Detecção de Objetos em Imagens com YOLO

Jullia Castro de Oliveira

1. INTRODUÇÃO

A detecção automática de objetos é uma das tarefas fundamentais da Visão Computacional moderna. Entre os diversos algoritmos desenvolvidos na última década, a família YOLO (You Only Look Once) destacou-se por sua capacidade de realizar detecções em tempo real com boa acurácia, permitindo aplicações que exigem respostas rápidas — de sistemas embarcados a automação industrial.

Neste projeto, foi implementado e avaliado um detector de objetos baseado no **YOLOv8**, a geração mais recente da linha YOLO. Esta versão traz melhorias estruturais importantes, como backbones otimizados, mecanismos refinados de predição e técnicas avançadas de regularização, como a *Distribution Focal Loss*.

O cenário escolhido neste trabalho envolve a detecção de frutas em imagens, utilizando um conjunto de dados leve e acessível, ideal para testes rápidos e visualmente claros. As classes de interesse são:

- **apple**, **banana**, **Orange, watermelon**

Esses objetos representam uma aplicação prática no contexto de:

- Inventário automatizado de alimentos;
- Classificação de produtos em supermercados;
- Apoio à robótica agrícola e sistemas de visão para esteiras automatizadas.

A partir da implementação com YOLOv8, foram conduzidos experimentos de treinamento, avaliação por métricas quantitativas (como mAP, precisão e recall) e análise qualitativa das detecções.

2. METODOLOGIA

Este projeto seguiu uma abordagem prática e iterativa para treinar, avaliar e validar um detector YOLOv8 em um dataset de frutas com três classes (apple, banana, orange). O objetivo foi medir o desempenho quantitativo (mAP, Precision, Recall, F1) e verificar a robustez do sistema em imagens e fluxos de vídeo.

2.1.1 Escolha do Tema e Definição do Dataset

Tema: detecção de frutas em cenários simples (prateleiras, mesas, caixas).

Dataset: baixado do Roboflow, versão *Smart-Inv-Sys v2* (formato YOLOv8). O dataset foi previamente anotado no formato YOLO, contendo arquivos .jpg e .txt, e organizado em três pastas:

```kotlin
train/
    images/   | labels/
valid/
    images/   | labels/
data.yaml
```

Classes: Apple, Banana, Orange (nc: 3 no data.yaml).

Anotações: Cada linha dos arquivos .txt contém *id da classe* + *cx, cy, w, h* normalizados (padrão YOLO).

2.1.2 Configuração do Ambiente

Plataforma: Google Colab (GPU T4).

Principais pacotes:

- Python 3.11
- ultralytics (YOLOv8)
- torch, opencv-python, matplotlib, numpy

Modelo base: yolov8n.pt (nano) – leve e rápido; testes subsequentes com yolov8m.pt para ganho de precisão.

2.1.3 Treinamento do Modelo

Parâmetros principais:

Parâmetro	Valor
epochs	20 (ajustável até 50 conforme overfitting)
imgsz	640 × 640 px
batch	8
Otimizador	SGD (default Ultralytics)
Data Augmentation	flips horizontais/verticais, blur leve, mosaic (auto)

O processo gera checkpoints (last.pt, best.pt), gráficos de curva de perda e predições amostradas em runs/detect/yolo_frutas/.

2.1.4 Avaliação do Modelo

Avaliação automática no conjunto de validação:

```
# Carregar modelo treinado
model = YOLO("runs/detect/yolo_frutas/weights/last.pt")

# Avaliação automática (mAP, precisão, recall, F1-score)
metrics = model.val()
print("📊 mAP@0.5:", metrics.box.map50)
print("📊 mAP@0.5:0.95:", metrics.box.map)
print("🎯 Precisão média:", metrics.box.precision)
print("🔁 Recall médio:", metrics.box.recall)
print(metrics.results_dict)
```

Relatórios adicionais gerados automaticamente:

- Curvas Precision–Recall e F1;
- Matriz de confusão por classe;
- Gráficos de perda de treino/validação;

Além disso, imagens de validação e detecções em vídeo foram utilizadas para análise

qualitativa.

