**生物特征识别**

**计算机笔迹鉴定**

**第八组总结报告**

**小组成员：郭雨琛，王腾科，洪兴华，童玮男，黄凯迪**

**一、计算机笔迹鉴定与生物特征识别的关系**

所谓生物特征识别技术是指通过计算机利用人体所固有的生理特征或行为特征来进行个人身份鉴定的技术。生理特征与生俱来，多为先天性的;行为特征则是习惯使然，多为后天性的。我们将生理和行为特征统称为生物特征。常用的生物特征包括:指纹、掌纹、虹膜、脸像、声音、笔迹等。可见，笔迹鉴定是生物特征识别技术的组成部分之一。与其他识别技术相比，笔迹鉴定在特征的复杂性、稳定性上都有很大的不同。



**二、本组选择的计算机笔迹鉴定总体方向**

1. **根据取样的方式选择离线笔迹鉴定：**

离线笔迹鉴定的对象是写在纸上的字符这些字符通过扫描仪或摄像机转化为计算机能处理的数字信号。 离线鉴定由于失去了很多的动态特征信息，难度较大。但因离线笔迹鉴定本身具有更广泛的应用范围，如很多的支票签名鉴定、合同签署签名及犯罪学上的笔迹鉴定都是离线的，离线笔迹鉴定具有更多的研究价值与意义 。

