

# 合肥工业大学

## 《机器视觉》课程实验

实验题目	课程实验三：学号识别
学生姓名	高志鸿
学    号	2023217584
专业班级	智能科 23-3 班

## 一、实验目的

- 掌握从“真实拍摄图片”到“MNIST 规范输入”的完整预处理流程：灰度化、二值化、去噪、轮廓筛选、数字分割、标准化对齐。
- 理解并实现基于 CNN 的手写数字分类模型，完成 0-9 十分类任务。
- 将单个数字识别扩展到“多数字序列识别”（学号），实现端到端的学号检测与输出可视化。
- 学会用中间结果（分割图、概率分布、结果叠加图）定位错误来源并优化流程。

## 二、实验环境

- 操作系统：Windows
- 开发语言：Python
- 主要依赖：PyTorch、torchvision、OpenCV (cv2)、NumPy
- 代码位置：`exp3/`
  - 训练：`exp3/train.py`
  - 推理：`exp3/main.py`
  - 预处理：`exp3/preprocess.py`
  - 模型结构：`exp3/model.py`
- 数据/模型/输入输出：
  - MNIST 数据：`exp3/data/MNIST/`（自动下载）
  - 训练好的模型：`exp3/models/mnist\_cnn.pt`
  - 输入图片：`exp3/input/input.jpg`
  - 输出目录：`exp3/output/`

## 三、实验原理与方法

本实验采用“传统图像处理分割 + CNN 分类”的经典两阶段方案：先在真实拍摄图片中分割出每个数字并标准化为 MNIST 风格的  $28 \times 28$  单通道图，再用 CNN 对每个数字进行分类，最后按从左到右顺序拼接得到学号。

### 1. CNN 数字分类模型（`exp3/model.py`）

- 网络结构 `Net`：两层卷积（ $1 \rightarrow 32 \rightarrow 64$ ）+ ReLU + MaxPool + Dropout + 两层全连接（ $9216 \rightarrow 128 \rightarrow 10$ ），输出 `log\_softmax` 形式的对数概率。
- 使用 Dropout（0.25 / 0.5）抑制过拟合，提升泛化能力。

### 2. 模型训练（`exp3/train.py`）

- 训练集/测试集：MNIST，训练时带数据增强以适应真实手写与拍摄形变：
  - `RandomRotation(15)`、`RandomAffine(translate/scale/shear)`
  - `Normalize((0.1307,), (0.3081,))` 与推理保持一致

- 优化器与学习率：Adadelata（`lr=1.0`），StepLR 每轮衰减（`gamma=0.7`），训练 `epochs=10`，最终保存权重到 `exp3/models/mnist\_cnn.pt`。

### 3. 学号图片预处理与分割（`exp3/preprocess.py`）

- 灰度化后使用自适应二值化 `adaptiveThreshold(Gaussian, THRESH\_BINARY\_INV, blockSize=21, C=2)` 适应光照不均。
  - 传统二值化处理是将全局阈值“写死”，这样就会导致阴影或者噪点对二值化造成影响。
  - 此处处理是自适应二值化，即局部二值化。取  $3 \times 3$  范围内灰度值的中值减 2 作为阈值，此处理可以适应光照不均。
- 闭运算（`MORPH\_CLOSE`， $2 \times 2$ ）连接断裂笔画并抑制噪点。

