

Avaliação Prática - AV3: Sistema Híbrido de Processamento de Imagem com YOLOv8

Descrição do Projeto

Este projeto explora a aplicação de técnicas de Processamento Digital de Imagens (PDI) em conjunto com a inteligência artificial para análise de cenas urbanas. O objetivo principal foi desenvolver um sistema em Python capaz de detectar e analisar objetos (como veículos e pedestres) em imagens de ruas, utilizando uma abordagem híbrida que combina métodos clássicos de PDI com o detector de objetos YOLOv8. O contexto é o desenvolvimento de ferramentas para monitoramento e análise urbana, onde a extração automática de informações de imagens é cada vez mais relevante.

Técnicas de Processamento Digital de Imagens Aplicadas

O desenvolvimento foi realizado utilizando Python, com suporte das bibliotecas OpenCV, NumPy, Matplotlib e Ultralytics (para YOLOv8). Foram aplicadas as seguintes técnicas de PDI:

- 1. Conversão para Escala de Cinza:** Transformação da imagem colorida original para uma representação monocromática. Esta é uma etapa comum de pré-processamento que simplifica a imagem, reduzindo a dimensionalidade dos dados (de 3 canais de cor para 1 canal de intensidade) e sendo um requisito para certos algoritmos, como a detecção de bordas Canny.
- 2. Filtragem Gaussiana:** Aplicação de um filtro de suavização Gaussiano (com kernel 7x7 neste projeto). Este filtro ajuda a reduzir ruídos de alta frequência na imagem, borrando-a ligeiramente. Foi utilizado como uma etapa de pré-processamento antes de aplicar o YOLOv8, para avaliar seu impacto na detecção de objetos.
- 3. Detecção de Bordas Canny:** Algoritmo para identificar bordas significativas na imagem. Ele opera sobre a imagem em escala de cinza e geralmente após alguma suavização (implícita no algoritmo ou aplicada previamente) para destacar contornos e limites de objetos.

4. Detecção de Objetos com YOLOv8: Utilização do modelo pré-treinado `yolov8n.pt` para identificar e localizar múltiplos objetos na imagem. O YOLO (You Only Look Once) é um detector de objetos em tempo real de última geração que fornece a classe do objeto detectado (ex: carro, pessoa) e sua localização através de uma caixa delimitadora (bounding box).

Resultados Obtidos

A seguir são apresentados os plots gerados com Matplotlib, exibindo os resultados das etapas de processamento aplicadas a uma imagem de teste de uma cena urbana.

Figura 1: Imagem Original

Figura 1: Imagem Original



Descrição: Plot da imagem de entrada colorida, representando uma rua com diversos veículos (carros, ônibus), pedestres e elementos urbanos. Esta é a base para todas as análises subsequentes.

Figura 2: Imagem em Escala de Cinza

Figura 2: Imagem em Escala de Cinza



Descrição: Plot do resultado da conversão da imagem original para escala de cinza. A informação de cor foi removida, restando apenas as intensidades luminosas.

Figura 3: Imagem com Filtro Gaussiano Aplicado

Figura 3: Imagem com Filtro Gaussiano (7x7)



Descrição: Plot da imagem original após a aplicação de um filtro Gaussiano 7x7. Nota-se uma leve suavização/borrão, que visa reduzir ruídos antes de outras etapas ou da detecção com YOLOv8.

Figura 4: Detecção de Bordas com Canny

Figura 4: Detecção de Bordas (Canny)



Descrição: Plot do resultado da aplicação do detector de bordas Canny. As linhas brancas representam as bordas detectadas na imagem, destacando os contornos dos objetos e estruturas.

Figura 5: Detecção de Objetos YOLOv8 na Imagem Original

Figura 5: Detecção YOLOv8 (Imagen Original)



Descrição: Plot do resultado da aplicação do YOLOv8 diretamente na imagem original. As caixas coloridas delimitam os objetos detectados, com rótulos indicando a classe (ex: 'car', 'person', 'bus') e a confiança da detecção. Diversos veículos e pedestres foram identificados.

Figura 6: Detecção de Objetos YOLOv8 na Imagem com Filtro Gaussiano

Figura 6: Detecção YOLOv8 (Imagem com Filtro Gaussiano)



Descrição: Plot do resultado da aplicação do YOLOv8 na imagem pré-processada com o filtro Gaussiano (Figura 3). As detecções são mostradas de forma similar à Figura 5.

Análise e Comentários sobre os Resultados

As técnicas clássicas de PDI (escala de cinza, filtro Gaussiano, Canny) foram aplicadas com sucesso, gerando as saídas esperadas e demonstrando etapas fundamentais de pré-processamento e extração de características, conforme visualizado nos plots das Figuras 2, 3 e 4.

A aplicação do YOLOv8 (Figuras 5 e 6) demonstrou sua capacidade de detectar múltiplos objetos de diferentes classes na cena urbana complexa.

Comparação YOLOv8 (Original vs. Pré-processada): Comparando visualmente a Figura 5 (YOLO na original) e a Figura 6 (YOLO na imagem com filtro Gaussiano), observamos que o pré-processamento com filtro Gaussiano não pareceu trazer uma melhoria significativa na detecção para esta imagem e configuração específicas. Em ambos os casos, o YOLOv8 identificou um número similar de objetos principais (carros, pessoas, ônibus). Uma análise mais detalhada dos arquivos CSV gerados (não incluídos no relatório, conforme as diretrizes) poderia revelar pequenas diferenças nas confianças das detecções ou na precisão das caixas delimitadoras, mas visualmente, o impacto do filtro Gaussiano como pré-processamento para o YOLOv8 foi mínimo neste teste. Em

alguns cenários com mais ruído, a suavização poderia ser benéfica, mas aqui não alterou drasticamente o resultado da detecção pela rede neural.

O projeto cumpriu o requisito de aplicar no mínimo três técnicas de PDI (Escala de Cinza, Filtro Gaussiano, Canny) além da detecção com YOLOv8, e realizou a comparação solicitada entre a detecção na imagem original e na pré-processada.

Conclusão

Este trabalho demonstrou com sucesso a implementação de um sistema básico em Python para análise de imagens urbanas, combinando técnicas clássicas de PDI e o detector de objetos YOLOv8. Foram aplicadas e visualizadas através de plots Matplotlib as técnicas de conversão para escala de cinza, filtragem Gaussiana e detecção de bordas Canny. O YOLOv8 foi utilizado para detectar objetos na imagem original e em uma versão pré-processada, permitindo uma análise comparativa. Os resultados indicam que, para esta imagem específica, o pré-processamento com filtro Gaussiano teve impacto limitado na performance do YOLOv8. O projeto estabelece uma base funcional que pode ser expandida com técnicas de fusão mais avançadas e análises mais profundas dos objetos detectados.