更深和更宽的Transformer，哪个更好？NUS团队：我站Wider！

原创 CV开发者都爱看的 极市平台 2021-08-09 22:00:00 手机阅读

收录于话题

#Transformer 45 #神经网络结构设计 25

作者丨小马

编辑丨极市平台

极市导读

在本文中，为了在更少的可训练参数下获得更好的性能，作者提出了一个框架：WideNet来有效地部署可训练参数，通过更宽的网络而不是更深。在0.72×可训练参数的情况下，作者的最佳模型比Vision Transformer (ViT)高出1.46%。 >>加入极市CV技术交流群，走在计算机视觉的最前沿

写在前面

Transformer-based结构最近在各项任务上取得了惊人的成果。为了进一步提高Transformer的有效性和效率，在现有的工作中主要有两种思路:(1)扩大可训练参数范围;(2)通过参数共享实现浅层化或在模型深度上进行压缩。

然而，当可供训练的token较少时，较大的模型往往不易于发挥它强大的建模表征能力；另外，当模型非常大时，就需要更多的并行操作。由于特征表示能力有限，较小的Transformer往往达不到大模型的performance。

在本文中，为了在更少的可训练参数下获得更好的性能，作者提出了一个框架来有效地部署可训练参数，通过更宽的网络而不是更深。

具体实现上，作者采用了用一个混合专家(mixture-of-experts，MoE)结构代替前馈网络(FFN)，沿模型宽度进行缩放。接着，作者使用参数不共享的多个Layer Norm在Transformer层之间共享MoE层。这样的部署起到了转换各种语义表示的作用，使模型更具有参数效率和有效性。

为了验证框架的有效性，作者设计了WideNet，并在ImageNet-1K数据集上进行了实验。在0.72×可训练参数的情况下，作者的最佳模型比Vision Transformer (ViT)高出1.46%。采用0.46×和0.13×参数时，作者提出的WideNet仍然比ViT和ViT-MoE分别高出0.83%和2.08%。

1. 论文和代码地址

Go Wider Instead of Deeper

Fuzhao Xue, Ziji Shi, Futao Wei, Yuxuan Lou, Yong Liu, Yang You
Department of Computer Science, National University of Singapore, Singapore

{f.xue,ziji.shi}@u.nus.edu, weifutao2019@gmail.com, yuxuanlou@u.nus.edu, {liuyong,youy}@comp.nus.edu.sg

论文地址：https://arxiv.org/abs/2107.11817

代码地址：未开源

2. Motivation

上面提到了，目前主要有两种方式来提高Transformer的有效性和效率：

1）第一种是沿宽度缩放Transformer到更多可训练的参数。通过稀疏条件计算，这些稀疏模型可以扩展到具有可比较FLOPs的超大模型。

2）另一种是减少可训练参数变成一个小模型。为此，有人提出在Transformer Block之间重用可训练参数。

但是，两种思想都有其局限性。对于大型模型，将Transformer Block中的部分前馈网络(FFN)层替换为MoE层是缩放可训练参数的一种典型而有效的方法。在每个MoE层中，为了优化单个token表示，只有少数专家被激活，因此基于MoE的Transformer拥有与普通Transformer相当的FLOPs。然而，**在训练和推理过程中，需要在TPU或GPU上并行地保持这些模型，所以performance不能被线性的提升** 。

另一个局限性是**基于MoE的模型的稀疏性不能在相对较小的数据集上很好地扩展** 。对于小模型，虽然通过降低模型的深度可以显著减少可训练参数，但这些“比较浅”模型的性能仍然低于原来的Transformer。这些较小的模型只是简单地压缩了原始模型的深度，这种结构导致**模型学习能力的损失是不可避免的** 。

在本文中，作者提出了一个更有效地部署可训练参数的参数部署框架：going wider instead of deeper。然后在Transformer中实现这种思想，实例化为WideNet。WideNet在深度上参数共享，并使用混合专家(MoE)在宽度上扩展可训练参数。在所有的Transformer Block中，作者采用了参数共享的MoE层和Multi-Head Self-Attention层。

