NOAA FISHERIES

PIFSC/ESD/ARP

AI/ML Lunch & Learn: Object Detection

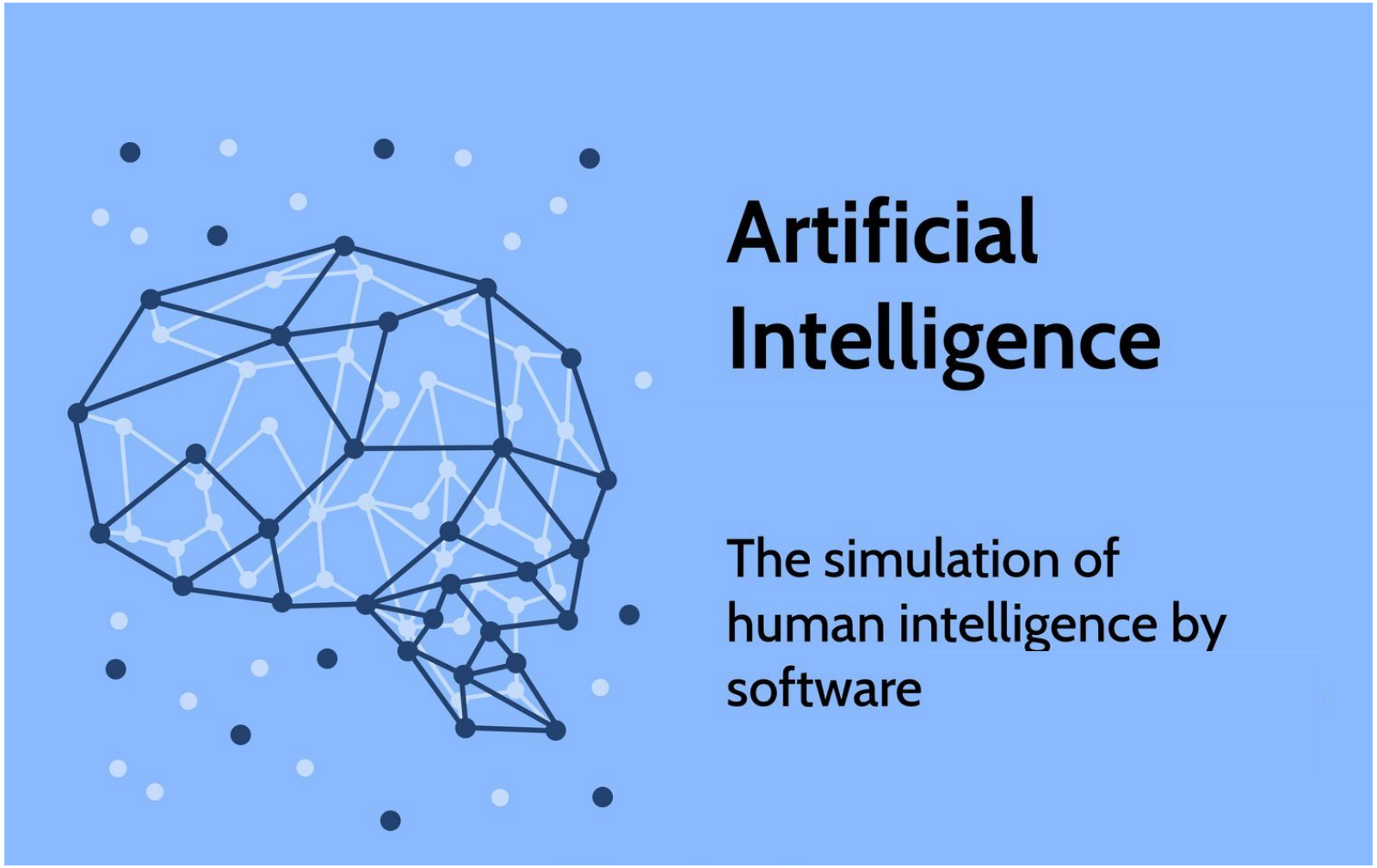
2025-01-08

michael.akridge@noaa.gov

What is Object Detection?

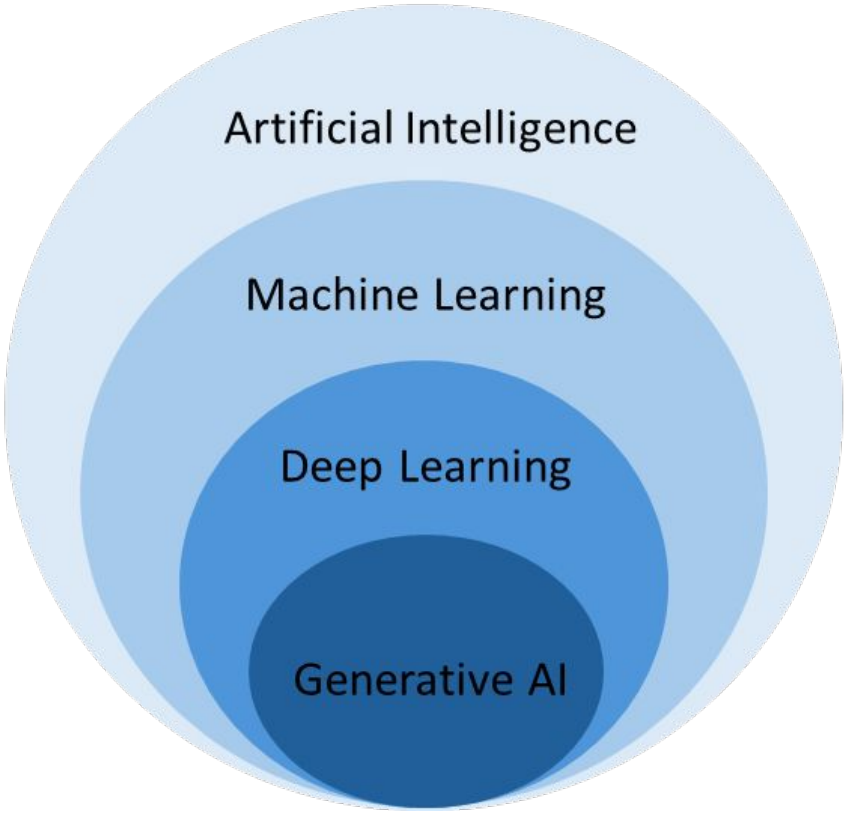
NOAA FISHERIES

AI



NOAA FISHERIES

Object Detection: Deep learning powered AI



Machine learning model

An algorithm that identifies patterns in a set of data and makes predictions over it.

What is AI? A complex pattern learner.

Artificial Intelligence (AI): The field of study making computer algorithms exhibit human-like intelligence and behaviors.

Machine Learning (ML): A subfield of AI using statistical methods to enable systems to learn from data.

Deep Learning (DL): A submethod of ML using deep neural networks to enable systems to learn from data.

Generative AI: A subset of models within DL that learn (from data) how to create brand new content in the form of text, images, etc.

NOAA FISHERIES

Object Detection: Computer Vision

A CV field w/ a focus on identifying and locating specific objects in images

Key Components

1. **Object Localization:** Identifying the position of objects in the form of bounding boxes (rectangles) or other techniques

2. **Object Classification:** Assigning a class or label to each detected object.

NOAA FISHERIES

Is this a Cat?

Image Classification

What is in the image & where is it?

Object Detection

NOAA FISHERIES

NOAA FISHERIES

Examples In Hawaii: Wildfire Detection System

Hawaiian Electric

Hawaiian Electric deploys high resolution video cameras with artificial intelligence (AI) for early detection of wildfires

Strategically placed cameras will monitor 24/7 for ignitions

Release Date: 7/16/2024

Download PDF

HONOLULU, July 16, 2024 – Hawaiian Electric has begun deploying a network of high-resolution video cameras using artificial intelligence (AI) technology to provide enhanced situational awareness and early detection of ignitions in elevated fire risk areas near company infrastructure. The public will also have access to the live feeds from any of the cameras.

Hawaiian Electric recently installed the first camera station in Lahaina and has plans to deploy a total of 78 stations in elevated fire risk areas on the five islands served by the company, with each location having two cameras to provide a full 360-degree view. The camera feeds will be monitored 24 hours a day, seven days a week. The $14 million project is the latest step in Hawaiian Electric's ongoing effort to reduce the risk of wildfires associated with company equipment.

"We are continuing to take action to address the growing risks from wildfires across our service territory using a variety of technologies and methods," said Jim Alberts, Hawaiian Electric senior vice president and chief operations officer. "Installing publicly viewable AI-assisted video cameras in elevated fire risk areas will enable the company, fire agencies, and emergency operations centers the ability to identify potential wildfires early and respond quickly."

Hawaiian Electric signed a five-year contract with California-based ALERTWest. ALERTWest will install and maintain the camera stations as well as provide around-the-clock monitoring for potential ignitions by experienced wildfire safety professionals. Approximately 50% of the project costs will be covered by federal funds allocated under the federal Infrastructure Investment and Jobs Act (IIJA) estimated at $90 million in grant funding covering various costs related to Hawaiian Electric's resiliency and wildfire mitigation work. Hawaiian Electric also will be able to achieve cost savings by leveraging its existing telecom network to provide communications support for the project.

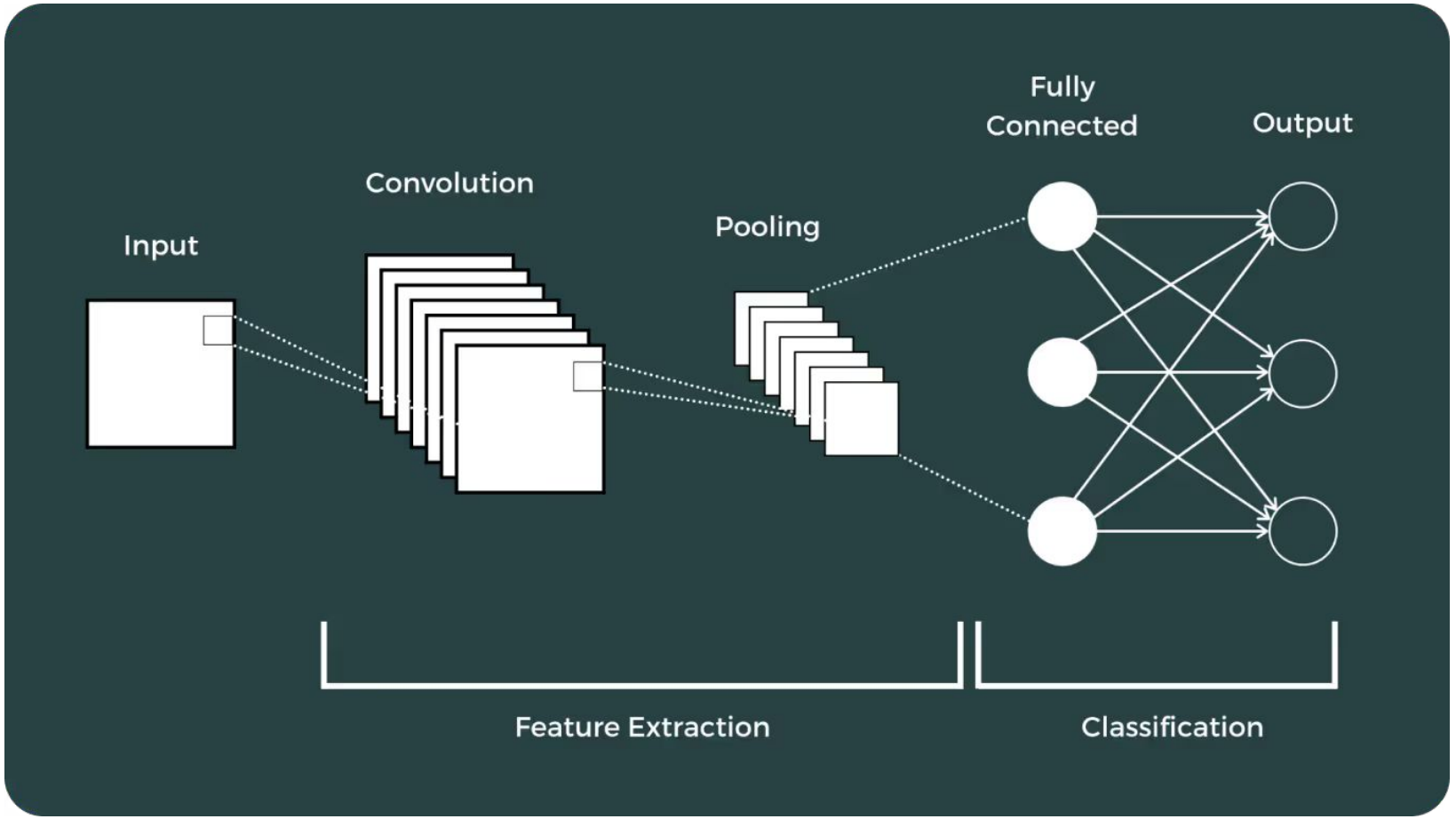
NOAA FISHERIES

Examples In Hawaii: Wildfire Detection System

How does this all work?

NOAA FISHERIES

Convolutional Neural Networks (CNNs)



"Convolutional refers to a process that involves intertwining two sources of information."

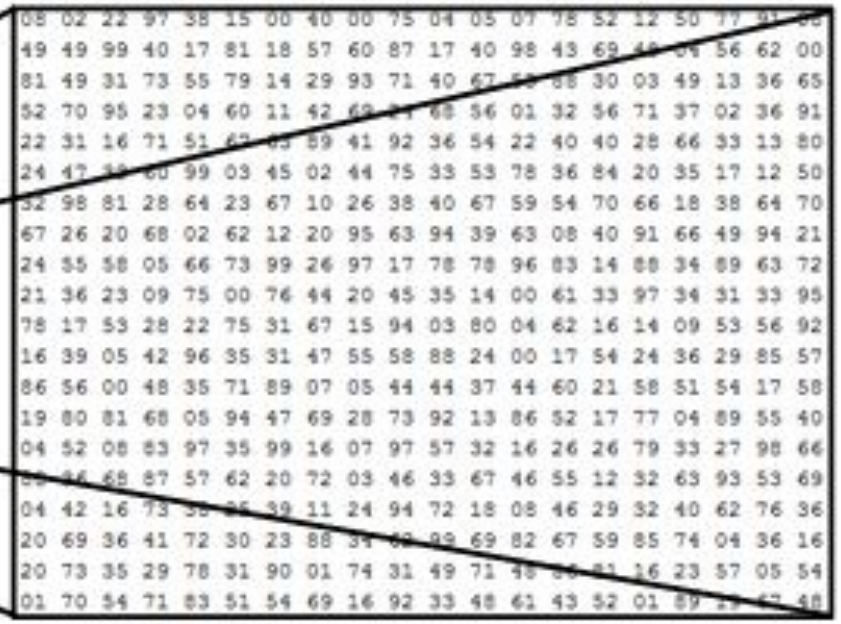
NOAA FISHERIES

A bunch of Data

+ Bunch of Math =

Classifier

NOAA FISHERIES



What the computer sees

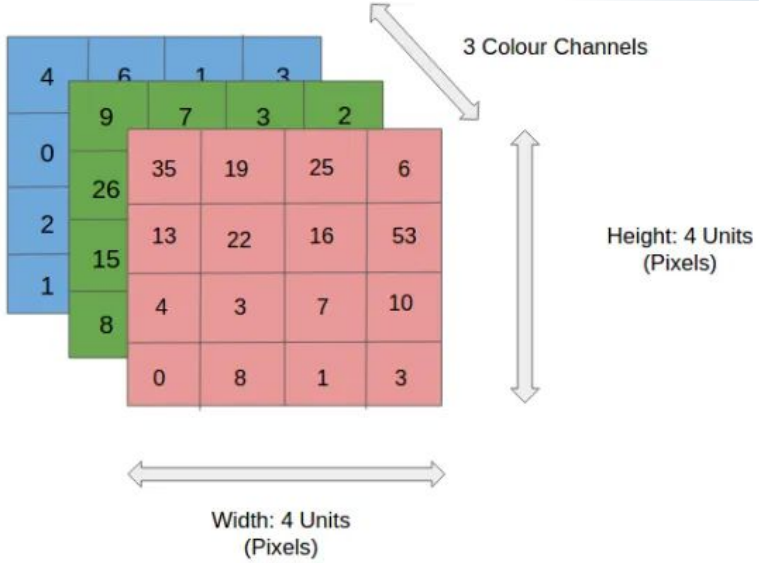
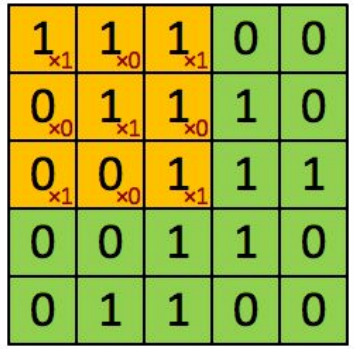
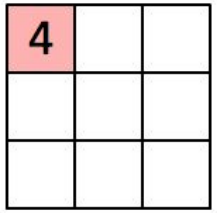


image classification

82% cat
15% dog
2% hat
1% mug



Image



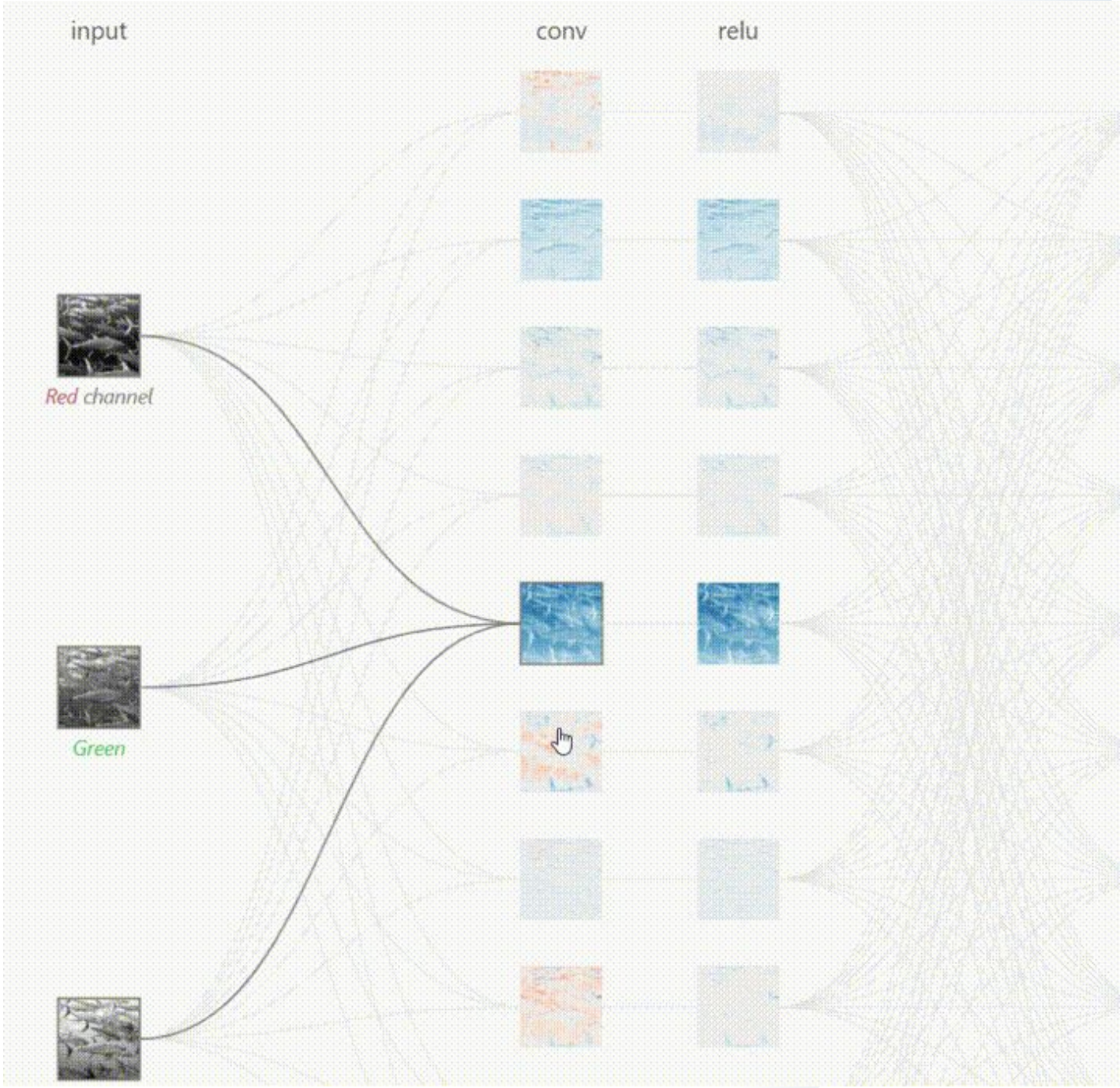
Convolved Feature

https://cs231n.github.io/
https://github.com/cs231n/cs231n.github.io
https://online.stanford.edu/courses/soe-ymls-machine-learning-specialization
https://online.stanford.edu/programs/artificial-intelligence-graduate-certificate

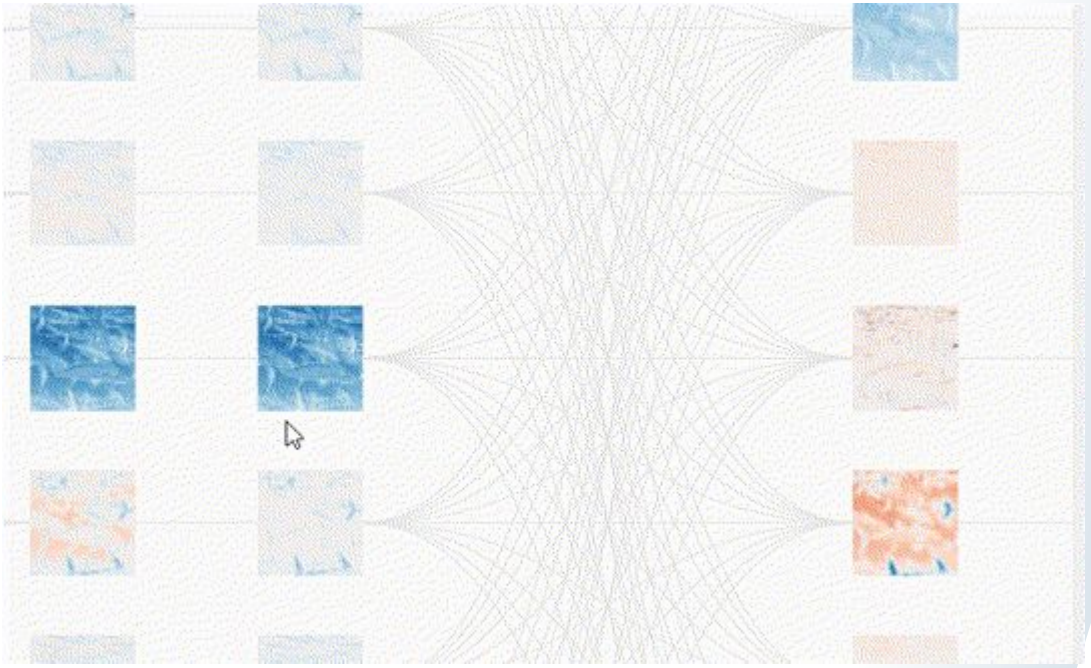
NOAA FISHERIES

The convolutional layer is the first layer of a CNN and is where most of the computations take place.

It uses a filter or kernel, which is a small matrix of weights, to move across an input image to detect specific features.



https://poloclub.github.io/cnn-explainer/



Learn More here:
https://poloclub.github.io/cnn-explainer/

NOAA FISHERIES

NOAA FISHERIES

Common Types Object Detectors

NOAA FISHERIES

Common Types of Object Detectors

(Two or Multi-Stage Detectors)
Region-Based Object Detectors

- R-CNN
- Fast R-CNN
- Faster R-CNN

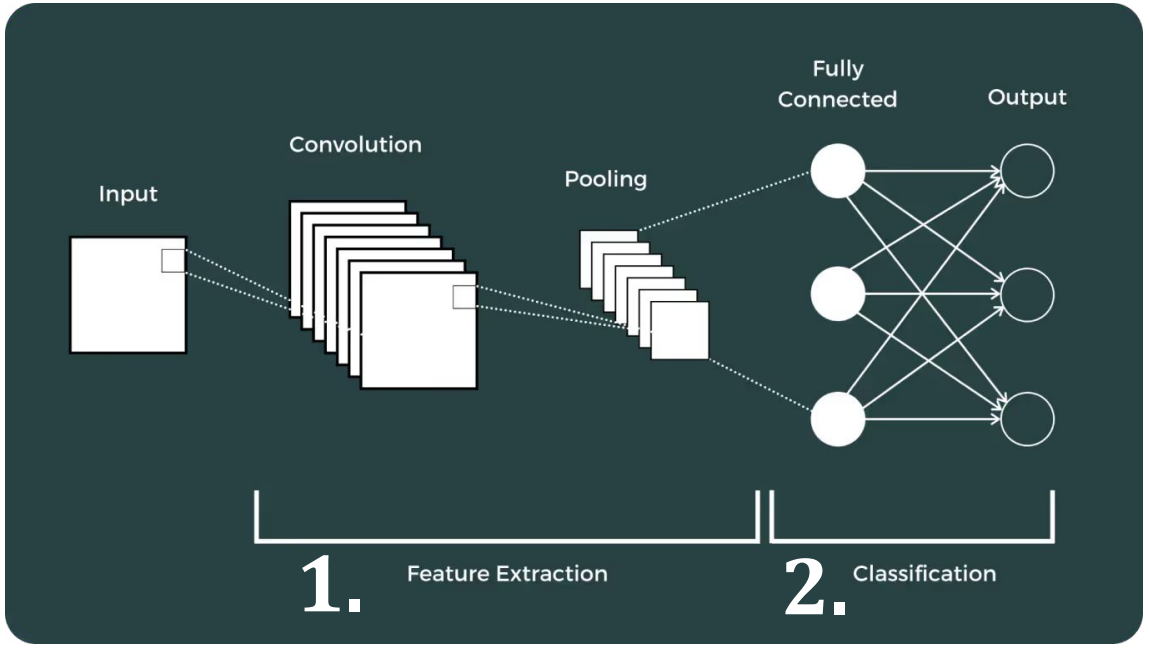
(One Stage Detectors)
Single Shot Object Detectors

- YOLO (**Y**ou **O**nly **L**ook **O**nce)
- SSD (Single Shot MultiBox Detector)

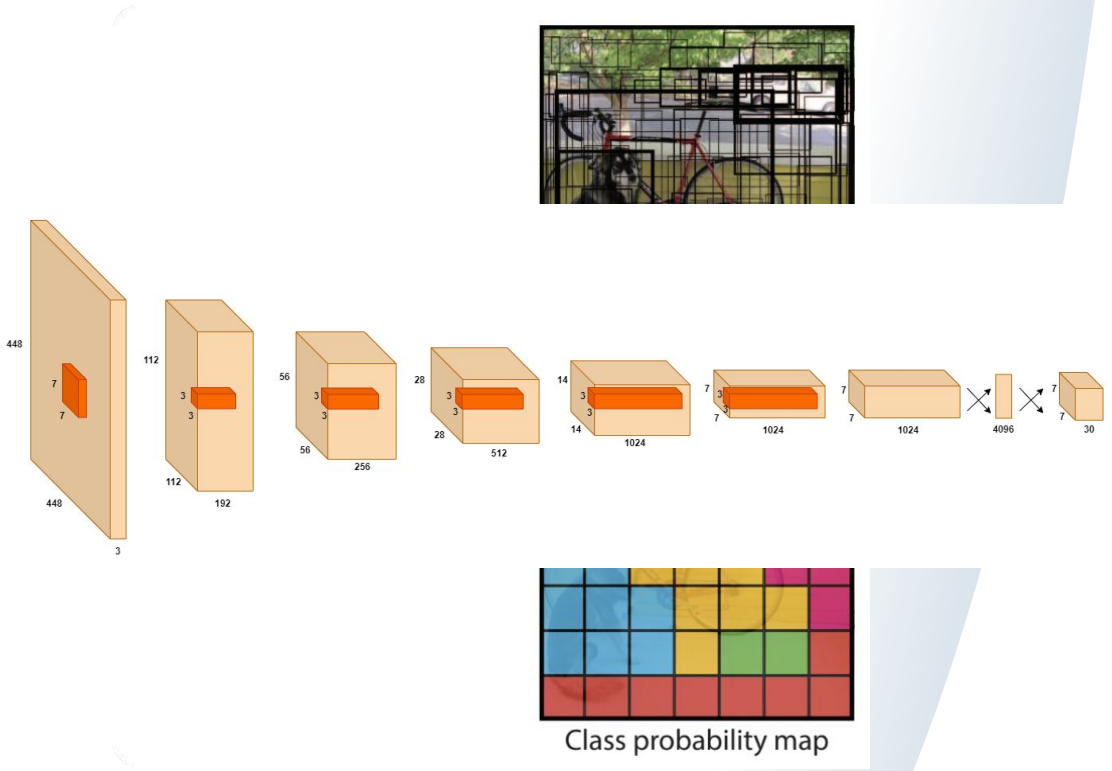
NOAA FISHERIES

Common Types of Object Detectors

Region-Based Object Detectors
(Two or Multi-Stage Detectors)



