预激活与自注意力机制在图像分类中的应用

黄杰辰
2018013362
计86
hjc18thu@163.com

李炳睿
2018013391
计84
zplibingrui@163.com

蔡韫睿
2018013360
计82
caiyr18@163.com

1 简介

Google Landmark Recognition Challenge[1]是Google已经连续开展数年举办的在线开放图像识别比赛，这一任务有其实际价值（如Google开发的帮助人们整理旅行相册的地标识别引擎），也有其理论难度。而本项目的背景正是Google在2020年发起的这一地标分类大赛，虽然比赛已经逾期，但我们打算使用其数据集Google-Landmarks Dataset[2]，我们将利用神经网络来使机器学习不同地理位置的建筑风格，实现图像识别。同时考虑到实际的资源问题，我们的目标是在学习、复现已有成熟的模型的基础上，比较分析其性能和优缺点；然后根据这些提出一些新的更好的模型，让它能以更低的计算代价训练出更优秀的效果；并希望它能够具有鲁棒性，优化一些整个图像分类领域不可避免的问题（如梯度爆炸等）。

近年来卷积网络的方法不断涌现，从最开始的CNN模型，到后来一直加以改进的VGG、ResNet模型，它们在大规模的图像处理问题上取得了巨大的成功。但是它们也并非完美，如VGG存在收敛速度过慢、Inception存在模型超参数过多、设计复杂等问题，因此在模型方面我们试图取其精华去其糟粕，有效缓解以上的问题；而且就本次项目而言，数据集本身存在图片清晰度不一、噪声过大等挑战，我们的模型需要有更强大的局部特征提取能力。在本文中我们以VGG/ResNet[1]提出的架构哲学为基础，研读了多种流行的卷积网络设计后[2, 3]，提出了两个新的、可以有效解决上述困难的模型，它们各有特点和擅长的方面：

(1)我们基于Inception-ResNet提出了一种引入预激活单元(Pre-Activation)的Inception变体，姑且称新模型为Inception-Res-PA，它将inception模型中ReLU激活层带来的不稳定性转换为输入输出的恒等映射，即在ReLU前加入BN，有助于避免梯度消失和爆炸问题。

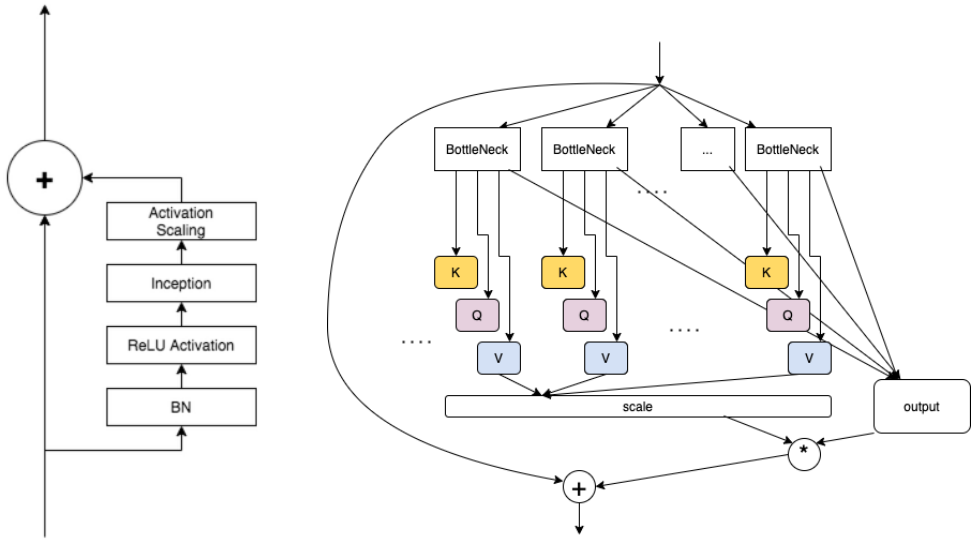
(2)我们在ResNet中引入了自注意力机制(Self-Attention)，姑且称新模型为ResNeAt：

$$Scale = Sigmoid(Softmax(QK^T/d) \cdot V)$$

这里的Q/K/V为Bottleneck中的卷积处理后的三个向量。它的特点是将特征图的不同通道学到相互注意力能力，同时延续cardinality的概念[2]，旨在减少参数量的使用，也让模型更加简洁易收敛。

我们利用本次地标大赛数据集来全面评价我们的两个模型。在Inception-Res-PA与inception模型的对比中，我们通过数据可视化的方法可以明显发现其收敛快、层数可以更深的优点；在ResNeAt与ResNeXt的对比中，我们考察其参数量，发现ResNeAt要

[1] https://www.kaggle.com/c/landmark-recognition-challenge
[2] https://www.kaggle.com/google/google-landmarks-dataset



