SST Classification with RNN(LSTM)

Name: Shi Minglei Students ID: 2020011245 Department: Automation

1 Implementation

1. 1 Project Structure

- Project
    - .data
    - data
    - output
    - main.py
    - network.py
    - engine.py
    - losses.py
    - optimizer.py
    - plot.py
    - dataset.py
    - cmd.sh

I design my project structure as file trees showed above. Each folder or module is illustrated below.

- **.data**

  There are SST-5 dataset in this folder. Supply the data in these experiments.

- **data**

  There are word2vec pretrained word embeddings given by teaching assistant.

- **output**

  There is some log information outputted by each experiment such as model information, hyerparameters and so on.

- **main.py**

  In main.py, we define the model, criterion, optimizer, training process, plot function and some log information.

- **network.py**

  In network.py, we define three models needed in these experiments, one MLP model and two ConvNets model.

- **engine.py**

  In engine.py, we define train_one_epoch function and training process, in the meanwhile, we define validate and test function.

- **losses.py**

  In losses.py, we define different criterion functions include Cross Entropy loss and Euclidean

loss.

- **optimizer.py**

In optimizer.py, we define different optimizer include SGD without momentum, SGD with momentum, Adam, AdamW and other functions.

- **plot.py**

In plot.py, we define plot functions to plot loss and accuracy curve in our experiments. This module is the same as ploy.py supplied in hw3.

- **cmd.sh**

The instructions used in these experiments.

1.2 Model Architecture

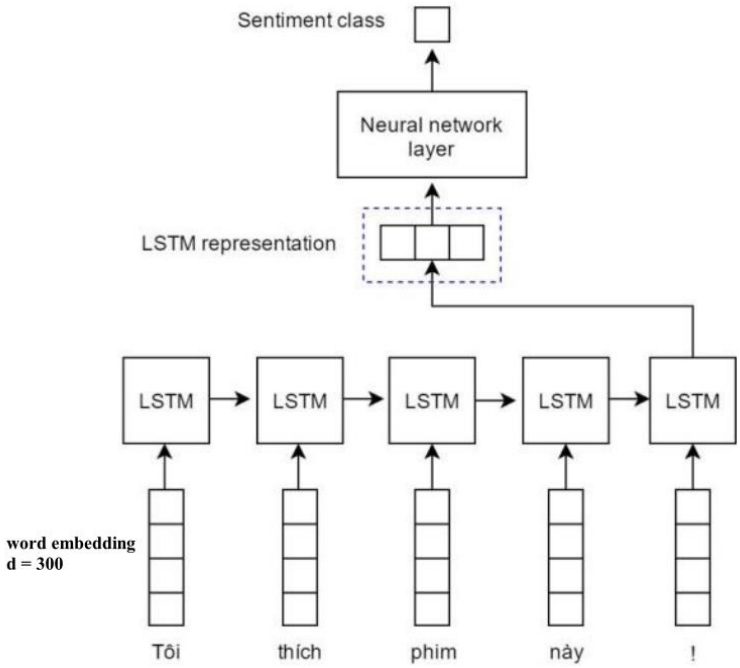

Figure 1 Basic LSTM

We use LSTM with attention model in our experiments.

And we stack two LSTMs together to form a stacked LSTM, with second LSTM taking in outputs of the first LSTM and computing the final results.

Attention Implementation:

$$Q = XW_Q$$
$$K = W_K$$
$$A = softmax(QK)$$
$$V = XW_V$$
$$ouput = Linear(VA)$$

Attention Module we implement is a little bit different from the traditional one, but it still works well with the same effect.

So the whole model is showed below:

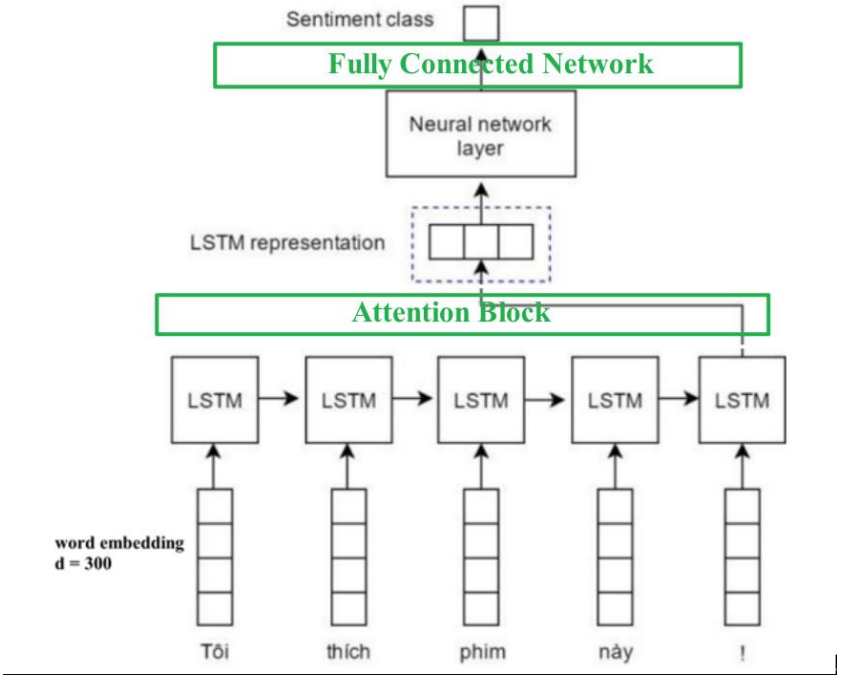

Figure 2 LSTM with attention

1.3 Train & Test

We use instructions in cmd.sh and use the control variables method to conduct training and testing experiments.

