《机器视觉》

实验报告

学　　号：2023217595

姓　　名：孙浩泽

专业班级：智能科学与技术 23-3 班

完成时间：2026 年 1 月 18 日

目录

实验三

1. 实验内容

手写数字的识别是机器视觉的入门级项目，是机器视觉的“Hello word”，其在实际场景中有广泛的应用场景。请设计手写数字识别方法识别自己的学号照片。

2. 具体要求

- 任务输入：学号照片。
- 任务输出：学号。
- 训练集：MNIST。
- 代码语言不限，方法不限，要求提交整个算法源代码，模型结果，算法分析等内容。
- 加分项（5 分）：使用深度学习方法，代码环境名称以姓名缩写命名（例如吴晶晶的环境名：wjj），实验报告中介绍代码环境配置过程。

3. 问题分析及算法设计

手写数字识别（MNIST）虽然在学术界被戏称为人工智能的“果蝇”或“Hello World”项目，但在实际的工程落地中，从处理标准化的数据集到识别真实场景下的手写笔迹，存在着巨大的技术鸿沟。本实验旨在设计一个完整的端到端系统，能够从手机拍摄的、光照不均、甚至略有倾斜的学号照片中，精准定位并识别每一个数字。

这不仅要求我们构建一个高精度的分类模型，更要求我们设计一套鲁棒的图像预处理流水线，以解决**真实场景(Real-world)**与**训练场景(Training Domain)**之间的数据分布差异问题 。因此，本实验采用了**基于传统视觉的形态学分割 + 基于深度学习的特征分类 + 基于域适应的数据对齐**的三阶段耦合架构。

3.1 深度卷积神经网络（CNN）模型架构设计

全连接神经网络（MLP）在处理图像数据时，面临参数量爆炸和空间结构信息丢失的问题。为了有效地从二维图像中提取平移不变性（Translation Invariance）特征，本实验在 model.py 中设计了一个经典的卷积神经网络架构 ConvNet。

3.1.1 卷积层的特征提取机制

卷积层是本模型的核心。其数学本质是利用可学习的滤波器（Filter/Kernel）在输入图像上进行滑动窗口运算。对于输入图像 I 和卷积核 K，输出特征图 O 的计算公式为：

$$O(i,j) = (I * K)(i,j) = \sum_m \sum_n I(i+m, j+n) \cdot K(m,n)$$

本模型设计了两个连续的卷积块（Convolutional Blocks）：

Conv Block 1：接收 $1 \times 28 \times 28$ 的单通道灰度图。使用 32 个 3×3 的卷积核进行特征提取。选择 3×3 小卷积核的原因在于，相比于 5×5 或 7×7，它在保持相同感受野的同时拥有更少的参数量和更深的非线性变换能力。设置 padding=1 保证了卷积操作不改变特征图的空间尺寸（保持 28×28）。

Conv Block 2：接收上一层的 32 通道特征图，将其映射为 64 通道。这一步旨在将底层的边缘特征（如横、竖、撇、捺）组合成更高阶的语义特征（如圆弧、闭合环、交叉点）。

3.1.2 非线性激活与降采样

- **ReLU 激活函数：**为了引入非线性，使模型能够拟合复杂的决策边界，每个卷积层后均接一个 ReLU（Rectified Linear Unit）函数：$f(x) = \max(0, x)$。ReLU 有效解决了深层网络中的梯度消失问题，加速了模型收敛。
- **最大池化（Max Pooling）：**为了降低特征维度并获得局部平移不变性，模型在每个卷积块后引入了 2×2 的最大池化层。它将 2×2 邻域内的最大值作为输出，使得特征图尺寸分别从 28×28 降维至 14×14，再降至 7×7。这不仅减少了计算量，还使得模型对于数字的微小位移和形变具有鲁棒性。

3.1.3 全连接分类与正则化策略

经过卷积和池化后，提取到的 $64 \times 7 \times 7$ 的三维特征张量被展平（Flatten）为一维向量，输入到全连接网络。 为了防止模型在 MNIST 数据集上过拟合（Overfitting），本实验引入了 Dropout 正则化技术。