(a) Inception-Res-PA (b) ResNeAt

Figure 1: 我们提出的两个新模型，分别对Inception、ResNet进行了改进，达到更快收敛、更高准确率的效果

明显少于ResNeXt。因此可以认为我们的两个模型均正确实现了各自优化方面，且他们比原先的架构在准确率上都有了明显的提升，也达到了我们项目本身的目标。

以上结果可以强有力地证明我们模型的正确性和优越性，它优化了现在图像分类模型中的一些共性问题。并且我们相信残差网络原理在各个图像问题上都是相通的，期望这两个模型的思想可以适用于其他更多的大数据集上。

2 相关工作

在大规模图像分类任务中，自VGGNet[4]和GoogLeNet[5]等重要的里程碑相继被提出后，卷积神经网络的方法变得十分流行且有效。

2.1 残差学习与短路连接

在He等人的论文中[1]，残差连接的方法被引入，为了说明这种方法在图像分类、物体检测等方面的好处，他们给出了一系列理论和实践上可信的证据。相关工作指出，网络的深度在分类等任务中有重要的影响，但是在深度增加的同时，模型的复杂性相应提高，相对于层数较少的模型它变得更难训练。在实践中也确实如此，当深度较深的模型开始收敛时，退化问题暴露了出来：即准确率很快达到饱和，训练错误在更高的水平，并且这个问题不是由过拟合导致的。

但事实上，当那些更深的层将自身变为一个恒等映射(Identity Map)时，它总能达到相对较浅的模型的效果，因此它的最好效果不会比他们更差，但作者发现恒等映射是难以拟合的。因此，作者提出了残差连接的办法，对于原先的输出$\mathcal{H}(x)$，将其显式地表示成残差函数$\mathcal{F}(x) := \mathcal{H}(x) - x$，那么原来的输出变为$\mathcal{F}(x) + x$。由于$\mathcal{H}(x)$是复杂的非线性的，$\mathcal{H}(x)$和$\mathcal{F}(x)$的复杂程度相当，但是$\mathcal{F}(x)$的训练难度变得更低，因为只需要将参数训为全0就能达到残层模型的效果。

最终，他们采用了一个由如下公式定义的残差单元:

$$y = \mathcal{F}(x, W_i) + x$$

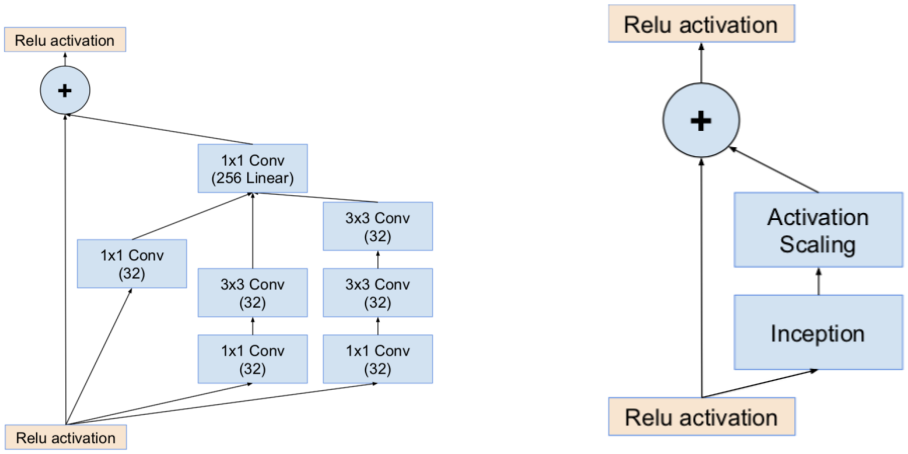
这里的$\mathcal{F}(x, W_i)$代表残差函数，在实际的使用中作者采用了2到3层的卷积网络；此外，作者将输入通过一个短路连接直接作用到$\mathcal{F}$上，短路连接的添加对参数量没有影响，事实上相比VGG仍有更低的复杂程度。在实验中，ResNet成功解决了退化问题，并且达到了

比VGG更好的效果。但ResNet最深能够达到122层，当模型达到上千层时，退化问题仍能够显现出来。此外，我们认为残差连接对准确率提升没有影响，它只是解决了模型变深的退化问题，而准确率的增加还是得益于模型深度的增加。

2.2 多路卷积

在GoogLeNet[5]被提出以来，Inception的思路得到发展，即对同一个输入采用多路卷积的方法分别得到不同的子输出，再将多个子输出求和得到最后的输出。多路的卷积采用不同大小的卷积核以提取不同的特征。从Inception的提出到现在，Inception已经来到了第四个版本，但事实上在Inception-v3及以前，模型结构只是在子网络的范围内进行改动，由于当时计算工具的限制，模型也被分成多个子网络分别训练。例如从Inception-v1(GoogLeNet)到Inception-v2，引入了BatchNorm层，以让收敛更加稳定；从Inception-v2到Inception-v3，将一个$n \times n$的大卷积核，转化成大小为$1 \times n$ 和$n \times 1$的两个卷积核，以减小参数量，减少模型的复杂度。而在此之上，Inception-v4则更加大胆地改变了整个模型的结构, 去掉了一些不必要的特性, 在不影响整个模型质量的基础上, 加快了训练速度。

