

# 常见验证码的弱点与验证码识别

insight-labs (/author/insight-labs) · 2013/06/08 11:36

## 0x00 简介

验证码作为一种辅助安全手段在Web安全中有着特殊的地位，验证码安全和web应用中的众多漏洞相比似乎微不足道，但是千里之堤毁于蚁穴，有些时候如果能绕过验证码，则可以把手动变为自动，对于Web安全检测有很大的帮助。

全自动区分计算机和人类的图灵测试（英语：**Completely Automated Public Turing test to tell Computers and Humans Apart**，简称**CAPTCHA**），俗称验证码，是一种区分用户是计算机和人的公共全自动程序。在**CAPTCHA**测试中，作为服务器的计算机会自动生成一个问题由用户来解答。这个问题可以由计算机生成并评判，但是必须只有人类才能解答。由于计算机无法解答**CAPTCHA**的问题，所以回答出问题的用户就可以被认为是人类。(from wikipedia)

大部分验证码的设计者都不知道为什么要用到验证码，或者对于如何检验验证码的强度没有任何概念。大多数验证码在实现的时候只是把文字印到背景稍微复杂点的图片上就完事了，程序员没有从根本了解验证码的设计理念。

验证码的形式多种多样，先介绍最简单的纯文本验证码。

## 纯文本验证码

纯文本，输出具有固定格式，数量有限，例如：

- 1+1=?
- 本论坛的域名是?
- 今天是星期几?
- 复杂点的数学运算

这种验证码并不符合验证码的定义，因为只有自动生成的问题才能用做验证码，这种文字验证码都是从题库里选择出来的，数量有限。破解方式也很简单，多刷新几次，建立题库和对应的答案，用正则从网页里抓取问题，寻找匹配的答案后破解。也有些用随机生成的数学公式，比如 随机数 [+-\*/] 随机运算符 随机数=?，小学生水平的程序员也可以搞定.....

这种验证码也不是一无是处，对于很多见到表单就来一发的spam bot来说，实在没必要单独为了一个网站下那么大功夫。对于铁了心要在你的网站大量灌水的人，这种验证码和没有一样。

下面讲的是验证码中的重点，图形验证码。

## 图形验证码

先来说一下基础：

识别图形验证码可以说是计算机科学里的一项重要课题，涉及到计算机图形学，机器学习，机器视觉，人工智能等等高深领域.....

简单地讲，计算机图形学的主要研究内容就是研究如何在计算机中表示图形、以及利用计算机进行图形的计算、处理和显示的相关原理与算法。图形通常由点、线、面、体等几何元素和灰度、色彩、线型、线宽等非几何属性组成。计算机涉及到的几何图形处理一般有 2维到n维图形处理，边界区分，面积计算，体积计算，扭曲变形校正。对于颜色则有色彩空间的计算与转换，图形上色，阴影，色差处理等等。

在破解验证码中需要用到的知识一般是 像素，线，面等基本2维图形元素的处理和色差分析。常见工具为：

- 支持向量机(SVM)
- OpenCV
- 图像处理软件(Photoshop,Gimp...)
- Python Image Library



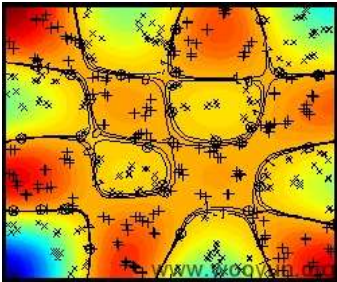
(/author/insight-labs)  
insight-labs  
(/author/insight-labs)

支持向量机SVM是一个机器学习领域里常用到的分类器，可以对图形进行边界区分，不过需要的背景知识太高深。

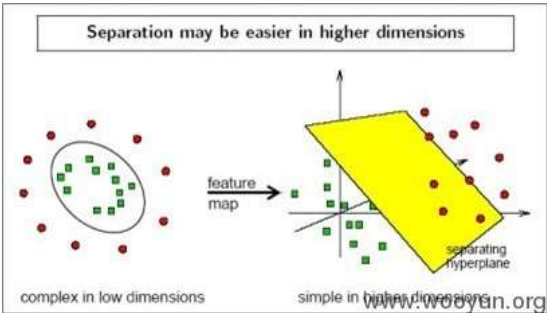
OpenCV是一个很常用的计算机图像处理和机器视觉库，一般用于人脸识别，跟踪移动物体等等，对这方面有兴趣的可以研究一下

PS,GIMP就不说了，说多了都是泪啊.....

