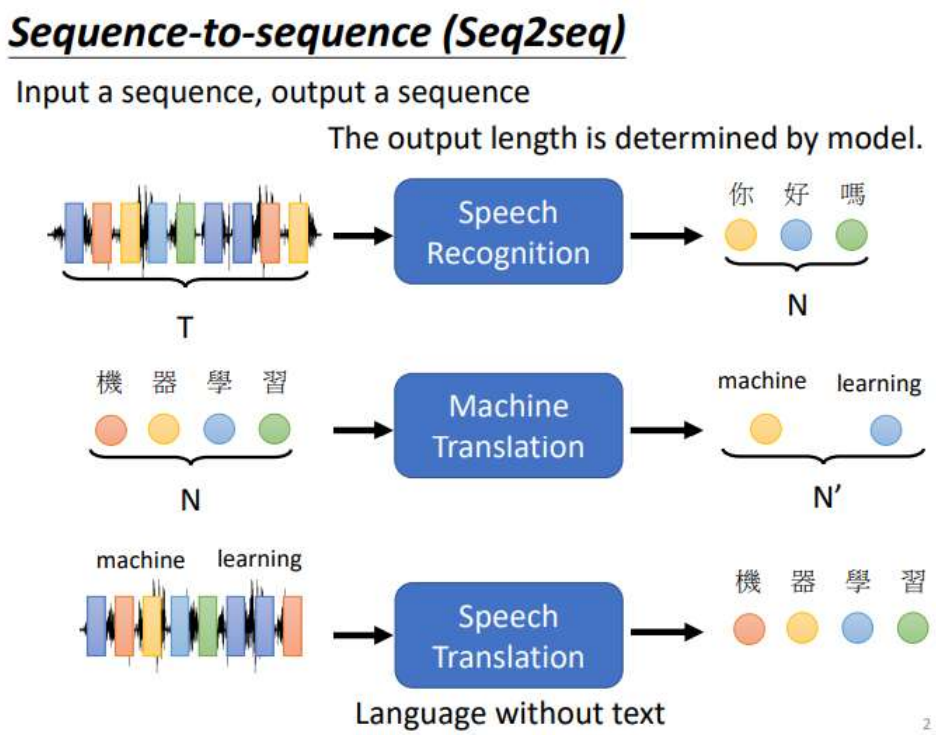
Transformer

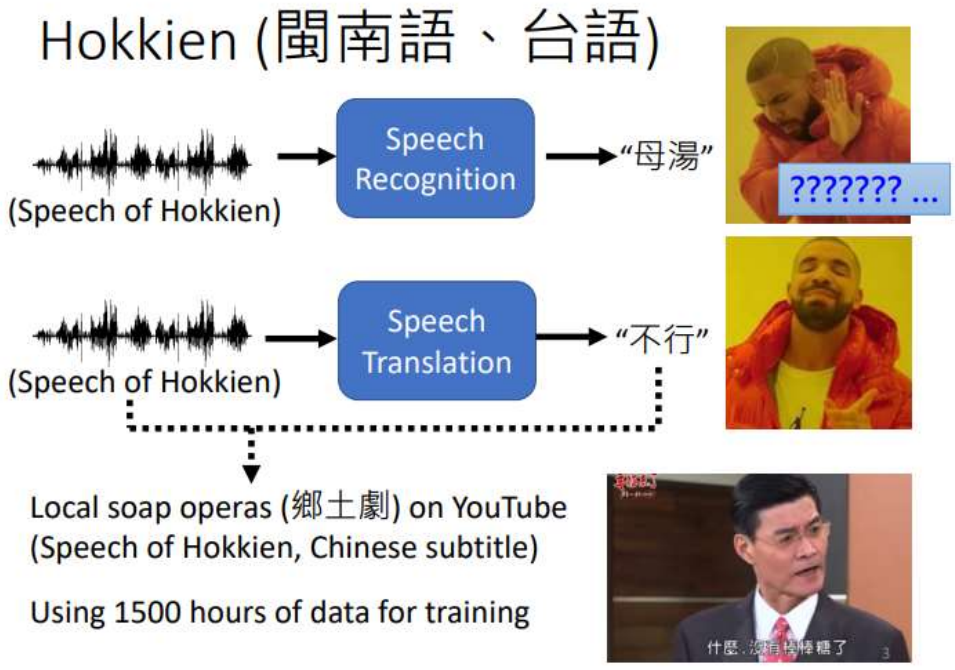
Create at 2022/06/22

- 上課資源：
  1. 【機器學習2021】Transformer (上) (https://www.youtube.com/watch?v=n9TlOhRjYoc)
  2. 【機器學習2021】Transformer (下) (https://www.youtube.com/watch?v=N6aRv06iv2g)
- 參考資料：
  1. Non-Autoregressive Sequence Generation (由助教莊永松同學講授) (https://www.youtube.com/watch?v=jvyKmU4OM3c)
  2. Pointer Network - 從輸入複製東西能力的模型 (https://www.youtube.com/watch?v=VdOyqNQ9aww)
- 參考網站：
  1. Deep Learing For Human Language Processing 2020 Spring (https://speech.ee.ntu.edu.tw/~hylee/dlhlp/2020-spring.php)

Sequence-to-sequence (Seq2seq)

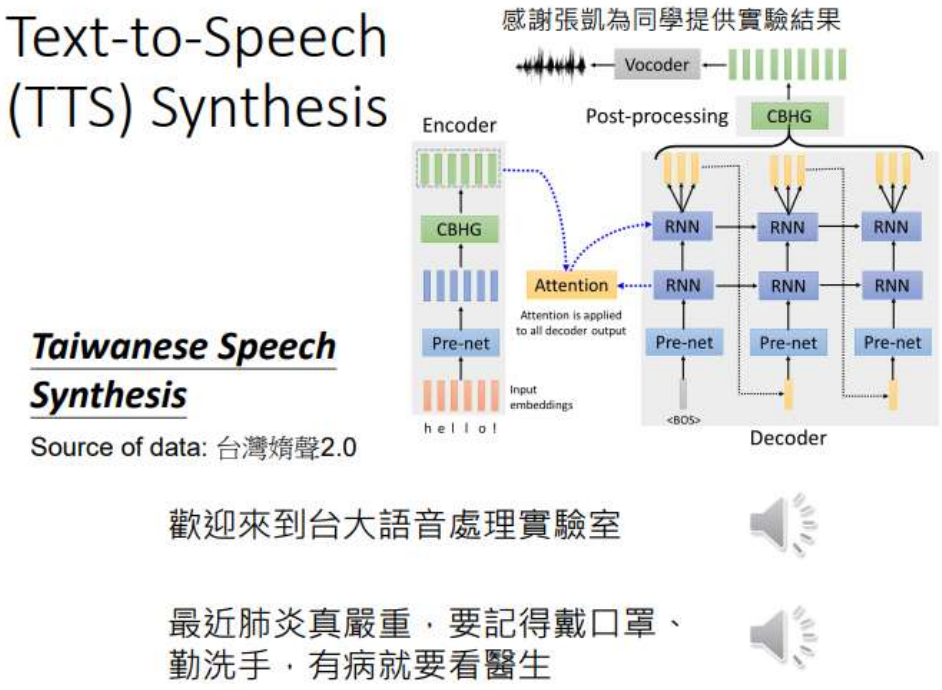


- Transformer 就是 sequence-to-sequence (Seq2seq) 的 model
  - input : sequence
  - output : sequence (多長不曉得，由機器決定)
- 應用 :
  - 語音辨識
  - 機器翻譯
  - 語音翻譯

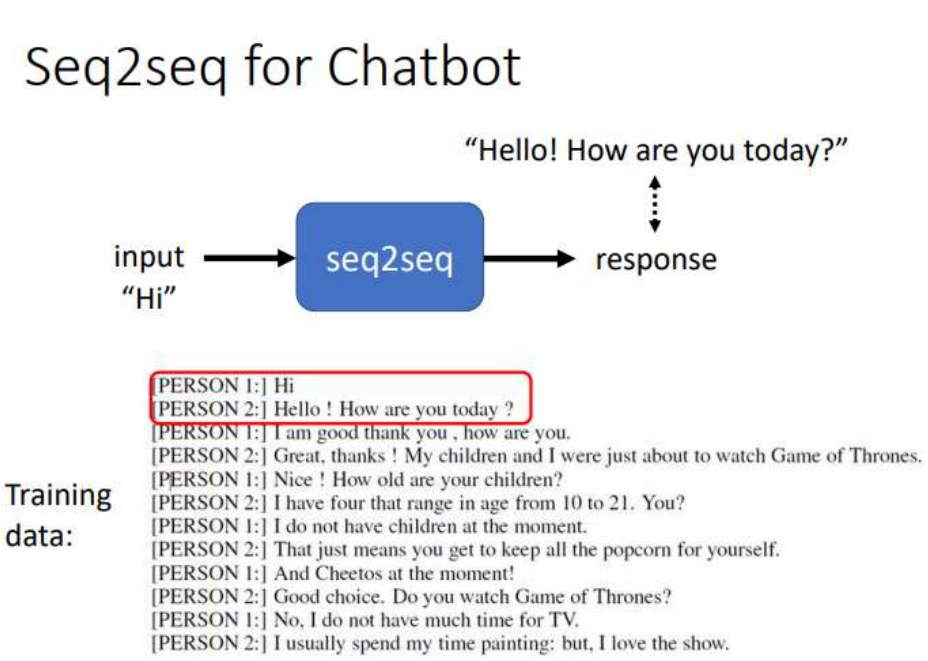


不透過中間的羅馬拼音之類的轉換
直接硬 train，行得通嗎？

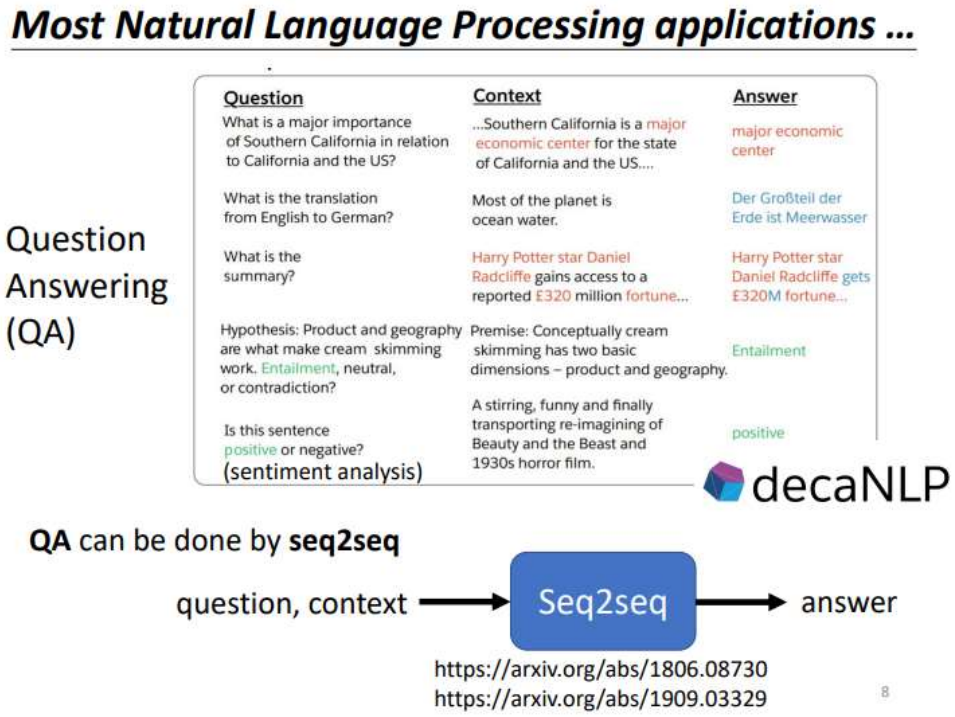
不是沒有可能



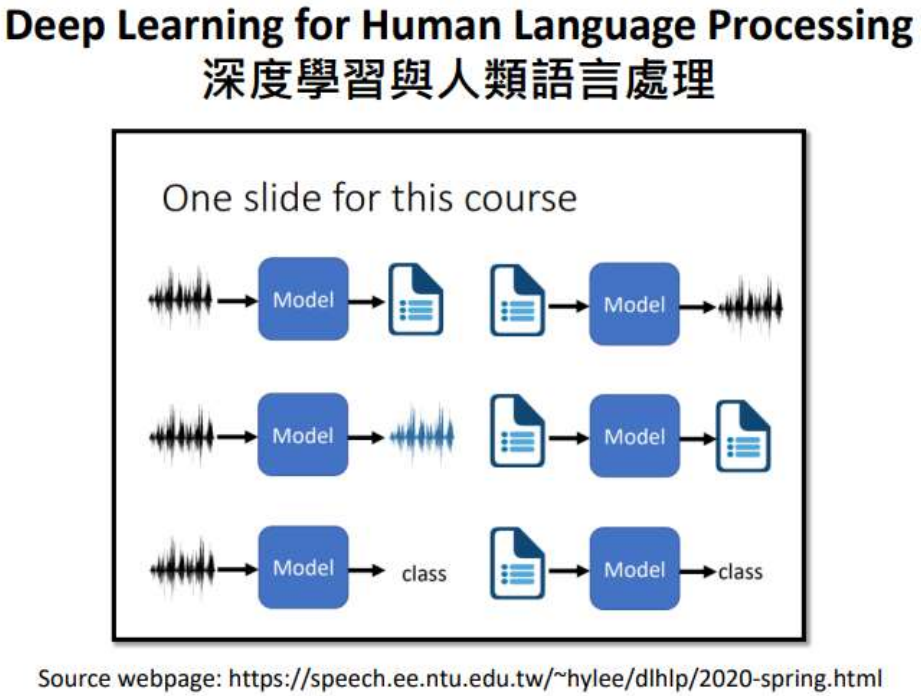
- 語音合成
  - 輸入：文字
  - 輸出：聲音訊號



- 可以用 Seq2seq model 訓練一個聊天機器人
  - 輸入：一句話 (文字 vector sequence)
  - 輸出：給出回應 (文字 vector sequence)