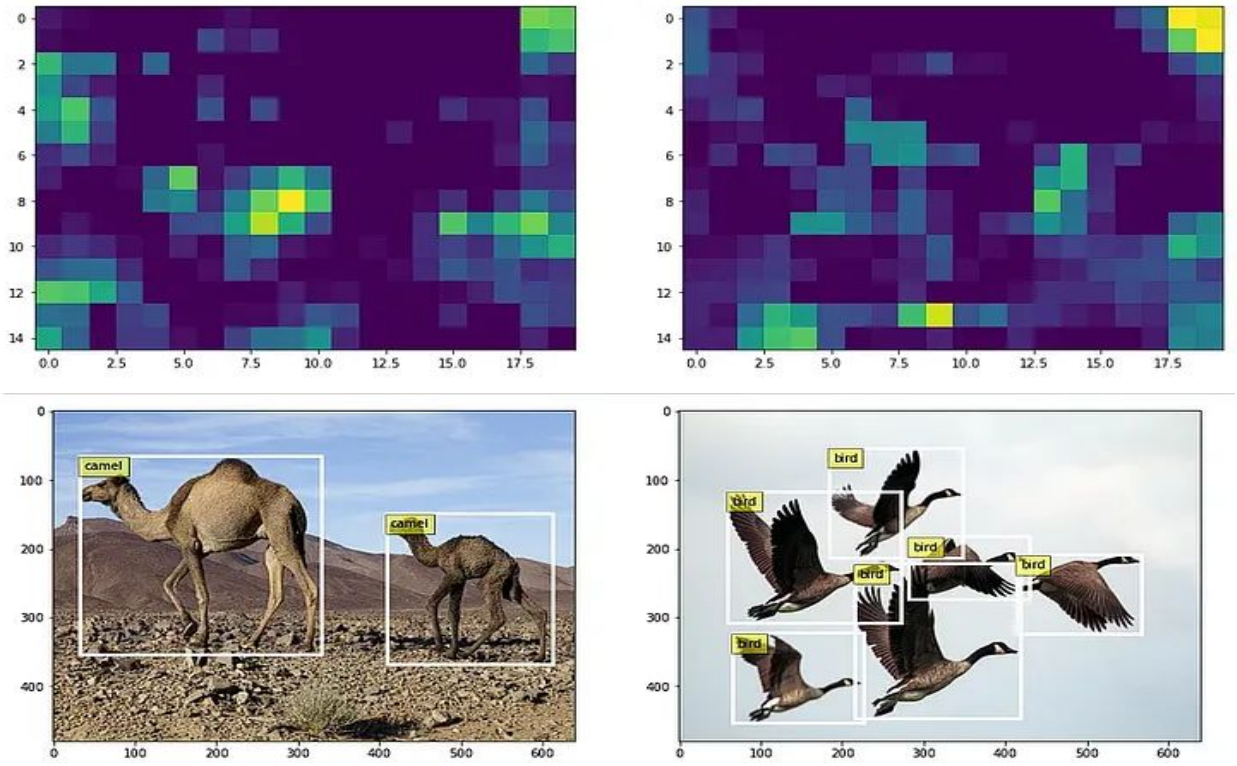
Single Shot Object Detectors
(One Stage Detectors)



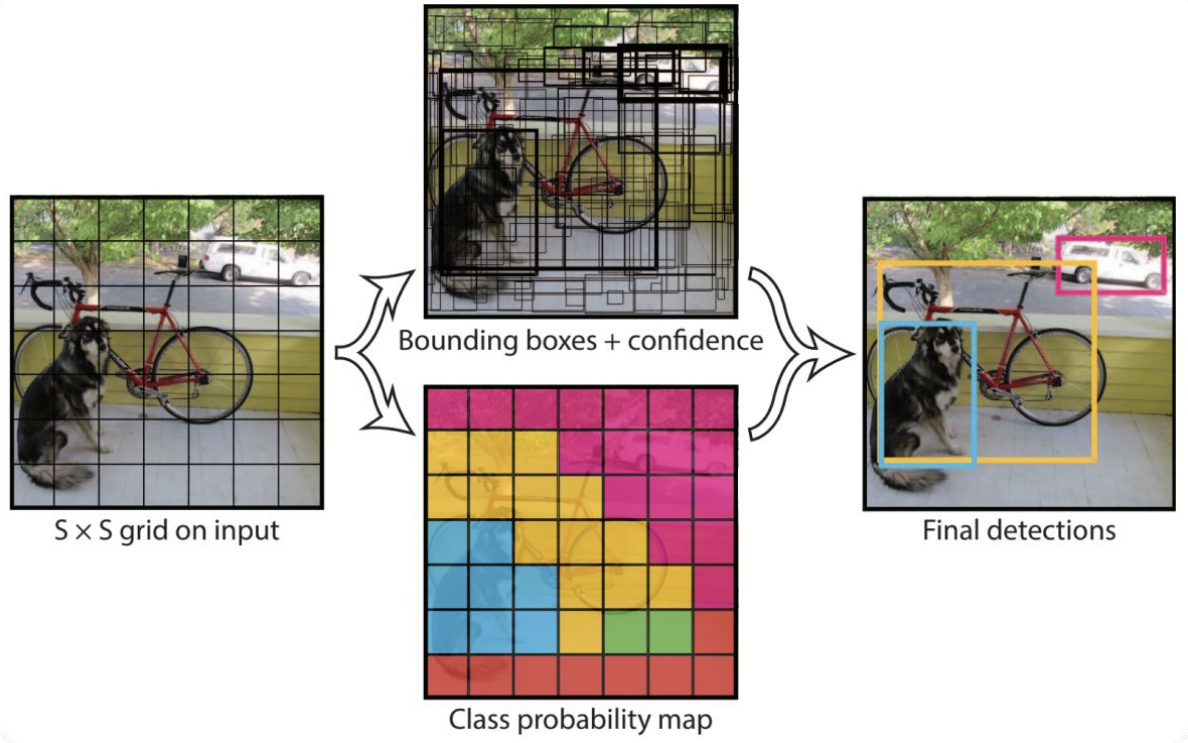
NOAA FISHERIES

Common Types of Object Detectors

Region-Based Object Detectors
(Two or Multi-Stage Detectors)



Single Shot Object Detectors
(One Stage Detectors)



NOAA FISHERIES

How does training an Object Detection model work?

NOAA FISHERIES

Object Detection: Several Different Options

Training Method	Description	Use Case	Pros	Cons
From Scratch	Train from random initialization on new data	Custom models, large datasets	Full control	Data and compute-intensive
Fine-tuning	Adapt a foundational (pre-trained) model to a specific task	Domain-specific tasks	Efficient and high-performance	Risk of overfitting
Zero-shot Learning	Leverage pre-trained models without training	No labeled data available	No task-specific data needed	limited performance on edge-case/complex data
Transfer Learning	Use a model trained for a related task as a starting point	Related but different tasks	Speeds up training	Requires task/domain compatibility

NOAA FISHERIES

What in an Object Detection Model?

NOAA FISHERIES

Object Detection Model

The model is like a blueprint.

- It's a set of instructions that a computer can follow to identify objects in an image or video.

Model Weights:

- Model weights are the stored patterns(rules) the model has learned during training.

Common Model files

- .pt = model architecture & model weights.
- .pth = just model weights

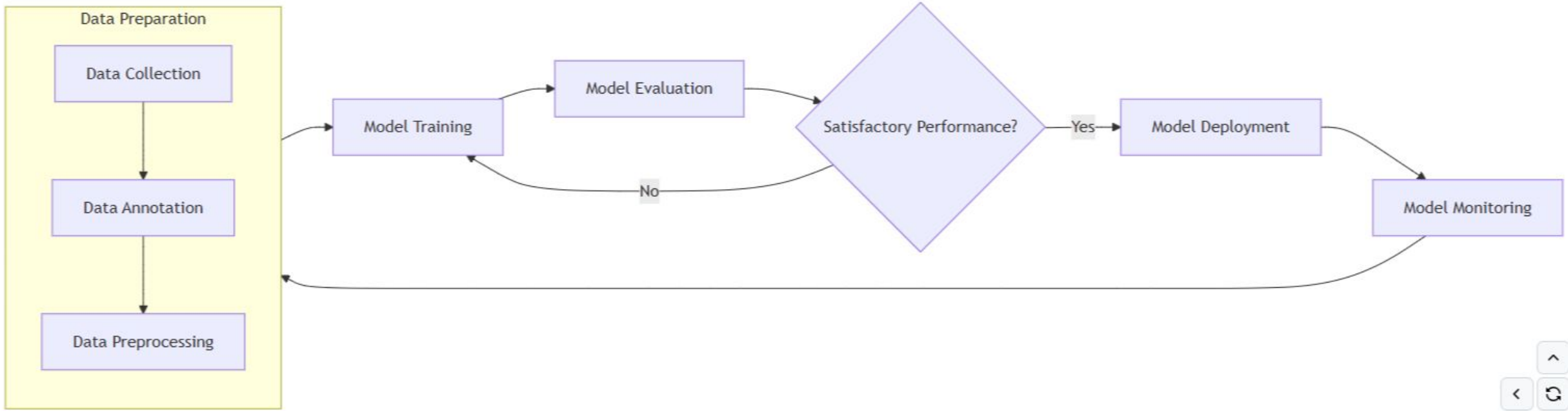
NOAA FISHERIES

How to get started?

NOAA FISHERIES

Getting a Project Started

- Create Project Plan Summary
  - Define Scope
  - Create a class list



80% Data Management
20% Model Building

"In machine learning, 80% of your time is spent preparing and cleaning the data, and only 20% of the time is spent on the actual model building."

- Andrew Ng (former Stanford AI Director/Professor)

NOAA FISHERIES

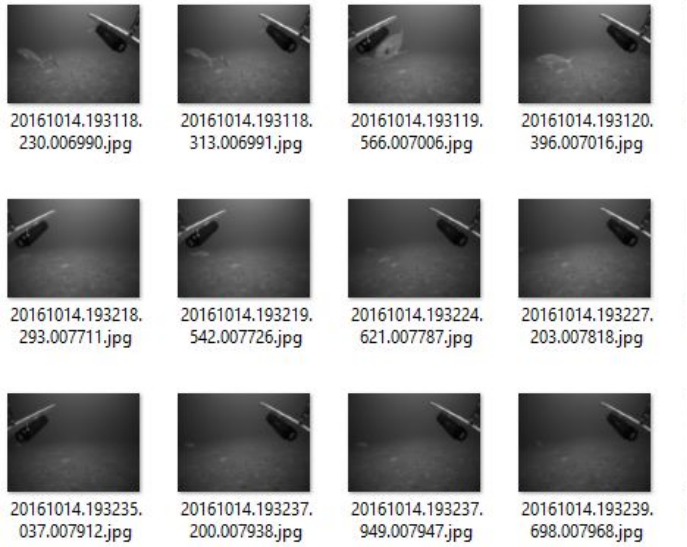
AI Ready Dataset:

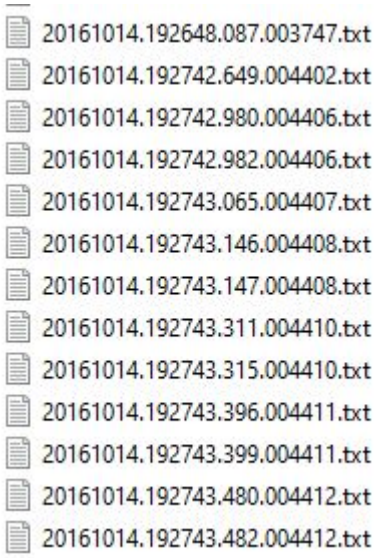
What Does it Look Like?

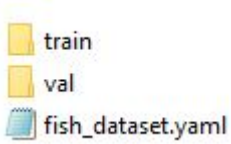
NOAA FISHERIES

AI-Ready Training Dataset

1. Images + 2. Annotations (labels) = Training Dataset



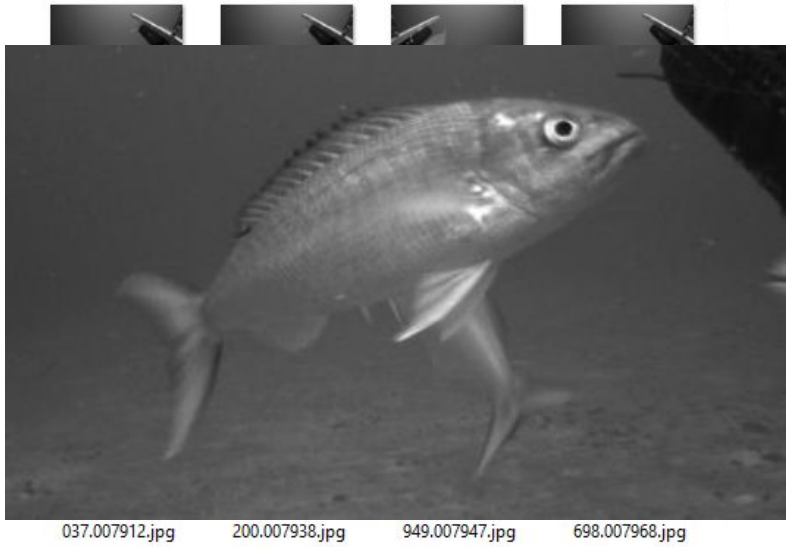


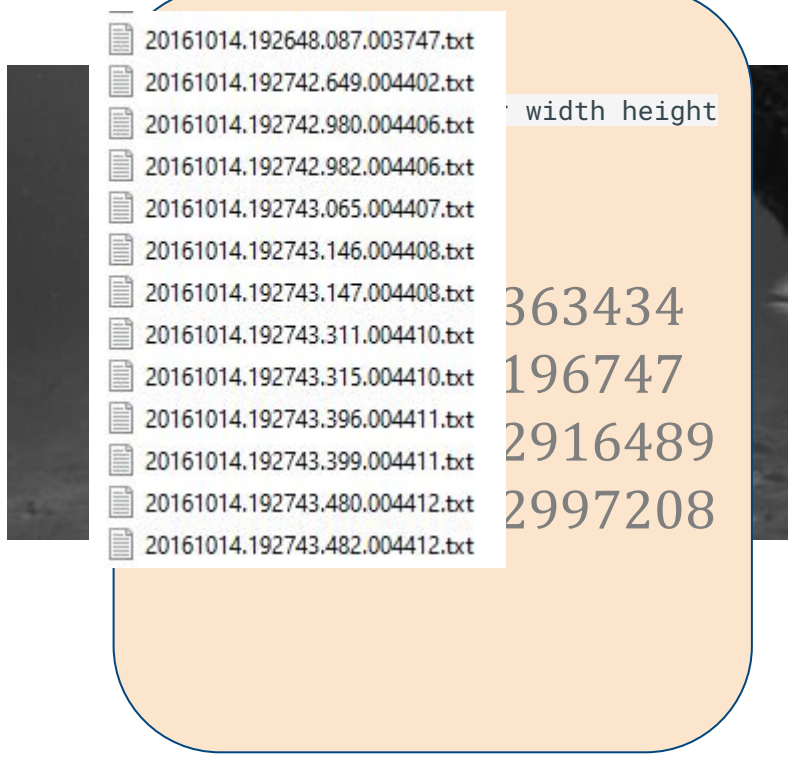


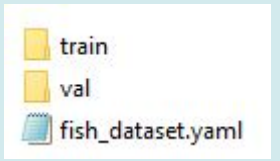
NOAA FISHERIES

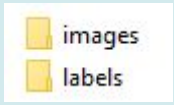
AI-Ready Training Dataset

1. Images + 2. Annotations (labels) = Training Dataset









#fish_dataset.yaml

train: images/train
val: images/val

Classes
names:
0: fish
1: coral
2: sand

NOAA FISHERIES

Is your Data AI Ready?
Check out the ESIP AI-Ready Data Checklist

Link:
https://github.com/ESIPFed/data-readiness/blob/main/checklist-published/ai-ready-data-checklist-v.1.0.md

About the checklist

The checklist is developed using the 2019 draft readiness matrix developed by the Office of Science and Technology Policy Subcommittee on Open Science as a basis. The checklist has been improved based on further research and user feedback. Definitions for some concepts are listed at the end of this document. This checklist is developed through a collaboration of ESIP Data Readiness Cluster members include representatives from NOAA, NASA, USGS, and other organizations. The checklist will be updated periodically to reflect community feedback.

How to use the checklist

Ideally for AI-ready assessment, a dataset should be defined as the minimum measurable bundle (i.e., a physical parameter/variable of observational datasets or model simulations). The assessment at this scale will enable better integration of data from different sources for research and development. However, it can be an intensive process for manual assessment without automation. Therefore, we recommend current assessments be done on the data file level. If the dataset has different versions, the checklist should be applied to each dataset type (e.g. raw, derived).

How to cite this checklist:

> ESIP Data Readiness Cluster (2023): Checklist to Examine AI-readiness for Open Environmental Datasets v.1.0. ESIP. Online resource. https://doi.org/10.6084/m9.figshare.19983722.v1

History

Version: 1.0;
Last updated: June 21, 2023.

General Information

- Link to the dataset landing page (*questions below could be automatically filled from the landing page in the future*)
  - Name of the dataset
  - The current version of the dataset
  - Point of contact for the dataset
  - When was the dataset originally published?
- Is this raw data or a derived/processed data product? *Raw/Derived*
- Is this observational data, simulation/model output, or synthetic data? *Observed/Modeled/Synthetic*
- Is the data single-source or aggregated from several sources? *Single-source/Aggregated*

Data Quality

- Questions on timeliness:
  - Will the dataset be updated? *Yes, it will be updated. / No, it will not be updated*
  - If the data will be updated, how often will it be updated? [choose from]
    - Updated when new data are added.
      - Choose the update frequency that best describes the dataset: *near-real-time (irregularly) / hourly or sub-hourly / daily or sub-daily / weekly / monthly / yearly / longer than a year*
      - Will there be different stages of the update (e.g., updated with preliminary data first and replaced by a later update of the full record)? *Yes/No/Not applicable*

ESIP Data Readiness Cluster

This is the repository for the Data Readiness Cluster to maintain the AI-ready data checklist. The cluster is a community-driven group focusing on developing recommendations and community standards on AI-ready open environmental data. Although the work currently focuses on environmental data, the product could be extended to data from other domains.

The goal of AI-readiness assessment:

- For data producers/providers, the purpose is to understand to what extent the data being assessed meets the common research data management practices and principles that are relevant to AI/ML application development. The assessment result can be used to justify targeted improvements for the dataset when resources become available.
- For projects generating new datasets, the purpose of the AI-readiness checklist can be used to guide the development of the dataset. For example:
  - What documentation do you want to provide accompanying the dataset?
  - Do you have a proper data quality assessment that will make the development of downstream AI/ML applications efficient?

How to use the checklist for assessment:

The current version of the checklist is available here (last updated 2023-12-20). The checklist will be maintained and updated by the community.

To assist with the assessment, we have created a fillable Google sheet template. You can make a copy of the Google sheet for your assessment. Each dataset should be assessed separately as the checklist is designed for individual datasets. More effort is ongoing to address the need for linked datasets.

If you are in the early stages of developing AI/ML applications with open environmental datasets. We encourage you to assess the input data used for your applications. Although you may not have the ability to change other people's datasets, this will help you document the effort spent on preparing the dataset for your development.

NOAA FISHERIES

What's available to help the process?

NOAA FISHERIES

Platforms vs Libraries vs Tools

- What to know:
  - are they easy to install and use?
  - Free/open source?
  - Maintained/Support/documentation?
- What's important to ask:
  - Can it export the annotation data out in a common format?
  - Can you export the model?

Type	AI/ML Libraries	Tools	Platforms/Software
Example	Pytorch, Tensorflow, Ultralytics	Label Studio, CVAT, TagLab	Label Studio, CVAT, Roboflow, Coralnet, Viame, CleamML
Use	for building, training, deploying, developing and more	Data Prep, Annotation, labeling of datasets (optional ML backends)	Integrate multiple functionalities

NOAA FISHERIES

Annotation Tools

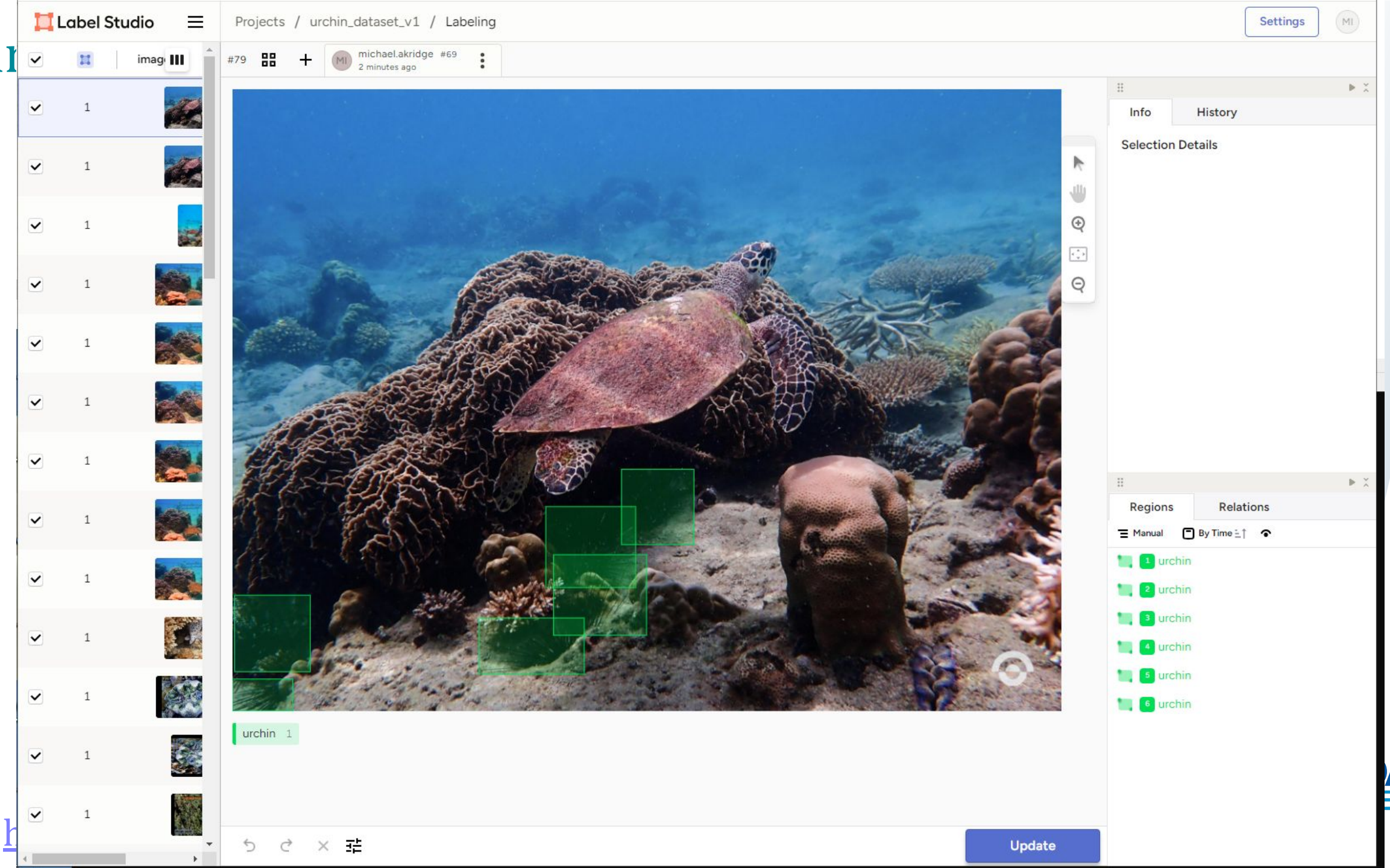
Image and Video Annotation

Name	Image	Video	Free	Open Source	AI Assisted Annotation	Notes
Label Studio	✅	✅	✅	✅	✅	Multi-type data labeling and annotation tool with standardized output format.
CVAT	✅	✅	✅	✅	✅	Open-source tool for video and image annotation.
VIAME	✅	✅	✅	✅	✅	See documentation.
Scalabel	✅	✅	✅	✅		Modern video and image annotation platform.
Tator	✅	✅	✅	✅		See Tutorials and Source Code.
BIIGLE	✅	✅	✅	✅		Check GitHub.
FishID	✅	✅	✅	SOON		Email to request access: fishidglow@outlook.com.
TagLab	✅	❌	✅	✅	✅	
~~VoTT (Archived)~~	✅	❌	✅	✅	❌	Officially archived, last update in 2020.
SEAMS	✅	❌	✅	✅	❌	SEafloor Annotation and Mapping Support Open-source tool for video and image annotation.

