NOAA Fisheries Steller Sea Lion Population Count

Lam Kin Ho*, Chih-Jou Huang*, Dong-You Tsou*, Ping-Yeh Chou*
Dept. of Computer Science
National Yang Ming Chiao Tung University
Hsinchu, Taiwan

Abstract

*We address the challenge of automated Steller sea lion population counting from aerial imagery, a task vital for ecological monitoring yet hindered by dense crowds, occlusions, and varied object scales. Existing detection and segmentation approaches struggle in this domain due to frequent overlaps and annotation inconsistencies. In this work, we present **YouCount**, a robust YOLO11-based pipeline, complemented by a novel Adaptive Bounding Boxes Coordinates Annotation Framework (**ABC**). Our method adaptively adjusts bounding boxes and employs dual patch generation strategies to effectively capture dense and overlapping instances. Extensive experiments on the NOAA Fisheries Steller Sea Lion Population Count dataset demonstrate the superiority of our approach, achieving an RMSE of **11.65312** and securing **2nd** place in the Kaggle competition. Ablation studies further validate the effectiveness of our adaptive annotation and training strategies. Our code is available at https://github.com/cloud-peterjohn/Counting-Sea-Lions.git.*

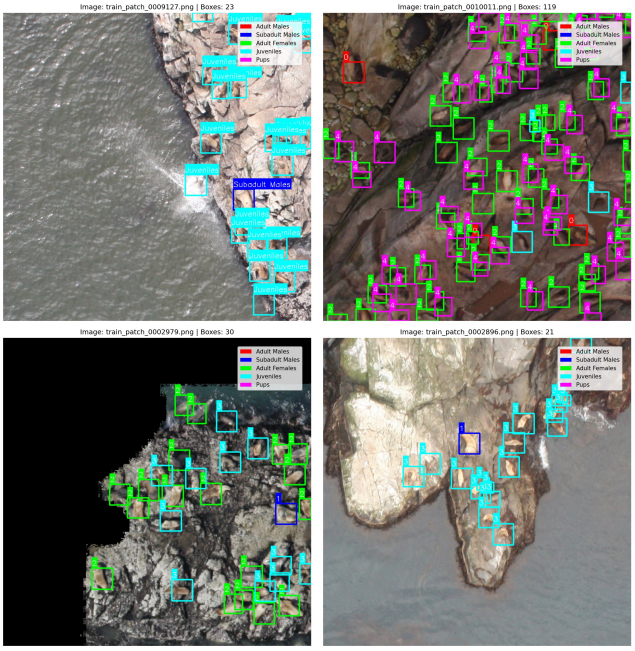


Figure 1. Visualization of Our Method.

1. Introduction

The population of Steller sea lions [3] in the western Aleutian Islands has declined dramatically, prompting urgent conservation efforts. Accurate population estimates are crucial for effective conservation, enabling scientists to better understand the factors contributing to the lack of recovery of Stellers in certain area. However, counting sea lions requires specially trained scientists and is an extremely time-sensitive task.

Automating this process with advanced computer vision [2, 12, 13] techniques improves monitoring efficiency, freeing valuable resources for conservation while benefiting endangered species. Previous approaches have primarily relied on detection-based [22, 24] and segmentation-based [20, 21] methods. Detection-based methods, on the other hand, employ Fast-RCNN-like [5] networks to predict bounding boxes for counting task. However, aerial images present significant challenges due to overlapping and clustered sea lions, leading to frequent missed detections. Segmentation-based methods, such as UNet [7, 9], moderately alleviates certain aspects of the overlap problem. Nevertheless, current segmentation-based methods consistently count the number of mask center points exceeding the confidence threshold, which is highly susceptible to segmentation instability, leading to counting deviations. To address these limitations, we propose a **YO**LO-11 based **Count**ing methods, named **YouCount**. Moreover, we present an **A**daptive **B**ounding Boxes **C**oordinates Annotation Framework, named **ABC**. Our method consists of two steps. We first annotate the dataset, utilizing ABC to adaptively adjust bounding box sizes and segment images into

*Equal contribution.

patches. For model training, we employ YOLO11 and integrate various data augmentation techniques to enhance robustness. Finally, we count the sea lion instances predicted by the YOLO model that exceed the confidence threshold.

In our experiments, we achieved an RMSE of 11.65312 and secured **2nd** place in the Kaggle competition. We further compare our method with various detection and segmentation approaches,demonstrating its superior performance. Additionally, we conducted ablation studies on the data pre-processing method, ABC, demonstrating the effectiveness of our annotation approach.

2. Related Work

2.1. Annotation Methods

The training dataset provided by NOAA Fisheries Alaska Fisheries Science Center consists of raw images alongside corresponding annotated images, where different sea lion categories are distinguished by uniquely colored dots. Previous segmentation approaches treated each annotation point as the center of a fixed-radius circle to generate a mask, while detection methods employed fixed-size squares centered at annotation points as bounding boxes. However, these methods overlooked significant variations in sea lion size, leading to inconsistent performance. In densely populated areas, fixed-size annotations frequently result in excessive overlap, further complicating identification. Moreover, most sea lions are neither perfectly circular nor square-shaped—their actual forms are better approximated by rectangles or ellipses, making traditional annotation methods suboptimal.

