

Model conversion

Model conversion

1. Jetson Orin YOLO11 (benchmark)
 2. Enable optimal performance of the motherboard
 - 2.1. Enable MAX power mode
 - 2.2. Enable Jetson clocks
 3. Model conversion
 - 3.1. CLI: pt → onnx → engine
 - 3.2. Python: pt → onnx → engine
 4. Model prediction
 - CLI usage
- Frequently Asked Questions
- ERROR: onnxslim
- References

1. Jetson Orin YOLO11 (benchmark)

YOLO11 benchmark data comes from the Ultralytics team, which tests models in multiple formats (data is for reference only)

2. Enable optimal performance of the motherboard

2.1. Enable MAX power mode

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

```
sudo nvpmodel -m 2
```

2.2. Enable Jetson clocks

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

```
sudo jetson_clocks
```

3. Model conversion

According to the test parameters of different format models provided by the Ultralytics team, we can find that the inference performance is best when using TensorRT!

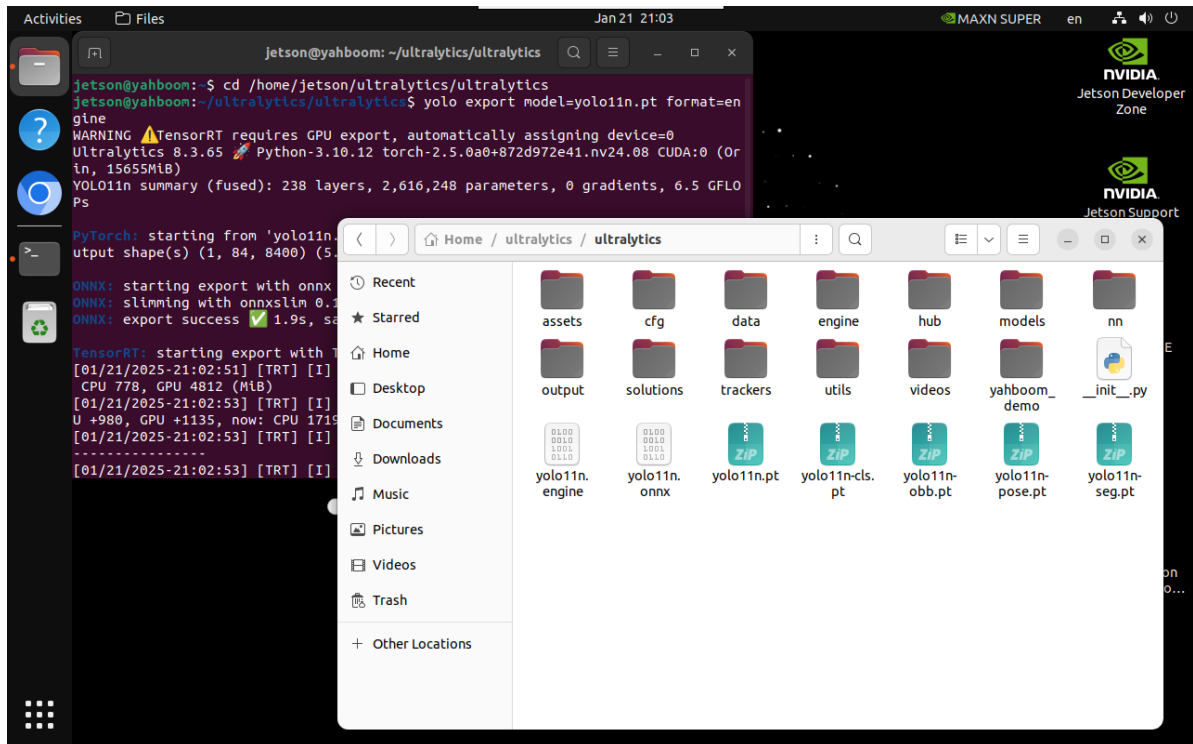
When using the export mode of YOLO11 for the first time, some dependencies will be automatically installed. Just wait for it to be completed automatically!

