基于Xception模型的花卉分类

刘喆骐
liuzheqi20@mails.tsinghua.edu.cn

徐硕
xus19@mails.tsinghua.edu.cn

摘要

在kaggle中有一项竞赛，目标是将104种花卉进行分类，我们参加了这个竞赛，尝试使用8种不同的深度学习模型（包括Inception、ResNet50、Xception等）来完成这个任务，经过尝试与调参，我们最终发现Xception模型的表现效果最好。基于此，我们尝试使用TensorsFlow的有关库手动搭建Xception模型，代替原有的预训练模型完成该任务，我们发现如果不使用预训练模型的权重初始化模型，得到的效果会不理想，而在权重初始化后，我们搭建的模型也取得了较好的成绩。

介绍

在kaggle种有一项竞赛需要我们建立一个机器学习模型，来对从五个不同公共数据集中提取的104种花卉图像进行分类。该竞赛提供的数据集共包含JPEG编码的4种不同格式的花卉图片，大小分别为192*192、224*224、331*331、512*512，每种格式的图片都分为train、val、test三个部分，其中train集和val集中的图片带有标签。数据集中共有12753张训练图片、3712张验证图片、7382张测试图片。此外，数据集提供的图片为tfrec格式，这是TensorFlow自带的一种数据格式，在读取图片时可以更加方便。

为了解决该分类问题，我们先尝试了8种不同的预训练模型在该数据集上的表现，这8种预训练模型包括：MobileNetV2、EfficientNetB0、Xception、NASNetMobile、DenseNet、Inceptionv3、ResNet50以及VGG16。我们发现Xception模型在该数据集上的表现效果最好。

此后，我们尝试用TensorsFlow的有关库手动搭建出了Xception模型，并对经过用预训练模型的权重初始化模型以及未经过用预训练模型的权重初始化模型的训练结果进行了比较。

背景

图像分类是计算机视觉领域的一个重要问题。它的目的是将给定的图像数据集分成若干个类别，每个类别都有许多图像。这个问题的重要性在于，它可以应用于许多领域，如自动驾驶、医学图像分析、机器人感知等。

在过去几十年中，许多方法都被提出来解决图像分类问题。例如，在20世纪70年代提出的最大类间方差法（Fisher's Linear Discriminant）是一种基于线性分类器的方法，可以有效地将图像分类。在20世纪90年代，随着深度学习技术的发展，卷积神经网络（Convolutional Neural Network，CNN）也被广泛应用于图像分类任务中。CNN通过在图像中提取特征并使用多个卷积层来解决图像分类问题。在近几年，一些新的机器学习方法也被提出，如基于自动编码器的方法和生成对抗网络（Generative Adversarial Network，GAN）。

在图像分类领域，深度学习模型是目前最为流行的方法之一。深度学习模型通过提取图像中的特征并使用多个卷积层来解决图像分类问题，并且在近年来得到了极大的发展。目前，许多不同的深度学习模型都被提出，每种模型都有自己的优缺点。

Xception模型是一种卷积神经网络（Convolutional Neural Network，CNN），它使用了深度可分离卷积的方法来构建网络。在传统的卷积层中，每个卷积核都会与输入数据的所有通道进行卷积，并得到一个输出通道。这样的卷积层通常需要较多的参数，并且在计算时需要较大的计算量。

深度可分离卷积的方法则是将传统卷积层中的卷积核拆分成两部分：深度卷积核和点卷积核。深度卷积核只与输入数据的单个通道进行卷积，而点卷积核则是将深度卷积核的输出在通道维度上进行卷积。这样的卷积层通常参数较少，并且计算量也更小。

Xception模型使用了这种深度可分离卷积的方法来构建网络，并通过对网络进行精心设计，使得模型在保持同等准确率的情况下，能够大大减少模型的参数数量和计算量。这使得Xception模型更加轻巧，更加容易被部署在移动设备上。此外，Xception模型还使用了跨域的残差连接（Cross Domain Residual Connection）的方法来构建网络，这能够有效地防止模型的欠拟合，提高模型的泛化能力。

