机器学习工程师纳米学位毕业项目

猫狗大战

庞浩

2017 年 2 月 23 日

目　录

1 定义

1.1 项目概述

这是一个 Kaggle 娱乐型竞赛项目，目标是训练一个模型，在给定的图像中分辨猫和狗。这是一个图像分类问题，是典型的计算机视觉问题。

本项目使用的模型是卷积神经网络（Convolutional Neural Network，CNN）。“卷积神经网络”一词表明该网络是用了卷积这种数学运算。卷积是一种特殊的线性运算。CNN 是指那些至少在网络的一层中使用卷积运算来替代一般矩阵乘法运算的神经网络。

CNN 在深度学习的历史中发挥了重要作用。它是将研究大脑获得的深刻理解成功用于机器学习应用的关键例子，也是第一个表现良好的深度模型之一。CNN 还是第一个解决重要商业应用的神经网络，并且仍然是当今深度学习商业应用的前沿。

项目选择的数据集是 Kaggle 竞赛提供的数据，训练集包括 12500 张被标记为猫的图片和 12500 张被标记为狗的图片，测试集包括 12500 张未标记图片。对于每一张测试集中的图像，模型需要预测出是狗图像的概率（1 代表狗，0 代表猫）。

1.2 问题陈述

Kaggle 竞赛提供的数据是从真实世界采集的包含猫或狗的图像。图像分辨率差异较大，质量参差不齐，图片中的猫和狗颜色丰富，品种多样，姿态各异，背景环境复杂，这些都增加了分类难度。

在处理图像的过程中，实际存在两个子过程：特征提取和对象分类。在特征提取环节，模型以模型原始像素点为基础，从中提取边缘、纹理、对象轮廓、色彩组合，等不同抽象级别的特征。在对象分类部分，需要将抽取的特征输入到一个分类其中，得到分类的结果，常见的分类器有 Softmax，SVM 等。

1.3 评价指标

在 2013 年举行的第一次猫狗大赛中，使用的评价指标为分类正确率。对于有限的待测试样本，正确率指标是一个离散值，在深度学习还未普及的时候，用于度量模型的分类能力是没问题的。

本次猫狗大赛（Dogs vs. Cats Redux: Kernels Edition）中，使用的评价指标为对数损失，分数计算方法如下：

$$\text{LogLoss} = -\frac{1}{\text{n}}\sum_{i=1}^{n}[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)]$$

这个指标是一个连续值。在深度学习大行其道的今天，模型对图像的分类能力都很强，为了对模型进行更细致的评价，改用这一指标是合理的。模型不但需要能够分类正确，而且需要分类十分肯定。

从建模的角度看，由于评价指标从正确率变为了对数损失，因此在建模像 L2 正则化这样的优化方式就不宜使用了，这样的正则化在防止过拟合时会惩罚对分类结果的过度肯定，这是不利于得到较高分数的。另外，在多模型整合时也不能直接采用投票或求平均值的方式了，这对优化对数损失也没有直接的好处。

2 分析

2.1 数据可视化

竞赛数据集有两个子集，一个训练数据集，一个测试数据集，训练数据集中图片的文件名为标签，说明了图片类别是猫还是狗。

图 1 训练集图片示例

训练集共有图片 25000 张，其中猫狗各一半，测试集共有图片 12500 张。从图 1 的可视化结果能够看出，图片具有丰富的多样性。大部分图像质量和分辨率良好，个别图像分辨率过小或质量较差，例如训练集右下角的那副图像，亮度有些偏大，可能会影响识别。

另外，由于猫狗所占比重一样，在训练时不需要调整类别权重。

2.2 算法和技术

2.2.1 深度学习

深度学习并不是一个全新的概念。一般来说，截止到目前已经有三次深度学习的发展浪潮：在 20 世纪 40 年代到 60 年代深度学习被称为控制论，20 世纪 80 年代到 90 年代深度学习被誉为连接机制，并于 2006 年开始，以深度学习之名复兴。

在领域的发展初期，一些最早的学习算法，旨在模拟生物学习的计算模型，其结果是产生了人工神经网络（Artificial Neural Network，ANN）。那时，深度学习模型对应的观点是学习系统是受生物大脑所启发。但是，它们一般都没有被设计成生物功能的真实模型。在 20 世纪 50 年代，感知机成为第一个能根据每一个类别输入的样例来学习权重的模型。同时期出现的，用于调节权重的随机梯度下降（Stochastic Gradient Descent, SGD）训练算法，仍然是当今深度学习的主要训练算法。

现在，神经科学被视为深度学习研究的一个重要灵感来源，但它已不再是该领域的主要导向。主要原因是没有足够的关于大脑的信息作为指导。要获得对大脑实际使用算法的深刻理解，我们需要有能力同时检测至少数千相连神经元的活动。现代科学无法做到这一点，甚至连大脑的最简单、最深入研究的部分人们都还远远没有理解。