2.2. Counting Methods

Previous methods have primarily relied on object detection or instance segmentation approaches.

Object detection methods [15, 23], utilized Fast-RCNN with VGG-16 [19] backbones or ResNet-50 [6] backbones to predict bounding boxes and confidence scores for individual sea lions, with counting based on bounding boxes exceeding a predefined confidence threshold. However, aerial images often contain densely packed sea lions with extensive overlap, leading to significant bounding box intersections. Moreover, variations in image capture height and location introduce inconsistencies in sea lion spatial distributions and size differences, causing biases and imbalances in training and inference.

Instance segmentation methods [1, 11] employed UNet to predict masks centered at sea lion coordinates for counting. This paradigm mitigates the overlap issues caused by fixed-size bounding boxes, improving training stability and result accuracy. However, the variability in sea lion postures can still lead to inaccurate center predictions and instability in the results.

Furthermore, both segmentation and detection approaches share several common issues:

- Variations in sea lion size across images cause inconsistent performance.
- Pups are often too small to detect reliably.
- Densely packed sea lions make individual identification difficult.
- Distinguishing between different sea lion categories also remains a persistent challenge.

3. Methods

3.1. Preliminary

3.1.1. YOLO

You Only Look Once (YOLO) is a pioneering real-time object detection algorithm introduced by Redmon et al. in [16]. Unlike traditional two-stage detectors that first generate region proposals and then classify each proposal, YOLO frames object detection as a single regression problem. It directly predicts bounding boxes and class probabilities from the entire input image in one forward pass of a neural network. This design enables YOLO to achieve remarkable inference speeds while maintaining competitive accuracy, making it suitable for applications requiring fast and efficient detection.

In this study, we adopted YOLO11 [18], developed by the Ultralytics team, as the primary architecture for object detection. YOLO11 is the latest evolution in the YOLO series, building upon the lightweight, real-time, and multi-task capabilities of its predecessors such as YOLOv8 [8]. YOLO11 introduces an enhanced architecture with improvements to both the backbone and neck, significantly boosting its feature extraction capabilities. This makes it particularly effective in handling complex scenarios such as multi-object detection. Furthermore, the YOLO11m variant achieves higher mean Average Precision (mAP) [4] on the COCO dataset [10] while reducing the number of parameters by approximately 22% compared to YOLOv8m.

3.2. Counting Sea Lions

Our methods is outlined in Figure 2

3.2.1. ABC

To localize sea lions in high-resolution images, we first generate initial bounding boxes (bboxes) by drawing a square centered at each annotated point. The side length of these squares is fixed initially. Next, we compute the Intersection over Union (IoU) between every pair of bboxes to determine the scaling factor for each bbox,following standard object detection practice [17]. A minimal allowable dimension is set to prevent bboxes from becoming too small. Based on the horizontal and vertical distances between pairs of points, we also decide the direction in which each bbox should be resized.

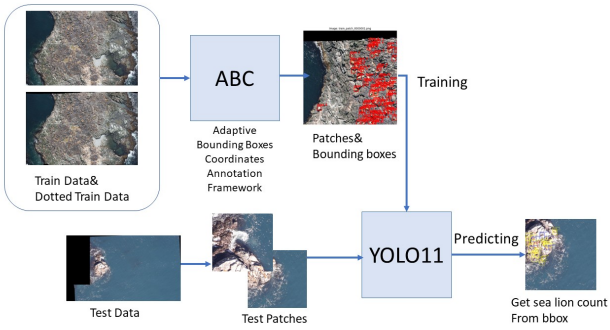


Figure 2. Overview of our methods.

Given the large size of the original images, patch extraction is necessary for model training. We adopt two patching strategies:

1. **Overfitting-based Patching**
   This strategy samples patches centered around bboxes. The number of patches is set to 30% of the total number of bboxes in the image. The number of patches is constrained to be between 1 and 3, and must not exceed the total number of bboxes. Then, a random subset of bboxes is selected as the centers for cropping patches.
2. **Tiling-based Patching**
   In this method, we tile the image using a fixed patch size and extract patches by sliding a window over the image. The stride is set to 95% of the patch size, allowing adjacent patches to overlap. For each grid cell, a patch is extracted using the center of the cell as the reference point.

For both methods, patches are cropped from the image, and the corresponding bounding boxes are adjusted so they remain within the boundaries of each patch. If a bbox is clipped such that its area becomes less than 20% of its original area, it is discarded.

Additionally, because the dotted images contain large black regions that obscure visual information, we compute the black pixel ratio within each patch. If this ratio exceeds 0.5, indicating that more than half of the patch area is visually unusable, the patch is discarded.

Finally, all valid patches and their associated bboxes are rescaled to match the target patch size. Specifically, the original patch size is 1280, and the target patch size is 640, resulting in a downscaling factor of 0.5.