2 Experiments

Description: Since the hyerparameters of each network is infinite, we select a few that are representative to illustrate the results.

2.1 Hyerparameters and Results

MLP containing one hidden layer with 128 units. Batch size = 64 SGD with Momentum:0.9
Model 1: LSTM with Attention
Mode1 2: LSTM without Attention
Mode1 3: LSTM with Attention, but not using pre-trained word embeddings

Table 2-1 Model performance with different hyerparameters

Tag	Model	Num-layers	Hidden_dim	Learning rate	Weight_decay	Dropout	Optimizer	Max epoch	Test Acc
V1.1	1	1	128	1e-3	1e-4	0.5	Adam	10	40.62%
V1.2	1	1	256	1e-3	1e-4	0.5	Adam	10	40.37%

V1.3	1	1	384	1e-3	1e-4	0.5	Adam	10	40.96%
V1.4	1	1	512	1e-3	1e-4	0.5	Adam	10	40.10%
V1.5	3	1	256	1e-3	1e-4	0.5	Adam	10	37.72%
V1.6	1	2	256	1e-3	1e-4	0.5	Adam	10	39.45%
V1.7	1	1	128	1e-3	1e-4	0.2	Adam	10	40.45%
V2.1	1	1	128	1e-3	1e-4	0.5	Adam	2	45.66%
V2.2	1	1	256	1e-3	1e-4	0.5	Adam	2	43.70%
V2.3	1	1	384	1e-3	1e-4	0.5	Adam	2	47.34%
V2.4	1	1	512	1e-3	1e-4	0.5	Adam	2	45.78%
V2.5	3	1	256	1e-3	1e-4	0.5	Adam	2	36.59%
V2.6	1	2	256	1e-3	1e-4	0.5	Adam	2	44.90%
V2.7	1	1	128	1e-3	1e-4	0.2	Adam	2	46.27%
V3.1	1	1	128	1e-3	1e-4	0.5	Adam	1	43.99%
V3.2	1	1	256	1e-3	1e-4	0.5	Adam	1	44.85%
V3.3	1	1	384	1e-3	1e-4	0.5	Adam	1	43.77%
V3.4	1	1	512	1e-3	1e-4	0.5	Adam	1	44.98%
V3.5	3	1	256	1e-3	1e-4	0.5	Adam	1	34.67%
V3.6	1	2	256	1e-3	1e-4	0.5	Adam	1	42.02%
V3.7	1	1	128	1e-3	1e-4	0.2	Adam	1	40.22%
V4.1	1	2	256	1e-3	1e-4	0.5	SGD	2	23.12%
V4.2	1	2	256	5e-3	1e-4	0.2	Adam	2	39.14%
V4.3	1	2	256	5e-4	1e-4	0.2	Adam	2	43.07%
V4.4	1	2	256	1e-2	1e-4	0.2	Adam	2	41.13%
V4.6	2	1	384	1e-3	1e-4	0.5	Adam	2	23.12%
V4.7	2	1	384	1e-3	1e-4	0.5	Adam	10	23.12%

2. 2 Plot

Since all the loss curve and accuracy curve are similar with each other, we select a few that are representative.

