



Northeastern University

CONSTRUCTION SAFETY

Open hole / Unsafe Excavation Detection

Van Kieu Dang

Northeastern University

ALY6890: Capstone Project





OBJECTIVES



Open hole / unsafe excavation detection

This model combined with PPE detection model will be deployed to our Jetson-based edge inference systems for job sites allowing us to monitor and report on unsafe conditions surrounding excavation.



Knowledge acquisition

- Pipeline, workflow
- Object detection
- Automation
- Tools and Platforms



CONTENT

I. Insight Exploration



1. Iterative workflow



2.1 Data collection



2.2 Annotation



2.3 Pre-processing



3. Model training



4. Model Performance Comparison

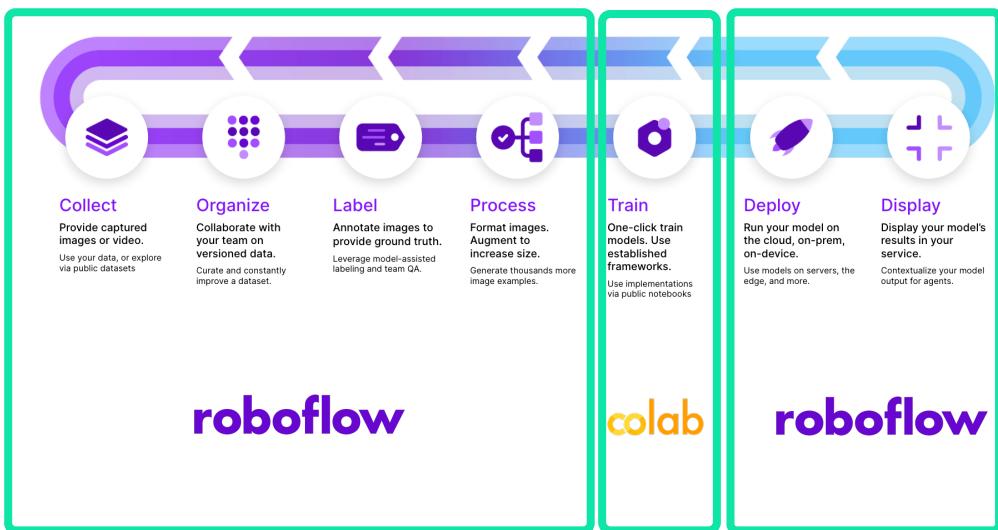


5. Recommendations

III. Challenges and Progress

I. KEY INSIGHTS

1. Iterative Workflow



1

Identify **misclassifications** (FP, FN) and makes necessary improvements.



2

Update **hyper-parameters** used.

```
# Create the Trainer
trainer = Trainer(
    accelerator="gpu",
    max_epochs=200,
    precision='32-true',
    callbacks=[early_stopping_callback],
    accumulate_grad_batches=16,
    gradient_clip_val=0.5,
    log_every_n_steps=5,
    enable_model_summary=True
)

# Use GPU acceleration
# Train for a maximum of 50 epochs
# Use 32-bit floating point precision
# Early stopping callback
# Accumulate gradients over 16 batches before
# Clip gradients if their norm exceeds 0.5
# Set frequency of logging training metrics
# Enable model summary
```

3

Adjust settings, **revisit** data augmentation methods, or **incorporate** additional labeled data.

I. KEY INSIGHTS



2.1. Data Collection

Various scenarios that models can generalize effectively.



Diverse and representative



Balanced class distribution

Class Balance

open-hole
Safety-bollard
Safety-barrier
Safety-cone



Negative mining / augmentation

Images

694

0 missing annotations
Ø 47 null examples

Annotations

1,206

1.7 per image (average)
across 4 classes



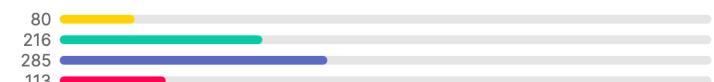
Leverage Health check - Dimension

Dimension Insights

Size Distribution

The purple box indicates the median width by median height image (640x512).

