MLDS HW 2-2 Chinese Chatbot

1. Model description

Data Preprocess:

* + 1. Build word dictionary (including word2ix, ix2word) :

首先從clr\_conversation.txt 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，以提供模型訓練及預測使用。

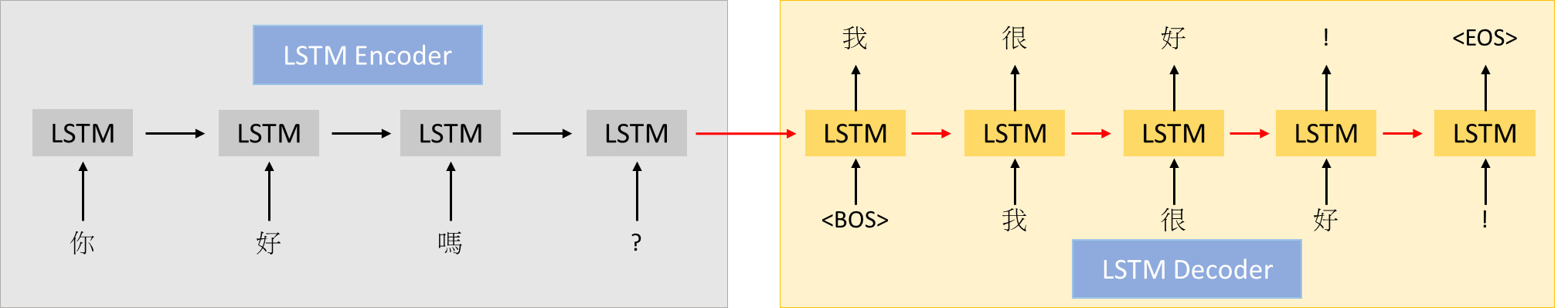
* + 1. Training data preprocess:

clr\_conversation.txt中每一行為一句話，對話和對話之間以+++$+++做分隔。首先利用+++$+++收集每段對話conv到對話清單convs中，再從每段對話conv中，根據前後兩個句子sent製作 [question, answer] pair 的清單，形成training samples。

將training samples 分成多個大小為batch\_size的batch；而在每個batch裡

將每個question做padding 至該batch中question句子的最大長度並reverse，形成encoder input ；在每個answer前加上<eos>後，再做padding至batch中answer句子的最大長度，形成decoder input； 最後將每句話中的每個詞彙根據字典word2ix轉成相對應的ID vector，生成訓練資料。

Model Structure



Model 架構為encoder-decoder 架構，由兩層size為512維的LSTM所組成並共享權重；一個單層的LSTM作為encoder，encode 在data preprocess步驟中所產生的encoder input (question)，另一個LSTM則作為decoder，decoder 的target為decoder input shift by 1; 吃入encoder output 及前一次decoder 的輸出(answer)作為input。Loss function 使用sequence\_loss計算，而back propagation 則選用AdamOptimizer來更新模型的參數。

Inference:

讀取input test data，將每一句話根據word2ix轉成相對應的ID vector，接著對每個句子做padding至test data 中的最大長度，餵進seq2seq的模型中；當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(greedy decoding)；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，最終產生與該test data相對應的答句 。另外，除了greedy decoding 外，也有使用beam search來實現inference時decode的model。