3.2.2. YouCount

During training, annotated images from the dataset—including bounding boxes and class labels for sea lions—are input into the YOLO model. The model performs multiple layers of convolution and feature extraction, simultaneously predicting object locations and classes across multiple anchor boxes. Various data augmentation techniques are applied during training to simulate different lighting conditions, perspectives, and object scales. The training process runs over multiple epochs, with a learning rate scheduler gradually reducing the learning rate to promote stable convergence[16].

In the inference stage, each image patch (originally sized 1440×1440) is resized to 640×640 to match the input requirements of the YOLO model. The model extracts features and predicts bounding boxes with associated confidence scores and class probabilities. Non-Maximum Suppression (NMS) is then applied to filter out highly overlapping predictions, keeping only the most confident ones [14].

The final output includes the predicted locations and classes of detected objects. After obtaining the bounding boxes, we further refine the object counts by applying a weighting mechanism based on proximity to patch boundaries: if a detected object is near one edge, its count is weighted by 0.7; if it is near two or more edges, the weight becomes 0.5. This approach mitigates the overcounting of objects that span across multiple image patches and improves the overall counting accuracy. The estimated counts from all patches belonging to the same original image are then aggregated to produce the final count of each sea lion class per image.

To further improve the final counting accuracy, we adopted a simple post-processing step inspired by the top-ranked solutions [11, 15]). Specifically, we applied category-wise scaling factors to the predicted counts based on observed bias patterns. The adjustments were as follows:

`pups` $\times 1.3$, `juveniles` $\times 0.85$, `adult_females` $\times 0.96$, `adult_males` $\times 1.55$, and `subadult_males` $\times 1.2$.

4. Experiments

4.1. Experiments on Counting Sea Lions

4.1.1. Experiment Setups

We utilize the dataset from the NOAA Fisheries Steller Sea Lion Population Count competition, which comprises the following components:

- `Train/*.jpg`: 946 training images, each filename corresponds to a train_id
- `TrainDotted/*.jpg`: Annotated versions of training images showing the location of each animal
- `Test/*.jpg`: Test image collection

The objective is to count the number of sea lions in 5 distinct classes: Adult males, Subadult males, Adult females, Juveniles, and Pups.

In our main experiments, we adopt the following training configurations in Table 1.

Table 1. Hyperparameter Configuration

Category	Parameter	Value	Description
Data Augment	Hue Adjustment	±1.5%	Adjusts image hue, affecting color balance
	Saturation Adjustment	±70%	Modifies image saturation, enhancing or reducing color intensity
	Brightness Adjustment	±40%	Alters image brightness, increasing or decreasing overall luminance
	Rotation Angle	±3°	Randomly rotates the image to improve robustness to viewpoint changes
	Translation Scale	±2%	Randomly shifts the image to simulate different perspectives
	Scaling Range	±30%	Resizes the image to improve adaptability to objects of varying sizes
	Copy-Paste Probability	8%	Copies and pastes objects to increase data diversity
	Random Erasing Probability	8%	Randomly removes image regions to enhance model robustness
	MixUp Probability	4%	Combines images for augmentation, improving generalization
Model Train	Batch Size	2	Number of samples per training batch
	Epochs	100	Number of times the model iterates over the dataset
	Confidence Threshold	0.5	Minimum confidence required for object detection
	NMS IoU Threshold	0.7	Intersection-over-Union threshold for Non-Maximum Suppression
	Image Size	640×640	Fixed input image dimensions

4.1.2. Result and Analysis

In this project, we evaluated our sea lion counting model using Root Mean Squared Error (RMSE) on both the public and private test sets provided by the competition. On the public test set, our model achieved an RMSE of **11.68292**, while on the private test set, it obtained a slightly lower RMSE of **11.65312**.

Figure 3(a)(d) shows the predicted bounding boxes over the entire image. The model can detect individual sea lions accurately, even in crowded regions where multiple sea lions are partially overlapping. Notably, the model successfully distinguishes between adjacent individuals without merging them into a single bounding box, demonstrating its robustness in handling occlusion. In addition, the detector can identify pup instances, which are typically smaller and more visually ambiguous due to their size and color similarity with the background. These results suggest that the model performs well across different density levels and can handle fine-grained instance separation.

The detection density heatmap Figure 3(b) largely aligns with the actual spatial distribution of sea lions in the original image. We observe no significant false positive hotspots in background regions such as ocean, indicating that the model has learned to localize detections to semantically meaningful areas. It provides reassuring evidence that the model's spatial prediction behavior is consistent and well-grounded.

Figure 3(c) presents the detection distribution and confidence score distribution of sea lion categories for a sample image (Image 163): The left chart is a bar plot showing the number of detections per sea lion category in the image. Among these, "adult_females" is clearly the most detected category (over 250 instances), while "subadult_males" has very few detections. This may reflect the actual lower presence of this category in the image.

