

Sequence Generation

Hung-yi Lee

李宏毅

Outline

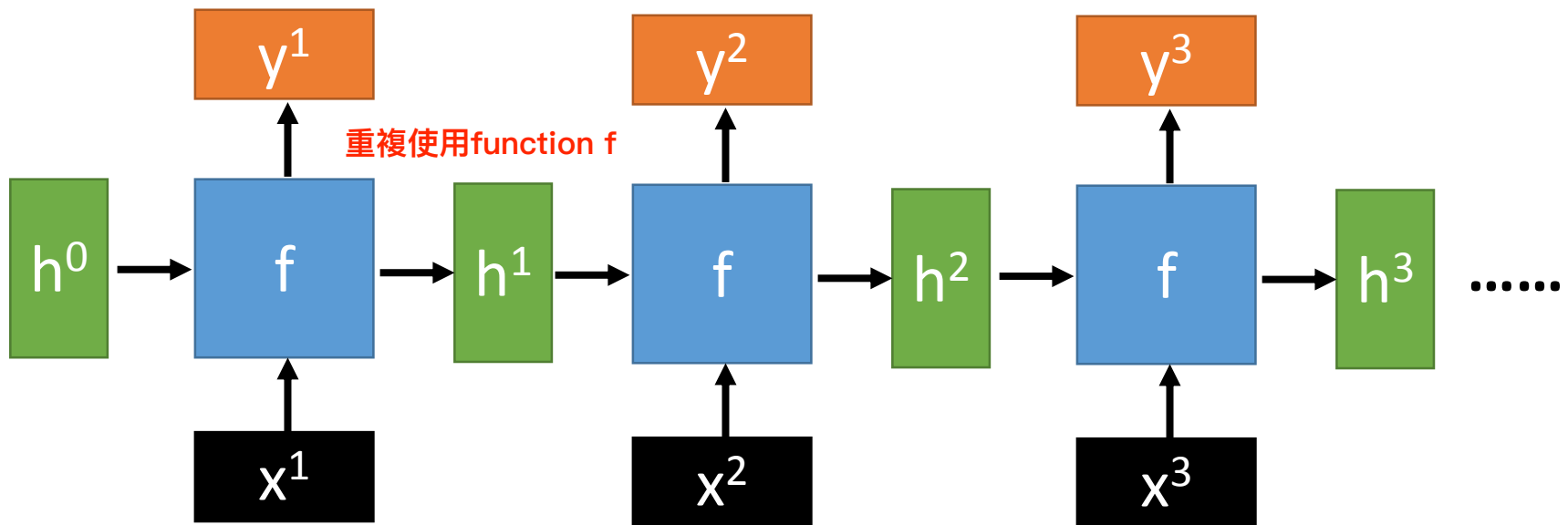
- RNN with Gated Mechanism
- Sequence Generation
- Conditional Sequence Generation
- Tips for Generation

RNN with Gated Mechanism

Recurrent Neural Network

- Given function f : $h', y = f(h, x)$

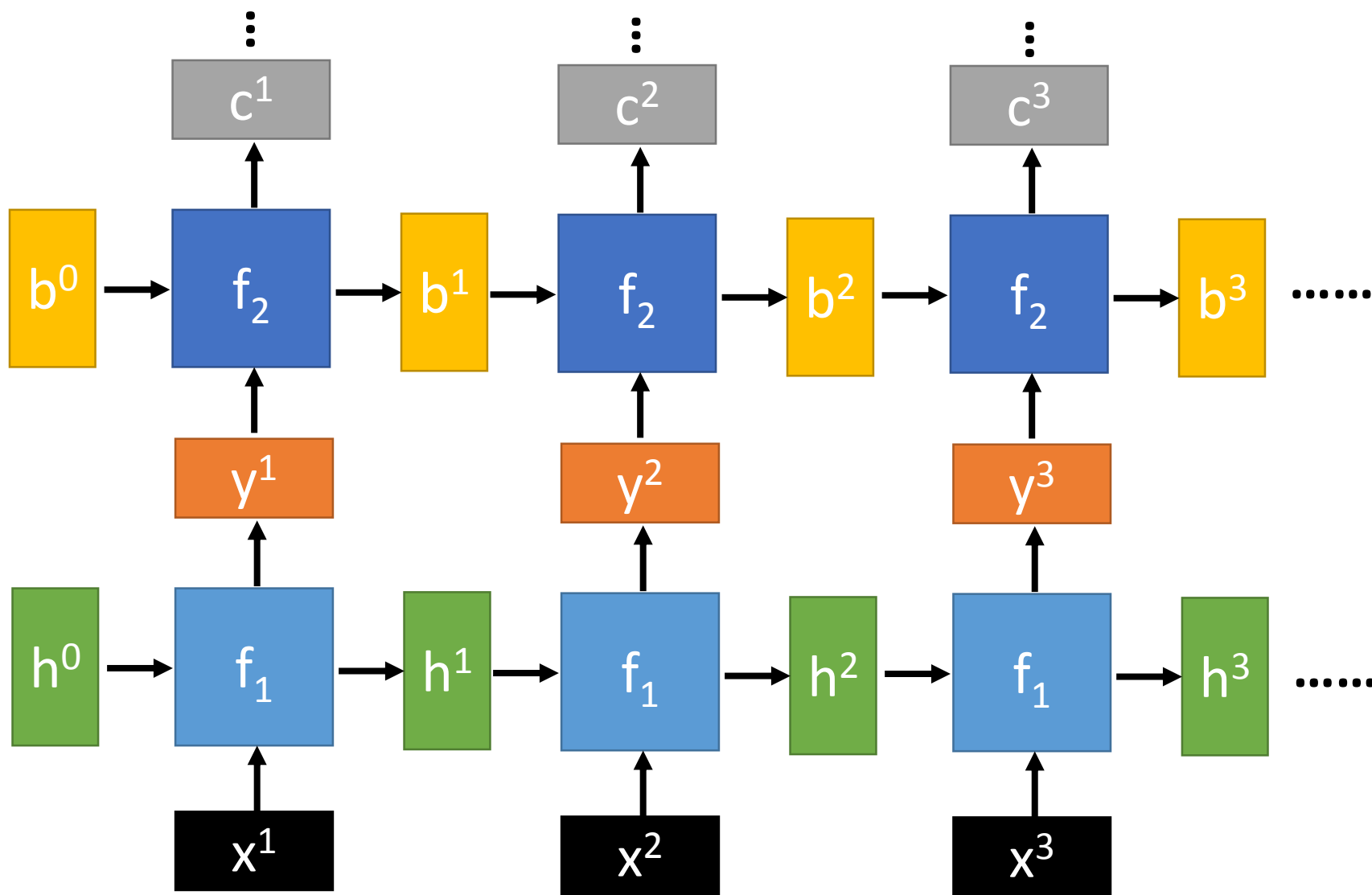
h and h' are vectors with the same dimension



No matter how long the input/output sequence is, we only need one function f

Deep RNN

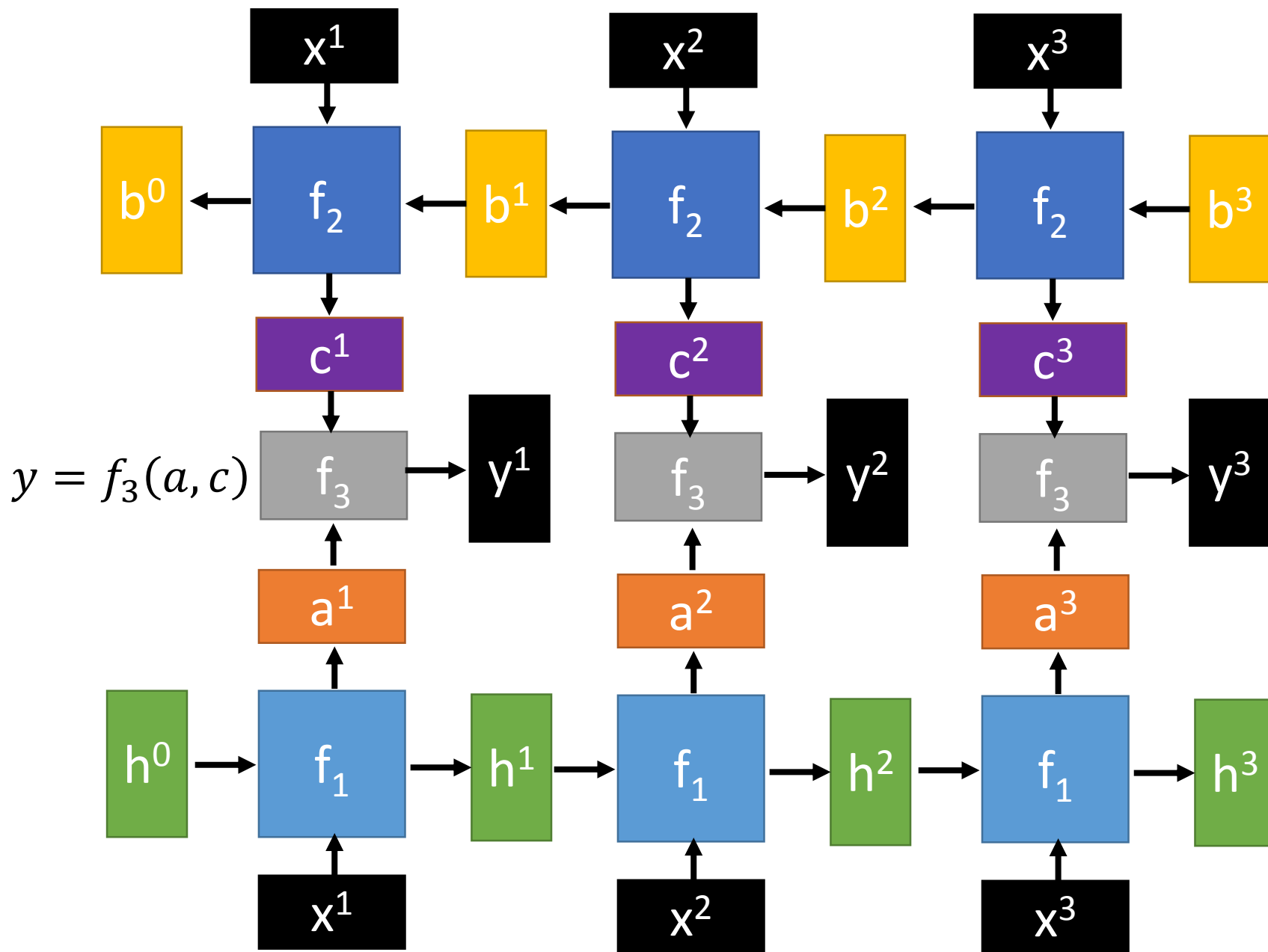
$$h', y = f_1(h, x) \quad b', c = f_2(b, y) \quad \dots$$



f1跟f2吃的state方向相反

Bidirectional RNN

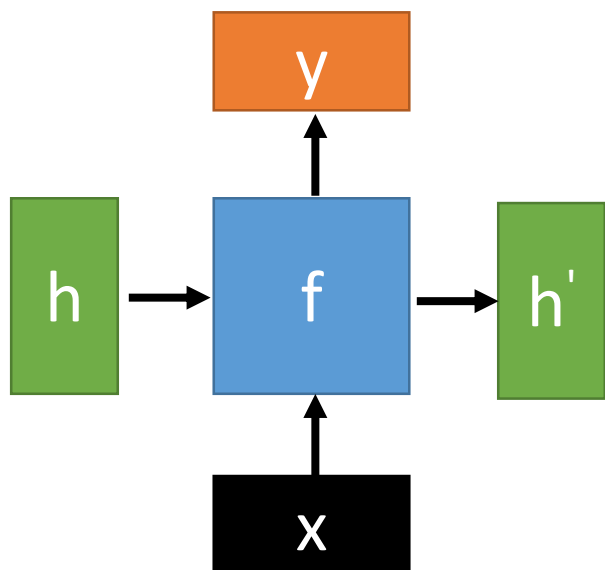
$$h', a = f_1(h, x) \quad b', c = f_2(b, x)$$



Naïve RNN

一般RNN的activation function都是用tanh instead of ReLu, 如果用ReLu效果會變差

- Given function $f: h', y = f(h, x)$



$$h' = \sigma(W^h h + W^i x)$$

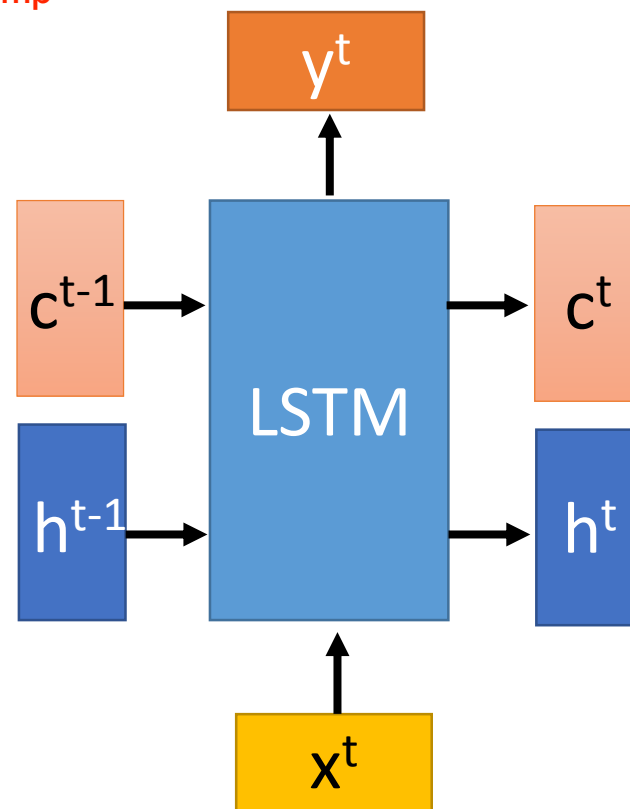
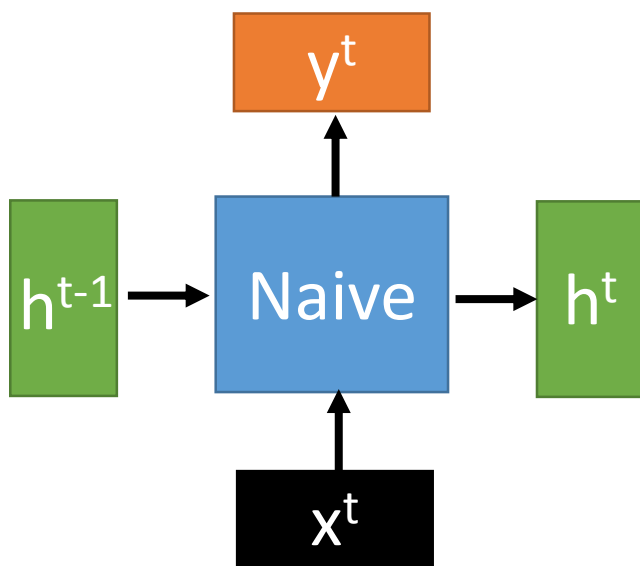
$$y = \sigma(W^o h')$$

softmax

Ignore bias here

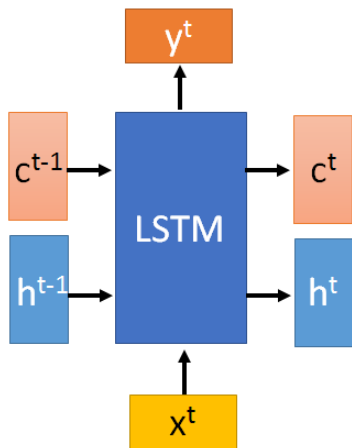
LSTM比較強的原因是因為可以記住較久以前的time stamp

LSTM

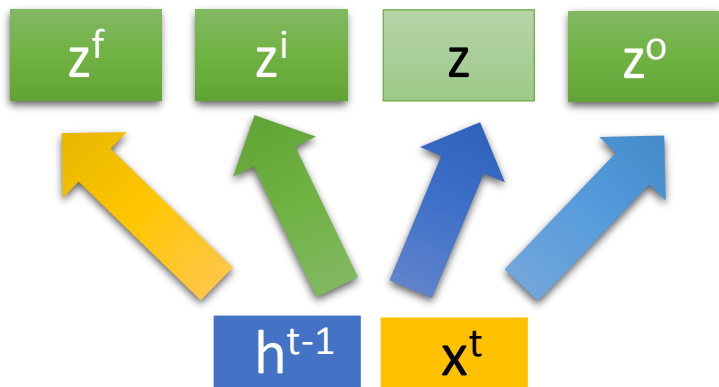


變化較慢的memory \Rightarrow c changes slowly $\Rightarrow c^t$ is c^{t-1} added by something

變化較快的memory \Rightarrow h changes faster $\Rightarrow h^t$ and h^{t-1} can be very different



c^{t-1}



四個維度相同但是value不同的transform matrix

$$z = \tanh\left(W \begin{bmatrix} x^t \\ h^{t-1} \end{bmatrix} \right)$$

$$z^i = \sigma\left(W^i \begin{bmatrix} x^t \\ h^{t-1} \end{bmatrix} \right)$$

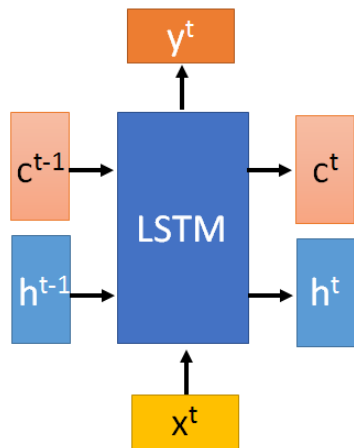
Input gate

$$z^f = \sigma\left(W^f \begin{bmatrix} x^t \\ h^{t-1} \end{bmatrix} \right)$$

forget gate

$$z^o = \sigma\left(W^o \begin{bmatrix} x^t \\ h^{t-1} \end{bmatrix} \right)$$

output gate



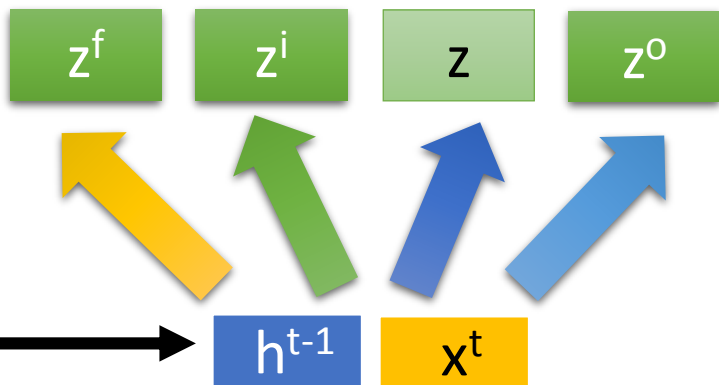
把三個串在一起呈上一個超大的transform matrix
然而為了怕參數過多造成overfitting，強迫與c相乘的部分為diagonal matrix，也就是他只能做scalar變化