Python Image Library是pyhon里面带的一个图形处理库，功能比较强大，是我们的首选。

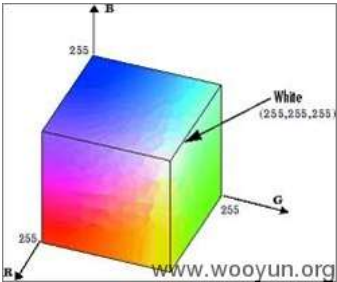


SVM图像边界区分

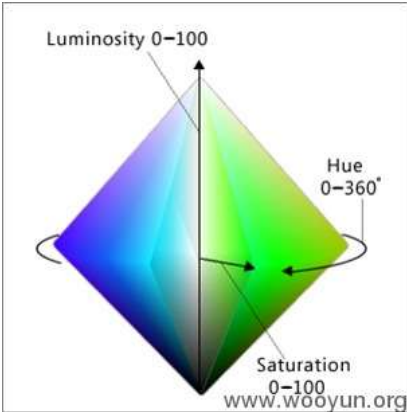


SVM原理，把数据映射到高维空间，然后寻找能够分割的超平面

识别验证码需要充分利用图片中的信息，才能把验证码的文字和背景部分分离，一张典型的jpeg图片，每个像素都可以放在一个5维的空间里，这5个维度分别是，X,Y,R,G,B，也就是像素的坐标和颜色，在计算机图形学中，有很多种色彩空间，最常用的比如RGB，印刷用的CYMK，还有比较少见的HSL或者HSV，每种色彩空间的维度都不一样，但是可以通过公式互相转换。



RGB色彩空间构成的立方体，每个维度代表一种颜色



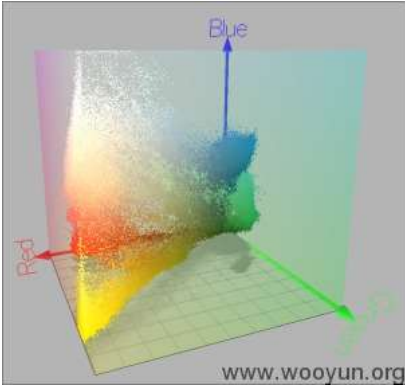
HSL（色相饱和度）色彩空间构成的锥体，可以参考：

<https://zh.wikipedia.org/wiki/HSL%E5%92%8CHSV%E8%89%B2%E5%BD%A9%E7%A9%BA%E9%97%B4>  
(<https://zh.wikipedia.org/wiki/HSL%E5%92%8CHSV%E8%89%B2%E5%BD%A9%E7%A9%BA%E9%97%B4>)