1. **根据考察的对象和提取特征的方法选择文本相关：**

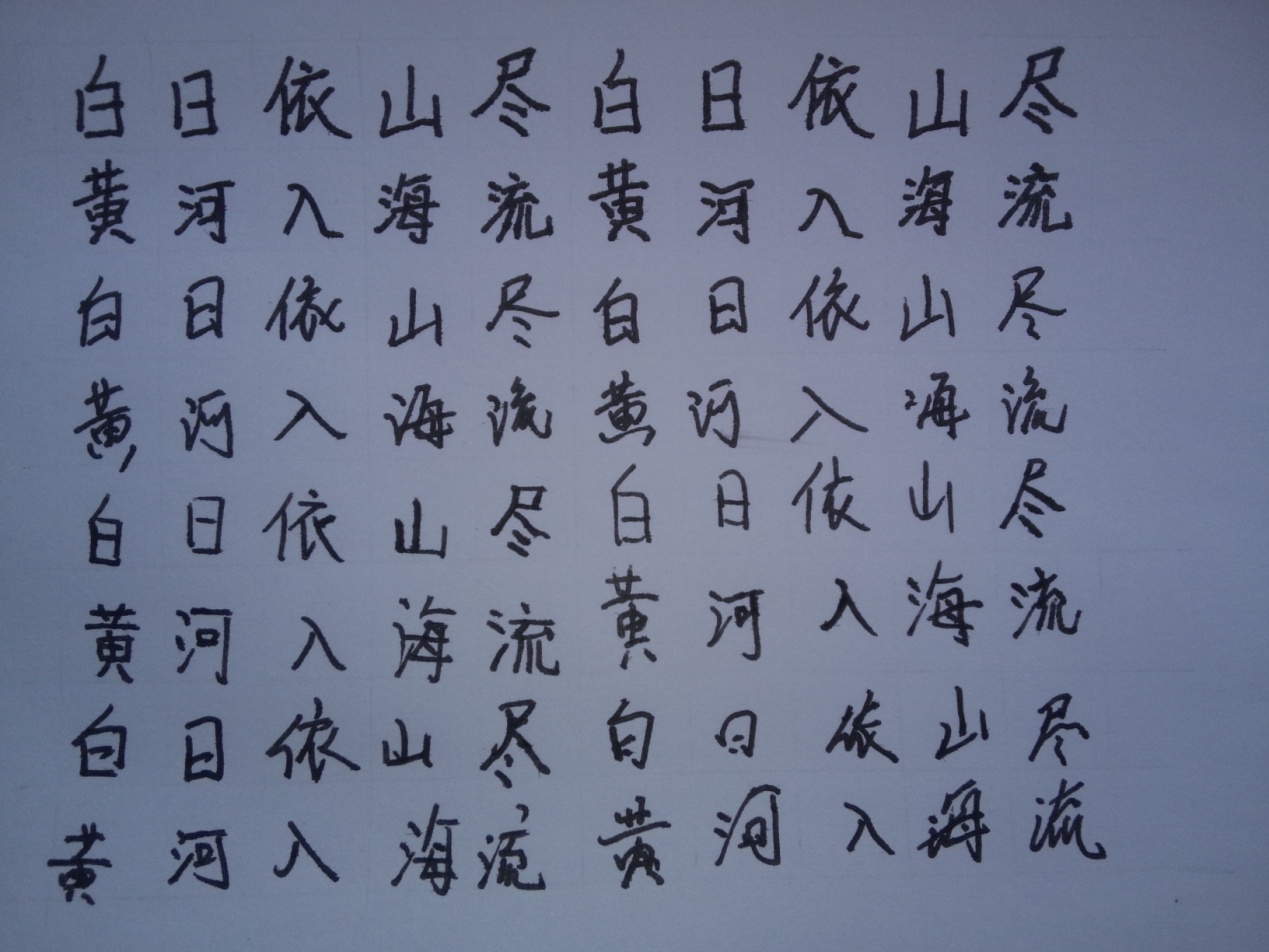
用文本有关方法进行鉴定时，从检验笔迹和样本笔迹中选择相+

+1--同的单字(称为特征字)进行比较，在相同字符的基础上鉴定。因而提取的笔迹特征是依赖于字符类别，也就是说依赖于文本内容的。

文本有关特征可以通过正交变换和直方图方法提取。此外，文本有关方法还有标准模板变形法、高阶相关法、笔划匹配法、多通道分解与匹配、简化的Wiener分布应用等。

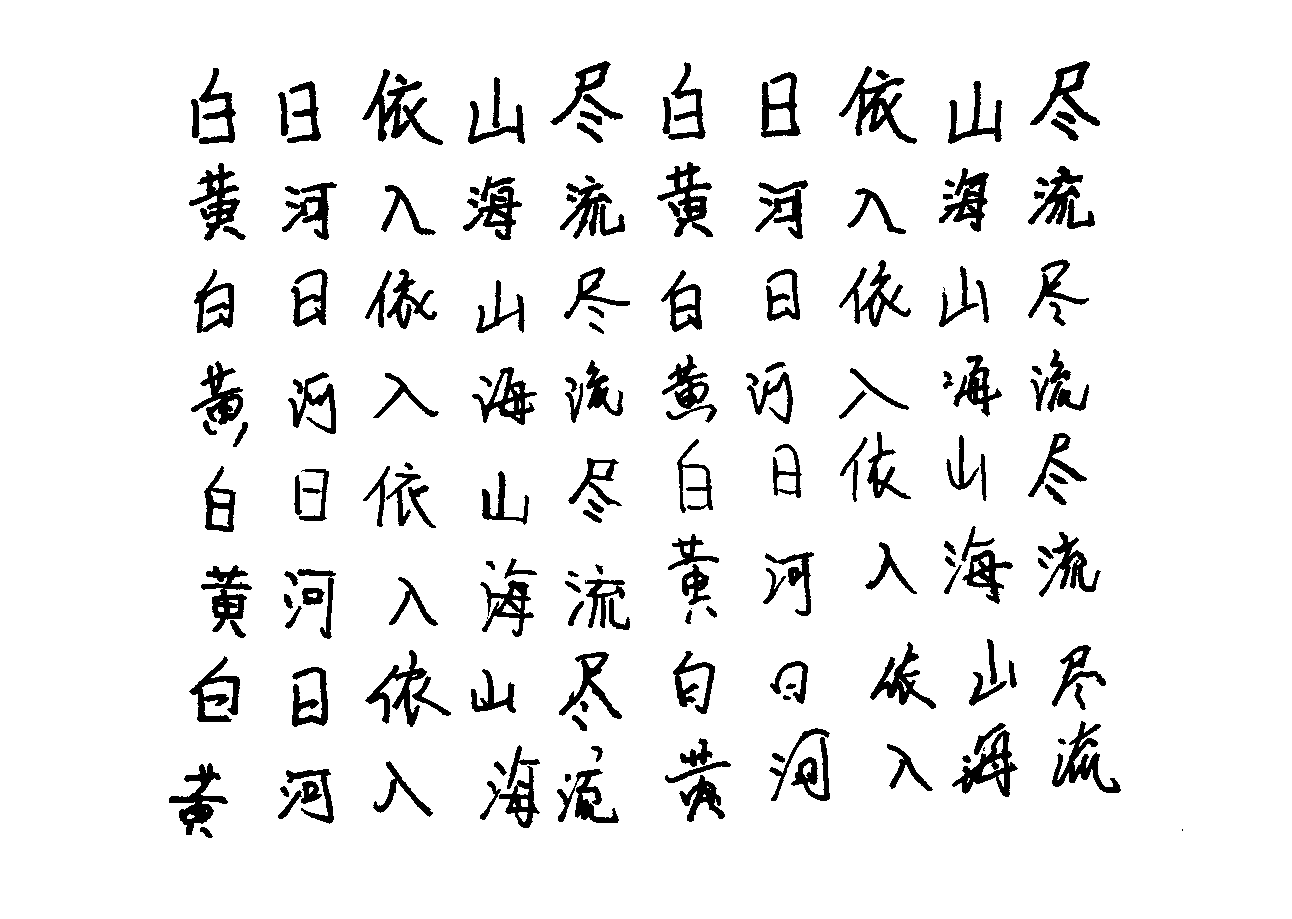
三、**图像预处理**

1.笔迹图像的采集：检材和样本笔迹图像的采集是计算机自动鉴定笔迹中最重要的一部分。因为计算机通常为二进制识别，所以要将待鉴定的笔迹转化为计算机可以运算的数字信号，实践中主要通过扫描仪扫描和数码照相机拍照等方式将笔迹形成数字化图像，输入计算机。



2.图像的预处理 ：基于图像采集设备自身的原因以及图像本身也存在一些无用的信息等，这时就需要通过对采集的图像进行平滑、去除噪音、锐化、归一化、二值化、图像分割等处理，使图像特征更为清晰、显著，从而为笔迹特征的选取提供更有利的条件。

经过灰度化（加权平均值法）、去噪（均值滤波法）、二值化后的图像如下：



接下来对图片进行汉字切分和尺寸归一化的处理得到纹理图像：















![C:\Users\Administrator\AppData\Roaming\Tencent\Users\1030304876\QQ\WinTemp\RichOle\_0](_~94KDHI@`MGJA_8N`N.png](data:image/png;base64,)

至此图像预处理完成。

