Applied Deep Learning HW1

r11944049 呂瑋浩

October 2022

1 Q1: Data processing

使用sample code，計算出現頻率前10000名的詞並加上[PAD]、[UNK]，分別代表Padding、Unknown，Padding是為了將句子補充到最大長度，而Unknown則是不出現在pre-trained embedding的字，在slot tagging中，由於每個字都有label，所以計算Crossentropy時，必須忽略掉Padding的字，在本次實驗中使用9作為label(0-8為原本的label)，pre-trained embedding使用glove.840B.300d，代表有840B tokens, 2.2M vocab, 並產生300維的vector。

2 Q2: Describe your intent classification model

我實做的model包含一層BiLSTM，hidden size為256，兩層fully connected network，在Kaggle上得到了0.91777的分數。

1. $S = \{s_1, s_2, ..., s_N\}, s \in \mathbf{R}^{300}$ 為句子的Word embedding
2. 將S通過一層BiLSTM

$$out_t, h_t, ct = BiLSTM(s_t, h_{t-1}, c_{t-1})$$

3. 最後h_t 通過兩層fully connected network，中間採用Batch normalize、LeakyReLU、Dropout

我使用了Cross entropy作為Loss function，Optimization algorithm使用Adam，learning rate初始為1e-4，weight decay設為1e-5，並採用Step scheduler，每15epoch將learning rate調小0.1，Batch size設為128，並使用了Early stopping來防止Overfitting，每10個epoch保存validation sets上表現最好的參數。

3 Q3: Describe your slot tagging model.

我實做的model包含一層BiLSTM，hidden size為1024，一層fully connected network，在Kaggle上得到了0.79410的分數

1. $S = \{s_1, s_2, ..., s_N\}, s \in \mathbf{R}^{300}$ 為句子的Word embedding

2. 將S通過BiLSTM

$$out_t, h_t, ct = BiLSTM(s_t, h_{t-1}, ct-1)$$

3. 最後將out 通過fully connected network，中間採用LeakyReLU、Dropout

我使用了Cross entropy 作為Loss function，並採用L2 regularization，其中因為會將句子padding到最大長度，所以當label為9時會忽略不計。

$Loss = Crossentropy(\hat{y}, y) + 1e-6 * \Sigma w_i^2$, where w_i is the model weight

使用Adam為Optimizaion algorithm，learning rate為1e-3，採用Step scheduler，每15epoch將learning rate調小0.1，Batch size設為128，並使用了Early stopping來防止Overfitting，每10個epoch保存validation sets上表現最好的參數。

4 Q4: Sequence Tagging Evaluation

	precision	recall	f1-score	support
date	0.78	0.79	0.79	206
first_name	0.96	0.96	0.96	102
last_name	0.83	0.77	0.80	78
people	0.78	0.74	0.76	238
time	0.84	0.83	0.83	218
micro avg	0.82	0.80	0.81	842
macro avg	0.84	0.82	0.83	842
weighted avg	0.82	0.80	0.81	842

Figure 1: Classification report

Token accuracy: 預測的token為單位

$$Token\ accuracy = \frac{correct\ tokens\ predicted}{all\ tokens}$$

Joint accuracy: 預測的sequence為單位，整個sequence都預測對才算正確

$$Joint\ accuracy = \frac{correct\ sequence\ predicted}{all\ sequence}$$

Joint accuracy比Token accuracy嚴格許多，所以會使用Precision、Recall、F1 score來輔助判斷，當Token accuracy很高，Joint accuracy很低時能知道是哪邊出現預測錯誤。

		Real label	
		Positive	Negative
Predicted label	Positive	True Positive (TP)	False Positive (FP)
	Negative	False Negative (FN)	True Negative (TN)

Precision:在預測為正的樣本中有多少是對的

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

Recall:在真實為正的樣本中有多少是預測對的

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

F1 score: Precision與Recall的調和平均

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

support: Real label為該類別的數量

micro avg: 不對類別分類計算混淆矩陣

macro avg: 根據每個類別Recall、Precision做算術平均

weighted avg: 根據每個類別Recall、Precision做加權平均

5 Q5: Compare with different configurations

5.1 Intent classification

Hidden size	Train	Dev	Test
256(Q2)	98.860%	91.167%	91.777%
512	99.720%	92.067%	92.533%
1024	**99.760**%	**92.600**%	**92.577**%

Layer	Train	Dev	Test
1	99.760%	**92.600**%	92.577%
2	**99.800**%	92.100%	**92.622**%

我調整Q2的模型(BiLSTM)的兩個參數：Hidden size、layers，來增進Performance，Hidden size在1024時的表現是最好的，但是速度大幅下降，到了訓練Hidden size 2048的模型時，訓練所需時間太長了，所以嘗試增加Layer，而透過實驗發現一層BiLSTM在validation set上的表現比兩層BiLSTM好，所以選用一層BiLSTM且Hidden size為1024的model。

Model	Train Acc	Dev Acc	Test Acc
BiLSTM	99.760%	**92.600**%	**92.577**%
BiGRU	99.740%	92.300%	92.311%
CNN-BiLSTM	**99.780**%	92.333%	92.488%

接著我實作了三種不同的Model，BiGRU為BiLSTM的變形，在Validation sets上並沒有比BiLSTM好，同樣在Test sets上也是，然而在訓練速度上BiGRU比BiLSTM還快。我實做的CNN-BiLSTM為在BiLSTM前加上一層Convolution layer(kernel=3)，CNN會透過將字與字做權重相加提取中間的特徵來增加Performance，也消除一些很少出現的字避免模型Overfitting，實驗結果反而沒有比單純使用BiLSTM好。

5.2 Slot tagging

Hidden size	Train	Dev	Test
256	95.003%	78.600%	76.675%
512	98.012%	79.300%	78.123%
1024(Q3)	**98.771**%	**80.700**%	**79.410**%

Layer	Train	Dev	Test
1	**98.771**%	80.700%	79.410%
2	97.943%	**82.200**%	**80.536**%

我調整Q3的模型(BiLSTM)的兩個參數:Hidden size、layers，來增進Performance，實驗結果發現Hidden size為1024、Layer為2是表現最好的，這個任務比Q1還複雜一些，所以Hidden size調小會出現Underfitting的現象，將Layer調為兩層會表現得更好。

Model	Train Acc	Dev Acc	Test Acc
BiLSTM	97.943%	82.200%	80.536%
BiGRU	**98.371**%	**82.400**%	**80.750**%
CNN-BiLSTM	97.805%	81.200%	80.643%

接著我實作了三種不同Model，BiGRU在Validation及Testing sets上都比BiLSTM好，訓練速度也快。我實作的CNN-LSTM為在BiLSTM前加上一層Convolution layer(kernel=3)，在Slot tagging中有first name、last name的label這種前後是相關的label，我認為Convolution layer能提取這類特徵，進而增進表現，而實驗結果則是Validation set上的表現是最差的，Testing sets上表現比LSTM好，比GRU差，並且在訓練速度上CNN-LSTM的所需時間約為BiLSTM的兩倍，但是只提升0.1%的準確度，如果要解決前後字的相關問題，另一個可能的方法是加上CRF layer來增加表現。