ADL 2022 HW1 Report

R10922176 陳冠頴

Q1. Data processing

a. Each word appear in training set will be mapped to a distinct index. The index of 1 is represent the unknow word "[UNK]" and index of 0 represent the padding word "[PAD]". All of the sentences will be padded to length of 128.

b. I use the sample code and use the glove.840b.300d to initialize the embedding model. Each row of embedding model is represented a word vector which the dimension of word vector is 300.

Q2. Describe your intent classification model.

a. Your model

I. Architecture

1. Embedding

   Glove embedding with shape of (6491, 300).

   $x = LayerNorm(Embedding(input))$, where input is a sentence with length 128 and each token in the sentence is encoded by its corresponding index.

2. RNN

   $output_t, h_n = GRU(x_t)$, where x_t is the word embedding of the t-th token.

3. MLP

   i. $m0 = ReLU(Linear(BatchNorm1d(output_{-1})))$, where $output_{-1}$ is the output feature from the last layer of final state of the GRU and the output dimension of *Linear* is 512.

   ii. $m1 = ReLU\left(Linear\left(BatchNorm1d(m_0)\right)\right)$, where the output dimension of the *Linear* is 150 which is the number of classes.

II. Hyperparameters

GRU	hidden_size	num_layers	bidirectional	dropout	epochs
	1024	2	True	0.2	300

b. Performance of your model

I. Accuracy on training set: 0.99987

II. Accuracy on validation set: 0.90967

III. Public score on Kaggle: 0.92044

c. The loss function you used

CrossEntropyLoss

d. The optimization algorithm

AdamW	Learning rate	Weight decay	Batch size
	0.001	1.0	256

Q3. Describe your slot tagging model.

a. Your model

I. **Architecture**

1. Embedding

   Glove embedding with shape of (4117, 300).

   $x = LayerNorm(Embedding(input))$, where input is a sentence with length 128 and each token in the sentence is encoded by its corresponding index.

2. RNN

   $output_t, h_n = GRU(x_t)$, where x_t is the word embedding of the t-th token.

3. MLP

   i. $m0_t = ReLU(Linear(BatchNorm1d(output_t)))$, where $output_t$ is the output feature from the last layer of t-th state of the GRU and the output dimension of *Linear* is 512.

   ii. $m1_t = ReLU\left(Linear\left(BatchNorm1d(m0_t)\right)\right)$, where the output dimension of the *Linear* is 150 which is the number of classes.

4. When inference the test dataset, I will remove the result which exceed the original length of the sentence.

II. Hyperparameters

GRU	hidden_size	num_layers	bidirectional	dropout	epochs
	512	2	True	0.2	300

b. Performance of your model

I. Accuracy on training set: 0.94312

II. Accuracy on validation set: 0.79600

III. Public score on Kaggle: 0.80000

c. The loss function you used

CrossEntropyLoss

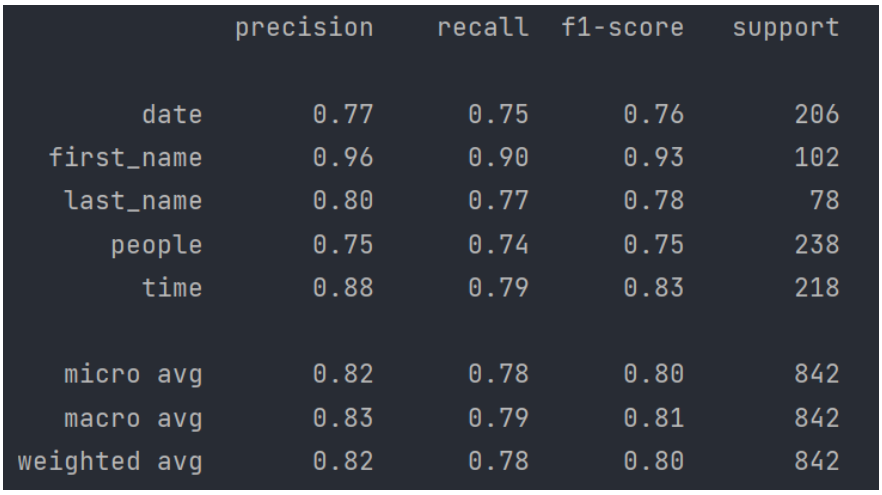
d. The optimization algorithm

AdamW	Learning rate	Weight decay	Batch size
	0.01	1.0	256

Compare with the intent classification task, I set the relatively large learning rate at the begin and use CosineAnnealingLR(T_max=5) as the leaning rate scheduler to decrease the learning rate when the model is training.

Q4. Sequence Tagging Evaluation

a. Code implement in seqeval_on_validset.py



```
              precision    recall  f1-score   support

        date       0.77      0.75      0.76       206
  first_name       0.96      0.90      0.93       102
   last_name       0.80      0.77      0.78        78
      people       0.75      0.74      0.75       238
        time       0.88      0.79      0.83       218

   micro avg       0.82      0.78      0.80       842
   macro avg       0.83      0.79      0.81       842
weighted avg       0.82      0.78      0.80       842
```

b. Token accuracy measured the accuracy in token level.
Joint accuracy measured the accuracy in sentence level which means that the accuracy is 100% if and only if all of the token in the sentence is correct.

Q5. Compare with different configurations

1. Experiment of difference weight decay on intent classification task.

Except of the weight decay, all of the hyperparameters are the same as Q2.

Weight Decay	Best Epoch	Train Acc	Train Loss	Val. Acc	Val. Loss
0.0	13	0.99960	0.00318	0.88233	0.52617
0.0001	177	1.00000	0.00177	0.89733	0.42944
0.01	124	0.93306	0.54444	0.71900	1.16115
1.0	**222**	**0.99986**	**0.00486**	**0.90966**	**0.38400**

The figure1 below is the training curve of weight decay 0.0, and the figure2 below is the training curve of weight decay 1.0.

2. Finding

We can see the best epoch is 13 when the weight decay is 0.0 and the public score on Kaggle is 0.89200. I thought that the model trained on epoch more than 13 is overfitting, so I tried to set different values of weight decay to do the regularization on model. After some experiments I got the best model by set weight decay to 1.0 which had best performance on validation set. The public score on Kaggle has improved from 0.89200 to 0.92044.

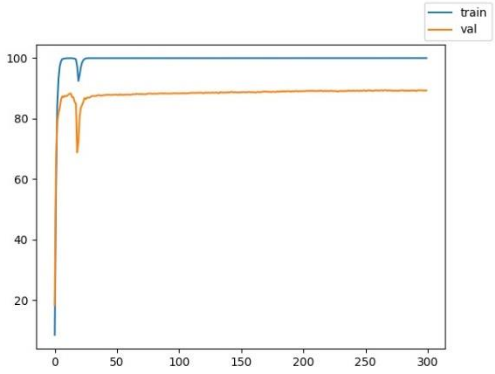


Figure1

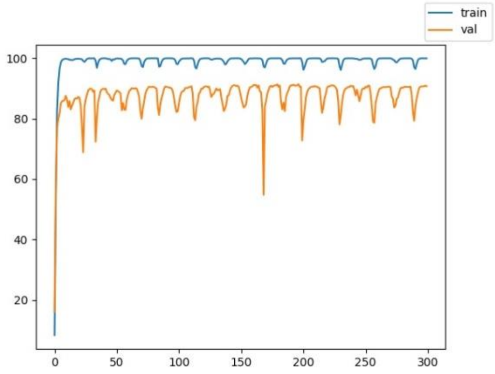


Figure2