上海大学 2023 ~ 2024 学年
课程报告成绩评价表

课程名称: **《模式识别》** 课程编号: **08306089**

报告名称: **基于 HOG+SVM 与 YOLO 的路标识别分类**

姓　　名: **郑力铖** 学　　号: **21122873**

报告评语:

报告成绩:

方案设计（20 分）		验收（20 分）		书面报告（60 分）			总分
可行性（10 分）	创新性（10 分）	规范性（10 分）	演示效果（10 分）	规范性（20 分）	完整性（20 分）	科学性（20 分）	

任课教师:

评阅日期:　　年　　月　　日

基于 HOG+SVM 与 YOLO 的路标识别分类

郑力铖 (21122873)

摘要: 作为交通系统的基本要素, 交通标志提供了关于驾驶员、行人等的道路状况的重要信息, 来降低事故风险。随着计算机视觉和人工智能的快速发展，交通标志识别算法已被应用于先进的驾驶员辅助系统和自动驾驶系统, 以帮助驾驶员和自动驾驶车辆准确获取道路信息。然而在实际应用中, 交通标志识别仍然具有挑战性。本文对比了经典的机器学习方法 HOG+SVM, 和通用视觉识别模型 YOLOv5 和 YOLOv8 方法，探讨在自动驾驶场景下的路标识别方法。
关键词: 图像识别, 支持向量机, 深度学习, 神经网络。

1 引言

1.1 问题提出

作为高级驾驶员辅助系统（ADAS）和自动驾驶系统（ADS）的重要组成部分，交通标志识别（TSR）技术可以帮助驾驶员和自动驾驶汽车获取重要的道路信息 [1]。交通标志识别 TSR 是一种为自动驾驶车辆识别道路上的交通标志的智能识别系统，是物体检测的一个子任务，类似于人脸识别。

TSR 系统通常由检测和分类两个阶段组成。检测通常使用形状和颜色来识别交通标志的特征，从而从自然场景中提取交通标志。更好的分类能更准确地识别的内容检测到的交通信号。交通标志的识别度，对于避免道路事故具有重大意义 [2]。本文通过对比不同方法的实际应用效果，旨在为自动驾驶系统选择最合适的路标识别方法提供参考。考虑到实时性、准确性和鲁棒性等因素，读者将能够更好地了解何种方法在特定条件下更为适用。

1.2 求解方案分析

一般来说，交通标志识别有两种检测方法，一种是传统方法（人工定制的特征），另一种是深度学习方法（学习具有深度神经网络的特征）。对于传统方法而言，整个实现流程通常包括的提取区域兴趣（包括交通标志），提取特征（例如 HOG），以及然后将这些特征发送到分类器（例如 SVM）。然而，传统方法在多类任务中的表现的性能较差。近年来，深度学习方法在许多任务中表现出优异的性能，例如图像分类和检测。对于对象检测，深度学习方法（例如 YOLO[3]，Faster R-CNN[4]）在主流基准上表现良好，准确度和速度都较传统方法有较大优势。

早在 2005 年，方向梯度直方图（HOG）就被提出，HOG 用来提取图片特征，并和支持向量机（SVM）结合，最早用来进行行人检测。HOG+SVM 这种经典的机器学习方法在过去取得了一定的成功，但随着技术的进步，人们开始寻求更加高效、准确的解决方案。

近年来，YOLO 算法以其高速度和高精度迅速走红。因此，本文深入研究了 YOLOv5 和 YOLOv8 这两种先进的目标检测模型。这两种模型都是基于 YOLO 模型的优化，利用

深度学习技术，通过神经网络层次结构和先进的目标检测算法，在图像中快速准确地识别路标。相较于传统方法，它们在处理复杂场景和多类别标识时表现更为出色。

1.3 论文概述

本文的其余部分的组织如下。在第二章我们介绍了梯度直方图与支持向量机(HOG+SVM)，以及对象检测算法 YOLO 的算法概述。在第三章我们实现了 HOG+SVM，YOLOv5 和 YOLOv8 在数据集上的实验代码，介绍了算法的总体框架，并做了一部分改进。第四章分别对 HOG+SVM，和 YOLO 的两种算法进行了对比，同时在 YOLOv5 和 YOLOv8 间的实验结果做了对比。最后在第五章做了讨论和结论。

