

Image based Malware Classification

Comparative Analysis of YOLOv8 and YOLOv11 for Malware Image Classification

Sylan Padmakumar Pranjal Khare Anand Pandey Vikash
Kumar Ayush Monga

Capstone Project 2025 - 2026

Table of Contents

1 introduction

2 Workflow

3 Results

Introduction

In our project we are using the YOLO ML models to classify malware. Our dataset consists of malware from 9 different families.

We have chosen to do a comparative study of the different YOLO models to identify their efficiency in classifying malware.

Table of Contents

1 introduction

2 Workflow

3 Results

Dataset Preparation

Dataset Acquisition:

- Dataset acquired from Microsoft malware classification challenge.
- Consists of bytes files acquired from malware belonging to 9 different families.
- The malware families consist of: Ramnit, Lollipop, Kelihos ver3, Vundo, Simda, Tracur, Kelihos ver1, Obfuscator.ACY, Gatak

Dataset Conversion

The bytes files had to be converted into images for classification by the YOLO image models, the following methods were used:

- The files are arranged into 9 different class directories.
- We have used the 3-gram rgb method proposed by arXiv:2207.00421.

Dataset Preparation

dataset was split in the following way to comply with YOLO training specifications:

Dataset split:

- 80% → train
- 10% → validation
- 10% → test

Software Workflow

Training Workflow

- Used GPU compute instance hired from a cloud compute provider.
- Installed all the requirements(Pytorch, Ultralitics, Matplotlib, numpy, pandas) on a python virtual environment.
- Verified CUDA and GPU availability using Torch.
- Downloaded pre-trained classification weights.

Hardware

Used a pytorch template VM with the following hardware specifications

- Used Nvidia RTX A5000 GPU (24GB GDDR6 with error-correction code (ECC)).
- 32GB Ram and 7 CPUs.

Note

GPU and compute was acquired from www.jarvislabs.ai.

Table of Contents

1 introduction

2 Workflow

3 Results

Results

Model complexity and final classification metrics:

Model	Parameters	GFLOPs	Final Top-1 Acc
YOLOv8m-cls (final)	15,774,185	41.6	98.5%
YOLO11m-cls (final)	10,353,161	39.3	98.9%
YOLOv8n-cls (smoke)*	1,446,409	3.3	93.7%
YOLO11n-cls (smoke)*	1,537,553	3.2	94.9%

* YOLOv11n-cls and YOLOv8n-cls stats are from a small smoke run. Exact small-model numbers vary by configuration.

Results

The conclusions are as follows:

- YOLOv8 performed well as expected for the classification tasks proving it as a strong baseline for future performance comparison.
- YOLOv11 had a minor accuracy increase over YOLOv8 but used drastically less parameters for training.

Reasons for YOLOv7 failing:

- YOLOv7 Classification scripts incompatible with current PyTorch version.
- Missing model YAMLs and checkpoint formats caused errors.
- Was unable to obtain YOLOv7 classification checkpoints for training.
- Not trainable without patching the code.

Note

YOLOv7 cannot be trained as support for it ended in 2023.

YOLOv8 Smoke Test

Figure: YOLOv8 Smoke Test to check if all systems are ready

YOLOv8 Smoke test Image Batches

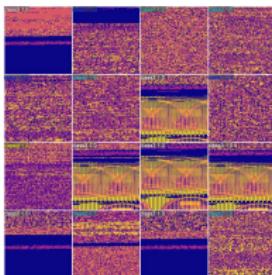
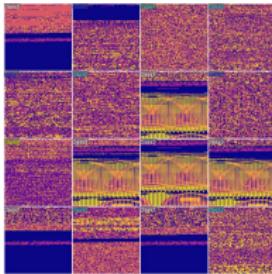


Figure: YOLOv8 validation batches.

