计算机视觉通选课教学大纲

目录

1 课程定位

属性	说明
课程性质	通选课，面向多专业本科生
学时安排	32 学时（前 10 周 ×3 学时 + 第 11 周 ×2 学时）
设计理念	OBE 成果导向 + 学生中心 + 项目驱动
最终项目	**自动阅卷系统**（支持选择题、判断题、简答题）

2 OBE 能力矩阵

能力维度	具体描述	可验证产出
A1-图像处理	能对图像进行预处理、增强、分割	每周实验代码
A2-版面分析	能定位试卷中各题型区域	第 4 周作业
A3-特征识别	能识别填涂、符号、手写文字	第 5-8 周作业
A4-系统集成	能将各模块整合为完整系统	最终项目
A5-AI 协作	能用 AI 工具辅助编程开发	全过程体现

3 课程故事线

本课程以**“造一个能改卷子的 AI 助教”**为主线，通过 11 周的渐进式学习，让学生从零开始构建一个完整的自动阅卷系统。

第二章： **让试卷更清晰**——拍照模糊怎么办?（第 3 周）

第三章： **找到题目位置**——选择题、简答题在哪里?（第 4 周）

第四章： **识别填涂答案**——怎么知道选了 A 还是 B?（第 5 周）

第五章： **识别对错符号**——怎么看到是 还是 ×?（第 6 周）

第六章： **阅读手写文字**——能看懂学生写的答案吗?（第 7-8 周）

第七章： **整合成 AI 助教**——让它改完一整张卷子（第 9-11 周）

4 详细周次设计（32 学时）

4.1 第一阶段：基础准备（第 1-2 周，6 学时）

4.1.1 第 1 周：计算机视觉导论与图像基础

故事问题：机器是怎么“看见”试卷的？

时间	内容	活动	产出
40min	CV 导论：从人脸识别到阅卷系统	案例展示	理解课程目标
50min	图像的数字表示：像素→RGB→矩阵	Jupyter 实验	能用代码显示图像
40min	OpenCV 入门：读取、显示、保存	跟着做	读取试卷图片
30min	实验：给试卷加滤镜	动手实践	提交实验结果

课后作业：用 OpenCV 实现 3 种图像滤镜效果

4.1.2 第 2 周：AI 辅助编程工具实战

故事问题：怎么让 AI 帮我写代码？

时间	内容	活动	产出
30min	AI 编程工具介绍	演示	了解工具
40min	Prompt 工程：CV 领域专用模板	互动练习	能写有效 Prompt
50min	实战：用 AI 辅助实现人脸检测	分组实践	跑通第一个 CV 项目
40min	代码调试与优化技巧	实战演示	学会调试方法

课后作业：用 AI 辅助实现一个手势识别程序

4.2 第二阶段：核心技能（第 3-8 周，18 学时）

4.2.1 第 3 周：图像预处理与增强

故事问题：试卷拍照模糊怎么办？

时间	内容	项目贡献	产出
40min	图像去噪（高斯/中值滤波）	清理试卷噪点	
50min	二值化与阈值分割	分离填涂与背景	
40min	透视矫正（修正拍照角度）	矫正倾斜试卷	
30min	实验：预处理流水线	动手实践	提交对比效果图

课后作业：实现试卷图像预处理完整流程

4.2.2 第 4 周：试卷版面分析

故事问题：怎么知道选择题、简答题在哪里？

时间	内容	项目贡献	产出
40min	边缘检测与轮廓查找	找到题目边界	
50min	连通域分析与区域分割	分隔各题型区域	
40min	文本行检测（PaddleLayout）	定位简答题位置	
30min	实验：版面结构可视化	动手实践	提交标注结果

课后作业：实现试卷版面分析，标注三种题型区域

4.2.3 第 5 周：选择题识别（填涂检测）

故事问题：怎么知道选了 A 还是 B？

课后作业：实现选择题填涂识别模块

时间	内容	项目贡献	产出
30min	OMR 原理：光学标记识别	理解填涂检测	
60min	像素密度统计算法	核心识别逻辑	
50min	多选项处理（A/B/C/D）	完整选择题识别	
20min	实验：识别一张选择题答卷	动手实践	提交识别结果

4.2.4 第 6 周：判断题识别（符号匹配）

故事问题：怎么看到是 还是 ×？

时间	内容	项目贡献	产出
40min	形状匹配算法原理	理解符号识别	
50min	轮廓特征提取（圆度、凸性）	区分 和 ×	
40min	模板匹配实战	判断题识别	
30min	实验：识别判断题答案	动手实践	提交识别结果

课后作业：实现判断题符号识别模块

4.2.5 第 7 周：OCR 基础与文字识别

故事问题：怎么让机器“阅读”文字？

时间	内容	项目贡献	产出
40min	OCR 技术原理与发展	理解文字识别	
60min	PaddleOCR 快速上手	中文识别工具	
40min	印刷体文字识别实战	识别试卷题号	
20min	实验：识别试卷标题	动手实践	提交识别结果

课后作业：用 OCR 识别试卷中的印刷文字

4.2.6 第 8 周：手写简答题识别

故事问题：能看懂学生写的答案吗？

时间	内容	项目贡献	产出
40min	手写识别的挑战	理解技术难点	
60min	端到端手写识别（TrOCR）	手写文字识别	
40min	识别结果后处理	优化识别效果	
20min	实验：识别手写简答	动手实践	提交识别结果