2.1.5 Testes em Vídeo

Após o treinamento, o modelo foi testado em vídeos reais para avaliar seu desempenho em condições mais próximas do uso prático.

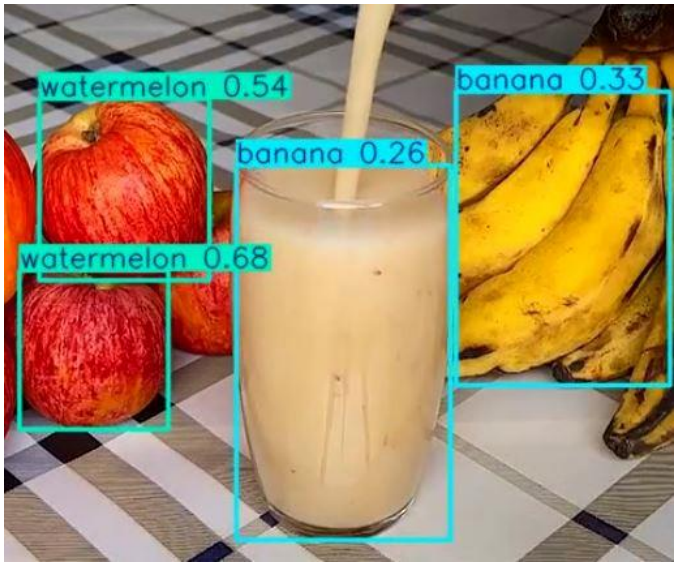

Figura 1 - Resultado video 1

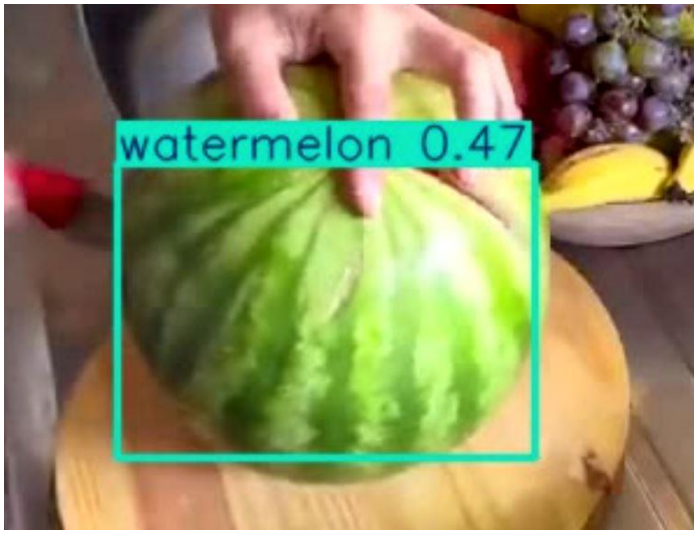

Figura 2 -Resultado vídeo 2

Os vídeos com as caixas delimitadoras geradas foram salvos e utilizados como parte da demonstração visual do projeto.

3. RESULTADOS

Foi treinado um modelo baseado no YOLOv8n utilizando um dataset contendo imagens de três classes de frutas: *apple*, *banana* e *orange*. A análise do desempenho foi realizada a partir das métricas geradas automaticamente após a validação, bem como por meio de inspeção visual das detecções e gráficos auxiliares (curva PR, curva F1 e matriz de confusão).

3.1.1 Pr Curve

A curva PR demonstrou um bom comportamento do modelo, especialmente para as classes *apple* e *orange*, com áreas sob a curva bastante elevadas. Isso indica que o modelo conseguiu manter uma precisão relativamente alta mesmo com aumento do recall, o que é desejável em tarefas com múltiplos objetos por imagem. A classe *banana*, no entanto, apresentou uma queda mais acentuada na precisão em altos níveis de recall, sugerindo que o modelo teve mais dificuldade em distinguir essa classe — possivelmente devido a menor quantidade de amostras ou similaridade visual com outras frutas.