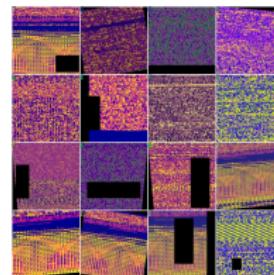
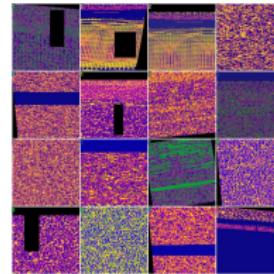


Figure: YOLOv8 training batches.

Confusion Matrix for YOLOv8 Smoke test

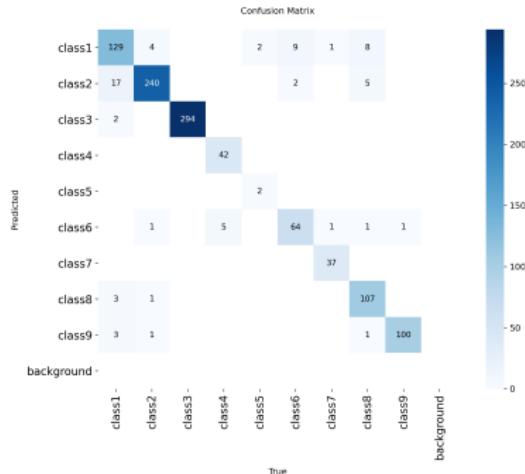


Figure: YOLOv8 confusion matrix normalized.

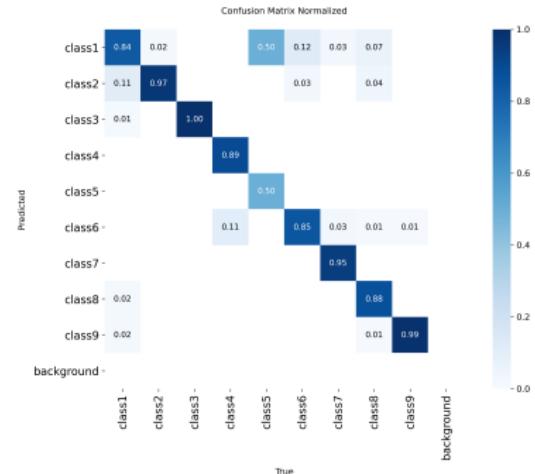


Figure: YOLOv8 confusion matrix normalized.

