吊打BERT Large的小型预训练模型ELECTRA终于开源！真相却让人...

原创 rumor酱 夕小瑶的卖萌屋 3月12日

来自专辑

卖萌屋@自然语言处理

一只小狐狸带你解锁 **炼丹术&NLP** 秘籍

还记得去年写下《ELECTRA：超越BERT，19年最佳NLP预训练模型》时兴奋的心情，在我等到都快复工的时候，终于看到了它的身影和源码[1]：

Rank	Name	Model	URL	Score
1	Alibaba DAMO NLP	StructBERT		90.3
2	T5 Team - Google	T5		90.3
3	ERNIE Team - Baidu	ERNIE		90.1
4	Microsoft D365 AI & MSR AI & GATECH	MT-DNN-SMART		89.9
5	ELECTRA Team	ELECTRA-Large + Standard Tricks		89.4

才第五吗？没事，期望越大，失望越大

谷歌在github放出的预训练模型效果是这样的：

Released Models

We are initially releasing three pre-trained models:

Model	Layers	Hidden Size	Params	GLUE score	Download
ELECTRA-Small	12	256	14M	77.4	link
ELECTRA-Base	12	768	110M	82.7	link
ELECTRA-Large	24	1024	335M	85.2	link

燃！鹅！在论文中声称的效果却是这样的

Model	Train / Infer FLOPs	Speedup	Params	Train Time + Hardware	GLUE
ELMo	3.3e18 / 2.6e10	19x / 1.2x	96M	14d on 3 GTX 1080 GPUs	71.2
GPT	4.0e19 / 3.0e10	1.6x / 0.97x	117M	25d on 8 P6000 GPUs	78.8
BERT-Small	1.4e18 / 3.7e9	45x / 8x	14M	4d on 1 V100 GPU	74.7
BERT-Base	6.4e19 / 2.9e10	1x / 1x	110M	4d on 16 TPUv3s	82.2
DistilBERT	– / 1.4e10	– / 2x	66M	–	77.8
ELECTRA-Small	1.4e18 / 3.7e9	45x / 8x	14M	4D on 1 V100 GPU	79.0
ELECTRA-Base	6.4e19 / 2.9e10	1x / 1x	110M	4D on 16 TPUv3s	85.1

Table 1: Comparison of small models on the GLUE dev set. BERT-Small/Base are our implementation and use the same hyperparameters as ELECTRA-Small/Base. Infer FLOPs assumes single length-128 input. Training times should be taken with a grain of salt as they are for different hardware and with sometimes un-optimized code.

Model	Train FLOPs	Params	CoLA	SST	MRPC	STS	QQP	MNLI	QNLI	RTE	Avg.
BERT	1.9e20 (0.27x)	335M	60.6	93.2	88.0	90.0	91.3	86.6	92.3	70.4	84.0
XLNet	9.6e20 (1.3x)	360M	63.6	95.6	89.2	91.8	91.8	89.8	93.9	83.8	87.4
RoBERTa-100K	6.4e20 (0.90x)	356M	66.1	95.6	**91.4**	92.2	92.0	89.3	94.0	82.7	87.9
RoBERTa	3.2e21 (4.5x)	356M	68.0	**96.4**	90.9	**92.4**	92.2	90.2	**94.7**	86.6	88.9
BERT (ours)	7.1e20 (1x)	335M	67.0	95.9	89.1	91.2	91.5	89.6	93.5	79.5	87.2
ELECTRA	7.1e20 (1x)	335M	**69.3**	96.0	90.6	92.1	**92.4**	**90.5**	94.5	**86.8**	**89.0**
Test set results for models with standard single-task finetuning (no ensembling, task-specific tricks, etc.)											
BERT	1.9e20 (0.27x)	335M	60.5	94.9	89.3	86.5	89.3	86.7	92.7	70.1	83.8
SpanBERT	7.1e20 (1x)	335M	64.3	94.8	**90.9**	89.9	89.5	87.7	94.3	79.0	86.3
ELECTRA	7.1e20 (1x)	335M	**68.2**	**96.9**	89.6	**91.0**	**90.1**	**90.1**	**95.4**	**83.6**	**88.1**

Table 2: Comparison of large models on the GLUE dev and test sets. BERT dev results are from Clark et al. (2019). See Appendix B for some discussion on GLUE test set comparisons.

