算法设计

灰度图像二值化

需要参数：

阈值T

实现描述：

1. 建立一个新图像，长宽与原图像相同，类型为二值图像
2. 从（0,0）位置获取原图像的一个像素点P
3. 获取该像素点P的灰度值G
4. 如果G>T在新图像相应位置设置G=255，如果G<T在新图像相应位置设置G=0
5. 如果原图像存在下一像素P，则获取下一像素并重复步骤3,4，否则执行步骤6
6. 输出新图像

零散白像素点修复

需要参数：

二值图像B，预判阈值T

实现描述：

1. 建立一个新图像N，长宽与原图像相同，类型为二值图像
2. 从（0,0）位置开始获取原图像的一个像素点P
3. 设定标记F=false
4. 判断像素点P是否位于图像边界，即点P(X,Y)，(X-1，Y),(X+1，Y),(X,Y-1),(X,Y+1)是否都存在，如果P位于图像边界，跳转至步骤10
5. 获取像素P的灰度值G，如果G=0跳转至步骤10
6. 获取与P八连通的8个像素p[0]-p[7]
7. 获取p[0]-p[7]的灰度值g[0]-g[7]
8. 得到g[0]-g[7]中为0的个数r
9. 如果r>T，修改标记F=true
10. 如果标记F为false，在新图像相同位置像素设定灰度值与原图像的灰度值相同，如果标记为true，在新图像相同位置像素设定灰度值G=0
11. 如果原图像存在下一像素P，则获取下一像素，跳转至步骤3，否则执行步骤12
12. 输出新图像

垂直方向去噪（针对性算法）

需要参数：

探索深度D

实现描述：

1. 建立一个新图像N，长宽与原图像相同，类型为二值图像
2. 从（0,0）位置开始获取原图像的一个像素点P
3. 设定标记F=false
4. 判断像素点P是否为黑色像素，即判断像素点P的灰度G=0是否成立，如果不成立，跳转至步骤7
5. 设P点坐标（X,Y），图片高度为H，判断（X,Y+1）,(X,Y+2)……(X,Y+i)其中i<=D且i<=H是否为全为黑色像素，如果是，跳转步骤7
6. 修改标记F=true
7. 如果标记为false，在新图像相同位置像素设定灰度值与原图像的灰度值相同，如果标记为true，在新图像相同位置像素设定灰度值G=255
8. 如果原图像存在下一像素P，则获取下一像素，跳转至步骤3，否则执行步骤9
9. 输出新图像

八方向去噪算法（针对性算法）

需要参数：

判定阈值T，识别层数S

实现描述：

1. 建立一个新图像N，长宽与原图像相同，类型为二值图像
2. 从（0,0）位置开始获取原图像的一个像素点P
3. 设定标记F=false
4. 判断像素点P是否为黑色像素，即判断像素点P的灰度G=0是否成立，如果不成立，跳转至步骤9
5. 设P坐标（X,Y），取一下一系列坐标像素(X,Y+I)(X,Y-I)(X-I,Y)(X+I,Y)(X-I,Y-I)(X-I,Y+I)(X+I,Y-1)(X+I,Y+I) 其中I<=S 如果上述坐标像素有不存在的即超出图片边界，则此位置使用这个方向上的边界像素代替，即位置坐标的取值在图像的长宽范围内，例如(X-I,Y)的取值为(-1,10)则此坐标使用(0,10)位置像素代替
6. 找出步骤五所取坐标像素中白色像素的个数R
7. 如果R>=T\*S执行步骤8，否则执行步骤9
8. 修改标记F=true
9. 如果标记为false，在新图像相同位置像素设定灰度值与原图像的灰度值相同，如果标记为true，在新图像相同位置像素设定灰度值G=255
10. 如果原图像存在下一像素P，则获取下一像素，跳转至步骤3，否则执行步骤11
11. 输出新图像

模板滤波算法（类似中值滤波）

需要参数：

模板高H，模板宽W，选择位置S

实现描述：

1. 建立一个新图像N，长宽与原图像R相同，类型为二值图像，长为A，宽为B
2. 设定处理区域M (W/2,H/2)-(A-W/2+1,B-H/2+1)
3. 从区域M的开始位置(W/2,H/2)获取原图像的一个像素点P
4. 以像素点P(X,Y)的位置为卷积核取[H\*W]的模板Z（模板运算这里不做详细说明）
5. 获取到模板Z中的像素的灰度值g[1]g[2]……g[H\*W]
6. 将上述灰度值数组从小到大排序
7. 判断是否满足1<=S<=H\*W，如果不满足，取S=H\*W/2
8. 在新图像相同位置像素点P’(X,Y)设定灰度值G=g[S]
9. 判断在处理区域M中是否存在下一像素，如果存在则获取下一像素并跳转至步骤4
10. 将原图像R-M（这里表示差集）的区域内像素的灰度值复制到新图像N中
11. 输出新图像