此外，借鉴残差连接的思想，Inception团队基于Inception-v4提出了Inception—ResNet[3], 将Inception的思想引入每个残差单元，其中的一个残差单元如图2(a)所示，最终的模型深度达到了层，比Inception-v4模型有更深的深度。但事实上，Inception-ResNet引入残差连接后仍作出了一系列改进，最主要的在每个残差Inception单元中，都加入一个1×1的卷积核。这样做的目的有两个，一是增加输出的通道数，让每个单元的输出与输入的shape相匹配, 从而用来补偿残差单元带来的维度下降, 这样能使训练更加稳定；二是起到放缩的作用，如图2(b)经作者的实验发现这样能使训练过程更加稳定。



(a) Inception-ResNet A类残差单元　　(b) Inception-ResNet 放缩机制

Figure 2: Inception-ResNet中的残差单元

虽然作者提出了多种让训练更加稳定的措施，但是在通道数很高的时候仍是会有训练过程中不稳定的情况，在本篇文章中，我们提出了一种方法期望能使收敛过程更加稳定，同时达到收敛速度更快的效果。

2.3 恒等映射

在He等人提出了ResNet之后，该团队又提出了在残差单元中保持恒等映射的方法[6]。作者在文章中提出，在短路连接中保持一个干净的通路，甚至没有ReLU和BatchNorm层的作用，是很重要的。当我们保持这样一个干净的通路后，设某个残差单元的输入是x_l，该残差单元的输出是x_{l+1}，同时也是下一个残差单元的输入，我们有以下的公式：

$$x_{l+1} = x_l + \mathcal{F}(x_l, W_l)$$

从而当x_l经过很多层后，有如下公式：

$$x_L = x_l + \sum_{i=l}^{L-1} \mathcal{F}(x_i, W_i)$$

我们可以看到输入x_l仍然保持在这里。因此当进行后向传播时，有如下的梯度公式

$$\frac{\partial \mathcal{E}}{\partial x_l} = \frac{\partial \mathcal{E}}{\partial x_L}\frac{\partial x_L}{\partial x_l} = \frac{\partial \mathcal{E}}{\partial x_L}(1 + \frac{\partial}{\partial x_l}\sum_{i=l}^{L-1} \mathcal{F}(x_i, W_i))$$

从这个式子中可以看出，在这样的结构下，等式右边括号中左边一项永远保持1，但右边一项不会一直是-1，从而梯度不可能恒为0。在此基础上，这种结构避免了梯度的消失和爆炸问题，从而支持让模型拥有更深的深度，同时也对准确率有所提升。

为了保持这样一条恒等映射的通路，ReLU层需要在原来的残差单元中改变位置。为此，作者进行了多次实验，最后得到了如图3所示的结构，这种结构采用了预激活的方式；同时，为了让ReLU更好地发挥出它应有的效果，在ReLU层之前引入了BatchNorm。

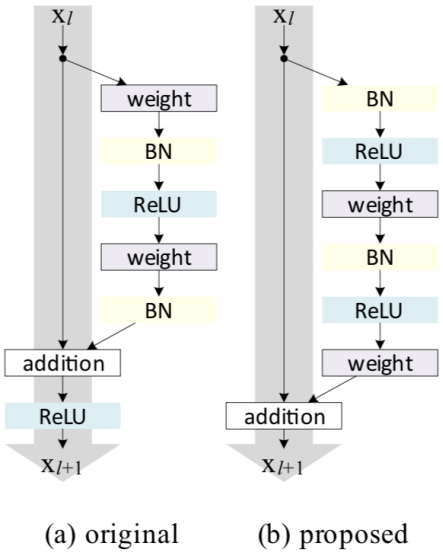


Figure 3: ResNet残差单元与引入恒等映射的残差单元

2.4 注意力机制

注意力机制模仿了生物观察行为的内部过程，即一种将内部经验和外部感觉对齐从而增加部分区域的观察精细度的机制。注意力机制可以快速提取稀疏数据的重要特征。而自注意力机制是注意力机制的改进，其减少了对外部信息的依赖，更擅长捕捉数据或特征的内部相关性。

Transformer就成功抛弃了传统的CNN和RNN，完全由self-attention模块构成，在自然语言处理领域得到了广泛的应用[7]，实际上，在视觉领域注意力机制也有重要的应用，用于感受野的捕捉。

提高卷积的感受野，即空间上融合更多特征融合，或者是提取多尺度空间信息，是优化卷积神经网络的重要工作。对于channel维的特征融合，普通的卷积操作默认对输入特征图的所有channel进行融合。而SENet[8]——2017年ImageNet比赛冠军——的创新点在于用attention来关注channel之间的关系，希望模型可以自动学习到不同channel特征的重要程度。为此，SENet提出了Squeeze-and-Excitation 模块，将典型ResNet模块输出特征图作Adaptive Average Pooling，再对其线性变换得到放缩向量，用该放缩向量对原输出作逐元素放缩。