Github repo中官方的解释是精调的震荡比较大，他们测试了很多随机种子后取了中位数。

Expected Results

Here are expected results for ELECTRA on various tasks. Note that variance in fine-tuning can be quite large, so for some tasks you may see big fluctuations in scores when fine-tuning from the same checkpoint multiple times. The below scores show median performance over a large number of random seeds. ELECTRA-Small/Base/Large are our released models. ELECTRA-Small-OWT is the OpenWebText-trained model from above (it performs a bit worse than ELECTRA-Small due to being trained for less time and on a smaller dataset).

那么问题来了，为什么论文中没有取中位数呢(*￣m￣)?

虽然ELECTRA的思想仍是很惊艳的，但这样的结果不免让我们对原论文的数据产生质疑，对该质疑更详细的讨论见评论区置顶留言。

不过，有了word2vec和transformer的大家都懂的前车之鉴，我们这一次也理解为瑕不掩瑜了(*￣m￣)

回顾目前单模型的进展（只参考原论文数据）：

模型	GLUE dev	GLUE test
BERT-Base	-	79.6
StructBERT-Base	-	80.9
BERT-Large	-	82.1
SpanBERT-Large	-	82.8
StructBERT-Large	-	83.9
T5	-	86.18
XLNet-Large	87.4 (without WNLI)	89.47 (ensemble)
ELECTRA-Large	89.0	88.1
RoBERTa-Large	89.1	88.5 (ensemble)

如果按照github中给出的85.2，ELECTRA跟RoBERTa还是有很大差距的。那今天我们就来分析一下ELECTRA的优点和缺陷，如果对文中的观点有质疑，请在评论区一起讨论～

PS：不了解这个模型的同学可以先看第二章ELECTRA简介～

后台回复【electra】获取论文PDF噢~~

ELECTRA优缺点

通过自己的思考和知乎大佬们的提点[2]，ELECTRA主要有如下优点：

- **任务难度的提升（知乎@香侬科技）**

原始的MLM任务是随机进行mask，样本预测难度是不一样的，比如“夕小瑶的卖[MASK]屋”和“夕[MASK]瑶的卖萌屋”中，卖萌是比较常见的词语，而夕小瑶则是命名实体，没见过的话更难预测出来。生成器相当于对输入进行了筛选，使判别器的任务更难，从而学习到更好的表示

- **效率的提升（知乎@徐啸）**

运算上，判别器进行2分类而不是V分类，复杂度降低。参数利用上，判别器不需要对完整的数据分布进行建模，作者在实验中也发现越小的模型提升越明显，可见参数被更完整地训练了

- **Token自身信息的利用（知乎@Towser）**

做MLM的任务时，都是mask掉token自身，利用上下文对自己进行预测。而ELECTRA是同时利用自身和上下文预测，和NLU任务的Finetune阶段也比较一致

- **收敛速度**

从论文中的分析实验来看：

（1）ELECTRA 15%：让判别器只计算15% token上的损失

(2) Replace MLM：两个生成器。一个生成器替换被mask的token，另一个生成器用替换后的输入继续进行15%的MLM训练。这样可以消除这种pretrain-finetune之间的diff

(3) All-Tokens MLM：同Replace MLM，只不过BERT的目标函数变为预测所有的token，为了防止信息泄露加入了copy机制，以D的概率直接拷贝输入，以1-D的概率预测新token，相当于ELECTRA和BERT的结合

Model	ELECTRA	All-Tokens MLM	Replace MLM	ELECTRA 15%	BERT
GLUE score	85.0	84.3	82.4	82.4	82.2