The right chart is a histogram of confidence scores for the detected bounding boxes. The x-axis represents the confidence score assigned by the model, and the y-axis shows the number of detections within each score interval. Most detections are concentrated between 0.5 and 0.65, with an average confidence score of 0.52. This suggests that the model's overall detections are of moderate confidence, indicating potential instability in some predictions.

Table 2. Class Distribution between Total Training Data

cls	Male	Subadult	Female	Juvenile	Pup
train	5392	4345	37537	20118	16285

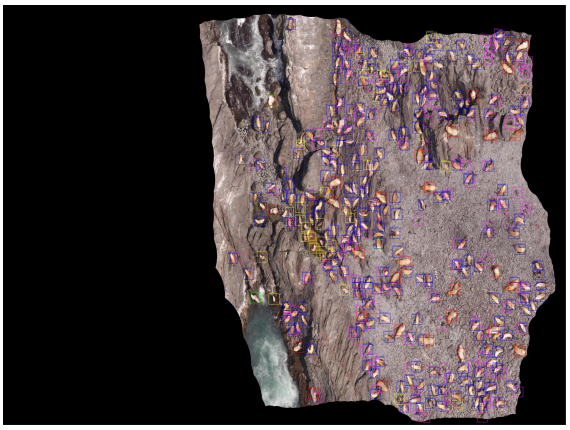
4.2. Ablation Experiments

4.2.1. Ablation Study on Resolution and Post-Processing

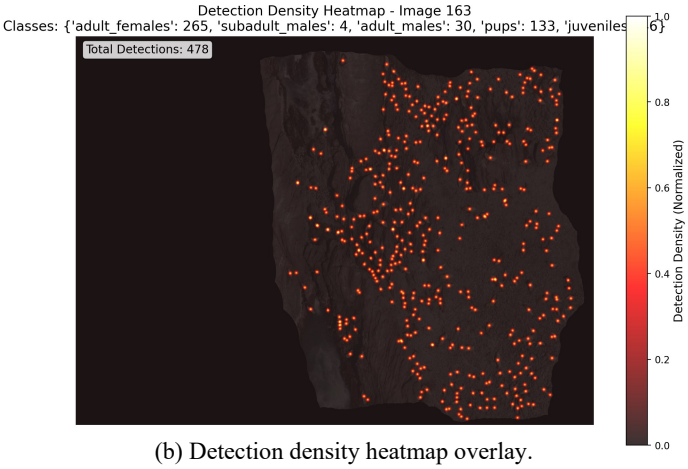
Table 3. Ablation study on input resolution and post-processing.

Test Data Scale	Post-processing	RMSE
640/1280	No	16.59
640/1440	No	12.45
640/1440	Yes	11.65

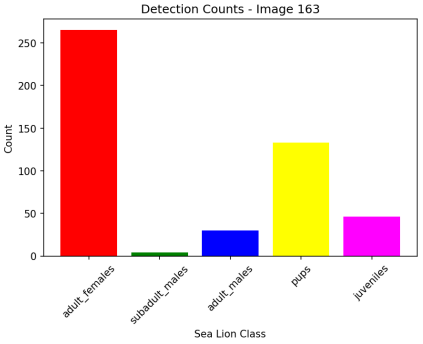
In our initial experiment, we set the test patch size to 1280 pixels when cropping patches from the original images, and then resized these patches down to 640 pixels. In a subsequent experiment, we increased the patch size to 1440 pixels, while still resizing to 640 pixels.The training data remained unchanged throughout these experiments. This effectively increased the downscaling ratio. As a result, the RMSE improved by approximately 4.1. We hypothesize that the test data images were taken at closer distances to the sea lions, causing the animals to occupy a larger proportion of each patch. By increasing the patch size before

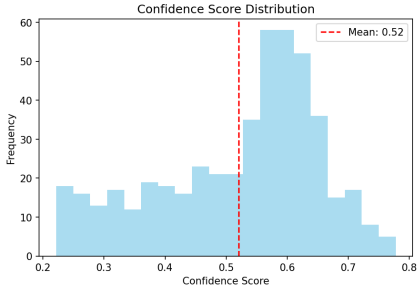


(a) Full-image bounding box visualization.

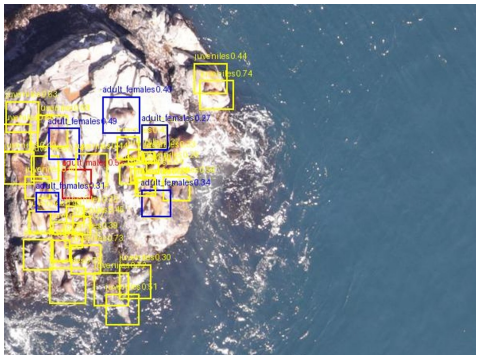


(b) Detection density heatmap overlay.





(c) Class distribution bar chart for a sample image.



(d) Patch-level bounding box visualization.

Figure 3. Visualization examples: (a) Full-image bounding box results, (b) Detection density heatmap, (c) Per-image class statistics, (d) Patch-level bounding box results.

resizing, the relative size of sea lions in the resized patches from both train and test data became more consistent, leading to more accurate predictions.