了解到色彩空间的原理，就可以用在该空间适用的公式来进行像素的色差判断，比如RGB空间里判断两个点的色差可以用3维空间中两坐标求距离的公式：

$$\text{distance}=\text{sqrt}[(r1-r2)^2+(g1-g2)^2+(b1-b2)^2]$$

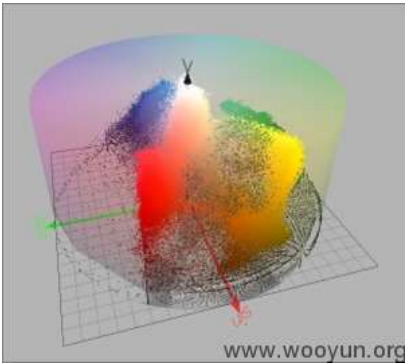
更加直观的图片，大家感受一下：



随便把一张图片的每个像素都映射到RGB色彩空间里就能获得一个这样的立方体。

通过对像素颜色进行统计和区分，可以获得图片的颜色分布，在验证码中，一般来说使用近似颜色最多的像素都是背景，最少的一般为干扰点，干扰线和需要识别文字本身。

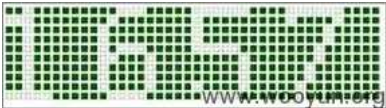
对于在RGB空间中不好区分颜色，可以把色彩空间转换为HSV或HSL：



## 0x01 验证码识别的原理和过程

### 第一步： 二值化

所谓二值化就是把不需要的信息通通去除，比如背景，干扰线，干扰像素等等，只剩下需要识别的文字，让图片变成2进制点阵。



### 第二步： 文字分割

为了能识别出字符，需要对要识别的文字图图片进行分割，把每个字符作为单独的一个图片看待。



### 第三步： 标准化

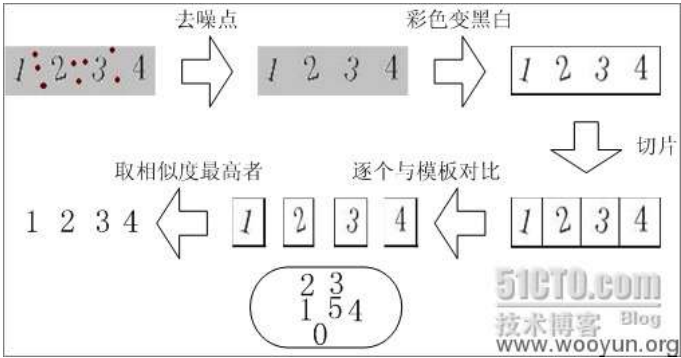
对于部分特殊的验证码，需要对分割后的图片进行标准化处理，也就是说尽量把每个相同的字符都变成一样的格式，减少随机的程度

最简单的比如旋转还原，复杂点的比如扭曲还原等等

第四步：识别

这一步可以用很多种方法，最简单的就是模板对比，对每个出现过的字符进行处理后把点阵变成字符串，标明是什么字符后，通过字符串对比来判断相似度。

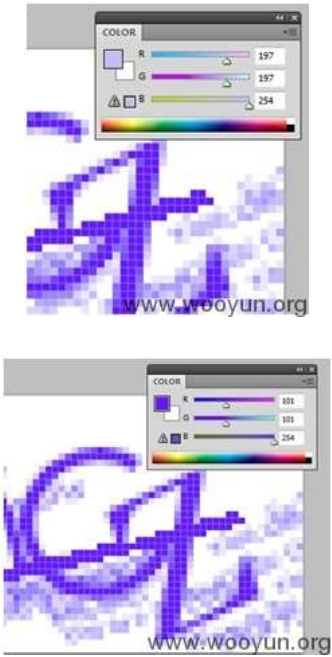
在文章的后半部分会详细解释每步的各种算法



二值化算法

对于大部分彩色验证码，通过判断色差和像素分布都能准确的把文字和背景分离出来，通过PS等工具把图片打开，用RGB探针针对文字和背景图的颜色分别测试，在测试多张图片后，很容易可以发现文字和背景图的RGB差距总是大于一个固定的阈值，即使每次图片的文字和背景颜色都会变化，比如：

新浪和discuz的验证码



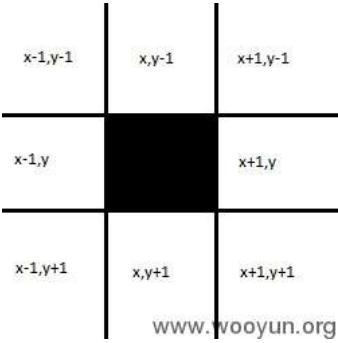
通过对文字部分和干扰部分取样可以发现，文字部分的R、G值一般在100左右，B值接近255，但是背景干扰的R、G值则大大高于文字部分，接近200，比较接近文字轮廓部分的像素的RG值也在150以上。通过程序遍历一遍像素就可以完全去掉背景。



Discuz的验证码同理

对于一些和文字颜色相同但是较为分散和单一的干扰像素点，我们可以用判断相邻像素的方法，对于每个点判断该点和相邻8个点的色差，若色差大于某个值，则+1，如果周围有超过6个点的色差都比较大，说明这个点是噪点。对于图像边界的一圈像素，周围没有8个像素，则统统清除，反正文字都在图片的中间位置。

如下图：假如当前像素的坐标是x,y 图形坐标系的原点是图像的左上角



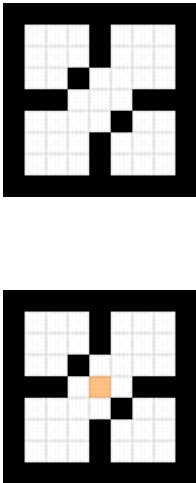
干扰线对于识别验证码增加了一些难度，不过干扰线只有很小的几率会以大角度曲线的方式出现，大部分时间还是小角度直线，去除算法可以参考<http://wenku.baidu.com/view/63bac64f2b160b4e767fcfed.html> (<http://wenku.baidu.com/view/63bac64f2b160b4e767fcfed.html>)