可以发现，对效果提升最重要的其实是all-tokens的loss计算，这种方式相比只计算15%的token，大大增加了模型收敛速度。

虽然ELECTRA有很多优点，但个人认为它还有以下缺陷：

- **显存占用增多**

之前都是训一个BERT，现在相当于两个，即使共享参数去训练，由于Adam等优化器的关系，需要保存的中间结果数量也是翻倍的。

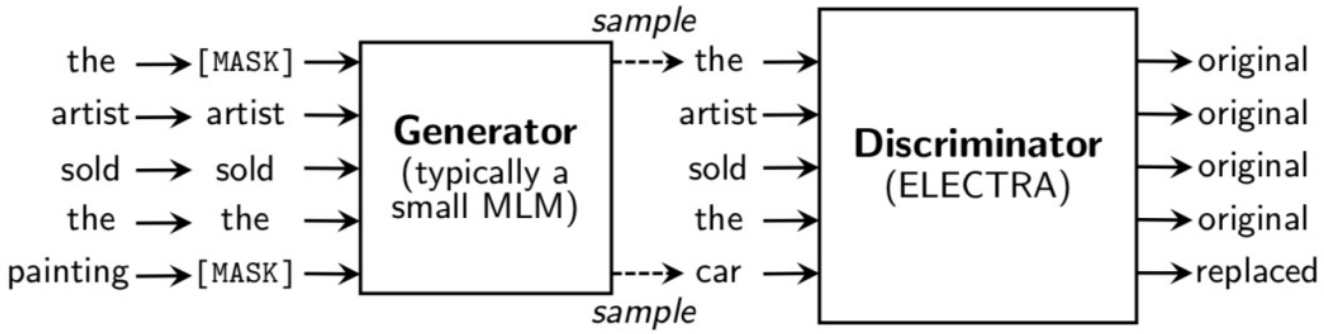
- **多任务学习超参数调整**

ELECTRA的loss由生成器和判别器分别构成，原文中给判别器loss的权重是50，因为二分类的loss会比V分类低很多。但这个数值太过绝对，最好是变成可学习的参数动态调整。

ELECTRA简介

模型结构

ELECTRA的全称是Efficiently Learning an Encoder that Classifies Token Replacements Accurately，以1/4的算力就达到了RoBERTa的效果。模型结构如下：



Generator和Discriminator可以看作两个BERT，生成器的任务是MLM，判别器的任务是Replaced Token Detection，判断哪个字被替换过。

但上述结构有个问题，输入句子经过生成器，输出改写过的句子，因为句子的字词是离散的，所以梯度在这里就断了，判别器的梯度无法传给生成器，于是生成器的训练目标还是MLM（作者在后文也验证了这种方法更好），判别器的目标是序列标注（判断每个token是真是假），两者同时训练，但**判别器的梯度不会传给生成器**，目标函数如下：

因为判别器的任务相对来说容易些，RTD loss相对MLM loss会很小，因此加上一个系数，作者训练时使用了50。

另外要注意的一点是，**在优化判别器时计算了所有token上的loss，而以往计算BERT的MLM　loss时会忽略没被mask的token** 。作者在后来的实验中也验证了在所有token上进行loss计算会提升效率和效果。

事实上，ELECTRA使用的Generator-Discriminator架构与GAN还是有不少差别，作者列出了如下几点：

	ELECTRA	GAN
输入	真实文本	随机噪声
目标	生成器学习语言模型，判别器学习区分真假文本	生成器尽可能欺骗判别器，判别器尽量区分真假图片
反向传播	梯度无法从D传到G	梯度可以从D传到G
特殊情况	生成出了真实文本，则标记为正例	生成的都是负例（假图片）

