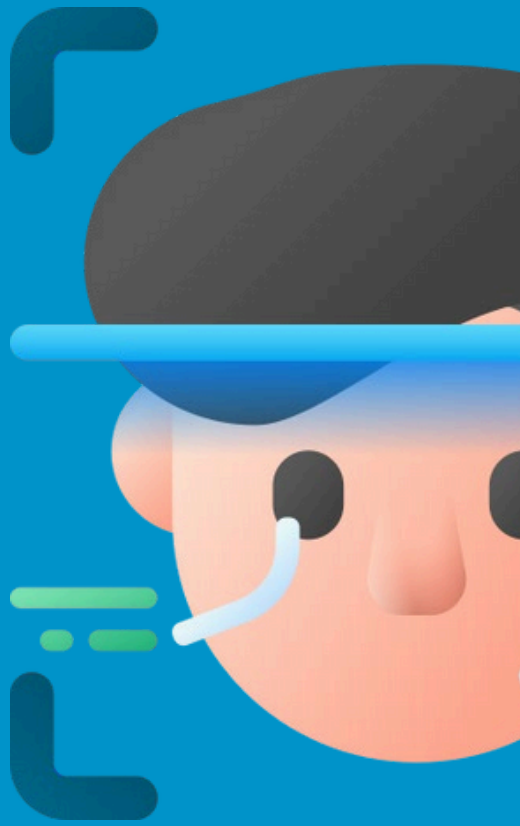
Efficient Computer Vision Models for Silkworm Feeding Prediction and Habitat Analysis



Arianna Viola
Rossella Milici

Outline

- Introduction
- State of the Art
- Model Evaluation
- Proposed methods
- Datasets and Metrics
- Experimental setup
- Implementation details
- Conclusion and Future Works
- References

Problem Statement

Efficient and sustainable silkworm rearing requires accurate, real-time monitoring of feeding conditions, which is traditionally done through manual inspection. This process is labor-intensive, prone to human error, and not scalable. The project aims to:

1. Perform binary classification using lightweight neural network architectures to determine whether silkworms require feeding (feeding vs. no feeding).
2. Develop unsupervised segmentation methods to distinguish between silkworms, mulberry leaves, and background in rearing bed images—without relying on labeled pixel-wise data.

State of Art

Task	Approach Type	Key Methods/Models	Notes
Feeding classification	Binary	EfficientNetV2, RepNeXt, MobileViT	Lightweight, deployable
Segmentation	Unsupervised	DINOV2, K-Means, Watershed,Sam	No need for pixel-wise labels
Segmentation	Supervised	SegFormer	Minidatset made of images manually annotated on LabelMe
Evaluation	Quantitative/Qualitative	Accuracy, F1, Precision, Recall, ROC AUC	Evaluate the binary classification

Evaluation metrics

- Accuracy: the correct predictions over the whole predicted sample

$$\frac{TP+TN}{TP+TN+FP+FN}$$

- Precision: the ratio of the correct predictions over the whole correct samples

$$\frac{TP}{TP+FP}$$

- Recall: the ratio of correct predictions for a class to the total number of cases in which it occurs

$$\frac{TP}{TP+FN}$$

- F1-score: the Harmonic mean between Precision and Recall

$$\frac{2XPRECISIONXRECOLL}{PRECISION+RECOLL}$$

GradCam

The GradCAM class implements a technique that allows you to "see" through the eyes of the neural network. GradCAM generates a "heatmap" that overlays the original image. The "hottest" areas (red/yellow) in the map indicate the exact pixels and regions that were most influential in the model's decision.

This class queries a previously trained model to understand its decision-making process.

Dataset

The experiment uses a custom dataset located at /kaggle/input/silk-dataset/.
The image filenames and their corresponding class labels are defined in a CSV file named 0_data.csv.