对于1个像素粗细的干扰线，在字符为2个像素以上的时候，可以用去噪点算法作为滤镜，多执行几次，就可以完美的把细干扰线去掉。

对于像素数比干扰点稍大的干扰色块，可以采用的算法有：

### 油漆桶算法（又叫种子填充算法，Floodfill）

种子填充算法可以方便的计算出任意色块的面积，对于没有粘连字符或者粘连但是字符每个颜色不一样的验证码来说，去除干扰色块的效果很好，你只需要大概计算一下最小的和最大的字符平均占多少像素，然后把这段区间之外像素数的色块排除掉即可。



上下左右4个方向填充还有8个方向填充的不同

判断颜色分布：

对于大多数彩色验证码来说，文字基本在图片中心的位置，每个字符本身的颜色是一样的，也就是说对于文字来说，同一种颜色基本都集中在一个固定的区域范围内，通过统计图片中的像素，按近似颜色分组，同时分析每个颜色组在图片中的分布范围，假如说有一种颜色大部分像素都在图片边缘，那么这个颜色肯定不属于要识别的字符，可以去掉。

对于干扰线，并没有一种十分有效的方式能完全去除并且不影响到文字，不过如果能够成功分割字符的话，少量干扰线对于识别率影响不大。

### 字符分割算法

破解验证码的重点和难点就在于能否成功分割字符，这一点也是机器视觉里的一道难题，对物件的识别能力。对于颜色相同又完全粘连的字符，比如google的验证码，目前是没法做到5%以上的识别率的。不过google的验证码基本上人类也只有30%的识别率

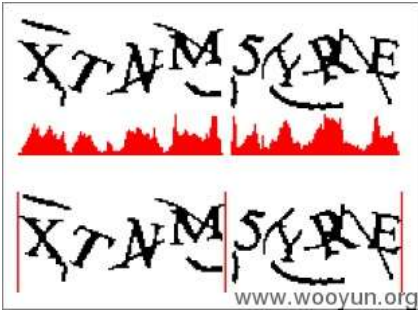
对于字符之间完全没有粘连的验证码，比如这个-> \_->



分割起来是非常的容易，用最基本的扫描线法就可以分割，比如从最左侧开始从上到下（y=0---|||y=n）扫描，如果没有遇到任何文字的像素，就则往右一个像素然后再扫描，如果遇到有文字像素存在，就记录当前横坐标,继续向右扫，突然没有文字像素的时候，就说明到了两个字符直接的

空白部分，重复这个步骤再横向扫描就能找到每个字符最边缘4个像素的位置，然后可以用PIL内建的crop功能把单独的字符抠出来。

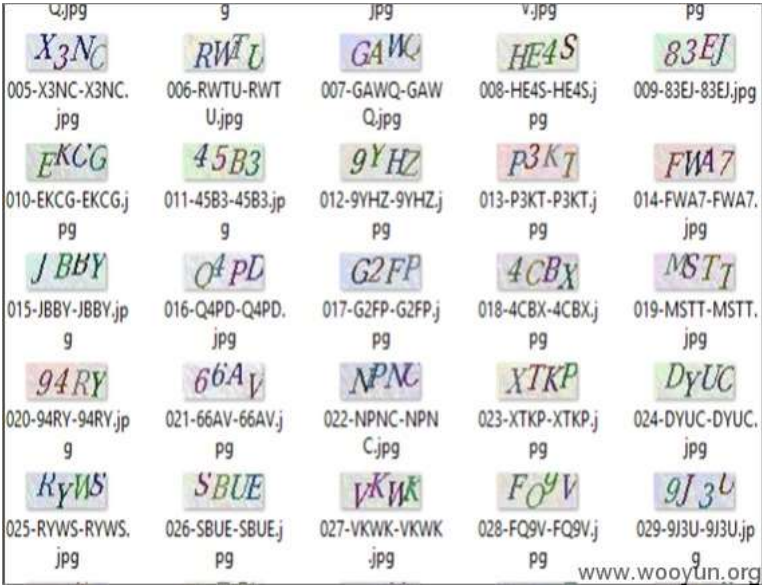
对于有少许粘连但是只是在字符边角的地方重叠几个像素的验证码，可以用垂直像素直方图的统计方法分割。如下图：