small
medium
large
jumbo



I. KEY INSIGHTS



2.2. Data Annotation

Insights to improve Consistency and Data quality



Multi-scale and complex annotation



Large



Medium

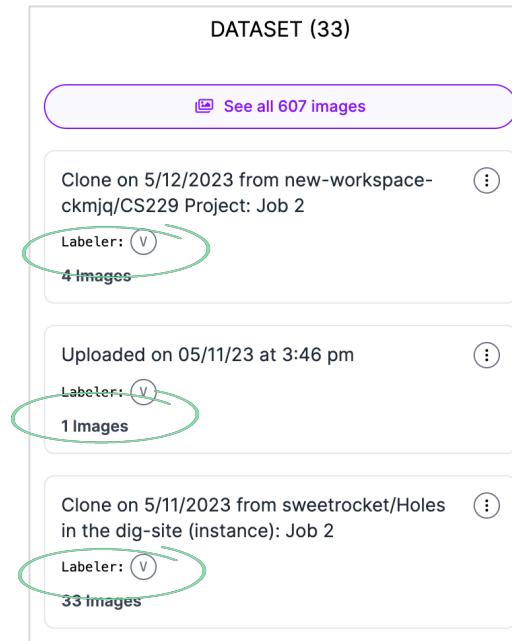


Small



Annotator guidelines

Only a few members are mapping annotations for **consistency** of data, strictly following guidelines.



DATASET (33)

See all 607 images

Clone on 5/12/2023 from new-workspace-ckmjg/CS229 Project: Job 2
Labeler: V 4 Images

Uploaded on 05/11/23 at 3:46 pm
Labeler: V 1 Images

Clone on 5/11/2023 from sweetrocket/Holes in the dig-site (instance): Job 2
Labeler: V 33 Images



Smart Annotation

Leverage smart annotation from Roboflow to **boost data preparation progress**.



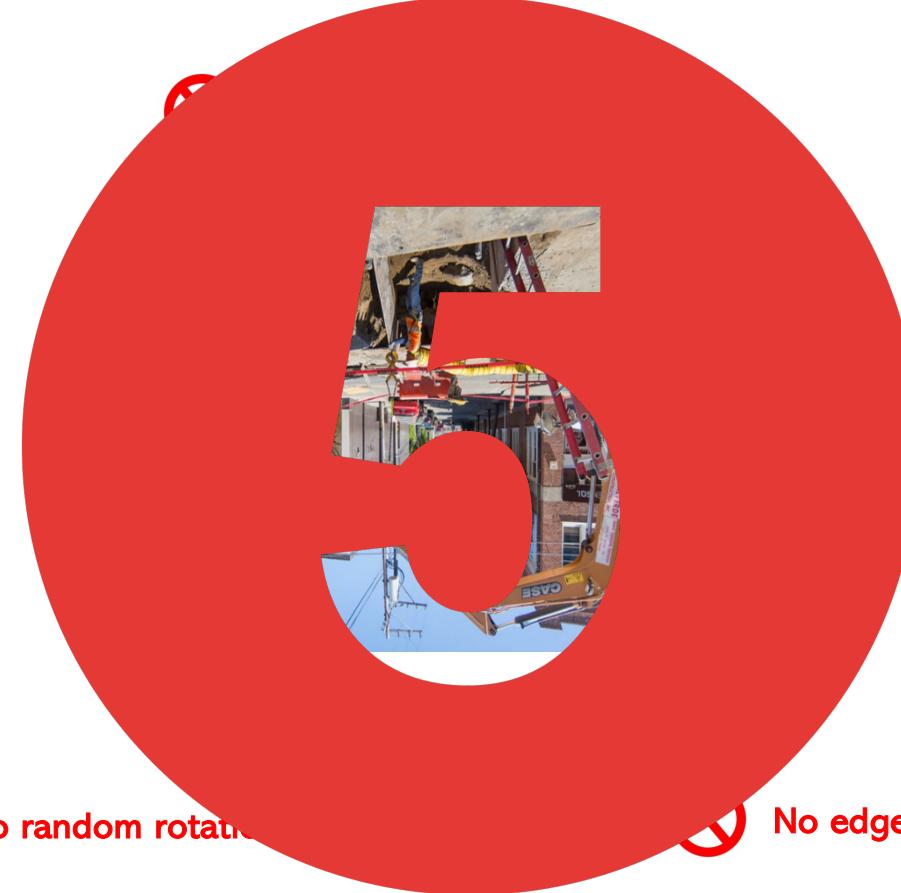
I. KEY INSIGHTS



2.3. Data Preprocessing

To increase dataset size and prepare data for training

 No black and white



 No perspective transformation



 No random rotation

 No edge detection (unnecessary)