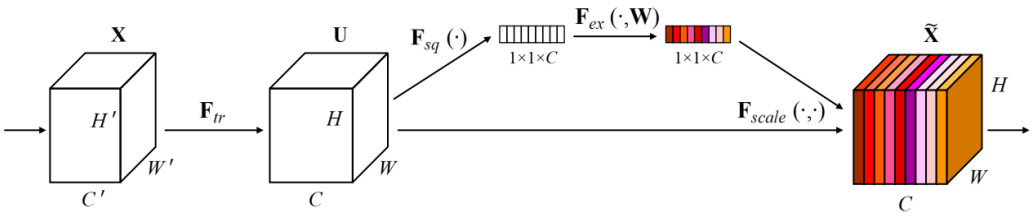


Figure 4: Squeeze-and-Excitation模块，展示了通道维度上的注意力机制

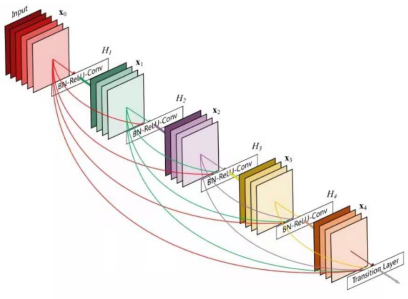


Figure 5: DenseNet密集连接模型

另外，Xiang Li等人提出的SKNet同样把attention机制在深度卷积模型中发扬光大[9]，这些attention模块的设计都是具有灵活性、可移植的，一方面减少了模型参数量，一方面保持或超过了原模型的性能。

2.5 密集卷积网络

在此前卷积神经网络提高效果的方向是要么深（比如ResNet，解决了网络深时候的梯度消失问题）要么宽（比如Inception），而DenseNet则是一种从特征（feature）入手，通过对输入特征的极致利用实现一个密集卷积网络，来达到更好的效果和更少的参数。

DenseNet与ResNet有一些思想上的相似，都是将浅层的信息继续保留到更深层；不过它是一种密集连接机制，在保证网络中层与层之间最大程度的信息传输的前提下，更为极端地直接将所有层连接起来。 具体而言，以第l层输出为例，Densenet首先用[x0,x1,...]表示将0到l-1层的输出做通道的合并，然后第l层输出就是对合并的通道进行一系列变换后再加上第l-1层输出

$$x_l = F([x_0, x_1, ..., x_{l-1}]) + x_{l-1}$$

这一公式就是DenseNet的核心思想，它建立了不同层之间的连接关系，充分利用了feature，进一步减轻了梯度消失问题，也可以再加深网络，而且训练效果非常好。

[10]

3 方法

3.1 Inception-ResNet with Pre-Activation

在Inception-ResNet的原文中提到，在通道数过高时，模型的训练过程容易发生不稳定的情形，因此我们想要提出一种使得训练更加稳定的方法，期望这种方法能加快模型的收敛。在Inception-v2/BN-Inception[11]中提到，ReLU层用来处理准确率饱和问题以及梯度消失问题，但同时它会带来输出空间的不平滑，导致训练过程的不稳定；在Inception-ResNet的[3]原文中也提到，ReLU可能让较深的模型的训练变得不稳定。因此我们认为如果能够减少ReLU层在Inception-ResNet中的作用，训练不稳定的问题很可能得到解决。

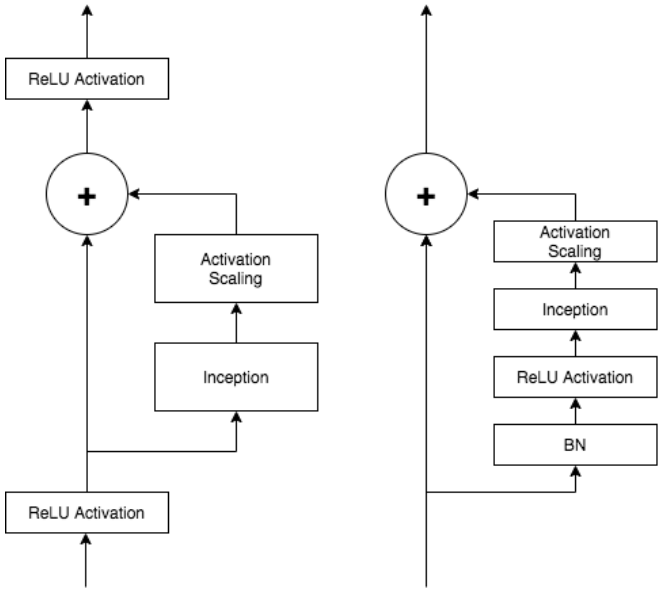


Figure 6: 左边的模型是原有Inception-ResNet Block，右边的模型是我们提出的引入预激活单元的Block

而关于如何从现有的结构改进，我们参考了He等人提出的恒等映射[6]的方法，如图6所示，我们对残差单元保留了一条恒等映射的通道，从而ReLU层的位置得到改变。此外，这种方法本身也能够控制梯度的范围，使每次迭代都能够稳定的变化，从而让整个训练过程更加稳定。经过我们的实验后，在新提出的模型中，训练稳定的同时，收敛速度有所加快，准确率也有些许的提升。

