Applied Deep Learning HW1 report

B08310054 歐崇愷

1. Data Processing
   - I use NLTK to tokenize 'text' field and didn't remove stop word and didn't set the max input length of text
   - I use GLOVE as a fixed, pretrained word embedding. Embed each word into 300 dim vector
2. Describe your intent classification model
   - I use LSTM model
     - h_t, c_t = LSTM(w_t, h_{t-1}, c_{t-1}), where w_t is the word embedding of the t-th token.
     - h_t is a 2-dim vector, representing the hidden state of LSTM at the t-th timestamp
     - choose softmax as activation function
   - performance of your model
     - **0.92222 acc**
   - **cross entropy loss**

$$\text{Loss} = -\sum_{i=1}^{\substack{\text{output} \\ \text{size}}} y_i \cdot \log \hat{y}_i$$

   - I use torch.optim.Adam as the optimizer, with
     - Learning rate = 0.003
     - Batch size = 128
3. Describe your slot tagging model
   - I use LSTM model
     - h_t, c_t = LSTM(w_t, h_{t-1}, c_{t-1}), where w_t is the word embedding of the t-th token.
     - h_t is a 2-dim vector, representing the hidden state of LSTM at the t-th timestamp
     - choose softmax as activation function
   - performance of your model
     - **0.76300 acc**
   - **cross entropy loss**

$$\text{Loss} = -\sum_{i=1}^{\substack{\text{output} \\ \text{size}}} y_i \cdot \log \hat{y}_i$$

- I use torch.optim.Adam as the optimizer, with
    - Learning rate = 0.0003
    - Batch size = 1

4. Sequence Tagging Evaluation

```
Training Loss:1.3896,   Training Acc 0.89
Testing Loss:1.4199,   Testing Acc 0.74
              precision    recall  f1-score   support

        date       0.69      0.74      0.71       206
  first_name       0.88      0.72      0.79       102
   last_name       0.83      0.44      0.57        78
      people       0.67      0.63      0.65       238
        time       0.79      0.79      0.79       218

   micro avg       0.74      0.69      0.72       842
   macro avg       0.77      0.66      0.70       842
weighted avg       0.75      0.69      0.71       842
```

- 在自己計算的 joint acc 中得到 0.74 的命中率，然而在 token

    accurancy 中僅得到約 0.71 的命中率，由此可見在較長句子

    中 token 的預測率較為不理想，才會使 token accuracy 小於

    joint accuracy.

5. Compare with different configurations

- In intent classification case，我初始化參數為 learning rate

    0.05, hidden dim 512, numbers of layer 2，發現訓練不起來，

    卡在 acc = 0.6，嘗試降低 hidden dim 至 256 表現略微提升，

    調整 numbers of layer 至 1 或 3 反而更差，最後調整 learning

    rate 至 0.005 後 acc 上漲至 0.92

- In slot tagging case，我初始化參數為 learning rate 0.005, hidden dim 512, numbers of layer 2，發現訓練不起來，acc = 0.01，嘗試降低 hidden dim 至 64 表現略微提升，最後調整 learning rate 至 0.0003 後 acc 上漲至 0.73