光猫外观缺陷检测算法说明文档

一、产品缺陷分类及评分机制

1、缺陷类别

系统通过拍摄光猫的三个侧面进行图像检测，检测以下四类外观瑕疵：

1)外壳破损：识别外壳上的裂痕、断裂等严重影响设备功能或外观的问题。

2)标签破损：检查标签的清晰度，是否存在缺失、撕裂等情况。

3)外壳泛黄：检测设备外壳是否存在颜色的变黄老化情况。

4)壳体划痕：识别外壳表面是否有较明显的划痕，影响美观。

2.评分标准：

为了确保光猫路由器的质量控制与产品审查，我们制定了以下缺陷评分规则。此评分规则将帮助质量检验团队对每一台路由器进行详细评估，确保每个产品的质量达到公司标准。以下是各种缺陷的详细评分标准：

缺陷类型及其评分规则

1. 外壳破损

轻微：小于 1 平方厘米的破损，扣除 1-3 分。

中等：1-3 平方厘米的破损，扣除 4-6 分。

严重：超过 3 平方厘米的破损，扣除 7-10 分。

2. 标签破损

轻微：标签边缘轻微脱落或有轻微划痕，扣除 1-3 分。

中等：标签部分脱落或文字模糊不清，扣除 4-6 分。

严重：标签大面积损毁或完全不可识别，扣除 7-10 分。

3. 外壳泛黄

轻微：外壳轻微变色，但不影响整体外观，扣除 1-3 分。

中等：外壳明显泛黄，影响美观，扣除 4-6 分。

严重：外壳严重泛黄，影响产品识别或美观程度极大降低，扣除 7-10 分。

4. 壳体划痕

轻微：表面有细小的划痕，长度小于 2 厘米，扣除 1-3 分。

中等：划痕明显，长度 2-5 厘米，扣除 4-6 分。

严重：划痕深且长，超过 5 厘米，扣除 7-10 分。

总体评分方法

每台光猫路由器的初始分为 100 分。根据上述评分规则，每检测到一个缺陷，根据其严重程

度相应扣分。

二、算法模型训练以及部署

1、模型算法介绍

YOLOv5 算法简介：

YOLOv5（You Only Look Once version 5）是一款非常流行的目标检测模型，广泛用于实时目标检测任务。这个模型是在 YOLO 系列的前几个版本上进一步优化和改进的结果。

1) 模型架构： YOLOv5 的架构主要包括：输入层、多个卷积层、BN（Batch Normalization）层、Leaky ReLU 激活函数、多尺度预测和输出层。这种架构能够在单个前向传递中同时预测多个边界框和类别概率。
2) 图像处理： 在进行目标检测前，YOLOv5 首先将输入图像调整到一个统一的尺寸（例如，640x640 像素）。这样做是为了保证不同尺寸和比例的图像都能被模型有效处理。
3) 特征提取： YOLOv5 使用 CSPNet（Cross Stage Partial Network）作为其主要的特征提取网络。CSPNet 通过减少计算量来增加网络的学习能力和运行速度。
4) 边界框预测： 模型在多个尺度上进行预测，使用不同大小的特征图来预测不同尺寸的边界框。这样可以增加模型对于小物体的检测能力。
5) 非极大值抑制（NMS）： 在边界框的预测后，YOLOv5 使用非极大值抑制来去除重叠的预测框，只保留最有可能包含目标的框。

CenterNet 算法简介：

1) 模型架构： CenterNet 主要由一个卷积神经网络构成，该网络负责从输入图像中提取特征。网络的输出是三个主要部分：中心热图（heatmap），宽高预测（size prediction），以及偏移量（offsets）。
2) 中心点检测：
3) 热图预测：模型预测一个热图，每个类别有其对应的热图层。热图的峰值位置代表物体中心的可能位置。
4) 宽高预测：对于热图中的每一个中心点，模型还需要预测该目标的宽度和高度。

5) 偏移量预测：由于将输出空间下采样，中心点的位置可能不够精确，模型还会预测一个偏移量来校正这一下采样带来的误差。

6) 损失函数： CenterNet 使用了多任务损失来同时训练中心点热图、宽高预测和偏移量预测。这通常包括二值交叉熵损失用于热图的训练，L1 损失用于宽高和偏移量的训练。

7) 后处理： 在网络输出之后，通过查找热图上的局部最大值来确定中心点的位置，然后使用宽高预测和偏移量预测来计算出目标的精确边界框（bounding box）。

2、模型训练步骤

a. 数据采集与准备：

使用工业相机从固定视角拍摄光猫图片，以确保图像的一致性和可比性。

收集包含四种主要缺陷（标签破损、刮痕、泛黄和裂痕）的图像，每种缺陷至少需 800 张图片，确保数据的多样性和充分性。同时，需要大量无缺陷的光猫图片，用于训练中的负样本，帮助模型更好地学习识别无缺陷的正常情况。

b. 模型训练与参数调整：

对数据进行合理的训练测试划分，通常建议使用 70%的数据进行训练，30%的数据用于测试，以评估模型的泛化能力。设置训练轮次至少 200 轮（epochs），以确保模型有足够的时间学习数据特征。监控训练精度和测试精度，两者差距不应过大，以防模型过拟合。选择随机梯度下降（SGD）作为优化器，调节学习率和动量，优化训练过程并提高模型的收敛速度和性能。

c. 模型评估与优化：

通过不断的测试和验证，对模型进行评估，确保其在处理实际图像时的准确性和可靠性。分析模型在不同类型缺陷识别上的表现，识别出模型的弱点并进行针对性的优化。实施交叉验证等技术，进一步验证模型的稳定性和泛化能力，避免数据分布不均导致的偏差问题。

3、模型部署

a. 系统要求

Python: 推荐使用 Python 3.7 或以上版本，确保与现有库的兼容性。

依赖库: Flask, OpenCV, PyTorch, torchvision, Pillow 等。

硬件: 推荐使用具有 CUDA 支持的 GPU，以加速模型推理过程。

b. 安装必要的 Python 库。

创建一个 Flask 应用，用于处理前端发来的图片请求和返回检测结果。

设置路由，接收图像文件，处理并返回检测结果。

c. 集成 YOLOv5 模型:

将训练好的 YOLOv5 模型文件 (best.pt) 集成到 Flask 应用中。

使用 PyTorch 加载模型，并设置为评估模式以进行推理。

编写图像处理函数，包括图像的加载、转换（调整大小、归一化）和缺陷检测。

用户可以通过按钮上传图像，后台 Flask 服务处理这些图像并将带有缺陷标注的图像返回显示。

d. 启动服务:

运行 Flask 应用，确保服务正常监听请求。