**四、纹理图像的特征提取与模式匹配**

将图片中每一行最左边和左右边的黑色像素点以及每一列最上边和最下边的黑色像素点保存到四个数组当中同时将行与行之间像素点的列之差（即黑色像素点的变化率）保存到另外四个数组中，把上述八个数组当做变量带入到函数StrDetect当中，得到每个字上下左右4个边界的轮廓特征，并以字符形式表示。

我们将四个边上的纹理特征分为三大类

第一类：竖（V）；左斜（L）；右斜（R）；突变（P）

第二类：左半圆弧（C）；右半圆弧（Q）

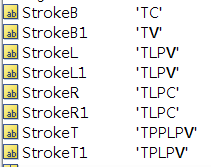
第三类：结构待定（T）

最后将得到的字符串纹理特征带入到模式匹配算法中，分析得到笔记鉴别的最后结果。

此处已两个同一个人写的“白”字为例：

得到的纹理特征以及匹配度rate1=0.7500

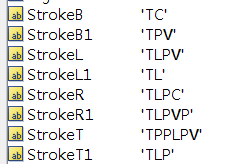
 



C:\Users\Administrator\AppData\Roaming\Tencent\Users\1030304876\QQ\WinTemp\RichOle\ZZKBTY7YBLG[}~T${BA@@Z6.png

而以上的第一个“白”字与另外一个人写的“白”字相比较得到的纹理特征和匹配度如下：



C:\Users\Administrator\AppData\Roaming\Tencent\Users\1030304876\QQ\WinTemp\RichOle\O)UBE$N1EYFL8NOD4H$775D.png

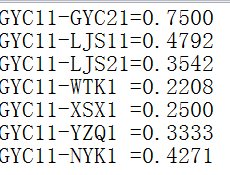
可见第二次的匹配度只有0.2208，与第一次的0.75相比反映出第一组的两个字更相似，更像是同一个人写的。