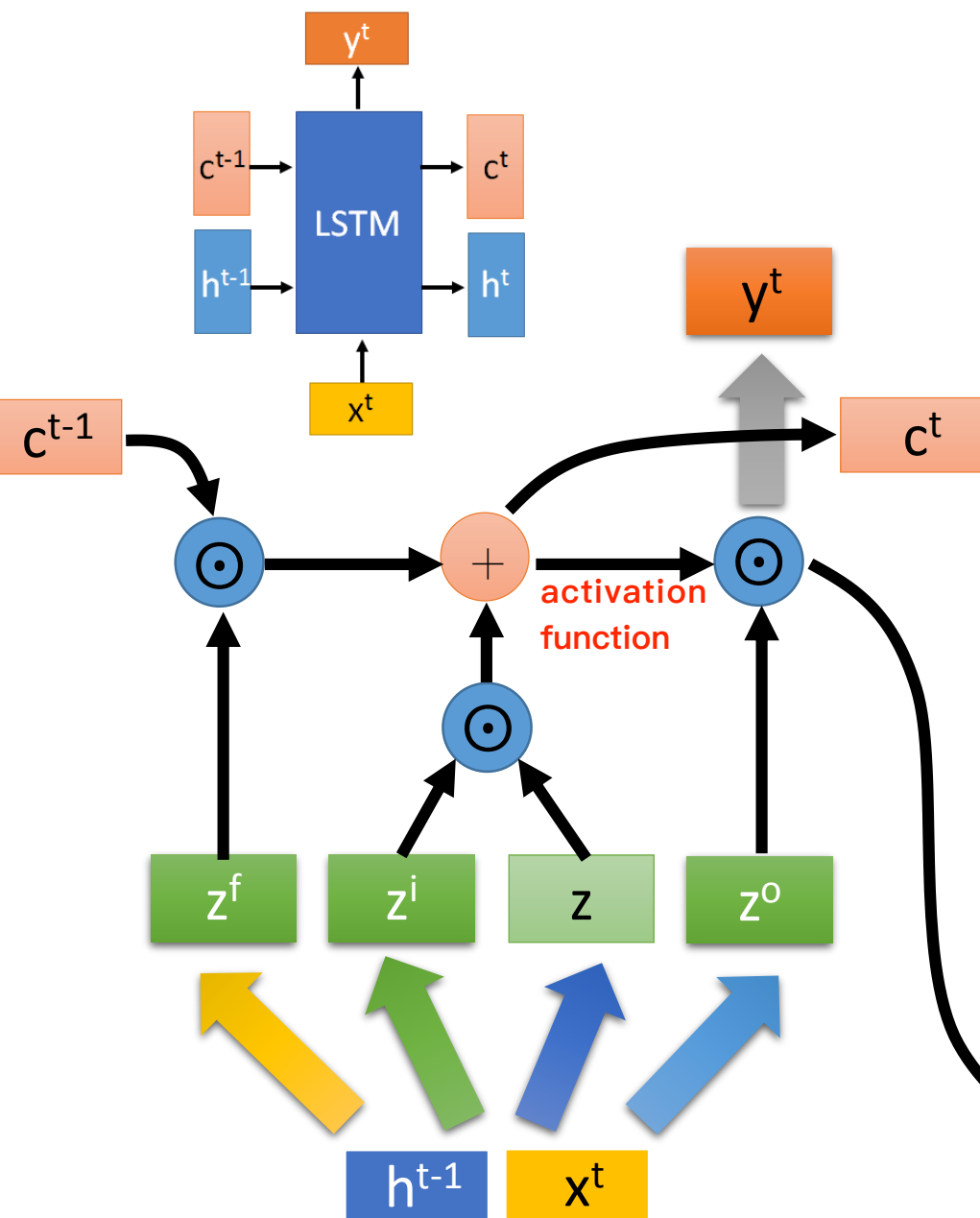
$$z = \tanh\left(\begin{bmatrix} W & \text{diagonal} \end{bmatrix} \begin{bmatrix} h^{t-1} \\ c^{t-1} \end{bmatrix} \right)$$

一種方法

“peephole”

z^o z^f z^i obtained by the same way





c 的變化比較小

$$c^t = z^f \odot c^{t-1} + z^i \odot z$$

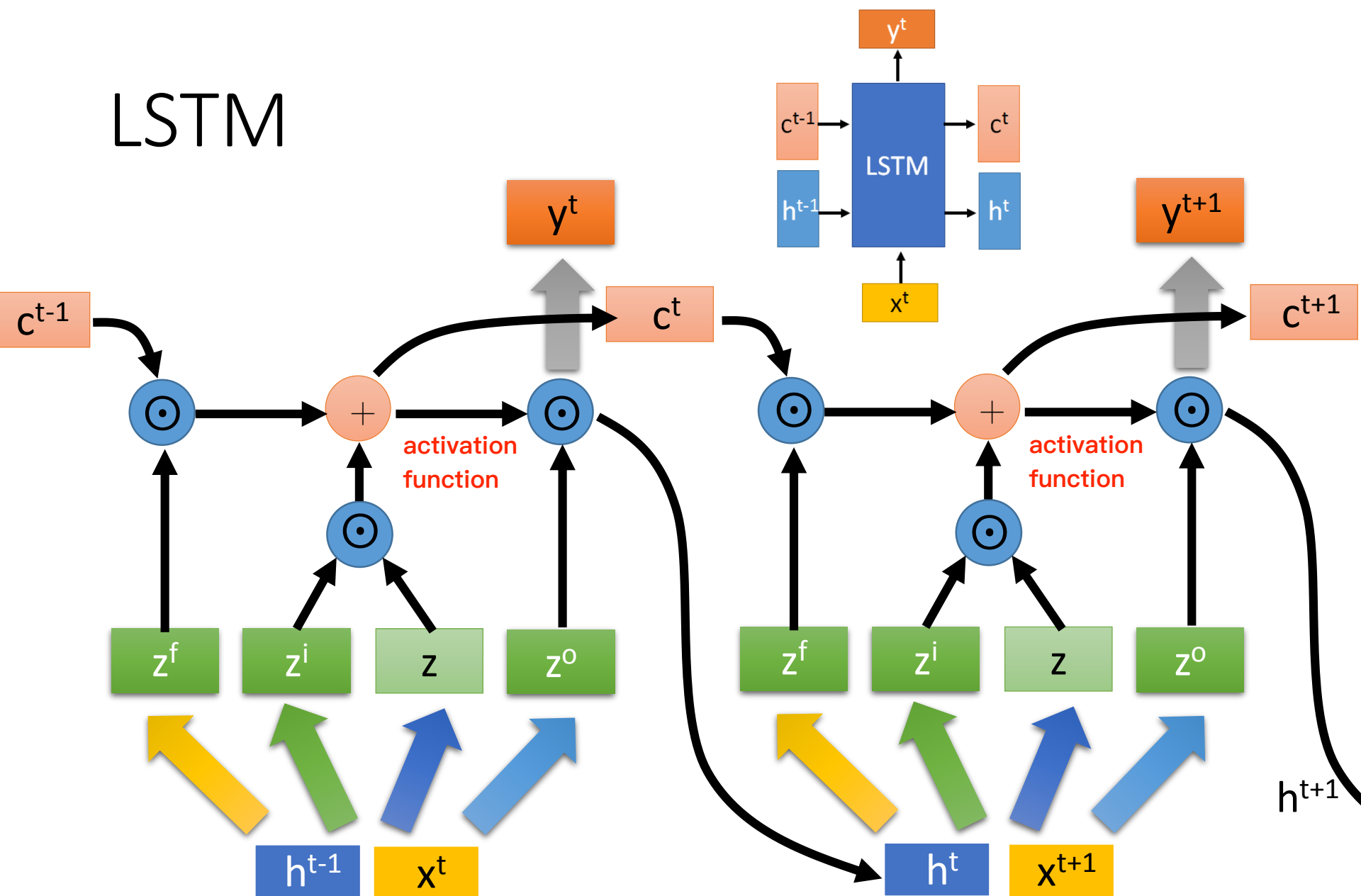
forget gate input gate

$$h^t = z^o \odot \tanh(c^t)$$

output gate

$$y^t = \sigma(W'h^t)$$

LSTM



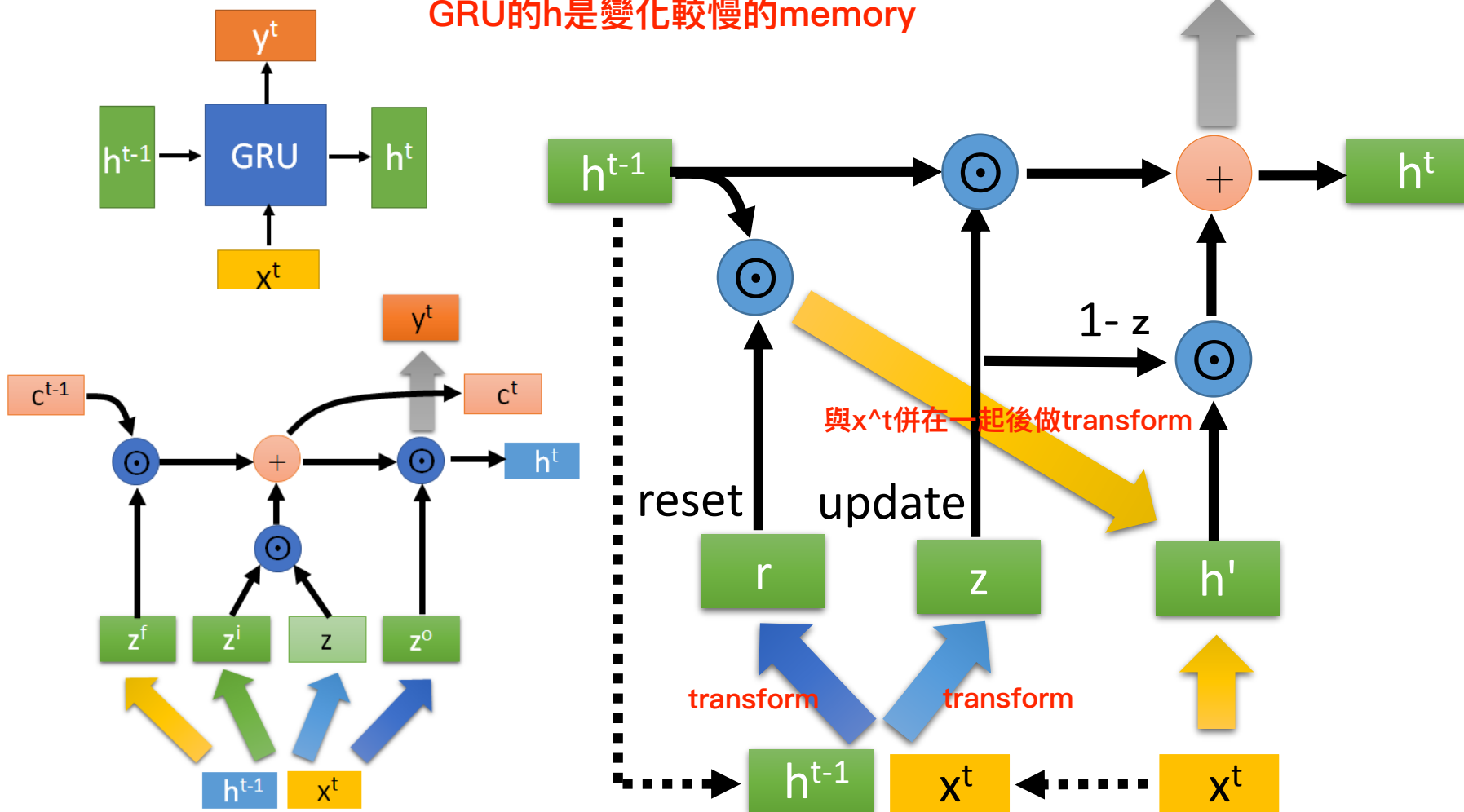
參數量減少避免overfitting

GRU

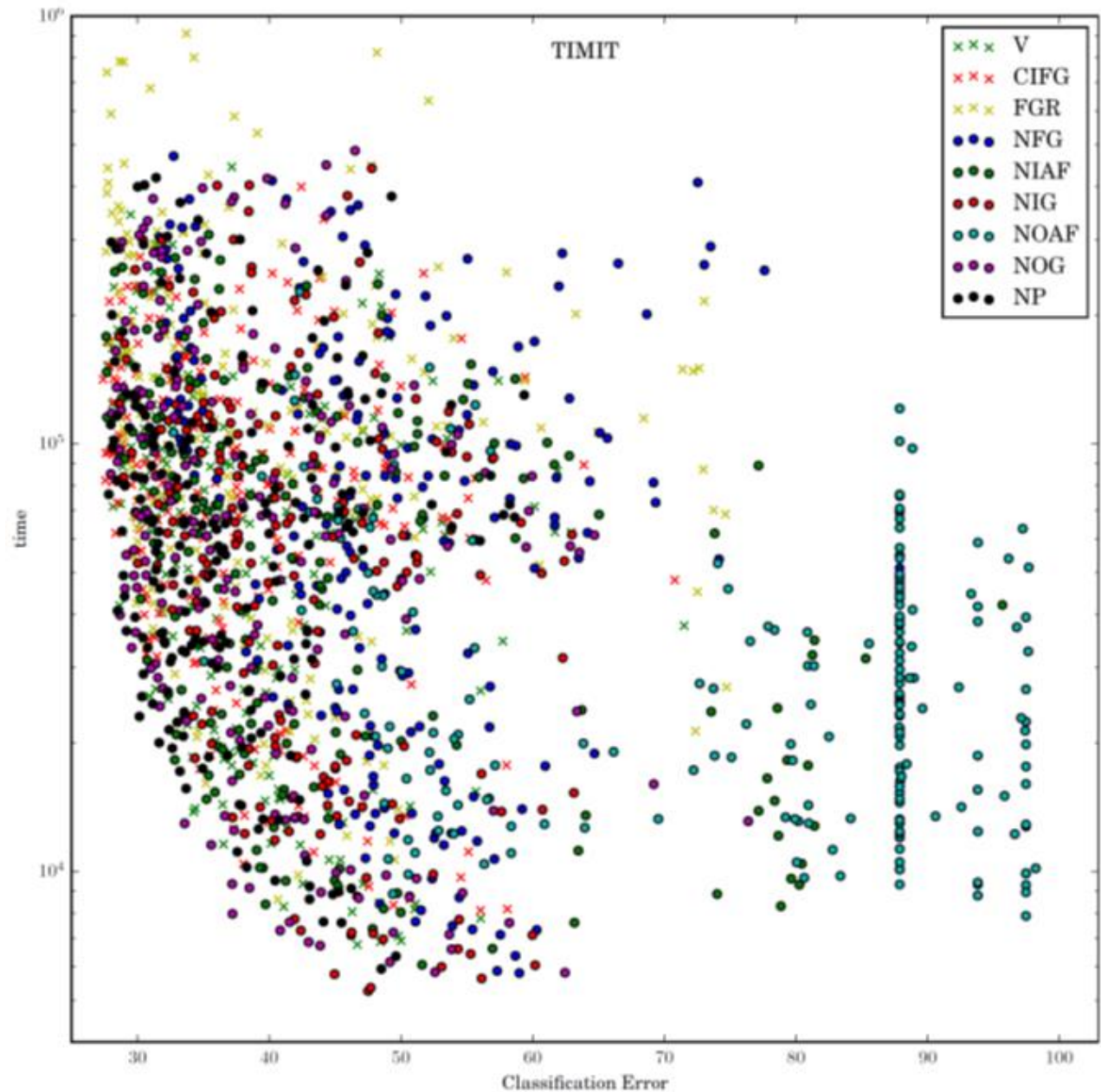
input/forget gate是連動的概念

$$h^t = z \odot h^{t-1} + (1 - z) \odot h'$$

GRU的h是變化較慢的memory

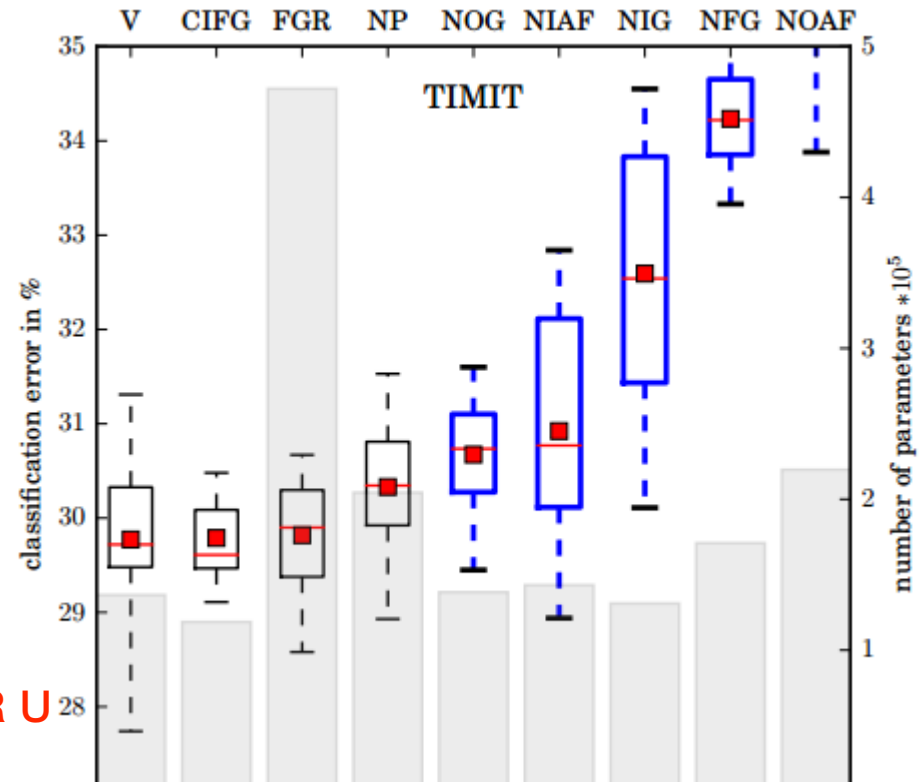


LSTM: A Search Space Odyssey



LSTM: A Search Space Odyssey

1. No Input Gate (NIG)
2. No Forget Gate (NFG)
3. No Output Gate (NOG)
4. No Input Activation Function (NIAF)
5. No Output Activation Function (NOAF)
6. No Peepholes (NP)
7. Coupled Input and Forget Gate (CIFG)
8. Full Gate Recurrence (FGR)



Standard LSTM works well

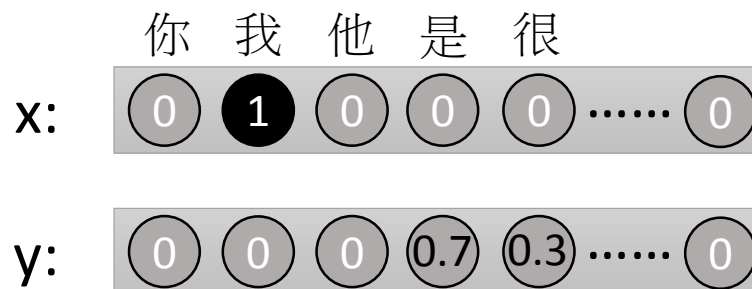
Simply LSTM: coupling input and forget gate, removing peephole

Forget gate is critical for performance

Output gate activation function is critical

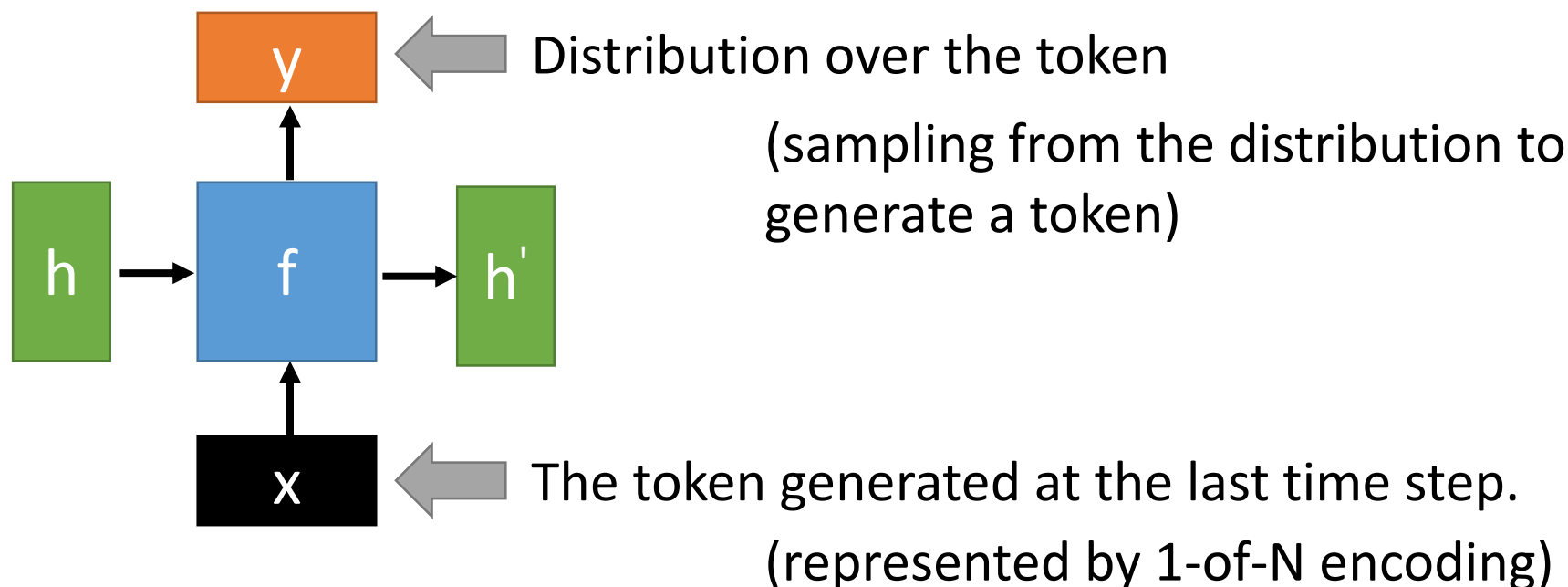
Sequence Generation

Generation



- Sentences are composed of characters/words
- Generating a character/word at each time by RNN

