Development of the Delfi Project: Converting Classification to Detection for Zooplankton Analysis

[Kananat Siangsano]

February 13, 2025

1 Introduction

The Delfi project faced an initial challenge where the available dataset was structured for classification rather than detection tasks. The original dataset consisted of PNG images organized by species folders, which presented several limitations for our intended detection-based approach. The primary challenge stemmed from the fact that the classification dataset contained pre-cropped individual organisms, lacking crucial information about their original positions within the raw image files.

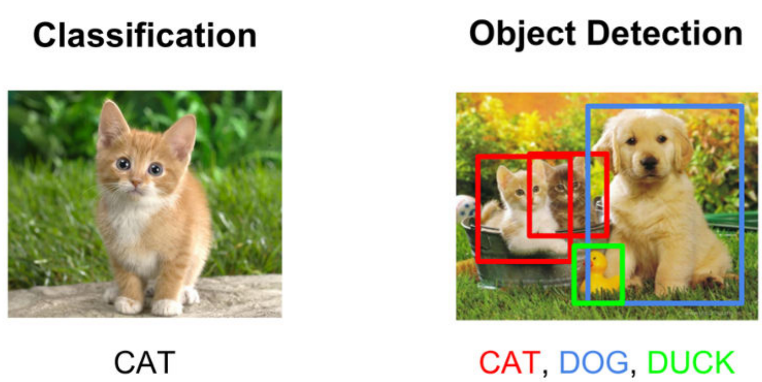

Figure 1: Comparison between classification (Left) and detection (Right) tasks. (1).

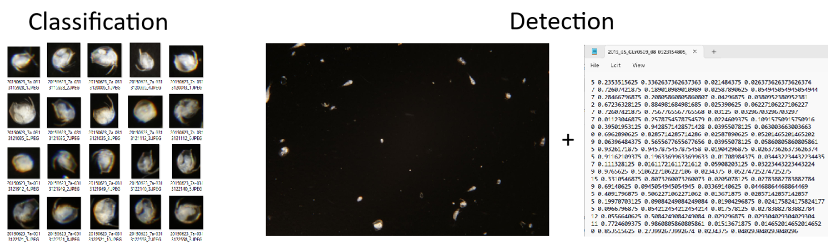

Figure 2: Dataset structure comparison. Classification datasets (Left) organize images by class folders, while detection datasets (Right) maintain image-label pairs with coordinate information.

2 Methodology

2.1 Dataset Synthesis

To address the dataset limitation, we developed a Python-based solution to synthesize a detection dataset from the existing classification data. The process involved:

- Placing cropped organisms on blank black backgrounds
- Implementing dynamic rotation and size adjustments for each organism
- Recording precise coordinate placement information
- Generating labeled detection dataset with position data

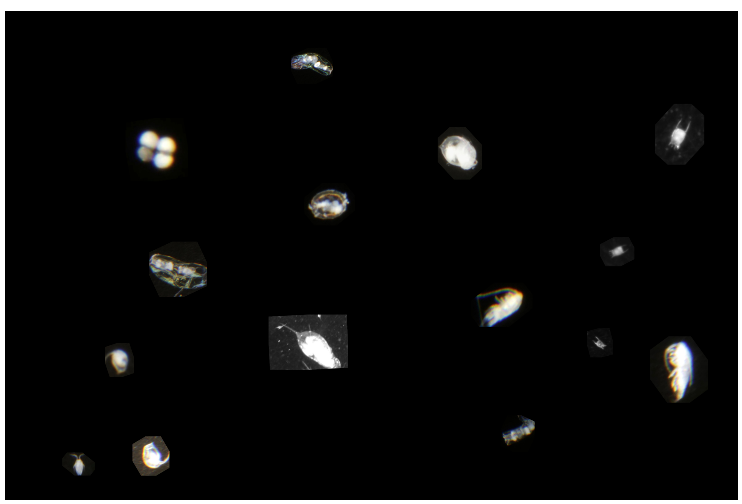

Figure 3: Example of the synthesized image

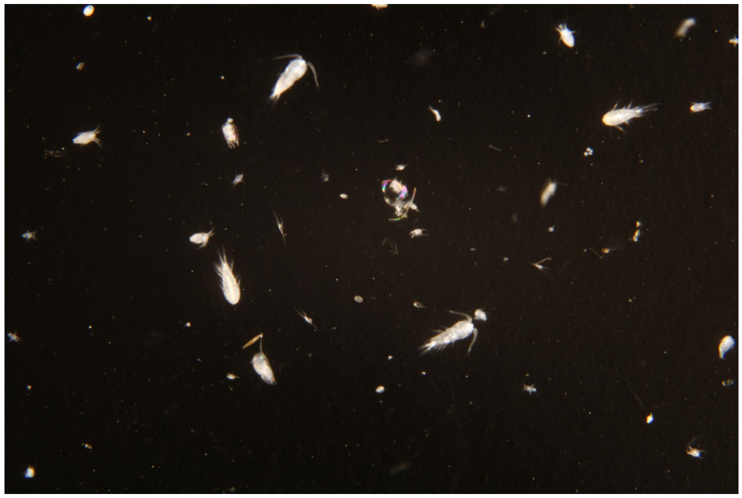

Figure 4: Example of the raw image

This approach produced synthetic images that approximated raw image characteristics, though differences remained in background consistency, particularly regarding irregular features such as light gradients present in raw images.

