ADLxMLDS HW2 B03705006 侯舜元

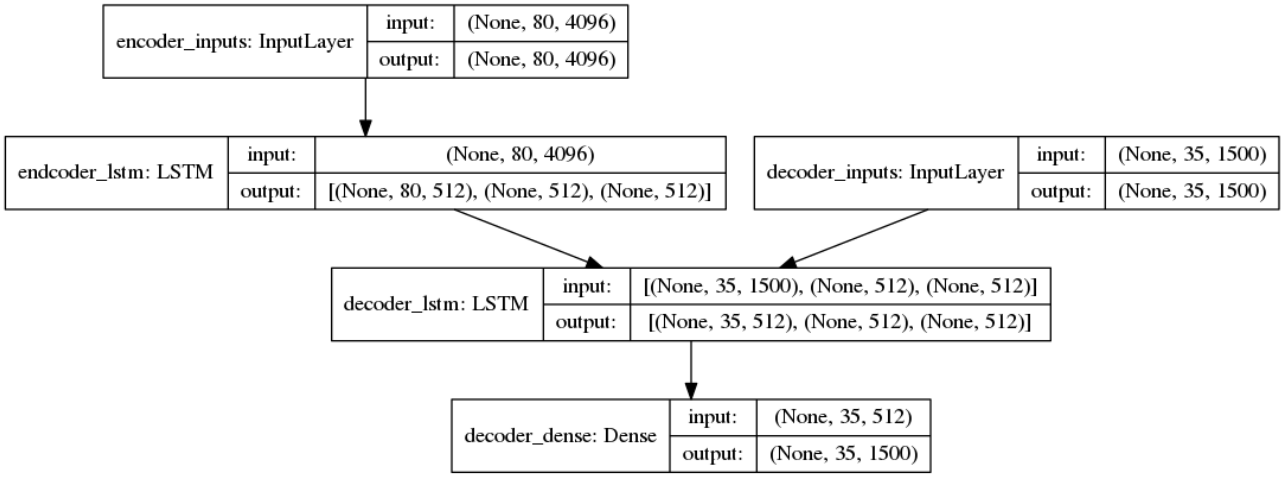
Model description (2%)

資料前處理：

影片的 feature 未做處理，將影片的 label (captions)做 one-hot coding encoder，max words 取 1024 ~ 2048 不等。在 training 時每部影片會隨機取 2~5 個 capture 去 train。

Model 架構圖：

Training Model

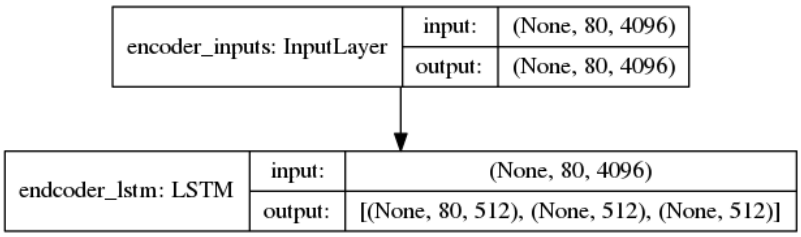


Training Model：

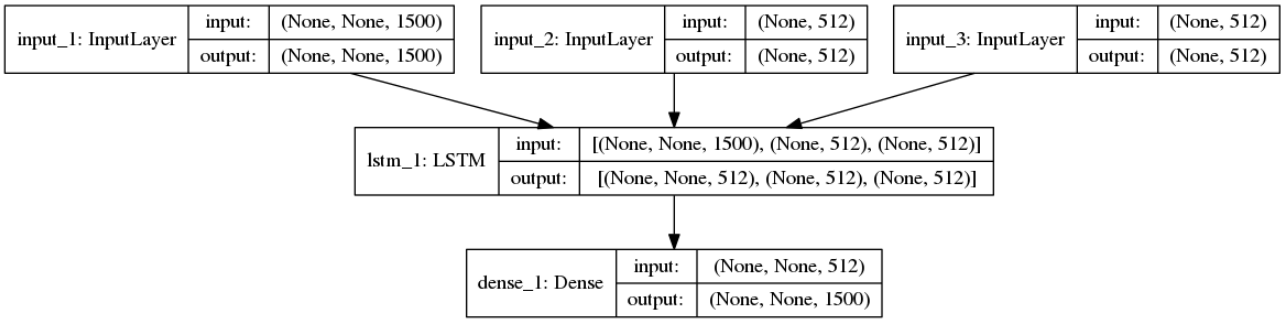
將影片的 feature 丟到 encoder LSTM，並將 LSTM 的 return_state 設為 True，output 回傳 LSTM 的 state values (最後一個 time step）。Decoder inputs 為影片 capture 的前 [0 : MAX_SEQ_LEN-1]個字元， Decoder targets 則為[1: MAX_SEQ_LEN]。Encoder 階段結束後將 state values 丟給 decoder LSTM 當作其 initial states，就可以接續著 decode 出結果了。