3.2 ResNeAt

为了减少深度卷积模型的参数量，节省计算资源，我们对传统模型进行了许多改进的探索，最终提出了一种基于self-attention机制的方法。我们的模型以ResNet为基础，由于减少参数量的核心算法在于self-attention，我们姑且称新模型为**ResNeAt**。

3.2.1 Bottleneck结构

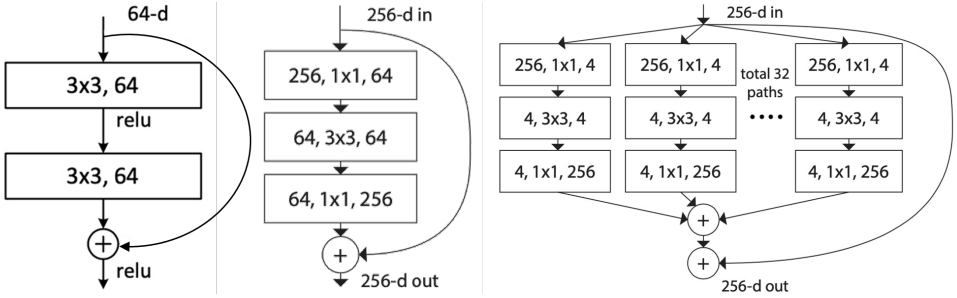


Figure 7: **左**：来自ResNet-34，ResNet-34以该残差模块为基本单元，层层堆叠。**中**：为了加深网络的层数同时保持较低的计算成本，ResNet-50/101/152等模型应用了改进后的bottleneck结构。其特点是应用了1×1卷积核来对特征图升维或降维。**右**：ResNeXt提出的聚合卷积模型，该例子中Cardinality为32。

ResNet不仅提出了残差连接，还提出了一种叫Bottleneck的模块，如图7中图所示，其主要特点是用1×1卷积核，先对特征图降维，再做带padding的3×3卷积，最后用1×1卷积核恢复输入的维度。[1]

ResNeAt也使用bottleneck结构，这样做的目的是，在合理地保持计算复杂度的前提下，提高数据特征图的维度，如图7中三者具有相似的计算复杂度，但bottleneck结构可以处理高维数据。

3.2.2 聚合的卷积变换与Cardinality

ResNeXt提出了将bottleneck模块内部卷积分组化的方法[2]，采用分组、分别做卷积变换、再聚合的思路，引入了Cardinality这一概念，指代分组的数量。例如图7右，是分组化后Cardinality=32的bottleneck模块。

实际上，ResNeAt延伸了聚合变换的概念，ResNeXt中的分割-变换-聚合，是用简单的分组-卷积-求和的方法，而我们使用多个分组卷积，每个分组独立地求出本分组的K/Q/V，最后再用self-attention机制将各通道聚合起来。即，在self-attention的K/Q/V矩阵中，包含多个分组分别得到的子矩阵，而子矩阵的数量实际就是Cardinality。

正如self-attention本身的优势[7]，我们这样做也成功地减少了模型参数量，只用局部的邻接通道来求K/Q/V，进而计算全局的通道与通道之间的关系。

3.2.3 实现Self-Attention

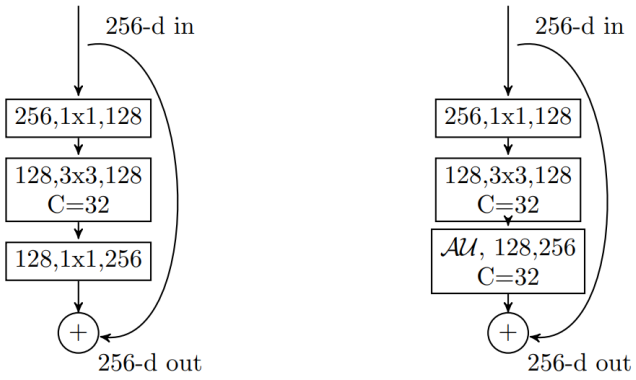


Figure 8: 左：ResNeXt模块，右：ResNeAt模块，其中AU的定义见正文

Attention以及self-attention机制已经在计算机视觉领域得到了广泛应用，我们没有用Stand-Alone Self-Attention中用self-attention直接取代卷积的颠覆性的方法[12]，而是基于深度卷积模型，如图8所示，引入了自注意力机制，旨在使特征图的不同通道学到相互注意力能力。

