Model prediction

1. Best performance mode

1.1. Enable MAX mode

Enabling MAX Power Mode on Jetson will ensure that all CPU and GPU cores are turned on:

```
sudo nvpmodel -m 0
```

1.2. Enable Jetson Clocks

Enabling Jetson Clocks will ensure that all CPU and GPU cores run at maximum frequency:

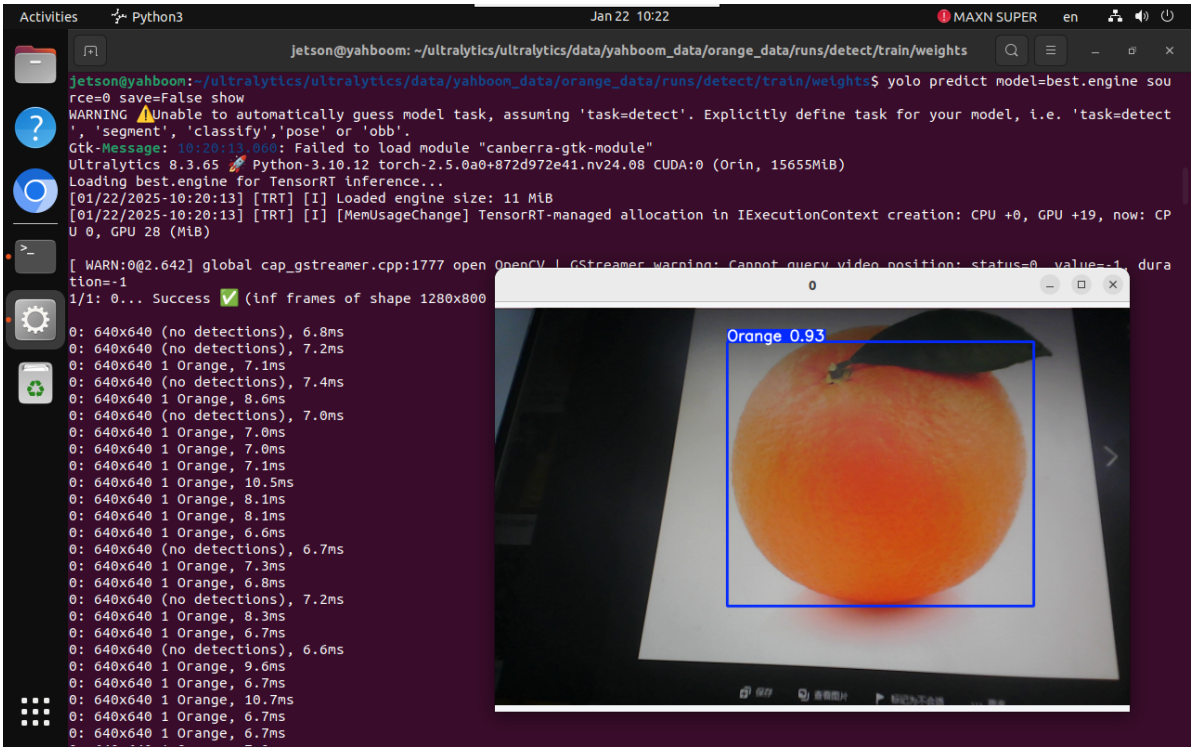
```
sudo jetson_clocks
```

2. Model prediction

2.1. CLI usage

CLI currently only supports calling USB cameras. CSI camera users can directly modify the previous python code to call onnx and engine models!

```
yolo predict model=best.engine source=0 save=False show # If there are multiple cameras, follow the number after the source
```



2.2, Python usage

Use Python to call USB camera and CSI camera to identify oranges.

2.2.1, USB camera

Use best.engine to predict camera images:

