1. In a relatively up-to-date Python environment:

```
pip install ultralytics
```

Or git clone from https://github.com/ultralytics/ultralytics. It's recommended to have PyTorch already installed. Here's a link to ultralytic's own install instructions for further clarifications: https://docs.ultralytics.com/quickstart/#install-ultralytics

Training Custom Dataset on Yolov8

1. Preparing Dataset: Assuming you've collected and annotated your dataset into yolov8 format(if you haven't roboflow can convert different annotations into yolov8 format), make or make sure your <data>.yaml is accurate.

Open

data.yaml
~/YOLOv6_etc/robot_yolo8

```yaml
train: /home/aralab/YOLOv6_etc/robot_yolo8/train/images
val: /home/aralab/YOLOv6_etc/robot_yolo8/valid/images
test: /home/aralab/YOLOv6_etc/robot_yolo8/test/images

nc: 2
names: ['crack', 'spall']

```

The paths to train, val, and test must be accurately referencing the folders containing the images for training, validation, and testing. nc refers to the number of classes the model is meant to recognize; names is what the names of classes are/the labels.

2. Choosing a Yolov8 model:

Model	size (pixels)	mAP^{val} 50-95	Speed CPU ONNX (ms)	Speed A100 TensorRT (ms)	params (M)	FLOPs (B)
YOLOv8n	640	37.3	80.4	0.99	3.2	8.7
YOLOv8s	640	44.9	128.4	1.20	11.2	28.6
YOLOv8m	640	50.2	234.7	1.83	25.9	78.9
YOLOv8l	640	52.9	375.2	2.39	43.7	165.2
YOLOv8x	640	53.9	479.1	3.53	68.2	257.8

Choose the pre-trained model best suited for your needs.The screenshot is from https://github.com/ultralytics/ultralytics

3. Training Model on Custom Dataset:

On Terminal:

One GPU:

```
yolo detect train data=data.yaml model=yolov8n.pt epochs=100 imgsz=640
```

Multiple GPUs(Example below is specifically two GPUs):

```
yolo detect train data=data.yaml model=yolov8n.pt epochs=100 imgsz=640 device=0,1
```

Resume Interrupted Training:

```
yolo train resume model=path/to/last.pt
```

<model_name>.pt is the model which is also sometimes called weights.

On Python:

```
from ultralytics import YOLO
model = YOLO('yolov8n.pt')  # load a pretrained model

# Train the model
results = model.train(data='data.yaml', epochs=100, imgsz=640)

# Train the model with 2 GPUs
results = model.train(data='data.yaml', epochs=100, imgsz=640, device=[0, 1])

# Train the model on Apple chips
results = model.train(data='data.yaml', epochs=100, imgsz=640, device='mps')

# Resume Interrupted Training
model = YOLO('path/to/last.pt')  # load a partially trained model
results = model.train(resume=True)
```

When the training is done, it will tell you where it put the models but it is generally in current_directory/runs/detect/train#/weights. The best model is named best.pt.

Source and more info: https://docs.ultralytics.com/modes/train/#arguments

4. Validating Custom Model:

On Terminal:

```
yolo detect val model=path/to/best.pt
```

On Python:

```
model = YOLO('path/to/best.pt')  # load a custom model

# Validate the model
metrics = model.val()  # no arguments needed, dataset and settings remembered
metrics.box.map    # map50-95
metrics.box.map50  # map50
metrics.box.map75  # map75
metrics.box.maps   # a list contains map50-95 of each category
```

Source and more info: https://docs.ultralytics.com/modes/val/#key-features-of-val-mode

5. Predicting using Custom Model:

On Terminal:

```
yolo predict model=best.pt source='path/images/bus.jpg' save=True
```

On Python:

```
from ultralytics import YOLO

# Load a pretrained YOLOv8n model
model = YOLO('best.pt')

# Define path to what you want to be predicted on
source = 'path/to/image.jpg'

# Run inference on the source
results = model(source, save=True, conf=0.48)
```

save=<bool> saves the model prediction onto the source in runs/detect/predict#

conf=<float> is the minimum confidence the model is on a prediction.

save_txt=<bool> saves results of prediction into a txt

results in results=model(....) contain all the predictions for the source in a variety of formats; this includes the bounding boxes, the xywh, xyxy, etc.

What the model is making predictions on or source itself can be most images and videos.

For more specifics and other arguments:
https://docs.ultralytics.com/modes/predict/#working-with-results