YOLOv8 Final Result

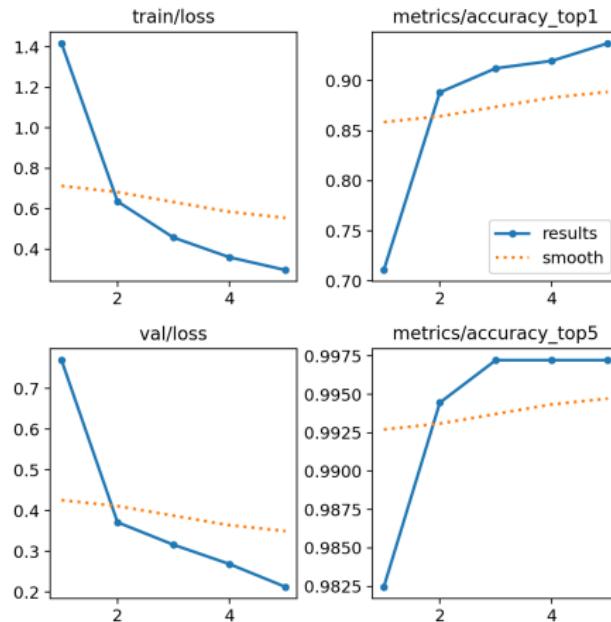


Figure: YOLOv8 smoke test result

YOLOv8 Final run

The screenshot shows a Jupyter Notebook environment with several open tabs and cells. The main area displays the output of a Python script named `YOLOv8.py`. The output shows training metrics for 100 epochs, with each epoch consisting of multiple batches. The metrics include GPU usage, loss values, instance counts, and top5 accuracy. The final accuracy reported is 98.5%.

```
classes top1_acc top5_acc 100% 17/17 37.1it/s 0.5s
all 0.999
Epoch GPU mem loss Instances Size
95/100 1.63G 0.0167 18 224: 100% 272/272 16.8it/s 36.2s
classes top1_acc top5_acc 100% 17/17 37.0it/s 0.5s
all 0.985 0.999
Epoch GPU mem loss Instances Size
96/100 1.63G 0.0167 18 224: 100% 272/272 18.2it/s 34.9s
classes top1_acc top5_acc 100% 17/17 36.9it/s 0.5s
all 0.985 0.999
Epoch GPU mem loss Instances Size
97/100 1.63G 0.0158 18 224: 100% 272/272 19.3it/s 34.3s
classes top1_acc top5_acc 100% 17/17 37.0it/s 0.5s
all 0.983 0.999
Epoch GPU mem loss Instances Size
98/100 1.63G 0.01516 18 224: 100% 272/272 18.5it/s 34.7s
classes top1_acc top5_acc 100% 17/17 46.0it/s 0.4s
all 0.983 0.999
Epoch GPU mem loss Instances Size
99/100 1.63G 0.01605 18 224: 100% 272/272 17.6it/s 35.5s
classes top1_acc top5_acc 100% 17/17 49.4it/s 0.4s
all 0.983 0.999
Epoch GPU mem loss Instances Size
100/100 1.63G 0.01527 18 224: 100% 272/272 18.7it/s 34.5s
classes top1_acc top5_acc 100% 17/17 39.5it/s 0.4s
all 0.983 0.999

100 epochs completed in 0.415 hours.
Optimizer stripped from /home/run/yolov8/yolov8_final/weights/best.pt, 31.7MB
Optimizer stripped from /home/run/yolov8/yolov8_final/weights/best.pt, 31.7MB

Validating /home/run/yolov8/yolov8_final/weights/test.pt...
Ultralytics 8.3.229 | Python 3.10.16 torch 2.9.1+cu128 CUDA 10 (NVIDIA RTX A5000, 24123MiB)
YOLOv8cls summary (Fused): 42 layers, 15,774,185 parameters, 0 gradients, 41.0 GFLOPs
Training validation: validate_val() -> validating 1000 images in 9 classes [green]
val: /home/colormap_truncate_split/val... found 1063 images in 9 classes [green]
test: /home/colormap_truncate_split/test... found 1065 images in 9 classes [green]
classes top1_acc top5_acc 100% 17/17 30.4it/s 0.6s

Speed: 0.1m preprocess, 0.3m inference, 0.0m loss, 0.0m postprocess per image
Results saved to /home/run/yolov8/yolov8_final
Learn more at https://docs.ultralytics.com/models/train
(was: runffcc535442#dadd)→
```

Figure: YOLOv11 - Trained for 100 epochs and achieved an accuracy of 98.5%.

YOLOv8 Final Image Batches

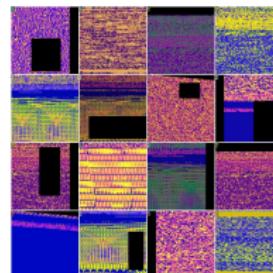
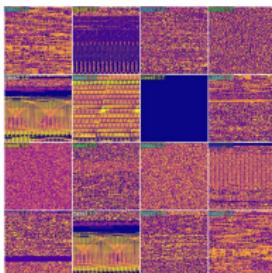
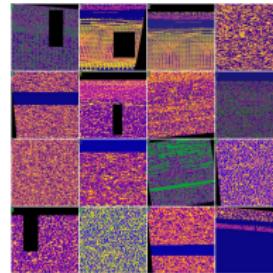
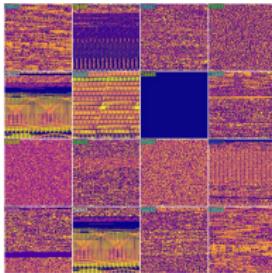


Figure: YOLOv8 validation batches.