I. KEY INSIGHTS



3. Model Training

When it comes to model training, various aspects we can take into account such as hyper-parameters, pre-trained models, optimizers, and so on. In this part, we only highlight some insights from model tuning:

- img-size: Define input image size, (default value: [640, 640]).
- workers: Define maximum number of dataloader workers (default value: 8).
- batch-size: Determine batch size.
- epochs: Define the number of training epochs.
- data: Our dataset location.
- weights: Specify a path to weights to start transfer learning from. Here we choose the generic COCO pretrained checkpoint from Yolov7.
- cache-image: Cache images for faster training.
- device: cuda device, i.e. 0 or 0,1,2,3 or cpu.
- p (patience): early stop hyper-parameter (Model will stop after N epochs without improvement). In this case, N =20.

```
# Training the model
#cd /content/yolov7
!python train.py --img-size 640 --batch-size 16 --workers 8 --epochs 200 --device 0 --data {dataset.location}/data.yaml
```

YOLO

```
# Define the EarlyStopping callback
early_stopping_callback = EarlyStopping(
    monitor='validation/loss',
    mode='min',
    patience=20.          # Update the number of epochs for early stop
)
# Set model
model = Detr(lr=1e-4, lr_backbone=1e-5, weight_decay=1e-4)
batch = next(iter(TRAIN_DATALOADER))
outputs = model(pixel_values=batch['pixel_values'], pixel_mask=batch['pixel_mask'])

# Create the Trainer
trainer = Trainer(
    accelerator="gpu",
    max_epochs=200,
    precision='32-true',
    callbacks=[early_stopping_callback],
    accumulate_grad_batches=16,
    gradient_clip_val=0.5,
    log_every_n_steps=5,
    enable_model_summary=True
)

# Start training
trainer.fit(model)
```

DETR



Optimization

- Choose a small **learning rate**: 0.001 / 1e-4.
- More **epochs**.
- Set **early stop**.
- Medium **batch size**.
- Set '**cache**' for faster training.
- Apply **Weight decay** to prevent overfitting.
- Unfreeze up to n-1 layers to train.
- Use **Non-max suppression**.
- ...

