

暨南大学本科实验报告专用纸(附页)

暨南大学本科实验报告专用纸

课程名称 数字图像处理实验 成绩评定
实验项目名称 基于图论聚类算法的图像分割 指导教师 汤知日
实验项目编号 006 实验项目类型 设计型 实验地点 教504
学生姓名 张朋洋 学号 2022104334 学院 智能科学与工程
工程学院 专业 物联网工程 实验时间 2024 年 12 月 11 日

实验目的

1. 实现 Graph-theoretic 聚类算法，对下述图像进行分割（基于颜色 or 亮度？）。
2. 对比 k-means 与 mean shift 聚类算法，分析实验结果和各自优劣势。

实验工具

3. Python
4. cv2

实验内容

（一）实验（方法）原理

1. 图论算法分割

(1) Minimum_cut:

MinCut 的目标是将图分成两个子图，使得分割的“代价”最小。代价通常是通过图中边的权重来衡量的。具体而言，给定一个加权图，MinCut 通过切断边的集合来最小化割边的总权重，从而将图分成两个互不相交的子集。它是为了最小化切割的边的总权重。它可能导致生成的子图大小不均，分割结果可能不太理想。

(2) Normalized_cut:

Normalized Cut 是在 MinCut 的基础上对分割结果做了改进，它不仅考虑切断边的权重，还通过对子图大小的正则化来避免 MinCut 的缺陷。Ncut 试图最小化切割边的总权重，同时最大化割后两个子图的“相似性”与“均衡性”。

目标：最小化切割边的总权重，并通过一个正则化项来平衡两个子图的大小和相似性。

公式：

$$Ncut(A,B) = \frac{cut(A,B)}{assoc(A,V)} + \frac{cut(A,B)}{assoc(B,V)}$$

其中， $cut(A, B)$ 是 A 和 B 之间切断的边的权重， $assoc(A, V)$ 是 A 中节点与整个图的相似度总和， $assoc(B, V)$ 是 B 中节点与整个图的相似度总和。它避免了 MinCut 在处理图不均衡时的的问题，生成的子图更具相似性和均衡性。

暨南大学本科实验报告专用纸(附页)

2. K-means 与 mean-shift 聚类分割:

这两种方法的原理在实验五中已经给出了详尽介绍和分析, 本次实验报告不再赘述。

(二) 实验步骤:

1. Normalized-cut 分割:

(1) 将图像

(2) 发现因为计算量过于庞大, 正常思路无法计算出结果, 于是, 选择以下优化方法减小总体计算量。

<code>numpy.core._exceptions._ArrayMemoryError: Unable to allocate 228. GiB for an array with shape (175084, 175084) and data type float64</code>

表 1 计算量过大的报错

(3) 方法一: 稀疏矩阵

(4) 方法二: 用 matlab 实现

(5) 方法三: 图伸缩, 减小分辨率等

2. Minicut 分割:

(1) 存在同样原始计算量过大的问题

(2) 采用多尺度优化方法, 减小计算量

(3) 代码分为两部分:

① `build_graph`: 构建图, 包含缩放和图的初始化。

② `segment_image`: 主函数, 使用多尺度方法处理图像。

(4) 参数说明:

① `coarse_scale`: 初始粗分割的图像缩放比例 (例如 0.5 表示原始图像的 50% 尺寸)。

(5) 多尺度实现细节:

① 首先在缩小图像上进行粗略分割, 减少计算量;

② 将粗分割结果放大并细化至原始分辨率, 保证结果准确性。

(6) 优化点:

① 图像缩小时, 减少了图节点数量, 节省了内存;





② 粗分割用于引导细化分割, 减少了细分割的搜索空间。

(三) 实验结果

1. 对比图:

方法	结果图
----	-----

暨南大学本科实验报告专用纸(附页)

<p>原图</p>	
<p>基于亮度的 normalized_cut</p>	
<p>多尺度 Mini-cut</p>	
<p>Normalized-cut (论文原作者在 github 上公布的 (部分原代码))</p>	

暨南大学本科实验报告专用纸(附页)

K-means	
Mean_shift	

2. 所有实验代码及相关文件已经上传到:

[GitHub:https://github.com/zmydsg/DIP.git](https://github.com/zmydsg/DIP.git)

实验总结

(一) 分析

1. K-means 聚类

(1) 优点: 它适用于数据分布比较规则, 簇形状接近圆形或球形。在处理具有较少噪声且簇间有明显边界时表现才比较好。

(2) 缺点: 由于其对初始中心的依赖, 不同的初始点可能结果相差很大, 容易陷入局部最优~并且对噪声敏感。

(3) 在一个简单的二维数据集(如具有明确边界的圆形簇)中, K-means 通常能够很好地分割簇。但如果数据簇形不规则, 或存在噪声, K-means 的表现会明显下降。

2. Mean_shift

(1) 优点: 它不受簇的形状限制, 具有较强的鲁棒性, 受噪声影响较小。不需要预设簇的数量。

(2) 缺点: 算法计算复杂度较高。对带宽参数较为敏感。

(3) 总得从实验来说, 它适合具有不规则形状的簇或数据存在噪声时使用。并且可以自适应地选择簇的数量。

3. 多尺度 Mini_cut:

(1) 图像缩小时, 减少了图节点数量, 节省了内存;

(2) 粗分割用于引导细化分割, 减少了细分割的搜索空间。

(3) 算法过程:

暨南大学本科实验报告专用纸(附页)

- 构建图：将图像的每个像素点视为图中的节点，图的边表示像素之间的关系（如颜色相似度、空间邻近等）。

- 权重计算：计算每一对像素间的边权，常见的方法有基于像素值差异、颜色信息、纹理特征等。边权较小的像素对，表示它们属于不同的区域，反之表示它们属于相同的区域。

- 最小割求解：使用最小割算法（如 Max-Flow Min-Cut 定理）计算图的最小割。常用的算法包括 Ford-Fulkerson 算法和 Edmonds-Karp 算法。

- 分割图像：通过最小割将图像分成两个部分，这两个部分代表了不同的图像区域（如前景和背景）。

（4）优缺点：

- 优点：

全局优化：Min-Cut 算法能够考虑整个图像的全局信息，提供较为精确的分割结果。

处理复杂图像：能够处理具有复杂背景和细节的图像。

- 缺点：

计算复杂度高：最小割问题的计算复杂度较高，尤其是图像分辨率较高时，算法的计算量会大幅增加。

需要手动设定参数：图像分割中涉及到的权重计算和阈值选择可能需要根据具体情况调整。

4. Normalized_cut

（1）优点：

全局优化：Ncut 考虑了全局信息，能够有效避免局部最小值，提供更合理的分割结果。

避免过度分割：通过标准化割的代价，Ncut 能够避免传统 Min-Cut 算法中因某一部分的割代价过小而导致过度分割的现象。

（2）缺点：

计算复杂度高：由于涉及到特征分解和谱聚类，Ncut 的计算复杂度较高，对于大规模图像可能需要较长的计算时间。

需要调节参数：Ncut 算法的效果依赖于图像的相似度度量方式和其他参数的设置。

（二）结论

1. 在所有可能的割中，最小割是指边的权重和最小的割。

2. N_cut 的核心目标是最小化分割的代价，同时避免割掉大部分图的节点（即避免某个部分过小）。

3. 关于 mean_shift 与 K_means 的分析与总结在实验五已经很详尽了，这里就不赘述了。