Additionally, many competing teams applied post-processing techniques such as ranking adjustments (e.g., rank1 [15] and rank2 [11]). Following this idea, we experimented with adjusting the class-wise prediction ratios through post-processing, which further reduced the RMSE by about 0.8. We infer that fine-tuning the relative proportions of detected classes helps correct biases in the model's outputs, leading to improved overall counting accuracy.

5. Conclusion

In this paper, we presented YouCount, a robust and efficient pipeline for automated Steller sea lion counting from aerial imagery. Through our novel Adaptive Bounding Boxes Coordinates Annotation Framework (ABC), we introduced a principled approach for handling dense and overlapping instances, while our dual patch generation strategy ensured comprehensive data coverage. Leveraging the YOLO11x architecture with tailored augmentation and rigorous quality control, our method achieved a strong RMSE of 11.65312, earning 2nd place in the Kaggle competition. Extensive experiments and ablation studies validate the effectiveness of both our adaptive annotation and training strategies. We believe our approach provides a solid foundation for wildlife monitoring and can be readily generalized to similar ecological counting tasks.

NOAA Fisheries Steller Sea Lion Population Count

Supplementary Material

6. Experiment Details

6.0.1. Dataset Description

Competition Dataset We utilize the official dataset from the NOAA Fisheries Steller Sea Lion Population Count competition, which comprises the following components:

- `Train/*.jpg`: 946 training images, each filename corresponds to a train_id
- `TrainDotted/*.jpg`: Annotated versions of training images showing the location of each animal
- `Test/*.jpg`: Test image collection
- `coords-threeplusone-v0.4.csv`: Coordinate annotation file
- `sample_submission.csv`: Example of a valid submission file

Classification Task The objective is to count the number of sea lions in 5 distinct classes:

1. Adult males
2. Subadult males
3. Adult females
4. Juveniles
5. Pups

6.0.2. Environment Setup

Virtual Environment To ensure reproducibility, we create a dedicated virtual environment for this project:

```
python -m venv sealion-counting
source sealion-counting/bin/activate
```

Software Dependencies The following packages are installed in the virtual environment:

Table 4. Software Environment Configuration

Framework/Library	Version
PyTorch	2.0.1
Ultralytics	8.0.196
OpenCV	4.8.0
NumPy	1.24.3
Pandas	2.0.3
Matplotlib	3.7.1
Pillow	10.0.0
Python	3.8+

6.0.3. Training Configuration

We adopt the following training configuration to optimize model performance:

Table 5. Training Hyperparameter Settings

Parameter	Value
Batch Size	2
Epochs	100
Confidence Threshold	0.5
NMS IoU Threshold	0.7
Image Size	640×640
GPU Memory Requirement	8 GB

6.0.4. Data Augmentation Strategy

The experiment employs comprehensive data augmentation techniques to enhance model generalization:

Table 6. Data Augmentation Parameter Configuration

Augmentation Technique	Parameter Range
Hue Adjustment	±1.5%
Saturation Adjustment	±70%
Brightness Adjustment	±40%
Rotation Angle	±3°
Translation Scale	±2%
Scaling Range	±30%
Copy-Paste Probability	8%
Random Erasing Probability	8%
MixUp Probability	4%

6.0.5. Visualization Features

The system implements comprehensive visualization capabilities to aid in result analysis and quality assessment:

Bounding Box Visualization

- **Color-coded boxes** for each sea lion class:
  - **Adult Males**: Red (RGB: 255, 0, 0)
  - **Subadult Males**: Green (RGB: 0, 255, 0)
  - **Adult Females**: Blue (RGB: 0, 0, 255)
  - **Juveniles**: Yellow (RGB: 255, 255, 0)
  - **Pups**: Magenta (RGB: 255, 0, 255)
- **Confidence scores** displayed on each detection
- **Individual patch visualizations** and full-image reconstructions

Detection Density Heatmaps

- **Gaussian-smoothed density maps** showing detection concentrations
- **Background image overlays** for context
- **Enhanced scaling** techniques for small detection counts:
  - Higher resolution processing (2×) with downsampling

  - Gaussian influence area for each detection (20-pixel radius)
  - Power scaling (square root) to enhance low values
  - Normalized visualization (0-1 range)
- **Colorbar labels** and detection count information

Statistical Visualizations
- **Per-image statistics**:
  - Class distribution bar charts
  - Confidence score histograms
  - Detection count summaries
- **Global statistics**:
  - Overall class distribution
  - Confidence score patterns
  - Detection density analysis

6.0.6. Results and Analysis

Performance Metrics The system's performance is evaluated using:
- **Count accuracy**: Comparison between automated and manual counts

Kaggle Competition Performance:
- **Final Score**: 11.65312 RMSE
- **Public Leaderboard**: 11.68292 RMSE
- **Final Rank**: 2nd place

Detailed information can be found in Figure 4

Table 7. Competition Leaderboard Results

Rank	Score
1st	10.85644
2nd (Ours)	**11.65312**
3rd	12.50888
4th	13.03257
5th	13.18968
6th	13.92760

We visualize our results, as shown in Figure 5 6 7

Visualization Insights The visualization capabilities reveal several key insights:
- **Clustering patterns** of different sea lion classes
- **Confidence distribution** across detection types
- **Spatial relationships** between sea lion groups
- **Detection density** variations across images

Submission and Description	Private Score	Public Score	Selected
final_submission.csv Complete (after deadline) · now	11.65312	11.68292	☐

Figure 4. Results of Kaggle Competition.

