南京農業大學

SRT 计划项目总研究报告

项目名称：基于 Keras 深度学习框架的多标签图像分类研究

申 请 者：邱 日

学　　院：信息科学与技术学院

专　　业：计算机科学与技术

指导教师：顾兴健　　职称：讲师

2018 年 5 月 11 日

目 录

基于 Keras 深度学习框架的多标签图像分类研究

项目编号：201710307010Y 项目主持人：邱日　成员：周宽；秦天涯　邱日
指导教师　顾兴健

transfer learning for multi-label image classification based on Keras

Student majoring in computer science and technology　Ri Qiu,Kuan Zhou,QTY
Tutor　Xinjian Gu

Abstract: Deep Neural Network has recently achieved outstanding performance in a variety of computer vision tasks,especially classification related tasks,including multi-class classification and multi-label classification.In multi-label classification problems, an image is associated with a set of labels simultaneously. Due to our very limited compute resources and little money,it is almost impossible to build and train a powerful image classification model totally from zero. However,by storing knowledge gained while solving one problem and applying it to a different but related problem, better results can be obtained, and this strategy is called transfer learning. ResNet-50(deep Residual Networks) is a powerful residual learning framework, these residual networks are easier to optimize, and can gain accuracy from considerably increased depth. Build and train a model very little data and still yield reasonable results is feasible with transfer learning and pre-trained models, namely ResNet-50. Building on prior work in this field, we do transfer learning with and without fine-tuning as well as the training of a neural network model from scratch on a much smaller multi-label image dataset.This multi-label image dataset is provided by NJU LAMDA,this dataset contains 2000 natural scenes images.

To address the above problem,we propose a novel deep transfer neural network based on multi-label learning for classification tasks.The existing approaches tend to treat multi-label problems as those for multi-class problems.Our model mainly consists of two sub-networks: the transfer learning network and the multi-label learning network. The transfer neural network is an extended ResNet-50 architecture. In data pre-processing stage, we convert 2000 images and their labels into tfrecord format(native file format used in Tensorflow allows programmers to shuffle, batch and split datasets with its own functions).we will shuffle and augment data,this helps prevent overfitting and help the model generalize better. We use the bottleneck features of a pre-trained network, namely ResNet50, which is pre-trained on the ImageNet dataset.Because the ImageNet dataset contains several nature scene classes , such as 'tree','desert','sea' and so on among its total of 1000 classes, this base model will already have learned features that are relevant to our multi-label classification problem under natural scene. To further improve this base model's (pre-trained Resnet model) performance, we 'fine-tune' the last convolutional block of the ResNet model as well as the top-level classifier. Our experimentations with real-world datasets demonstrate the effectiveness of the proposed approach.

Here we will simply introduce two neural network libraries that we used :TensorFlow and Keras. TensorFlow™ is an open source software library for high performance numerical computation. Its flexible architecture allows easy deployment of computation across a variety of platforms. Keras is a high-level neural networks API, written in Python and capable of running on top of TensorFlow, CNTK, or Theano. It was developed with a focus on enabling fast experimentation. Keras enable us to go from idea to result with the least possible delay.

Key words: Multi-label classification; transfer learning;pretrained model;ResNet;Keras;DNN

第 1 章

1.1 研究目的

计算机视觉中多标签分类 (Multi-label) 问题可以看做是多类别 (Multi-class) 问题的泛化，多标签分类将一个实例分到多个类别中去.

