Reconhecimento de EPI utilizando Inteligência artificial e Visão Computacional

Luanne Zati de Lima

Objetivos

O projeto tem como objetivo desenvolver um sistema de reconhecimento de Equipamentos de Proteção Individual (EPI) para monitorar a segurança de funcionários em indústrias. O
sistema utiliza técnicas de inteligência artificial e visão computacional para analisar imagens
ou vídeos das câmeras de segurança e identificar se as pessoas estão utilizando os EPIs
necessários, como capacetes, óculos de proteção, luvas, coletes refletivos, entre outros. Caso o sistema detecte a ausência de qualquer EPI obrigatório, um alerta será enviado aos líderes ou responsáveis pela segurança da fábrica.

Ferramentas Tecnológicas

OpenCV

TensorFlow

Python

Open CV

TensorFlow

PostgreSQL

Colab

Implementações Referência

01 https://youtube.com/watch?v=4Xd0zJDDDZU

Desenvolvimento

01 Modelo: ExPoint GitHub

02 Reconhecimento:

- Capacete
- Óculos
- Luvas
- Máscara
- Colete

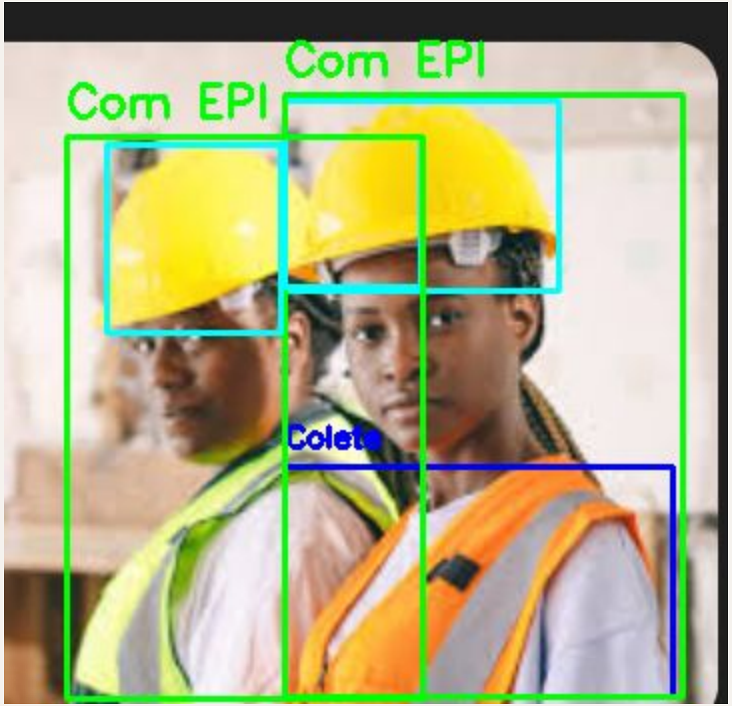
03 Yolov8

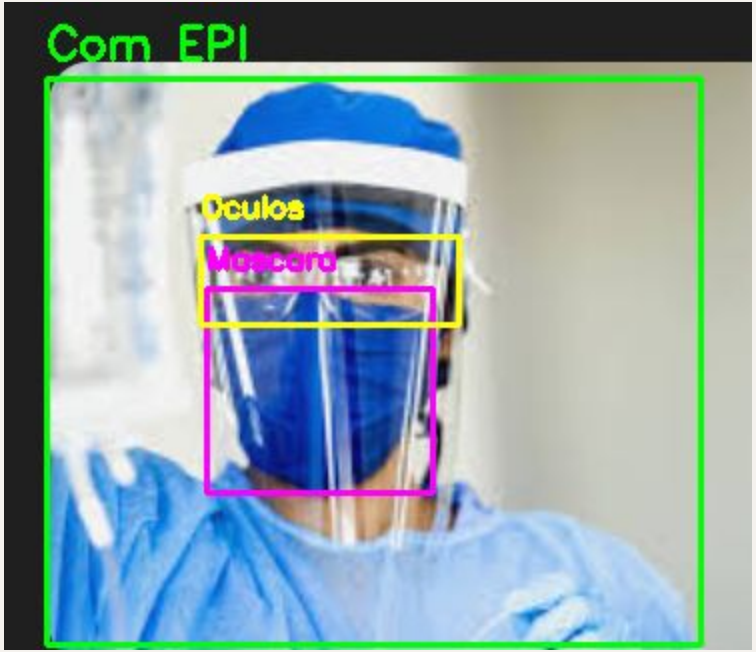
Desenvolvimento

O Código:

