AttentiveCLS Pooler

Team 1: Seungil Lee, Jongchan Park, Eugene Seo, and Dohyung Kim

November 4, 2021

1 Introduction

BERT [1] has become a standard architecture for NLP research ever since it was published. BERT computes the representation for every token, but it uses the output representation of the special token `[CLS]`, for sentence-level tasks (e.g., sentiment analysis). However, various strategy can be applied to get sentence-level representation, so we are going to design new method and try some of them in this homework.

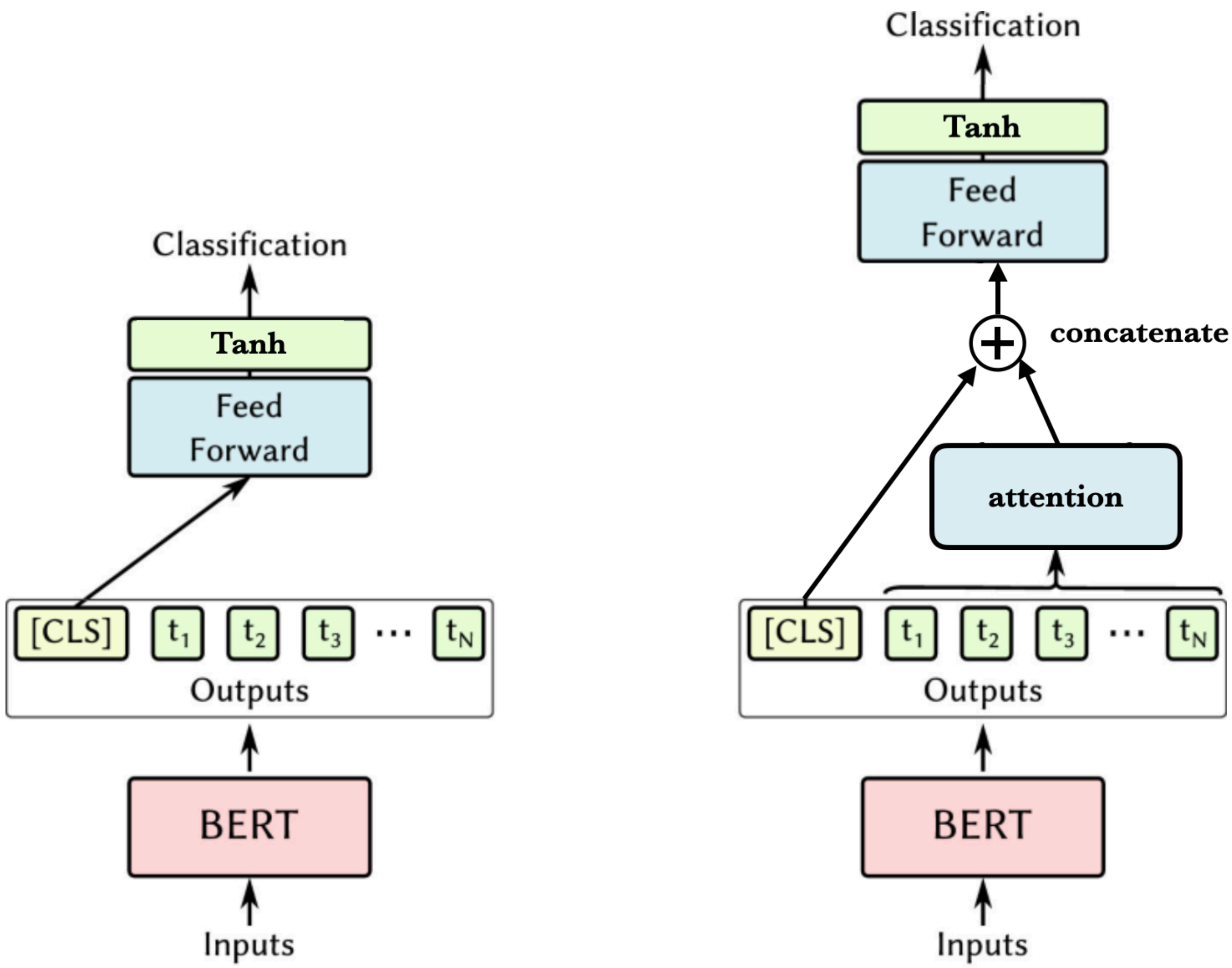
We have designed new pooler named `AttentiveCLS` pooler and evaluated its performance on CoLa, MNLI and MRPC tasks, which are the representatives of sentence-level tasks. The results were compared with `MeanMaxTokens` pooler, which is suggested in the homework description, also with the original `BERTPooler` in `huggingface` library (`https://huggingface.co/`).

2 AttentiveCLS Pooler

Though original BERT pooler simply adopts last output of `[CLS]` token, we tried to exploit information from other tokens as well. Nowadays, attention mechanism is widely used to get the importance of the given sequence, so we added a attention pooling layer on the top of the tokens other than `[CLS]` token. Then, the output of this attention layer is concatenated with the last output of `[CLS]` token.

We also apply linear transformation with $W \in \mathbb{R}^{H \times 2H}$ and tanh activation just like as the `BERTPooler` in `huggingface` implementation.

The `BERTPooler` in `huggingface` implementation applies linear transformation with $W \in \mathbb{R}^{H \times H}$ and tanh activation (See `BertPooler` class). Similarly, we apply linear transformation with $W_{\mathrm{MMT}} \in \mathbb{R}^{H \times 2H}$ and tanh activation.


Classification
Tanh
Feed Forward
[CLS]
t_1
t_2
t_3
t_N
Outputs
BERT
Inputs
Classification
Tanh
Feed Forward
concatenate
attention
[CLS]
t_1
t_2
t_3
t_N
Outputs
BERT
Inputs

(a) (b)

The tasks in this homework are as follows:

1. Implement the `MeanMaxTokens` pooler (See `MeanMaxTokensBertPooler` class in `bert_poolers.py`).
2. Implement your own BERT pooler (See `MyBertPooler` class) and describe its architecture and rationale in your report. It does not have to be completely novel.
3. Choose one dataset in GLUE [2], and compare the test performance of three poolers (See `run_glue.py`).
4. Discuss the result. Negative results are fine, the point is how you interpret and explain it.

3 Experiment

The files you should submit are

1. Your team's `bert_poolers_{team_no}.py` (e.g, `bert_poolers_0.py`).
2. Your team's two-page `report_{team_no}.pdf` (e.g., `report_0.pdf`). Use this LaTeX file as a template, and do not change style attributes in this file. References are not included in the page-limit.

4 Experiment

Comprehensive evaluation based on clarity, validity, and interestingness. You will get zero points if you violate academic integrity (e.g., plagiarism and data manipulation).

5 Result

References

[1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[2] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations*, 2019.