1.1.1 研究背景

随着移动互联网时代的到来, 图像数据与日俱增, 这就造就了图像大数据时代的到来, 传统的图像单标签分类问题已经无法满足含有复杂语义图像的分类识别要求, 这时候就需要分类识别速度快, 精度高的多标签图像分类技术, 该技术的出现推动了图像大数据时代的发展。关于多标签分类技术，一般过程是这样的：假如有了训练集（用于训练的图片）及测试集（用来测试的图片），首先提取训练集中的有标签图片，添加多层神经网络训练这些图片，得到训练模型，最后用训练模型检验测试集，来测试模型识别的精准度。这样的话，通常需要很大的测试集和训练次数，至少要有 10000 张测试图片、10000 次最大训练次数，才能得到一个较好的图片识别精准度，如果我们按照上述的步骤来完成我们的多标签分类的研究，这样做的意义并不大：一是这样需要非常大的电脑计算能力和计算开销，我们的机器难以达到这种要求；二是就算我们的电脑能支持这种开销，最终得到训练模型，别人完全可以单纯的通过增加测试图片和增加训练次数来超过我们的模型识别图像的精准度，且 Google 已经提供了不少花费了几十甚至数百个 GPU 小时的训练模型，那么这样我们的研究工作就没什么意义了。所以我们工作便放在了另一个方面上，可以这么想：你的训练图片和训练次数越多，你的图像识别精度越高，然而达到了一个较高的上限后，无论你再怎么添加训练图片和训练次数，你的图像识别精度就难以提升了，这时怎么提高精度呢? 这便是我们所要研究的工作，我们用到了深度学习之——迁移学习，通过使用已有的预训练模型，提取其中神经网络中信息的权重，迁移到我们神经网络，我们学习了这些特征，然后通过再训练来修正完善这些特征，便在已有的训练模型的基础上提高了图像识别的精准率，这样我们的研究变具有了价值性。我们工作的重点在于预训练模型中找到能够输出可复用特征（feature）的层次（layer），然后利用该层次的输出作为输入特征来训练那些需要参数较少的规模更小的神经网络.

1.1.2 研究意义

随着照片和视频分享网站的普及（Flicker,Instagram 和 Youtube 等等），大量的未标记的和弱标记的图像数据也随着传播开来。这越来越多的数据图像需要有效的数据检索来达到获取这个图像的最终目的。尽管图像检索已经被研究许多年了，搜索引擎检索目的图像仍然大多基于文本内容而非原始图像语义。因此，自动将图像标注成人类已经提供好的关键词或者标签的技术已经受到了大量的研究关注。该技术从开

始到现在的发展大致经历了三个阶段，分别是：基于文本的图像检索，基于内容的图像检索，基于语义的图像检索。基于文本的图像检索，由于手工标注工作量大，个人理解也有差异，导致检索结果出错率较高；基于内容的图像检索，是根据图像的底层特征进行检索，没有反映图像的内容信息；而基于语义的图像检索，弥补了上述两种方法的缺陷，使检索结果和使用者所需要的信息尽可能的一致。对于语义标注可以人工标注或者机器标注，由于上述所讲个人的理解因人而异，所以现在大多使用机器自动标注，即将等待标注的图像输入计算机系统，经过分割，将它划为等大的方格，而每个小方格的特征都被一个视觉特征词代替，让系统通过一系列算法推理出可能划分的语义，并将整个图像转为语义序列，相邻而重复的视觉词去掉然后将每个视觉词标注它所对应的物体标签。

1.1.3 研究目的

以多标签图像标注为目的，在 Keras 深度学习框架下，使用 Python 语言，结合 TensorFlow 库，开发图像标注的核心算法。结合 ResNet50 预训练模型和以上算法训练出属于我们自己的模型，在预训练模型的基础上提高其泛化能力及其图片识别的精准率，最后实现对图片的多标签分类。

1.2 Multi-label 问题简介

计算机视觉中分类问题是一个重要课题,图像分类有两类问题: 一是多分类 (multi-class) 问题，二是多标签 (multi-label) 分类问题. 我们要研究的是 multi-label classification 问题，在多标签分类中，一个实例可以对应多个标签. 比如，一张山水画就对应 ' 山 ' 和 ' 水 ' 或更多标签. 多标签分类问题可以看做是多分类问题的泛化，解决起来要比多分类问题困难很多.

