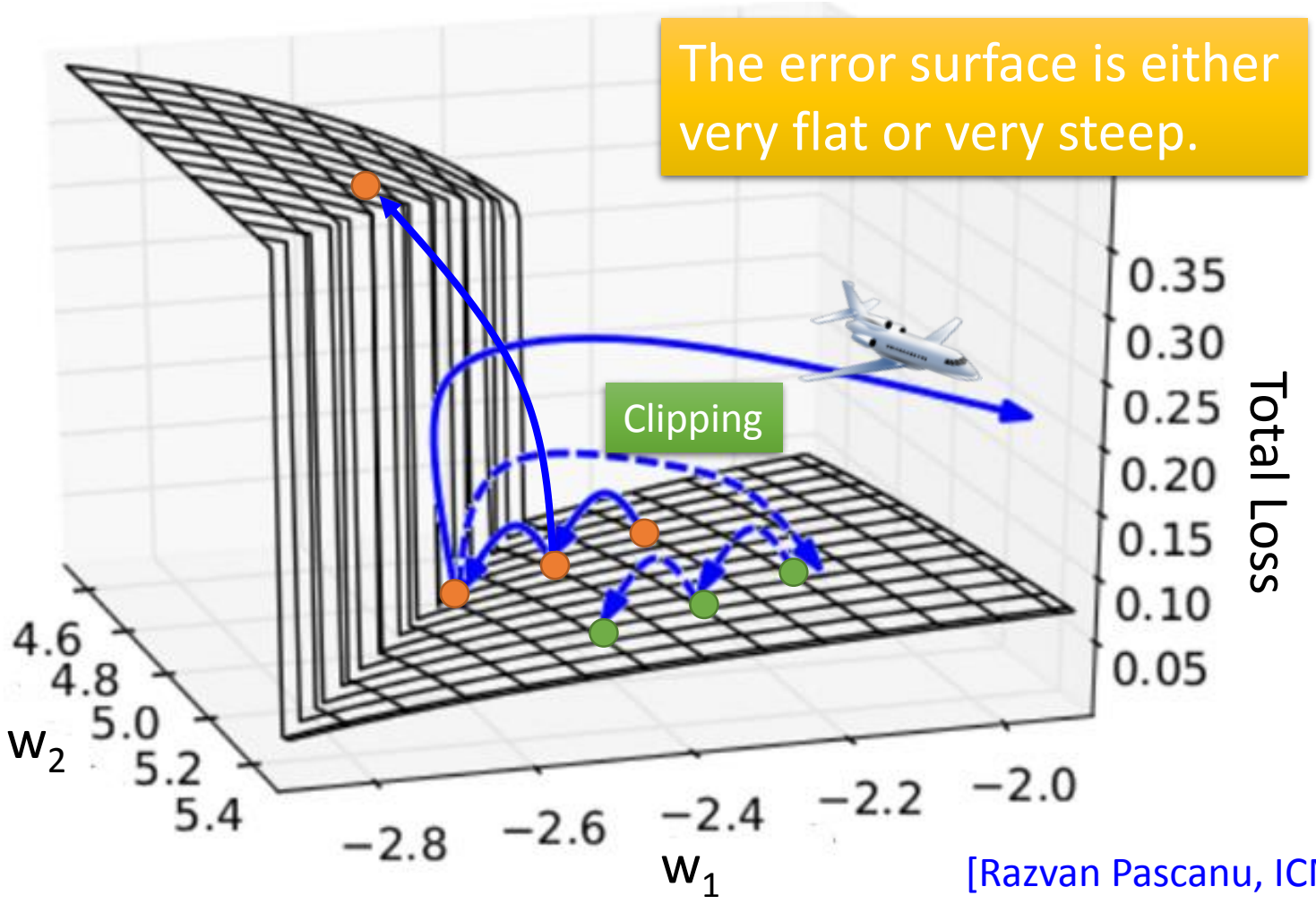
感謝 曾柏翔 同學
提供實驗結果

Unfortunately ……

- RNN-based network is not always easy to learn



The error surface is rough.



[Razvan Pascanu, ICML'13]

Why?

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

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

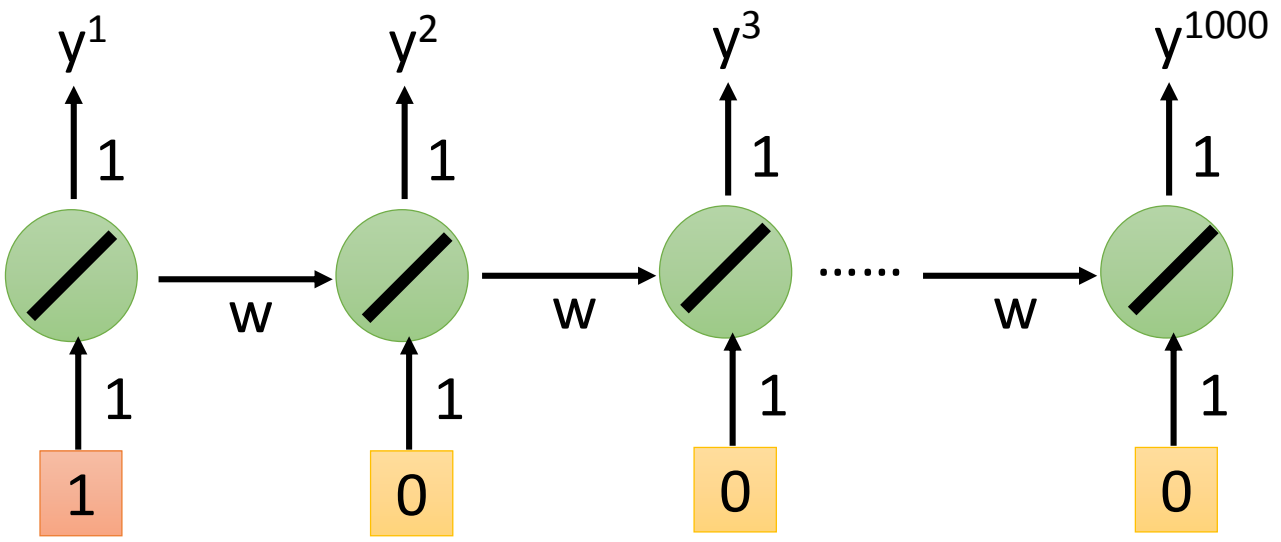
Large $\partial L/\partial w$ → Small Learning rate?

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

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

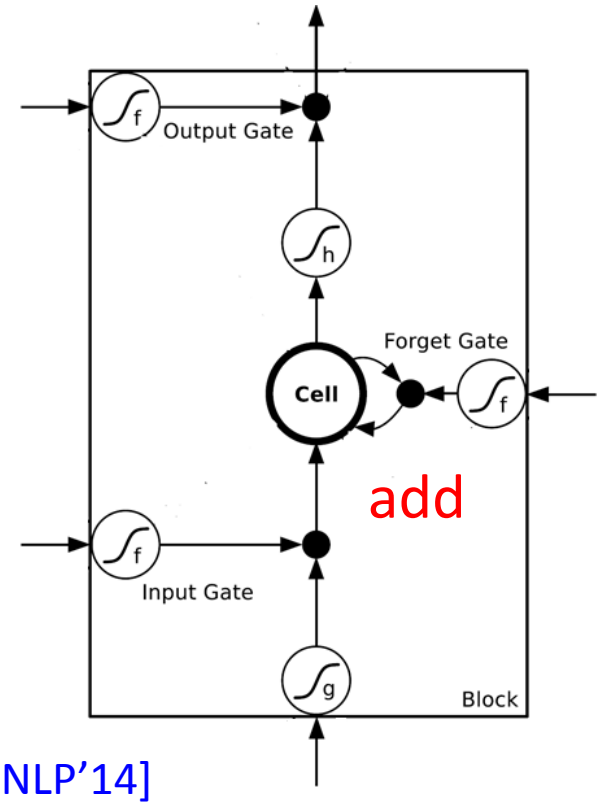
small $\partial L/\partial w$ → Large Learning rate?

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



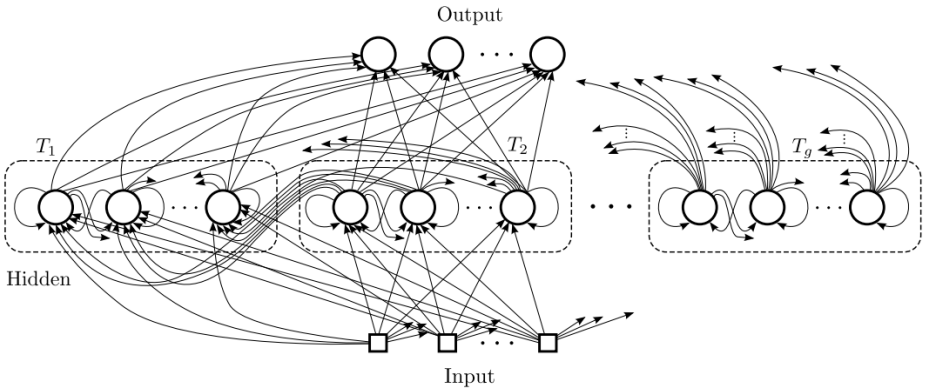
No Gradient vanishing
(If forget gate is opened.)

Gated Recurrent Unit (GRU): simpler than LSTM

[Cho, EMNLP'14]

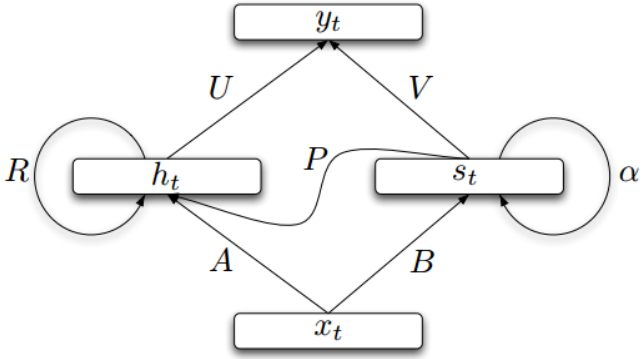
Helpful Techniques

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]

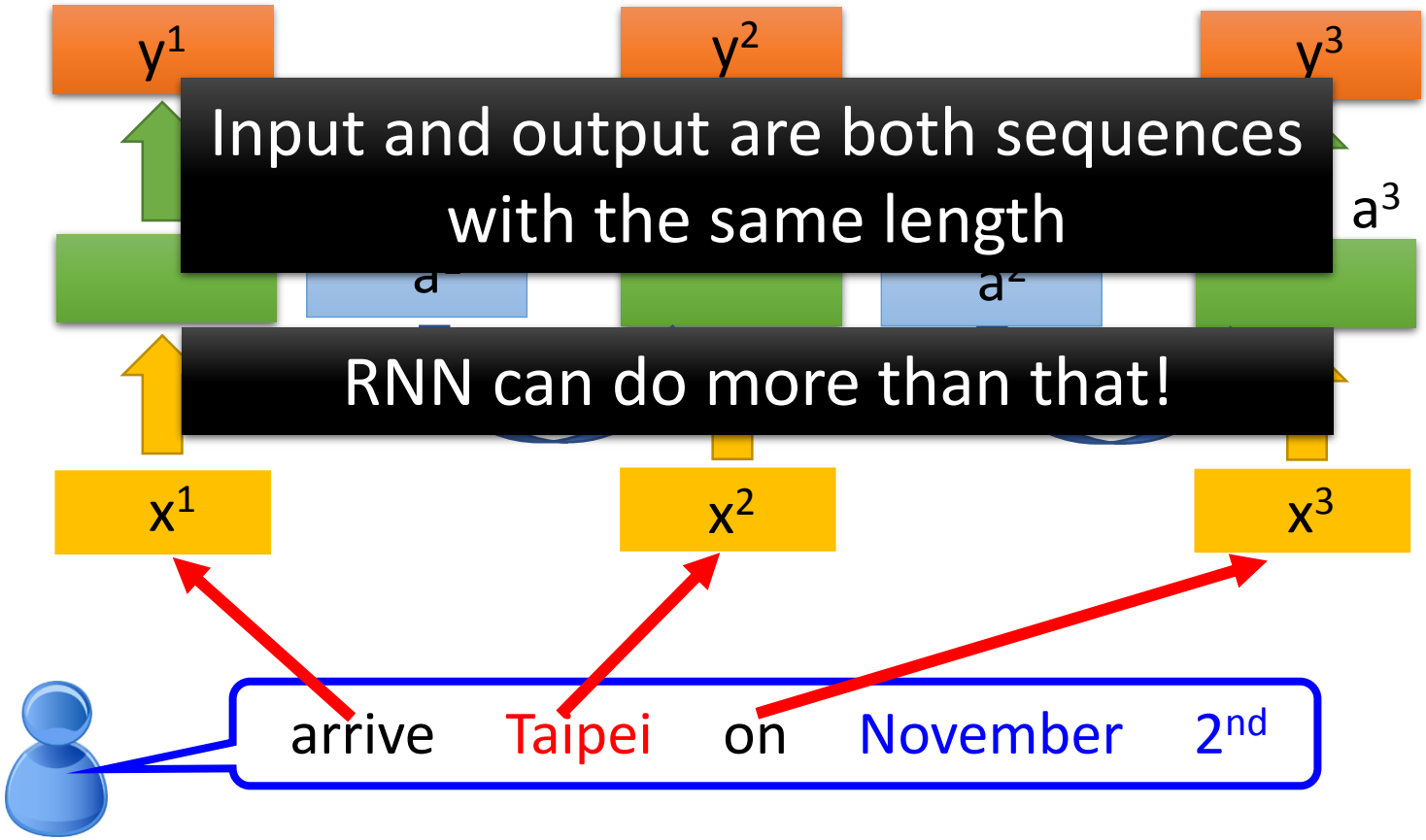
- Outperform or be comparable with LSTM in 4 different tasks