我们定义卷积自注意力单操作元，称为

$$o = \mathcal{AU}_C(x)$$

这里x为输入，C为Cardinality，o为单元输出，下面我们展开解释。在$\mathcal{AU}$中，我们对输入做

$$\begin{aligned} K &= \mathcal{CV}_C(x) \\ Q &= \mathcal{CV}_C(x) \\ V &= \mathcal{AP}(\mathcal{CV}_C(x)) \\ O &= \mathcal{CV}_C(x) \end{aligned}$$

这里$\mathcal{CV}_C$指分组卷积，组数为C，$\mathcal{AP}$指Adaptive Average Pooling，它对任意一张二维特征图逐元素取平均，得到只有一个元素的特征图。$K/Q/V$矩阵的意义与功能与常见的self-attention定义相同[7]，三者的形状分别为$(ch, s), (ch, s), (ch, 1)$，$ch$为总通道数，$s$为特征图大小。接下来是常见的self-attention变换

$$Scale = \sigma(Softmax(QK^T/\sqrt{s}) \cdot V)$$

注意这里用s的平方根放缩，这一方法也来自于self-attention原论文[7]。得到一个放缩向量$Scale$，形状为$(ch, 1)$由于经过了sigmoid激活，它的元素均在0-1之间。接着，我们对输出向量做放缩

$$o = Scale \odot O + x$$

得到了整个$\mathcal{AU}$的输出。

3.2.4 参数量比较

下面分别计算图8中的ResNeXt和ResNeAt基本单元参数量做比较。左图中，卷积操作总参数量为$256 \times 128 + 128 \times 3 \times 3 \times 128 \div 32 + 128 \times 256 = 70144$，右图中，由于self-attention的存在，ResNeAt基本单元参数量为$256 \times 128 + 128 \times 3 \times 3 \times 4 + 128 \times 256 \times 4 \div 32 = 41472$。另外，同样结构的ResNet，参数量应为$256 \times 128 + 128 \times 3 \times 3 \times 128 + 128 \times 256 = 212992$。可见ResNeAt参数量大大减少。

三个主要模型的具体结构见表1，可见三者结构类似，由上述计算可知，ResNeAt总参数量大约为ResNeXt的59%。

stage	output	ResNet-50	ResNeXt-50	**ResNeAt-50**
conv1	112×112	7×7,64,stride 2	7×7,64,stride 2	7×7,64,stride 2
conv2	56×56	3×3 max pool,stride 2	3×3 max pool,stride 2	3×3 max pool,stride 2
		$\begin{pmatrix} 1\times1,\ 64 \\ 3\times3,\ 64 \\ 1\times1,\ 256 \end{pmatrix} \times3$	$\begin{pmatrix} 1\times1,\ 128 \\ 3\times3,\ 128,\ C{=}32 \\ 1\times1,\ 256 \end{pmatrix} \times3$	$\begin{pmatrix} 1\times1,\ 128 \\ 3\times3,\ 128,\ C{=}32 \\ \mathcal{AU}\ \ 256,\ C{=}32 \end{pmatrix} \times3$
conv3	28×28	$\begin{pmatrix} 1\times1,\ 128 \\ 3\times3,\ 128 \\ 1\times1,\ 512 \end{pmatrix} \times4$	$\begin{pmatrix} 1\times1,\ 256 \\ 3\times3,\ 256,\ C{=}32 \\ 1\times1,\ 512 \end{pmatrix} \times4$	$\begin{pmatrix} 1\times1,\ 256 \\ 3\times3,\ 256,\ C{=}32 \\ \mathcal{AU}\ \ 512,\ C{=}32 \end{pmatrix} \times4$
conv4	14×14	$\begin{pmatrix} 1\times1,\ 256 \\ 3\times3,\ 256 \\ 1\times1,\ 1024 \end{pmatrix} \times6$	$\begin{pmatrix} 1\times1,\ 512 \\ 3\times3,\ 512,\ C{=}32 \\ 1\times1,\ 1024 \end{pmatrix} \times6$	$\begin{pmatrix} 1\times1,\ 512 \\ 3\times3,\ 512,\ C{=}32 \\ \mathcal{AU}\ \ 1024,\ C{=}32 \end{pmatrix} \times6$
conv5	7×7	$\begin{pmatrix} 1\times1,\ 512 \\ 3\times3,\ 512 \\ 1\times1,\ 2048 \end{pmatrix} \times3$	$\begin{pmatrix} 1\times1,\ 1024 \\ 3\times3,\ 1024,\ C{=}32 \\ 1\times1,\ 2048 \end{pmatrix} \times3$	$\begin{pmatrix} 1\times1,\ 1024 \\ 3\times3,\ 1024,\ C{=}32 \\ \mathcal{AU}\ \ 2048,\ C{=}32 \end{pmatrix} \times3$
fc	1×1	global average pool, 1000-d fc		

