遞迴神經網路與序列模型

蔡炎龍&教研處

「版權聲明頁」

本投影片已經獲得作者授權台灣人工智慧學校得以使用於教學用途，如需取得重製權以及公開傳輸權需要透過台灣人工智慧學校取得著作人同意；如果需要修改本投影片著作，則需要取得改作權；另外，如果有需要以光碟或紙本等實體的方式傳播，則需要取得人工智慧學校散佈權。

– 台灣人工智慧學校

課程內容

講師投影片
投影片
影片播放列表
程式碼: ~/courses-tpe/RNN/part3

1.神經網絡的技巧

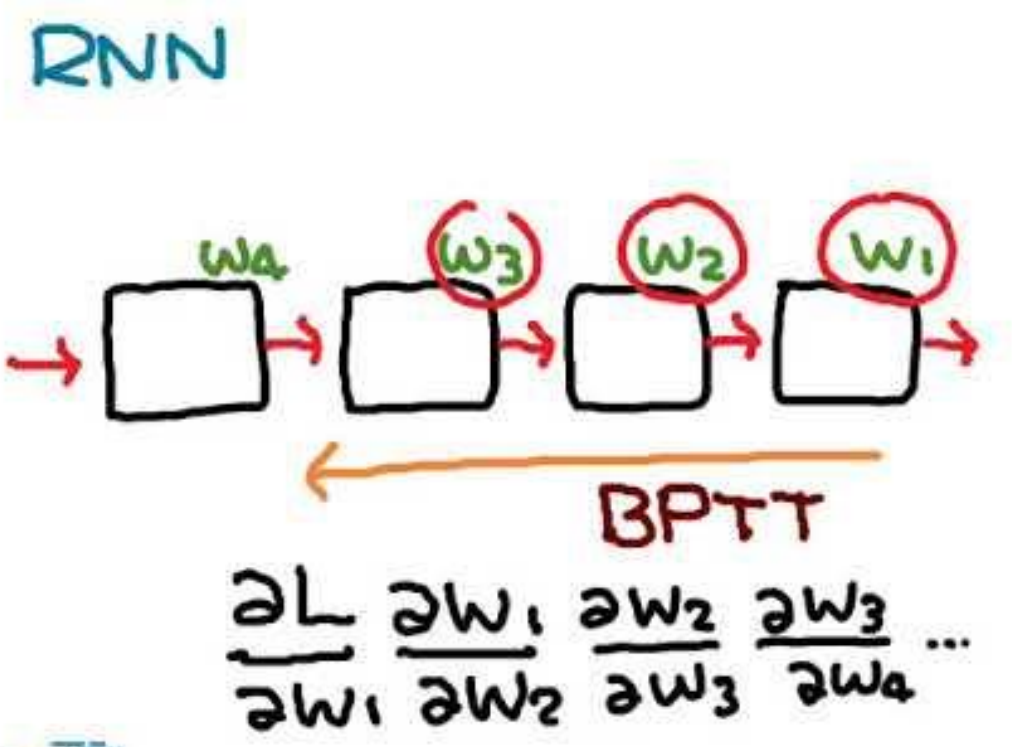
- LSTM為什麼好訓練
- Initialize
- Regularization
- Optimizer
- Batch Normalization

2.seq2seq 講解&實作

3.Attention model

4.問卷調查

為什麼 LSTM 好訓練?



台灣人工智慧學校

神經網路調校之前的提醒

初始化方法的選擇

重點

權重等參數選取原則

- 不要全部設成 0 (原因用腳想就知道)
- 取小一點的值

初始化的理論

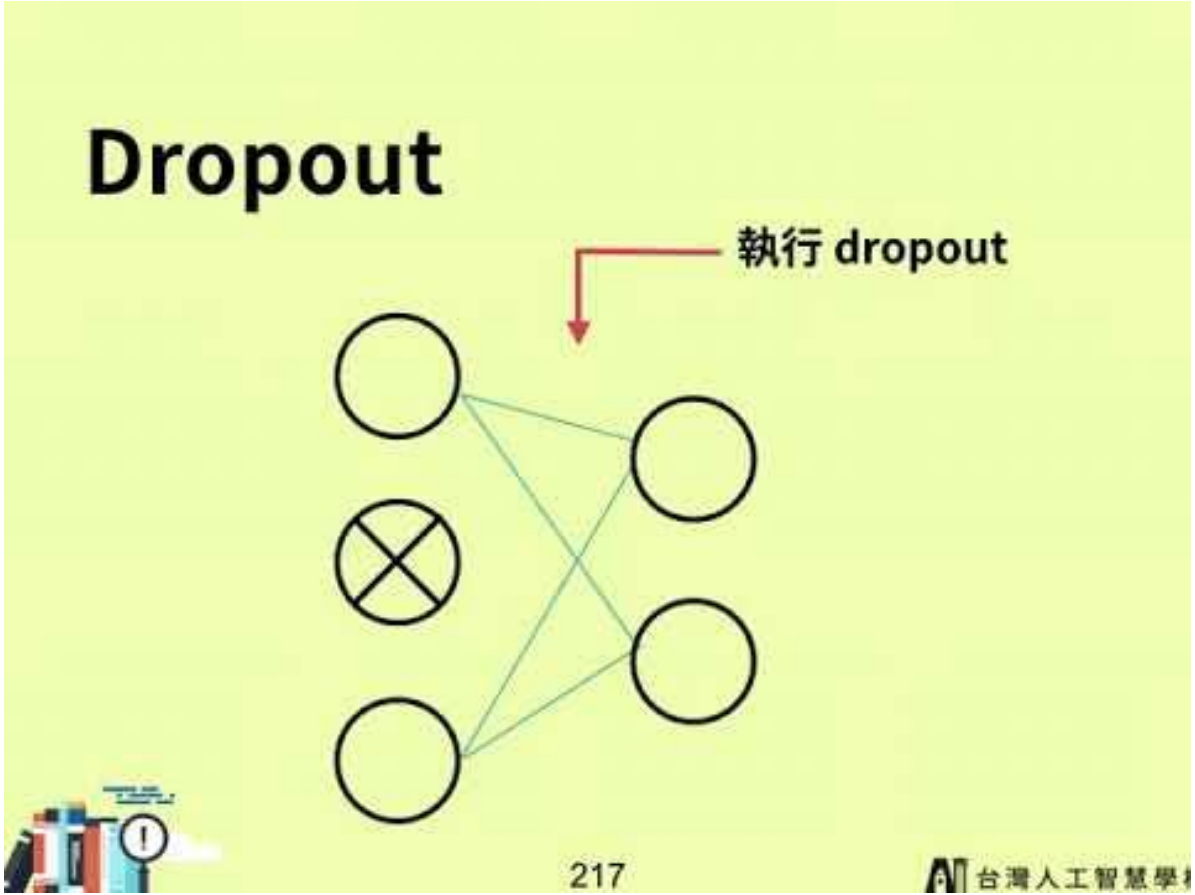
如果該獨立的都獨立, 平均值又都是 0, 我們有:

$$\mathrm{Var}(Y) = \sum_{i=1}^{n} \mathrm{Var}(W_i)\,\mathrm{Var}(X_i)$$

如果每個 W_i, X_i 變異數基本上一樣的話, 也就是說

$$\mathrm{Var}(W_i) = \mathrm{Var}(\overline{W})$$

Regularization之Dropout



不同參數初始化的方法

所以我們可以由平均值 0, 變異數為 $\frac{1}{n}$ 的分布中取值

$$\sigma = \sqrt{\frac{1}{n}}$$

原始論文

Xavier Initialization

$$n = n_{\text{in}} + n_{\text{out}}$$

L2 Regularization

L2 Regularization

$$\text{loss function} + \frac{\lambda}{2n}\|w\|_2^2$$

學習法改良的兩個方向

兩個改善方向

- momentum: 方向的穩定和加速器
- 可變速的 learning rate

Momentum

概念

Momentum

問題是你順著那個方向到了另一點, 你可能會發現那個方向又變了! 於是你又向另一個方向...