More Applications ……

Probability of "arrive" in each slot

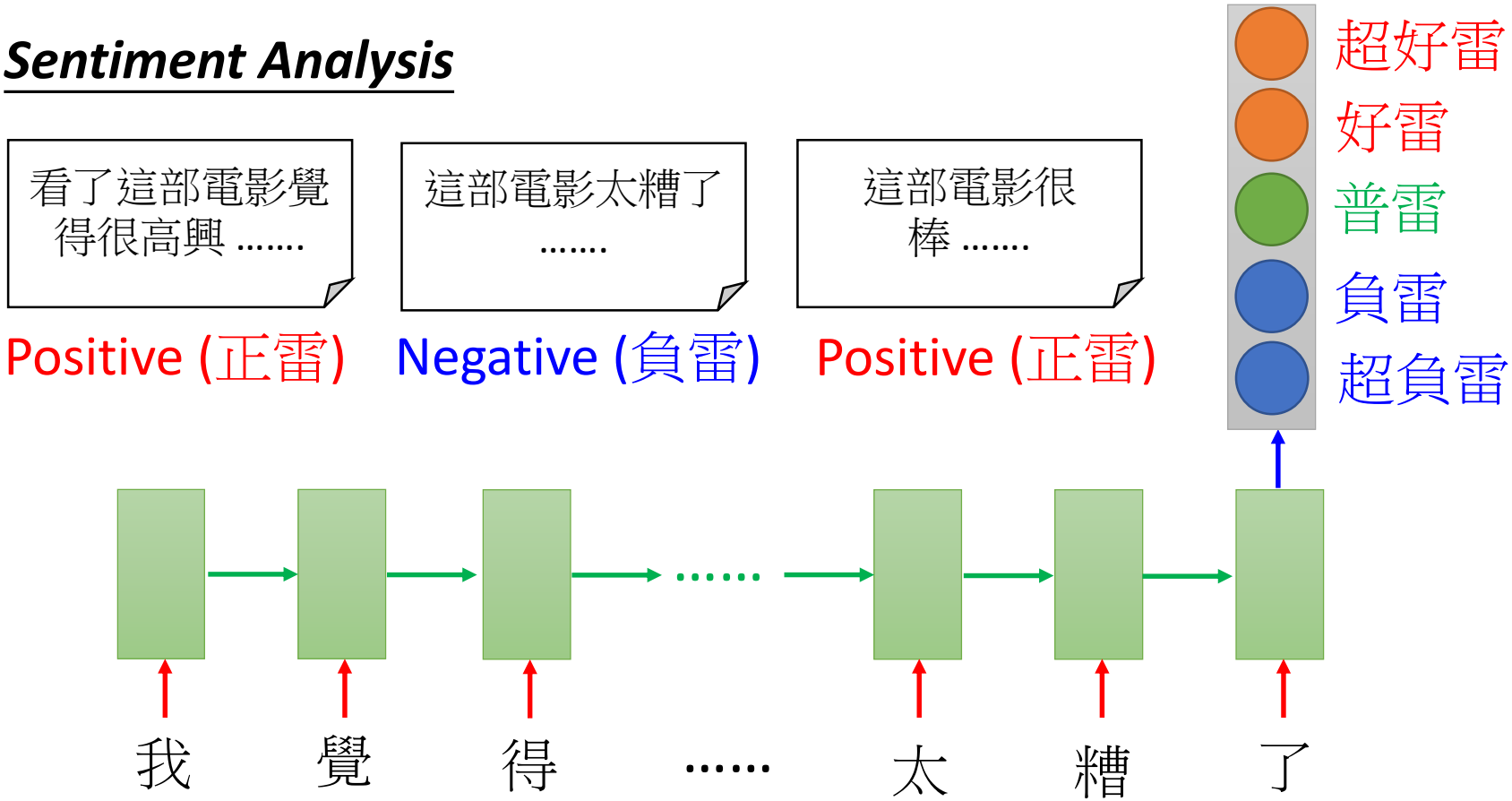
Probability of "Taipei" in each slot

Probability of "on" in each slot



Many to one

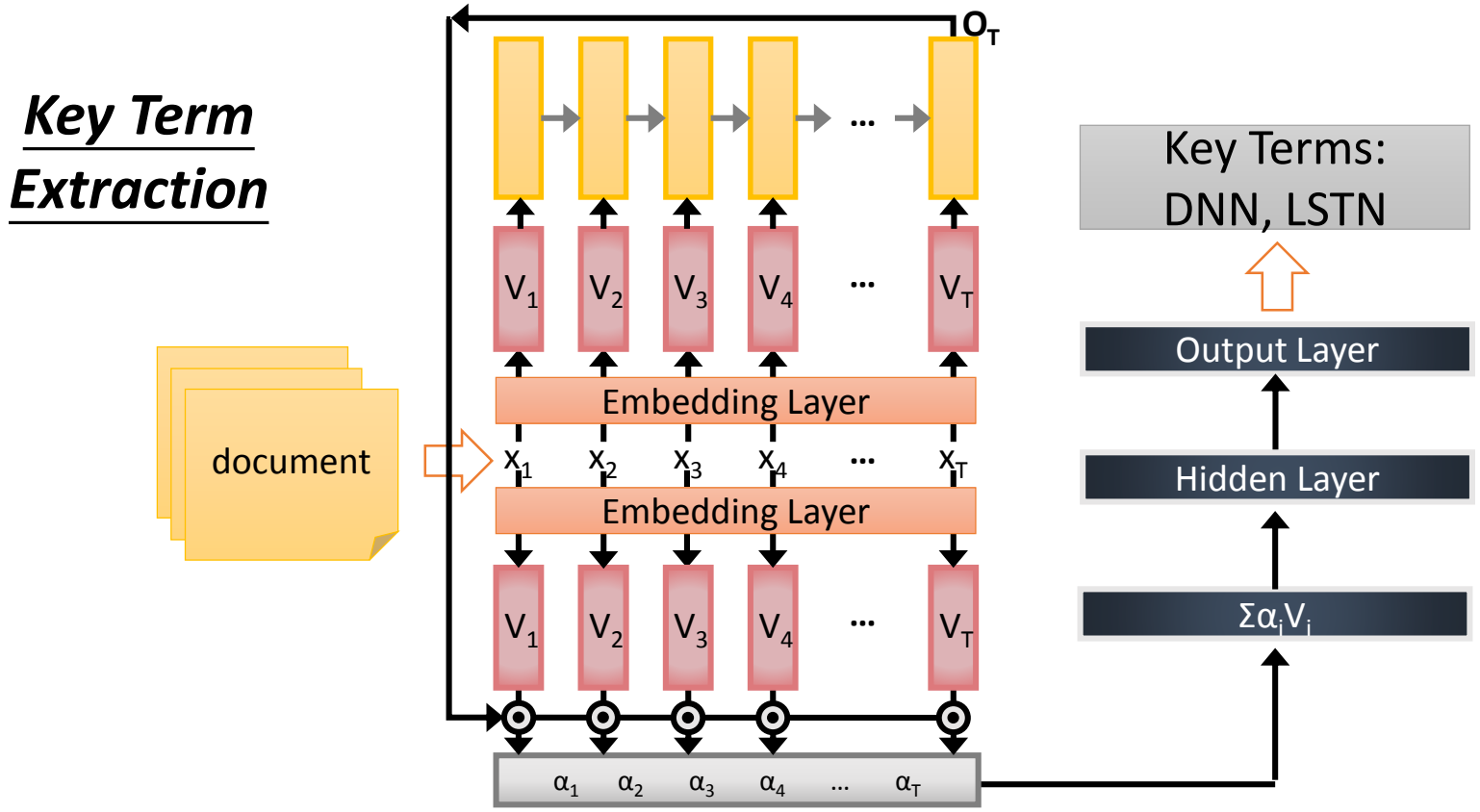
- Input is a vector sequence, but output is only one vector



Many to one

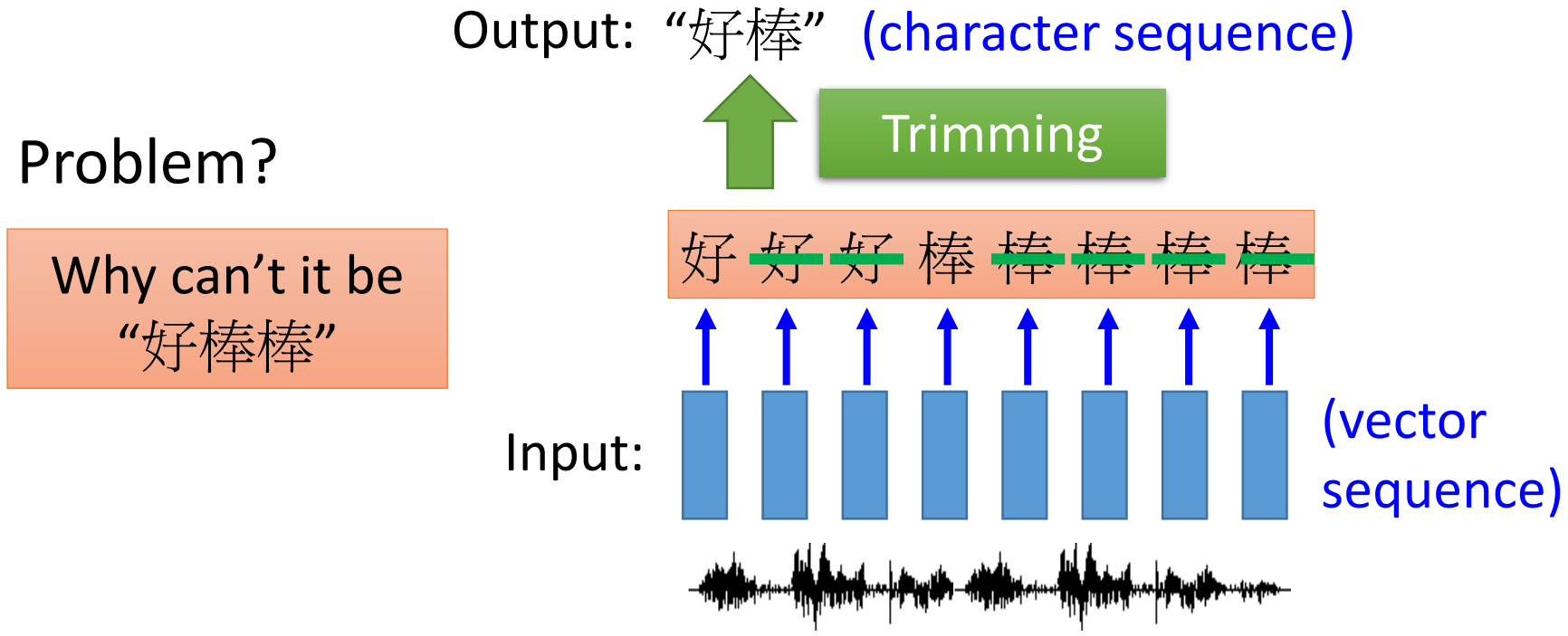
[Shen & Lee, Interspeech 16]

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



Many to Many (Output is shorter)

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

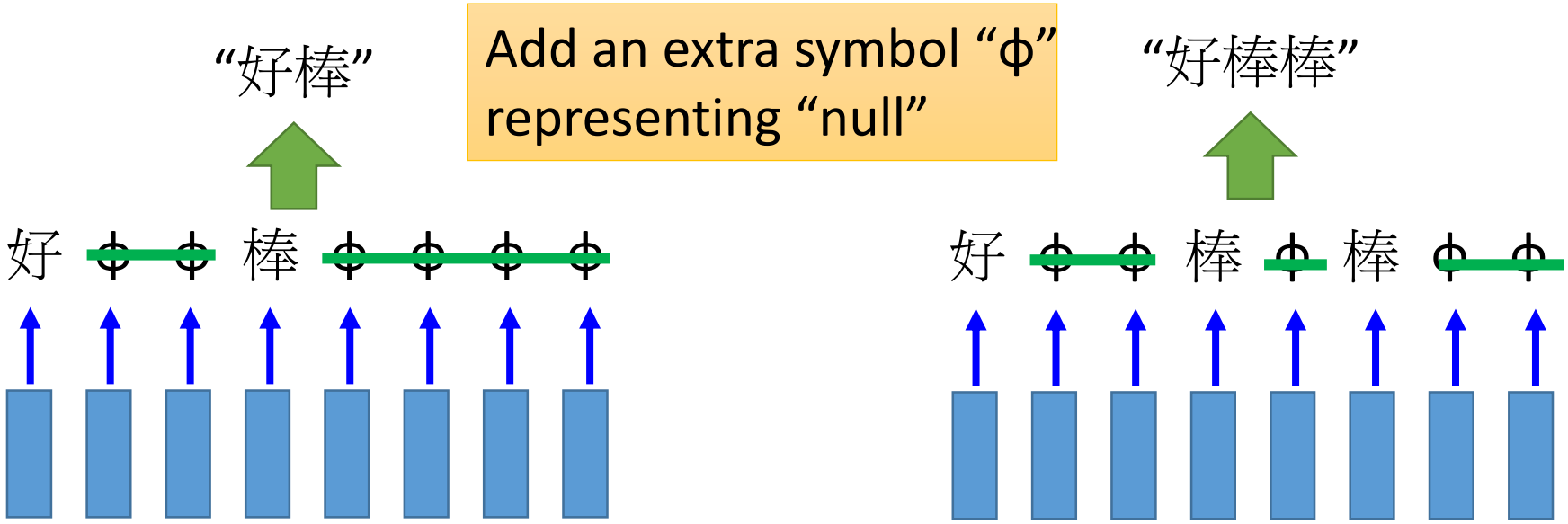


Problem?

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]



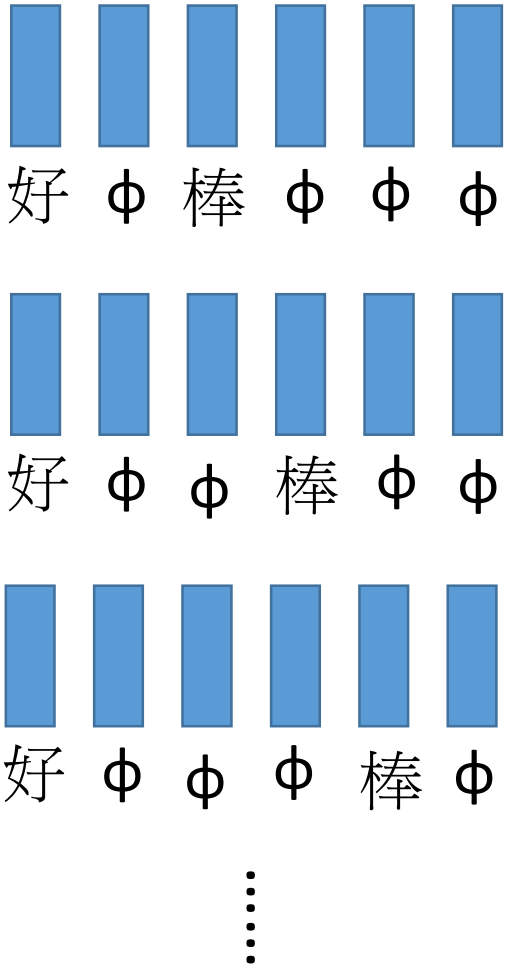
Many to Many (Output is shorter)

- CTC: Training

Acoustic Features:

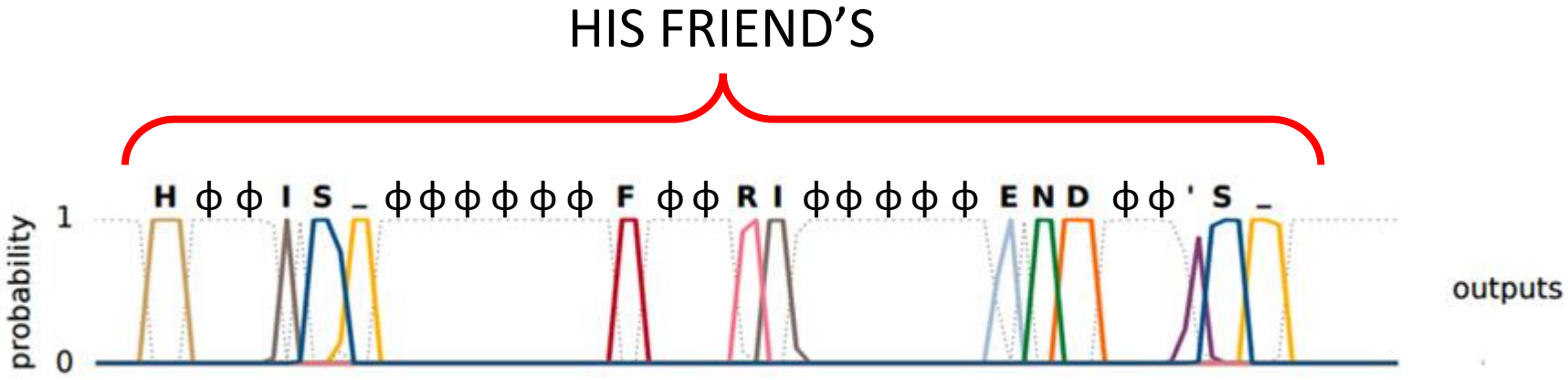
Label: 好 棒

All possible alignments are considered as correct.



Many to Many (Output is shorter)

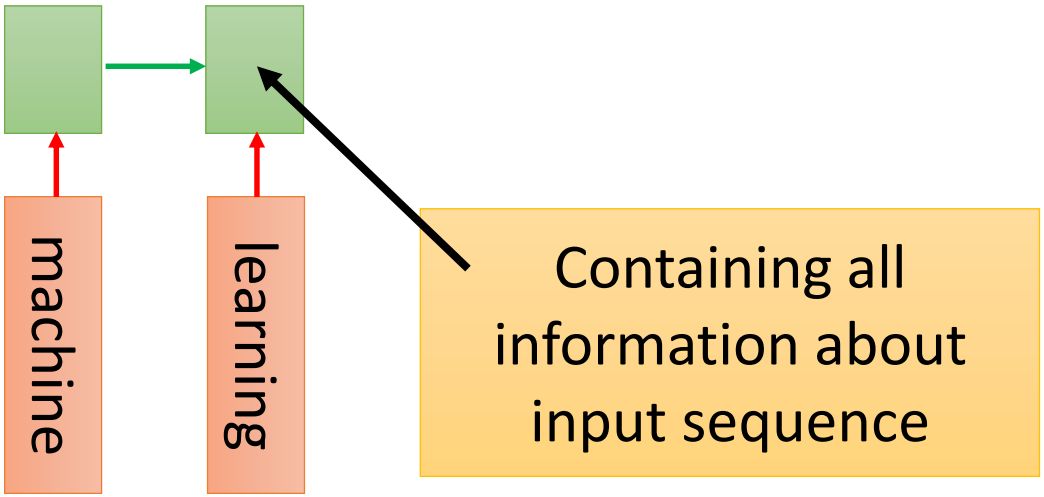
- CTC: example



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.

