Sentiment Analysis Report

- 选题为情感分析。
- 模型为[Funnel-Transformer: Filtering out Sequential Redundancy for Efficient Language Processing]论文所提出的Funnel Transformer.对其简介可见repo中的PPT。其最大特点是每经过一个block(几层Transformer)之后，就会pooling序列到一半长度。
- 采用sst2数据集，用`funnel/small`配上论文的设置复现了结果之余，还做了实验检验自己关于模型结构的想法。

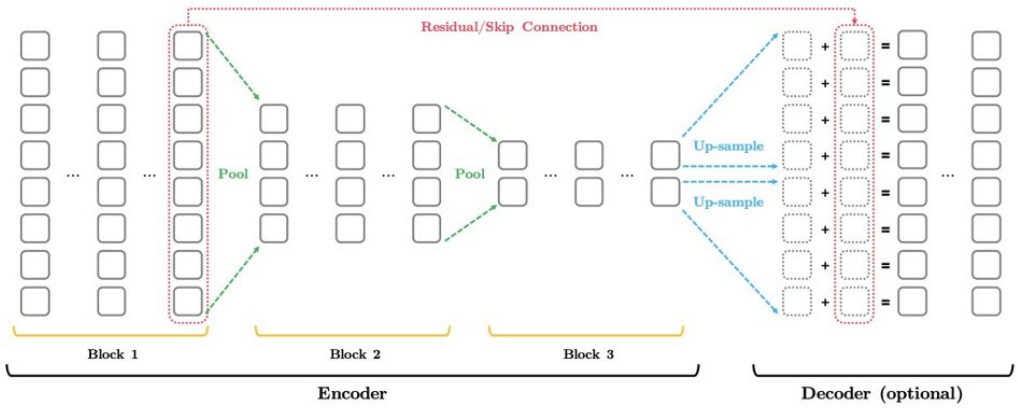

Figure 1: High-level visualization of the proposed Funnel-Transformer.

- 本实验的代码已经上传到GitHub [li1117heex/pyramidformer (github.com)]。
- 代码用的是Transformers的框架，预训练模型也来自于Transformers。

核心实验

作者的参数和实验结果为下表中对应的'sst2'列:

Model size	CoLA	SST-2	MRPC	STS-B	QQP	MNLI	QNLI	RTE	GLUE-AVG
L24H1024	66.5	94.3	92.8/90.0	91.5	89.6/92.2	89.4	94.1	84.5	87.8
B10-10-10	68.6	95.0	93.0/90.0	91.0	88.9/91.7	89.1	93.6	84.5	**87.9**
B8-8-8	66.6	94.8	92.6/89.7	90.7	88.8/91.7	89.0	93.6	82.1	87.3
L12H768	64.3	93.1	92.1/89.2	90.8	88.7/91.7	86.4	92.1	75.4	85.4
B6-6-6	64.3	94.2	92.8/89.7	90.1	88.7/91.6	87.4	92.5	78.3	**86.0**
B6-3x2-3x2	63.9	94.2	93.0/90.2	89.5	88.4/91.4	87.0	92.2	77.6	85.7
B4-4-4	62.8	93.6	92.5/89.2	89.2	88.4/91.3	86.0	91.6	74.3	84.8
L6H768	62.1	91.1	90.8/86.8	88.9	88.2/91.3	83.9	89.7	66.7	82.6
B3-4-4	59.0	93.1	90.8/87.5	88.7	88.1/91.0	85.8	91.1	72.5	**83.6**

Model size	IMDB	AG	DBpedia	Yelp2	Yelp5	Amazon2	Amazon5	FLOPs	#Params
L24H1024	4.724	**5.053**	0.653	1.874	28.84	2.425	32.85	1.00x	1.00x
B10-10-10	**4.324**	5.250	**0.639**	**1.789**	**28.68**	**2.419**	**32.72**	0.73x	1.22x
B8-8-8	4.364	5.408	0.651	1.729	28.76	2.447	32.85	0.58x	1.00x
L12H768	5.248	5.355	0.657	1.953	29.24	2.596	33.04	1.00x	1.00x
B6-6-6	**4.792**	**5.237**	**0.650**	**1.850**	**28.73**	**2.499**	**32.79**	0.88x	1.39x
B6-3x2-3x2	4.924	5.342	0.671	1.913	29.00	2.523	32.85	0.88x	1.00x
B4-4-4	5.152	5.382	0.659	2.032	29.33	2.566	33.03	0.58x	1.00x
L6H768	6.220	5.395	0.674	2.287	30.16	2.759	33.57	1.00x	1.00x
B3-4-4	**5.396**	**5.342**	**0.653**	**2.000**	**29.60**	**2.591**	**33.09**	1.00x	1.53x