Table 1: 模型结构

4 实验

基于上述模型与实现方法，我们做了下面的实验。

4.1 实验设置

4.1.1 数据集

我们的数据来源于Google地标分类比赛提供的数据集，这是一个基于GPS定位图像聚类的大规模数据集。原数据集有约15k类地标，共约1200k张图片，对于我们的训练而言太大，因此我们对其进行了精简，具体操作为：

- 选取出现频数最高的前1000个地标，取与它们关联的所有图片（每个地标的图片大概在几百量级，这样重建的数据集大约有200k图片）；
- 以某个比例（如0.1）随机取图片，进一步缩小数据集

最终数据集共包含了1k类共80k余张图片，其中训练集约60k张，验证集和测试集各约10k张。训练图片如9所示，特征提取难度大，存在很多问题。因此我们对这些图片做一些预处理和增广，让它可以更好地在我们的模型上进行训练。这些预处理包括：

- 将所有图片大小放缩成相同的256×256，并随机裁剪为224×224，便于输入输出
- 将图片进行随机翻转和颜色增强的处理，便于细粒度分类

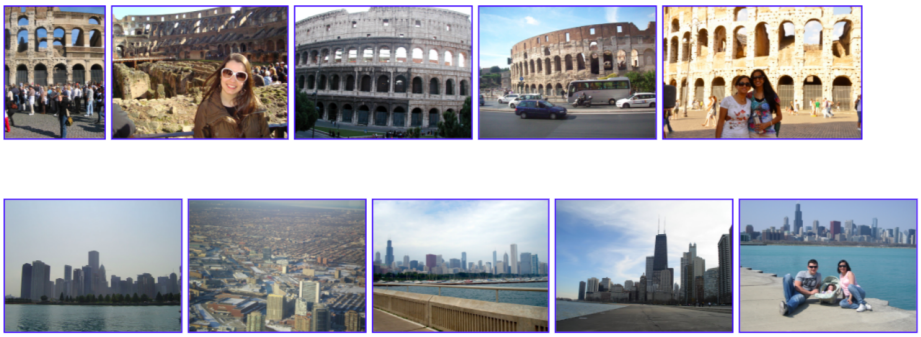


Figure 9: 这些图片，许多可能来自于游客照片，数据噪声大，对于同一建筑，拍摄角度不同、时间不同、尺寸不同，还有背景杂乱、部分遮挡、无关物体乱入等等问题

4.1.2 实验内容

我们的实验一共包括Baseline、复现已有模型、实现创新模型三个部分。

首先在寻找baseline时，由于我们的创新都是基于ResNet网络进行的，而且残差网络已经被证明在更深层次的训练上有很好的表现，因此我们直接将其中的ResNet-50作为baseline，得到了一组实验数据(Top5为结果在前5的准确率，Top1为结果在最正确的准确率)

	Top5 acc	Top1 acc
ResNet50	0.843	0.774

Table 2: Baseline: 用ResNet-50实现的测试集准确率结果

接着我们研读了多种流行的卷积网络设计，借助其论文的参考，自己手写复现了一些近年来提出的更优化的模型，包括ResNeXt、Inception-ResNet、DenseNet、SE-Net，在某些方面可以作为与我们自己模型对比的参考;

最后我们提出了两个新模型，即前文提到的ResNeAt、Inception-Res-PA，并将其与baseline以及ResNeXt、Inception-ResNet进行数据可视化上的比较。

4.1.3 参数设置

为控制变量，所有实验使用几乎相同的一套参数，如下所述：训练数据使用mini-batch，batch大小为64，采用每轮shuffle的机制。初始学习率设置为0.0005，衰减策略为：每轮计算验证集准确率，如果验证集准确率较上一轮有所下降，则以0.5的比例衰减学习率。使用Adam优化策略，betas=(0.9, 0.999)，以交叉熵作为损失函数。所有模型统一训练25轮。

4.2 实验结果

在本数据集上，我们对7个不同模型做了实验。实验最终结果如表3所示。可以看出，baseline的top1准确率达到了77.4%左右，说明我们的数据集分布正常，一定程度上证明了实验结果的有效性。我们必须承认，在所有模型中，我们创新的模型并不是效果最好的，如表所示，SENet在该数据集上训练得到了最高的测试准确率。

- 图10是引入PA后的Inception与baseline、普通Inception的训练曲线对比。可以看出，Inception的训练曲与baseline类似，而引入PA后，Inception模型收敛明显加快，这是预激活机制保证的梯度稳定性的作用。从验证集曲线来看，引入PA的模型泛化能力也更强。在测试集上，Incetpino-Res-PA的top1准确率比Inception-Res高1.7%。
- 图11是ResNeAt与baseline、ResNeXt的对比。可以看出，即使参数规模大大减少，ResNeAt也能达到相似的效果，在收敛速度上甚至略优于ResNeXt。

	Top5 acc	Top1 acc
ResNet	0.843	0.774
ResNeXt	0.855	0.793
DenseNet	0.855	0.799
Inception-Res	0.845	0.783
SE-Net	**0.860**	**0.809**
Inception-Res-PA	0.851	0.800
ResNeAt	0.848	0.784