0_data.csv (21.64 kB)

Detail Compact Column

# classificazione	▵ foto
0 1	1351 unique values
0	IMG_2663.jpg
0	IMG_2664.jpg
0	IMG_2665.jpg
0	IMG_2666.jpg
0	IMG_2667.jpg
0	IMG_2668.jpg
0	IMG_2669.jpg
0	IMG_2670.jpg
0	IMG_2671.jpg
0	IMG_2672.jpg
0	IMG_2673.jpg
0	IMG_2674.jpg

Experimental SetUp

1) Data and Preprocessing

Dataset:The experiment uses a custom dataset located at /kaggle/input/silk-dataset/. The image filenames and their corresponding class labels are defined in a CSV file named 0_data.csv;
Image Specifications: All images are resized to a standard resolution of 224x224 pixels;
Normalization : The images are normalized using the standard ImageNet mean ([0.485, 0.456, 0.406]) and standard deviation ([0.229, 0.224, 0.225]). It ensures that the input images have a similar statistical distribution to the data the model was originally trained on;

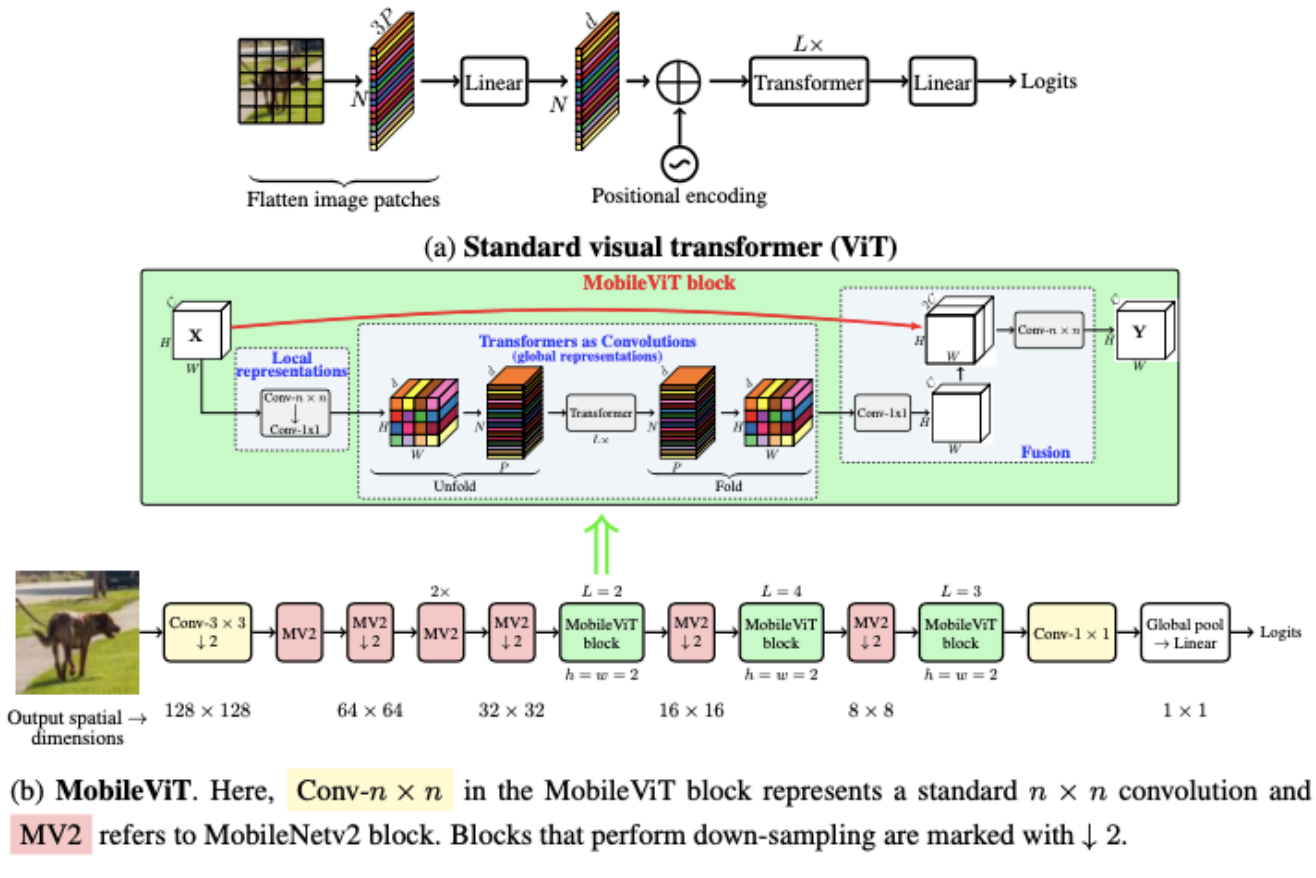
2) Image Classification

Training Hyperparameters: Splitted in 80 (train) /20 (validation);
Epochs: The model is trained for 10 epochs;
Batch Size: The data is fed to the model in batches of 32 images at a time;
Hardware: The training process is configured to run on a CUDA-enabled GPU if available, falling back to the CPU otherwise;

3) Unsupervised Segmentation: DINOV2, K-Means,Segformer with labelme,Watershed and SAM;

Proposed method for binary classification:

MobileVit

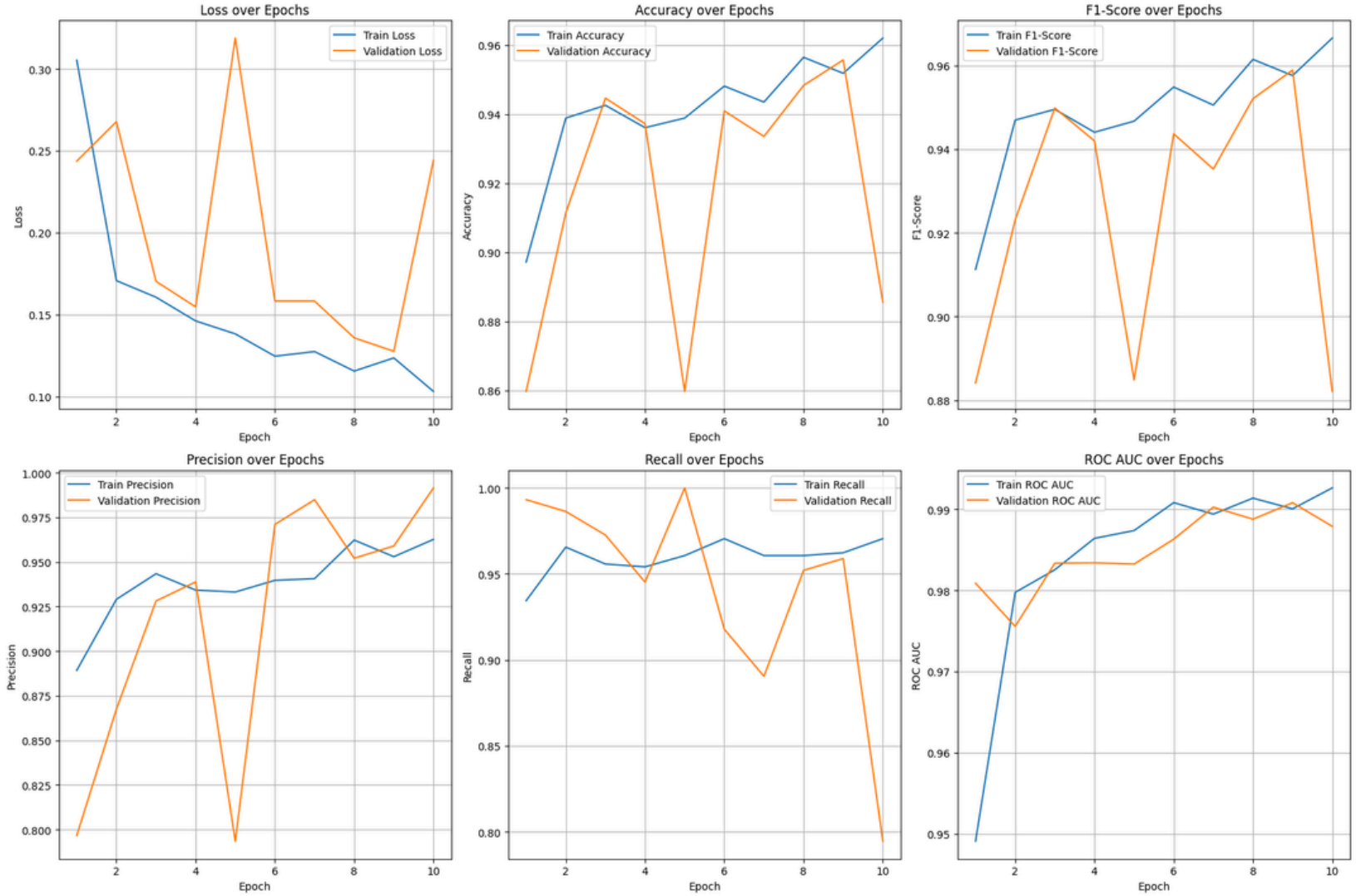


(b) **MobileViT**. Here, Conv-$n \times n$ in the MobileViT block represents a standard $n \times n$ convolution and MV2 refers to MobileNetv2 block. Blocks that perform down-sampling are marked with $\downarrow 2$.

