Machine Learning for Visual Computing Coursework指导

L.C. 2024.10

要求

需要提交zip文件，包含ipynb文件（回答问题和修改代码）和models文件夹（模型权重）

前序工作

共有5个问题

[问题 1]： 线性拟合（10 分）简单

[问题 2]： 图像检索（15 分）中等

[问题 3]： 图像去噪（30 分）中等

[问题 4]： 纹理合成（15 分）

[问题 5]： 神经隐式表示（30 分）

前序准备（逐一运行已编写好的代码即可）包括：

1. 需要提前下载所需的数据集（***IRIS***、***MNIST***、***CelebA***、***NoisyOCR***）
2. 环境配置与依赖

- 主要使用numpy、matplotlib、scipy、sklearn、PIL、torch等Python库
- 可以在本地或Google Colab环境中运行
- 设置了随机种子(RAND_ST = 42)确保实验结果可复现

3. 数据集准备 实验涉及三个主要数据集:

- IRIS数据集:用于分类任务的经典数据集,包含3个类别的鸢尾花数据
- MNIST数据集:手写数字图像数据集,包含0-9共10个类别
- 自定义数据集:需要从UCL服务器下载的cw_datasets.zip

4. 数据预处理

- IRIS和MNIST数据集都使用了train_test_split进行划分,训练集占90%,测试集占10%
- MNIST数据进行了归一化处理(除以255)
- 提供了灵活的类别数量选择功能(num_classes参数)

5. 可视化工具 提供了多个可视化函数:

- plot_correlation():用于展示变量之间的相关性；plot_eigenvecs():用于可视化特征向量；plot_images(): 用于显示图像数据；plot_confusion_matrix():用于展示分类结果的混淆矩阵

6. 评估指标 包含了多个性能评估函数:

- true_positive()；true_negative()；false_positive()；false_negative() 这些函数可用于计算分类任务的各项评价指标

7. 文件管理

- 提供了灵活的文件路径配置，支持本地和Google Drive两种方式
- 包含了自动下载和解压数据集的功能

问题1 线性拟合（10 分）

学习链接

a) 实现正则方程求解函数 `nsolve`，输入矩阵 X 和目标向量 y，并返回优化权值 w。用你自己的模拟数据/随机创建的数据测试你的代码。(5 分)

1. `nsolve` 函数实现:
   - 需要实现正规方程(Normal Equation)的求解
   - 正规方程的公式是: $w = (X^T X)^{-1} X^T y$
   - 使用 `numpy` 库来进行矩阵运算
2. 测试代码:
   - 需要创建一些随机的输入数据 X 和对应的标签 y
   - 使用 `nsolve` 函数来计算权重 w
   - 使用这个 w 来预测 y,并与真实的 y 进行比较

b) 实现 `line_fit(X,y)`，拟合输入数据的线性函数。在以下任务中测试你的代码：使用其余三个变量作为输入（`sepal length (cm)`, `sepal width (cm)` 和 `petal width (cm)`），通过线性拟合预测 ***IRIS*** 数据集的 `petal length (cm)`。报告验证集上的 $L2$ 损失，并绘制图表显示 y 与您对测试集的预测之间的相关性。(2分)

1. `line_fit` 函数实现：
   - 使用1a的 `nsolve` 函数计算权重
   - 计算预测值和 $L2$ 损失
2. 在 ***IRIS*** 训练集上测试：
   - 使用三个特征（除了花瓣长度）作为输入来预测花瓣长度
   - 计算并打印训练集上的 $L2$ 损失
3. 在 ***IRIS*** 测试集上测试：
   - 使用训练好的模型在测试集上进行预测
   - 计算并打印测试集上的 $L2$ 损失
4. 绘制真实值与预测值的相关性图：
   - 使用散点图展示真实值和预测值的关系

c) 实现 `poly_fit(X,y)`，该函数应拟合输入数据的二度多项式。在以下任务中测试您的实现：使用多项式预测 ***IRIS*** 数据集的 `petal length (cm)`，将其余三个变量作为输入（`sepal length (cm)`, `sepal width (cm)` 和 `petal width (cm)`）。二次多项式应考虑所有可能的成对项，即在有两个输入变量 x 和 y 的情况下，$w_1x^2 + w_2xy + w_3y^2 + w_4x + w_5y + w_6$。报告验证集上的 $L2$ 损失，并绘制图表显示 y 与你在测试集上的预测之间的相关性。(3分)