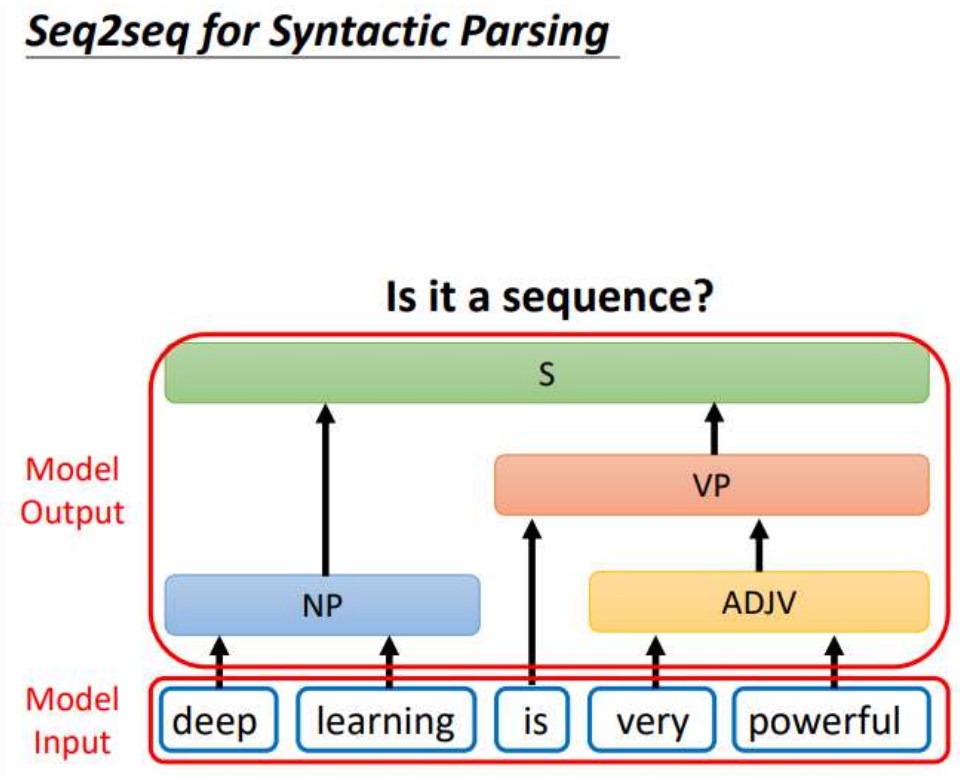
- Seq2seq model 在 NLP (natural language processing) 領域的使用很廣泛
- 很多 NLP 的任務，可以想成是 question answering
  - 給機器讀一段文字
  - 問機器一個問題
  - 希望機器回答一個正確的答案
- QA 問題可以用 Seq2seq 來解
  - 用 Seq2seq 的 model
    - 輸入：問題 + 文章
    - 輸出：問題的答案

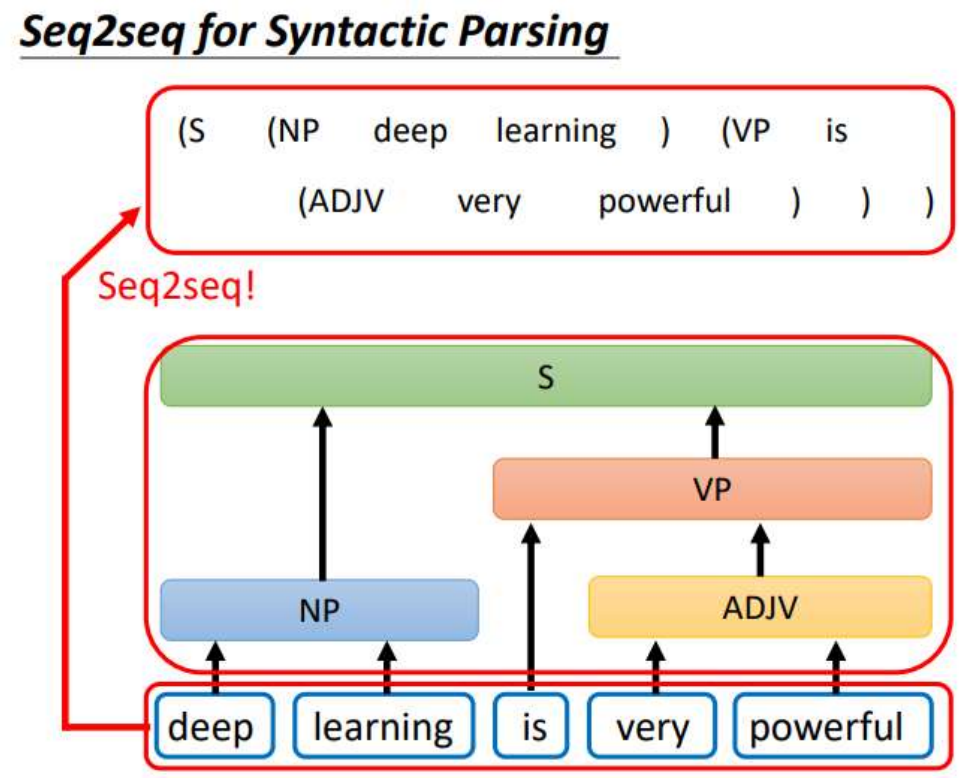


Source webpage: https://speech.ee.ntu.edu.tw/~hylee/dlhlp/2020-spring.html

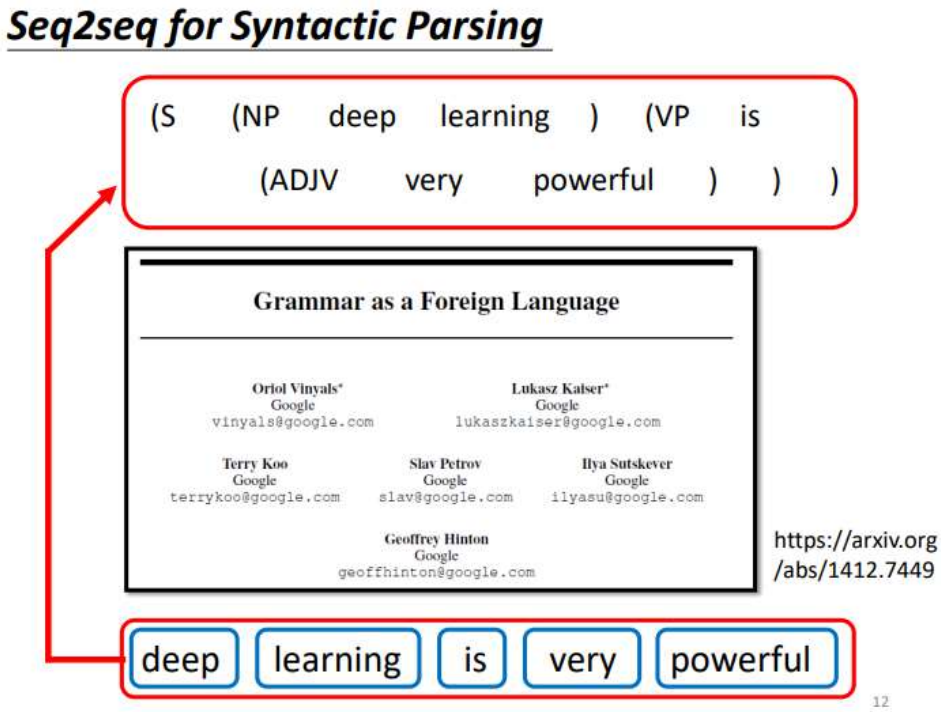
- 為不同的任務客製化各式各樣不同的模型，往往可以得到比單用 Seq2seq model 更好的結果

有很多感覺不是 seq2seq model 的問題，但是都可以硬用 seq2seq model 去解它



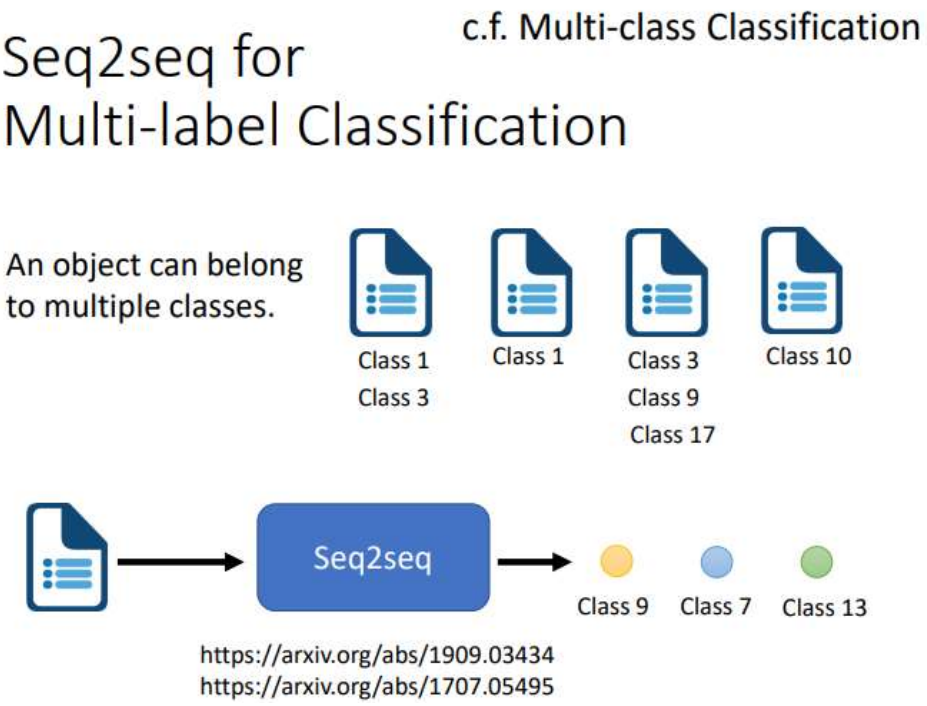


- 文法剖析
  - 輸入：一段文字 (sequence)
  - 輸出：一個樹狀的結構 (可以看作是一個 sequence)
- 可以用 seq2seq model 硬解



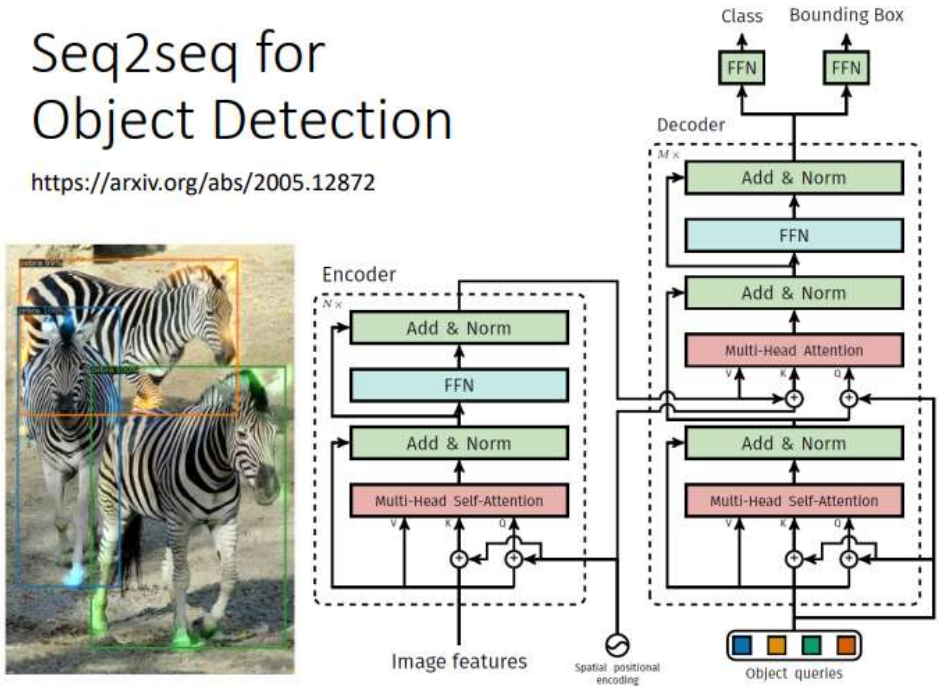
- 把文法剖析當作是一個翻譯的問題
- 把文法當作是另外一個語言

Seq2seq for Multi-label Classification



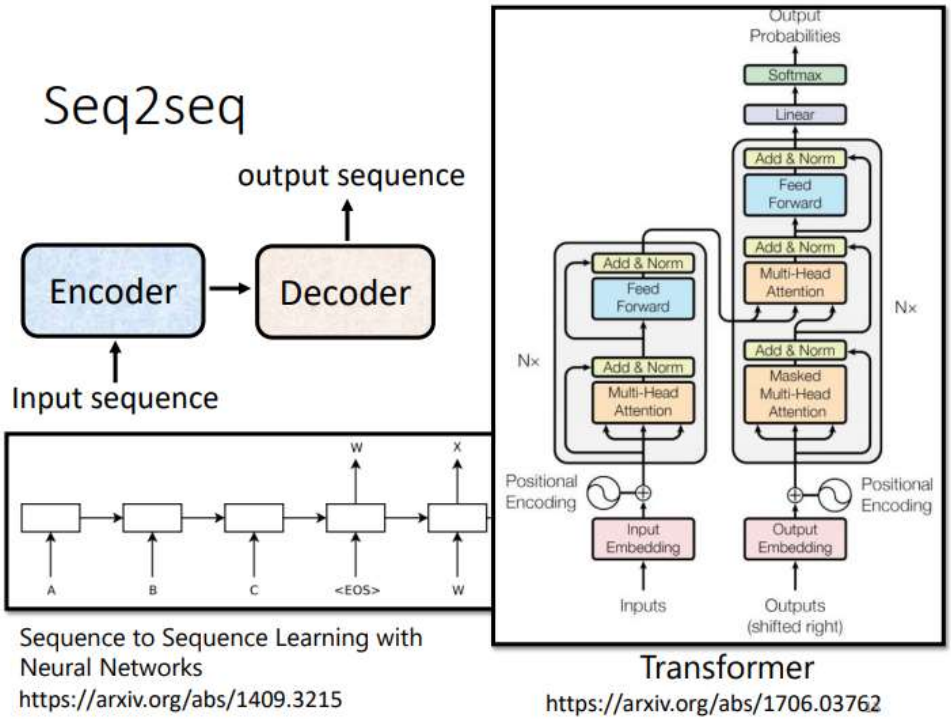
- Multi-class Classification
  - 有不只一個 class 機器要做的事情
  - 從數個 class 裡面選擇某一個 class 出來
- Multi-label Classification
  - 同一個東西，可以屬於多個 class
- 在做文章分類的時候，某篇文章可以屬於 class 1 & 3
  - 用 seq2seq 讓機器自己決定輸出幾個 class