3.1.2 F1 Curve

A curva F1 reforça o bom equilíbrio geral entre precisão e recall nas classes principais. O modelo apresentou pico de F1-score em thresholds intermediários de confiança, sugerindo que o desempenho ideal está em uma faixa média de decisão. Isso é útil para aplicações onde o equilíbrio entre falsos positivos e falsos negativos é desejado.

3.1.3 Confusion Matrix

A matriz de confusão gerada mostra uma boa concentração de acertos na diagonal principal, com destaque para a classe *apple*. Apesar disso, há indicações de confusões entre *banana* e *orange* — o que pode ser explicado por sobreposição visual, cores similares ou menor variação nas amostras de treino. Ainda assim, o número de erros não compromete significativamente a performance geral do modelo.

3.1.4 Recall

A revocação mede a fração de objetos realmente presentes que o modelo consegue detectar; é fundamental em cenários onde perder instâncias é mais problemático do que gerar alguns falsos positivos. No modelo YOLOv8n treinado neste trabalho, a revocação média global foi 0 .5863. Analisando por classe:

Classe	Recall
apple	0.636
banana	0.444
orange	0.632

Os valores mostram que o detector recupera bem maçãs e laranjas, mas ainda perde várias bananas— reflexo do menor número de exemplos e da semelhança visual entre frutas amarelas. Para aplicações em que é crítico localizar todas as instâncias (ex.: contagem em linha de produção), aumentar o recall para *banana* deve ser prioridade, mesmo que isso traga leve queda de precisão

4. ANÁLISE QUALITATIVA DE ERROS

Além das métricas numéricas, foram inspecionadas as predições visuais geradas em val_batch*_pred.jpg e comparadas aos rótulos em val_batch*_labels.jpg.

4.1.1 Falsos Positivos

Onde ocorrem: superfícies estampadas ou regiões com cores semelhantes às frutas (ex.: toalhas, embalagens).

Impacto: eleva a contagem final, mas não prejudica severamente a mAP porque as caixas erradas costumam ter confiança baixa.

4.1.2 Falsos Negativos

Ao comparar as predições com os rótulos reais (val_batchX_labels.jpg), percebeu-se a presença de **objetos que não foram detectados**, caracterizando falsos negativos. Esses casos foram mais comuns em:

Mais frequentes quando:

- Frutas estão **parcialmente ocultas ou sobrepostas**;
- **Tamanhos muito reduzidos** na imagem (cantos distantes);
- **Iluminação irregular** produz sombras fortes.

Esses casos explicam o recall moderado e indicam a necessidade de mais variação de escala e iluminação no data-augmentation.

4.1.3 Erros de Classificação

Raros, mas ocorreram trocas entre *banana* ↔ *orange* quando a coloração laranja-amarelada é parecida. Aumentar exemplos e aplicar augmentations de matiz/saturação pode reduzir o problema.

4.1.4 Precisão na Delimitação (Bounding Boxes)

Algumas caixas ficam curtas ou largas demais em frutas alongadas ou cortadas pelo enquadramento. Isso afeta mais a mAP@0.5:0.95 do que a mAP@0.5. Treinar com imagens onde as frutas tocam bordas ajuda o modelo a ajustar melhor o box-regression.

5. MÉTRICAS RESUMO

Métrica global	Valor
mAP@0.5	0.6540
mAP@0.5:0.95	0.4455
Precisão média	0.6793
Recall médio	0.5863

No conjunto, o YOLOv8n treinado é adequado para protótipos e aplicações onde uma **mAP ~65 %** a IoU 0.5 é aceitável. Para uso industrial, recomenda-se uma iteração adicional visando recall mais alto e AP superior a 0.80 para todas as classes.

