MLDS HW 2-1 Video Caption Generation

1. Model description:

Data Preprocess:

* + 1. Build word dictionary (including word2ix, ix2word and bias\_init\_vector) :

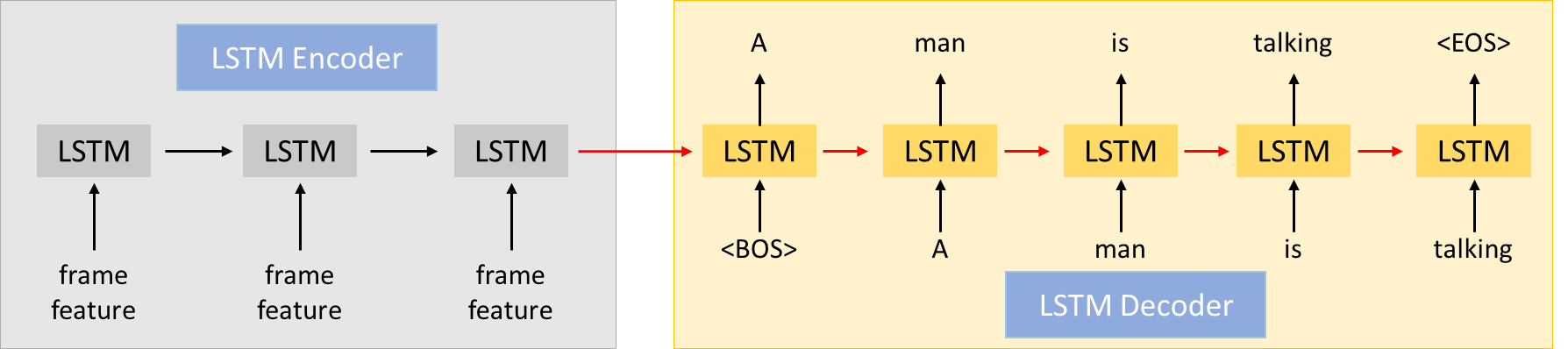
首先從training data 及 testing data的caption中收集詞彙， 共有6347個詞彙，並依照所設的詞彙出現次數threshold(2)，將總詞彙壓縮為3870個；根據所收集的詞彙建立word2ix, ix2word 字典，以及bias\_init\_vector以提供模型訓練及預測使用。

* + 1. Training data preprocess:

training data 中的每一個video長度約為5~20s, 利用CNN對每個video抽取共80 個4096維的frame features. (助教提供)

將training data 中的每個caption 開頭加上<bos>，結尾加上<eos>，不足16字的caption做padding (加<pad>) 至 16字，並捨棄大於16字的caption； 最後將每句caption中的每個詞彙根據字典word2ix轉成相對應的ID vector。

由於一個video會有多個caption，此model會對於每個video 隨機選擇其中一個caption作為label，對應相應的frame features，生成訓練資料。

Model Structure

Model 架構為encoder-decoder 架構，由兩層size為256維的LSTM所組成並共享權重；一個單層的LSTM作為encoder，encode pretrained的CNN frame features，另一個LSTM作為decoder，吃入video embedding features及前一次decoder 的輸出作為input，產生該video相對應的caption。Loss function 使用cross entropy，而back propagation 則選用AdamOptimizer來更新模型的參數。

Inference:

將input test video 中80個frame的features輸入encoder中，接著decoder讀到<bos>時便開始運作: 在time step t 輸出經softmax後機率最大(argmax)的詞彙；並將此詞彙作為time step t+1 的decoder input ，持續重複此步驟直到deocder輸出<eos>後結束，最終產生此影片的caption。

Model details

Parameters

Input dimension = 4096;

LSTM hidden dimension = 256;

#video LSTM steps = 80;

#caption LSTM steps = 16;

Optimizer = Adam optimizer

Learning rate = 0.001

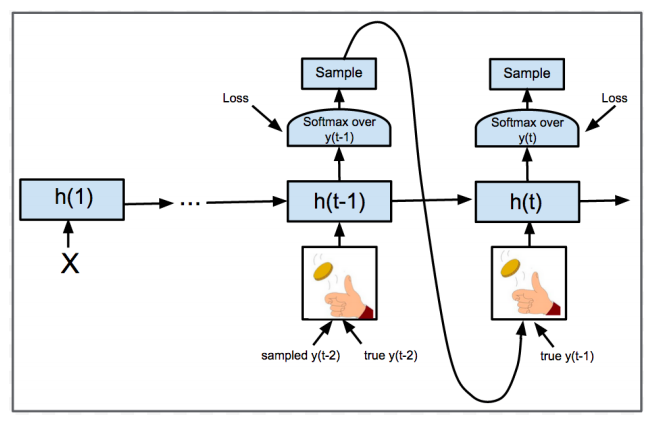
Batch size = 32

Epoch = 205

1. How to improve your performance:

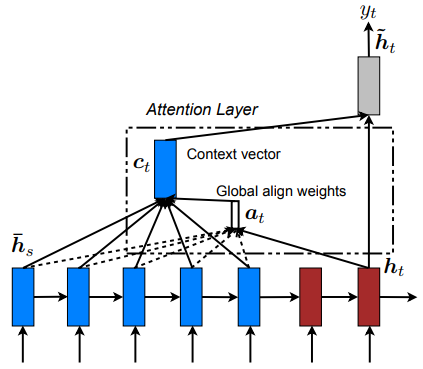
Write down the method that makes you outstanding