但其實你好好走原來方向, 也許一路也就下山了!

Learning Rate變速器和Adam學習法

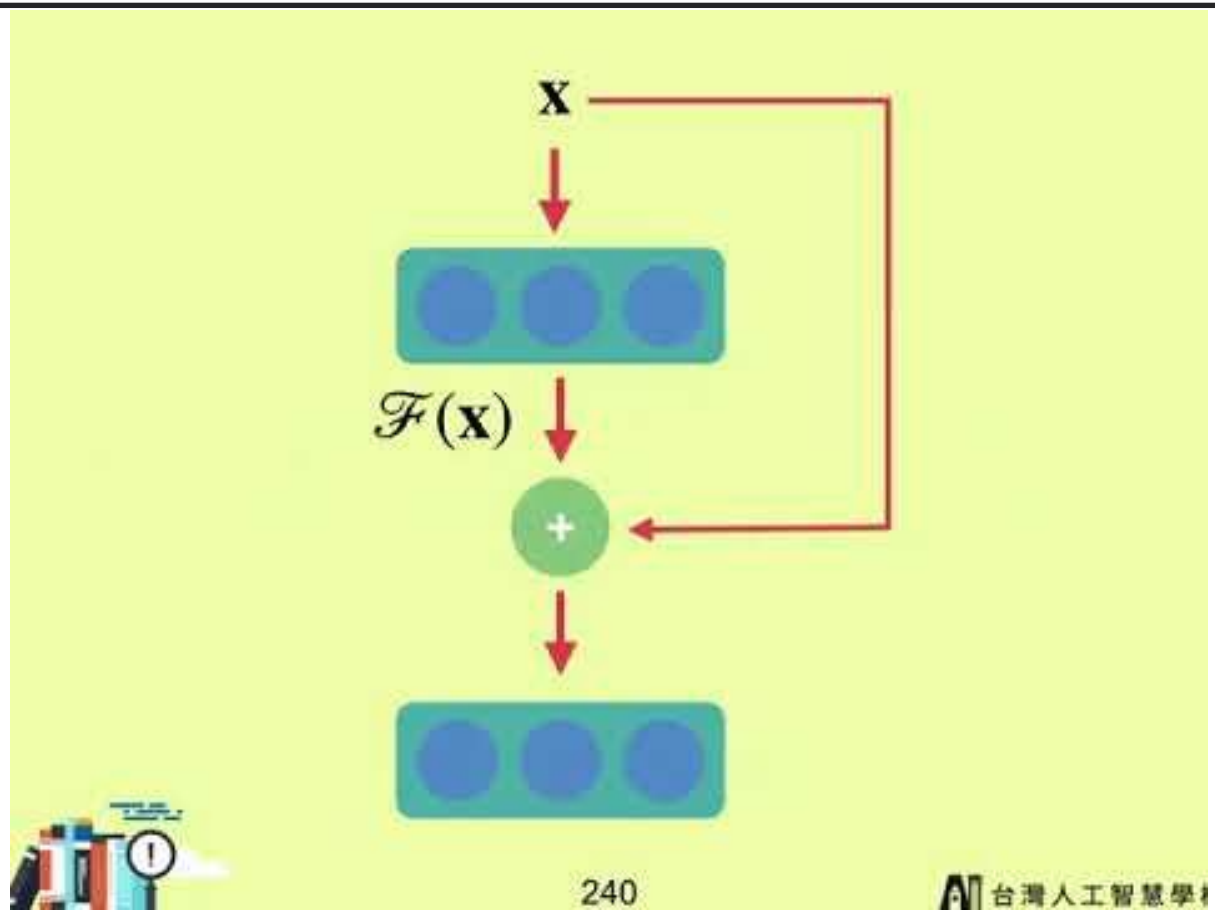
概念

Learning Rate 變速器

一個很自然的想法: learning rate 在接近目標時可能要小一點。於是有了各種「減速法」。

BatchNormalization

ResNet



RNN可用ResNet嗎

SELU

$$\text{selu}(x) = \lambda \cdot \begin{cases} x & \text{if } x > 0 \\ \alpha e^x - \alpha & \text{if } x \leq 0 \end{cases}$$

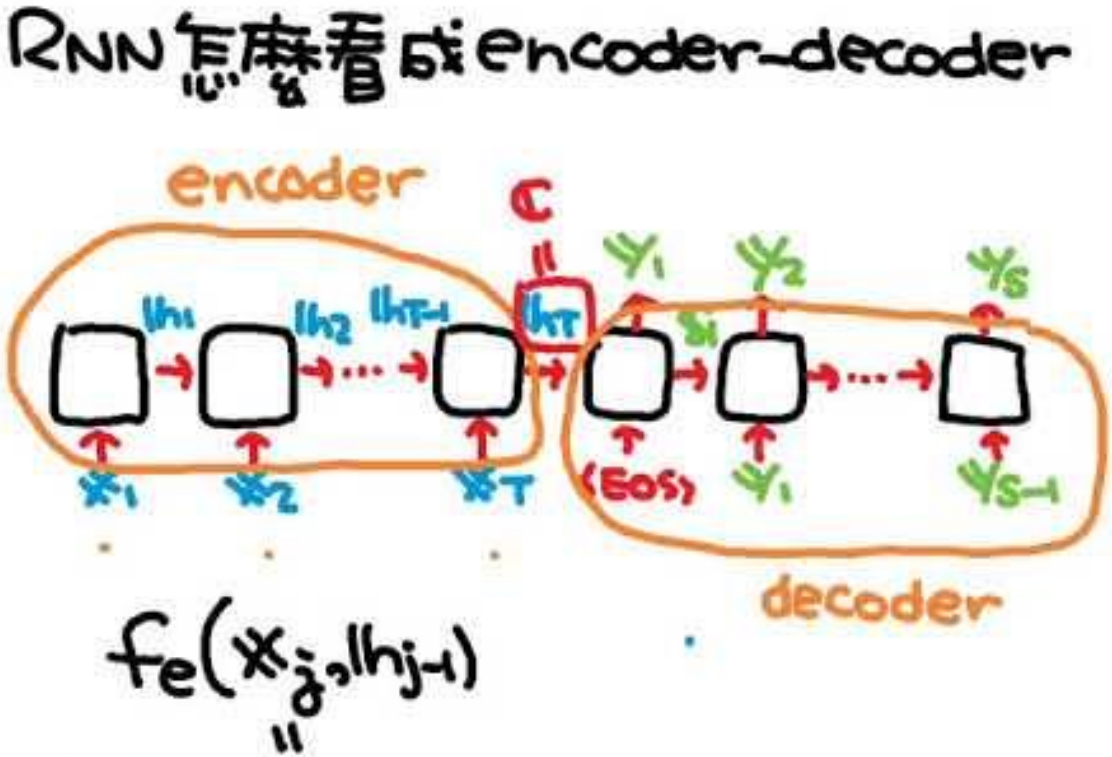
最炫的是參數其實是常數:

