

基于改良网络的狗品种分类任务

摘要

本文通过对比分析和模型调优，研究了基于 GoogLeNet、ResNet 和 VGG 改良网络的犬品种分类任务。作者首先介绍了深度学习的概念及其在图像识别中的应用，然后详细阐述了 VGG、ResNet 和 GoogLeNet 三种网络模型的结构和特点。在实验设计部分，作者使用了斯坦福犬数据集，并通过数据增强和预处理来提高模型性能。在模型调优方面，作者对三种网络模型进行了改良，包括优化器的选择、超参数的调整以及网络结构的改进。实验结果表明，改良后的网络模型在犬品种分类任务上取得了显著的性能提升。最终，作者对实验结果进行了总结，并对未来的研究方向提出了展望。

关键词

深度学习；图像识别；卷积神经网络；犬品种分类；模型调优

Dog breed classification task based on improved network

Abstract

This paper studies the dog breed classification task based on improved networks of GoogLeNet, ResNet, and VGG through comparative analysis and model tuning. The author first introduces the concept of deep learning and its application in image recognition, then elaborates in detail on the structure and characteristics of the three network models: VGG, ResNet, and GoogLeNet. In the experimental design section, the author uses the Stanford Dogs Dataset and improves model performance through data enhancement and preprocessing. In terms of model tuning, the author has improved the three network models, including the selection of optimizers, adjustment of hyperparameters, and improvements in network structure. The experimental results show that the improved network models have achieved significant performance enhancement in the dog breed classification task. Finally, the author summarizes the experimental results and proposes prospects for future research directions.

Key word

Image recognition; Convolutional neural network; Dog breed classification; Model tuning

目 录

摘要	I
Abstract	I
1 绪论	1
1.1 深度学习	1
1.2 图像识别	1
1.2.1 图像识别技术原理	2
1.2.2 图像识别技术的分析	2
2 基于卷积神经网路的图像分类	4
2.1 VGG 网络模型	4
2.2 ResNet: 残差网络模型	5
2.3 GoogLeNet: 多尺度感受野网络模型	7
3 犬的品种分类任务实验设计	8
3.1 数据集和预处理	8
3.2 网络模型设计和调优	11
3.2.1 ResNet18 模型	13
3.2.2 GoogLeNet 模型	14
3.2.3 VGG 模型	15
4 模型对比分析	17
5 模型分类效果	20
6 实验结论	21
7 设计感言	22
参考文献	22

1 绪论

1.1 深度学习

深度学习（Deep Learning）是机器学习（Machine Learning）领域中一个新的研究方向，它被引入机器学习使其更接近于最初的目标——人工智能（Artificial Intelligence）。

深度学习就是模仿人脑的“神经网络”建立一个类似的学习策略（如图 1），基于大量的数据进行训练和学习，在这个过程中不断修改模型的参数，得到的一个更好的模型，使得这个模型能类似于人一样，识别和处理一些问题，这就是其学习性，并且有些模型的能力甚至能够超越人类的水平。深度学习的最终目标是让机器能够像人一样具有分析学习能力，能够识别文字、图像和声音等数据。

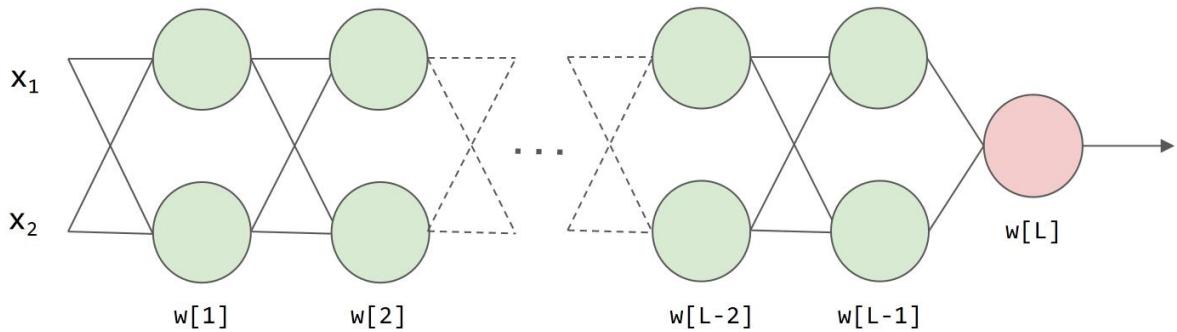


图 1 深度人工神经网络

在过去的几十年里，训练模型的数据集是非常小的，当时的人们难以收集并且构建足够大，有效的数据集，从而导致模型没有充足地得到训练，而随着数字化社会的来临，现在的数据量都非常巨大，制作有效的数据集变得容易得多，模型也就能够得到充足的训练，这也是深度学习兴起的一大重要原因。

同时无论是在 CPU 还是 GPU 上面的，都在不断得发展，使得高计算量的深度学习模型能够更快的得到训练，以往需要训练两个月的数据量，现在只需要三天就能够完成。尤其是在最近这几年，我们也见证了算法方面的极大创新。许多算法方面的创新，一直是在尝试着使得神经网络运行的更快。

1.2 图像识别

图像识别技术是信息时代的一门重要的技术，其产生目的是为了让计算机代替人类去处理大量的物理信息。随着计算机技术的发展，人类对图像识别技术的认识越来越深刻。图像识别技术的过程分为信息的获取、预处理、特征抽取和选择、分类器设计和分类决策。简单分析了图像识别技术的引入、其技术原理以及模式识别等，之后介绍了神经网络的图像识别技术和非线性降维的图像识别技术及图像识别技术的应用。从中可以总结出图像处理技术的应用广泛，人类的生活将无法离开图像识别技术，研究图像识别

技术具有重大意义。

1.2.1 图像识别技术原理

其实，图像识别技术背后的原理并不是很难，只是其要处理的信息比较繁琐。计算机的图像识别技术和人类的图像识别在原理上并没有本质的区别，只是机器缺少人类在感觉与视觉差上的影响罢了。人类的图像识别也不单单是凭借整个图像存储在脑海中的记忆来识别的，我们识别图像都是依靠图像所具有的本身特征而先将这些图像分了类，然后通过各个类别所具有的特征将图像识别出来的，只是很多时候我们没有意识到这一点^[1]。当看到一张图片时，我们的大脑会迅速感应到是否见过此图片或与其相似的图片。其实在“看到”与“感应到”的中间经历了一个迅速识别过程，这个识别的过程和搜索有些类似。在这个过程中，我们的大脑会根据存储记忆中已经分好的类别进行识别，查看是否有与该图像具有相同或类似特征的存储记忆。机器的图像识别技术也是如此，通过分类并提取重要特征而排除多余的信息来识别图像。机器所提取出的这些特征有时会非常明显，有时又是很普通，这在很大的程度上影响了机器识别的速率。总之，在计算机的视觉识别中，图像的内容通常是用图像特征进行描述。

1.2.2 图像识别技术的分析

2015年2月15日新浪科技发布一条新闻：“微软最近公布了一篇关于图像识别的研究论文，在一项图像识别的基准测试中，电脑系统识别能力已经超越了人类。人类在归类数据库 Image Net 中的图像识别错误率为 5.1%，而微软研究小组的这个深度学习系统可以达到 4.94% 的错误率。”这也说明未来图像识别技术有更大的研究意义与潜力。而且，计算机在很多方面确实具有人类所无法超越的优势，也正是因为这样，图像识别技术才能为人类社会带来更多的应用^[2]。

1) 非线性降维的图像识别技术

计算机的图像识别技术是一个异常高维的识别技术。不管图像本身的分辨率如何，其产生的数据经常是多维性的，这给计算机的识别带来了非常大的困难。想让计算机具有高效地识别能力，最直接有效的方法就是降维。降维分为线性降维和非线性降维。例如主成分分析（PCA）和线性奇异分析（LDA）等就是常见的线性降维方法，它们的特点是简单、易于理解。但是通过线性降维处理的是整体的数据集合，所求的是整个数据集合的最优低维投影。经过验证，这种线性的降维策略计算复杂度高而且占用相对较多的时间和空间，因此就产生了基于非线性降维的图像识别技术，它是一种极其有效的非线性特征提取方法。此技术可以发现图像的非线性结构而且可以在不破坏其本征结构的基础上对其进行降维，使计算机的图像识别在尽量低的维度上进行，这样就提高了识别速率。例如人脸图像识别系统所需的维数通常很高，其复杂度之高对计算机来说无疑是

巨大的“灾难”。由于在高维度空间中人脸图像的不均匀分布，使得人类可以通过非线性降维技术来得到分布紧凑的人脸图像，从而提高人脸识别技术的高效性。

2) 神经网络的图像识别技术

神经网络图像识别技术是在传统的图像识别方法和基础上融合神经网络算法的一种图像识别方法。这里的神经网络是指人工神经网络，也就是说这种神经网络并不是动物本身所具有的真正的神经网络，而是人类模仿动物神经网络后人工生成的。在神经网络图像识别技术中，遗传算法与 BP 网络相融合的神经网络图像识别模型是非常经典的，在很多领域都有它的应用。在图像识别系统中利用神经网络系统，一般会先提取图像的特征，再利用图像所具有的特征映射到神经网络进行图像识别分类。以汽车拍照自动识别技术为例，当汽车通过的时候，此时检测设备就会启用图像采集装置来获取汽车正反面的图像。获取了图像后必须将图像上传到计算机进行保存以便识别。最后车牌定位模块就会提取车牌信息，对车牌上的字符进行识别并显示最终的结果。在对车牌上的字符进行识别的过程中就用到了基于模板匹配算法和基于人工神经网络算法。

