



# 模式识别课程报告

基于数据驱动的特征提取的道路分割实验研究

班级 \_\_\_\_\_ 模式识别 (1 班) \_\_\_\_\_

学号 \_\_\_\_\_ 23060189 \_\_\_\_\_

姓名 \_\_\_\_\_ 朱睿 \_\_\_\_\_

2023 年 12 月 18 日

# 一、实验目的

## 1、实验目标

利用至少两种不同的特征提取算法，对第一次作业中的道路图像中的路面进行特征建模。要求如下

- 利用任何一款流形学习方法（可直接使用，无需自己编程）对两种特征的有效性进行可视化对比分析
- 分别利用自己提取的特征对道路进行识别，并评估流形学习可视化得到的特征有效性评估结果与实际道路识别效果之间的相关性

## 2、实验涉及到的学习内容

- 基于数据驱动的特征提取方法
- 使用流形学习方法对提取到的特征进行可视化

# 二、实验具体完成情况

## 1、实验总体方案设计

实验需要利用聚类算法对道路进行二值化分割，具体分为如下 6 步

- **数据准备**: 构建真值图像，用于后续的算法评估
- **数据预处理**: 对于初始图像，采用中值滤波、均值滤波、高斯滤波等方式对图像进行预处理
- **特征提取**: 采用不同方法进行特征提取，用于后续的计算，例如原始 RGB 通道特征、主成分分析 (PCA)、核主成分分析 (KPCA)、线性判别 (LDA)、字典学习 (Dictionary Learning) 等
- **流形学习方法可视化**: 采用 t-SNE (t-Distributed Stochastic Neighbor Embedding)、等度量特征映射 (Isometric Feature Mapping, Isomap)、局部线性嵌入 (Locally Linear Embedding, LLE) 等方法对特征提取效果进行可视化
- **分类算法 (道路分割)**: 采用不同的算法进行道路分割，并进行比对，例如支持向量机 (SVM)、
- **数据后处理**: 采用开运算、闭运算及二者结合的方式对分割后的道路进行进一步优化，提升分割效果

由于本人对 python 较为感兴趣，且后续科研工具主要为 python 语言，因此报告主要基于 python 完成上述任务。

## 2、具体技术途径

针对实验方案中的 6 个步骤，基本原理和实现方法如下：

### (1) 数据准备：构建真值图像，用于后续的算法评估

**SAM-Adapter** 近期大模型的涌现给 AI 研究带来显著的发展，META 的工作 Segment Anything (SAM)，就是其中一个为图像分割任务设计的基础大模型。SAM 是一种交互型的图像分割大模型，通过提供的 prompt 如点、框、文本描述等粗略的提示，就可以分割出图像中指定的目标，其 demo 的效果十分惊艳。然而在某些特殊场景的图片上并不会带来如此惊艳的效果，可能是由训练数据的差异性导致。

针对于图像分割特定任务，我们采用 SAM-Adapter，其通过设计一个 Adapter 模块，使得在不微调 SAM 网络的情况下，将领域特定的信息或视觉提示注入到分割网络中，从而提高 SAM 在特定任务上的性能。

**评估指标** 在实验中，我们构建了 4 个评估指标，分别为准确率 (Accuracy)、精确率 (Precision)、召回率 (Recall)、F1 值，其计算方法如下

我们采用 TP (True Positive, 真阳性)、FP (False Positive, 假阳性)、TN (True Negative, 真阴性) 和 FN (False Negative, 假阴性) 来描述不同结果的样本的数量。

