猫狗大战

I 问题的定义

项目概述

最近，在大规模图像识别中，卷积神经网络发挥了重大作用。这主要归功于大的图像数据库，比如 ImageNet，以及计算机性能的提升，比如 GPU。尤其是，ImageNet Large-Scale Visual Recognition Challenge(ILSVRC)挑战赛对深度视觉识别框架的发展起到了很大的推动作用。从较浅的高维度神经网络到深度卷积神经网络，ILSVRC 挑战赛孵化出了好几代大规模图像分类系统。

自从 Alexnet 卷积神经网络问世后，其优秀的图像识别能力使得卷积神经网络获得了很大的关注。随后 VGG 网络、Googlenet 网络、Resnet 等各种卷积神经网络应运而生。随着卷积神经网络的不断发展，目前图像分类已经发展的相当成熟。图像分类的发展促进了机器视觉其他方面的发展，比如图像检测，图像分割等。图像分类模型性能的不断提高，是机器视觉不断发展的标志。Kaggle 竞赛对图像分类、图像分割等各种技术的发展起着推动性作用。本文来探索 Kaggle 竞赛中一个简单的图像分类任务。

问题陈述

本文对猫和狗的图像进行分类。使用的数据集是 kaggle 竞赛上的猫狗数据集。其下载链接为(https://www.kaggle.com/c/dogs-vs-cats-redux-kernels-edition/data)。数据集中包含各种各样的猫狗图像，训练集有 25000 张，其中猫狗分别占 12500 张，分别放在相应的类别文件夹中。测试集包含 12500 张图像。数据集具有多样性，拥有不同颜色，不同大小，不同角度，不同数量，不同场景，不同姿态等特征。图像的分辨率也是大小不一的，并且图像中或者有猫，或者有狗，但不会同时有猫和狗。要找到一种模型，在这些数据集上训练之后，使得模型具有分辨猫图像和狗图像的能力。

评价指标

因为该任务是要对猫和狗进行分类，所以这是一个二分类任务。在分类任务中，有两种评价指标。一种是准确率，另一种是精确度和召回率。准确率适合对称性分类任务，精确度和召回率适合不对称性分类任务。因为本文的猫狗数据集中猫和狗的图片比例相同，因此我选择分类准确率作为评价指标。

II 分析

数据探索

猫狗数据集分为训练集和测试集。训练集包含 25000 张图像，其中猫狗分别占 12500 张。测试集包含 12500 张图像。相比之下，ILSVRC 竞赛的数据集包含 120 万张图像，1000 个类别，平均每个类别包含 1200 张图像。对比猫狗数据集与 ILSVRC 竞赛的数据集，虽然总的

图像数差距很大，但如果只比较单个类别的图像数量，猫狗数据集比 ILSVRC 数据集的单个类别数高 10 倍。可以说，相比之下，猫狗数据集中单个类别的数据量很丰富。

因为测试集是不含标签的，因此为了验证训练效果，我从训练集随机挑选出 2500 张图像作为验证集，其中猫狗各占 1250 张。此时训练集包含 22500 张猫狗图像，验证集包含 2500 张猫狗图像。

探索性可视化

对猫狗数据集中的训练集和验证集的分析图像如图 1、图 2 所示。从这两幅图中可以看出，在训练集和验证集中猫和狗的图像分别占有的比例相同。因为验证集是从最初的训练集中分离出来的，因此训练集和验证集的图像服从同一分布。