投影切割算法

需要参数：

分割字符数A，基准线keys[A+1]，基准判定阈值T，起始扫描值S

注释：基准线，判定阈值，扫描起始值根据事先分析获取到的样本集的黑色像素点分布状况图人为确定。

实现描述：

1. 初始化两个数组cut\_start[A],cut\_end[A],分别记录字符切割的起点和终点
2. 初始化两个列表list\_start,list\_end记录沿着宽度方向像素扫描复合结果的值
3. 沿着宽度方向对图像进行扫描，记录每个宽度方向位置的该纵列的黑色像素点个数，保存在数组px[]中
4. 从px数组的第S个位置起，直到结束。若满足条件px[i-1]≠0 且px[i]=0，则将i记录到list\_end中
5. 从px数组的第S个位置起，直到结束。若满足条件px[i+1]≠0 且px[i]=0，则将i记录到list\_start中
6. 判断每组keys[j]-T<=u<=keys[j]+T （其中(1<=j<=A)）是否有u在list\_start存在，如果存在则cut\_start[j]=u，如果不存在则cut\_start[j]=keys[j]
7. 判断每组keys[j]-T<=u<=keys[j]+T （其中(2<=j<=A+1)）是否有u在list\_end存在，如果存在则cut\_end[j-1]=u，如果不存在则cut\_end[j-1]=keys[j]
8. 分别以每组的cut\_start[k],cut\_end[k]（1<=k<=A）为竖直方向分割线对原始图片进行切割

切割单字符多余部分算法

需要参数：

顶部基准线TL，底部基准线BL，顶部判断阈值TTH，底部判断阈值BTH

实现描述：

1. 沿着单字符图片的高度方向逐行扫描，将每行的黑色像素点个数记录到数组px[]中
2. 找到第一个满足px[i+1] ≠0且px[i]=0的位置，如果满足TL-TTH<=i<=TL+TTH记录cut\_start=i否则记录cut\_start=TL
3. 找到最后一个满足px[i-1] ≠0 且px[i]=0的位置，如果满足BL-BTH<=i<=BL+BTH记录cut\_end=I,否则记录cut\_end=BL
4. 以cut\_start和cut\_end为界将图片进行分割，保留之间的部分

二值化

根据输入的样本图片可以看出，图像主要由字符（黑色）和背景（白色）构成。图像大小为50px\*20px 字符部分约占1/3，字符骨干的灰度值Gc<20。白色背景的分布比较均匀，灰度值Gb>180，即背景的灰度Gb ∈[180,255]。其余部分即视为字符与背景的连接边缘，该类像素的灰度值在[21,179]之间都有分布且分布均匀，每个灰度值对应的像素个数很少。因此我们如此设置该二值化过程的二值化阈值T。

1. 先对原图像作出灰度直方图
2. 在定义域的子集中观测一个合理的界定值
3. 在区间中找到所有的
4. 求出的中位数
5. 记录二值化阈值

找到二值化阈值后对图像进行二值化处理

则有

图像

O

X

Y

P(x,y)

本文中处理的图像坐标原点设在图像左上角，图像宽度方向为X轴，高度方向为Y轴

对于其中的给定一点像素为方便表述及后续处理，定义像素位置向量

定义像素位置复数

。

对于输出为实数的（离散）函数 ，其二元函数表达形式 二维向量表达形式，

复变表达形式 在本文中表达相同意义，并取其表述简单的形式对算法进行描述。

取一个像素位置四连通域或者八连通域（视具体算法需求而定）的像素

取四连通域的四个像素即取该像素位置的上下左右四个像素，在该案例中使用像素的位置复数表达较为简单。

由复数的有关知识可知在上述坐标系中 所表示的位置分别为复数 所表示的像素位置的 右下左上 四个像素。

对于取八连通区域的八个像素，即像素的上下左右，左上左下右上右下八个像素。

为方便处理定义旋转用模长为1的复数

同时为保证旋转45°时依旧是格点复数，即x,y是整数，需要在旋转45°时做模长伸缩变换。

像素八连通域的八个像素位置可以用如下方式取到

为方便后续使用，定义

在上述坐标系下 其中k=0,1,2,3,4,5,6,7时分别表示原像素的右，右下，下，左下，左，左上，上，右上 八个位置的像素位置复数。

函数 输入为像素的位置信息，输出为二值化后的灰度值即0或者1 。

零散白像素点修复过程

在上一步执行二值化的过程中，由于使用的原始灰度图像是jpg的压缩图像，字符与背景的连接处即字符边缘的图像信息不明确，有些许的灰度跃变，即字符到背景的灰度变化不是平滑的。这样的图像二值化后，字符会有“凹坑”现象，同时，字符内部可能会有单个像素的空洞，为修补这些凹坑和空洞使字符看上去平滑而使用该步骤。

