深度學習應用 - 作業一

系級：電機所碩一　姓名：楊冠彥　學號：R11921091

Q1: Data processing

我使用 sample code 。

a. 針對 intent 及 slot 我都使用 sample code 的方式，根據空格去分隔文字，並依據每個 words 出現的頻率建成字典。

b. 在這個作業中，我直接使用了 token dictionary 和 Glove 的 glove.840B.300d 的 pre-trained embedding 。這個 pre-trained embedding 為 840B tokens、2.2M vocab、300 維度向量且有區分大小寫的 embedding 。另根據 preprocess.sh 回傳的統計結果，約有 83% 的 token 在 intent classification 中被 GloVe 涵蓋，在 slot classification 中則涵蓋了 72%。

Q2: Describe your intent classification model.

a. model

我將我的模型區分成embedding layer、bi-LSTM、Fully-connected layer 三層。經過前述 Data processing的tokenizer會輸出long type 的長度為L的vector 。

Embedding layer

將 input sequence token 透過 glove embedding 轉換為300維 word vector 。

Input: 一個long type 的長度為 L 的vector

Output: (L, D_{emb})

bi-LSTM

首先，使用一層bi-LSTM來做特徵提取， h_t 是時間為 t 時的隱藏狀態（輸出）， X_0 是目前的輸入，也就是input sequence， X_T則是目前隱藏節點的輸出，

$$X_T, (h_t, c_t) = LSTM(X_0, (h_0, c_0))$$

Fully-connected layer

將bi-LSTM輸出 h_t 輸入Fully-connected layer，即

$$y = (y(0), \ldots, y(149)) = MLP(concat(h_t[-1], h_t[-2]))$$

(如果 bidirection = False 則

$$y = (y(0), \ldots, y(149)) = MLP(concat(h_t[-1]))$$)

最終預測輸出

預測將會是分數最高的類別，即

$$y* = argmax(y)$$

一些bi-LSTM model的設定:

1. hidden size: 256
2. num layer: 2
3. dropout: 0.5

一些MLP的設定:

1. hidden size: 512*512
2. output size: 512*150
3. dropout: 0.5

b. Performance

- Validation accuracy: 0.905
- Public accuracy: 0.91422

c. Loss function

我使用標準Cross entropy loss，定義 $y*$為 intent classifier output， gt 為 ground truth ，所以

$$Loss = CrossEntropyLoss(y*, gt)$$

d. Optimization algorithm, learning rate and batch size etc.

- optimization algorithm: Adam (weight decay=0)
- learning rate: 1e-3 = 0.001
- batch size: 32
- number of epoch: 100

- max lengh: 64

Q3: Describe your slot tagging model.

a. model

我將我的模型區分成embedding layer、bi-LSTM、Fully-connected layer 三層。經過前述Data processing的tokenizer會輸出long type 的長度為L的vector

Embedding layer

將 input sequence token 透過 glove embedding 轉換為300維 word vector 。

Input: 一個long type 的長度為L的vector

Output: (L, D_{emb})

bi-LSTM

首先，使用一層bi-LSTM來做特徵提取， h_t 是時間為 t 時的隱藏狀態（輸出）， X_0 是目前的輸入，也就是input sequence， X_T則是目前隱藏節點的輸出，

$$X_T, (h_t, c_t) = LSTM(X_0, (h_0, c_0))$$

Fully-connected layer

將bi-LSTM輸出 h_t 輸入Fully-connected layer，即

$$y = (y(0), \ldots, y(9)) = MLP(X_T)$$

最終預測輸出

預測將會是分數最高的類別，即

$$y* = argmax(y)$$

一些bi-LSTM model的設定:

1. hidden size: 256
2. num layer: 2
3. dropout: 0.5

一些MLP的設定:

1. hidden size: 512*512
2. output size: 512*150

3. dropout: 0.5

b. Performance

- Validation accuracy: 0.817
- Public accuracy: 0.79356

c. Loss function

我使用標準Cross entropy loss，定義 $y*$為 intent classifier output， gt 為 ground truth ，所以

$$Loss = CrossEntropyLoss(y*, gt)$$

d. Optimization algorithm, learning rate and batch size etc.