1.3 研究方法

我们确定以机器学习中的深度学习模式来进行我们的研究。工具上采用 Tensorflow 和 Keras 这两个神经网络库。其中 Tensorflow 是谷歌官方开源的通用深度学习系统，而 Keras 则进行了更高层的封装，现已成为谷歌 Tensorflow 的默认 API

1.3.1 开发工具

我们的技术路线是 Tensorflow+keras+Kaggle

TensorFlow 介绍 TensorFlow 是谷歌基于 DistBelief 进行研发的第二代人工智能学习系统，其命名来源于本身的运行原理。Tensor（张量）意味着 N 维数组，Flow（流）意味着基于数据流图的计算，TensorFlow 为张量从流图的一端流动到另一端计算过程。TensorFlow 是将复杂的数据结构传输至人工智能神经网中进行分析和处理过程的系统 TensorFlow 可被用于语音识别或图像识别等多项机器学习和深度学习领域，且完全开源，任何人都可以用。TensorFlow 作为一个机器学习库能做很多事情：

- 可自行设计神经网络结构；

- 不需要通过反向传播求解梯度，Tensorflow 支持自动求导；
- 通过 C++ 编写核心代码，简化了线上部署的复杂度 (通过 SWIG 实现 Python，Go 和 JAVA 接口)；
- 谷歌的 Tensorflow 中内置 TF.Learn 和 TF.Slim 等组件，并兼容 Sciket-learn estimator 接口 (evaluate、grid、search、cross、validation)；
- 数据流式图支持自由的算法表达，可实现深度学习以外的机器学习算法；
- 可写内层循环代码控制计算图分支的计算，可将相关的分支转化为子图并执行迭代计算；
- 可进行并行设计，充分利用硬件资源。
- 具有灵活的移植性，编译速度较快；
- 在数据并行模式上：Tensorflow 主要面向内存足以装载模型参数环境，从而实现计算效率的最大化；
- 支持卷积神经网络 (Convolutional Neural Network ,CNN), 循环神经网络 (Recurrent Neural Network,RNN), 支持深度强化学习及计算密集的科学计算 (偏微分方程求解);

Keras 介绍 Keras 是一个高层神经网络 API，Keras 由纯 Python 编写而成并基 Tensorflow、Theano 以及 CNTK 后端。Keras 为支持快速实验而生，能够把你的 idea 迅速转换为结果，Keras 的特点：

- 简易和快速的原型设计（keras 具有高度模块化，极简，和可扩充特性）
- 支持 CNN 和 RNN，或二者的结合
- 用户友好：Keras 是基于 TensorFlow。用户的使用体验始终是它的中心内容。Keras 遵循减少认知困难的最佳实践：Keras 提供一致而简洁的 API，能够极大减少一般应用下用户的工作量，同时，Keras 提供清晰和具有实践意义的 bug 反馈。
- 模块性：模型可理解为一个层的序列或数据的运算图，完全可配置的模块可以用最少的代价自由组合在一起。具体而言，网络层、损失函数、优化器、初始化策略、激活函数、正则化方法都是独立的模块，你可以使用它们来构建自己的模型。
- 易扩展性：添加新模块超级容易，只需要仿照现有的模块编写新的类或函数即可。创建新模块的便利性使得 Keras 更适合于先进的研究工作。
- 与 Python 协作：Keras 没有单独的模型配置文件类型（作为对比，caffe 有），模型由 python 代码描述，使其更紧凑和更易 debug，并提供了扩展的便利性。

Kaggle 平台介绍 Kaggle 主要为开发商和数据科学家提供举办机器学习竞赛、托管数据库、编写和分享代码的平台。该平台已经吸引了 80 万名数据科学家的关注. Kaggle 提供了优秀的云编程的环境，还可以免费使用其服务器，达到在云上训练神经网络模型的目的. 我们的电脑配置都比较差，学校也没有给我们提供计算的服务器，经费远远不够租用 AWS 服务器, 给我们做实验和编写代码带来极大困难. 采用 Kaggle 平台

虽然计算资源还是比较有限，但略微改善了我们的实验环境. 而且 Kaggle 用 docker 部署，用到的 tensorflow、Keras 等库基本上都预装好，使我们免除配置环境的麻烦.

