MLP再添新砖，Facebook入局！ResMLP:完全建立在MLP上的图像分类架构

原创 CV开发者都爱看的 极市平台 2021-05-10 18:30:00 手机阅读

作者丨happy
审稿丨邓富城
编辑丨极市平台

极市导读

继谷歌MLP-Mixer引爆CV圈后，各高校也纷纷入场，facebook也不例外，在今天提出一种完全建立在MLP上的架构ResMLP用于图像分类，得出了与MLP-Mixer相似的结论。 >>本周二（5月11日）【极市直播】张新宇：CVPR 2021-Alpha Refine：通过精确的边界框估计提高跟踪性能

ResMLP: Feedforward networks for image classification with data-efficient training

Hugo Touvron[1,2] Piotr Bojanowski[1] Mathilde Caron[1,3]
Matthieu Cord[1,2] Alaaeldin El-Nouby[1,3] Edouard Grave[1]
Armand Joulin[1] Gabriel Synnaeve[1] Jakob Verbeek[1] Hervé Jégou[1]

[1]Facebook AI [2]Sorbonne University [3]Inria

paper: https://arxiv.org/abs/2105.03404

code1: https://github.com/lucidrains/res-mlp-pytorch（第三方）

code2: https://github.com/facebookresearch/deit(大概开源在此)

> 继上周谷歌的MLP-Mixer引爆CV圈后，清华大学、牛津大学刊出关于MLP的尝试，大有MLP->CNN->Transformer->MLP轮回之趋势。今天Facebook同样入局，提出了ResMLP，得出了与MLP-Mixer相似的结论。

前景回顾

在前几天(也就是5月5日)谷歌一记重拳 `MLP-Mixer` 引爆了CV圈，挖了新坑：**“移除掉自注意力与卷积后的网络，仅仅采用MLP仍可取得与CNN、Transformer相媲美的性能”**。笔者曾对其有初步的解读，感兴趣者可移步：谷歌最新提出无需卷积、注意力，纯MLP构成的视觉架构！网友：MLP is All You Need？

次日(即5月6日)，清华大学丁霄汉博士(重参数方案ACNet、RepVGG、DBB等方案的一作)提交了MLP相关的文章，揭示了“**将重参数卷积嵌入圈连接层同样可以取得非常不错的效果**”。

同日，清华大学胡事民团队也提交了其团队关于自注意力的探索，提出了一种新的注意力机制： External Attention ，基于两个外部的、小的、可学习的和共享的存储器，只用两个级联的线性层和归一化层就可以取代了现有流行的学习架构中的“Self-attention”，揭示了线性层和注意力机制之间的关系。

当天“哭晕在厕所”的牛津小伙(Luke)先在github上提交了他关于MLP的实验code与预训练模型（https://github.com/lukemelas/do-you-even-need-attention），后在arxiv上看出了Do You Even Need Attention?一文。再一次说明：“即使不使用注意力机制，纯MLP也可以取得非常强的性能”。也就是说，**ViT的强大性能可能不是源自注意力机制**，而是其他因素。关于MLP的争论与介绍可参见 AI科技评论 的报道：CV圈杀疯了！继谷歌后，清华、牛津等学者又发表三篇MLP相关论文，LeCun也在发声

今天，Facebook(DeiT、CaiT的作者团队)同样入局MLP，提出了ResMLP，得出了与前文类似的结论。

Abstract

本文提出一种完全建立在MLP上的架构ResMLP用于图像分类。它是一种交替执行如下两个模块的简单残差网络：(1) 一个作用于图像块的线性层，独立于通道；(2) 一个作用于通道的两层前馈网络，独立于图像块。

当采用先进的训练策略(重度数据增广、可选知识蒸馏)进行训练时，所提方法在ImageNet上取得了令人惊讶的精度-复杂度均衡结果。

Introduction

最近，源自自然语言处理的Transformer在CV圈搅翻了天！当采用充分大的数据训练时，ViT在ImageNet上取得了与CNN相当甚至更优的性能。为什么会搅翻天呢？自从2012年AlexNet以来，为了提升ImageNet上的性能，全世界的研究员设计了大量的架构，引入了诸多的先验信息，甚至还采用NAS等技术才将ImageNet的精度提升到了80+%；然而外来户Transformer轻松的就达到了这个精度，且不需理会卷积架构的内在假设与平移不变性。真乃“一记重拳”！

从Transformer角度来看，更长周期的训练、更多的参数、更多的数据、更多的正则技术足以覆盖ImageNet分类的重要先验信息。

本文则将上述趋势进一步前推，提出了ResMLP(Residual Multi-Layer Perceptrons)：一种纯基于MLP的架构，见下图。所提架构极为简单：它采用平展后的图像块作为输入，通过线性层对其进行投影，然后采用前述所提两个残差操作对投影特征进行更新；最后将所得块特征进行均值池化后进行分类。

