

Documentação: Wrapper de Inferência YOLOv8 (ONNX + OpenCV)

Este script fornece uma interface simples e leve para realizar inferências com modelos YOLOv8 (Detecção e Classificação) utilizando apenas **OpenCV** e **NumPy**.

Ele elimina a necessidade de instalar bibliotecas pesadas como `torch`, `torchvision` ou `ultralytics` no ambiente de produção, sendo ideal para sistemas embarcados e ROS 2.

Pré-requisitos

Para utilizar este script, você precisa apenas das seguintes bibliotecas instaladas:

```
pip install opencv-python numpy
```

Nota: Certifique-se de que seus modelos `.pt` foram exportados para `.onnx` com `opset=12` para garantir compatibilidade com o OpenCV: `yolo export model=seu_modelo.pt format=onnx opset=12`

Classe 1: InferenceYOLO (Detecção de Objetos)

Utilizada para modelos treinados para detectar objetos e retornar caixas delimitadoras (Bounding Boxes).

Inicialização

```
yolo = InferenceYOLO(MODEL_PATH, IMAGE_PATH, INPUT_SIZE, SCORE_THRESHOLD, NMS_THRESHOLD)
```

Parâmetro	Tipo	Descrição
MODEL_PATH	str	Caminho para o arquivo <code>.onnx</code> de detecção.
IMAGE_PATH	str	Caminho para a imagem de entrada.
INPUT_SIZE	tuple	Tamanho da entrada da rede (Largura, Altura). Ex: <code>(640, 640)</code> .
SCORE_THRESHOLD	float	Confiança mínima (0.0 a 1.0) para considerar uma detecção válida.
NMS_THRESHOLD	float	Limiar para a supressão de não-máximos (remove caixas duplicadas).

Métodos Disponíveis

```
bestboxes()
```

Retorna uma lista com as coordenadas das caixas detectadas após o filtro NMS.

- **Retorno:** Lista de listas `[[x, y, w, h], ...]` (Canto Superior Esq, Largura, Altura).

```
bestscores()
```

Retorna a confiança (probabilidade) de cada caixa detectada.

- **Retorno:** Lista de floats `[0.95, 0.88, ...]`.

```
bestclassids()
```

Retorna o ID numérico da classe de cada caixa detectada.

- **Retorno:** Lista de inteiros `[0, 1, 0, ...]`.

```
imshow()
```

Abre uma janela do OpenCV exibindo a imagem original com as caixas e rótulos desenhados.

- **Uso:** Pressione qualquer tecla para fechar a janela.

Classe 2: `InferenceClass` **(Classificação de Imagem)**

Utilizada para modelos treinados para classificar a imagem inteira em uma categoria (ex: "Legível" vs "Ilegível"), sem retornar coordenadas.

Inicialização

```
classificador = InferenceClass(MODEL_PATH, IMAGE_PATH, INPUT_SIZE, CLASS_NAMES)
```

Parâmetro	Tipo	Descrição
MODEL_PATH	str	Caminho para o arquivo <code>.onnx</code> de classificação.
IMAGE_PATH	str	Caminho para a imagem de entrada.
INPUT_SIZE	tuple	Tamanho da entrada. Geralmente <code>(224, 224)</code> ou <code>(640, 640)</code> .
CLASS_NAMES	list	Lista de strings com os nomes das classes na ordem correta.

Métodos Disponíveis

```
classresult()
```

Retorna o resultado da predição com maior probabilidade.

- **Retorno:** Lista contendo `[ID_Classe, Confiança, Nome_Classe]`.
- **Exemplo:** `[0, 0.98, 'ilegivel']`.

```
imshow()
```

Abre uma janela do OpenCV exibindo a imagem original com o texto da classe predita escrito nela.

Exemplo de Uso Completo

Abaixo, um exemplo de como instanciar e usar ambas as classes em um único script.

```

from seu_script_de_inferencia import InferenceYOLO, InferenceClass

# --- EXEMPLO 1: Classificação ---
# Definindo as classes (deve ser a mesma ordem do treino)
classes_manometro = ['ilegivel', 'legivel']

# Instanciando
classificador = InferenceClass(
    MODEL_PATH="runs/classify/yolo_manometer/weights/best.onnx",
    IMAGE_PATH="foto_manometro.jpg",
    INPUT_SIZE=(640, 640),
    CLASS_NAMES=classes_manometro
)

# Obtendo resultados
resultado = classificador.classresult()
print(f"Resultado Classificação: ID={resultado[0]}, Score={resultado[1]}, Nome={resulta

# Visualizando
classificador.imshow()

# --- EXEMPLO 2: Detecção ---
# Instanciando
detector = InferenceYOLO(
    MODEL_PATH="runs/detect/yolo_plataforma/weights/best.onnx",
    IMAGE_PATH="foto_drone.png",
    INPUT_SIZE=(640, 640),
    SCORE_THRESHOLD=0.5,
    NMS_THRESHOLD=0.45
)

# Obtendo resultados
caixas = detector.bestboxes()
scores = detector.bestscores()

print(f"Foram detectados {len(caixas)} objetos.")
for i in range(len(caixas)):
    print(f"Objeto {i}: Box={caixas[i]}, Confiança={scores[i]}")

# Visualizando
detector.imshow()

```

⚙️ Detalhes Técnicos (Letterbox)

O script inclui uma função interna chamada `letterbox`. Ela é responsável por redimensionar a imagem de entrada mantendo a proporção original (aspect ratio).

- Se a imagem for retangular e o modelo esperar quadrada, o script adiciona **bordas cinzas** (padding) automaticamente.
- As coordenadas retornadas pela detecção já são **automaticamente corrigidas** para corresponderem aos pixels da imagem original, removendo o efeito do padding e da escala.