Avaliação Prática - AV3: Sistema de Detecção de Buracos com YOLOv5

Descrição do Projeto

Este projeto foi reorientado para focar especificamente na **detecção de buracos em estradas**, utilizando uma abordagem baseada em aprendizado profundo. O objetivo principal é desenvolver um sistema em Python capaz de identificar buracos em imagens, empregando o detector de objetos YOLOv5.

Inicialmente, o projeto visava uma detecção genérica de objetos urbanos com YOLOv8. No entanto, devido à solicitação de focar em buracos e às dificuldades encontradas para obter pesos YOLOv8 pré-treinados publicamente disponíveis para essa classe específica, optou-se por adaptar o pipeline para utilizar um modelo YOLOv5, na esperança de encontrar pesos treinados para buracos.

Desafios e Solução Adotada (YOLOv5)

Durante o desenvolvimento, enfrentamos os seguintes desafios:

1. **Incompatibilidade YOLOv5 -> YOLOv8:** Tentamos integrar pesos YOLOv5 (encontrados no GitHub com nome sugestivo de `pothole_yolov5s.pt`) na biblioteca Ultralytics YOLOv8, mas foi constatada incompatibilidade direta entre os formatos.
2. **Indisponibilidade de Pesos YOLOv8 para Buracos:** Realizamos buscas extensivas por pesos YOLOv8 pré-treinados especificamente para detecção de buracos em repositórios públicos (GitHub, Hugging Face), mas não encontramos modelos prontamente disponíveis para download direto.
3. **Conversores:** Não foram encontradas ferramentas oficiais ou confiáveis para converter pesos YOLOv5 para o formato YOLOv8.
4. **Verificação dos Pesos YOLOv5:** Após baixar o arquivo `pothole_yolov5s.pt` e integrá-lo ao pipeline YOLOv5 via PyTorch Hub, a execução com a imagem `buracos.jpg` (fornecida pelo usuário e contendo buracos visíveis) revelou que o modelo detectou um caminhão e um carro, mas **não detectou os buracos**. Além disso, o carregamento do modelo revelou que ele contém as 80 classes padrão do dataset COCO, e não uma classe específica para buracos. Isso indica fortemente

que, apesar do nome do arquivo, os pesos carregados são do modelo YOLOv5s padrão, e não um modelo treinado especificamente para buracos.

Diante disso, a solução adotada foi **manter o pipeline funcional com YOLOv5 via PyTorch Hub**, utilizando os pesos `pothole_yolov5s.pt` (que se revelaram ser os pesos padrão YOLOv5s). O sistema está tecnicamente funcional, mas **não realiza a detecção específica de buracos** com os pesos atualmente carregados.

Técnicas de Processamento Digital de Imagens Aplicadas

O desenvolvimento manteve o uso de Python com OpenCV, NumPy e Matplotlib. As técnicas aplicadas incluem:

1. **Conversão para Escala de Cinza:** Pré-processamento para simplificar a imagem.
2. **Filtragem Gaussiana:** Suavização para redução de ruído (kernel 7x7).
3. **Detecção de Bordas Canny:** Identificação de contornos.
4. **Detecção de Objetos com YOLOv5 (Modelo Padrão):** Utilização do modelo `pothole_yolov5s.pt` (identificado como YOLOv5s padrão) carregado via PyTorch Hub para identificar objetos gerais na imagem.

Resultados Obtidos (com Imagem `buracos.jpg`)

A seguir são apresentados os plots gerados com Matplotlib, exibindo os resultados das etapas de processamento aplicadas à imagem `buracos.jpg` fornecida.

Figura 1: Imagem Original (`buracos.jpg`)

Imagem Original (Buracos)

Descrição: Plot da imagem de entrada colorida, mostrando uma rua com buracos visíveis e um caminhão.

Figura 2: Imagem em Escala de Cinza

Escala de Cinza (Buracos)

Descrição: Plot do resultado da conversão da imagem original para escala de cinza.

Figura 3: Imagem com Filtro Gaussiano Aplicado

Filtro Gaussiano (Buracos)

Descrição: Plot da imagem original após a aplicação de um filtro Gaussiano 7x7.

Figura 4: Detecção de Bordas com Canny

Descrição: Plot do resultado da aplicação do detector de bordas Canny.

Figura 5: Detecção de Objetos com YOLOv5 na Imagem `buracos.jpg`

Detecção YOLOv5 (Buracos)

Descrição: Plot do resultado da aplicação do modelo YOLOv5 (`pothole_yolov5s.pt` , identificado como padrão) na imagem `buracos.jpg` . O modelo detectou um caminhão (`truck`) com confiança 0.71 e um carro (`car`) com confiança 0.38. Nenhum buraco foi detectado, apesar de estarem visíveis na imagem.

Análise e Comentários sobre os Resultados

As técnicas clássicas de PDI foram aplicadas conforme esperado (Figuras 2, 3, 4).

A aplicação do modelo YOLOv5 (Figura 5) na imagem real `buracos.jpg` confirmou que os pesos carregados (`pothole_yolov5s.pt`) correspondem ao modelo YOLOv5s padrão treinado no dataset COCO. O modelo identificou corretamente o caminhão e um carro ao fundo, mas falhou em detectar os buracos evidentes na via.

Isso reforça a conclusão de que **não conseguimos localizar e integrar um modelo YOLO (v5 ou v8) publicamente disponível e comprovadamente treinado para a detecção específica de buracos**. O arquivo de pesos encontrado, apesar do nome, não era especializado.

O projeto cumpriu o requisito de aplicar técnicas de PDI e um modelo de detecção (YOLOv5 padrão), mas a tarefa específica de detectar buracos não pôde ser realizada devido à falta de um modelo treinado adequado.

Conclusão

Este trabalho adaptou o sistema para tentar focar na detecção de buracos, superando desafios de compatibilidade ao integrar um modelo YOLOv5 via PyTorch Hub. No entanto, a investigação revelou que os pesos encontrados, apesar da nomenclatura, eram do modelo padrão YOLOv5s (COCO), incapaz de detectar buracos. O pipeline está funcional para detecção de objetos gerais com YOLOv5, mas a detecção específica de buracos exigiria encontrar ou treinar um modelo dedicado.

O projeto demonstra a aplicação de PDI e YOLO, mas evidencia a dificuldade em obter modelos pré-treinados específicos para certas tarefas e a importância de verificar a origem e o conteúdo dos pesos utilizados. A documentação reflete a jornada técnica, incluindo as limitações encontradas e as soluções implementadas.