- optimization algorithm: Adam (weight decay=0)
- learning rate: 1e-3 = 0.001
- batch size: 32
- number of epoch: 100
- max lengh: 64

Q4: Sequence Tagging Evaluation

Seqeval

```
seqeval classification report
              precision    recall  f1-score   support

        date       0.78      0.76      0.77       206
  first_name       0.97      0.92      0.94       102
   last_name       0.91      0.76      0.83        78
      people       0.75      0.77      0.76       238
        time       0.85      0.89      0.87       218

   micro avg       0.82      0.81      0.82       842
   macro avg       0.85      0.82      0.83       842
weighted avg       0.82      0.81      0.82       842

Joint Acc: 0.817
Token Acc: 0.9703459637561779
```

- Joint accuracy

$$\text{Joint accuracy} = \frac{\text{正確被預測的}sequence}{\text{所有被預測}sequence\text{的數量}}$$

- Token accuracy

$$\text{Token accuracy} = \frac{\text{正確被預測的}token}{\text{所有被預測}token\text{的數量}}$$

Seqeval 會把每個 sequence predict/ground truth 拆解為(tag, begin, end)的序列，

> ex.
> y_true = [['O', 'O', 'O', 'B-MISC', 'I-MISC', 'I-MISC', 'O'], ['B-PER', 'I-PER', 'O']]
> y_pred = [['O', 'O', 'B-MISC', 'I-MISC', 'I-MISC', 'I-MISC', 'O'], ['B-PER', 'I-PER', 'O']]
> -> y_true = [[(MISC, 3, 6)], [(PER, 0, 2)]]
> -> y_pred = [[(MISC, 2, 6)], [(PER. 0, 2)]]

再對每個 tag 計算 true positive(TP)，也就是每個 squence 同樣(tag, begin, end)數量，

- $Recall$: 表預測能命中多少真實的正樣本

$$\text{Recall} = \frac{TP}{TP + FN}$$

- $Precision$: 表預測出為正樣本的準確度

$$\text{Precision} = \frac{TP}{TP + FP}$$

- $F1 - score$: 為$precision$和$recall$的調和函數

$$\text{F1-score} = \frac{2}{\text{Recall}^{-1} + \text{Precision}^{-1}}$$

- $Support =$ 真實$(tag, begin, end)$為該tag的數量
- $Microavg =$

$$\frac{\text{所有}tag\text{的}TP}{\text{所有}tag\text{的}TP + FP(\text{或}FN)}$$

- $Macroavg$: 對所有的$precision$及$recall$做平均
- $Weightedavg$: 根據tag數量做$weightedsum$

Q5: Compare with different configurations

1. hidden size

我想知道hidden size對準確度的影響，找出相對好的hidden size與epoch組合。

GRU

分別測試hidden size 為16、64、128、256、512，其餘參數設定與Q2和Q3完全一致。

Intent

Hidden Size	Best Epoch	Train Loss	Train Acc.	Val. Loss	Val. Acc.
16	100	1.870003	0.3982	1.666143	0.639333
64	86	0.179835	0.950866	1.093286	0.891
128	36	0.137901	0.962333	0.838733	0.902
*256**	46	0.113211	0.969666	0.869564	0.907666
512	76	0.259974	0.9338	0.908806	0.887

*: 為該張表中驗證準確度最高的結果

Slot

Hidden Size	Best Epoch	Train Loss	Train Acc.	Val. Loss	Val. Acc.
16	36	0.08825	0.380728	0.066056	0.405
64	97	0.020173	0.701546	0.020226	0.735
128	98	0.008617	0.833103	0.015606	0.777
256	99	0.004206	0.897984	0.016017	0.831
*512**	100	0.002134	0.947404	0.017795	0.841

*: 為該張表中驗證準確度最高的結果

LSTM

Intent

Hidden Size	Best Epoch	Train Loss	Train Acc.	Val. Loss	Val. Acc.
16	99	0.761380	0.760266	1.406203	0.840333
64	100	0.072918	0.980333	1.015609	0.903666
128	86	0.073756	0.980333	0.810521	0.91266

Hidden Size	Best Epoch	Train Loss	Train Acc.	Val. Loss	Val. Acc.
*256**	70	0.050195	0.9996	0.720059	0.92
512	80	0.050459	0.988333	1.127070	0.914333