Light-weight, General-purpose, and Mobile-friendly Vision Transformer.

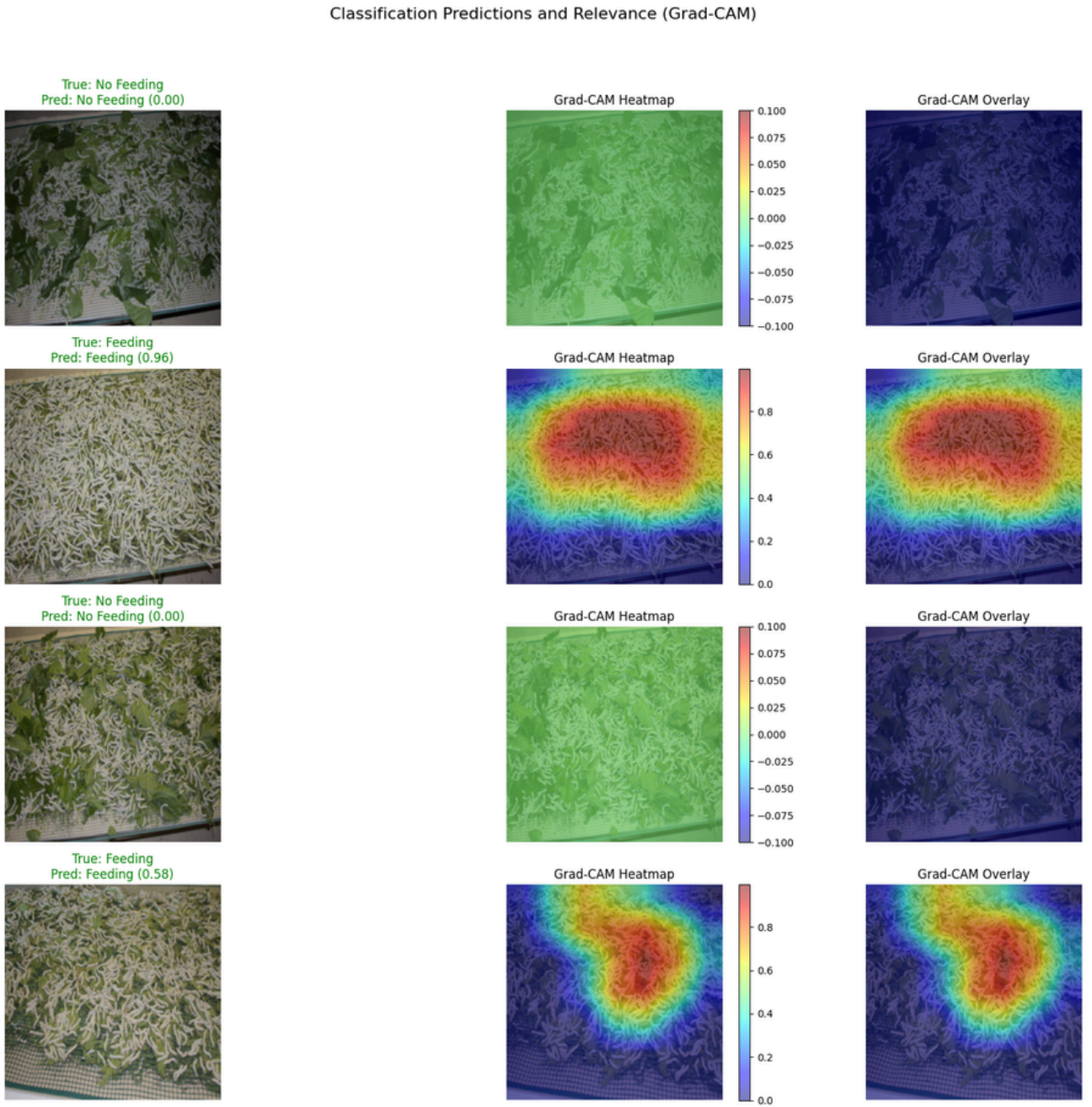
ANALYSIS PERFORMANCE

```
--- Final Classification Metrics (Validation Set) ---
   Metric    Value
 Accuracy 0.885609
 F1-Score 0.882129
Precision 0.991453
   Recall 0.794521
  ROC AUC 0.987890
```