表 1 为常用图像识别方法的对比。在实际应用环境中，由于瑕疵形态多样且存在噪声干扰等特性，各种检测和识别方法都有其优势和不足。在某些情况下，为了提高检测精度，可能会综合使用多种方法。大多数算法是为特定小规模数据集设计的，因此它们的泛化能力较弱，检测结果容易受到图像质量、现场环境、倾斜角度等因素的影响，特别是在背景条件复杂的情况下。

表 1 常用图像识别方法对比

分类	方法	效果	缺点
检测	Canny 边缘检测 和霍夫变换	速度快，精度低	对图像倾斜敏感
	投影法	速度快，精度低	对光斑、污渍敏感
识别	模板匹配	速度快，精度低	计算成本高，抗干扰差
	SVM	速度中等，精度中等	特征提取要求高
	三层 BP 神经网络	速度慢，精度高	易出现过拟合

随着深度学习的发展，由于卷积神经网络自带的特征提取能力，其在图像识别任务上表现出优越的性能，这使其逐渐发展成为目前图像识别领域的主流方法。

2 基于卷积神经网路的图像分类

卷积神经网络的发展可以追溯至 20 世纪 80 年代, 其结构只是简单的神经网络结构。2012 年, Krizhevsky 提出了一种卷积神经网络模型 AlexNet 该模型用于识别 ImageNet 数据集, 包含 8 层卷积层, 3 层全连接层, 采用非线性激活函数和 Dropout 等技术, 同时利用深度网络的设计和并行计算的手段, 使得 ImageNet 的误差降低了约 12.4% , 开创了图像分类任务高准确率的先河。随后十多年里, 陆陆续续推出了各种先进的卷积神经网络模型, 2014 年, Simonyan 和 Zisserman 提出 VGG 模型; 同年谷歌提出了 Inception 模型; 2015 年, He 等人提出了一种残差连接的结构 ResNet 模型; 2017 年, Huang 等人提出了密集连接的 DenseNet^[3]; 2020 年, Dosovitskiy 等人提出 Vision-Transformer 模型。为了更好地提高图像识别的性能, 很多研究开始通过结合其他机器学习技术来增强卷积神经网络的性能。本文接下来详细讲解其中几种网络模型。

2.1 VGG 网络模型

VGGNet 是比较深的卷积神经网络^[4], 包含 16 个或 19 个卷积层和池化层, 可以提取比较复杂的图像特征, 在一些具有挑战性的图像识别任务中取得了非常好的成绩。由于 VGGNet 采用较小卷积核, 使得模型具有较小的复杂度, 能够有效地提取图像的特征, 识别性能较好。但是 VGGNet 极其深的网络结构导致了其极大的计算复杂度, 需要更多的计算资源甚至 GPU, 并且容易出现模型过拟合问题, 需要采用适当的方法进行正则化。

图 2 是 VGG-16 的网络结构示意图, 有 13 层卷积和 3 层全连接层。VGG 网络的设计严格使用 3×3 的卷积层和池化层来提取特征, 并在网络的最后面使用三层全连接层, 将最后一层全连接层的输出作为分类的预测。

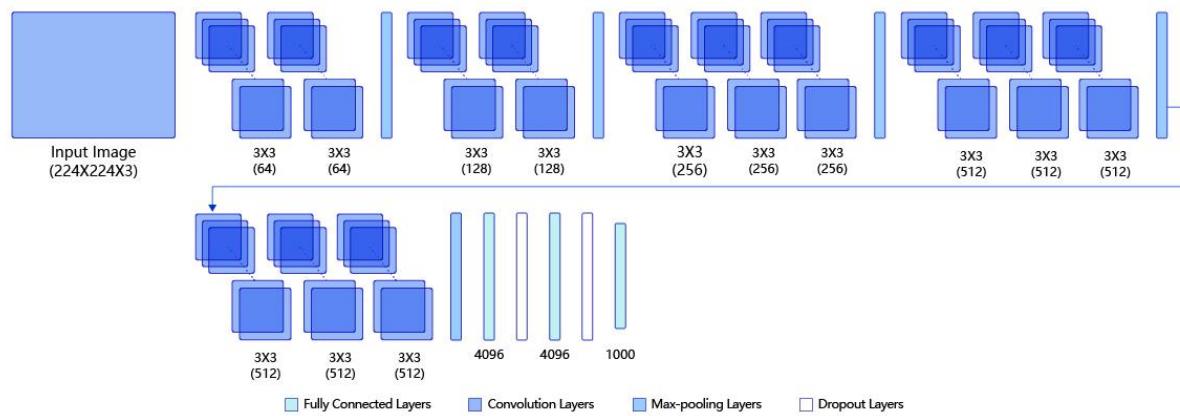


图 2 VGG 模型网络结构示意图

整个网络都使用了同样大小的卷积核尺寸 3×3 和最大池化尺寸 2×2 。 1×1 卷积的意义主要在于线性变换, 而输入通道数和输出通道数不变, 没有发生降维。2 个 3×3 的卷积层串联相当于 1 个 5×5 的卷积层, 感受野大小为 5×5 。同样地, 3 个 3×3 的卷积层串

联的效果则相当于 1 个 7×7 的卷积层。这样的连接方式使得网络参数量更小，而且多层次的激活函数令网络对特征的学习能力更强。

VGG 中还有一个显著特点：每次经过池化层（maxpooling）后特征图的尺寸减小一倍，而通道数增加一倍（最后一个池化层除外）。在 VGG 中每层卷积将使用 ReLU 作为激活函数，在全连接层之后添加 dropout 来抑制过拟合。使用小的卷积核能够有效地减少参数的个数，使得训练和测试变得更加有效。比如使用两层 3×3 卷积层，可以得到感受野为 5 的特征图，而比使用 5×5 的卷积层需要更少的参数。由于卷积核比较小，可以堆叠更多的卷积层，加深网络的深度，这对于图像分类任务来说是有利的。VGG 模型的成功证明了增加网络的深度，可以更好的学习图像中的特征模式。

VGGNet 在训练时有一个小技巧，先训练浅层的网络 VGG11，再复用 VGG11 的权重来初始化 VGG13，如此反复训练并初始化 VGG19，能够使训练时收敛的速度更快。

2.2 ResNet：残差网络模型

ResNet 是一个非常深的卷积神经网络，最多可以达到 152 层，该网络的主要创新之处在于使用了残差学习方法，有效地解决了训练极深的网络时出现梯度消失的难题^[5]。ResNet 可以构建更深的神经网络，从而提高了分类准确度，如 ResNet-101 和 ResNet-152 等都表现出了很好的效果。但 ResNet 的网络结构相对较为复杂，其训练时间相比 VGG 等模型要长，因此需要更大的训练集和计算资源来支持。

从理论上来讲，加深深度学习网络可以提升性能。深度网络以端到端的多层次方式集成了低/中/高层特征和分类器，且特征的层次可通过加深网络层次的方式来丰富。举一个例子，当深度学习网络只有一层时，要学习的特征会非常复杂，但如果有多层，就可以分层进行学习，如图 3 所示^[6]，网络的第一层学习到了边缘和颜色，第二层学习到了纹理，第三层学习到了局部的形状，而第五层已逐渐学习到全局特征。网络的加深，理论上可以提供更好的表达能力，使每一层可以学习到更细化的特征。

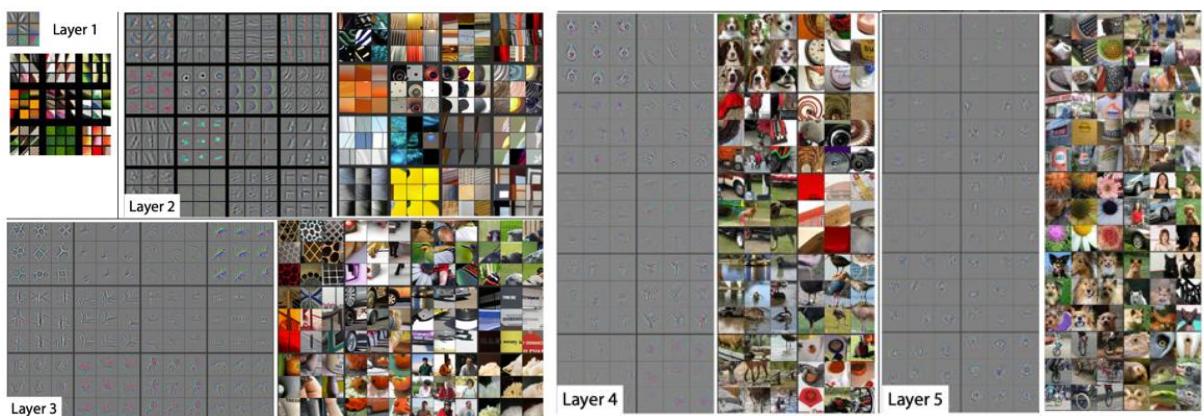


图 3 特征可视化

然而，通过堆叠层数加深网络并不一定是有效的。最显著的问题就是梯度消失或梯度爆炸。我们都应该知道神经网络的参数更新依靠基于链式求导的梯度反向传播（Back Propagation），而 sigmoid 的导数最大值为 0.25，那么随着网络层数的增加，小于 1 的小数不断相乘导致逐渐趋近于零，从而产生梯度消失。梯度爆炸也是同样的道理，当权重初始化为一个较大值时，虽然和激活函数的导数相乘会减小这个值，但是随着神经网络的加深，梯度呈指数级增长，就会引发梯度爆炸。但是从 AlexNet 开始，神经网络中就使用 ReLU 函数替换了 Sigmoid，同时 BN（Batch Normalization）层的加入，也基本解决了梯度消失/爆炸问题。