* + 1. Schedule sampling



model在training時把ground truth 作為RNN的輸入(teacher forcing)，而在inference 時則將RNN上時刻的輸出作為下時刻的輸入(last time step’s output)；為了模擬model在inference的真實情況；可以在開始訓練時先以ground-truth來做輸入；隨著迭代次數的增加，model的參數逐漸收斂，可以逐漸以上時刻的輸出作為下時刻的輸入；而具體的實現作法是以丟擲硬幣來決定。而這次的實作是利用tensorflow 中的ScheduledEmbeddingTrainingHelper來實作，根據廣義的bernoulli distribution決定輸入的型態。

* + 1. Attention mechanism



Attention mechanism的實作是採用tensorflow中的LoungAttention layer，將所有的encoder output 作為memory傳進attention layer, 每個decoder output會經由context base function對每一個encoder output計算alignment score； Normalize所有scores 後，將每個encoder output 與相對的score相乘後加總(weighted sum)以得到context vector. 最後同時考慮context vector與decoder output以得到最後的結果。

Why do you use it

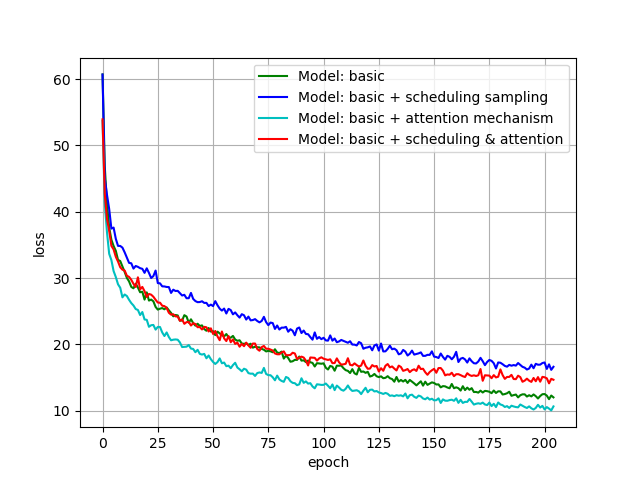
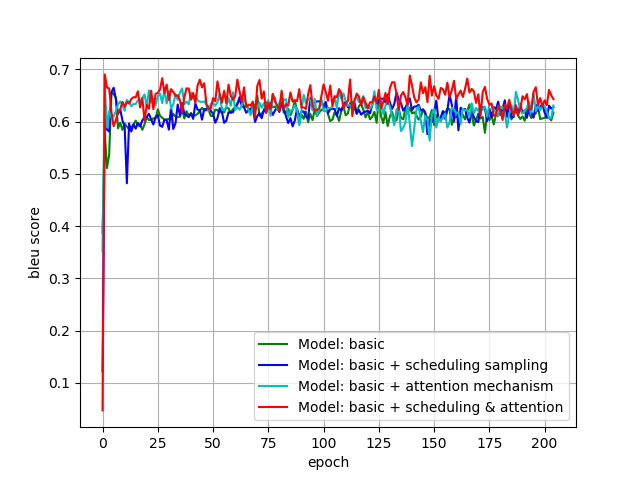
* + 1. Schedule sampling

為解決exposure bias 的問題，即RNN model 在training 時接受ground truth input，而在inference時卻是接受last time step’s output，若在t-1時刻的到較糟糕的輸出結果，由於model無法得知真實的輸出，model只能根據此糟糕的結果去預測下一時刻t的輸出，造成一步錯，步步錯的情形。

* + 1. Attention mechanism

在Encoder-Decoder的結構中，輸入序列無論長短都會被encode成一固定長度的向量，decode時則受限於該固定長度的向量；當輸入序列較長時難以保留全部的資訊，使model的性能變得很差。Attention mechanism 可以幫助model對輸入的每個部份賦予不同的權重，抽取出更加關鍵且重要的資訊，使模型能做出更加準確的判斷。

Analysis and compare your model without the method



根據上圖可知model在epoch > 175後BLEU score 變化不大

* + 1. Schedule sampling

從上圖觀察得知，在model加入schedule sampling 後的loss上升，且BLEU score 的差異不大；推測原因可能為schedule sampling 需在epoch數大一點才能看出效果，training epoch數太小反而會適得其反。

* + 1. Attention mechanism

從上兩張圖觀察可知，在加入attention mechanism後確實能有效降低loss，而在BLEU score 的表現上，在前幾個epochs中 attention-based model確實有較高的BLEU score，而到訓練後期，basic model 和 attention-based model 在validation set上的表現雖差異不大，但仍可看見確實有改善。

* + 1. Schedule sampling + Attention mechanism

從上圖中可看出，當basic model 同時加入schedule sampling 和attention mechanism後，雖然loss 比只加入attention mechanism 的model來得高(推測可能是受到schedule sampling 的影響，在epoch數小時成效不佳)，但是BLEU score有明顯增加，大多數都在0.6以上；可見這兩個技巧的確對model的訓練確實有幫助。

1. Experimental results and settings

實驗中發現，epoch在大於175後BLEU score就已變化不大(收斂)，再繼續train會發生overfit而導致BLEU score下降。最終使用以下的設定，以得到最佳的BLEU score

Input dimension = 4096;

Word threshold = 2 (3870 words);

Luong Attention;

Schedule sampling rate = 0.2;

LSTM hidden dimension = 256;

#video LSTM steps = 80;

#caption LSTM steps = 16;

Optimizer = Adam optimizer;

Loss = sparse\_softmax\_cross\_entropy\_with\_logits;

Learning rate = 0.001;

Batch size = 32;

Epoch = 205;

Max BLEU score = 0.68

1. 分工:

|  |  |  |
| --- | --- | --- |
| r06944046 | r06922141 | b03901156 |
| 33.3% | 33.3% | 33.3% |