在实际应用中，Xception模型在ImageNet数据集上取得了最优秀的成绩，并且也在其他许多视觉任务中表现出了良好的效果。由于其轻巧的特点，Xception模型也被广泛用于移动设备上的视觉应用中。

在本研究中，我们尝试了 8 种不同的深度学习模型，发现Xception模型的表现效果最好。因此，我们选择使用Xception模型来完成这个图像分类任务。

为了进一步提高模型的表现，我们尝试使用TensorFlow的库来手动搭建Xception模型，代替原有的预训练模型完成这个任务。我们发现，如果不使用预训练模型的权重初始化模型，得到的效果会很差，而在权重初始化后，我们搭建的模型也取得了较好的成绩。

方法

在分类任务中，我们主要使用了两种技术，分别是数据增强和以及图像分类的预训练模型。

数据增强是指在训练机器学习模型时，使用各种方法增加训练数据的数量的过程。通过增加训练数据的数量，可以使模型更加泛化，从而在测试数据上表现得更好。在此次训练中，我们随机对图像进行了截取、旋转、裁剪操作。

在图像分类任务中，预训练模型是指使用大型数据集预先训练出来的模型。这些预训练模型可以被用于解决各种不同的图像分类任务，而且可以在训练数据较少的情况下取得较好的性能。这些模型已经在大型数据集（例如 ImageNet）上预先训练过，并且在多种图像分类任务中取得了较好的性能。常用的图像分类预训练模型有VGG、ResNet、Inception、DenseNet、Xception等。使用预训练模型时，通常会在预训练模型的基础上，在新的数据集上进行微调（fine-tuning），以获得更好的性能。此外，我们也通过手动建立Xception模型，比较了经过权重初始化和未经过权重初始化但结构相同的两个模型在该数据集上表现的差距。

实验

在此次实验中，我们使用的数据集共包含JPEG编码的 4 种不同格式的花卉图片，大小分别为 192*192、224*224、331*331、512*512，每种格式的图片都分为train、val、test三个部分，其中train集和val集中的图片带有标签。数据集中共有 12753 张训练图片、3712 张验证图片、7382 张测试图片。我们使用 512*512 的图片进行验训练。

在我们的实验中，采取F1 值作为评价标准。首先，我们通过固定预训练模型和其他超参数，对数据增强手段进行优化；在得到最佳的数据增强方法后，我们仅改变预训练模型，观察不同预训练模型在该数据集上的效果，寻找在该数据集上最佳的预训练模型；最后，我们针对该预训练模型进行调参，来得到最好的分类结果。

在得到Xception作为最佳预训练模型后，我们还采用TensorFlow库手动搭建出了一个Xception模型，来比较经过预训练确定初始权重和随机权重的两模型之间的区别。

1.1. 数据增强部分

在此次训练任务中，我们随机对图像进行了截取、旋转、裁剪操作。

在进行数据增强前后花卉图片示例如图 1 所示。

Figure 1: 左图未经数据增强，右图进行了数据增强

在其他条件一致时，数据增强前的F1 值一般要比数据增强后的高。在使用Xception训练 35 轮时，使用数据增强的F1 值为 0.942，未使用数据增强的F1 值为 0.931，说明数据增强可以高模型的准确率。在此后的实验中，我们都采取了数据增强的策略。

1.2. 预训练模型的确定

我们共采取了 8 种预训练模型（加载了ImageNet上预训练权重）对该数据集进行训练，其他设置完全相同（35 轮，相同的学习率策略）。得到的结果如下表所示。

经过实验，我们发现Xception模型在该数据集上的效果最好，因此我们将就Xception模型做更多的优化工作。

此外我们还发现VGG16 与NASNetMobile模型在该数据集上几乎失效。这是由于VGG16 模型庞大，需要更多的训练轮数，而NASNetMobile则是由于无法加载预训练的权重导致结果不理想。

模型	MobileNetV2	EfficientNetB0	Xception	NASNetMobile
准确率（F1 score）	0.91	0.652	0.942	0.006
模型	DenseNet	Inceptionv3	ResNet50	VGG16
准确率（测试集）	0.935	0.938	0.924	0.211