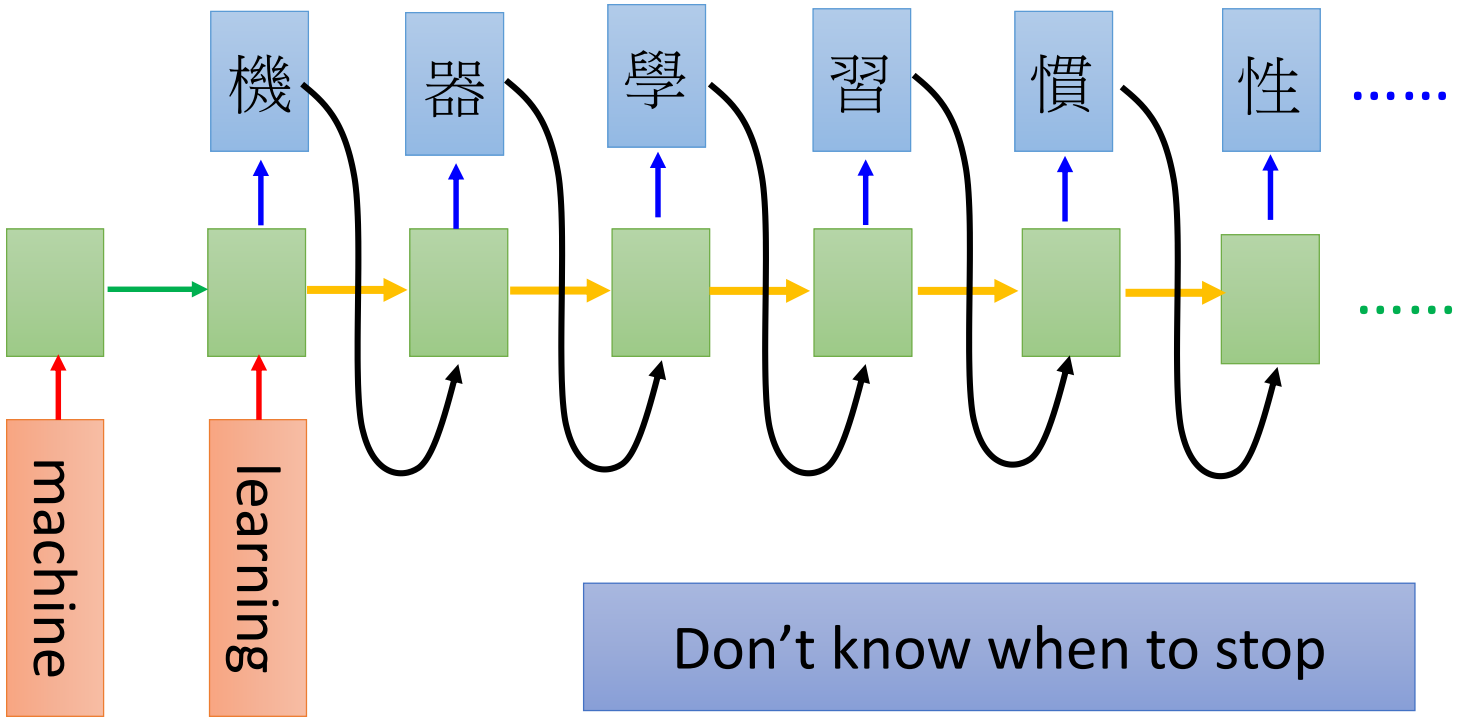
Many to Many (No Limitation)

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

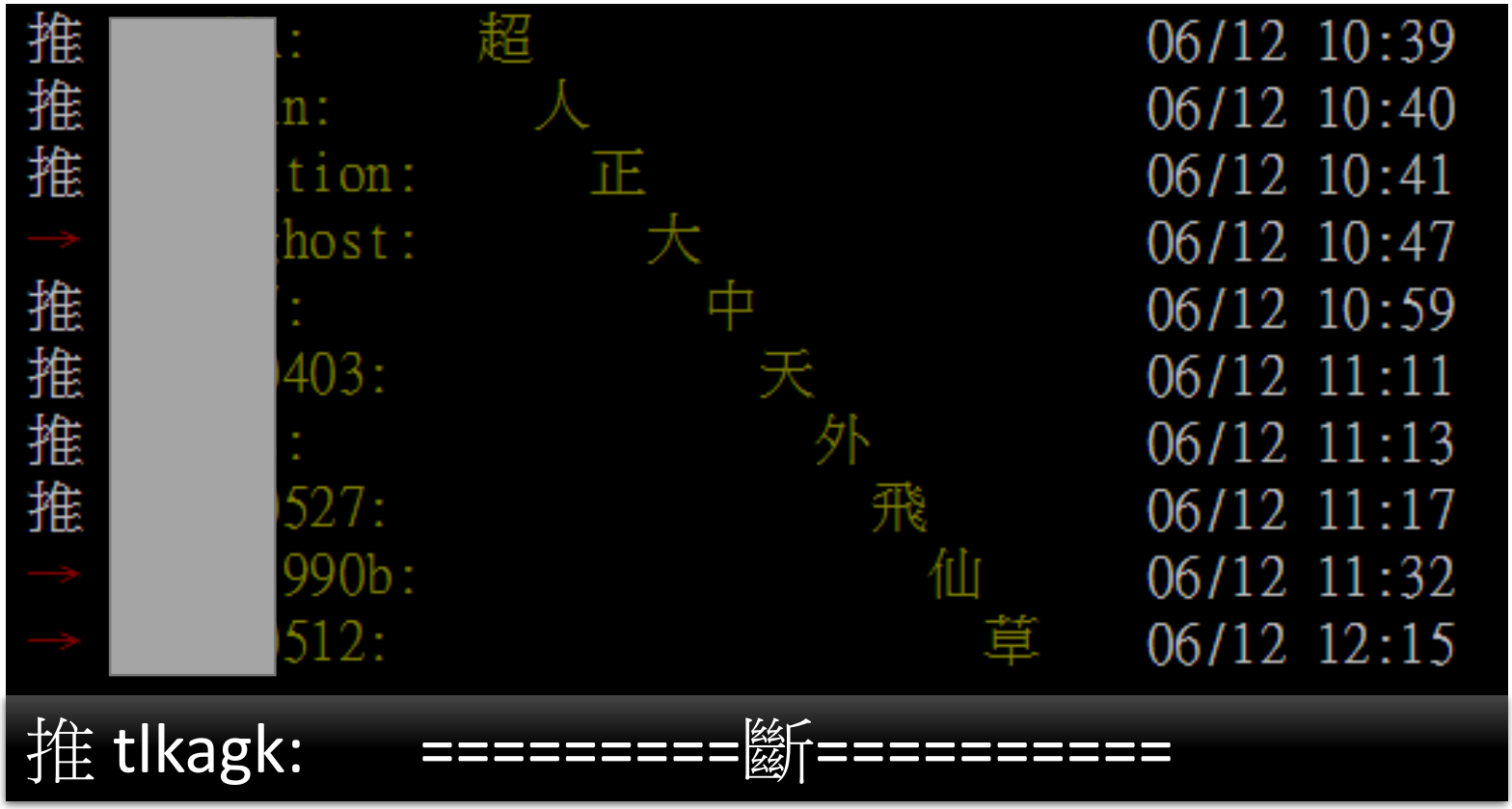


Many to Many (No Limitation)

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



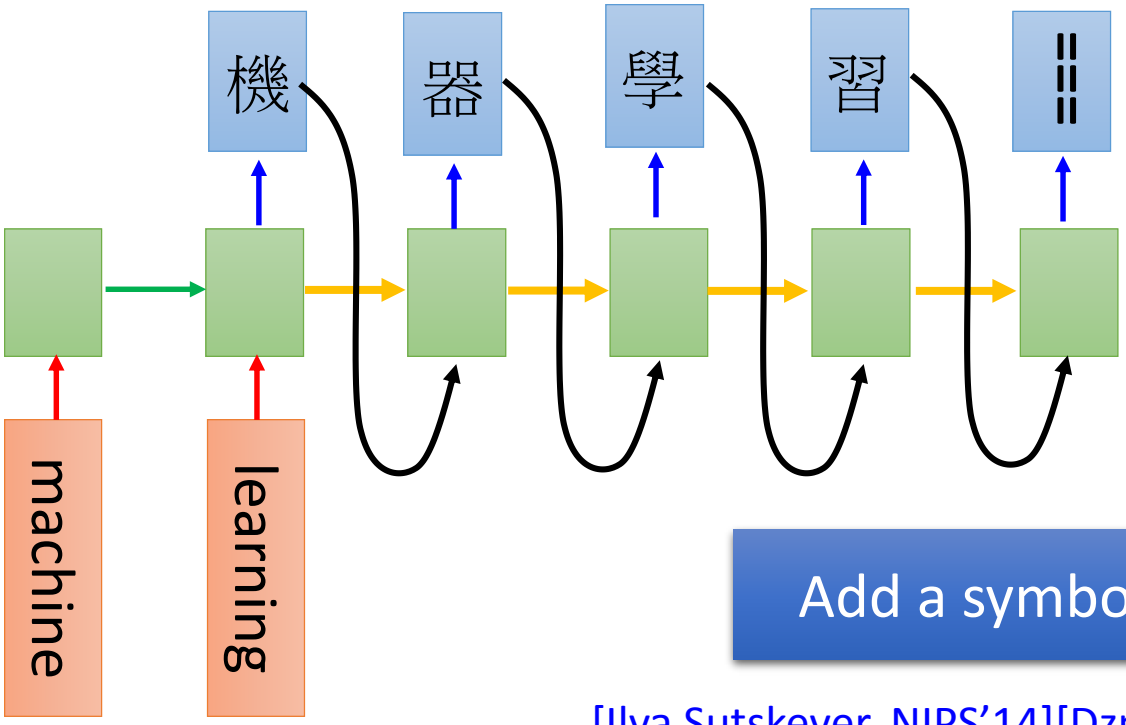
Many to Many (No Limitation)



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)



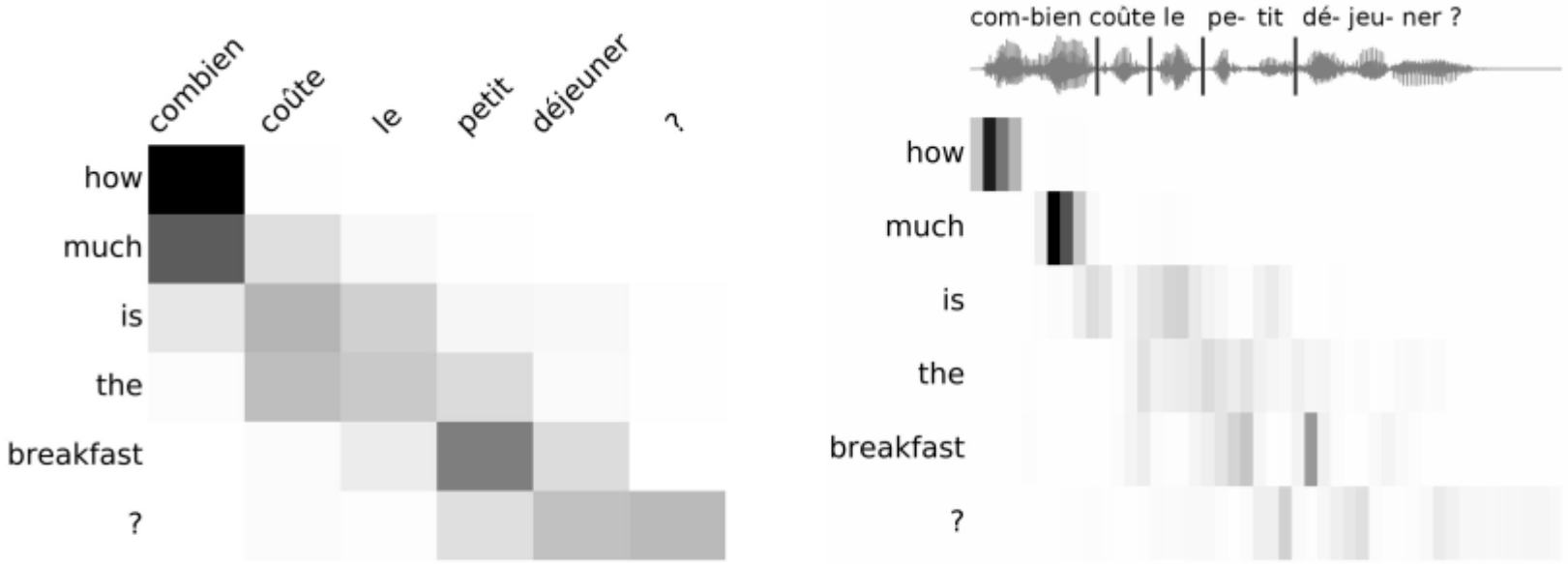
Add a symbol "===" (斷)

[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

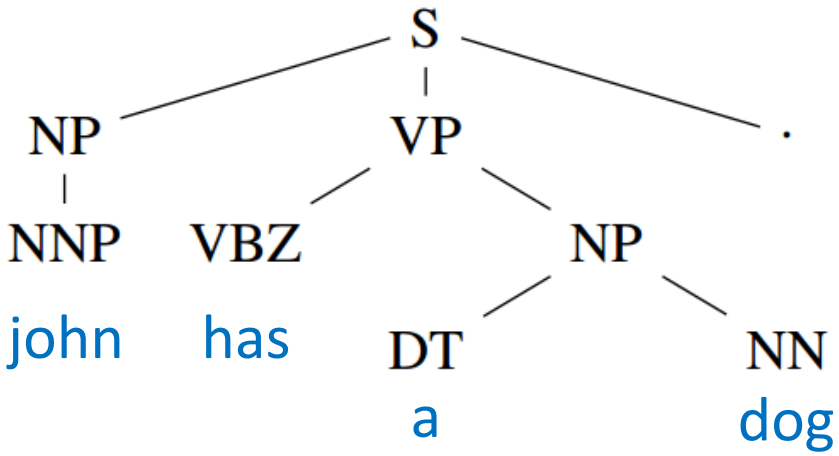
(b) Speech translation alignment

Figure 1: Alignments performed by the attention model during training

Beyond Sequence

- Syntactic parsing

John has a dog . →

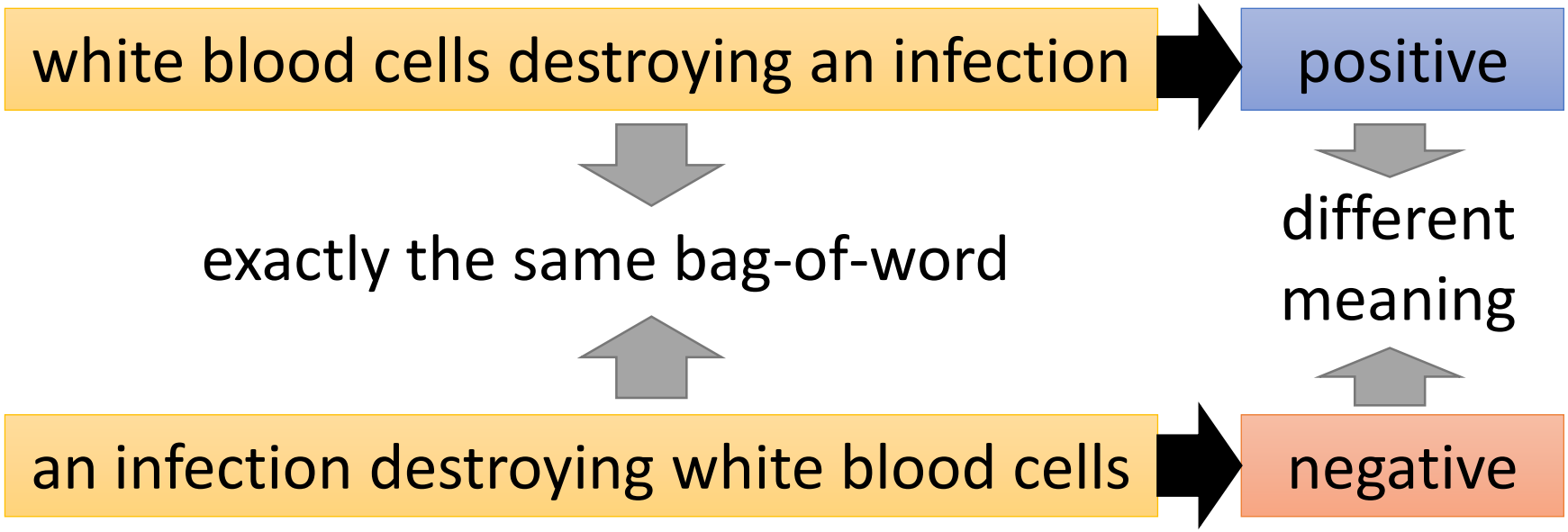


John has a dog . → (S (NP NNP)$_{\text{NP}}$ (VP VBZ (NP DT NN)$_{\text{NP}}$)$_{\text{VP}}$.)$_{\text{S}}$