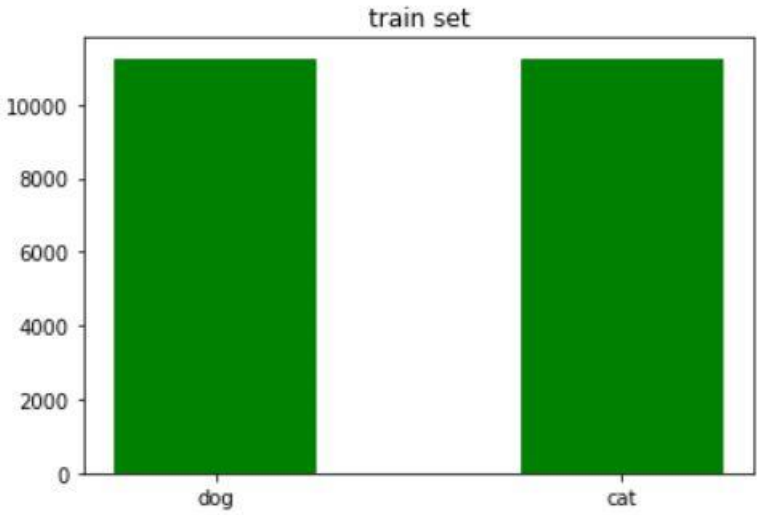

图 1 训练集可视化分析

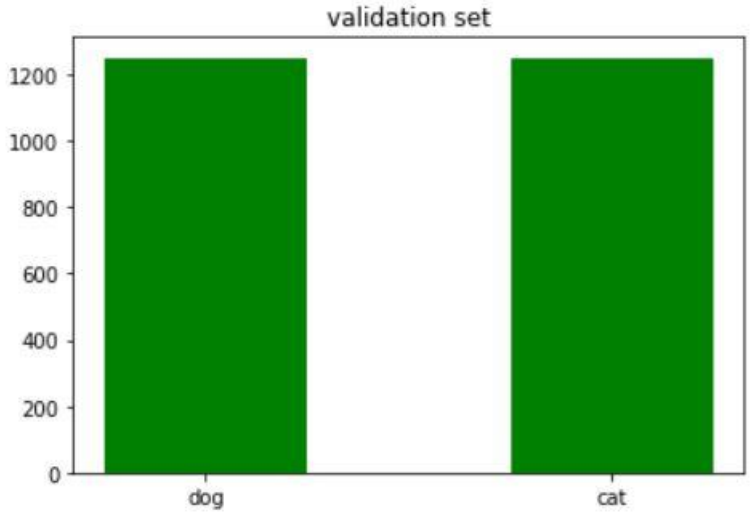

图 2 验证集可视化分析

算法和技术

Alexnet 模型

Alexnet 是 2012 年 Imagenet 竞赛的冠军模型，top-5 识别率达到 80.2%。AlexNet 包含 5 个卷积层和 3 个全连接层，模型示意图如图 3 所示：

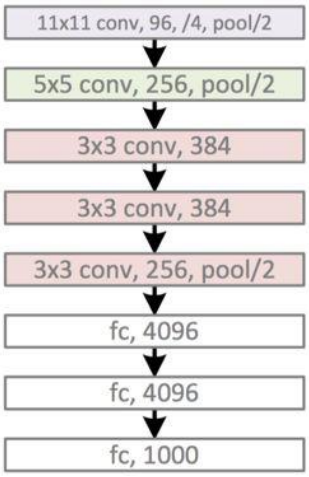

图 3 Alexnet 模型结构图

AlexNet 采用了 Relu 激活函数，取代了之前经常使用的 sigmoid 函数和 tanh 函数。AlexNet 另一个创新是 LRN(Local Response Normalization）局部响应归一化。LRN 模拟神经生物学上侧抑制（lateral inhibitio）的功能。侧抑制指的是被激活的神经元会抑制相邻的神经元。LRN 局部响应归一化借鉴侧抑制的思想实现局部抑制，使得响应比较大的值相对更大，提高了模型的泛化能力。LRN 只对数据相邻区域做归一化处理，不改变数据的大小和维度。

VGG 模型

VGG 网络是 2014 年 Imagenet 竞赛的亚军模型。VGG 是在传统的 Alexnet 卷积网络框架基础上研究增加深度对网络性能的影响。整个网络都使用了同样大小的 3*3 卷积核尺寸和 2*2 最大池化层，网络结构简洁。VGG 包含 VGG16 和 VGG19，VGG16 网络包含 16 层，VGG19 网络包含 19 层。其中 VGG19 网络的性能与 VGG16 网络的性能很接近，因此 VGG16 更被大家常用。本实验使用 VGG16 进行训练。VGG16 模型的结构图如图 4 所示。