1.3. 参数调节

我们尝试使用了调整训练轮数和调整学习率曲线的方法。

训练轮数为 35 轮时，使用图 3 所示的学习率曲线，得到F1 值为 0.940。

将数据训练轮次调为 50 轮，使用 tensorflow 的 ReduceLROnPlateau 函数，若验证集 loss 在两个连续 epoch后不下降，则将学习率减半。得到测试集F1 值为 0.93。

将数据训练轮次调为 50 轮，并使用不同的学习率曲线。得到的结果如图 2-7 所示。

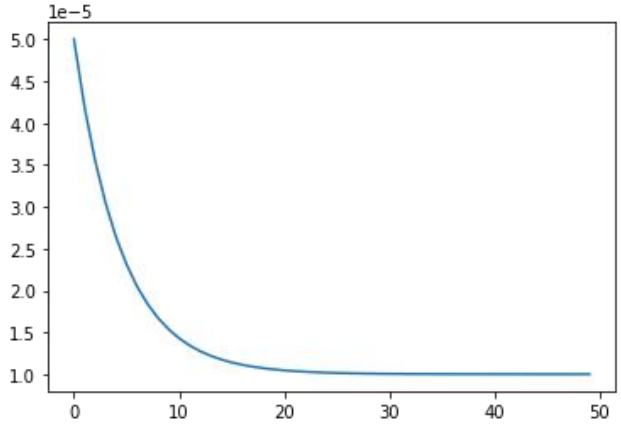

Figure 2: 使用指数下降曲线（每次学习率乘 0.9），F1 值 0.881

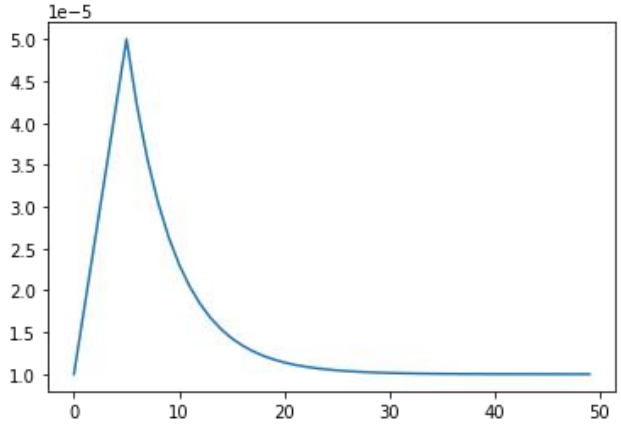

Figure 3: 先线性上升，后指数下降曲线，F1 值 0.942

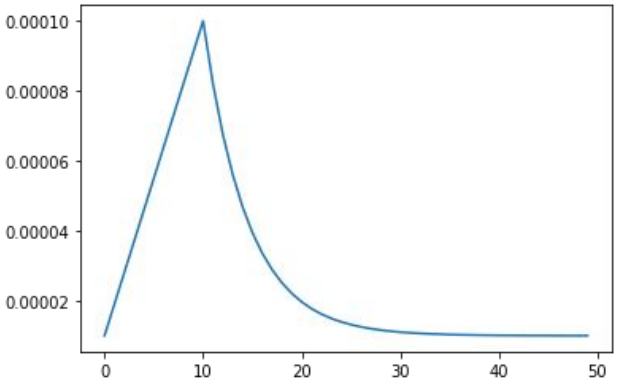

Figure 4: 延长上升期时间（斜率不变），后指数下降曲线，F1 值 0.947

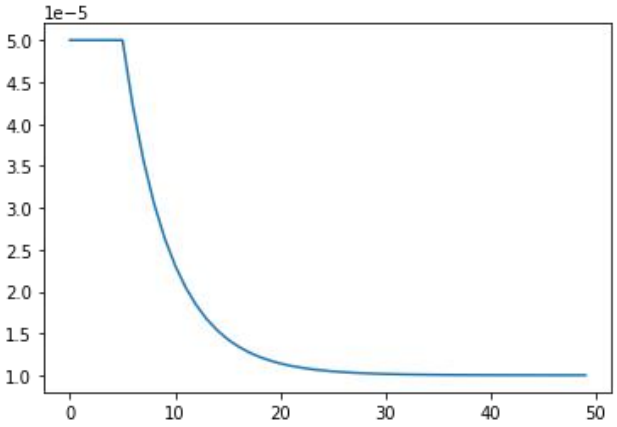

Figure 5: 使用学习率先不变，后指数下降曲线，F1 值 0.920

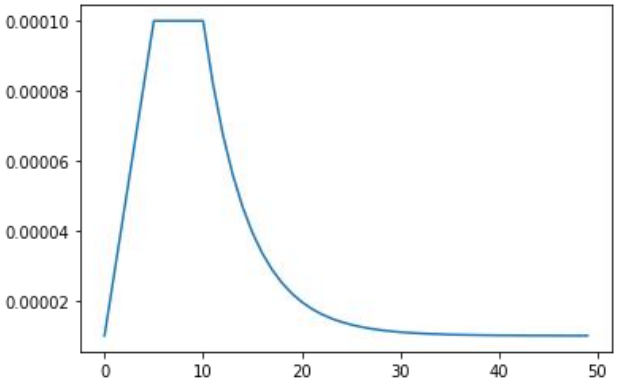