**准确率 (Accuracy)**：准确率衡量模型预测的整体正确性，即正确预测的样本数与总样本数之比。准确率的计算方法如下：

$$\text{Accuracy} = \frac{TP + TN}{TP + TN}$$

**精确率 (Precision)**：精确率衡量模型在预测为正类的样本中的正确性，即真阳性样本数与所有预测为正类的样本数之比。精确率的计算方法如下：

$$\text{Precision} = \frac{TP}{TP + FP}$$

精确率用于评估模型的预测准确性，高精确率表示模型将负样本误判为正样本的可能性较低。

**召回率 (Recall)**：召回率衡量模型对正类样本的预测覆盖率，即真阳性样本数与所有真实正类样本数之比。召回率的计算方法如下：

$$\text{Recall} = \frac{TP}{TP + FN}$$

召回率用于评估模型的查全率，高召回率表示模型正确检测到了大部分正样本。

**F1 值**: F1 值是精确率和召回率的调和平均值，综合考虑了模型的准确性和覆盖率。F1 值的计算方法如下：

$$F1 = \frac{2 \times (Precision \times Recall)}{Precision + Recall}$$

F1 值的范围在 0 到 1 之间，值越高表示模型在准确性和覆盖率之间取得了更好的平衡。

## (2) 数据预处理：对于初始图像，采用不同滤波方式对图像进行预处理

**中值滤波 (Median Filtering)** 中值滤波是一种非线性滤波方法，用于图像去噪。它的原理是将像素点周围的邻域像素值进行排序，然后用中间值来代替当前像素的值。中值滤波对于去除图像中的椒盐噪声或斑点噪声效果良好，因为它能够有效地消除异常值的影响。

**均值滤波 (Mean Filtering)** 均值滤波是一种线性滤波方法，用于平滑图像并减少噪声。它的原理是将像素点周围的邻域像素值取平均值，然后用该平均值来代替当前像素的值。均值滤波对于高斯噪声等均值为零的噪声效果较好，但在去除椒盐噪声等具有较高峰值的噪声时效果较差。

**高斯滤波 (Gaussian Filtering)** 高斯滤波是一种线性滤波方法，用于图像平滑和去噪。它的原理是将像素点周围的邻域像素值按照高斯分布进行加权平均，然后用加权平均值来代替当前像素的值。高斯滤波对于平滑图像并保留边缘信息效果较好，因为它能够考虑到像素之间的空间关系。

**双边滤波 (Bilateral Filtering)** 双边滤波是一种非线性滤波方法，用于图像平滑和去噪。它的原理是同时考虑像素之间的空间距离和像素值之间的差异，通过加权平均来计算当前像素的值。双边滤波在保留图像边缘信息的同时能够有效地去除噪声，适用于保持图像细节和边缘清晰的应用场景。

**均值迁移滤波 (Mean Shift Filtering)** 均值迁移滤波是一种基于样本点密度的非线性滤波方法，用于图像平滑和分割。它通过在像素点周围的邻域内计算样本点的均值偏移来对图像进行平滑处理。均值迁移滤波对于保持图像细节和纹理信息的同时能够有效地去除噪声和进行图像分割。

这些滤波方法在图像处理领域中被广泛应用，每种方法都有其适用的场景和特点。选择合适的滤波方法取决于图像的特性、噪声类型以及对图像细节和边缘保留的要求。

## (3) 特征提取：采用不同方法进行特征提取，用于后续的计算

**原始 RGB 通道特征** 原始 RGB 通道特征是通过直接使用图像的原始 RGB 颜色通道作为特征表示的方法。对于一张 RGB 图像，它包含了红色 (R)、绿色 (G) 和蓝色 (B)

三个颜色通道。原始 RGB 通道特征将每个像素点的 RGB 值作为特征向量的元素。假设图像的尺寸为  $M \times N$ , 则特征向量的维度为  $M \times N \times 3$ 。

**主成分分析 (PCA)** 主成分分析是一种常用的降维技术, 通过线性变换将高维数据转换为低维表示。它通过找到数据中最大方差的方向 (主成分) 来实现降维。给定一个包含  $N$  个样本的数据集  $X$ , 其中每个样本  $x_i \in \mathbb{R}^d$ , PCA 的目标是找到投影矩阵  $W$ , 将原始数据映射到新的低维空间, 使得投影后的数据具有最大的方差。

PCA 的数学表达式如下:

- 计算数据的协方差矩阵  $C$ :

$$C = \frac{1}{N} \times \sum (x_i - \mu)(x_i - \mu)^T$$

其中,  $\mu$  是数据的均值向量。

- 对协方差矩阵  $C$  进行特征值分解得到特征值  $\lambda_1, \lambda_2, \dots, \lambda_k$  和对应的特征向量  $v_1, v_2, \dots, v_k$ 。
- 将特征向量按照对应的特征值降序排列, 选择前  $k$  个特征向量组成投影矩阵  $W$ 。
- 将原始数据  $x$  映射到低维空间:

$$y = W^T(x - \mu)$$

**核主成分分析 (KPCA)** 核主成分分析是对传统 PCA 方法的扩展, 用于处理非线性关系的数据降维。KPCA 通过使用核函数将数据映射到高维特征空间, 然后在高维空间中进行 PCA 分析。这种非线性映射使得 KPCA 能够提取非线性特征。

KPCA 的数学表达式如下:

- 计算核矩阵  $K$ :

$$K_{ij} = k(x_i, x_j)$$

其中,  $k(\cdot, \cdot)$  是核函数,  $x_i$  和  $x_j$  是数据集中的样本。

- 对核矩阵  $K$  进行中心化, 得到中心化核矩阵  $\tilde{k}$ 。
- 对中心化核矩阵  $\tilde{k}$  进行特征值分解, 得到特征值  $\lambda_1, \lambda_2, \dots, \lambda_k$  和对应的特征向量  $\alpha_1, \alpha_2, \dots, \alpha_k$ 。
- 将原始数据  $x$  映射到低维空间:

$$y = \sum \alpha_i k(x, x_i)$$

**线性判别分析(LDA)** 线性判别分析是一种用于特征提取和降维的线性分类方法。LDA 通过寻找投影方向，使得在降维后的空间中，不同类别的样本尽可能分开，同类样本尽可能接近。

假设有  $N$  个样本，每个样本  $x_i \in \mathbb{R}^d$ ，对应的类别标签为  $y_i \in 1, 2, \dots, C$ ，其中  $C$  表示类别的数量。

LDA 的数学表达式如下：

- 计算类别均值向量：

$$\mu_i = \frac{1}{N_i} \times \sum_j x_j$$

其中， $N_i$  是类别  $i$  的样本数量。

- 计算类内散度矩阵  $S_w$  和类间散度矩阵  $S_b$ ：

$$S_w = \sum_i \sum_x (x_j - \mu_i)(x_j - \mu_i)^T$$

$$S_b = \sum_i N_i (\mu_i - \mu)(\mu_i - \mu)^T$$

其中， $\mu$  是所有样本的均值向量。

- 对矩阵  $S_w^{-1} S_b$  进行特征值分解，得到特征值  $\lambda_1, \lambda_2, \dots, \lambda_k$  和对应的特征向量  $w_1, w_2, \dots, w_k$ 。
- 将特征向量按照对应的特征值降序排列，选择前  $k$  个特征向量组成投影矩阵  $W$ 。
- 将原始数据  $x$  映射到低维空间：

$$y = W^T x$$

**字典学习 (Dictionary Learning)** 字典学习是一种无监督学习方法，用于从数据中学习一组基础原子（字典），以表示数据。字典学习的目标是通过最小化数据的重构误差，学习出能够有效表示数据的原子集合。给定一个包含  $N$  个样本的数据集  $X$ ，其中每个样本  $x_i \in \mathbb{R}^d$ ，字典学习的目标是找到一个字典  $D = [d_1, d_2, \dots, d_k]$ ，其中  $d_j \in \mathbb{R}^d$  为字典的原子，以及一个稀疏编码矩阵  $C \in \mathbb{R}^{k \times N}$ ，使得  $X \approx DC$ ，其中  $k$  是字典的大小。

字典学习的数学表达式如下：

- 初始化字典  $D$  和稀疏编码矩阵  $C$ 。
- 迭代优化以下两个步骤直至收敛：
  - 固定字典  $D$ ，更新稀疏编码矩阵  $C$ ：

$$\min ||X - DC||_2^2 + \lambda ||C||_1$$

$\min \|X - DC\|^2 + \|\mathbf{C}\|$  其中,  $\lambda$  是控制稀疏性的参数。

b. 固定稀疏编码矩阵  $C$ , 更新字典  $D$ :

$$\min \|X - DC\|_2^2 \quad s.t. \|d_j\|_2 \leq 1, \forall j$$

通过迭代优化, 字典学习可以学习到数据的稀疏表示, 以及适合于表示数据的原子字典。字典学习在图像去噪、图像压缩、图像恢复等任务中有广泛应用, 能够自动学习到有效的特征表示。