α = 1.67326324235437728481704299167 17

λ = 1.05070098735548049341933498529 46

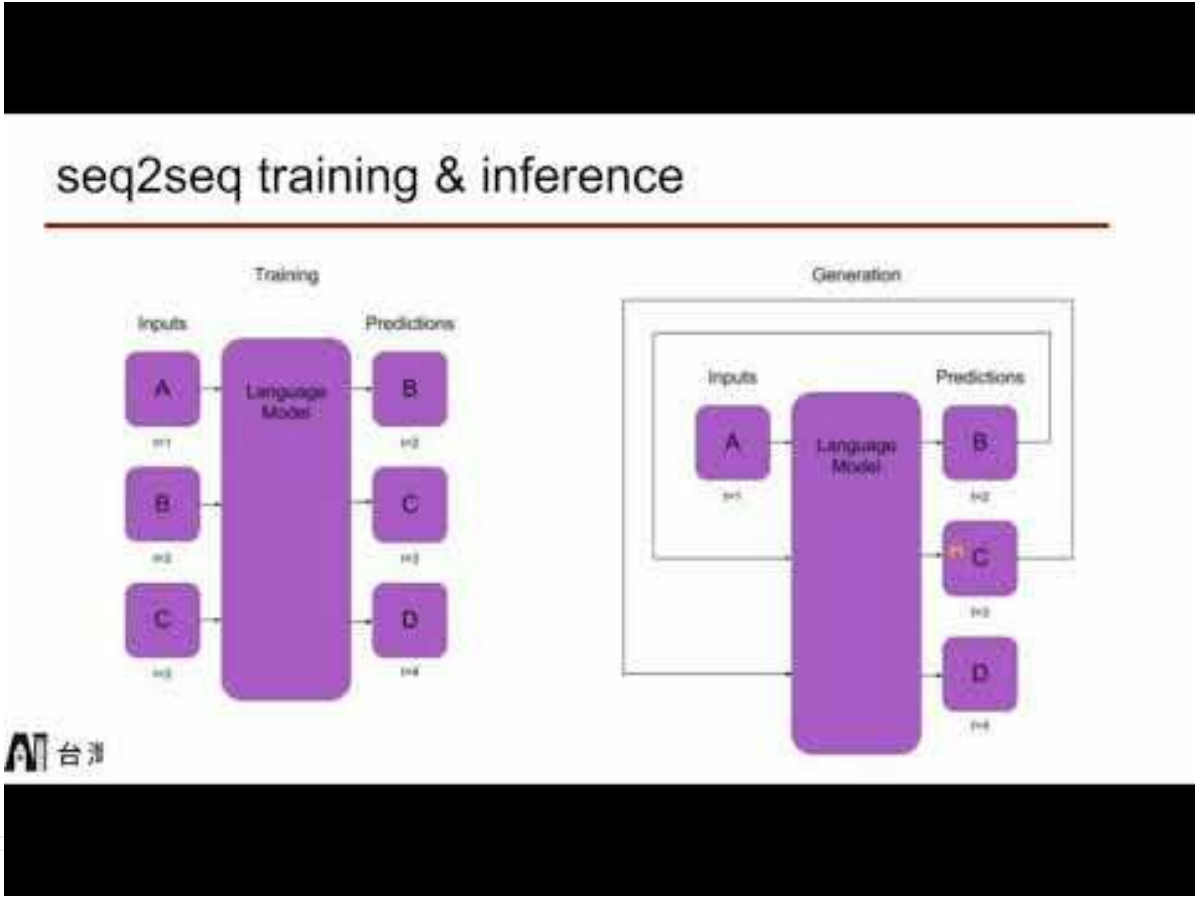
seq2seq 簡介

Encoder - Decoder



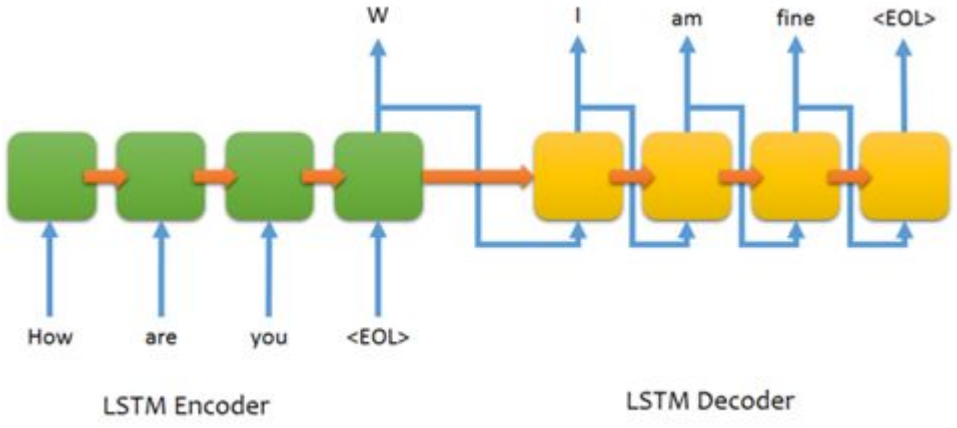
台灣人工智慧學校

RNN - seq2seq



sequence to sequence 架構

- Structure learning
- 通常爲兩個 RNN 組合
- Encoder: 對資訊進行編碼，產生編碼向量
- Decoder: 對編碼向量做解碼的動作

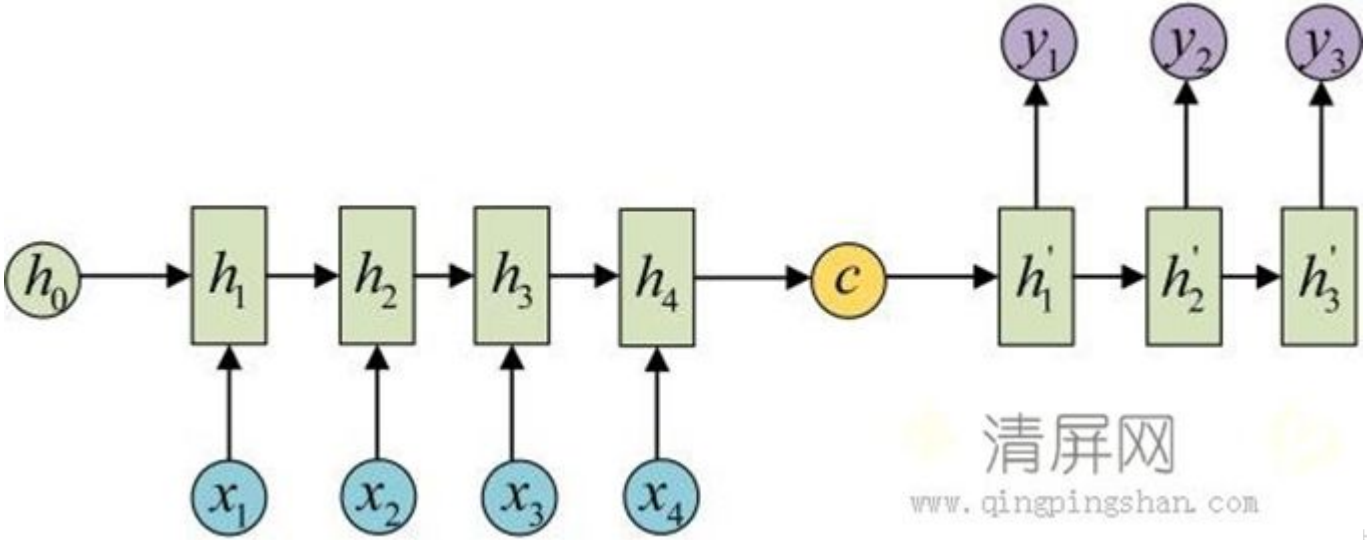


Application of seq2seq

- Machine translation
- Text summarization
- Chatbot (Conversation)
- Reading Comprehension
- Speech to natural language
- Video Caption