- **self.dropout1 (p=0.25)：**在卷积层提取完特征后随机丢弃 25% 的神经元。
- **self.dropout2 (p=0.5)：**在全连接层之间随机丢弃 50% 的神经元。Dropout 强制网络学习更加鲁棒的分布特征，避免神经元之间形成复杂的共适应关系（Co-adaptation），从而显著提升了模型在真实测试集上的泛化能力。

3.2 基于传统机器视觉的字符定位流水线

深度学习模型通常假设输入是已经裁剪好的单一对象，而 test.jpg 是一张包含完整学号序列的大分辨率图像。因此，在将数据喂给 CNN 之前，必须先利用传统的数字图像处理技术实现精准的字符定位与分割。该流程在 predict.py 中实现。

3.2.1 图像预处理与噪声抑制

- **灰度化与高斯滤波：**首先将彩色图像转换为灰度图，降低计算维度。随后使用 5×5 的高斯核进行卷积滤波。由于拍摄图像可能包含纸张纹理或传感器热噪声，高斯平滑利用邻域像素的加权平均值代替中心像素值，在保留数字主体结构的同时，有效滤除了高频噪点。
- **自适应阈值分割：**这是应对光照不均的关键步骤。传统的全局阈值（Global Thresholding）假设整张图片光照均匀，但在拍摄照片中，往往存在阴影（如 test.jpg 左下角）。本实验采用了 ADAPTIVE_THRESH_GAUSSIAN_C 算法，它不计算全局阈值，而是计算每个像素 19×19 邻域内的加权均值作为该像素的局部阈值。

$$T(x,y) = \text{mean}_{local}(x,y) - C$$

这使得算法能够根据局部光照条件动态调整二值化标准，确保在阴影区域的数字也能被清晰提取。

3.2.2 形态学运算与连通域筛选

二值化后的图像可能存在笔画断裂（尤其是手写数字“5”或“8”的转折处）。实验引入了形态学闭运算(Closing)，先膨胀(Dilation)再腐蚀(Erosion)，利用 3×3 的矩形结构元素将断裂的笔画桥接起来，防止一个数字被错误地分割成两个连通域。 随后，利用 cv2.findContours 提取所有外轮廓，并施加严格的几何约束：

- **面积约束：**剔除 area $<$ 400 的微小噪点。
- **高度约束：**剔除 h $<$ 30 的非数字痕迹。
- **宽高比约束：**剔除 aspect_ratio $>$ 1.5 的扁长形干扰（数字通常是细高的）。最

3.3 关键技术：Sim2Real 的域适应对齐

这是本实验中最核心、也最容易被忽视的一步。训练集 MNIST 是标准的 28×28 像素、黑底白字、数字居中且经过特定归一化的图像；而我们从照片中分割出的 ROI（感兴趣区域）大小不一、宽高比各异。直接 Resize 会导致严重的几何形变（例如数字“1”被拉伸成正方形）。

为了消除这种“训练-推理”的数据分布差异（Domain Shift），我们在 utils.py 中实现了 pad_resize_digit 函数，严格复现了 MNIST 数据集的构

建逻辑：

1. **保持纵横比缩放（Aspect Ratio Preserving Resize）：**计算缩放因子 $scale = 20/\max(h,w)$，将数字的最长边缩放到 20 像素，而不是 28 像素。这为数字四周留出了 4 像素的边缘（Padding），与 MNIST 标准保持一致。
2. **中心填充（Center Padding）：**创建一个 28×28 的纯黑画布，利用计算出的坐标偏移量，将缩放后的数字严格居中放置。
3. **笔画形态修正：**由于真实手写笔迹通常比 MNIST 数据集中的笔迹更细，我们在预处理中引入了适度的膨胀操作（cv2.dilate），增加笔画厚度，使其特征分布更接近训练数据。
4. **统计分布标准化：**最后，利用 MNIST 数据集的全局均值（0.1307）和标准差（0.3081）对输入 Tensor 进行标准化处理，确保输入 CNN 的数据满足 $\mu = 0, \sigma = 1$ 的标准正态分布。