最後要做的事情是從y這個distribution做sampling



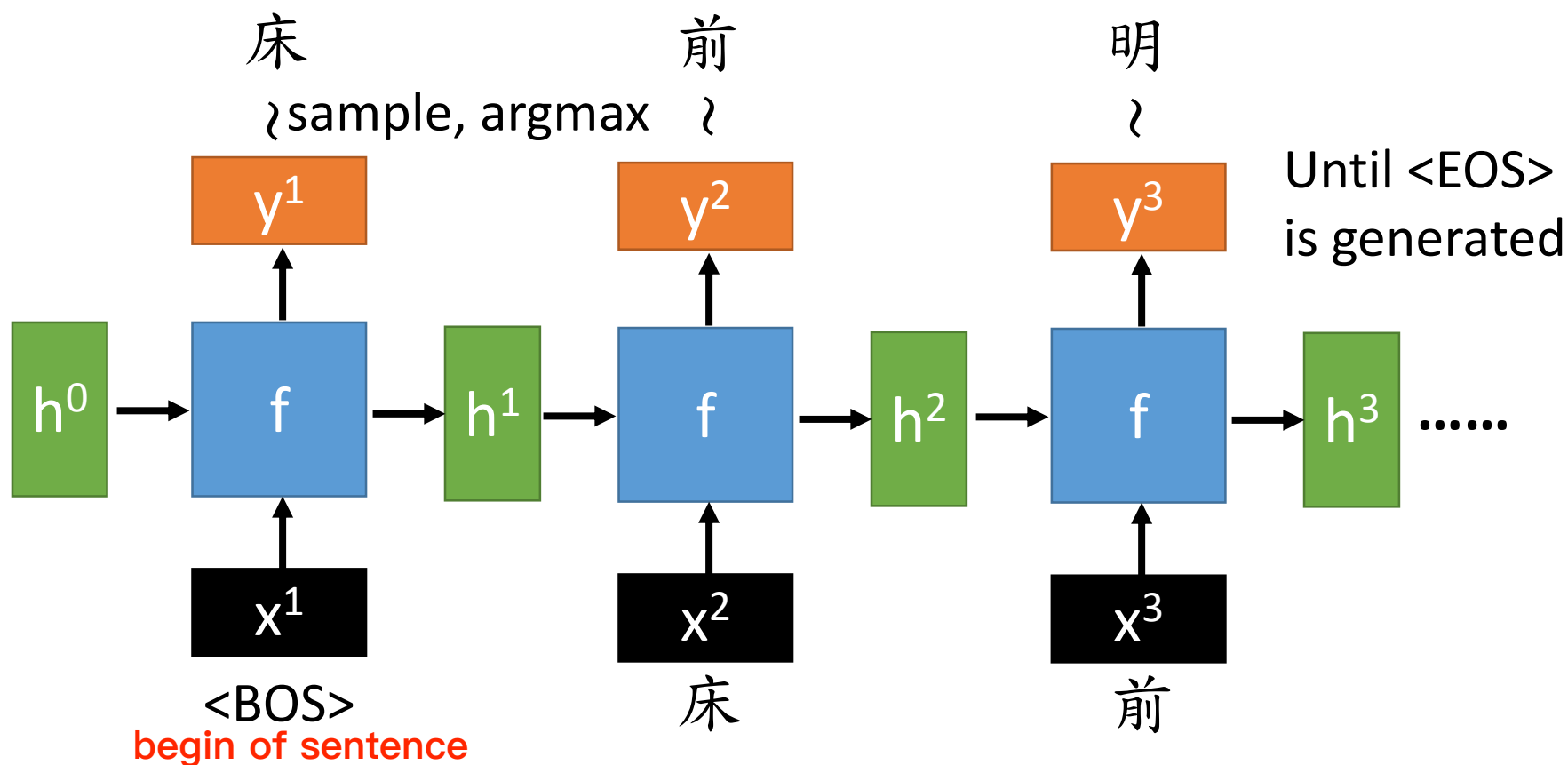
Generation

$$y^1: P(w | \langle \text{BOS} \rangle)$$

$$y^2: P(w | \langle \text{BOS} \rangle, \text{床})$$

$$y^3: P(w | \langle \text{BOS} \rangle, \text{床}, \text{前})$$

- Sentences are composed of characters/words
- Generating a character/word at each time by RNN

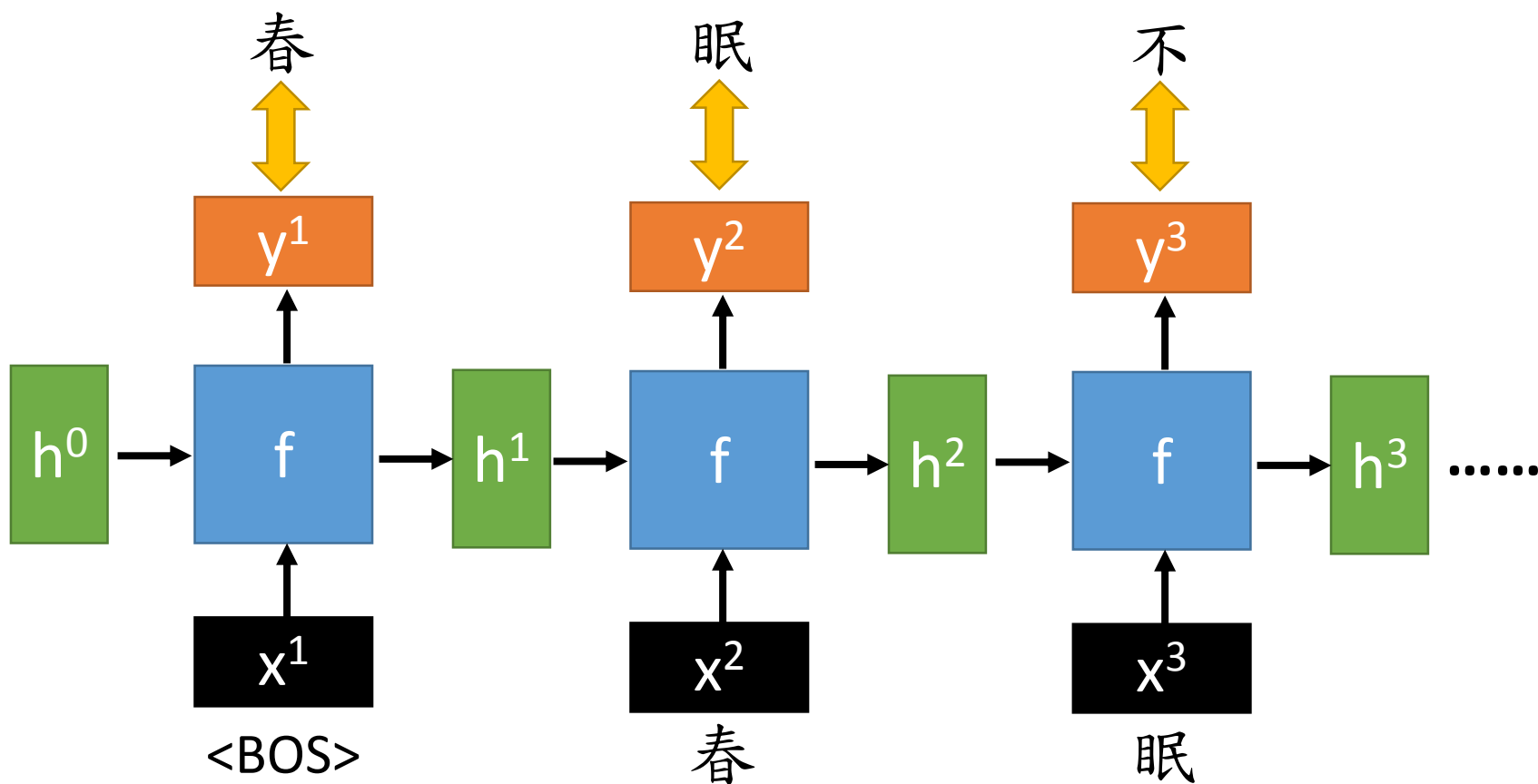


Generation

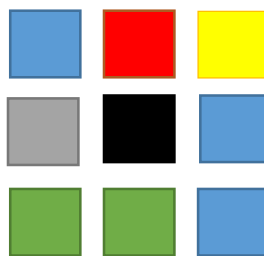
↕ : minimizing cross-entropy

- Training

Training data: 春 眠 不 覺 曉



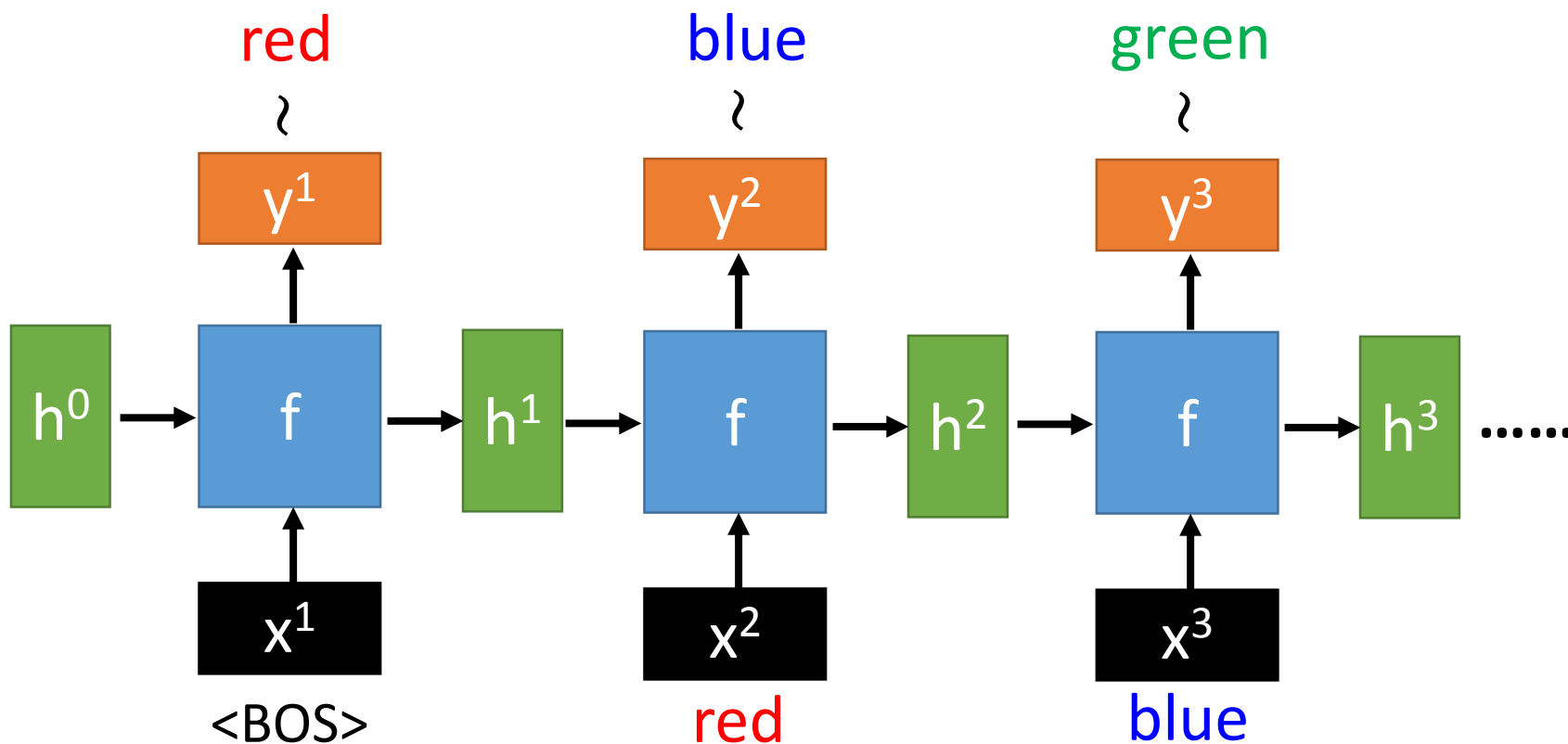
Generation



Consider as a sentence
blue red yellow gray

Train a RNN based on the
“sentences”

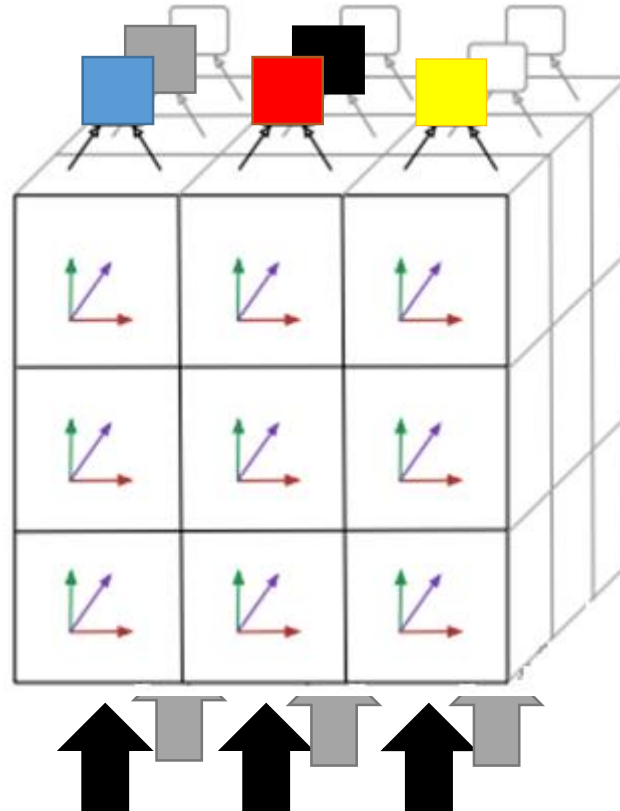
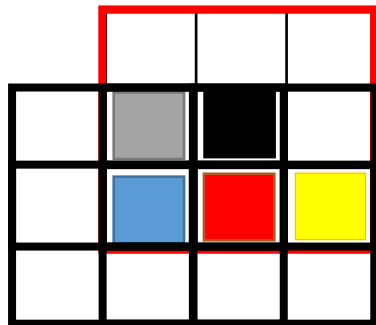
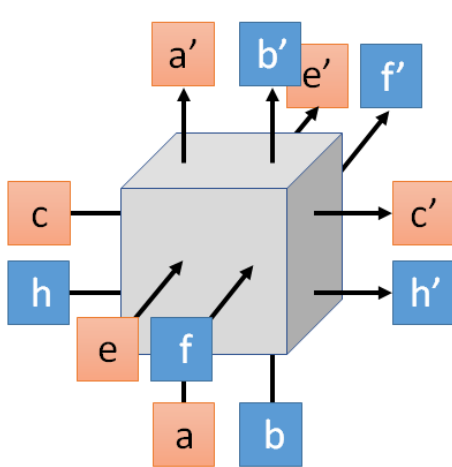
- Images are composed of pixels
- Generating a pixel at each time by RNN



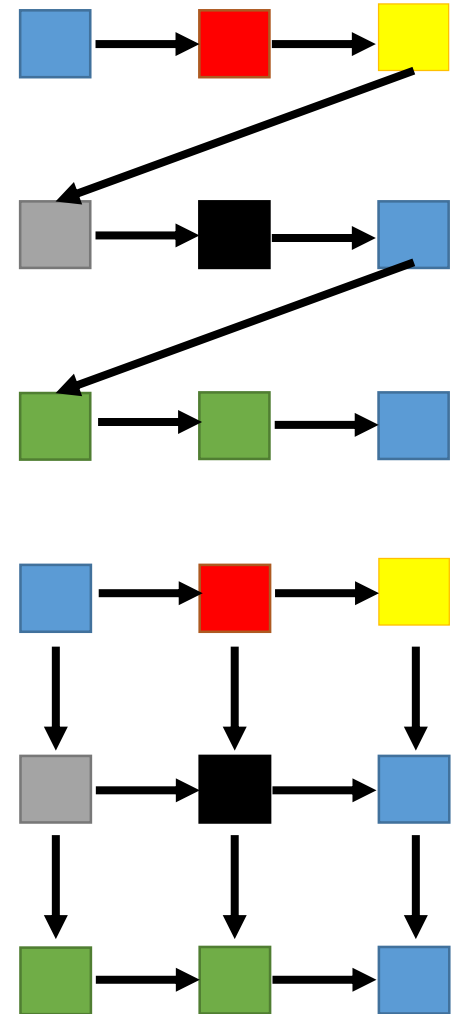
Generation - PixelRNN

只看上下左右附近的pixel

- Images are composed of pixels



3 x 3 images



Conditional Sequence Generation

Conditional Generation

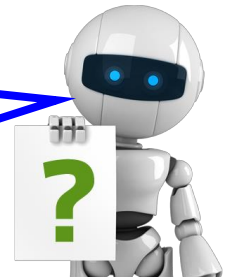
- We don't want to simply generate some random sentences.
- Generate sentences based on conditions:

Caption Generation

Given
condition:



"A young girl
is dancing."



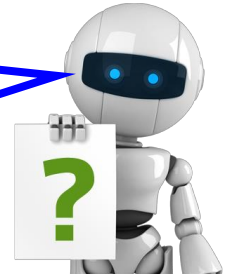
Chat-bot

Given
condition:



"Hello"

"Hello. Nice
to see you."