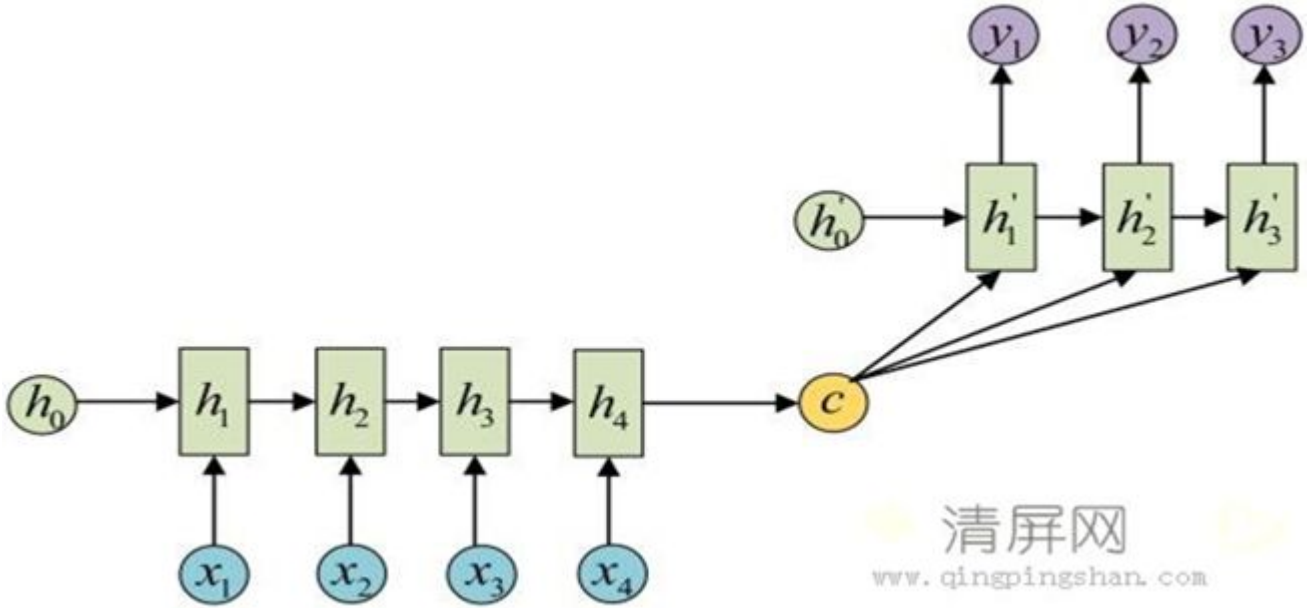
一般的 seq2seq 架構

- encoder 將輸入 x 做編碼，取出最後一個時間點，得到向量 c (state)
- 向量 c 當作 decoder 的 initial state 做編碼



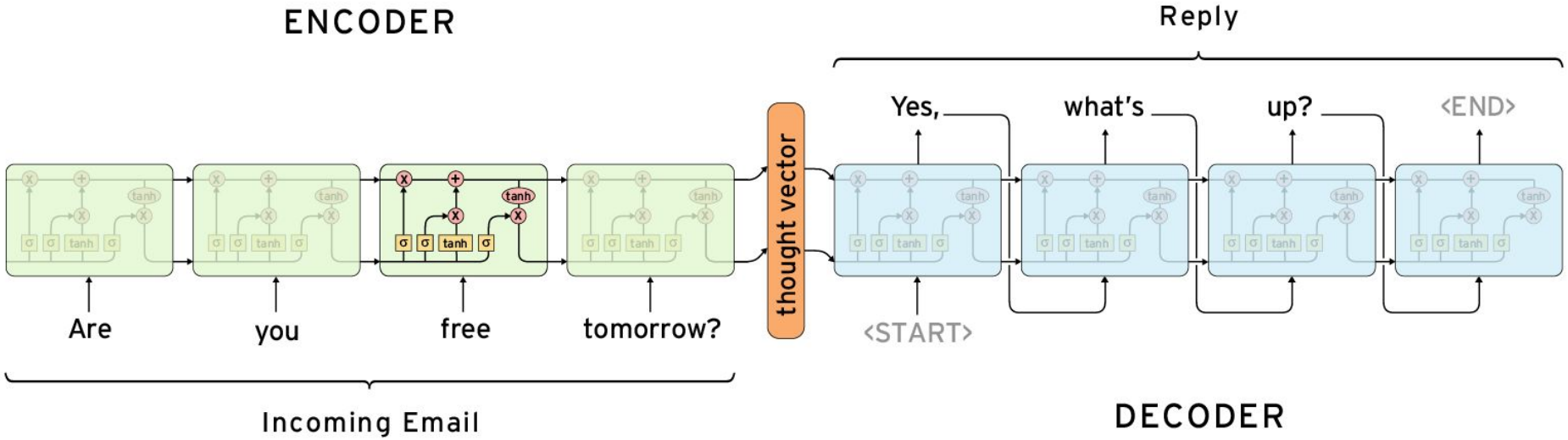
一般 seq2seq 其他架構

- 向量 c 當作 decoder 每個時間點的 input
- decoder 的 initial state 則爲 0 / random initialize