1.3.2 迁移学习

什么是迁移学习

迁移学习 (Transfer learning) 顾名思义就是就是把已学训练好的模型参数迁移到新的模型来帮助新模型训练。考虑到大部分数据或任务是存在相关性的，所以通过迁移学习我们可以将已经学到的模型参数（也可理解为模型学到的知识）通过某种方式来分享给新模型从而加快并优化模型的学习效率不用像大多数网络那样从零学习。

迁移学习的基本思路是利用预训练模型，即已经通过现成的数据集训练好的模型（这里预训练的数据集可以对应完全不同的待解问题，例如具有相同的输入，不同的输出）。开发者需要在预训练模型中找到能够输出可复用特征（feature）的层次（layer），然后利用该层次的输出作为输入特征来训练那些需要参数较少的规模更小的神经网络。由于预训练模型此前已经习得了数据的组织模式（patterns），因此这个较小规模的网络只需要学习数据中针对特定问题的特定联系就可以了。

迁移学习的一般过程

一般的迁移学习过程是这样的：训练好一个网络（我们称它为 base network）→把它的前 n 层复制到 target network 的前 n 层 →target network 剩下的其他层随机初始化 → 开始训练 target task。其中，在做 backpropogate（反向传播）的时候，有两种方法可以选择：（1）把迁移过来的这前 n 层 frozen（冻结）起来，即在训练 target task 的时候，不改变这 n 层的值；（2）不冻结这前 n 层，而是会不断调整它们的值，称为 fine-tune（微调）。这个主要取决于 target 数据集的大小和前 n 层的参数个数，如果 target 数据集很小，而参数个数很多，为了防止 overfitting（过拟合），通常采用 frozen 方法；反之，采用 fine-tune。

1.3.3 预训练模型

预训练模型 (pretrained model): 预训练模型是指前人为了解决某类问题所建立的模型。而当我们为了解决这种类似问题而借用这种模型, 这种模型就称为预训练模型。有了这个预训练模型，在我们解决这类问题的时候，我们可以借鉴前人的经验开始训练一个模型而不是从头开始为这个问题建立模型并且训练它。模型信息：如图 1.1所示.

我们用预训练模型是 ResNet50, ResNet50 函数原型

```
keras.applications.resnet50.ResNet50(include_top=True, weights='imagenet',
                                      input_tensor=None, input_shape=None,
                                      pooling=None,
                                      classes=1000)
```

模型	大小	Top1准确率	Top5准确率	参数数目	深度
Xception	88MB	0.790	0.945	22,910,480	126
VGG16	528MB	0.715	0.901	138,357,544	23
VGG19	549MB	0.727	0.910	143,667,240	26
ResNet50	99MB	0.759	0.929	25,636,712	168
InceptionV3	92MB	0.788	0.944	23,851,784	159
IncetionResNetV2	215MB	0.804	0.953	55,873,736	572
MobileNet	17MB	0.665	0.871	4,253,864	88

图 1.1 pre-trained models

50 层残差网络模型, 权重训练自 ImageNet 该模型在 Theano 和 TensorFlow 后端均可使用, 并接受 channels_first 和 channels_last 两种输入维度顺序模型的默认输入尺寸时 224x224. 参数如下

- include_top：是否保留顶层的全连接网络
- weights：None 代表随机初始化，即不加载预训练权重。'imagenet' 代表加载预训练权重
- input_tensor：可填入 Keras tensor 作为模型的图像输出 tensor
- input_shape：可选，仅当 include_top=False 有效，应为长为 3 的 tuple，指明输入图片的 shape，图片的宽高必须大于 197，如 (200,200,3)
- pooling：当 include_top=False 时，该参数指定了池化方式。None 代表不池化，最后一个卷积层的输出为 4D 张量。'avg' 代表全局平均池化，'max' 代表全局最大值池化。
- classes：可选，图片分类的类别数，仅当 include_top=True 并且不加载预训练权重时可用。

