文字提取概览

Target：输入一张原始的文章图像，可以先根据用户喜好选择多种风格转换，再将图中所有无关文字和线条剔除，保留所有黑色文字，再利用文字提取算法将图中文字分别提取，最后用红色矩形框将图中文字标记。

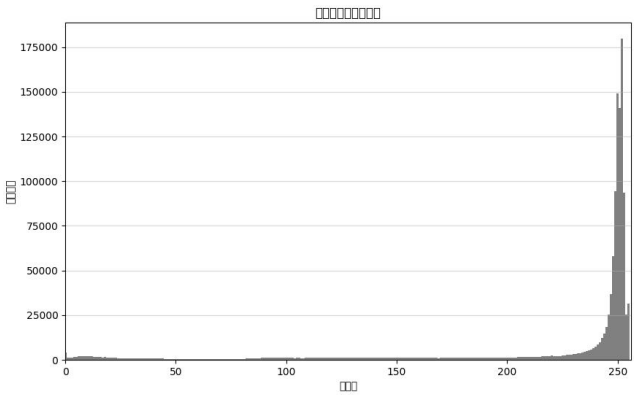
1. kmean 聚类算法提取黑色文字像素

输入的原始图像中会有多种非黑色的文字和线条，如下图

图中红色的边框属于冗余信息，我们利用 kmean 算法将图中不同颜色的像素进行分类，最终将黑色类别的像素保留。

1.1. kmean 算法思路来源

我们在黑色像素提取的初期思路是先将原图像转化为灰度图，根据灰度图中不同灰度值的分布确定黑色像素的灰度值，并保留这部分灰度值以达到保留黑色像素的效果。



以上是原图像的灰度值分布直方图。在图中显示黑色像素和红色像素的灰度值分布分布较为相似，难以找出确定的阈值以明确区分黑色像素和红色像素。因此我们决定通过更高维度的特征进一步区分黑色和其他颜色的像素，因此选择使用 kmean 聚类算法。

1.2. kmean 算法原理

K-means 算法是一种经典的无监督学习聚类算法，通过迭代将数据划分为 K 个簇，使得同一簇内的数据点尽可能相似，不同簇的数据点尽可能不同。以下是其核心原理和步骤：

1.2.1. 算法目标

最小化簇内平方和：

$$WCSS = \sum_{i=1}^{K} \sum_{x \in C_i} \|x - \mu_i\|^2$$

其中：

K：预设的簇数量。

C_i：第 ii 个簇。

μ_i：第 ii 个簇的质心（均值向量）。

1.2.2. 算法步骤

1.2.2.1. 初始化

随机选择 KK 个数据点作为初始质心

1.2.2.2. 分配阶段

将每个数据点分配到距离最近的质心所在的簇：

$$C_i = \{x: \|x - \mu_i\|^2 \leq \left\|x - \mu_j\right\|^2, \forall j \neq i\}$$