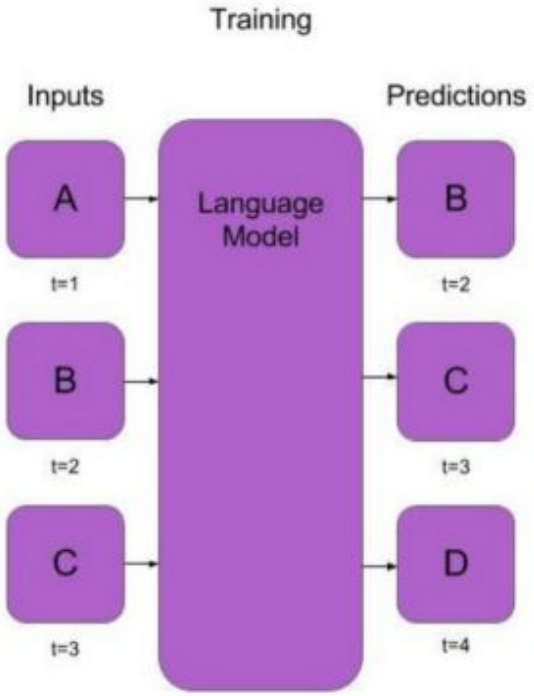


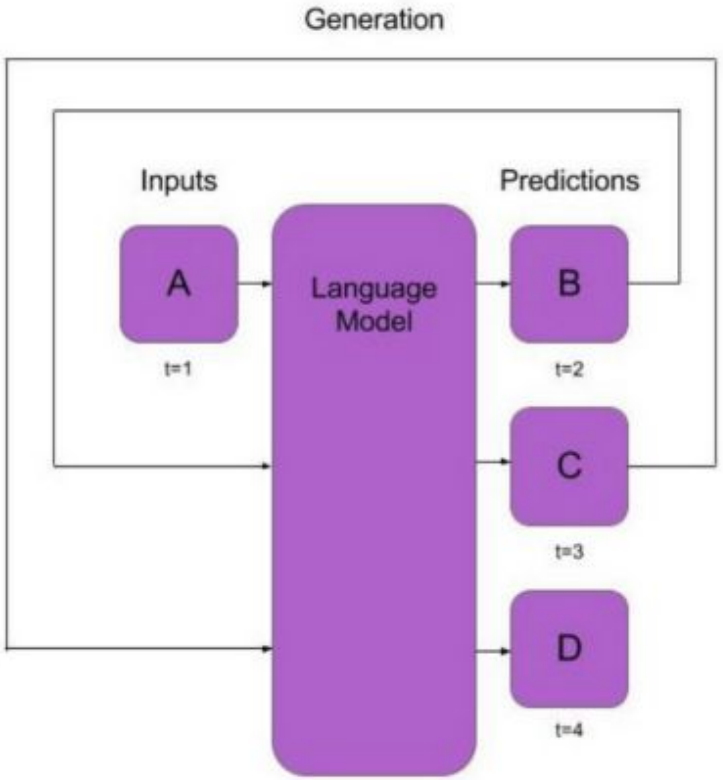
例如:

- encoder 將Incoming Email依序輸入做編碼 , 取出最後一個時間點的輸出向量得到thought vector。
- 以Thought vector當作 decoder 的 initial state, 依序做解碼得到Reply。