1.3.4 基本概念

激活函数

$$\sigma(x) = \frac{1}{1+e^{-x}}$$

图 1.2 sigmoid **函数**

激活函数（activation function）：运行时激活神经网络中某一部分神经元，将激活信息向后传入下一层的神经网络。神经网络之所以能解决非线性问题（如语音、图像识别），本质上就是激活函数加入了非线性因素，弥补了线性模型的表达力，把“激活的神经元的特征”通过函数保留并映射到下一层。这里我们使用的是 sigmoid 函数。这是传统神经网络中最常用的激活函数之一（另一个是 tanh）对应的公式和图像.

sigmoid Sigmoid 函数的优点在于，它的输出映射在（0,1）内，单调连续，非常适合用作输出层，并且求导比较容易。但是，它也有缺点，因为软饱和性，一旦落入饱和

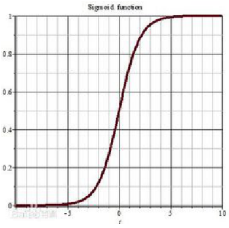

图 1.3 sigmoid **函数图像**

区，导数就会变得接近于 0，很容易产生梯度消失。

1.4 Resnet 简单原理

Resnet 要解决的大概是这样的问题：神经网络越来越深的时候，反传回来的梯度之间的相关性会越来越差，最后接近白噪声。因为我们知道图像是具备局部相关性的，那其实可以认为梯度也应该具备类似的相关性，这样更新的梯度才有意义，如果梯度接近白噪声，那梯度更新可能根本就是在做随机扰动。

有了梯度相关性这个指标之后，作者分析了一系列的结构和激活函数，发现 resnet 在保持梯度相关性方面很优秀。这一点其实也很好理解，从梯度流来看，有一路梯度是保持原样不动地往回传，这部分的相关性是非常强的。

resnet 的提出就是为了解决网络加深时的训练精度下降问题，也被作者称之为退化问题。

1.4.1 实验过程

数据的预处理

数据的预处理在机器学习中往往被忽略，其实它很重要，并且不容易做好。

数据集的确定 首先确定数据集，数据集的选取决定了最终训练效果的上限，如果数据集质量不错的情况下，那数据当然多多益善。我们的多标签图像分类任务中，可以选择著名的 ImageNet 、MS COCO 数据集或者更大的 OpenImage，我们前期也试图去使用。深度学习很重要的一个要求是大量的计算资源和充足的经费，然而作为普通的本科学生，学校提供的活动经费和我们可用的计算资源实在是太微不足道了，完全不能在这些数据集上完成训练任务。

Multi-label 问题中的数据集很多，但绝大多数都不适用于我们经费和计算资源极其有限的现实情况，因此找到一个大小对我们合适，而又符合我们研究目的的数据集是很困难的。我们先后尝试了很多，最后确定了南京大学周志华教授 LAMDA 组公开的多示例多标签数据集 miml-image-data，数据集由 2000 张自然风景图片和标签组成：图片的内容有山、水、日落、海、树这五类组成，其中每个图片内容包含一类或多类; 图片对应的标签在.mat 格式的文件中给出。.mat 是 Matlab 的文件格式，这个文件中有数据集的几个属性，有 target 等属性，图片标签由 [-1,1,-1,-1,1] 这种形式给出，依次对应那五类，-1 则代表没有，1 代表图片有这个标签。整个数据集压缩后大小为 28MB.

数据的清洗与读入 南大提供的数据集应该还是比较干净的，数据的清洗基本上只要统一大小。原始图片和.mat 的标签并不利于模型的读入，利用 tensorflow 和之上的 keras 做开发时把它们处理后保存为 tensorflow 的结构化数据格式是比较理想的方式，这样可以更高效和方便地加载数据。为了提高预测效果和增强模型的泛化能力，我们在此转化的过程中还做了数据的增广 (data augmentation), 将其中的图片改变大小和剪裁等，因此转化后的数据比原始数据大几倍.