现代术语“深度学习”超越了目前机器学习模型的神经科学观点。学习的多层次组合这一更普遍的原则更加吸引人，这可以应用于机器学习框架且不必是受神经启发的。

在 20 世纪 80 年代，连接机制是在认知科学的背景下出现的。认知科学是理解心智，并结合多个不同层次分析的跨学科方法。连接机制的中心思想是，当网络将大量简单计算单元连接在一起时，可以实现智能行为。这种见解同样适用于与计算模型中隐藏单元作用类似的生物神经系统中的神经元。这一时期涌现出了众多的重要成果，很多一直沿用至今：

- 分布式表示：这一想法是系统每个输入应该是由许多特征表示的，并且每个特征应参与多个可能输入的表示。
- 反向传播算法：到目前为止，仍是训练深度模型的主要方法。
- 循环神经网络及长短期记忆网络：如今仍在许多序列建模任务中广泛应用。
- 卷积神经网络：如今仍在许多计算机视觉建模任务中广泛应用。

那时，普遍认为深度网络是难以训练的。事实上，20 世纪 80 年代就存在的算法能工作得非常好，但是直到 2006 年都没有体现出来。这个问题可能只是因为计算复杂性太高，而以当时可用的硬件难以进行足够的实验。

神经网络研究的第三次浪潮始于 2006 年，这次浪潮普及了“深度学习”这一术语，强调研究人员现在可以训练以前不可能训练的更深的神经网络，并把注意力集中于深度的理论

意义。此时，深度神经网络已经优于与之竞争的基于其他机器学习技术以及手工设计函数的AI 系统。

事实上，这一阶段最重要的新进展是现在人们有了这些算法成功训练所需的资源。截至2016 年，一个粗略的经验法则是，监督深度学习算法一般在每类给定约 5000 标注样本情况下可以实现可接受的性能，当至少有 1000 万标注样本的数据集用于训练时将达到或超过人类表现。另外，随着数据量增加，模型的规模、复杂度和计算量也在快速地增加。

2.2.2 深度卷积神经网络

卷积网（Convolutional Network），也称为卷积神经网络（Convolutional Neural Network，CNN)，是一种专门用来处理具有类似网格结构的数据的神经网络。CNN 在诸多应用领域都表现优异。“卷积神经网络”一词表明该网络是用了卷积这种数学运算。卷积是一种特殊的线性运算。CNN 是指那些至少在网络的一层中使用卷积运算来替代一般矩阵乘法运算的神经网络。

在深度学习的大背景下，卷积神经网络的普遍深度已经达到几十层，这样的卷积神经网络被称为深度卷积神经网络（Deep Convolutional Neural Network，DCNN)。方便起见，在本文接下来的内容中，CNN 与 DCNN 等价。

CNN 在深度学习的历史中发挥了重要作用。它是将研究大脑获得的深刻理解成功用于机器学习应用的关键例子，也是第一个表现良好的深度模型之一。CNN 还是第一个解决重要商业应用的神经网络，并且仍然是当今深度学习商业应用的前沿。

CNN 是用反向传播训练的第一个有效的深度网络之一。现在仍然不完全清楚为什么 CNN 在一般的反向传播网络被认为已经失败时反而成功了。可能简单地归结为 CNN 比全连接网络计算效率更高，因此使用它运行多个实验并调整它们的实现和超参数更容易。利用现代硬件，大型全连接网络对许多任务也执行得很合理。也可能神经网络成功的主要阻碍是心理，比如:实践者没有期望神经网络有效,以至于他们没有认真努力地使用神经网络。无论如何，幸运的是 CNN 在几十年前就表现良好。在多个方面，它为余下的深度学习传递火炬，并为一般意义的神经网络被接受铺平了道路。

在 CNN 中，卷积层的数据处理通常包括三个步骤。首先，卷积层通过多次卷积计算获得一组线性激活函数；然后，非线性激活函数作用在每一个线性输出上进行探测；最后，可以使用池化函数来更进一步地调整卷积层的输出。这一过程包含了 CNN 的核心思想。

