Adaptive Deep Learning, 2022 Fall

Assignment 1

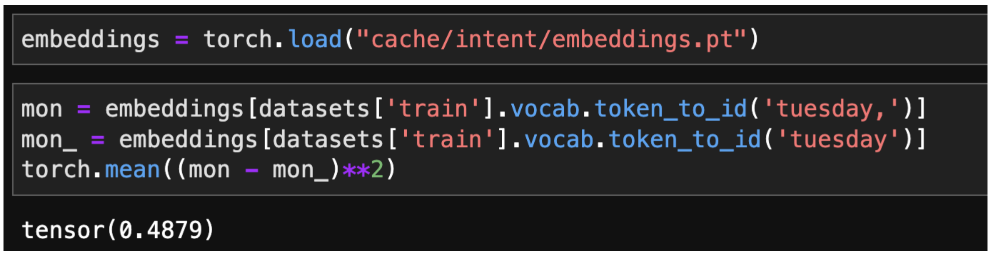
Chen, Chun Hsien

R10246002

October 16, 2022

Question 1: Data processing

(a) First, the input sentences are splitted by spaces. Second, we observed that a word with or without punctuation may have different word embeddings which is not close to each other, so I decided to remove all the punctuations.



```
embeddings = torch.load("cache/intent/embeddings.pt")
```

```
mon = embeddings[datasets['train'].vocab.token_to_id('tuesday,')]
mon_ = embeddings[datasets['train'].vocab.token_to_id('tuesday')]
torch.mean((mon - mon_)**2)
```

```
tensor(0.4879)
```

Figure 1: Distance between the embedding a word with or without a comma.

(b) The pretrained word embedding we used is GloVe (840B tokens, into 300d vector) for both intent classification and slot tagging.

(c) Texts in the same batch will be padded by 0 to same length. And the words not in the vocabulary is labeled by index 1. Each index corresponds to a 300 dimensional vector.

Question 2: Describe your intent classification model

(a) Baseline Model

Layers	Input dimension	Output dimension
Embedding	(N, L)	$(N, L, 300)$
BiGRU	$(N, L, 300)$	$(N, L, 2 \times \text{hidden\_size})$
MaxPooling	$(N, L, 2 \times \text{hidden\_size})$	$(N, 2 \times \text{hidden\_size})$
Linear	$(N, 2 \times \text{hidden\_size})$	$(N, 150)$

Table 1: Intent classification model architecture

Where N denotes the batch size, L is the text length after padded, noted that L can vary among batches. BiGRU stands for bidirectional GRU, with only one layer. After GRU extracts important sequential features, we take maximum on the second dimension of its output, which is done by the layer "Max-Pooling".

(b) Performance of your model. (public score on kaggle)

Baseline Model	Validation*	Public	Private
Accuracy	0.951	0.94044	0.94311

Table 2: Performance of intent
* means only training data is used, while the model used for generating prediction is trained on all data available, including validation data.

(c) $CrossEntropyLoss$ is used,

$$l(x,y) = \sum_{n=1}^{N} \frac{1}{\sum w_{y_n}} l_n, \ where\ l_n = -w_{y_n} log \frac{exp(x_{n,y_n})}{\sum_{i=1}^{C} exp(x_{n,i})} \tag{1}$$

where x is the input, y is the target, w is the weight, C is the number of classes, and N spans the minibatch dimension. w_{y_n} are set to be 1 for all n.

(d) Optimization algorithm

1. Optimizer: AdamW
2. Initial learing rate: 0.001, lr will decay by a factor l, if the loss doesn't get smaller for more than K epochs.
3. Epoch: 30
4. Batch size: 32

Question 3: Describe your slot tagging model.

(a) Baseline Model

Layers	Input dimension	Output dimension	Details
Embedding	(N, L)	$(N, L, 300)$	-
BiLSTM	$(N, L, 300)$	$(N, L, 2 \times$ hidden_size$)$	dropout=0.3
Linear1	$(N, L, 2 \times$ hidden_size$)$	$(N, L, 128)$	dropout=0.3
Linear2	$(N, L, 128)$	$(N, L, 128)$	dropout=0.3
Linear3	$(N, L, 128)$	$(N, L, 10)$	-

Table 3: Slot tagging model architecture
Where N denotes the batch size, L is the text length after padded, noted that L can vary among batches. 2 layers of BiLSTM is used, and the activation function between each linear layer is Hardswish, which is defined by $Hardswish(x) = \begin{cases} 0 & \text{if } x \leq -3 \\ x & \text{if } x \geq 3 \\ x(x+3)/6 & \text{otherwise} \end{cases}$

(b) Performance of your model. (public score on kaggle)

Baseline Model	**Validation***	**Public***	**Private**
Joint Accuracy	0.821	0.80965	0.82261

Table 4: Performance of intent
* means only training data is used, since the public score doesn't improve when full dataset is used for training, the final model is trained only with training data.

(c) Loss function you use
$CrossEntropyLoss$ is used

$$l(x, y) = \sum_{n=1}^{N} \frac{1}{\sum w_{y_n} \cdot I_{y_n \neq ignore\_index}} l_n \tag{2}$$

where,

$$l_n = -w_{y_n} log \frac{exp(x_{n,y_n})}{\sum_{i=1}^{C} exp(x_{n,i})} \cdot I_{y_n \neq ignore\_index} \tag{3}$$

where x is the input, y is the target, w is the weight, C is the number of classes, and N spans the minibatch dimension. w_{y_n} are set to be 1 for all n. In this case, $C = 10$, since we added another token for padding and labeled it as the 10^{th} class. Ignore_index is set to be 9, which corresponds to the tokens used for padding.