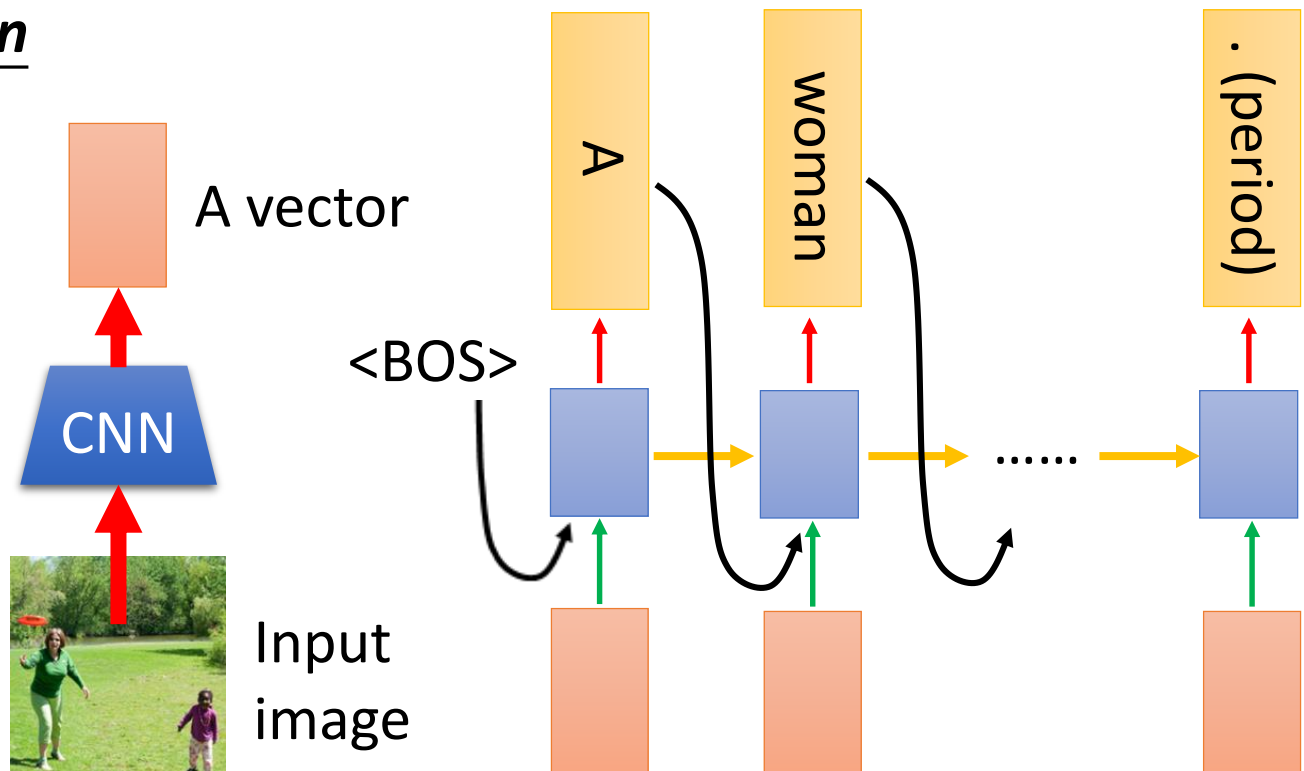


Conditional Generation

為了避免machine忘記input image的vector，在
每一個time stamp都餵進去image轉換的vector

- Represent the input condition as a vector, and consider the vector as the input of RNN generator

Image Caption Generation



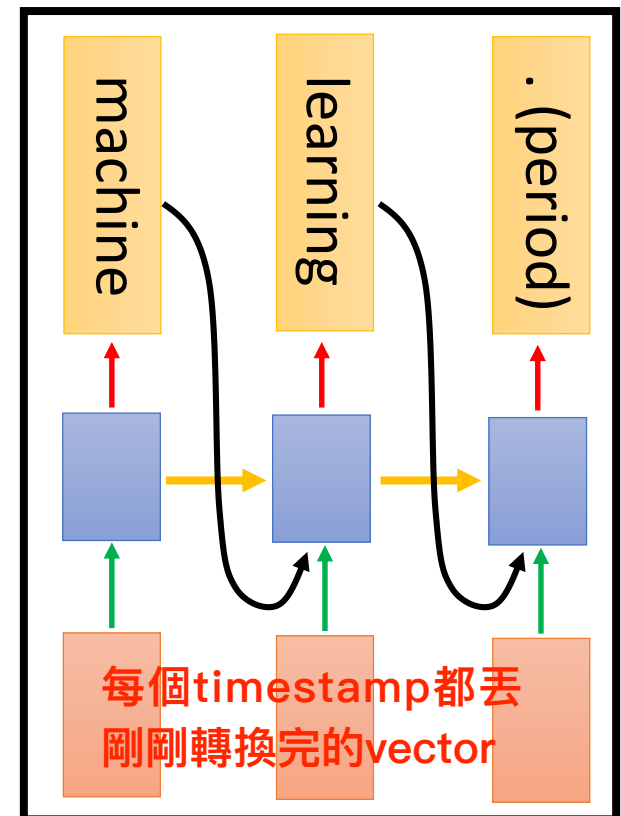
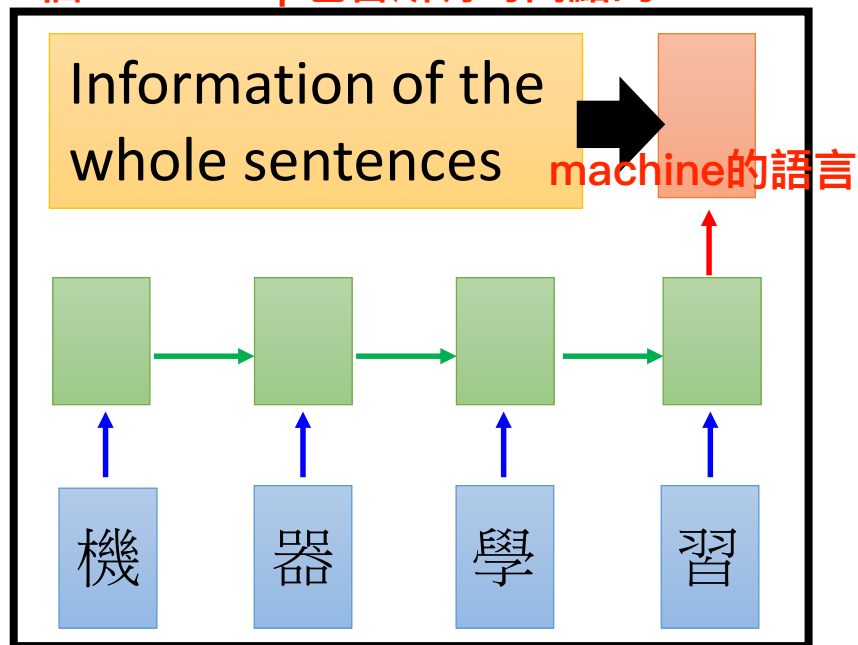
想辦法把input image轉換成一個vector

Conditional Generation

Sequence-to-sequence learning

- Represent the input condition as a vector, and consider the vector as the input of RNN generator

- E.g. Machine translation / Chat-bot
最後一個timestamp包含所有時間點的information



Encoder

← Jointly train →

Decoder

Conditional Generation

M: Hello

U: Hi

M: Hi

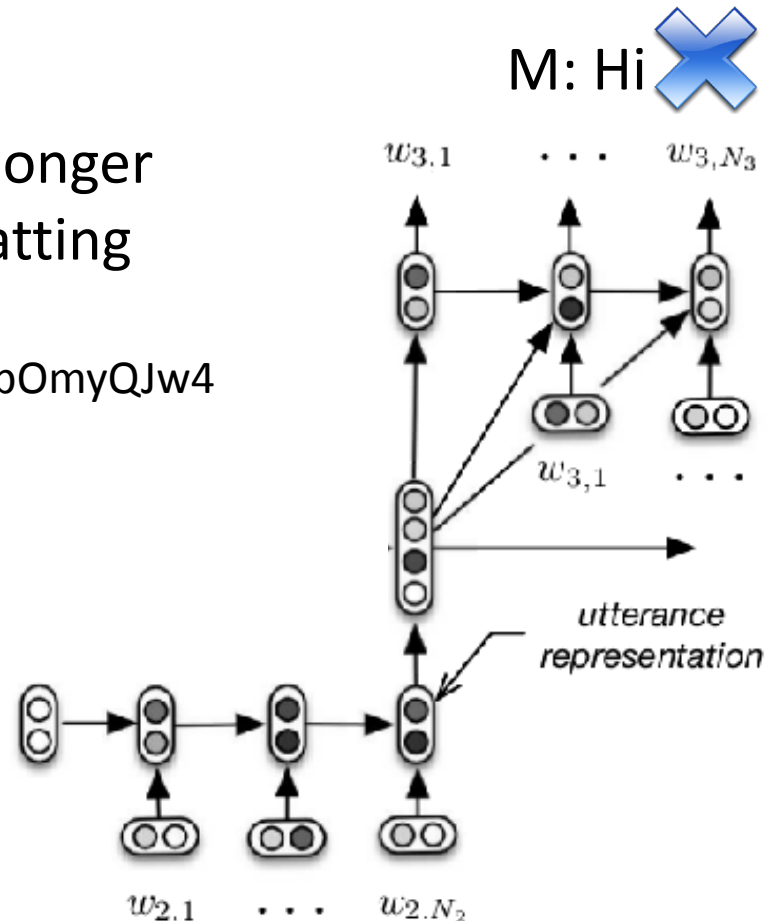
<https://www.youtube.com/watch?v=e2MpOmyQJw4>

machine不只要看當下的input vector，
還要包含觀看history產生過的句子

M: Hello

U: Hi

Need to consider longer
context during chatting

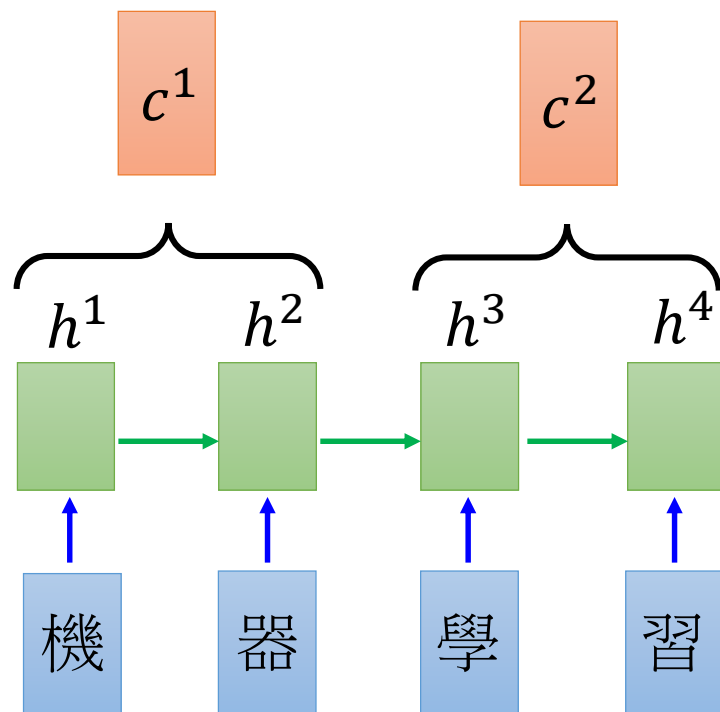


原本每個timestamp都吃encoder最後一個timestamp output的結果，但是我們沒有辦法保證所有的資訊都可以被encode盡最後一個vector，因此希望每個timestamp有decoder自己決定的input vector

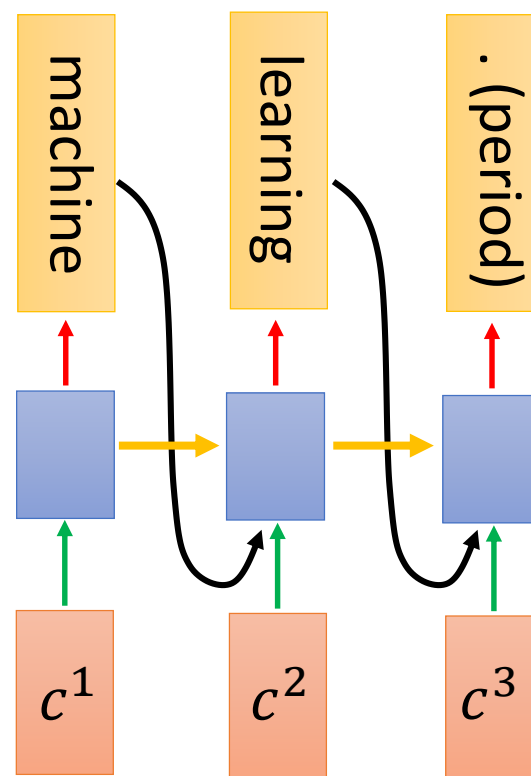
Dynamic Conditional Generation

Attention based model

decoder在每個timestamp吃到的information都是不一樣的



Encoder

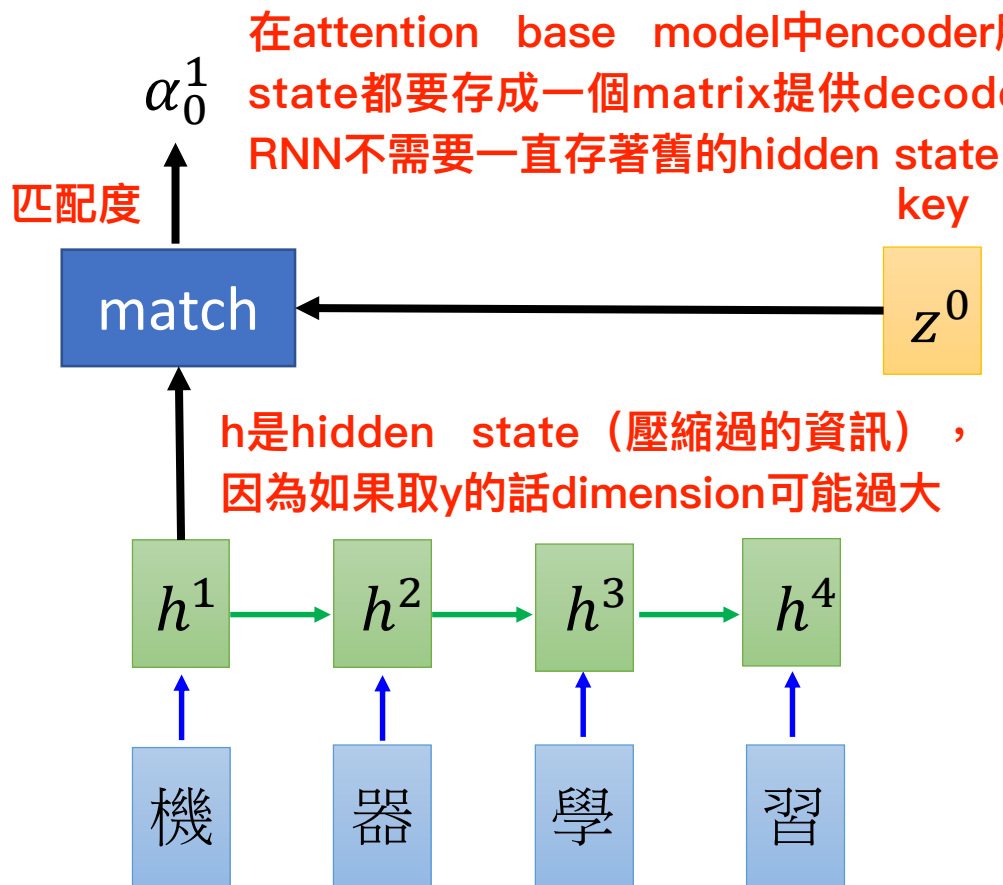


Decoder

每次input vector都是decoder從encoder output的database
搜尋找出相關的information轉換成vector

Machine Translation

- Attention-based model



與encoder/decoder同時學出來的

Jointly learned with other part of the network

α

match

h z

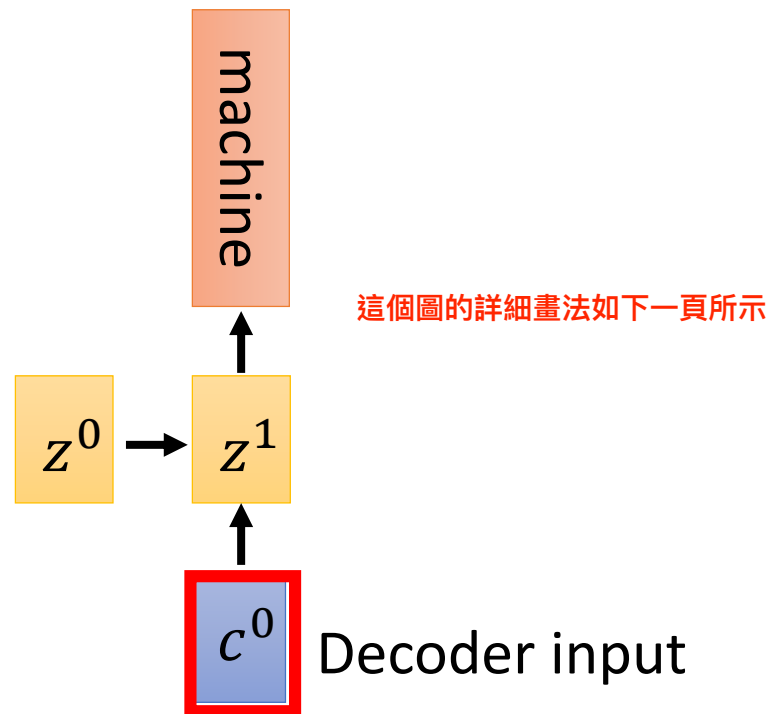
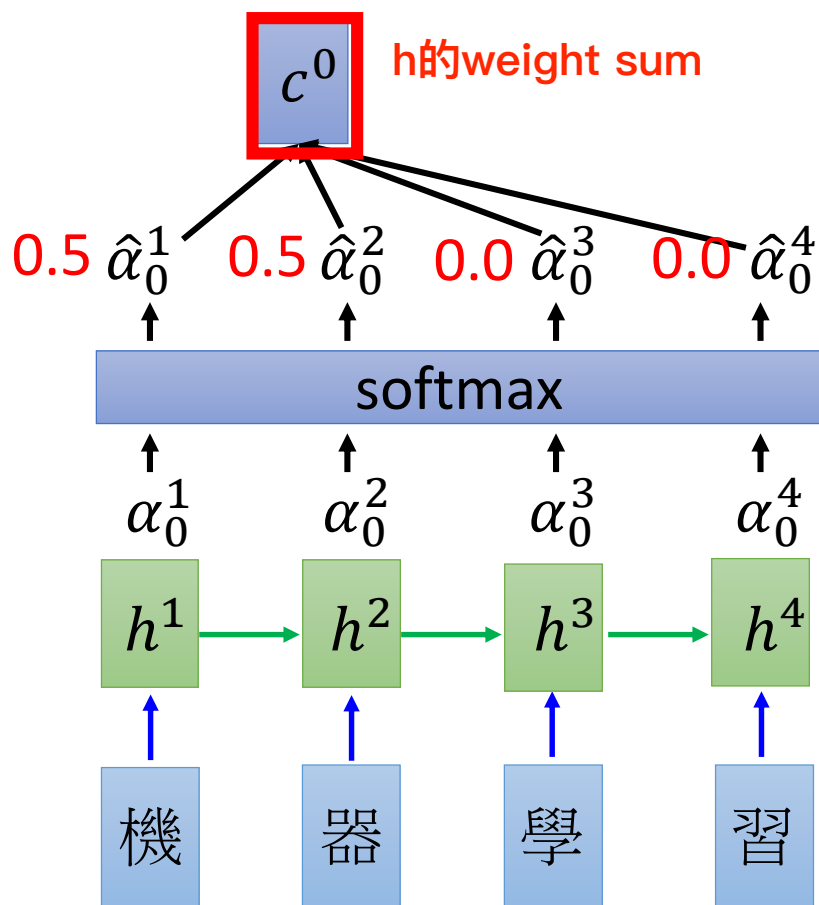
What is match ?