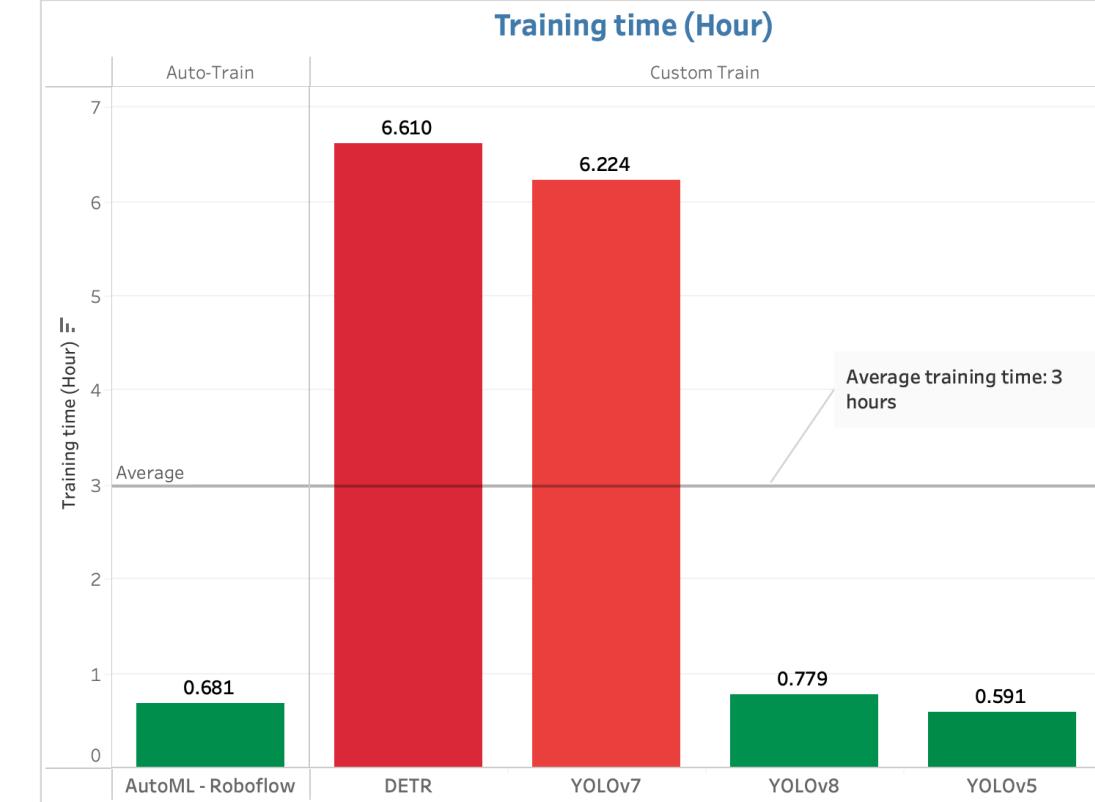
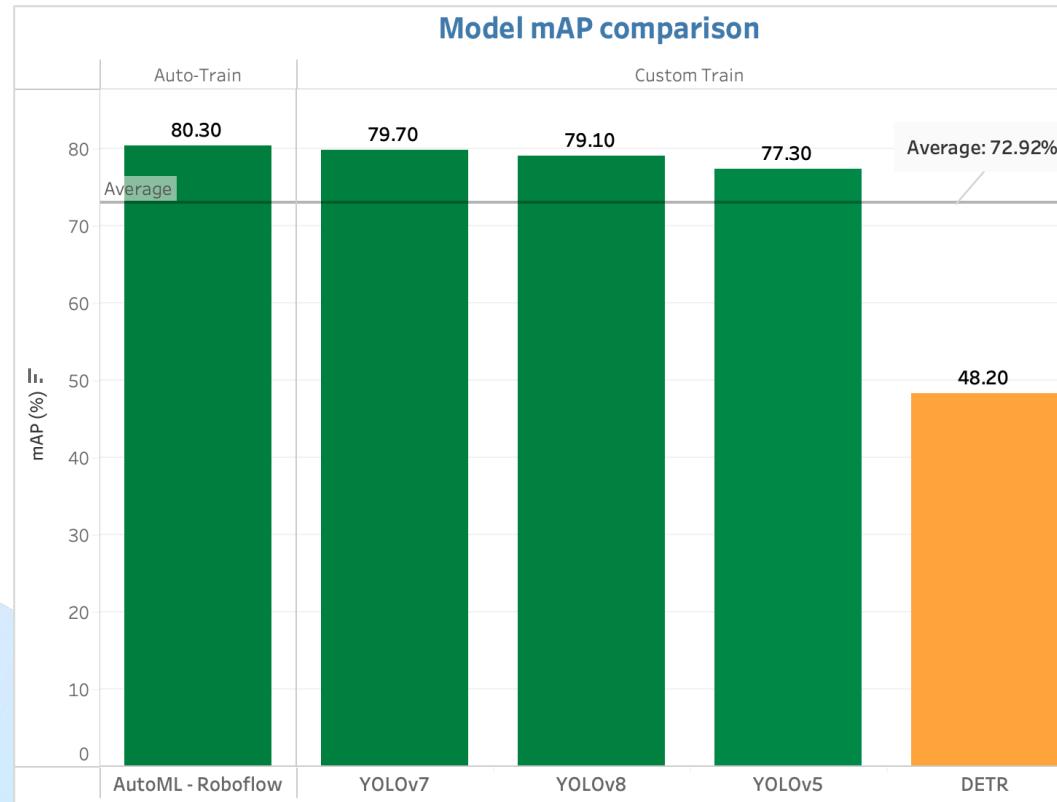
II. KEY RESULTS



5. Comparison

Performance for each model is displayed in detail on respective Tensorboard from our notebooks.

Note: Results are derived from the same dataset, same hardware, and same training set up such as: 200 epochs, patience of 20, batch size of 16,...



