如何提升大规模Transformer的训练效果？Primer给出答案

原创 首风小兔 少小强的卖萌屋 2021-10-29 17:00

文｜首风小兔
编｜智商掉了一地

> **如何进一步提升大规模Transformer的训练效果？Primer给出了答案，一起来看看吧！**

Transformer是深度学习界的明星模型，由于其具有高度的并行性，十分容易在并行计算系统中扩展至超大规模，自然语言处理任务一路见证了Transformer模型规模的爆炸式增长，

微软和Nvidia不久前联合发布的"Megatron-Turning"更是创造了最新记录：其参数数目已经达到了恐怖的5300亿。大规模Transformer通过横扫各大任务榜单，向所有人展示了"大模型+大数据"这一简单方法的高度通用性。

在更加有效的深度学习技术出现之前，许多已经正在使用Transformer的任务难免都会期待是否拥有一个更大的模型。就能够更进一步提升应用任务的效果？但训练大规模Transformer的高昂成本也同样令人望而生畏，大规模Transformer，让人又爱又恨。本文要介绍的**Primer**就是在该背景下开展的一个工作。

论文标题：

Primer: Searching for Efficient Transformers for Language Modeling

论文链接：

https://arxiv.org/abs/2109.08668

论文代码：

https://github.com/google-research/google-research/tree/master/primer

Arxiv访问慢的小伙伴也可以在 **【少小强的卖萌屋】**订阅号后台回复关键词 **【1029】** 下载论文PDF~

1 为什么会有Primer?

针对训练大规模Transformer存在高昂成本的问题，作者试图回答是否能够通过**模型架构自动搜索技术**，找到一个高效Transformer架构，实现以下目标：

1. 给定学习效果，相比标准Transformer，这个新的变种有着更低的训练代价。
2. 给定学习代价，相比标准Transformer，这个新的变种达到更好的学习效果。

作者给出的答案就是Primer（**PRIM**itives searched transfor**ER**）。

2 什么是Primer?

Primer 对Transformer的结构给出了两处修改，在下图中用红色圈出，在论文进行的各项实证研究中，这两项修改最为鲁棒，论文作者建议：**Transformer语言模型的使用者如果想尝试改进自己的模型，推荐从这两项修改开始。**

▲Primer对Transformer结构的两处重要修改

这两处由遗传算法自动搜索到的修改分别是：

1. 在Feed Forward 模块（FFN）部分，将原来的ReLU激活修改为Squared ReLU，也就是在ReLU之后再添加一个Square运算，作者声称这个小小的修改最为重要，十分有助于加速模型在更短时间内收敛到同样的学习效果。
2. 在自注意力机制中Q，K，V 映射之后添加 3×1 Depthwise 卷积，称之为Multi-DConv Attention (MDHA)；

上面两幅图已十分充足即所得地解释了论文的结果，对经常与深度学习算法打交道的同学来说，根据这幅图已经可以在1分钟之内修改好自己的Tranformer模型，将其变为Primer，然后，**保持所有其他因素不变**去试试能否在自己的任务上复现论文的效果：在更短的时间内，模型收敛到和原来模型同样的精度。

作者为确定搜索出的模型结构具有广泛的实用性，做了大量的覆盖性实验验证，验证变量包括：模型参数规模（20M到1.98）、训练时长（10到10^5加速器小时）、多个数据集（LM1B, C4, PG19）、多个硬件平台（TPUv2，TPUv3，TPUv4 和 V100），将Primer的修改插入多个Transformer 代码库（Tensor2Tensor[1]，Lingvo[2] 和 T5[3]）中的多个模型（dense Transformer，sparse mixture-of-experts Switch Transformer，Synthesizer）。在大量的试验中，作者发现只有**上面两个修改具有广泛的有效性**，作者还列举了一些有效但不总是有效的修改，给出了他们在实验中的表现：

1. **共享Q和K的 depthwise 表示**：共享一部分Q和K映射的映射矩阵，K由图1中的MDHA后加depthwise 卷积计算得到，$Q = KW$，实验发现：**大部分时候这个修改对学习效果都是有害的**，我们可以看到，这是一个令人类专家看来十分奇怪的模型修改，很像是一个典型的自动搜索产生的修改方案，论文中还给出了类似这样的奇怪修改，大部分也都没有能够改进学习代价。
2. **归一化层添加位置**：标准的Transformer实践在自注意力模块（Self-Attention）和FFN（Feed Forward）层之前添加归一化层，论文作者尝试在自注意力模块之前，FFN模块之后添加归一化层，这个修改**会有帮助，但并不总是有帮助。**
3. **自定义归一化**：Primer使用自定义归一化：$x(x-\mu)$ 替代$(x-\mu)^2$，这一修改会有帮助，并不总是有效。
4. **Bottleneck 映射大小的修改**：将原始Transformer模型隐藏层大小从512增减少到384，但将FFN模块的Upwards 映射部分的维度从2048增加到4608，收敛效果**在模型参数小于35M时有明显的改善，**但对更大的模型有效性降低。