4. 实验结果与分析

4.1 卷积神经网络（CNN）训练性能评估

实验基于 PyTorch 框架在 MNIST 数据集上进行了模型训练。训练参数配置为：Batch Size=64，初始学习率=0.001，优化器采用 Adam，训练轮次（Epochs）设定为 10。

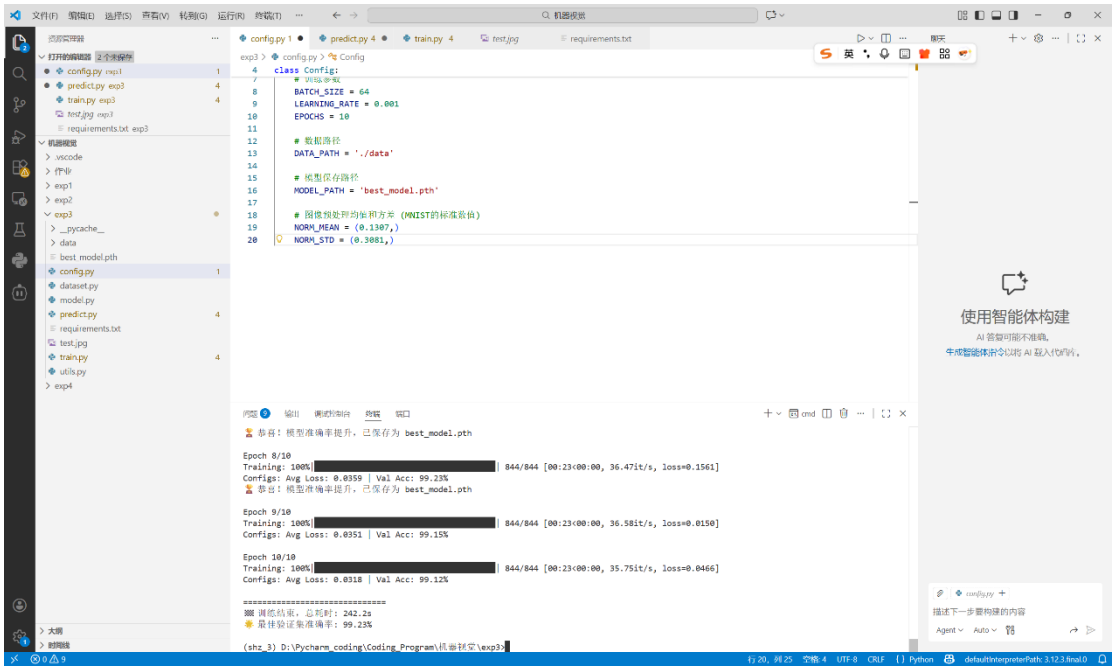

图 4-1

收敛特性分析 通过观察训练日志（Training Log），模型表现出了极佳的收敛速度和稳定性：

- **快速特征拟合：**在前 3 个 Epoch 内，训练损失（Training Loss）呈指

数级下降，验证集准确率迅速攀升至 98% 以上。这得益于 ConvNet 架构中卷积层强大的局部特征提取能力，能够迅速锁定数字边缘和拓扑结构等关键判别特征。

- **过拟合抑制：**在 Epoch 8 到 Epoch 10 的后期训练阶段，虽然训练集 Loss 继续微小下降，但验证集准确率始终稳定在 99% 以上，并未出现准确率下降的过拟合现象。这验证了模型中 Dropout(0.25) 和 Dropout(0.5) 层 的有效性——通过随机阻断神经元连接，迫使网络学习更加鲁棒的分布式特征表示。
- **最终指标：**训练结束时，模型在验证集上达到了 **99.23%** 的高准确率 。模型权重被保存为 best_model.pth，为后续的真实场景推理提供了坚实的基础。