- https://github.com/MichaelAkridge-NOAA/awesome-ocean-ai

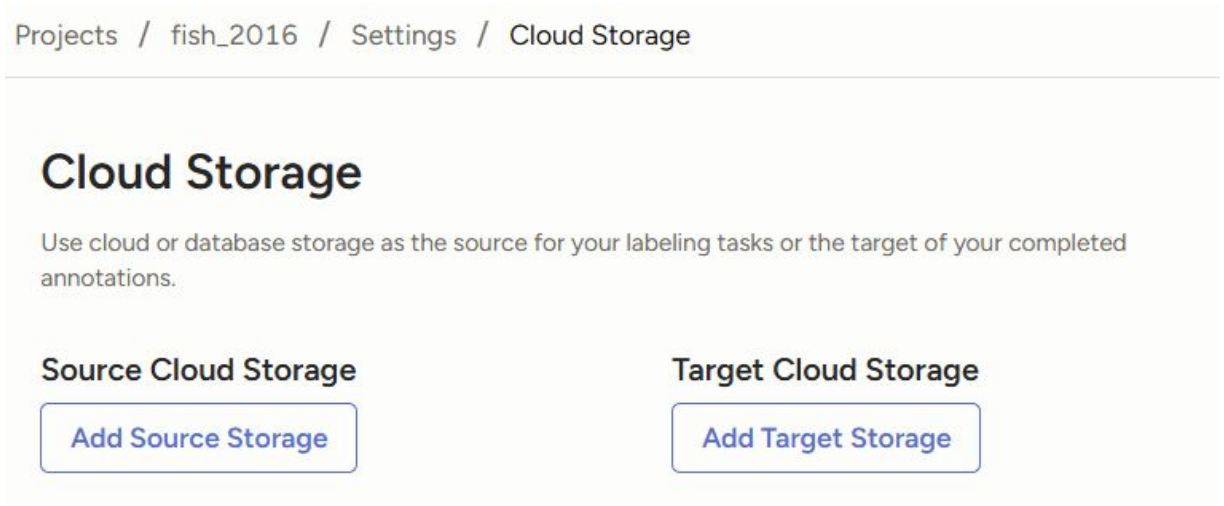
NOAA FISHERIES

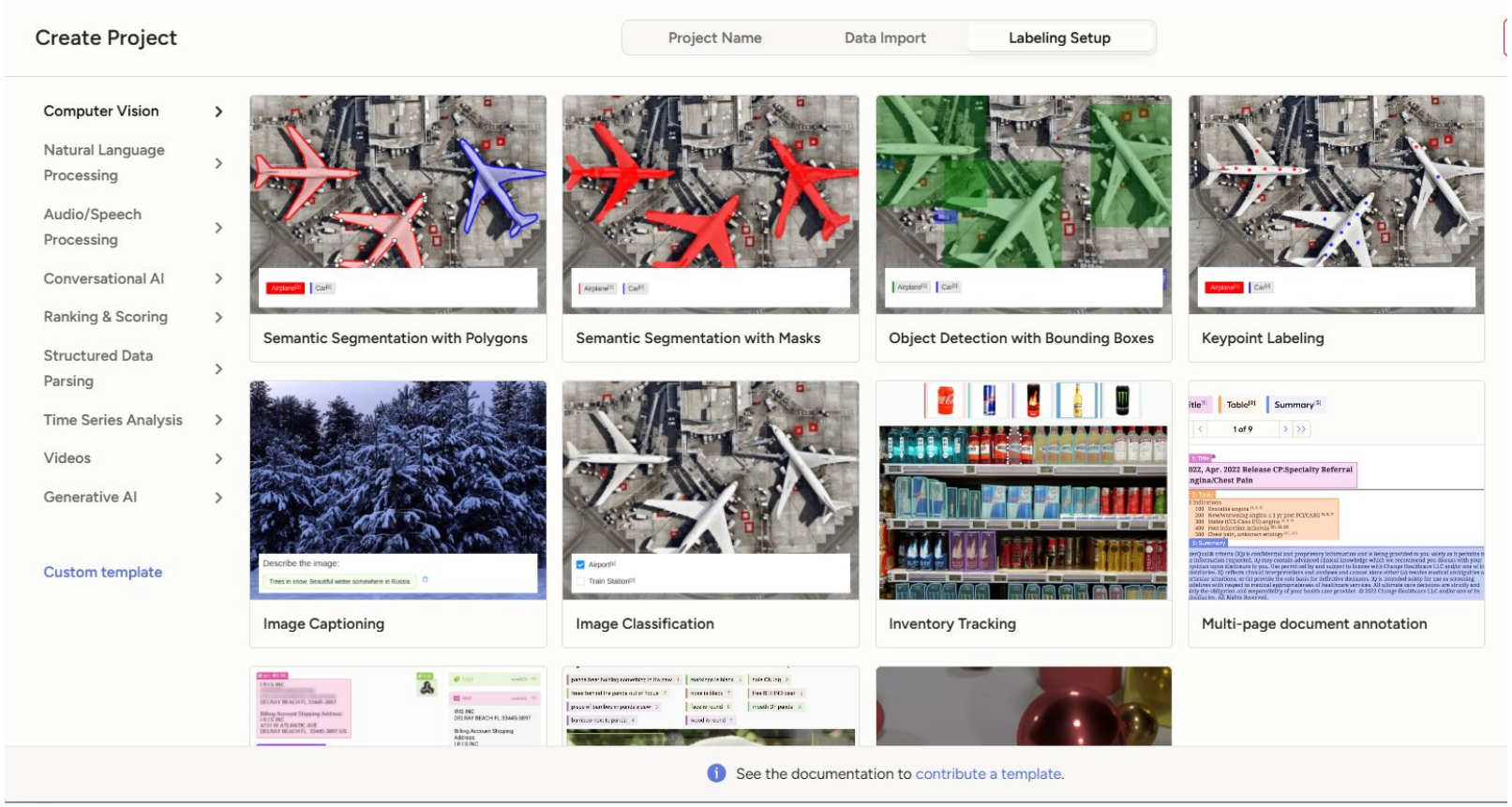
Ann

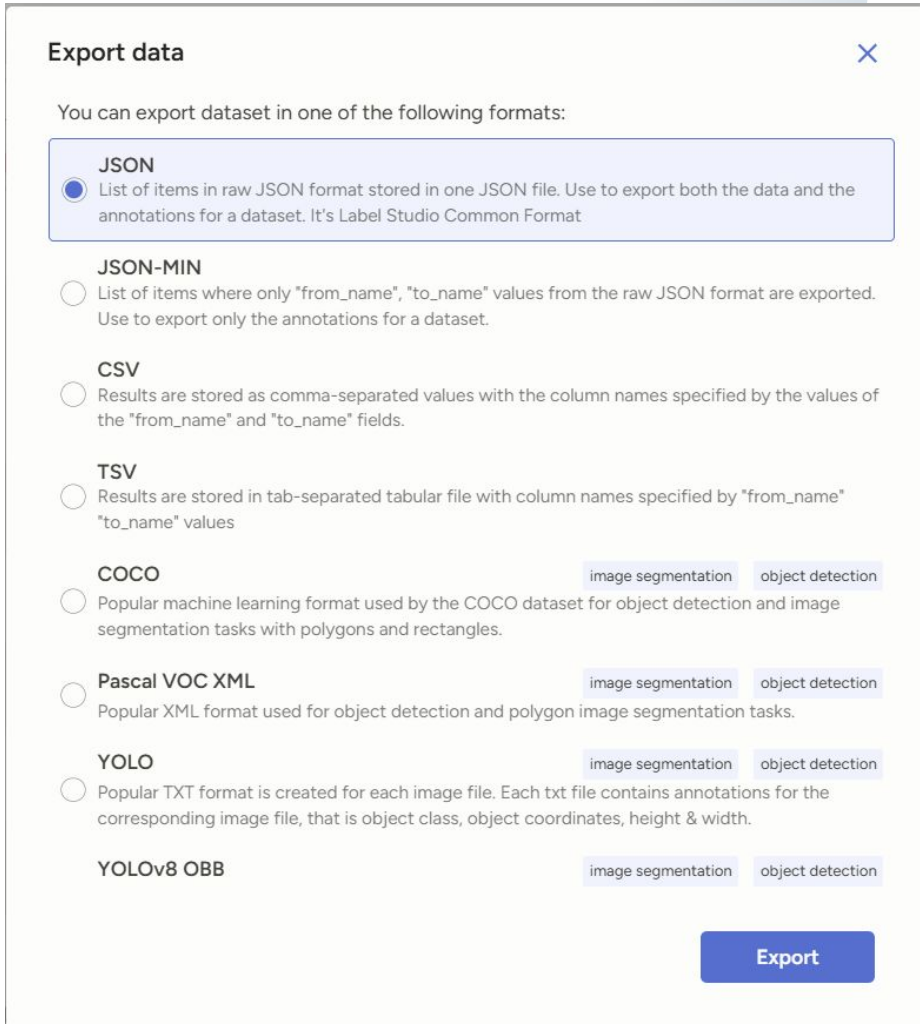

Label Studio
Projects / urchin_dataset_v1 / Labeling
Settings
#79
michael.akridge #69
2 minutes ago
Info
History
Selection Details
Regions
Relations
Manual
By Time
1 urchin
2 urchin
3 urchin
4 urchin
5 urchin
6 urchin
urchin 1
Update

- h

- Browser/web Based
- Simple Install
- Runs on all systems







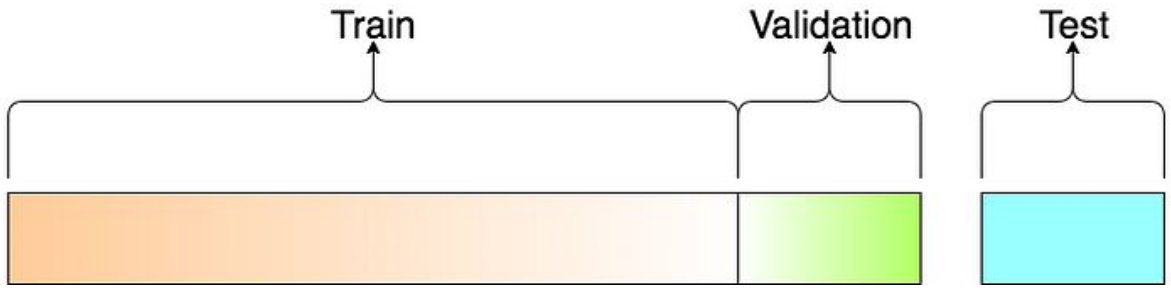
Dataset Best Practices

NOAA FISHERIES

Dataset Best Practices: For training an Object Detection Model

Standard Dataset Split

- **Training Set (70-80%):** Used to train your model.
- **Validation Set (15-20%):** Used to tune and validate the model during training
  - **Optional - Test Set (15-20%):** Used to evaluate the model performance on unseen data after training

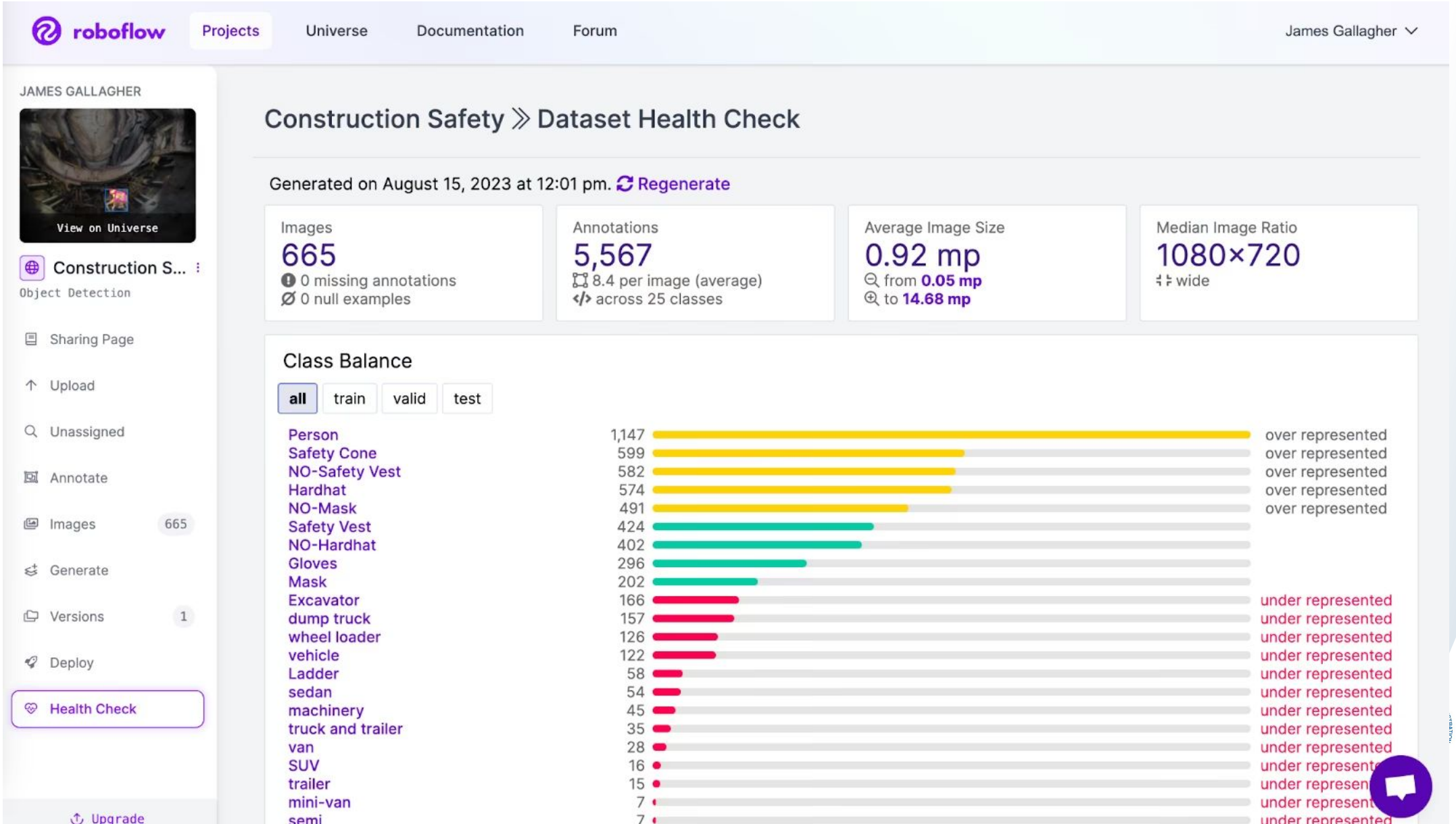


Best Practices

- **Images per class: ≥ 1500 images** per class recommended
- **Instances per class: ≥ 10000 instances** (labeled objects) per class recommended
- **Image variety**. Representative of environment.
- **Label accuracy**.
  - Labels must closely enclose each object.
  - No space should exist between an object and it's bounding box.
  - No objects should be missing a label.
- **Optional - Background images (1-10%).** Background images are images with no objects that are added to a dataset to reduce False Positives (FP).
- Have unique images in each split

NOAA FISHERIES

Dataset Management & Balance



NOAA FISHERIES

Lessons Learned

- **Data augmentation** techniques can help models generalize better
  - Many common data augmentation techniques(scale, shear, rotate, perspective, flip, mosaic, and mixup) are built into many training libraries (Link: YOLO Data Augmentation Explained)

Image Scale & Mosaic Augmentation Example

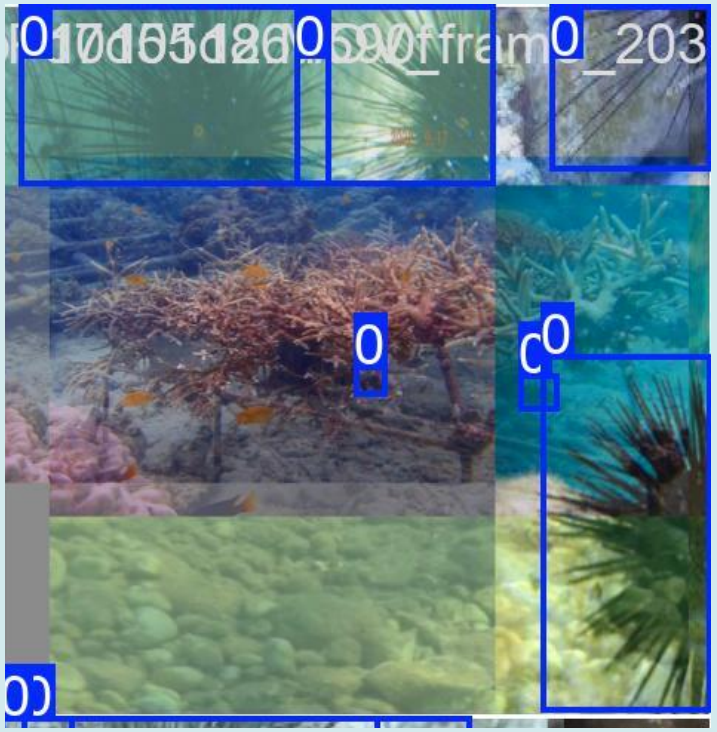
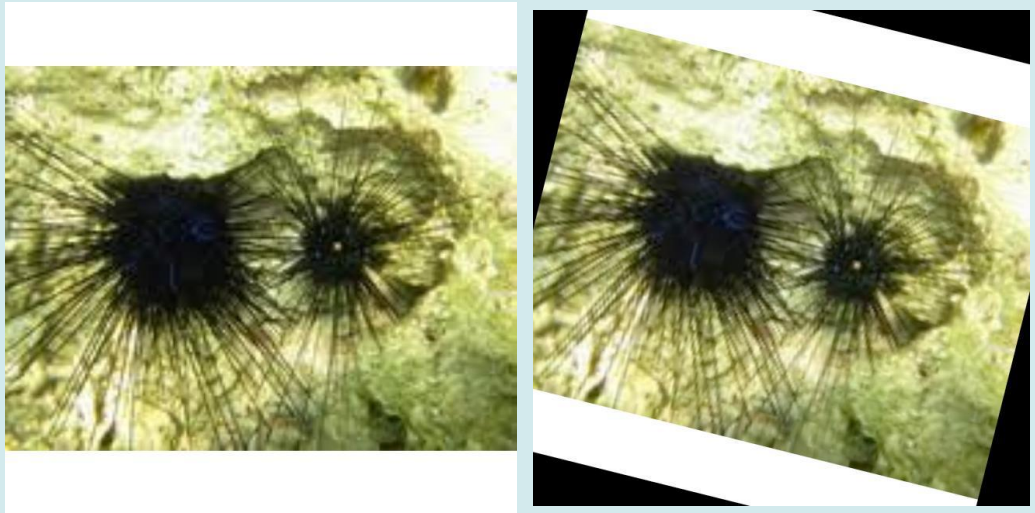
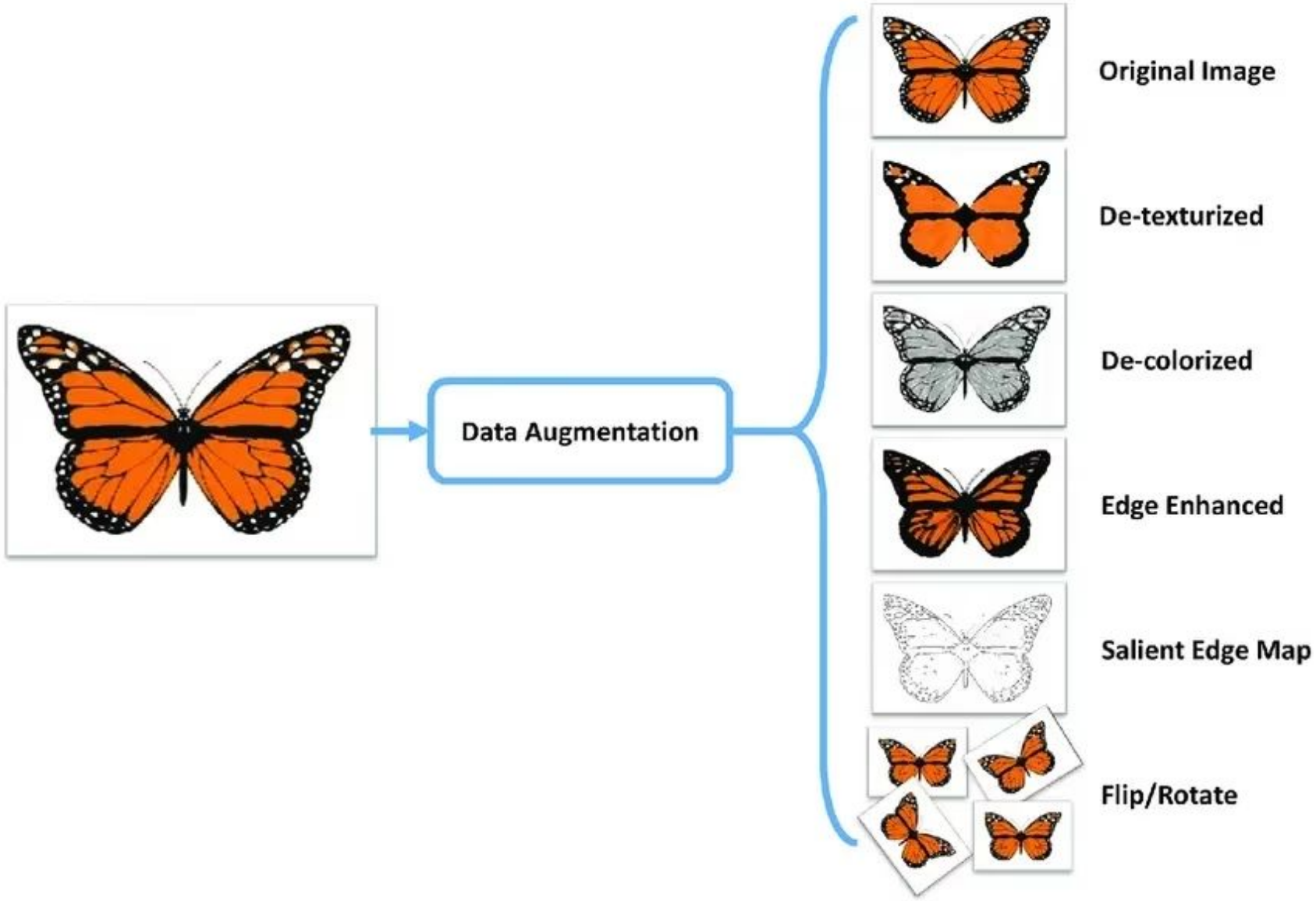


Image Angle/Degree rotation Augmentation Example





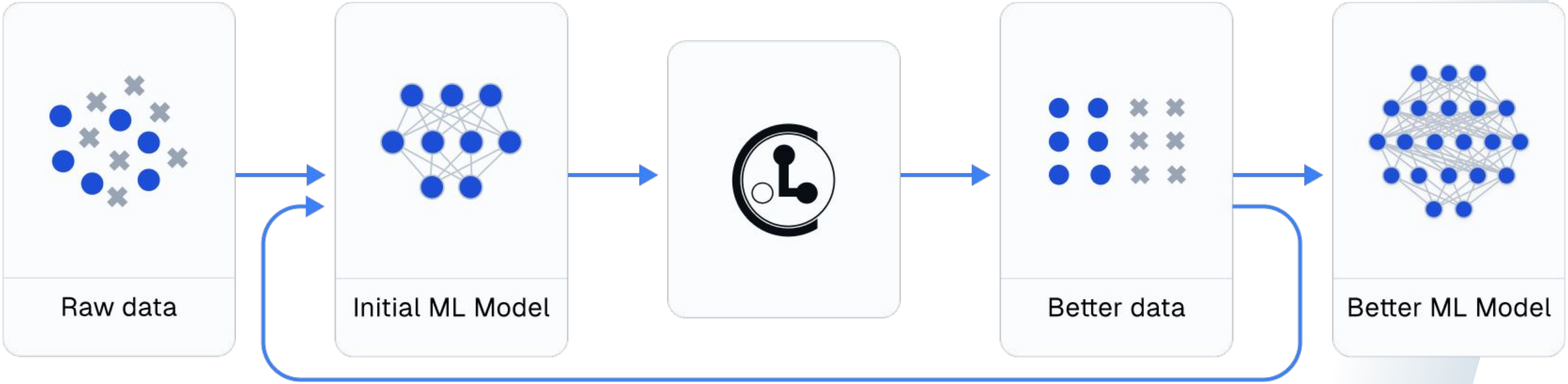
Turns 500 image dataset into a 5000 image dataset

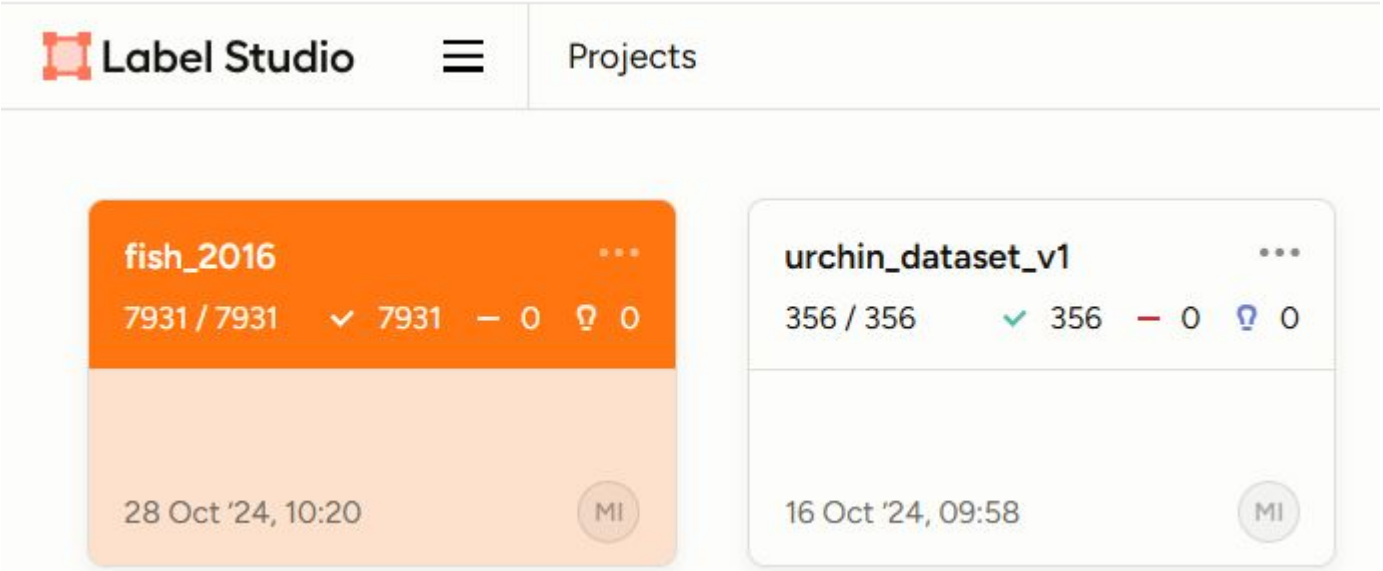
NOAA FISHERIES

After Data: When to Start Training a Model?

NOAA FISHERIES

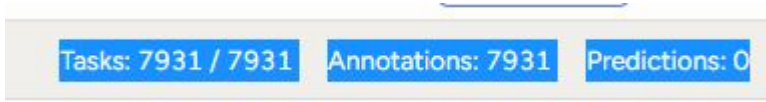
Data-Centric AI Model Flow: Start Right Now





Have the Machine do the Hard work:

- Pre-annotate/autolabel data
- Interactive labeling
- Model evaluation and fine-tuning



NOAA FISHERIES

The YOLO (Ultralytics) Framework

YOLO(You Only Look Once)

Object Detection: YOLO(You Only Look Once)

YOLO: A Brief History

YOLO (You Only Look Once), a popular object detection and image segmentation model, was developed by Joseph Redmon and Ali Farhadi at the University of Washington. Launched in 2015, YOLO quickly gained popularity for its high speed and accuracy.

- YOLOv2, released in 2016, improved the original model by incorporating batch normalization, anchor boxes, and dimension clusters.
- YOLOv3, launched in 2018, further enhanced the model's performance using a more efficient backbone network, multiple anchors and spatial pyramid pooling.
- YOLOv4 was released in 2020, introducing innovations like Mosaic data augmentation, a new anchor-free detection head, and a new loss function.
- YOLOv5 further improved the model's performance and added new features such as hyperparameter optimization, integrated experiment tracking and automatic export to popular export formats.
- YOLOv6 was open-sourced by Meituan in 2022 and is in use in many of the company's autonomous delivery robots.
- YOLOv7 added additional tasks such as pose estimation on the COCO keypoints dataset.
- YOLOv8 released in 2023 by Ultralytics. YOLOv8 introduced new features and improvements for enhanced performance, flexibility, and efficiency, supporting a full range of vision AI tasks,
- YOLOv9 introduces innovative methods like Programmable Gradient Information (PGI) and the Generalized Efficient Layer Aggregation Network (GELAN).
- YOLOv10 is created by researchers from Tsinghua University using the Ultralytics Python package. This version provides real-time object detection advancements by introducing an End-to-End head that eliminates Non-Maximum Suppression (NMS) requirements.
- **YOLO11** 🚀 **NEW**: Ultralytics' latest YOLO models delivering state-of-the-art (SOTA) performance across multiple tasks, including detection, segmentation, pose estimation, tracking, and classification, leverage capabilities across diverse AI applications and domains.

2015

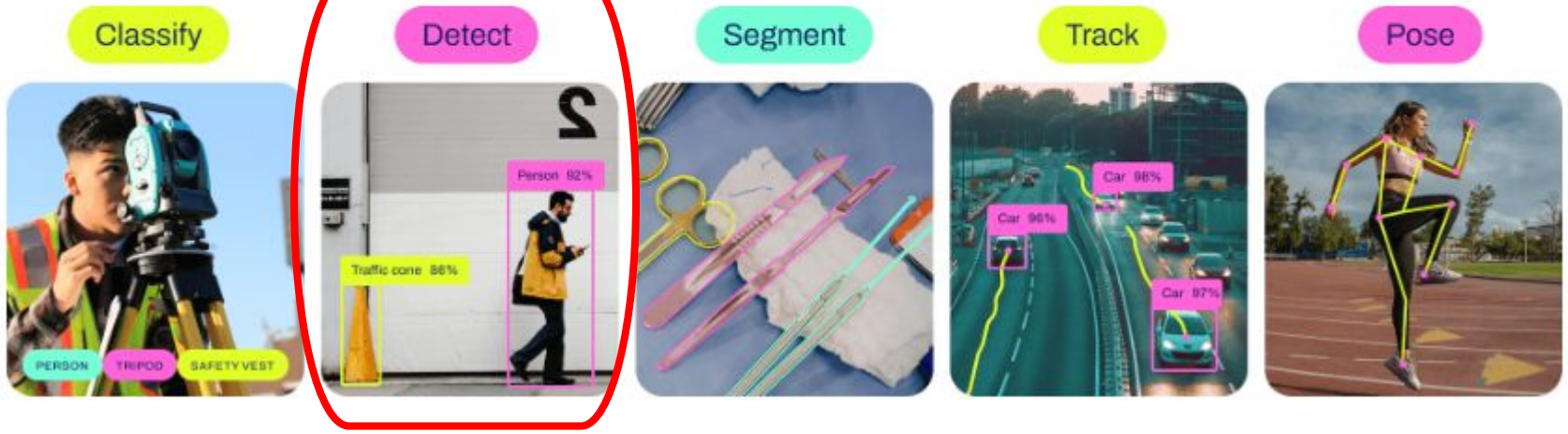
2025

NOAA FISHERIES

YOLO(You Only Look Once) -

Models

YOLO11 Detect, Segment and Pose models pretrained on the COCO dataset are available here, as well as YOLO11 Classify models pretrained on the ImageNet dataset. Track mode is available for all Detect, Segment and Pose models.