实验及结论

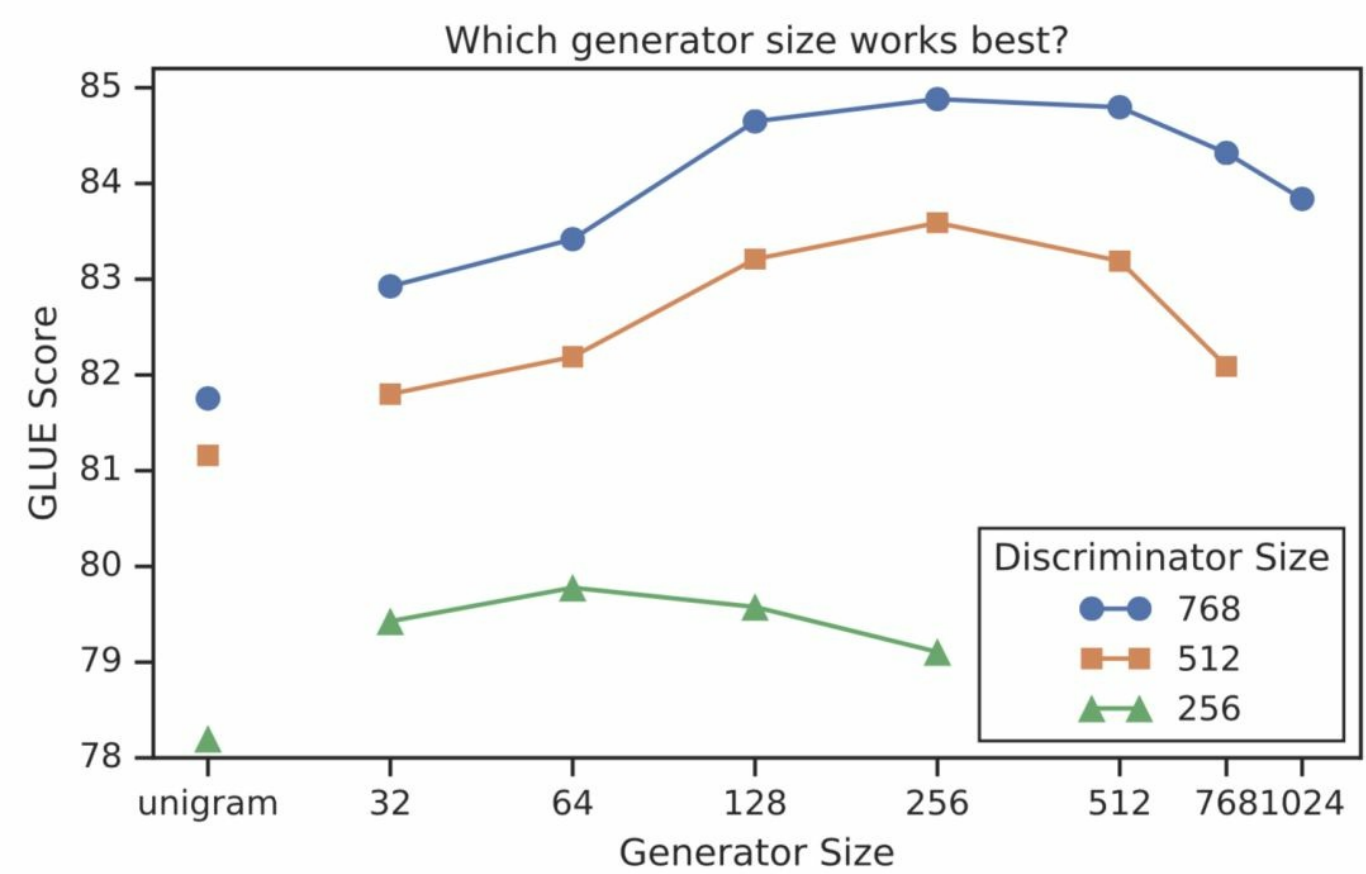
创新总是不易的，有了上述思想之后，可以看到作者进行了大量的实验，来验证模型结构、参数、训练方式的效果。

- **Weight Sharing**

生成器和判别器的权重共享是否可以提升效果呢？作者设置了相同大小的生成器和判别器，在不共享权重下的效果是83.6，只共享token embedding层的效果是84.3，共享所有权重的效果是84.4。作者认为 **生成器对embedding有更好的学习能力**，因为在计算MLM时，softmax是建立在所有vocab上的，之后反向传播时会更新所有embedding，而判别器只会更新输入的token embedding。最后作者只使用了embedding sharing。

- **Smaller Generators**

从权重共享的实验中看到，生成器和判别器只需要共享embedding的权重就足矣了，那这样的话是否可以缩小生成器的尺寸进行训练效率提升呢？作者在保持原有hidden size的设置下减少了层数，得到了下图所示的关系图：