预训练模型的加载

预训练模型采用 Resnet50 模型,Resnet 由何恺明提出，意为深度残差网络，在 kaggle 网站上可以下载到很多预训练模型，这里使用 Resnet50，下载到.h5 文件将它放在.keras 模型配置文件夹.keras/models 中。

为什么要使用预训练模型? Google 为在 Imagenet 上训练神经网络，使用了很多 GPU, 连续训练了几周，使用了大数据和大量计算资源，这些预训练好的模型可以看做以二进制存放的智慧. 作为普通本科生，学校提供给我们的项目经费和计算资源极其有限，若不使用预训练模型，不可能达到有意义的训练效果。而且当数据集也极其有限的时候，预训练模型显得更加必要。

使用预训练模型将给模型的训练一个很好的起点，最终也往往也取得比直接在目标数据集从 0 开始的训练更好的效果。

模型预测层的再建立 预训练模型大体由两部分组成，一是抽取特征的部分，二是用于预测的部分。迁移学习有一定的适用条件，预训练的模型在现实场景的 ImageNet 数据集上训练，我们自己使用的模型也是现实的自然风景，因此是符合迁移学习的适用条件的。预训练模型使用了大量的计算资源，有效地学习到了抽取的图像特征，这是宝贵的经验，应当留下来。我们的计算资源和经费极其有限，数据集的规模也很小，预训练模型学习到的特征应当尽可能多地保留下来。

有了预训练模型，我们的起点就比较高，因此学习速率 (learning rate) 应当调小, 否则将很快将之前在大数据训练中得到的合适权重带偏，然后在小数据集上很快陷入过拟合状态，预测效果反而下降。因此训练过程中应当把预训练模型的前面很多层的权重都冻结，我们可以训练预训练模型的最后几层，或者自己再在预训练模型除预测部分的特征提取的最后加上自己建立的卷积层、池化层等，在我们的小数据集训练。通过反向传播调整特征提取部分最后几层的权重，这样就达到了既不扰乱预训练模型训练得到的大部分合适权重，又能有针对性地在我们的小数据集上训练和强化的效果。这样我们就能通过小型数据集，利用极其有限的计算资源取得理想的训练效果。