Table 2: ELECTRA pretraining results at the base scale.

Hparam	**RTE**	**MRPC**	**STS-B**	**CoLA**	**SST-2**	**QNLI**	**MNLI**	**QQP**
Hidden dropout	0.1							
GeLU dropout	0.0							
Attention dropout	0.1							
Max sequence length	128							
Batch size	16	16	16	16	32	32	64	64
Number of epochs	10	10	10	10	5	3	3	5
Learning rate decay	Linear							
Weight decay	0.01							
Warmup proportion	0.1							
Adam epsilon	1e-6							

Hparam	**IMDB**	**AG**	**DBpedia**	**Yelp-2**	**Yelp-5**	**Amazon-2**	**Amazon-5**
Hidden dropout	0.1						
GeLU dropout	0.0						
Attention dropout	0.1						
Max sequence length	512	128	128	512	512	512	512
Batch size	32	32	64	128	128	128	128
Number of epochs	5	3	3	3	3	3	3
Learning rate decay	Linear						
Weight decay	0.01						
Warmup proportion	0.1						
Adam epsilon	1e-6						

Table 8: Hyper-parameters for finetuning on the GLUE benchmark and 7 text classification datasets.

我实验结果如下，基本对应。

loss	accuracy	F1
0.204	93.2	93.4

我的猜想

多加几个block会提升模型的表现，因为到后面的block的时候序列变得更短，信息会更加充分的汇集到用于分类的位于序列首位的[cls]token中。而时间消耗则基本不变，因为

在原本参数seq_len=128,模型有3个各4层的block,即`block_sizes=[4,4,4]`的情况下，最终序列长度为32。我分别对5，6，7个各4层block的模型设置进行了实验，结果如下：

block	layer per block	epoch	lr	loss	accuracy	F1
5	4	5	1e-6	0.553	74.1	77.0
6	4	5	1e-6	0.696	50.9	67.5
7	4	5	1e-6	0.693	50.9	67.5

（参数

非常不佳。甚至在6，7block时得到了recall=1的荒谬结果。

我觉得有2个可能原因：

1. 学习率太低
2. 模型层数过深

以下实验结果：

block	layer per block	epoch	lr	loss	accuracy	F1
7	2	5	1e-6	0.694	53.6	62.1
6	2	5	1e-6	0.666	60.8	60.0
7	4	5	1e-5	0.698	50.9	67.5
7	4	5	1e-4	0.698	50.9	67.5

还是很荒谬。

但是在2层block搭配`lr=1e-5`时，得到了不错的结果：

block	layer per block	epoch	lr	loss	accuracy	F1
7	2	5	1e-5	0.358	87.3	87.6
7	2	5	1e-4	0.698	50.9	67.5
7	2	5	5e-6	0.368	87.5	88.1

再减少block:

block	layer per block	epoch	lr	loss	accuracy	F1
6	2	5	1e-5	0.333	87.7	88.3
4	2	5	1e-5	0.356	88.1	88.6

和原来的模型表现只差几个点，时间也节省了不少，从10min降到了6min.

结论

- 增加block只有反效果
- 其实原本的模型每个block的层数减少一些也可以接受

不足

- 对于多出来的层，参数直接用原本最后一层复制过去。这也会影响新增的block的表现。如果能找到更好的初始化方法，或者训练更加充分会好一些。

结尾

这个模型我在微软实习的这段时间里曾经研究过，还曾经从头跑过pretrain,这次算实践了一下自己的idea.计算资源都来自微软服务器。