- Open-source (python codebase) and easy to use
- Extensive documentation , tutorials, with a large and active community(>60 million)
- Shared encoders between models (part of the CNN that learns relevant features). The encoder of one model can be used as starting point for others

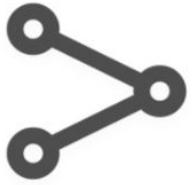
ultralytics

NOAA FISHERIES

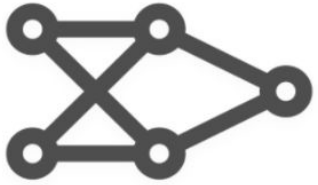
Base Foundation Model Selection

Model	size (pixels)	mAP^{val} 50-95	Speed CPU ONNX (ms)	Speed T4 TensorRT10 (ms)	params (M)	FLOPs (B)
YOLO11n	640	39.5	56.1 ± 0.8	1.5 ± 0.0	2.6	6.5
YOLO11s	640	47.0	90.0 ± 1.2	2.5 ± 0.0	9.4	21.5
YOLO11m	640	51.5	183.2 ± 2.0	4.7 ± 0.1	20.1	68.0
YOLO11l	640	53.4	238.6 ± 1.4	6.2 ± 0.1	25.3	86.9
YOLO11x	640	54.7	462.8 ± 6.7	11.3 ± 0.2	56.9	194.9

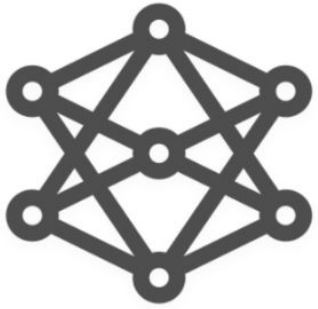
- speed vs accuracy tradeoffs
- training time
- Resources for Deployment



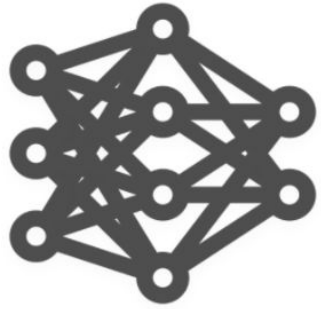
Small



Medium



Large



XLarge

YOLO Python Script to Train Model

```
# Load Library
import YOLO

# Load a foundational model
model = YOLO("yolo11n.pt")

# Train Model
model.train(data="fish_dataset.yaml", # path to dataset
    epochs=100,  # number of training epochs
    imgsz=640,  # training image size
    device=cpu,  # device to run on - cpu, gpu(0,1)
    …)
```

NOAA FISHERIES

note: When training a model, an epoch refers to one complete pass through the entire training dataset.

What is the right number of Training Passes (Epochs) to Use?

It depends. **Varies based on model, dataset size and task.**

Larger datasets require more while smaller datasets will need less.

Good baselines:

- 1-10 - test setup
- 50 - mockup model
- 100 min regular run
- 300 recommended starting point

NOAA FISHERIES

Do I need a Supercomputer to get started?

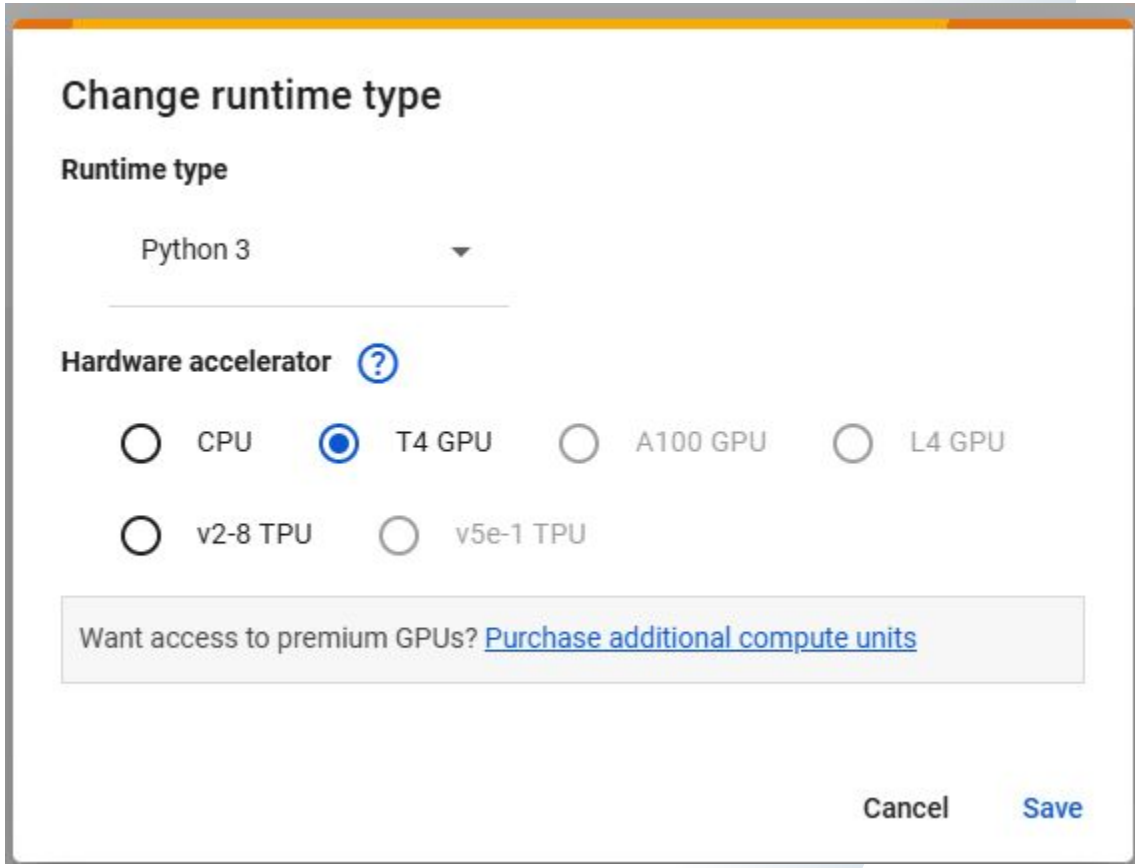
NOAA FISHERIES

Nope. Just a Web browser to Get Started.

Google Colaboratory

Colab is a hosted Jupyter Notebook service that requires no setup to use and provides free access to computing resources, including GPUs and TPUs. Colab is especially well suited to machine learning, data science, and education.

Google colab



NOAA FISHERIES

20241001 - YOLO11_TRAIN.ipynb

File Edit View Insert Runtime Tools Help Last edited on Dec 6, 2024

+ Code + Text

```
[ ]  import os
     import torch
     from ultralytics import YOLO

     device = 'cuda' if torch.cuda.is_available() else 'cpu'
     print(f"Using device: {device}")

     # Load the large model (YOLOv8x) that was fine-tuned for VME detection
     large_model = YOLO("/content/best.pt")  # Replace with your model path
     large_model = large_model.to(device)  # Move the model to GPU (CUDA) if available

     # Step 4: Create folders for the new dataset (images and labels)
     base_path = "/content/fish_dataset"

     # Step 11: Train the YOLOv11n model using the generated dataset
     small_model = YOLO("yolo11n.pt")  # Load the smaller YOLOv8n model

     # Train the model using the generated fish-only dataset
     small_model.train(data="/content/fish_dataset.yaml", epochs=50, imgsz=416, batch=16, lr0=0.001)

     print("Training complete!")

     # model save
     small_model.save("/content/yolov11n_fish_trained.pt")

     # model val stats
     metrics = small_model.val(data="/content/fish_dataset.yaml")  # This will evaluate precision, recall, and mAP
```

```
      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       1/50      1.45G      1.276      2.073       1.11          5       416: 100%|██████████| 93/93 [00:28<00:00,  3.25it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 12/12 [00:03<00:00,  3.06it/s]


      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       2/50      1.36G      1.194      1.457      1.083         11       416: 100%|██████████| 93/93 [00:26<00:00,  3.55it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 12/12 [00:02<00:00,  5.04it/s]
                   all        369        473      0.466      0.431      0.426      0.255
```

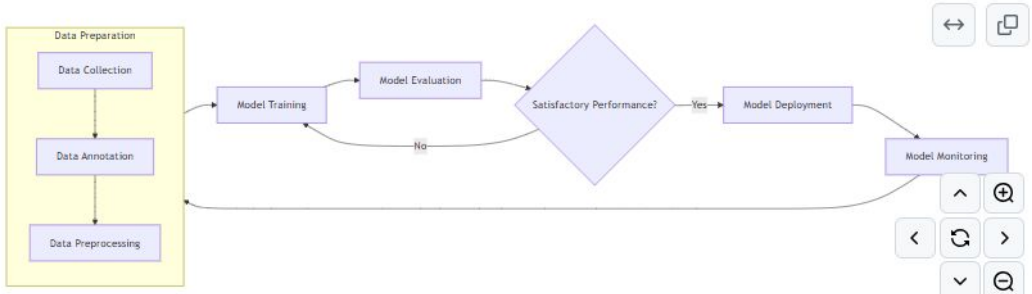
IOAA SHERIES

- https://github.com/MichaelAkridge-NOAA/open-science-ai-toolkit

Open Science AI Toolkit

Open source suite of tools, workflows, and processes designed to accelerate Open Science AI/ML development.

Overview



Installation

To install the entire toolkit:

```
pip install git+https://github.com/MichaelAkridge-NOAA/open-science-ai-toolkit.git
```

Features

The `noaa_ai_tools` subpackage provides functionalities like:

- **Dataset Preparation**: Easily prepare your datasets for training with functions for filtering and validating annotations.
  - **Filter Images with Labels**
  - **Create Labels for Background Images**
  - **Validate YOLO Format**
  - **Remap Class IDs**
  - **Split Dataset (train, val, test)**
- **Model Training**: Train and manage deep learning models, including support for YOLO models from Ultralytics(https://github.com/ultralytics/ultralytics).
- **Model Exporting**: Export trained models to various formats, including ONNX, TensorFlow Lite, Edge TPU, and NCNN.

```python
# Training hyperparameters
small_model.train(
    data=yaml_file_path,
    epochs=100,
    imgsz=640,
    batch=32,  # Adjust batch size based on GPU capacity
    lr0=0.001,  # Initial learning rate
    lrf=0.0001,  # Final learning rate (used for Cosine Annealing)
    optimizer='AdamW',  # Use AdamW optimizer for better performance
    device=device,
    save_period=10,  # Save model checkpoint every 10 epochs
    patience=10,  # Early stopping if no improvement after 10 epochs
    augment=True,  # Enable data augmentation
    mosaic=True,  # Use mosaic augmentation
    mixup=True,  # Use MixUp augmentation
    cos_lr=True,  # Cosine annealing learning rate
    project='training_logs',  # TensorBoard logging directory
)

print("Training complete!")

# Unfreeze all layers after the initial phase
for param in small_model.model.model.parameters():
    param.requires_grad = True

# Save the trained model
trained_model_path = os.path.join(base_path, "yolo11m_urchin_trainedv3.pt")
small_model.save(trained_model_path)
print(f"Trained model saved to {trained_model_path}")

# Save the model weights separately for further use
weights_path = os.path.join(base_path, "yolo11m_urchin_weightsv3.pth")
torch.save(small_model.model.state_dict(), weights_path)
print(f"Weights saved to {weights_path}")

# Evaluate model performance
```

Lessons Learned

NOAA FISHERIES

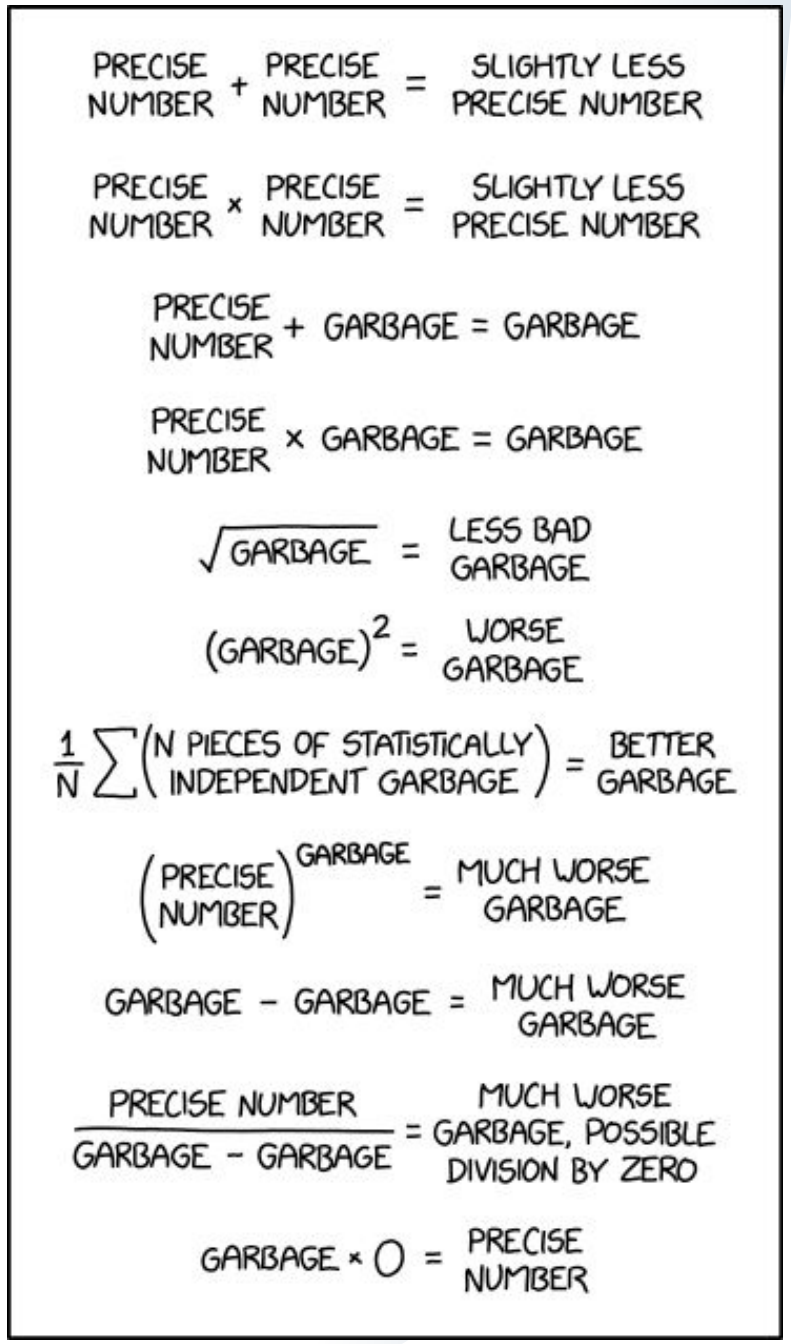
Lessons Learned

Garbage in, garbage out

> "the concept that flawed, biased or poor quality ("garbage") information or input produces a result or output of similar ("garbage") quality."

- still very relevant today
- The better your training dataset the better your model will be

GARBAGE MATH: HTTPS://XKCD.COM/2295/



Lessons Learned: How to Create a better model

More data ≠ Better Model

Better data = Better Model

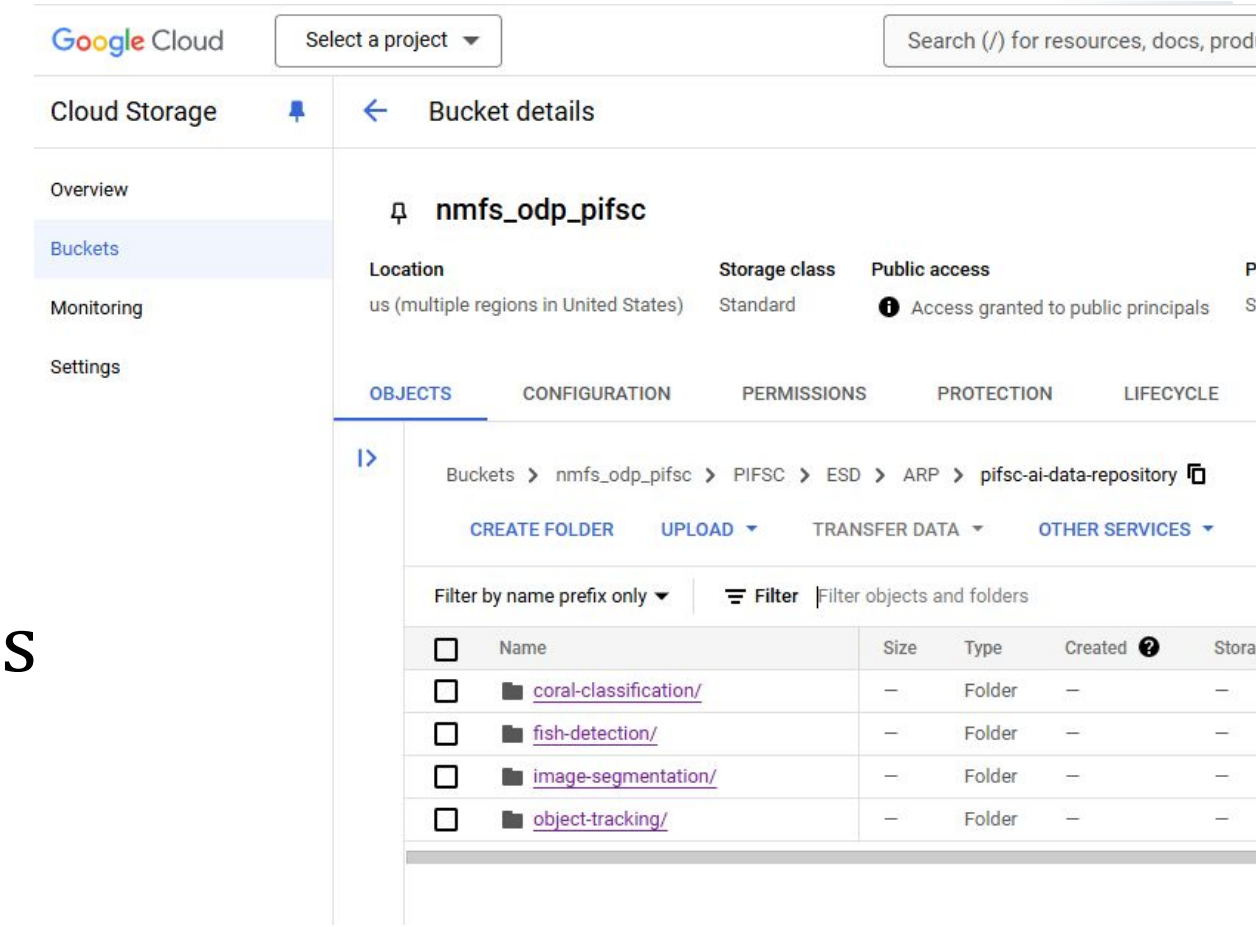
NOAA FISHERIES

Lessons Learned

Storage & Access of AI Ready Datasets is Lacking

- In general, AI ready datasets are very sparse and/or hard to access easily

- When making a dataset:
  - Choose a standard format like COCO (json)
  - Although, its very easy to convert from/to any format

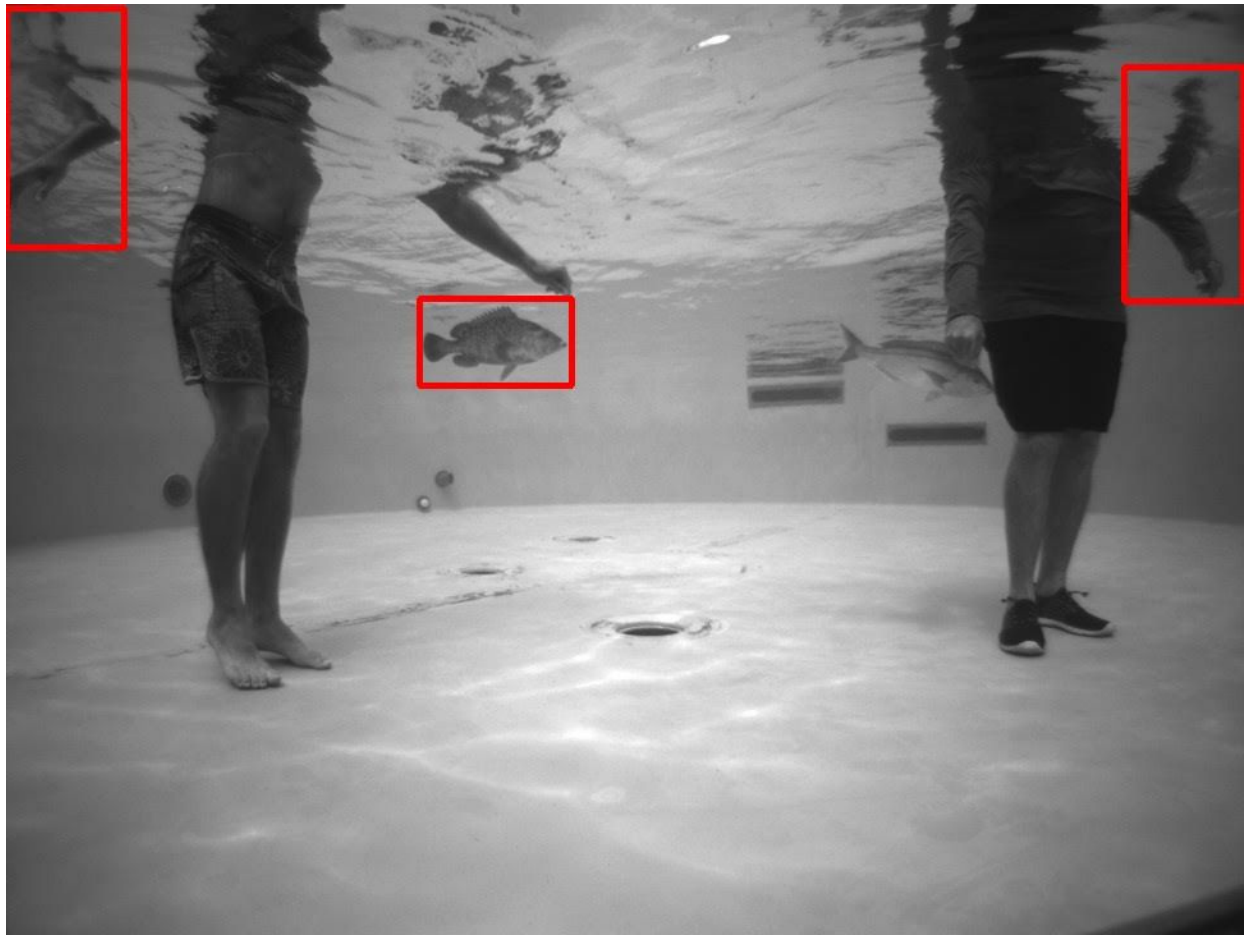


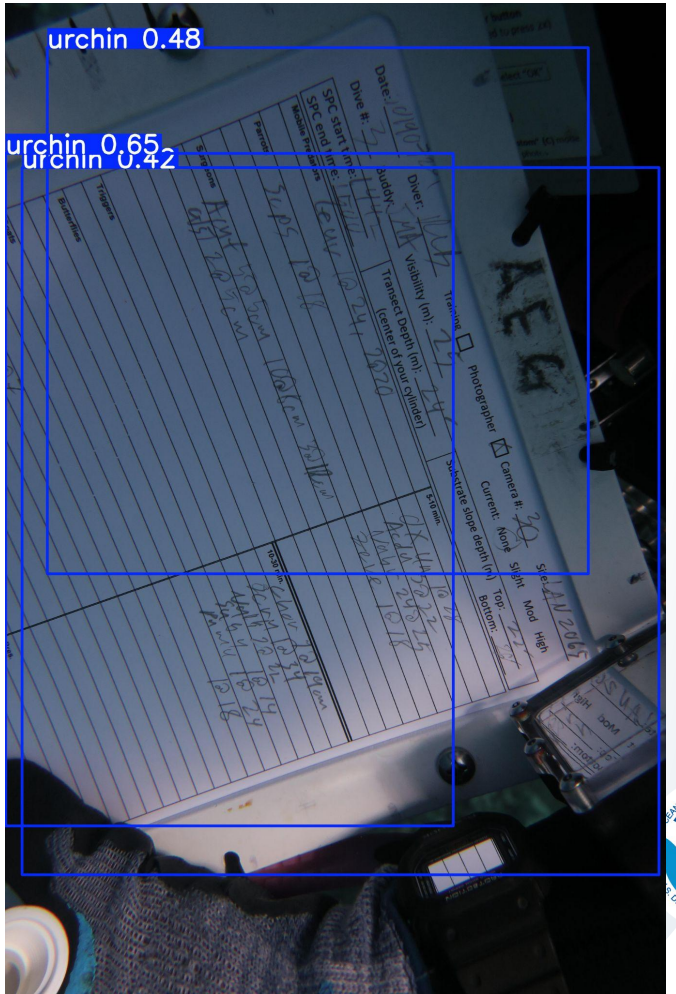
Cloud AI data repo in the works

NOAA FISHERIES

Lessons Learned

- **When making a Training Dataset:**
  - The model won't know what it hasn't seen