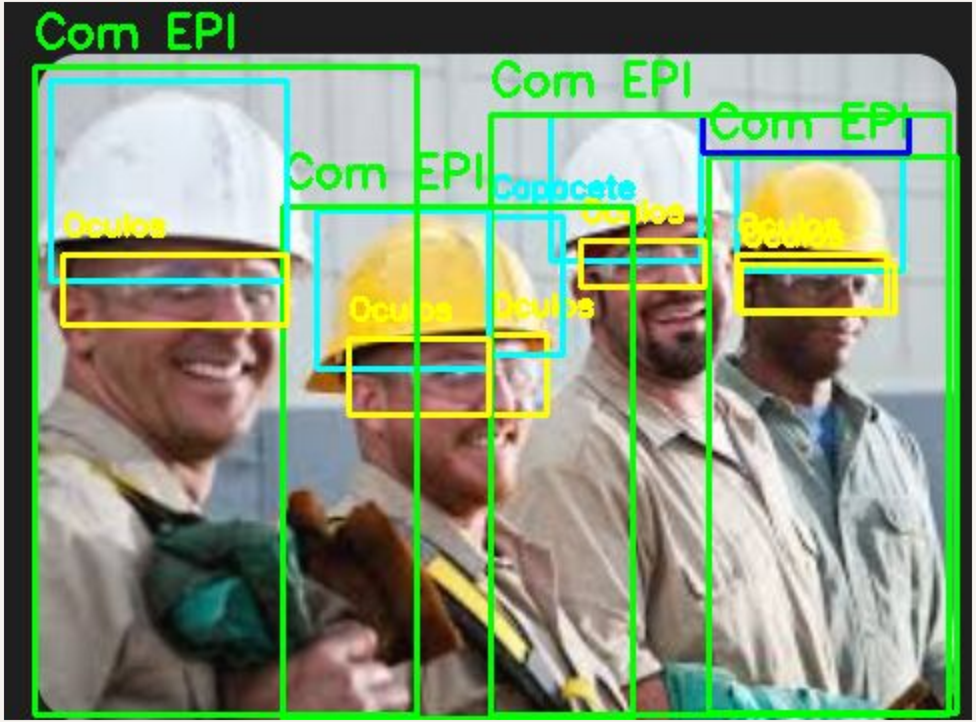
- Faz o reconhecimento se há pessoas na imagem
- Se tiver, faz a segmentação dessa pessoa para identificar a presença ou não de EPI