2 相关算法概述

2.1 梯度直方图（HOG）

梯度直方图（Histogram of oriented gradient，简称 HOG）[5] 是应用在计算机视觉和图像处理领域，用于目标检测的特征描述器。这项技术是用来计算局部图像梯度的方向讯息的统计值。在 HOG 特征描述符中，梯度方向的分布，也就是梯度方向的直方图被视作特征。图像的梯度 (x 和 y 导数) 非常有用，因为边缘和拐角 (强度突变的区域) 周围的梯度幅度很大，并且边缘和拐角比平坦区域包含更多关于物体形状的信息。方向梯度直方图 (HOG) 特征描述符常和线性支持向量机 (SVM) 配合使用，用于训练高精度的目标分类器。

2.2 支持向量机（SVM）

支持向量机（Support Vector Machine，常简称为 SVM，又名支持向量网络）[6] 是在分类与回归分析中分析数据的监督式学习模型与相关的学习算法。给定一组训练实例，每个训练实例被标记为属于两个类别中的一个或另一个，SVM 训练算法创建一个将新的实例分配给两个类别之一的模型，使其成为非概率二元线性分类器。SVM 模型是将实例表示为空间中的点，这样映射就使得单独类别的实例被尽可能宽的明显的间隔分开。然后，将新的实例映射到同一空间，并基于它们落在间隔的哪一侧来预测所属类别。

2.3 对象检测算法 YOLO

YOLO（You Only Look Once）[3] 是一种流行的对象检测和图像分割模型，由华盛顿大学的 Joseph Redmon 和 Ali Farhadi 开发。它的主要特点是在一次前向传递中，同时完成对象定位和分类。相较于传统的对象检测方法，YOLO 在速度和准确性上取得了很大的突破。这使得它适用于实时目标检测任务，如视频分析和自动驾驶。

YOLOv5 已是目前最广泛使用的 YOLO 网络。相较于其前身有一系列改进，包括使用更强大的骨干网络、添加了超参数优化、采用自适应训练策略以适应不同数据集，以及支持多尺度训练，提高对小目标的检测能力等。并集成实验跟踪和自动导出到流行导出格式等工程上的新功能。YOLOv8 是 Ultralytics 的 YOLO 的最新版本。作为目前最尖端、最先进的 YOLO 版本，YOLOv8 在先前版本的成功基础上，引入了新功能和改进，以增强性能、灵活性和效率。

3 算法实现描述

3.1 HOG+SVM 实现

HOG+SVM 算法的实现流程如下表所述：

Algorithm 1 HOG+SVM 路标识别算法步骤

Input: 训练数据集；待预测数据；

Output: 预测数据的类别，与兴趣区域位置;

1: 加载数据;
2: 图片预处理：二值化，开闭运算;
3: 边缘检测提取候选区域;
4: 在候选区域中提取 HOG 特征图;
5: 用 SVM 分类器判别 HOG 特征图。**return** 候选区域框位置，SVM 类别预测值;

3.2 YOLO 实现

YOLO 实现与 HOG+SVM 不同，将照片直接缩放成网络输入的形状大小即可，接下来便是网络本身的计算，而后网络会输出候选区域框的位置和类别预测值。如下图1即为 YOLOv5n 的网络结构。YOLOv8n 在这之上修改了 CSP 部分的网络结构，由三个卷积层变为了两个卷积层，而增多了更多的跳层连接和额外的 Split 操作。