Figure 6: 使用学习率先线性上升，维持稳定，后指数下降曲线，F1 值 0.952

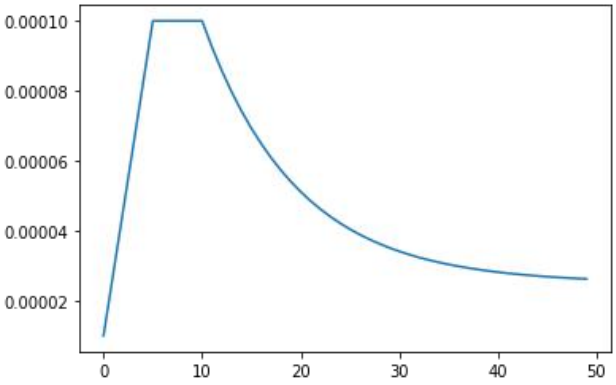

Figure 7: 使用学习率先线性上升，维持稳定，后指数下降曲线（学习率每次乘 0.8），F1 值 0.932

可以看到使用平台期加上上升期最后加入指数下降的效果最好。

1.4. 自行构建Xception模型

根据xception模型的原理和架构，我们尝试自行构建了一个xception模型，并开展了训练。由于是自行定义的模型，所以无法使用tensorflow预先在ImageNet训练的权重。模型一共可以分为 3 个flow, 分别是Entry flow、Middle flow、Exit flow；分为 14 个block，其中Entry flow中有 4 个、Middle flow中有 8 个、Exit flow中有 2 个。具体结构如图 8。

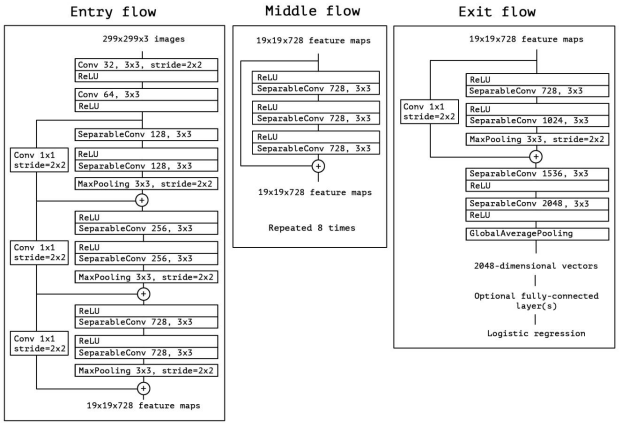

Figure 8: Xception模型的具体结构

这其中的内部主要结构就是残差卷积网络搭配可分离卷积块（SeparableConv）层实现各个block。在Xception模型中，常见的两个block的结构如图 9。

