Report

COMS4995 005 hw2
Huibo Zhao hz2480
Tingmeng Li tl2758

Python 3.6
Part1 Pytorch
Part2 Tensorflow, keras

1.1

Architecture:

The order are represented as sequential order.

Convolution layer with input channel to be 3, output channel to be 10, kernel size to be 3 and padding to be 1.
Rectified linear unit function.
Max pooling layer with input to be 2 and kernel size to be 2.
Convolution layer with input channel to be 10, output channel to be 32, kernel size to be 3 and padding to be 1.
Rectified linear unit function.
Max pooling layer with input to be 2 and kernel size to be 2.
Convolution layer with input channel to be 32, output channel to be 64, kernel size to be 3 and padding to be 1.
Rectified linear unit function.
Max pooling layer with input to be 2 and kernel size to be 2.
Reshape (flatten) layer that transform the data of size 1024 (64*4*4)
Applied linear transformation (fully connected layer) with input size of 1024 (64*4*4) and output size of 120.
Rectified linear unit function.

Applied linear transformation (fully connected layer) with input size of 120 and output size of 96.
Rectified linear unit function.
Applied linear transformation (fully connected layer) with input size of 96 and output size of 10.

Training Policy:

We set the batch size to be 4 and using stochastic gradient descent with learning rate initially set to be 0.001. We train the network with that learning rate for 10 epochs and then decrease the learning rate to be 0.00025 and train another 5 epochs. We didn't use any dropout for this part.

Results:

Final train accuracy: 0.944
Final validation accuracy: 0.702
Best validation accuracy: 0.709
Test accuracy: 0.7062

1.2

Architecture:

The order are represented as sequential order.

Convolution layer with input channel to be 3, output channel to be 192, kernel size to be 5 and padding to be 2.
Rectified linear unit function.
Batch normalization layer.
Max pooling layer with input to be 2 and kernel size to be 2.
Convolution layer with input channel to be 192, output channel to be 168, kernel size to be 3 and padding to be 2.
Rectified linear unit function.
Batch normalization layer.

Max pooling layer with input to be 2 and kernel size to be 2.
Convolution layer with input channel to be 168, output channel to be 96, kernel size to be 3 and padding to be 2.
Rectified linear unit function.
Batch normalization layer.
Max pooling layer with input to be 2 and kernel size to be 2.
Reshape (flatten) layer that transform the data of size 2400 (96*5*5)
Applied linear transformation (fully connected layer) with input size of 2400 (96*5*5) and output size of 120.
Rectified linear unit function.
Applied linear transformation (fully connected layer) with input size of 120 and output size of 96.
Rectified linear unit function.
Applied linear transformation (fully connected layer) with input size of 96 and output size of 10.

Training Policy:

We set the batch size to be 4 and using stochastic gradient descent with learning rate initially set to be 0.001. Different from part1.1, we set weight_decay, which acts as regularization, to be 0.0005 this time. In total, We train the network for 10 epochs but we modified the training policy: we start with learning rate 0.001 and we decrease the learning rate only if previous epoch accuracy / current epoch accuracy > 0.995, which we identify that this is the moment to alter the learning rate.

Final train accuracy: 0.946
Final validation accuracy: 0.750
Best validation accuracy: 0.750
Test accuracy: 0.7497

Part 2

We used Keras LSTM for this part. We implemented a character-level sequence-to-sequence model to process the input and generate output. The summary of the process is:

1. Turn the sentences into 3 Numpy arrays, containing a one-hot vectorization of German sentences as encoder_input_data, one-hot vectorization of English sentences as decoder_input_data, and another one-hot vectorization of English sentences with one time-stamp forward as decoder_target_data.
2. Train a basic LSTM model to predict decoder_target_data given encoder_input_data and decoder_input_data. Our model uses teacher forcing.
3. To decode a test sentence, we repeatedly:
   1) Encode input and retrieve initial decoder state.
   2) Run one step of the decoder with this initial state and a "start of sequence" token as target. The output will be the next target character.
   3) Repeat with the current target token and current states

Model summary:

```
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_1 (InputLayer)            (None, None, 122)    0
__________________________________________________________________________________________________
input_2 (InputLayer)            (None, None, 95)     0
__________________________________________________________________________________________________
lstm_1 (LSTM)                   [(None, 256), (None, 388096      input_1[0][0]
__________________________________________________________________________________________________
lstm_2 (LSTM)                   [(None, None, 256),  360448      input_2[0][0]
                                                                 lstm_1[0][1]
                                                                 lstm_1[0][2]
__________________________________________________________________________________________________
dense_1 (Dense)                 (None, None, 95)     24415       lstm_2[0][0]
==================================================================================================
Total params: 772,959
Trainable params: 772,959
Non-trainable params: 0
__________________________________________________________________________________________________
None
```

BLEU scores:

Average BLEU score is 0.330

5 best translated sentences:

Translated sentence: Tom is a bad learn.
Expected output: Tom is awesome.
BLUE score: 0.669

Translated sentence: I want to see you anymore.
Expected output: I don’t have time for you.
BLEU score: 0.639

Translated sentence: It’s a book at the door.
Expected output:Is that a new perfume?
BLEU score: 0.639

Translated sentence: It’s a bad look at the door.
Expected output: We are sorry we can’t help you.
BLEU score: 0.615

Translated sentence: Don’t you have a car for you.
Expected output:They’re late, as usual.
BLEU score: 0.615