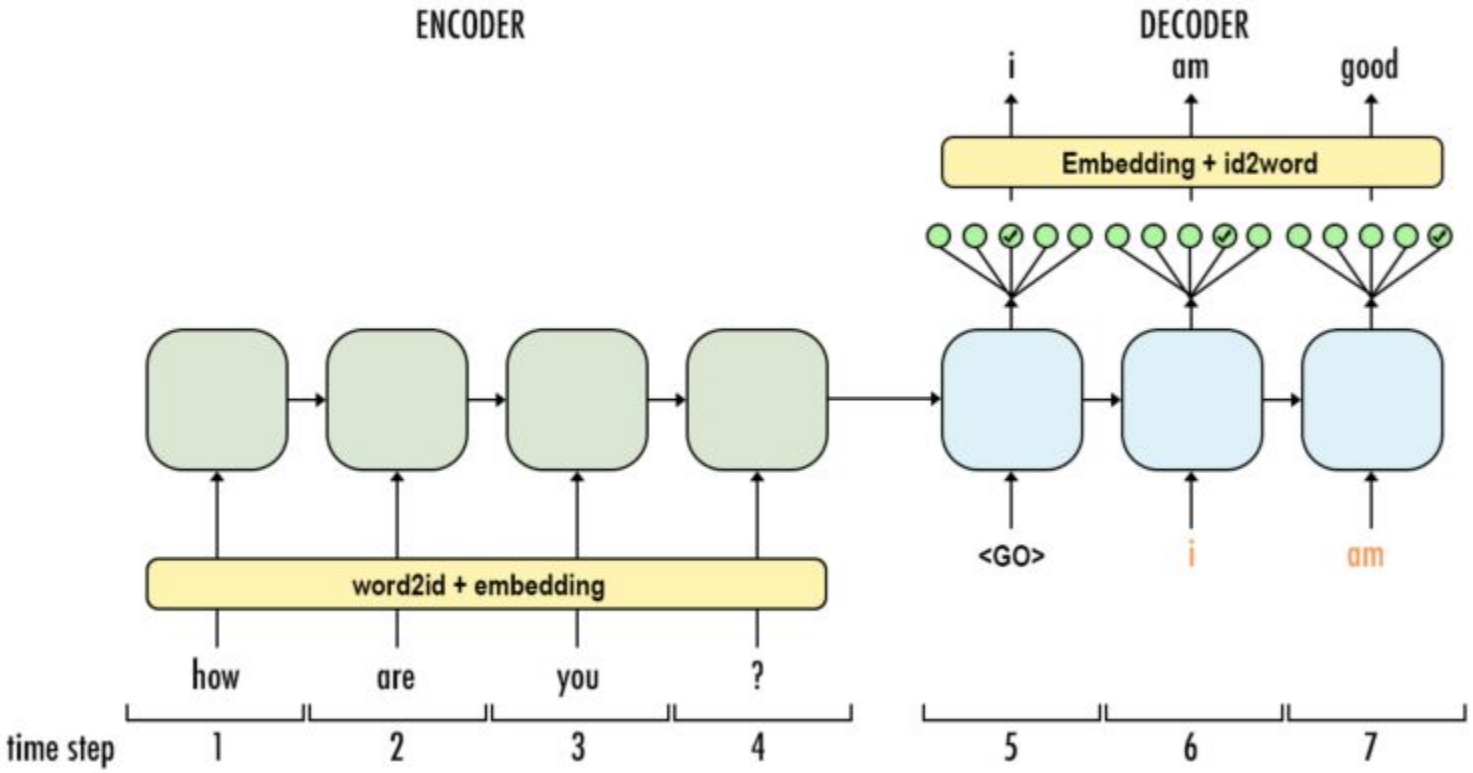


seq2seq training & inference

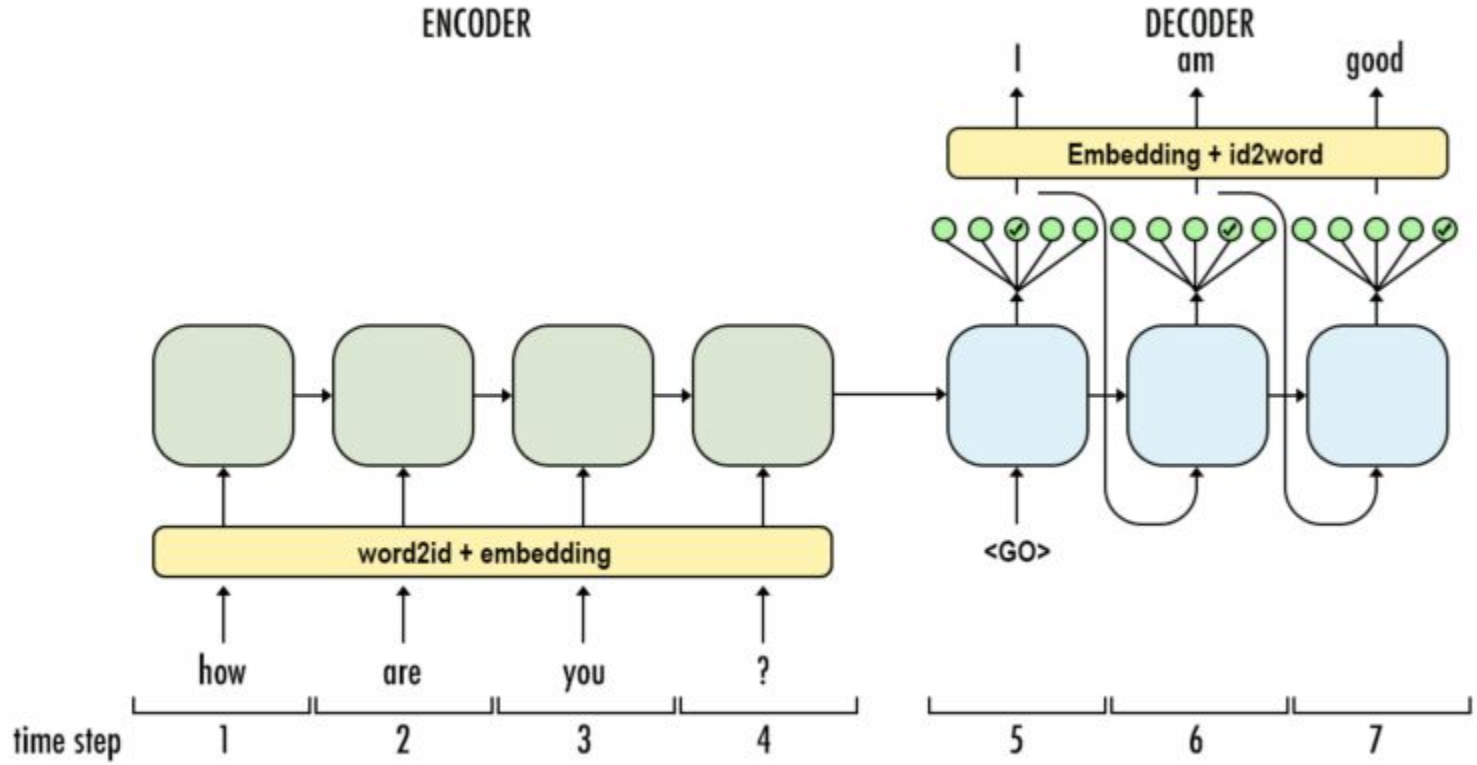




seq2seq training



seq2seq inference



Special character in NLP

- <BOS> : 開始字符，在 decoder 的第一個 timestep 輸入
- <EOS> : 結束字符，輸出 <EOS> 後就不再產生字
- <UNK> : 未知字符，word vector 沒有的字
- <PAD> : 補空字符，接在 <EOS> 後 (tensor shape 固定下)

seq2seq in tensorflow

- tf.contrib.seq2seq
- 重要的 API
  - TrainingHelper
    - use target data as next decoder input
  - GreedyEmbeddingHelper
    - use argmax of the output to next decoder input
  - BasicDecoder
    - 處理流程
  - dynamic_decode
    - 產生 output

Training decoding layer code example

```
def training_decoding_layer(dec_embed_input, summary_length,
                            dec_cell, initial_state, output_layer,
                            vocab_size, max_summary_length):

    training_helper = tf.contrib.seq2seq.TrainingHelper(
                           inputs=dec_embed_input,
                           sequence_length=summary_length,
                           time_major=False)

    training_decoder = tf.contrib.seq2seq.BasicDecoder(
                            dec_cell,
                            training_helper,
                            initial_state,
                            output_layer)

    training_logits, _ = tf.contrib.seq2seq.dynamic_decode(
                               training_decoder,
                               output_time_major=False,
                               impute_finished=True,
                               maximum_iterations=max_summary_length)
    return training_logits
```

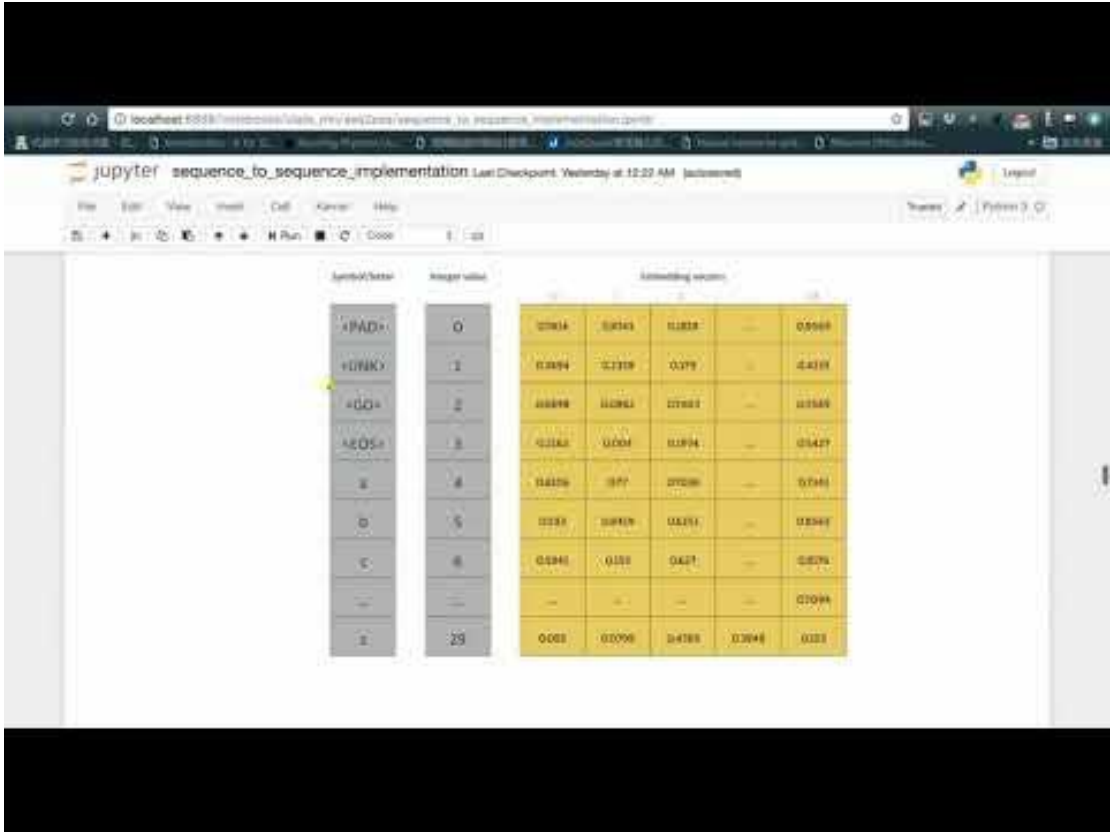
Inference decoding layer code example

```
def inference_decoding_layer(embeddings, start_token, end_token,
                             dec_cell, initial_state, output_layer,
                             max_summary_length, batch_size):


    start_tokens = tf.tile(tf.constant([start_token],
                                       dtype=tf.int32),
                                       [batch_size],
                                       name='start_tokens')

    inference_helper = tf.contrib.seq2seq.GreedyEmbeddingHelper(
                            embeddings,
                            start_tokens,
                            end_token)

    inference_decoder = tf.contrib.seq2seq.BasicDecoder(
                             dec_cell,
                             inference_helper,
                             initial_state,
                             output_layer)

    inference_logits, _ = tf.contrib.seq2seq.dynamic_decode(
                               inference_decoder,
                               output_time_major=False,
                               impute_finished=True,
                               maximum_iterations=max_summary_length)

    return inference_logits
```