图上半部分是垂直像素直方图的一种直观展示，假如图片宽度为100像素，则把图片切割为100个1像素的竖线，下面的红色部分为当前x坐标上所有黑色像素的总和。这么一来可以很容易的通过直方图的波峰波谷把4个字母分割开。图片的下半部分是扫描线分隔法，因为干扰线和字符旋转的存在，只有M和5直接才出现了连续的空白部分。

除了垂直像素直方图，还可以从不同的角度进行斜线方向的像素数投影，这种方式对于每次全体字符都随机向一个角度旋转的验证码效果很好。对于每次字符大小和数量都一样的验证码还可以用平均分割法，也就是直接先把中间的文字部分整体切出来，然后按宽度平均分成几份，这种方式对字符粘连比较多用其他方式不好分割的验证码很有用，之前的megaupload的3位字母验证码就是通过这种方式成功分割的。

另外对于彩色的验证码，还可以用颜色分割，比如12306的：



12306的验证码，每个字符颜色都不一样，真是省事啊。

作为验证码识别里的难点，分割字符还有很多种算法，包括笔画分析曲线角度分析等等，不过即便如此，对粘连的比较厉害的字符还是很难成功的。

标准化

标准化的意思是指对于同一个字符，尽可能让每次识别前的样本都一致，以提高识别率。而验证码设计者则会用随机旋转，随机扭曲还有随机字体大小的方式防止字符被简单方法识别。

还原随机旋转的字符一般采用的是旋转卡壳算法：





此算法非常简单，对一张图片左右各旋转30度的范围，每次1度，旋转后用扫描线法判断字符的宽度，对于标准的长方形字体，在完全垂直的时候肯定是宽度最窄的。嗯？纳尼？上面的图是中间的最窄？好像的确是这样，不过只要每次旋转后的结果都一样，对于识别率不会有影响。

扭曲还原的算法比较蛋疼，效果也不怎么样（其实我不会），不过如果识别算法好的话，对扭曲的字符只要人能认出来，识别率也可以达到接近人类的水准。

还有一些常用到的算法，对于提高识别率和减少样本数量有一定帮助：

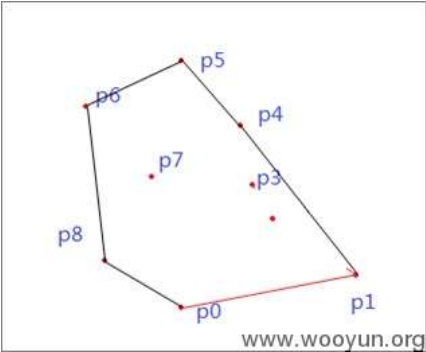
骨架细化：腐蚀算法



腐蚀算法的原理有点像剥洋葱，从最外层沿着最外面的一层像素一圈一圈的去掉，直到里面只剩下一层像素为止。腐蚀算法里面需要用到另一个算法，叫做凸包算法，用来找一堆像素点里面最外围的一层。

最后就是把字符变成统一大小，一般而言是把全部字符都缩到和验证码里出现过的最小的字符一个大小。

详情请自行google.....



分割算法差不多就到这里了，都是一些比较基础的内容。下面是最终的识别。

0x02 识别

其实到了这一步，单独的字符已经分离出来了，可以训练tesseract ocr来识别了，样本数量多的话，识别率也是很高的。不过在这里还是要讲一下，如何自己来实现识别过程。

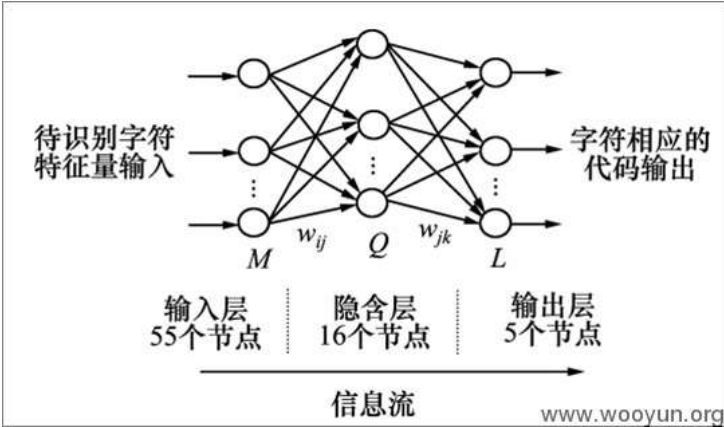
第一步，样本现在应该已经是一个矩阵的形式了，有像素的地方是1，背景是0，先肉眼识别一下，然后把这个矩阵转换为字符串，建立一个键值对，标明这串字符串是什么字符。之后就只需要多搜集几个同样字符的不同字符串变形，这就是制作模板的过程，。