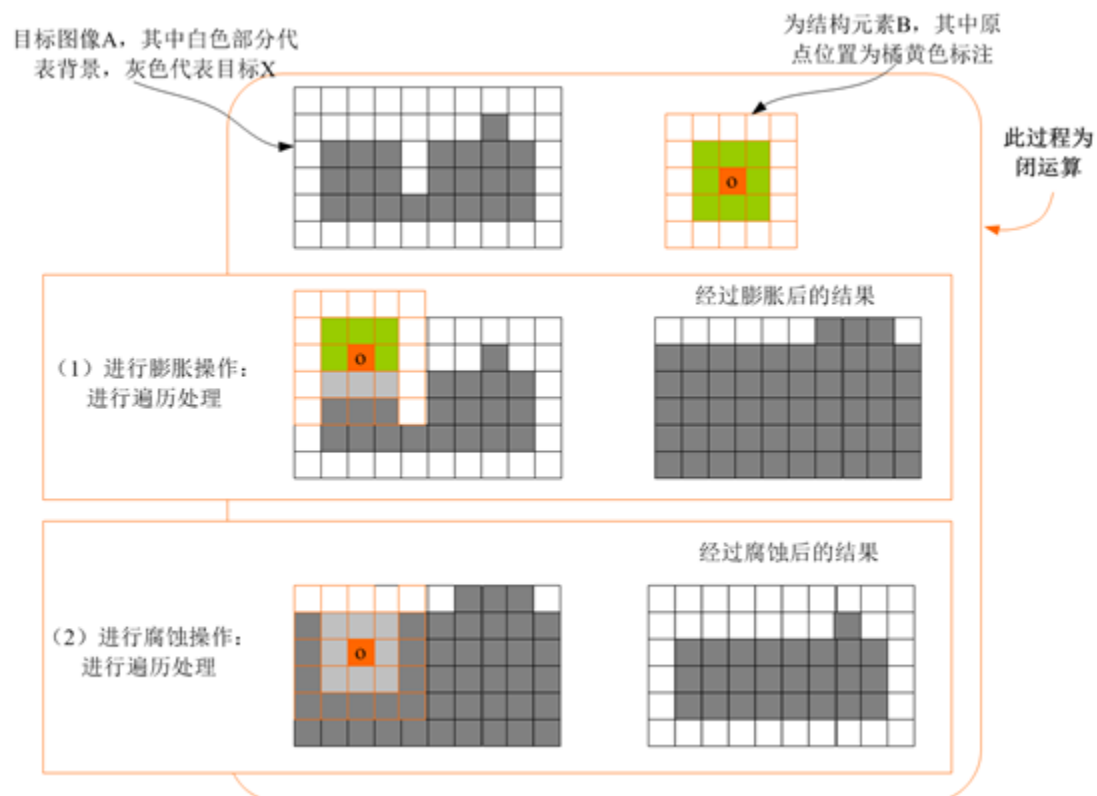


Figure 1 闭运算原理

- 轮廓提取与候选框筛选：过滤极小噪点，并基于候选框高度中位数做动态筛选，使保留下来的区域更像“数字”。
- 粘连字符处理：当  $w > 1.5 \times h$  时使用垂直投影找“波谷”作为分割点；只有波谷深度满足  $\min\_val < \text{avg\_proj} \times 0.60$  才切分，减少误切宽写数字。
- MNIST 风格标准化：每个数字先等比缩放到  $20 \times 20$  框内，再放入  $28 \times 28$  画布；通过质心（Center of Mass）对齐到 (14,14)，使分布更接近 MNIST 训练集。
- 轻微形态学加粗：使用水平  $2 \times 1$  膨胀修复笔画断裂（更有利于某些数字的底

部横线连续性)，同时尽量不破坏如“0”的内部空洞结构。

## 4. 序列识别与结果输出（`exp3/main.py`）

- 对 `exp3/input/` 中图片逐张处理：
  - `preprocess\_image()` 得到按  $x$  排序的数字列表
  - 每个数字经 `ToTensor + Normalize` 后送入模型预测
  - 将预测结果拼接为学号字符串，并叠加到原图生成 `{name}\_result.png`
- 同时保存调试产物：`{name}\_digit\_0.png ... {name}\_digit\_9.png` 便于核对分割是否正确；将结果追加写入 `exp3/output/result.txt`。

# 四、实验内容与步骤

## 1. 准备数据与环境

- 确认 MNIST 数据位于 `exp3/data/`（首次训练会自动下载）。
- 将待识别学号图片放入 `exp3/input/`，例如 `input.jpg`。