Seq2seq for Object Detection



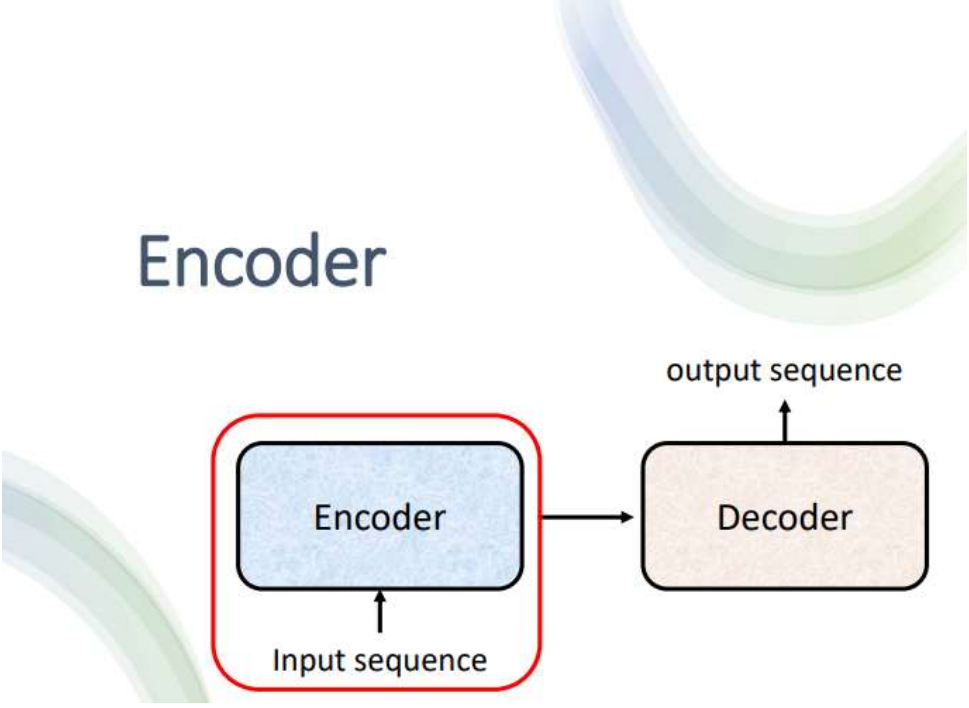
- Object detection 是給機器一張圖片，把圖片裡面的物件框出來
- 可以用 seq2seq 硬做，但是這邊不細講

Seq2seq 怎麼做呢？

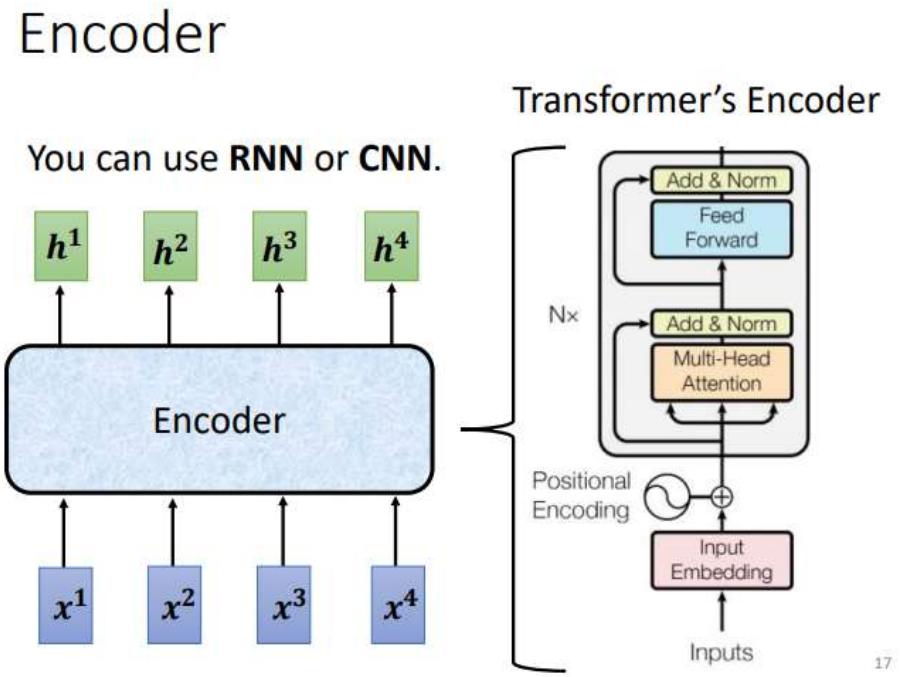


- 裡面分成兩塊
  - Encoder
  - Decoder
- input sequence 用 Encoder 處理這個 sequence，再把處理好的結果丟給 Decoder，由 Decoder 決定要輸出怎麼樣的 sequence

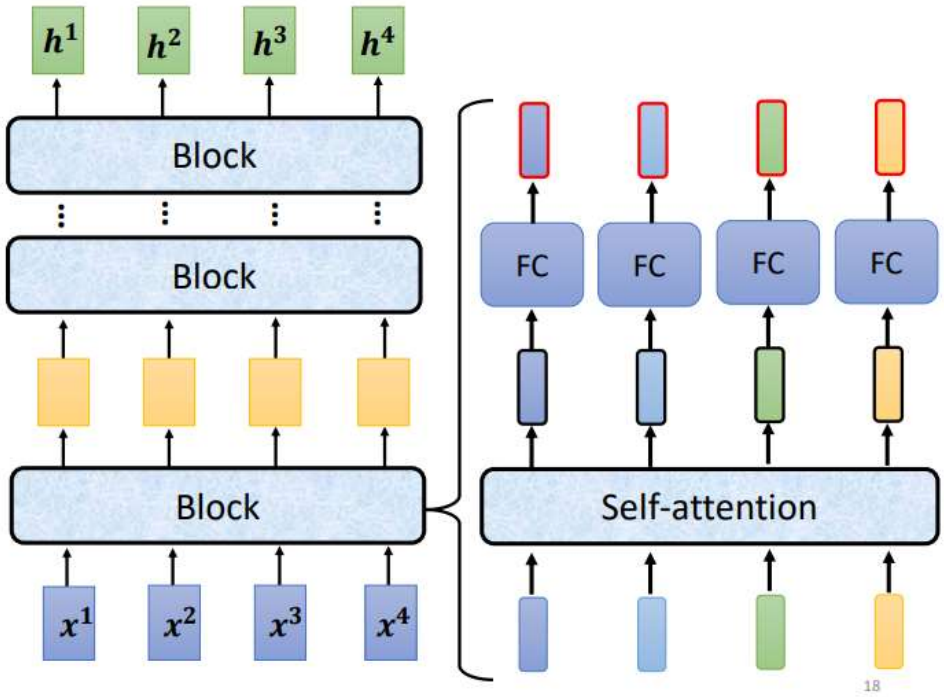
Encoder



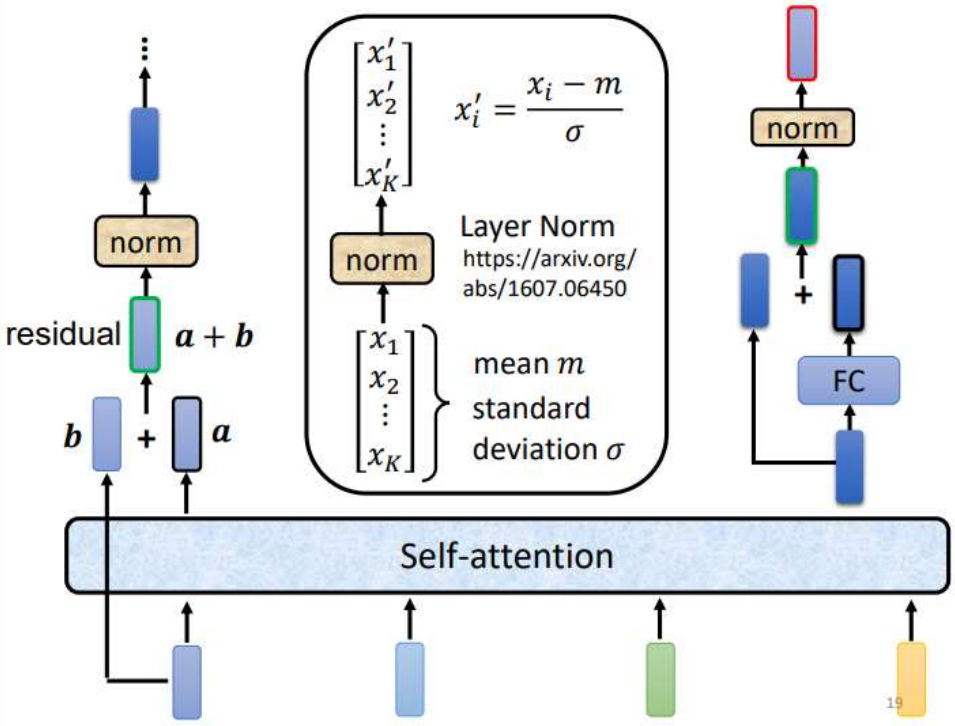
Encoder 的部分



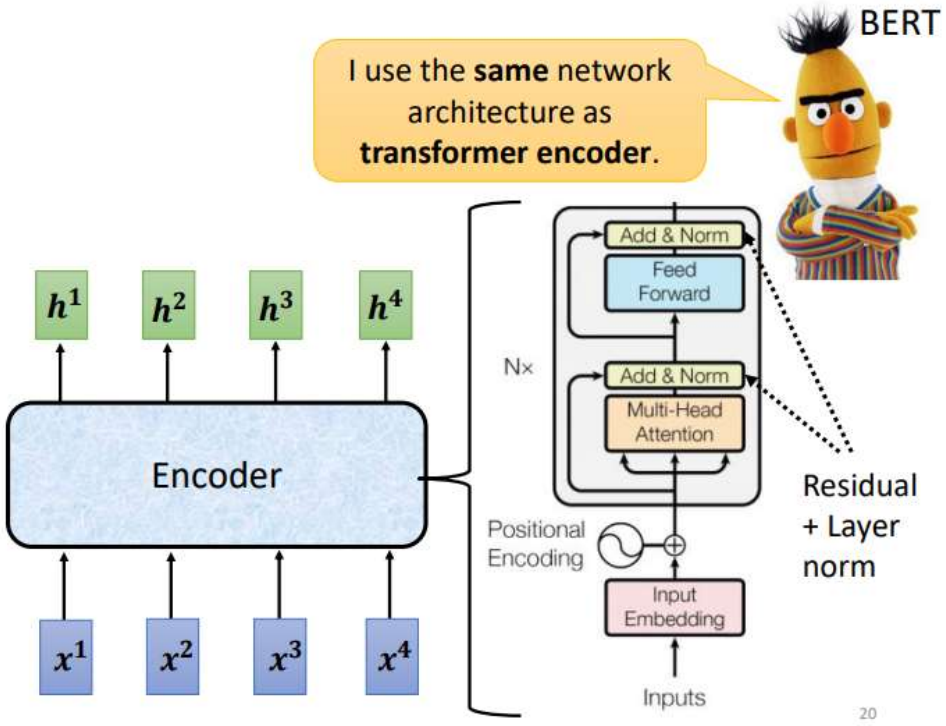
- Seq2seq's model Encoder
  - 輸入：一排向量
  - 輸出：另外一排向量
- Transformer's Encoder 用的就是 self-attention



- Encoder 裡面會分成很多的 block
- 每個 block 都是
  - 輸入：一排向量
  - 輸出：另外一排向量
- 最終的 block 輸出最終的 vector sequence
- 每個 block 不是 neural network 的一層，是因為每個 block 裡面做的事情是好幾個 layer 在做的事情
- 在 Transformer 的 Encoder 裡面，每個 block 做的事情
  - 輸入一排向量做 self-attention
  - 考慮整個 sequence 的資訊
  - 輸出另外一排向量
  - 再將輸出的向量丟到 fully connected 的 feed forward network 裡面
  - 再輸出另外一排向量，這一排向量就是 block 的輸出



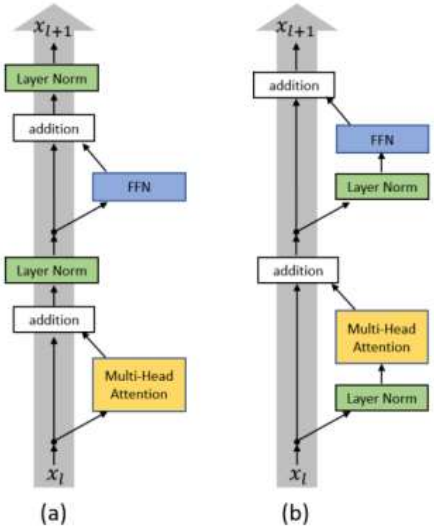
- 實際上做的事情
  - 在 transformer 裡面加入了一個設計
  - 不只是輸出向量
  - 還要把向量加上它的 input，得到新的 output
  - 接著做 normalization (layer normalization)
    - 這邊不是用 batch normalization
    - 這邊用的叫 layer normalization
      - 比 batch normalization 簡單
      - 輸入一個向量，輸出一個向量
      - 對同一個 feature 不同的 dimension，計算輸入向量的 μ、σ
      - 做 normalization 之後的輸出才是 Fully Connected Network 的輸出
      - 而 Fully Connected Network 也有 residual 的架構，並且還要再做一次 normalization
  - 這樣的 network 架構，叫做 residual connection



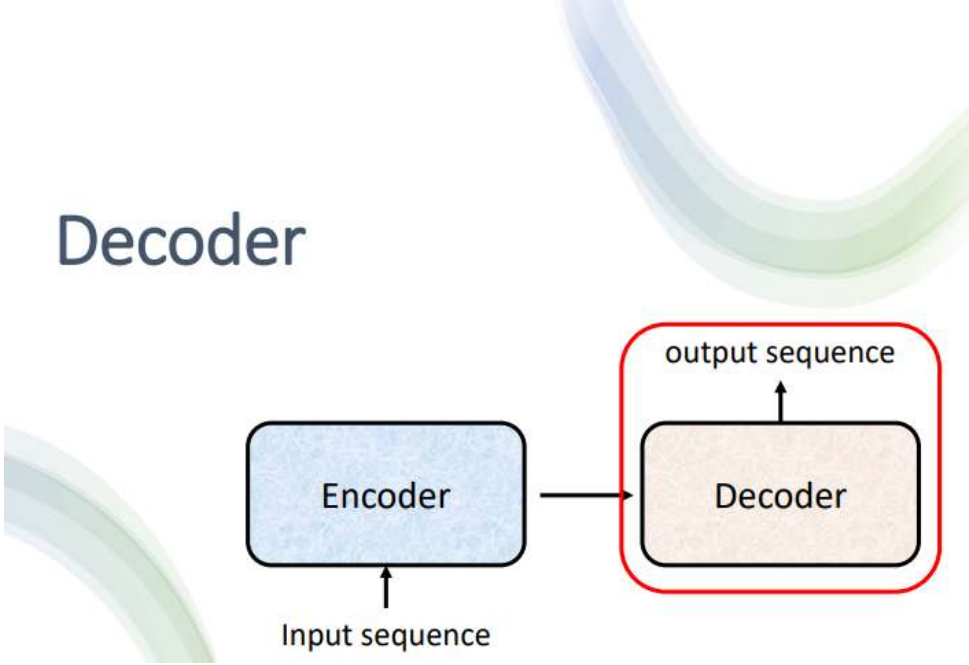
- input 的地方有加上 positional encoding
- 然後做 Multi-Head Attention (就是 self-attention 的 block)
- Add & Norm 就是 residual + layer normalization
- 接著做 fully connection 的 feed forward network
- 之後再做一次 Add & Norm
- 這是一個 block 的輸出
- 這個 block 會在複雜的 BERT 模型裡面用到
- BERT 其實就是 transformer encoder

To learn more .....