当然要得出更加精确的笔迹鉴定结果，我们收集的样本由6个人提供了8份笔迹，其中两人没人提供了两份样本作为参照和比较，因为是文字相关的识别每份笔迹都是“白日依山尽 黄河入海流”这10个字。

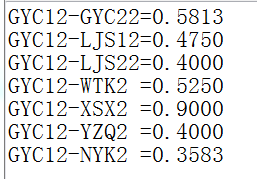
这意味着要想得到最精确的匹配分析结果，要将10组，每组8个相同的字之间都进行匹配分析，再将得到的匹配度进一步处理后才能得到最终结论。

部分匹配度数据如下：

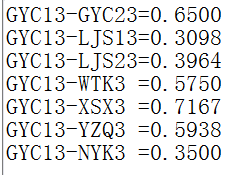
第一组“白”字：



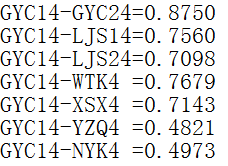
第二组“日”字：



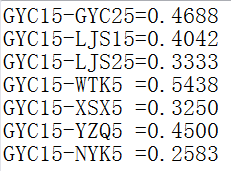
第三组“依”字：



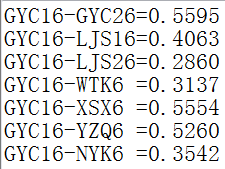
第四组“山”字：



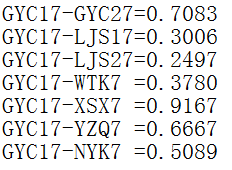
第五组“尽”字：



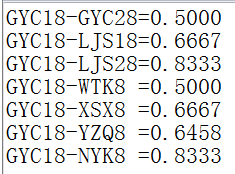
第六组“黄”字：



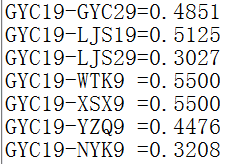
第七组“河”字：



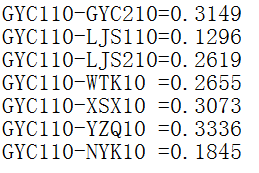
第八组“入”字：



第九组“海”字：



第十组“流”字：



五、匹配结果分析

1. 单字匹配规则：

经过多次试验我们发现阈值取0.5时，算法准确率最高，即最后的得到的匹配度大于0.5认为该单字字迹与训练样本是同一人所写，否则不是。

1. 平均匹配度：

根据多组数据得到的匹配度进行统计，算出整个句子（段落，或文章）的平均匹配度。同样把阈值设为0.5，若整体的平均匹配率大于0.5，则认为这篇样本文章与训练样本是同一个人所写，否则不是。