## 2. 训练模型

- 运行 `exp3/train.py`，训练结束后生成 `exp3/models/mnist\_cnn.pt`。

## 3. 学号识别推理

- 运行 `python -m exp3.main`，脚本会自动遍历 `exp3/input/` 下的图片进行识别。

## 4. 查看输出与中间结果

- 分割结果：`exp3/output/input\_digit\_0.png` ~ `input\_digit\_9.png`
- 辨识叠加图：`exp3/output/input\_result.png`
- 文本结果：`exp3/output/result.txt`（按运行次数追加记录）
- 控制台输出：会打印每个数字的预测类别与 Top-3 概率，便于判断不确定数字。

# 五、实验数据记录（本项目实际运行记录）

- 输入图片：`exp3/input/input.jpg`
- 输出文件（本次运行生成/覆盖）：
  - `input\_digit\_0.png` ~ `input\_digit\_9.png`
  - `input\_result.png`
  - `result.txt`

- 自适应推理运行结果（控制台打印）：
  - 识别学号：`2023217584`
- `result.txt` 中也保留了历史多次运行记录，本次最新结果为：`input.jpg: 2023217584`（文件末行）。

## 六、实验结果与分析

### 1.实验结果

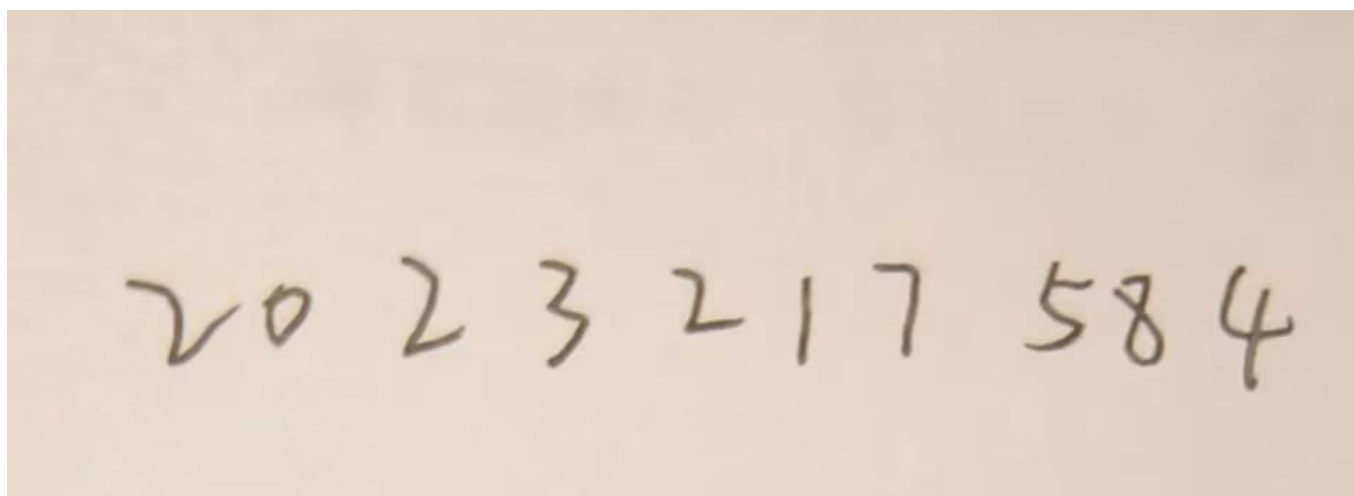


Figure 2 输入

单个数字分割结果：



识别结果：

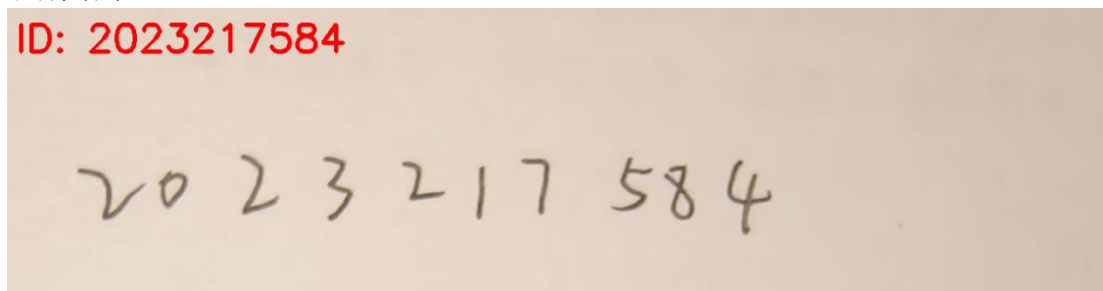


Figure 3 输出

### 2.结果分析

- 从最终输出看，本实验成功将真实拍摄的学号图片转换为 10 张标准化数字图，并依次完成分类与拼接，得到正确学号 `2023217584`。
- 该任务的主要难点在“分割”而非“分类”：MNIST 分类模型本身对规范 28×28 输入的识别能力较强，但真实图片存在光照不均、笔画断裂、粘连数字、

字体大小不一致等问题，会显著影响分割质量。一旦分割顺序、数量或单字形态发生偏差，即使模型分类准确也会导致最终学号错误或位数不对。

