Homework1

R08525116 吳承哲

Q1: Data processing

a. How to tokenize the data.

1. Load the data from train.json and eval.json.
2. Find all intents and give them id number and save as a dictionary. There are 150 classes.
3. Save all the words to counter class which is one kind of dictionary, so we can choose common words we need. There are totally 6489 words.
4. Build a vocab class, and put the words in.
5. Build a SeqClsDataset, and write the collate_fn to return data of tensor format.
   5.1 Intent: Encode the texts to vector and pad them with 0, and then turn the string labels to index number. The max length is 28 because the longest sentence contains 28 words.
   5.2 slot: Encode the texts to vector and pad them with 0. Turn string labels to
   5.3 index number and pad them with -100. The reason why use -100 is the cross entropy loss would ignore the index by default. The max length is 35 because the longest sentence contains 35 words.
6. Build dataloader and call the collate_fn we write to process the data, and then get the input data which is tensor format.

b. The pre-trained embedding: Glove

Pick the vectors of words in our data from glove.

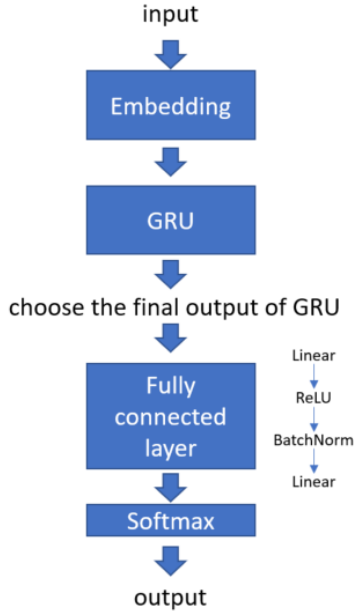
Q2: Describe the intent classification model.

a. **Model**

$$h_0 = Embedding(x)$$
$$h_t, c_t = GRU(h_{t-1}, c_{t-1})$$
$$o = \text{Softmax}(FC(h_{tf}))$$

Here x is input, h_0 is output of embedding layer. The h_t is output of GRU, and c_t is hidden state to next GRU layer. The h_{tf} is the last dimension of h_t, and o is output.

Layer	GRU	Linear
Configuration	Hidden size: 256 Layer number: 2 Dropout: 0.3 Bidirectional: True	Hidden size*2 → Hidden size → Classes number



b. **Performance of model**

Unit: (%)

Train accuracy = 99.80%

Val accuracy = 93.29%

Public score = 92.40%

c. **The loss function:** Cross entropy

d. **The optimizer algorithm:** Adam

Learning rate: 0.001

Batch size: 256

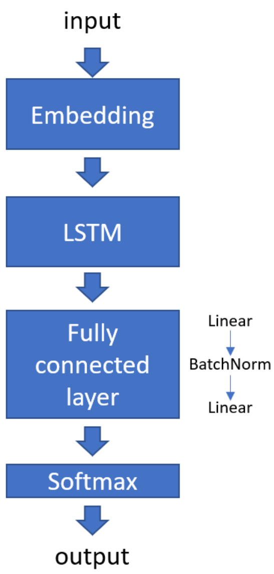
Q3: Describe the slot tagging model.

a. **Model**

$$h_0 = Embedding(x)$$
$$h_t, c_t = LSTM(h_{t-1}, c_{t-1})$$
$$o = \text{Softmax}(FC(h_t))$$

Here x is input, h_0 is output of embedding layer. The h_t is output of LSTM, and c_t is hidden state to next LSTM layer. The o is output.

Layer	LSTM	Linear
Configuration	Hidden size: 256 Layer number: 2 Dropout: 0.2 Bidirectional: True	Hidden size*2 → Hidden size → Classes number



b. Performance of model

Unit: (%)

Train joint accuracy: 91.67

Val joint accuracy: 80.07

Val accuracy: 96.39

Public score: 75.656

c. The loss function: Cross entropy

d. The optimize algorithm: Adam

Learning rate: 0.001

Batch size: 128

Q4: Sequence Tagging Evaluation

a. Use seqeval to evaluate the model in Q3 on validation set

	precision	recall	f1-score	support
date	0.75	0.74	0.74	206
first_name	0.93	0.88	0.90	102
last_name	0.89	0.73	0.80	78
people	0.77	0.71	0.74	238
time	0.84	0.90	0.87	218
micro avg	0.81	0.79	0.80	842
macro avg	0.84	0.79	0.81	842
weighted avg	0.81	0.79	0.80	842

b. Explain the difference between the evaluation method in seqeval. Token accuracy, and joint accuracy.