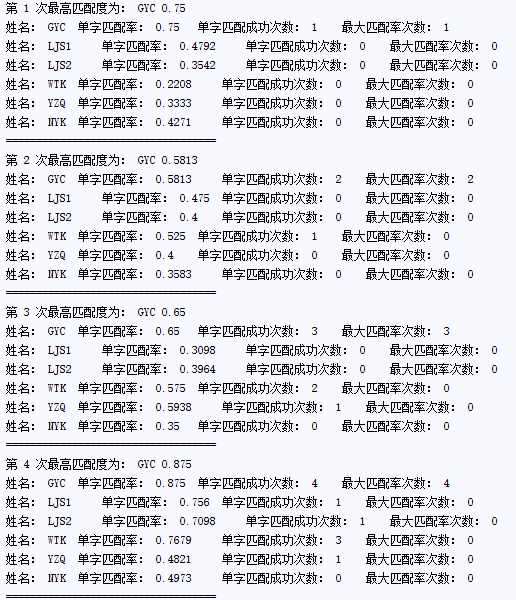
1. 匹配成功次数：

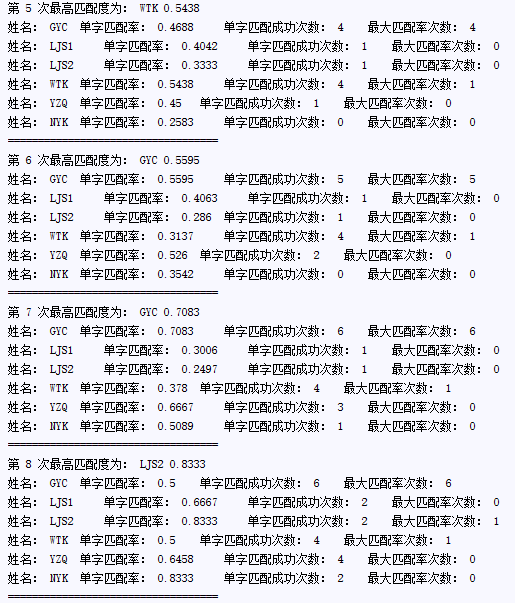
对每一次单字匹配，若匹配成功，则累加一次匹配成功次数

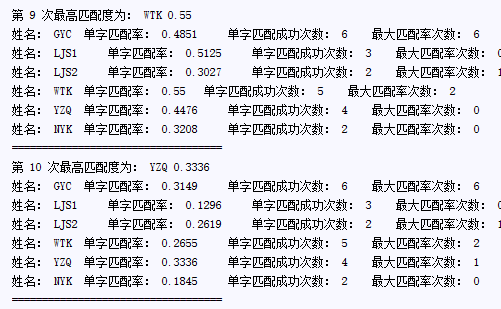
该参数可作为判断的参考

1. 最高匹配度次数：

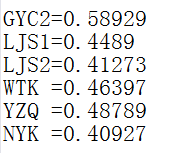
统计样本段落与每个训练样本中匹配度最高的次数，同样可以作为判断的参考。

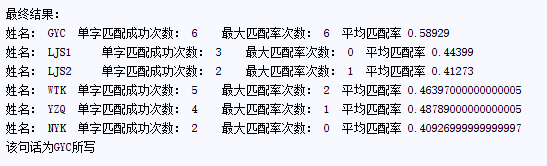






根据上面10组匹配度的经过平均算法后得到整体的均匹配度：





由结果知：第一个最终匹配值GYC2与GYC1的匹配度大于0.5，判断是同一人所写，其余均小于0.5判定不是同一人所写。

纹理特征提取算法程序如下：

num=input('请输入第num个字：');

I0=imread(['D:\GG\GYC1\GYC1' int2str(num) '.png']);% 必须为二值图像

%I0=abs(256-I0);

I=im2bw(I0,0.4);

[y0 x0]=size(I);

Range=sum((~I)');

Hy=0;

for j=1:y0

if (Range(j)>=1)

Hy=Hy+1;

end

end

RangeX=sum((~I));

Wx=0;

for i=1:x0

if (RangeX(i)>=1)

Wx=Wx+1;

end

end

I

Amp=24/Hy; % 将文字图像归一化到24像素点的高度。

I=imresize(I,Amp);

[y x]=size(I);

%I=bwmorph(~I,'skel',Inf);

%I=~I;

tic

%====== 基本结构 =======%

% 第一类：竖（V）；左斜（L）；右斜（R）；突变（P）

% 第二类：左半圆弧（C）；右半圆弧（Q）

% 的三类：结构待定（T）；

%=====================================%

Left=zeros(1,y); % 左端轮廓检测

for j=1:y

i=1;

while ((i<=x)&&(I(j,i)==1))

i=i+1;

end

if (i<=x)

Left(j)=i;

end

end

for j=1:y-1

LeftD(j)=Left(j+1)-Left(j);

end

%========== 结构特征提取 =============%

j=1;

while ((Left(j)<1)&&(j<y))

j=j+1;

end

Y1=j;

j=y;

while ((Left(j)<1)&&(j>1))

j=j-1;

end

Y2=j-1; % 去掉急剧变化的两端

%============== 右边 ==================%

Right=zeros(1,y); % 左端轮廓检测

for j=1:y

i=x;

while ((i>=1)&&(I(j,i)==1))

i=i-1;

end

if (i>=1)

Right(j)=i;

end

end

for j=1:y-1

RightD(j)=Right(j+1)-Right(j);

end

%=====================================%

Top=zeros(1,x); % 顶端轮廓检测

for i=1:x

j=1;

while ((j<=y)&&(I(j,i)==1))

j=j+1;

end

if (j<=y)