2.2.3 卷积层的基本原理

在泛函分析中，卷积是通过两个函数x和w生成第三个函数的一种数学算子，表征函数x与经过翻转和平移w的重叠部分的面积。

设：$x(a)$，$w(a)$是$\mathbb{R}$上的两个可积函数，作积分：

$$s(t) = \int x(a)w(t-a)da$$

这样，随着t的不同取值，这个积分就定义了一个新函数$s(t)$，称为函数x与w的卷积，记为：

$$s(t) = (x * w)(t)$$

这种运算就叫做卷积运算，卷积运算通常用*表示。

一般地，当使用计算机处理数据时，输入会被离散化，需要使用离散形式的卷积。事实上，从离散角度看卷积，会更加清楚。

对于两个序列$x(a)$和$w(a)$，一般可以将其卷积定义为：

$$s(t) = \sum x(a)w(t-a)$$

在 CNN 的术语中，第一个参数（函数x）叫做输入，第二个参数（函数w）叫做核函数。输出有时被称为特征映射。

如果把二维图像I作为输入，则也相应的需要使用二维的核K：

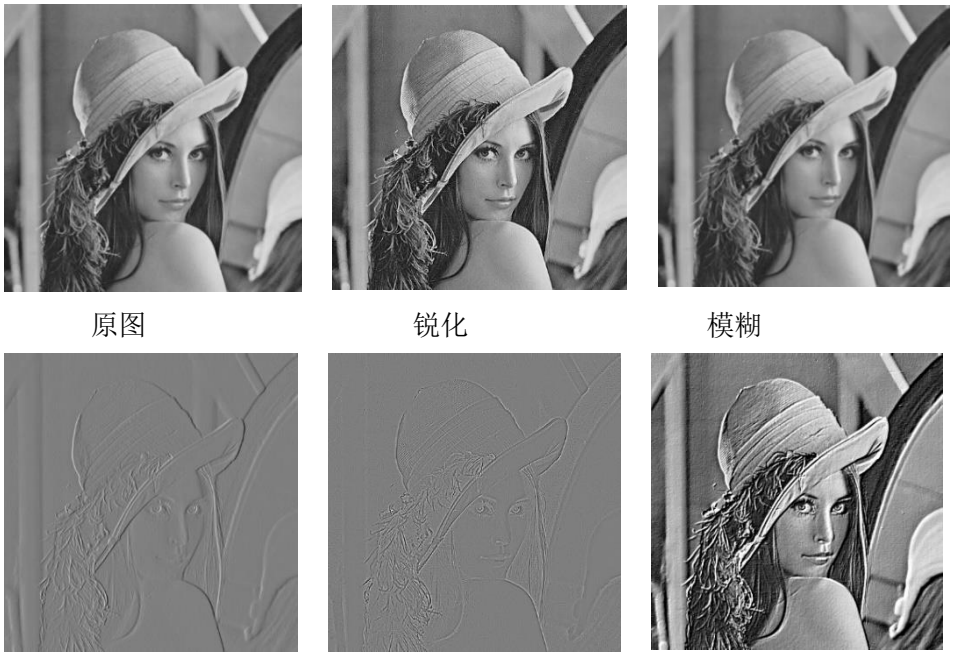
$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(m,n)K(i-m, j-n)$$

卷积核在计算机视觉和图像处理中被称为“滑动函数”，在信号处理中被称为滤波，而且是线性滤波，即：输出像素是输入域像素的加权和。不同的卷积核有不同的功能，例如，下面五个卷积核分别可以对原始图像进行会锐化（Sharpen）、模糊（Blur）、边缘增强（Edge Enhance）、边缘检测（Edge Detect）和凸印（Emboss）。

$$Sharpen = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix} \quad Blur = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad Emboss = \begin{bmatrix} -2 & -1 & 0 \\ -1 & 1 & 1 \\ 0 & 1 & 2 \end{bmatrix}$$

$$Edge\ Enhance = \begin{bmatrix} 0 & 0 & 0 \\ -1 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad Edge\ Detect = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

在实际图像上应用这五种卷积核做滑动窗口卷积运算，结果图 2-3 所示：



原图　　锐化　　模糊

边缘增强　　边缘检测　　凸印

图 2 五种卷积核的实际直观表现

从神经网络的角度看，卷积运算是通过三个重要的思想来帮助改进机器学习系统的：

1、稀疏交互（sparse interactions）

在传统的深度神经网络中，使用的仅仅是在感知机基础上的深度网络，每一层都是全连接的，例如：一个 100×100 的图像可以表示为一个长度为 10000 的向量，如果在第一个隐藏层的节点数和输入层一样，那么需要的参数就达到了 10000×10000=10^8 个，导致训练量非常大，而且过多的参数还非常容易过拟合。

假设使用 3×3 大小的卷积核处理图像，原始图像中的像素点x在卷积核滑动时仅有 9 个可能的位置，如果对图像从下往上看，最多只有 9 个输出受到x的影响。设输出图像上的某一点为y，如果对图像从上往下看，则只有 9 个输入能够影响到y。若连接不是稀疏的，则所有输出都会受到x的影响，所有的输入都能够影响到y。

事实上，处于 CNN 网络更深的层中的单元，它们的接受域要比处在浅层的单元的接受域更大。这意味着在 CNN 中，即使直接连接都是很稀疏的，处在更深层中的单元仍可以间接地连接到全部或者大部分输入图像。

2、参数共享（parameter sharing）

参数共享是指在一个模型的多个函数中使用相同的参数。在 CNN 中，卷积核的每一个元素都作用在输入的每一个位置上。卷积运算中的参数共享保证了只需要学习一个参数集合，而不是对于每一个位置都需要学习一个单独的参数集合。

由于图片的底层特征是与特征在图片中的位置无关的。比如边缘，无论在图片什么位置的边缘，都可以用相同的卷积核进行特征提取。假设使用 3×3 大小的卷积核处理图像，输出层的每一个像素都是由输入层对应位置的 3×3 大小的局部图像，与相同的一组 3×3 大小的卷积核参数做内积，再经过非线性单元计算而来的。这样无论图片原始大小如何，只需要使用 3×3 个参数就够了。一组参数只能得到一个特征映射，实践中一般会有多组参数，同时提取多组不同的特征。