Design by yourself

- Cosine similarity of z and h inner product
- Small NN whose input is z and h , output a scalar
- $\alpha = h^T W z$
做一個transform

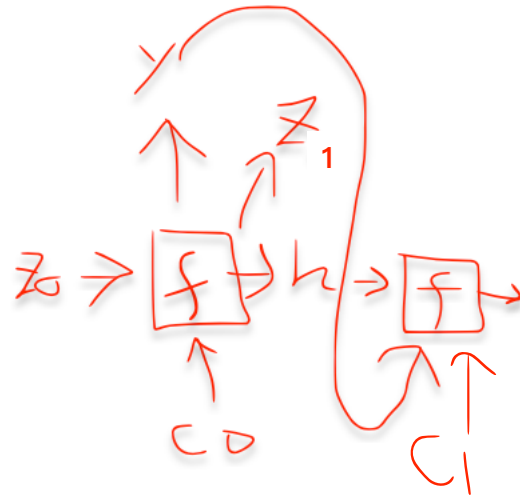
Machine Translation

- Attention-based model



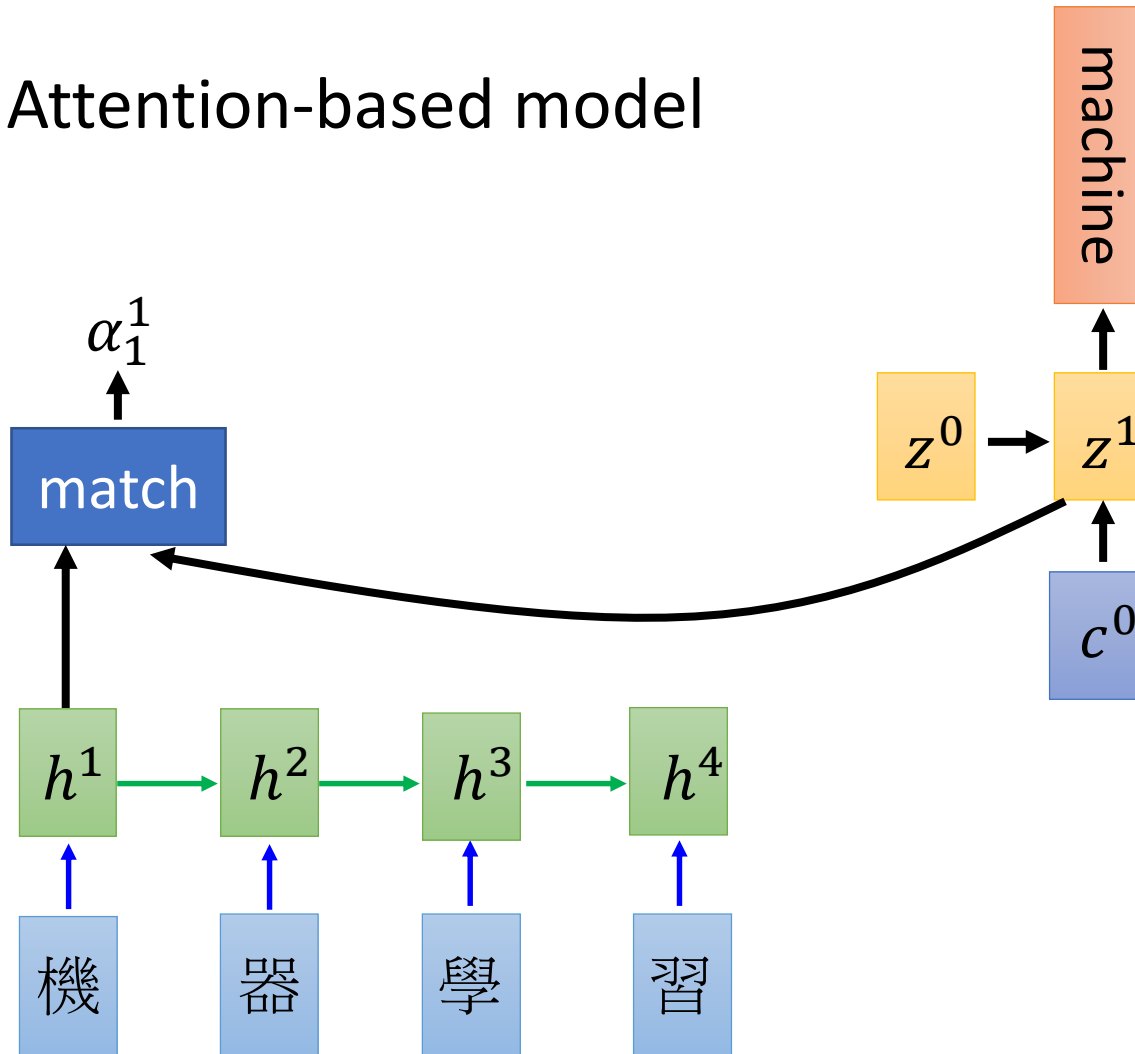
這個圖的詳細畫法如下一頁所示

$$c^0 = \sum \hat{\alpha}_0^i h^i$$
$$= 0.5h^1 + 0.5h^2$$



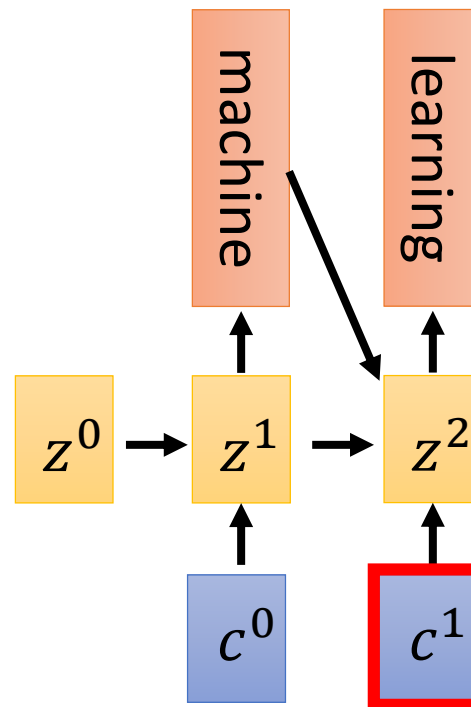
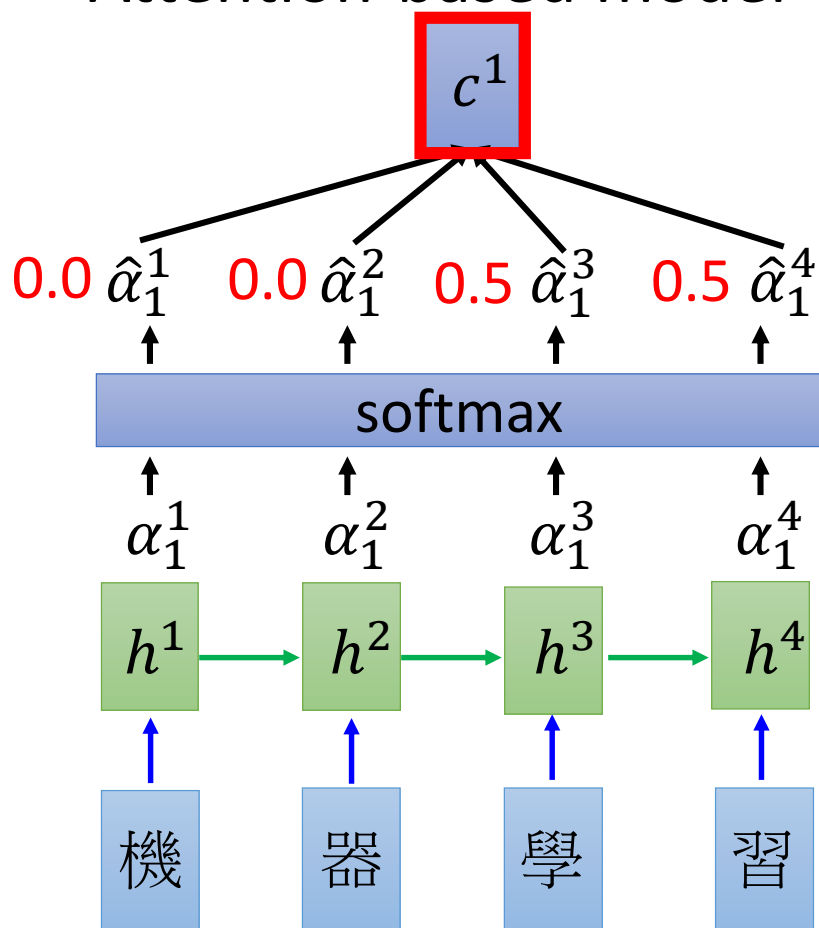
Machine Translation

- Attention-based model



Machine Translation

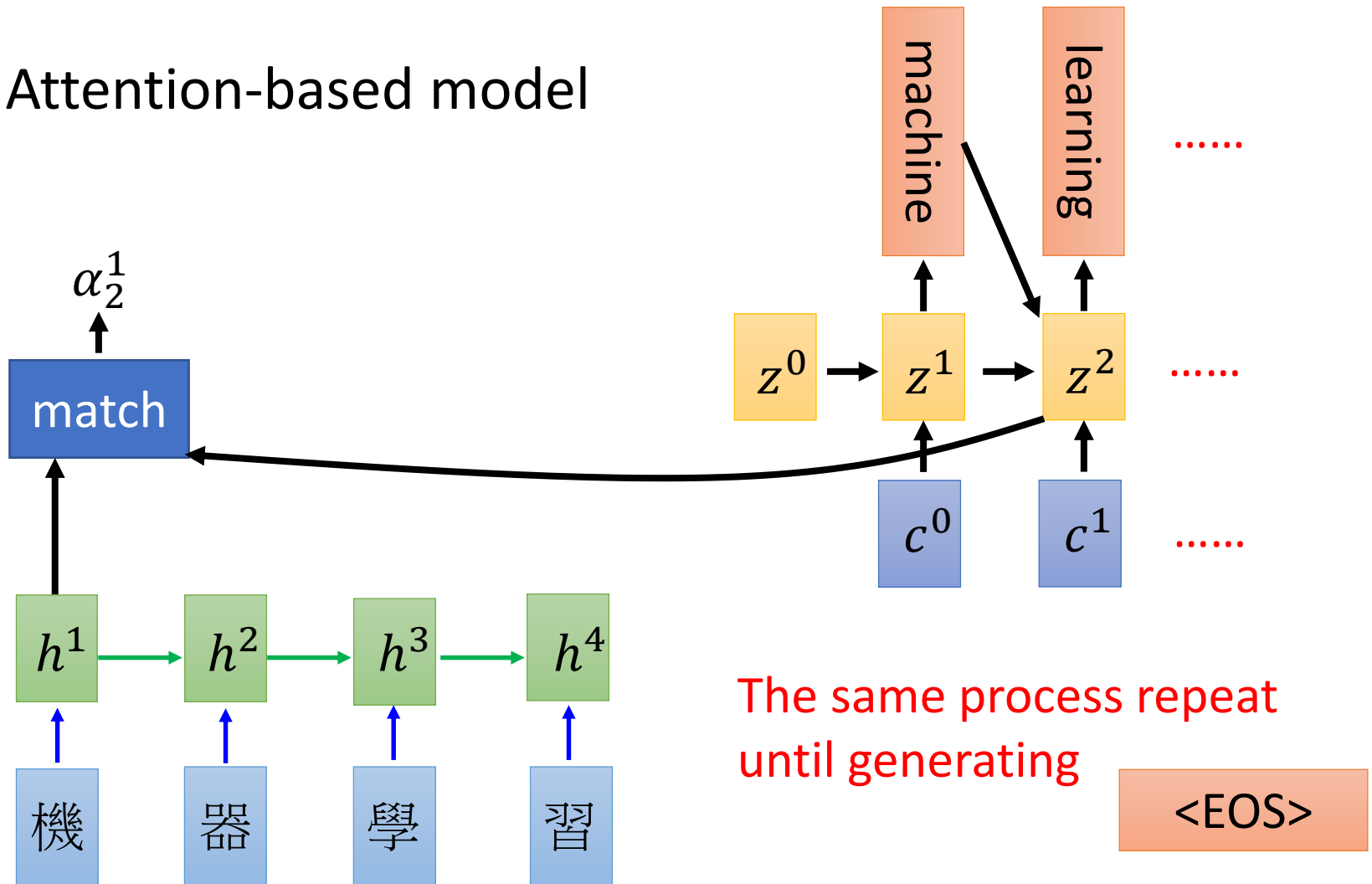
- Attention-based model



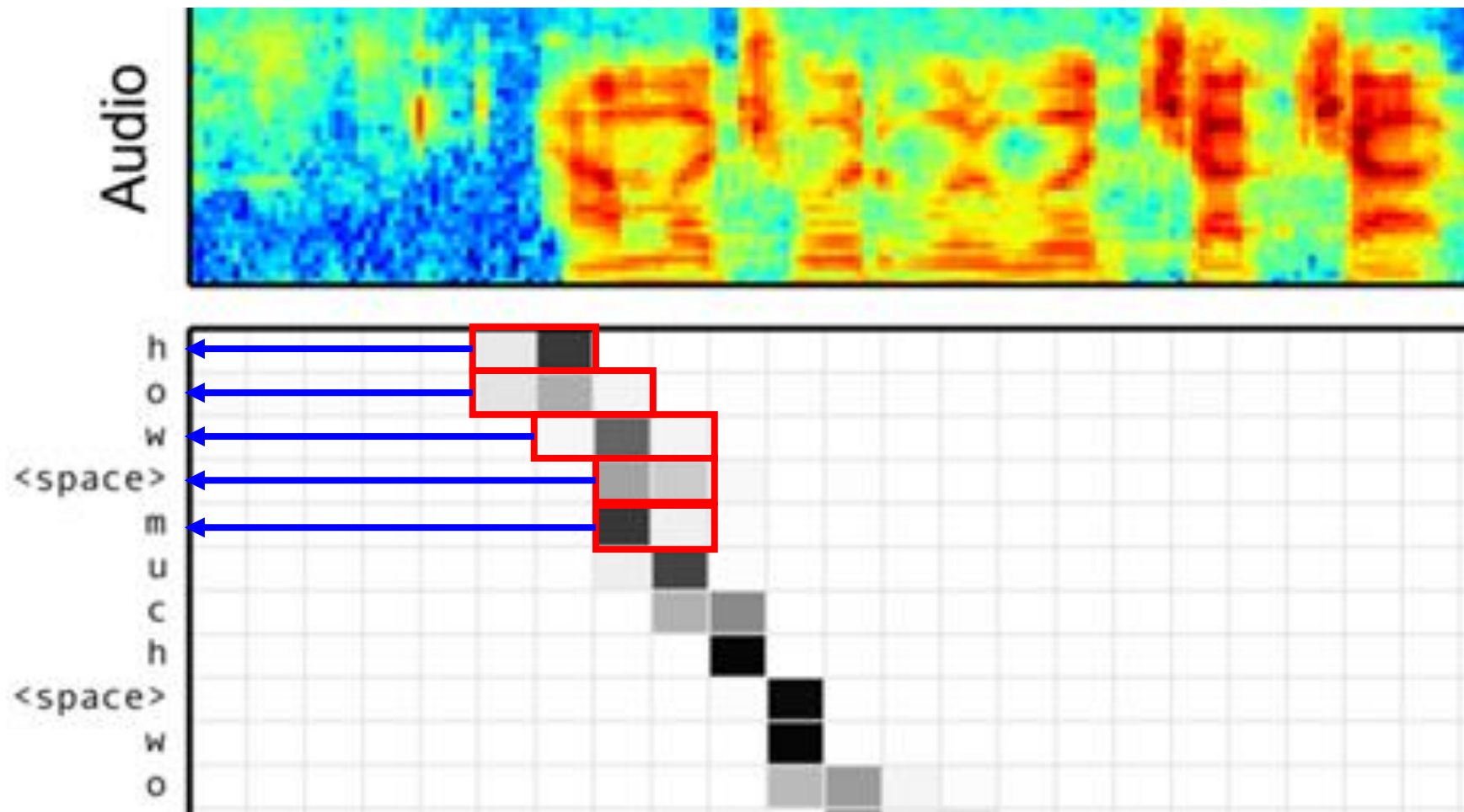
$$c^1 = \sum \hat{\alpha}_1^i h^i$$
$$= 0.5h^3 + 0.5h^4$$

Machine Translation

- Attention-based model



Speech Recognition



Model	Clean WER	Noisy WER
CLDNN-HMM [22]	8.0	8.9
LAS	14.1	16.5
LAS + LM Rescoring	10.3	12.0

William Chan, Navdeep Jaitly, Quoc V. Le, Oriol Vinyals, “Listen, Attend and Spell”, ICASSP, 2016

Image Caption Generation

把image先做segmentation後轉換成一把vector (sequence) ，
就可以套用attention based model(RNN)

A vector for
each region

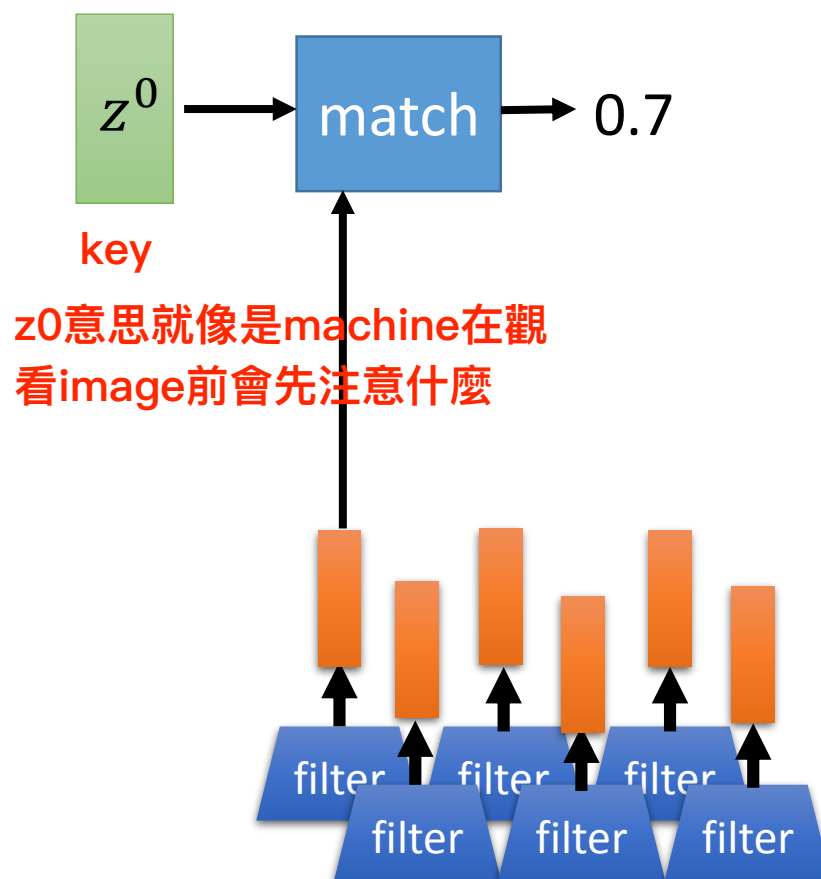
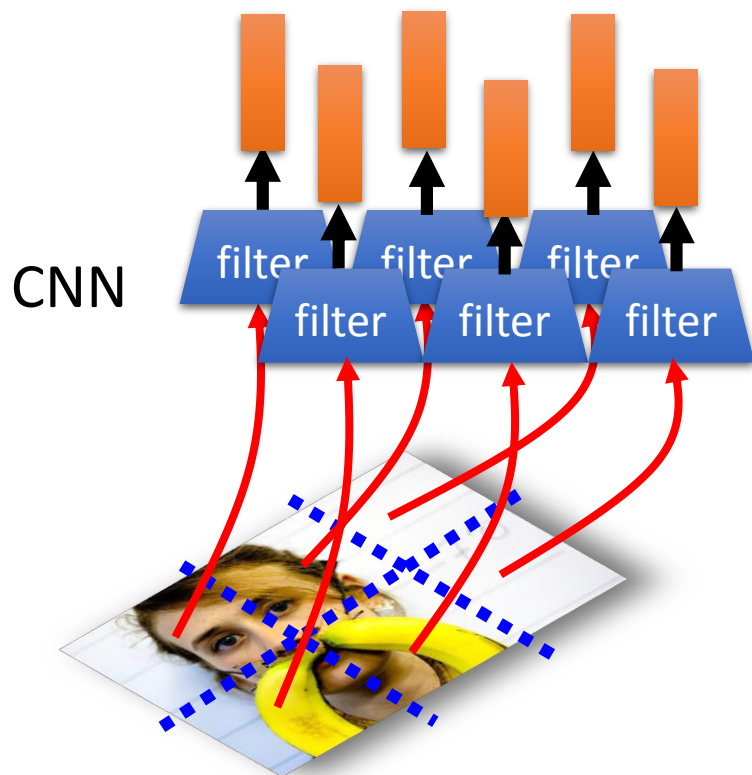