Top(i)=j;

end

end

for i=1:x-1

TopD(i)=Top(i+1)-Top(i);

end

%==============================%

i=1;

while ((Top(i)<1)&&(i<x))

i=i+1;

end

X1=i;

i=x;

while ((Top(i)<1)&&(i>1))

i=i-1;

end

X2=i-1; % 去掉急剧变化的两端

%===================================%

Bottom=zeros(1,x); % 底部轮廓检测

for i=1:x

j=y;

while ((j>=1)&&(I(j,i)==1))

j=j-1;

end

if (j>=1)

Bottom(i)=j;

end

end

for i=1:x-1

BottomD(i)=Bottom(i+1)-Bottom(i);

end

%========== 数字 1 的宽度特征 =========%

Width=zeros(1,y);

for j=1:y

Width(j)=Right(j)-Left(j);

end

W=max(Width);

Po=0; % 用于检测笔划

Ne=0;

NS=0;%笔划数

for i=X1+4:X2-4

for j=1:y-1

if ((I(j+1,i)-I(j,i))>0) % 由黑到白

Po=Po+1;

if ((Po>=2)&&(j<=fix(0.7\*y)))

Po=3;

end

else if ((I(j+1,i)-I(j,i))<0) % 由白到黑

Ne=Ne+1;

if ((Ne>=2)&&(j<=fix(0.7\*y)))

Ne=3;

end

end

end

end

NS=[NS max(Po,Ne)];

Po=0;

Ne=0;

end

Comp=max(NS);

%========== 轮廓结构特征提取 ==========%

if (min(W,Wx)>10)

StrokeT=StrDetect01(TopD,X1,X2,3,6); % 顶部基本结构检测

StrokeL=StrDetect01(LeftD,Y1,Y2,3,5); % 左边基本结构检测

StrokeR=StrDetect01(RightD,Y1,Y2,3,5); % 右边基本结构检测

StrokeB=StrDetect01(BottomD,X1,X2,3,6); % 底部基本结构检测

%========== 识别 ==========%

Digit=Recognition(StrokeT,StrokeL,StrokeR,StrokeB,Comp)

else

Digit='1'

end

t=toc;

%======= 显示 ======%

px=(1:x);

py=(1:y);

figure(1);

imshow(I);

figure(2);

plot(Left);grid

title('左轮廓');

figure(3);

plot(Top);grid

title('上轮廓');

figure(4);

plot(Right);grid

title('右轮廓');

figure(5);

plot(Width);grid

title('宽度');

n=input('请输入对比编号：');

switch n

case 1

I0=imread(['D:\GG\GYC2\GYC2' int2str(num) '.png']);% 必须为二值图像

case 2

I0=imread(['D:\GG\LJS1\LJS1' int2str(num) '.png']);% 必须为二值图像

case 3

I0=imread(['D:\GG\LJS2\LJS2' int2str(num) '.png']);% 必须为二值图像

case 4

I0=imread(['D:\GG\WTK\WTK' int2str(num) '.png']);% 必须为二值图像

case 5

I0=imread(['D:\GG\XSX\XSX' int2str(num) '.png']);% 必须为二值图像

case 6

I0=imread(['D:\GG\YZQ\YZQ' int2str(num) '.png']);% 必须为二值图像

case 7

I0=imread(['D:\GG\NYK\NYK' int2str(num) '.png']);% 必须为二值图像

end

I=im2bw(I0,0.4);

[y0 x0]=size(I);

Range=sum((~I)');

Hy=0;

for j=1:y0

if (Range(j)>=1)

Hy=Hy+1;

end

end

RangeX=sum((~I));

Wx=0;

for i=1:x0

if (RangeX(i)>=1)

Wx=Wx+1;

end

end

Amp=24/Hy; % 将文字图像归一化到24像素点的高度。

I=imresize(I,Amp);

[y x]=size(I);

%I=bwmorph(~I,'skel',Inf);

%I=~I;

tic

%====== 基本结构 =======%

% 第一类：竖（V）；左斜（L）；右斜（R）；突变（P）

% 第二类：左半圆弧（C）；右半圆弧（Q）

% 的三类：结构待定（T）；

%=====================================%

Left=zeros(1,y); % 左端轮廓检测

for j=1:y

i=1;

while ((i<=x)&&(I(j,i)==1))

i=i+1;

end

if (i<=x)