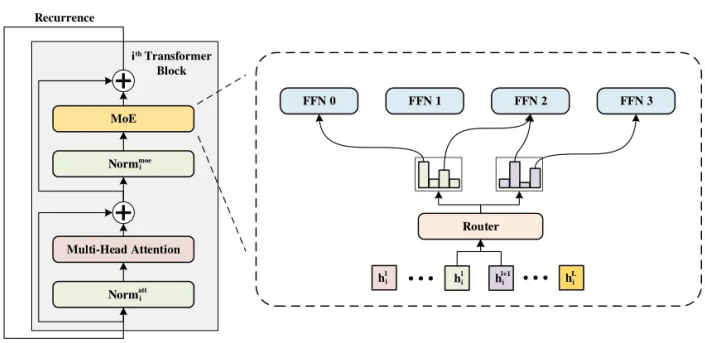
因此，通过MoE层，WideNet可以在宽度上训练更多的参数（**更宽** ），通过参数在Transformer Block之间共享，WideNet可以在深度上训练更少的参数（**更浅** ）。

那么这些参数都共享了，怎么才能保证语义的多样性呢？作者采用了参数不共享的Norm层来提高多样性。Norm层可训练参数不同，使Transformer Block能以不同的表示形式输入。由于MoE层增强了每个Transformer Block的建模能力，因此可以在相同的可训练参数下有效地对不同的语义进行建模。通过一个attention层，一个MoE层和一些参数不共享的Norm层，作者实现了going wider instead of deeper的思想。

与简单的沿宽度缩放相比，WideNet是一个参数效率更高的框架，这使得模型足够小，可以适应下游任务。WideNet中的每个专家可以通过更多的token表示进行训练，使其具有更好的泛化性能。

与单纯随深度压缩的模型相比，WideNet中的所有Transformer Block共享同一个MoE层而不是一个FFN层。这种结构最大限度地提高了每个Transformer Block的建模能力。更多的专家可以以更强的能力为更复杂的token表示建模。另一个区别是参数不共享的Norm层，这些层提供了一些额外的可训练参数，可以将输入表示转换为其他语义信息。在这种情况下，有了足够强大的MoE层，WideNet仍然可以从不同的层中很好地建模语义信息。

3. 方法



在本文中，作者研究了一种新的可训练参数部署框架，并在Transformer上实现了该框架，模型结构如上图所示。在本文中，作者使用Vision Transformer作为Backbone，因此在attention层或FFN层之前对特征进行Normalize。

在WideNet中，作者用MoE层代替FFN层。采用参数共享的Transformer，以实现更有效的参数部署。在每个MoE层中，有一个路由器来选择K个专家来学习更复杂的表示。为了特征语义信息的多样性，Layer Norm层的参数是不共享的。

3.1. 条件计算与MoE

作者的核心理念是沿着宽度部署更多的可训练参数，沿着深度部署更少的可训练参数。基于这个思想，作者使用MoE将Transformer缩放到大量的可训练参数。

给定E个专家（expert，**每个专家在MoE中其实都是一个FFN**）和输入x，MoE模型的输出为：

$$\mathrm{MoE}(x)=\sum_{i=1}^{E} g(x)_i e(x)_i$$

其中$e()$是一个非线性函数，也就图中的FFN；$g()$是可训练的路由器（router）。

从上面的公式可以看出，MoE将多个专家的输出加权求和了，这样计算量其实还是比较大的。为了减少计算量，作者只是将其中的几个专家的结果求和了，意思就是说g函数的输出其实是一个非常稀疏的矩阵，所以每次只选择几个专家，并没有采纳所有专家的“意见”。

3.2. Routing

为了保证g的输出是一个稀疏矩阵，因此作者采用了TopK()函数，按照贡献度来选择前K个专家，表示如下：

$$\mathrm{g}(x)=\mathrm{TopK}(\mathrm{softmax}(f(x)+\epsilon))$$

$f()$是一个从D维（特征的维度）到E维（专家的个数）的线性映射，$\epsilon \sim N(0,1/E^2)$是一个用于专家路由探索的高斯噪声。当K远小于E的时候，g函数的输出是一个稀疏矩阵。

3.3. Balanced Loading

在基于MoE的Transformer中，模型将每个token分派给K个专家。在训练期间，如果MoE模型没有规则，大多数token可能会被分派给一小部分专家。这种不平衡的分配会**降低MoE模型的吞吐量**。更重要的是，**大多数附加的可训练参数没有得到充分的训练**，使得稀疏条件模型无法超越相应的稠密模型。