Table 3: 测试集的实验结果，包括baseline、已有模型、创新模型

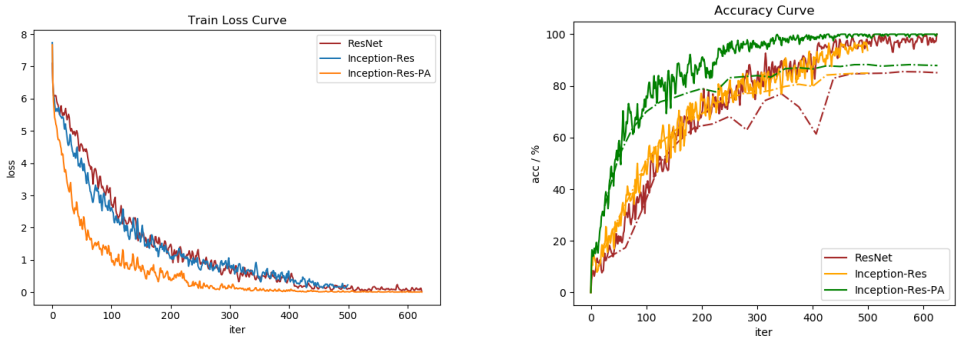


Figure 10: Inception-Res-PA与Inception的对比

5 Honor Code

在本报告中，为了清楚地展示模型，我们使用了多处来自原论文的图片，这些论文都在我们的引用列表中。在本项目中，部分底层模块实现使用了网上的代码。

6 总结

在本次项目中，我们利用Google地标分类比赛的数据集，尝试了多种不同的神经网络，并提出了两个创新的模型（ResNeAt、Inception-Res-PA）。我们较为全面地对比现有的比较流行的几种残差网络对于图像分类问题的表现，而且根据这些表现来改进的两个模型都可以让训练更稳定、收敛速度更快、准确率更高，达到了本次项目的目标；同时它们继承了ResNet让训练层数更深的特性（甚至做得更好），因此我们认为所提出的改进思想可以在其他大数据集上也有更好的表现，这值得以后的工作中进行进一步探究，从而增强其鲁棒性。

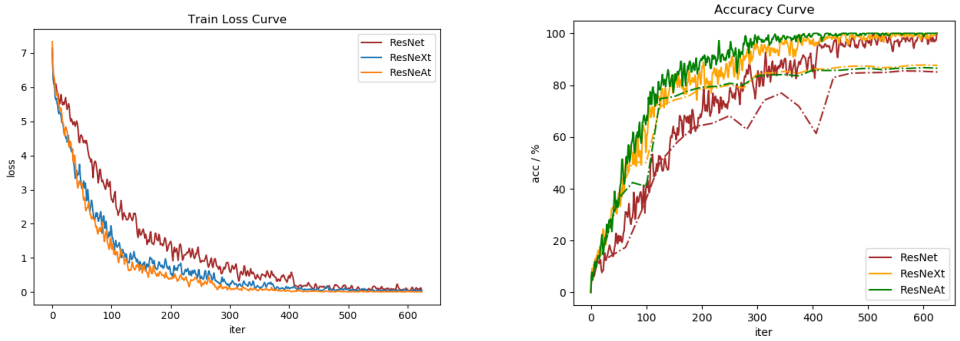


Figure 11: ResNeAt与ResNeXt的对比

我们的工作也有诸多不够完善之处，最主要的是实验不够详尽全面，如果需要进一步说明创新模型的可泛化性，则未来需要设计多方面的对比以及消融实验。

参考文献

References

[1] He, K., X. Zhang, S. Ren, et al. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[2] Xie, S., R. B. Girshick, P. Dollár, et al. Aggregated residual transformations for deep neural networks. *CoRR*, abs/1611.05431, 2016.

[3] Szegedy, C., S. Ioffe, V. Vanhoucke. Inception-v4, inception-resnet and the impact of residual connections on learning. *CoRR*, abs/1602.07261, 2016.

[4] Simonyan, K., A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[5] Szegedy, C., W. Liu, Y. Jia, et al. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9. 2015.

[6] He, K., X. Zhang, S. Ren, et al. Identity mappings in deep residual networks. *CoRR*, abs/1603.05027, 2016.

[7] Vaswani, A., N. Shazeer, N. Parmar, et al. Attention is all you need. *CoRR*, abs/1706.03762, 2017.

[8] Hu, J., L. Shen, G. Sun. Squeeze-and-excitation networks. *CoRR*, abs/1709.01507, 2017.

[9] Li, X., W. Wang, X. Hu, et al. Selective kernel networks, 2019.

[10] Huang, G., Z. Liu, L. van der Maaten, et al. Densely connected convolutional networks, 2018.

[11] Ioffe, S., C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167, 2015.

[12] Ramachandran, P., N. Parmar, A. Vaswani, et al. Stand-alone self-attention in vision models. *CoRR*, abs/1906.05909, 2019.