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

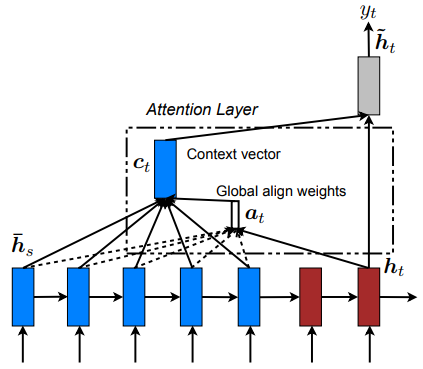
Batch size = 32

Epoch = 205

1. How to improve your performance

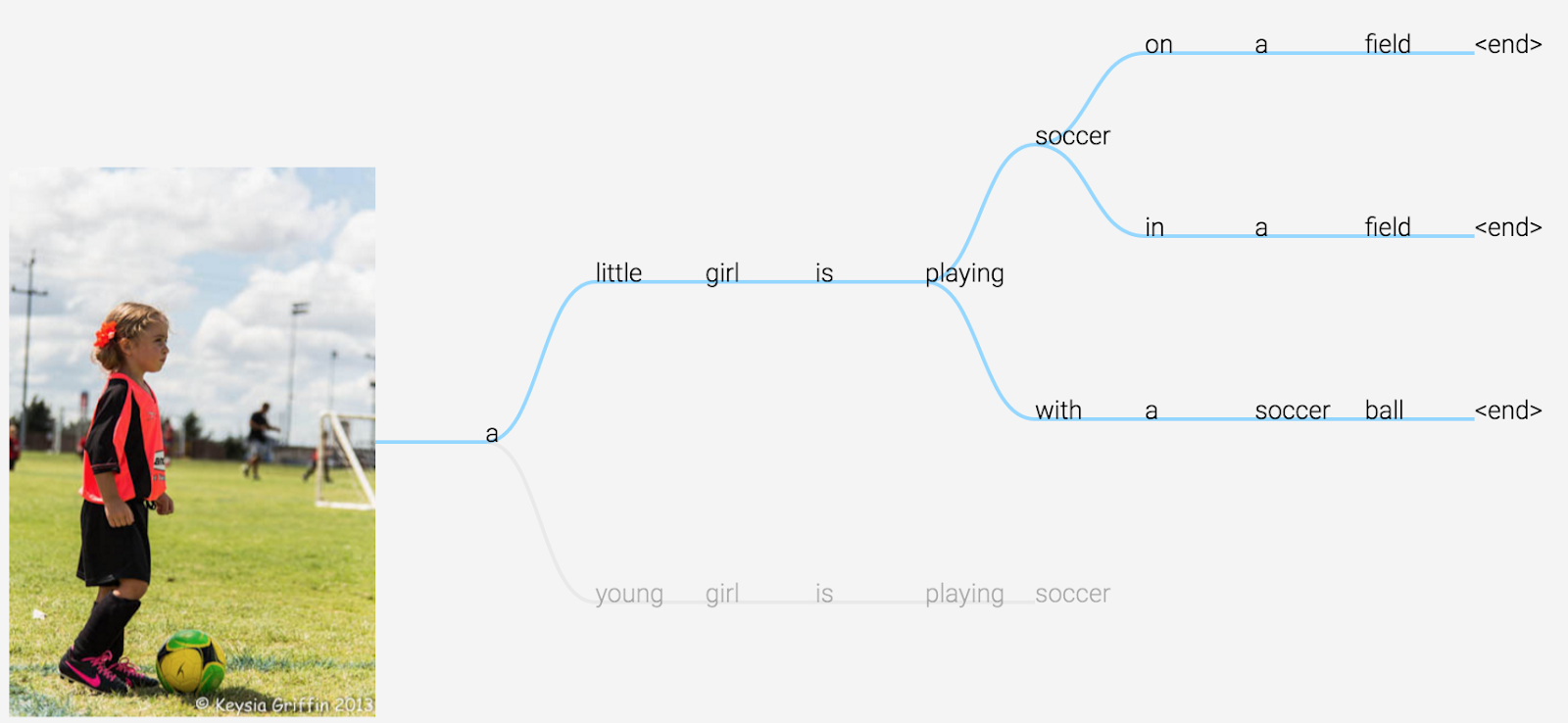
Write down the method that makes you outstanding

* + 1. 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。

* + 1. Beam Search



Beam search 的實作是採用tensorflow 中的BeamSearchDecoder，首先對encoder output進行tile\_batch，也就是複製beam\_size份；接著在decode的過程中每次皆保留概率前beam\_size高的詞彙組合，捨棄其餘組合，重複此步驟至遇見<EOS>結束為止。最終將這些句子依照得分排序，得分最高者作為最終輸出。