因此，为了平衡负载，需要避免两件事：(1)分派给单个专家的token过多；(2)单个专家收到的token太少。

为了解决第一个问题，需要设置缓冲容量B。也就是说，对于每个专家，不管分派给该专家多少token，最多只保留B个token。如果分配了超过B个token，那么剩下的标记将被丢弃：

$$B=CKNL$$

C是超参数，用于控制为每个专家保留token的比例；K是每个token选择的专家数量；N是Batch Size；L是每张图像中patch token的数量。

为了解决第二个问题，作者使用可区分的负载均衡损失。对于每个路由操作，给定E个专家和一个Batch中的NL个Token，在训练时的模型总损失中加入以下辅助损失：

$$l_{balance}=\alpha E \cdot \sum_{i=1}^{E} m_i \cdot P_i$$

$$m_i = \frac{1}{L}\sum_{j=1}^{L} \mathrm{h}(x_j)_i$$

m是一个向量，$h(\cdot)$为TopK选取的索引向量。可以看出$h(\cdot)$是不可微的。然而，需要一个可微分的损失函数才能来优化MoE。因此，作者定义P_i为：

$$P_i = \mathrm{softmax}(f(x) + \epsilon)_i$$

可以看出P_i是routing结果softmax的第i个元素，因此P_i是可微的

负载均衡损失的目的是实现均衡分配。当最小化$l_{balance}$时，m和P都接近均匀分布。

3.4. Sharing MoE across Transformer blocks

WideNet采用了一个参数共享的Transformer Block，这么做主要有两个原因：

首先，作者的目的就是提出一个在参数上更加有效的结构；

第二，作者使用MoE层来获得更强的建模能力。

另外为了克服稀疏条件计算产生的过拟合问题，作者给每个专家提供足够的token。

3.5. Individual Layer Normalization

上面作者已经在不同Transformer Block的Self-Attention、MoE上共享了参数，因此，为了提高特征的泛化性，模型使用了不同的LN。（**这个觉得这一步很妙，因为LN本身参数就不多，所以其他结构参数共享，LN参数不共享能够在很小的overhead下提高特征的多样性。**）

第i个参数Transformer Block的计算方式如下：

$$\begin{aligned} x' &= \mathrm{LayerNormal}_i^{att}(x) \\ x &= \mathrm{MHA}(x') + x \\ x'' &= \mathrm{LayerNormal}_i^{moe}(x) \\ x &= \mathrm{MoE}(x'') + x \end{aligned}$$

LayerNorm的计算方式如下：

$$\mathrm{LayerNormal}(x) = \frac{x - \mathrm{E}[x]}{\sqrt{\mathrm{Var}[x] + \epsilon}} * \gamma + \beta$$

3.6. Optimization

虽然在每个Transformer Block中重用了路由器的可训练参数，但由于输入表示的不同，分配也会不同。因此，给定T次具有相同可训练参数的路由操作，优化的损失如下：

$$loss = l_{main} + \lambda \sum_{t=1}^{T} l_{balance}^{T}$$

其中l_{main}就是模型的主要目标，比如说对于分类任务，l_{main}就是cross-entropy loss。

4.实验

4.1. Pretraining Performance

4.1.1. Hyper-parameters

Parameter	Value
Epoch	300
Warmup Epochs	30
Batch Size	4096
Learning rate	0.01
Weight Decay	0.1
Dropout	0.1
Label smoothing	0.1
Mixup prob.	0.5

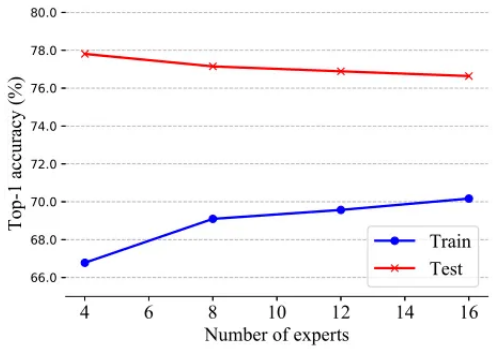
预训练的超参数如上表所示。

4.1.2. Main results

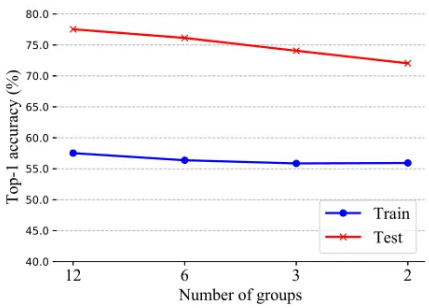
Model	Top-1	Parameters
ViT-B	78.63	87M
ViT-L	77.49	305M
ViT-MoE-B	77.87	128M
ViT-MoE-L	77.41	406M
WideNet-B	77.54	29M
WideNet-L	79.49	40M
WideNet-H	**80.09**	63M