- On Layer Normalization in the Transformer Architecture
- https://arxiv.org/abs/2002.04745
- PowerNorm: Rethinking Batch Normalization in Transformers
- https://arxiv.org/abs/2003.07845

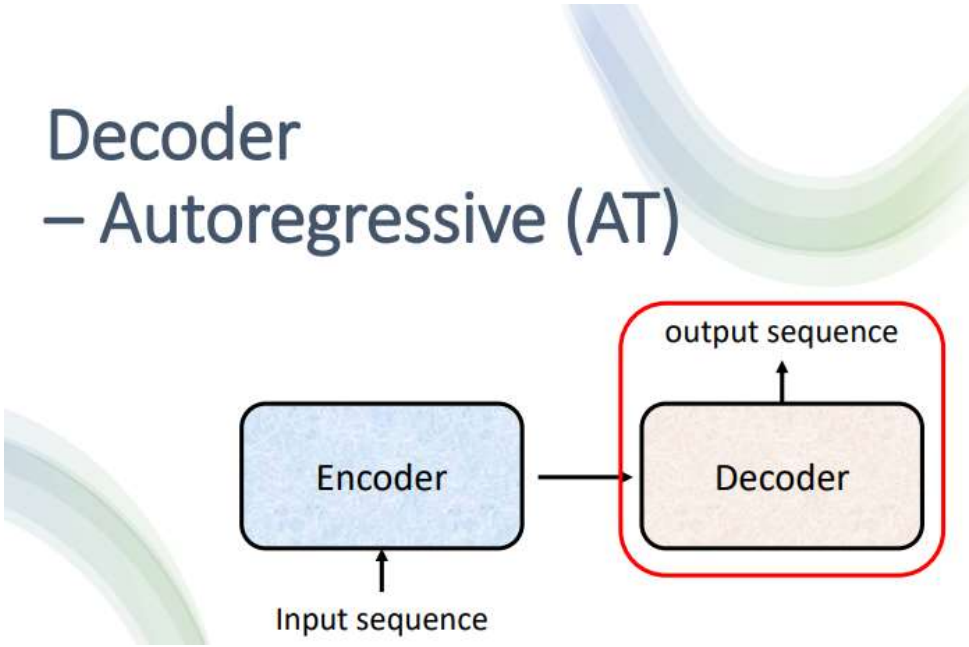


Decoder



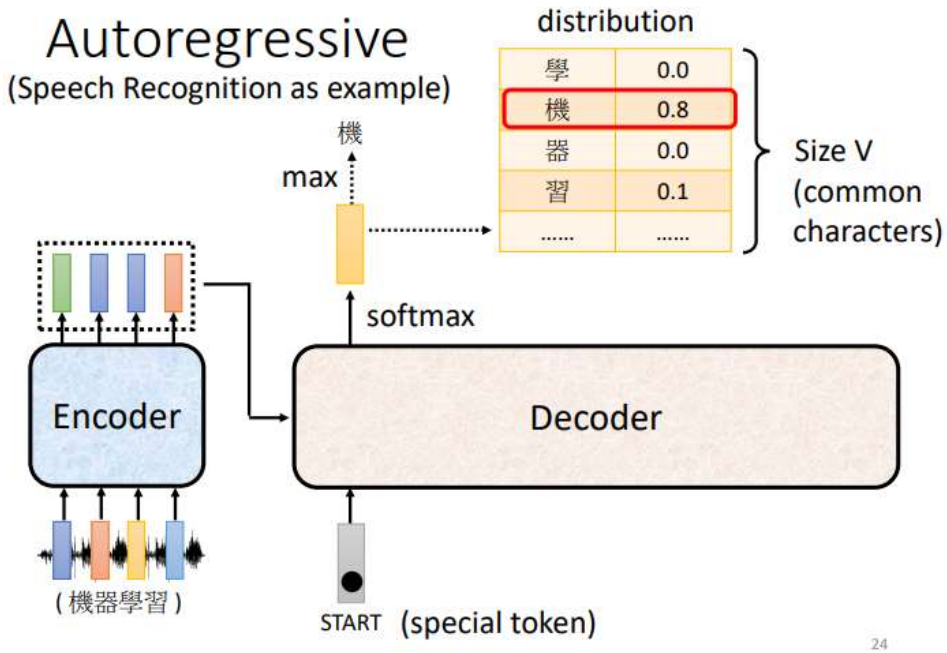
Decoder 的部分

Decoder - Autoregressive (AT)

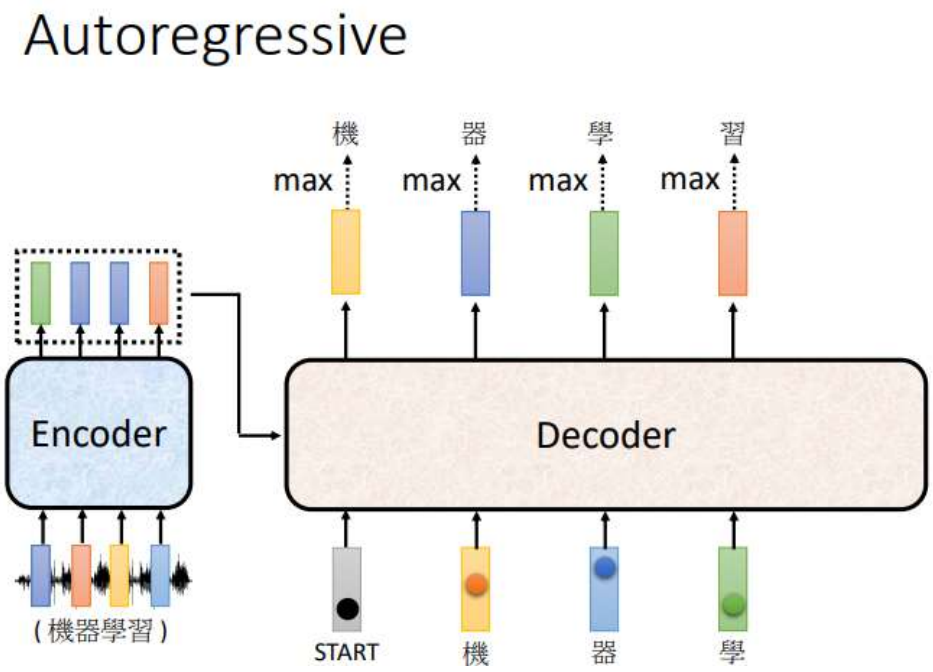


Decoder 分成兩種

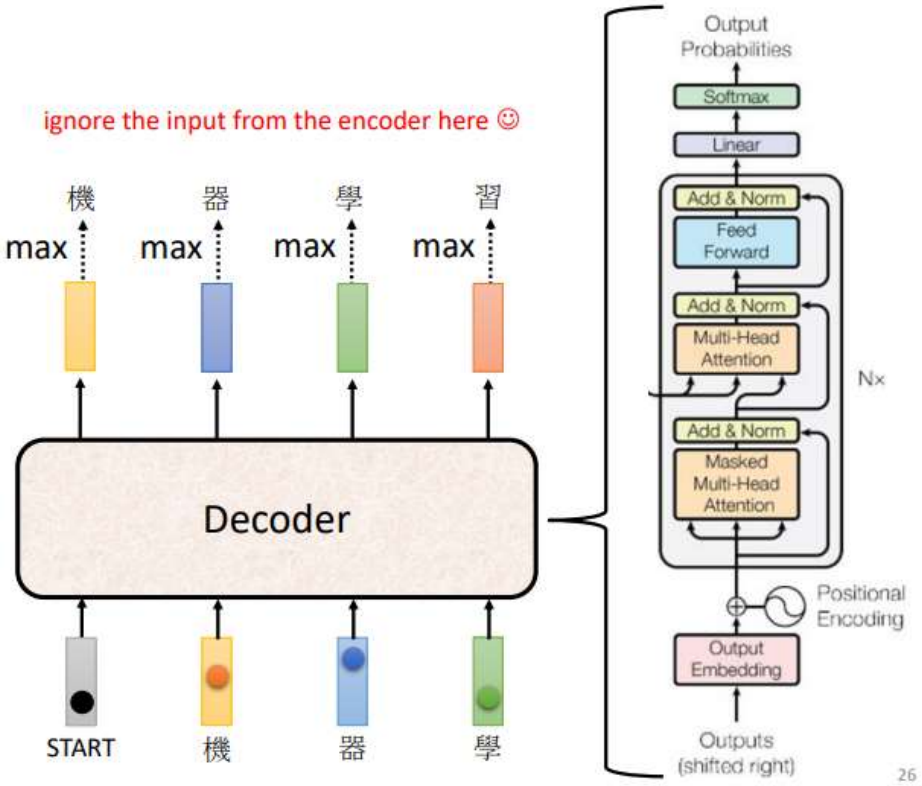
- Autoregressive 的 Decoder (比較常見)



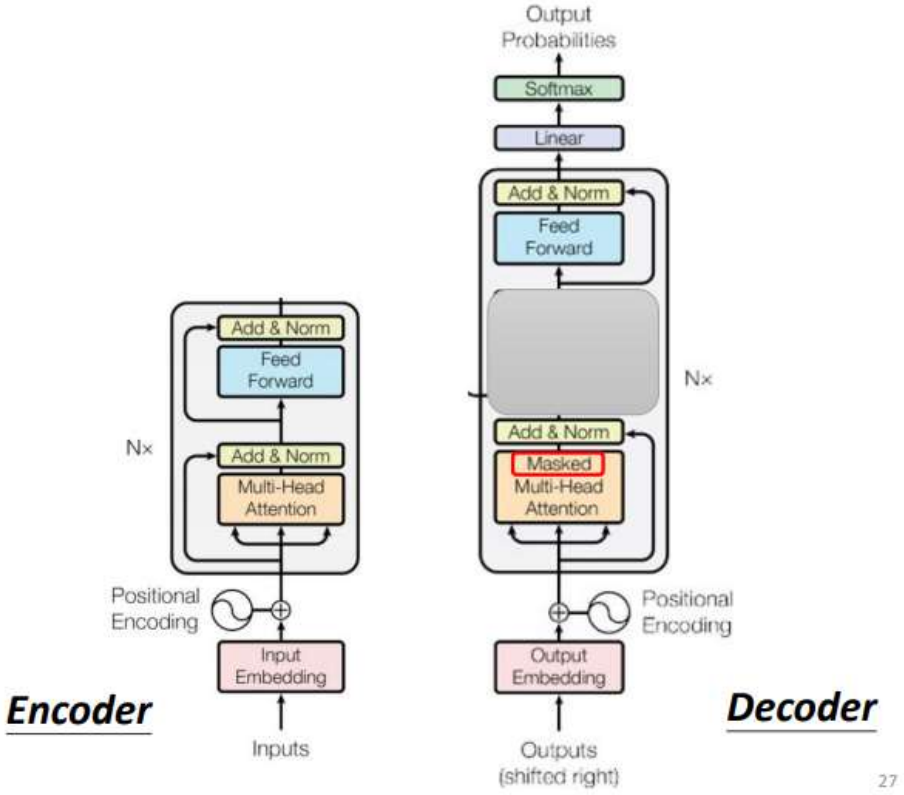
- 怎麼運作呢？
  - 用語音辨識來說明
    - 輸入：一段聲音
    - 輸出：一串文字
- 把一段聲音輸入給 Encoder，輸出會變成一排 Vector
- Decoder 要做的事情就產生輸出，產生語音辨識的結果
  - 怎麼產生語音辨識的結果？
    - 把 Encoder 的輸出先讀進去
    - 先給它一個特殊的符號代表開始，可以用一個 One-Hot 的 vector 表示
    - Decoder 會產生一個向量，這個向量的長度跟你的 vocabulary size 相同
    - 每一個中文字會對應到一個數值
    - 在產生這個向量之前，會先跑一個 softmax，跟做分類一樣，做分類在得到最終的輸出之前會先跑一個 softmax
    - 所以在產生的向量裡的分數是一個 distribution，相加起來是 1，而分數最高的那個字，就是最終的輸出



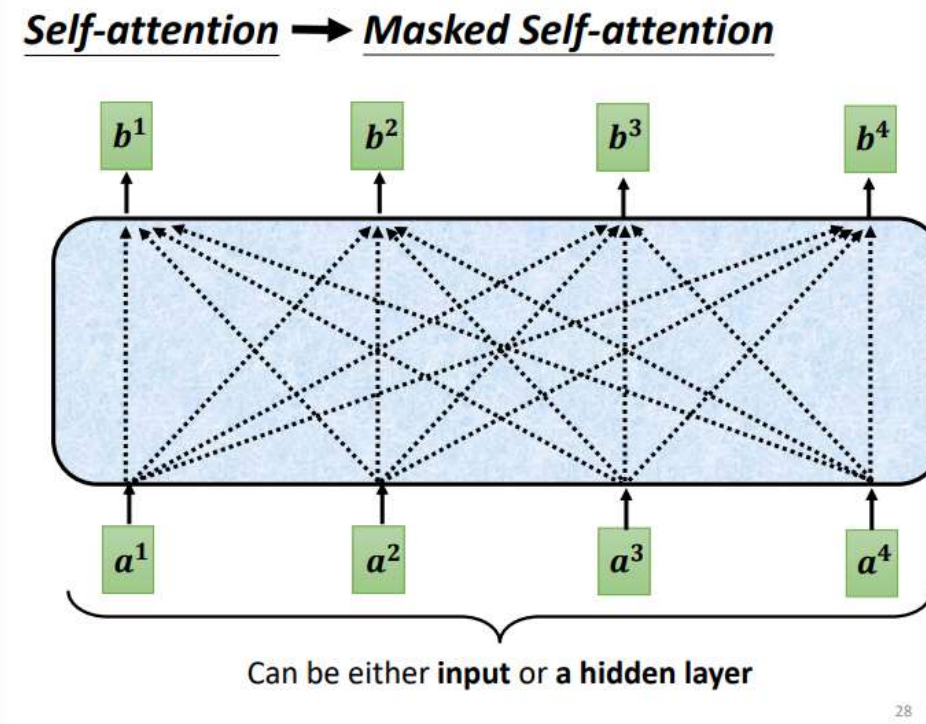
- 把 “機” 當作是 Decoder 新的 input
- 現在 Decoder 的 input 除了 START 還有 “機” (是 one-hot vector) 作為 input
- 輸出分數最高的 “器”
- 反覆以上動作