取像素的四或者八连通域的时候，会遇到像素在图片边缘的情况，因此需要根据实际情况人为扩充函数

图像区域

其中width表示图像宽度，height表示图像高度。对于图像区域外即简单地定义此时的即白色点。

取四连通域像素进行判断时，我们认为一个白点周围有三个即以上的黑点，那么该白点应当被修正为黑点；

取八连通域像素进行判断时，我们认为一个白点周围有五个及以上的黑点，那么该白点应当被修正为黑点。

所有的修正后的信息直接输出在一个新建的图像上，对原始的输入图像不直接做修改。

垂直方向去噪算法

针对图像中黑色干扰线的特征，主要为横线且线的厚度较小，对字符有影响区域的干扰线厚度大致不超过3个像素而有效字符的厚度一般在5个像素以上，该算法虽然会削薄字符但对字符的骨干影响不大，基本可以保持字符形状不被破坏，因此有该针对性的去噪算法。

这里定义一个探索深度D，以便于算法的调整。

若

则计算

则去噪修正后的灰度为

所有的修正后的信息直接输出在一个新建的图像上，对原始的输入图像不直接做修改。

八方向广度探索去噪



层2

层1

对于的像素进行如下操作：

所选取的测试像素点位于待修正像素点的八个方向即辐射角满足

实际所取的层数S，和设定的层参考阈值T为经验参数。

所有的修正后的信息直接输出在一个新建的图像上，对原始的输入图像不直接做修改。

统计排序滤波器算法

设定图像矩形模板窗口大小

其中m为模板宽，n为模板高

如果采用非矩形模板窗口，则把这个模板窗口记为

统计排序滤波器算法需要将一系列输入值从小到大排序而后选择其中一个位置的值进行输出。

设计函数 (sort and index)排序并标序，函数的输入为有穷数列，输出从小到大经过排序的有序数列

设计函数选择器，函数的输入为有序数列和选择位置，输出为实数（整数）。

像素的模板窗口区域

设计函数窗口取数器，输入模板，输出为一个数集。

则设计一个统计排序滤波器，输入像素位置，模板，选择位置，输出为一个实数（整数）。

其中i表示索引xi表示元素，元素可以重复

列表

所有的修正后的信息直接输出在一个新建的图像上，对原始的输入图像不直接做修改。

垂直投影分割

沿着图片的宽度方向逐列扫描每一列中黑色像素的个数

通过前期对图片样本的采样分析，得到一个字符存在区域的经验集合，采用该经验集合的原因在于缩小扫描时间和针对图像左侧对字符几乎无影响的噪声的干扰以简化算法达到一个合理的效果。

经验区间

通过前期对图片样本的分析，得到一组首字符开始，末字符结尾和中间字符分割的基准。这个基准通过对当前样本集的所有图片进行采样。

J(x)表示当前样本在x列的黑色像素个数是否为0。

观察此曲线，用观察法确定首字符开始的基准线和末字符结束的基准线。

此外通过在经验区间求解上述函数的极大值点。即解离散不等式组

解出解集后，通常的样本集可以直接将解集从小到大定义为基准线

如果有特殊情况，如解集中元素个数大于三个或者不足三个可以人为根据经验样本个别判定。

基准线集

经验参数：基准线距离阈值

而后对个体样本进行如下处理

求解各字符首分隔位置

解离散不等式组

如果每组不等式的解集 则否则

求解各字符末分割位置

解离散不等式组

如果每组不等式的解集 则否则

由此，对应每张图片样本可以得到集合

取，并将分割区域内的图片信息输出到一张新的图片，对原始的输入图片不做修改。

单字符图片冗余部分去除

对得到的字符分割后的单字符原始样本图片（宽度不定，高度50）样本逐行扫描每一行的黑色像素个数

根据YB的图像，可是获得切除冗余部分的首尾经验准轴

使得98%的样本字符在hcs和hce之间

换言之hcs和hce可以通过解方程

对于每一个样本的实际切割修正可以通过如下操作实现，设定一个容差阈值th

顶部切割位置解如下方程组实现

如果解集则取否则

底部切割位置解如下方程组实现

如果解集则取否则

将分割区域内的图片信息输出到一张新的图片，对原始的输入图片不做修改。

算法执行步骤：

1. 输入原始灰度图像（实现）
2. 确定原始灰度图像的二值化阈值（实现）
3. 二值化原始图像（实现）
4. 修复由于图像压缩导致二值化可能错误的点，使字符边缘平滑（实现）
5. 使用合理的去噪组合去除图像噪声（实现）
6. 通过对全体样本分析确定垂直分割的基准线（实现）
7. 对每个样本进行字符分割（实现）
8. 对全体样本分析确定切除冗余部分的分割基准线（实现）
9. 对每个样本进行冗余切除（实现）
10. 对字符样本进行大小归一化（调用image.scaleFilter，双线性插值）
11. 对样本进行标签分类（手动处理）
12. 对分类完的样本输出到标准训练格式文件（实现）
13. 使用训练输入文件进行SVM训练得到结果分类模型（调用libsvm）
14. 输入需要预测的图片并进行处理（实现）
15. 使用模型进行预测（实现）