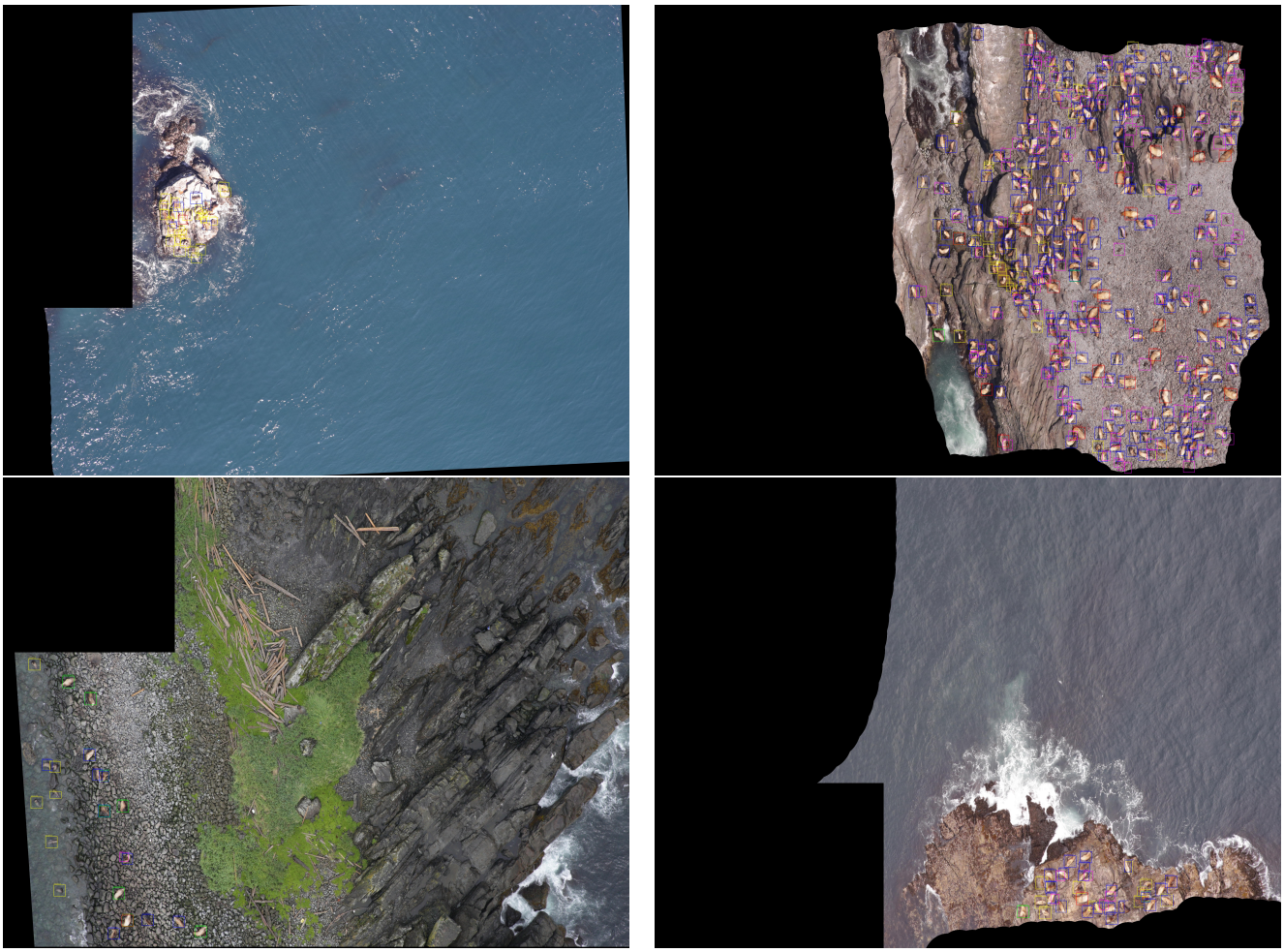


Figure 5. Visualization of full-image bounding box results.