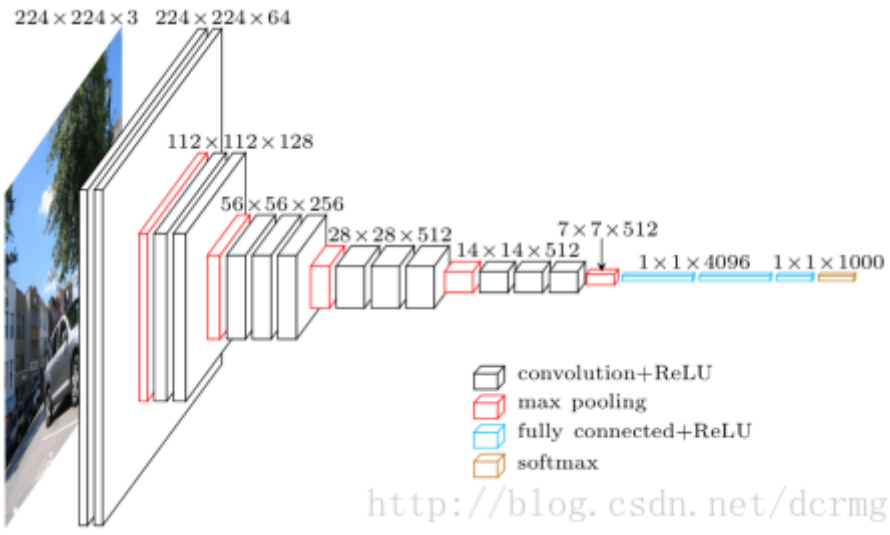

图 4 VGG16 模型的结构图

VGG 网络使用了 Multi-Scale 的方法做数据增强，将原始图像缩放到不同尺寸 S，然后再随机裁切 224*224 的图片，这样能增加很多数据量，对于防止模型过拟合有很不错的效

果。VGG 网络使用了更小的 3*3 卷积核和更深的网络。两个 3*3 卷积核的堆叠相当于一个 5*5 卷积核的感受野，三个 3*3 卷积核的堆叠相当于一个 7*7 卷积核的感受野。这样一方面可以减少网络参数量，另一方面可以拥有更多的非线性变换，增加了 CNN 对特征的学习能力。在 VGG 网络的卷积结构中，引入 1*1 的卷积核，在不影响输入输出维度的情况下，引入非线性变换，增加网络的表达能力，降低计算量。该 VGG 模型不仅在 ILSVRC 数据集中表现好，而且在其他数据集表现也很好。

ResNet 模型

网络的深度对模型的性能至关重要。实验发现，当增加网络层数后，深度网络出现了退化问题。即当网络深度增加时，网络准确度出现饱和，甚至出现下降。深层网络存在着梯度消失或者爆炸的问题，这使得深度学习模型很难训练。

何凯明博士提出了残差学习来解决退化问题。因为残差学习相比原始特征直接学习更容易。ResNet 网络参考 VGG19 网络，在其基础上通过短路机制加入了残差单元。ResNet 相比普通网络每两层间增加了短路机制，这就形成了残差学习。残差学习解决了由于网络深度增加引起的梯度爆炸和梯度消失问题，因此能够有效加深网络的深度，使得结果更好。残差模块结构如图所示。ResNet 网络有各种不同深度的网络，ResNet-18、ResNet-34、ResNet-50、ResNet-101、ResNet-152。网络越深，其性能越好。由于硬件性能的限制，本文使用 ResNet-50 来进行迁移学习。

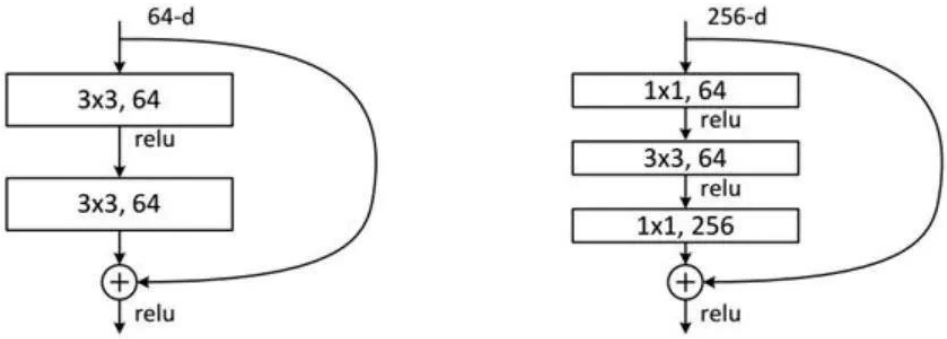

图 5 残差模块

Inception V3

GoogLeNet 首次出现在 2014 年 ILSVRC 比赛中获得冠军。该模型通常称为 Inception V1。Inception V1 是一个拥有 22 层的深度网络，参数量为 5 million。同一时期的 VGG 网络性能和 Inception V1 性能十分接近，但是 VGG 网络的参数量远大于 Inception V1。除了 Inception V1，在此基础上还有 Inception V2、Inception V3。