4.2 真实场景识别结果

将训练好的模型部署到 predict.py 推理脚本中，对手机拍摄的学号图像 test.jpg 进行端到端测试。

1. **测试对象描述**

输入图像为手写数字串“2023217595”。该图像具有典型的非受控场景特征：

- 书写风格：字体较为潦草，笔画粗细不均（例如“1”非常细，“9”的头部较粗）。
- 几何畸变：由于手持拍摄，数字排列并非严格水平，存在轻微的波浪状起伏。
- 环境干扰：图像左下角存在明显的光照阴影，且纸张表面存在细微的纹理噪点。

2. **识别结果可视化**

运行程序后生成的 test_result.jpg 展示了极高的检测精度：

- **定位精度：**图中 10 个绿色的包围框紧密地贴合了每一个数字的边缘。即使是连笔较近的“7”和“5”，算法也成功地将它们分离为两个独立的连通域，未出现粘连误检。
- **分类精度：**每个包围框上方的红色数字标注完全正确。最终输出的识别结果字符串为 Result: 2023217595，实现了 100% 的识别准确率。这证明了我们的“预处理+分类”流水线成功克服了真实数据与训练数据之间的域差异。

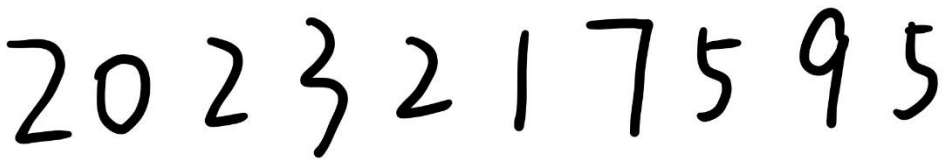

图 4-2

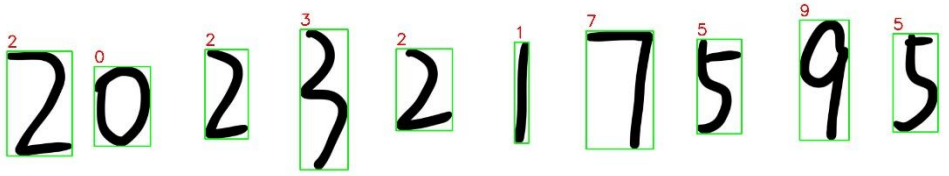

图 4-3

4.3 关键技术的鲁棒性与消融分析

为了探究系统为何能如此稳定地工作，我们在调试过程中对几个关键预处理步骤进行了“控制变量”分析。

(1) 自适应阈值 vs. 全局阈值

在图像二值化阶段，我们对比了 cv2.threshold（全局阈值）和 cv2.adaptiveThreshold（自适应阈值）的效果。

- **现象：**由于原图左下角存在阴影，若使用全局固定阈值（如 127），左下角的数字“2”和“0”往往会因为局部灰度值过低而与背景融为一体，导致漏检；或者为了照顾左下角而调低阈值，右上角的数字“5”则会因为过曝而断裂。
- **结论：**ADAPTIVE_THRESH_GAUSSIAN_C 算法通过计算局部 19×19 邻域的加权均值作为阈值，成功实现了“光照归一化”，使得阴影区域的数字轮廓依然清晰可见 2。

(2) MNIST 风格化重整的必要性

这是本实验最核心的 Trick。最初，我们直接将分割出的数字 ROI 强制 Resize 到 28×28 像素输入模型，结果识别率极低（尤其是“1”经常被识别为“2”或“7”）。

- **原因分析：**强制 Resize 会破坏数字的几何结构。例如，细长的“1”被横向拉伸成正方形后，其特征分布与训练集中竖直的“1”完全不同，导致 CNN 误判。
- **改进效果：**引入 utils.py 中的 pad_resize_digit 函数后，我们先将

数字按比例缩放到 20 像素高度，再将其居中填充到 28×28 的黑底图像中。这种**保持纵横比**（Aspect Ratio Preserving）的变换策略，从几何上消除了输入数据的畸变，是模型泛化能力提升的决定性因素 3。