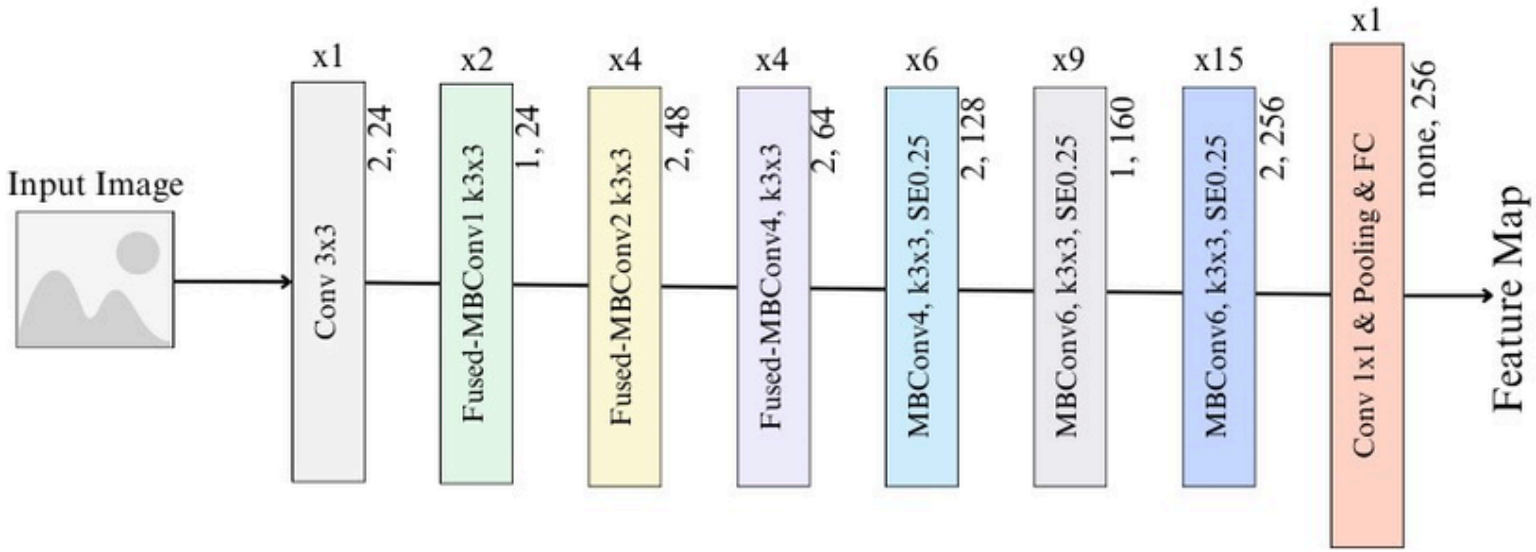
Green->The model did not localise a specific area and took its decision by analyzing the image globally.