Left(j)=i;

end

end

for j=1:y-1

LeftD(j)=Left(j+1)-Left(j);

end

%========== 结构特征提取 =============%

j=1;

while ((Left(j)<1)&&(j<y))

j=j+1;

end

Y1=j;

j=y;

while ((Left(j)<1)&&(j>1))

j=j-1;

end

Y2=j-1; % 去掉急剧变化的两端

%============== 右边 ==================%

Right=zeros(1,y); % 左端轮廓检测

for j=1:y

i=x;

while ((i>=1)&&(I(j,i)==1))

i=i-1;

end

if (i>=1)

Right(j)=i;

end

end

for j=1:y-1

RightD(j)=Right(j+1)-Right(j);

end

%=====================================%

Top=zeros(1,x); % 顶端轮廓检测

for i=1:x

j=1;

while ((j<=y)&&(I(j,i)==1))

j=j+1;

end

if (j<=y)

Top(i)=j;

end

end

for i=1:x-1

TopD(i)=Top(i+1)-Top(i);

end

%==============================%

i=1;

while ((Top(i)<1)&&(i<x))

i=i+1;

end

X1=i;

i=x;

while ((Top(i)<1)&&(i>1))

i=i-1;

end

X2=i-1; % 去掉急剧变化的两端

%===================================%

Bottom=zeros(1,x); % 底部轮廓检测

for i=1:x

j=y;

while ((j>=1)&&(I(j,i)==1))

j=j-1;

end

if (j>=1)

Bottom(i)=j;

end

end

for i=1:x-1

BottomD(i)=Bottom(i+1)-Bottom(i);

end

%========== 数字 1 的宽度特征 =========%

Width=zeros(1,y);

for j=1:y

Width(j)=Right(j)-Left(j);

end

W=max(Width);

Po=0; % 用于检测笔划

Ne=0;

NS=0;%笔划数

for i=X1+4:X2-4

for j=1:y-1

if ((I(j+1,i)-I(j,i))>0) % 由黑到白

Po=Po+1;

if ((Po>=2)&&(j<=fix(0.7\*y)))

Po=3;

end

else if ((I(j+1,i)-I(j,i))<0) % 由白到黑

Ne=Ne+1;

if ((Ne>=2)&&(j<=fix(0.7\*y)))

Ne=3;

end

end

end

end

NS=[NS max(Po,Ne)];

Po=0;

Ne=0;

end

Comp=max(NS);

%========== 轮廓结构特征提取 ==========%

if (min(W,Wx)>10)

StrokeT1=StrDetect01(TopD,X1,X2,3,6); % 顶部基本结构检测

StrokeL1=StrDetect01(LeftD,Y1,Y2,3,5); % 左边基本结构检测

StrokeR1=StrDetect01(RightD,Y1,Y2,3,5); % 右边基本结构检测

StrokeB1=StrDetect01(BottomD,X1,X2,3,6); % 底部基本结构检测

%========== 识别 ==========%

Digit=Recognition(StrokeT,StrokeL,StrokeR,StrokeB,Comp)

else

Digit='1'

end

t=toc

结果分析Python3.5代码如下：

**class sort**(object)**:  
 def \_\_init\_\_**(self,*name*)**:** self.name**=***name* self.iter**=**0  
 self.max\_iter**=**0  
 self.score**=**0  
 self.total\_score**=**0  
 self.avg\_score**=**0  
  
GYC**=**sort(**'GYC'**)  
LJS1**=**sort(**'LJS1'**)  
LJS2**=**sort(**'LJS2'**)  
WTK**=**sort(**'WTK'**)  
YZQ**=**sort(**'YZQ'**)  
NYK**=**sort(**'NYK'**)  
  
**def add\_iter**(*sort*)**:  
 if** *sort*.score**>**0.5**:** *sort*.iter**+=**1  
 **return  
  
def return\_items**(*sort*)**:** print(**'姓名：'**,*sort*.name,**'**\t**单字匹配率：'**,*sort*.score,**'**\t**单字匹配成功次数：'**,*sort*.iter,**'**\t**最大匹配率次数：'**,*sort*.max\_iter)  
 **return  
  
def return\_result**(*sort*)**:** *sort*.avg\_score**=***sort*.total\_score**/**10  
 print(**'姓名：'**,*sort*.name,**'**\t**单字匹配成功次数：'**,*sort*.iter,**'**\t**最大匹配率次数：'**,*sort*.max\_iter,**'**\t**平均匹配率'**,*sort*.avg\_score)  
  