Inception 网络的核心是 Inception 模块。Inception 模块由 1*1 卷积，3*3 卷积，5*5 卷积，3*3 最大池化四个基本结构并联组成，并对四个成分运算结果进行通道上组合。通过多个卷积核提取图像不同尺度的信息，最后进行融合，可以得到图像更好的表征。Inception-V3 是借鉴 VGG 模型的方法，把 Inception 模块中的大卷积核转换成多个串联的 3*3 的卷积核。这种方法可以降低模型的参数量，增强模型的非线性。

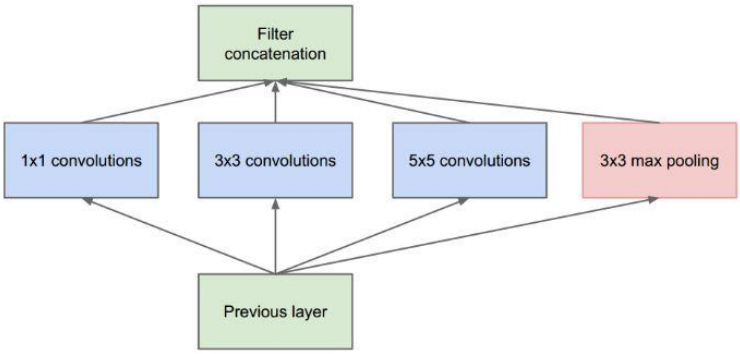

图 6 Inception 模块

Xception 模型

Xception 是 google 提出的对 Inception v3 的另一种改进，主要是采用 depthwise separable convolution 来替换原来 Inception v3 中的卷积操作，在基本不增加网络复杂度的前提下提高了模型的效果。文章中将 depthwise separable convolution 分成两步，一步是 depthwise convolution，另一步是 pointwise convolution。Depthwise separable convolution。即先用 M 个 3*3 卷积核一对一卷积输入的 M 个 feature map，生成 M 个结果，然后用 N 个 1*1 的卷积核正常卷积前面生成的 M 个输出，最后生成 N 个 features。

EfficientNet 模型

对于深度卷积神经网络，扩大网络宽度、深度或分辨率的任何尺寸都可以提高精度。那么该如何来平衡网络宽度，深度和分辨率对网络性能的影响呢？

EfficientNet 使用 EfficientNet-B0 为基准架构，使用一种复合模型缩放的方法，即使用一个符合系数来均匀地缩放网络宽度、深度以及分辨率。图 7 为 EfficientNet-B0 的体系结构。在 EfficientNet-B0 的基础上，使用不同的符合系数来扩大网络规模得到 EfficientNet-B1 到 EfficientNet-B7。把 EfficientNet 一系列模型与现有的优秀的卷积神经网络的性能对比图如图 8 所示。

Stage i	Operator $\hat{\mathcal{F}}_i$	Resolution $\hat{H}_i \times \hat{W}_i$	#Channels $\hat{C}_i$	#Layers $\hat{L}_i$
1	Conv3x3	224×224	32	1
2	MBConv1, k3x3	112×112	16	1
3	MBConv6, k3x3	112×112	24	2
4	MBConv6, k5x5	56×56	40	2
5	MBConv6, k3x3	28×28	80	3
6	MBConv6, k5x5	14×14	112	3
7	MBConv6, k5x5	14×14	192	4
8	MBConv6, k3x3	7×7	320	1
9	Conv1x1 & Pooling & FC	7×7	1280	1

图 7 EfficientNet-B0 结构

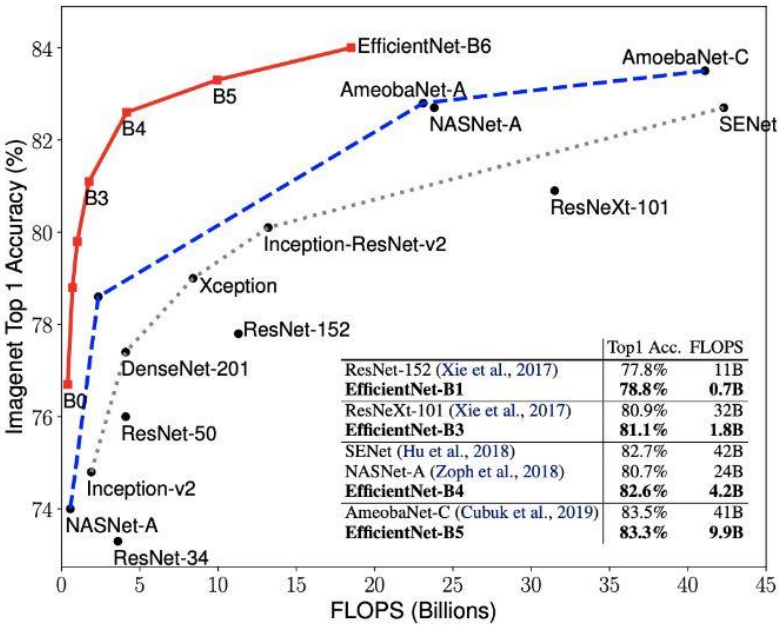

图 8

基准模型

我选择传统的卷积网络框架 Alexnet 网络作为基准模型。该模型包含 5 个卷积层，3 个全连接层，60 million 参数量。该模型是卷积神经网络步入图像分类的第一步。

