Spring 2023
Introduction to Artificial Intelligence

Homework 2

April. 28, 2023

Introduction

The goal of this programming assignment is to 1) Try how to preprocess a text dataset for sentiment classification, 2) implement a old-school basic language model, **Bi-Gram**, 3) understand the concept of classical neural network for text, **RNN**, 4) get a taste of a complete supervised learning process, and 5) compare the results and performance with SOTA method (**DistilBert** in this assignment). Please make sure you understand the concept of n-gram, recurrent neural network and BERT before working on this assignment.

When you do this assignment, please:

- Finish codes between **#Begin your code** and **#End your code**. Search for **TO-DO** to find the part you need to implement
- Feel free to add additional functions, but preventing the existing code from modifying
- Implement the required function by yourself instead of using existing methods
- Do not import additional packages, packages imported in the provided code should be enough to finish the homework. However, please feel free to ask at teams if you find it necessary to use additional packages.
- You can change run.sh and run_colab.ipynb.
- Only at the preprocess.py part can you import additional packages you need to implement the preprocessing function.
- Use Python 3.10 in this assignment

Data

The dataset in this assignment is the IMDB dataset, which consists of 50k movie reviews. You can find it in the "data" folder. The example and statistics of the dataset are as below.

	Train	Test
Positive	20,000	5,000
Negative	20,000	5,000
Total	40,000	10,000

Review	Sentiment
Saw the move while in Paris in May 2006 ... It is important to have some understanding of the French society of Today to really enjoy the humor of this movie…	Positive
Scary Movie 2 was a grave disappointment. Simply referencing movies, like Mission Impossible 2 does not lead to comedy …Overall, a very poorly done movie.	Negative

Requirements

Code (60%)

Part 0: Implement a different preprocessing method (10%)

- We provide the code of removing stopwords in preprocess.py.
- You can implement your preprocessing method in the "preprocessing_function" function (e.g., stemming).
- Briefly explain the two methods(the provided and yours) in the report.
- You can use the existing list you want (e.g., stopwords provided by nltk).
- You can import packages needed in preprocess.py.
- Hint: Try to observe the dataset and you will find there are some symbols that can be eliminated.

Part 1: Implement the bi-gram language model (20%)

- Implement the bi-gram model in the "get_ngram" function in ngram.py.
- Implement the bi-gram model in the "train_sentiment" function in ngram.py.
- You may need to modify the outputs in the "train" function, but you need to keep self.model at least.
- You can run ngram.py solely for checking the correctness of your implementation with the simple input. We also provide an explanation in the code for reference only.

Part 2: Implement BERT model (15%)

- In Part 2-1, construct a classifier that can classify based on the output of a pretrained BERT model in bert.py. Determine the number of linear layers and dropout layers by yourself, feel free to stack layers in this link.
- In Part 2-2, you should implement the training function in main.py. You can refer to this tutorial which introduces how to use pytorch to completely build and train the model.

- Feel free to change the configuration in second_part function which is in main.py. You can change the parameters in the optimizer, but you need to store all your changes in these two configuration dict (rnn_config and bert_config).
- The score of this part is:
    a. Test F1-score > 94%: 100%
    b. Test F1-score > 93%: 90%
    c. Test F1-score > 90%: 80%
- Note that when measuring the F1-score of your model, you can only use the simplest text preprocessing method, remove_stopwords.

Part 3:Implement LSTM model (15%)

- In part 3-1, you should simply pass any parameters you want to initialize your model in rnn.py. Of course, if you want to hardcode all the parameters in the definition of the YourModel class, then you don't need to pass any parameters here.
- In part 3-2 and part 3-3, you should build your recurrent neural network by subclassing nn.Module. The code you need to implement is __init__() and forward().
- The model you have to build is the RNN model, please use at least one kind of recurrent layer defined in this page.
- To enhance your efficiency, please read the following tutorial before starting to implement this task:
    1. Build Neural Network from Scratch in PyTorch
    2. Basic concept of RNN and LSTM
- The score of this part is:
    - Test F1-score > 85%: 100%
    - Test F1-score > 80%: 90%
    - Test F1-score > 75%: 80%
    - Test F1-score > 70%: 60%
- Note that when measuring the F1-score of your model, you can only use the simplest text preprocessing method, remove_stopwords.

