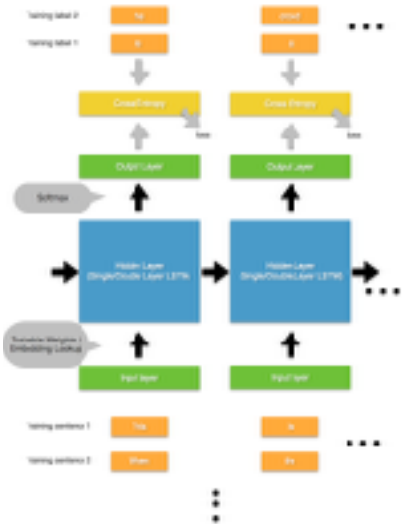
Hw1 report — Language Model

R05943135 杯屎賴—
R05943135江承恩 R05943011 沈恩禾
R05921016傅鈞笙 R05942072 吳昭霆

環境

CPU	GPU	Memory	OS	Libraries
Intel(R) Xeon(R) CPU E5-2630 v2 @ 2.60GHz	GTX 980	HYNIX HMT42GR7BFR4C -RD MEMORY 16GB * 8	Ubuntu 15.04 Mint 17.1 Rebecca	Tensorflow 1.0

Model描述



- **LSTM cell**: LSTM組成單元，使用tf.contrib.rnn.BasicLSTMCell()。
- **unit_num**: LSTM 的 input size 與 output size。
- **num_layers**: LSTM縱向堆疊的深度。
- **Cost function:** Cross entropy 之總和。
- **Optimizer:** GradientDescentOptimizer。
- **Input:** 單詞ID，size = 1。
- **Output:** 單詞機率，size = num_vocabulary。

Training Data處理參數

- **num_vocabulary**：以詞頻擷取(num_vocabulary數-1)之詞彙，超出範圍以<unk>取代。
- **train_num_steps**: Training sentence 長度。
- **test_num_steps**: 克漏字選項為中央，往左右各擷取(**test_num_steps-1)/2** 長度。
- **<start> <end> Padding**: 在句首句尾加上<start> <end> token。
- **<end> Padding**: 句首句尾加上<end> token直到句子達到train_num_steps大小。
- **Sentence concatenation**: Training data 是否有跨句的例子。

Training參數

- **Training batch size**: 嘗試20 , 64 , 128。
- **learning rate**: learning rate起始值，嘗試 1.0 , 0.8 ,0.001。
- **learning rate decay**: 每過一個epoch，learning rate乘上的倍數，嘗試0.8 , 0.5, 0.4。
- **output dropout probability**: 嘗試1.0與0.65。
- **initial scale:** weights初始值絕對值範圍，使用0.05。

實驗

	num layers	unit num	Train num step	padding	concatenate	Test perplexity	Accuracy (public/private)	epoch	Batch size	Test step
A	2	256	5	both	yes	159	0.311/0.375	6	128	5
B	2	650	20	both	yes	136	0.28/	2	20	5
C	1	256	20	both	yes	184	0.313/0.365	10	64/128	5
D	1	168	20	both	yes	192	0.315/0.338	10	128	5
E	1	168	20	both	yes	192	0.328/0.363	10	128	5

- 在perplexity停滯時皆手動調整learning rate到0.001繼續。
- Vocabulary 10000達到之正確率為20%上下，其後僅使用12000。
- 為加速training，當num_units大於256時output dropout使用0.65，小於256時使用1。
- 較好的結果中，valid perplexity和train perplexity皆會較為接近，如A train/valid為59/68，B train/valid為83/89。

嘗試改進方法

- **字詞處理：**
  把字詞全部換成小寫，並且把句子中的標點符號刪除。

- **<start> <end> padding:**
  在句首跟句尾分別各加上兩個 <start> <start>，<end> <end> ，每一句隨機取連續的五個字來當作 training data 。這麼做的原因是因為我們把 test 的克漏字選項放在中間，一開始是選擇前二後二來算機率，所以把 training data 做一樣的處理，加入 <start> ，<end> 則是希望能 train 到原本句首跟句尾的資料（ex: <start><start> 句首 … … ; … … 句尾 <end> <end>） 。

- **padding:**
  將原本的資料一句一句排下來，然後統一一個 size，作為最後要留下的字數。超過size的句子刪掉後面的字，不足的則是在句尾加上 <end> 以補齊。更之後再試著將字數過少的句子刪掉，避免做padding時 <end> 對於句子的重要性高過原本的資料。一開始直接將 size 設為20 ，可以發現 train 的 perplexity 可以降到更低 ，但是 test data 的 perplexity 反而增加到很高，並且沒有隨著 epoch 下降，推測原因是因為 <end> 太多導致對於 model 來說，有很高機會會判斷成 <end>。 因此後來我們根據 train data 的長度來調整不同的 size 以及決定要捨棄的過短的句子。

- **tokenizer:**
  首先，我們使用nltk 的 sentence 及 word tokenizer, 由於它們有用到pre-trained的models因此能將data parse得更好; 其次，使用regular expression的tokenizer將一句話的標點符號eliminate; 最後，我們用pos_tagged將一些可能會造成model perplexity過高的詞(例如專有名詞)，替換成相對應的token(NNP)放到sentence下去train, 確實在validation汗testing的perplexity 都有大幅降低。

- **Number of layers:**

一開始建立兩層的layer ，後來發現有點難train，perplexity降不下來，後來改為一層。

- **hidden_layer_size:**

嘗試不同的數字後發現在128~256 之間結果較佳，最後決定使用 168。

- **learning rate, learning rate decay:**

一開始將 learning rate 設為 1 ，對於一開始要降低 perplexity 有不錯的效果，但過了幾個 epoch 之後 perplexity 容易卡在一個範圍內浮動 ，後來加入了 decay rate ，每過一個 epoch 會乘上 0.5 (或其他數字) ，發現在過了幾個 epoch 後可以達到較低的 perplexity。

分析

- **word embedding:**

對於沒看過的字或是沒有看過的組合也可以經由 word embedding 來得到他們之間的關係。比方說，在 train 中並沒有看過 a + b ，但是 a 之後接 b 的機會並不會為 0，如此可以改善遇到未見過 data 的預測精準度。

- 動詞型態

我們目前的方法沒有對於詞性去做整理，造成對於同一個詞來說，也許只是詞性改變，但是卻可能判斷成不同的字或是 unknown。

- **Sentence concatenation:**

Training data 中是否要去除跨句的情況，例如：a dog. Everyone was shock by。若是選擇去除跨句的情況，則用<end>將句子pad 到num_steps 長度，嘗試結果發現去除跨句的情況似乎對正確率沒有什麼幫助，雖然test perplexity會大幅下降，但主要只是因為padding部分拉低了平均的loss。

分工

	江承恩	沈恩禾	傅鈞笙	吳昭霆
工作項目	架設model	word embedding	資料前處理	調整model