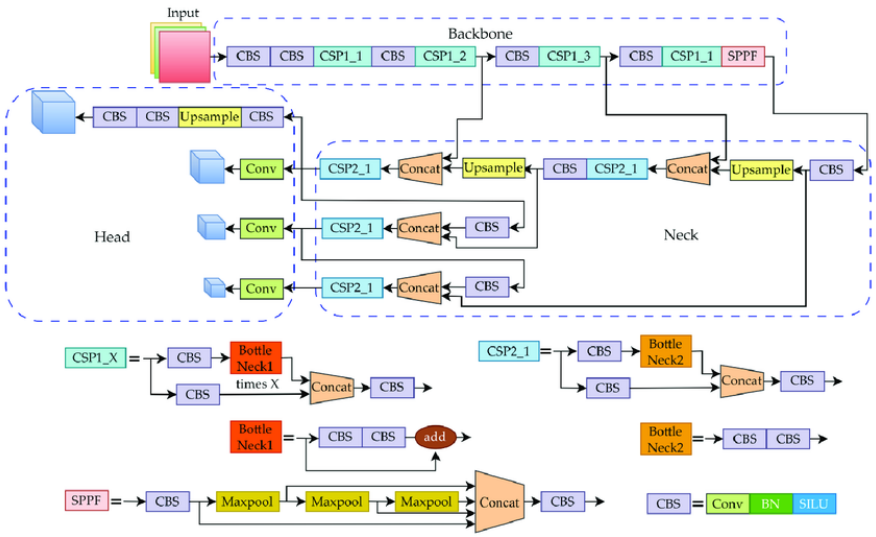

图 1：YOLOv5n 的网络结构。

4 实验描述

4.1 实验数据和实验方案

实验所使用的数据集来自于**kaggle.com**中的**road-mark-detection**数据集，该数据集已划分好训练集、验证集和测试集，并且已经缩放到 640*640 的图片大小。共含训练集 2167 张，验证集 417 张，测试集 192 张。

首先，考虑到机器学习算法对于多分类任务的短板，我们选取 1200 张已标注的训练数据上，仅选取 6 种路标训练 HOG 特征，并由 SVM 进行分类，计算每种类别的精确率、召回率和 F1 得分。分析该方法的性能。

在这之后，我们部署 YOLOv5 和 YOLOv8，v5 作为目前最广泛使用的 YOLO 引擎，对比 v8 作为目前的最优算法 SOTA。在 mAP、精确率、召回率和 F1 得分上作对比，比较两者性能。且与 HOG+SVM 的实验结果做对比，并分析原因。

4.2 实验的评价指标

本文主要使用精确率（Precision）、召回率（Recall）、F1 分数（F1 score）和平均精度均值（mAP）对模型进行评价。

精确率是指模型预测为正类别的样本中，实际为正类别的比例。召回率是指实际为正类别的样本中，模型成功预测为正类别的比例。F1 分数是精确率和召回率的调和平均，它综合考虑了两者之间的平衡。其三者的计算公式为：

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

$$\text{F1 score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

mAP 主要用于评估目标检测任务中的性能。它是所有类别的平均精度（AP）的平均值。每个类别的 AP 是计算 Precision-Recall 曲线下的面积。mAP 提供了对模型在多类别场景中的整体性能的综合评估。数学上，mAP 的计算公式可以表示为：

$$mAP = \frac{1}{n} \sum_{i=1}^{n} AP_i$$

其中，n 是类别的总数，AP_i 是第 i 个类别的 Average Precision。

4.3 HOG+SVM 实验及结果分析

在针对 1200 张图片学习针对的 6 种路标后，HOG+SVM 算法的精确率、召回率和 F1 得分如表1所示。因其得分相当不可看，因此也无法进行合适的绘图。这样结果的原因，一是由于本身轮廓检测需要背景单一纯净，训练现实场景的 roi 噪声过大而难以选准。第二是由于 SVM 算法本身针对多分类的复杂现实问题的性能一般。

表 1: HOG+SVM 的路标识别多指标分析

	精确率	召回率	F1 得分
直行标志	0.3706	0.3251	0.3464
左转标志	0.5889	0.3869	0.4670
右转标志	0.3750	0.4565	0.4118
禁止鸣笛	0.4999	0.5682	0.5319
人行横道	0.8095	0.5965	0.6869
禁止通行	0.7288	0.7049	0.7167

如图2所示，虽然在背景是阴天的情况下，在空中的车道指示路标可以非常好的被识别出，但在一般情况下，如图3中的高架桥下的真实情况所示，图片左上角的右转路标的 roi 明显较小，且将摩托车错检为了人行横道标识，并且这样错检的情况在测试中相当普遍，这导致了该方法在指标上的落后，现实场景下也难以应用。