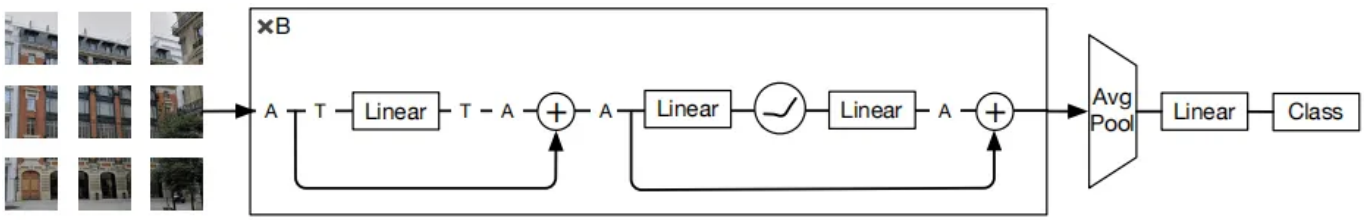

Figure 1: The ResMLP architecture: After flattening the patch into vectors, our network alternately processes them by (1) a communication layer between vectors implemented as a linear layer; (2) a two-layer residual perceptron. We denote by A the operator Aff, and by T the transposition.

所提架构受启发于ViT，但更简单，区别在于：**没有采用任何形式的注意力，仅仅包含线性层与GELU非线性激活函数**。由于所提架构比Trasnformer的训练更为稳定，我们不需要与batch或者跨通道相关的操作，比如BatchNorm、GroupNorm、LayrNorm等。所提方案的训练过程基本延续了DeiT与CaiT的训练方式。

由于所提方案的线性特性，该模型中的块交互易于可视化、可解释。尽管第一层学习到的交互模式与卷积滤波器非常类似，但在更深的层我们观察到块间更微妙的交互作用：包含轴向滤波器形式、长期依赖性等。

总而言之，本文贡献包含以下几点：

- 尽管简单，**无需额外数据、规范化(如BN)技术，ResMLP可以在ImageNet上取得令人惊艳的精度-复杂度均衡**；
- 这些模型可以从蒸馏方法中受益进而继续提升模型性能；
- 由于块间简单的线性通信，我们可以网络从不同层之间学习何种类型的空域交互。

Method

所提ResMLP架构以$N \times N$非重叠块作为输入，这些块将独立的经由线性层处理并构成N^2d维嵌入特征；所得嵌入特征将送入后续一系列残差多层感知器层中生成N^2d维输出嵌入特征；将上述输出嵌入特征进行平均得到d维图像表达；最后将上述图像表达送入线性分类层预测图像对应的标签。训练过程中采用交叉熵损失。

Residual Multi-Perceptron Layer 所提网络由一系列具有相同结构的层(线性层后接前馈层)构成。类似Transformer层，每个子层采用了跳过连接并行，但并未采用LayerNorm，这是因为采用如下仿射变换训练已经非常稳定：

$$Aff_{\alpha,\beta}(x) = Diag(\alpha)x + \beta$$

其中，α, β表示可学习向量。需要注意的是：该层**推理无耗时**，因其参数可与前接线性层合并。$Aff(x)$独立的作用于X的每一列，尽管与BatchNorm、LayerNorm非常类似，但该操作不依赖任何batch统计；它与近期提出的LayerScale非常接近，但LayerScale没有偏置项。

总而言之，所提$Multi-Perceptron$层将N^2d维输入特征堆叠为$d \times N^2$矩阵X，输出N^2d维输出特征，计算公式如下：

$$Z = X + Aff((\mathbf{A}Aff(X)^T)^T)$$
$$Y = Z + Aff(\mathbf{C}GELU(\mathbf{B}Aff(Z)))$$

其中，A, B, C表示主要的学习参数。参数矩阵A 的维度维$N^2 \times N^2$，用于混合所有位置的信息，而前馈层则作用于每个位置。因此，中间激活矩阵Z具有与矩阵X、Y相同的维度。最后参数矩阵B和C的维度类似Transformer层，即$4d \times d, d \times 4d$。

所提方法与Transformer层的主要区别在于：我们**采用线性交互替换自注意力；**自注意力具有数据依赖性，而所提线性层则不具备。相比卷积的局部感受野、权值共享特性，所提线性层具有全局感受野，参数不共享，空域独立于通道实施。

相比ViT，本文所提方法主要有以下几点区别：

- 不包含任何自注意力模块；
- 不包含额外的 `class` token；
- 不包含任何形式的位置嵌入信息；
- 移除了LayerNorm，引入了一种简单的可学习仿射变换。