3 Scaling Law 对比

由于这篇论文是一篇实验研究，文章用了长达35页的篇幅解释了在TensorFlow中进行模型架构搜的设计、搜索空间设计、诸多无视律的修改，阅读这篇文章时，**研究神经网络架构搜索的读者，可以只关注模型搜索空间设计和搜索算法设计部分；研究自然语言处理任务本身的读者，可以只关注上面两个简单的结论，**在这里我们只重点给摘要作者如何通过实验验证Primer能够减少大规模Transformer训练代价这一关键论点。

作者对Transformer模型的以下几组变量进行全排列组合：（1）层数：$L \in \{6, 9, 12\}$，（2）隐藏层大小 $d \in \{384, 512, 1024\}$，（3）FFN 模块upwards projection 比率 $r \in \{4, 8, 12\}$，产生出模型参数从23M到385M变化的一系列模型，作者在LM1B搜索任务上，使用序列长度64，batch大小4096 token，分别在TPUv2和V100上训练24小时，用Tensor2Tensor和T5中的几种典型Transformer变种作为对比对象，与Primer进行对比。

▲35M模型参数下，几种Transformer模型与Primer加速因子对比

上图给制了35M参数模型的实验结果，以标准Transformer模型训练24小时的学习效果为基准，横轴是几种Transformer模型变种，纵轴是各个变种达到基准效果的加速比，从中可以看到Primer在所有实验组中都有1.7倍以上的加速率。

▲几种Transformer模型与Primer scaling曲线对比

上图又进一步展示了Primer在所有模型规模下的实验数字。横轴是时间，纵轴是验证集上的损失，不同颜色的曲线代表不同的模型规模。左图是对所有Transformer变种用MSE实验去拟合出达到给定精度的最佳模型大小，结制成二维线图，也就是scaling law[4]曲线，可以看到Primer的scaling曲线在最下方，这意味着在所有模型大小情况下，Primer都能更快地达到指定学习精度。

4 小结与讨论

尽管这篇论文长达35页，但是关键结论十分简单，而剩下的篇幅都在阐述如何进行模型搜索和在不同任务上重复验证Primer能否节约Transformer的训练代价，对不关心这些细节的实践者，完全可以跳过作者的长篇大论来直接使用作者的结论。

关于这两个由遗传算法自动搜索到的修改，第一个：squared ReLU 在其它学习任务中已经被多次使用，能够进一步增强ReLU的激活值，或许容易被人类专家注意和想到，第二个：卷积能增加特征向量的局部稳定性，由于Q，K，V projection已经足够简洁，在Q和K projection之后添加depthwise卷积，可能是一个连人类专家也不容易主动尝试的想法，被自动搜索找到了。

读过这篇论文，Primer依然可能有一定的限制，这里指出值得注意的三点。

1. 尽管Primer的目标是减少大规模Transformer训练的代价，但是文章实验的大模型也远远小于GPT-3，当模型参数进一步提升时，这两个修改是否有效，也未经作者的验证，可能还是在从侧面说明，尽管作者想使用自动搜索模型结构这项技术去减少Transformer的训练代价，但进行实验本身的代价依然过于昂贵。
2. 作者自己也指出，实验只在自回归语言模型上进行，而初步测试表明了这两项修改应用于其它类型任务时并不总是有效，也就是说**这两项修改有可能只适用于部分任务。**
3. 一个小小的遗憾是作者在搜索空间构造时并没有对自注意力机制这样的高阶模块的潜在变种进行搜索。毕竟这是Transformer的核心。由于这一步也存在着大量的选择空间，或许也潜藏着压缩Transformer模型训练代价的可能性。

[1] Ashish Vaswani, Samy Bengio, Eugene Brevdo, Francois Chollet, Aidan N. Gomez, Stephan Gouws, Llion Jones, Lukasz Kaiser, Nal Kalchbrenner, Niki Parmar, Ryan Sepassi, Noam Shazeer, and Jakob Uszkoreit. Tensor2tensor for neural machine translation. arXiv preprint arXiv:1803.07416, 2018. (https://arxiv.org/pdf/1803.07416.pdf)

[2] Lingvo: A TensorFlow Framework for Sequence Modeling. (https://blog.tensorflow.org/2019/02/lingvo-tensorflow-framework-for-sequence-modeling.html)

[3] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. Journal of Machine Learning Research, 21(140):1–67, 2020. (https://arxiv.org/pdf/1910.10683.pdf)

[4] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. arXiv preprint arXiv:2001.08361, 2020. (https://arxiv.org/abs/2001.08361)