#### (4) 流形学习方法可视化: 采用不同方法对特征提取效果进行可视化

流形学习 (Manifold Learning) 是一种无监督学习方法, 用于从高维数据中发现其潜在的低维流形结构。在高维空间中, 数据可能呈现复杂的非线性关系, 而流形学习的目标是通过学习数据的流形结构, 将高维数据映射到低维空间, 以便更好地理解和可视化数据。以下是三种常见的流形学习方法的介绍:

**t-SNE (t-Distributed Stochastic Neighbor Embedding)** t-SNE 是一种流行的降维和可视化方法, 通过保留数据样本之间的相似性关系, 将高维数据映射到二维或三维空间。t-SNE 的核心思想是, 在低维空间中, 通过高斯分布来表示样本之间的相似性, 而在高维空间中, 则通过  $t$  分布来表示相似性。t-SNE 使用梯度下降的方法最小化高维空间和低维空间之间的 Kullback-Leibler (KL) 散度, 从而实现降维和可视化。

**Isomap (Isometric Feature Mapping)** Isomap 是一种基于流形学习的降维方法, 它通过保持数据点之间的测地距离来捕捉数据的流形结构。Isomap 中的关键步骤是构建数据点之间的邻接图, 并计算它们之间的最短路径距离。然后, 使用多维缩放 (MDS) 算法将最短路径距离映射到低维空间, 得到最终的降维结果。

**LLE (Locally Linear Embedding)** LLE 是一种非线性的流形学习方法, 它假设数据在局部区域内近似线性, 并通过保持局部线性关系来进行降维。LLE 的主要步骤包括两个阶段: 首先, 对每个数据点找到其最近的邻居; 然后, 通过最小化数据点与其邻居之间的线性重构误差, 计算低维表示。LLE 的优点是能够捕捉复杂的流形结构, 但对于大规模数据集可能计算复杂度较高。