Generally, the class we care about is positive, and other is negative. In the classification task, there are four situations.

TP (true positive): label is positive and prediction is also positive.

TN (true negative): label is negative and prediction is also negative.

FP (False positive): label Is positive but prediction is negative.

FN (False negative): label is negative but prediction is positive.

Precision

Precision is the percentage of correctness among the positive predictions.

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

Recall

Recall is the percentage of correctness among the positive samples.

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

F1 score

F1 score is considered the ability of precision and recall at the same time, it is harmonic mean between precision and recall.

$$\frac{1}{precision} + \frac{1}{recall} = \frac{2}{f1 - score}$$

$$\mathrm{f1 - score} = \frac{2TP}{2TP + FP + FN}$$

Micro average

Sum up the TP, FP, FN of every class first, and then apply them to get the statistics.

Macro average

Take the average of metrics between every class.

Weighted average

Be the same as macro average, but multiply the weight to each class.

Joint accuracy

In the slot tagging task, if the prediction of each word in the sentence is true, it is considered as one correctness prediction. Therefore, this metric is regarded one sentence as one unit.

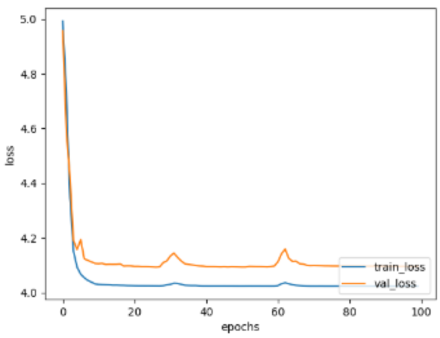
Token accuracy

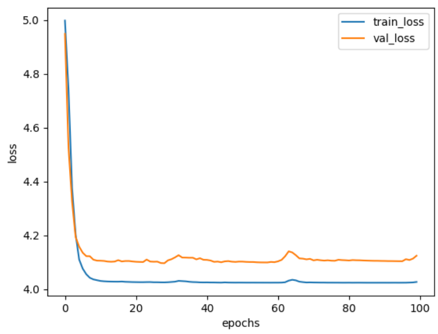
In the slot tagging task, the metric is regarded one word as one unit to evaluate the accuracy.

Q5: Compare with different configurations

Intent

Model	Accuracy (%) (train/val/public score/private score)	Configuration
Q2 model	99.80 / 93.29 / 92.400 / 92.755	
model	99.73 / 93.66 / 92.888 / 93.377	Delete relu

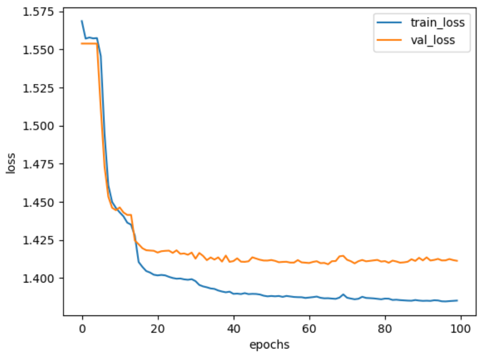


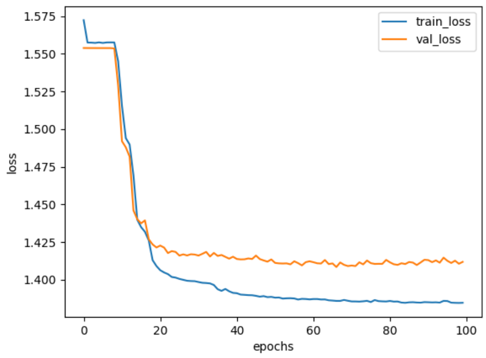


First, I used two linear layers, and got the worse result, Then I added relu and batch normalization, and I got a better result. I thought the model need to be more complex, so I added relu activation function. Batch normalization could make model faster and more stable by re-centering and re-scaling. However, when I deleted relu, I got a better result than Q2 model, I thought the task may not be so difficult that the model didn’t need to be complex. I also fine turn the hidden size and dropout rate, and the settings I used were the best in my experiment.

Slot tagging

Model	Joint Accuracy (%) (train/val/public score/private score)	Configuration
Q3 model	91.67 / 80.07 / 75.656 / 77.170	
model	91.17 / 78.61 / 71.903 / 73.847	Add relu





In the task, it is similar as intent classification. The relu activation function couldn’t improve the efficiency of the model effectively. Therefore, Without relu is a better architecture in this task. Besides, the hidden size and dropout rate I used is also the best settings than other.