HW 1 - Final Report

Computer Vision - Surgical Applications

0970222

Noam Carmon 207234170
Shahar Hillel 207530551
Raz Biton 315507780

1 Exploratory Data Analysis

1.0.1 a. Image Visualization (with/without label)

We first look at the images to understand the data and the challenges involved in detection.

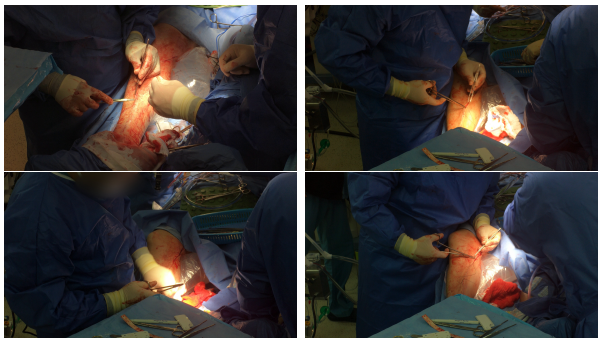

Figure 1: Sample training images (labeled and unlabeled examples).

To understand the data visually, we labeled images from the training set and overlaid the bounding box to better understand the annotations:

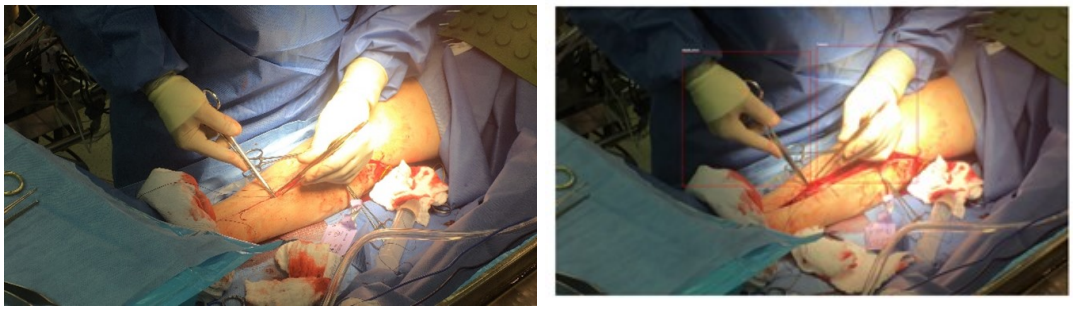

Figure 2: Sample training images (left: with labels, right: without labels).

1.0.2 Insights from Looking at the Data

After viewing several labeled images from the training set, we made the following observations:

- Most images are close-up views of leg suturing.
- Hands and tools are sometimes occluded or cut off, making detection harder.
- Lighting varies, some images are bright, while others have shadows or glare.
- Tools are often overlapping or in realistic positions, as in actual surgeries.

These observations suggest that, although the dataset is small, it captures realistic surgical variability—including occlusions, lighting differences, and visual clutter—which makes detection more challenging but may help the model generalize better to unseen surgical scenarios, such as the OOD (out-of-distribution) video.

1.0.3 Data Distribution Analysis

Training Data:

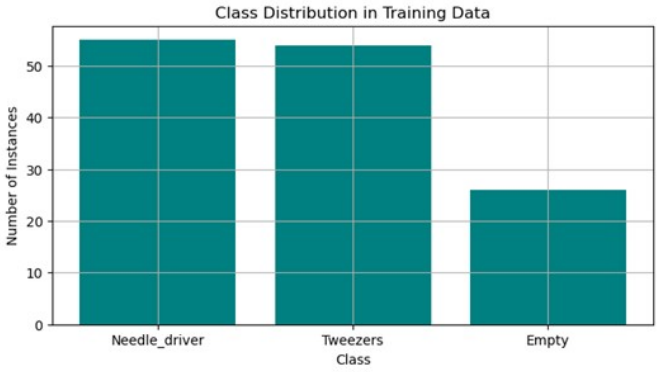

Figure 3: Class distribution in the training set.

- Total labeled images: 61
- Total bounding boxes: 135
- Average boxes per image: 2.21
- Most frequent class: `Needle_driver`
- Least frequent class: `Empty`

Validation Data:

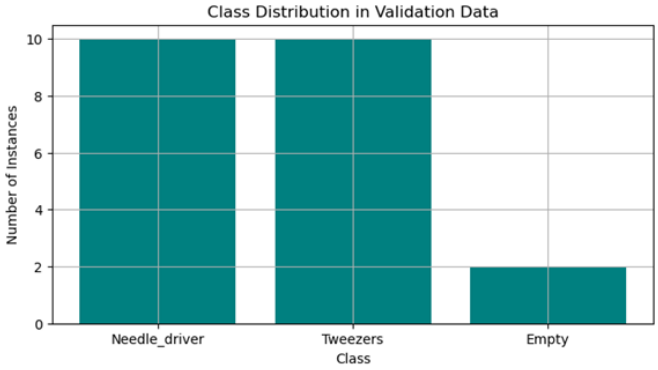

Figure 4: Class distribution in the validation set.

- Total labeled images: 10
- Total bounding boxes: 22
- Average boxes per image: 2.20
- Most frequent class: `Needle_driver`
- Least frequent class: `Empty`

2 Experiments

2.1 Baseline Model

Data Loading, Pre-processing, and Cleaning:
We used the provided labeled dataset. Data was loaded using the Ultralytics YOLO framework, which supports direct ingestion of YOLO-formatted annotations.