3、等变表示（equivariant representations）。

对于 CNN，参数共享的特殊形式使得网络层具有对平移等变的性质。如果函数$f(x)$与$g(x)$满足$f\big(g(x)\big)=g\big(f(x)\big)$，则$f(x)$对于变换$g$具有等变性。对于卷积来说，如果令$g$是输入的任意平移函数，那么卷积函数对于$g$具有等变性。当卷积产生了一个二维映射来表明某种属性在输入的什么位置出现了。如果移动输入中的对象，它的表示也会在输出中移动同样的量。卷积对其他的一些变换并不是天然等变的，例如对于图像尺度或者角度的变换，需要其他的一些机制来进行处理。

可以把卷积的使用当作是对网络中一层的参数引入了一个无限强的先验概率分布。这个先验表达了该层应该学得的函数只包含局部连接关系，并且对平移具有等变性。

2.2.4 技术实现

项目实现 CNN 使用的是 Google 开源的 TensorFlow。项目同时使用了开源机器学习库 Keras，作为对 TensorFlow 的上层封装，以便能够使用更多的高级 API，加快开发的速度。另外，由于使用 CNN 处理大分辨率全彩色图片所需的计算量非常的大，项目使用了 GPU 进行计算加速。在安装 TensorFlow 和 Keras 的同时还需要安装 CUDA 和 cuDNN。

目前，主流的深度学习框架均包含以下五个核心组件：

1. 张量（Tensor）
2. 基于张量的各种操作（Operation）
3. 计算图（Computation Graph）
4. 自动微分（Automatic Differentiation）工具
5. 代数计算、GPU 加速等拓展包（BLAS、cuBLAS、cuDNN）

张量是所有深度学习框架中最核心的组件，因为后续的所有运算和优化算法都是基于张量进行的。将各种各样的数据抽象成张量表示，然后再输入神经网络模型进行后续处理是一种非常必要且高效的策略。因为如果没有这一步骤，开发者就需要根据各种不同类型的数据组织形式定义各种不同类型的数据操作，这会浪费大量的精力。更关键的是，当数据处理完成后，还可以方便地将张量再转换回想要的格式。

事实上，整个神经网络都可以简单视为为了达到某种目的，针对输入张量进行的一系列操作过程。而所谓的“学习”就是不断纠正神经网络的实际输出结果和预期结果之间误差的过程。这里的一系列操作包含的范围很宽，可以是简单的矩阵乘法，也可以是卷积、池化等稍复杂的运算。

有了张量和基于张量的各种操作之后，下一步就是将各种操作整合起来，输出我们需要的结果。但随着操作种类和数量的增多，有可能引发各种意想不到的问题，包括多个操作之间应该并行还是顺次执行，如何协同各种不同的底层设备，以及如何避免各种类型的冗余操作等等。计算图正是为解决这一问题产生的。因为计算图的引入，开发者得以从宏观上俯瞰整个神经网络的内部结构，计算图也可以从宏观上决定代码运行时的 GPU 内存分配，以及分布式环境中不同底层设备间的相互协作方式。

计算图带来的另一个好处是让模型训练阶段的梯度计算变得模块化且更为便捷，也就是自动微分法。由于可以将神经网络视为由许多非线性过程组成的一个复杂的函数体，而计算图则以模块化的方式完整表征了这一函数体的内部逻辑关系，因此微分这一复杂函数体，即求取模型梯度的方法就变成了在计算图中简单地从输入到输出进行一次完整遍历的过程。目前许多计算图程序包都已经预先实现了自动微分。

由于上述的大部分实现都是基于高级语言的，而即使是执行最简单的操作，高级语言也会比低级语言消耗更多的 CPU 周期，因此运算缓慢就成了高级语言的一个天然的缺陷。目前，主流深度学习框架都是利用脚本语言实现前端建模，用低级语言如 C++实现后端运行。一个显著的加速手段就是利用现成的扩展包，开发者可以根据个人喜好灵活选择。另外，一般的 BLAS 库只是针对普通的 CPU 场景进行了优化，但目前大部分的深度学习框架都已经开始采用并行 GPU 的运算模式，因此利用诸如 NVIDIA 推出的针对 GPU 优化的 cuBLAS 和 cuDNN 等更据针对性的库是更好的选择。

2.3 基准指标

在图像分类问题上，最有名的就是 ImageNet 竞赛了。到目前为止，对 ImageNet 数据集上的 1000 个类别进行分类，最好的 Top-5 错误率大约在 3%左右。本次竞赛仅针对猫狗进行分类，若想错误率在 3%以内，则对数损失必须控制在 0.1 以内。

考虑到训练 CNN 需要大量的硬件资源与计算时间，受限于客观条件，项目设定的目标为分类对数损失分数小于 0.08。

3 具体方法

3.1 迁移学习

3.1.1 模型选择

CNN 的核心是对图像特征的表示，从最基本的边缘、纹路，到较高抽象层次的对象。CNN 通过各层卷积核及其对应的权重来记录这些特征，同时，通常情况下利用最后一层的 softmax 进行概率分配，从而得出分类结果。由此可见，大部分卷积神经网络是用来做特征提取的，只有最后的很少几层是用来得出分类结果的。

从零开始训练一个卷积神经网络，需要进行细致的网络结果设计，并调整大量的参数进行优化。幸运的是，在知名计算机视觉竞赛 ImageNet 中，类别就含有猫和狗，同时，许多优秀的网络结构都提供了在 ImageNet 数据集上的训练结果。由此，利用现有的模型以及预训练出的权重，就能很好的表征猫和狗的特征，再辅以一个简单的分类器就能够获得一个相对理想的结果。