Figure: YOLOv8 training batches.

Confusion Matrix for YOLOv8

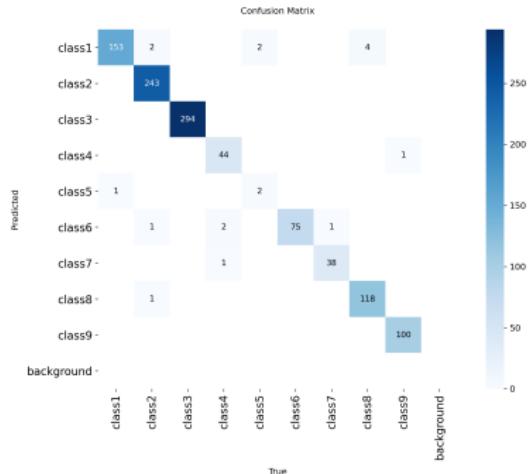


Figure: YOLOv8 confusion matrix normalized.

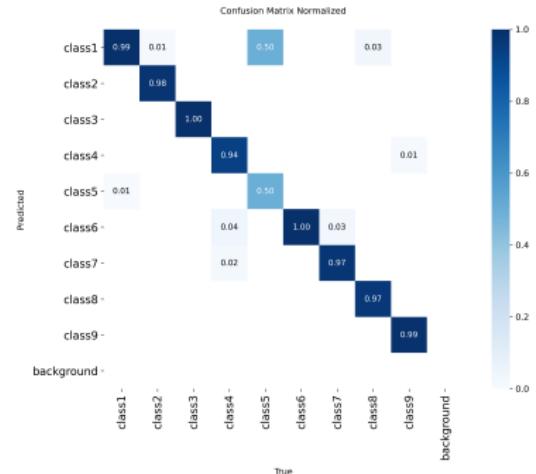


Figure: YOLOv8 confusion matrix normalized.

YOLOv8 Final Result

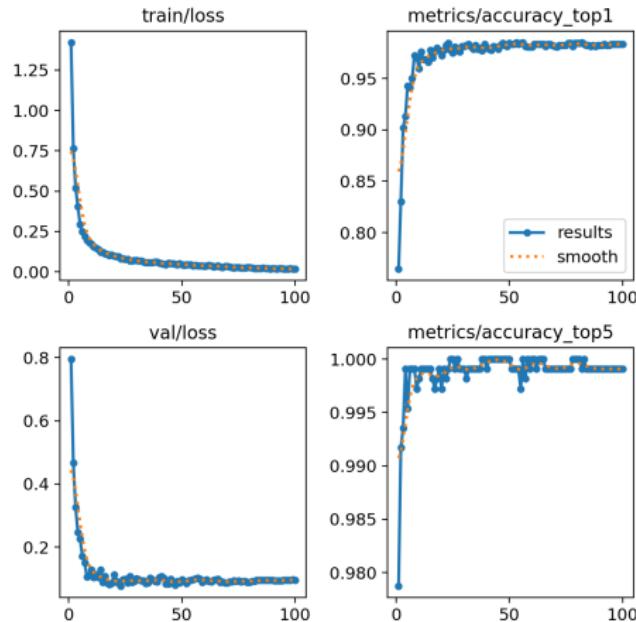


Figure: YOLOv8 Final result

YOLO 11 smoke test

The screenshot shows a Jupyter Notebook interface with a terminal window displaying the output of a YOLOv11 smoke test. The terminal output includes training logs, performance metrics, and validation results.

```

val: Scanning /home/colormap_truncate_split/val... 1083 images, 0 corrupt: 100% 1083/1083 11.6Bit/s 0.0s
optimizer: 'optimizer=auto' found, ignoring 'lr=0.01' and 'momentum=0.937' and determining best 'optimizer', 'lr' and 'momentum' automatically...
Using sizes 224 train: 224 val
Logging results to /home/runs/yolo11/yln_smoke_cls2
Starting training for 5 epochs...

