Homework assignment for the "Deep Learning for NLP" module*

Artem Chernodub, Ph.D.
Ukrainian Catholic University, Faculty of Applied Sciences
Grammarly
chernodub@ucu.edu.ua

December 11, 2020

1 Introduction

The homework assignment is dedicated to *sequence labeling* or *sequence tagging* problem. Unlike of similar tasks like Named Entity Recognition or Part-Of-Speech tagging where you classify each word-level token in the text, now you need to classify each sentence in a paragraph.

2 The problem

You've got training data containing source text files and corresponding text files with labels: {*train, dev, test*}.{*src, lbl*}. In source text files, each line is a paragraph which contains N sentences, one sentence from the paragraph may be capitalized. Lines in text files with labels contains sentence-level labels which highlight capitalized sentences. For example:

Source text file line:

The quick brown fox jumps right over the lazy dog. WAKE UP, NEO! It's time to work.

Label text file line:

O C O

*Yeah, it's time to start working on it right now.

3 The method

Homework's solution is exploiting HuggingFace Transformers library. You need to leverage RobertaForTokenClassification class for sentence-level classification. To perform this, you may apply two tricks. Firstly, add special Transformer's *[CLS]* token to the end of each sentence, i.e.

> The quick brown fox jumps right over the lazy dog. [CLS] WAKE UP, NEO! [CLS] It's time to work. [CLS]

Secondly, you need to classify these *[CLS]* tokens only. To ignore certain tokens, please, use magic PyTorch number -100 as target training labels for ignored tokens.

4 Tasks

Please, add the following modifications to the homework's template code:

1. Data analysis. Calculate frequencies for train/dev/test data, plot distributions for:
   (a) number of sentences in paragraphs;
   (b) labels depending on sentence's orders in paragraphs (5% of score).
2. Modify *UpperSentDetectorDataset* class in *dataset.py*. This class serves data storage and preprocessing. Please, add code, which:
   (a) loads paragraphs from source text files;
   (b) tokenizes text using *RobertaTokenizer* tokenizer;
   (c) splits paragraphs to sentences using NLTK-based *split_sentences* function from *utils.py*;
   (d) adds *[CLS]* token to each sentence;
   (e) if labels text file is available, load it and convert sentence-level labels to token-level ones;
   (f) provides data access to optimizer (30% of score).
3. Modify *UpperSentDetectorModel* class in *model.py*. This class implements neural network model for tagging the sentences. Add forward pass to calculate output labels and loss function for training the model (20% of score).

4. Modify main training cycle in *train.py*:
   (a) add *Adam* optimizer with learning rate $lr = 10^{(-6)}$;
   (b) perform model's forward pass, loss calculation, make weight's updates;
   (c) implement early stopping based on "save best" method: save model which shows best $F_{0.5}$ score on *dev* dataset. Please, use *get_f_score* function from *utils.py*.

   If everything is done properly, you'll easily obtain $F_{0.5} > 0.99$ after 5 epochs (25% of score).
5. Implement the calculation of sentence-level $F_{0.5}$ / F_1 scores / *True Positive / True Negative / False Positive / False Negative* rates from scratch. Don't use standard functions from scikit-learn or similar external packages. See *get_tp_tn_fp_fn* fucntion in *utils.py* (10% of score).
6. Find cases from test set where **your model** had failed and visualize them in an impressive manner. Find ten random parapraphs from news websites, capitalize one sentence in half of them, run through your system and visualize (10% of score).

5 Deadlines

The deadline is 30/12/2020, 9:00 AM CET 9:00 AM CET. The penalty for missing the deadline: 50% for up to one week, 100% for more than one week. Submit your developed source code and links of the trained models. I must be capable to run and evaluate your models using *predict.py* and *evaluate.py* scripts. For splitting sentences, use only NLTK-based *split_sentences* function from *utils.py*.

6 Installation notes

You need *python 3.6* installed and Linux/Ubuntu/MacOS. Installation for the homework's code:

```
pip install nltk==3.5 pathlib==1.0.1 torch==1.7.0 transformers==4.0.0 sklearn
python -c "import nltk;nltk.download('punkt')"
```

Good luck!