6. GRÁFICOS - RESULTADOS

- **Precision e Recall:**

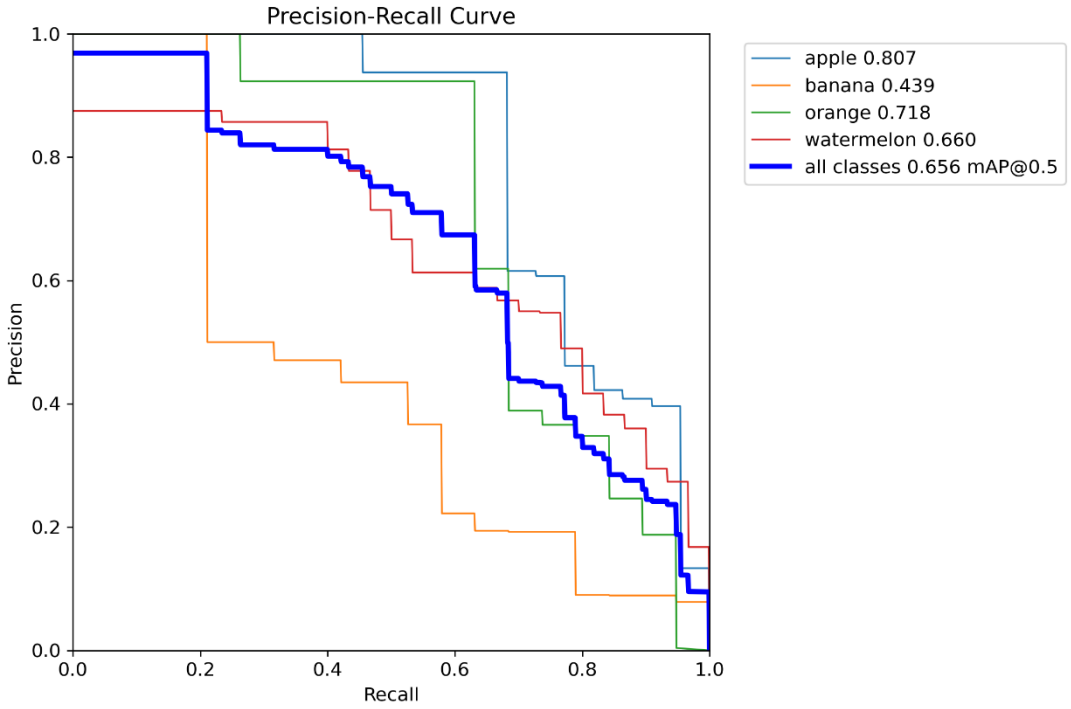

Figura 3 - PR CURVE

- **F1-Score:**

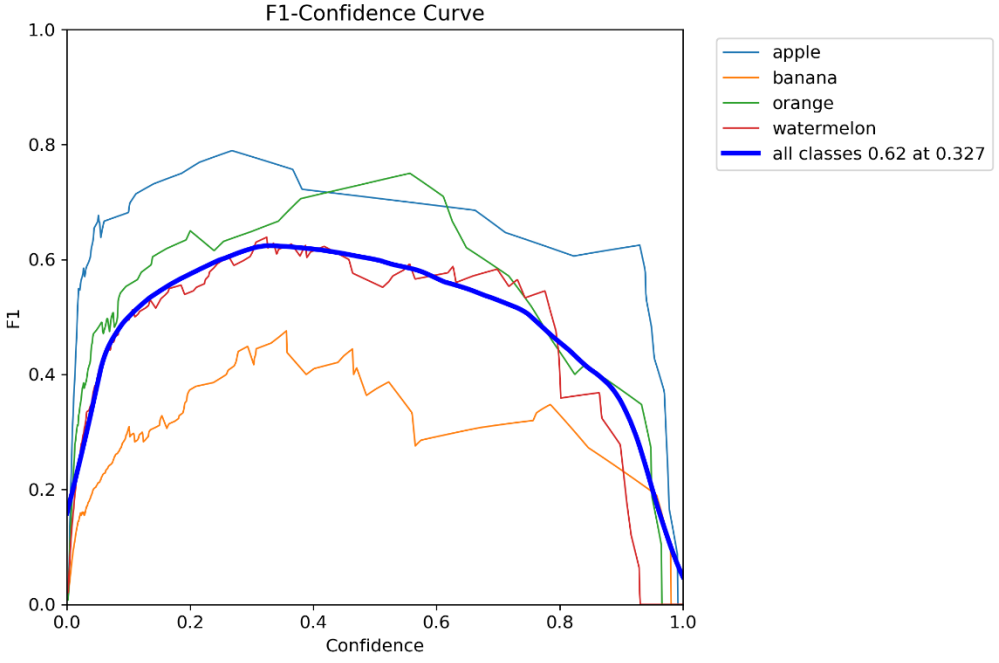

Figura 4 - F1 SCORE