```

Epoch	GPU mem	loss	Instances	Size
1/5	0.7956	1.254	59	224: 100%
	classes	top1_acc	top1_acc:	100% 9/9 8.1Bit/s 1.1s
	all	0.79	0.993	
2/5	0.9776	0.5137	59	224: 100%
	classes	top1_acc	top1_acc:	100% 9/9 23.8Bit/s 0.4s
	all	0.769	0.995	
3/5	0.9866	0.4027	59	224: 100%
	classes	top1_acc	top1_acc:	100% 9/9 16.8Bit/s 0.5s
	all	0.893	0.998	
4/5	0.9946	0.304	59	224: 100%
	classes	top1_acc	top1_acc:	100% 9/9 31.4Bit/s 0.3s
	all	0.958	0.998	
5/5	0.9946	0.2349	59	224: 100%
	classes	top1_acc	top1_acc:	100% 9/9 23.4Bit/s 0.4s
	all	0.969	0.998	

5 epochs completed in 0.018 hours.
Optimizer striped from /home/runs/yolo11/yln_smoke_cls2/weights/best.pt, 3.2MB
Optimizer striped from /home/runs/yolo11/yln_smoke_cls2/weights/best.pt, 3.2MB

```

Validating /home/runs/yolo11/yln_smoke_cls2/weights/best.pt...
Using GPU 0.228 PyTorch 1.7.1 is total: 2.32GB (NVIDIA RTX 4090, 24128MB)
YOLOv11 cls (0.228 PyTorch) is total: 2.32GB (NVIDIA RTX 4090, 24128MB)
YOLOv11 cls (0.228 PyTorch) is total: 1,083,553 parameters, 0 gradients, 3.2 LOPs
train: /home/colormap_truncate_split/train... found 8689 images in 9 classes ✓
val: /home/colormap_truncate_split/val... found 1083 images in 9 classes ✓
test: /home/colormap_truncate_split/test... found 1083 images in 9 classes ✓
    classes top1_acc top1_rec 100% 9/9 14.4Bit/s 0.6s
    all 0.969 0.998
Speed: 0.1ms preprocess, 0.3ms inference, 0.0ms loss, 0.0ms postprocess per image
Results saved to /home/runs/yolo11/yln_smoke_cls2
!Learn more at https://docs.ultralytics.com/models/train
(wenv) root@0fb0532edc:~#

```

Figure: YOLOv11 smoke test to check if all systems are ready.

YOLOv11 Smoke test Image Batches

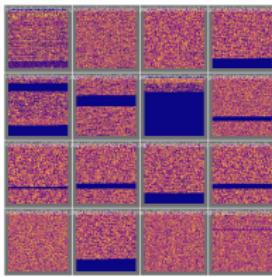
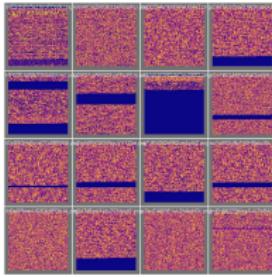


Figure: YOLOv11 validation batches.

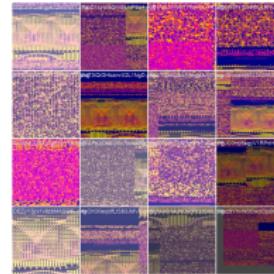
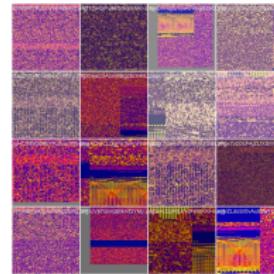


Figure: YOLOv11 training batches.