课后作业：实现简答题手写识别模块

4.3 第三阶段：项目实战（第 9-11 周，8 学时）

4.3.1 第 9 周：系统架构与分组开发

故事问题：把所有模块组合起来

时间	内容
40min	项目架构讲解与模块划分
80min	分组开发 + 教师巡回指导
20min	进度检查与问题解答

系统框架（已提供）：

```
auto_grading_system/
    input/                  # 测试试卷
    output/                 # 识别结果
    modules/
        preprocess.py       # 预处理（已实现）
        layout.py           # 版面分析（已实现）
        choice.py           # 选择题识别（学生实现）
        judge.py            # 判断题识别（学生实现）
        essay.py            # 简答题识别（学生实现）
        grading.py          # 评分逻辑（学生实现）
    main.py                 # 主程序（部分实现）
```

分组要求：

- 3-4 人/组
- 包含不同专业背景
- 明确分工：前端展示、后端整合、算法优化

4.3.2 第 10 周：核心开发与调试

故事问题：让系统真正跑起来

开发要求：

时间	内容
30min	集成指导：如何拼接各模块
120min	分组开发 + 教师一对一指导
30min	测试集验证（10 张标准试卷）

等级	完成度	分数段
基础版	完成一种题型识别	60-75
进阶版	完成两种题型识别	76-88
完整版	完成三种题型识别	89-100

4.3.3 第 11 周：成果展示与总结

故事问题：展示你的 AI 助教

时间	内容
30min	分组展示（每组 5 分钟演示 + 2 分钟答辩）
40min	互评与教师点评
20min	课程总结与 CV 技术展望
10min	优秀项目颁奖

5 课程设计详细方案

5.1 项目名称：智能阅卷系统

5.2 功能需求矩阵

题型	输入	处理	输出	技术方案
选择题	填涂区域图像	像素密度统计	选项 (A/B/C/D)	OpenCV countNonZero
判断题	符号区域图像	形状匹配	符号 (/×)	轮廓特征 + 模板匹配
简答题	手写文字区域图像	端到端 OCR	文字内容	PaddleOCR / TrOCR

5.3 系统流程

$$\text{试卷图片} \rightarrow \text{图像预处理} \rightarrow \text{版面分析} \rightarrow \begin{matrix}\text{选择题识别}\\ \text{判断题识别}\\ \text{简答题识别}\end{matrix} \rightarrow \text{答案匹配} \rightarrow \text{评分报告}$$

5.4 分组策略

角色	职责	适合专业
组长	统筹协调、进度管理	所有专业
算法负责人	识别算法实现与调优	CS/EE
前端负责人	界面展示、可视化	设计/媒体
测试负责人	测试用例、文档撰写	所有专业

5.5 评估标准

维度	占比	评分标准
功能完整性	40%	三种题型全实现 (40 分) / 两种题型 (30 分) / 一种题型 (20 分)
识别准确率	25%	测试集准确率 >90%(25 分) / >80%(20 分) / >70%(15 分)
代码质量	15%	结构清晰、注释完整、可复用
展示效果	10%	演示流畅、问题回答清晰
团队协作	10%	分工明确、组内互评

6 教学资源设计

6.1 代码脚手架（三层支持）

Layer 1: 完全封装版（面向零基础）

```
from cv_toolkit import ChoiceRecognizer

recognizer = ChoiceRecognizer()
result = recognizer.detect(image)  # 一行调用
```

Layer 2: 参数可调版（面向进阶）

```
from cv_toolkit import ChoiceRecognizer

recognizer = ChoiceRecognizer(
    threshold=127,      # 可调参数
    option_count=4,
    min_density=0.3
)
result = recognizer.detect(image)
```

Layer 3: 算法实现版（面向挑战）

```
# 学生需要实现核心算法
def detect_answers(image, threshold=127):
    binary = cv2.threshold(image, threshold, 255, cv2.THRESH_BINARY)[1]
    # TODO: 实现密度统计算法
    pass
```

6.2 实验数据集

```
datasets/
  test_papers/          # 测试试卷（20张，三种题型）
  templates/            # 标准答案模板
  samples/
    choice/             # 选择题样本
    judge/              # 判断题样本
    essay/              # 简答题样本
  results/              # 标注结果（用于验证）
```

7 差异化教学策略

7.1 跨专业适配方案

学生类型	特点	支持策略
CS 专业	编程基础好	提供算法实现版，鼓励深入优化
EE 专业	数学基础好	强调算法原理，提供数学推导
文科专业	编程薄弱	提供封装 API，强调应用场景

7.2 AI 辅助编程 Prompt 模板

场景	Prompt 模板
理解代码	“这段 OpenCV 代码是什么意思？请用通俗语言解释：{代码}”
调试问题	“我的填涂识别结果不准确，这是代码和输入输出，请帮我分析问题”
优化算法	“如何提高这张试卷的 OCR 识别准确率？当前使用 PaddleOCR”

8 考核方案

考核环节	占比	说明
平时作业	30%	每周实验（8 次 ×3-4 分）
课程项目	50%	功能完整度 + 代码质量 + 识别准确率
展示报告	15%	演示效果 + 文档质量 + 答辩表现
团队互评	5%	组内贡献度评价

9 时间投入建议

周次	课内学时	课外学时	累计产出
1-2 周	6h	4h/周	能用 OpenCV + AI 工具
3-4 周	6h	6h/周	预处理 + 版面分析
5-6 周	6h	6h/周	选择题 + 判断题识别
7-8 周	6h	8h/周	OCR + 手写识别
9-11 周	8h	12h/周	完整阅卷系统

10 设计总结

课程设计核心优势

- **零浪费**：每周产出直接用于最终项目
- **零门槛**：AI 工具 + 三层脚手架
- **零脱节**：所有内容服务于三种题型识别
- **故事化**：从“看见”到“理解”再到“评分”的完整叙事
- **可扩展**：基础版 → 进阶版 → 挑战版，满足不同层次需求