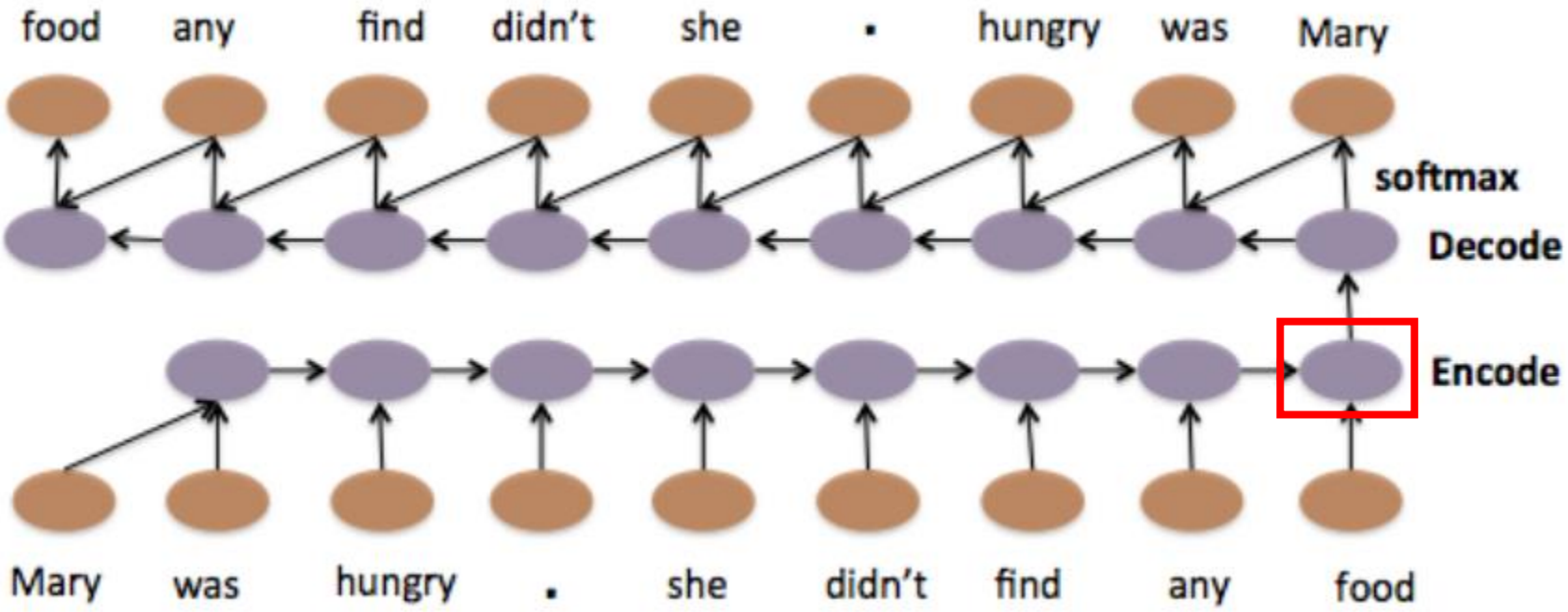
Oriol Vinyals, Lukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, Geoffrey Hinton, Grammar as a Foreign Language, NIPS 2015

Sequence-to-sequence Auto-encoder - Text

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

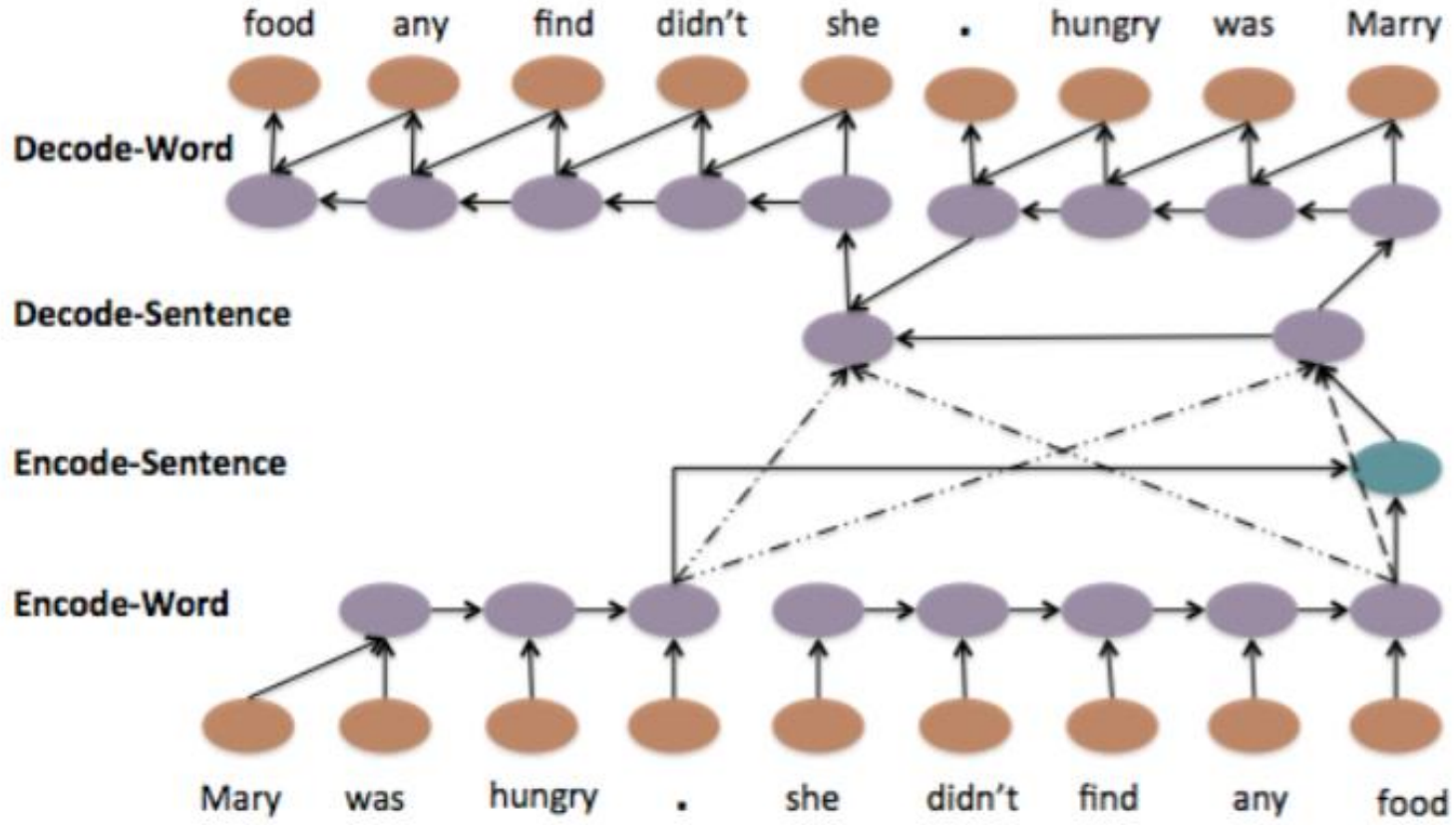


Sequence-to-sequence Auto-encoder - Text



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

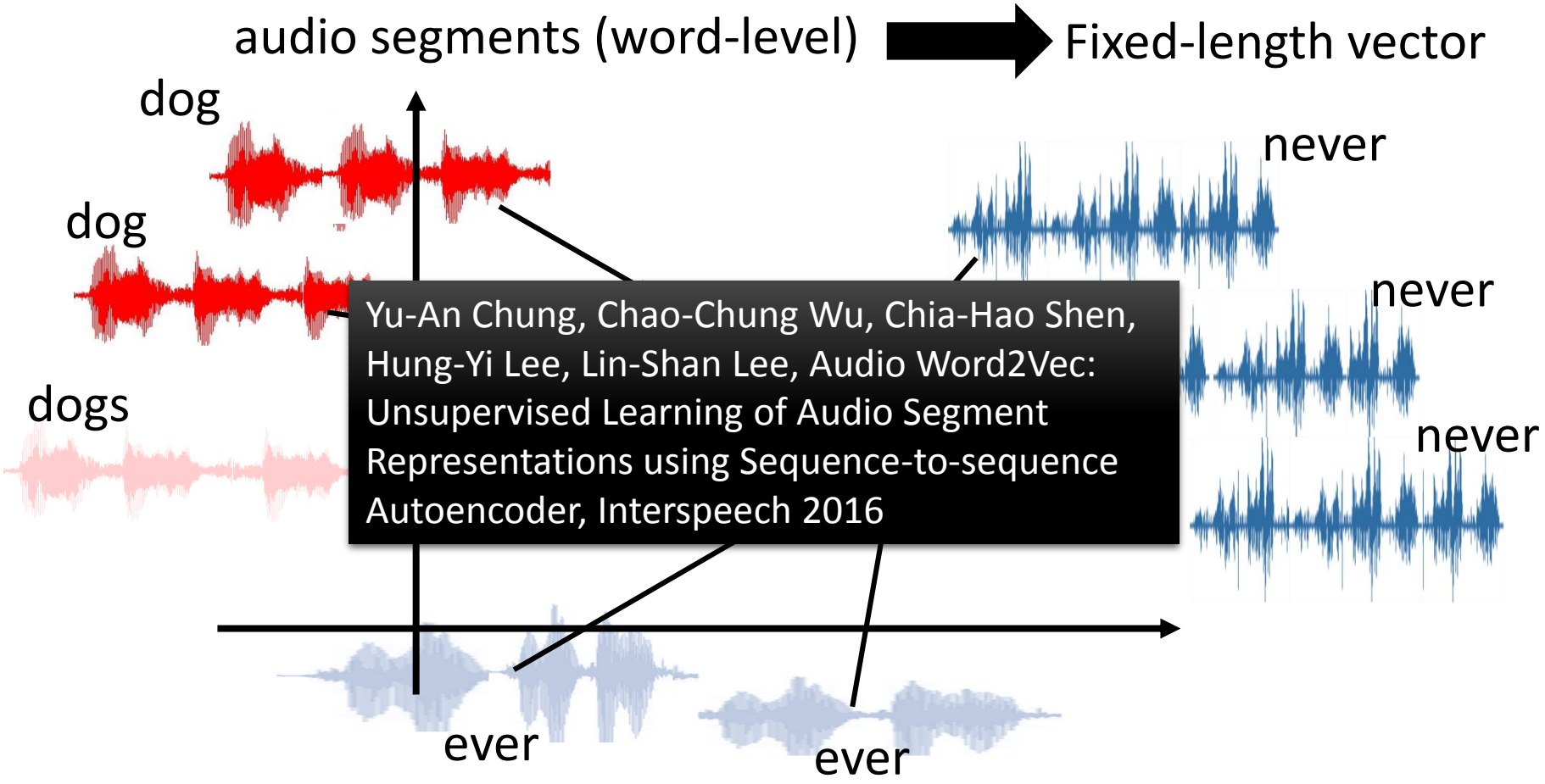
Sequence-to-sequence Auto-encoder - Text



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

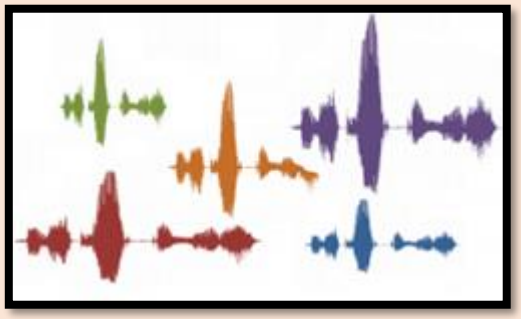
- Dimension reduction for a sequence with variable length



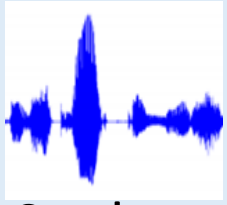
Sequence-to-sequence Auto-encoder - Speech

Audio archive divided into variable-length audio segments

Off-line



Audio Segment to Vector



Spoken Query

Audio Segment to Vector

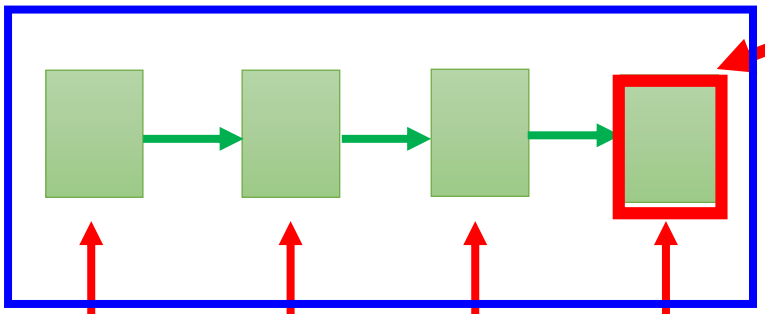
Similarity

Search Result

On-line

Sequence-to-sequence Auto-encoder - Speech

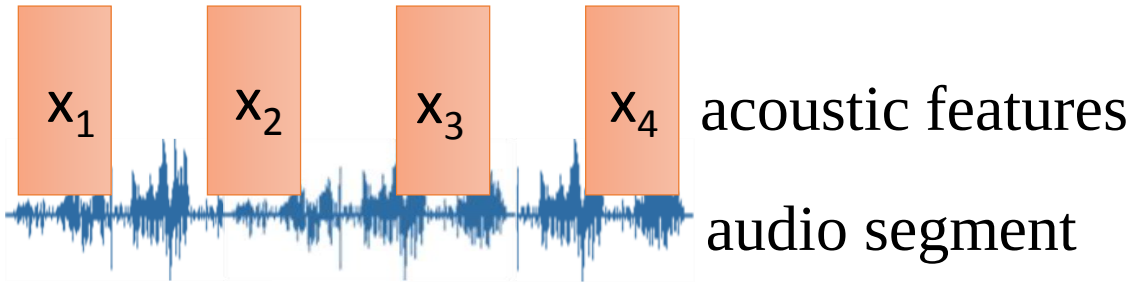
RNN Encoder



The values in the memory represent the whole audio segment

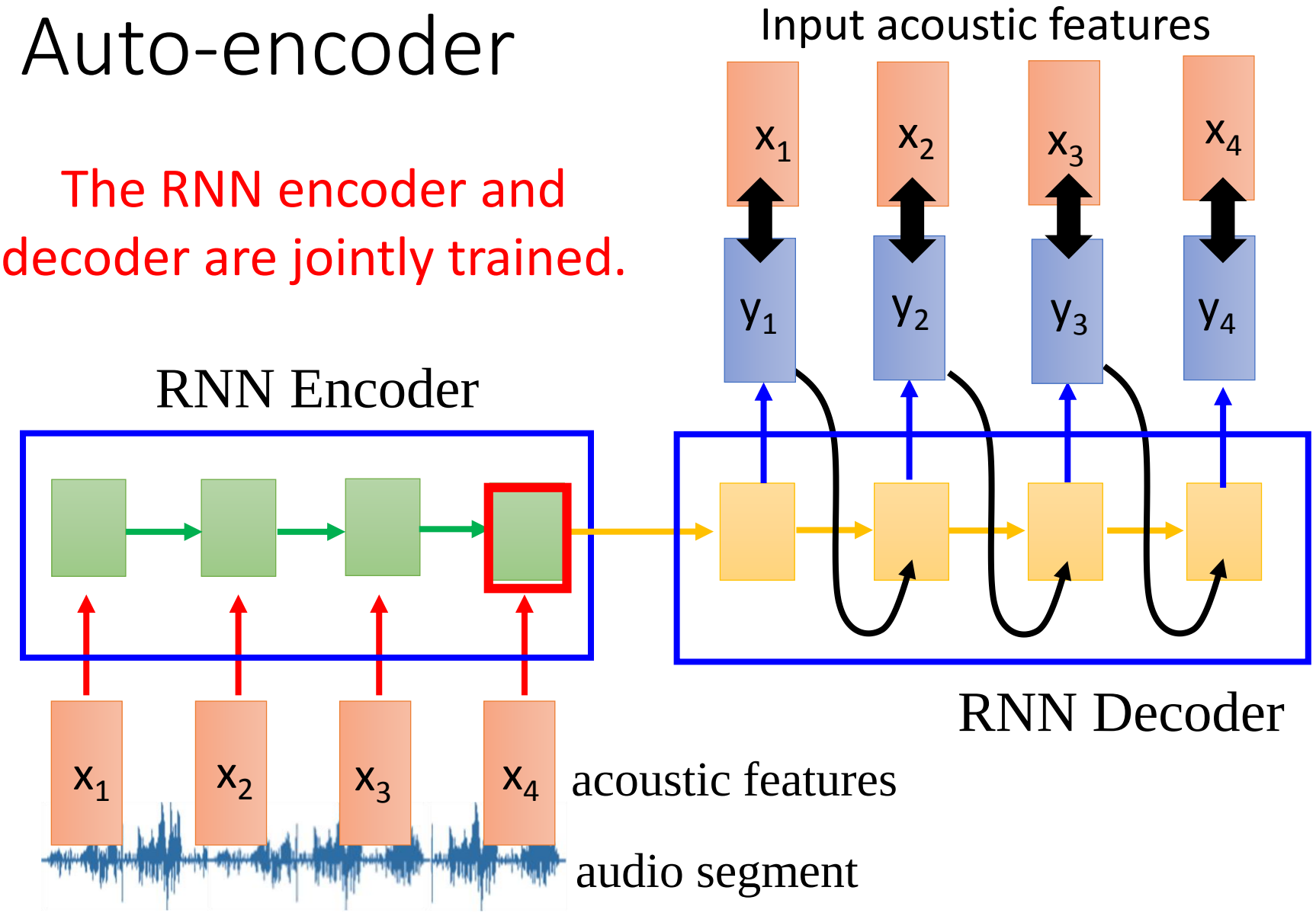
The vector we want

How to train RNN Encoder?