3.1、CLI: pt → onnx → engine

Convert the PyTorch model to TensorRT: The conversion process will automatically generate an ONNX model

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

```
yolo export model=yolo11n.pt format=engine
# yolo export model=yolo11n-seg.pt format=engine
# yolo export model=yolo11n-pose.pt format=engine
# yolo export model=yolo11n-cls.pt format=engine
# yolo export model=yolo11n-obb.pt format=engine
```



3.2、Python: pt → onnx → engine

Convert the PyTorch model to TensorRT: The conversion process will automatically generate an ONNX model

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

```
python3 model_pt_onnx_engine.py
```

```
from ultralytics import YOLO

# Load a YOLO11n PyTorch model
# model = YOLO("/home/jetson/ultralytics/ultralytics/yolo11n.pt")
model = YOLO("/home/jetson/ultralytics/ultralytics/yolo11n-seg.pt")
# model = YOLO("/home/jetson/ultralytics/ultralytics/yolo11n-pose.pt")
# model = YOLO("/home/jetson/ultralytics/ultralytics/yolo11n-cls.pt")
# model = YOLO("/home/jetson/ultralytics/ultralytics/yolo11n-obb.pt")
# Export the model to TensorRT
model.export(format="engine")
```

The image displays a Linux desktop environment. On the left, a vertical dock contains icons for Activities, Applications, and a file manager. The top panel shows the date and time as 'Jan 21: 08' and the system status as 'MAXN SUPER en'. The terminal window, titled 'Jetson@yahboom: ~/ultralytics/ultralytics/yahboom_demo', shows the following commands and output:

```
Jetson@yahboom:~$ cd /home/jetson/ultralytics/ultralytics/yahboom_demo
Jetson@yahboom:~/ultralytics/ultralytics/yahboom_demo$ python3 model_pt_onnx_eng
ine.py
WARNING ⚠️TensorRT requires GPU export, automatically assigning device=0
Ultralytics 8.3.65 Python-3.10.12 torch-2.5.0a0+872d972e41.nv24.08 CUDA:0 (0r
in, 15655MiB)
YOLO11n-seg summary (fused): 265 layers, 2,868,664 parameters, 0 gradients, 10.4
GFLOPS

PyTorch: starting from '/home/jetson/ultralytics/ultralytics/yahboom_demo/pt
h input shape (1, 3, 640, 640) BCHW and output shape (160, 160)) (5.9 MB)

ONNX: starting export with onnx 1.17.0 opset 19...
ONNX: slimming with onnxslim 0.1.47...
ONNX: export success ✅ 2.1s, saved as '/home/jetson/ultralytics/yahboom_demo/o
11n-seg.onnx' (11.2 MB)

TensorRT: starting export with TensorRT 10.7.0...
[01/21/2025-21:05:42] [TRT] [I] [MemUsageChange] Int
CPU 871, GPU 5161 (MiB)
[01/21/2025-21:05:44] [TRT] [I] [MemUsageChange] Int
U +977, GPU +1158, now: CPU 1804, GPU 6330 (MiB)
[01/21/2025-21:05:44] [TRT] [I] -----
```

The file manager window, titled 'Home / ult...ics / ultralytics', shows a directory view of the 'ultralytics' folder. The files and folders are:

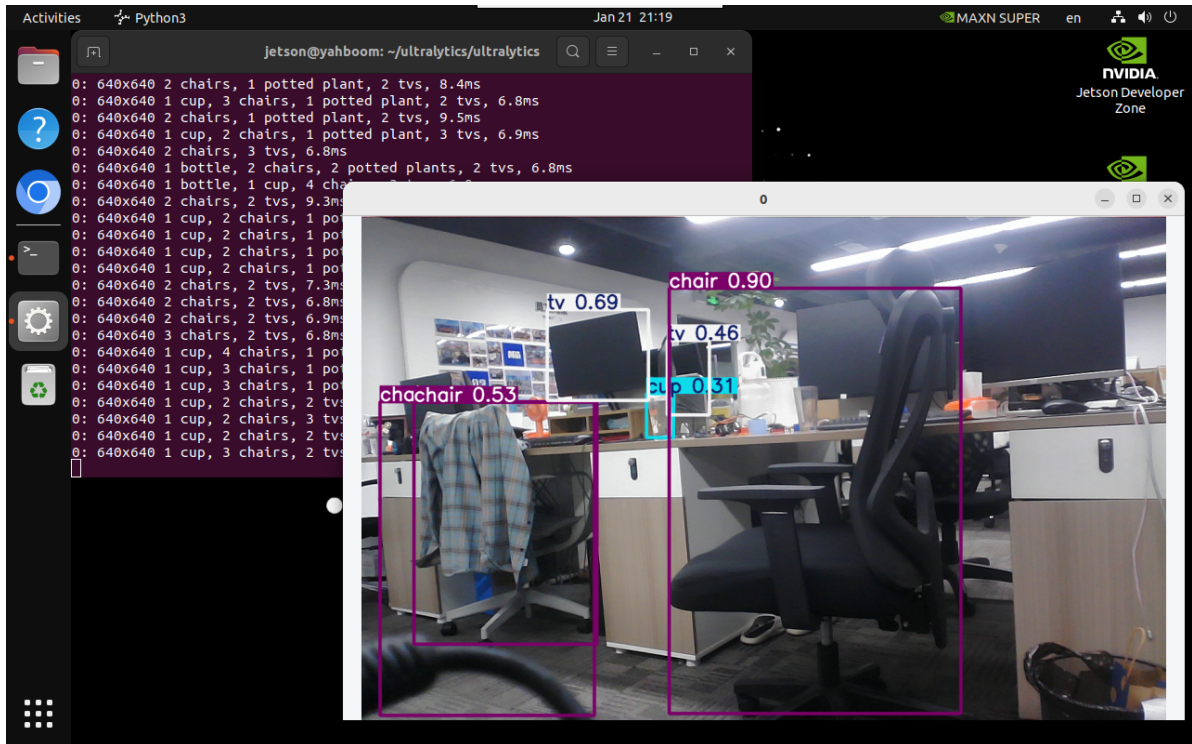
- assets
- cfg
- data
- engine
- hub
- models
- nn
- output
- solutions
- trackers
- utils
- videos
- yahboom_demo
- _init_.py
- yolo11n.onnx
- yolo11n.pt
- yolo11n-seg.engine
- yolo11n-seg.onnx
- yolo11n-seg.pt
- yolo11n-cls.pt
- yolo11n-obb.pt
- yolo11n-pose.pt

CLI usage

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

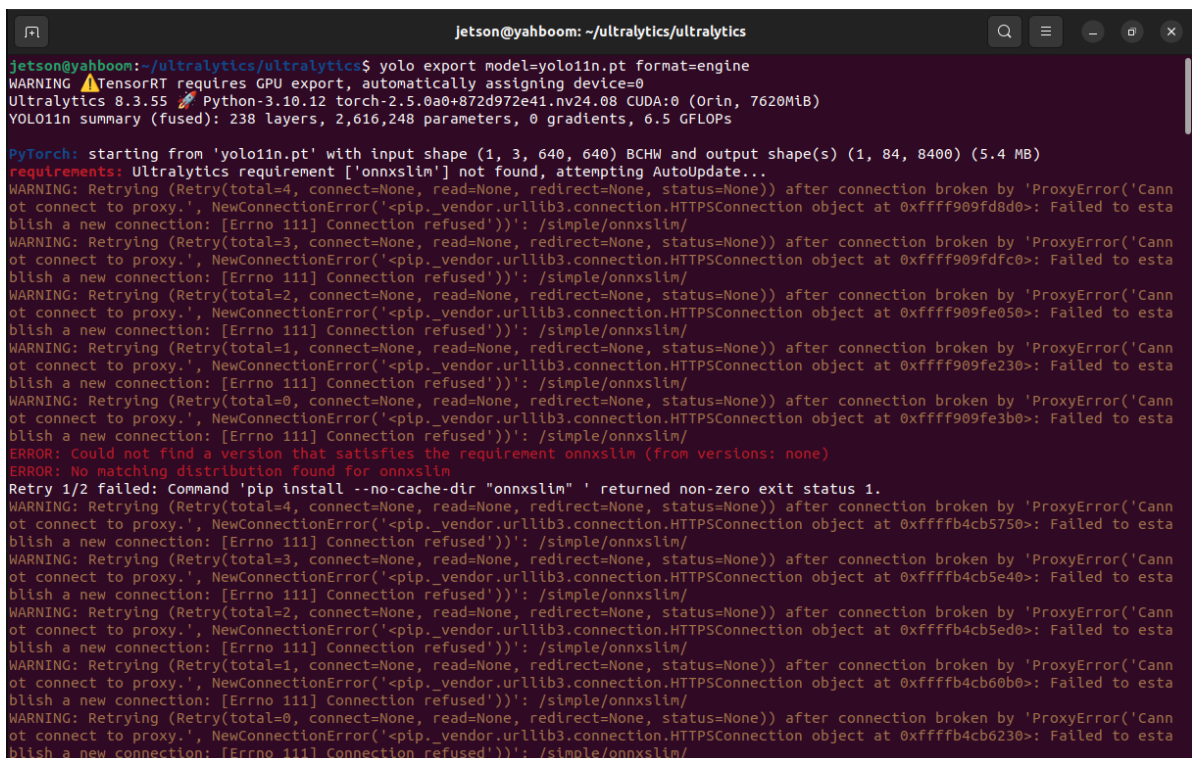
[illegible]

```
yolo predict model=yolo11n.engine source=0 save=False show
```