Report (30%)

- The goal of writing a report is to learn to analyze the questions and your observations. We rate your report mainly based on the discussions and analysis. You **don't need to paste your code and explanation of your code** in this assignment, but you will only get some scores (or even zero scores) if your code cannot run successfully as the requirements.
- You are required to submit a report and it can be written in Chinese or English.
- Save the report as a **.pdf** file
- The report should **at least** include the following items.
    - **For Part 0:**

        - Briefly explain the method you implemented and give an example (such as the E.g in the remove_stopwords function) in the report.
    - **For Part 1:**
        - Briefly explain the concept of perplexity in report and discuss how it will be influenced.
        - Screenshot the outputs and tell your observations about the differences in the perplexity caused by the preprocessing methods(1. without preprocess 2. with remove stopwords 3. with your method).
    - **For Part 2:**
        - Briefly explain the two pre-training steps in BERT. Self-supervised Learning by Hung-yi Lee tutorial
        - Briefly explain four different BERT application scenarios
        - Discuss the difference between BERT and distilBERT?
        - Screenshot the required test F1-score.
        - (BONUS 5%) Explain the relation of the Transformer and BERT and the core of the Transformer. Application of deep learning_7-1 to 9-1 by Vivian
    - **For Part 3:**
        - Briefly explain the difference between vanilla RNN and LSTM.
        - Please explain the meaning of each dimension of the input and output for each layer in the model. For example, the first dimension of input for LSTM is batch size.
        - Screenshot the required test F1-score.
    - **Discussion:**
        - Discuss the innovation of the NLP field and your thoughts of why the technique is evolving from ngram -> LSTM -> BERT.
        - Describe problems you meet and how you solve them.

Demo (10%)

- It will be a simple demo, we will announce the details for the demo after the homework deadline, your **points from the coding part will be evaluated based on the demo.** However, we might still run your code if there's any problem.
- Please bring your own device, and show the results you executed on colab or terminal on your own device. (The pictures are example outputs you should "at least" contain on colab or terminal. Your training epochs or outputs may be different from the example, but the information of f1 score, precision and recall must be kept.)
    - For Part 1: The results of training on data preprocessed by remove stopwords and the preprocessing method you wrote.

- For Part 3: The results of training on data preprocessed by remove stopwords.

- For Part 2: The result of training on BERT preprocessed by remove stopwords.

```
100% 5000/5000 [29:37<00:00,  2.81it/s]
Epoch: 0, F1 score: 0.9318, Precision: 0.9321, Recall: 0.9318, Loss: 0.2294
100% 1/1 [31:23<00:00, 1883.64s/it]
```

- The 10% of the demo credit will be given by
  - 1% : Attending the demo.
  - 9% : Three simple questions about your code or the tasks.

- The purpose of the demo is for checking you write your own code and the code is executable.

Discussion

TAs had opened a channel **HW2** 討論區 on Microsoft Teams of the course, you can ask questions about the homework in the channel. TAs will answer questions in the channel as soon as possible.

Discussion rules:

1. Do not ask for the answer to the homework.
2. Check if someone has asked the question you have before asking.
3. We encourage you to answer other students' questions, but again, do not give the answer to the homework. Reply to the messages to answer questions.
4. Since we have this discussion channel, do not send emails to ask questions about the homework unless the questions are personal and you do not want to ask publicly.

Submission

1. **The deadline for this homework is 4/17 (Mon.) 23:59:00.**
2. Please submit one zip file that contains all the Python code files (i.e., main.py, preprocess.py, ngram.py, bert.py...), and report with the format hw2_{StudentID}.pdf (e.g., hw2_111123456.pdf). Note that you should not include the dataset in your zip file.
3. Submit the zip file with the filename of hw2_{StudentID}.zip (e.g., hw2_111123456.zip).
4. Late submission leads to a score of (original score)*0.85^{days}, for example, if you submit your homework right after the deadline, you will get (original score)*0.85 points.

5. **We only accept one zip file**, wrong format, or naming format cause -10 points to your score (after considering late submission penalty).
6. TA will run your code if necessary.
7. **Plagiarism** is not allowed! You will get a 0 if we find that.
8. If there is anything you are not sure about submission, ask in the discussion forum.

Files

File name	Description
main.py	The main code of this assignment.
preprocess.py	The code preprocessing the text.
ngram.py	The code for the n-gram model. You can run this code solely for unit tests.
bert.py	The code with DistilBert for sentiment classification.
rnn.py	The code for the RNN model.
data folder	Train, test datasets.

References

- Introduction to Bert by Prof. Hung-Yi Lee.
- [DLHLP 2020] Language Modeling by Prof. Hung-Yi Lee.
- Language Modeling post by Lena Voita.
- Beyond Computing 講座 (2022/04/09)