II. KEY RESULTS



6. Recommendations

1

No	Category	Model	mAP (%)	Speed / Training time (Hour)	Model size (MB)	Framework Compatibility
1	Custom train	YOLOv5	77.3	0.591	14.5	Darknet, OpenCV, Pytorch, Tensorflow, Keras...
2	Custom train	YOLOv7	79.7	6.224	74.8	Darknet, OpenCV, Pytorch, Tensorflow, Keras...
3	Custom train	YOLOv8	79.1	0.779	22.5	Darknet, OpenCV, Pytorch, Tensorflow, Keras...
4	Custom train	DETR	48.2	6.61	166.6	Pytorch
5	Auto train	AutoML - Roboflow	80.3	0.681	N/A	Variety



2

Focus more on **Data** than **Modeling** after this phase.

III. SUMMARY



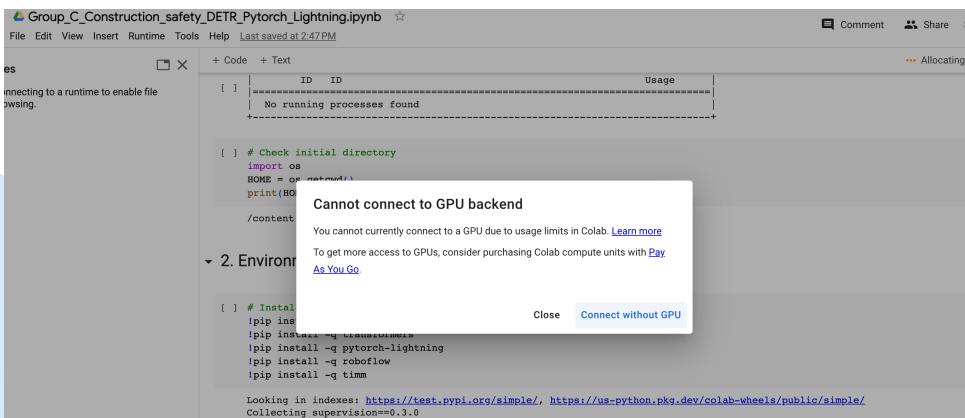
CHALLENGES

- Open ended project without target images and available dataset.
- Target objects are hard to detect.



No standards, different size, different shapes,...

- Limitation of hardware: Google Colab free tier



```

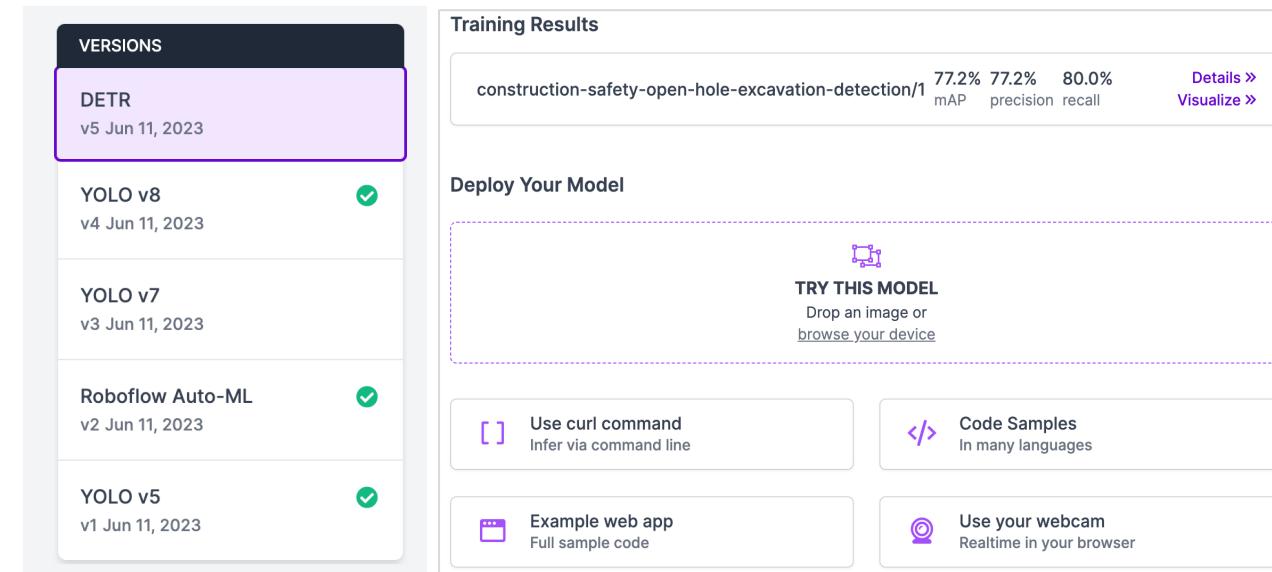
Group_C_Construction_safety_DETR_Pytorch_Lightning.ipynb
File Edit View Insert Runtime Tools Help Last saved at 2:47PM
Comment Share
Connecting to a runtime to enable file browsing.
[ ] ID ID Usage Allocating
[ ] No running processes found
+ Text
[ ] # Check initial directory
import os
HOME = os.getenv('HOME')
print(HOME)
/content
[ ] Cannot connect to GPU backend
You cannot currently connect to a GPU due to usage limits in Colab. Learn more
To get more access to GPUs, consider purchasing Colab compute units with Pay As You Go.
[ ] 2. Environment
[ ] # Install dependencies
!pip install torch torchvision
!pip install -q pytorch-lightning
!pip install -q roboflow
!pip install -q timm
Looking in indexes: https://test.pypi.org/simple/, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting supervision==0.3.0