Overlay->Heat map is overlapped to the original image.



Proposed method for binary classification:

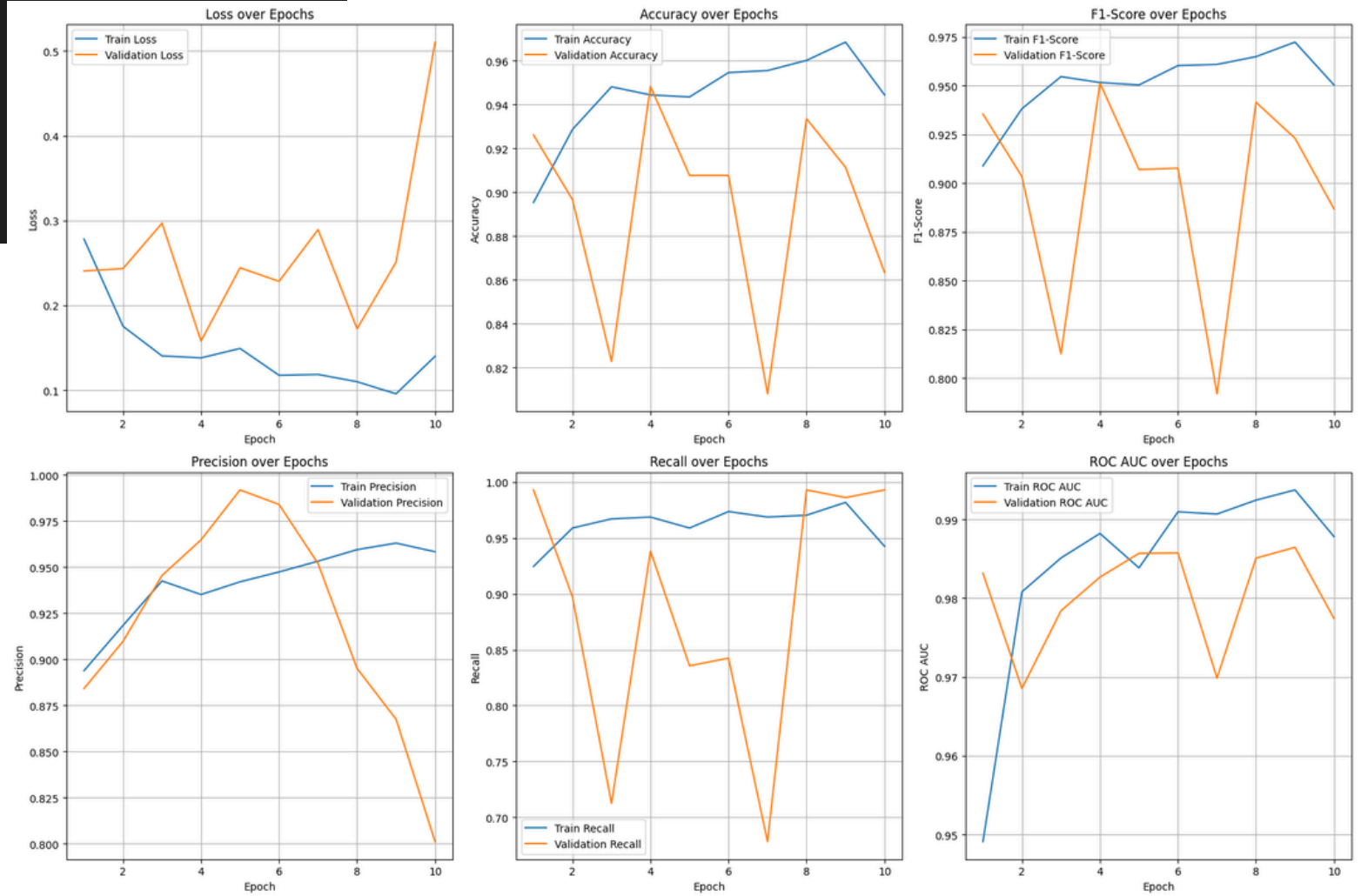
EfficientNetV2

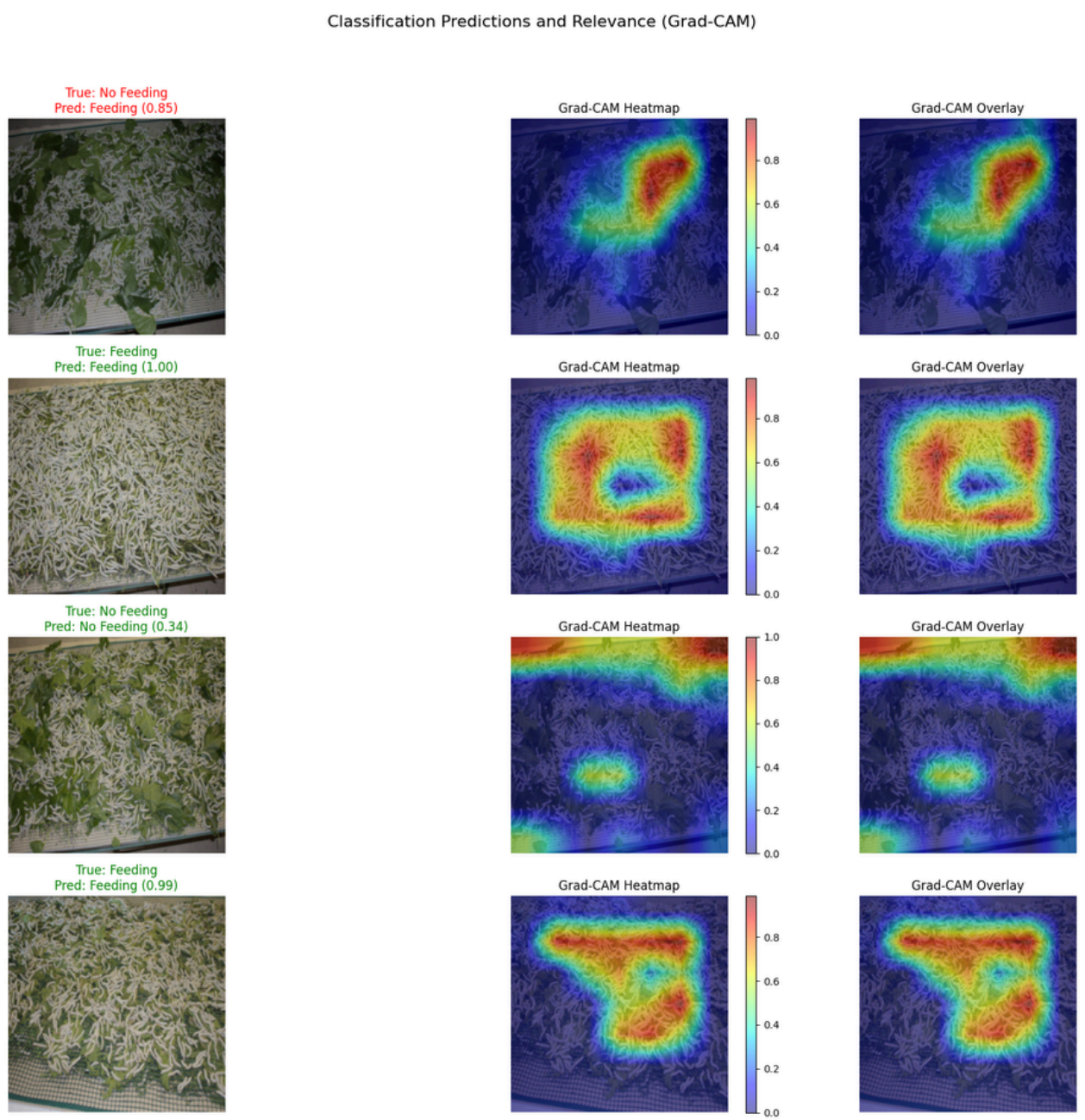


Smaller Model and Faster Training.