1.2.2.3. 更新阶段

重新计算每个簇的质心（即簇内所有点的均值）：

$$\mu_i = \frac{1}{|C_i|} \sum_{x \in C_j} x$$

1.2.2.4. 迭代

重复分配和更新步骤，直到满足终止条件（如质心变化小于阈值、WCSS 收敛或达到最大迭代次数）。

1.2.3. 关键细节

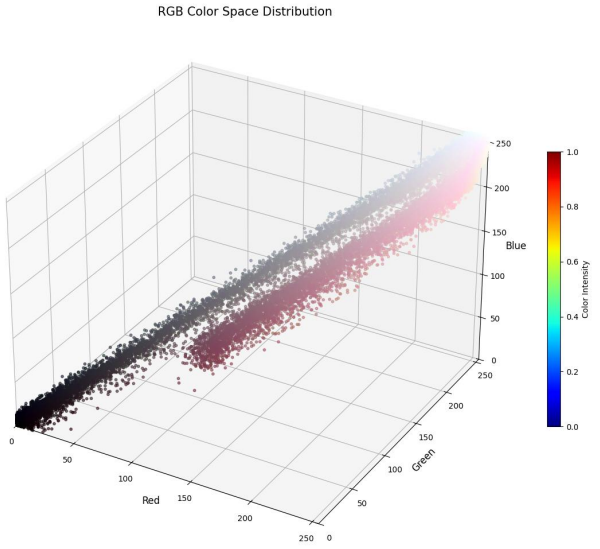
距离度量：通常使用欧氏距离（L2 范数），也可用曼哈顿距离（L1）等。

K 值选择：通过肘部法（Elbow Method）、轮廓系数（Silhouette Score）或业务需求确定。

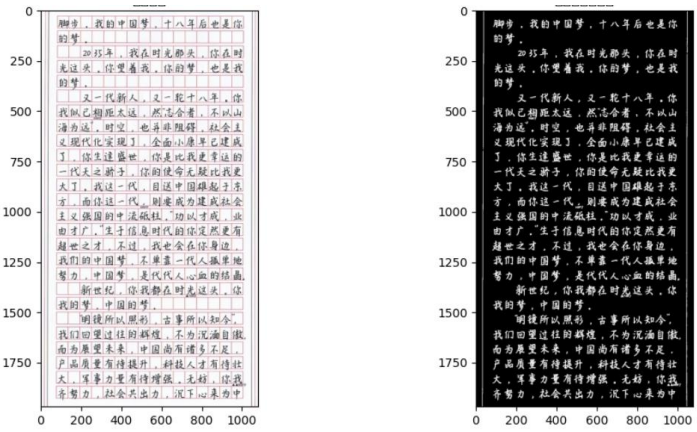
收敛性：算法保证局部最优，但不一定是全局最优（受初始质心影响）。

1.3. kmean 算法应用

对于任何一张图片，每个像素都有三个通道，通过对图像绘制三维的三通道色彩分布图，如下图大致分为两个簇。



由此可以提取原图像中最接近黑灰色的簇，将其认定为手写文字的像素。提取结果如下图：



2. 图片风格转化

我们对图像提供了三种种风格进行转化。

2.1. 图像增强

2.1.1. 方法

使用 PIL 库的 ImageEnhance 模块进行对比度、锐度和颜色增强通过 enhance_factor 参数控制增强程度。

2.1.2. 原理

对比度增强: 调整图像像素值的分布范围，使亮部更亮，暗部更暗

锐度增强: 通过边缘增强算法突出图像细节

颜色增强: 调整色彩饱和度，使颜色更鲜艳

2.1.3. 应用场景

改善曝光不足或过度的照片

增强模糊图像的细节

使褪色或低饱和度图像更生动

2.2. 素描风格

2.2.1. 方法

将图像转换为灰度，反相颜色，应用高斯模糊，再次反相，与原灰度图像进行颜色减淡混合

2.2.2. 原理

基于边缘检测和亮度差异模拟素描效果，高斯模糊模拟铅笔的柔和线条，颜色减淡混合产生素描的高光效果

2.2.3. 应用场景:

艺术创作和设计

照片转手绘效果

图像预处理(如边缘特征提取)

2.3. 水彩风格

2.3.1. 方法

使用 OpenCV 的 stylization 函数

再应用 detailEnhance 进行细节增强

2.3.2. 原理

stylization 通过非线性滤波模拟水彩的扩散效果

detailEnhance 保留重要边缘同时平滑同质区域

参数 sigma_s 控制平滑程度，sigma_r 控制颜色混合程度

2.3.3. 应用场景

数字艺术创作

照片艺术化处理

游戏和动画素材制作

3. 边缘提取及合并

3.1. 边缘提取

3.1.1. 字符区域检测

使用轮廓检测方法定位字符区域

对图像反相处理（因 OpenCV 默认检测白色轮廓）

3.1.2. 噪声过滤

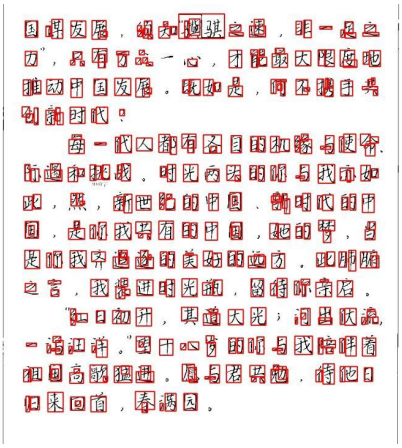
基于面积过滤

基于宽高比过滤

3.1.3. 结果标记

用红色矩形框标记有效字符区域

返回标记后的图像和轮廓信息



3.2. 边缘合并

算法思想：比较每两个红色矩形之间的距离，若小于所有矩形长和宽的平均值则对其进行合并，并标记已合并的矩形，确保不会重复标记。最后用绿色边框标记文字位置。