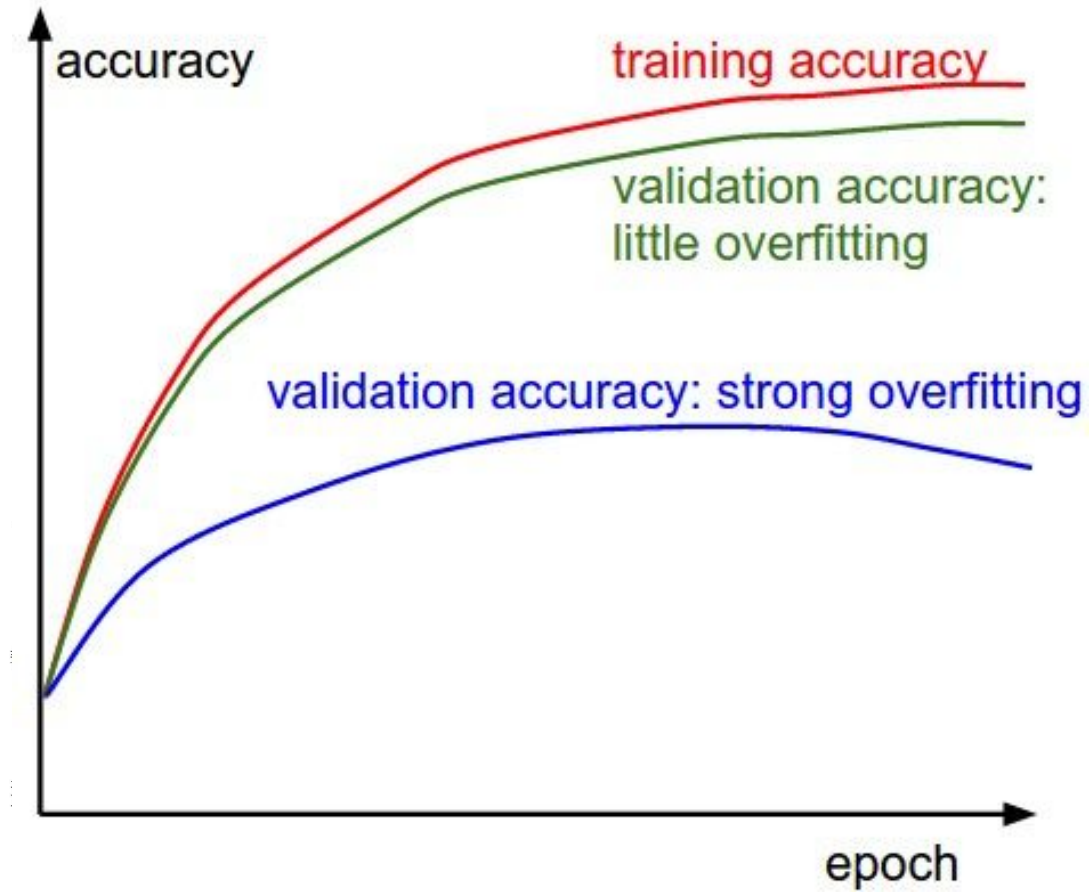


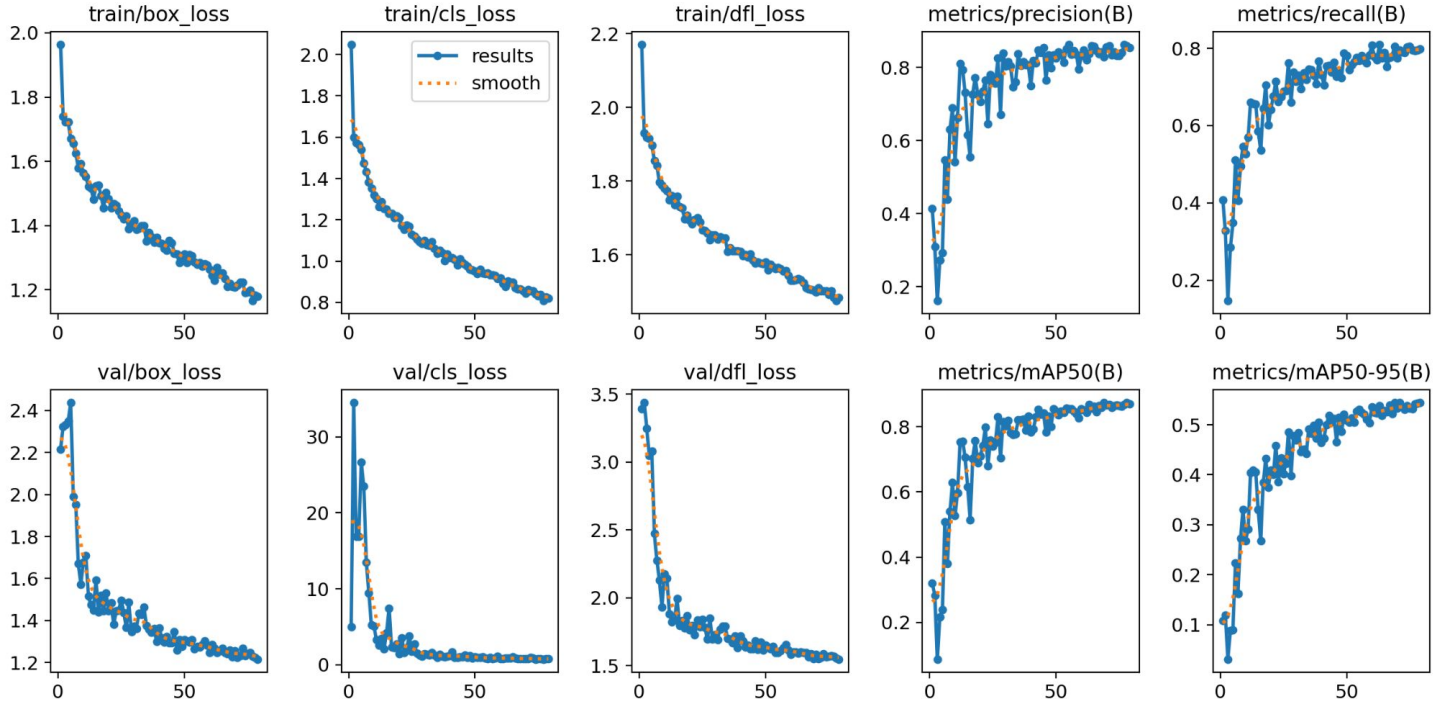


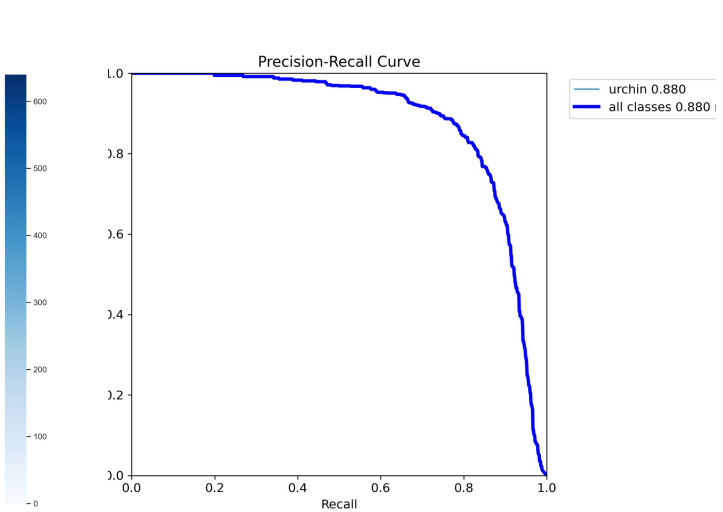
NOAA FISHERIES

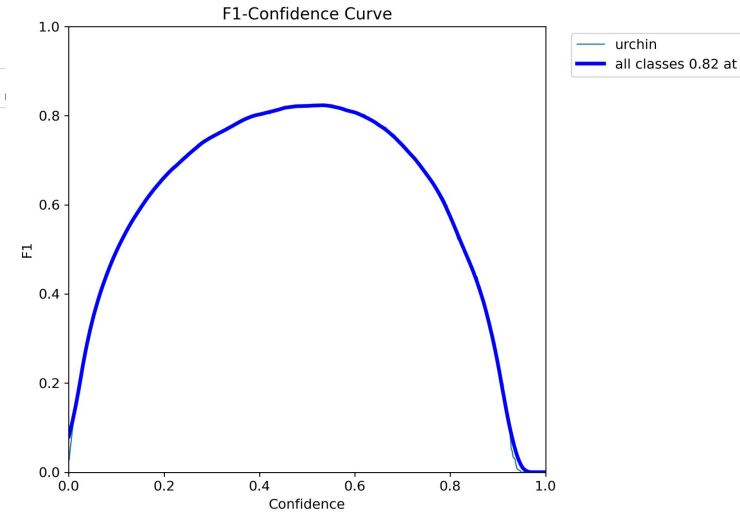
Lessons Learned

- **There are built in Evaluation Tools for Metrics**



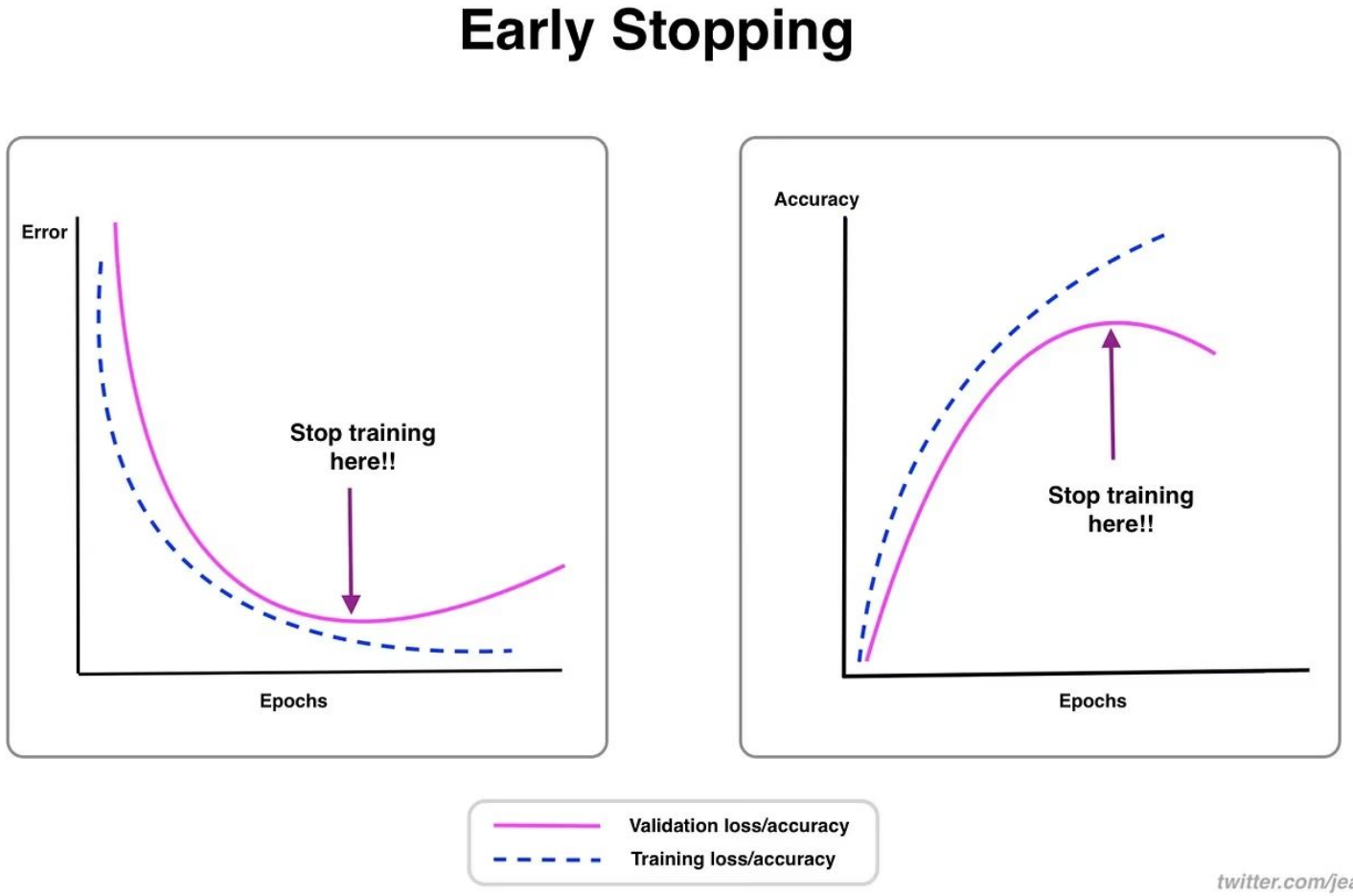






Lessons Learned

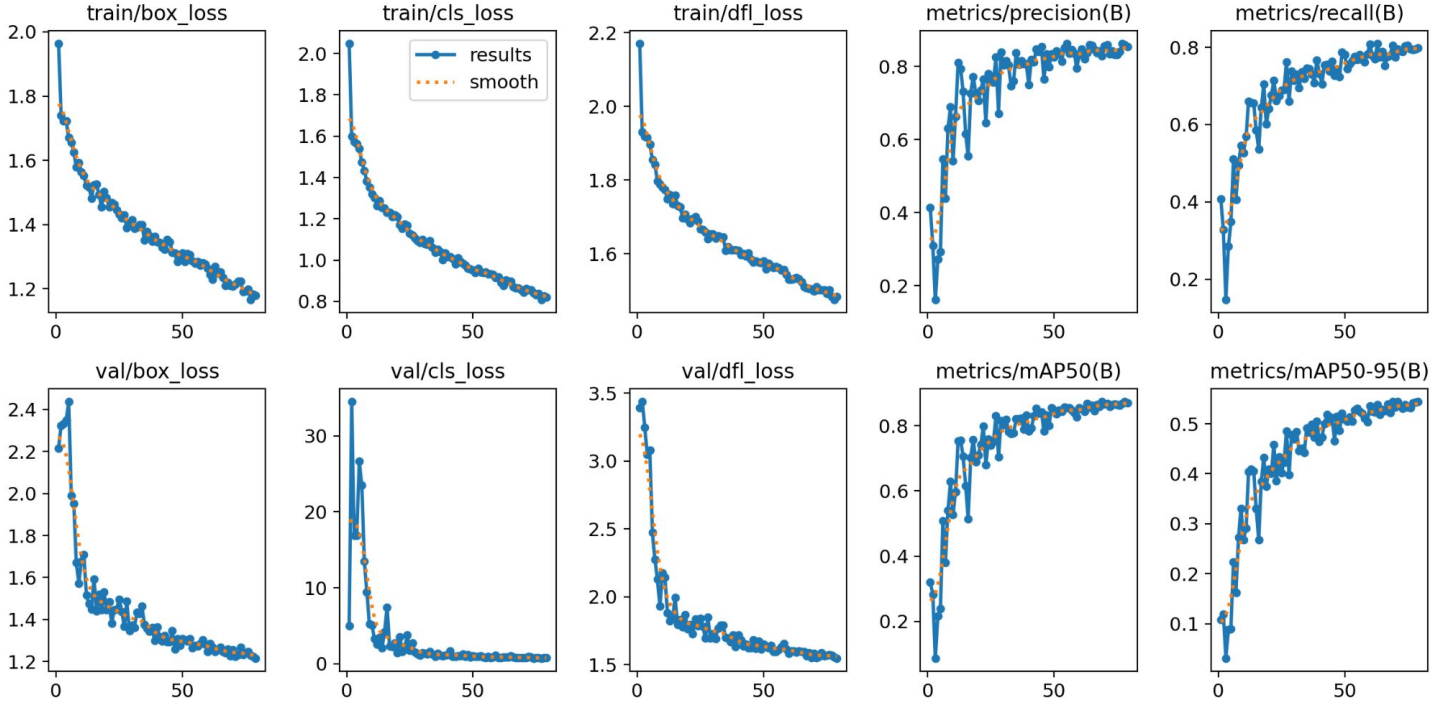
- **More training is not always better.**
- **Stopping early** can help **prevent** "**Overfitting**" (memorizing of the training set) and help optimize the model.

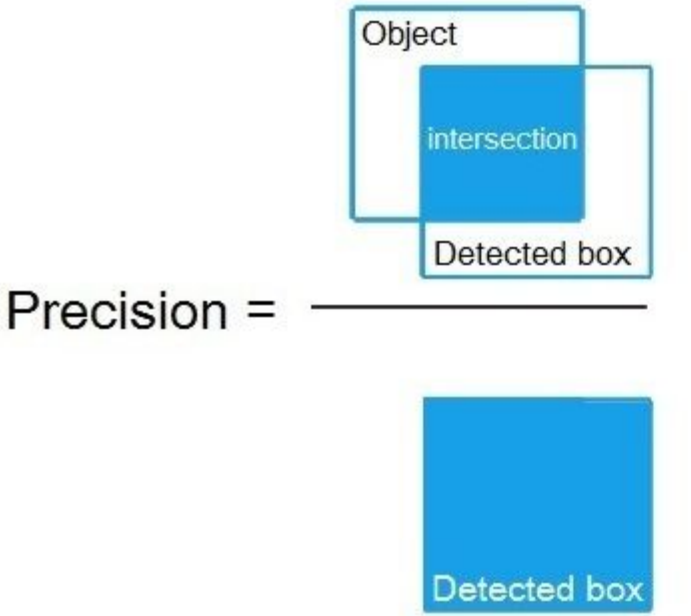


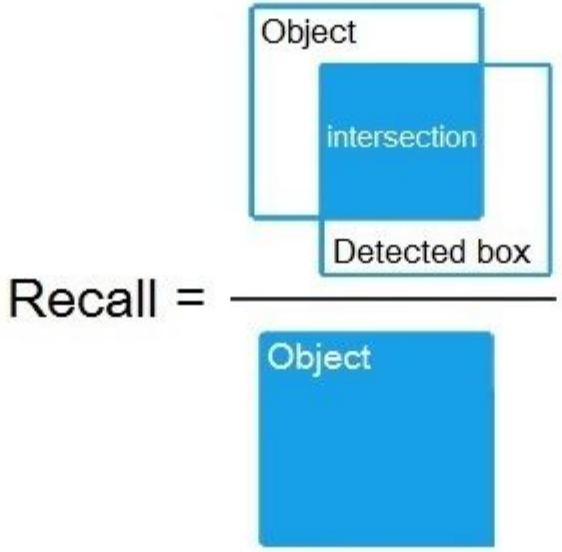
NOAA FISHERIES

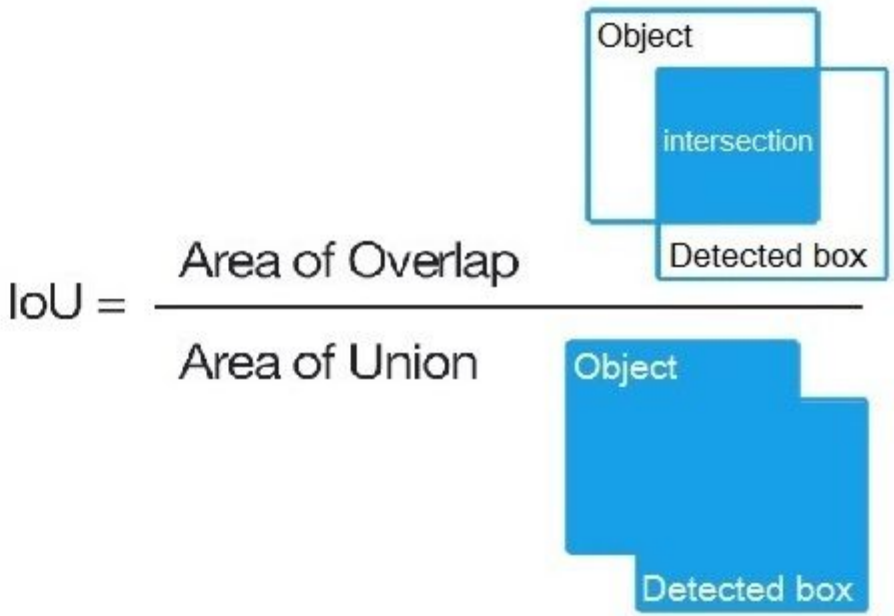
Lessons Learned

- **Evaluation Tools for Metrics**



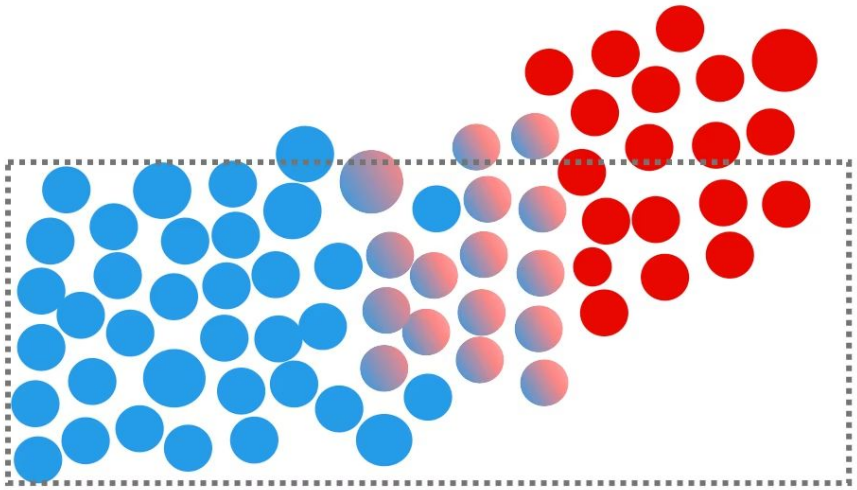




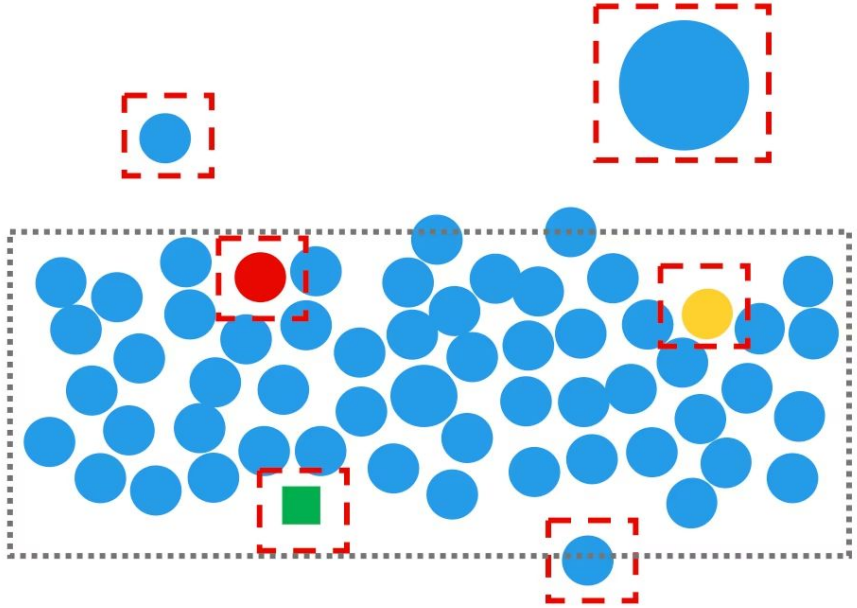


urchin
all classes 0.82 at

Data drift



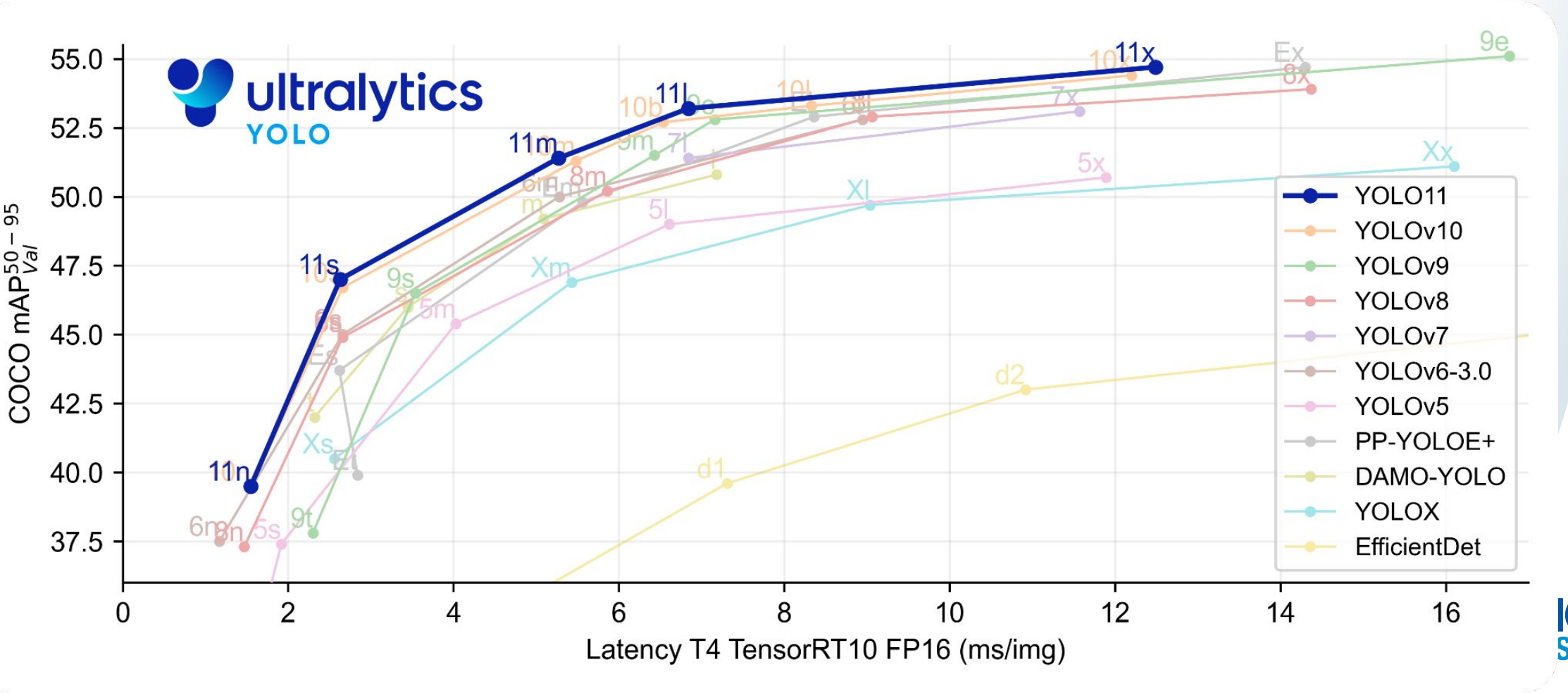
Outliers



NOAA FISHERIES

Lessons Learned

- **Changes, improvements, and updates with AI happen fast**



The Fun Stuff: Test Models, Advanced Techniques & Examples

NOAA FISHERIES

Marine Detect

This repository provides access to two **YOLOv8 object detection models** for identifying **species** of interest in **underwater environments**.

Datasets Details

The models utilize a combination of publicly available datasets (~ 90%) and Tēnaka-based datasets (~ 10%). Some datasets were already annotated, and others undergo manual labeling.

References to the public datasets used can be found in the 'References' section of this README.

The images used with annotations (YOLO format) can be downloaded using the following links: FishInv dataset, MegaFauna dataset.

Datasets split details

Model	Training + Validation Sets	Test Set
FishInv	12,243 images (80%, 20%)	499 images
MegaFauna	8,130 images (80%, 20%)	253 images

GitHub

more time for restoration.

ultralytics HUB BETA

Home

Datasets

Projects

Models

roboflow

Universe | Public Datasets | Model Zoo | Blog | Docs

Home > Datasets

DATASET TYPE

All Datasets 40

Object Detection 36

Classification 4

Computer Vision Datasets

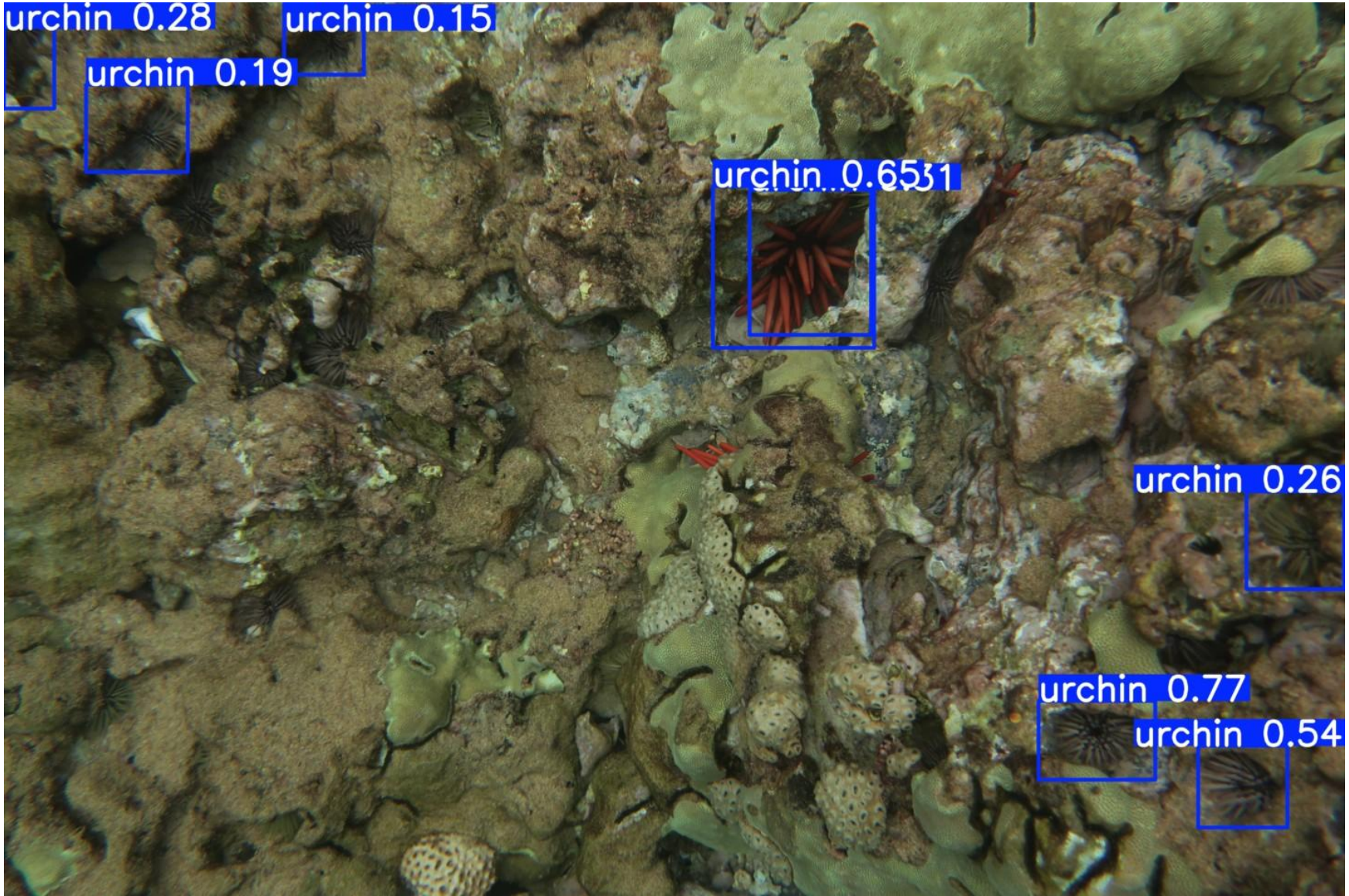
Roboflow hosts free public computer vision datasets in many po available.
If you'd like us to host your dataset, please get in touch.