2.2 Raw Image Preprocessing

To bridge the gap between synthetic and raw images, we implemented a preprocessing pipeline for the raw dataset. This process aimed to standardize background characteristics through the following algorithm:

1. Image segmentation into small subsections
2. Dynamic threshold value determination for each section
3. Background pixel identification using threshold values
4. Background normalization by setting identified pixels to 0

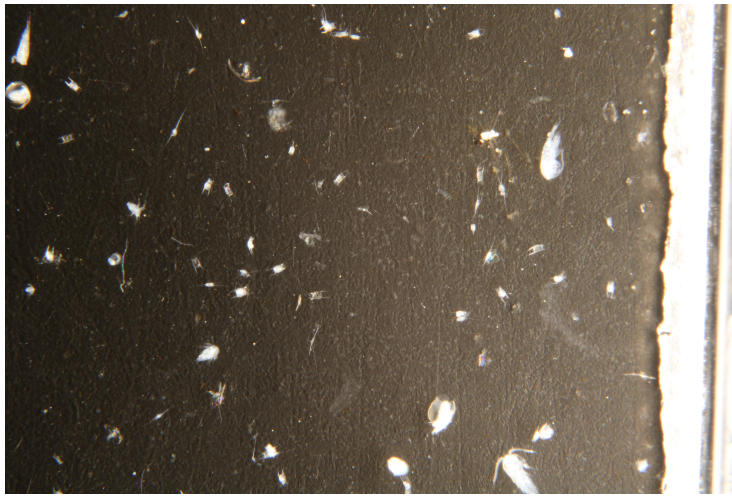

Figure 5: Raw image before preprocessing

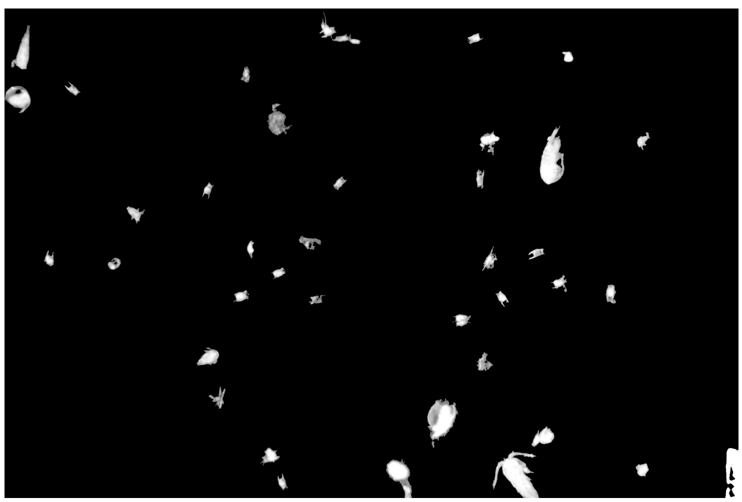

Figure 6: Raw image after preprocessing

This preprocessing resulted in images with uniform black backgrounds, more closely matching our synthetic training data.

2.3 Model Development and Active Learning

The development process followed several key stages:

1. Initial training of YOLOv11m using the synthesized dataset

2. Application of the trained model to preprocessed raw images
3. Generation of preliminary detection dataset from model predictions
4. Manual correction of the initial 100-image dataset
5. Implementation of an active learning cycle:
    - Model training on corrected dataset
    - Prediction generation on new raw images
    - Dataset expansion through prediction correction
    - Iterative model improvement through repeated training

3 Discussion

While starting the project with manual labeling of a detection dataset would have been ideal, our approach effectively utilized existing classification data to bootstrap the detection model development. The initial dataset, though requiring substantial correction, provided a foundation superior to starting from scratch. The implementation of active learning has proven effective for iteratively improving model performance, with each cycle enhancing the model's detection capabilities on raw images.

4 Conclusion

Through this methodology, we successfully transformed a classification-based dataset into a framework for detection-based analysis. The combination of synthetic data generation, preprocessing algorithms, and active learning has created a robust pipeline for zooplankton detection. While the current model's performance is limited by dataset size, the established framework allows for continuous improvement through iterative training cycles.

References

[1] ByteIota, "Object detection with TensorFlow," 2025, accessed: February 2025. [Online]. Available: https://byteiota.com/object-detection-tensorflow/