```



PROGRESS

- Data seem to be the most critical part of model performance.
- Acquire Pipeline, Automation, Tools (Roboflow + Colab) and Object Detection techniques as well as best practices from the company.
- Implemented successfully several models



VERSIONS

DETR	v5 Jun 11, 2023
YOLO v8	v4 Jun 11, 2023
YOLO v7	v3 Jun 11, 2023
Roboflow Auto-ML	v2 Jun 11, 2023
YOLO v5	v1 Jun 11, 2023

Training Results

construction-safety-open-hole-excavation-detection/1	77.2% mAP	77.2% precision	80.0% recall
--	-----------	-----------------	--------------

Details » Visualize »

Deploy Your Model

TRY THIS MODEL
Drop an image or [browse your device](#)

Use curl command Infer via command line

Code Samples In many languages

Example web app Full sample code

Use your webcam Realtime in your browser

- Models under development: Mask RCNN and Unet.



CONSTRUCTION SAFETY

THANK YOU



Appendix - Dataset

Search over 90,000 Open Source Computer Vision Projects

Van Kieu Dang - Kiel

Browse How to Search

Show Editable View >

Construction Safety - Open hole / Excavation Detection Object Detection

Overview

Images 694

Dataset 5

Model

API Docs

Health Check

Filename : Split : ALL TRAIN VALID TEST Classes: All Classes Sort By: Newest

De-Select Select All Select Images to Clone Search

Image	Label	File Name
	TRAIN	00000204.jpg
	TRAIN	th (6).jpg
	TRAIN	00000330.jpg
	TRAIN	3.jpg
	TRAIN	cccc.jpg
	TRAIN	th (20).jpg
	TRAIN	00090.jpg
	TRAIN	th (24).jpg
	VALID	00000017.jpg
	TRAIN	00410.jpg
	TRAIN	00311.jpg
	TRAIN	th (19).jpg
	TRAIN	00041.jpg
	TRAIN	00407.jpg
	TRAIN	djjc.jpg
	TRAIN	DALL-E 2023-06...
	TRAIN	DALL-E 2023-06...
	VALID	00396.jpg
	TRAIN	00100.jpg
	VALID	DALL-E 2023-06...
	TRAIN	00350.jpg
	TRAIN	00192.jpg
	VALID	00099.jpg
	VALID	00243.jpg

Link to the project: <https://universe.roboflow.com/northeastern-4sfxe/construction-safety-open-hole-excavation-detection>



Appendix – Model Versions

Construction Safety - Open hole / Excavation Detection Image Dataset

[Generate New Version](#)

VERSIONS	
DETR	v5 Jun 11, 2023
YOLO v8	✓ v4 Jun 11, 2023
YOLO v7	v3 Jun 11, 2023
Roboflow Auto-ML	✓ v2 Jun 11, 2023
YOLO v5	✓ v1 Jun 11, 2023

YOLO v5
Version 1 Generated Jun 11, 2023

[Export Dataset](#) [Edit](#)