Inference Model

Encoder



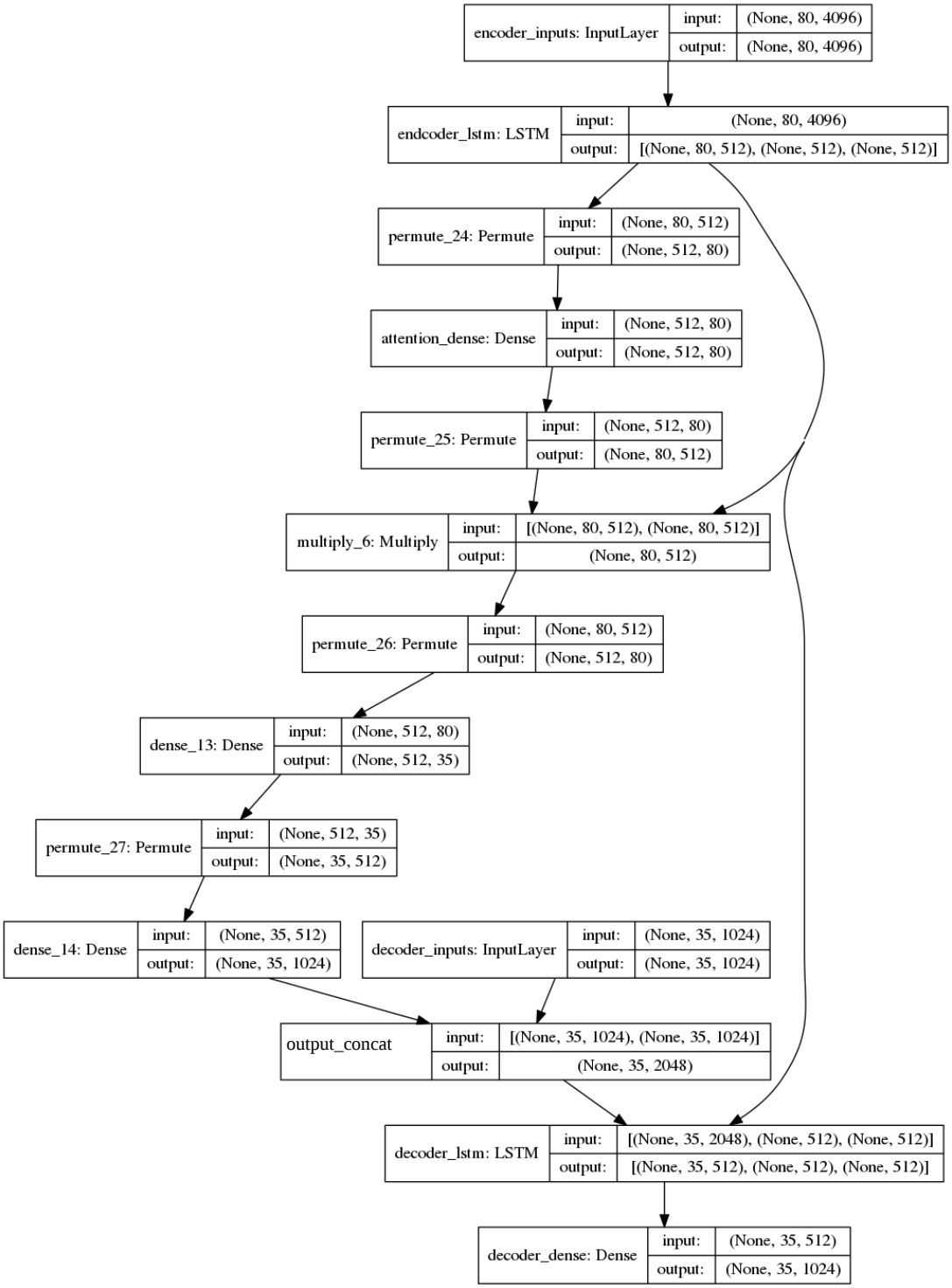
Decoder



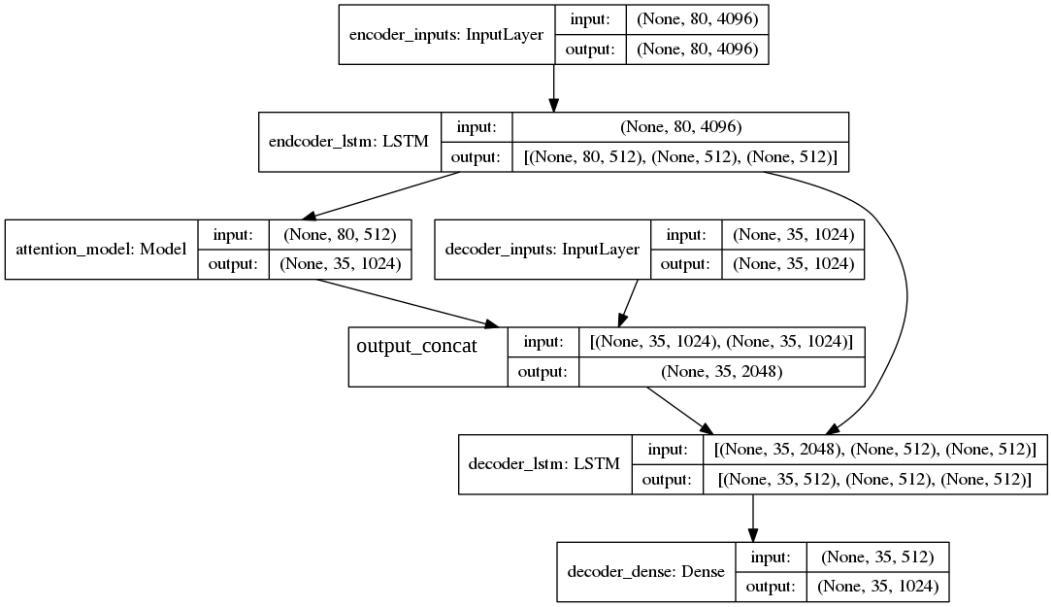
Inference 階段將一影片的 feature 丟給 inference encoder 回傳得到 state values 並傳給 inference decoder 當作其 input。而在 decode 句子時會以一 <bos> 開頭的字當作其 decoder input，去 predict 下一個字，再以 predict 結果和新的 state 去 predict 下一個字...
重複以上步驟直到達到中止條件(句子長度上限或是遇到<eos>)為止。
此 model 得到最佳的結果（分數）如下
Originally, average bleu score is 0.29555270794864785
By another method, average bleu score is 0.690094226941438

Attention Mechanism（2%)

Attention
Model



簡化好讀版本：



How do you implement attention mechanism? (1%)

有嘗試許多不同的 attention mechanism，由於受限於 Keras 的許多限制（例如無法回傳 LSTM 每個 time step 的 states 等），故我決定將 encoder model output 的 sequence 先通過 permute 和一個 softmax dense layer 做 attention（決定要關注於哪個 frame），維度轉換後再和 decoder input concat 起來一起丟進 decoder 裡 train。

Compare and analyze the results of models with and without attention mechanism. (1%)

最好的 attention mechanism 得到的結果在 val_loss 有比原先的無 attention 有更優秀的表現，但是其 bleu 分數較低

Originally, average bleu score is 0.282988050956

By another method, average bleu score is 0.648919382927

在 capture 的文意表現上，看看以下隨機取的三個例子做比較

有 attention

1 a man is adding ingredients in a bowl

2a cat is eating a

3 a car is driving down a motorcycle

無 attention

1 a man is putting ingredients in a bowl

2 a person is holding a dog

3 a man is driving down a road

第一個句子兩者相似且正確，第二句子正解為一不知名生物在人的身上聞塑膠盒，無 attention 表現較佳。而第三個句子的正解為一個黑條紋車子在街道上行駛，有 attention 在預測街道時可能又再度著重於該交通工具的 feature 反而預測出了 motorcycle，而無 attention 版本的較為正確。整體來說無 attention 版本意外的較好，但也可能是 test data 的巧合。