- **Matriz de Confusão:**

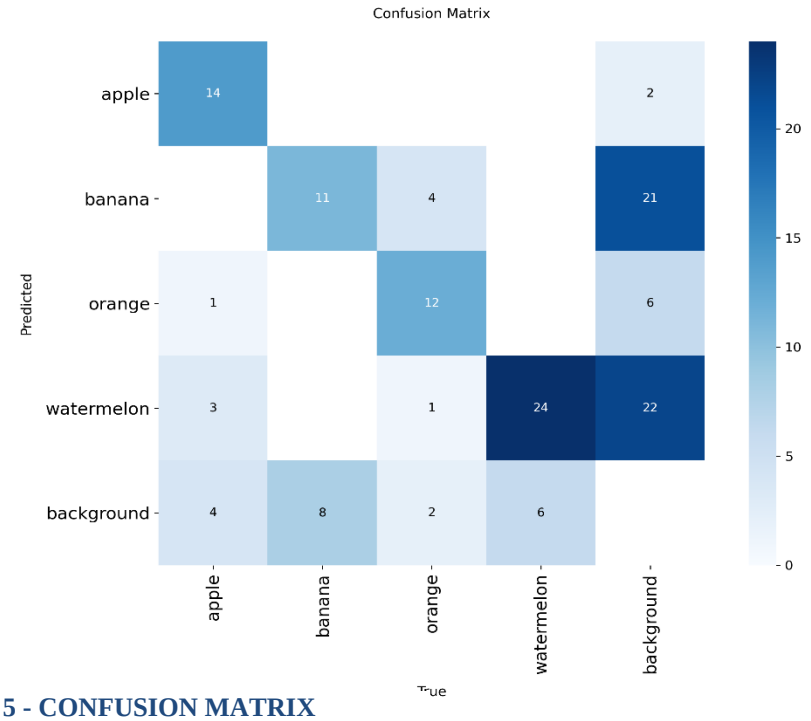

Figura 5 - CONFUSION MATRIX

- **Recall:**

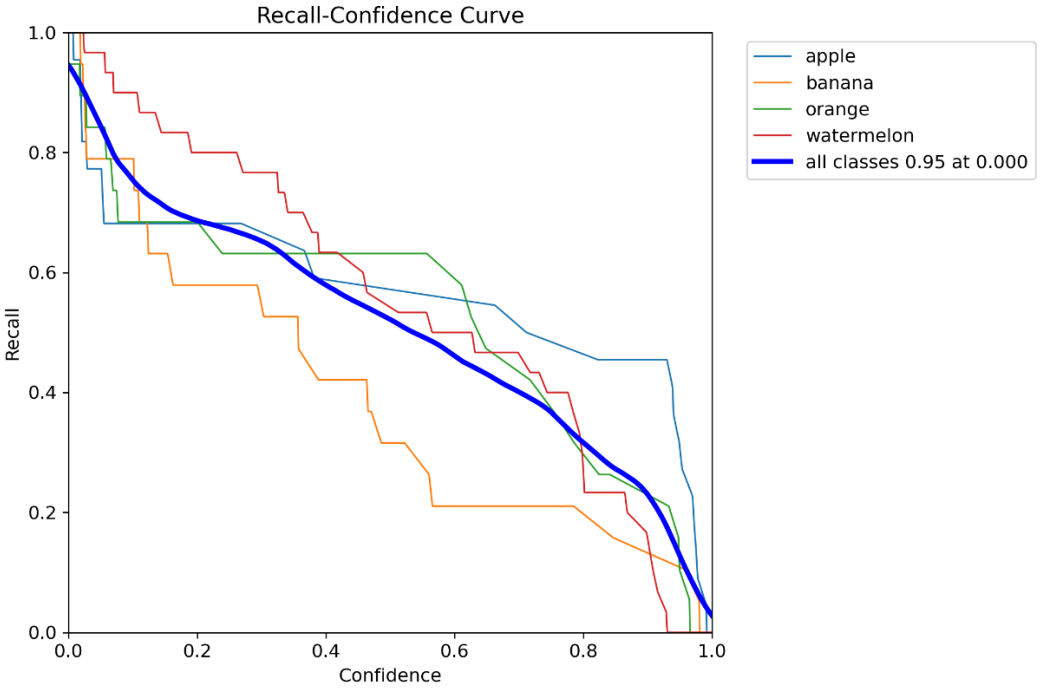

Figura 5 – R CURVE