如图 1.4 所示。

如图 1.5 所示。

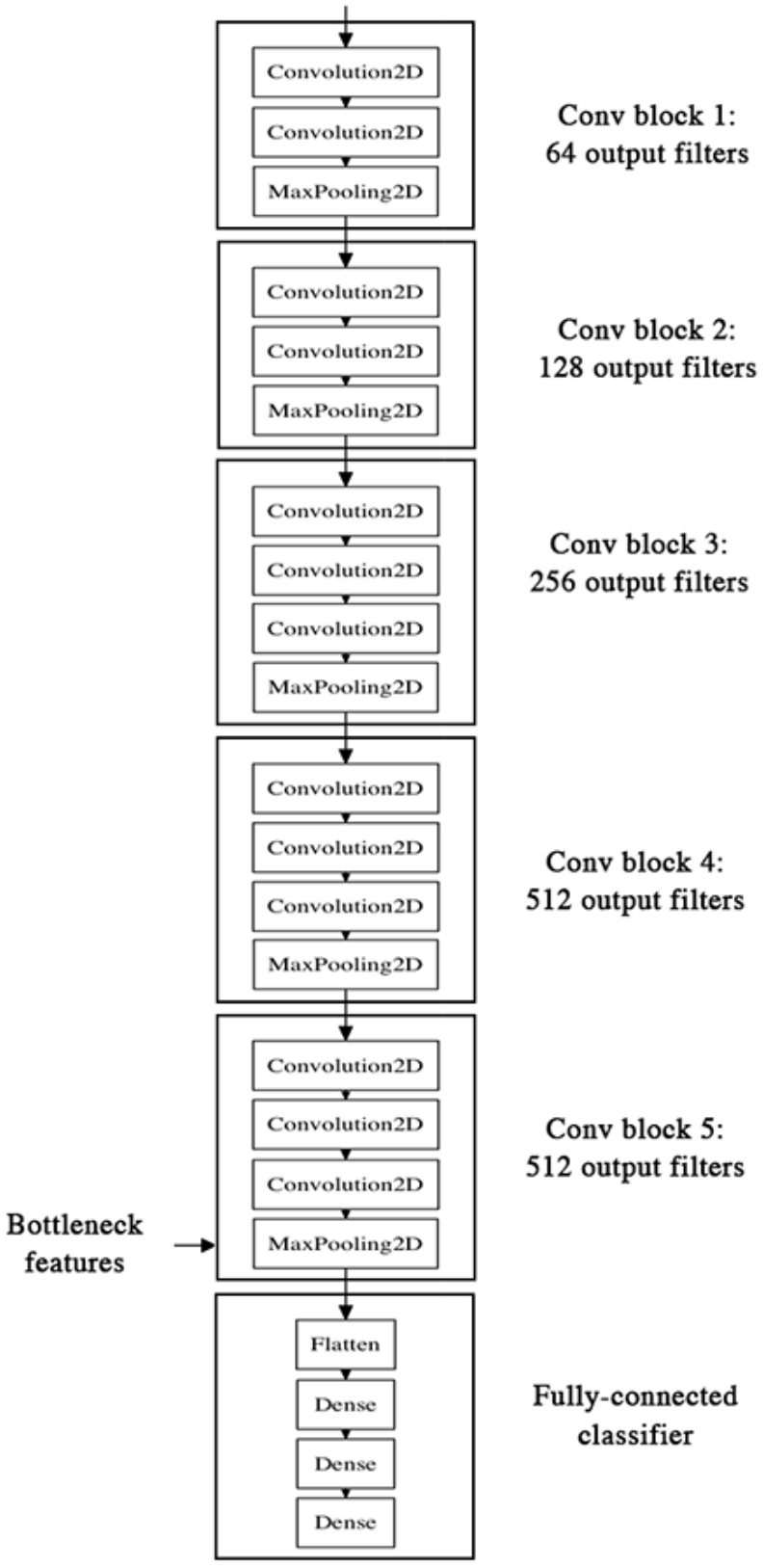
Convolution2D
Convolution2D
MaxPooling2D
Conv block 1:
64 output filters
Convolution2D
Convolution2D
MaxPooling2D
Conv block 2:
128 output filters
Convolution2D
Convolution2D
Convolution2D
MaxPooling2D
Conv block 3:
256 output filters
Convolution2D
Convolution2D
Convolution2D
MaxPooling2D
Conv block 4:
512 output filters
Convolution2D
Convolution2D
Convolution2D
MaxPooling2D
Conv block 5:
512 output filters
Bottleneck
features
Flatten
Dense
Dense
Dense
Fully-connected
classifier