在 torchvision 库的 models 模块中，包含有已训练好的 alexnet 模型，VGG 模型。我在最初的计划是，用猫狗数据集依次分别来训练以下几个模型：未训练的 alexnet 模型，已训练好的 alexnet 模型，已训练的 VGG 模型、已训练的 ResNet-50 模型、已训练的 EfficientNet-B3 模型。训练结束后，对各个模型的表现进行对比。

另外，对于猫狗数据集，人的表现可以到达 0%的错误率。因此 0%的错误率就是训练模型的目标。

III 方法

数据预处理

猫狗数据集包含各种分辨率图像，可网络需要输入的图像是固定的维度。因为 Alexnet、VGG 网络要求输入图像尺寸为 224 X 224 X 3，所以我们对图像进行下采样并剪切出一个固定分辨率为 224 X 224 的三通道 RGB 图像。在 torchvision 包的 transforms 模块中包含多中处理图像的方法，比如裁剪、翻转、缩放、标准化等。要将不同分辨率的图像转换成 224 X 224 的输入图像，首先要对图像进行缩放操作 transforms.Resize(256)，然后对缩放后的图像从中心裁剪出 224 X 224 的图像块 transforms.CenterCrop(224)。最后对图像标准化 transforms.Normalize()。即把训练集图像的每个像素减去所有训练图像的平均值，再除以标准差。其中的参数 mean 和 std 是从训练集 25000 张图像中随机选取 2500 张计算得到的近似值。除此之外，没有对训练集做其他预处理。

除了对图像缩放成想要的尺寸之外，我还做了图像增强，包括随机旋转、随机水平翻转、随机竖直翻转、随机裁剪。图像增强对降低网络的过拟合起到很好的效果。图像增强在下文

的实验训练过程中会导致验证集损失值小于训练集的损失值。

执行过程

第一，选择模型。

我预先计划在未训练的 alexnet 模型，已训练好的 alexnet 模型，已训练的 VGG 模型、已训练的 ResNet-50 模型、已训练的 EfficientNet-B3 模型上分别进行训练，并对比训练结果。Alexnet 模型是一个 8 层的卷积神经网络，VGG 是一个 16 层的卷积神经网络。可以提前预料的是，VGG 网络比 Alexnet 网络表现好，因为 VGG 网络的深度是 Alexnet 网络的两倍。但是因为 VGG 网络深度是 Alexnet 的两倍大，且 VGG 网络的参数量也是 Alexnet 的两倍大，因此训练速度会慢很多。我之所以选择尝试在 Alexnet 网络、VGG 网络上训练，是因为我想知道不同深度的卷积神经网络在该猫狗分类任务中表现如何。

第二，选择损失函数和优化器。

损失函数主要有两大类，回归损失和分类损失。回归损失主要包括均方误差损失函数，平均绝对误差损失函数等。分类损失主要包括均方误差损失函数，交叉熵损失函数，负对数似然损失函数等。本文是一个二分类问题，我选择分类损失中的交叉熵损失函数作为模型的损失函数。

在机器学习和深度学习中，除了常见的 SGD 优化器，还有 Adagrad，RMSProp，Adam 等各种优化器。Adagrad，RMSProp，Adam 等这些优化器都是在 SGD 的基础上一步步进行改进的结果。他们解决了 SGD 固定学习率的问题，使得不同的参数对应不同的学习率，并且在训练过程中学习率是一直变化的。

对于猫狗图像分类任务，我选择的优化器是 Adam 和 SGD。因为 Adam 训练速度比 SGD 快，因此最开始的训练的时候我选择 Adam 作为优化器，等到训练集损失值趋于平缓，再使用 SGD 作为优化器继续训练。SGD 虽然比 Adam 训练速度慢，但 SGD 训练起来更稳定，训练集损失值波动性小。