遗憾的是，在解决了梯度消失/爆炸后，我们仍然无法通过堆叠层数来达成总是有效的网络模型。如图 4，很明显，56 层的深层网络，在训练集和测试集上的表现都远不如 20 层的浅层网络，这种随着网络层数加深，accuracy 逐渐饱和，然后出现急剧下降，具体表现为深层网络的训练效果反而不如浅层网络好的现象，被称为网络退化。

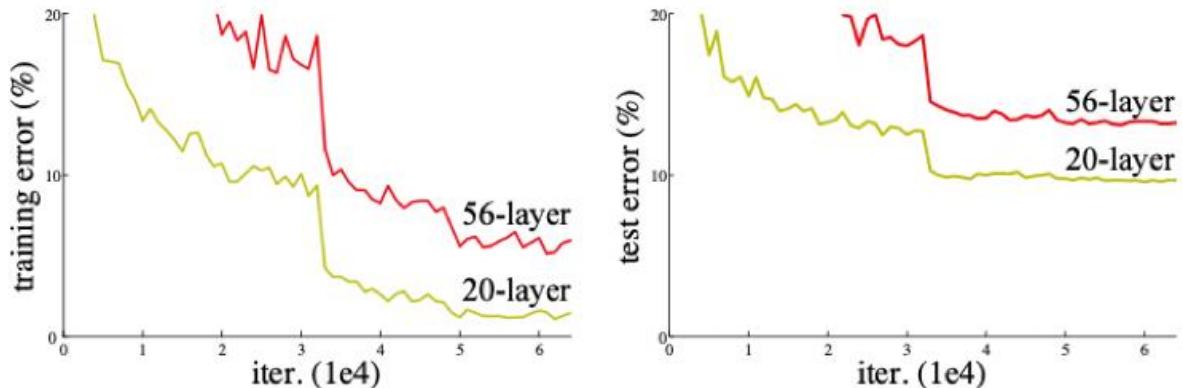


图 4 CIFAR-10 上 20 层和 56 层网络的训练误差和测试误差

理论上来讲，当浅层网络效果不错的时候，网络层数的增加即使不会引起精度上的提升也不该使模型效果变差。但事实上非线性的激活函数的存在，会造成很多不可逆的信息损失，网络加深到一定程度，过多的信息损失就会造成网络的退化。

为了加深网络结构，使每一次能够学到更细化的特征从而提高网络精度，需要实现的一点是恒等映射。而 ResNet 就是提出的残差结构让网络拥有恒等映射能力，即随着网络层数的增加，深层网络至少不会差于浅层网络。

残差结构的目的是，随着网络的加深，使逼近于 0，使得深度网络的精度在最优浅层网络的基础上不会下降。直接选取最优的浅层网络同样能达到较高的精度，然后最优的浅层网络结构并不易找寻，而 ResNet 可以通过增加深度，找到最优的浅层网络并保证深层网络不会因为层数的叠加而发生网络退化。

2.3 GoogLeNet：多尺度感受野网络模型

GoogLeNet 是一个比较新颖的卷积神经网络^[7]，具有 22 个层，该网络模型引入了 Inception 模块、global average pooling 层等创新性的结构，可以大大降低模型的参数量，同时提高了准确率。然而 GoogLeNet 的模型非常深，特别是在训练中期存在梯度消失和爆炸的风险。因此需要进行精细的参数初始化和正则化，甚至需要使用残差结构来解决，模型结构也比较复杂，计算时间和计算成本较大。GoogLeNet 网络模型如图 5。

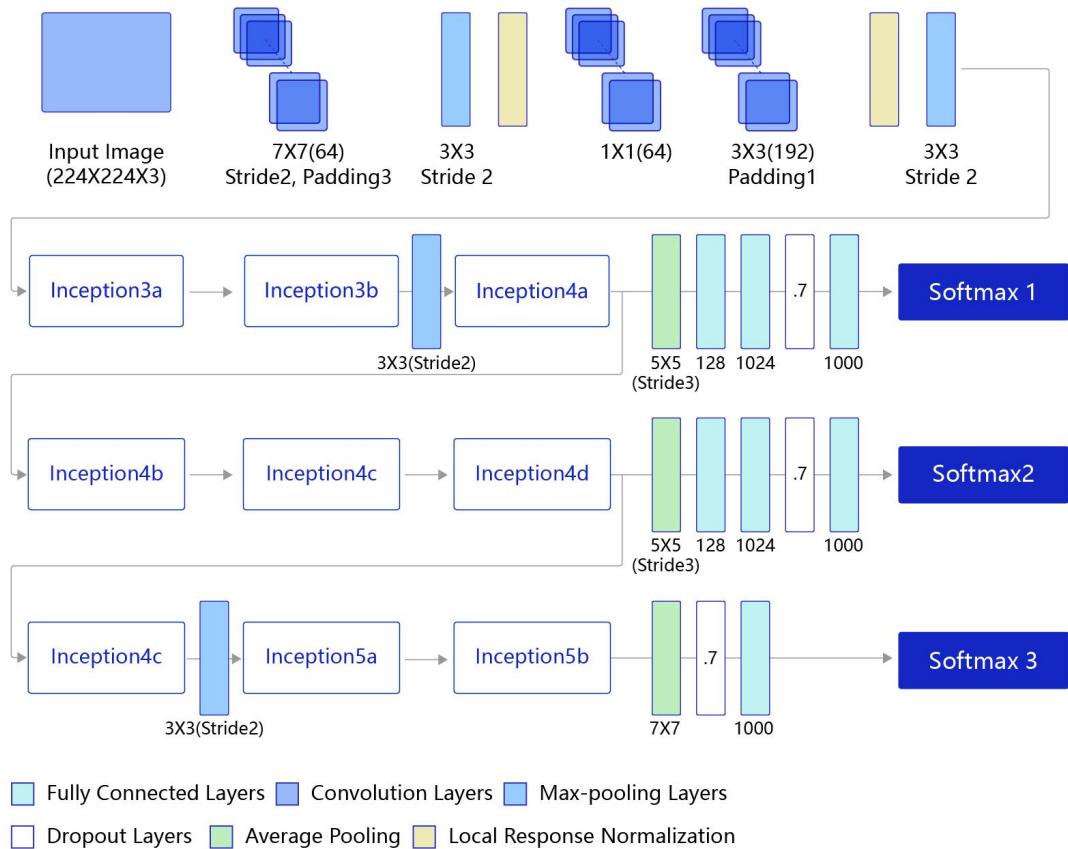


图 5 GoogLeNet 模型网络结构示意图

由于图像信息在空间尺寸上的巨大差异，选择合适的卷积核提取特征是一个挑战。空间分布范围更广的图像信息适合用较大的卷积核提取其特征；而空间分布范围较小的图像信息则适合用较小的卷积核提取其特征。为了解决这个问题，GoogLeNet 提出了一种被称为 Inception 模块的方案。Inception 模块的设计使用 3 个不同大小的卷积核对输入图片进行卷积操作，并附加最大池化，将这 4 个操作的输出沿着通道这一维度进行拼接，构成的输出特征图将会包含经过不同大小的卷积核提取出来的特征，从而达到捕捉不同尺度信息的效果。Inception 模块采用具体采用多通路(multi-path)的设计形式，每个支路使用不同大小的卷积核，最终输出特征图的通道数是每个支路输出通道数的总和，这将会导致输出通道数变得很大，模型参数量会变得非常大。为了减小参数量，在每个 3x3 和 5x5 的卷积层之前，增加 1x1 的卷积层来控制输出通道数；在最大池化层后面增加 1x1 卷积层减小输出通道数。

3 犬的品种分类任务实验设计

本次实验的目标在于对实验模型进行对比探究和性能调优。

3.1 数据集和预处理

斯坦福犬数据集 (Stanford Dogs Dataset) 包含来自世界各地的 120 种犬类的图像。该数据集是使用 ImageNet 中的图像和注释方法构建的，常用于细粒度图像分类任务。