Block1:主要在Entry flow和Exit flow中：

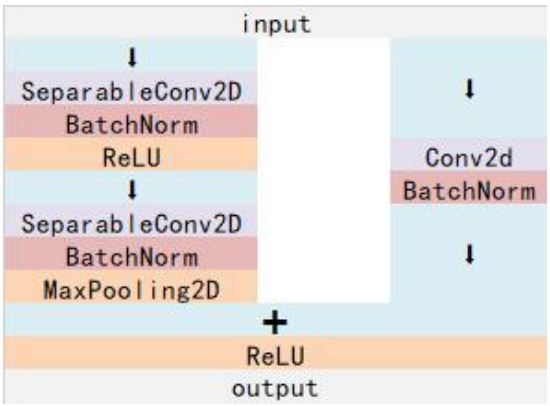

Figure 9: block1 的具体结构

Block2: 主要在Middle flow中。

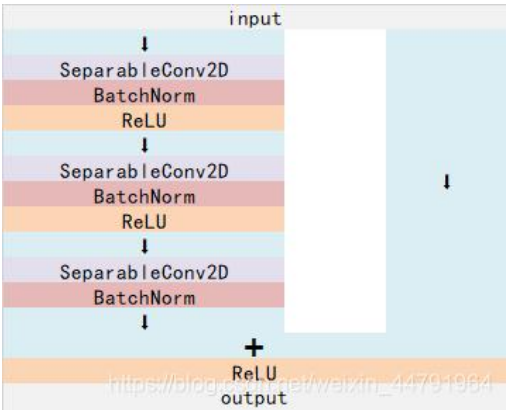

Figure 10: block2 的具体结构

通过运行，我们得到如下的结果。

当训练轮数为 35 轮时:

测试集f1 值	测试集准确率	测试集 Recall	验证集loss
0.846	0.858	0.843	0.5587

Loss和准确率图如下，可以发现在上述关键指标上不如加载预训练权重的xception模型。这说明在大模型（如ImageNet）训练后得到的权重在经过一定的调整以后能够比随机权重得到更加优越的结果。这也正是迁移学习的思想。