第三，超参数的选择

SGD 的学习率最初选择了 1e-4，之后以 10 的倍数递减。

因为笔记本显存有限的缘故，把 Batch_size 设为 64。

训练模型

1，在未训练的 Alexnet 模型上训练时，网络参数都是随机初始化的。用该网络进行训练，5 个 epoch 后，网络在验证集上的准确率达到 80.16%。再继续训练准确率还会有所提升。训练集和验证集的准确率和损失函数曲线如图 9、图 10 所示。

	训练集准确率	验证集准确率
1 epoch	67.07%	69.44%
2 epoch	71.91%	73.80%
3 epoch	75.15%	75.64%
4 epoch	76.56%	77.96%
5 epoch	75.15%	80.16%

图 9 随机初始化的 Alexnet 网络训练结果

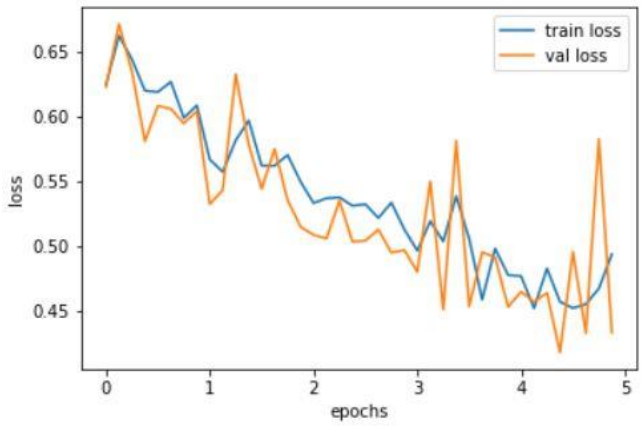

图 10 随机初始化的 Alexnet 网络损失曲线

2，用 pytorch 库 models 模块中预训练的 Alexnet 网络进行迁移学习。训练结果如图 11、图 12 所示。最后验证集的准确率可达到 95%，再继续训练准确率还会有所提升。由于训练集做了数据增强，使得训练集的损失值比验证集的损失值高。

	训练集准确率	验证集准确率
1 epoch	91.36%	94.28%
2 epoch	91.21%	94.88%
3 epoch	92.34%	94.96%
4 epoch	93.28%	95.60%
5 epoch	93.55%	95.04%

图 11 Alexnet 迁移学习训练结果

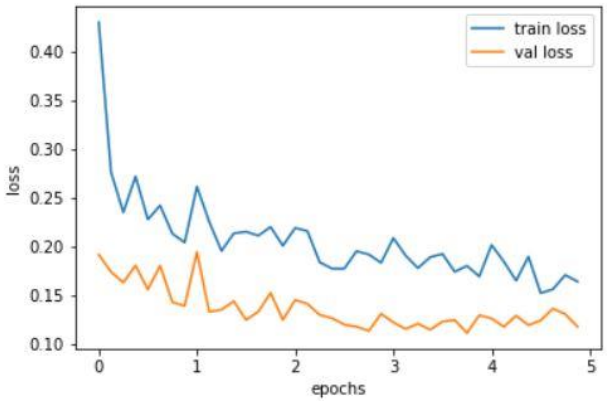

图 12 Alexnet 网络迁移学习损失曲线

3，接下来用 VGG16 网络进行迁移学习。VGG16 网络含有 16 层，138 Million 参数量。由于 VGG 网络较大，在自己的笔记本上无法展开训练，因此选择在云端进行训练。我选择了阿里云。训练之后的结果如图 13、图 14 所示。训练 5 个 epoch 后验证集的准确率可达到 98%，从损失函数曲线来看此时已基本收敛，再继续训练准确率还会有些许提升。

	训练集准确率	验证集准确率
1 epoch	94.49%	98.32%
2 epoch	93.12%	98.20%
3 epoch	95.00%	98.28%
4 epoch	95.00%	98.08%
5 epoch	96.36%	98.28%

