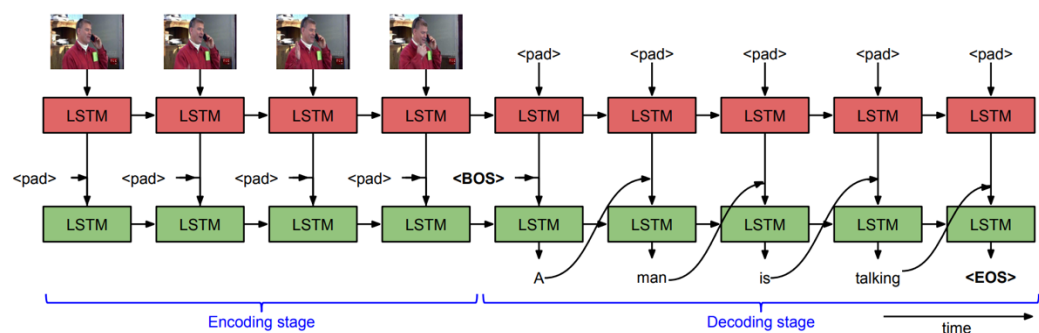


1. Model description

- 文字處理
 - 濾掉標點符號、轉小寫、並以空白做 split 取出 word tokens
 - 將 captions 轉成 token sequences，並在前後加上起始結束的 tag
 - 設定 sequence 最大的長度 max_sent_len，不足的部分做 padding
Ex: [<s>, a, man, is, walking, </s>, <pad>, <pad>, <pad>, <pad>]
 - 最後透過建立好的 dictionary，將 token sequence 轉成 index sequence 後再餵入 model
- 由於一個 video 有多個 caption，我的處理方式是每次先選好 video 後，再隨機選擇其中的一個 caption 當 label，所以每次 epoch 跑的數量都會是一樣 1450 筆 data，但 video features 會在不同 epoch 中對應到不同 label (caption)。
- S2VT Model

為 Conditional Generation

疊兩層 RNN (GRU/LSTM)，第一層作為 encoder，會把 pre-trained CNN Outputs 拿進來做 training 取出影像的代表特徵；第二層作為 decoder，會將影片的的代表特徵和前一次的 decoder 輸出一起放入 input (condition) 做 training 得到輸出結果

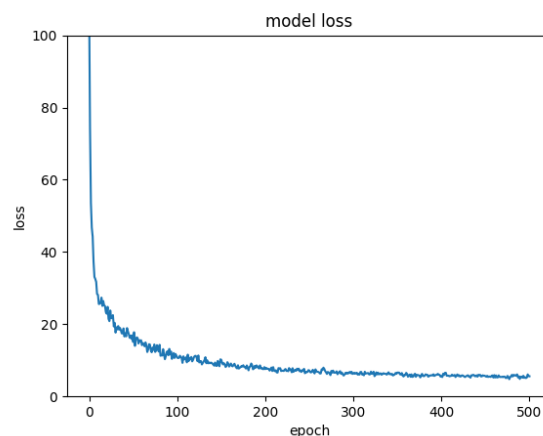


Sequence to Sequence – Video to Text:

<http://www.cs.utexas.edu/users/ml/papers/venugopalan.iccv15.pdf>

其中「前一次的 decoder 輸出」實際上在做 training 時是直接 reference 原本 label captions 正確的前一個字，而在 testing 時才是真正的去拿 model 前一個的輸出 (exposure bias)，因為如果不這麼做的話 model 會很難 train 起來，(因為 decoder 會一步錯，步步錯)。

- 參數設置：
 epochs: 500
 batch size: 128
 layer_dim: 768 (for word_embedding, encoding rnn, decoding rnn)
 learning rate: 0.001
 max_sent_len: 15
 loss: cross-entropy (透過 masking 來濾掉 padding 的部分)
 optimizer: adam
- 輸出處理：
 濾掉起始 tag 和 pad tag，並將輸出截斷至結尾 tag 第一次出現之時
- 結果：
 Loss 如下圖來到了 5 左右，Average bleu score 為 0.281514