- **Resultados gerais:**

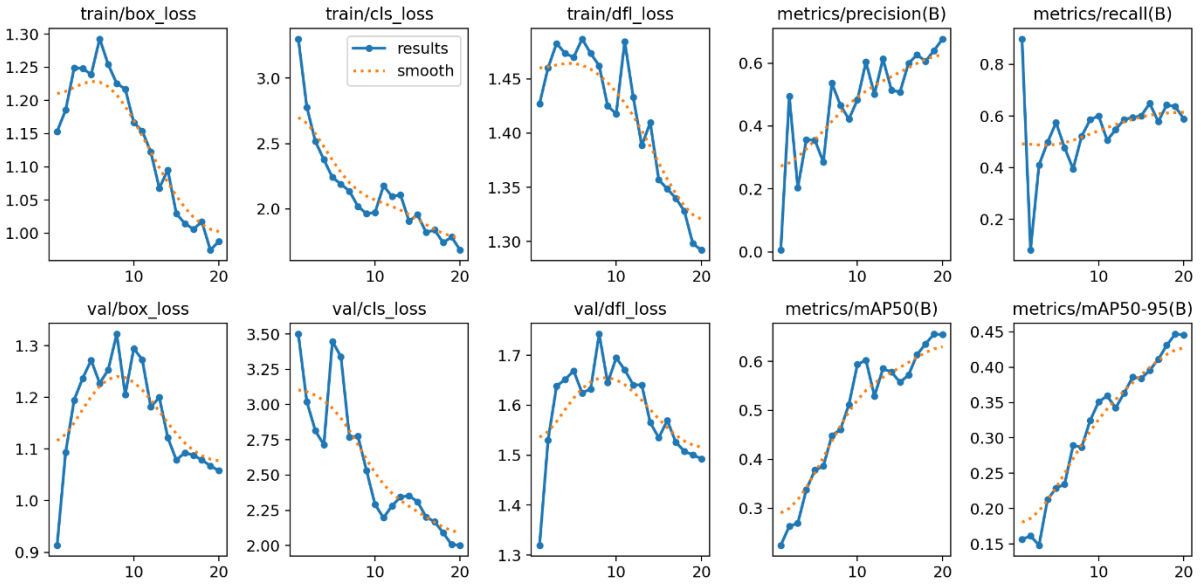

Figura 6 – RESULTS

Link dos vídeos:

https://drive.google.com/drive/folders/17NPiLodnsyGCmYuaX3jTUlHYftfHdsq_?usp=sharing