图 13 VGG16 迁移学习训练结果

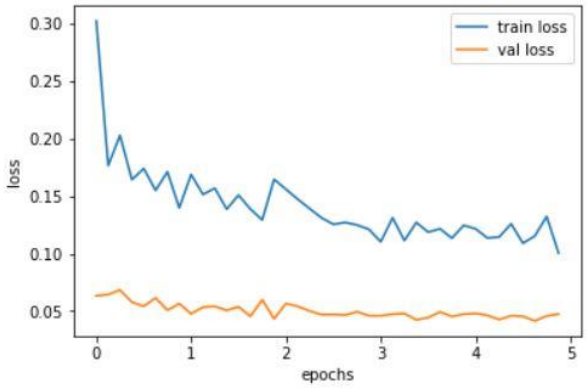

图 14 VGG16 损失曲线

4，用 ResNet 迁移学习。图 15 和图 16 是 ResNet18 网络迁移学习的结果。可以看出训练 5 个 epoch 后验证集的准确率可达到 97%，从损失函数曲线来看此时已基本收敛。由于对训练集应用了数据增强，使得训练集损失值比验证集损失值高。图 17 和图 18 是 ResNet50 网络迁移学习的结果。训练 5 个 epoch 后验证集的准确率可达到 97.84%，从损失函数曲线来看此时已基本收敛。

	训练集准确率	验证集准确率
1 epoch	90.39%	95.96%
2 epoch	90.85%	96.64%
3 epoch	92.18%	96.92%
4 epoch	92.81%	96.96%
5 epoch	91.71%	97.08%

图 15 Resnet-18 迁移学习训练结果

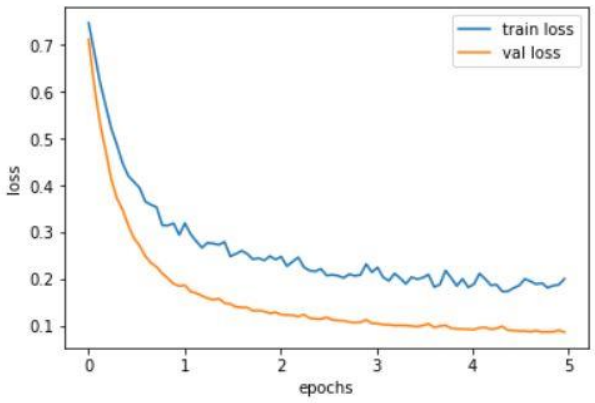

图 16 Resnet-18 迁移学习损失曲线

	训练集准确率	验证集准确率
1 epoch	93.32%	97.08%
2 epoch	94.06%	97.40%
3 epoch	93.98%	97.44%
4 epoch	94.18%	97.88%
5 epoch	94.37%	97.84%

图 17 Resnet-50 迁移学习训练结果

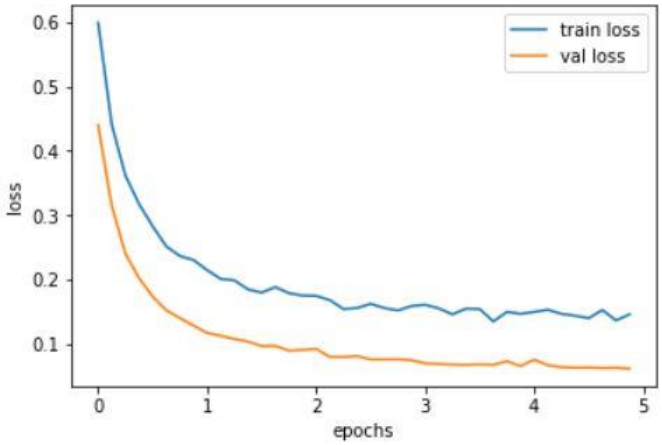

图 18 Resnet-50 迁移学习损失曲线

5，用 EfficientNet-B3 迁移学习。图 19 和图 20 是 EfficientNet-B3 网络迁移学习的结果。训练 5 个 epoch 后验证集的准确率可达到 96.52%，从损失函数曲线来看此时已开始收敛，再继续训练准确率还会有所提升。

	训练集准确率	验证集准确率
1 epoch	91.13%	95.32%
2 epoch	93.35%	95.88%
3 epoch	93.28%	95.96%
4 epoch	93.47%	96.24%
5 epoch	94.33%	96.52%

图 19 EfficientNet-B3 迁移学习训练结果

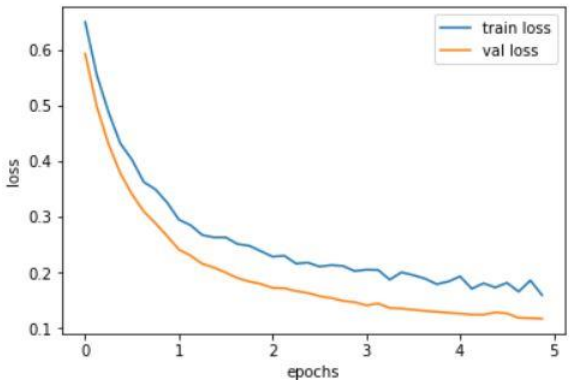

图 20 EfficientNet-B3 迁移学习损失曲线

完善

Alexnet、VGG16、resnet18、resnet50、EfficientNet-B3 在训练 5 个 epoch 之后，从损失函数曲线来看网络都已进入收敛区域，他们的准确率分别达到了 95.04%、98.28%、97.08%、97.84%、96.52%，相比人的水平仍然有 2%的差距。这 2%的错误率足以使得 kaggle 得分排在一千名以后。距离 0%的错误率还有很长一段路。