How to improve your performance (1%)

1. 將字典的大小（vocab_size）提昇，原本從 1024 開始 training，後來提升到 1500 和 2048。1500 的結果綜合來看（最高 blue score 和最佳的文意）最佳。
2. 使用 beam search，每次取最佳的兩個繼續預測下一個字，成功提高了句子通順度的表現

有 beam search	無 beam search
a woman is riding a ball	a woman is riding a ball of a
a man eats pasta	a man is pasta from a plastic container into a plastic container into a plastic container
a person is cutting a	a woman is cutting a of of a of and and and it in the
a girl is singing	a woman is talking on a couch
a dog is walking in a grass	a man is walking down a grass

由上表格可見，加上 beam search 可以防止重複預測許多不通順的介係詞，以及更完整正確的闡述影片的內容。
值得一提的是，加上 beam search 後反而使最 bleu score 下降，下降到 0.26753。

Experimental results and settings (1%)

1. 如何迅速的 improve bleu score:
(1) 提昇 vocab_size (1024 → 1500 , bleu 0.254 → 0.28)
(2) 在 decoding state 結束後，人工撰寫條件篩選、刪除連續重複的字串以及介係詞。可以在不更改 model 的情況下，迅速的將 bleu score 提昇 0.1~0.2 分，且文意會看起來較通順

2. Attention Mechanism 的選擇
第二頁圖中的 attention 架構是最成功的方法，另外以下有幾個比較失敗
(1)將 attention model 的 output 跟 decoder lstm 的 output concat 起來在丟到一個 softmax dense 裡 train 最後 output => val_loss 下降的非常緩慢，結果也較差
(2)將 attention model 的 output 跟 decoder lstm 的 output Multiple 相乘，再丟到最後 softmax dense => val_loss 一直降不下來，結果顯著較差

3. Parameter tuning
LSTM unit 數量的選擇從 320 到 512 會使 performance 上升，但從 512 到 1024 就沒顯著的效果。
Optimizer 使用 adam，learning rate 從預設的 0.001 往下調低到 0.0007～0.0005 會有比較好的結果。