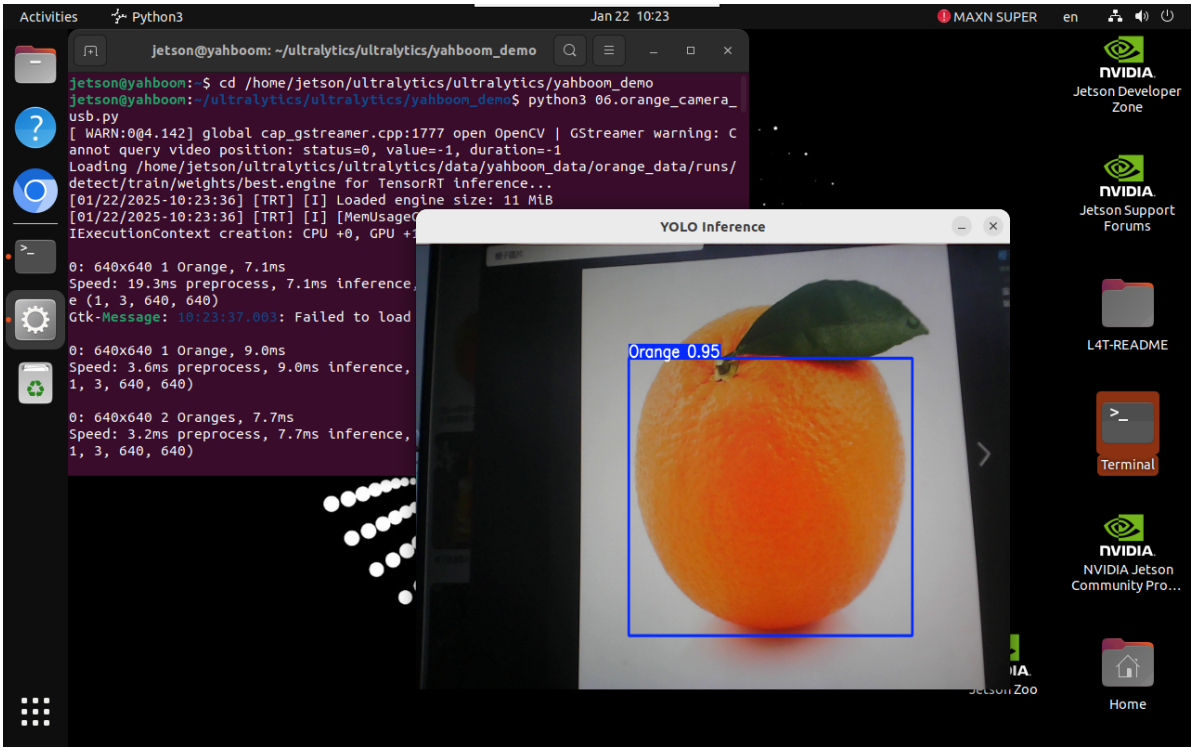
```
cd /home/jetson/ultralytics/ultralytics/yahboom_demo
```

Run the code: Click the preview screen, press the q key to terminate the program!

```
python3 06.orange_camera_usb.py
```

Effect preview

Video location of yolo recognition output: /home/jetson/ultralytics/ultralytics/output/



Sample code:

```
import cv2
from ultralytics import YOLO

# Load the YOLO model
# model =
YOLO("/home/jetson/ultralytics/ultralytics/data/yahboom_data/orange_data/runs/detect/train/weights/best.pt")
# model =
YOLO("/home/jetson/ultralytics/ultralytics/data/yahboom_data/orange_data/runs/detect/train/weights/best.onnx")
model =
YOLO("/home/jetson/ultralytics/ultralytics/data/yahboom_data/orange_data/runs/detect/train/weights/best.engine")

# Open the cammera
cap = cv2.VideoCapture(0)

# Get the video frame size and frame rate
frame_width = int(cap.get(cv2.CAP_PROP_FRAME_WIDTH))
frame_height = int(cap.get(cv2.CAP_PROP_FRAME_HEIGHT))
fps = int(cap.get(cv2.CAP_PROP_FPS))

# Define the codec and create a VideoWriter object to output the processed video
output_path =
"/home/jetson/ultralytics/ultralytics/output/06.orange_camera_usb.mp4"
fourcc = cv2.VideoWriter_fourcc(*'mp4v')  # You can use 'XVID' or 'mp4v'
depending on your platform
out = cv2.VideoWriter(output_path, fourcc, fps, (frame_width, frame_height))

# Loop through the video frames
while cap.isOpened():
    # Read a frame from the video
    success, frame = cap.read()
```

```
    if success:
        # Run YOLO inference on the frame
        results = model(frame)

        # Visualize the results on the frame
        annotated_frame = results[0].plot()

        # Write the annotated frame to the output video file
        out.write(annotated_frame)


        # Display the annotated frame
        cv2.imshow("YOLO Inference", cv2.resize(annotated_frame, (640, 480)))

        # Break the loop if 'q' is pressed
        if cv2.waitKey(1) & 0xFF == ord("q"):
            break
    else:
        # Break the loop if the end of the video is reached
        break
# Release the video capture and writer objects, and close the display window
cap.release()
out.release()
cv2.destroyAllWindows()
```

2.2.1, CSI camera

Use best.engine to predict the camera image:

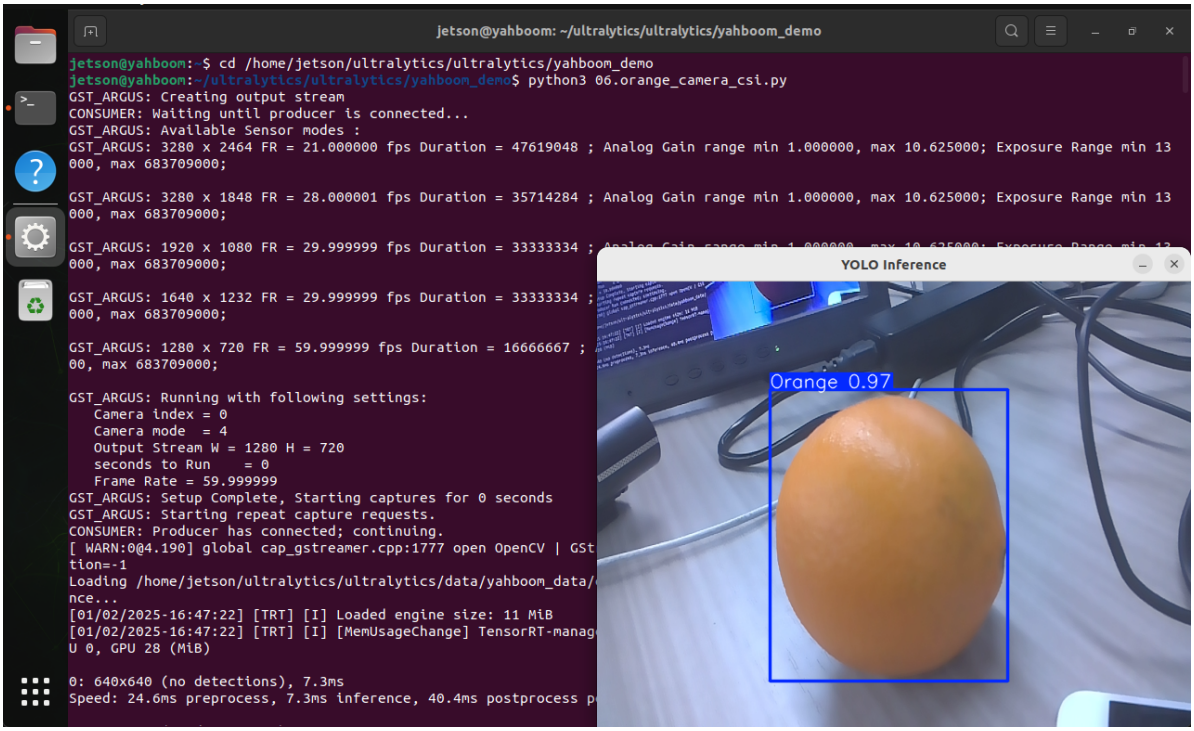
```
cd /home/jetson/ultralytics/ultralytics/yahboom_demo
```

Run the code: Click the preview image, press the q key to terminate the program!

```
python3 06.orange_camera_csi.py
```

Effect preview

Yolo recognizes the output video location: /home/jetson/ultralytics/ultralytics/output/



Sample code:

```
import cv2
from ultralytics import YOLO
from jetcam.csi_camera import CSICamera

# Load the YOLO model
# model =
YOLO("/home/jetson/ultralytics/ultralytics/data/yahboom_data/orange_data/runs/detect/train/weights/best.pt")
# model =
YOLO("/home/jetson/ultralytics/ultralytics/data/yahboom_data/orange_data/runs/detect/train/weights/best.onnx")
model =
YOLO("/home/jetson/ultralytics/ultralytics/data/yahboom_data/orange_data/runs/detect/train/weights/best.engine")

# Open the camera (CSI Camera)
cap = CSICamera(width=640, height=480)

# Get the video frame size and frame rate
frame_width = 640
frame_height = 480
fps = 30

# Define the codec and create a VideoWriter object to output the processed video
output_path =
"/home/jetson/ultralytics/ultralytics/output/06.orange_camera_csi.mp4"
fourcc = cv2.VideoWriter_fourcc(*'mp4v')  # You can use 'XVID' or 'mp4v'
depending on your platform
out = cv2.VideoWriter(output_path, fourcc, fps, (frame_width, frame_height))

# Loop through the video frames
while True:
    # Read a frame from the camera
    frame = cap.read()
```

```
    if frame is not None:
        # Run YOLO inference on the frame
        results = model(frame)

        # Visualize the results on the frame
        annotated_frame = results[0].plot()

        # Write the annotated frame to the output video file
        out.write(annotated_frame)

        # Display the annotated frame
        cv2.imshow("YOLO Inference", cv2.resize(annotated_frame, (640, 480)))

        # Break the loop if 'q' is pressed
        if cv2.waitKey(1) & 0xFF == ord("q"):
            break
    else:
        # Break the loop if no frame is received (camera error or end of stream)
        print("No frame received, breaking the loop.")
        break

# Release the video capture and writer objects, and close the display window
cap.release()
out.release()
cv2.destroyAllWindows()
```