为了提高准确率，单网络模型似乎无法满足我们的需求。最后，我参考了知乎上优达学城账户发布的一篇文章。把已经训练好的不同模型的特征向量结合在一起，后面加上自己创建的全连接层。训练模型时只对自己创建的全连接层进行训练。文中选择了 resnet50，Xception，Inception-V3 这三个模型进行结合。因为这三个模型都很大，为了节省计算量和训练时间，预先把训练集图像转换成各个网络的特征向量，并保存下来。也就是说，一幅图像映射出三个不同的特征向量，之后将这三个不同的特征向量结合成一个特征向量。结合后的特征向量作为自己创建的全连接网络的训练集进行训练。这种方法降低了训练网络时对电脑配置的要求，而且大大提高了训练速度。其训练结果如图 21 所示。验证集的错误率在 0.4%左右。将结果上传至 kaggle 竞赛，loss 得分为 0.041，排名在前 2%。图 22 为 kaggle 竞赛得分截图。可见这种多模型融合的方法表现很优秀。

	训练集准确率	验证集准确率
1 epoch	97.69%	99.40%
2 epoch	99.17%	99.60%
3 epoch	99.37%	99.64%
4 epoch	99.45%	99.60%
5 epoch	99.49%	99.60%

图 21 多模型融合训练结果

图 22 多模型融合 kaggle 得分

IV 结果

模型的评价与验证

Alexnet、VGG16、resnet18、resnet50、EfficientNet-B3 在训练 5 个 epoch 之后，从损失函数曲线来看网络都已进入收敛区域，他们的准确率分别达到了 95.04%、98.28%、97.08%、97.84%、96.52%，相比人的水平仍然有 2%的差距。

对于猫狗数据集，人的表现可以到达 0%的错误率。因此训练模型的目标就是 0%的错误率。对比 Alexnet、VGG16、resnet18、resnet50、EfficientNet-B3，在训练 5 个 epoch 之后，他们的准确率分别达到了 95.04%、98.28%、97.08%、97.84%、96.52%。我发现，在这五种网络模型中，VGG16 网络的迁移学习准确率更高。由此可以得出，VGG16 网络学习出的特征相比其他网络更好，泛化能力更强。

VGG 网络、Alexnet 网络、Resnet 网络以及 EfficientNet 网络都获得了较低的错误率，但距离 0%的错误率还有一段距离。最后把 Resnet50，Xception，Inception-V3 这三个模型结合在一起后，发现可以获得 0.6%的错误率，可见这种多模型融合的方法表现很优秀。

V 项目结论

深度模型在本文的猫狗图像分类任务中表现很好，准确度都超过了 98%。只是要想达到 100%的准确度，用多模型融合的方法可以更快的实现我们的目的。只是多个模型结合在一起，使得计算量成倍增加，不适合在配置较低的硬件上实时预测。

对项目的思考

对猫狗图像分类这样一个项目，听起来很简单的事情，发现训练起来却不简单。Alexnet、VGG16、resnet18、resnet50、EfficientNet-B3 在训练 5 个 epoch 之后，他们的准确率分别达到了 95.04%、98.28%、97.08%、97.84%、96.52%。我猜想，之所以该实验中 VGG16 的迁移效果是最好的，是因为 VGG16 的参数量是其他模型的十几倍。像 resnet、EfficientNet 这些轻量级模型迁移效果差一些。但是因为我只训练 5 个 epoch，并不能最终决定那个模型是表现最好的。不过，在计算资源有限的情况下，轻量级模型是我们的首选。

需要做出的改进

由于硬件资源有限，网络的训练时间不长。如果增大训练时间，还可以进一步提高各个模型的准确率。多模型融合这种方法虽然可以达到较高的准确率，却需要更多的计算资源。我认为接下来的任务是探索不仅准确率高而且计算效率也高的模型。

十分感谢优达学城，在这个项目中我学到了很多东西。