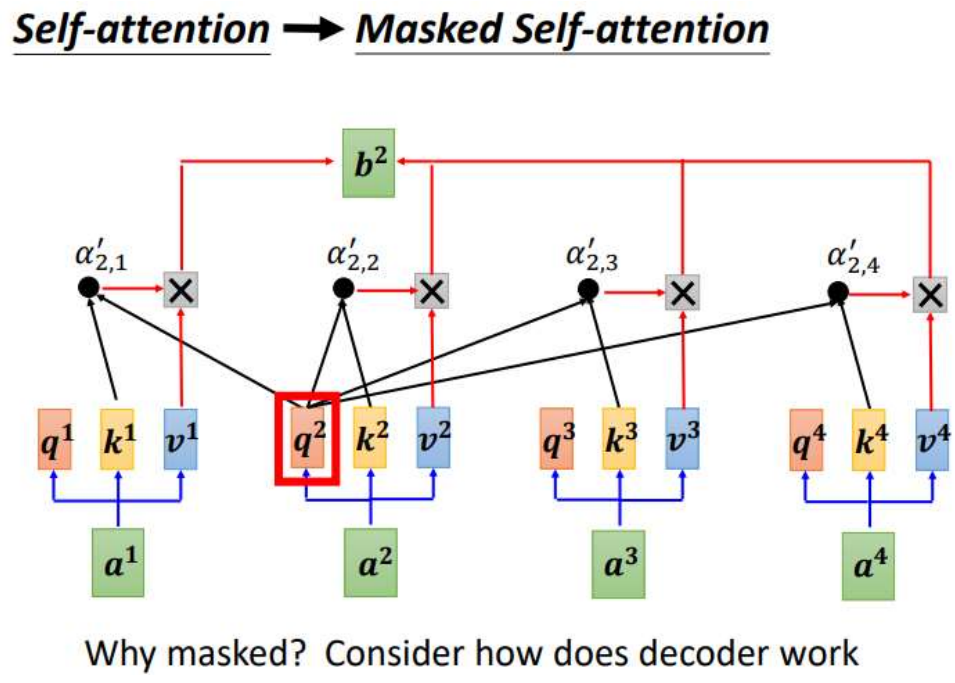
- 在 transformer 裡面，Decoder 內部的結構

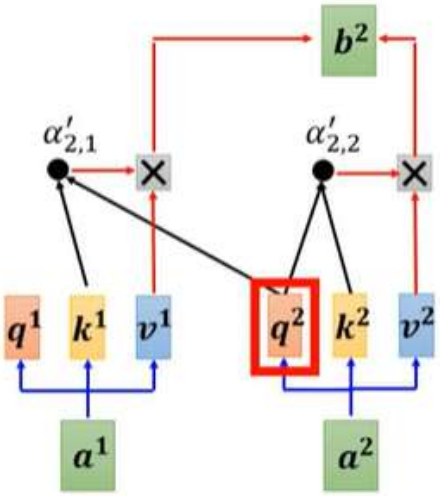


- 比較 Encoder 和 Decoder 之間的差異
- 把 Decoder 的中間遮起來，和 Encoder 的結構沒有太大的差異
- 在 Decoder 的 Multi-Head Attention Block 上多了 “Masked”



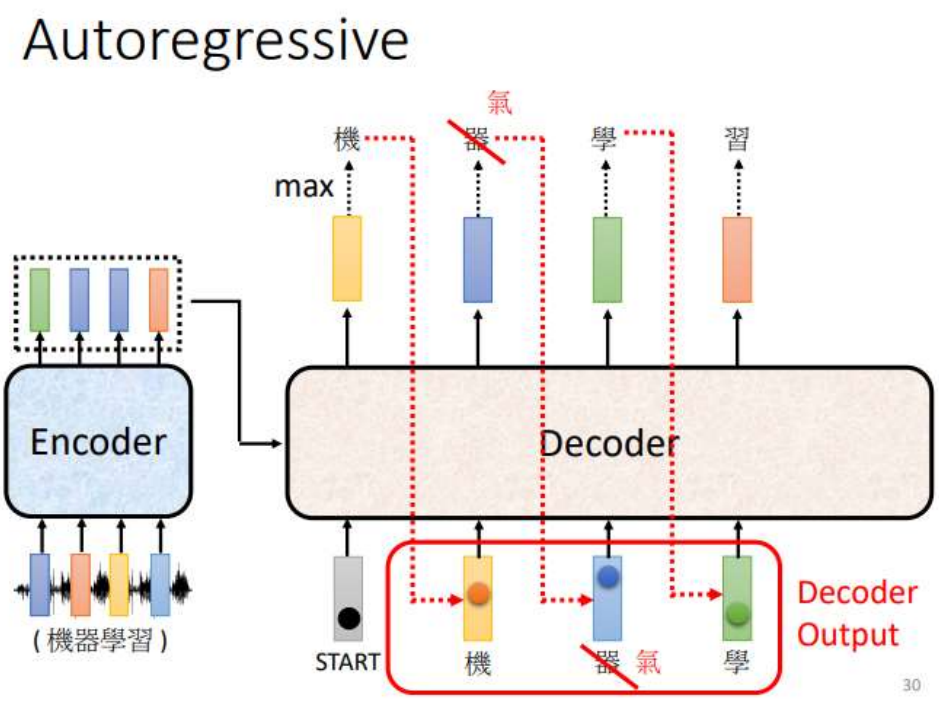
- 原來的 self-attention
  - 每一個輸出，都要看過每一個輸入才做決定
- 把 self-attention 轉成 **Masked Self-attention**
  - 不能看右邊的部分
    - 當在做 b^1 輸出的時候，只能考慮 a^1 的資訊，不能再考慮 a^2、a^3、a^4
    - 在做 b^2 輸出的時候，只能考慮 a^1、a^2 的資訊，不能再考慮 a^3、a^4
    - 在做 b^3 輸出的時候，只能考慮 a^1、a^2、a^3 的資訊，不能再考慮 a^4
    - 在做 b^4 輸出的時候，能考慮整個 sequence 的資訊

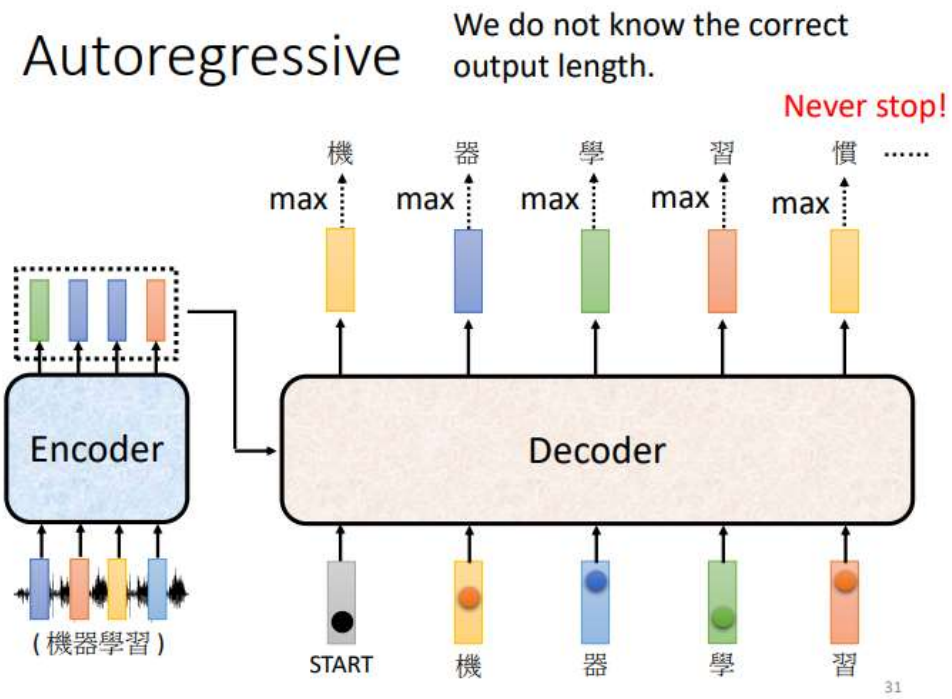




Why masked? Consider how does decoder work

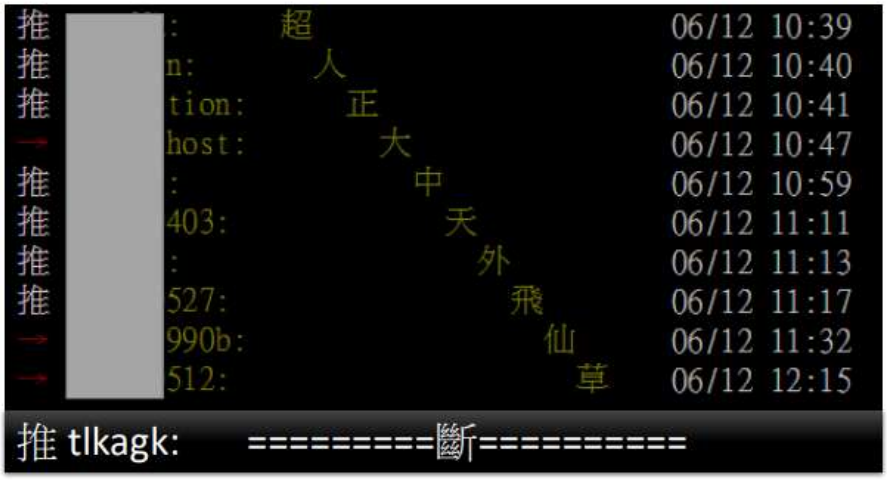
- 要產生 b^2 的時候
  - 只拿第二個位置的 query 去跟第一、二個位置的 key 去計算 attention

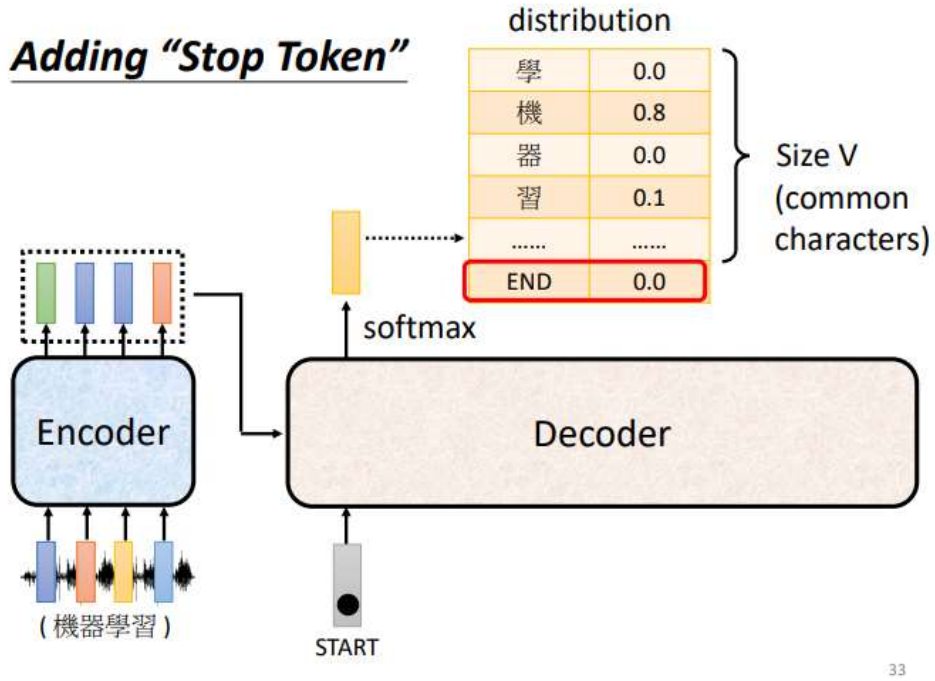




- Decoder 必須自己決定輸出的 sequence 長度

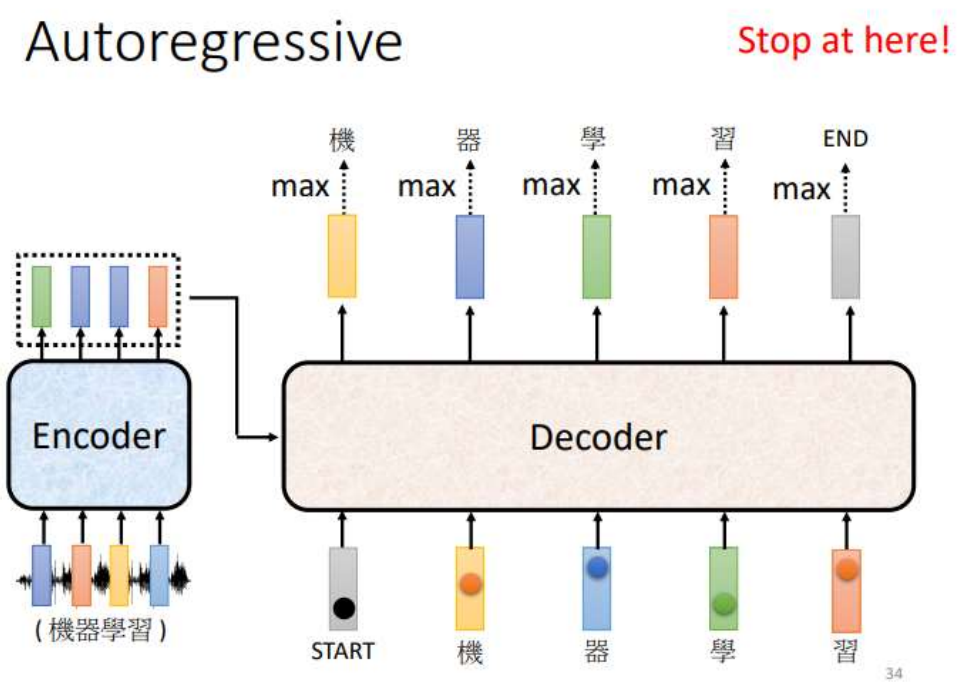
推文接龍 (Tweet Solitaire)





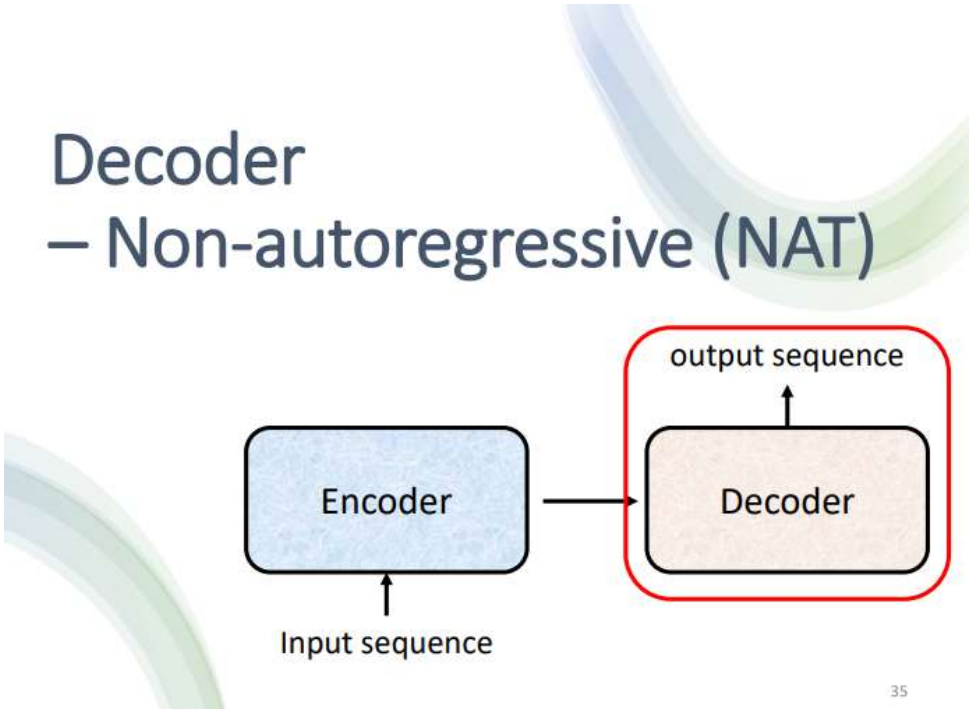
- 要準備一個特別的符號 "END"