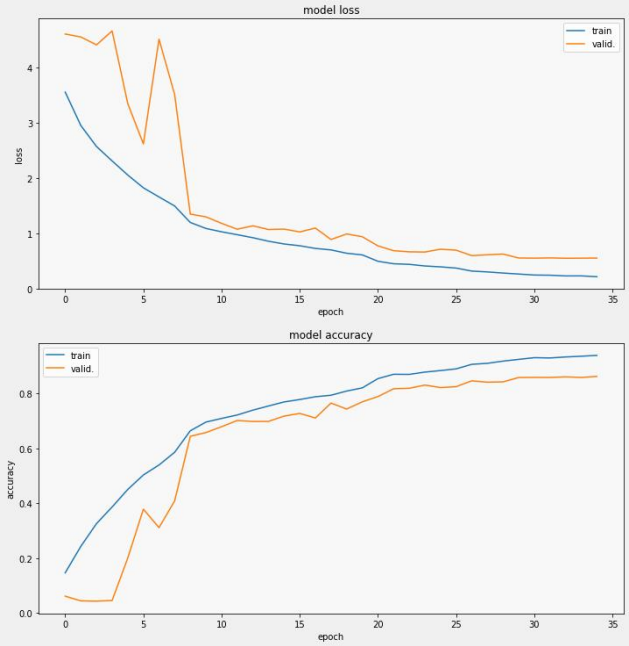

Figure 11:准确率和loss图

当增加训练轮数到 70 轮：

测试集f1 值	测试集准确率	测试集Recall	验证集loss
0.838	0.855	0.831	0.5694

后 35 轮的Loss和准确率图如下，可以观察到出现了一定的过拟合，在训练集上的准确率达到了 0.95，而测试集准确率只有 0.855。这说明难以再提高准确率。

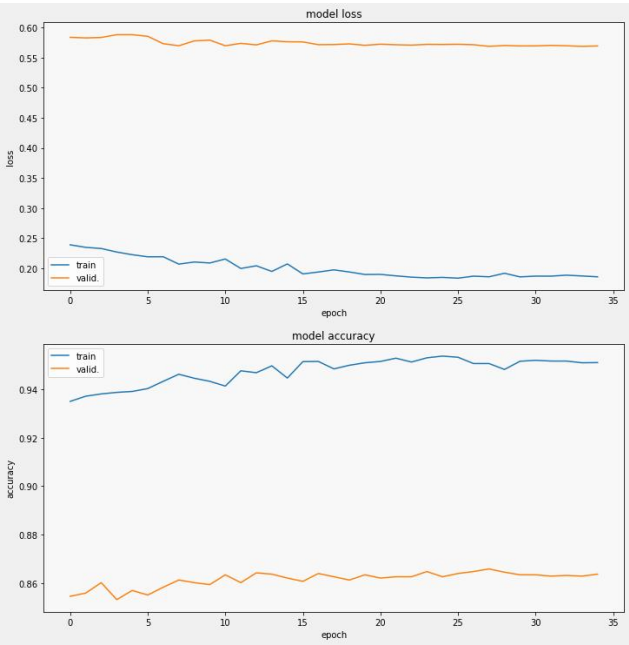

Figure 12:准确率和loss图

作为对比，我们训练未添加ImageNet预训练权重的tensorflow自定义的xception网络，得到的结果如下。

当训练轮数为 35 轮时

测试集f1 值	测试集准确率	测试集Recall	验证集loss
0.800	0.834	0.789	0.6478

Loss和准确率图如下，

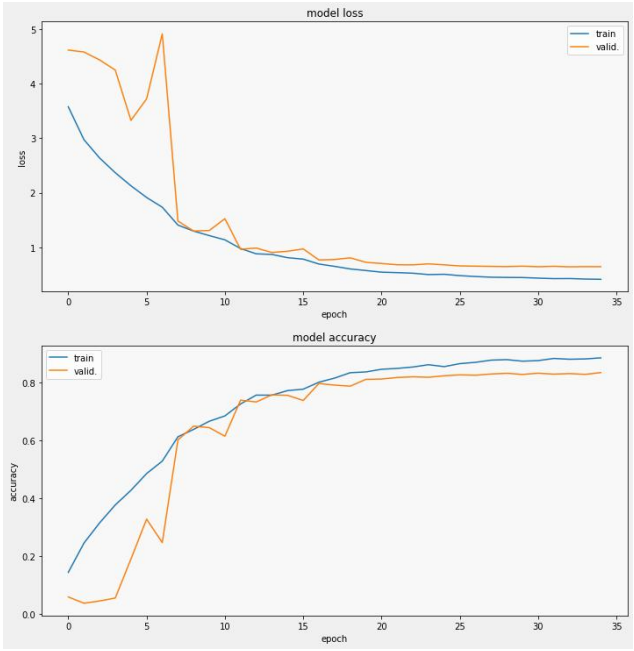

Figure 13:准确率和loss图

结果和我们定义的xception模型接近，说明我们自定义的xception模型没有出现错误。

总结

在此次研究中，我们发现

1.数据增强有利于提升分类模型的准确率；

2.Xception在该数据集上表现良好，而VGG16、NASNetMobile在该数据集几乎失效；

3.在随机权重下，Xception模型在分类任务中也有一定的效果，但在同一训练轮数的准确率上不及使用在ImageNet上预训练的权重的模型，体现了迁移学习的思想。

参考文献

[1] Chollet, F. (2017). Xception: Deep learning with depthwise separable convolutions. arXiv preprint arXiv:1610.02357.

[2]Jiang, X., Li, X., Li, G., Li, L., & Wang, X. (2019). Fisher Discriminant Analysis for Hyperspectral Image Classification: A Survey. IEEE Geoscience and Remote Sensing Magazine, 7(2), 18-36.

[3]LeCun, Y., Bottou, L., Bengio, Y. and Haffner, P. (1998). Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11), 2278-2324.

[4]Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., Manzagol, P. A. (2008). Extracting and composing robust features with denoising autoencoders. Proceedings of the 25th International Conference on Machine Learning, 1096-1103.
[5]Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... & Bengio, Y. (2014). Generative adversarial nets. In Advances in neural information processing systems (pp. 2672-2680).