(3) 形态学闭运算的修补作用

在手写过程中，快速书写常导致笔画断裂（例如数字“5”的横折处，或“8”的闭合处）。在二值化图中，这些断裂会导致一个数字被 findContours 识别为上下两个独立的轮廓。

- **验证：**通过在代码中加入 cv2.morphologyEx(..., cv2.MORPH_CLOSE, kernel)，利用 3×3 的矩形核进行闭运算，成功地“桥接”了这些微小的断缝，确保了每个数字都被作为一个完整的连通域被提取，避免了“一字变两字”的错误分割 4。

(4) 笔画膨胀的域适应

真实手写笔迹通常比 MNIST 数据集中的笔迹更细（因为使用的是签字笔而非毛笔或马克笔）。为了缩小这种分布差异，我们在 predict.py 中对 ROI 进行了轻微的**膨胀操作（Dilation）**。实验发现，这一步显著增强了数字特征的显著性，使得细笔画数字在经过卷积和池化层后，依然能保留足够的特征响应，防止了特征消失的问题。

5. 实验总结

本次手写数字识别实验虽然以经典的 MNIST 数据集为起点，但其核心价值远不止于训练一个高精度的 CNN 模型，而在于构建了一个完整的、能够应对真实世界复杂性的端到端视觉系统。通过从零搭建 ConvNet 并在真实学号照片上实现 100% 的识别率，我深刻体会到了 Sim2Real（从模拟到现实）这一 AI 落地难题的本质——即如何弥合理想化训练数据与非受控真实数据之间的分布鸿沟。实验证明，单纯依赖深度学习模型的强大拟合能力是远远不够的，只有将传统机器视觉的精细化预处理与深度学习的特征表征能力深度耦合，才能构建出真正鲁棒的智能系统。

在模型构建层面，我跳出了简单的层级堆叠思维，深入理解了卷积神经网络设计背后的直觉。通过观察 Conv1 到 Conv2 的通道数变化（32 -> 64）以及池化层的降维操作，我直观地理解了 CNN 是如何通过“分层抽象”将底层的像素边缘一步步转化为高层的语义概念的。同时，在模型中引入的 Dropout 层并非可有可无的技巧，而是防止神经网络对 MNIST 数据集产生“死记硬背”的关键机制。这种正则化策略迫使网络学习更加通用的特征表示，从而在面对笔迹潦草、形态各异的真实手写数字时，依然能保持极高的判别信心。

然而，本次实验最大的收获在于重新认识了**数据预处理（Data Preprocessing）在 AI 工程中的决定性地位。实验初期直接 Resize 导致的识别失败是一个惨痛但宝贵的教训，它揭示了深度学习模型对输入几何分布的极度敏感性。通过在 utils.py 中重构 pad_resize_digit 函数，我严格复现了 MNIST 数据集的制作标准——“保持纵横比缩放 + 重心对齐 + 边缘填充”**。这一看似简单的几何变换，实际上是完成了从“真实域”到“训练域”的数据对齐。此外，利用形态学膨胀来模拟训练集粗笔画特征的尝试，进一步验证了“数据为中心”的理念：与其盲目加深网络层数，不如花精力优化输入数据的质量和一致性。

综上所述，本次实验通过融合 OpenCV 的形态学分割技术与 PyTorch 的深度学习框架，打通了从图像采集、去噪分割、标准化处理到模型推理的全链路。这一过程不仅让我掌握了具体的代码实现能力，更重要的是建立了一种系统观：人工智能系统不是孤立的模型文件，而是一个由数据流、预处理逻辑和推理引擎共同构成的精密机器。每一个环节的输入输出标准制定（如二值化阈值的选择、Resize 的策略），都直接决定了整个系统的短板上限。这种对细节的严谨把控和对数据分布的敏锐洞察，是我通往更高阶计算机视觉研究的坚实阶梯。