1. `poly_fit` 函数实现：
   - 创建二次多项式特征，包括所有可能的两两组合。
   - 使用1a的 `nsolve` 函数计算权重。
   - 计算预测值和 $L2$ 误差。

2. 后续同1b

问题2 图像检索（15分）

图像检索资料总结

a) 只使用torchvision和torch库，实现四种不同的特征空间：rgb距离、直方图距离、vgg特征和vgg特征gram矩阵。（**4 分**）
代码向量化处理(使用batch处理替代循环)。（**1 分**）

1. 首先，实现VGG19模型的加载
2. 实现`get_features`函数

   img: PIL图像或图像列表 PIL image or list of images

   mod: 特征提取模式 Feature extraction mode

   1: RGB特征 RGB features

   2: 颜色直方图 Color histogram

   3: VGG特征 VGG features

   4: VGG Gram矩阵 VGG Gram matrix

   返回： features: 提取的特征(numpy数组)
3. 实现`find_img`函数

   RGB距离：直接使用图像的像素值作为特征

   直方图距离：使用图像的颜色直方图作为特征

   VGG特征：使用VGG19网络提取的深度特征

   VGG特征Gram矩阵：使用VGG19特征的Gram矩阵，捕捉纹理信息
4. 测试

b) 实现一个算法来从数据库中检索最相似图像的索引。使用一个隐藏的查询和隐藏的数据库来测试（隐藏的数据集图像大小从 (64, 64, 3) 到 (256, 256, 3)）。确保你的代码适用于所有数据库大小和分辨率。测试将获得一个包含 N 个条目和 M 个查询的单个数据库 (**4 分**)
代码向量化处理(使用batch处理替代循环)。（**1 分**）

1. 适应不同大小和分辨率：使用`transforms.Resize((224, 224))`将所有图像调整为相同的大小，这样可以处理从(64, 64, 3)到(256, 256, 3)的所有图像
2. 矢量化处理：
   - 在`get_features_q2b`函数中，我们使用`torch.stack`来批量处理图像
   - 在`find_img_q2b`函数中，我们一次性获取所有数据库图像的特征，并使用矩阵运算来计算距离
3. 适用于所有数据库大小：我们的实现不依赖于固定的数据库大小，可以处理任意数量的图像
4. 使用VGG19特征：我们使用预训练的VGG19模型提取特征，这种方法对于图像检索任务有很好的表现

c) 讨论并记录， M-times-N 结果中的每一个都是什么。（**5 分**）

1. 分析每种特征提取方法的优缺点
2. 解释为什么会得到这样的检索结果

问题3 降噪 (30 分)

a) 实现 `denoiseGauss(image)`，使用5×5高斯滤波器对`cw_datasets/Filtering/`文件夹下的`noisy_image.png`进行去噪。（**5分**）

1. 实现两个核心函数：
   - `gkern`: 生成5×5的高斯核，可指定标准差
   - `denoise_gauss`: 对输入图像应用高斯滤波
2. 在'cw_datasets/Filtering/'目录下的'noisy_image.png'上测试实现效果

b) 使用Pytorch实现卷积神经网络来对图像进行去噪。我们提供了用于训练和测试的128×128的噪声和无噪声图像对，位于`cw_datasets/Denoising/`文件夹下。你可以使用Pytorch的所有基础设施。网络应具有足够的深度和复杂度以保证良好的收敛效果。请在每一层后使用ReLU非线性激活函数。（**20分**）

