2019 Spring COM526000 Deep Learning - Homework 4

Recurrent Neural Network

Due: June 13, 2019

INSTRUCTIONS

1. **High-level APIs are forbidden in this homework, such as *Keras, TFlearn, slim, pytorch*.**
2. In problem 1, the preprocessed data can be obtained by running the code in *hw4_1_preprocess.py*, which will download the dataset and split it into training set (25,000 reviews) and testing set (25,000 reviews). You can further form a validation set from training set if needed. Each review is clipped (or padded with zero) to 120 words long. Sentences are split into words and those words are mapped to unique integer indices according to a dictionary (built for 10,000 most frequently used words).
3. In problem 2, you need to preprocess the data by yourself. The process is similar as the case in problem 1. **Notice that you need to use utf-8 (or utf-8-sig) encoding while loading/saving *.txt* files.**
4. You have to provide the source codes and the result of independent testing set in problem 2 (*hw4_1_StudentID.py, hw4_2_StudentID.py, test_StudentID.txt*). Answer the questions stated in the problems in your report (*hw4_StudentID.pdf*) .
5. Compress all required files into *hw4_StudentID.zip* for submission (to iLMS).
6. **You should write your own codes independently. Plagarism is strictly prohibitted.**

PROBLEMS

1. (60%) **Gated recurrent neural network**
   Recurrent structures are useful while dealing with sequential data. In this problem, we will train a model to classify movie reviews in **IMDb dataset** into good reviews (label 1) and bad reviews (label 0). We treat each review as a sample in the dataset with 120 "time points" (words) and use them as inputs.

   (a) (5%) Before feeding samples into the model, you have to further transform the words (now they are encoded as integer indices) to vector representations. We call this procedure **Embedding**. Explain why do we need embedding.

   Now we have to choose which recurrent model to apply. Gated models such as Gated Recurrent Unit (**GRU**) and Long Short-Term Memory (**LSTM**) are popular choices.

   (b) (5%) Explain the idea of gated models and the main differences between GRU and LSTM.

   (c) (10%) Train a vanilla RNN (simple RNN) first to do the binary classification task. Specify the model structure in your report clearly (don't forget the embedding layer!). You have to evaluate the performance by the test accuracy. Also, you have to plot learning curve as in previous homeworks, receiver operating characteristic curve (**ROC**, as shown in Figure 1) and precision-recall curve (**PRC**, as shown in Figure 2) with their area-under-curve (**AUROC** and **AUPRC**). These two metrics are useful while evaluating model performance.

   (d) (5%) Based on the definition of ROC, explain why random guess forms the red dotted line in Figure 1.

   (e) (5%) What curve would random guess form in the PRC case?

   (f) (20%) Repeat (c) with GRU and LSTM.

   (g) (10%) Change the maximum length of each review from 120 words to 256 words in (c) and (f). State your observations and comments.

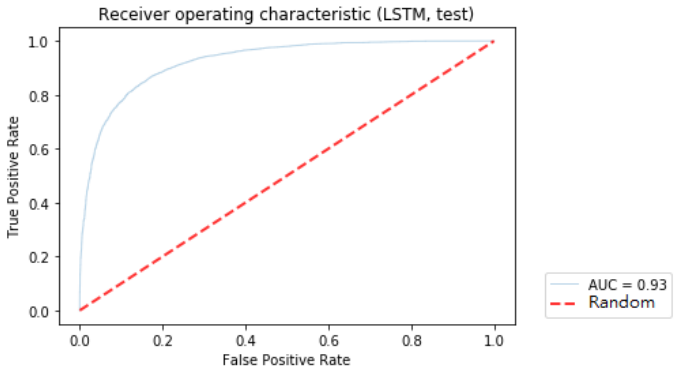

Figure 1: ROC curve.