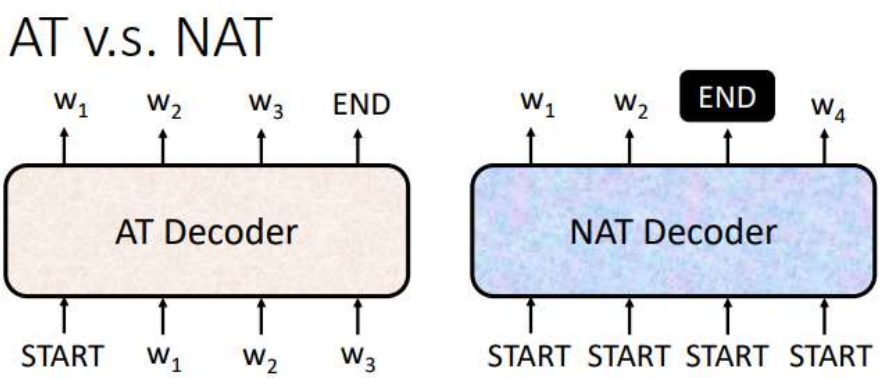
Autoregressive Decoder



- 把 "習" 當作輸入之後，Decoder 就要能夠輸出 "END"
- 就 Decoder 看到 Encoder 輸出的 Embedding
  - 看到輸入 "START"、"機"、"器"、"學"、"習"
  - 要能輸出 "END"
  - 整個 Decoder 產生 Sequence 的過程就結束

Decoder - Non-autoregressive (NAT)





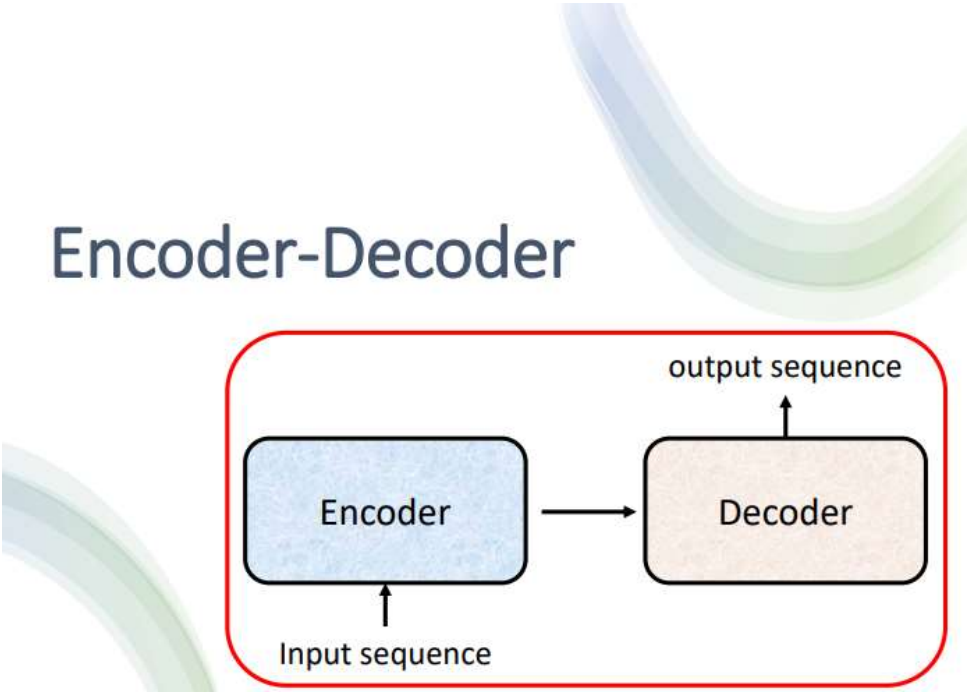
- How to decide the output length for NAT decoder?
  - Another predictor for output length
  - Output a very long sequence, ignore tokens after END
- Advantage: parallel, more stable generation (e.g., TTS)
- NAT is usually worse than AT (why? **Multi-modality**)

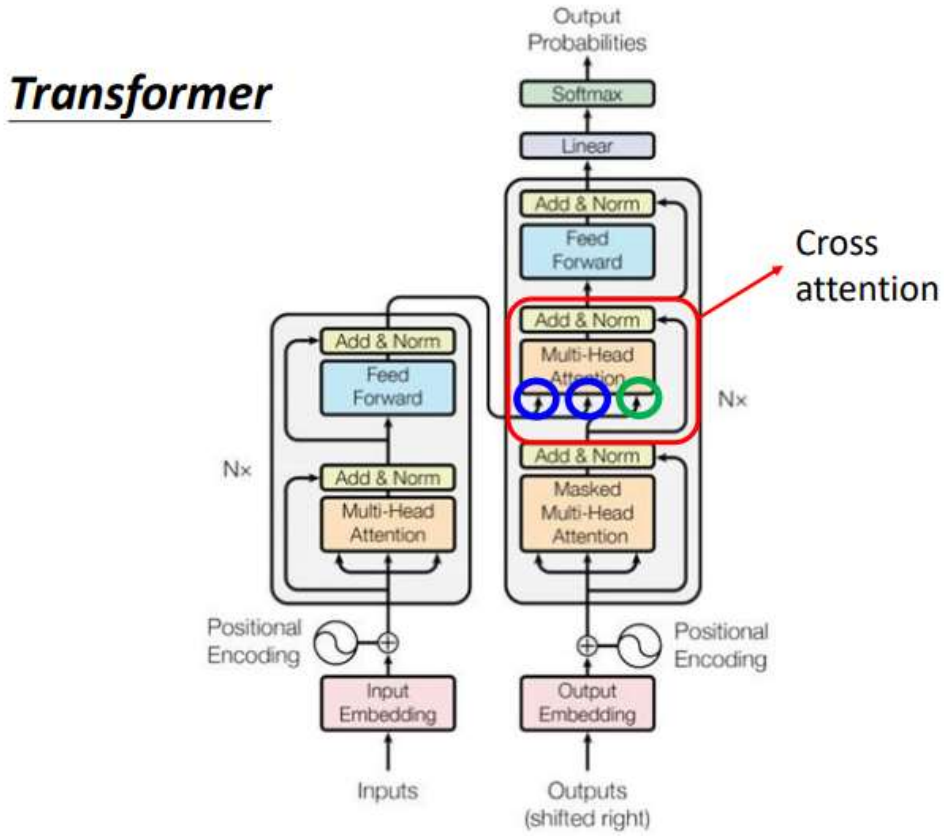
36

- AT
  - 先輸入 START，然後輸出 w_1
  - 輸入 w_1，輸出 w_2，直到輸出 “END” 為止
- NAT
  - 一次把整個句子產生出來
  - 一次輸入一整排的 START，一次輸出一排 token 就結束了
  - 只需要一個步驟
  - 問題：
    - 不知道輸出的長度是多少，怎麼知道 START 要放幾個當作 NAT Decoder 的輸入
      - 方法 1：另外扔一個 classifier，這個 classifier 吃 encoder 的 input，輸出一個數字代表 decoder 應該要輸出的長度
      - 方法 2：直接給一堆 START 的 token，然後看什麼地方輸出 “END”，“END” 的右邊就當作它沒有輸出

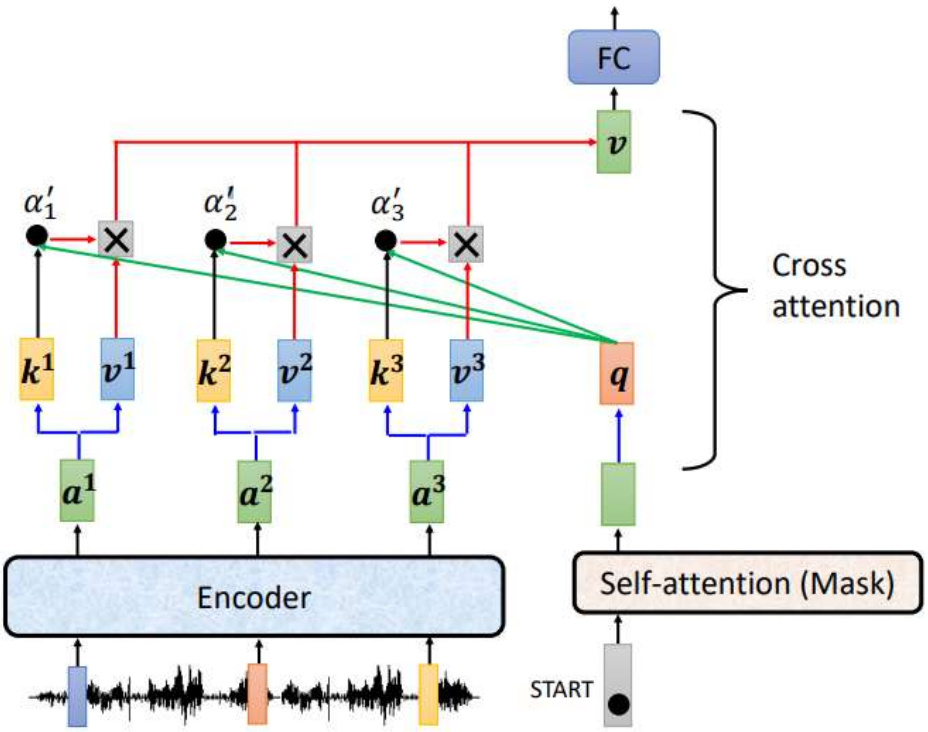
- NAT
  - 優點：平行化，所以速度比較快，比較能控制它輸出的長度
  - 缺點：performance 往往不如 AT
    - 問題：Multi-Modality

Encoder-Decoder 如何傳遞資訊？



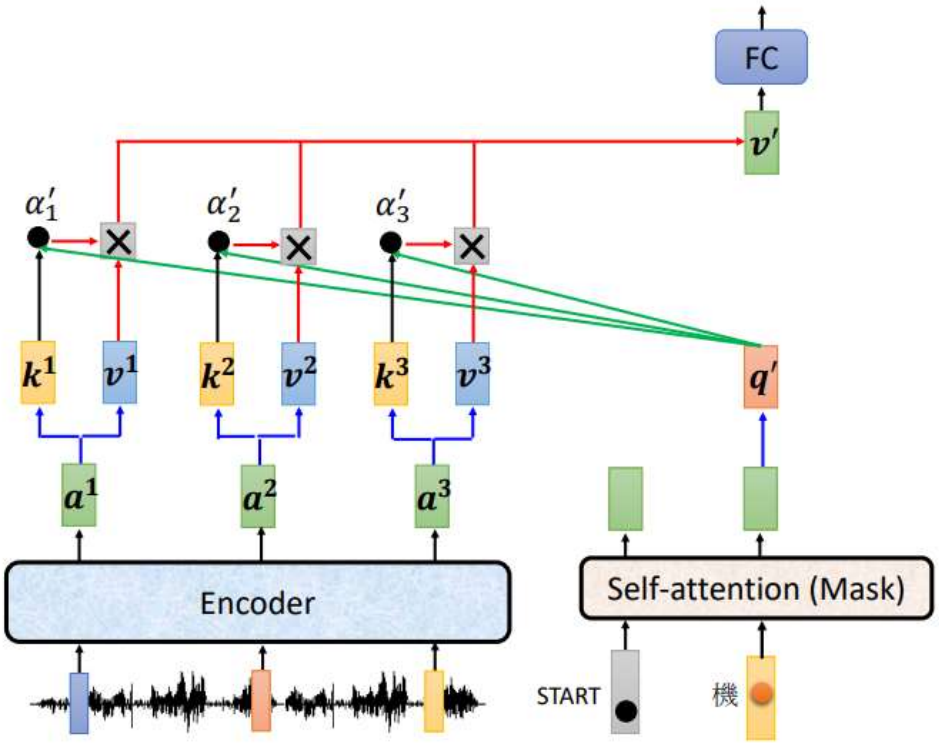


- 觀察剛剛遮起來的那塊，這塊稱為 Cross attention
- 是連接 Encoder 跟 Decoder 之間的橋樑
- 從左邊兩個箭頭 (藍色)，可以讀到 Encoder 的輸出
- 右邊一個箭頭 (綠色)，讀到 Decoder 的輸入

