我的AI年---深度学习在图片处理中的应用

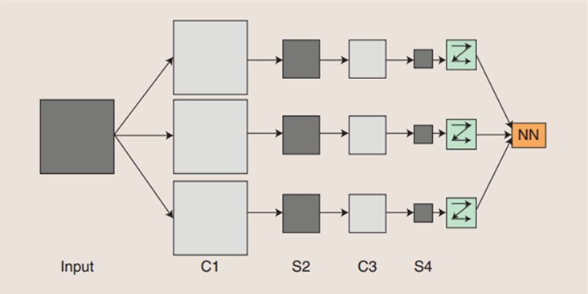
随着阿法狗在围棋方面的优秀表现，这一年中机器学习又迎来一个新的热潮。周末难得有时间静下心来写写文章，权当一个心得体会分享，有兴趣的同志们可以一起探讨。机器学习本来是个“古老”的课题了，早在上世纪七八十年代就有很多优秀的研究成果，只不过并未引起如此重视，更没有如今的如此多的实际应用。粗略的来讲，机器学习可以分为监督学习、无监督学习、半监督学习，也可分为浅度学习、深度学习等等，划分的标准五花八门，但是总体来讲目前的机器学习还是属于浅层次学习，只是停留在表象的识别，分类，聚类，预测等等领域，至于真正深层次的逻辑学习，推理等等还是几乎空白的。在各种学科中机器学习是极少见的实践走在理论前面的学科之一，它的复杂性和强大的数学依赖性让很多想自己研究的人望而却步，从入门到放弃。本文只介绍深度学习在图片处理和识别中的实践应用，后面的例子拿腾讯的几种常用验证码来做个实验。本文借鉴谷歌在街景识别中的一些理论依据来做。建议大家阅读谷歌在街景识别方面的论文（Ian.J Goodfellow, Yaroslav Bulatov, Vinay Shet等人的论文）或者斯坦福的机器学习公开课。

下面我们不发散思维，只讲图片的处理和识别。

在图片处理中，有一些很成熟的应用，比如说车牌号识别，人脸识别等等，车牌号识别其实很简单，大家都知道它就是由0到9字母a到z再加上数量有限的那几个省份的汉字组成，人脸识别和一些街景识别验证码的识别就稍微有点难度了，因为它是无规则的，至少在形式上样式上我们看起来是没什么规律的，就是同一个字母鬼知道下一个图片中它会是什么样的形态出现，大家感受下：



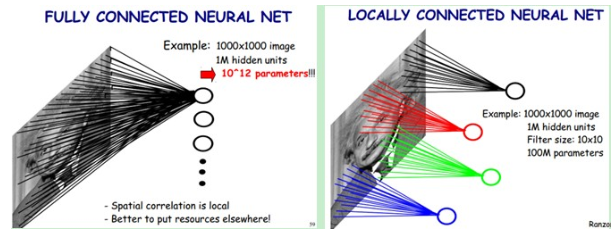
是不是挺难识别出是什么来的，但是人的眼睛一看就知道是什么了，为什么呢，因为每个字母尽管形态千变万化、大小千变万化、出现角度无规律，但并不代表它没有特征，比如9再变形也不会变成1，它下面总会上面一个圈圈带着一笔勾而不是像1一样杵在那里。这就是特征。只要有特征我们就能搞定它！这种情况我们恐怕不能使用像车牌识别之类的浅层学习中的一些处理方式了，使用SVM、BP、最大熵算法都是搞不定，因为这些常用的算法是基于样本统计的神经网络。这类 图片的处理当然也有很多优秀的深度算法，但是目前的实践来看，最好的莫过于CNN神经网络，使用多层卷积的神经网络来学习图片的特征，逐渐建立模型，这就是谷歌在街景识别中的核心思路。



如上图所示，C层为特征提取层，每个神经元的输入与前一层的局部感受野相连，并提取该局部的特征，一旦该局部特征被提取后，它与其他特征间的位置关系也随之确定下来；S层是特征映射层，网络的每个计算层由多个特征映射组成，每个特征映射为一个平面，平面上所有神经元的权值相等。特征映射结构采用影响函数核小的sigmoid函数作为卷积网络的激活函数，使得特征映射具有位移不变性；

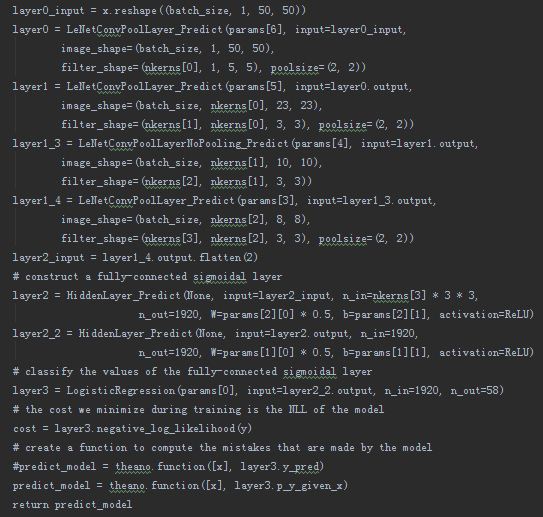
正常情况下要根据自己的图片特征和精准度要求来控制要做几层的，Google的街景识别中是使用11层，但是11层的计算量是非常大的，当时谷歌用一万个节点还要跑近一个月来跑出模型来（样本也很大），我们做实验就用个六七层就可以了。