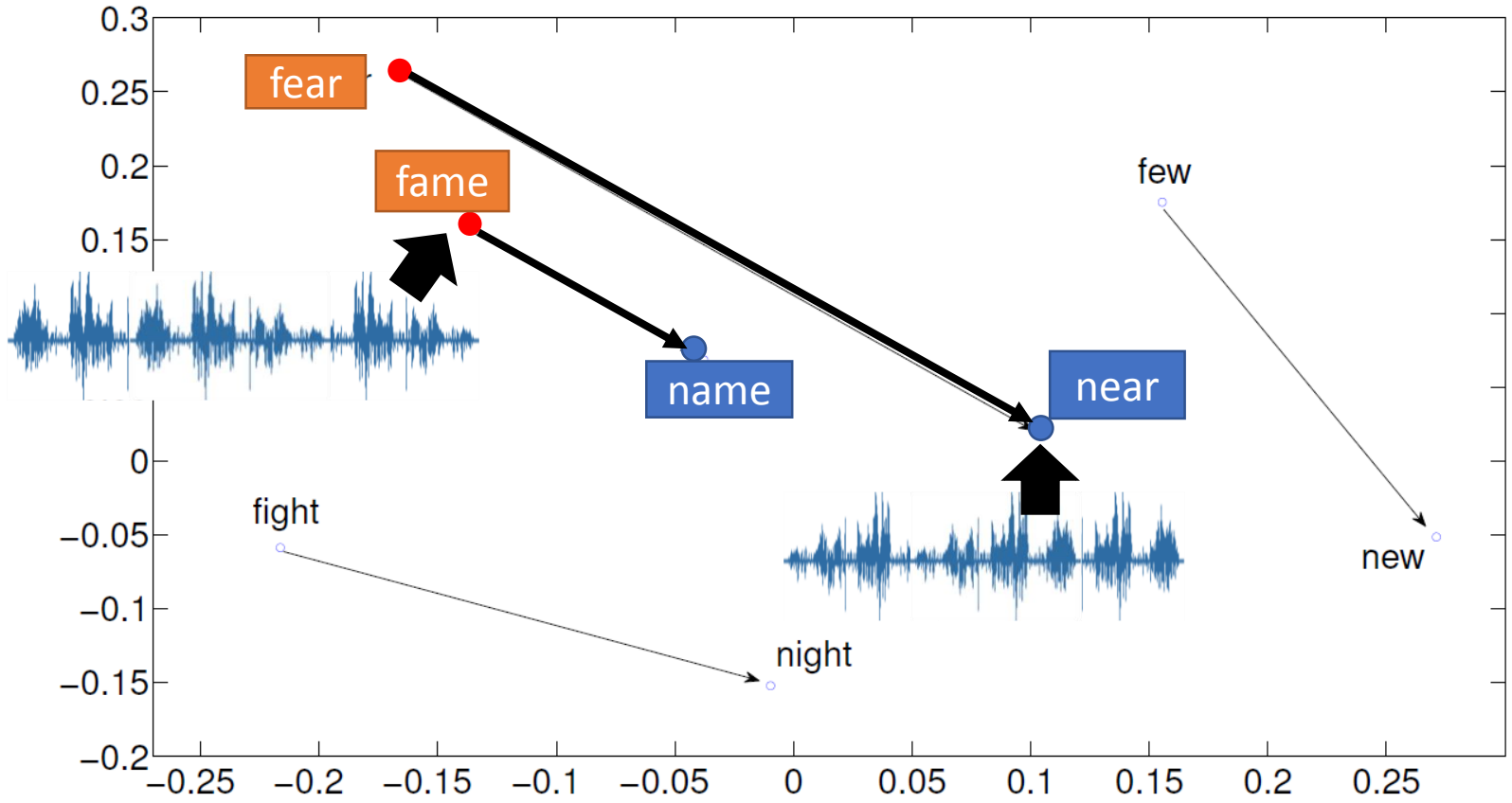
Sequence-to-sequence Auto-encoder

The RNN encoder and decoder are jointly trained.

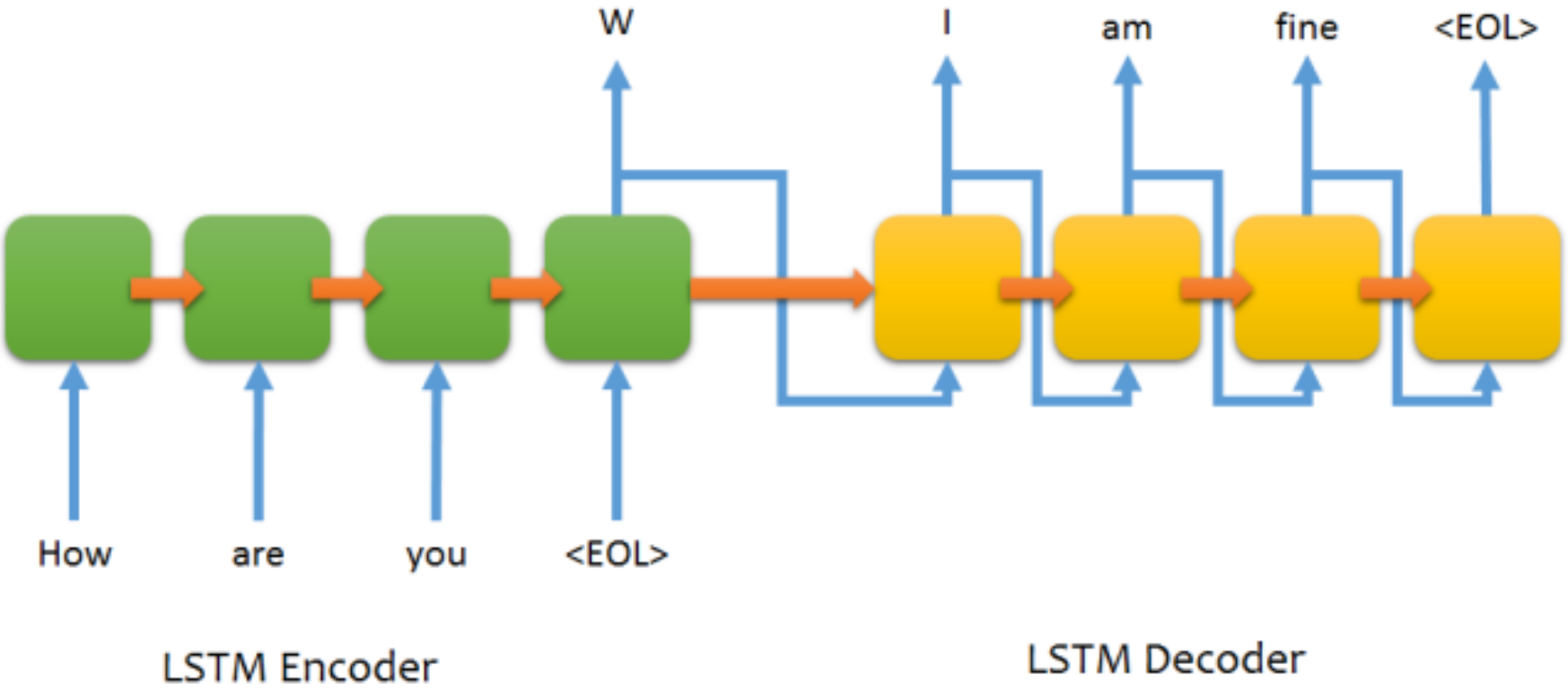


Sequence-to-sequence Auto-encoder - Speech

- Visualizing embedding vectors of the words



Demo: Chat-bot



電視影集 (~40,000 sentences)、美國總統大選辯論

Demo: Chat-bot

- Develop Team
  - Interface design: Prof. Lin-Lin Chen & Arron Lu
  - Web programming: Shi-Yun Huang
  - Data collection: Chao-Chuang Shih
  - System implementation: Kevin Wu, Derek Chuang, & Zhi-Wei Lee (李致緯), Roy Lu (盧柏儒)
  - System design: Richard Tsai & Hung-Yi Lee

NTU IOX center

Demo: Video Caption Generation

Video

A group of people is knocked by a tree.

A girl is running.

A group of people is walking in the forest.

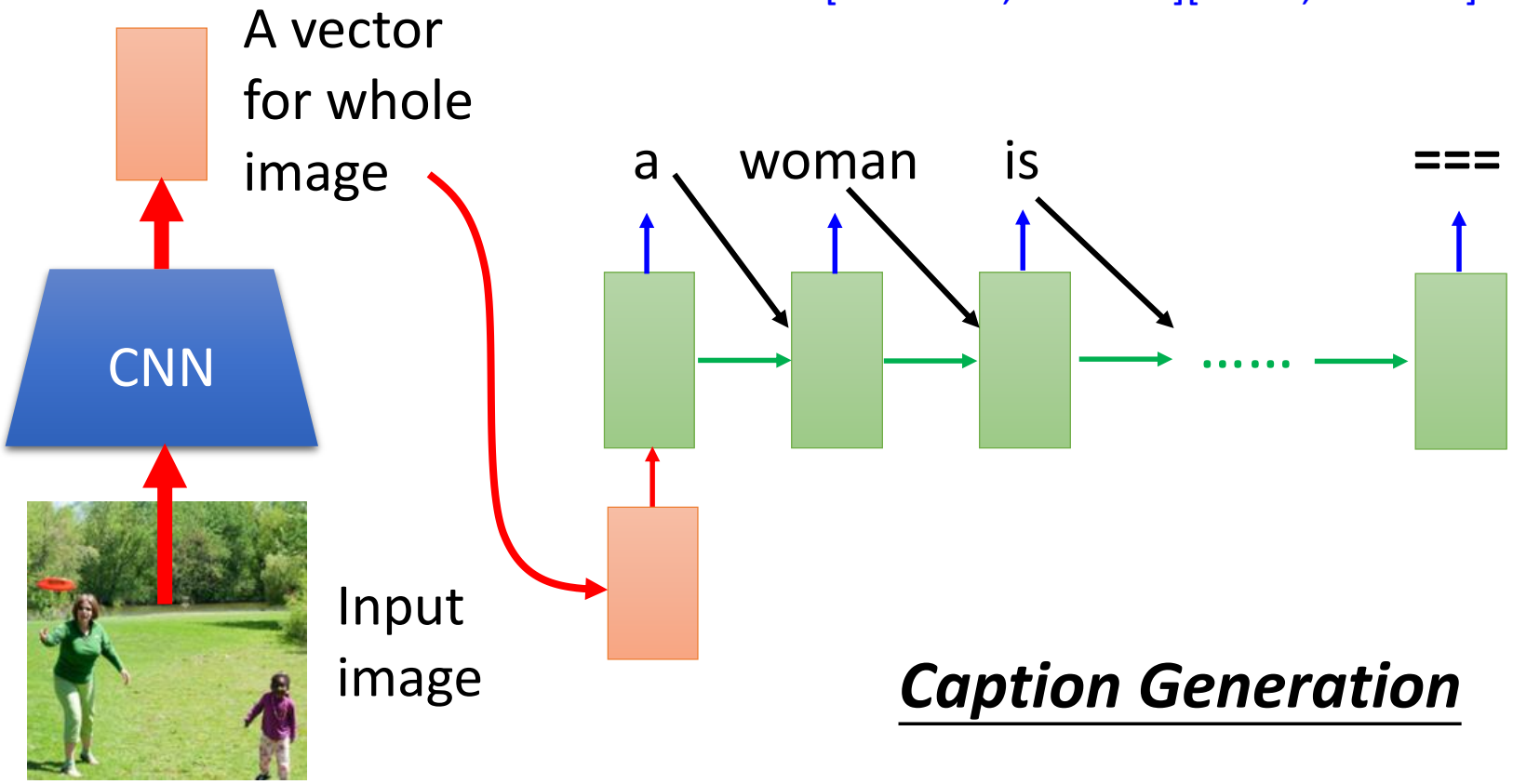
Demo: Video Caption Generation

- Can machine describe what it see from video?
- Demo: 台大語音處理實驗室 曾柏翔、吳柏瑜、盧宏宗
- Video: 莊舜博、楊棋宇、黃邦齊、萬家宏