YOLO 11 Final Run

The screenshot shows a Jupyter Notebook interface with a terminal window titled "Terminal 1". The terminal displays the command-line output of a YOLOv11 training session. The logs show multiple epochs (95, 96, 97, 98, 99, 100) being completed on a GPU, with metrics like loss, instances, and size. The final message indicates 100 epochs completed in 0.481 hours, validation results, and the saving of best.pt and final.pt models.

```
classes top1_acc top5_acc: 100% 17/17 39.9it/s 0.4s
all 0.988 0.998
Epoch 95/100 GPU:mem loss Instances Size
1.946 0.0216 18 224: 100% 272/272 17.6it/s 16.9s
classes top1_acc top5_acc: 100% 17/17 36.6it/s 0.5s
all 0.989 0.998
Epoch 96/100 GPU:mem loss Instances Size
1.946 0.02173 18 224: 100% 272/272 16.8it/s 16.2s
classes top1_acc top5_acc: 100% 17/17 39.4it/s 0.4s
all 0.989 0.998
Epoch 97/100 GPU:mem loss Instances Size
1.946 0.02196 18 224: 100% 272/272 16.9it/s 16.1s
classes top1_acc top5_acc: 100% 17/17 36.8it/s 0.5s
all 0.989 0.998
Epoch 98/100 GPU:mem loss Instances Size
1.946 0.02111 18 224: 100% 272/272 16.5it/s 16.5s
classes top1_acc top5_acc: 100% 17/17 37.5it/s 0.5s
all 0.988 0.998
Epoch 99/100 GPU:mem loss Instances Size
1.946 0.0211 18 224: 100% 272/272 16.5it/s 16.7s
classes top1_acc top5_acc: 100% 17/17 48.6it/s 0.4s
all 0.988 0.998
Epoch 100/100 GPU:mem loss Instances Size
1.946 0.02088 18 224: 100% 272/272 16.8it/s 16.2s
classes top1_acc top5_acc: 100% 17/17 44.3it/s 0.4s
all 0.988 0.998
100 epochs completed in 0.481 hours.
Optimizer stripped from /home/runs/yolol11/yolol11_final_cli/weights/last.pt, 20.9MB
Optimizer stripped from /home/runs/yolol11/yolol11_final_cli/weights/best.pt, 20.9MB
Validating /home/runs/yolol11/yolol11_final_cli/weights/best.pt...
Ultralytics 8.3.22 | PyTorch-1.10.16 torch-2.3.1+cu128 CUDA-RT 450000_24129MB
YOLOv11 trained on 1000 images, 1000 classes, 1000 gradients, 39.3 GFLOPs
train: /home/colormap_truncate_split/train... found 8929 images in 9 classes ✓
val: /home/colormap_truncate_split/val... found 1083 images in 9 classes ✓
test: /home/colormap_truncate_split/test... found 1095 images in 9 classes ✓
classes top1_acc top5_acc: 100% 17/17 28.2it/s 0.6s
all 0.989 0.998
Speed: 0.1m preprocess, 0.4m inference, 0.0m loss, 0.0ms postprocess per image
Results saved to /home/runs/yolol11/yolol11_final_cli
!Leave note at https://docs.ultralytics.com/nodes/train
(wvenv) root@ebcfdb553bed:~#
```

Figure: YOLOv11 - Trained for 100 epochs and achieved an accuracy of 98.9%.

YOLOv11 Final Image Batches

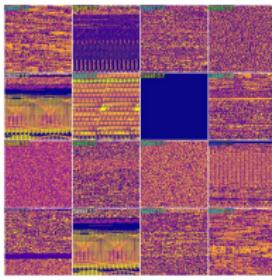
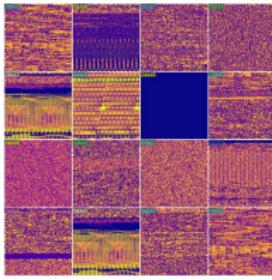


Figure: YOLOv11 validation batches.