CNN最牛逼的地方有两个：局部感受野和权值共享。通俗点说，什么叫局部感受野，就是你看到一栋大楼时其实没有必要从最上面楼顶打量到最下面才意识到这是栋大楼，而是只看到了一个角或者其中一部分基本就可以确定这是一栋楼了；权值共享，其实是在做特征映射的时候用的，也就是用公用的一个权值来映射整张图片的一个特征，然后不停变换这个共享的权值来提取不同的特征。贴一张网上的图片：



如果不使用局部感受野和权值共享的话计算量是相当惊人的，比如上图1000\*1000的图片，如果不使用局部感受野的话每个神经元要和每个像素连接，那就是1000000\*1000000，如果用局部感受野，假设每个像素只和10\*10的神经元连接那就是1000000\*100，计算量降低了一万倍。可是这时候发现计算量级还是蛮高的，那就是权值共享起作用了，感受野是10\*10的时候每个神经元对应的是100个参数，总共有1000000个神经元，如果把100个参数暂且让它相等，参数的数目又降低到了100。其实这就是一个卷积的过程！数学真是一个很伟大的东西！ 什么是卷积，就是不同时间或者空间的因素对某个结果的贡献大小的一个累加嘛，我们的数学课本上大多把这些东西讲的很复杂，一上来就一个晦涩难懂的公式，然后又试图用貌似最严谨的语言来描述，导致的结果就是很多人看不懂或者没兴趣看了。再比如什么叫傅里叶变换，我们去翻书时很多都是罗列定义然后再推导公式，其实很简单一句话就可以解释傅里叶变换，就是换个角度看问题嘛！比如一个很复杂的曲线，你无法看出应该用什么公式来描述时，换个角度去看会发现其实它就是一个什么sin cos加个其他什么已知的曲线的累积的结果而已。

理论基础有了，再来实践一把吧。我们暂且使用6层，当然处理过程可能比较慢，这时候可以用cuda来加速处理，也就是把图片处理移到GPU上去处理（cuda这又是一个展开来讲能讲好多的东西，且不去追究，姑且就认为是一种能加速图片处理的方法好了）：

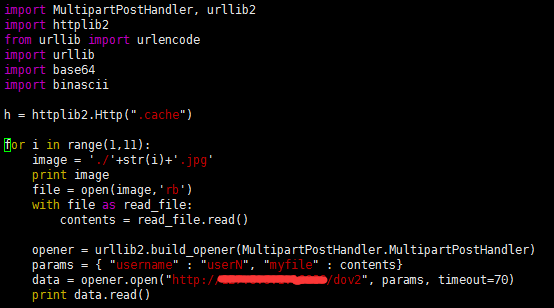


具体过程由于篇幅原因就不罗列了，下面直接看下效果如何。

把训练好的模型部署到一个server上，然后写一个Python进程来接受处理图片然后交给这个模型去处理，然后把模型的结果返回去。

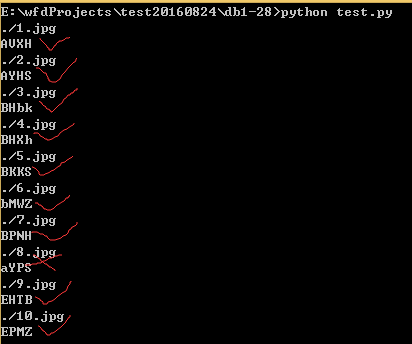
我们写个测试脚本，把要测试的图片放到一个目录下

test.py



1、

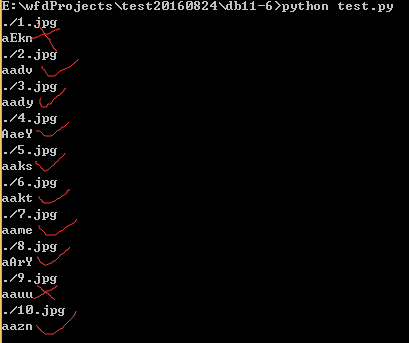




可以看到大部分能识别正确，第八张识别错了；

2、

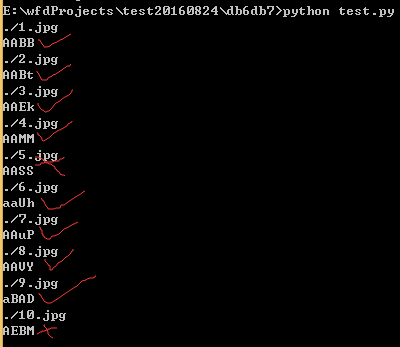




这种有比较重的背景，也是可以大部分识别正确，只有一张错了；

3、





这种情况更复杂了，各种类型的掺杂在一起，这时候处理也还是可以大部分识别正确只错了两个。

图片处理其实是机器学习最普通的一个应用而已，像视频鉴黄、语音识别等等也大都异曲同工。这方面的处理原理其实不是很复杂，难就难在如何提高正确率，如何降低运算成本，有兴趣的童鞋们欢迎一起研究探讨。希望能在不久的将来看到真正牛逼的机器学习的成果能越来越多的出现在大家的生活中。