这些流形学习方法在降维和可视化高维数据时都具有一定的优势和适用性。选择合适的方法取决于数据的特点以及所需的降维效果。

#### (5) 分类算法 (道路分割): 采用不同的算法进行道路分割, 并进行比对

#### (6) 数据后处理: 采用不同方式对分割后的道路进行进一步优化, 提升分割效果

数据后处理, 我们采用形态学处理的方式。形态学方法是一种基于数学形态学理论的图像处理技术, 用于图像的形状分析和特征提取。形态学方法主要应用于二值图像,

通过定义结构元素（也称为模板或核）和基本的形态学操作来改变图像的形状和结构。

形态学方法包括一系列的形态学操作，其中最常用的是开运算（Opening）和闭运算（Closing）。

**开运算（Opening）** 开运算是先进行腐蚀操作，然后再进行膨胀操作。它主要用于去除二值图像中的小型噪点和细小的边缘连接。开运算能够平滑图像，同时保持主要对象的形状和结构。具体步骤如下：

- 腐蚀：使用结构元素对图像进行腐蚀操作，使边界向内缩小。
- 膨胀：对腐蚀结果进行膨胀操作，恢复对象的大小并平滑边界。

开运算的特点：

- 去除小型噪点和细小边缘连接。
- 平滑对象的形状。
- 保持主要对象的大小和结构。

**闭运算（Closing）** 闭运算是先进行膨胀操作，然后再进行腐蚀操作。它主要用于填充图像中的小型空洞和细小的断开区域。闭运算能够平滑图像，同时保持主要对象的形状和结构。具体步骤如下：

- 膨胀：使用结构元素对图像进行膨胀操作，填充空洞和连接断开的区域。
- 腐蚀：对膨胀结果进行腐蚀操作，恢复对象的大小并平滑边界。

闭运算的特点：

- 填充小型空洞和细小断开区域。
- 平滑对象的形状。
- 保持主要对象的大小和结构。

**先开运算后闭运算** 这种操作顺序的特点是先将较小的噪声和细小结构从图像中去除，然后再对图像进行平滑和连接，以保持主要对象的形状。

**先闭运算后开运算** 这种操作顺序的特点是先将空洞和断开区域填充和连接，然后再对图像进行平滑和去除细小结构，以保持主要对象的形状。

### 三、实验结果与分析

#### (1) 数据准备：真值图像构建

我们利用成熟的 SAM-Adapter 方法（这是一种基于预训练大模型的图像分割方法，具备很好的 0 样本推理能力），对图像道路进行分割，可以得到道路的 mask 标签，如图1所示

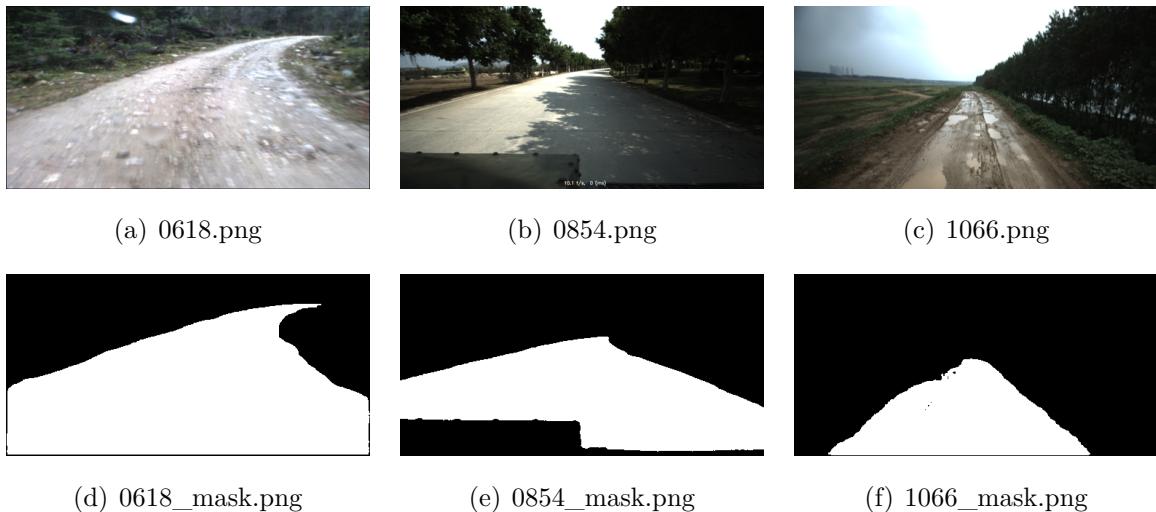


图 1: 道路 mask 标签示意图

#### (2) 数据预处理：滤波

在先前的实验中，我们发现在分割结果中，总是会有一些噪点，经过分析，我们发现：道路通常是较为亮的点，即 RGB 的值均比较高的点，而在非道路中，也可能存在若干点亮度也较高，其与道路样本更加接近，使得可能把一些小的孤立点分割为道路；另一方面，道路中也存在一些颜色较暗的点，其与非道路样本更加接近，使得道路上的一些暗点可能被分割为非道路。

为此，我们计划采用滤波的方式，对原始图像进行预处理。一方面，非道路的亮点会与周围的非道路样本更为接近，降低其被分割为道路的可能性；另一方面，道路的暗点会与周围的道路样本更加接近，增加其被分割为道路的可能性。

在实验中，采用的滤波方式有中值滤波、均值滤波、高斯滤波、双边滤波、均值迁移滤波，我们设置的滤波参数如表 1 所示，原先我们计划对于滤波设置不同的参数，对不同滤波参数也进行比较，考虑到实验量较大，且本次实验的核心不在于此，因此，我们精心为每一种滤波仅挑选了一组参数进行实验。