ROBOFLOW TRAIN

MODEL TYPE: YOLOV5 MODEL UPLOAD

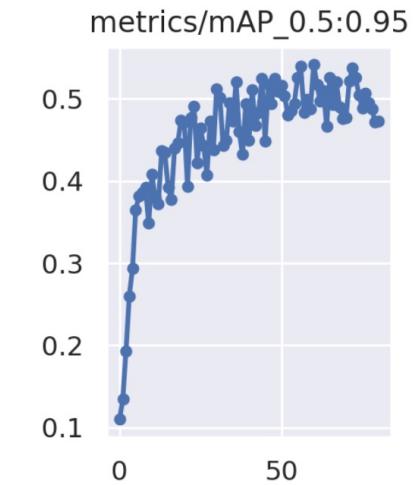
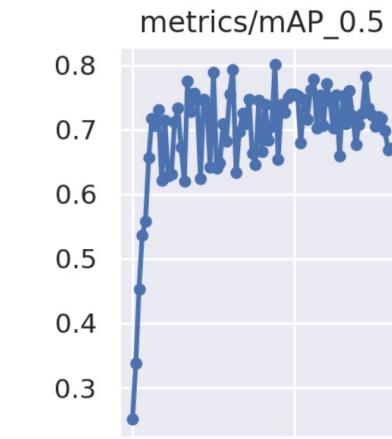
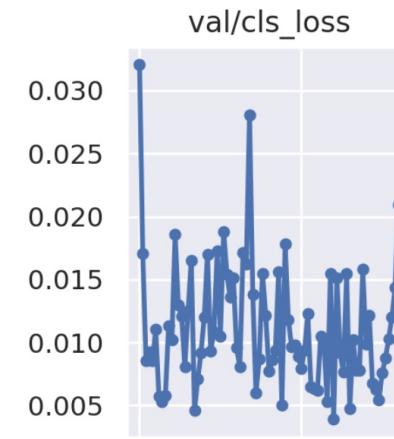
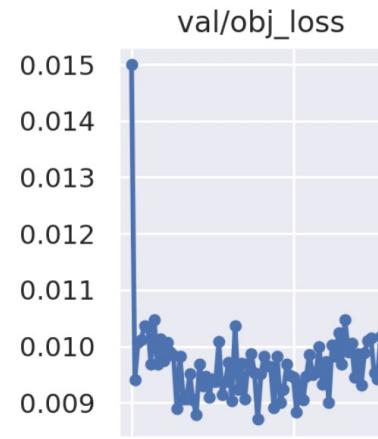
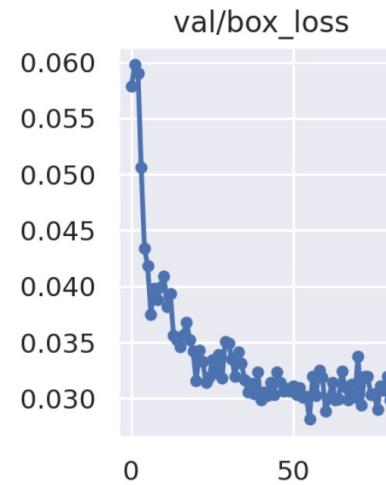
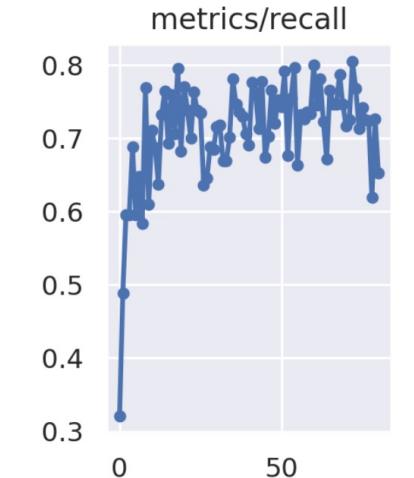
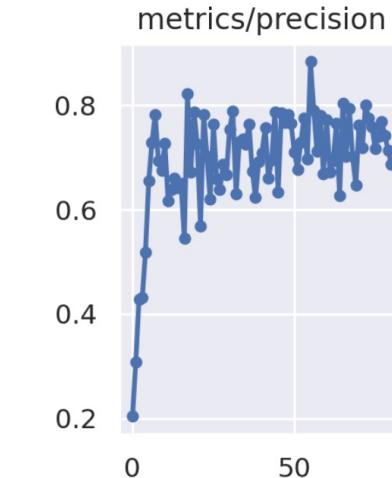
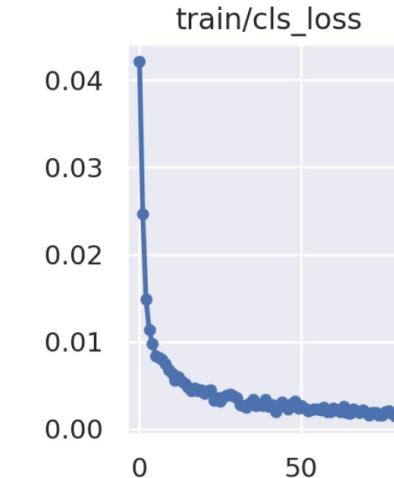
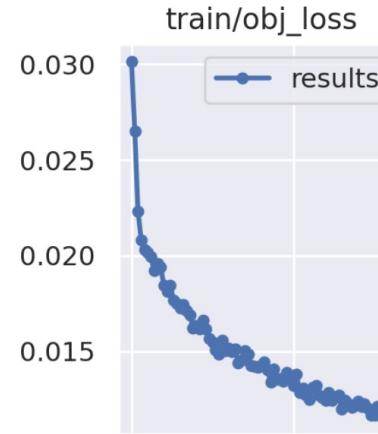
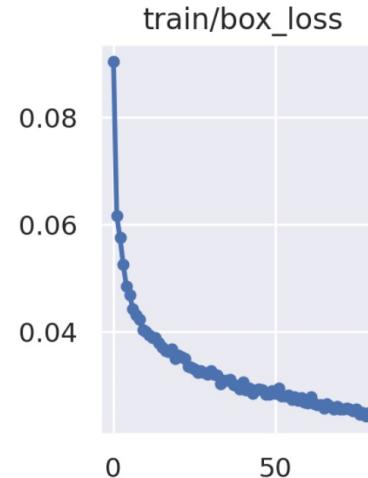
Training Results