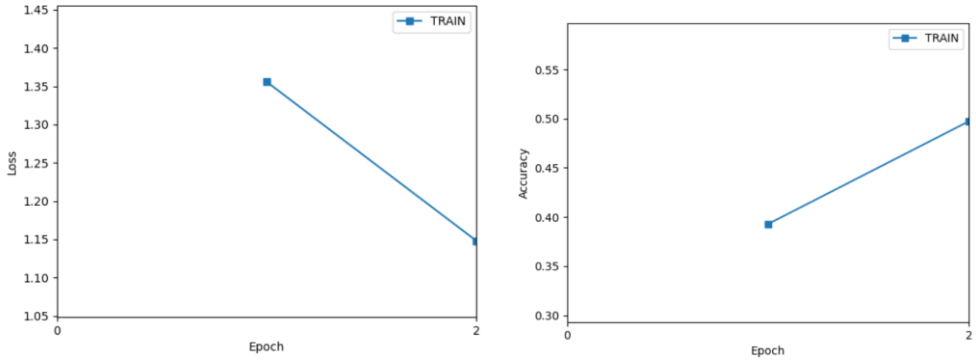

Figure 2-1 Training Loss and Accuracy v2.3

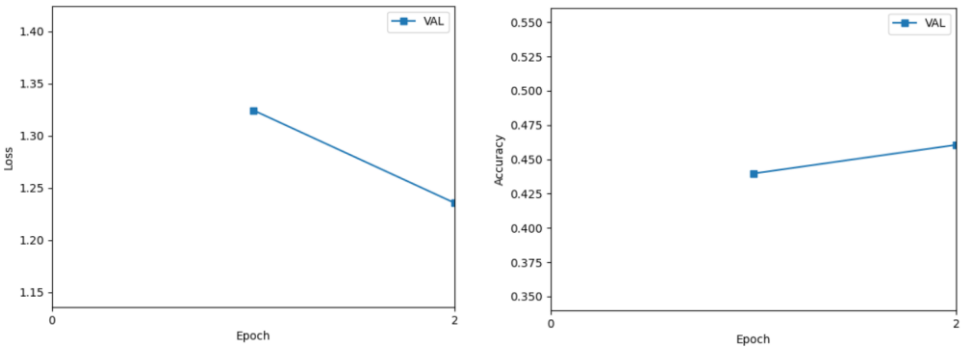

Figure 2-2 Validating Loss and Accuracy v2.3

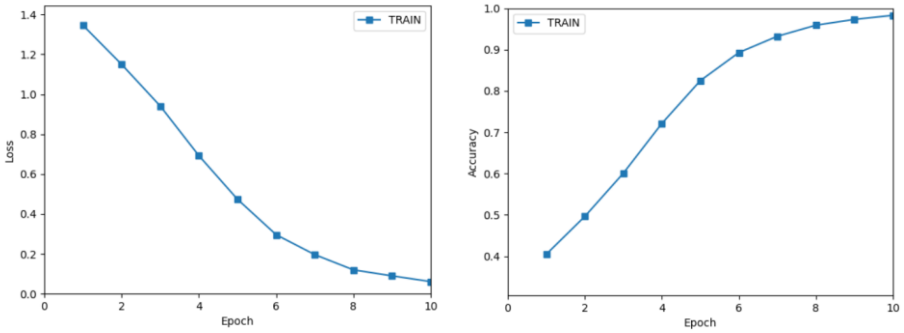

Figure 2-3 Training Loss and Accuracy v1.3

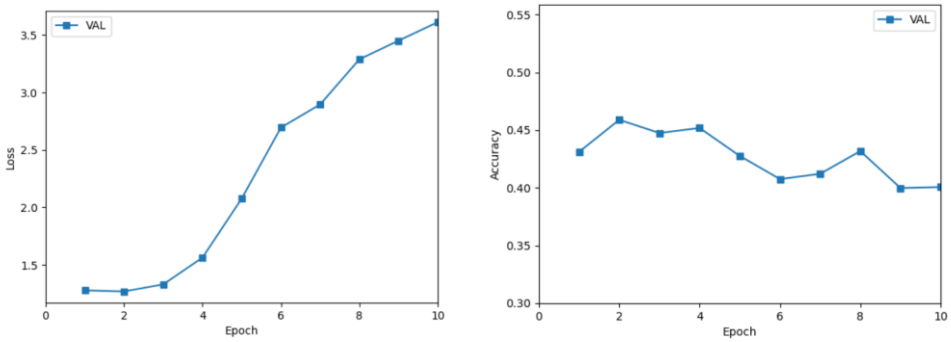

Figure 2-4 Validating Loss and Accuracy v1.3

2.3 Analysis

Fixed Parameters

- **Batch Size**

From hw2 we know that batch size only has slight influence on our experiment so we keep the batch size fixed at 64.

- **Weight decay**

In our experiment, we fix weight decay at 0.0001. Since the dataset is small and model has limited parameters, weight decay term doesn't play a key role in our accuracy.

- **Criterion**

Since we are solving a classification problem, so we fix the criterion as cross entropy loss.

Variable Parameters and Analysis

- **Model**

(1) LSTM with Attention

(2) LSTM without Attention

(3) LSTM with Attention, but not using pre-trained word embeddings

We can see that LSTM with Attention outperform that without Attention and using pre-trained word embeddings is significant according to the experiments

- **Dataset**

SST-5.

- **Max Epoch**

In the experiments, we set 1, 2, 10 epochs in different experiments. We find that the epochs play a key role in results.

According to the experiments and the curve of loss and accuracy, the model get a better performance on test when the epochs are small. Because with more epochs training loss is small and training accuracy is high, but it overfits badly. See figure 2-3 and figure 2-4

- **Learning Rate**

In the experiments, we set 5e-4, 1e-3, 5e-3, 1e-2 in different experiments.

We can find that the models perform well with small learning rate i.e., 0.001 and underperform with other learning rate, especially the large learning rate. The reason may be if the learning rate is large, the model may miss the optimal both locally and globally.

According to the results, in these experiments, we should choose a small learning rate with more epochs.

- **Optimizer**

In the experiments, we select SGD with momentum or Adam.

When other hyerparameters are fixed, Training with Adam always outperform that with SGD.

According to the results, we should select Adam in most cases.

3 Acknowledgment

Thanks for the homework! It's nice to practice what I have learned, especially the implementation of LSTM models using pytorch. And I have learned how to construct a project using

structured file organization. Thanks for checking! Best wishes.