基于图片内容的图片检索

邵钰杉
班级：软件 51
学号：2014011003
邮箱: sys14@mails.tsinghua.edu.cn

杨昕磊
班级：软件 51
学号：2015013197
邮箱: yangxinlei5@126.com

冯硕
班级：软件 51
学号：2015013210
邮箱: thss15_fengs@163.com

摘要

这次任务中，我们实现了使用人工神经网络对十分类的图片类型的判断。并实现了针对一张图片，由图片位于各分类的可能性进行对图片的 10-最近邻查询，返回图片库中国与查询图片最近似的 10 张图片。

类别描述

H.1[计算机视觉] 图片特征检索

H.2[数据与结构] 数据检索

一般术语

图片检索和搜索

关键词

人工神经网络、图像、特征、检索

1. 导论

我们小组先构造了不同的神经网络，并用这些神经网络对训练集图片进行训练，得到了可用于对图片进行 10 分类的神经网络。之后比较了不同神经的效果。最后选用效果最好的神经网络对图片获取 10 分类的可能性。最后用 10-最近邻查询来获得十张与查询图片最接近的图片。

2. 神经网络部分

2.1 公共框架

2.1.1 序贯模型（Sequential）

神经网络可以有非常复杂的结构，而其中最简单的模型，即网络的线性堆叠称为序贯模型，由于我们需要针对每一张图片做出一个类型的判断，即单输入对应单输出，故可以使用序贯模型来解决。

2.1.1.1 生成器（generator）

对输入的数据进行一定的处理，如进行随机偏移，旋转等操作，以减少过拟合的概率。

2.1.1.2 损失函数（loss）

用以估计预测的结果与真实值的不一致程度。

2.1.1.3 优化器（optimizer）

用以对机器学习模型中边的权值进行调整。

2.1.2 常用层（Core）

2.1.2.1 全连接层（Dense）

即最常用的全连接层，可以理解为一张完全二分图$K_{m,n}$

2.1.2.2 激活层（Activation）

为神经网络加入非线性因素，对一个层的输出施加激活函数。

2.1.2.3 退出层（Dropout）

为避免神经网络过拟合加入的层，实现方式为随机断开一些已经连接的神经元。

2.2 卷积神经网络框架（CNN）

2.2.1 卷积层（Convolutional）

2.2.1.1 二维卷积层（Conv2D）

二维卷积层，即对图像的空域卷积。该层对二维输入进行滑动窗卷积。

2.2.1.2 二维最大值池化层（MaxPooling2D）

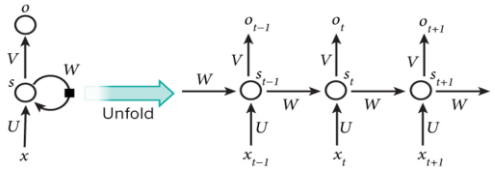
为空域信号施加最大值池化

2.3 循环神经网络框架（RNN）

2.3.1 循环层（Recurrent）

2.3.1.1 简单循环网络层（SimpleRNN）

全连接 RNN 网络，RNN 的输出会被回馈到输入。



2.4 神经网络实现及实验

2.4.1 实验环境：

操作系统：CentOS Linux release 7.3.1611 (Core)

内存：128.0GB

中央处理器：Intel(R) Xeon(R) CPU E5-2630 v4@2.20GHz*10

图形处理器：NVIDIA GeForce GTX1080

2.4.2 卷积神经网络实现

2.4.2.1 设计架构

本实验中我们组使用了 CNN 神经网络进行图片分类。使用的模型是蓄罐模型由此建立神经网络，每一层神经网络里面加入卷积层 Conv2D(32, (3, 3))，加入激活层 Activation('relu')，加入迟化层 MaxPooling2D(pool_size=(2, 2))，去除冗余数据 Dropout(0.2)。这是这是完整的一层神经网络。将神经网络输出传入到分类器中。首先将原来的神经网络数据降维后输入全连接层，最后使用激活层进行输出 Activation('softmax')。

2.4.2.2 训练模型

在训练模型过程中，使用图片生成器 train_generator 给模型提供训练集提供输入。train_generator 会将训练集中的图片随机进行旋转切片等变化，使得模型见过足够多的图片。