**def get\_sore**(*url*,*k*)**:  
 with** open(*url*,**'r'**) **as** fd**:** flists**=**fd.readlines()  
 prop**=**0.0  
 i**=**0  
 **for** flist **in** flists**:** i**+=**1  
 **if** i**==**1**:** GYC.score**=**float(flist)  
 GYC.total\_score**+=**GYC.score  
 add\_iter(GYC)  
 **elif** i**==**2**:** LJS1.score**=**float(flist)  
 LJS1.total\_score**+=**LJS1.score  
 add\_iter(LJS1)  
 **elif** i**==**3**:** LJS2.score**=**float(flist)  
 LJS2.total\_score**+=**LJS2.score  
 add\_iter(LJS2)  
 **elif** i**==**4**:** WTK.score**=**float(flist)  
 WTK.total\_score**+=**WTK.score  
 add\_iter(WTK)  
 **elif** i**==**5**:** YZQ.score**=**float(flist)  
 YZQ.total\_score**+=**YZQ.score  
 add\_iter(YZQ)  
 **elif** i**==**6**:** NYK.score**=**float(flist)  
 NYK.total\_score**+=**NYK.score  
 add\_iter(NYK)  
 **if** prop**<**float(flist)**:** prop**=**float(flist)  
  
 **if** GYC.score**==**prop**:** GYC.max\_iter**+=**1  
 print(**'==================================='**)  
 print(**'第'**,*k***+**1,**'次最高匹配度为：'**,GYC.name,GYC.score)  
 **elif** LJS1.score**==**prop**:** LJS1.max\_iter**+=**1  
 print(**'==================================='**)  
 print(**'第'**,*k***+**1,**'次最高匹配度为：'**,LJS1.name,LJS1.score)  
 **elif** LJS2.score**==**prop**:** LJS2.max\_iter**+=**1  
 print(**'==================================='**)  
 print(**'第'**,*k***+**1,**'次最高匹配度为：'**,LJS2.name,LJS2.score)  
 **elif** WTK.score**==**prop**:** WTK.max\_iter**+=**1  
 print(**'==================================='**)  
 print(**'第'**,*k***+**1,**'次最高匹配度为：'**,WTK.name,WTK.score)  
 **elif** YZQ.score**==**prop**:** YZQ.max\_iter**+=**1  
 print(**'==================================='**)  
 print(**'第'**,*k***+**1,**'次最高匹配度为：'**,YZQ.name,YZQ.score)  
 **elif** NYK.score**==**prop**:** NYK.max\_iter**+=**1  
 print(**'==================================='**)  
 print(**'第'**,*k***+**1,**'次最高匹配度为：'**,NYK.name,NYK.score)  
 **return  
  
def true\_writer**()**:  
 if** GYC.avg\_score**>**0.5**:** print(**'该句话为%s所写' %**GYC.name)  
 **elif** LJS1.avg\_score**>**0.5**:** print(**'该句话为%s所写' %**LJS1.name)  
 **elif** LJS2.avg\_score**>**0.5**:** print(**'该句话为%s所写' %**LJS2.name)  
 **elif** WTK.avg\_score**>**0.5**:** print(**'该句话为%s所写' %**WTK.name)  
 **elif** YZQ.avg\_score**>**0.5**:** print(**'该句话为%s所写' %**YZQ.name)  
 **elif** NYK.avg\_score**>**0.5**:** print(**'该句话为%s所写' %**NYK.name)  
 **return  
  
for** i **in** range(10)**:** file\_name**=**str(i)**+'.txt'** url**='E:**\\**笔迹识别**\\**'+** file\_name  
*# print('第%i次匹配' %i)* get\_sore(url,i)  
 return\_items(GYC)  
 return\_items(LJS1)  
 return\_items(LJS2)  
 return\_items(WTK)  
 return\_items(YZQ)  
 return\_items(NYK)  
  
print(**'==================================='**)  
print(**'最终结果：'**)  
return\_result(GYC)  
return\_result(LJS1)  
return\_result(LJS2)  
return\_result(WTK)  
return\_result(YZQ)  
return\_result(NYK)  
true\_writer()