1. 数据集实现：
   - 创建继承自`Dataset`的`DenoisingDB`类
   - 处理'cw_datasets/Denoising/'中的128×128图像块
   - 实现数据加载和预处理功能
2. 网络架构：
   - 设计编码器-解码器CNN结构
   - 包含足够的网络深度以确保良好收敛
   - 每层后添加ReLU激活函数
   - 确保正确处理通道维度
3. 训练过程：
   - 实现MSE损失函数
   - 使用Adam优化器设置训练循环
   - 监控并保存最佳模型
   - 选择合适的批量大小和学习率
4. 测试和可视化：
   - 加载训练好的模型
   - 与高斯滤波结果进行比较
   - 计算PSNR值
   - 展示视觉对比结果
5. 关键要求：
   - 正确归一化图像数据
   - 妥善处理输入和目标图像
   - 确保张量维度一致性
   - 保存模型权重以供评估

c) 比较高斯核去噪方法和神经网络去噪方法，讨论哪种方法表现更好以及原因。请控制在5句以内。（**5分**）

Neural network denoising significantly outperforms Gaussian filtering as shown by higher PSNR values. While Gaussian filtering causes over-smoothing and loss of details, neural networks preserve fine textures and edges effectively. This superior performance comes from the network's ability to learn complex denoising patterns from training data. The multi-layer architecture allows processing noise at different scales while preserving image details. The non-linear nature of neural networks enables better handling of various noise patterns compared to simple Gaussian filtering.

问题4 纹理合成 (15分)

a) 编写代码从示例图像中提取风格的均值、方差和Gram矩阵。我们将使用隐藏的风格/纹理来测试这一点。**(3分)**

1. Gram矩阵 (get_vgg_gram_matrix): 计算特征图的gram矩阵,反映特征之间的相关性
2. 均值 (get_vgg_mean): 计算特征图的均值
3. 方差 (get_vgg_var): 计算特征图的方差

此外还需要实现:

- Normalization类: 对输入图像进行标准化处理
- StyleFeaturesExtractor类: 使用VGG19提取特征并存储统计量

b) 通过优化图像像素来生成给定统计特性的纹理。我们也会用我们的隐藏纹理来测试这一点。**(6分)**

StyleLoss类: 计算两种损失

- gram损失: 基于gram矩阵的差异
- mean_var损失: 基于均值和方差的差异

优化函数:

- get_style_model_and_losses: 构建包含损失层的模型
- run_texture_synthesis: 优化过程的主函数

c) 如何改进起始输入以获得更具视觉吸引力的结果。**(1分)**

d) 找出一个非平凡的纹理，它在均值上看起来可以接受，但在其他方面不行。**(1分)**

e) 找出一个非平凡的纹理，它只在Gram矩阵上看起来可以接受。**(1分)**

f) 失败案例。我们给你一个有规则图案的示例。它失败了。你能修复它吗？进行你自己的研究并解释。**(2分)**

需要深入理解VGG特征和不同统计量的意义

优化过程需要合理设置参数确保收敛

需要分析不同类型纹理的特点来解决后面的分析题

g) 你如何利用本节的答案来为第2题生成一个好的隐藏测试图像。写不超过一句话。**(1分)**

问题5 隐式神经表示 (30分)

a) [5分] 首先训练一个MLP模型来拟合单张图像。 主要思路:

- 实现基础MLP架构
- 实现随机像素采样函数
- 实现图像预测和评估函数
- 完成训练循环

b) [5分] 添加位置编码(positional encoding)来改进预测效果。 主要思路:

- 实现位置编码函数
- 修改MLP架构以支持位置编码的输入
- 对比有无位置编码的效果

c) [8分] 将框架扩展到训练序列帧而不是单帧。 主要思路:

- 修改采样函数以支持时间维度
- 更新预测和评估函数以处理多帧
- 实现新的训练循环

d) [2分] 对序列帧训练时比较有无时间位置编码的效果。 主要思路:

- 实现时间维度的位置编码
- 比较并报告两种方式的结果

e) [5分] 实现批处理以避免处理大量像素时的内存问题。批大小不超过16000。 主要思路:

- 实现数据分块函数
- 修改预测函数以支持批处理

f) [5分] 生成100帧的视频,帧均匀采样于第一帧和最后帧之间。 主要思路:

- 实现视频生成和保存功能
- 验证运动的连续性