上表展示了ViT和WideNet在ImageNet-1K上的结果。在可训练参数较少的情况下，WideNet-H比ViT-B高出1.46%。即使最小的模型WideNet-B，在可训练参数少4倍以上的情况下，达到与ViT-L和ViT-MoE-B相当的性能。

4.2. MoE Analysis



从上图可以看出随着专家数量增加，虽然模型的学习能力也会更强，但是会导致过拟合。



从上表可以看出，当使用更少的group时，这意味着更少的路由操作，会导致有明显的性能下降。

4.3. Ablation Study

4.3.1. Contributions of key modifications

Model	Top-1	Parameters
WideNet-B	77.54	29M
w/ shared Layer Norm	76.28	29M
w/o MoE layer	Nan	9M
w/o parameter sharing	**77.87**	128M
WideNet-L	**79.49**	40M
w/ shared Layer Norm	78.33	40M
w/o MoE layer	76.89	15M
w/o parameter sharing	77.41	406M
WideNet-H	**80.09**	63M
w/ shared Layer Norm	76.64	63M
w/o MoE layer	78.97	23M
w/o parameter sharing	OOM	

作者进行消融实验，以探究的三个关键修改（Independent Layer Normalization，scaling width with MoE layer，compressing depth with parameter sharing）的影响。

作者首先用共享层替换单个层的Norm。可以观察到，在几乎相同的训练参数下，性能下降。

此外，作者将MoE层恢复到FFN层。如果没有MoE层，训练将会非常困难，可训练的参数会少得多。例如，没有MoE层的WideNet-B会遇到梯度爆炸，性能会显著下降。

最后，如果在Transformer Block之间不共享参数，可以观察到轻微的性能下降和显著的参数增加。

4.3.2. Comparison with comparable speed or computation cost

Model	#Blocks	FNN dim	Para Sharing	Top-1	#Para	Time
ViT-L	24	4096	×	77.49	305M	0.08K
ViT-L	24	4096	√	76.89	15M	0.07K
WideNet-L	12	4096	√	**78.19**	40M	0.07K
ViT-L	24	8192	√	75.81	24M	0.09K
WideNet-L	24	4096	√	**79.49**	40M	0.14K

从上表可以看出，与ViT-L相比，WideNet-L的计算成本更高。然而，当WideNet-L使用的Transformer Block比ViT-L少时，WideNet-L在训练时间略短和13.1%的参数情况下，比ViT-L高出0.7%。

5. 总结

在本文中，作者提出“**go wider instead of deeper** ”来使得参数部署更高效和有效。基于这个思想作者提出WideNet。WideNet首先通过在Transformer Block之间共享参数来压缩可训练参数和深度。为了最大化每个Transformer Block的建模能力，作者将FFN层替换为MoE层。然后，参数不共享的LayerNorm提供了一种更有效的参数化方法来增强语义表示。作者通过实验也证明了WideNet能够在参数更少的情况下达到更好的性能。