Demo: Image Caption Generation

- Input an image, but output a sequence of words

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

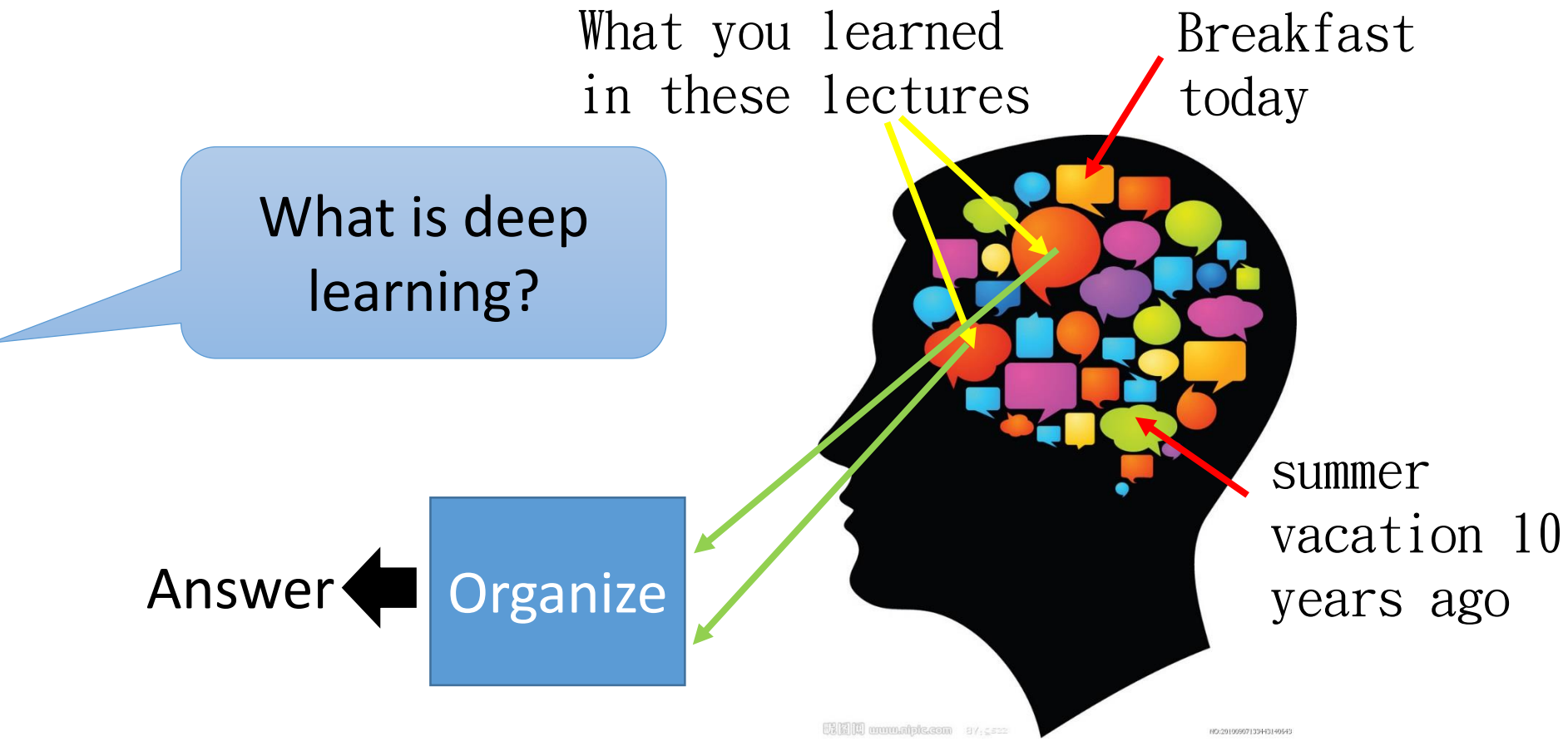


Demo: Image Caption Generation

- Can machine describe what it see from image?
- Demo:台大電機系 大四 蘇子睿、林奕辰、徐翊祥、陳奕安

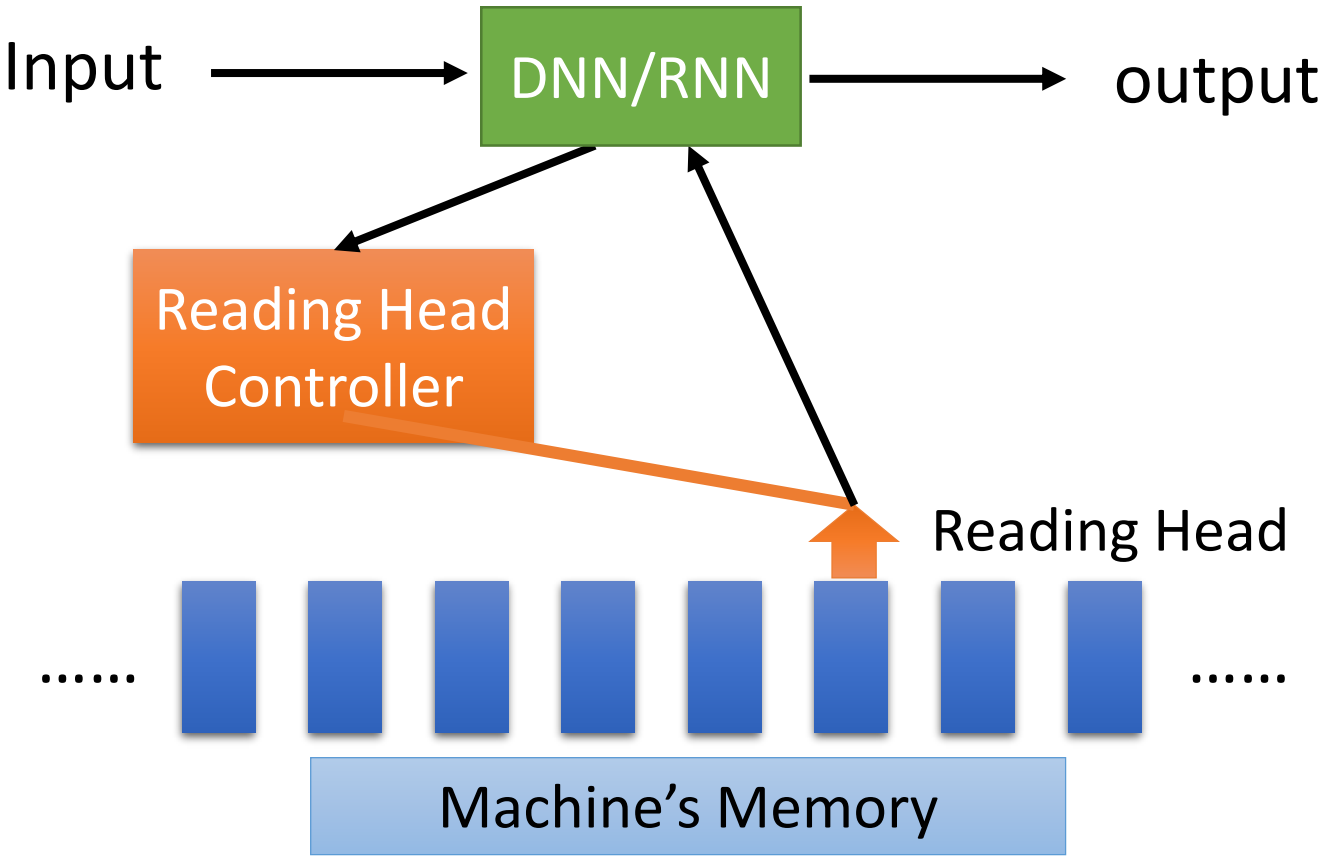
http://news.ltn.com.tw/photo/politics/breakingnews/975542_1

Attention-based Model



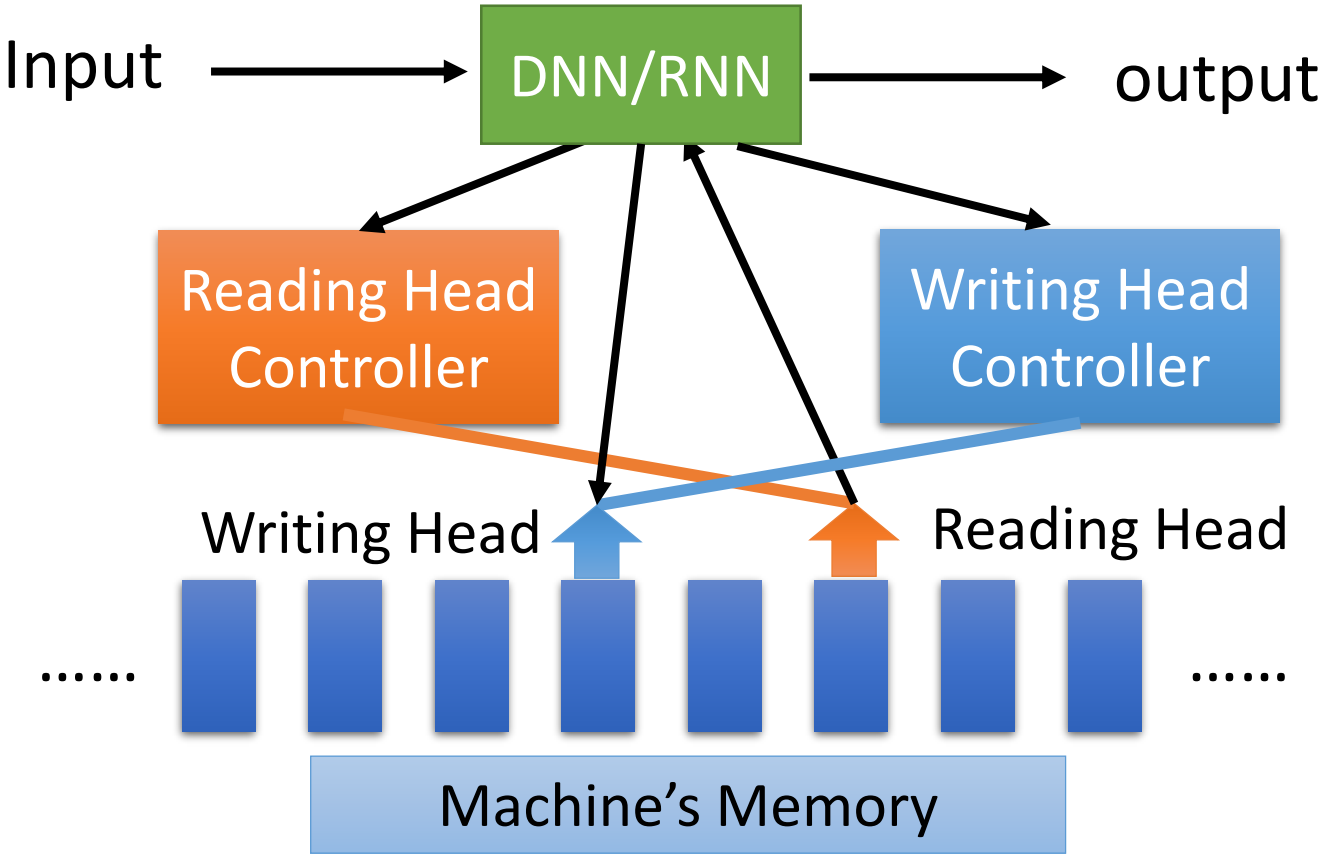
http://henrylo1605.blogspot.tw/2015/05/blog-post_56.html

Attention-based Model



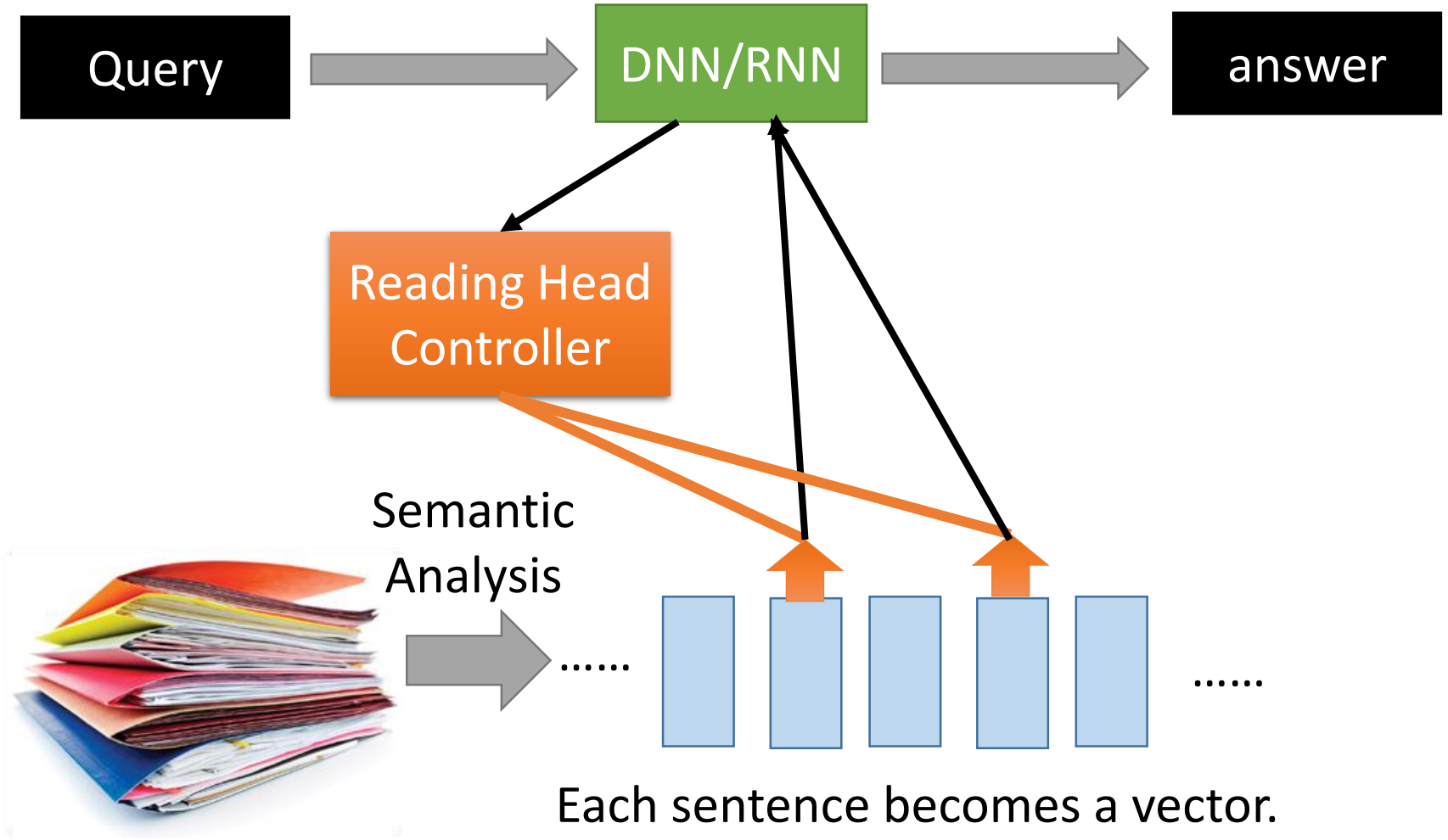
Ref:
http://speech.ee.ntu.edu.tw/~tlkagk/courses/MLDS_2015_2/Lecture/Attain%20(v3).ecm.mp4/index.html

Attention-based Model v2



Neural Turing Machine

Reading Comprehension



Reading Comprehension

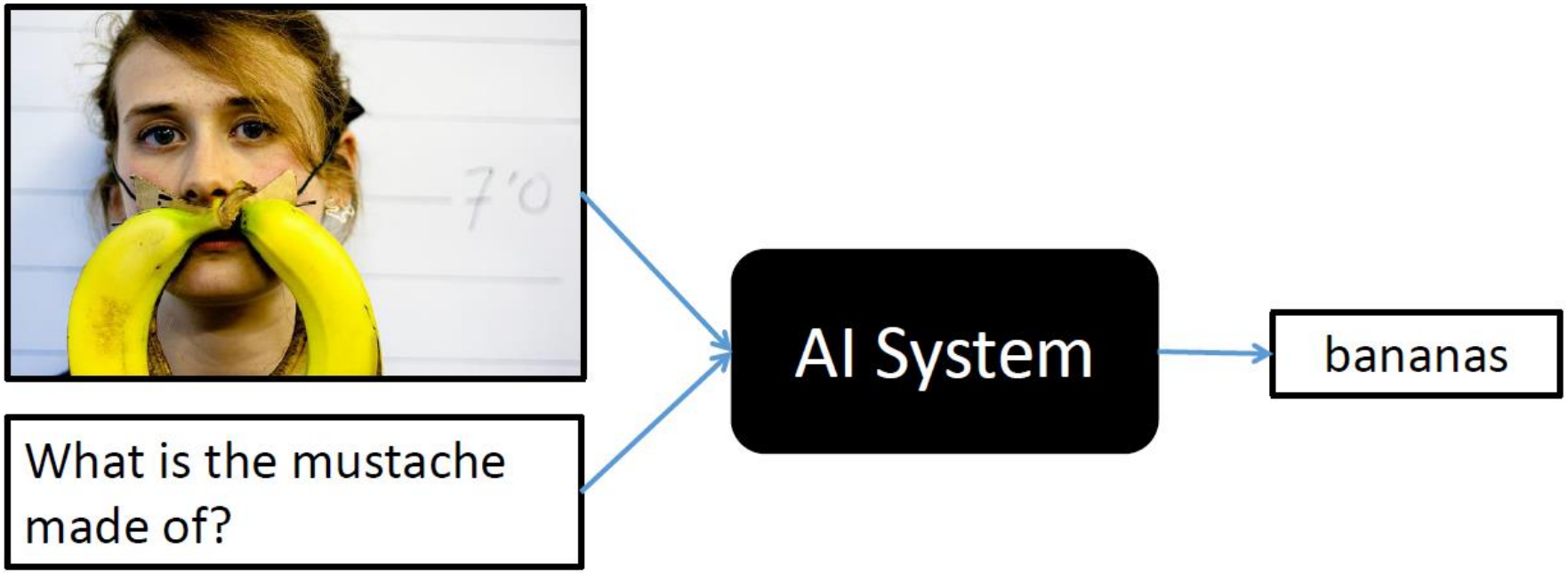
- End-To-End Memory Networks. S. Sukhbaatar, A. Szlam, J. Weston, R. Fergus. NIPS, 2015.

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**			

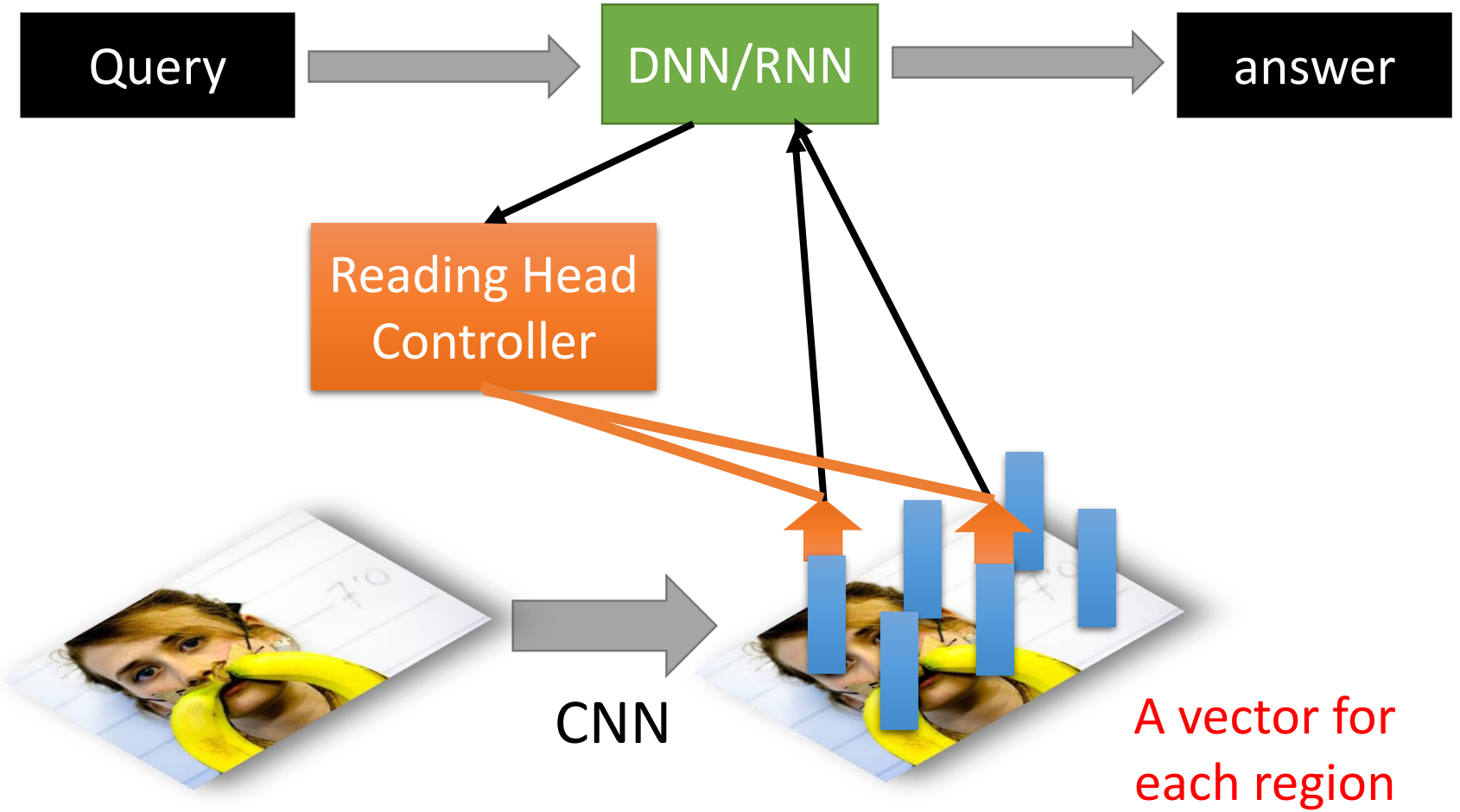
Keras has example:
https://github.com/fchollet/keras/blob/master/examples/babi_memnn.py

Visual Question Answering



source: http://visualqa.org/

Visual Question Answering



Speech Question Answering

- **TOEFL Listening Comprehension Test by Machine**
- Example:

  Audio Story: (The original story is 5 min long.)

