DL HW1 Language Model Report

壹.Environment

- GPU:Tesla K80, OS: Debian 8
- Library: python3, tensorflow: 1.0.0, numpy: 1.12.0

貳.Model Description & How do you improve your performance:

1. Embedding to Embedding

作法：

- input: 任何embedding
- output: 50d的embedding

架構：

- 一層LSTM，雙向

Optimization：

- 方法一： cost為句子中每個字的embedding與prediction做cosine similarity
- 方法二： cost為句子中每個字的embedding與prediction做MSE
- 方法三： cost為句子中最後一個字(backward為第一個字)的embedding與最後一個字的prediction做cosine similarity

n_step(**分為有**padding**及無**padding)：

- padding作法：
  因Optimization的方法不一，故padding的方法也會有異
  1. Optimization方法一
     - Forward: 句點後補0的embedding（因為數字部分在parse training data的時候就濾掉了，且testing data裡並無0）
     - Backward: 句點後補0的embedding
  2. Optimize方法三（因為不希望他學到的最後一個字都一樣）
     - Forward: 前面增加0的embedding，再補句點，再接句子
     - Backward: 句點後補0的embedding
- no_padding作法：
  1. training: random取開頭(但最大必須在結尾前n_step個字)
  2. testing: 試著從頭開始，若從頭開始前n_step的字還沒到要做的答案的位置的話，則自要做的答案往回推前n_step的字

Weighted Cosine-similarity

因為有forward和backward兩個方向，每個選項都可以得到兩個cosine-similarity。因為使用rnn模型，越後面的output應該越有價值，所以在combine這兩個similarity時，我們會考慮克漏字的位置。(越靠近句子開頭，backward-similarity的比重就越大，反之則是forward比重提升)

理由：

因為embedding裡只考慮相近關係，並未考慮前後關係，故其embedding的意義應與字義比較相關，所以我們希望RNN能把其embedding增加語法的意義進去

實驗數據(因為只限三頁故放不下全部實驗數據)

- Embedding: glove_50
- N_Hidden: 256
- direction: both

epoch	n_step	padding	stop_word	optimization	accuracy
6	6	False	Exist	method_2	0.30000
4	10	False	Removed	method_1	0.32885
4	40	True	Removed	method_3	0.29231
2	10	False	Exist	method_1	0.30385

2.One-hot vector RNN

統計corpus中前12000頻繁的字作為字庫，將每個字對應到一個維度，input是12000維的one-hot vector。

架構

- 兩層256維hidden layer的lstm
- output經過softmax成12000維的vector
- training時有dropout

實驗

- Adam Optimizer (learning rate=0.01)
- forward LM。testing時考慮目標前的n-1字的機率，與包含候選字預測下一個字的機率相乘
- 移除unknown的字，或是使用一個維度作為unknown的字
- 在長度<n_steps的句子使用0 padding

結果(因為只限三頁故放不下全部實驗數據)

epoch	n_step	remove unknown	accuracy
10	10	No	0.30577
10	10	Yes	0.39038

3.Total similarity (best)

方法

利用word embedding model中得到的embedding來計算每個選項和句子中其他字的cosine similarity的總和，最後以總和最大的作為預測答案。

改進

- 除了pre-trained好的模型外，我們也利用LSA(Latent Sematic Analysis)產生了一個模型，得到很好的結果。
- 由於句子中有很多不影響結果的單字(如a, if, the)，因此嘗試了只看content word還有只看有dependency的字兩種方法，結果都有進步。

實驗項目

- 不同word embedding 模型：使用Glove(6B words, 50維)、Google Word2Vec(GoogleNews 300維)、在training set上訓練的word2vec(w2v_train)、LSA
- 考慮句子中不同的字：測試了全部字都考慮、僅考慮content word(名詞、動詞、形容詞、副詞)、僅考慮和選項有dependency的字三種情形。

結果(因為只限三頁故放不下全部實驗數據)

model	words	accuracy	model	words	accuracy
Glove	ALL	0.31932	LSA	ALL	0.45000
Google	ALL	0.30192	LSA	content	0.50000
w2v_train	ALL	0.41346	LSA	dependency	0.45962

參.Team Devision

Model 1: 簡瑋德,李承軒
Model 2: 劉岳承
Model 3: 黃兆緯