关于《校园人类风险行为识别系统》数据脱敏处理的证明

**一、证明声明**

本团队郑重声明：为参加“第七届国际青年人工智能大赛——校园场景AI创新应用赛”，本项目《校园人类风险行为识别系统》中所使用的全部训练数据（包括 YOLO 图像识别训练集与 DeepSeek 大模型微调语料集），均已按照《中华人民共和国个人信息保护法》及国家标准 GB/T 35273-2020《信息安全技术 个人信息安全规范》的要求，完成了严格、规范的数据脱敏处理与合规性审查。经处理后的数据集不包含任何可直接或间接识别特定个人身份的信息，确保所有数据在采集、处理、存储及训练环节中的安全性与合法合规性。

**二、脱敏数据范围**

（一）YOLO 图像目标检测训练集

数据集名称：校园场景人类行为识别图片集（跌倒/行走/打闹）  
数据量：5125张图片  
原始来源：团队自行采集与模拟实验场景拍摄（部分参考开源公开安全数据集）  
涉及敏感信息：人脸影像  
脱敏范围：所有图像中出现的人员人脸

（二）DeepSeek 微调语料集

数据集名称：校园安全智能问答与告警语料示例集  
数据量：共 100000 条样本（示例）  
数据类型：纯文本语料（instruction / input / output 格式）  
示例内容：  
{"instruction": "请根据你的知识回答以下问题", "input": "什么是物联网？", "output": "物联网是指通过互联网相互连接的物理设备网络，这些设备能够收集和交换数据。"}  
{"instruction": "你是本项目的AI助手，请根据你的知识回答用户的问题", "input": "系统如何工作？", "output": "处理流程：1)视频输入与采样 2)目标检测与分类 3)状态帧比例持续判断 4)告警信息生成 5)结果推送展示。"}  
{"instruction": "你是本项目的告警助手，根据输入的高危信息，进行告警并输出到最终结果到前端用户", "input": "摔倒检测告警：时间16:55:33，位置阅览室角落", "output": "🚨紧急处理通知：请立即前往现场协助并记录事件详情。"}  
合规说明：所有语料均为项目团队自主编写或改编自公开知识问答语料，不涉及任何个人姓名、联系方式、地理位置或可识别身份信息。语料中仅包含技术性、学术性与场景描述类内容。

**三、脱敏技术与流程**

脱敏原则：遵循“目的明确、最少够用、公开透明、确保安全”的原则，对所有包含潜在隐私信息的样本进行不可逆处理。  
  
核心技术：  
人脸检测：YOLOv10s-face 模型（Ultralytics 框架）  
脱敏算法：像素化马赛克处理（15x15）  
元数据处理：清除 EXIF 信息，去除拍摄时间、GPS位置等元数据  
复核策略：自动化检测 + 人工抽检，漏检率 < 1%  
DeepSeek 微调语料：人工审查 + 正则清洗，移除可能含地名、人名的句子片段。  
  
处理流程：  
原始数据采集 → 数据备份 → 自动检测与脱敏（Python脚本） → 人工抽样核验 → EXIF清除 → 合规复核 → 脱敏数据集归档。

**四、合规性说明**

本项目的数据处理过程严格遵守 GB/T 35273-2020《信息安全技术 个人信息安全规范》的以下核心条款：  
• 最小化原则：仅保留与行为识别任务相关的像素信息；  
• 不可逆原则：人脸区域经像素化模糊后无法复原；  
• 匿名化原则：删除元数据与可追溯标识；  
• 安全存储原则：脱敏前数据仅存储于离线加密磁盘，不上传公共云端；  
• 教育场景合规性：处理后的数据符合校园教育数据使用的安全标准，具备在教育系统内安全部署与二次研究的能力。

**五、责任人与日期**

数据安全负责人：朱梓华

团队名称：青盾护卫

完成日期：2025年10月26日

签名/盖章：\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**附件**

1. 脱敏前后图像对比样例（含马赛克效果示例）



2. YOLOv10s-face 自动检测与脱敏 Python 脚本

from ultralytics import YOLO  
import cv2  
import os  
from tqdm import tqdm  
  
# === 配置部分 ===  
INPUT\_DIR = r"" # 原始图片路径  
OUTPUT\_DIR = r"" # 输出脱敏图片  
MODEL\_PATH = r"yolov10s-face.pt" # YOLO人脸检测模型  
PIXELATE = True # True=马赛克, False=模糊  
PIXEL\_BLOCK = 15 # 马赛克颗粒大小  
CONF\_THRES = 0.25 # 置信度阈值  
EXPAND\_RATIO = 0.2 # 扩大检测框比例  
  
os.makedirs(OUTPUT\_DIR, exist\_ok=True)  
  
# === 加载模型 ===  
model = YOLO(MODEL\_PATH)  
print("✅ YOLO人脸检测模型加载成功")  
  
# === 批处理所有图片 ===  
for filename in tqdm(os.listdir(INPUT\_DIR)):  
 if not filename.lower().endswith(('.jpg', '.jpeg', '.png')):  
 continue  
  
 path = os.path.join(INPUT\_DIR, filename)  
 img = cv2.imread(path)  
 if img is None:  
 continue  
  
 h, w = img.shape[:2]  
 results = model.predict(source=img, conf=CONF\_THRES, verbose=False)  
  
 for result in results:  
 boxes = result.boxes.xyxy.cpu().numpy() # [x1, y1, x2, y2]  
 for (x1, y1, x2, y2) in boxes:  
 # 扩大检测框  
 x1 = max(0, int(x1 - (x2 - x1) \* EXPAND\_RATIO))  
 y1 = max(0, int(y1 - (y2 - y1) \* EXPAND\_RATIO))  
 x2 = min(w, int(x2 + (x2 - x1) \* EXPAND\_RATIO))  
 y2 = min(h, int(y2 + (y2 - y1) \* EXPAND\_RATIO))  
  
 face = img[y1:y2, x1:x2]  
  
 if face.size == 0:  
 continue  
  
 # 打马赛克  
 if PIXELATE:  
 small = cv2.resize(face, (max(1, (x2 - x1) // PIXEL\_BLOCK), max(1, (y2 - y1) // PIXEL\_BLOCK)))  
 face\_blur = cv2.resize(small, (x2 - x1, y2 - y1), interpolation=cv2.INTER\_NEAREST)  
 else:  
 face\_blur = cv2.GaussianBlur(face, (31, 31), 30)  
 img[y1:y2, x1:x2] = face\_blur  
  
 cv2.imwrite(os.path.join(OUTPUT\_DIR, filename), img)  
  
print("🎯 所有人脸已脱敏处理完成！输出目录：", OUTPUT\_DIR)

3. DeepSeek 微调语料合规性说明摘要

4. 抽检统计表（检测召回率与漏检率）