除了均值池化外，我们还引入CaiT中的 `class-attention` ：它包含两层与Transformer相同的结构，但仅仅 `class token` 基于冻结的块嵌入进行更新。我们将其引入到所提方案中并进行了适配：**采用线性层替换了注意力交互**。这种方式可以进一步提升模型性能，但同时也会增加额外的参数量、计算量。我们将这种方案称之为 `class-MLP` 。

Experiments

Experimental setting

Datasets 训练数据为ImageNet，除了ImageNet本身的验证集外，我们还在ImageNet-real、ImageNet-v2数据集上进行了验证测试。

Training paradigms 训练方式考虑了以下两种形式：

- 监督学习：采用softmax分类+交叉熵损失训练，本文主要聚焦于此；

- 知识蒸馏：采用ConvNet通过知识蒸馏方式引导ResMLP训练。

Hyper-parameter setting 在监督学习中，我们采用 Lamb 羽化期，学习率为5×10^{-3}，权值衰减0.2。超参设置于DeiT类似，知识蒸馏时的老式模型为RegNety-16GF。

Main Results

	Arch.	#params ($\times 10^6$)	throughput (im/s)	FLOPS ($\times 10^9$)	Peak Mem (MB)	Top-1 Acc.
State of the art	CaiT-M48↑448ϒ [50]	356	5.4	329.6	5477.8	86.5
	NfNet-F6 SAM [5]	438	16.0	377.3	5519.3	86.5
Convolutional networks	EfficientNet-B3 [46]	12	661.8	1.8	1174.0	81.1
	EfficientNet-B4 [46]	19	349.4	4.2	1898.9	82.6
	EfficientNet-B5 [46]	30	169.1	9.9	2734.9	83.3
	RegNetY-4GF [40]	21	861.0	4.0	568.4	80.0
	RegNetY-8GF [40]	39	534.4	8.0	841.6	81.7
	RegNetY-16GF [40]	84	334.7	16.0	1329.6	82.9
Transformer networks	DeiT-S [49]	22	940.4	4.6	217.2	79.8
	DeiT-B [49]	86	292.3	17.5	573.7	81.8
	CaiT-XS24 [50]	27	447.6	5.4	245.5	81.8
Feedforward networks	ResMLP-12	15	1415.1	3.0	179.5	76.6
	ResMLP-24	30	715.4	6.0	235.3	79.4
	ResMLP-36	45	478.7	8.9	291.3	79.7

Table 1: **Comparison between architectures on ImageNet classification.** We compare different architectures for images based on convolutional networks, Transformers and feedforward networks with comparable FLOPS and number of parameters. We report Top-1 accuracy on the validation set of ImageNet with different measure of complexity: throughput, FLOPS, number of parameters and peak memory usage. All the models use 224×224 images as input. All the Transformers and feedforward networks uses 14×14 patches of size 16×16. Feedforward networks use a working dimension of d=384. The throughput is measured on a single V100 GPU with batch size fixed to 32. For reference, we include the state of the art with ImageNet training only (86.5% Top-1 with very large models).

上表对比了所提方法与ConvNet、Transformer在监督学习框架下的性能。从表中对比可以看到：

- 尽管ResMLP在精度、FLOPs以及吞吐量的均衡方面不如ConvNet、Transformer，但其性能仍非常优异；
- 事实上，这里所对比的ConvNet经过了多年的研究与精心优化才达到了如此好的性能；而本文所提方法只是最简单的适配，未经过多的优化。

Arch.	Teacher	w/o Dist.	w/ Dist.
DeiT-S	RegNety-16GF	79.9	81.2
ViT-B	RegNety-16GF	81.8	83.4
ResMLP-12	RegNety-16GF	76.6	77.8

Table 2: **Impact of distillation on ResMLP.** We report Top-1 accuracy on the validation set of ImageNet when training models with and without knowledge distillation using a pre-trained model. Accuracies for Transformer-based models are taken from Touvron *et al.* [49].

上表对比了知识蒸馏的影响性。从中可以看到：

- 类似于DeiT，ResMLP同样能从ConvNet受益；
- 表中结果表明：前馈网络仍存在过拟合问题。额外的正则技术与蒸馏可以进一步提升模型的性能。

Architecture	FLOPs	res.	$CIFAR_{10}$	$CIFAR_{100}$	Flowers102	Cars	$iNat_{18}$	$iNat_{19}$
EfficientNet-B7	37.0B	600	98.9	91.7	98.8	94.7	_	_
ViT-B/16	55.5B	384	98.1	87.1	89.5	_	_	_
ViT-L/16	190.7B	384	97.9	86.4	89.7	_	_	_
Deit-B/16	17.5B	224	99.1	90.8	98.4	92.1	73.2	77.7
ResNet50	4.1B	224	_	_	96.2	90.0	68.4	73.7
Grafit/ResNet50	4.1B	224	_	_	97.6	92.7	68.5	74.6
ResMLP-12	3.0B	224	98.1	87.0	97.4	84.6	_	_
ResMLP-24	6.0B	224	98.7	89.5	97.9	89.5	64.3	72.5

Table 3: **Evaluation on transfer learning.** We compare models trained on ImageNet on for transfer to datasets covering different domains. The ResMLP architecture takes 224×224 images during training and transfer, while the ViTs and EfficientNet-B7 work with higher resolutions, see "res." column.