图 2: 质量较好的 HOG+SVM 检测结果

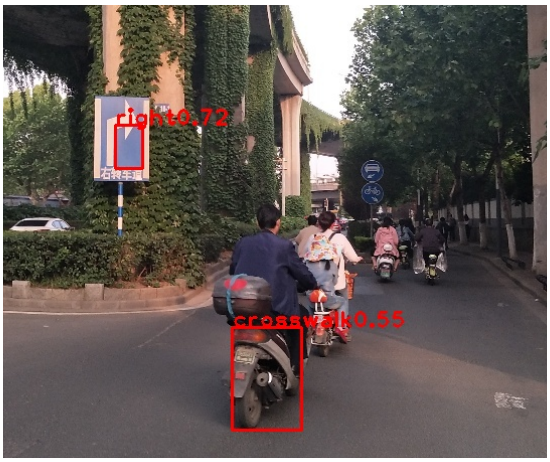

图 3: 明显错检与 roi 不一致问题

4.4 YOLOv5 与 v8 实验及结果分析

在经过 4.2 节实验，验证了 HOG+SVM 算法在现实场景中的局限性后，我们部署当下最流行的通用检测模型 YOLOv5，和 YOLO 的最新 SOTA YOLOv8。实验时我们自己设计了一个绘制准确度-召回率图的代码，因 v5 与 v8 两者的 PR 曲线差距不大，本文即以 v5 为例。

如图4，YOLOv5 达成了所有分类 mAP@0.5 值为 0.908 的高输出。黄车道线 Liniya2 由于与白车道线 Liniya1 相似，且数据量较少，容易产生错检，未能非常好的识别。而其他类别有非常好的收敛和学习，mAP@0.5 值均超过 0.8。

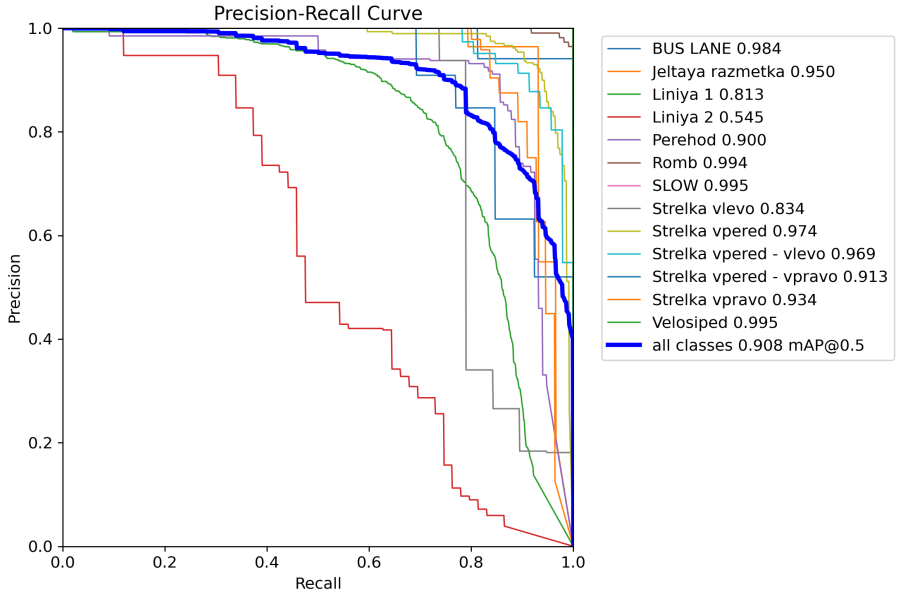

图 4: YOLOv5 的准确度-召回率图，YOLOv8 与之类似。

如图5，对 YOLOv5 的归一化混淆矩阵的分析中，我们发现公交车标识、慢行标识、自行车道三种标识，网络都能百分百准确地识别出。这也是由于这些标识本身就含有文字或含有的信息较多。而对于车道线这样的图形信息较少的标识，如图中的 Liniya1 与 Liniya2，网络还未能做到非常好的识别。综合来看，如图6中的 mAP50-95，可以看到 YOLOv8 在精确性能上优于 YOLOv5。