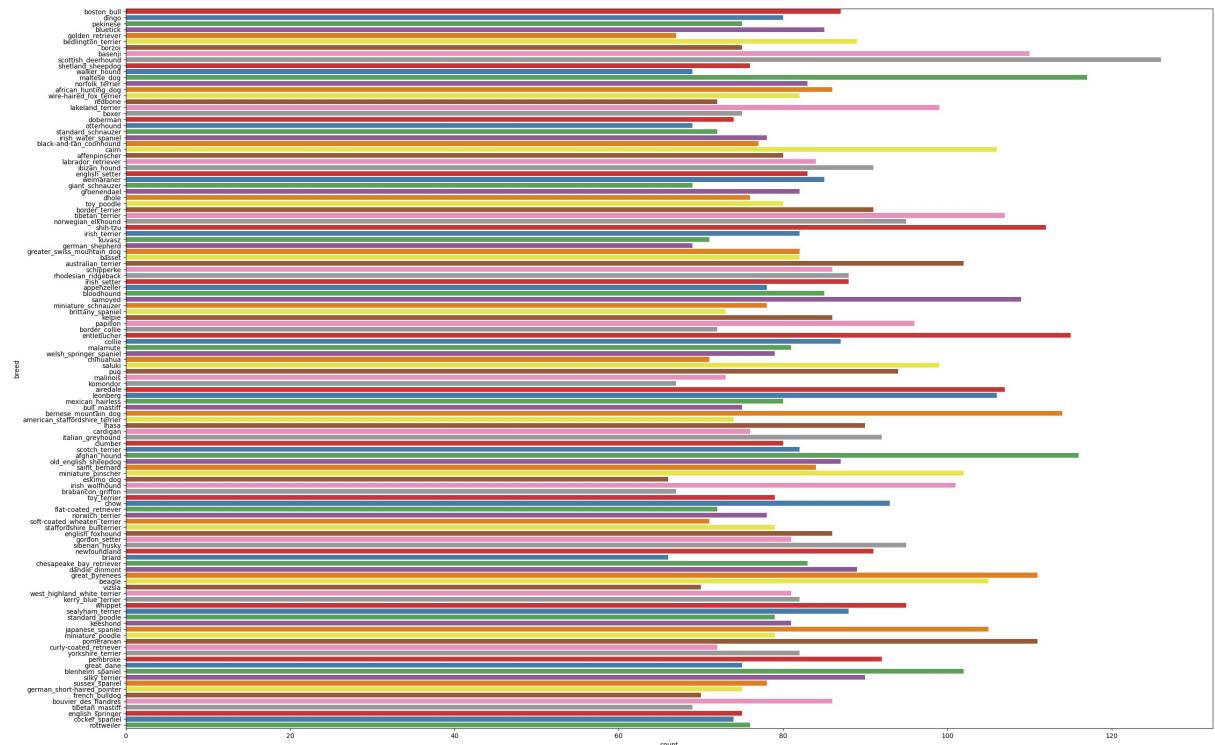


图 6 120 种犬类图像数量统计可视化

犬数据集共计 20580 张图像，是中等规模数据集。每一犬类图像的数量统计如图 6，可以清晰的看出，各类别的图像数量相对均匀，这对模型的无偏差学习至关重要。



图 7 随机抽样可视化

图 7 为训练图像的随机抽样可视化，可以看到，多数图像主体突出，轮廓清晰，而我们后续更大规模的抽样检查更加印证了以上描述。这有利于模型更轻易的学习到犬类的各种尺度的特征。

在学习了图像数据增强章节及相关参考资料后，多数都默认了数据增强操作。例如图 8 将数据增强分为训练集增强和测试集增强，先后做了随机旋转、中心开始裁剪、随机水平翻转、调整色彩和明暗、设置灰度率等操作。

```
data_transforms = {
    'train': transforms.Compose([
        transforms.RandomRotation(45),
        transforms.CenterCrop(224),
        transforms.RandomHorizontalFlip(p=0.5),
        transforms.RandomVerticalFlip(p=0.5),
        transforms.ColorJitter(brightness=0.2, contrast=0.1, saturation=0.1, hue=0.1),
        transforms.RandomGrayscale(p=0.025),
        transforms.ToTensor(),
        transforms.Normalize(mean: [0.485, 0.456, 0.406], std: [0.229, 0.224, 0.225])
    ]),
    'test': transforms.Compose([
        transforms.Resize(256),
        transforms.CenterCrop(224),
        transforms.ToTensor(),
        transforms.Normalize(mean: [0.485, 0.456, 0.406], std: [0.229, 0.224, 0.225])
    ]),
}
```

图 8 默认的图像数据增强操作

然而，经过实验对比发现，对于犬类数据集，无需增强，简单的预处理如 Resize 和 Normalize（归一化）就已足够模型学习并取得较好的性能。

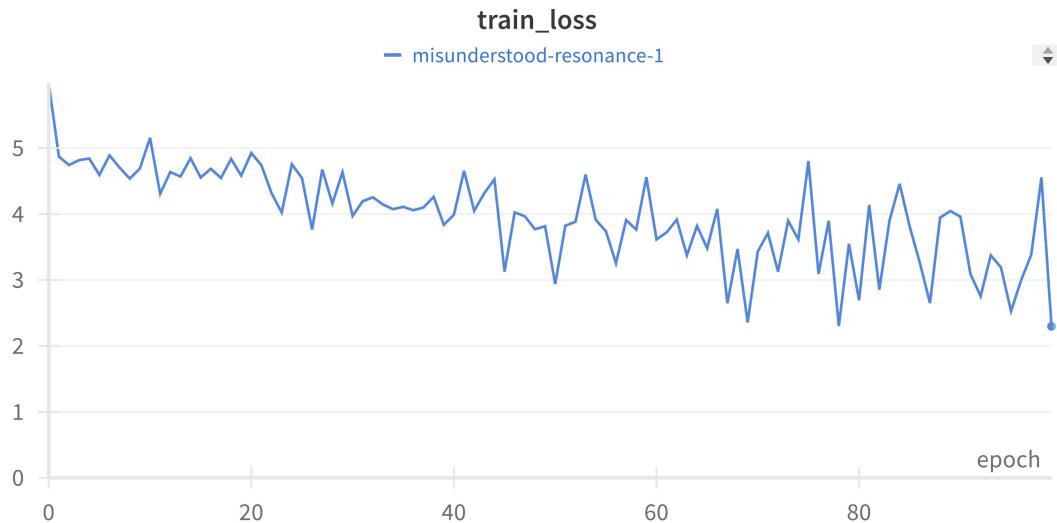


图 9 数据增强对照组

如图 9，这是在保持其他如模型、学习率、batch-size 等变量的情况下，数据增强对照组的训练损失曲线，可以观察到损失曲线振动频繁，在 60 到 80 个 epoch 之间还有一段损失大尺度上行的阶段。整体损失难以下降，直到 99 个 epoch 之后损失值才接近 2。



图 10 数据增强实验组

图 10 为采取无数据增强的实验组，同样控制了其他变量不变。很明显可以看出，实验组的训练损失曲线更加稳定，震荡较少，收敛速度更快，只在 20 个 epoch 时损失值就接近 2 了，并且有加速下降的趋势。

对比来看，可以得出结论，数据增强操作对于本数据集是不必要的。这里我们分析给出以下两种可能的解释：

(1) 数据足够多：犬类数据集共计 20580 张图像，是中等规模数据集，数据量足够大以至于模型“吃饱”，已经足够模型学习到具有鲁棒性的特征。

(2) 色彩增强不当：我们将前文的增强操作可视化对比，如图 11 的色彩增强对比。可以看到色彩增强后，图片相对于原图色彩失真了，而这样的色彩变换并不对所有图像具有一致性，即图像色彩越复杂，色彩变换差异越大，这反而相当于为模型的学习添加噪声，可能对模型依靠色彩来辨别犬类造成不利。

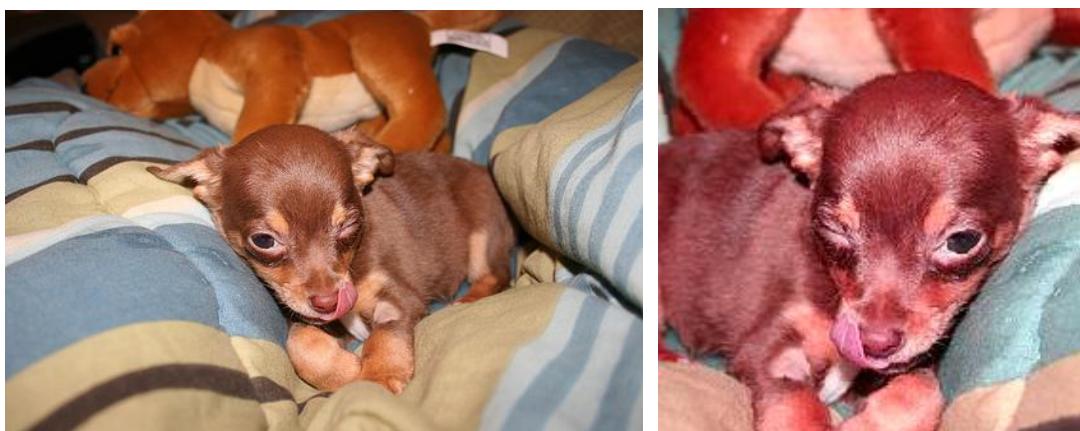


图 11 训练集中原图（左）和色彩增强图（右）

(3) 剪裁增强不当：中心剪裁有利于突出图像主体、去除干扰背景。然而对于复杂的真实数据集，作为主体的犬只存在较大概率并不位于中心。如图 12，剪裁后的犬只剩下了身子，头部丢失。我们随后进行了更多次的随机检查，发现这样的情况不在少数，这对模型的持续学习和分辨造成了干扰，也可能是训练损失曲线震荡的原因。