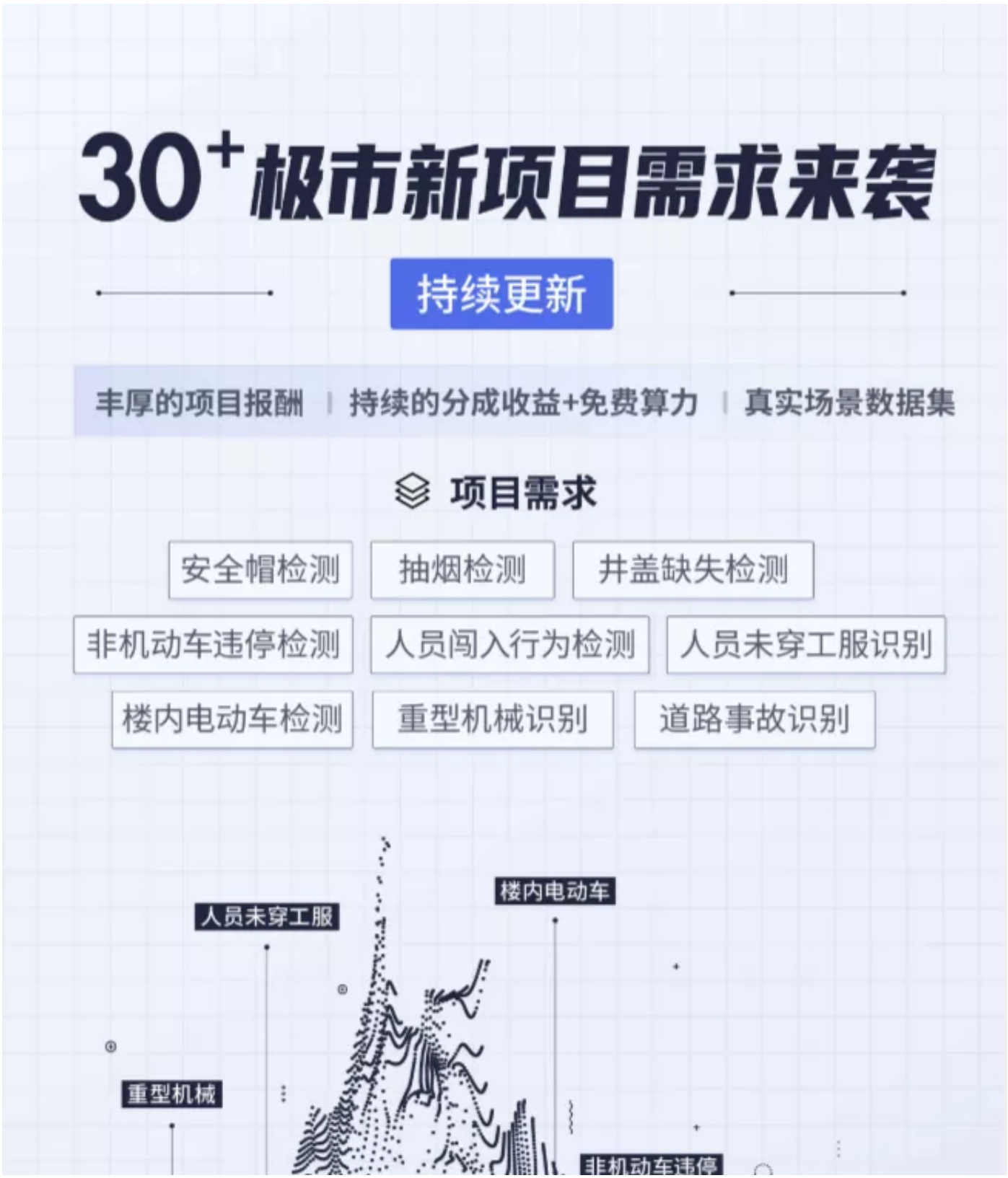
公众号

极市干货

深度学习环境搭建：如何配置一台深度学习工作站?

实操教程：OpenVINO2021.4+YOLOX目标检测模型测试部署 | 为什么你的显卡利用率总是0%?

算法技巧（trick）：图像分类算法优化技巧 | 21个深度学习调参的实用技巧




安全帽
More
道路事故
抽烟
人员闯入
More
More
井盖缺失
长按识别二维码
获得项目报酬及项目报名

极市平台签约作者#

小马
公众号：FightingCV

厦门大学人工智能系20级硕士，FightingCV公众号运营者
研究领域：研究方向为多模态内容理解，
专注于解决视觉模态和语言模态相结合的任务，促进Vision-Language模型的实地应用。
知乎：FightingCV

作品精选

CVPR2021最佳学生论文提名：Less is More
Transformer一作又出新作！HaloNet：用Self-Attention的方式进行卷积
超越Swin，Transformer屠榜三大视觉任务！微软推出新作：Focal Self-Attention

投稿方式：
添加小编微信Fengcall（微信号：fengcall19），备注：**姓名-投稿**

△长按添加极市平台小编

觉得有用麻烦给个在看啦~

阅读原文