Udacity 程式練習範例

- seq2seq/sequence_to_sequence_implementation.ipynb
  - 實際跑一遍及理解程式

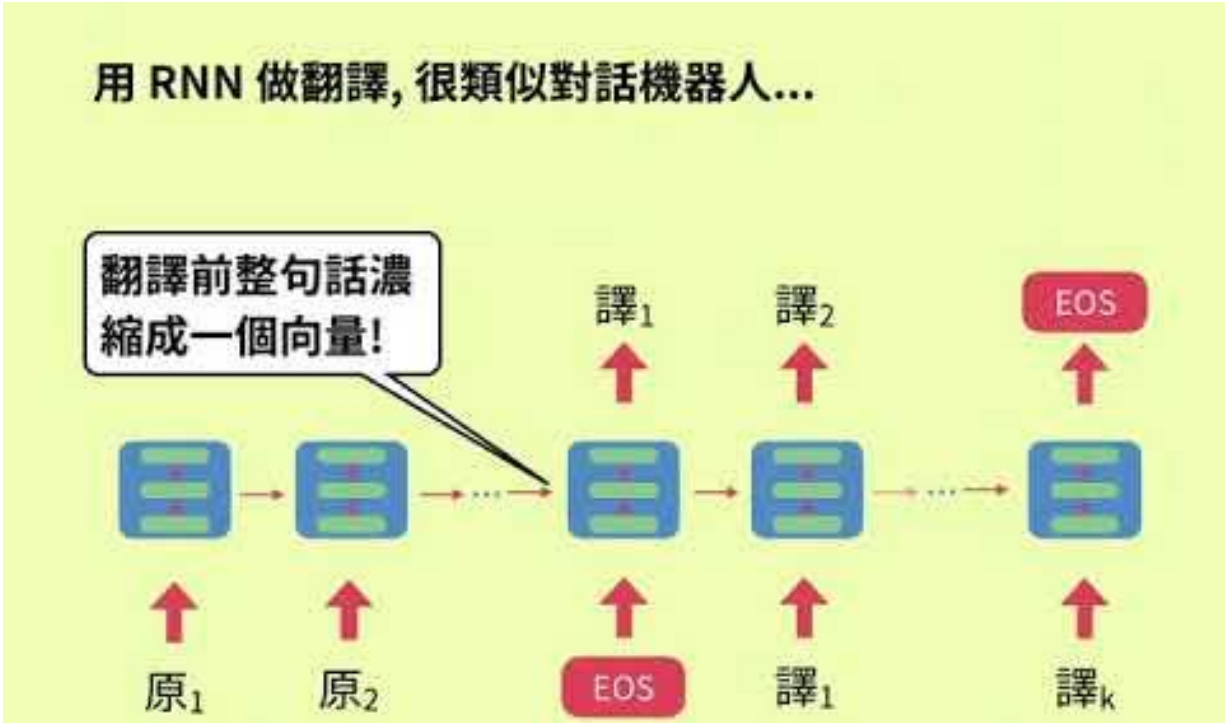
程式範例解說 (sequence_to_sequence_implementation.ipynb)