Why do you use it

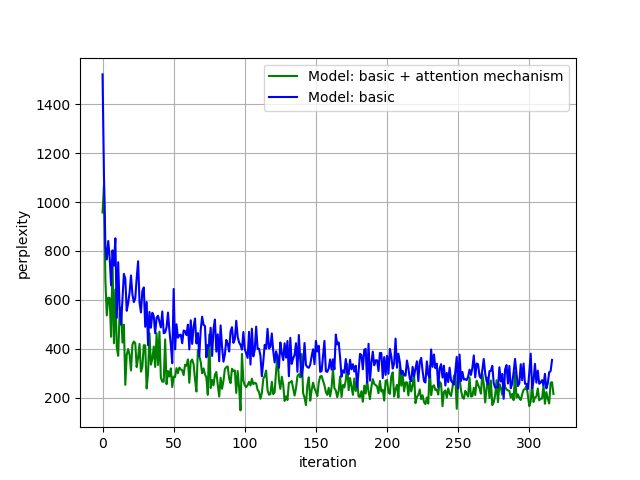
* + 1. Attention mechanism

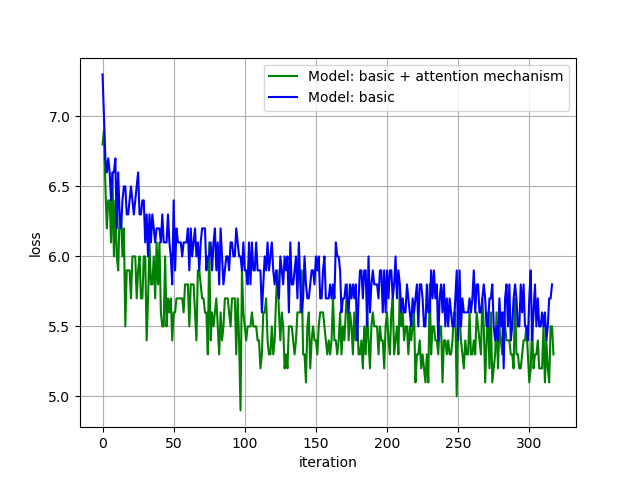
在Encoder-Decoder的結構中，輸入序列無論長短都會被encode成一固定長度的向量，decode時則受限於該固定長度的向量；當輸入序列較長時難以保留全部的資訊，使model的性能變得很差。Attention mechanism 可以幫助model對輸入的每個部份賦予不同的權重，抽取出更加關鍵且重要的資訊，使模型能做出更加準確的判斷。

* + 1. Beam Search

當model在inference 時若採用greedy search，則decoder取概率最高(argmax)的詞彙作為當前輸出，並將此詞彙作為預測下個詞彙的decoder input，重複此步驟直到model輸出<EOS>結束。很顯然在每一步都取概率最大的詞並不能保證最後生成的句子概率是最大的，因此可以利用beam search來做動態規劃、全局的decode，每次皆留下整體概率top-N的詞彙組合，捨棄概率低的結果；如此比greedy search更能獲得接近最佳解的解。

Analysis and compare your model without the method (比較前325個iteration)





* + 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perplexity的結果上，也可發現加入attention mechanism後的model其perplexity確實較低，顯示attention mechanism所抽取的資訊確實能幫助model的training。

* + 1. Beam search

由下列例子中可看出，經由Beam search後所得的Output sentence更合乎語意，且

也更像對話

Example case:

|  |  |  |
| --- | --- | --- |
|  | Input Sentence | Output Sentence |
| Directly Inference | 真是個乖孩子 | 我不懂，你 |
| Inference by Beam Search | 真是個乖孩子 | 我不知道你的意思 |

1. Experimental results and settings

實驗中發現，loss 會大約卡在4.7左右無法降下，增加layer的深度反而更糟；也有試過SGD optimizer ，也是會存在一樣的情形。最終經過tradeoff後使用以下的設定，以得到最佳的Chatbot

hidden\_dim = 512

num\_of\_layers = 2

embedding\_size = 128

learning\_rate = 0.0005

batch\_size = 128

num\_of\_epochs = 50

steps\_per\_checkpoint = 100

1. 分工

由於另兩位同學決定退選，故此次作業皆由一人b03901156完成