construction-safety-open-hole-excavation-detection/1	77.2% mAP	77.2% precision	80.0% recall	Details » Visualize »
--	--------------	--------------------	-----------------	--

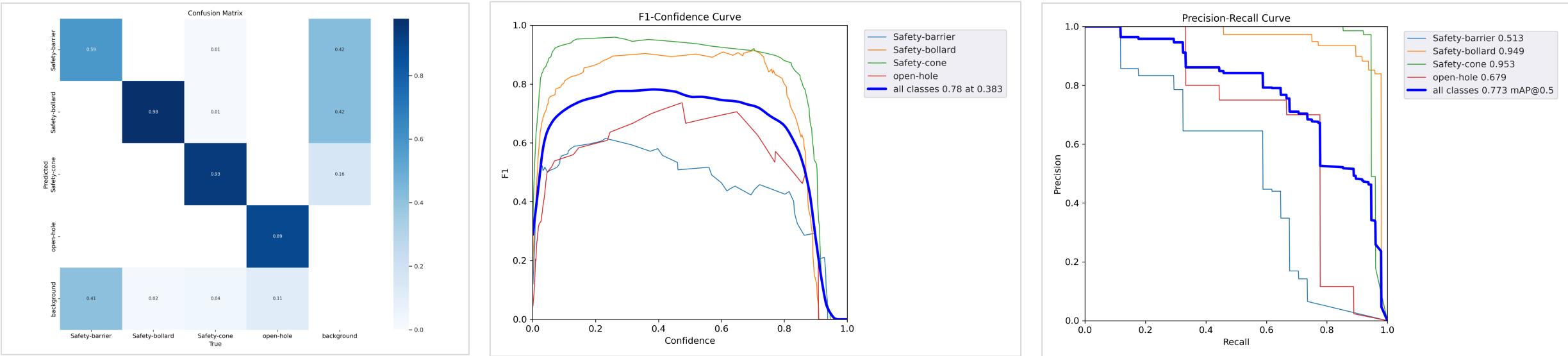
Deploy Your Model


TRY THIS MODEL
Drop an image or
[browse your device](#)

Appendix – YOLO v5 Performance Details



Appendix – YOLO v5 Performance Details



Ground truth



Prediction