Roboflow 100
Object Detection Benchmark
100 datasets | 7 Domains | 224k Images

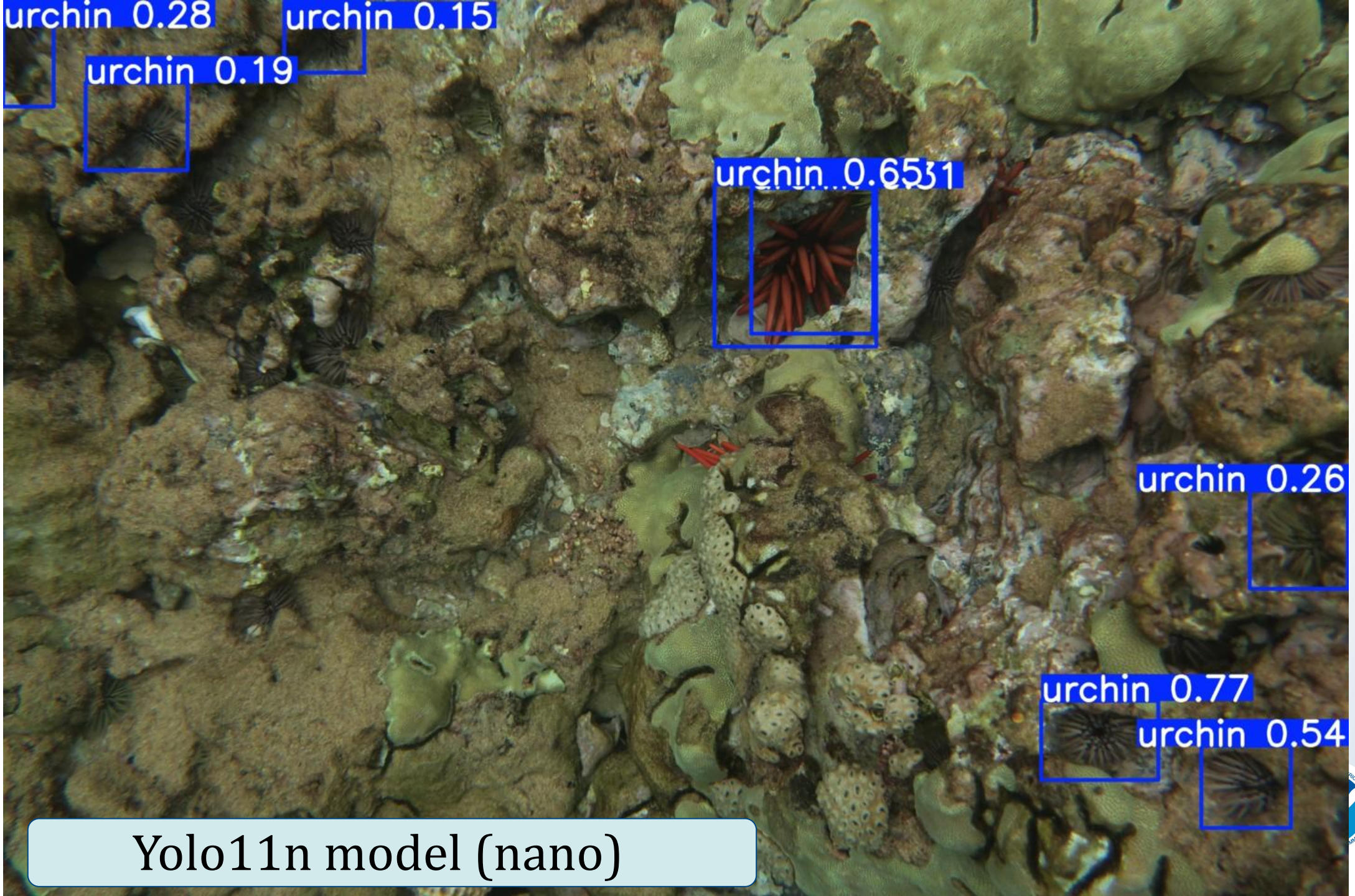
Microsoft COCO 2017 Dataset
Object Detection (Bounding Box)
120358 images | 12 exports | Last updated a month ago

Object Detection Tests: Urchins

Yolo11n model (nano)



NOAA FISHERIES

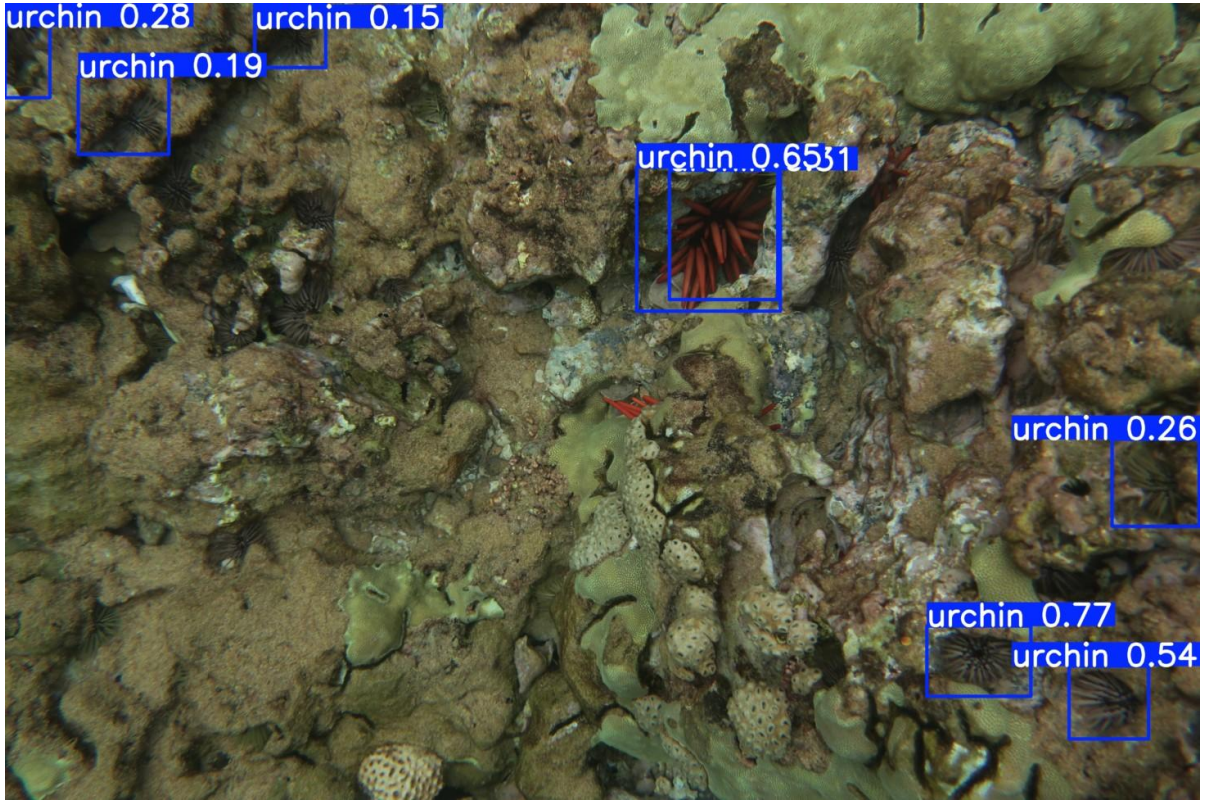


Yolo11n model (nano)

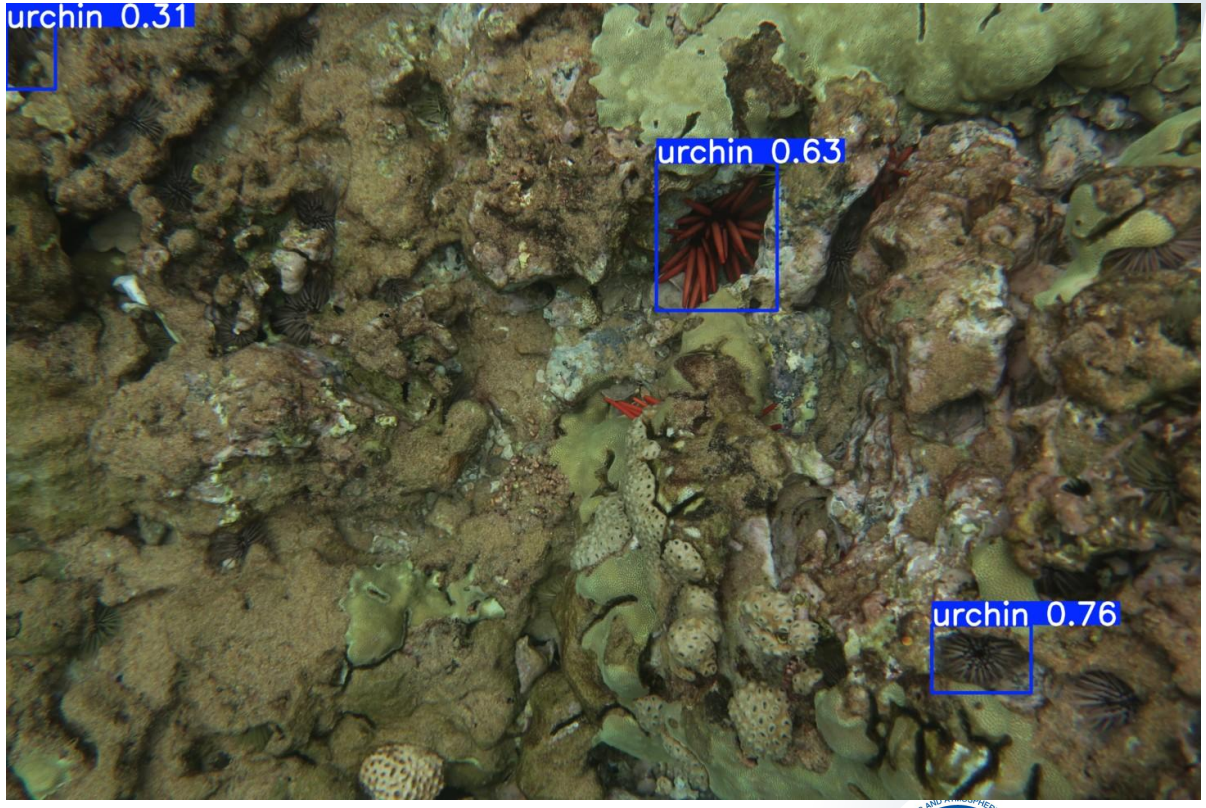
NOAA FISHERIES

Lessons Learned

Yolo11n at 50 epochs



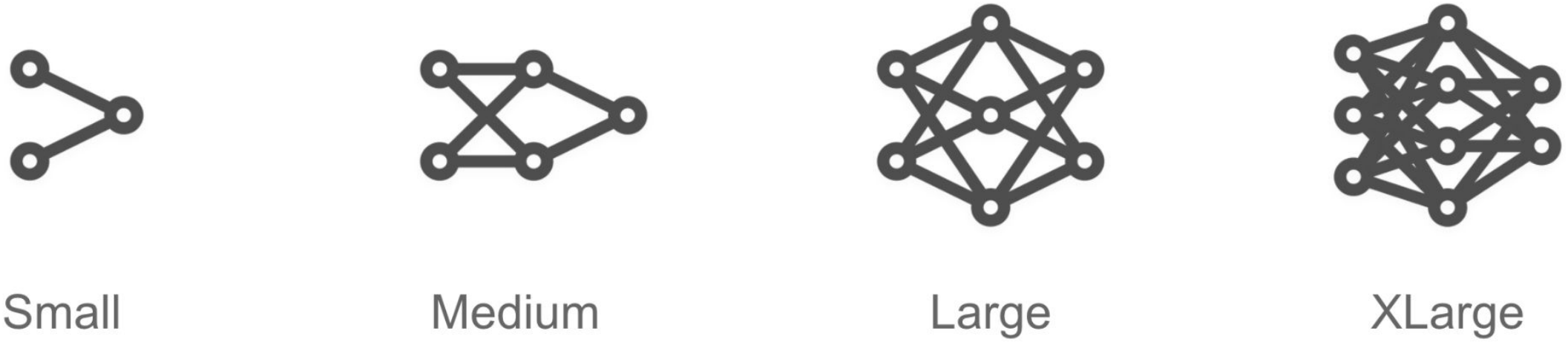
Yolo11n at 100 epochs



note: When training a model, an epoch refers to one complete pass through the entire training dataset.

Lessons Learned

- **Choose a model that makes sense for the task**
  - **nano for on edge and speed**, and simple/large objects
  - **XL for complexity**
  - **when unsure, try medium**

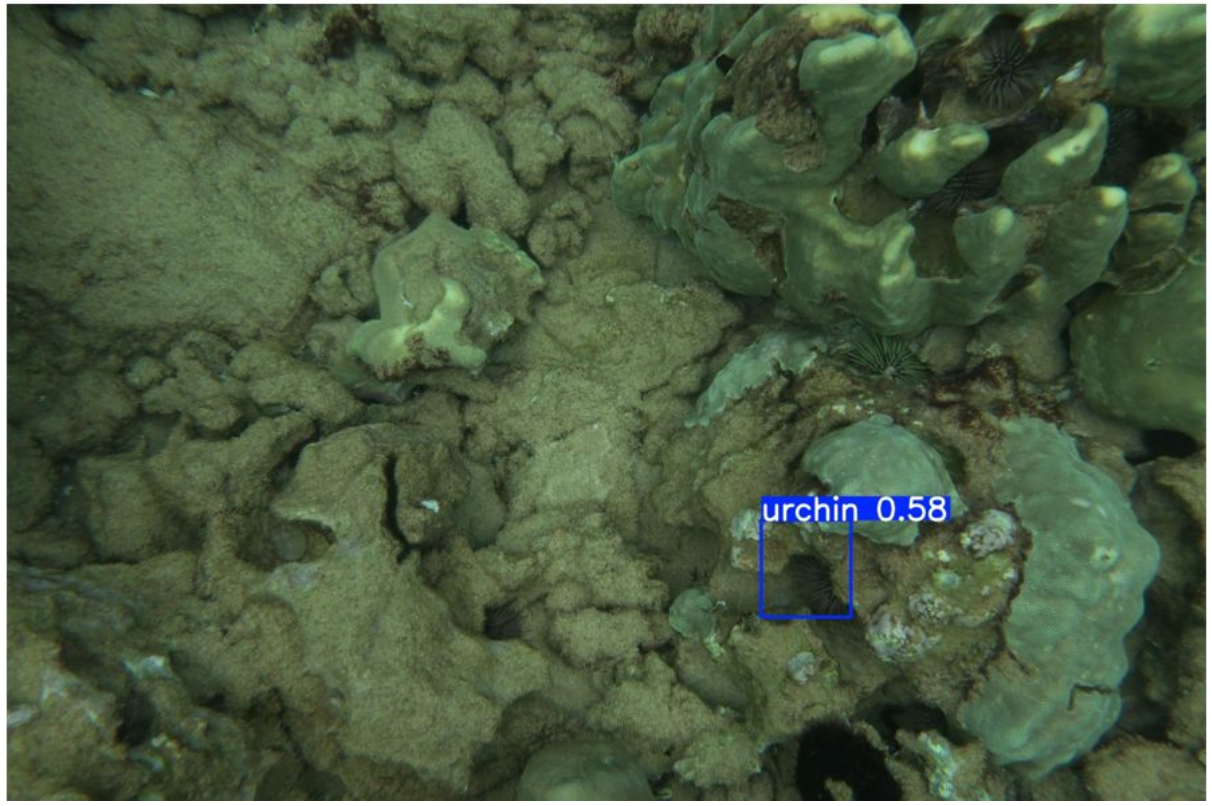


~45mins **~4 hours** **~8 hours**

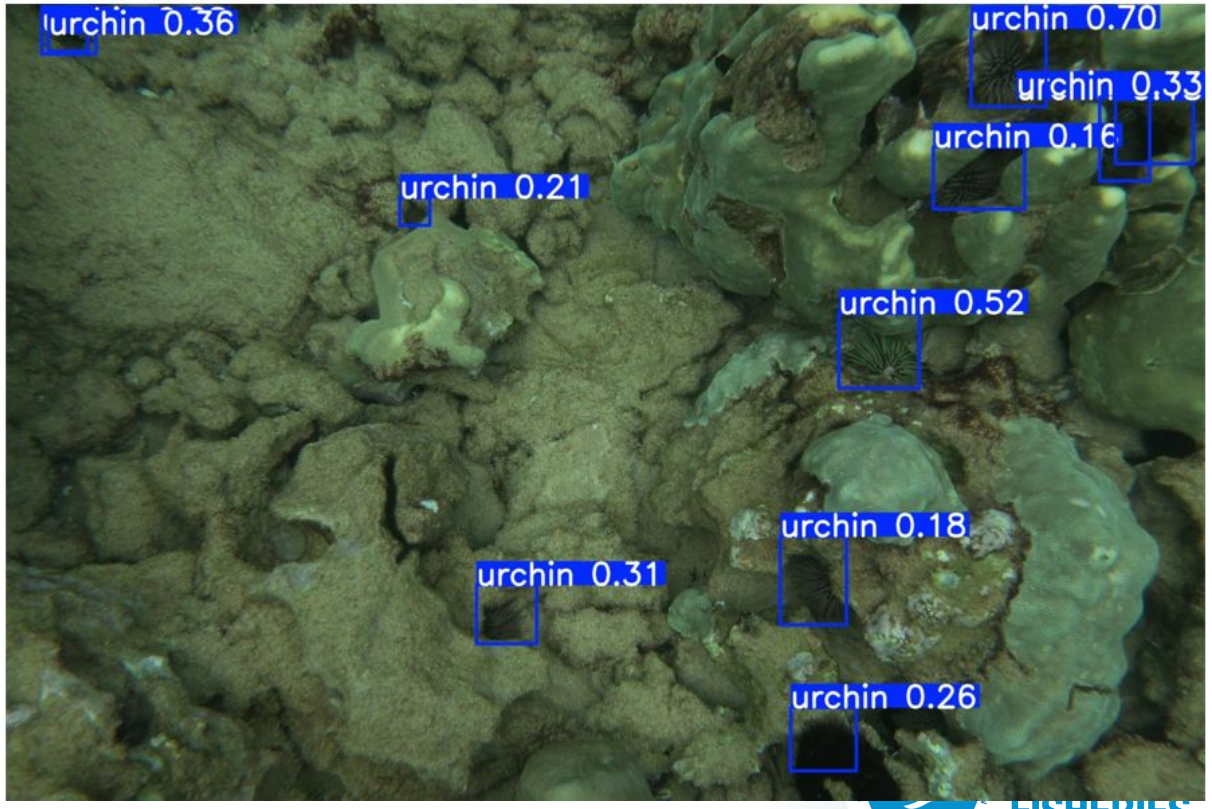
NOAA FISHERIES

Yolo11 Nano Model vs Yolo11 Medium Model Performance

Urchin Detector yolo11n(nano)

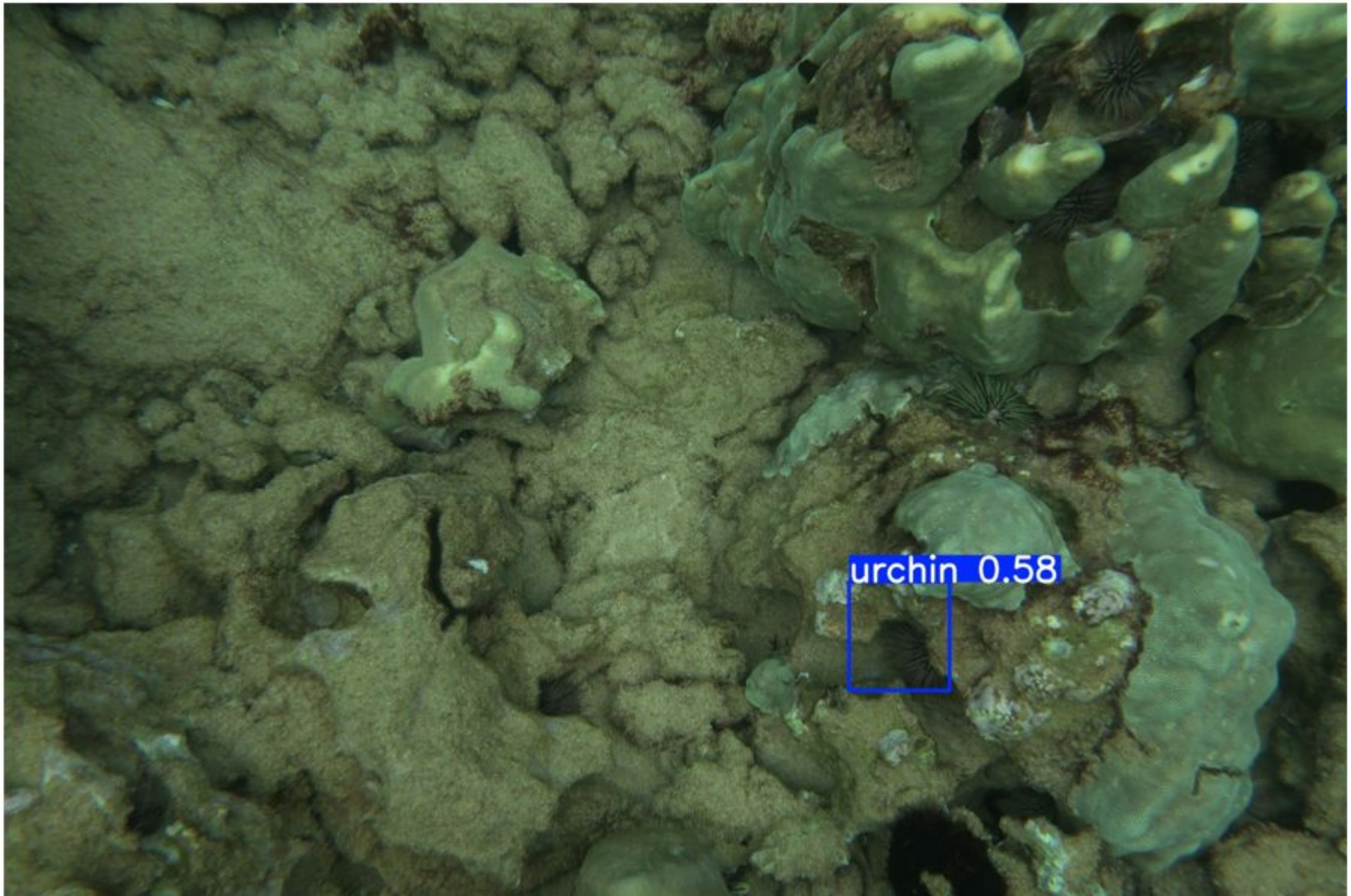


Urchin Detector yolo11m(medium)



FISHERIES

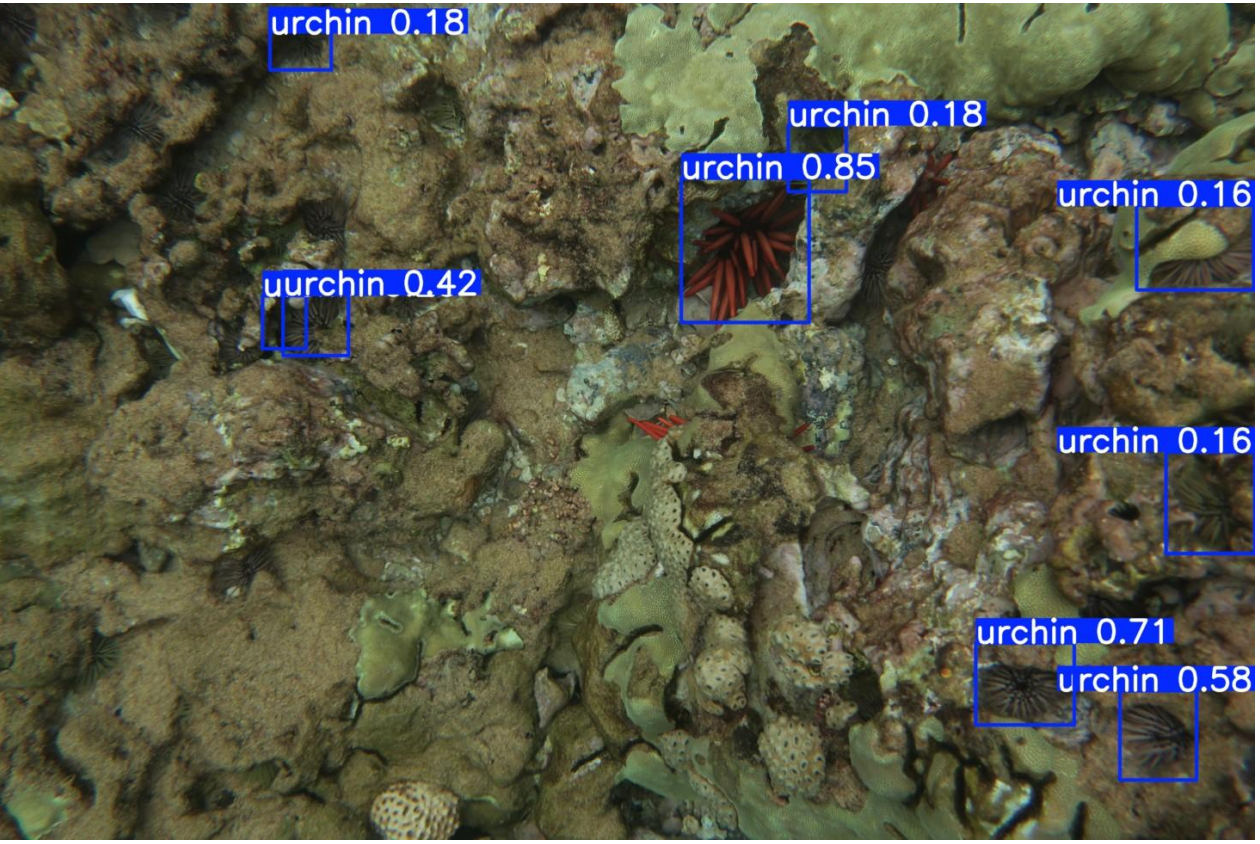
Urchin Detector yolo11n(nano)