Frequently Asked Questions

ERROR: onnxslim



Solution: Enter the onnxslim installation command in the terminal

```
sudo pip3 install onnxslim
```

```
jetson@yahboom: ~/ultralytics/ultralytics
Collecting onnxslim
  Downloading onnxslim-0.1.45-py3-none-any.whl.metadata (4.2 kB)
Requirement already satisfied: onnx in /usr/local/lib/python3.10/dist-packages (from onnxslim) (1.17.0)
Requirement already satisfied: sympy in /usr/local/lib/python3.10/dist-packages (from onnxslim) (1.13.1)
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-packages (from onnxslim) (23.2)
Requirement already satisfied: numpy>=1.20 in /usr/local/lib/python3.10/dist-packages (from onnx->onnxslim) (1.23.5)
Requirement already satisfied: protobuf>=3.20.2 in /usr/local/lib/python3.10/dist-packages (from onnx->onnxslim) (4.25.5)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in /usr/local/lib/python3.10/dist-packages (from sympy->onnxslim) (1.3.0)
Downloading onnxslim-0.1.45-py3-none-any.whl (142 kB)
Installing collected packages: onnxslim
Successfully installed onnxslim-0.1.45
WARNING: Running pip as the 'root' user can result in broken permissions and conflicting behaviour with the system package manager, possibly rendering your system unusable. It is recommended to use a virtual environment instead: https://pip.pypa.io/warnings/venv. Use the --root-user-action option if you know what you are doing and want to suppress this warning.
jetson@yahboom:~/ultralytics/ultralytics$ yolo export model=yolo11n-cls.pt format=engine
WARNING ⚠️ TensorRT requires GPU export, automatically assigning device=0
Ultralytics 8.3.55 🚀 Python-3.10.12 torch-2.5.0a0+872d972e41.nv24.08 CUDA:0 (Orin, 7620MiB)
YOLO11n-cls summary (fused): 112 layers, 2,807,024 parameters, 0 gradients, 4.2 GFLOPs

PyTorch: starting from 'yolo11n-cls.pt' with input shape (1, 3, 224, 224) BCHW and output shape(s) (1, 1000) (5.5 MB)

ONNX: starting export with onnx 1.17.0 opset 19...
ONNX: slimming with onnxslim 0.1.45...
ONNX: export success ✅ 0.9s, saved as 'yolo11n-cls.onnx' (10.8 MB)

TensorRT: starting export with TensorRT 10.3.0...
[12/31/2024-12:05:11] [TRT] [I] [MemUsageChange] Init CUDA: CPU +2, GPU +0, now: CPU 654, GPU 3655 (MiB)
[12/31/2024-12:05:13] [TRT] [I] [MemUsageChange] Init builder kernel library: CPU +927, GPU +683, now: CPU 1624, GPU 4338 (MiB)
[12/31/2024-12:05:13] [TRT] [I] -----
[12/31/2024-12:05:13] [TRT] [I] Input filename:      yolo11n-cls.onnx
[12/31/2024-12:05:13] [TRT] [I] ONNX IR version:  0.0.9
[12/31/2024-12:05:13] [TRT] [I] Opset version:    19
[12/31/2024-12:05:13] [TRT] [I] Producer name:    pytorch
[12/31/2024-12:05:13] [TRT] [I] Producer version: 2.5.0
[12/31/2024-12:05:13] [TRT] [I] Domain:
[12/31/2024-12:05:13] [TRT] [I] Model version:    0
[12/31/2024-12:05:13] [TRT] [I] Doc string:
[12/31/2024-12:05:13] [TRT] [I] -----
TensorRT: input "images" with shape(1, 3, 224, 224) DataType.FLOAT
```

References

<https://docs.ultralytics.com/guides/nvidia-jetson/>

<https://docs.ultralytics.com/integrations/tensorrt/>