Opencv 手写数字识别（Opencv3.0+KNN+HOG特征）

# 写在前面

最近在学习Opencv，本人android开发狗，对Opencv纯属兴趣。一个破本科毕业的我，发现智商完全不够用，书到用是方恨少，都怪自己数学太渣。好在Opencv封装得比较好，如果只是使用的话，大概知道原理就知道该怎么用。经过学习总结，写了一个小Demo，一方面是自己做的笔记，另一方面也可以给**初学者**做一个参考，让大家少走一些弯路。有些东西本来很容易理解，但是太过于概念化反而不好理解，吃了多次亏（可能是自己太渣:-）。其实这个Demo的原理也适用于任何文字手写识别，只需要修改样本库就可以了。

# 开发环境

Opencv 3.0

Xcode 7.3

Opencv 跨平台，如果不是用的Mac，自己去搜一下环境配置，网上很多资料，这里不细说。如果你使用的Mac 我这里有一些老外的简洁配置视频，文末和所有代码分享给大家。

# 用到的理论知识

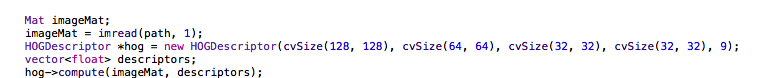
1. Opencv的Mat基本操作：很简单啦，就是图片读取roi拷贝等。
2. KNN算法：K最近邻(kNN，k-NearestNeighbor)分类算法，一个简单有效的分类算法，看了多个大牛写的文章，很多人觉得KNN精确度相对较好，我觉得重要的是简单有效。不懂的话，大家百度一下，我相信你学过向量，基本上就能明白KNN算法。
3. HOG 特征：方向梯度直方图(Histogram of Oriented Gradient, HOG)特征，这个需要大家有一定基础知识，采集过程有点复杂，实话是说我也并没有完全看懂，但我知道了大概原理。但是网上很多文章介绍这个东东。  
    这里我想说一些我的看法：图像识别的核心技术我认为是图像的描述特征，而不是高大上的机器学习算法，什么是图像特征?抛开很多乱七八糟的数学符号不说,特征嘛，唯一标示一个对象的东西了。只不过图像特征在Opencv里多是用向量来描述的（说到底就是用一个多维向量来确定一张图片）。当然其实也可以用任何格式来描述，也可以创造自己的特征算法，只有能行之有效。无所谓是用向量啊，函数啊，点集啊，只要能唯一描述一张图片或者一类图片就行。你甚至可以直接使用图片的所有像素RGB值组成一个向量来作为图像特征。所以我的建议是当提到图像特征时不要拘泥于书中的算法，完全可以大开脑洞，试着去设计自己的特征算法。说不定比某位前辈的算法更精确高效。  
    当然了，本文中使用的就是HOG特征，因为看了别人的文章觉得HOG特征在这种应用场合效果应该还不错，HOG特征最成功的地方是行人检测，大家可以去网上搜一下相关资料。

# 手写数字识别过程介绍

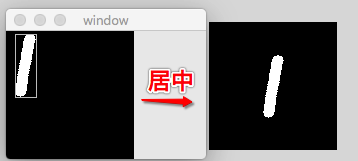
第一步:

当然是数据采集啦，我们根据已有的数据和分类去确定未知的数据属于哪一类，专业名词叫“监督学习”，去它的专业名字，我只是个学渣，你给我念八股文☺。

为了方便先贴出一段代码:

这就是Opencv3.0 采集图像Hog特征的代码,正如我前面所说，Opencv里面用一个向量来标示一张图片，至于这个向量怎么得出来的，就是算法的事情了。如果你感兴趣可以深入研究。这段代码的意思是从path这个路径的图片采集出它的HOG特征，descriptors这个vector存储的就是path这个路径下的图片的特征。换言之，现在就是用descriptors来标示这张图片了，在以后的图片匹配啊，计算啊，就用这个vector了，基本上和这个图像没关系了。在这个Demo项目中，样本和测试样本都是128X128的，在实际应用中可能样本和测试样本大小不一样，但要想办法优化原图之后进行归一化，至于HOGDescriptor参数为什么要这样设置，大家去网上了解一下HOG特征原理就知道了，这里我设置的参数是为了快速和准确而取的一个折衷。按照这个参数采集的最终特征值是384个,也就是说，现在这张图片要被这个长度为384的vector来代替。

值得注意的是，手写的时候可能图像并不能居中，这样可能会增加误差，在本Demo中通过轮廓寻找把内容重新拷贝到手写去中间，来减少误差。原理如下图:



第二步：

使用Opencv自带的机器学习算法API，组装数据，



调用Opencv自带的KNN 分类API，这里设定K值大小:1=<k<=10（因为我们这里是0-9共10个数字，相当于10个分类）,最好不要设置成1 ，有时候因为样本太少可能会导致误差太大,稍微设置大一点，可以减少误差。这里的dataMat就是采集到的所有HOG特征，labelMat 就是每一行对应代表的是那个值.画一个草图就像这样:



第三步:

通过以上的步骤，现在我们就可以预测或者说是分类识别了:



这里我使用10张测试图片，命名是0-9.png,一来方便读取而来方便对比结果,predict 就是预测分类，注意这里的去预测这个图片时候不是调用图片的Mat而是这个图像的HOG特征，没想到我仅仅用了对每个手写数字用了20个样本，使用了完全不同的10个样本来测试，竟然都识别出来了.效果还是蛮不错的。

最终结果：按‘C’键清除画板，按’S’键识别