Training Techniques:
We trained a YOLOv8n model as our baseline. YOLOv8n was chosen for its lightweight architecture and fast training times, enabling quicker iterations. The initial training configuration was:

- Epochs: 50
- Image size: 650 pixels
- Batch size: 16
- Learning rate: 1e-3

This baseline served as a reference point for further optimization and pseudo-labeling.

Regularization:
To address potential overfitting, especially given the small dataset size, we incorporated L2 regularization via the `weight_decay` parameter. Various values were tested during hyperparameter tuning to assess their impact on generalization.

Hyperparameter Tuning:
We used Optuna, a framework for automated hyperparameter optimization. The tuning process involved running multiple short training trials (10 epochs each) to evaluate different configurations. The following ranges were explored:

- Learning rate: 0.00135
- Momentum: 0.925
- Weight decay: 0.00013
- Batch size: 8

Using the best configuration found, we trained the model for 75 epochs. This tuning significantly improved performance, as evaluated using mean Average Precision (mAP) on the validation set.

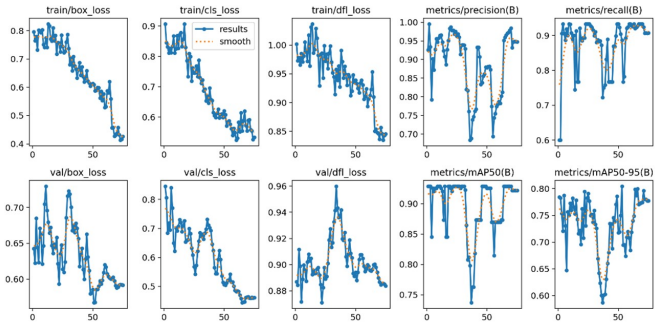

Figure 5: Training and validation losses and mAP's graphs.

2.2 Pseudo-Label Model (In-Distribution - ID)

For the semi-supervised phase, we utilized the in-distribution (ID) videos to expand our training dataset. Due to the high frame similarity in surgical videos, we subsampled the videos by selecting every 5^{th} frame to avoid redundancy and overfitting.

Pseudo-Label Generation:

We used the fine-tuned baseline YOLOv8n model to generate pseudo-labels on the ID video frames. To ensure label quality, we applied a filtering mechanism: only images in which all predicted bounding boxes exceeded a specified confidence threshold were retained. We evaluated several thresholds:

- Confidence thresholds tested: [0.5, 0.6, 0.7, 0.8]

Higher thresholds resulted in fewer training images due to stricter filtering. For example:

- At 0.5: 1,700 images retained
- At 0.8: 1,000 images retained

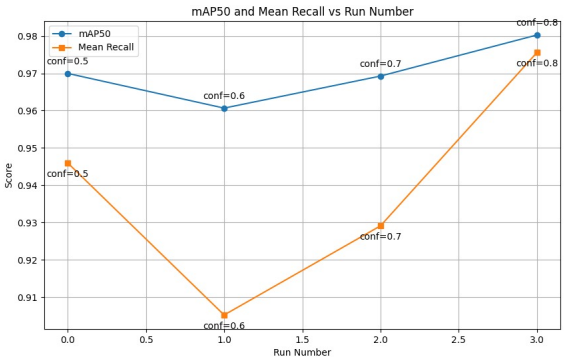

Figure 6: Effect of pseudo-label confidence threshold on validation performance for ID data. As confidence increases, recall drops due to fewer retained frames, but mAP50 can improve due to higher label quality.

Training on Pseudo-Labels:

Each filtered set of pseudo-labeled frames was used to retrain the model for 100 epochs. For training:

- Only the pseudo-labeled images were used as the training set
- All original labeled images (from training and validation, total of 71 images) were used for validation

This setup allowed us to evaluate how adding pseudo-labeled data improved generalization, depending on the confidence threshold.

2.3 Pseudo-Label Model (Out-of-Distribution - OOD)

We repeated the pseudo-labeling and retraining process on the out-of-distribution (OOD) video, which featured a different surgical procedure setting and camera setup.

Pipeline Details:

- Every 5^{th} frame was sampled (to reduce computation due to higher frame rates)
- Pseudo-labels were generated using the same baseline model

- Only frames where all detections exceeded the chosen confidence threshold were retained

Threshold Impact:

- At 0.5: 2,500 usable images retained
- At 0.8: 1,500 usable images retained

Each filtered set was used to fine-tune the model for 100 epochs, evaluated on the full labeled set. This experiment tested the model's ability to generalize to unseen conditions using semi-supervised learning techniques.

Figure 7: Visual comparison of the OOD video at different confidence thresholds.

3 Discussion and Conclusions

Baseline Model

- We showed that even with a very small labeled dataset (only 61 images), it's possible to train a functional object detection model using a lightweight architecture like YOLOv8n.
- The model achieved decent performance on the validation set, but it struggled in harder cases—such as when the tools were partially hidden or when lighting conditions changed—indicating limited robustness.

In-Distribution (ID) Pseudo-Label Model

- By using pseudo-labels from ID videos, we were able to greatly increase the training data without needing new manual annotations. This helped the model generalize better.
- We chose to use every 5th frame to avoid too much repetition, which kept training efficient and reduced the risk of overfitting.
- We found that setting the right confidence threshold for pseudo-labels was important. Lower thresholds gave us more training images but added noise. Higher thresholds gave fewer images, but they were more accurate. This trade-off directly affected model performance.

Out-of-Distribution (OOD) Pseudo-Label Model

- When we applied the same semi-supervised method to the OOD video—taken from a different setup—we saw that the model could still learn and perform the task.
- Although the performance was generally lower compared to the ID case, the results showed that pseudo-labeling can still help the model adapt to unseen scenarios, even if the new data comes from a different distribution.