图 12 训练集中原图（左）和中心剪裁增强图（右）

因此，数据增强操作对于本数据集是不必要的。不仅加重计算负担，甚至有可能带来额外的噪声。总结来说，在该数据集下数据增强对模型性能的提升没有帮助。

3.2 网络模型设计和调优

针对犬类的品种分类任务，我们采用了 ResNet、GoogLeNet、VGG 三个经典的网络模型进行实验。整个实验主要分为优化器实验、超参数实验、模型实验。

在优化器实验中，我们主要选择对比了 mini-batch SGD 优化器和 Adam 优化器。

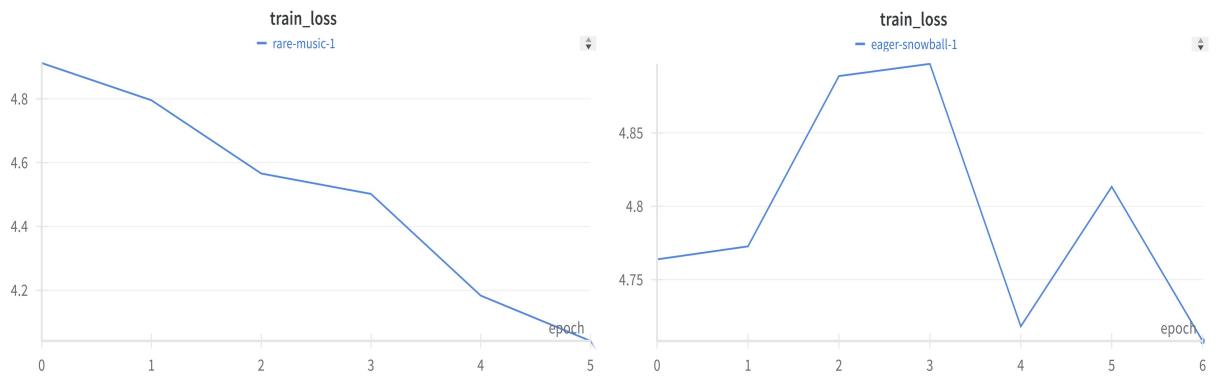


图 13 SGD（左）Adam（右）

如图 13，我们控制其他变量，对 SGD 和 Adam 训练损失曲线进行对比。值得一提的是，两者的学习率（Learning Rate）都设置为了 0.01，SGD 额外设置了 0.9 的动量项。可以看到两者具有相当大的差距，Adam 在 4.75 的损失值附近水平震荡，无法收敛；而 SGD 一路下行，损失快速下降。

为了使两者能够公平对决，我们进一步实验认为 Adam 在 0.001 的学习率下有不错的收敛能力，故安排两者在各自最优的设置下进行实验对比。



图 14 ResNet18 下 SGD 优化器最优性能



图 15 ResNet18 下 Adam 优化器最优性能

图 14 为 SGD 最佳性能下的训练损失曲线，可以看到，经过 25 个 epoch 后损失降为 0，这意味着模型拟合得很好，获得了最高的准确率；在第 5 个和第 10 个 epoch 之间出现了损失曲线“抬头”，而后又调转之下的现象。图 15 为对照组的 Adam，可以看到，其损失曲线相对于 SGD 来说更加平滑，但在损失值为 1 时发生了损失停滞的现象。

理论上，带一阶动量的 SGD 的下降方向是该位置的一阶动量方向。而 Adam 自适应学习率优化算法为每个参数设定了不同的学习率，在不同维度上设定不同步长，因此其下降方向是缩放过 (scaled) 的一阶动量方向。

通过图 14 和图 15 的对比结合理论分析，由于自适应的策略，Adam 优化器在梯度下降的尺度上相对较小，虽然有益于快速收敛，但也容易发生图 15 的停滞现象。而 SGD 优化器刚好相反，拥有像图 14 所展现的那样跳出局部最优的能力。

在超参数实验中，主要对比了学习率（Learning Rate）、批大小（Batch Size）的参数大小影响。学习率已经在前文做了探讨，而批大小可以在过拟合与欠拟合之间寻找平衡，在实验后，我们认为设置到 16 最合适。然而，我们仍然对 16 的批大小带来积极影响保持怀疑，因为其都在 SGD 优化器的背景下，我们认为存在 16 的批大小仅适用于 SGD 情形下的可能（mini-batch SGD 刚好要求批大小在合适的情况下尽可能小）。

经过以上实验，我们确认了针对该数据集数据分布最佳的优化器及其参数设置，即在学习率为 0.01、批大小为 16 的前提下，mini-batch SGD 带 0.9 权重的动量。此外，为了更好的解决实际训练中遇到的问题，我们在不改变经典模型设计思想的前提下，有针对性的对网络进行设计改良。结果表明，我们的改良取得了显著的效果。

3.2.1 ResNet18 模型

对于 ResNet18 模型，我们遵循了原论文给出的设计。然而在训练过程中，出现了损失一开始就无法下降的问题。这可能是潜在的梯度大小变化率过大导致的。因此，我们在每一基础卷积层中引入 2D 的批归一化（Batch Normalization），解决了因网络层与层之间的数据分布漂移导致的梯度大小变化率过大的问题。

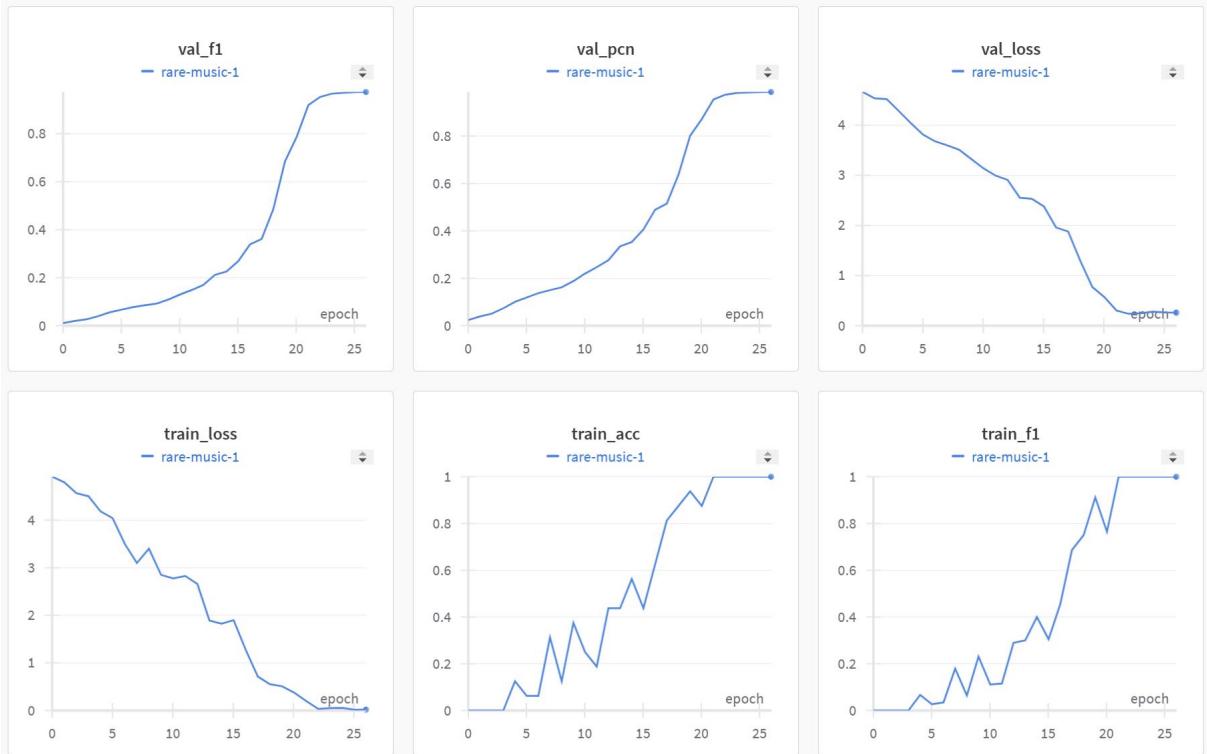


图 16 ResNet18 模型评价指标曲线

图 16 为 ResNet18 模型评价指标曲线，可以看到无论是准确率还是 f1 分数，这些指标都针对该次模型训练的性能表现给出一致的高评价（测试集同样）。其中 F1 分数是精确率和召回率的调和平均数，它试图平衡两者。F1 分数的值范围在 0 到 1 之间，1 表示完美的精确率和召回率，0 表示两者都很差。因此可以看出，该次训练性能接近完美。