本项目使用的是 Keras 库，其官方 Github 网站[1]提供了 VGG16、VGG19、InceptionV3、ResNet50 和 Xception 这 5 个模型及其在 ImageNet 数据上的训练结果。考虑到 VGG16 和 VGG19 是较老的模型，这里只使用 InceptionV3、ResNet50 和 Xception 这三个模型。

3.1.2 数据预处理及模型调整

由于是基于现有模型和权重进行计算，因此数据预处理必须与想用的模型匹配。通过阅读源码可知，这三个模型在读取图片时所做的预处理是不同的。

InceptionV3 和 Xception 这两个模型的图片预处理方式为将数据从（0，255）缩放到（-1，1）区间内。ResNet50 模型的图片预处理方式为 RGB 每一个色彩通道都减去 ImageNet 数据集相应通道的均值（103.939，116.779，123.68），并将通道顺序从 RGB 调整为 BGR。

另外，这三个模型默认的输入分辨率也不同。Xception 和 InceptionV3 的默认输入分辨率为 299x299，ResNet50 的默认输入分辨率为 224x224。输出的形式也不一样，有用 AveragePooling2D 层输出的，有用 GlobalAveragePooling2D 输出的。

为了便于使用，这里对这三个模型进行了相应的改造：

- 统一使用 299x299 大小的图片作为输入
- 统一使用 GlobalAveragePooling2D 作为特征输出，输出的特征数为 2048

[1] https://github.com/fchollet/deep-learning-models

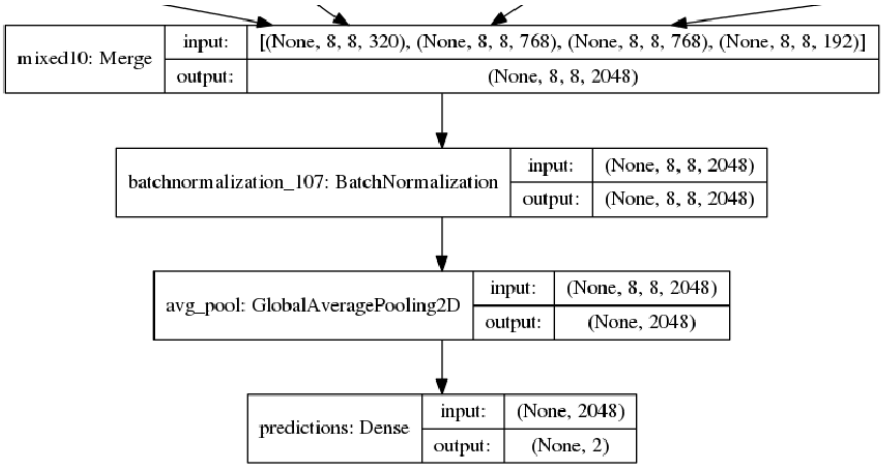


图 3 InceptionV3 输出部分

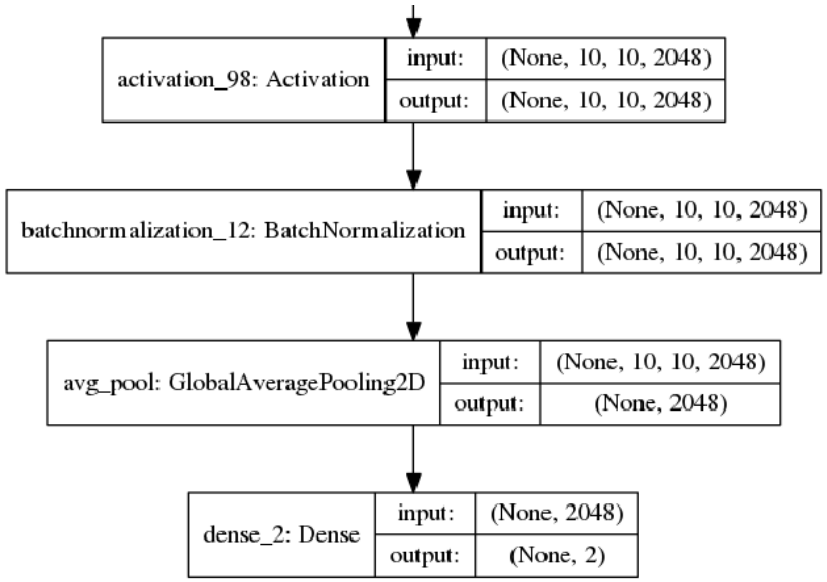


图 4 ResNet50 输出部分

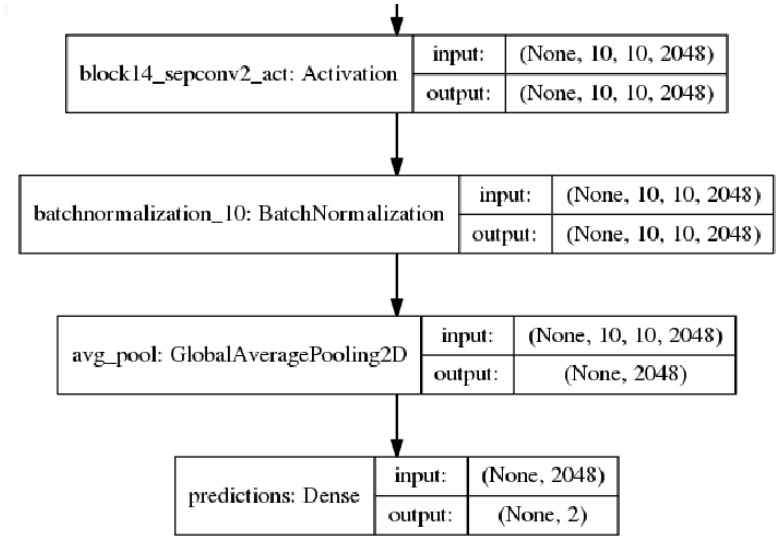


图 5 xception 输出部分