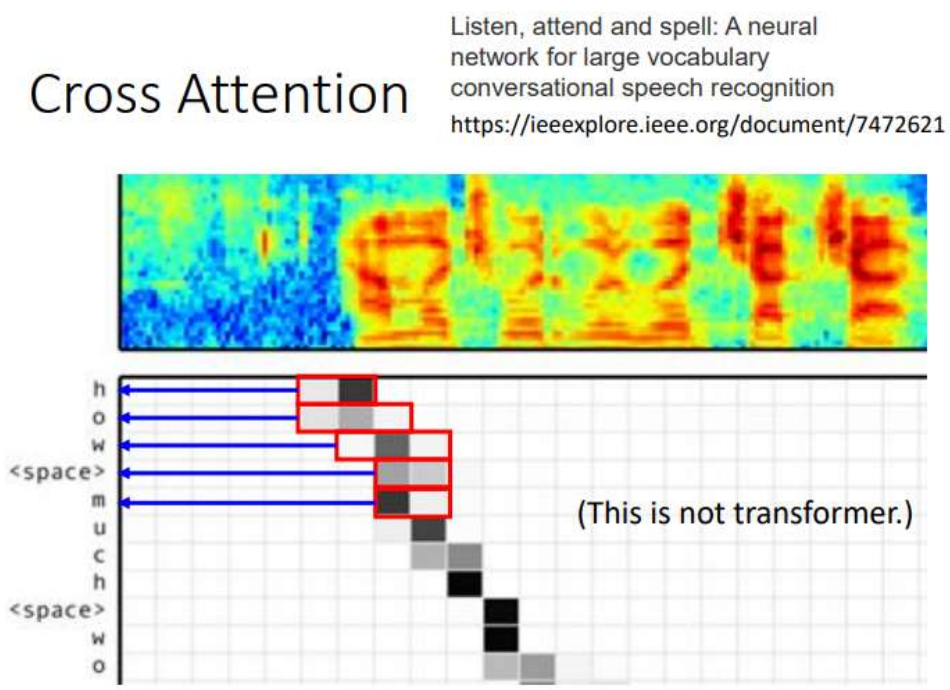


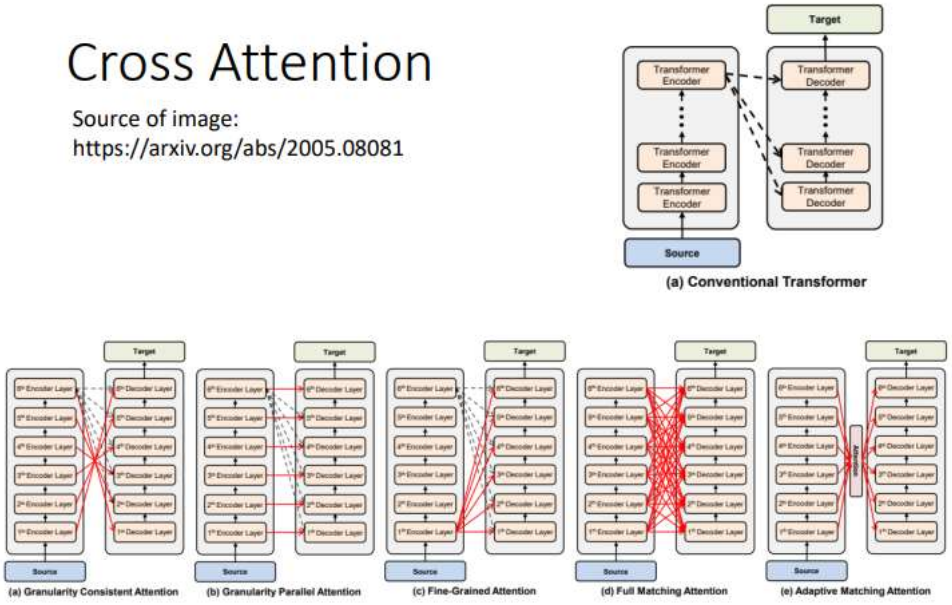
- 運作過程
  - Encoder :
    - 輸入一排向量，輸出一排向量 a^1、a^2、a^3
  - Decoder :
    - 先吃 START 當作 special token 輸入，通過 mask self-attention，得到一個向量
      - 就算 self-attention 是有做 mask，還是一樣輸入多少長度的向量，輸出就是多少向量
    - 把這個向量乘上一個矩陣做 transform，得到 query q
  - Encoder :
    - a^1、a^2、a^3 產生 key k^1、k^2、k^3
  - 把 q 跟 k^1、k^2、k^3 去計算 attention 的分數，得到 α_1、α_2、α_3
    - α'_1、α'_2、α'_3 代表可能做過 softmax、normalization
  - 接著把 α'_1、α'_2、α'_3 乘上 v^1、v^2、v^3
  - 再把它 weighted sum 加起來得到 v
  - v 就是接下來會丟到 Decoder 裡的 fully-connected network 做處理

- q 來自 Decoder
- k、v 來自 Encoder
- 這個步驟就叫做 Cross Attention



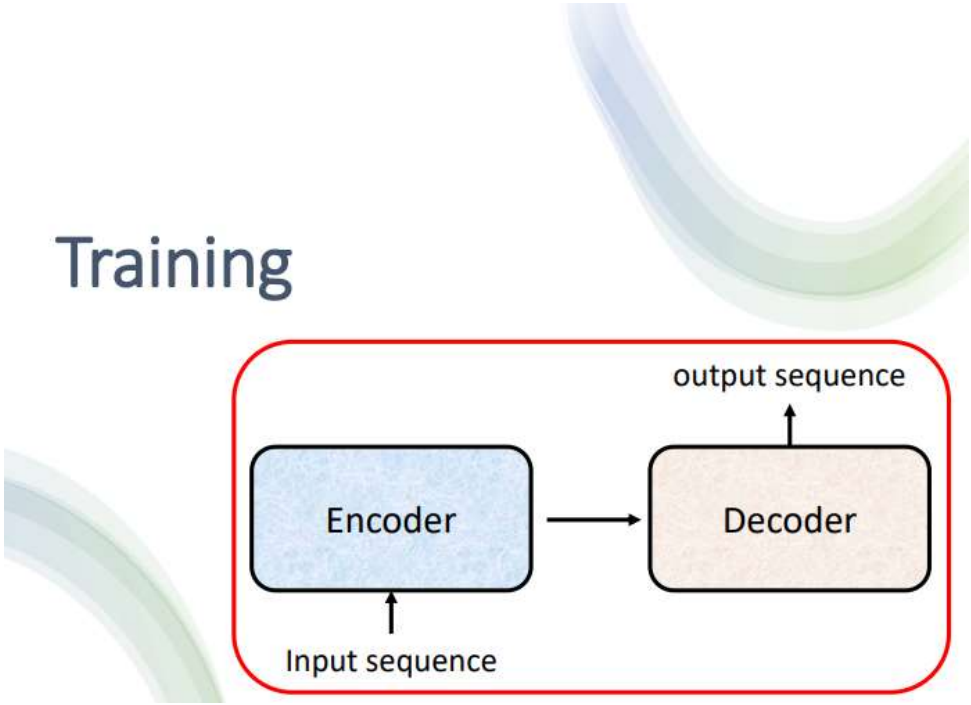
- 過程與上一張投影片相同



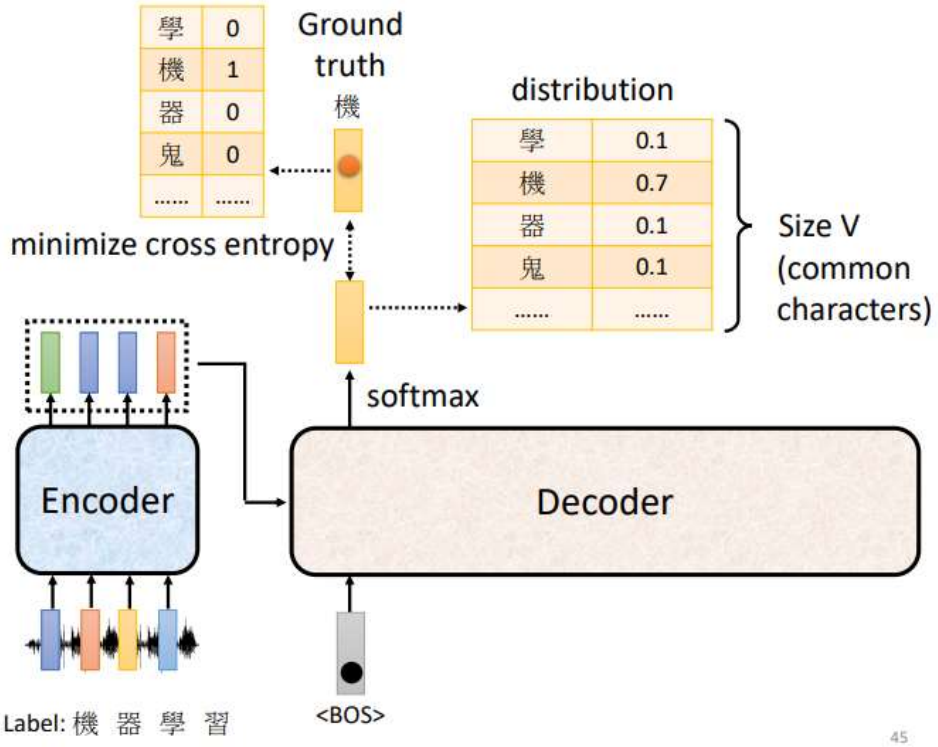


- 可以有各式各樣不同的連接方式

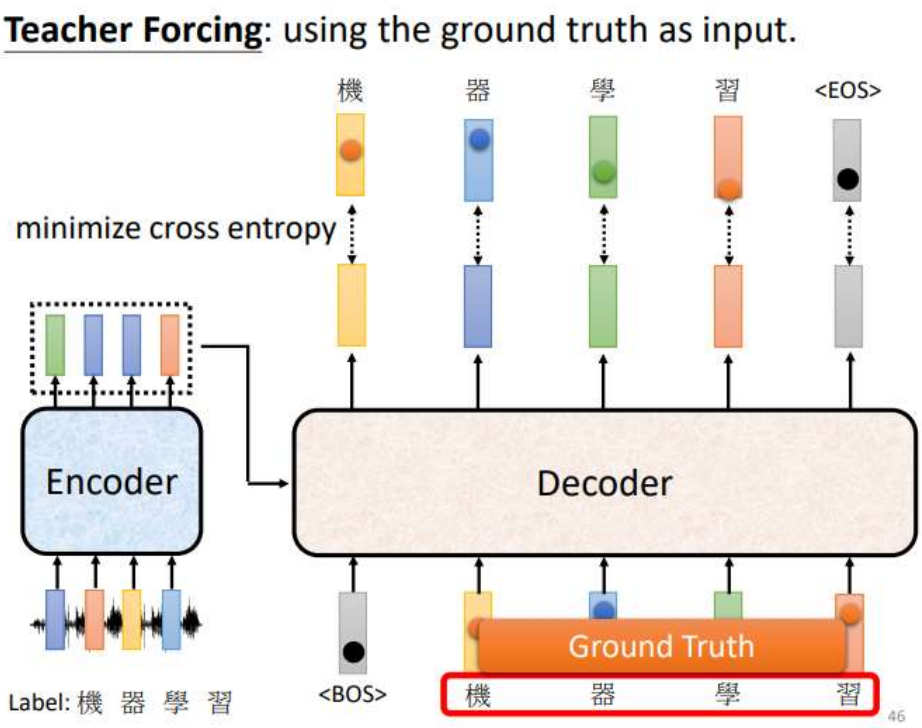
Training



- 如何做訓練的？
- 前面都是介紹模型訓練好之後是怎麼運作的

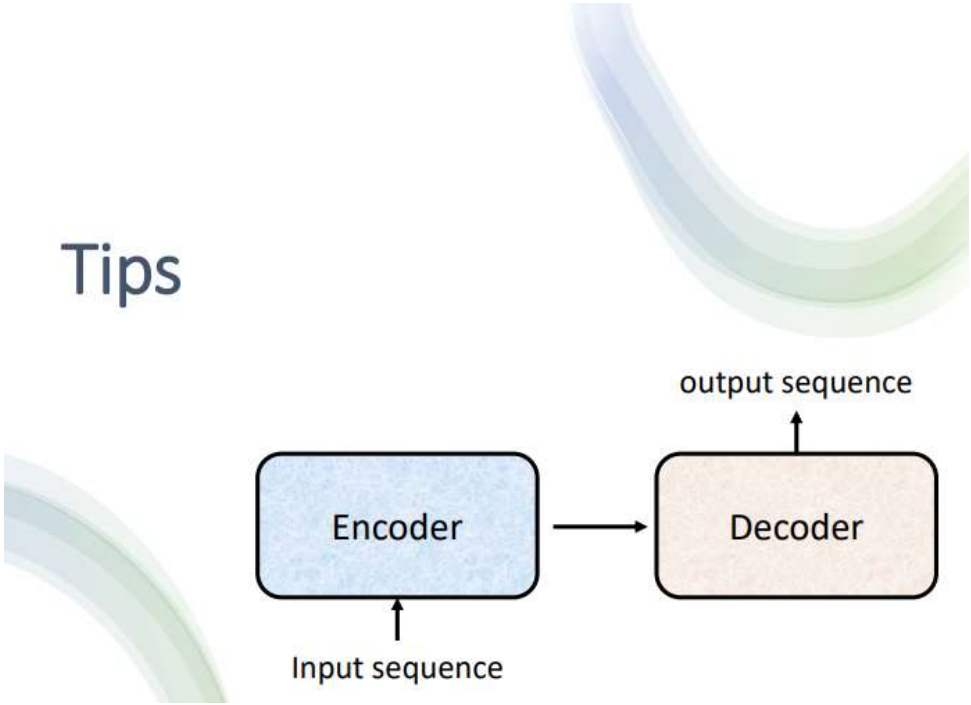


- 把 START 輸入到 Decoder 的時候，第一個輸出要跟 “機” 越接近越好
  - Decoder 的輸出是一個 distribution，希望這個機率的分佈跟 One-Hot vector 越接近越好
  - 所以會去計算 ground truth 跟 distribution 它們之間的 cross entropy
  - 希望 cross entropy 越小越好



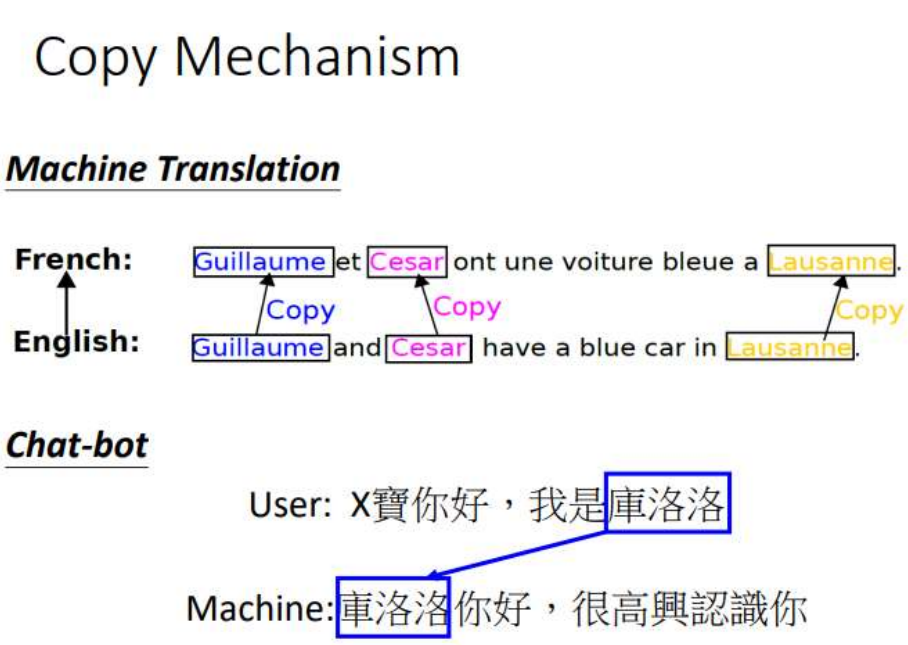
- 實際訓練：
  - 我們知道輸出應該是 “機”、“器”、“學”、“習” 四個字
  - 告訴 Decoder 希望我們的輸出跟四個字的 One-Hot Vector 越接近越好
  - 在訓練的時候，每個輸出跟它的 One-Hot Vector 都會有一個 Cross Entropy
  - 希望所有 Cross Entropy 的總和越小越好
  - 最後還要記得輸出 “END”，也要跟它的 One-Hot Vector 它的 Cross Entropy 越小越好
  - Decoder 的輸入是正確答案，在訓練的時候會給 Decoder 看正確答案，這個情況稱為 **Teacher Forcing**
- **Teacher Forcing**：使用 ground truth 當作 Decoder 輸入