- 本项目通过自适应二值化 + 闭运算提升前景稳定性，通过“高度中位数筛选 + 投影波谷分割”提升候选数字框的可靠性，并用“质心对齐”将输入分布尽量对齐 MNIST，从而使分类阶段更稳定。

## 七、结论

- 构建了完整的学号识别系统：图像预处理分割 → MNIST 风格标准化 → CNN 十分类 → 序列拼接输出。
- 在当前测试样例 `input.jpg` 上，系统能够输出正确学号 `2023217584`，并生成可用于展示的分割结果图与叠加结果图。
- 实验验证了“数据分布对齐（Normalization/质心对齐）+ 分割鲁棒性”是让 MNIST 模型迁移到真实场景的关键。

## 八、实验体会与个人理解

这次实验让我明确意识到，真正决定系统上限的往往不是模型结构有多复杂，而是数据是否被“喂对了”。同样的 CNN，在 MNIST 上可能轻松达到较高准确率，但一旦输入换成真实拍摄的学号图片，光照、背景、笔画粗细、倾斜与粘连都会让数据分布发生偏移。如果没有一套稳定的预处理把输入拉回到 MNIST 的分布附近，模型再强也很难稳定输出正确结果。因此我在实验中把更多精力放在预处理流水线的可靠性上，通过自适应二值化、形态学去噪、候选框动态筛选与质心对齐，让每个分割数字尽可能“像 MNIST”。

我也体会到“分割是序列识别的生命线”。分类器只负责“这张  $28 \times 28$  是什么数字”，而学号识别需要保证“切出了正确数量、顺序正确、每个字符不被误切或漏切”。一旦某个数字粘连导致少切一个，或者宽写数字被误切成两个，最终拼接结果就会位数错乱；又或者笔画断裂使某个数字结构改变，即便只错一位也会导致学号整体错误。也正因为如此，中间产物（`input\_digit\_\*.png`）对排查问题非常关键：它能把复杂系统拆成可观察的小问题，迅速判断错误是来自分割还是来自分类，从而有针对性地改进。

最后，这个实验让我形成了更工程化的调试方法：不只看最终学号是否正确，还要关注每一步输出是否“可解释、可复现”。例如观察二值化是否稳定、候选框是否覆盖完整数字、质心对齐后数字是否居中、模型对每位数字的 Top-3 概率是否具有足够置信度。这样的闭环思路让整个系统更可靠，也使我在答辩时能清晰说明每个设计选择背后的原因与效果。