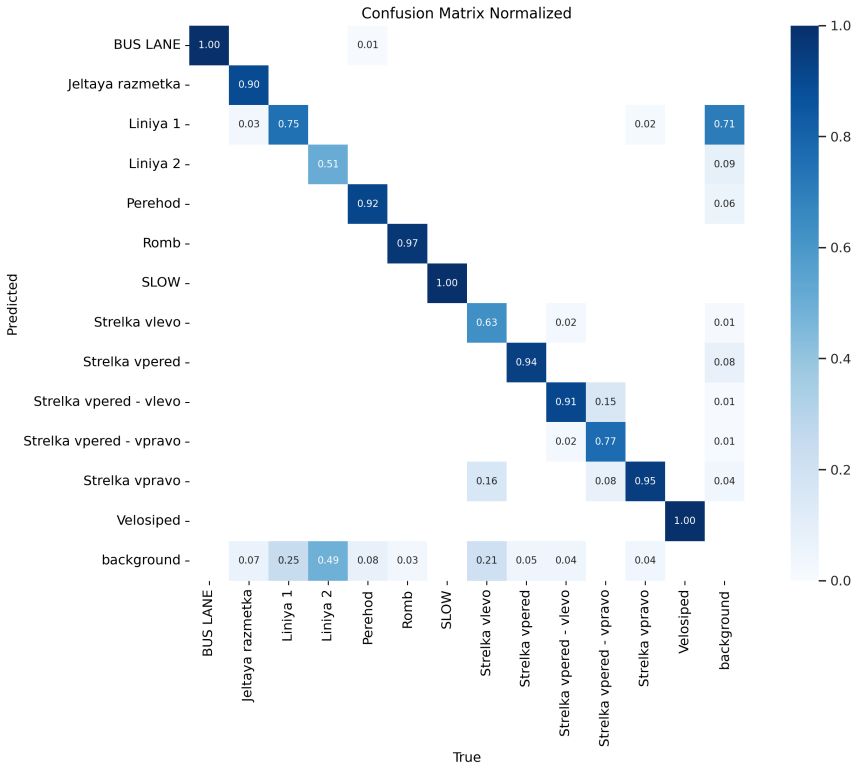

图 5：YOLOv5 的归一化混淆矩阵图，YOLOv8 与之类似。

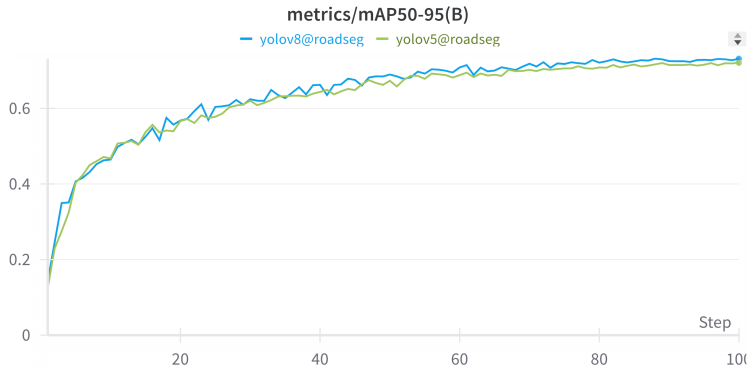

图 6：YOLOv5 与 YOLOv8 的 mAP 对比。YOLOv8 略优与 YOLOv5。

5 总结与展望

本文对比了经典的机器学习方法 HOG+SVM，与 YOLOv5 和 YOLOv8 在现实环境下的路标识别能力。设计了一个针对现实场景的算法对比实现，结果看出 HOG+SVM 算

法虽然不需要强大的算力进行训练，但实际效果也较差，无法投入实际运用。YOLO 网络在测试中表现出了强大的能力，但是对应的也需要一定时间用于训练。关于 YOLOv5 和 YOLOv8，在精准率和召回率这两项指标上两者各有千秋，但在 mAP 指标上 YOLOv8 一致保持优于 YOLOv5。YOLOv8 在精确性能上优于 YOLOv5，而 YOLOv5 因在网络参数的优势，训练速度上是优于 YOLOv8，因此在实际应用中，要综合考量精准性能和部署速度，从而选择最适合的算法。

对于在这一方向上的一些未来展望，近来的一些工作基于改善网络结构，对类似路标的小目标检测做了特殊优化。虽然在 mAP 指标上做到了一定提升，但是这些工作并没有提高识别的准确率。并且这一些工作都基于较老的 YOLO 网络，并没有在最新的 YOLOv8 网络基础上进行优化，因此也无法进行合适的比较。未来可以借鉴类似的一些优化经验，并迁移到 YOLOv8 或更新的最优网络基础上进行优化，进一步提高性能。

6 学习体会和建议

本学期在模式识别课上，方昱春老师深入地介绍了一些模式识别的理论基础，与不同算法模型和技术，以及它们在一些不同场景中的应用。建立起一定的理论基础后，通过研讨和小组作业等形式，上手实际项目，让我更好地理解了算法模型的工作原理、优缺点以及如何调整参数以提高性能。这培养了我解决问题的能力，从而选择合适的方法，并在实践中更加灵活地应用知识。希望今后能在课程中加入更多前沿领域的最新科研成果分享，并与小组作业有机结合起来，让同学们更好地感受到新成果带来的进步，并能有机会启发同学们在今后参与进前沿的计算机科研的的工作中去的热情。

参考文献

[1] J. Chen, K. Jia, W. Chen, *et al.*, "A real-time and high-precision method for small traffic-signs recognition," *Neural Computing & Applications*, vol. 34, pp. 2233–2245, 2022.