搜集了足够多的模板后，就可以开始识别了，最简单的方法：汉明距离，但是如果字符有少许扭曲的话，识别率会低的离谱。对比近似字符串用的最多一般是编辑距离算法(Levenshtein Distance)，具体请自己google。

两种算法的差别在于，对同样两个字符串对比10010101和10101010，汉明距离是6，但是编辑距离是2。

最后一种最NB的识别算法，就是神经网络，神经网络是一种模拟动物神经元工作模式的算法，神经网络有多种不同的结构，但是基本架构分为输入层，隐含层和输出层，输入和输出均为二进制。

( / )
( / n
ew
sen
d )
( / w
p-
logi
n.p
hp
?
acti
on
=lo
go
ut&
red
ire
ct\_t
o=
http
%3
A%
2F
%2
Fdr
op
s.w
ooy
un.
org
)



对于验证码识别来说，输入和输出节点不宜过多，因为多了很慢.....所以如果样本矩阵为20x20 400个像素的话，需要对应的也要有400个输入节点，因此我们需要对整个矩阵提取特征值，比如先横向每两个数字XOR一下，然后再竖向每两个数字XOR。

Python有很多封装好的神经网络库，你所需要的只是把特征值输入神经网络，再告诉他你给他的是什（字）符，这样多喂几次之后，也就是训练的过程，随着训练的进行，神经网络的内部结构会改变，逐渐向正确的答案靠拢。神经网络的优势是，对于扭曲的字符识别成功率非常高。另外神经网络在信息安全中还可以起到很多其他作用，比如识别恶意代码等等。

动画验证码

有些不甘寂寞的程序员又玩出了些新花样，比如各种GIF甚至flv格式的动画验证码，下面我来分析一下腾讯安全中心的GIF验证码。



晃来晃去的看似很难，放慢100倍一帧一帧再看看？



基本上每帧都有一个字符和其他的分开，用最简单的扫描法就能分割出来。

剩下的就很轻松了，旋转还原之后，先填充内部空白，缩小细化之后做成模板对比，识别率怎么也得有90%了。

原本一张图就能搞定的事情，偏偏给了我们8张图，而且每张图还有一点区别，平白无故增大了很多信息量。

另外就是一些所谓的高用户体验的验证码，比如freebuf的：

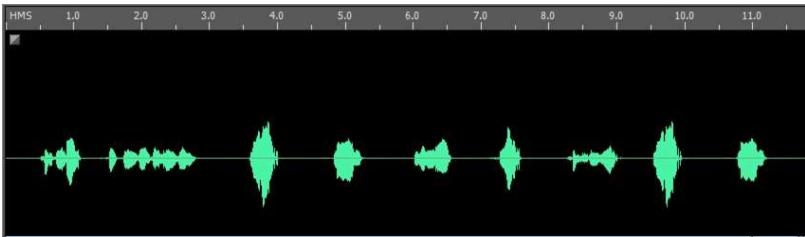


拖动解锁按钮会触发执行一段js，生成一串随机字符串，ajax给后端程序判断。

破解方式就当留给大家的思考题了，假如我想刷评论的话，怎么办。

还有就是声音验证码的识别，现在很多验证码为了提高用户体验和照顾视觉障碍的用户，都有声音验证码，一般来说是机器生成一段读数字的语音。但是在这方面上很多程序员都偷懒了，预先找了10个数字的声音录音，然后生成的时候把他们随机拼到一起，结果就是这样：





前3秒为语音提示，后面的是数字，有没有发现什么？

声音也是可以做成模板的哦

最后就是应该怎么样去设计验证码

- 整体效果
- 字符数量一定范围内随机
- 字体大小一定范围内随机
- 波浪扭曲(角度方向一定范围内随机)
- 防识别
- 不要过度依赖防识别技术
- 不要使用过多字符集-用户体验差
- 防分割
- 重叠粘连比干扰线效果好
- 备用计划
- 同样强度完全不同的一套验证码

附件添加一个破解验证码的实例包括程序大家自行研究吧：验证码识别  
(<http://static.wooyun.org/20141017/2014101711073747148.zip>)

★收藏    分享

昵称

验证码



写下你的评论...