上表对比了ResMLP在迁移学习方面的性能。从中可以看到：

- 相比现有架构，ResMLP极具竞争力。
- 采用充分大的数据、正则技术可以极大的降低模型过拟合的趋势。

全文到此结束，更多消融实验与分析建议查看原文。

参考code

```
# No norm layer
class Affine(nn.Module):
    def __init__(self, dim):
        super().__init__()
        self.alpha = nn.Parameter(torch.ones(dim))
        self.beta = nn.Parameter(torch.zeros(dim))
    def forward(self, x):
        return self.alpha * x + self.beta

# MLP on channels
class Mlp(nn.Module):
    def __init__(self, dim):
        super().__init__()
        self.fc1 = nn.Linear(dim, 4 * dim)
        self.act = nn.GELU()
        self.fc2 = nn.Linear(4 * dim, dim)
    def forward(self, x):
        x = self.fc1(x)
        x = self.act(x)
        x = self.fc2(x)
        return x

# ResMLP blocks: a linear between patches + a MLP to process them independently
class ResMLP_BLocks(nn.Module):
    def __init__(self, nb_patches ,dim, layerscale_init):
        super().__init__()
        self.affine_1 = Affine(dim)
        self.affine_2 = Affine(dim)
        self.linear_patches = nn.Linear(nb_patches, nb_patches) #Linear layer on patches
```

```
        self.mlp_channels = Mlp(dim) #MLP on channels
        self.layerscale_1 = nn.Parameter(layerscale_init * torch.ones(dim)) #LayerScale
        self.layerscale_2 = nn.Parameter(layerscale_init * torch.ones(dim)) # parameters

    def forward(self, x):
        res_1 = self.linear_patches(self.affine_1(x).transpose(1,2)).transpose(1,2)
        x = x + self.layerscale_1 * res_1
        res_2 = self.mlp_channels(self.affine_2(x))
        x = x + self.layerscale_2 * res_2
        return x

# ResMLP model: Stacking the full network
class ResMLP_models(nn.Module):
    def __init__(self, dim, depth, nb_patches, layerscale_init, num_classes):
        super().__init__()
        self.patch_projector = Patch_projector()
        self.blocks = nn.ModuleList([
            ResMLP_BLocks(nb_patches ,dim, layerscale_init)
            for i in range(depth)])
        self.affine = Affine(dim)
        self.linear_classifier = nn.Linear(dim, num_classes)
    def forward(self, x):
        B, C, H, W = x.shape
        x = self.patch_projector(x)
        for blk in self.blocks:
            x = blk(x)
            x = self.affine(x)
            x = x.mean(dim=1).reshape(B,-1) #average pooling
        return self.linear_classifier(x)
```

本文亮点总结

1.本文提出一种完全建立在MLP上的架构ResMLP用于图像分类。它是一种交替执行如下两个模块的简单残差网络：(1) 一个作用于图像块的线性层，独立于通道；(2) 一个作用于通道的两层前馈网络，独立于图像块。

2.本文贡献包含以下几点：

- 尽管简单，无需额外数据、规范化(如BN)技术，ResMLP可以在ImageNet上取得令人经验的精度-复杂度均衡；
- 这些模型可以从蒸馏方法中受益进而继续提升模型性能；
- 由于块间简单的线性通信，我们可以网络从不同层之间学习何种类型的空域交互。

如果觉得有用，就请分享到朋友圈吧！

极市平台
专注计算机视觉前沿资讯和技术干货，官网：www.cvmart.net
624篇原创内容

公众号

△点击卡片关注极市平台，获取最新CV干货

公众号后台回复“RegNet”获取资源链接～

极市干货

顶会干货： CVPR 二十年，影响力最大的 10 篇论文！ | CVPR2021 最新18篇 Oral 论文 | 学术论文投稿与Rebuttal经验分享

实操教程： PyTorch自定义CUDA算子教程与运行时间分析 | pytorch中使用detach并不能阻止参数更新

招聘面经： 秋招计算机视觉汇总面经分享 | 算法工程师面试题汇总

最新CV竞赛： 2021 高通人工智能应用创新大赛 | CVPR 2021 | Short-video Face Parsing Challenge

阅读原文 文章已于2021/05/11修改