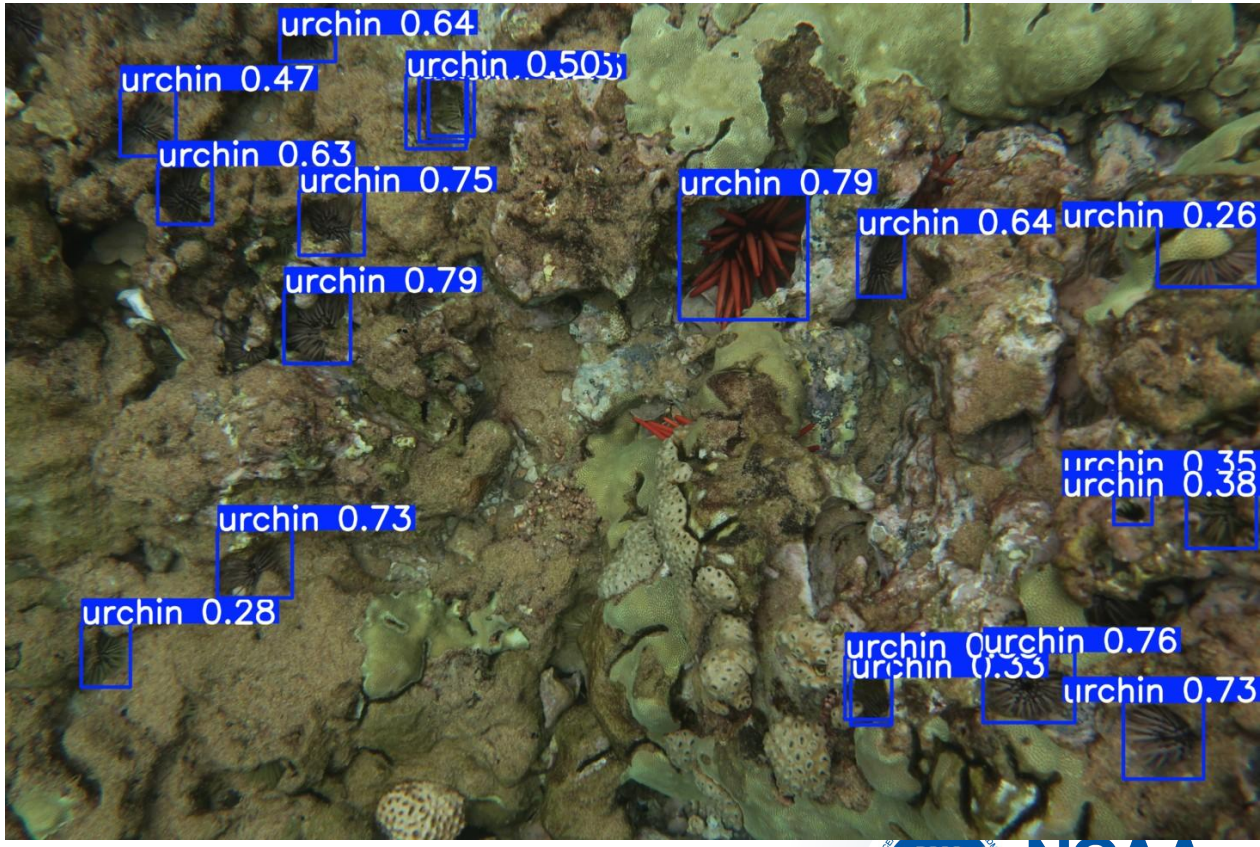
NOAA FISHERIES

Small vs Larger Training Dataset Performance

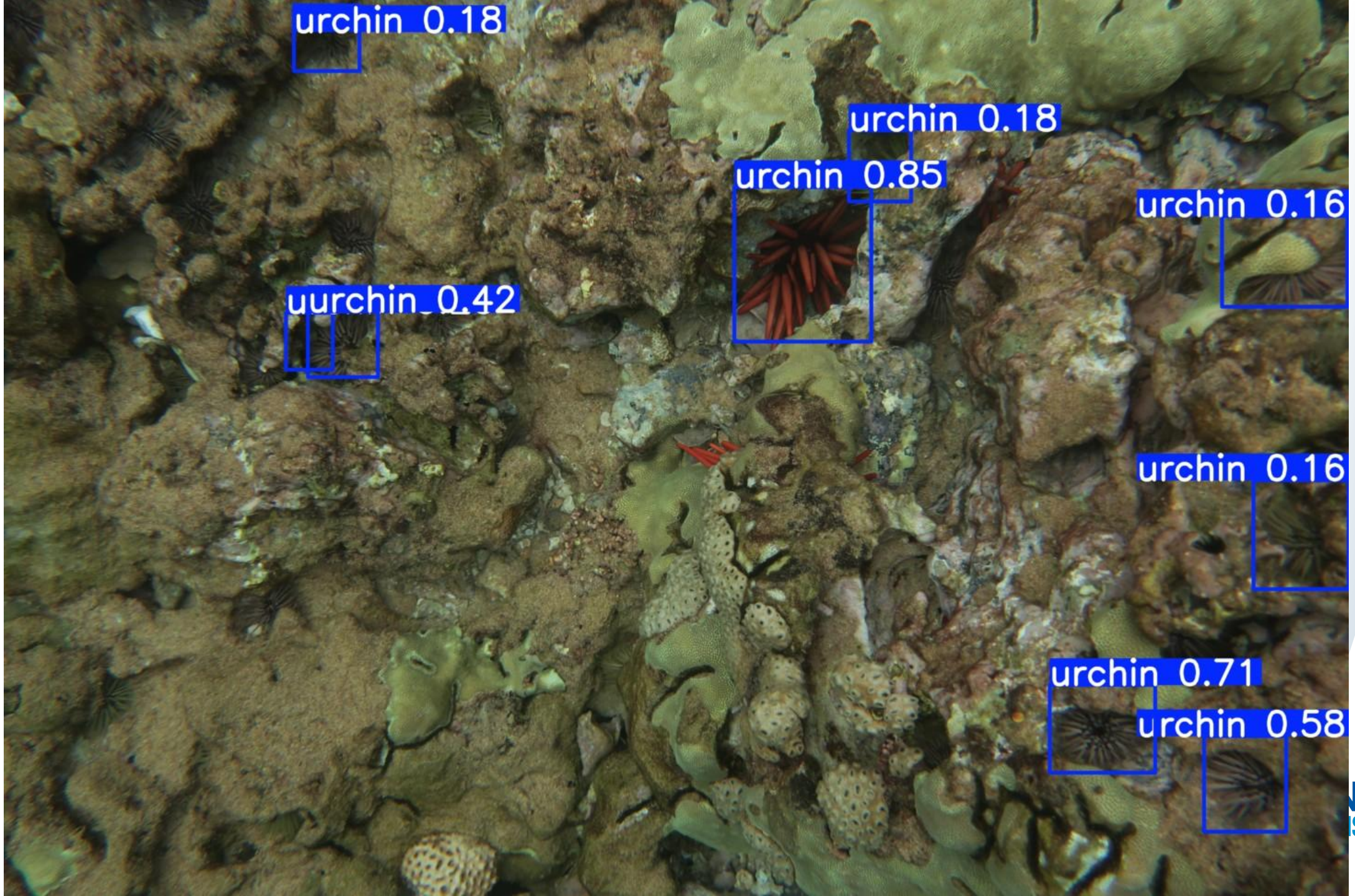
YOLO11m | 3,330 image training dataset



YOLO11m | 10,345 image training dataset



NOAA FISHERIES

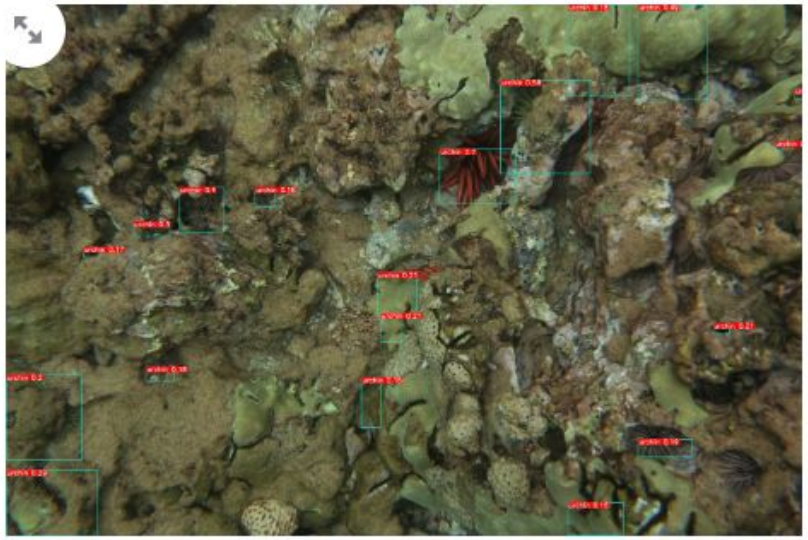

urchin 0.18
urchin 0.18
urchin 0.85
urchin 0.16
uurchin 0.42
urchin 0.16
urchin 0.71
urchin 0.58

Hybrid Object Detection: Using Patched Based Inference for small objects

Processing MAI-B4010_2019_18.JPG...

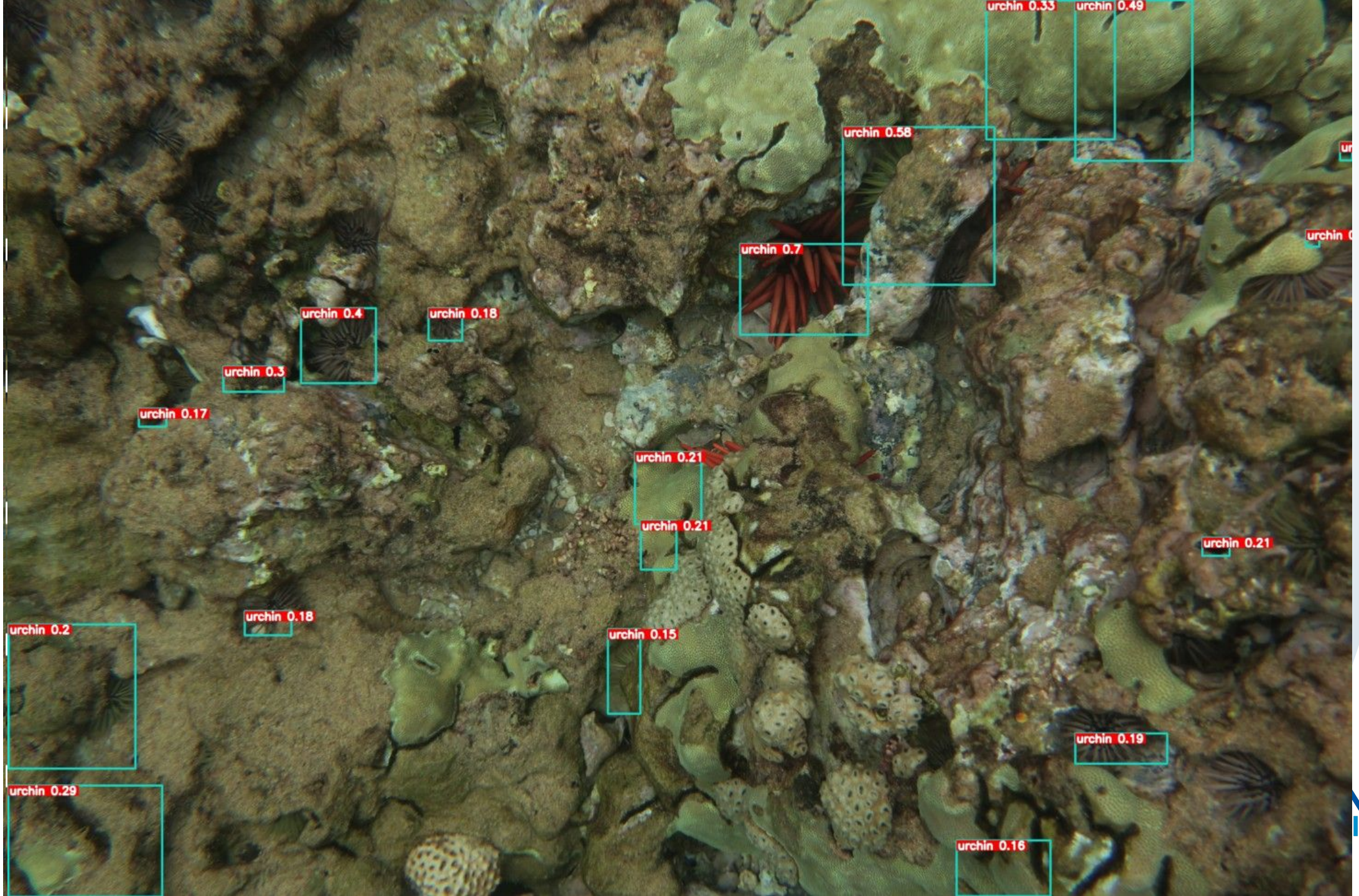
Original Image

Composite of Generated Crops



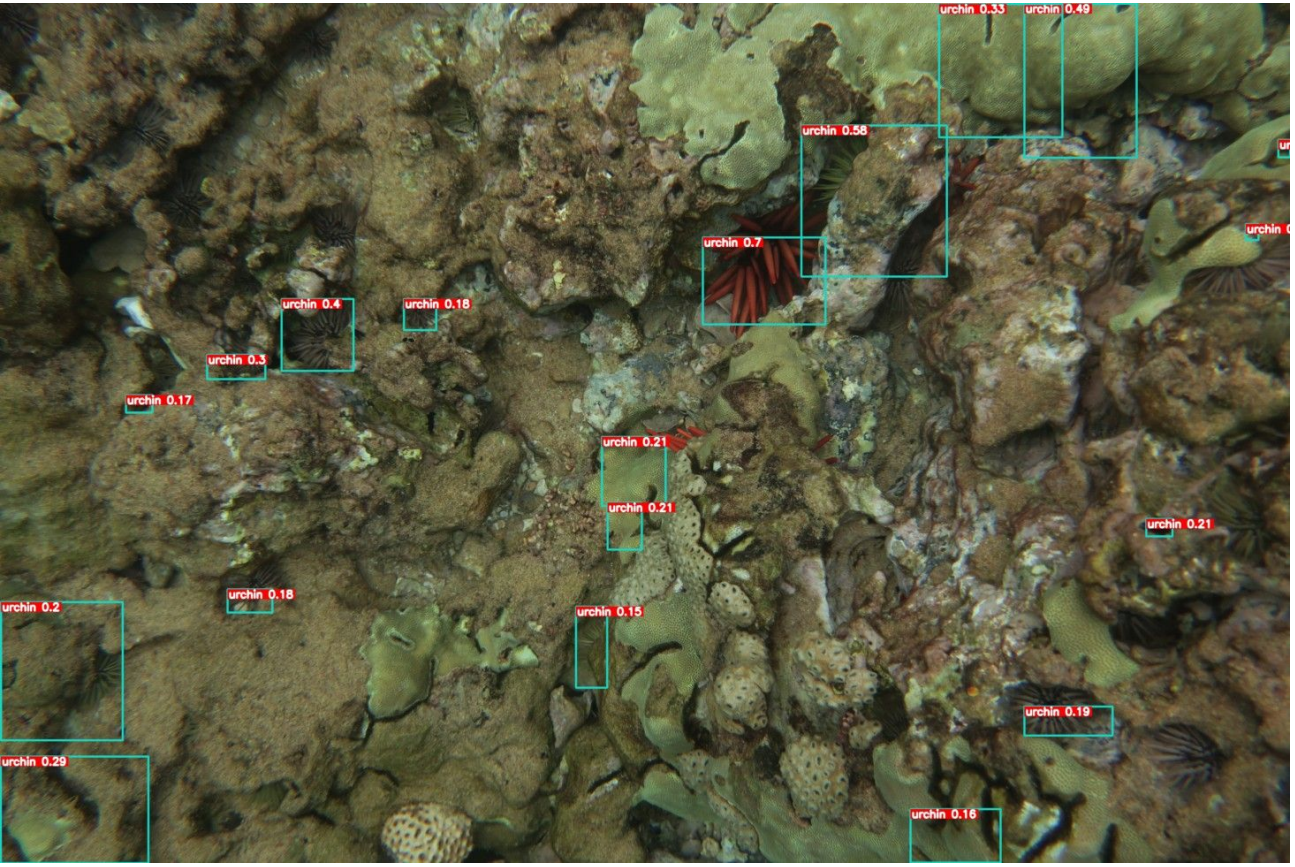
Detection Result

NOAA FISHERIES

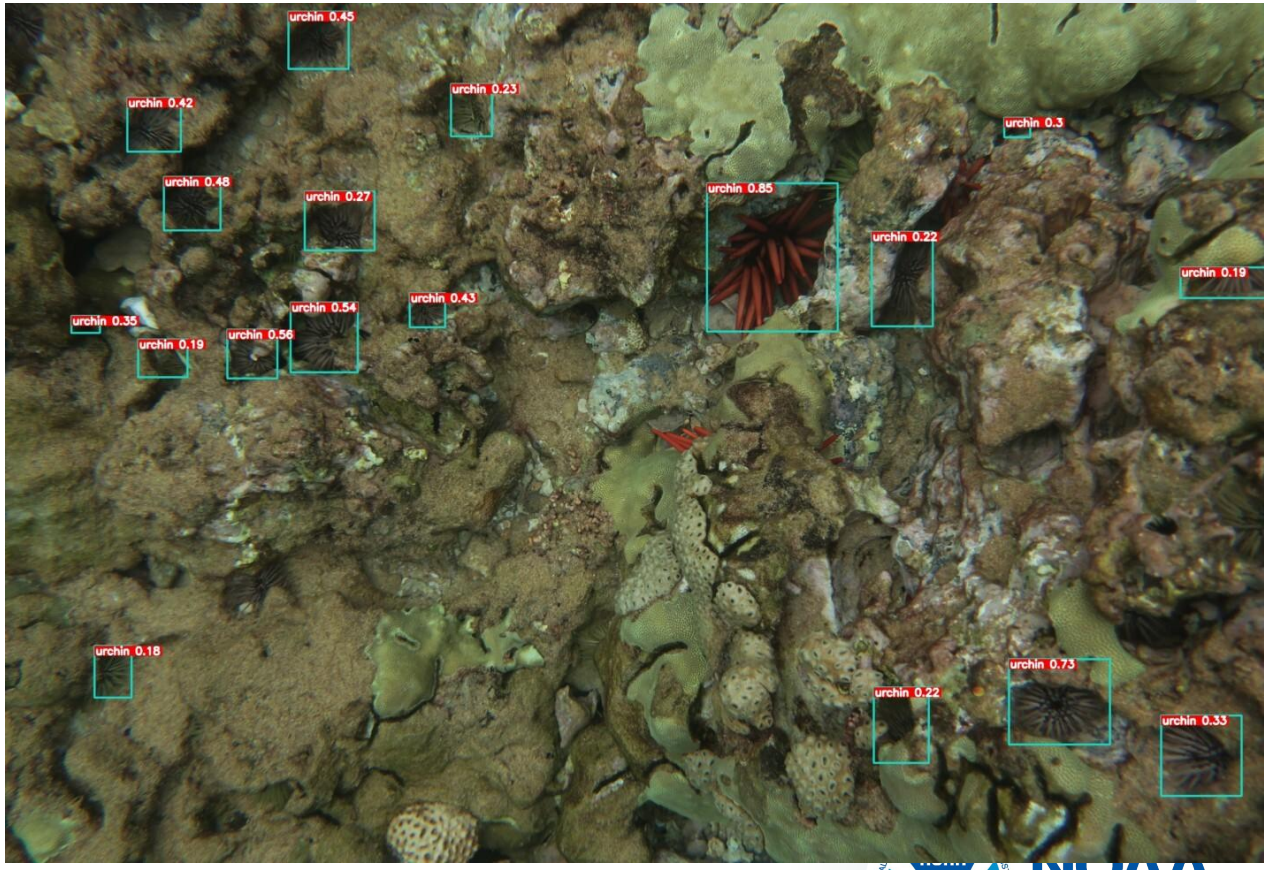

urchin 0.33
urchin 0.49
urchin 0.58
urchin 0.7
urchin 0.4
urchin 0.18
urchin 0.3
urchin 0.17
urchin 0.21
urchin 0.21
urchin 0.21
urchin 0.18
urchin 0.2
urchin 0.15
urchin 0.19
urchin 0.29
urchin 0.16

Patch Based vs Multi-Scale Patch Based

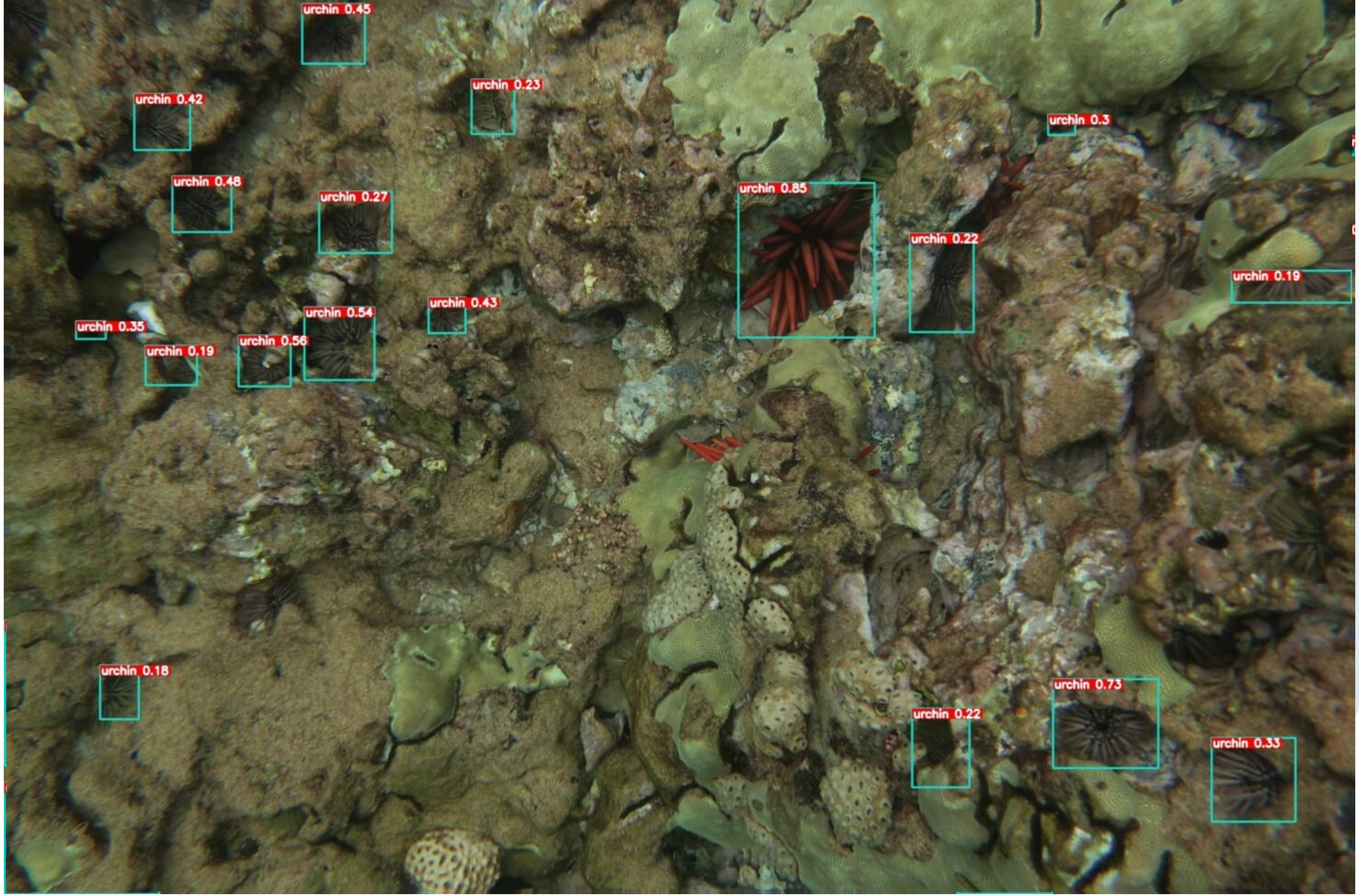
Patch Based Inference



Multi-Scale Patch Based Inference

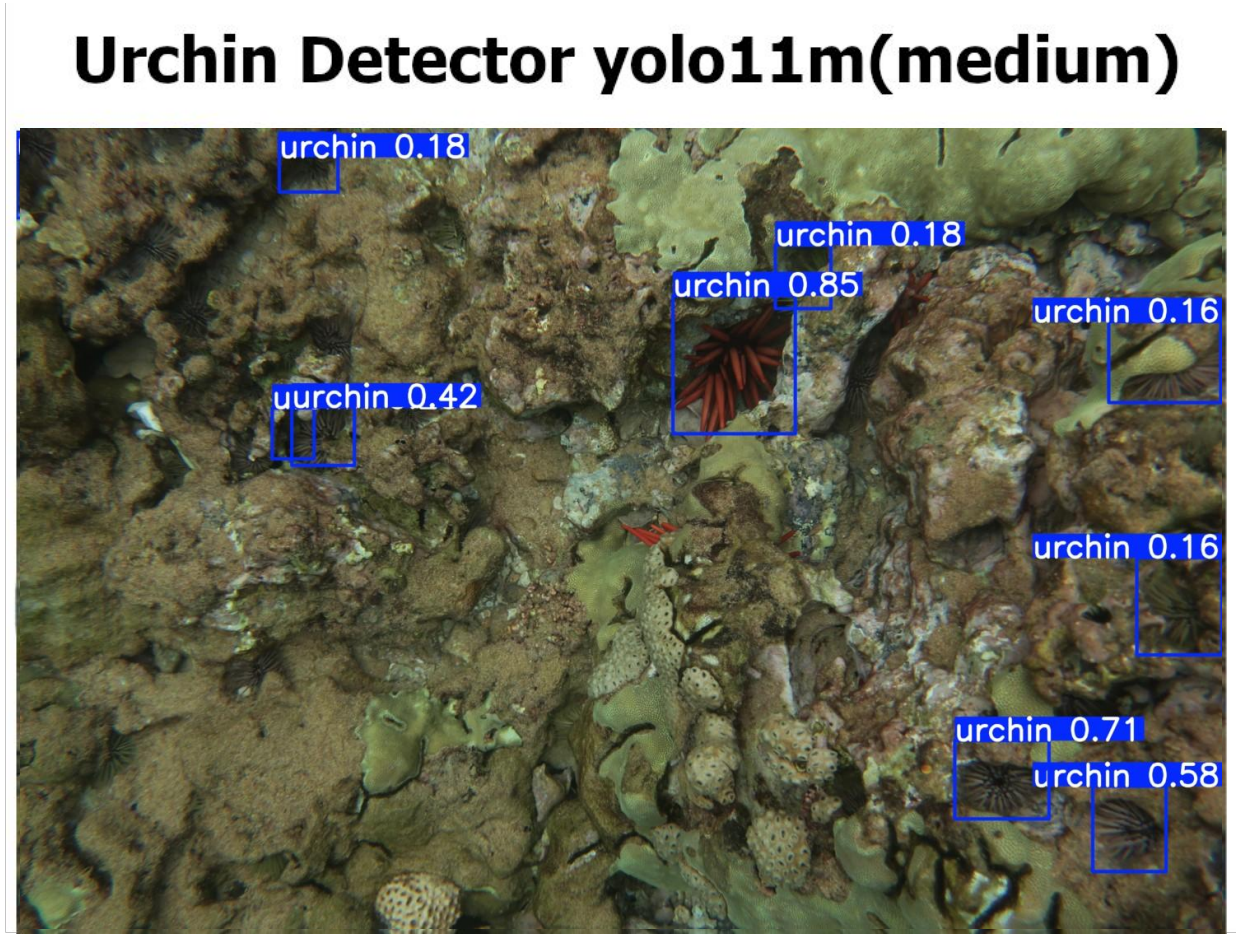


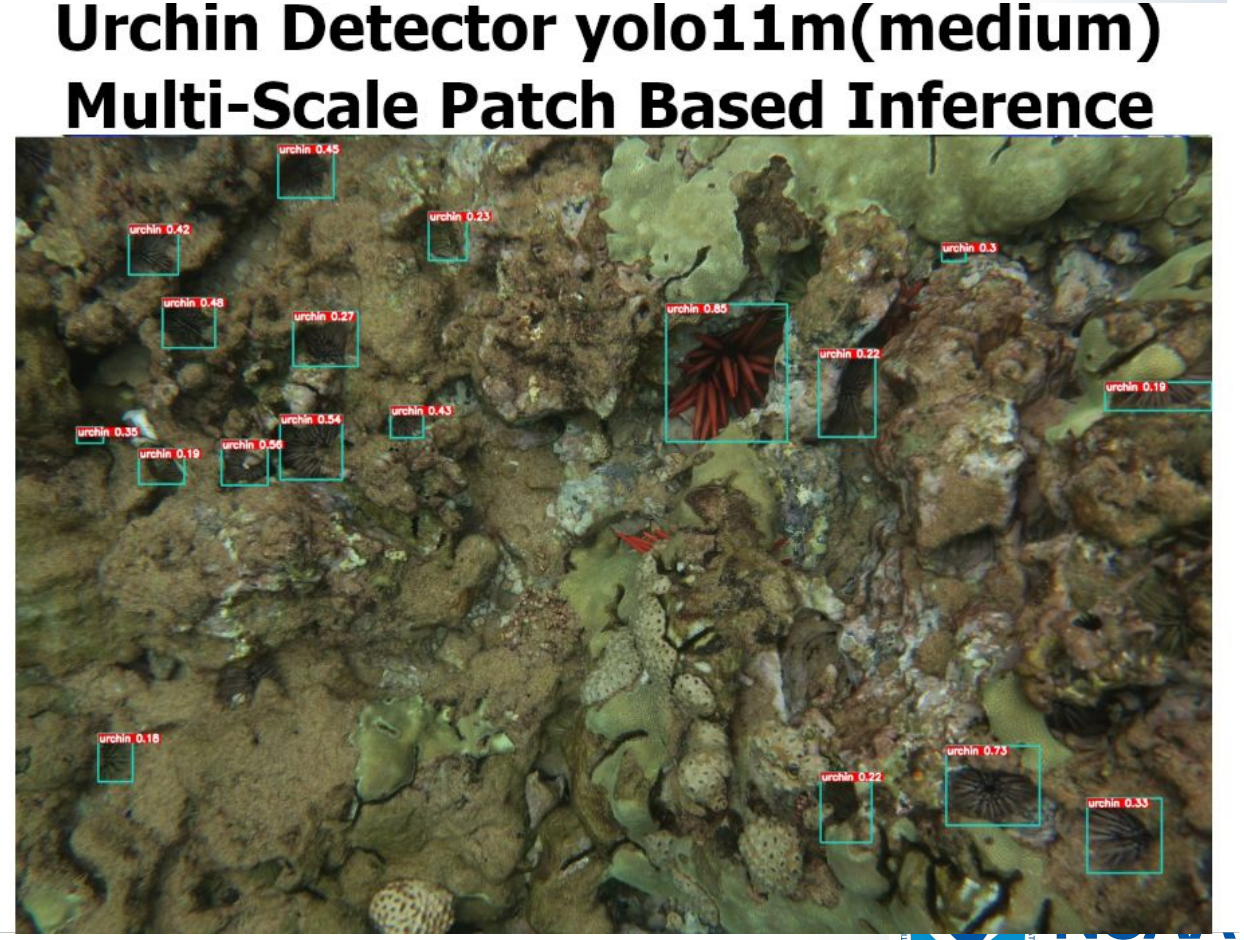
NOAA FISHERIES



NOAA FISHERIES

Multi-Scale Patch Based Inference Comparison



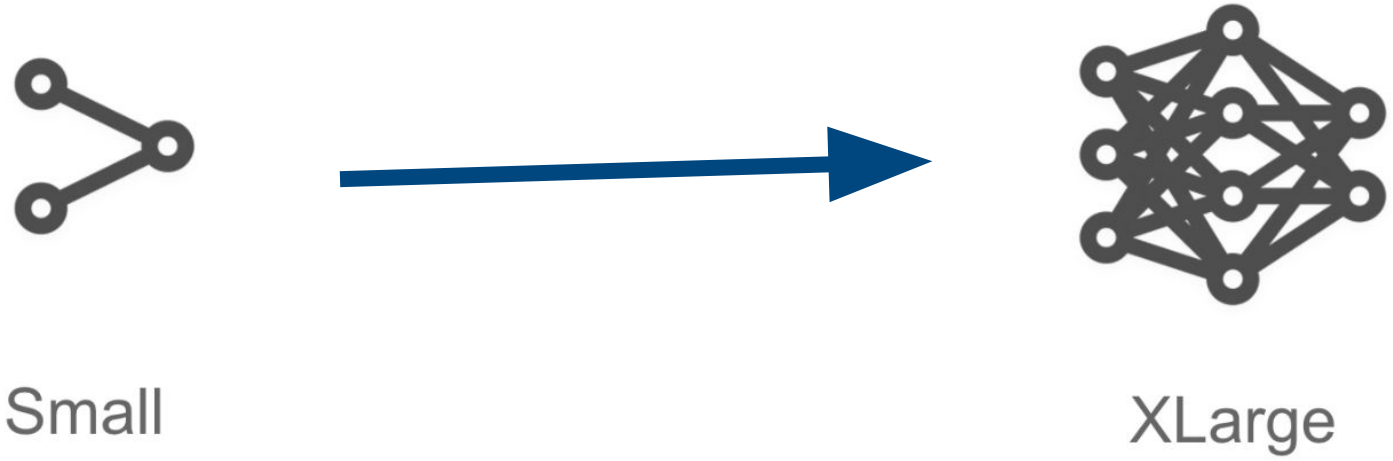


Lessons Learned

- **Explore a multi-model approach**

Use nano/small model for quick & real time data collection/detections

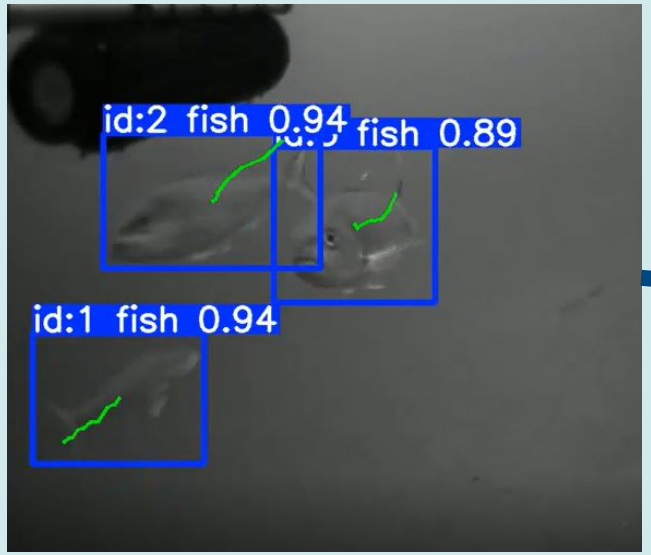
Pass detections to lager model back in the office for finer detections/classifications/sizing