发表



Hyrut 2016-01-19 12:24:08

不多说，文章很棒。

👤 回复



似水流年 2016-01-12 10:36:05

现在的图形验证码已经没有什么安全性可言了，12306的验证码看起来很难，实则用谷歌和百度的公共端口图像识别就可以搞定。目前安全性比较高的要算极验验证的验证码，全新的验证安全2.0，基于行为式验证的技术，游戏网站，论坛贴吧以及很多航空网站都在使用极验验证码。<http://www.geetest.com/>

👤 回复



2 2016-01-09 16:02:15

2

👤 回复



1234 2016-01-05 14:54:53

asdfasdf

👤 回复



路人甲 2016-01-01 15:17:46

很好很强大

👤 回复



thanatos 2015-12-09 16:39:29

流弊啊！不过看看今年的12306，我特么真是醉了

👤 回复



123 2015-11-29 13:49:30

哈哈哈哈哈

👤 回复



tester 2015-11-09 09:22:05

不明觉厉

👤 回复



小龙女 2015-11-07 23:46:03

@BeenQuiver 恩很对，加油哦！吆吆哒

👤 回复



测试者 2015-10-30 15:54:20

呵呵

👤 回复



奇迹通讯验证码专家 2015-10-30 11:14:13

@开发者 27746151 注验证码

👤 回复



开发者 2015-10-30 11:13:02

@奇迹通讯验证码专家 qq号有吗

👤 回复



忆人伤 2015-10-29 18:52:06

呵呵

👤 回复



huhufafa 2015-10-24 17:00:13

验证码能打码吗


👤 回复




枯荣 2015-08-06 16:28:47

赞赞赞赞赞

 回复

 **天道** 2015-03-23 16:39:28  
看来12306无论怎么做都奈何不了黄牛，黄牛软件都是大牛做的呀

 回复

 **文明** 2015-03-16 16:46:14  
老大 对于12306最近新出的验证码有什么话要说的吗


 回复

 **牛逼** 2015-01-22 11:15:32  
.....


 回复

 **你猜我猜不猜** 2015-01-21 13:57:06  
膜拜大神


 回复

 **huhu** 2015-01-19 12:55:49  
确实写得不错 学到了很多东西，赞 另外这个网站验证码好简单啊


 回复

 **YwiSax** 2015-01-19 11:03:51  
好强大


 回复

 **BeenQuiver** 2015-01-17 22:22:57  
图像处理真是博大精深，可惜没听课


 回复

 **万象时代** 2015-01-06 16:57:30  
非常高端的学术研究啊！


 回复

 **码农哥** 2014-08-22 22:16:41  
讲的相当详细，以后用到了来翻翻！


 回复

 **sun** 2014-06-28 13:07:31  
too naive


 回复

 **小怂** 2014-01-23 14:47:03  
大神，看不懂你程序里面的算法，有没有加了注释的程序能来一份？


 回复

 **瞌睡龙** 2013-10-23 11:09:22  
我觉得这货已经很干了，你需要的不是干货，是别人写好的识别程序放到你面前，你可以直接拿去用。此篇文章目的不在于写一个万能的识别程序，而是在引导下，可以自行针对不同的验证码，用不同算法写出最适合的识别程序。

 回复

 **net2** 2013-10-23 09:27:34  
有没有一些好点的破解带验证码表单的工具？fastocr不行。扯了这么多，都是科普，需要来点干货好么？

 回复

 **一天到晚吃** 2013-07-28 00:35:14  
不知道楼主试过天涯论坛发帖时的那个验证码不？最近在做这个

 回复



花心 2013-07-06 21:02:37  
太复杂了

回复



小一 2013-06-10 15:25:29  
这篇文章让我想起了酷壳的一篇文章《各式各样的验证码》，要想答对，先求个极限再说。。。

回复



熊猫 2013-06-09 23:20:12  
神!

回复



xsse 2013-06-09 21:41:16  
其中很弱的那个看起来好面熟啊，不过这个很赞~

回复



c4rp3nt3r 2013-06-09 13:52:06  
赞

回复



银冥币 2013-06-08 21:16:55  
勉强看懂了前半部分，后面就蒙了Orz

回复



小胖胖要减肥 2013-06-08 14:10:20  
你们团队还研究这个呀 龙哥

回复



goderci 2013-06-08 12:46:18  
信息量好大!

回复

感谢知乎授权页面模版