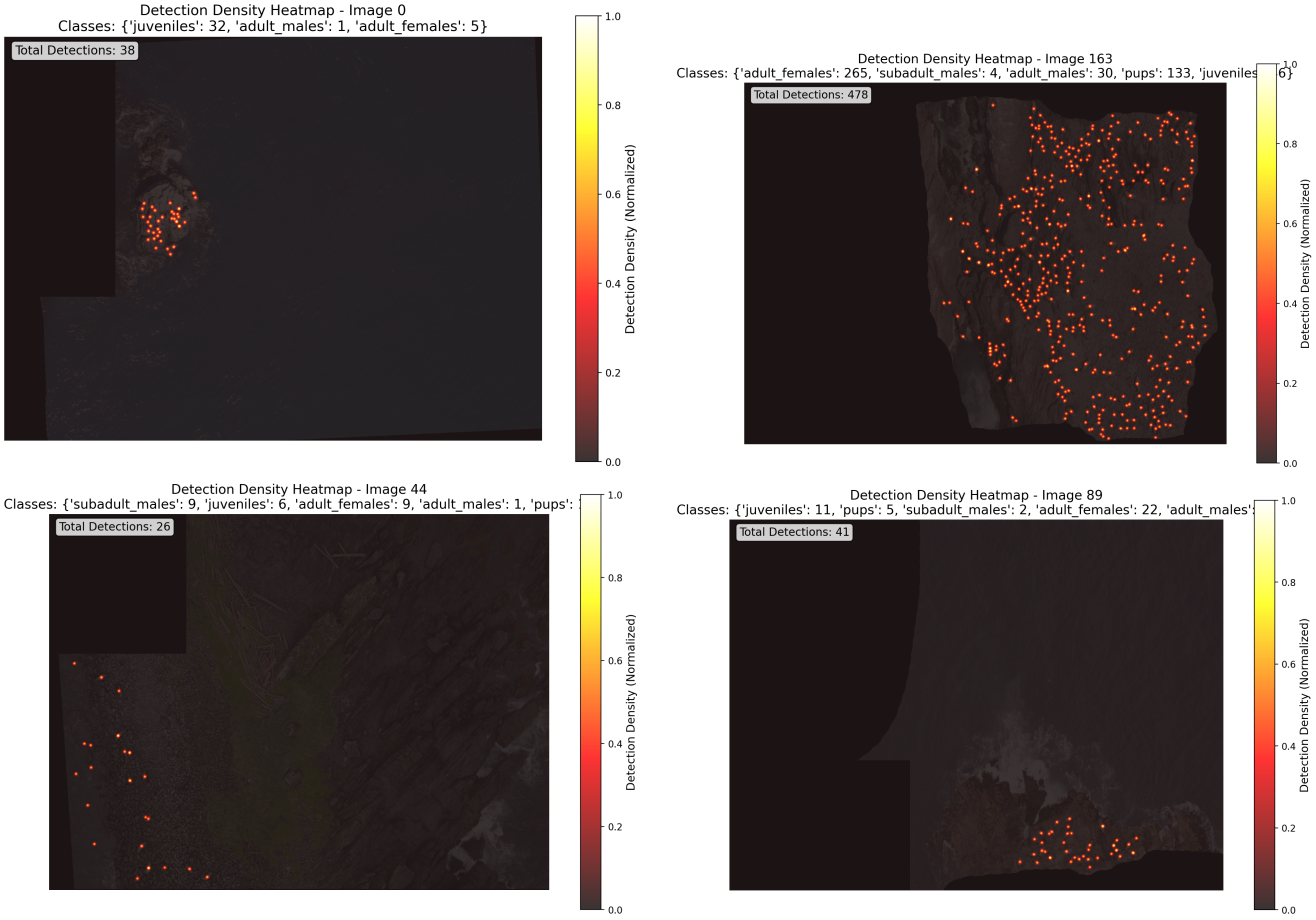


Figure 6. Visualization of detection density heatmap.