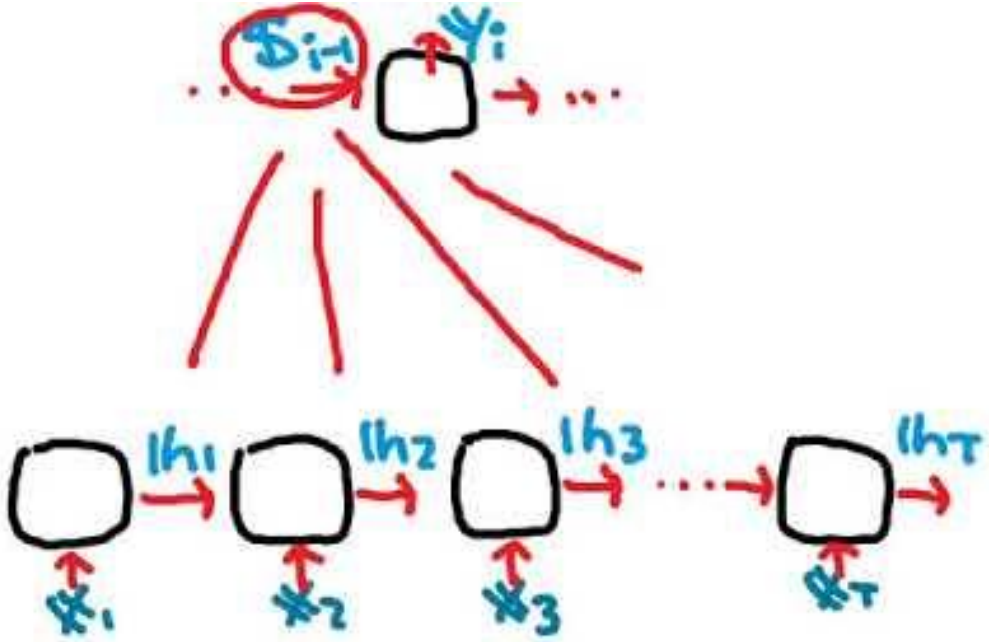
注意:程式碼需要複製到自己的路徑底下才能順利Run

Attention Model

Attention的概念

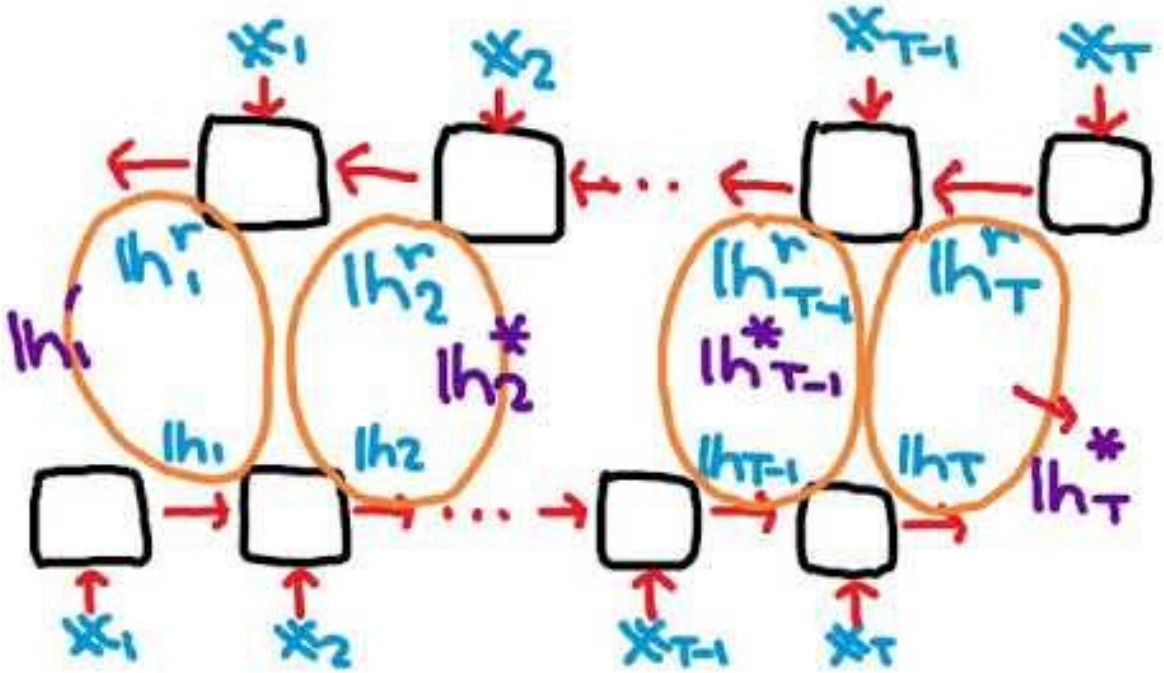


Attention的概念

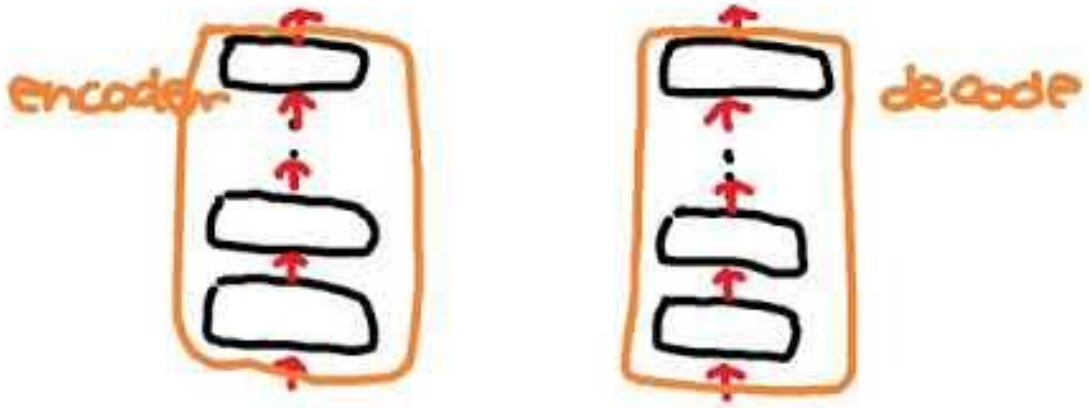


台灣人工智慧學校

Bidirectional RNN



Attention is all you need - Part1

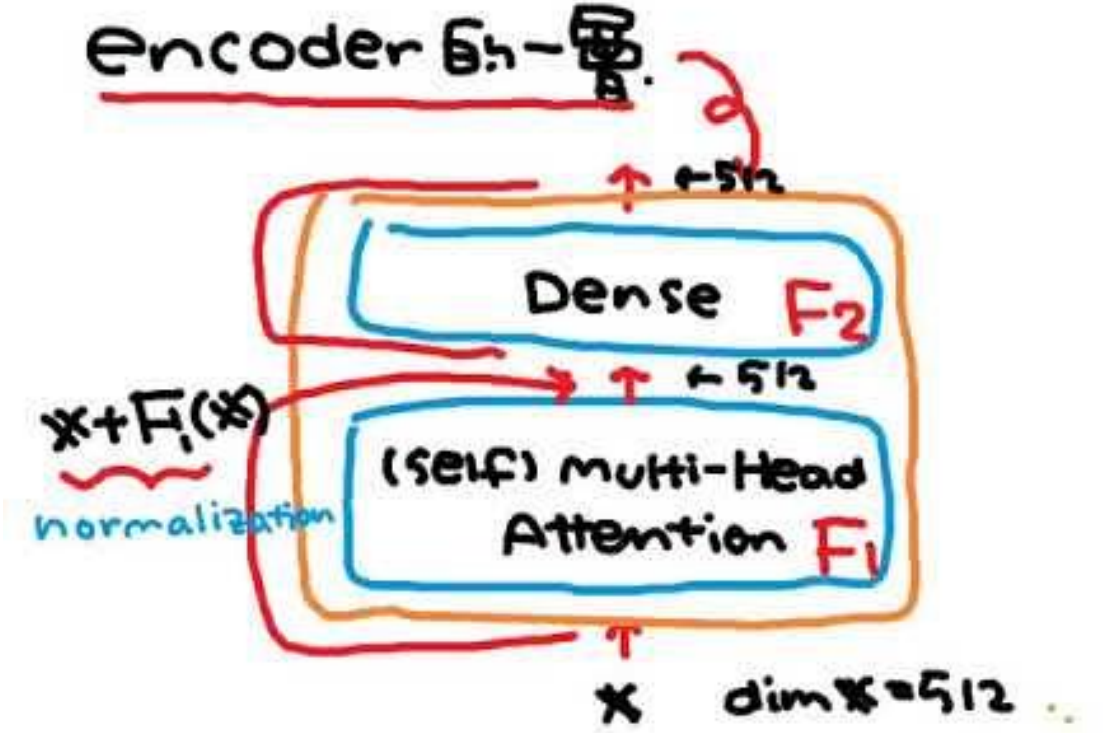


台灣人工智慧學校

Attention is all you need - Part2

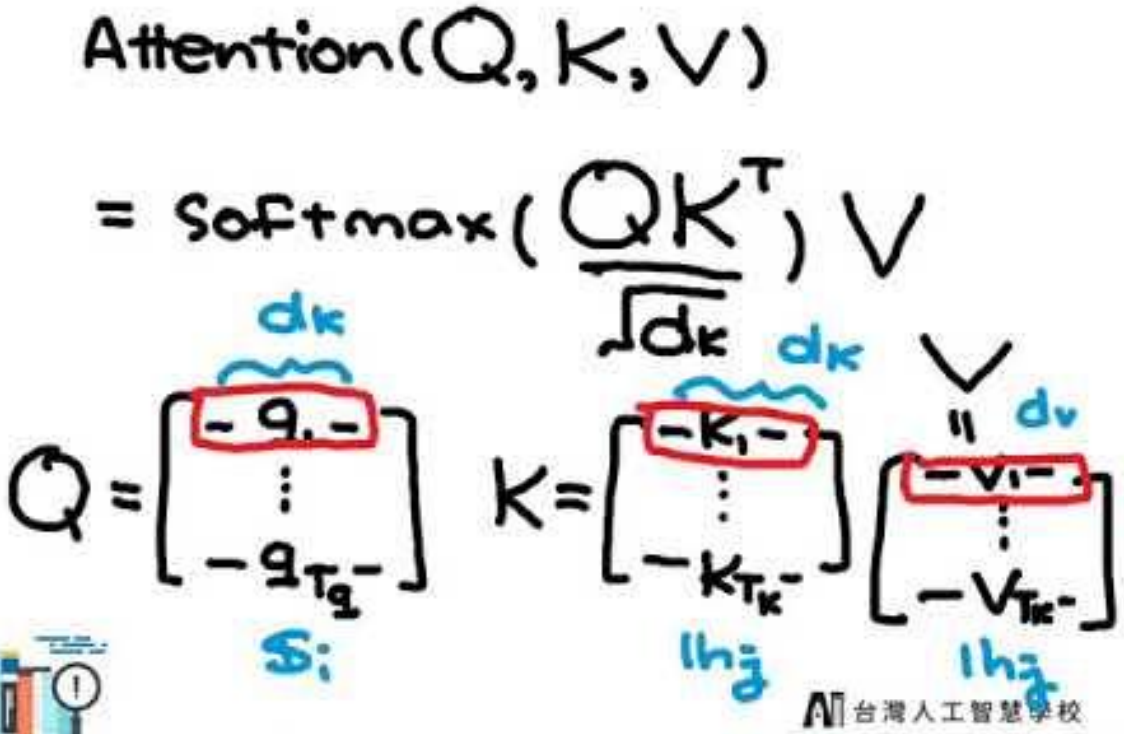
台灣人工智慧學校

Attention is all you need - Part3



台灣人工智慧學校

Attention is all you need - Part4



問卷調查

問卷調查表單連結