为了便于 Keras 加载数据，需要先将原始训练数据分割为训练集和验证集。这里，使用 1000 张猫图像和 1000 张狗图像做验证集，其余 23000 张图像做训练集。

3.1.3 模型的 fine-tune

除了将基础模型作为特征提取器，构建迁移模型外，直接利用基础模型做微调也是一个常见的策略。这一策略特别适合于使用小数据集构建图像分类模型。

各模型 finetune 之前的训练参数均为 8194 个，本项目针对模型各自的特点设置了 fine-tune 的范围，大幅增加了参与训练的参数个数：

- Xception：添加了 block13 和 block14 两部分，共 6796578 个参数。
- ResNet50：添加了 stage5 的部分，共 14984194 个参数。
- InceptionV3：所有 8x8x2048 的部分，共 11136130 个参数。

3.2 实现

3.2.1 模型的训练

为了提高训练集的样本数量，程序对训练集合验证集的图片都进行了数据增强变换，包括：水平翻转、在 0.05 范围内缩放、在 0.05 范围内上下/左右平移、在 5 度范围内旋转、幅度为 10 的随机通道偏移。

程序小批量梯度下降计算误差关于权重的梯度，每一批的大小为 32 个样本。优化方法是用的是 Adam 和 SGD，初始学习率为 0.001，学习率随着训练过程进行衰减：

$$lrate = 0.0001 \times 0.3^{(1+epoch)/3}$$

程序会自动保存 val_loss 最优的 5 个模型权重，以及训练的日志。

程序设置了 Early Stop，监视 val_loss。当 val_loss 连续 10 个 epoch 没有改进时停止训练。

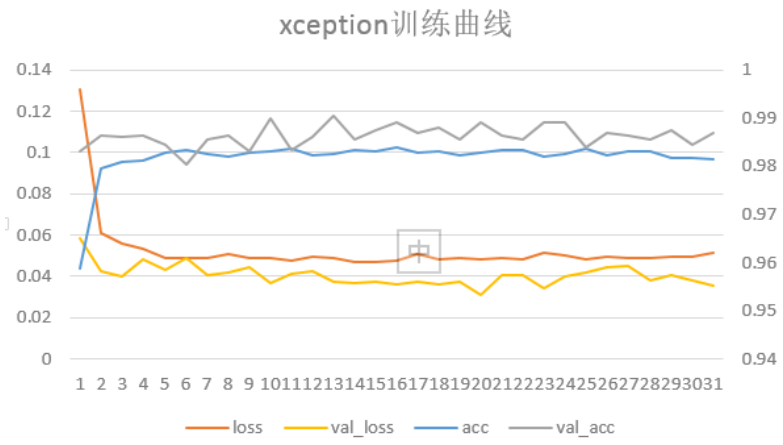


图 6 xception 模型训练曲线

3.2.1 结果的生成

程序批量读取测试集中的图片，并根据模型的不同进行相应的预处理。之后输入模型进行预测，并将预测结果按照格式要求写入到 csv 文件中。由于输出为预测为狗的概率，评价指标是对数损失，因此对输出进行截取有时会稍微改善最终的结果。