  Question: “ What is a possible origin of Venus’ clouds? ”

  Choices:

  (A) gases released as a result of volcanic activity

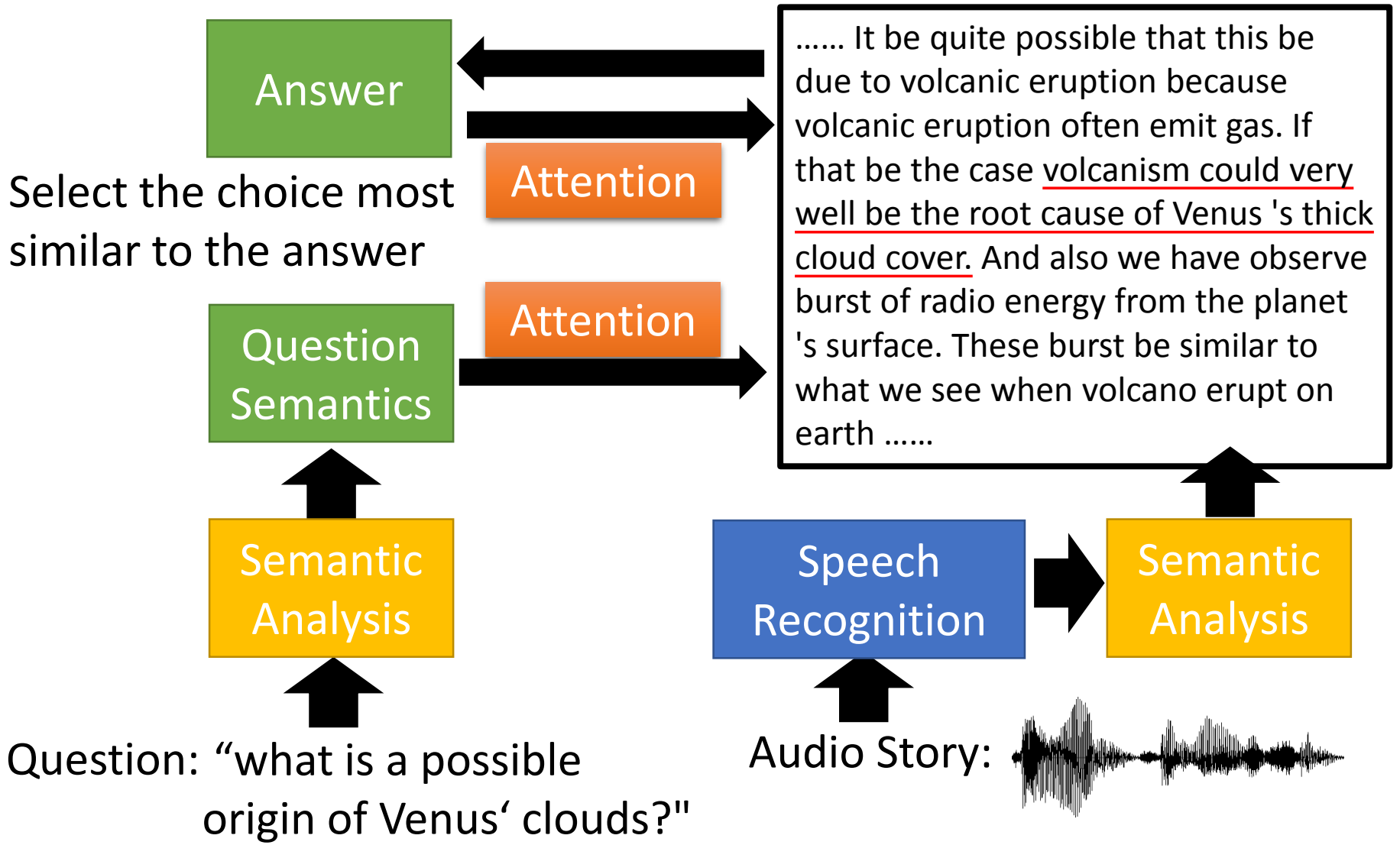
  (B) chemical reactions caused by high surface temperatures

  (C) bursts of radio energy from the plane's surface

  (D) strong winds that blow dust into the atmosphere

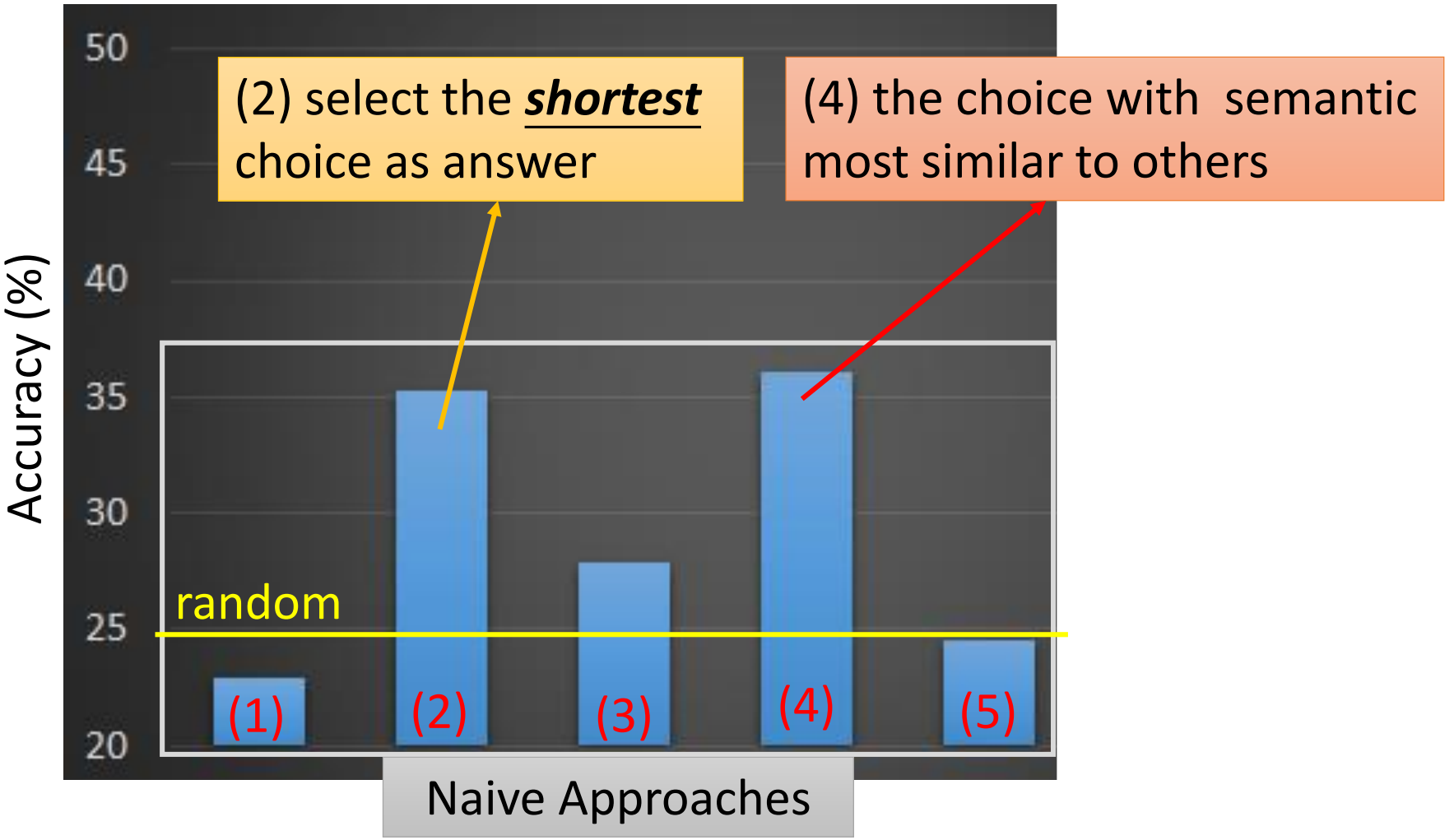
Model Architecture

Everything is learned from training examples

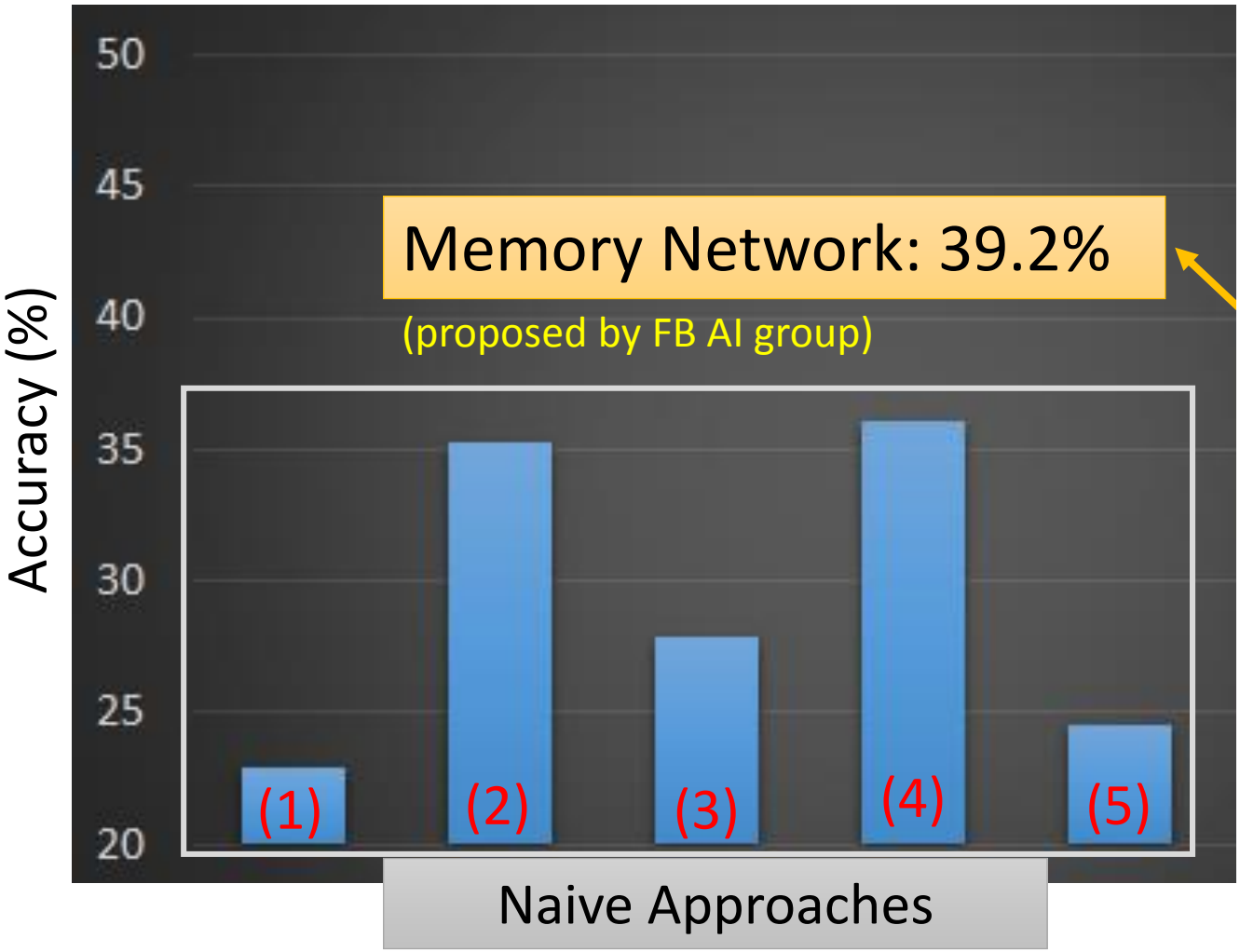


Simple Baselines

Experimental setup:
717 for training,
124 for validation, 122 for testing

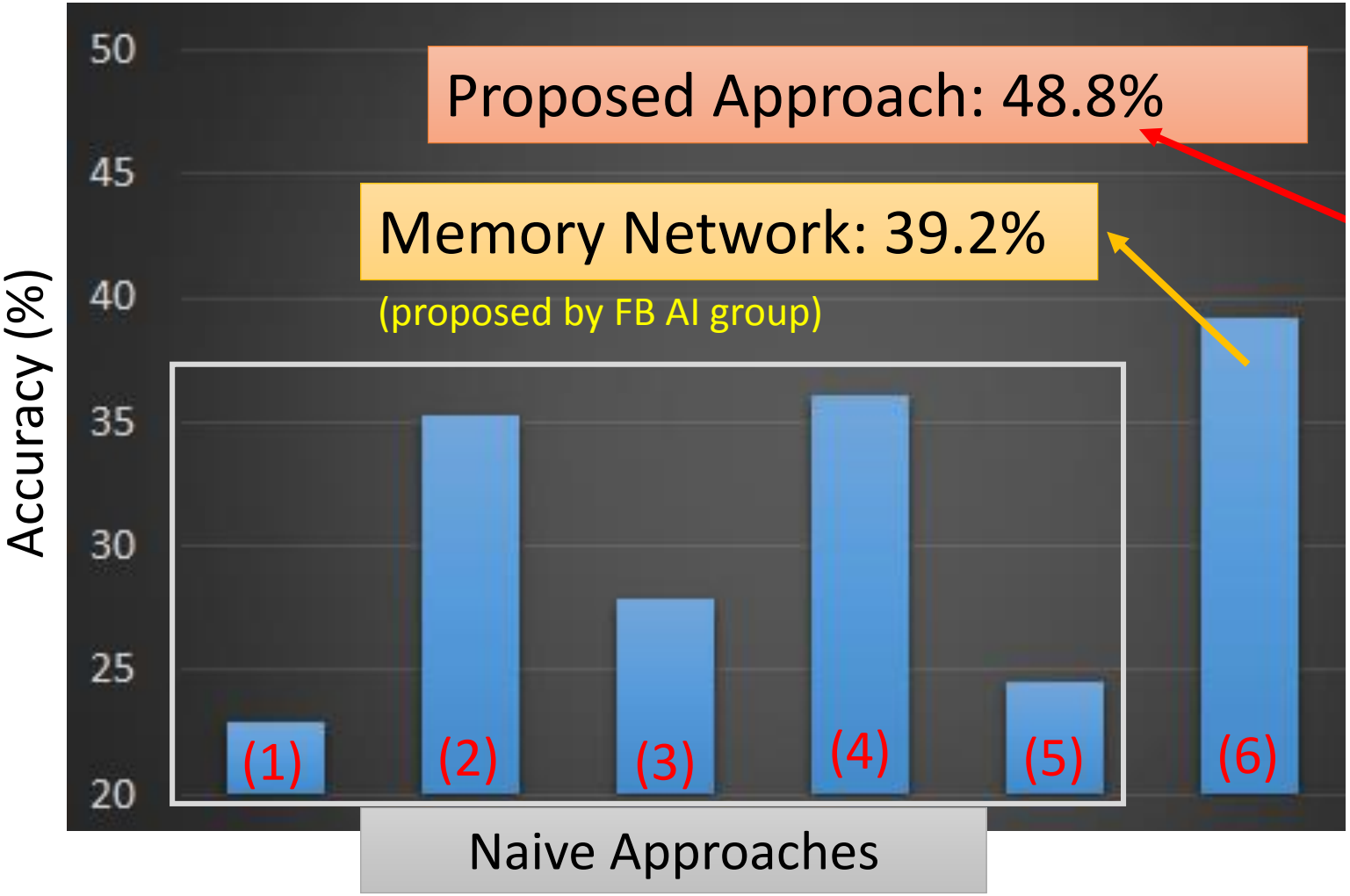


Memory Network



Proposed Approach

[Tseng & Lee, Interspeech 16]
[Fang & Hsu & Lee, SLT 16]