Exemplos





Exemplos


Com EPI
Com EPI
Com EPI
Com EPI
Capacete
Oculos
Oculos
Oculos
Oculos
Oculos


Sem EPI

Exemplos

Exemplos

Testando o modelo

```python
model = YOLO("/content/datsetpropriov2.pt")
results = model.val(data="/content/dataset_criado/data.yaml")
print(results)
```

```
Ultralytics 8.3.40 🚀 Python-3.10.12 torch-2.5.1+cu121 CPU (Intel Xeon 2.20GHz)
Model summary (fused): 168 layers, 11,127,519 parameters, 0 gradients, 28.4 GFLOPs
val: Scanning /content/dataset_criado/valid/labels... 68 images, 0 backgrounds, 0 corrupt: 100%|██████████| 68/68 [00:00

                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 5/5 [00:48<00
                   all         68        299       0.79      0.543      0.655      0.432
              capacete         64        101      0.917      0.752      0.865      0.574
                colete         50         77      0.796      0.532      0.665      0.483
                 luvas         35         55      0.498     0.0909      0.158     0.0831
               mascara         14         25      0.901      0.729      0.832      0.608
                oculos         31         41       0.84       0.61      0.754      0.411
Speed: 7.9ms preprocess, 674.4ms inference, 0.0ms loss, 1.1ms postprocess per image
Results saved to runs/detect/val5
ultralytics.utils.metrics.DetMetrics object with attributes:

ap_class_index: array([0, 1, 2, 3, 4])
box: ultralytics.utils.metrics.Metric object
confusion_matrix: <ultralytics.utils.metrics.ConfusionMatrix object at 0x7f450647a290>
curves: ['Precision-Recall(B)', 'F1-Confidence(B)', 'Precision-Confidence(B)', 'Recall-Confidence(B)']
curves_results: [[array([          0,    0.001001,    0.002002,    0.003003,    0.004004,    0.005005,    0.006006,    0
          0.024024,    0.025025,    0.026026,    0.027027,    0.028028,    0.029029,     0.03003,    0.031031,    0.0320
          0.048048,    0.049049,     0.05005,    0.051051,    0.052052,    0.053053,    0.054054,    0.055055,    0.0560
          0.072072,    0.073073,    0.074074,    0.075075,    0.076076,    0.077077,    0.078078,    0.079079,     0.080
          0.096096,    0.097097,    0.098098,    0.099099,      0.1001,      0.1011,      0.1021,      0.1031,      0.10
```

Testando o modelo

Conclusões sobre o teste do modelo

Precisão geral: 0.79 (muito bom, o modelo evita muitas predições erradas).
Recall geral: 0.543 (moderado; há objetos reais que o modelo não está detectando).
mAP50 geral: 0.655 (bom desempenho considerando o limiar de IoU de 0.5).
mAP50-95 geral: 0.432 (desempenho cai com limiares mais rigorosos, sugerindo que o modelo precisa melhorar em casos mais complexos).

Testando o modelo

Conclusões sobre o teste do modelo

Capacete:

- **Precisão**: 0.917 (excelente, o modelo comete poucos erros ao detectar capacetes).
- **Recall**: 0.752 (bom, está detectando a maioria dos capacetes reais).
- **mAP50**: 0.865 (muito bom, limiar de 0.5).
- **mAP50-95**: 0.574 (bom, mas ainda pode melhorar para limiares mais altos).

Colete:

- **Precisão**: 0.796 (bom, mas há margem para redução de falsos positivos).
- **Recall**: 0.532 (moderado, o modelo não está capturando muitos coletes reais).
- **mAP50**: 0.665 (razoável).
- **mAP50-95**: 0.483 (poderia melhorar para casos mais desafiadores).

Testando o modelo

Conclusões sobre o teste do modelo

Luvas:

- **Precisão**: 0.498 (fraco, o modelo faz muitas predições erradas sobre luvas).
- **Recall**: 0.0909 (muito baixo, o modelo quase não detecta luvas reais).
- **mAP50**: 0.158 (fraco).
- **mAP50-95**: 0.0831 (muito baixo, dificuldade em lidar com luvas).

Máscara:

- **Precisão**: 0.901 (excelente).
- **Recall**: 0.729 (bom, detecta muitas máscaras reais).
- **mAP50**: 0.832 (muito bom).
- **mAP50-95**: 0.608 (bom, mesmo em limiares mais rigorosos).

Testando o modelo

Conclusões sobre o teste do modelo

Óculos:

- **Precisão**: 0.84 (muito bom).
- **Recall**: 0.61 (bom, detecta uma boa parte dos óculos reais).
- **mAP50**: 0.754 (muito bom).
- **mAP50-95**: 0.411 (moderado, mas cai em limiares mais altos).

Testando o modelo

Conclusões sobre o teste do modelo

O modelo está se saindo bem com capacetes, máscaras e óculos. Estes itens têm boa precisão e recall.

O desempenho com coletes é moderado, mas ainda razoável.

O desempenho com luvas é o ponto mais fraco, tanto na precisão quanto no recall. Isso pode ser devido a:

- **Poucas amostras de luvas no dataset.**
- **Variabilidade nas imagens (diferentes tipos de luvas, iluminação, ou poses).**
- **Necessidade de mais refinamento no treinamento.**