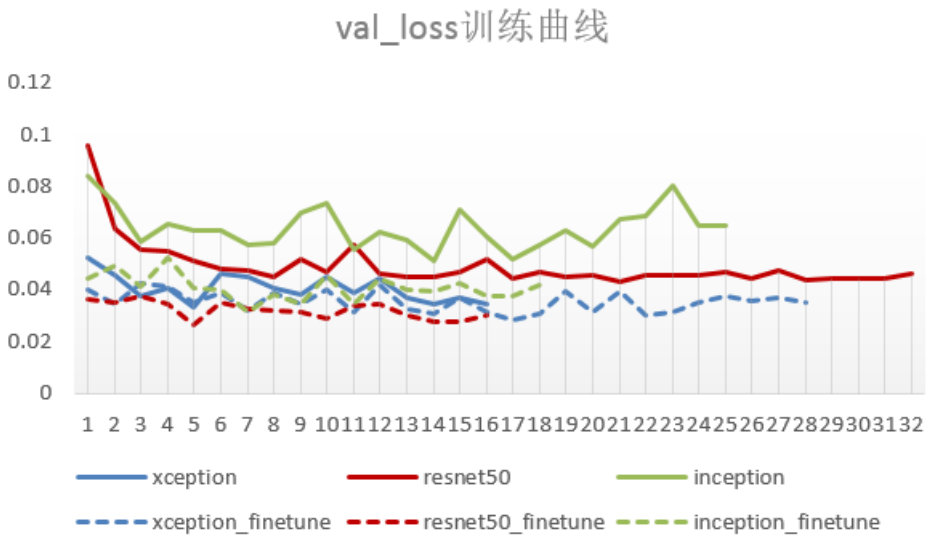


图 7 三个模型 val_loss 训练曲线比较

由图 7 可知，xception 模型的训练效果最好。使用这三个模型对测试集进行预测，并将结果上传至竞赛网站，获得的得分分别为：

- xception：0.05865　　xception finetune：0.05458
- resnet50：0.07193　　resnet50 finetune：0.05672
- inception：0.09765　　inception finetune：0.07739

3.3 改进

虽然从单一模型的角度，xception 取得了最好的效果，但是由于这三个网络的网络结构差异较大，对图像特征的提取也必然存在差异，因此，将这三个模型进行组合是下一步改进的方向。

由于评价指标使用的是对数损失，因此不能简单的做多模型平均或多模型投票，只能从特征层面进行组合以提高模型的分类水平。

由于在进行 Transfer-Learning 模型构建时，这三个模型的特征提取都是在 GlobalAveragePooling2D 层完成的，且特征数量均为 2048，可以针对这三个模型，提取所有训练集、验证集和测试集图像的特征，并将这些特征进行组合，之后重新设计分类器，得到多模型特征融合后的分类结果。

首先，训练集、验证集和测试集所有图片都输入到这三个模型，提取模型 avg_pool 层的输出，并保存为文件。同时，将数据对应的标签也保存到文件中。

接下来，将这三个模型的训练集、验证集和测试集特征在垂直方向分别进行堆叠，获得融合后的特征集合。

最后，构建一个简单的前馈神经网络，用于对融合好的特征进行分类。网络结构如下图所示，两个 Dropout 层的参数均为 0.5。

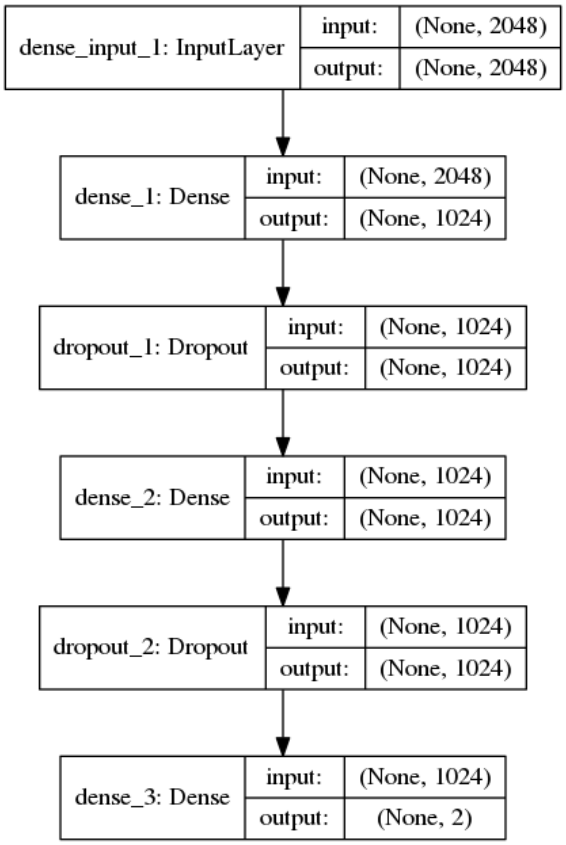


图 8 特征分类网络模型

网络使用 Adam 作为优化方法，学习率变化方式与之前一致。训练结果如下：

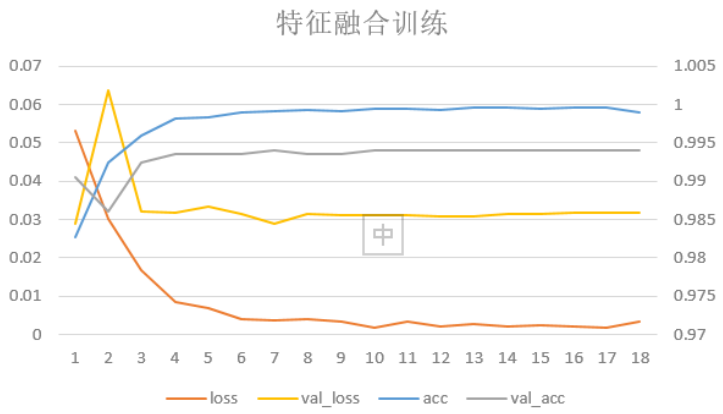


图 9 特征融合训练曲线

如图所示，得到的 val_loss 更小，但模型趋于过拟合。由于训练的 acc 已经十分接近于 1，且 val_acc 也已经非常高了，因此，在现有的条件和结构下，针对模型本身的优化空间已经不大了。