*: 為該張表中驗證準確度最高的結果

Slot

Hidden Size	Best Epoch	Train Loss	Train Acc.	Val. Loss	Val. Acc.
16	100	0.067648	0.368580	0.058628	0.39
64	99	0.021782	0.674627	0.024262	0.715
128	97	0.0084	0.840557	0.016649	0.802
256	95	0.004441	0.899917	0.015021	0.828
*512**	81	0.002904	0.929596	0.015282	0.83

*: 為該張表中驗證準確度最高的結果

小總結

對於Intent的資料集而言，我發現似乎到了一定程度就會overfitting，也就是盲目調高Hidden size不一定會取得好的結果，因為在hidden size=512時，驗證準確度都較hidden size=256來得差。slot的資料集則雖hidden size越高，驗證準確度也越高，但hidden size越大，代表抽出特徵越多，因此訓練時間也較久。另外，hidden size很小的話，驗證準確度也較差，甚至差異蠻大的。

2. Number of layers

我實驗將GRU/LSTM設定不同層數，分別測試1、2、3層，來尋找相對較好準確性的結果。

GRU

Intent

根據前面Hidden Size的實驗，驗證準確度(Val. Acc.)較高的為hidden size=256，所以以此為基準測試Number of Layer=1, 2, 3。

Number of Layer	Best Epoch	Train Loss	Train Acc.	Val. Loss	Val. Acc.
1	65	0.070977	0.979666	1.067042	0.904
*2**	46	0.113211	0.969666	0.869564	0.907666
3	70	0.168817	0.952666	0.961011	0.896

*: 為該張表中驗證準確度最高的結果

Slot

根據前面Hidden Size的實驗，驗證準確度(Val. Acc.)較高的為hidden size=512，所以以此為基準測試Number of Layer=1, 2, 3。

Number of Layer	Best Epoch	Train Loss	Train Acc.	Val. Loss	Val. Acc.
1	100	0.002587	0.935118	0.017515	0.83
*2**	100	0.002134	0.947404	0.017795	0.841
3	84	0.002184	0.947266	0.016682	0.839

*: 為該張表中驗證準確度最高的結果

LSTM

Intent

根據前面Hidden Size的實驗，驗證準確度(Val. Acc.)較高的為hidden size=256，所以以此為基準測試Number of Layer=1, 2, 3。

Number of Layer	Best Epoch	Train Loss	Train Acc.	Val. Loss	Val. Acc.
1	93	0.055822	0.9882	1.055977	0.919
*2**	70	0.050195	0.9996	0.720059	0.92
3	93	0.066227	0.984866	1.02114	0.914

*: 為該張表中驗證準確度最高的結果

Slot

根據前面Hidden Size的實驗，驗證準確度(Val. Acc.)較高的為hidden size=512，所以以此為基準測試Number of Layer=1, 2, 3。

Number of Layer	Best Epoch	Train Loss	Train Acc.	Val. Loss	Val. Acc.
1	94	0.002882	0.929734	0.017374	0.822
2	81	0.002904	0.929596	0.015282	0.83
*3**	84	0.002289	0.943677	0.017507	0.833

*: 為該張表中驗證準確度最高的結果

小總結

在 Intent 資料集中，Number of Layer=2 無論在LSTM或是GRU都會有比較好的表現；在 Slot 資料集中，Number of Layer=2 在 GRU 中有較好的表現，在 LSTM 中則是 Number of Layer=3 有較好表現。

Final model configuration

根據Q5比較結果，整理出最終最佳設定參數。

Intent Classifier

Name	Number of Layer	Hidden Size	Best Epoch	Train Loss	Train Acc.	Val. Loss	Val. Acc.
LSTM	2*	256	70	0.050195	0.9996	0.720059	0.92

Performance on Kaggle:

- Public accuracy: 0.92

Slot Classifier

Name	Number of Layer	Hidden Size	Best Epoch	Train Loss	Train Acc.	Val. Loss	Val. Acc.
GRU	2*	512	100	0.002134	0.947404	0.017795	0.841

Performance on Kaggle:

- Public accuracy: 0.82252

Reference

- Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation
- seqval