图 2 展示了在表 1 参数设置下，滤波前后的图像对比，具体滤波形式和图像类型均在图中进行了详尽的描述。

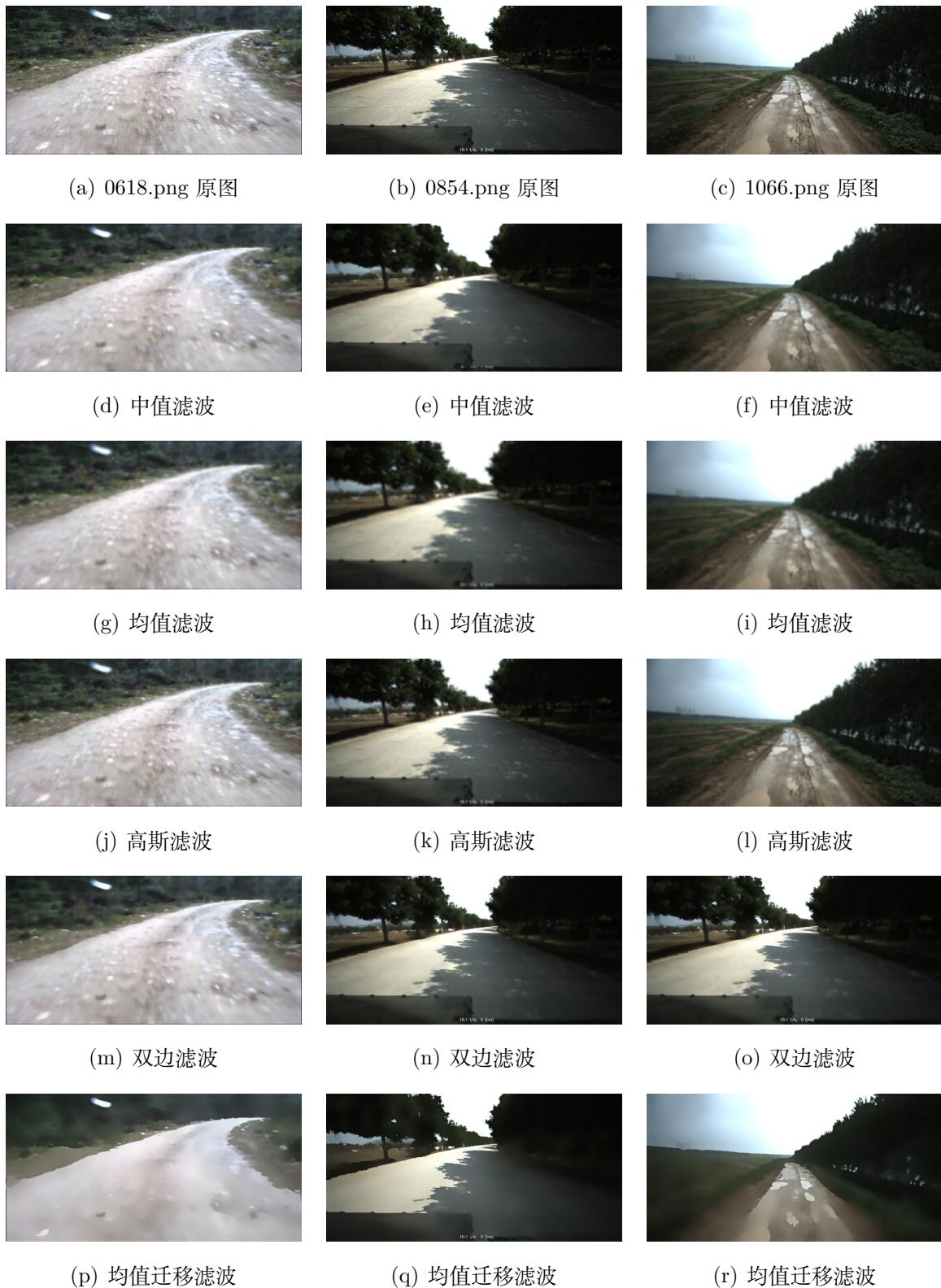


图 2: 原始图像滤波结果

表 1: 滤波实验参数设置

方法	参数设置
中值滤波	ksize=5
均值滤波	ksize=(5,5)
高斯滤波	ksize=(5,5)
双边滤波	d=9, sigmaColor=75, sigmaSpace=75
均值迁移滤波	sp=10, sr=50