3.2.2 GoogLeNet 模型

对于 GoogLeNet 模型，由于原论文的设计特别针对 1000 路图像分类，因此模型做的很深，复杂度很高。对于本次实验数据集的 120 路图像分类，原设计很容易过拟合和退化，并且浪费实验时间和计算资源。因此，我们在原设计的基础上降解了模型，并且加入了新的自己的设计思路。

```
self.front = Front()
# (N, ch1x1 + ch3x3_1 + ch3x3_2 + pool_ch, Hin, Win)
self.inception3a = Inception(in_channels: 192, ch1x1: 64, ch3x3_1_1: 128, ch3x3_1: 240, ch3x3_2_1: 32, ch3x3_2: 144, pool_ch: 64)
self.inception3b = Inception(in_channels: 512, ch1x1: 128, ch3x3_1_1: 224, ch3x3_1: 360, ch3x3_2_1: 64, ch3x3_2: 216, pool_ch: 64)
self.max_pool3 = nn.MaxPool2d(kernel_size: 3, stride=2, ceil_mode=True)

self.inception4a = Inception(in_channels: 768, ch1x1: 256, ch3x3_1_1: 512, ch3x3_1: 400, ch3x3_2_1: 432, ch3x3_2: 240, pool_ch: 128)
self.inception4b = Inception(in_channels: 1024, ch1x1: 384, ch3x3_1_1: 618, ch3x3_1: 960, ch3x3_2_1: 468, ch3x3_2: 576, pool_ch: 128)
self.max_pool4 = nn.MaxPool2d(kernel_size: 3, stride=2, ceil_mode=True)

if self.aux_logit:
    self.aux = InceptionAux(in_channels: 2048, self.n_classes)

self.avg_pool = nn.AdaptiveAvgPool2d((1, 1))
self.dropout = nn.Dropout(0.4)
self.fc = nn.Linear(in_features: 2048, self.n_classes)
```

图 17 改良的 GoogLeNet 代码设计

如图 17，我们将原设计的 9 层 inception 模块堆叠降解为了 4 层，并在每层通道数的设置上融合了自己的设计理念，即主要分为细节学习和完形学习。细节学习用于学习图像的细节性特征如纹理明暗等；完形学习组合前期学习到的细节性特征，并进一步拼凑学习全局特征。基于此，可以推断出应先细节学习后接着完形学习，并且细节学习的特征应尽可能多而丰富，以供完型学习有充分的特征进行拼凑辨认。

于是，将 inception 模块进行分工，前两层 inception 负责细节学习，后两层 inception 负责完形学习。不仅如此，在 inception 内部依据通道数的不同再分出三个层次：不同分支分别承担轻、中、重的学习任务，具体就是按照轻、中、重的程度按比例分配通道数。其中，小的感受野适合学习细节，因此通道数分配占比较大；大的感受野适合学习完形，因此通道数分配占比较小。然而，不同 inception 又轻微改变任务划分比例。

理论上分析，这样设计提倡学习任务的精细分工，将最大程度的提高模型学习效率。

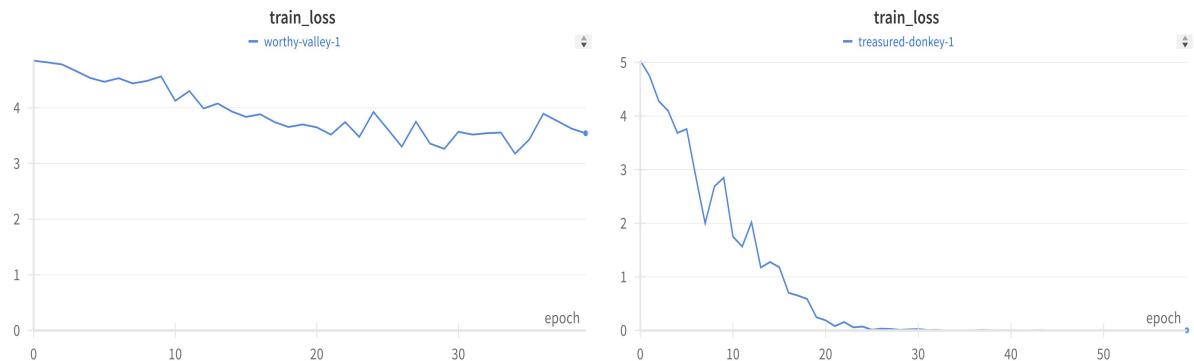


图 18 GoogLeNet 原设计（左）和改良设计（右）损失曲线对比

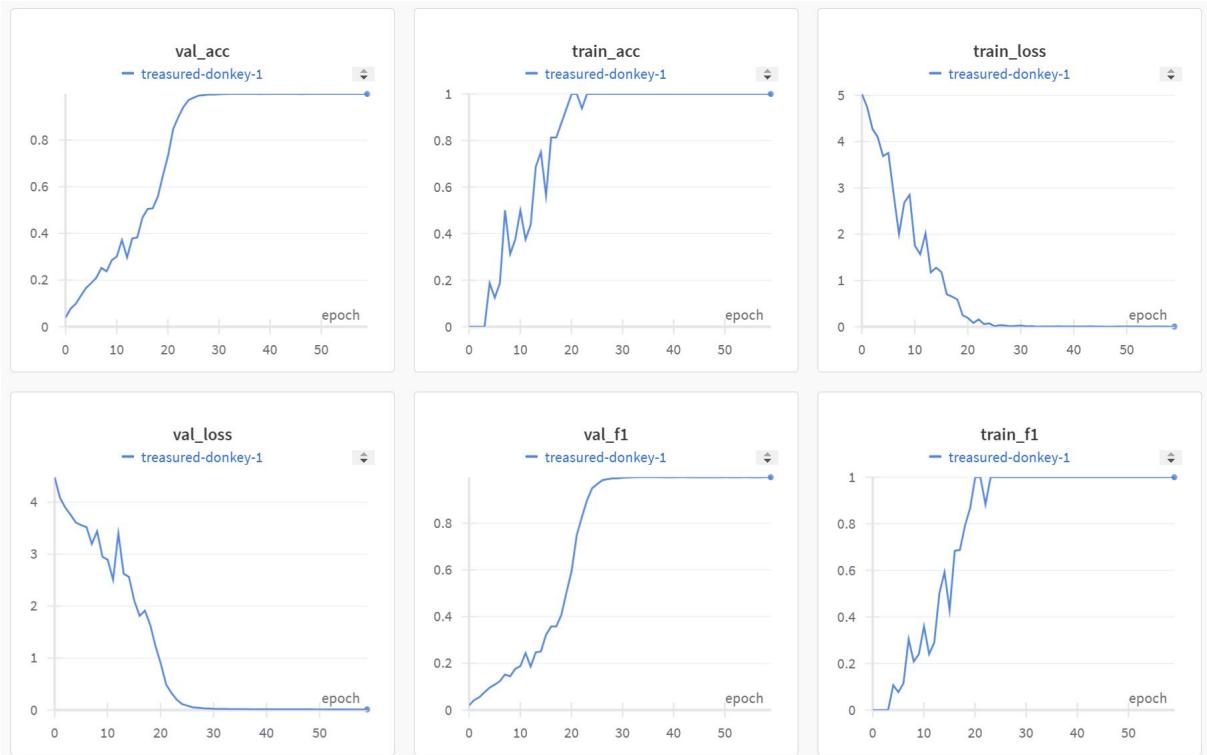


图 19 改良 GoogLeNet 模型评价指标曲线

结合图 18 和图 19，可以看到，相比 GoogLeNet 原设计的损失震荡不收敛，改良的 GoogLeNet 不仅收敛且收敛速度很快，在第 20 个 epoch 就达成收敛（超过前述 ResNet18 的第 23 个 epoch 收敛），准确率等评价指标也实现了对 ResNet18 的超越。如果说上次 ResNet18 的训练性能接近完美，那么该次改良 GoogLeNet 的训练性能就是完美。

3.2.3 VGG 模型

对于 VGG16 模型，我们同样对模型进行降解和重新设计，如图 20。

```
# Conv blocks (BatchNorm + ReLU activation added in each block)
self.layer1 = self.vgg_conv_block(in_list: [3, 64], out_list: [64, 64], k_list: [3, 3], p_list: [1, 1], pooling_k: 2, pooling_s: 2)
self.layer2 = self.vgg_conv_block(in_list: [64, 128], out_list: [128, 128], k_list: [3, 3], p_list: [1, 1], pooling_k: 2, pooling_s: 2)
self.layer3 = self.vgg_conv_block(in_list: [128, 256], out_list: [256, 256], k_list: [3, 3], p_list: [1, 1], pooling_k: 2, pooling_s: 2)
self.layer4 = self.vgg_conv_block(in_list: [256, 512], out_list: [512, 512], k_list: [3, 3], p_list: [1, 1], pooling_k: 2, pooling_s: 2)
self.layer5 = self.vgg_conv_block(in_list: [512], out_list: [1024], k_list: [3], p_list: [1], pooling_k: 2, pooling_s: 2)

# Final layer
self.avg_pool = nn.AdaptiveAvgPool2d((1, 1))
self.dropout = nn.Dropout(0.4)
self.layer8 = nn.Linear(in_features: 1024, self.n_classes)
```

图 20 改良的 VGG16 代码设计

除了大量减少卷积层，我们还观察原设计的分类层大量使用全连接，这造成了参数量暴增，训练十分困难以至于对实验展开造成不利。于是我们将原分类层的 2 个全连接替换为 1 个自适应池化层，只保留唯一的全连接。

得益于前两次模型改良的经验，我们认为对于该数据集，输出层设置 1024 个 logits（1024 个特征）就足够模型学习到 Discriminating Features。如果说将所有特征保持在

1024 个 logits 里显得比较稠密, 那么 2048 也是不错的选择, 虽然可能造成稀疏, 但给了模型自适应的空间, 灵活性增加。也可以结合 dropout 人为调整以达到平衡。