Image Caption Generation

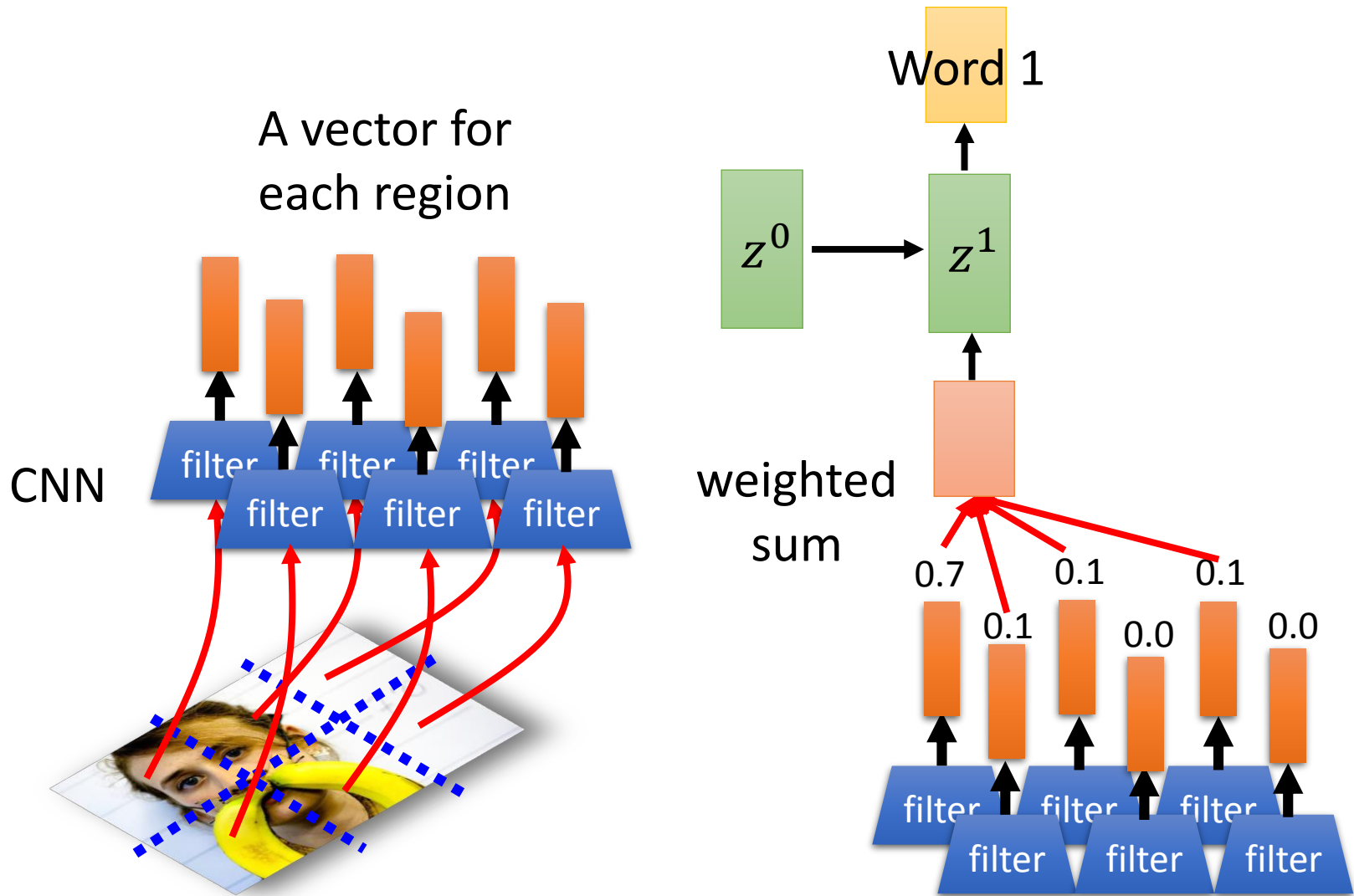


Image Caption Generation

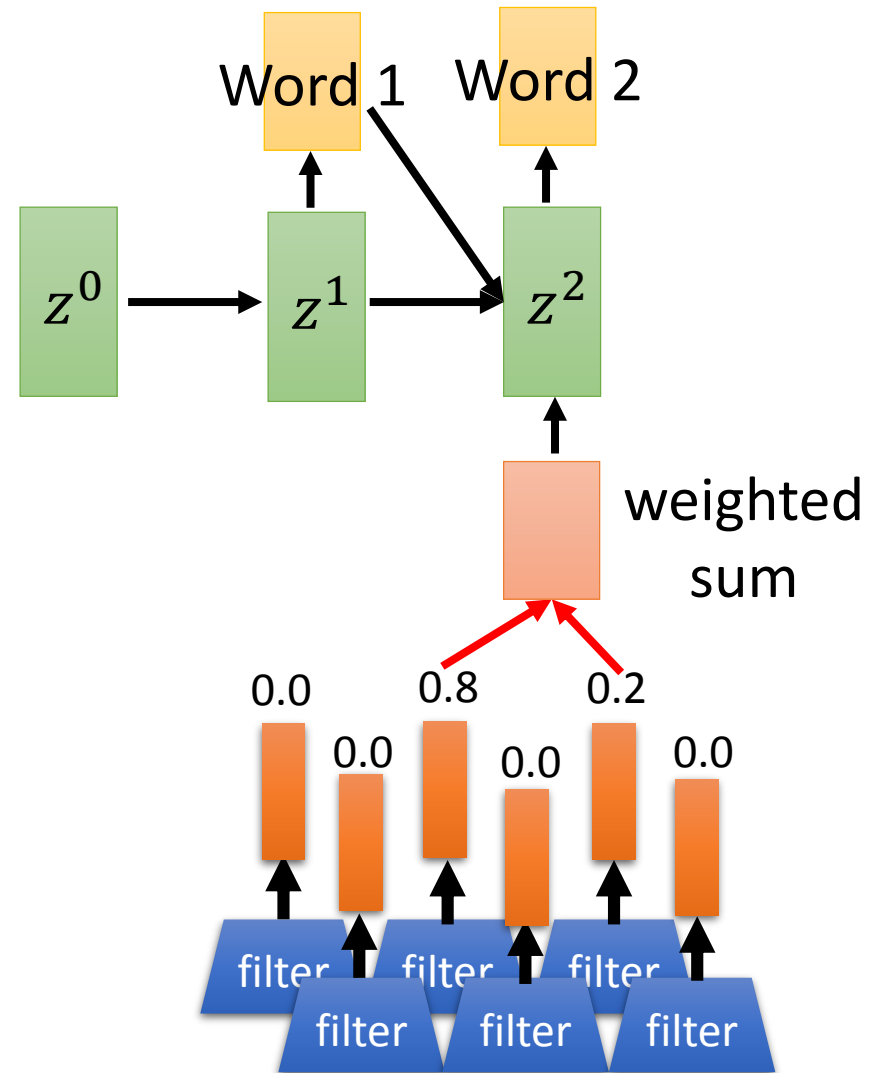
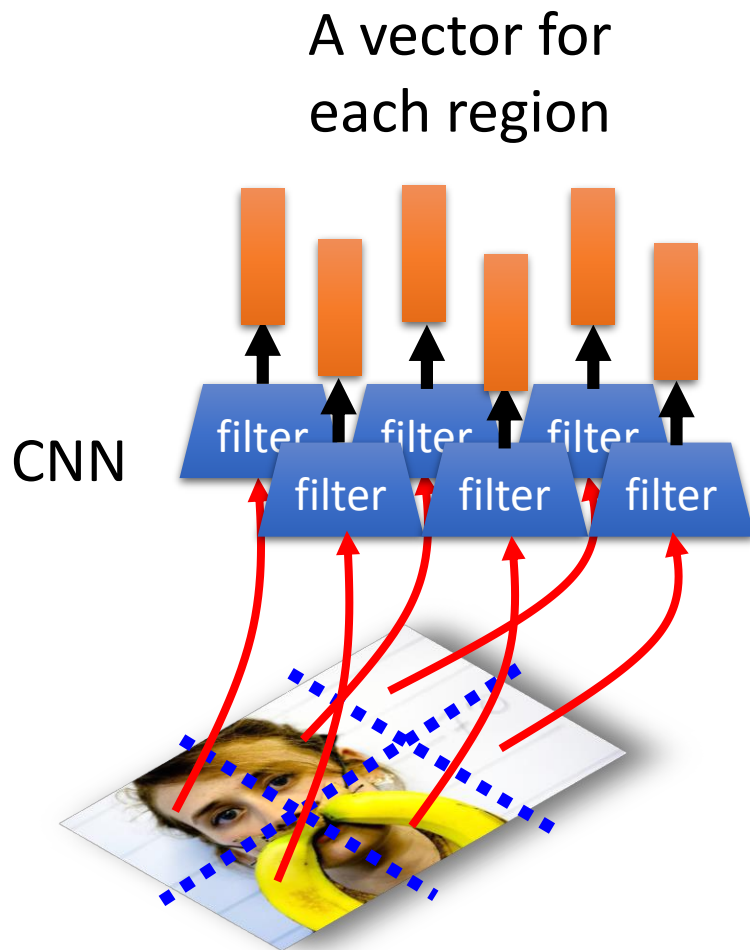
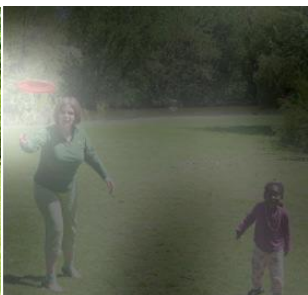


Image Caption Generation



A woman is throwing a frisbee in a park.



A dog is standing on a hardwood floor.



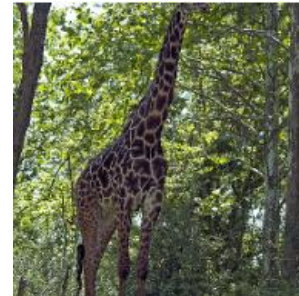
A stop sign is on a road with a mountain in the background.



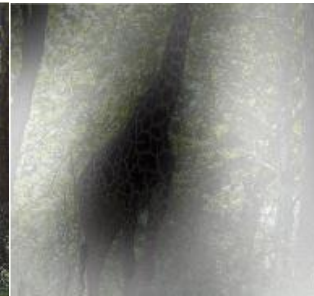
A little girl sitting on a bed with a teddy bear.



A group of people sitting on a boat in the water.



A giraffe standing in a forest with trees in the background.



Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, Yoshua Bengio, "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention", ICML, 2015

Image Caption Generation



A large white bird standing in a forest.



A woman holding a clock in her hand.



A man wearing a hat and a hat on a skateboard.



A person is standing on a beach with a surfboard.



A woman is sitting at a table with a large pizza.



A man is talking on his cell phone while another man watches.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, Yoshua Bengio, "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention", ICML, 2015



可以使用VGG19的convolutional layer最後一層找出的feature map當作segmentation
這些feature map就像是一把vector

Ref: A man and a woman ride a motorcycle

A **man** and a **woman** are **talking** on the **road**



Ref: A woman is frying food

Someone is **frying** a **fish** in a **pot**

Tips for Generation

希望machine在觀看video的時候不是只特定專注在某個frame，希望每個frame的attention要能夠平均分佈，避免machine指透過一個frame產生整個句子

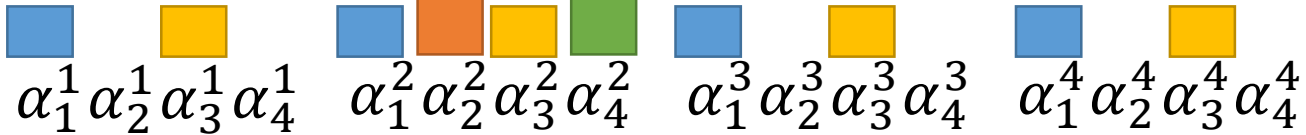
Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, Yoshua Bengio, "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention", ICML, 2015

Attention

component
 α_t^i
time



Bad
Attention



w_1 w_2 (woman) w_3 w_4 (woman) no cooking

Good Attention: each input component has approximately the same attention weight

E.g. Regularization term: $\sum_i \left(\tau - \sum_t \alpha_t^i \right)^2$

因此可以加個regularization，強迫一個frame在所有time stamp總合能夠接近某個數值tau

For each component Over the generation


在training的時候每個timestamp的input我們都是將正確答案餵進去，然而在testing的時候我們沒有正確答案

Mismatch between Train and Test

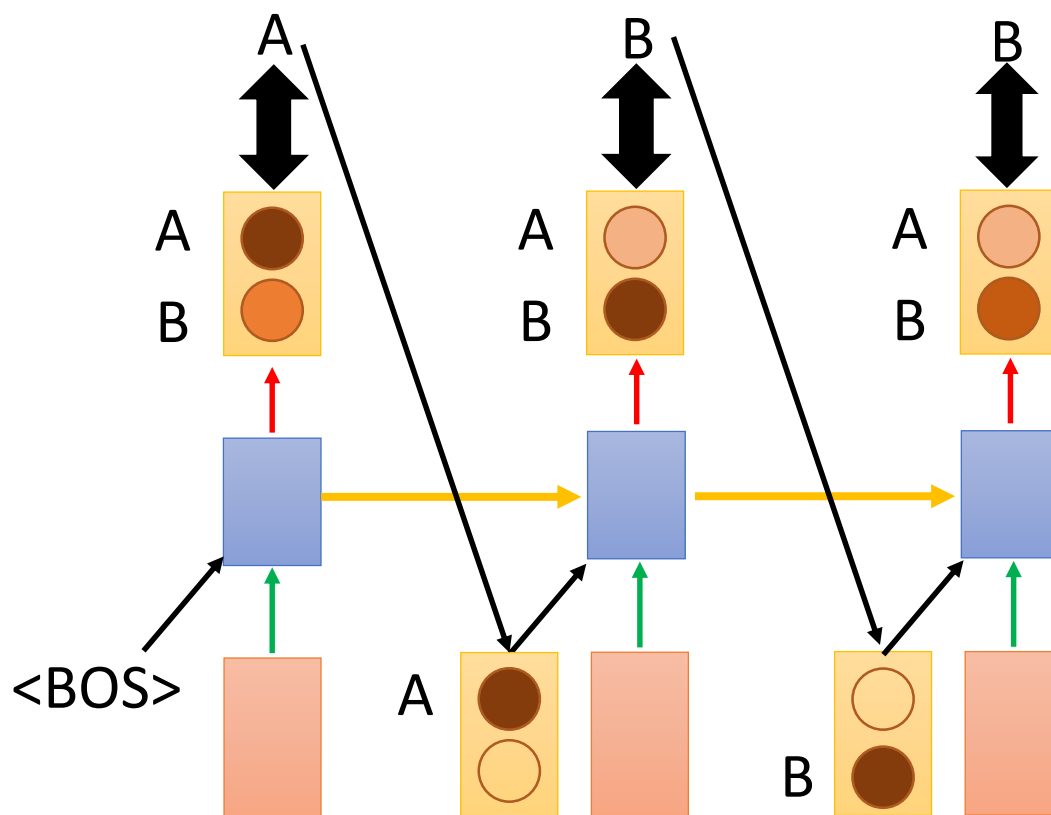
- Training

$$C = \sum_t C_t$$

Minimizing
cross-entropy of
each component

 : condition

Reference:



Mismatch between Train and Test

只好假設每個timestamp產生的結果都是正確的，然而這可能導致一步錯步步錯

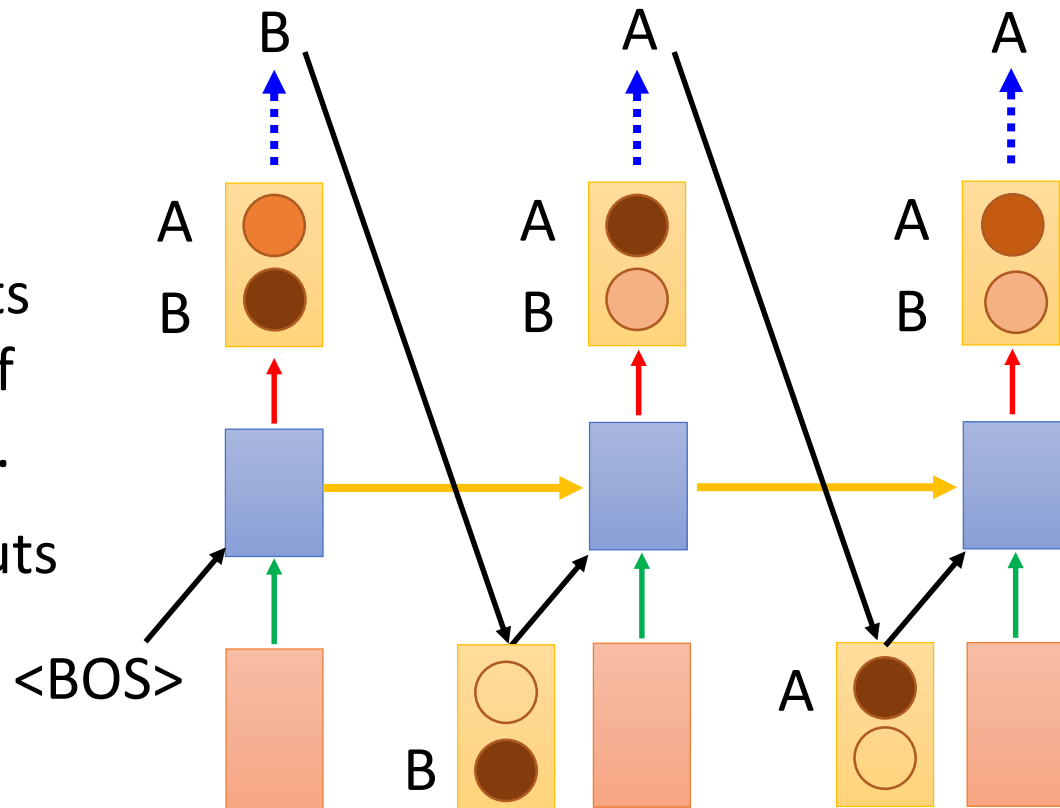
- Generation

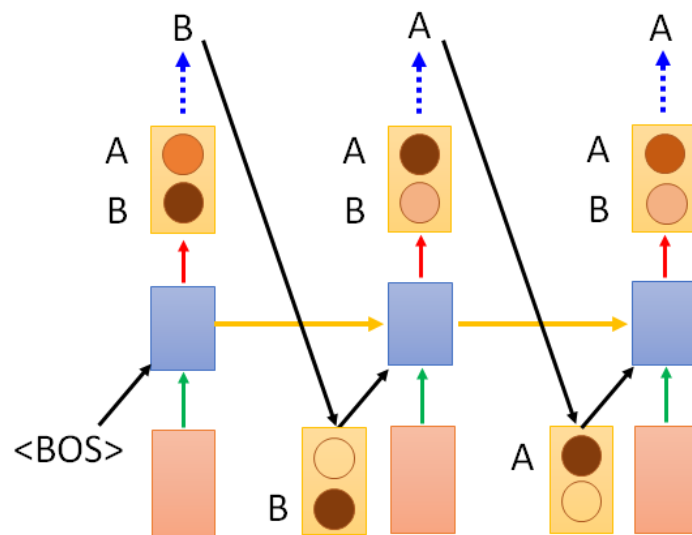
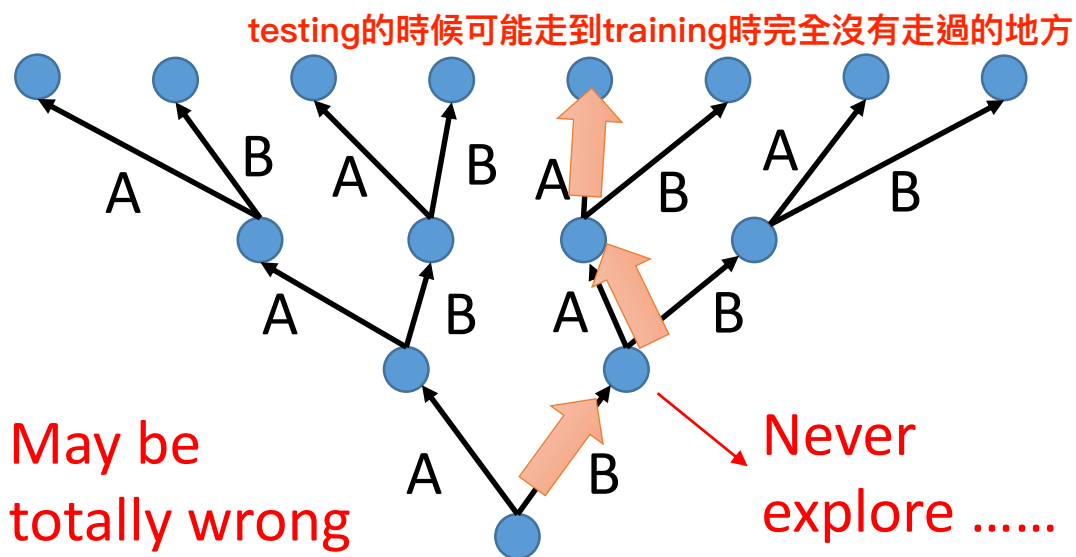
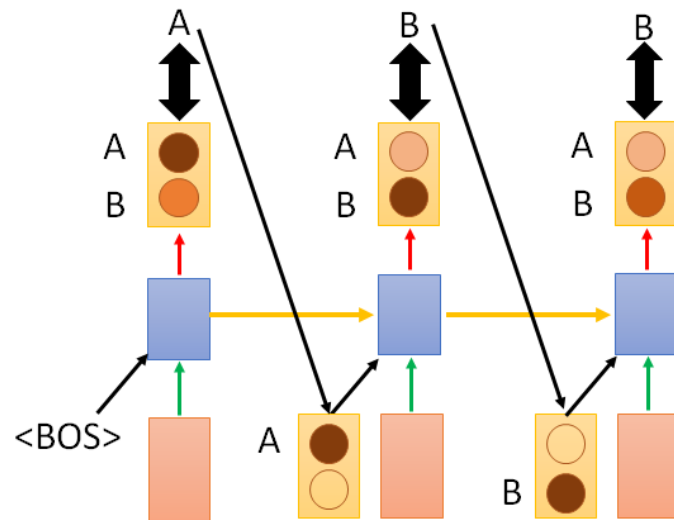
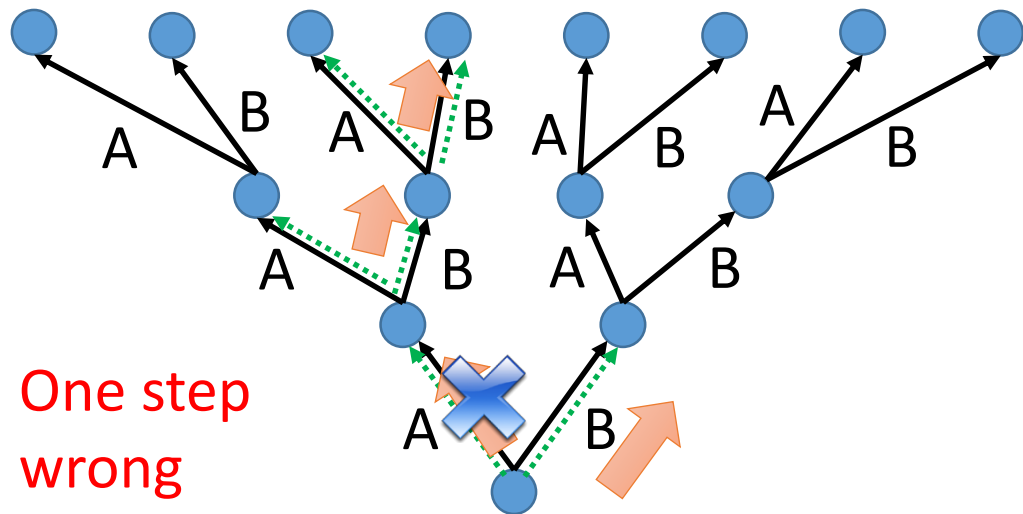
We do not know the reference

Testing: The inputs are the outputs of the last time step.

Training: The inputs are reference.

Exposure Bias





一步錯，步步錯

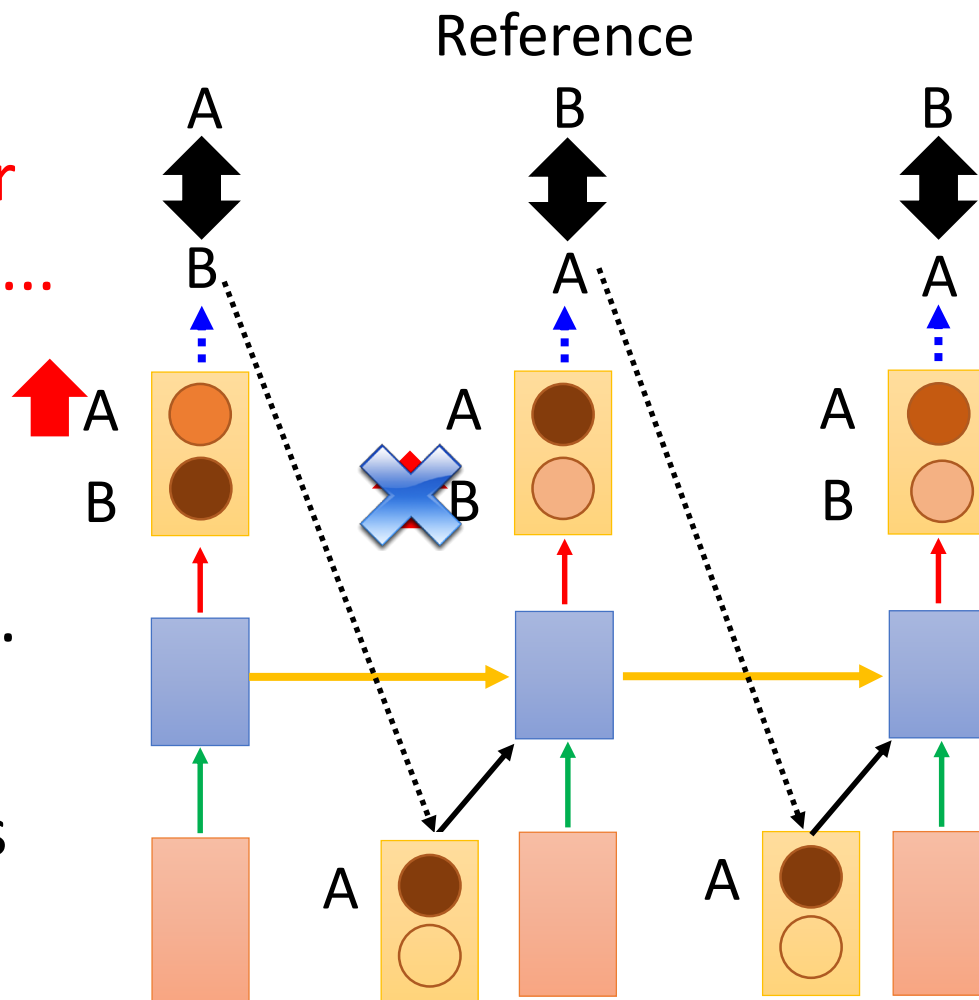
如果說把training process改成如testing process，順著上一部產生的結果繼續產生的話，training會變成不穩定
因為如果train到中後期，某個timestamp結果被翻轉了，則後面train出來的東西都白學了

Modifying Training Process?

When we try to
decrease the loss for
both steps 1 and 2

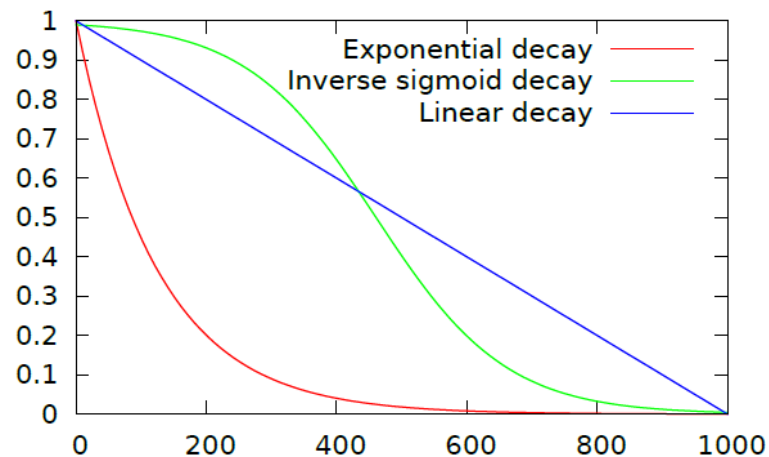
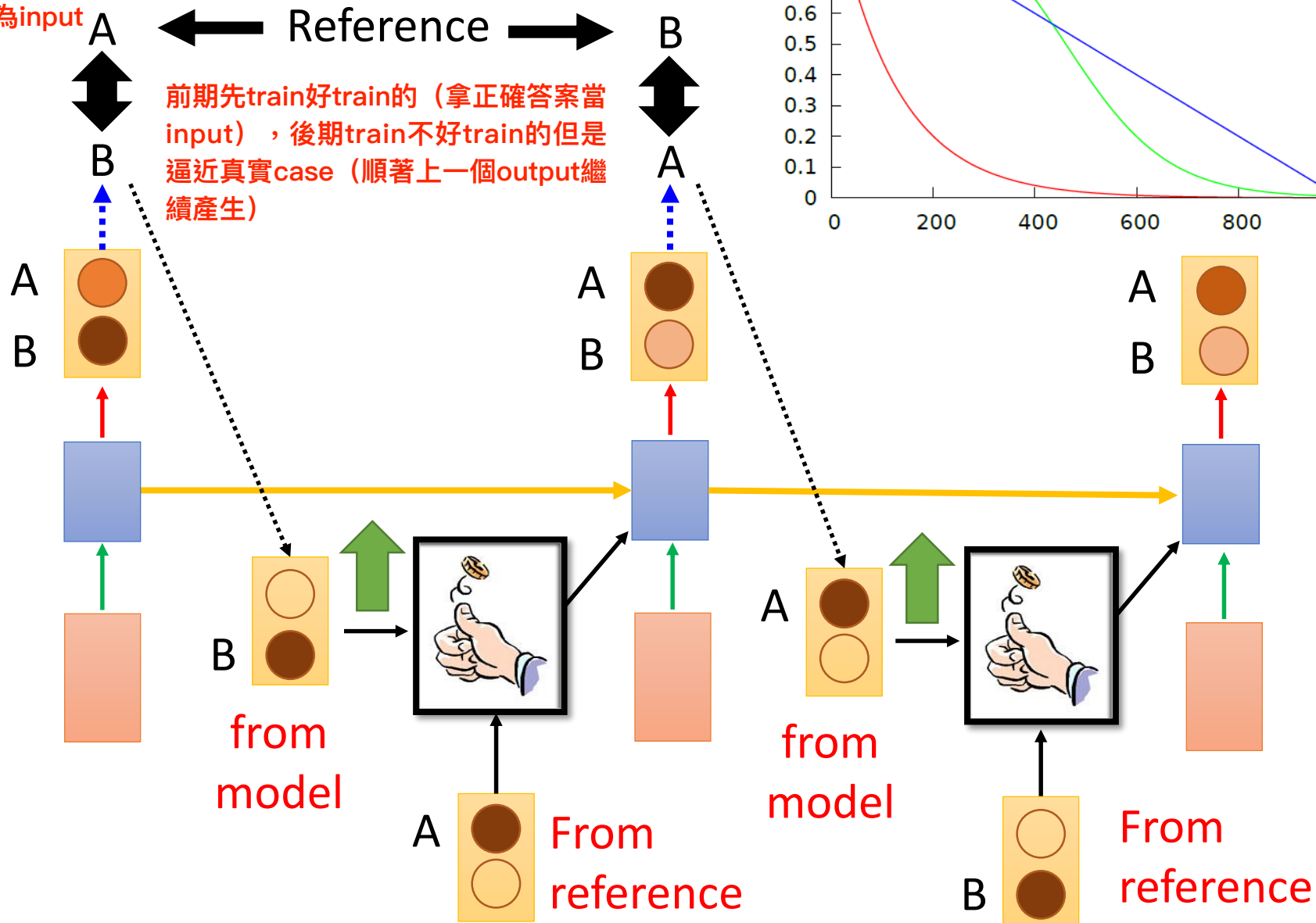
Training is
matched to testing.

In practice, it is
hard to train in this
way.



Scheduled Sampling 採取折衷措施

利用骰子決定到底要採取正確答案或是上個timestamp產生的結果作為input



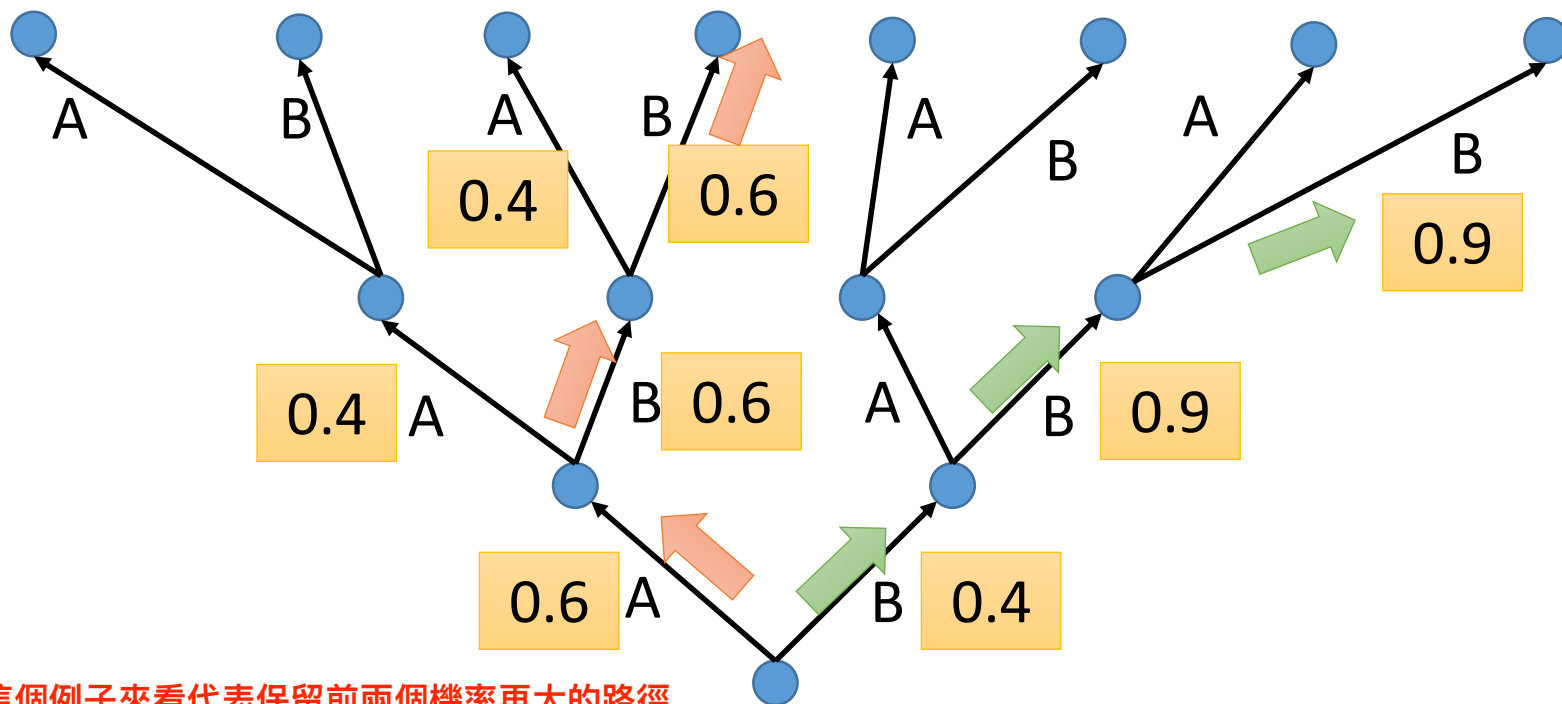
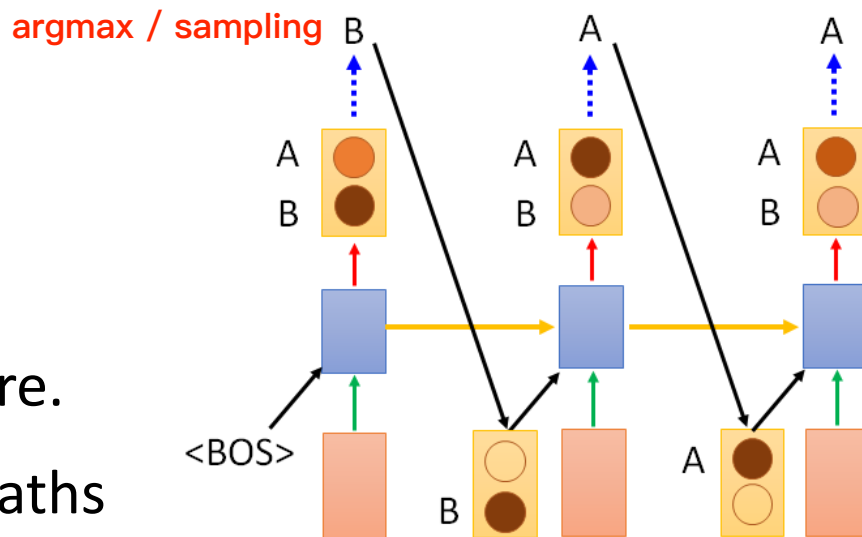
Beam Search

The green path has higher score.