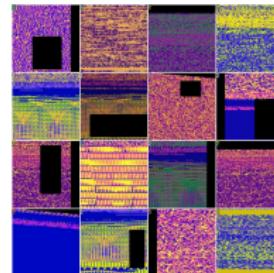
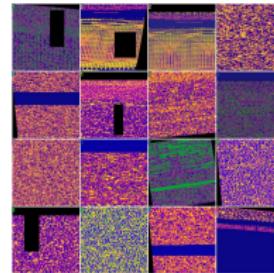


Figure: YOLOv11 training batches.

Confusion Matrix for YOLOv11

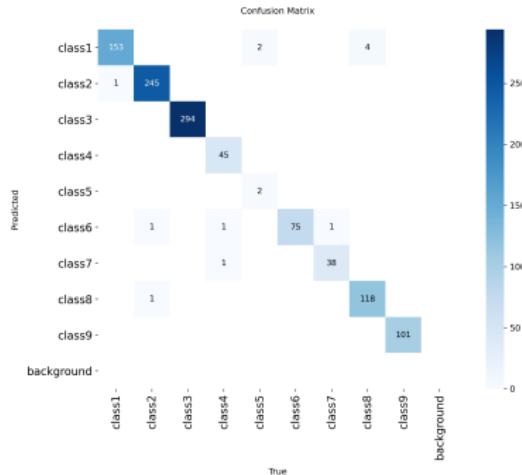


Figure: YOLOv11 confusion matrix normalized.

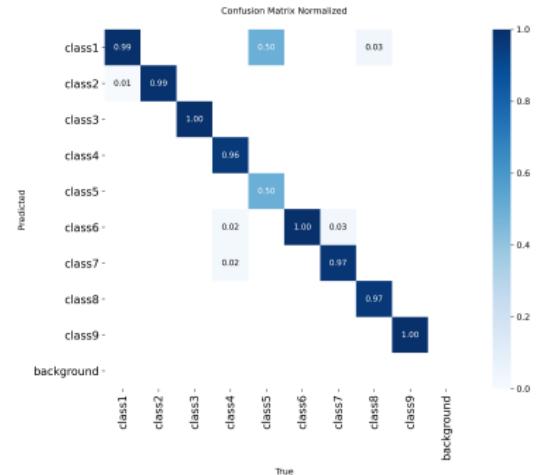


Figure: YOLOv11 confusion matrix normalized.

YOLOv11 final result

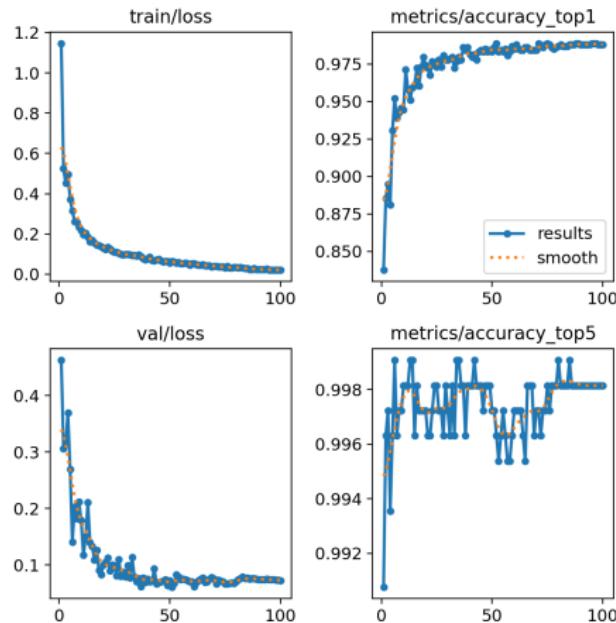


Figure: YOLOv11 results graphs