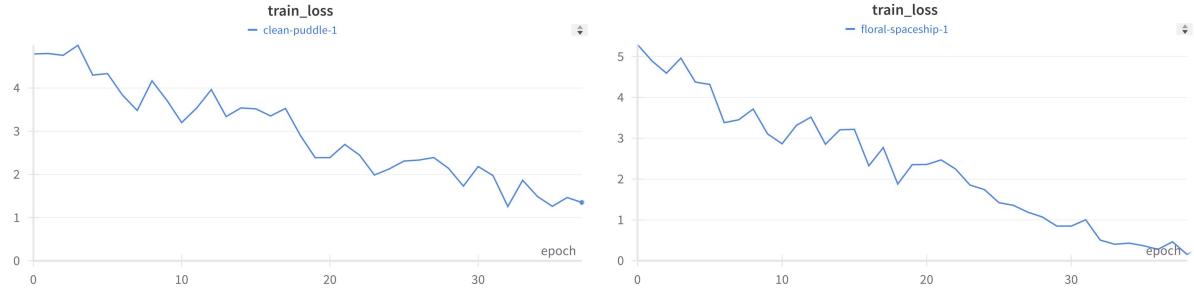


图 21 VGG_512_output_logits (左) 和 VGG_1024_output_logits (右) 损失曲线对比



图 22 改良 VGG(10)模型评价指标曲线

通过图 21 可以看到, 改良 VGG 输出层设置 512 个 logits 的情形下, 模型无法收敛, 而 1024 个 logits 性能更好。这验证了我们对于该数据集特征数量的经验, 512 个 logits 相对来说表达能力不足, 无法保持足够多有用特征。

图 22 显示, 经过 50 个 epoch 后, 改良 VGG 达成收敛, 评价指标显示该次训练也取得了接近完美的训练性能。然而, 不难发现, VGG 的评价指标曲线相比前述实验模型震荡更为明显。我们的解释认为, 与 GoogLeNet 的分支并联以及 ResNet 的残差跳连相比, VGG 是传统的串行结构, 那么在参数更新时, 下次更新会直接覆盖上次更新, 造成了扰动。索性模型具有适应能力, 不影响其持续收敛。

值得一提的是, 即便大幅降低了 VGG 模型的参数量, 其训练时间仍是前述模型的 3 倍。这提示了串行结构在并行结构下的效率劣势。

4 模型对比分析

本次实验中，模型对比主要分为性能对比和特征降维可视化对比。为了保证对比结论具有可靠性，我们做了较为充分的实验，如图 23。

Name	Last Run	Project Visibility	Runs	...
feat_1024_vgg16_enrich_sgd_0.95_0.4	2024-06-26	⌘ Team	1	...
feat_1024_vgg16_sgd_0.95_0.4	2024-06-26	⌘ Team	1	...
feat_512_vgg16_sgd_0.95_0.4	2024-06-26	⌘ Team	1	...
dropout0.4_vgg16_sgd_0.95	2024-06-25	⌘ Team	1	...
ave_pool_vgg16_sgd_0.95	2024-06-25	⌘ Team	1	...
redesign_vgg16_sgd_0.95	2024-06-25	⌘ Team	1	...
redesign_googlenet_sgd_0.95	2024-06-25	⌘ Team	1	...
adapt_googlenet_sgd_without_bn_0.95	2024-06-25	⌘ Team	2	...
adapt_googlenet_sgd_withbn_0.95	2024-06-25	⌘ Team	1	...
resnet18_sgd_newstart_0.95	2024-06-24	⌘ Team	1	...
resnet18_newstart_0.95	2024-06-24	⌘ Team	1	...
resnet50_dataall_transform_sgd_samedata_0...	2024-06-24	⌘ Team	1	...
resnet50_samedata_transform_sgd_samedat...	2024-06-24	⌘ Team	1	...

图 23 实验一览表（部分）

在性能对比中，我们首先对模型参数进行可视化，对比模型的体量。如图 24 所示。可以看出改良 GoogLeNet 即使大幅减少参数，还是具有相当大的体量。有意思的是，根据实验结果，VGG 虽然有最小的参数量，但是却有最久的训练时长。

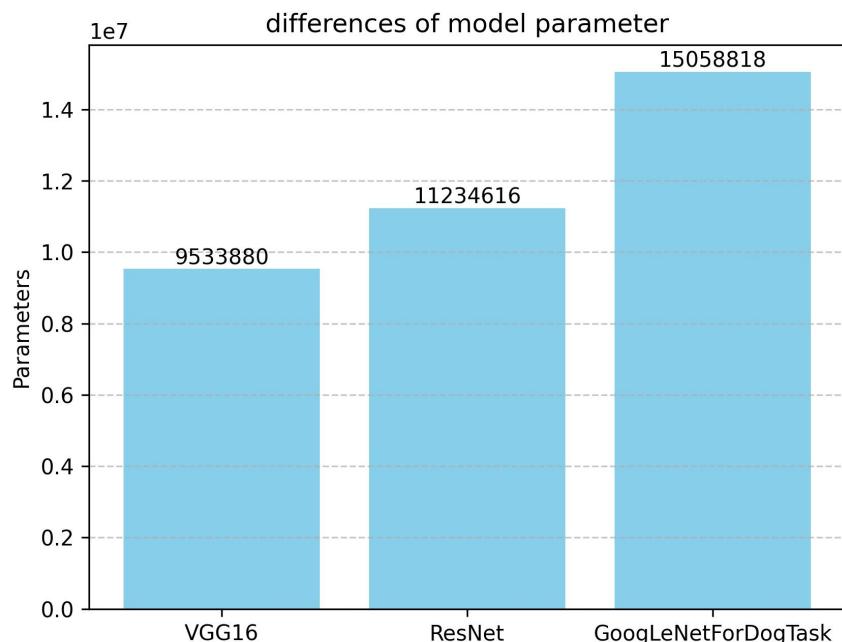


图 24 不同模型的参数差异

表 2 模型测试集性能指标对比

Model	Test Loss	Test Precision	Test Accuracy	Test F1 Score
VGG (Modified)	0.135	0.975	0.975	0.955
VGG (Meta)	2.019	0.445	0.445	0.302
GoogLeNet (Modified)	0.012	0.997	0.997	0.995
GoogLeNet (Meta)	3.432	0.279	0.279	0.175
ResNet (Modified)	0.313	0.983	0.983	0.969
ResNet (Meta)	3.489	0.239	0.239	0.147

表 2 是各模型版本在测试集的性能指标对比。可以清楚的看到三种模型在经过改良后取得了非常好的性能，而原初模型都无法适应本次实验数据集，故效果较差。需要注意的是，由于本实验统一设置了早停策略，即在验证集上连续三个 epoch 出现性能下降（损失不降反升）便停止训练。也许原初模型需要更多的尝试以跳出局部的限制，未来的工作可以进一步实验。然而就本工作而言，我们仍认为这是公平的。

在特征降维可视化中，我们利用了 openTSNE 框架下的 t-SNE 降维算法，对各个训练好的改良模型进行特征二维可视化。其中，t-SNE 降维算法的超参数按照官方文档的经典示例进行设置，具有自适应降维拟合的优势。