由于模型优化空间已经很有限了，在不使用其他数据的情况下，为了获得更好的分数，可以将全部数据投入到训练，以使模型学习到最多的特征。

将训练集和验证集在水平方向进行堆叠，形成新的训练集。投入训练时，监测目标改为 loss，其他策略不变。训练结果如下：

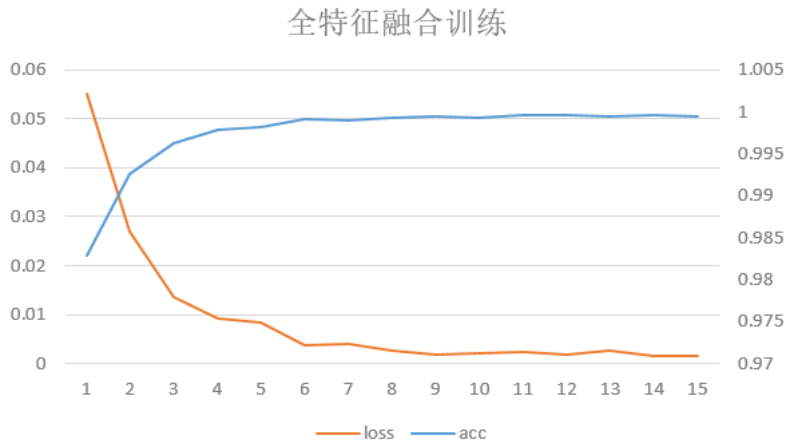


图 10 全特征融合训练曲线

将特征融合获得的结果上传至竞赛网站，得到的分数为 0.05000，相比单模型有 7.4%的提高。

这样的分数，目前在 Public Leaderboard 上排名第 61 位。

60	▼17	RichardDeurwaarder		0.04994	7	2mo
61	▲53	terrypang		0.05000	41	2h
62	new	MD79		0.05038	15	3h

图 11 Public Leaderboard 排名

4 结果

4.1 模型评价与验证

4.1.1 单模型选择

根据 Inception、ResNet50 和 xception 模型的定义可以发现，这三个网络都含有层间

的合并，不同的是：Inception 网络仅含有 concate 形式的 merge，ResNet50 仅含有 sum 形式的 merge，xception 两种形式的 merge 都有。同时，Inception 网络较浅，而 xception 和 ResNet50 网路较深。

由此可见，深度是网络能力的决定因素，利用 sum 形式的 merge，加深网络能够显著提高网络的能力。另外，网络的宽度对网络的能力也有提高的作用，但与深度相比只起次要作用。

4.1.2 特征组合策略

特征组合可以有很多种策略，受限于各方面资源，本项目仅尝试了最基本的特征组合方式。考虑到卷积的几何不变性，特征之间的相对位置也是含有信息的。如果不使用 GlobalAveragePooling2D 来提取特征，而是使用别的方式，就可以保留这样的位置信息，进而可以在特征组合后再次使用卷积神经网络进行分类。

另外，考虑到不同模型的能力不同，为不同模型的特征设置权重，或让能力较强的模型输出较多的特征，都是特征组合可以使用的策略。

4.2 结果分析

4.2.1 单模型结果

xception 模型得到 0.05458 的分数，显著优于 inception。相对 inception 的 0.07739 分，改进幅度为 29.1%。

ResNet50 模型的得分与 xception 模型十分接近。

xception 模型的分数可以在排行榜中排名第 88 位,ResNet50 模型的分数能排在 92 位，而 inception 模型的分数只能排在 226 位。

4.2.2 特征组合结果

特征组合相对单模型能够得到一些改善。相对单模型，特征提取后直接分类输出，特征组合还叠加了一个额外的分类网络，实际参与计算的网络层要大于单模型，因此，针对特征组合所获改善这一结果，有多少是来自于特征组合本身，有多少是来自于额外的分类网络，还是一个有待于研究的问题。

5 结论

猫狗大战是一个充满挑战的竞赛，利用有限的资源，寻找一条可行的技术路线，得到一个结果具有显著竞争力的模型是非常不容易的。CNN 是一个非常灵活的模型，围绕分类以及特征表示，有非常多的方法可以运用，使用 CNN 构建的计算机视觉分类模型是非常强大的。

另外，对于比赛，从竞争的角度，还有很多优化的方法。比如，测试集中的 2085 号图片，为狗的判别概率为 0.57。从分类的角度看，已经正确了，只是模型不够肯定。预测概率在 0.7 至 0.3 之间的图片共有 90 张，全部进行处理后，一定能够得到更好的结果。

图 12 测试集 2085 号图

还有一些图片即使是人工也无法判别其结果，比如测试集 5283 号图，存在这样的样本使得通过模型获得满分是不可能的了。

图 12 测试集 5283 号图

通过这个项目，参与到猫狗大战竞赛中来，大幅提高对 CNN 的理解及运用水平，是最大的收获。

参考文献

[1] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, Zbigniew Wojna. Rethinking the Inception Architecture for Computer Vision. arXiv:1512.00567

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

[3] François Chollet. Xception: Deep Learning with Depthwise Separable Convolutions. arXiv:1610.02357