Not possible to check all the paths

雖然一開始的機率小，但最後整條path機率大

然而無法窮舉所有可能，因此每次做beam search時都保留前N個機率最高的道路

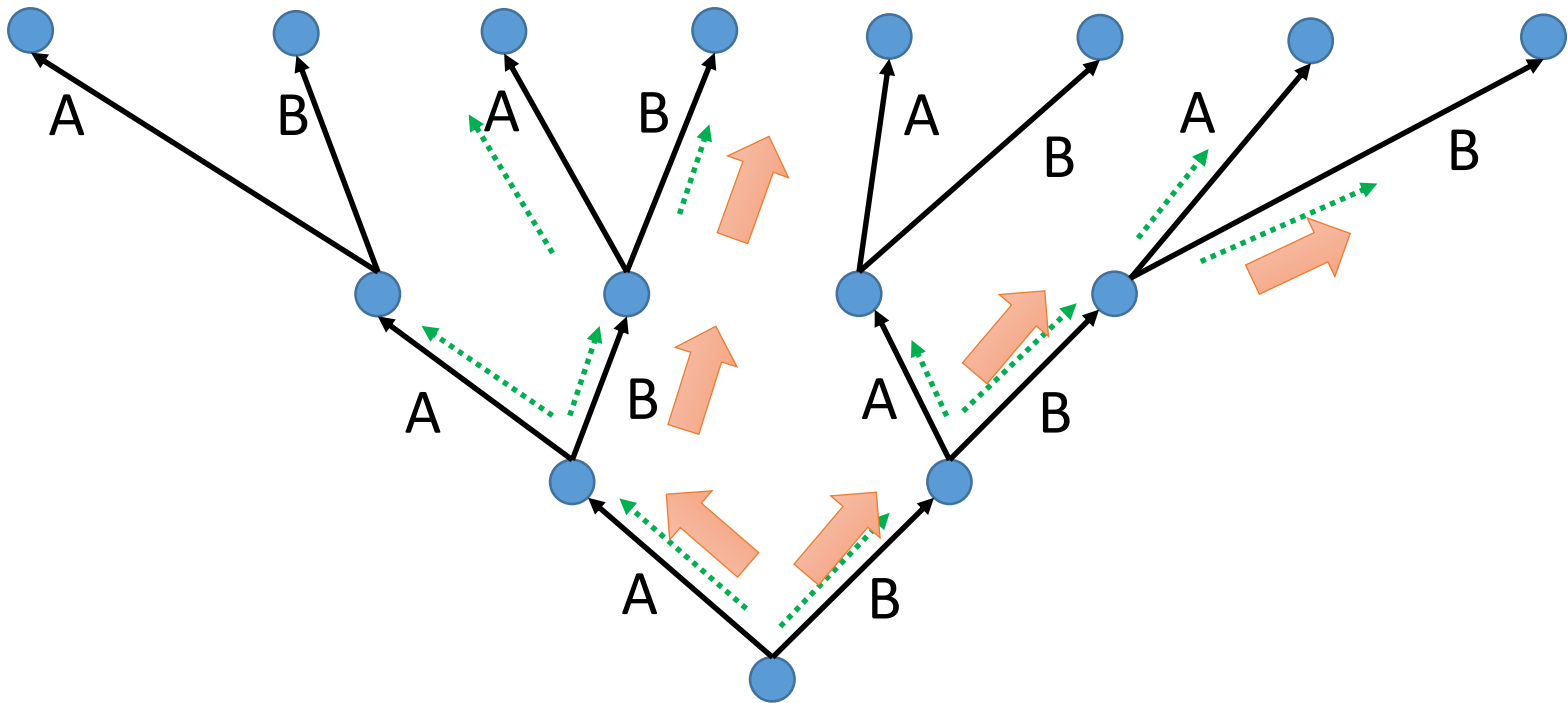
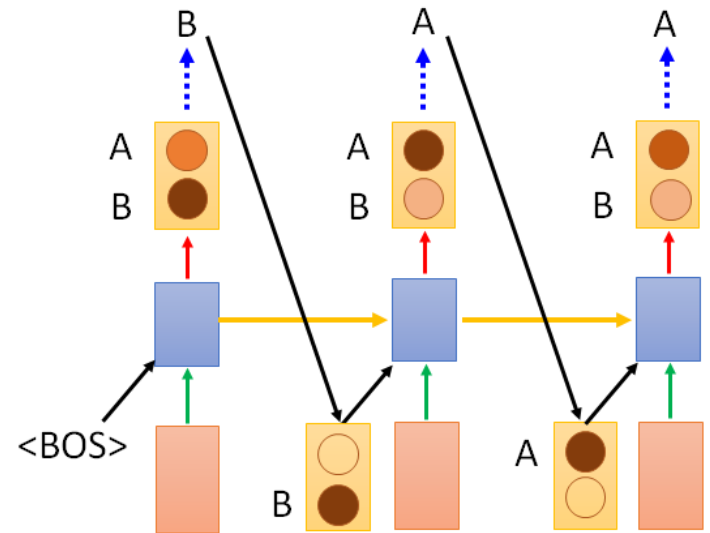


以這個例子來看代表保留前兩個機率再大的路徑

Beam Search

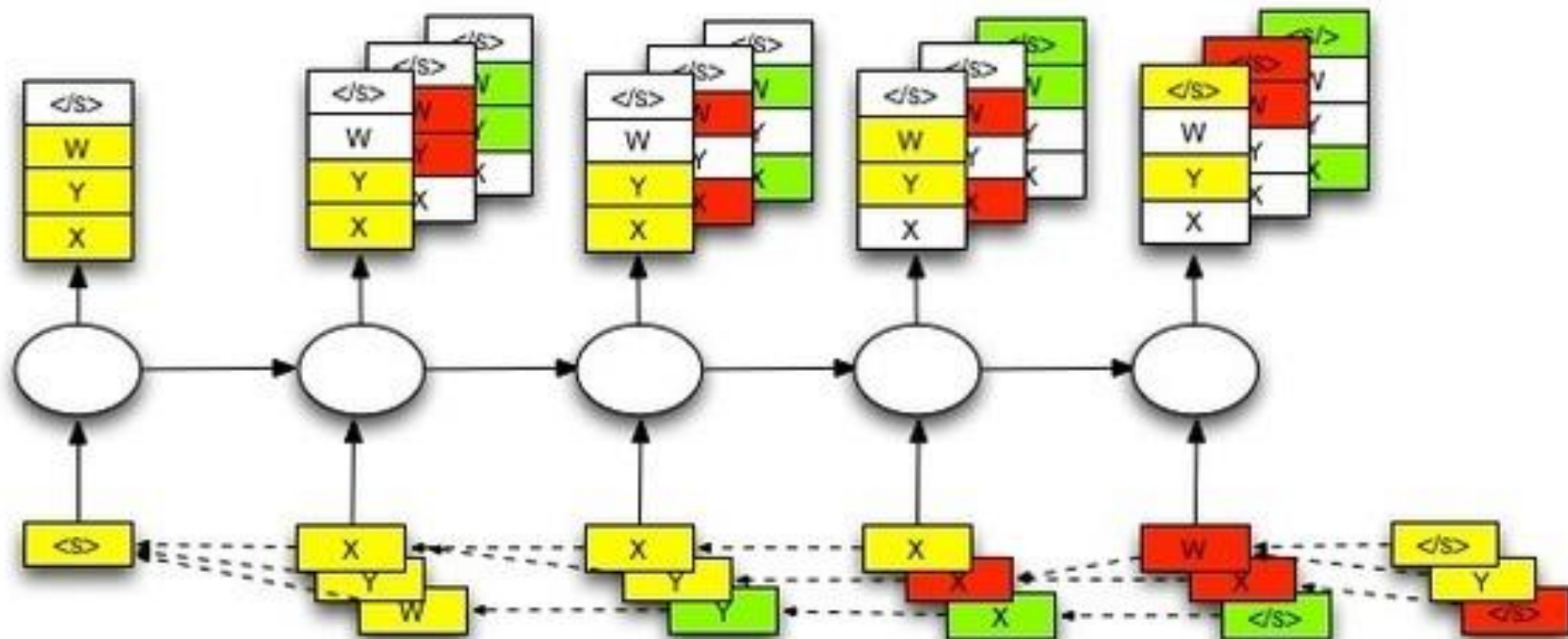
Keep several best path at each step

Beam size = 2



只有在testing才會用 training時無關

Beam Search



分開丟進去

The size of beam is 3 in this example.

如果說不要分開丟進去，直接平均所有input呢

Better Idea?

I am



You are



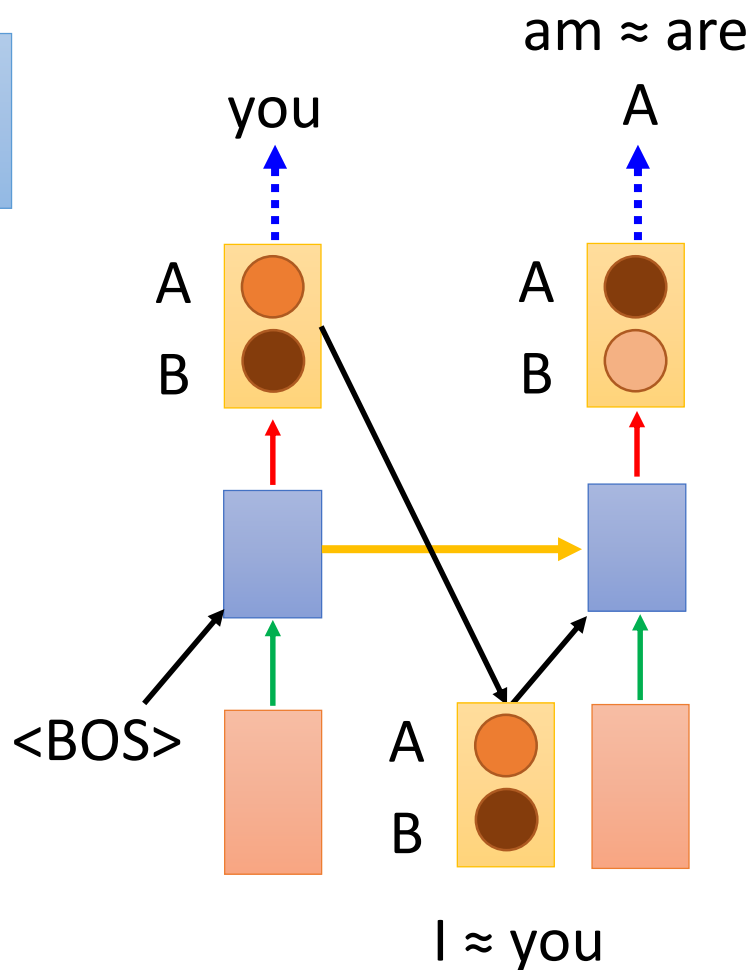
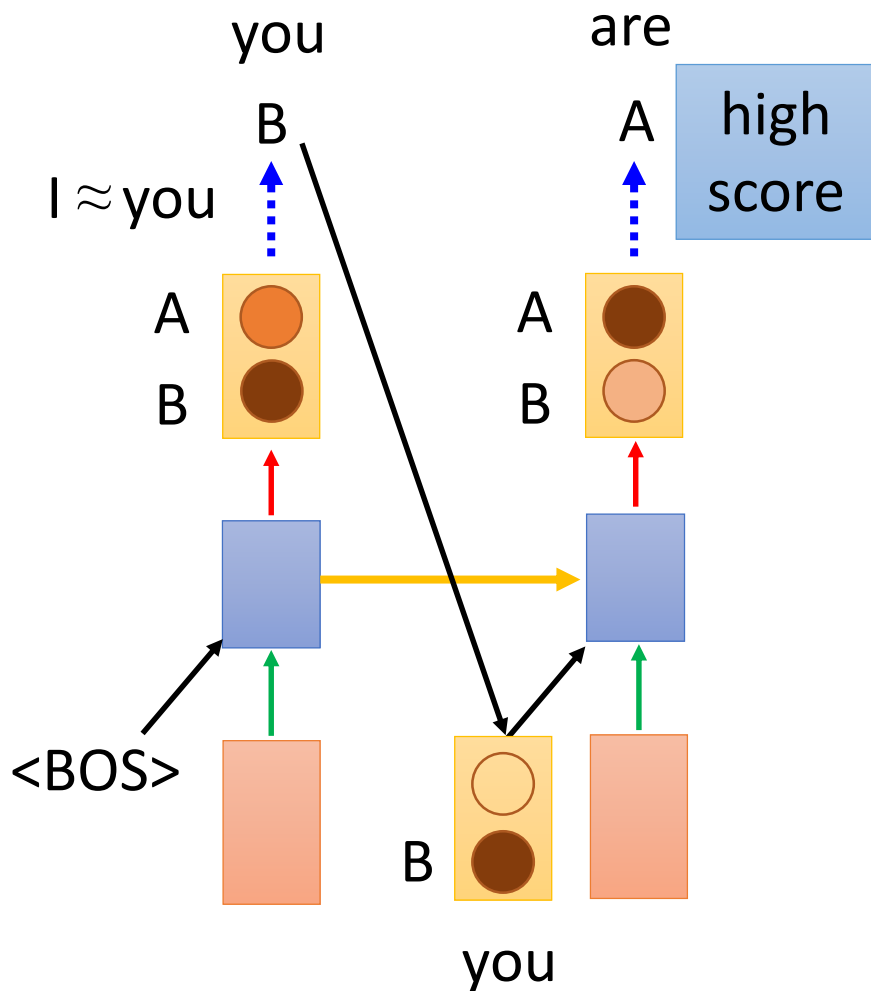
I are



You am



如果I/You機率都差不多有可能sample厝，造成mismatch



Object level v.s. Component level

Evaluation?

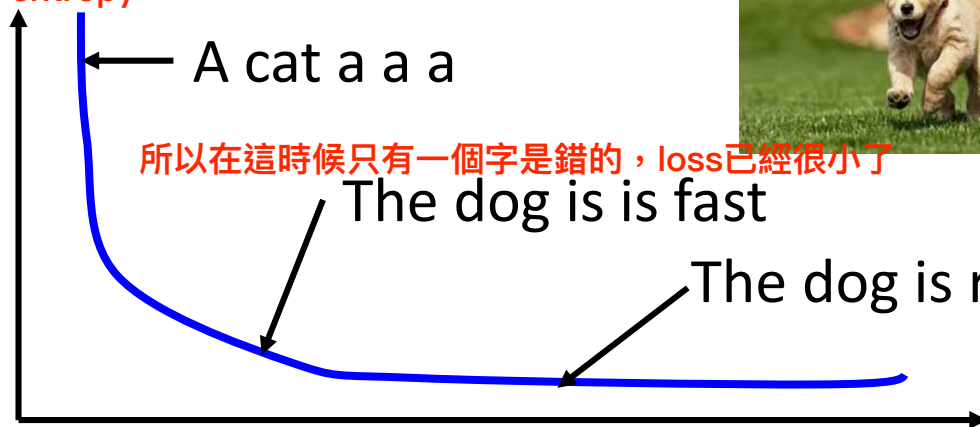
- Minimizing the error defined on component level is not equivalent to improving the generated objects

Ref: The dog is running fast

因為loss是每個詞彙分開做cross entropy

$$C = \sum_t C_t$$

Cross-entropy
of each step



Optimize object-level criterion instead of component-level cross-entropy. object-level criterion: $R(y, \hat{y})$

如果要設計好的loss function，通常都無法微分啊！

Gradient Descent?

y : generated utterance, \hat{y} : ground truth

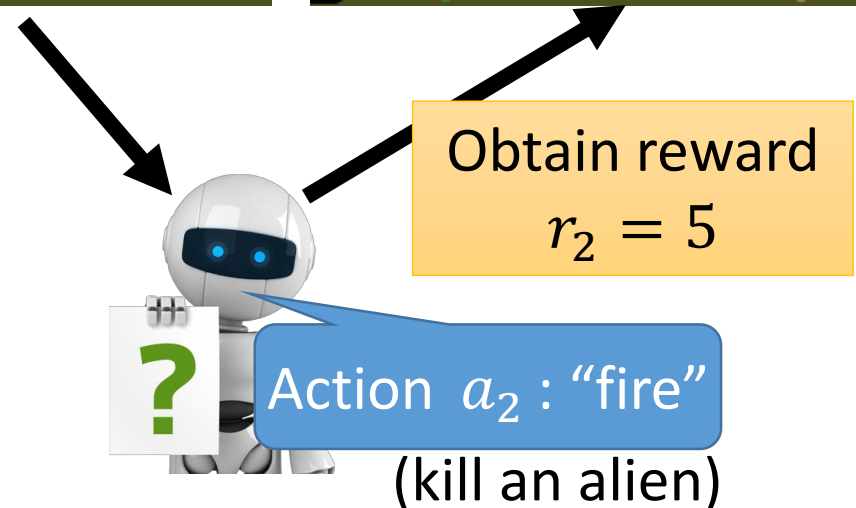
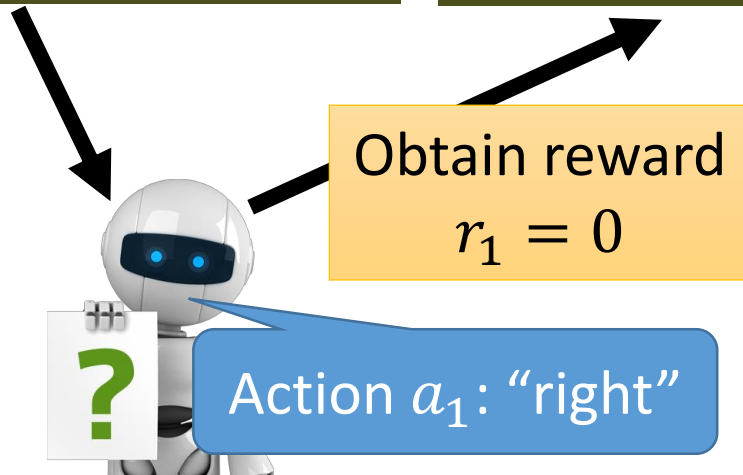
這年頭碰到不能微分的objective function，通通把它當成reinforcement learning的reward來train

Reinforcement learning?

Start with
observation s_1

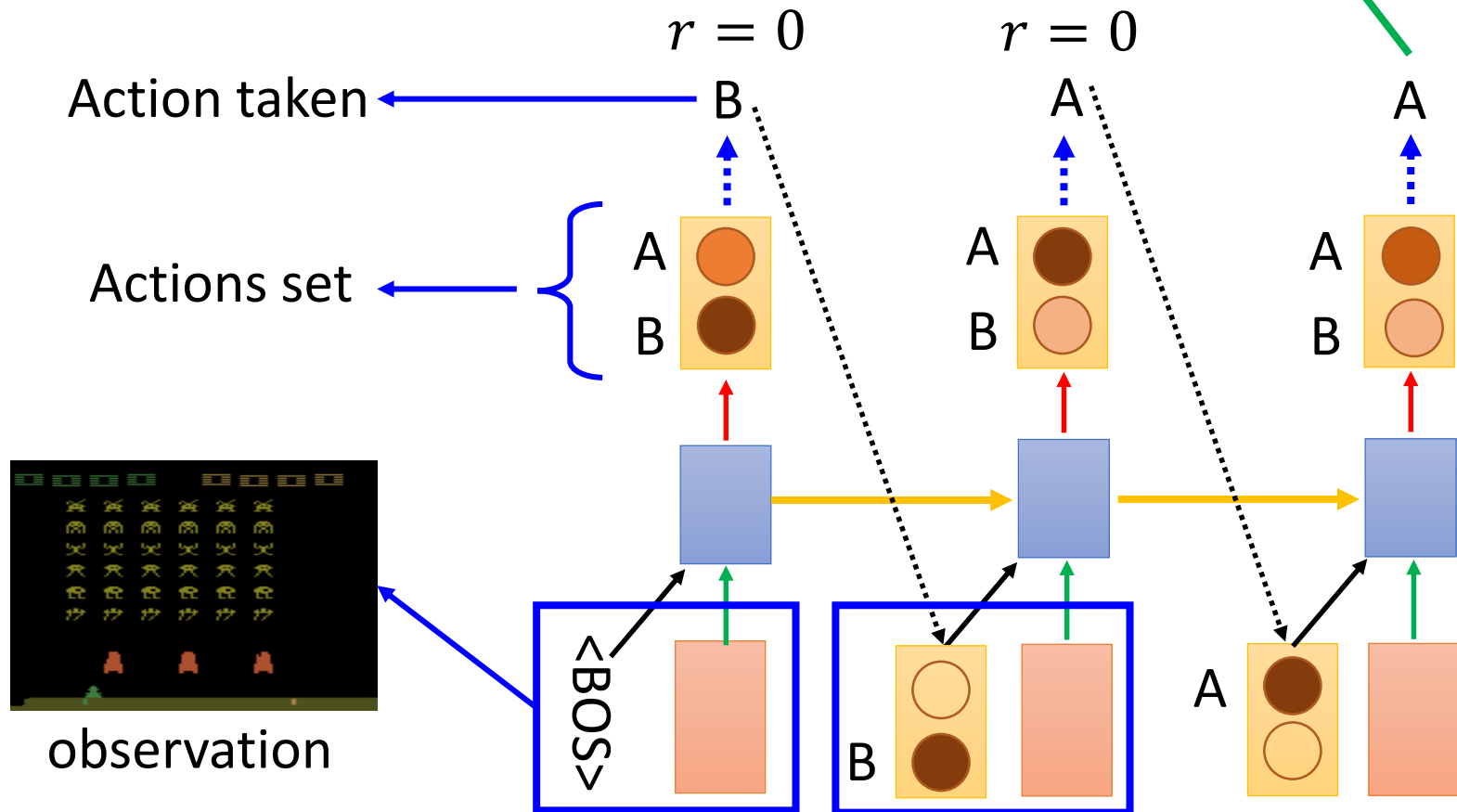
Observation s_2

Observation s_3



Reinforcement learning?

reward:
 $R(\text{"BAA", reference})$



Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, Wojciech Zaremba, "Sequence Level Training with Recurrent Neural Networks", ICLR, 2016

The action we take influence the observation in the next step