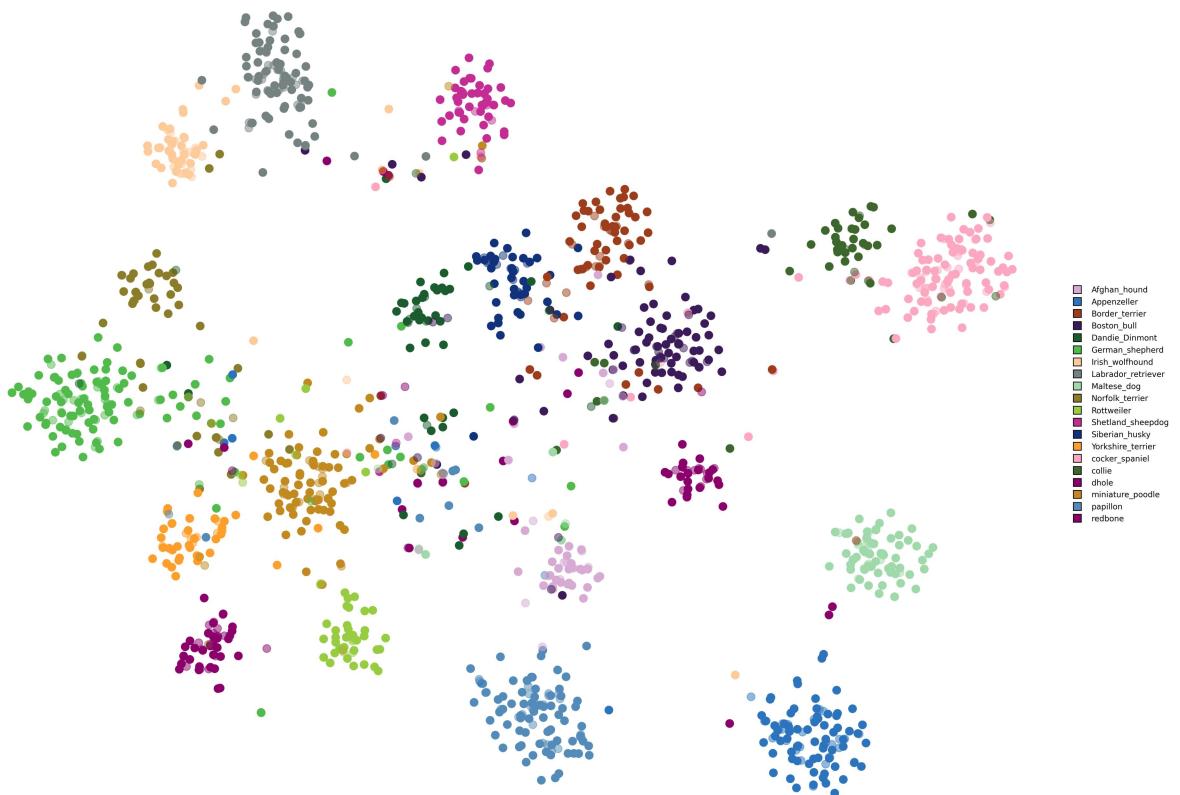


图 25 GoogLeNet 特征可视化（固定抽样 20 个类别）

如图 25, 不难看出, GoogLeNet 的聚类效果较好, 不同品种的狗在二维空间中有一定的聚类趋势, 多数类别形成了具有可分性的聚类, 并且聚类较为紧密。例如, “Papillon” 和“Appenzeller”似乎形成了自己的小群体。然而, 某些品种如“Siberian husky”和“Border terrier” 在图中较为分散, 这可能意味着模型在这些特征上区分度不高。另外, 图中没有明显的异常点, 这表明数据集可能较为干净, 没有显著的异常值。

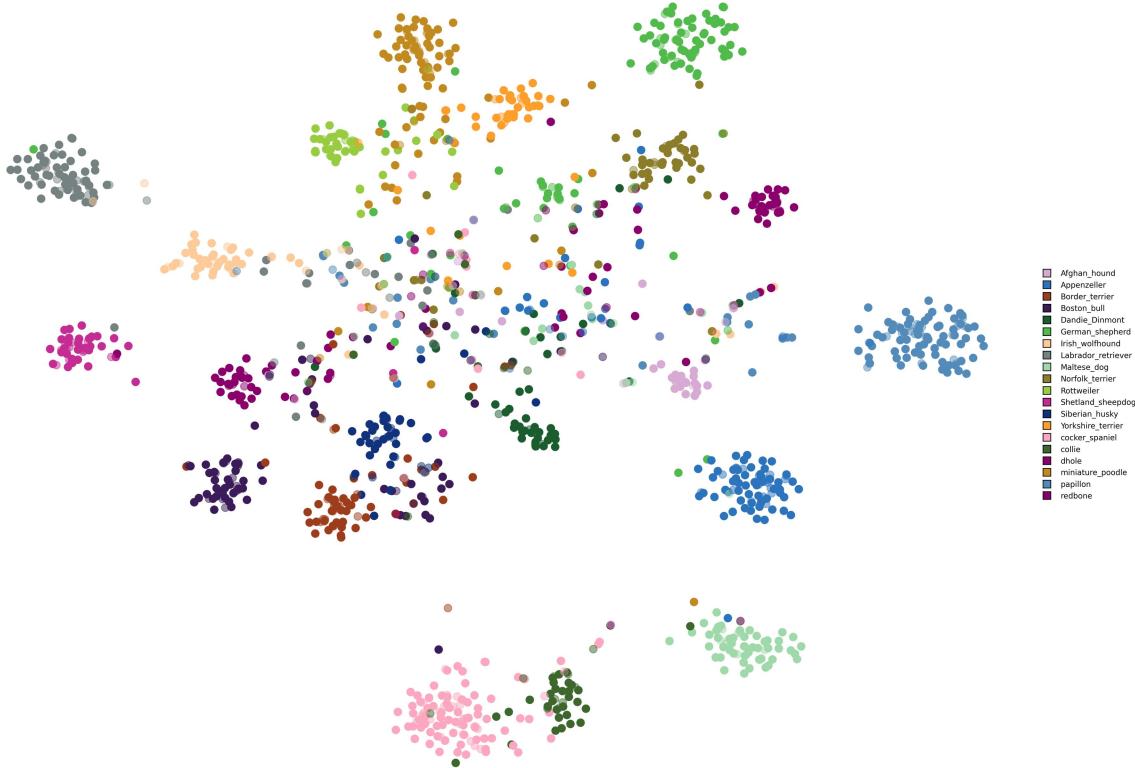


图 26 ResNet18 特征可视化 (固定抽样 20 个类别)

图 26 中可以看出, ResNet 的聚类效果在某些品种上似乎不如 GoogLeNet, 例如 “Boston bull” 和 “Dandie Dinmont” 的点较为分散, 尽管某些品种的聚类效果不佳, 但 “Irish wolfhound” 和 “Siberian husky” 在图中较为集中, 表明 ResNet 在这些特征上可能有一定的区分能力。另外, 总体而言不可分的样本数也较多, 集中在中心区域, 不分伯仲。与 GoogLeNet 相似, 图中没有明显的异常点。

图 27 显示的 VGG 特征可视化表明, 其聚类和类别可分效果是三个模型中最差的。只有较少的类别聚类成簇, 如 “Maltese Dog”。而不同类别之间的距离都比较靠近, 有的类别相互交错。结合评价指标, 可以推断 VGG 的准确率也许不错, 但鲁棒性较低, 面对一些干扰, 模型容易犯错。另外, 也可以从模型学习的角度理解: 模型主要通过大量神经元记住了各犬类, 而较少的关注犬类的特征模式。

总体而言, GoogLeNet 的聚类效果是最好的, 其次是 ResNet, 而 VGG16 在多数品种上的表现稍逊一筹, 这也和评价指标的结论形成相互印证。也不难总结到, 有些犬类的特征对于模型来说比较好学, 如 “Rottweiler” 等。



图 27 VGG 特征可视化（固定抽样 20 个类别）

5 模型分类效果



图 28 实例分类效果图（柯基犬）



图 29 实例分类效果图（边牧犬）

图 28、29 是网络图像的犬类实例，喂入模型后输出分类结果，并画在原图左上角。可以看到预测结果准确，进一步测试更多实例图像，结果表明与测试集性能基本一致。

6 实验结论

通过以上实验，我们总结了以下 7 点结论：

- (1) 对于犬类数据集，无需增强，简单的预处理如 Resize 和 Normalize（归一化）就已足够模型学习并取得较好的性能。
- (2) 由于自适应的策略，Adam 优化器在梯度下降的尺度上相对较小，虽然有益于快速收敛，但也容易发生图 15 的停滞现象。而 SGD 优化器刚好相反，拥有如图 14 所展现的跳出局部最优的能力。
- (3) 对于犬类数据集，适当降解模型有利于模型更好的学习，并提高学习效率。
- (4) 对于犬类数据集，输出层设置 1024 或 2048 个 logits 就足够模型充分学习。
- (5) 在 GoogLeNet 中增加的细节学习、完形学习以及精细分工的思路有利于模型学得更好，且收敛速度也加快。
- (6) VGG 串行架构的训练效率可能比不上对照实验模型并行架构的效率。
- (7) VGG 在特征降维可视化中聚类效果不佳，考虑到其性能指标好，故可能的结论是模型主要通过大量神经元记住了各犬类，而较少的关注犬类的特征模式。

总体而言，本研究展示了深度学习在细粒度图像分类中的潜力，并为未来的研究提供了有价值的参考。

7 设计感言

在本次深度学习课程设计过程中，我深刻体会到了深度学习模型的强大能力和调优的重要性。通过对经典网络模型的深入分析和改良，我不仅提升了自己的技术能力，也对深度学习在图像识别领域的应用有了更深刻的理解。

实验过程中遇到的挑战和困难，如梯度消失、过拟合、网络退化等问题，都促使我不断探索和尝试新的解决方案。最终，看到改良模型在实验中取得的优异性能，我感到非常自豪和满足。这次课程设计不仅锻炼了我的实践能力，也激发了我对未来人工智能研究的浓厚兴趣，特别是前沿的研究。我期待将所学知识应用于更广泛的领域，并为深度学习技术的发展做出自己的贡献。

参考文献

- [1] 张琦,张荣梅,陈彬.基于深度学习的图像识别技术研究综述[J]. 2019.
- [2] 郑远攀,李广阳,李晔.深度学习在图像识别中的应用研究综述[J].计算机工程与应用, 2019, 55(12):17.DOI:10.3778/j.issn.1002-8331.1903-0031.
- [3] Huang G , Liu Z , Laurens V D M ,et al.Densely Connected Convolutional Networks[J].IEEE Computer Society, 2016.DOI:10.1109/CVPR.2017.243.
- [4] Simonyan K , Zisserman A .Very Deep Convolutional Networks for Large-Scale Image Recognition[J].Computer Science, 2014.DOI:10.48550/arXiv.1409.1556.
- [5] He K , Zhang X , Ren S ,et al.Deep Residual Learning for Image Recognition[J].IEEE, 2016.DOI:10.1109/CVPR.2016.90.
- [6] Zeiler M D , Fergus R .Visualizing and Understanding Convolutional Networks[C]//European Conference on Computer Vision.Springer, Cham, 2014.DOI:10.1007/978-3-319-10590-1_53.
- [7] Szegedy C , Liu W , Jia Y ,et al.Going Deeper with Convolutions[J].IEEE Computer Society, 2014.DOI:10.1109/CVPR.2015.7298594.