訓練 Sequence To Sequence model 的 Tips



- 訓練 Sequence To Sequence model 的 Tips

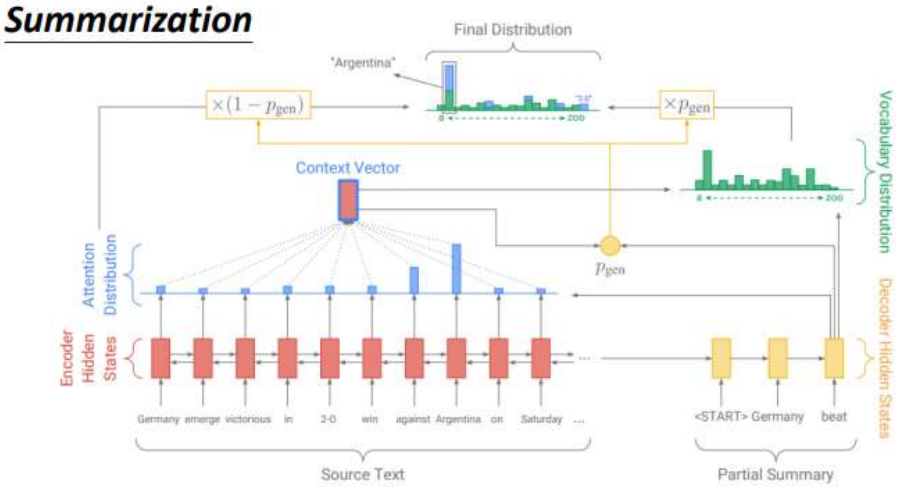
Copy Mechanism



- Chat-bot
  - 複製對於對話來說，是一個需要的技術能力
  - 複製一段它聽不懂的話

Copy Mechanism

https://arxiv.org/abs/1704.04368



- Summarization
  - 訓練一個模型，模型去讀一篇文章，產生文章的摘要

Guided Attention

Guided Attention

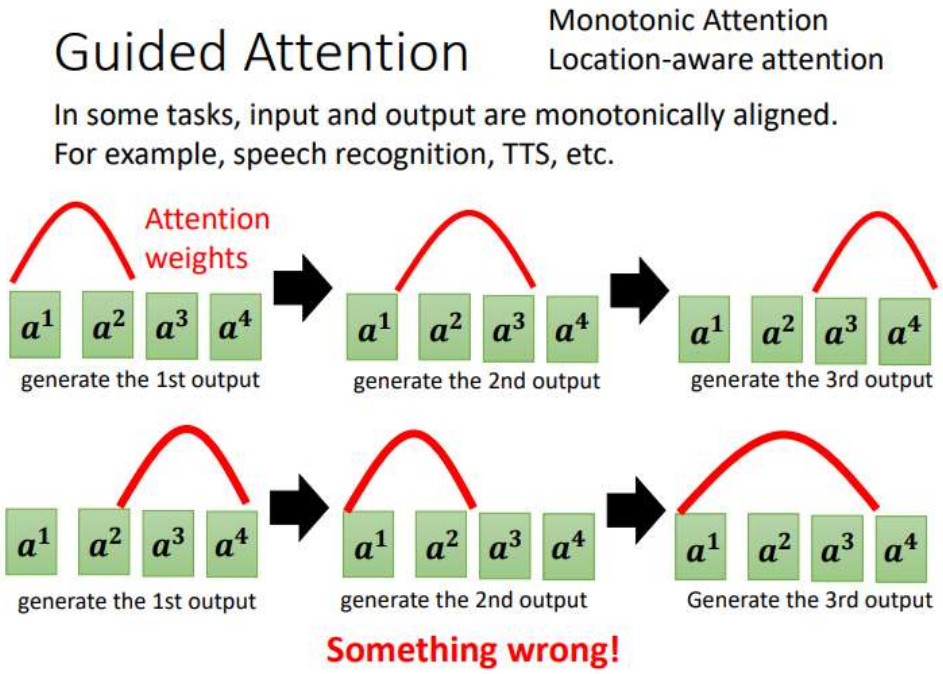
高雄發大財我現在要出征

發財發財發財發財

發財發財發財

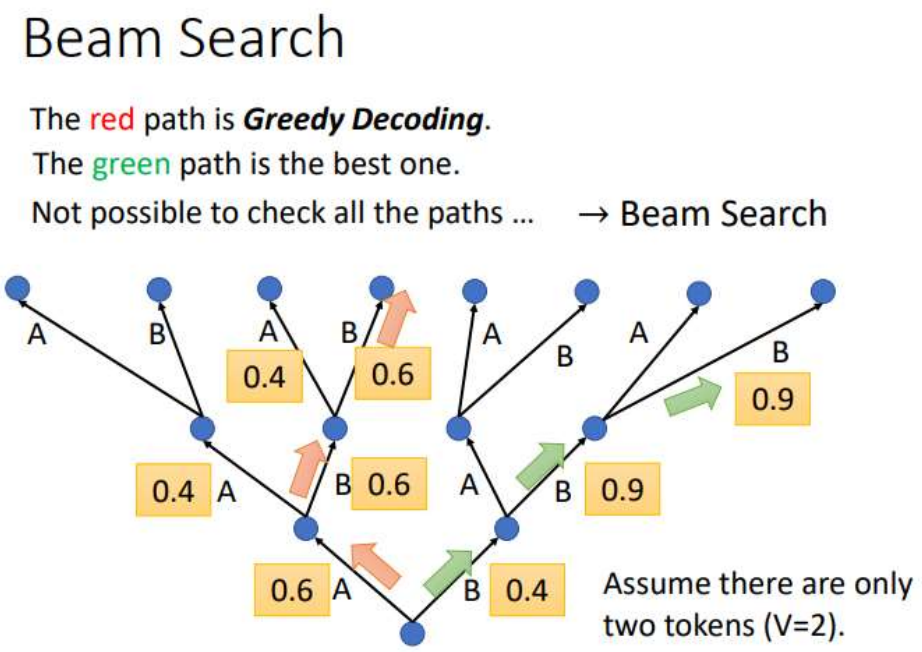
發財發財

發財 (Missing an input character!)



- 要求機器在做 attention 的時候，是有固定的方式
- 對語音合成或語音辨識來說，我們想像中的 attention 都是由左往右
- 紅色曲線代表 attention 的分數，越高就代表 attention 的值越大
- **Guided Attention 做的事情就是，強迫 attention 有一個固定的樣貌**

Beam Search



- 每次找分數最高的那個 token 當作輸入，稱為 **Greedy Decoding**
- Greedy decoding 不一定是最好的
- 綠色的路最後整體比紅色的路好

- **Beam Search**：用比較有效的方法找一個 Approximate，找一個估測的 Solution

Sampling

The Curious Case of Neural Text Degeneration

https://arxiv.org/abs/1904.09751

Context: In a shocking finding, scientist discovered a herd of unicorns living in a remote, previously unexplored valley, in the Andes Mountains. Even more surprising to the researchers was the fact that the unicorns spoke perfect English.

Beam Search, *b*=32:
"The study, published in the Proceedings of the National Academy of Sciences of the United States of America (PNAS), was conducted by researchers from the Universidad Nacional Autónoma de México (UNAM) and the Universidad Nacional Autónoma de México (UNAM/Universidad Nacional Autónoma de México/Universidad Nacional Autónoma de México/Universidad Nacional Autónoma de México/Universidad Nacional Autónoma de ..."

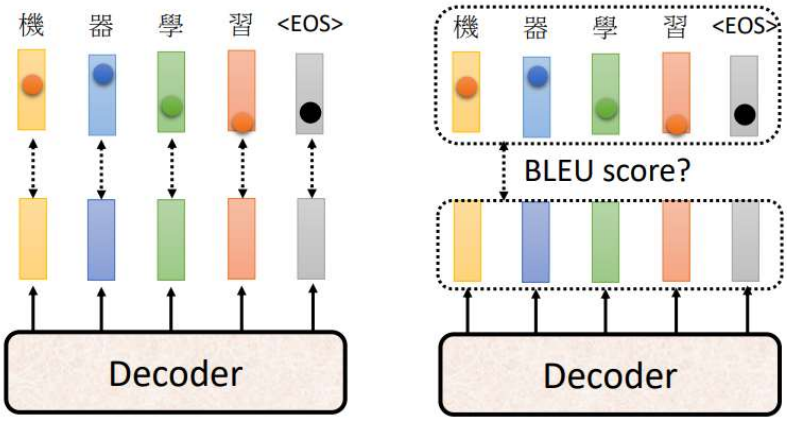
Pure Sampling:
They were cattle called Bolivian Cavalleros; they live in a remote desert uninterrupted by town, and they speak huge, beautiful, paradisiacal Bolivian linguistic thing. They say, 'Lunch, marge.' They don't tell what the lunch is," director Professor Chuperas Omwell told Sky News. "They've only been talking to scientists, like we're being interviewed by TV reporters. We don't even stick around to be interviewed by TV reporters. Maybe that's how they figured out that they're cosplaying as the Bolivian Cavalleros."

Randomness is needed for decoder when generating sequence in some tasks.

Accept that nothing is perfect. True beauty lies in the cracks of imperfection. ☺

- 假設一個任務它的答案非常明確，例如語音辨識，說一句話辨識的結果只有一個可能，沒有模糊地帶，Beam Search 就會比較有幫助
- 需要機器發揮創造力的時候，Beam Search 就會比較沒有幫助

Optimizing Evaluation Metrics?

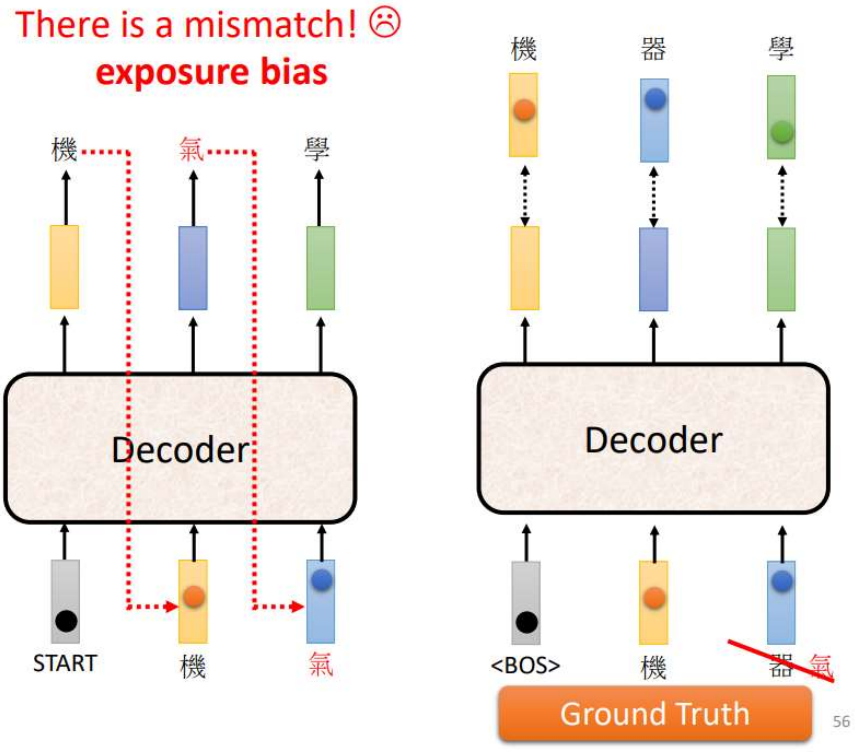


How to do the optimization?

When you don't know how to optimize, just use reinforcement learning (RL)! https://arxiv.org/abs/1511.06732

- BLEU score
  - Decoder 先產生一個完整個句子，再去跟正確的答案一整句做比較，算出 BLEU score
  - 但訓練的時候每一個詞彙是分開考慮，且 minimize 的是 cross entropy
  - minimize cross entropy 可以 maximize BLEU score 嗎？
    - 不一定，有點關聯但又沒有直接的關聯
- 在作業裡面，不是拿 cross entropy 來挑最好的 model
- 而是挑 BLEU Score 最高的那個 model
- 所以訓練的時候是看 cross entropy，但實際上在作業真正的評估的時候看的是 BLEU Score

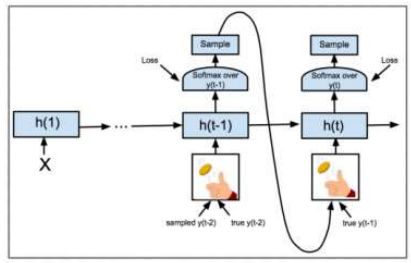
- **遇到在 Optimization 無法解決的問題，用 RL 硬 train 一發就對了**
- 遇到無法 Optimize 的 Loss Function 把它當作是 RL 的 Reward，把 Decoder 當作是 Agent Reinforcement Learning 問題硬做

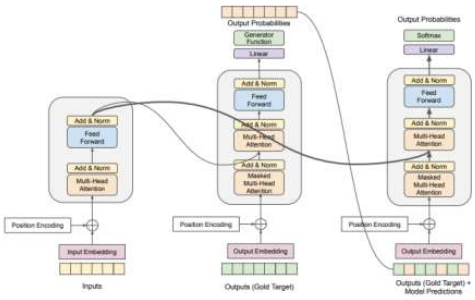


- 訓練跟測試是不一致的
- 測試的時候，Decoder 看到的是自己的輸出，所以測試的時候 Decoder 會看到一些錯誤的東西
- 但在訓練的時候，Decoder 看到的是完全正確的
- 不一致的現在稱為 **Exposure bias**

Scheduled Sampling

- Original Scheduled Sampling
  https://arxiv.org/abs/1506.03099
- Scheduled Sampling for Transformer
  https://arxiv.org/abs/1906.07651
- Parallel Scheduled Sampling
  https://arxiv.org/abs/1906.04331

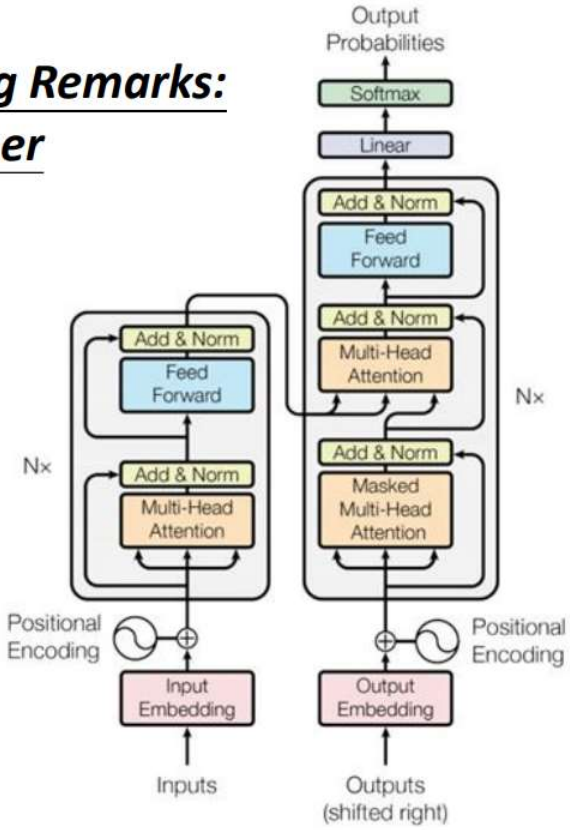




57

- 解決的辦法是，在訓練的時候偶爾給 Decoder 一些錯的東西，反而會學得更好，這招稱為 **Scheduled Sampling**
- Scheduled Sampling 會傷害到 transformer 的平行化的能力

Concluding Remarks: Transformer



tags: **2022 李宏毅_機器學習**