Python徒手实现识别手写数字——大纲

写在前面

其实我之前写过一个简单的识别手写数字的程序，但是因为逻辑比较简单，而且要求比较严苛，是在50x50大小像素的白底图上手写黑色数字，并且给的训练材料也不够多，导致准确率只能五五开。所以这一次准备写一个加强升级版的，借此来提升我对Python处理文件与图片的能力。

这次准备加强难度：

1. 被识别图片可以是任意大小；
2. 不一定是白底图，只要数字颜色是黑色，周围环境是浅色就行；
3. 加强识别手写数字的逻辑，提升准确率。

因为我还没开始正式写，并且最近专业课程学习也比较紧迫，所以可能更新的比较慢。不过放心，代码质量肯定是不会下降的，我会尽我所能写的逻辑明确、通俗易懂点。

所以这次面向的人群是拥有一定Python基础，对数学算发有一点了解（识别图像的算法嘛）的人。

但毕竟我不是专业的，也没有看那么多论文，所以我这里运用的算法仅仅是我一个粗浅的想法，只是为了练手而已。如果和实际应用脱节，还望莫怪。

当然，如果诸位有什么比较好的想法，可以在下方评论或者私信我，我们可以探讨一下，相互进步。

整体思路

大纲

对图片的预处理

在最开始的时候，我们假设只拥有一个训练库，里面是从0到9的手写数字图案若干组。

所以我们首先应该将这些图案读入程序中，然后运用**某种方式**保存好，用来后面识别图片。

这里的图案我们假设是大小不一的，里面手写的数字也是有大有小。所以我们可以将**包住手写数字图案的最小矩形**给裁剪出来，然后将裁剪出来的图案统一给拉伸成相同大小的图案。

以上操作得出一个矩阵，这个矩阵的值是图案的灰度值。对于训练用的图片和被检测的图片我们都是这样处理。

图像识别的算法处理

我这里想用两个方法来让数字识别准确点：

1. 识别所写数字的“洞数”；
2. 将图片转为1xn的向量，然后根据根据训练图片分出的类对被识别图片图片进行分类。

洞数就是某个数字是否有闭合的曲线，比如说7没有洞，6有一个洞，8有两个洞。所以我们根据洞数可以分成以下三类

- 0洞：1, 2, 3, 4, 5, 7
- 1洞：6, 9, 0
- 2洞：8

但是因为各种手写差异，比如说6, 9, 8之类的没有闭合，4上面闭合，所以会导致下面这种可能情况

- 0洞：1, 2, 3, 4, 5, 6, 7, 9
- 1洞：6, 8, 9, 0
- 2洞：8

虽然说这样分类0洞占大多数，但是聊胜于无。

对于将图片转为向量的意思就是将图片原本的二维矩阵展开称为一维向量。这个用numpy的函数可以可以很简单的实现。

对于这个分类，下面我就简单的讲一下原理。

假设我们在二维平面上有两个点$A=(1,1)$和$B=(5,5)$，我现在再放一个点$C=(2,2)$，那么请问，C点离哪一个更近？

学过初中数学的都会知道肯定是离A点更近。所以我们换一种说法，我们现在有两个类A和B，A类中包括了点$(1,1)$,B类中包括了点$(5,5)$，所以对于点$(2,2)$，它**可能**属于哪一类？

因为这个点离A类的点更近一点，所以它可能属于A类。这就是结论。那么对于3维空间，A类是点$(1,1,1)$和B类是$(5,5,5)$，那么对于点$(2,2,2)$肯定也是属于A类。

可以看出，我们这里是将**两个点的距离**来作为判断属于哪一类的标准。那么对于我们将图片拉成的1xn维向量，他实际上投影到n维空间上就是一个点，所以我们将训练向量分成10类，分别代表十个数字，那么被识别数字靠近哪一个类，那说明它有可能属于这一个类。

那么我们这里可以假设对于被识别向量，列出距离他最近的前十个向量分别属于哪一类别，然后根据名次加上一个权重，并计算出一个值。该值代表了可能是属于哪一个类，因此这就是我们得出的最终的一个结果——被识别手写数字图片的值。

难点

保存已训练图片的向量。这一条我想就直接保存在csv文件中，每一次运算时先判断是否有新的训练图片加入，如果有，则把新的图片向量也存入csv文件中。若没有，则直接读取所有向量保存在一个大矩阵中用于计算。

将手写数字从背景中分离。因为我这里令手写数字为黑色（灰度值为0），其他背景色尽量为，所以就令灰度值大于某个界限（如50）的点全部为255（白色），其余不变。这样子只要非255，那就是手写数字的点。

识别手写数字的洞。这个有算法，搞过程序设计竞赛的应该会了解。具体我就不细讲了，大概就是利用递归之类的去寻找。

求向量距离。这个更简单了，求解每一个训练向量与识别向量的距离就行，只不过当训练向量比较大的时候可能比较慢。

总结

以上就是全部思路，如果诸位有更好的想法，欢迎评论/私信我，让我们一起相互学习进步，谢谢。

如果你喜欢的话，请点个喜欢哦~