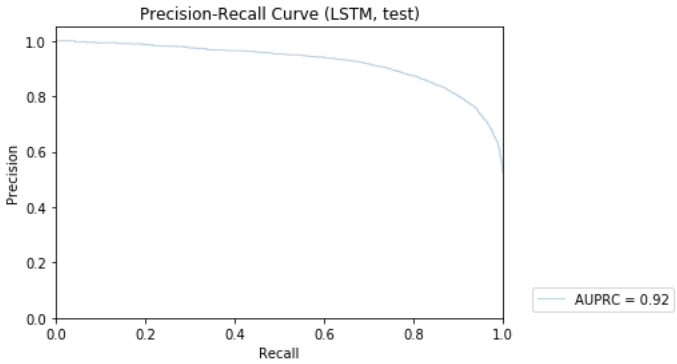

Figure 2: Precision-recall curve.

2. (40%) **Sequence-to-sequence learning**
   In problem 1, our model maps fixed-length input sequence (i.e. 120 words) to fixed-length output sequence (i.e. 0/1 label). When dealing with problems whose input and output sequence length are both variables, we will need sequence-to-sequence learning structure as shown in Figure 3 (for more details, please refer to chapter 10.4 in the textbook).
   In this problem, we would like to run a **machine translation** task (from English to French) using sequence-to-sequence learning. The contents of attached datasets (*en.txt* and *fr.txt*) are paired English/French sentences. We should use *en.txt* as input (source) and *fr.txt* as output (target) to train/test our model. Notice that each sentence in the dataset should be treated as a sample with variable sequence length (i.e. variable number of words).

   (a) (30%) Train a sequence-to-sequence model to accomplish the translation task. Specify how does your model work (e.g. how do you preprocess the sentences in the dataset, how do you design your encoder/decoder, what technique do you use to enhance the performance, etc.) and summarize the model structure. Report your test accuracy (state your definition for sentence accuracy clearly) and plot the learning curve. Show some translation results as examples of your work in report (e.g. as shown in Figure.4).

   (b) (10%) Use *test.txt* as input (source) to your trained model and save the translation result as *test_StudentID.txt*. **Make sure that you save the .txt file with utf-8 encoding.**

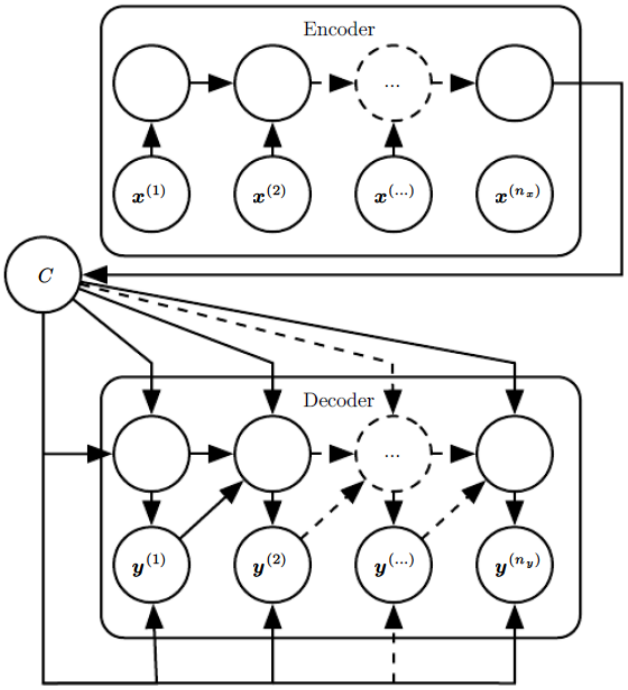

Figure 3: Sequence-to-sequence structure.
(*Deep Learning*, Ian.Goodfellow et al.)

```
Source (English)
  Word Indices:      [145, 65, 95, 198, 5, 110, 124, 223, 41, 162, 95, 84, 108, 66, 200]
  English Words: ['new', 'jersey', 'is', 'sometimes', 'quiet', 'during', 'autumn', ',', 'and', 'it', 'is', 'snowy', 'in',
'april', '.']

Translation (French)
  Word Indices:      [142, 204, 356, 257, 11, 178, 260, 98, 91, 254, 238, 356, 329, 70, 344, 161, 1]
  French Words: new jersey est parfois calme pendant l' automne , et il est neigeux en avril . <EOS>
```

Figure 4: Translation results demonstration.