### (3) 特征提取: 采用 PCA、KPCA、LDA、Dictionary\_Learning

在本次实验中, 我们主要采用了主成分分析 (PCA)、核主成分分析 (KPCA)、线性判别分析 (LDA) 和字典学习 (Dictionary Learning), 在实验中, 我们设置不同的参数, 具体实验设置如表 2 所示

表 2: 特征提取实验参数设置

方法	参数设置
Original RGB	None
PCA	n_components=1,2
KPCA	n_components=1,2 kernel=rbf, poly, sigmoid, cosine
LDA	n_components=1
Dictionary Learning	n_components=1,2

LDA 的参数  $n\_components$  只能为 1, 这是因为其在运算中  $n\_components \leq \min(features, n\_classes)$

通过计算, 我们可以得出, 总共对  $1 + 2 + 2 \times 4 + 1 + 2 = 14$  组特征提取参数进行实验对比。加之前面有 5 种滤波方式, 加上原图一共有 6 种不同的输入, 那么总共有  $6 \times 14 = 84$  种不同的实验设置。

### (4) 流形学习方法: 评估特征提取效果

在流形学习中, 我们采用 t-SNE, ISOMAP, lle 三种方法进行可视化特征提取效果的可视化展示, 由于 sklearn 是在 CPU 条件下运行, 运算速度特别慢, 因此对于这三种方法, 我们只针对训练集数据的特征提取效果进行可视化展示。

通过查阅资料, 我们发现关于如何高效使用 t-SNE 的网站([How to Use t-SNE Effectively](#)), 在这个网站中, 可以探究不同样本分布情况下, 不同参数对于 t-SNE 的效果展现。

在后续的调研中，我们发现前人在 t-SNE 的加速上做了很多工作，我们在 github 上找到了[tsne-cuda 库](#)，该库对于 t-SNE 进行了显著的加速，加速效果如图 3 所示

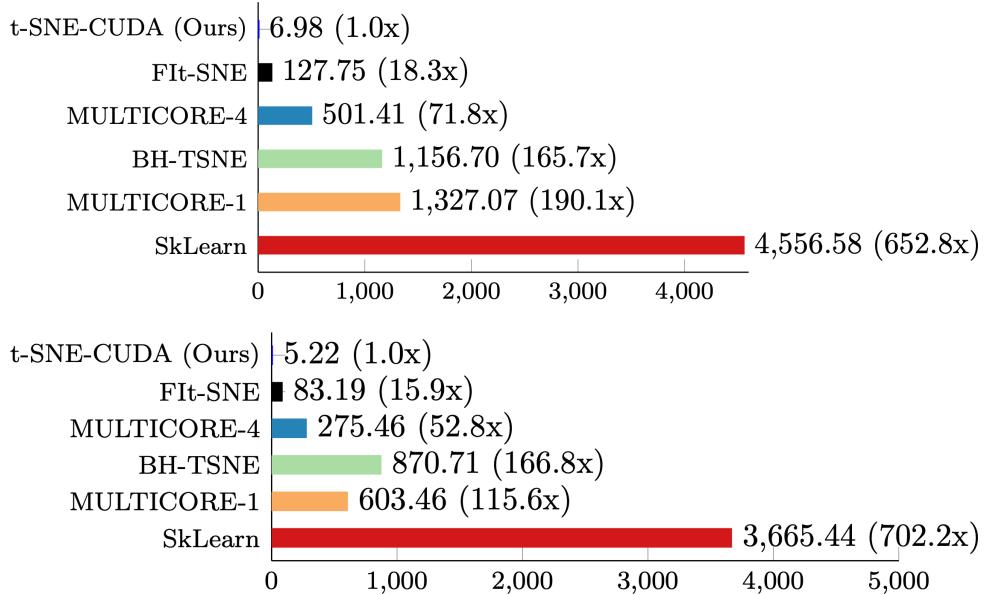


图 3: tsnecuda 在 MINIST 和 CIFAR 数据集上与其他库运算速度的对比

原本以为，可以采用 tsnecuda 进行加速，但是 tsnecuda 存在很多问题，其中之一就是不能用 pca 作为初始化。通过本地的实验对比，我们发现其效果远不如 sklearn。从图 4 中，我们不难发现，sklearn 上的效果明显要优于 tsnecuda。