(d) Optimization algorithm

1. Optimizer: RMSprop
2. Initial learing rate: 0.001, lr will decay by a factor l, if the loss doesn't get smaller for more than K epochs. Where l and K are parameters to be chosen.
3. Epoch: 100
4. Batch size: 16

Question 4: Sequence Tagging Evaluation

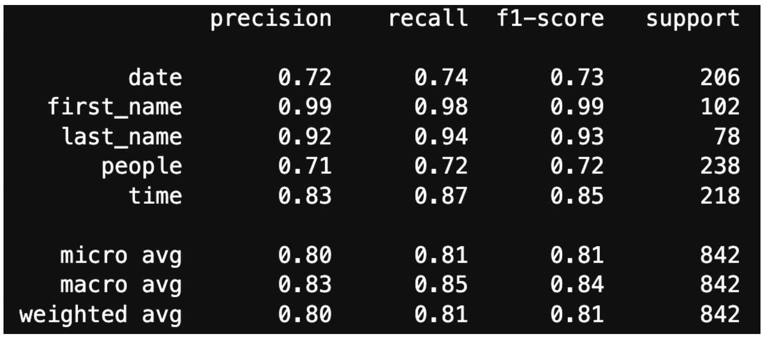
For each token, we denote true positive rate as **TP**, true negative rate as **TN**, false positive rate as **FP**, false negative rate as **FN**.

$$Precision = \frac{TP}{TP + FP}$$
$$Recall = \frac{TP}{TP + FN}$$
$$f1 - score = \frac{2 \times Precision \times Recall}{Precision + Recall}$$
$$Token\ accuracy = \frac{\text{number of tokens predicted correct}}{\text{total number of tokens}}$$
$$Joint\ accuracy = \frac{\text{number of texts predicted correct}}{\text{total number of texts}}$$

$-$ $macro\ avg =$ mean of the corresponding token metric

$-$ $micro\ avg$ is calculated by total number of TP, TN, FP, FN tokens with same formula as above.

$-$ $weighted\ avg$ is weighted by the number of labels



```
              precision    recall  f1-score   support

        date       0.72      0.74      0.73       206
  first_name       0.99      0.98      0.99       102
   last_name       0.92      0.94      0.93        78
      people       0.71      0.72      0.72       238
        time       0.83      0.87      0.85       218

   micro avg       0.80      0.81      0.81       842
   macro avg       0.83      0.85      0.84       842
weighted avg       0.80      0.81      0.81       842
```

Figure 2: Results of classification_report on evaluation data.

Question 5: Compare with different configurations

In this question, we compared the loss and accuracy result from different configuration of the baseline model defined in Q2 and Q3, including number hidden layers of BiGRU or BiLSTM, and hidden dimension.

- Intent classification

  (a) We can see that in this experiment, different layers of BiGRU doesn't have much impact on the rate of convergence. We have the smallest loss value when 2-layered of BiGRU is used, but we get the highest validation accuracy among all epoch when 1 layer BiGRU is used.

GRU layers	Epoch 1	Epoch 10	Epoch 20	Epoch 30
Loss				
1	1.51	0.00801	0.000176	4.13e-5
2	1.45	0.00384	9.03e-5	**2.03e-5**
3	1.60	0.02020	0.00057	4.04e-5
Accuracy				
1	**0.905**	**0.941**	**0.948**	**0.947**
2	0.890	0.937	0.938	0.940
3	0.863	0.910	0.932	0.936

Table 5: Compare the loss and accuracy among different layers of GRU in intent model.

(b) Further more, we compare GRU and LSTM, while other configs remain fixed.

Hidden size	128	256	512
GRU	0.942	0.947	0.948
LSTM	0.938	0.942	0.947

Table 6: Compare the validation accuracy using different RNN-based networks in intent model.

(c) Optimizer also matters a lot. It converges very fast with AdamW, and much slower when RMSprop is used for training.

Optimizer	Epoch 1	Epoch 10	Epoch 20	Epoch 30
AdamW	0.905	0.938	0.948	0.947
RMSprop	0.895	0.911	0.918	0.915

Table 7: Compare the effect of using different optimizers in the training of intent model.

- Slot tagging

  (a) 1 layer of LSTM converge the fast, but we have the smallest loss value when 2-layered of lstm is used, and also the highest validation accuracy.

LSTM layers	Epoch 1	Epoch 10	Epoch 20	Epoch 30
Loss				
1	0.327	0.0442	0.0187	0.0106
2	0.374	0.0521	0.0247	**0.0095**
3	0.584	0.0594	0.0347	0.0186
Joint Accuracy				
1	0.649	0.726	0.787	0.778
2	0.628	0.727	0.771	**0.805**
3	0.628	0.707	0.754	0.797

Table 8: Compare the loss and accuracy among different layers of lstm in slot model.

(b) Further more, we compare GRU and LSTM, while other configs remain fixed. We can see in this experiment, RNNs with hidden layer of size 256 performs better than others, and LSTM get slightly higher validation score than GRU.

Hidden size	128	256	512
LSTM	0.805	**0.821**	0.809
GRU	0.804	**0.816**	0.804

Table 9: Compare the validation accuracy using different RNN networks in slot model. Here, we list the highest validation score during training.

Discussion According to the observations in the above experiments, we can conclude a few points:

(1) More layers in GRU (LSTM) network increases model flexible but also complexity. Causing lower convergence rate and increases training time, and does not promise a better performance. In intent classification model, 1 layer of GRU performs the best, and reaches about 95% accuracy within 30 epochs.

(2) Optimizer can impact the convergence rate and the final result of training, with suitable optimzer, the accuracy can improve more than 3%, even if other configurations remain fixed, as table7 illustrates.

(3) Change GRU to LSTM does not results in significant improvements. LSTM has more parameters than GRU, while having more flexibility, the training is also more time consuming.