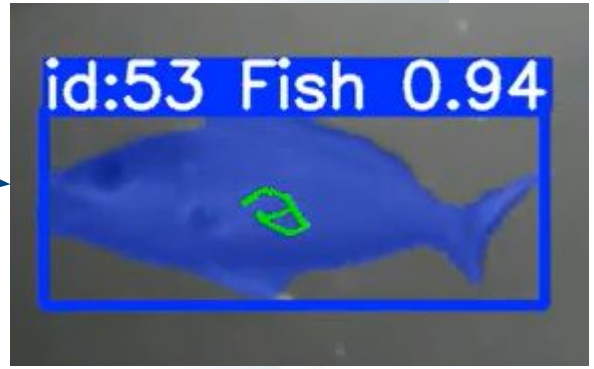
Auto-Annotation:

- **A Quick Way to Segmentation Datasets & Models**

Dataset
+ Trained Detection model
+ Segment Anything Model (SAM)
= Segmentation dataset



Train on segmentation dataset for a Segmentation Model

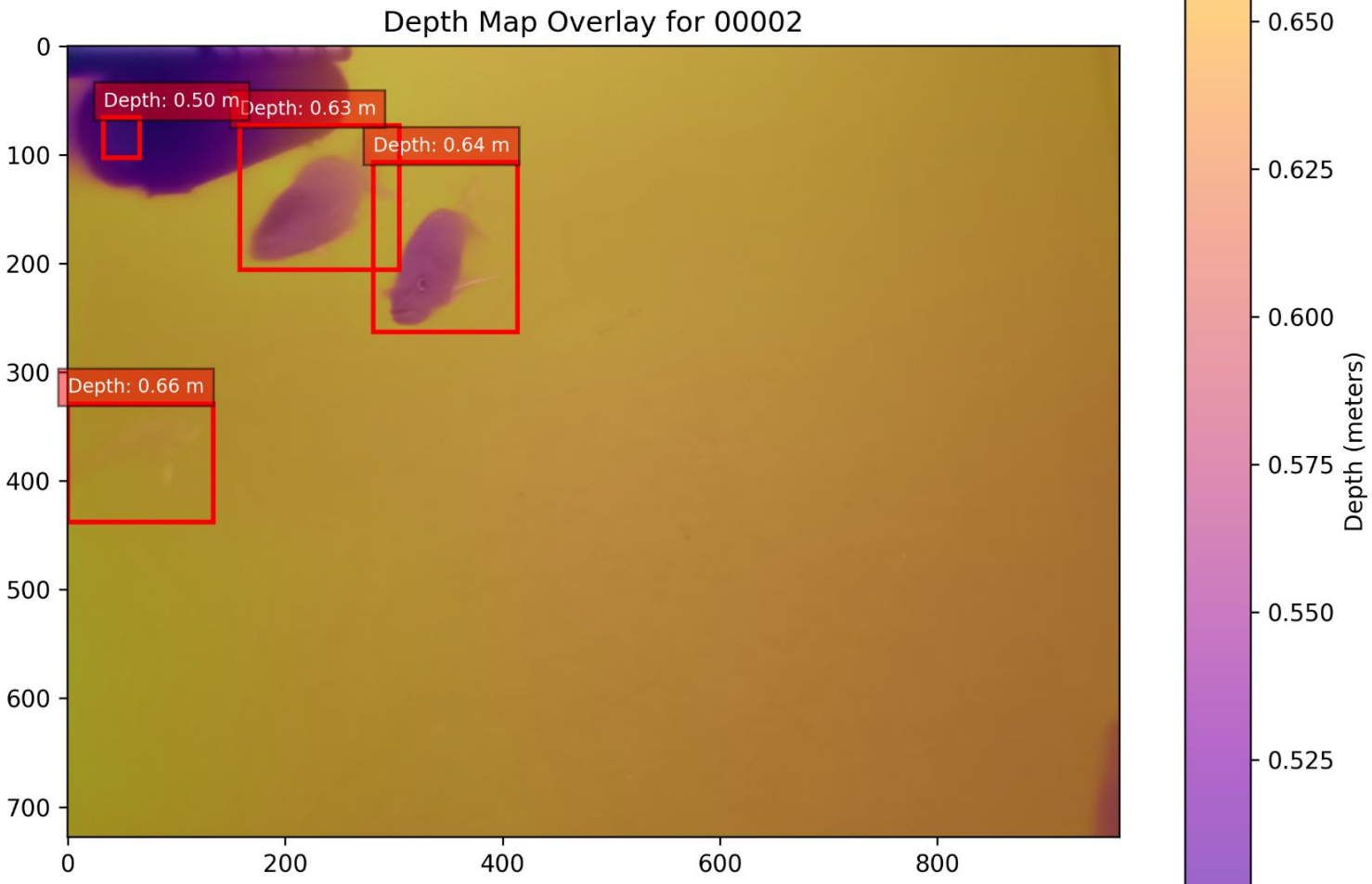


Object Detection Model

Segmentation Model

NOAA FISHERIES

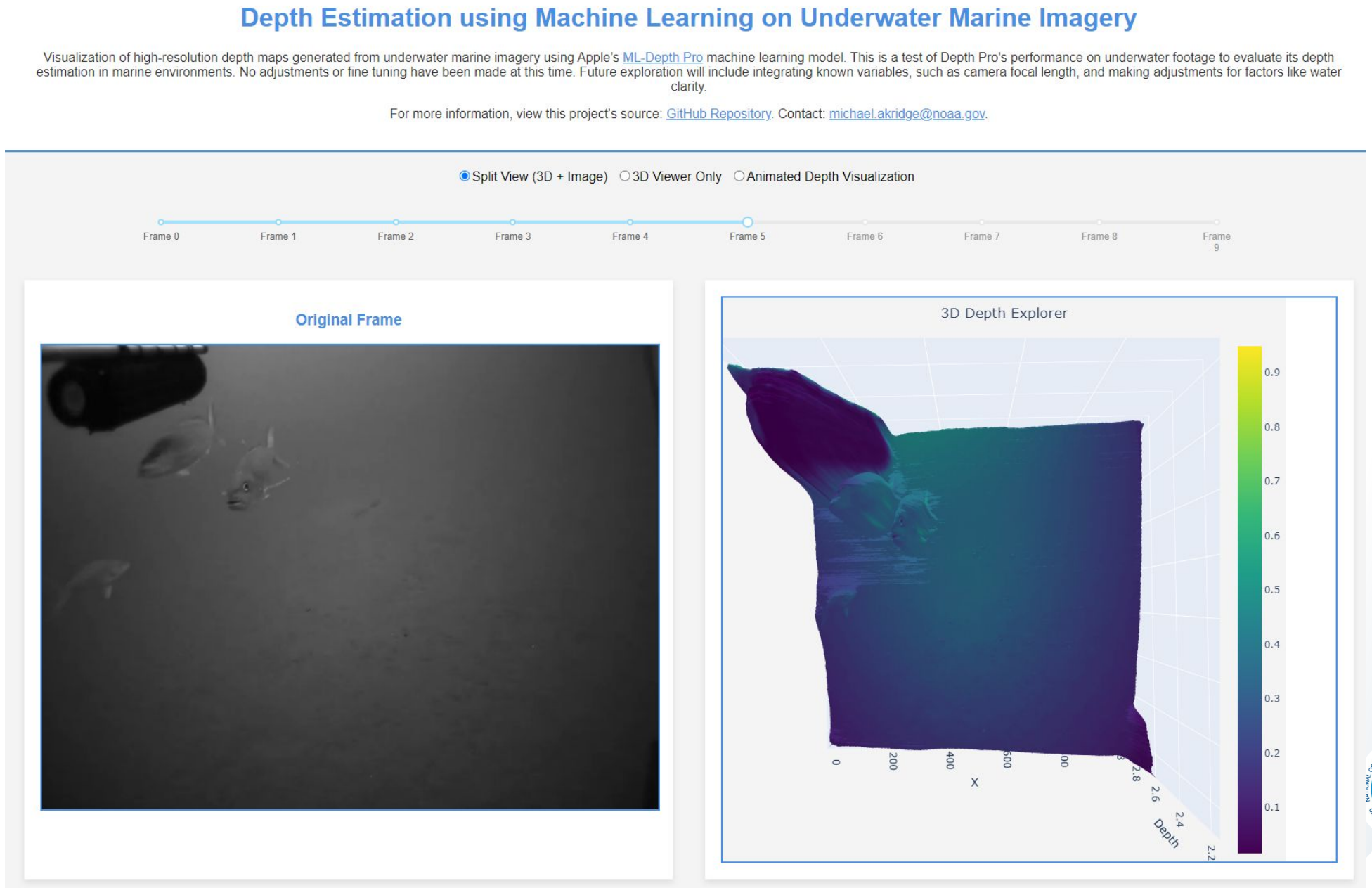
YOLO + Apple ML-Depth Pro

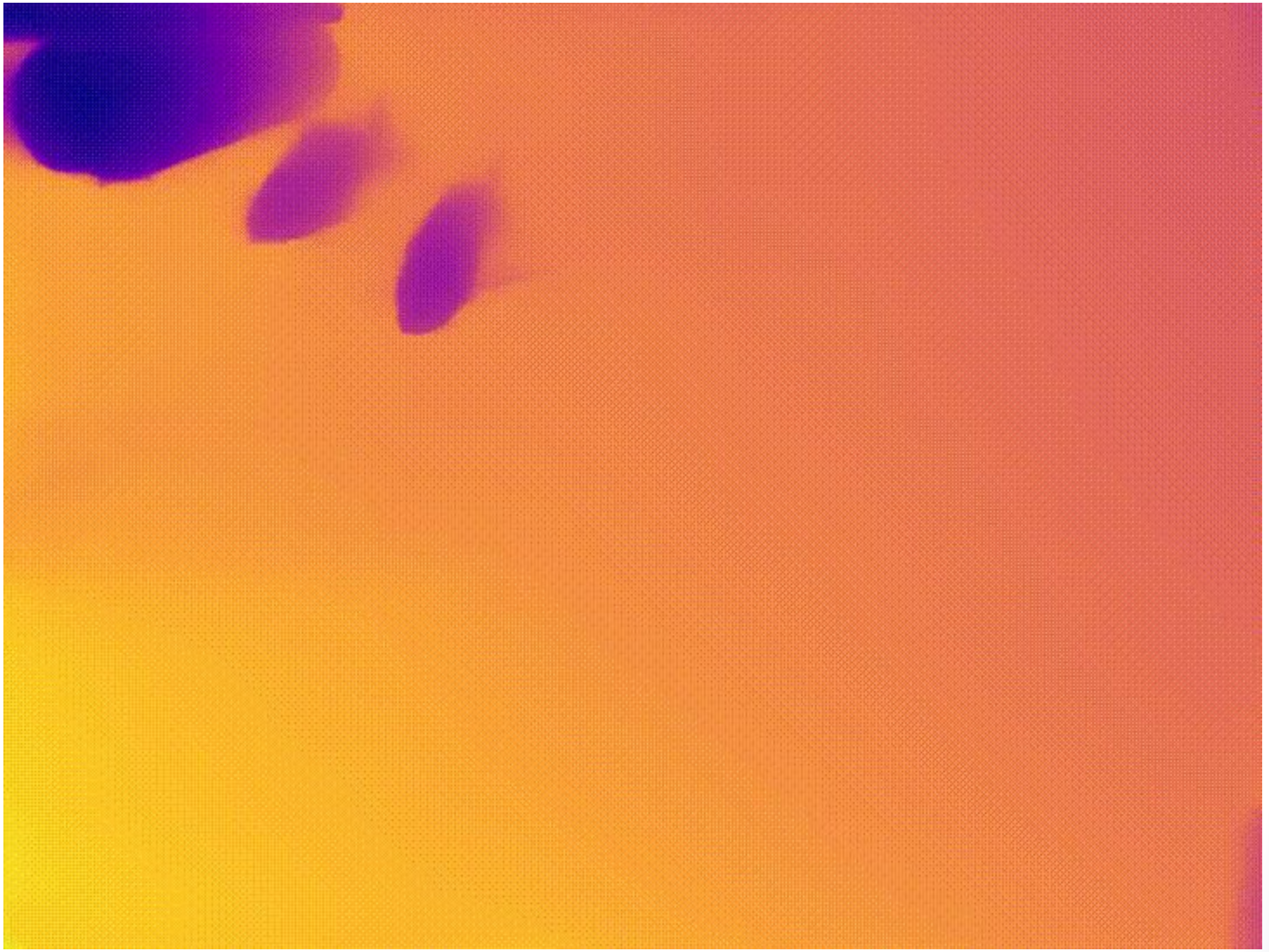


Depth Maps & 3D Point Clouds from a single image

NOAA FISHERIES

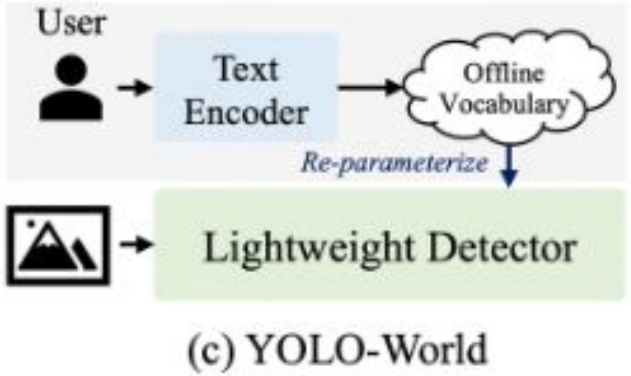
- https://connect.fisheries.noaa.gov/ml_depth_pro/





NOAA FISHERIES

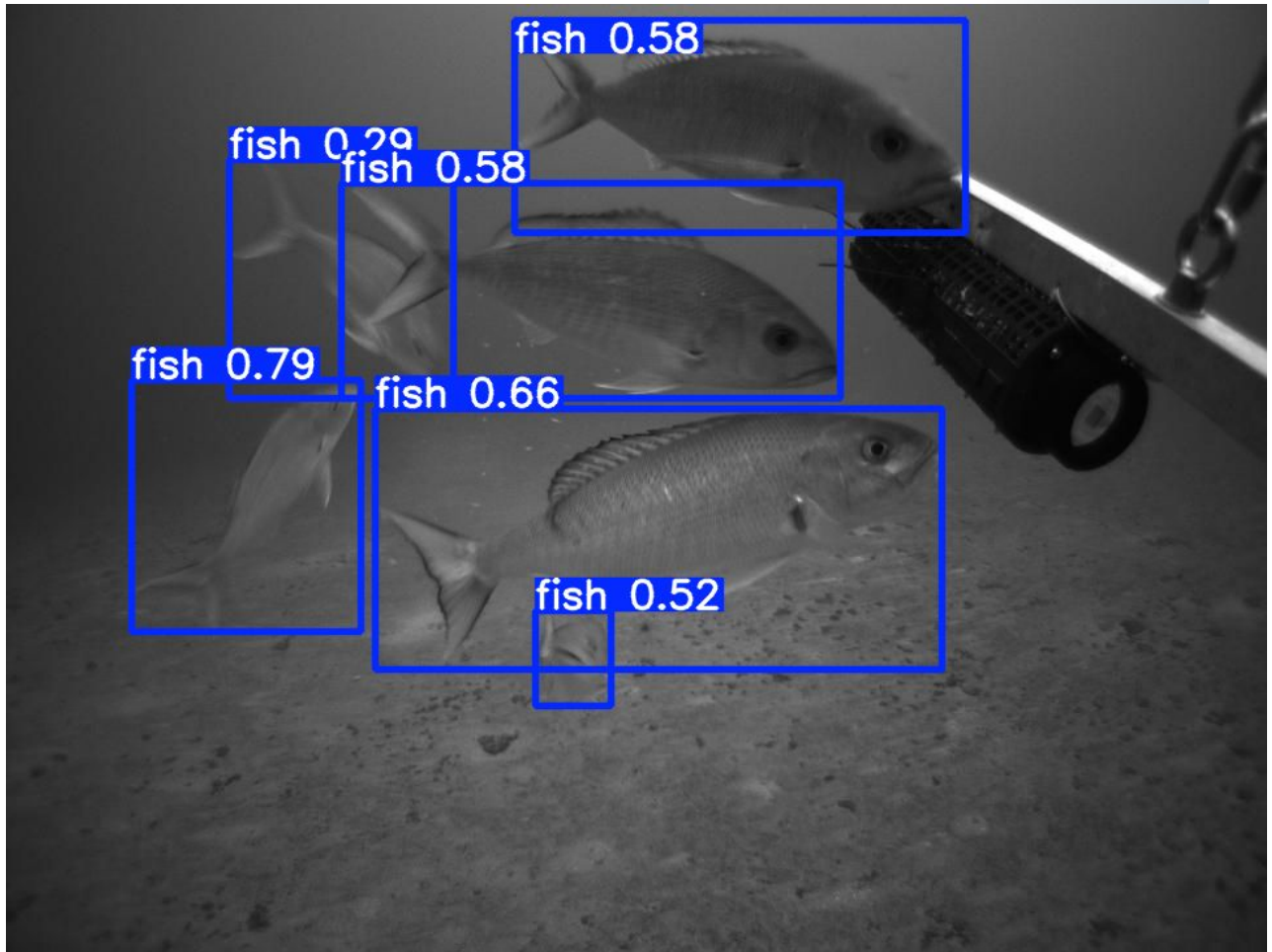
YOLO-World:"Prompt and Detect"



(c) YOLO-World

Zero-shot learning **enables a model to recognize object classes it has never been explicitly trained on.**

Speed: 15.2ms preprocess, 909.2ms inference, 21.4ms post process per image



Lessons Learned

- **Easy to get a DOI**

ORCID
Connecting research and researchers

Michael Akridge

https://orcid.org/0009-0007-0193-9705

Works (4)

AI Model: yolo11m-sea-urchin-detector

2024 | Research tool
DOI: 10.57967/HF/3342
CONTRIBUTORS:

Source: Michael Akridge

NOAA-PIFSC Model Zoo

updated 16 days ago

Collection of pre-trained & custom models for object detection, image segmentation, classification and more.

`akridge/yolo11-fish-detector-grayscale`
Object Detection • Updated 15 days ago • 78 • 1

`akridge/yolo11m-sea-urchin-detector`
Object Detection • Updated 21 days ago • 54

`akridge/yolo11-segment-fish-grayscale`
Image Segmentation • Updated 15 days ago • 23 • 1

NOAA FISHERIES

Community is Important: HuggingFace

Models

🌊Urchin Model 💾224

🐟Fish Models 💾272

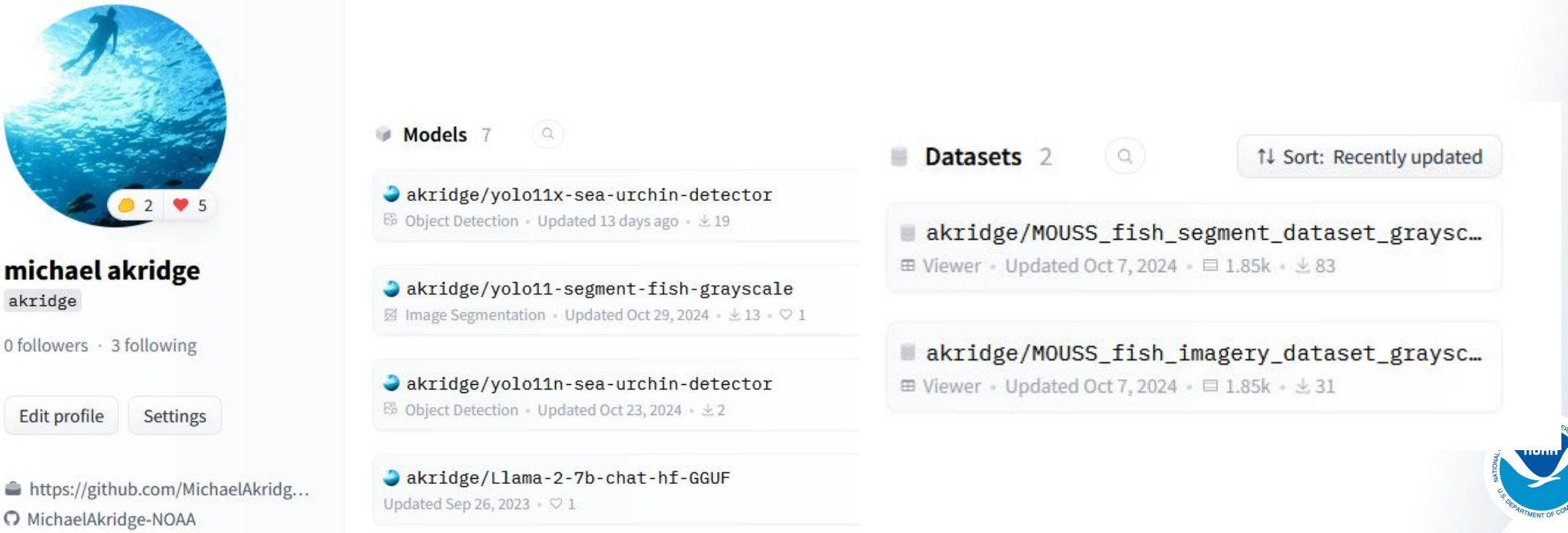
Total: 496 Downloads

AI-Ready Datasets

🔍Object Detection 💾1,480

🌊Segmentation 💾1,681

Total: 3,161 Downloads



NOAA FISHERIES

Thanks!

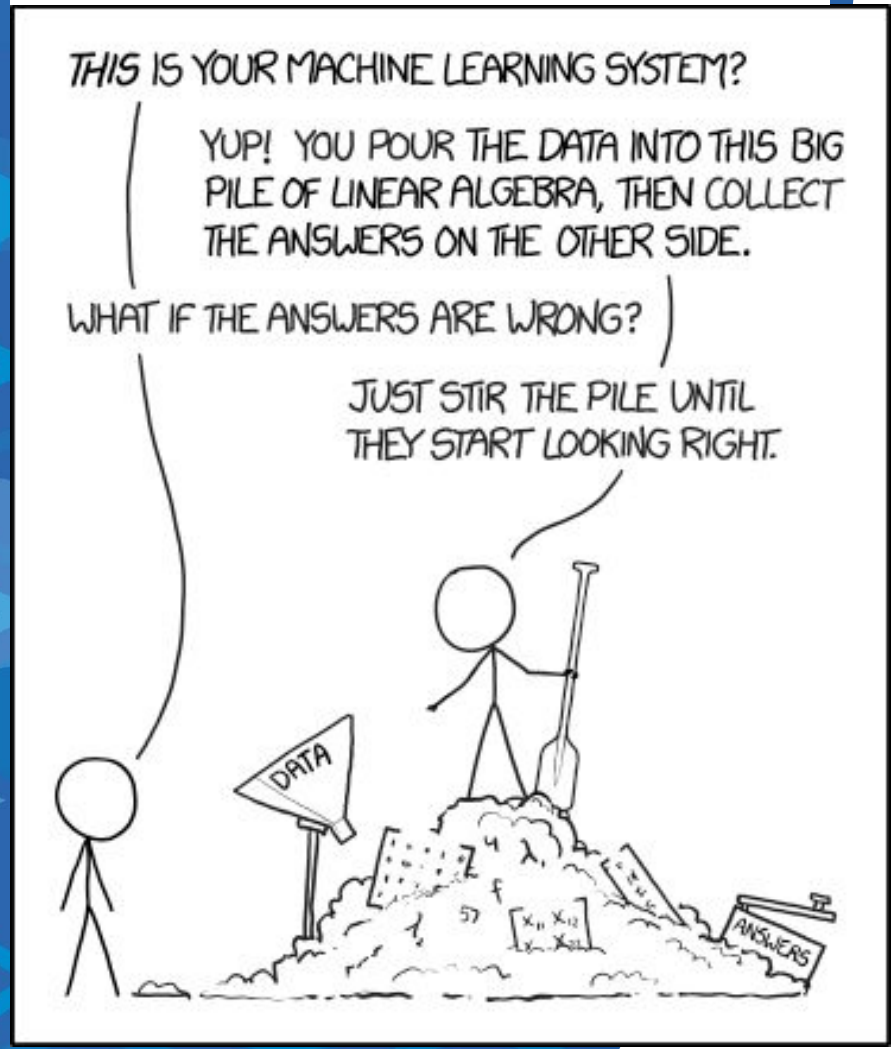
Links:

- YOLO Library:
  - https://github.com/ultralytics/ultralytics
- Label Studio
  - https://github.com/HumanSignal/label-studio
- Toolkit:
  - https://github.com/MichaelAkridge-NOAA/open-science-ai-toolkit
- Models:
  - https://huggingface.co/akridge
- **Welcome to the Community Computer Vision Course**
- **Cloud Data:**
  - **List of NOAA Open Data**
  - **PIFSC Cloud Console URL**: Google Cloud Storage Browser

MAKER Lab

NMFS OPEN SCIENCE

FATHOMNET



https://xkcd.com/1838/

NOAA FISHERIES