验证集的图片与训练集图片不相同，并且每一次验证时也会将验证集合图片进行随机旋转切片等变换，以防止模型因为验证集与训练集样本过小导致的过拟合。

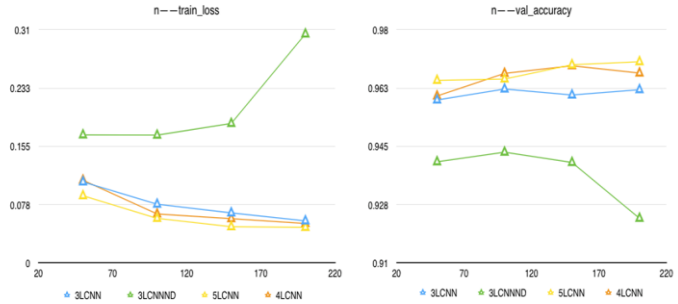
2.4.3 循环神经网络实现

在我们实现的对图片进行分类的 RNN 有如下结构。

在 RNN 的实现中，由于 simpleRNN 层只接受一维的向量，故在这里我们不再使用图片生成器，在这里我们使用 numpy 读取所有图片，接着用 numpy 的 reshape 将图片转化为一维，最后输入模型。

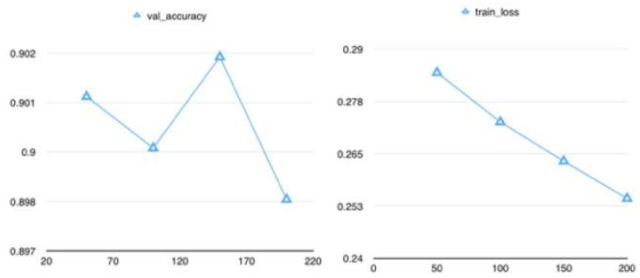
2.4.4 实验结果

模型	三层神经网络 CNN		三层神经网络 CNN（未 DropOut）		四层神经网络 CNN		五层神经网络 CNN	
迭代次数 n	val_acc	val_loss	val_acc	val_loss	val_acc	val_loss	val_acc	val_loss
50	0.9589	0.1216	0.9403	0.1882	0.9601	0.1146	0.9648	0.0968
100	0.9622	0.1278	0.9432	0.1873	0.9669	0.1048	0.9652	0.1042
150	0.9604	0.1353	0.9401	0.1854	0.9692	0.1131	0.9695	0.1050
200	0.9620	0.1422	0.9233	0.3069	0.9670	0.1261	0.9704	0.0942
迭代次数 n	train_acc	train_loss	train_acc	train_loss	train_accuracy	train_loss	train_accuracy	train_loss
50	0.9583	0.1074	0.9380	0.1700	0.9590	0.1096	0.9684	0.0890
100	0.9711	0.0777	0.9415	0.1697	0.9749	0.0647	0.9778	0.0588
150	0.9760	0.0661	0.9403	0.1853	0.9780	0.0584	0.9824	0.0477
200	0.9785	0.0555	0.9194	0.3045	0.9807	0.0520	0.9829	0.0468



cnn 实现结果

模型	循环神经网络RNN	
迭代次数n	val_accuracy	val_loss
50	0.9009	0.2941
100	0.8996	0.294
150	0.9019	0.2875
200	0.8983	0.2864
迭代次数n	train_accuracy	train_loss
50	0.9018	0.2844
100	0.9037	0.2726
150	0.9065	2632
200	0.9083	0.2543



rnn 实验结果

在上述的实验探究后，我们发现在其他条件相同的情况下，增加神经网络的层数可以有效地提升神经网络的最终分类的正确率。而在神经网络层数相同的情况之下，没有对网络进行简化（即没有 dropout）会使得网络过拟合。

在我们的实现中，卷积神经网络的结果要好于循环神经网络，故我们采用效果最好的 2 层卷积+5 层感知机的模型继续下面的实验。

3. 图片检索部分

3.1 最近邻检索

3.1.1 概述

对于每一个图片，得到一个 10 维向量，表示属于每个种类的概率。根据欧式距离，得到最近邻

3.1.2 实现方法

实现最近邻搜索，并没有用 rtree 或者 kdtree，而是用了 balltree。与 kdtree 和 rtree 不同的是，kdtree 和 rtree 用矩形分隔，balltree 用超球面分隔。

使用 balltree 的 query()方法得到最近邻。

3.2 GUI 及使用方法

3.2.1 实现方式

由于神经网络部分由 keras 实现，为了使用方便以及防止跨代码的调用，我们决定也使用 Python 来实现 GUI。

最终 GUI 由 pyqt 实现。

3.2.2 使用说明及效果图

运行 Pro2Gui.py 文件，即可得到如下图形界面。在搜索栏中输入 test 文件下中图片的完整名称，点击 search，即可得到 10 个和它最相近的图片。

效果图如下：