可以看到，**生成器的大小在判别器的1/4到1/2之间效果是最好的**。作者认为原因是**过强的生成器会增大判别器的难度**（判别器：小一点吧，我太难了）。

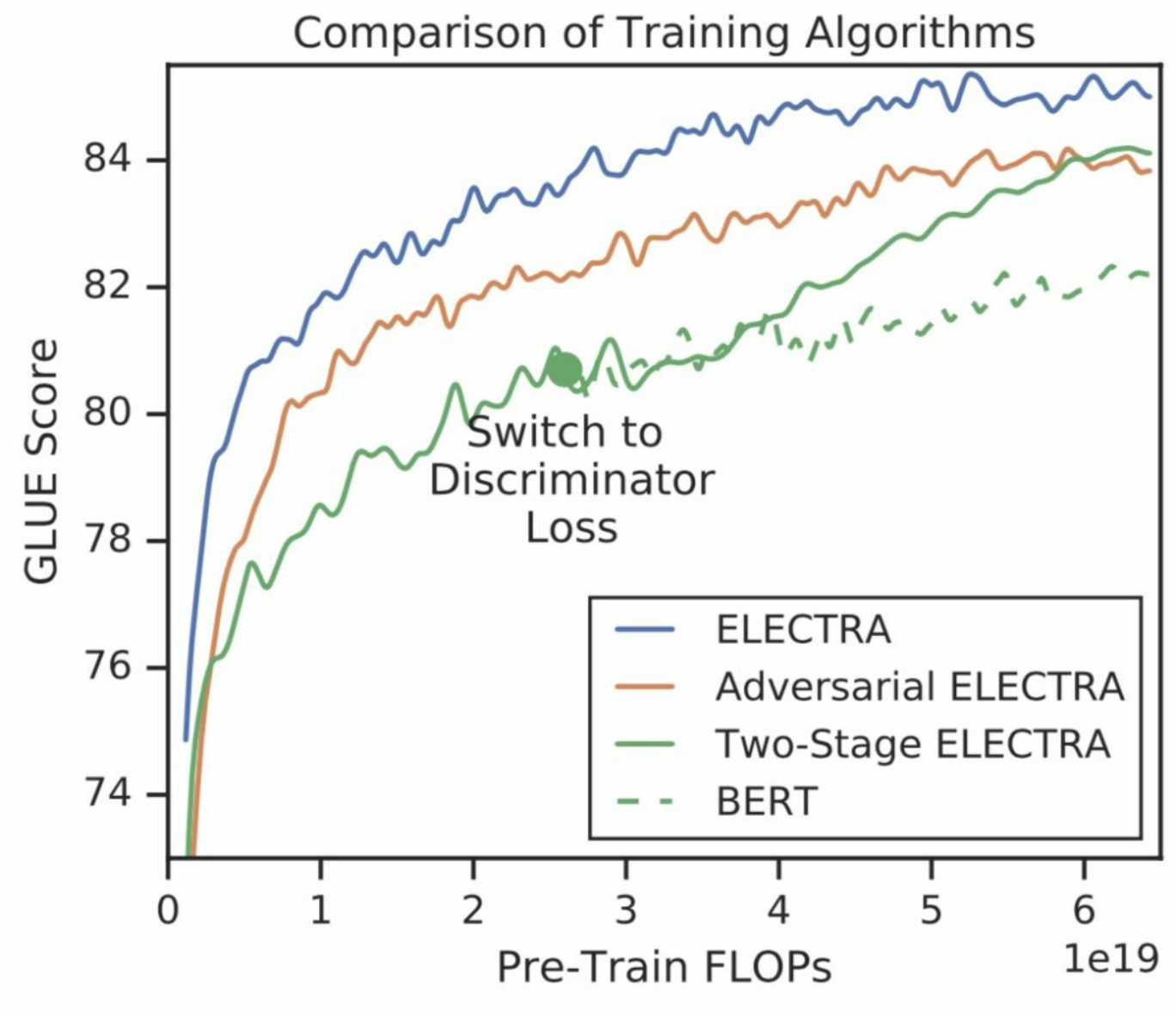
- **Training Algorithms**

实际上除了MLM loss，作者也尝试了另外两种训练策略：

(1) Adversarial Contrastive Estimation：ELECTRA因为上述一些问题无法使用GAN，但也可以以一种对抗学习的思想来训练。作者将生成器的目标函数由最小化MLM loss换成了**最大化判别器在被替换token上的RTD loss**。但还有一个问题，就是新的生成器loss无法用梯度下降更新生成器，于是作者用强化学习Policy Gradient的思想，将被替换token的交叉熵作为生成器的reward，然后进行梯度下降。强化方法优化下来生成器在MLM任务上可以达到54%的准确率，而之前MLE优化下可以达到65%。

(2) Two-stage training：即先训练生成器，然后freeze掉，用生成器的权重初始化判别器，再接着训练相同步数的判别器。

对比三种训练策略，得到下图：



可见“隔离式”的训练策略效果还是最好的，而两段式的训练虽然弱一些，作者猜测是生成器太强了导致判别任务难度增大，但最终效果也比BERT本身要强，进一步证明了判别式预训练的效果。

- **Small model? Big model?**

模型的效果可以参考文首的图片，ELECTRA-Small仅用14M参数量，以前13%的体积，就接近了BERT-Base的效果。ELECTRA-Base更是超越了BERT-Large。由于时间和精力问题，作者们没有把ELECTRA训练更久（应该会有提升），也没有使用各种榜单Trick，所以真正的GLUE test上表现一般。

后台回复【electra】获取论文PDF噢~~

参考文献

[1] https://github.com/google-research/electra

[2] https://www.zhihu.com/question/354070608/answer/885907890

夕小瑶的卖萌屋

关注&星标小夕，带你解锁AI秘籍

订阅号主页下方「撩一下」有惊喜哦

文章已于修改