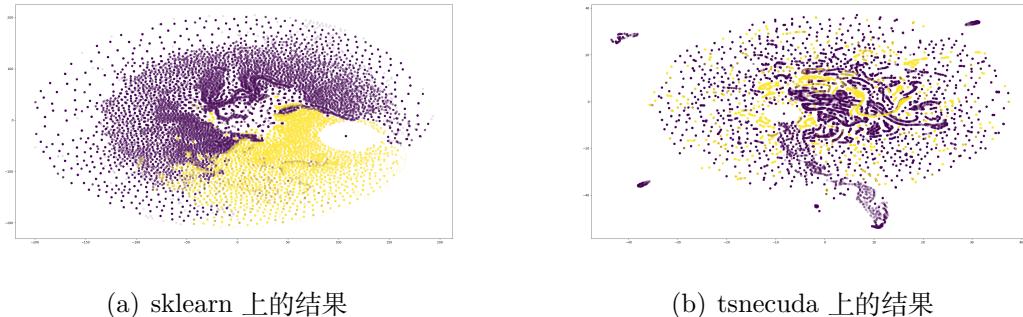


图 4: sklearn 和 tsnecuda 结果对比

从图 4 中，我们发现，sklearn 上能够明显分出紫色和黄色两个类别，但是 tsnecuda 上，黄色和紫色混杂在了一起，区分的效果明显差于前者。

抛弃了 tsnecuda 之后，通过调研，我们又发现了[cuML 库](#)，cuML 是一套实现机器学习算法和数学原语函数的库，与其他 RAPIDS 项目共享兼容的 api。cuML 使数据科学家、研究人员和软件工程师能够在 gpu 上运行传统的表格式 ML 任务，而无需深入 CUDA 编程的细节。在大多数情况下，cuML 的 Python API 与 scikit-learn 中的 API 相匹配。对于大型数据集，这些基于 gpu 的实现可以比 CPU 实现快 10-50 倍。

### (5) 分类算法：道路分割

#### (6) 数据后处理：采用形态学方法优化结果

#### 四、实验心得与体会

本次实验主要完成了两个方面的内容，利用 KMeans 和 DBSCAN 对道路进行分割。

在使用 KMeans 进行分割时，我们设置了不同的聚类数目，从 2 到 9，聚类的效果在 3 和 4 的时候达到比较好的状态。

在使用 DBSCAN 进行分割时, 我们对不同的  $\text{eps}$  (0.1,0.5,1.0), 不同的  $\text{min\_samples}$  (1,2, $\cdots$ ,9) 进行实验, 发现当  $\text{eps}$  为 0.1 和 0.5 时, 在  $\text{min\_samples}$  为 5 左右的效果比较好, 而当  $\text{eps}$  为 1.0 时, 整体分类效果较差, 几乎不能看见道路。

## 五、存在的主要问题和建议

- 使用 DBSCAN 求解时，运算速度明显慢于 KMeans。
  - 在 KMeans 实验中，可以采用选取若干道路的样本点，计算其中心，然后与聚类中心进行距离的判别，设置阈值，实现对于道路判别；由于我们道路拍摄视角较为固定，可以设置一个权重图，其为一个等腰三角形，位于图像正中间，尝试采用加权的 KMeans 聚类方法进行道路分割并实现对类别的判别。
  - 在 DBSCAN 实验中，使用了不同的 eps 和 min\_samples 进行实验，但是没有很好的对结果进行解释。

## 六、核心代码展示

```
def get_label(img, n_clusters, init='random'):  
    img_h, img_w, img_c = img.shape  
    img_data = img.reshape(img_h * img_w, img_c)  
    cluster = KMeans(n_clusters=n_clusters, init=init).fit(img_data)  
    label = cluster.labels_  
    label = label.reshape(img_h, img_w)  
    return label
```