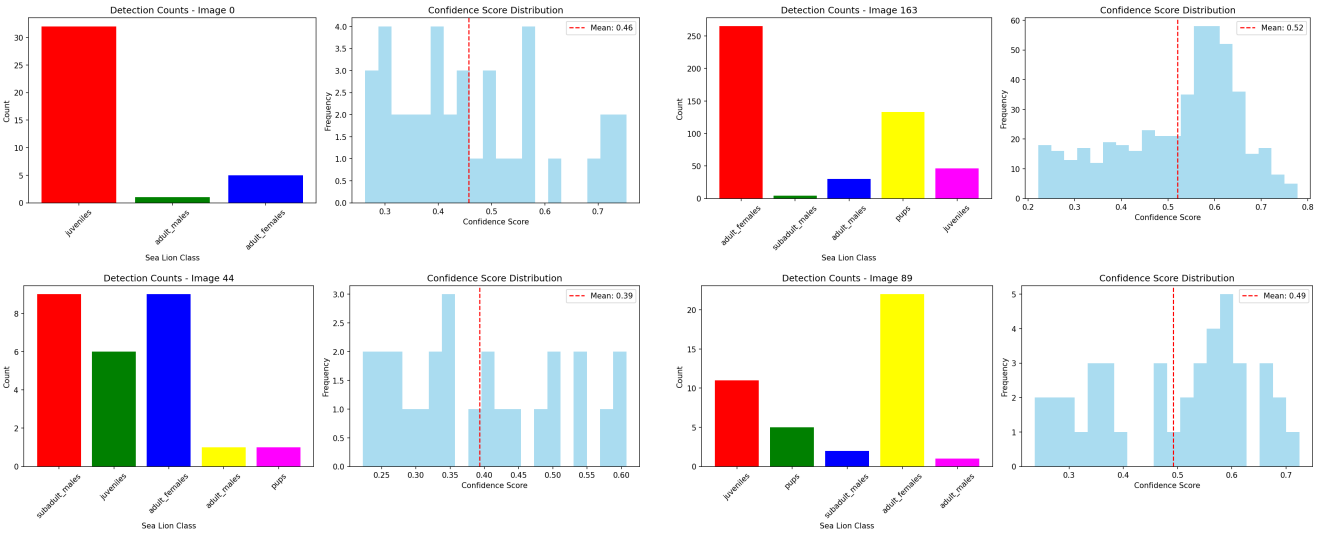


Figure 7. Visualization of per-image class statistics.

References

[1] Bestfitting. Segment and count solution for kaggle noaa fisheries steller sea lion population count, 2017. 3rd Place Solution using Star-UNet and Outline-UNet. 2

[2] Prithvijit Chattopadhyay, Ramakrishna Vedantam, Ramprasaath R Selvaraju, Dhruv Batra, and Devi Parikh. Counting everyday objects in everyday scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1135–1144, 2017. 1

[3] DataCanary, Katie, and Meg Risdal. Noaa fisheries steller sea lion population count. https://kaggle.com/competitions/noaa-fisheries-steller-sea-lion-population-count, 2017. Kaggle. 1

[4] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010. 2

[5] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015. 1

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 2

[7] Huimin Huang, Lanfen Lin, Ruofeng Tong, Hongjie Hu, Qiaowei Zhang, Yutaro Iwamoto, Xianhua Han, Yen-Wei Chen, and Jian Wu. Unet 3+: A full-scale connected unet for medical image segmentation. In *ICASSP 2020-2020 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 1055–1059. IEEE, 2020. 1

[8] Glenn Jocher. Yolov8: Ultralytics yolov8 documentation. https://docs.ultralytics.com, 2023. Accessed: 2025-06-02. 2

[9] M Krithika Alias AnbuDevi and K Suganthi. Review of semantic segmentation of medical images using modified architectures of unet. *Diagnostics*, 12(12):3064, 2022. 1

[10] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context, 2015. 2

[11] Konstantin Lopukhin. Counting sea lions using unet and regression on masks, 2017. 2nd Place Solution for Kaggle NOAA Fisheries Steller Sea Lion Population Count. 2, 3, 5

[12] Sergiu Mezei and Adrian Sergiu Darabant. A computer vision approach to object tracking and counting. *Studia Universitatis Babes-Bolyai, Informatica*, 55(1), 2010. 1

[13] Hamam Mokayed, Tee Zhen Quan, Lama Alkhaled, and V Sivakumar. Real-time human detection and counting system using deep learning computer vision techniques. In *Artificial Intelligence and Applications*, pages 205–213, 2023. 1

[14] Andreas Neubeck and Luc Van Gool. Efficient non-maximum suppression. In *18th International Conference on Pattern Recognition (ICPR)*, pages 850–855. IEEE, 2006. 3

[15] OutRunner. Use keras to count sea lions. Kaggle Notebook, 2017. Top-ranked solution in Kaggle competition NOAA Fisheries Steller Sea Lion Population Count. 2, 3, 5

[16] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016. 2, 3

[17] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized intersection over union: A metric and a loss for bounding box regression, 2019. 2

[18] Ultralytics Team. Yolo11: The next generation of yolo models. https://docs.ultralytics.com/models/yolo11/, 2024. Accessed: 2025-06-01. 2

[19] Andrea Vedaldi and Andrew Zisserman. Vgg convolutional neural networks practical. *Department of Engineering Science, University of Oxford*, 66, 2016. 2

[20] Xiaogang Wang et al. Deep learning in object recognition, detection, and segmentation. *Foundations and Trends® in Signal Processing*, 8(4):217–382, 2016. 1

[21] Hao Wu, Qi Liu, and Xiaodong Liu. A review on deep learning approaches to image classification and object segmentation. *Computers, Materials & Continua*, 60(2), 2019. 1

[22] Youzi Xiao, Zhiqiang Tian, Jiachen Yu, Yinshu Zhang, Shuai Liu, Shaoyi Du, and Xuguang Lan. A review of object detection based on deep learning. *Multimedia Tools and Applications*, 79:23729–23791, 2020. 1

[23] Yingxiaowei. Faster r-cnn for kaggle noaa fisheries steller sea lion population count, 2017. 6th Place Solution for Kaggle NOAA Fisheries Steller Sea Lion Population Count. 2

[24] Zhong-Qiu Zhao, Peng Zheng, Shou-tao Xu, and Xindong Wu. Object detection with deep learning: A review. *IEEE transactions on neural networks and learning systems*, 30 (11):3212–3232, 2019. 1