To Learn More ……

- 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/

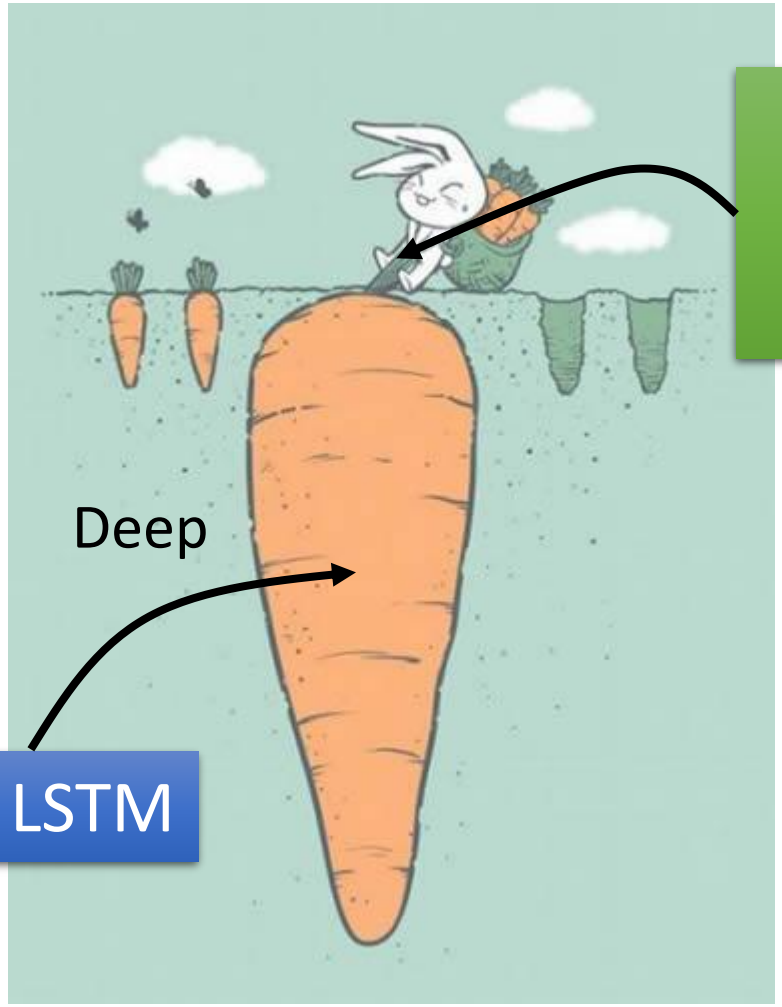
Deep & Structured

RNN v.s. Structured Learning

- RNN, LSTM
  - Unidirectional RNN does not consider the whole sequence
  - Cost and error not always related
  - Deep 勝

- HMM, CRF, Structured Perceptron/SVM
  - Using Viterbi, so consider the whole sequence 勝 ?
    - How about Bidirectional RNN?
  - Can explicitly consider the label dependency 勝
    - Cost is the upper bound of error 勝

Integrated Together



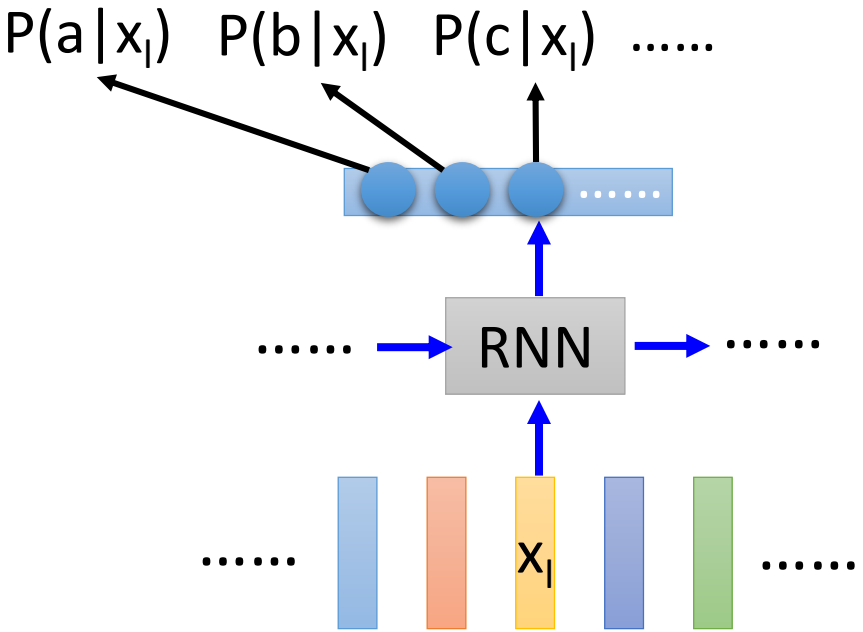
- Explicitly model the dependency
- Cost is the upper bound of error

http://photo30.bababian.com/upload1/20100415/42E9331A6580A46A5F89E98638B8FD76.jpg

Integrated together

- Speech Recognition: CNN/LSTM/DNN + HMM

$$P(x,y) = P(y_1|start)\prod_{l=1}^{L-1} P(y_{l+1}|y_l)\, P(end|y_L)\prod_{l=1}^{L} P(x_l|y_l)$$



$$P(x_l|y_l) = \frac{P(x_l,y_l)}{P(y_l)}$$

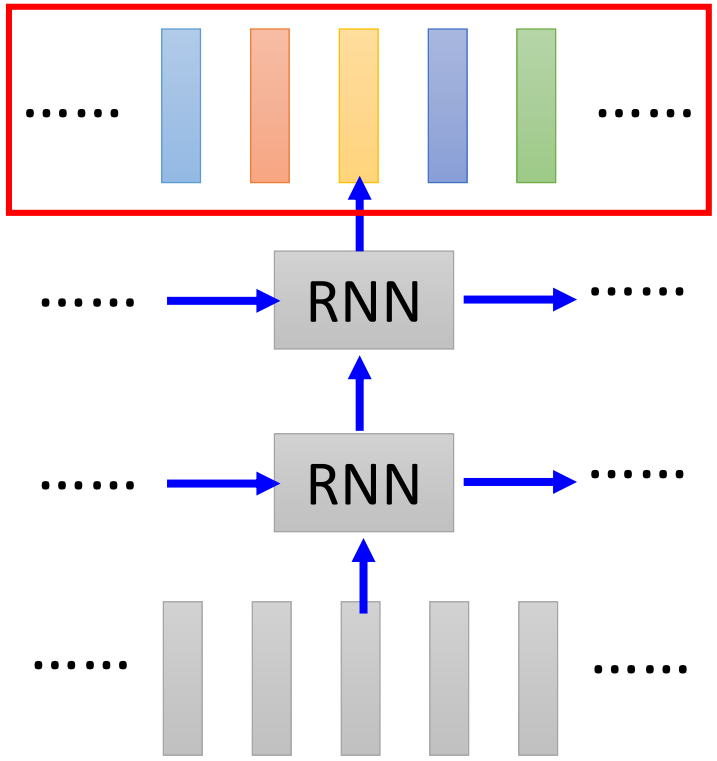
RNN

$$= \frac{P(y_l|x_l)\cancel{P(x_l)}}{P(y_l)}$$

Count

Integrated together

- Semantic Tagging: Bi-directional LSTM + CRF/Structured SVM



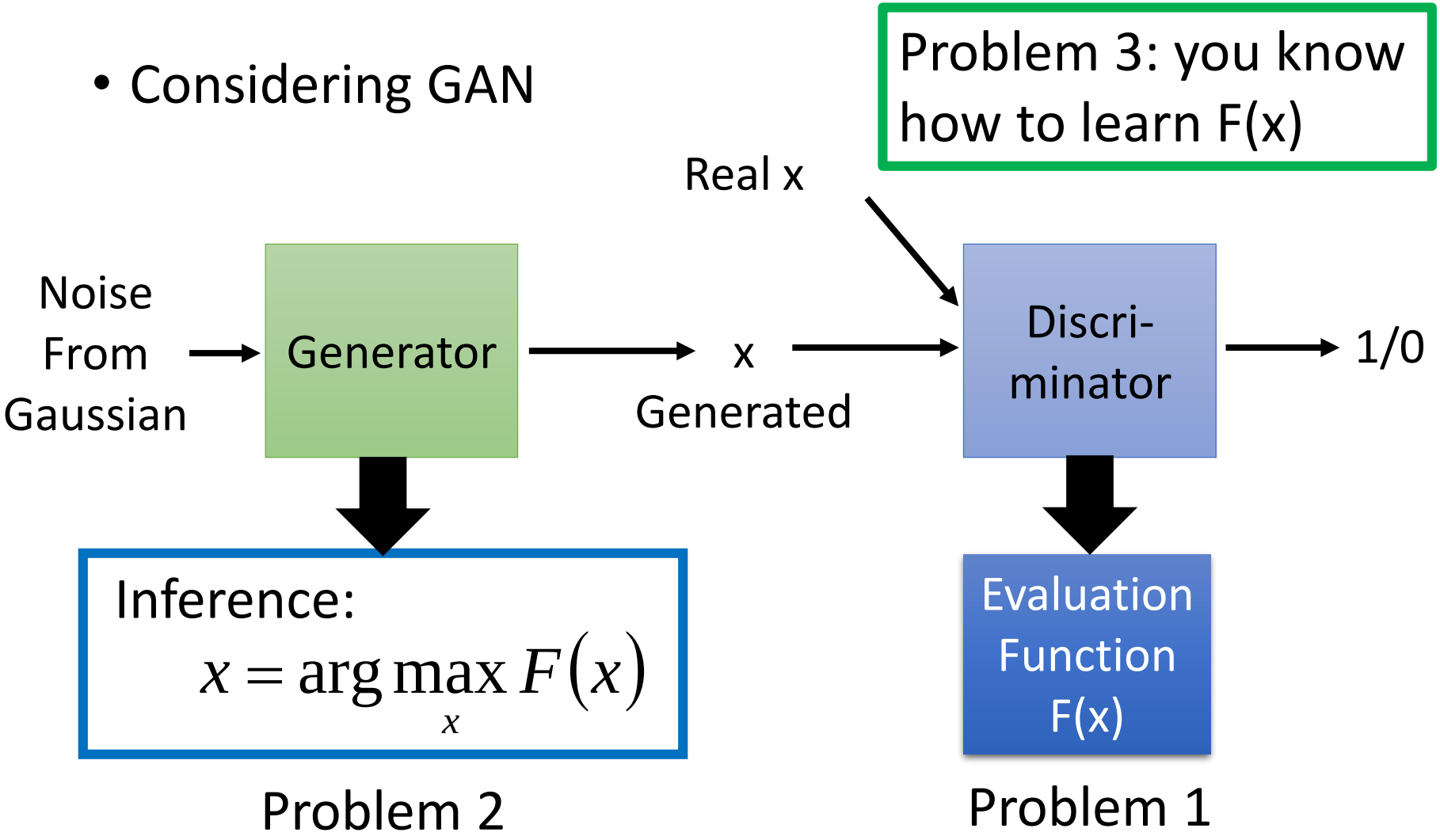
$w \cdot \phi(x, y)$

Testing:

$$\tilde{y} = \arg\max_{y \in \mathbb{Y}} w \cdot \phi(x, y)$$

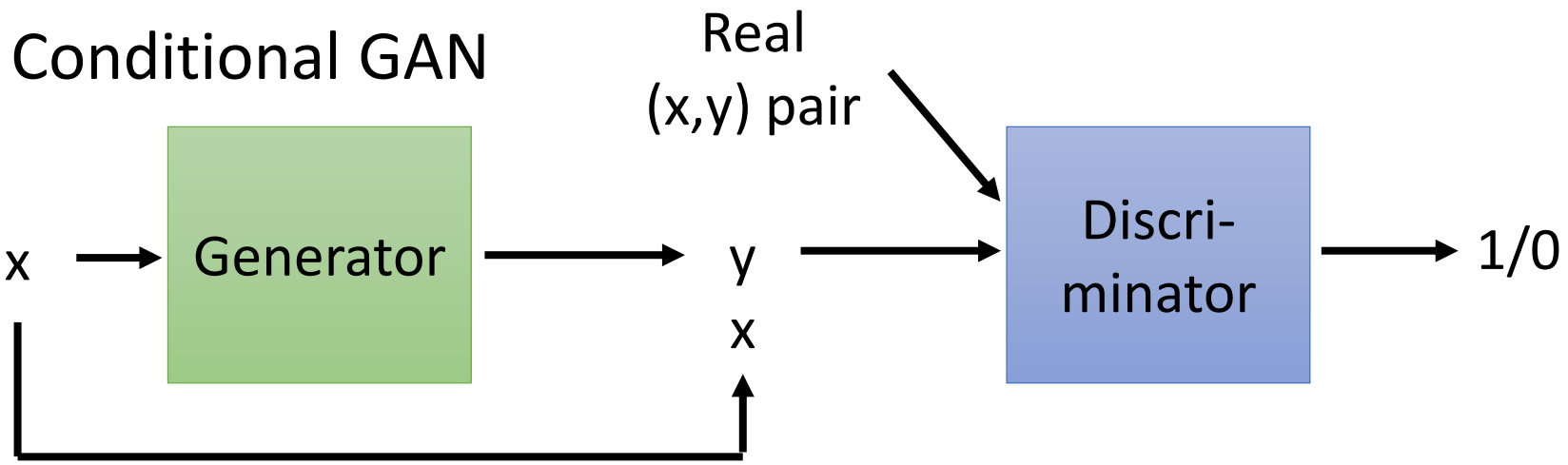
Is structured learning practical?

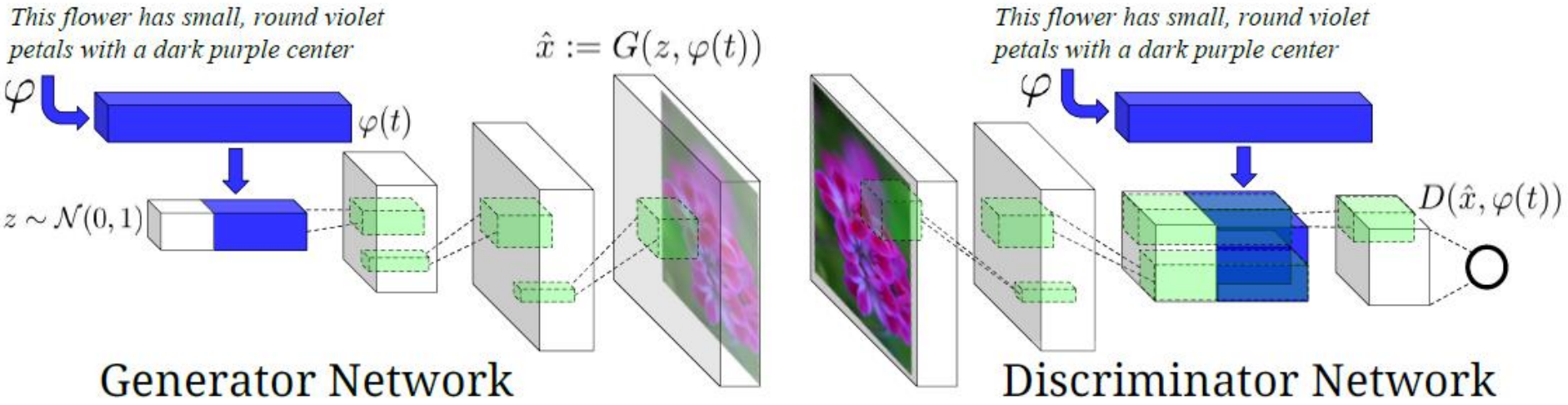
- Considering GAN



Is structured learning practical?

- Conditional GAN





Sounds crazy?
People do think in this way …

Deep and Structured will be the future.

- Connect Energy-based model with GAN:
  - A Connection Between Generative Adversarial Networks, Inverse Reinforcement Learning, and Energy-Based Models
  - Deep Directed Generative Models with Energy-Based Probability Estimation
  - ENERGY-BASED GENERATIVE ADVERSARIAL NETWORKS
- Deep learning model for inference
  - Deep Unfolding: Model-Based Inspiration of Novel Deep Architectures
  - Conditional Random Fields as Recurrent Neural Networks

Machine learning and having it deep and structured (MLDS)

- 和 ML 的不同
  - 在這學期 ML 中有提過的內容 (DNN, CNN …)，在 MLDS 中不再重複，只做必要的復習
- 教科書：“Deep Learning” (http://www.deeplearningbook.org/)
  - Part II 是講 deep learning 、Part III 就是講 structured learning

- Part II: Modern Practical Deep Networks
  - 6 Deep Feedforward Networks
  - 7 Regularization for Deep Learning
  - 8 Optimization for Training Deep Models
  - 9 Convolutional Networks
  - 10 Sequence Modeling: Recurrent and Recu
  - 11 Practical Methodology
  - 12 Applications
- Part III: Deep Learning Research
  - 13 Linear Factor Models
  - 14 Autoencoders
  - 15 Representation Learning
  - 16 Structured Probabilistic Models for Deep Learning
  - 17 Monte Carlo Methods
  - 18 Confronting the Partition Function
  - 19 Approximate Inference
  - 20 Deep Generative Models

Machine learning and having it deep and structured (MLDS)

- 所有作業都 2 ~ 4 人一組，可以先組好隊後一起來修
- MLDS 的作業和之前不同
  - RNN (把之前 MLDS 的三個作業合為一個)、Attention-based model 、Deep Reinforcement Learning 、Deep Generative Model 、Sequence-to-sequence learning
- MLDS 初選不開放加簽，以組為單位加簽，作業0的內容是做一個 DNN（可用現成套件）