[2] R. Zhang, Z. He, H. Wang, F. You, and K. Li, "Study on self-tuning tyre friction control for developing main-servo loop integrated chassis control system," *IEEE Access*, vol. 5, 2017.

[3] "Redmon j, divvala s, girshick r, farhadi a (2016) you only look once: unified, real-time object detection, pp 779–788."

[4] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: towards real-time object detection with region proposal networks," *IEEE Trans Pattern Anal Mach Intell*, vol. 39, 2016.

[5] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1, pp. 886–893 vol. 1, 2005.

[6] M. Hearst, S. Dumais, E. Osuna, J. Platt, and B. Scholkopf, "Support vector machines," *IEEE Intelligent Systems and their Applications*, vol. 13, no. 4, pp. 18–28, 1998.

A 附录

核心代码节选:

```
# HOG + SVM
from skimage import feature as ft
import cv2
def hog_feature(img_array, resize=(64,64)):
    """extract hog feature from an image."""
    img = cv2.cvtColor(img_array, cv2.COLOR_BGR2GRAY)
    img = cv2.resize(img, resize)
    bins = 9
    cell_size = (8, 8)
    cpb = (2, 2)
    norm = "L2"
    features = ft.hog(img,
                      orientations=bins,
                      pixels_per_cell=cell_size,
                      cells_per_block=cpb,
                      block_norm=norm,
                      transform_sqrt=True)
    return features

# YOLO training
from wandb.integration.ultralytics import add_wandb_callback
import wandb
from ultralytics import YOLO

wandb.init(project="YOLOv5")
model = YOLO("./yolov5nu.pt") # ./yolov8n.pt
add_wandb_callback(model, enable_model_checkpointing=True)
results = model.train(project="YOLOv5",
                      data="./datasets/data.yaml",
                      epochs=100,
                      save_period=10,
                      seed=42)
model_best = YOLO("./YOLOv5/train/weights/best.pt")
metrics = model_best.val()
wandb.finish()

# Calculate PR and F1
for img_name, obj_label in label_dict.items():
    obj_predict = predict_dict[img_name]
```

```
    for obj, coords in obj_predict.items():
        img_num_predict[obj] += len(coords)
    for obj, coords in obj_label.items():
        img_num_per_cls[obj] += len(coords)
        if obj in obj_predict.keys():
            for coord1 in coords:
                for coord2 in obj_predict[obj]:
                    if compute_iou(coord1, coord2)>=iou1:
                        tp_count[obj] +=1
for cls_name in cls_names:
    p = tp_count[cls_name]/img_num_predict[cls_name]
    r = tp_count[cls_name]/img_num_per_cls[cls_name]
    f1 = 2*p*r/(p+r)
    pre_rec[cls_name] = [p, r, f1]
return pre_rec

# PR curve drawing
def plot_pr_curve(px, py, ap,
                  save_dir=Path("pr_curve.png"),
                  names=()):
    """Plots a precision-recall curve."""
    fig, ax = plt.subplots(1, 1, figsize=(9, 6))
    py = np.stack(py, axis=1)

    for i, y in enumerate(py.T):
        ax.plot(px, y, linewidth=1,
                label=f"{names[i]} {ap[i, 0]:.3f}")

    ax.plot(px, py.mean(1), linewidth=3, color="blue",
            label="all classes %.3f mAP@0.5"%ap[:, 0].mean())
    ax.set_xlabel("Recall")
    ax.set_ylabel("Precision")
    ax.set_xlim(0, 1)
    ax.set_ylim(0, 1)
    ax.legend(bbox_to_anchor=(1.04, 1), loc="upper left")
    ax.set_title("Precision-Recall Curve")
    fig.savefig(save_dir, dpi=250)
    plt.close(fig)
```