图 1.4 VGG 模型

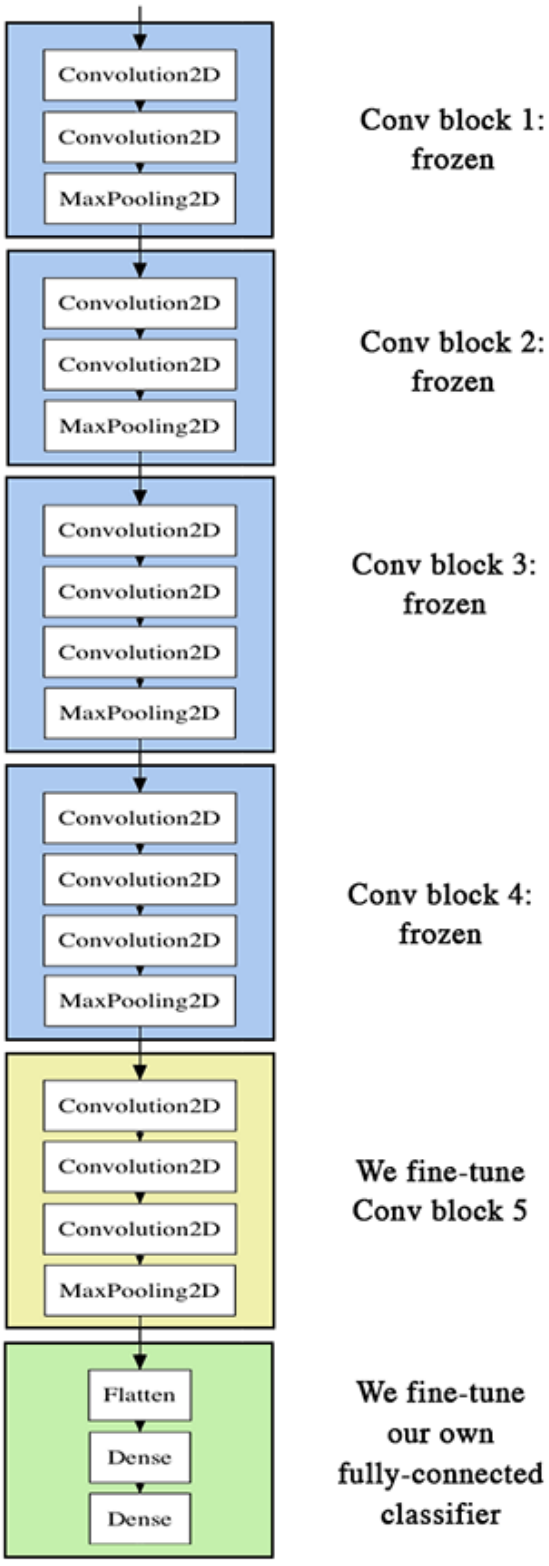

图 1.5 vgg16 modified

模型的训练

以识别的准确率作为 loss, 通过 keras 的 model.fit 函数在数据集上调整权重，训练模型。训练的过程保存在 history 中，然后我们可以通过后者画出准确率随时间变化的曲线图。

训练任务在我们的笔记本上进行。笔记本没有强大的英伟达显卡，内存也很有限，因此实验过程还是花了一个多小时。从图中，我们可以看到，由于使用了预训练模型，一开始起点还是比较高的，但是也很有限，不过要好于从 0 开始的乱猜，然后训练中效果提升得很快，当然其中也存在过拟合的情况，效果提升到一定程度之后，出现瓶颈，在那个点上下波动，之后有陷入过拟合的趋向，我们停止训练，保存为训练好的模型。

我们的准确度最终已经达到 99.3%，之后可以进行预测了.

1.4.2 模型的预测

模型大致分为两个部分，一是特征提取部分，二是预测部分。特征提取部分的建立前面已经论述过，这里来谈模型的预测部分。迁移学习中，迁移之前的预训练和迁移之后的训练，目标并不完全相同，因此预测层需重新建立。

Resnet50 的预训练目标是 1000 种日常物体多分类 (multi-class) 问题，我们的迁移目标是五种自然景物的多标签 (multi-label) 问题. 这里物体种类的数目 (classes) 不同, 而且问题性质也不同。实际上很多粗略解决多标签问题的方法都是简单地将其当成多分类问题来解决。这实际上不是很严谨，但由于训练比较充分，简单地取概率上前几个物体，识别效果还可以。

我们使用的方法是严格地把它作为多标签问题处理.softmax 函数适用于 multi-class 问题，用在 multi-label 问题中并不是很恰当，有种祖略的方式是对 N 个标签，使用 N 个 softmax 函数，这样所有的标签的概率加起来为 1。我们使用的是 sigmoid 激活函数，这样对于每个标签，都是独立的伯努利分布的概率，更加符合 multi-label 任务，经检验也比 softmax 的识别效果好。

1.5 结论和建议

1.5.1 研究成果

我们通过迁移学习，使预测模型开始就有大概 50% 的准确率，将预训练模型抽取的特征很好地保留下来，通过冻结预训练模型前面多层的权重，有效地避免在小数据集上训练将之前习得的图像特征给丢失，同时再在特征提取层面添加新层允许权值通过反向传播改变，达到在新任务中有效提取特征的目的.

1.5.2 创新点

深度学习需要大量的计算资源和资金的支持. 若无这些支持，从 0 开始设计并从头训练一个具有较高性能的的多标签分类模型基本是不可能的. 本次实验实验条件

极其有限，我们通过迁移学习，在小数据集上取得了不错的效果.

参考文献

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun，Deep Residual Learning for Image Recognition，arXiv:1512.03385