2. Attention mechanism

為 Dynamic Conditional Generation

在 encoder 與 decoder 之間加入 attention，讓 decoder 的 input 會專注在某部分上 (context 上下文的概念)。

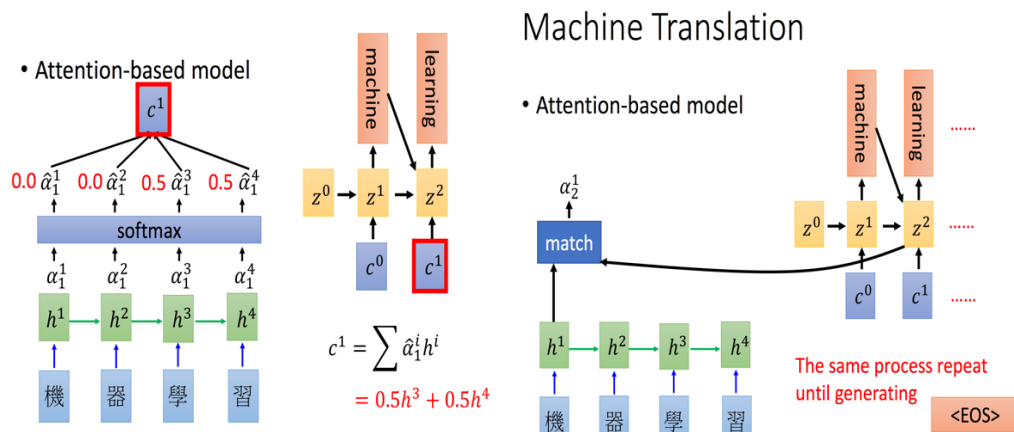
其作法是透過給 encoder 每個 timestep 輸出一個權重做 weight and sum 再餵入 decoder，而這組權重是透過一個 match function 所產生，這 match_function 會吃 [encoder layer 此次輸出] 和 [decoder layer 前個輸出]

得到一組權重 α

而我的 match function 是學出來的： $\alpha = \text{softmax}(h^T W z)$

(W 是學出來的，為 neural network 的一部分)

因此我的 decoder input 會再多出 context feature : $c = \sum \alpha h$



上課 slide

3. How to improve your performance

● Masking

masking 指的意思是說我們在餵 feature 進 model 時會另外有一個對應 feature 的 binary array，而這個 array 對應到 padding 時值為 0，非 padding 時值為 1

Why masking? => 因為 padding 值非常好預測到，而這會導致我們在計算 loss 時產生一種 model loss 已經很低的錯覺，也就是說 padding loss 其實不應該被算入 model loss 中，而透過 masking 我們就可以將 padding loss 濾掉

● Scheduled Sampling

誠如 1 所提到的 exposure bias 問題，為了解決此問題我們可以嘗試 Scheduled Sampling，其作法是在一開始 training 時我們都採用 from reference 來取前一個字，而漸漸的（我使用 linear decay）會有一個比例開始慢慢轉換成採用 from model 來取前一個字。

Why Scheduled Sampling? => 解決 exposure bias

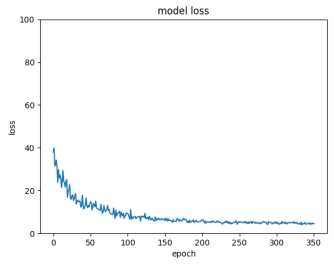
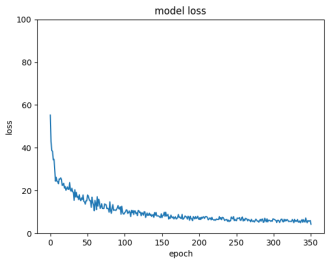
*這一部分我來不及放進 best model 中，不過有 implement 在 train.py/test.py/hw2_model.py 中，照理來說如果放進 best model 應該有助於 performance

4. Experimental results and settings

實驗基本參數設置如下：

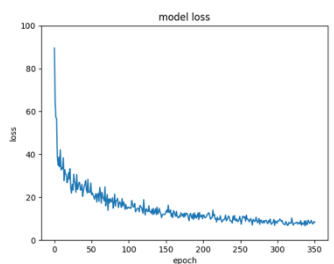
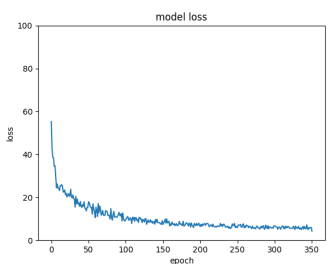
- batch size: 48
- layer_dim: 512 (for word_embedding, encoding rnn, decoding rnn)
- epochs: 350
- GRU
- Not Attend
- Decoder 前一個 step 的輸入為 Reference

● “LSTM” vs “GRU”

| | LSTM | GRU |
|--------------------|--|--|
| Epoch Time | 15.0075 | 14.5248 |
| Loss |  |  |
| Final Loss | 4.62784 | 5.2831 |
| Average Bleu Score | 0.2770 | 0.2680 |

LSTM 在 loss 上 和 Bleu Score 上略優於 GRU，速度上則是略輸

● “Attend” vs “Not Attend”

| | Attend | Not Attend |
|--------------------|---|---|
| Epoch Time | 42.8239 | 14.5248 |
| Loss |  |  |
| Final Loss | 8.39939 | 5.2831 |
| Average Bleu Score | 0.2711 | 0.2680 |

Attend 的時間會比原本久很多，而 Bleu Score 似乎沒有多大的進步