ANALYSIS PERFORMANCE

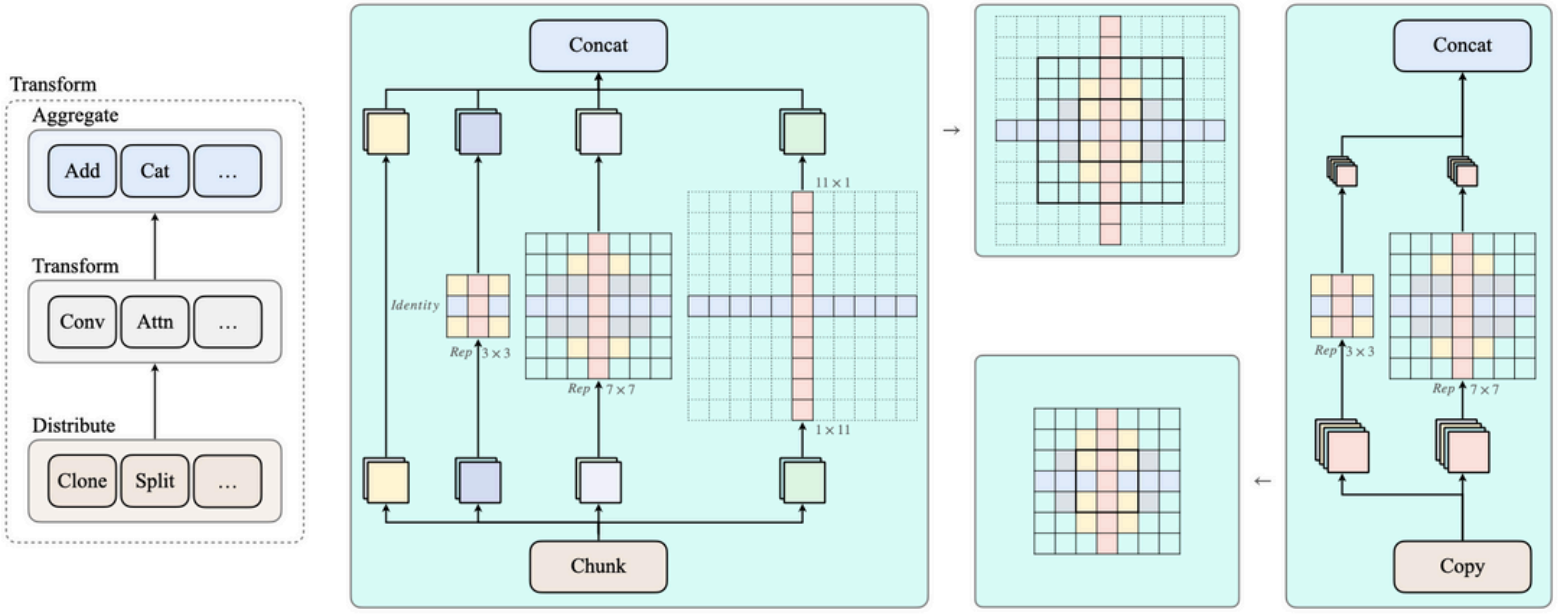
```
--- Final Classification Metrics (Validation Set) ---
   Metric    Value
 Accuracy 0.863469
 F1-Score 0.886850
Precision 0.801105
   Recall 0.993151
  ROC AUC 0.977425
```




Classification Predictions and Relevance (Grad-CAM)
True: No Feeding
Pred: Feeding (0.85)
Grad-CAM Heatmap
Grad-CAM Overlay
True: Feeding
Pred: Feeding (1.00)
Grad-CAM Heatmap
Grad-CAM Overlay
True: No Feeding
Pred: No Feeding (0.34)
Grad-CAM Heatmap
Grad-CAM Overlay
True: Feeding
Pred: Feeding (0.99)
Grad-CAM Heatmap
Grad-CAM Overlay

Proposed method for binary classification:

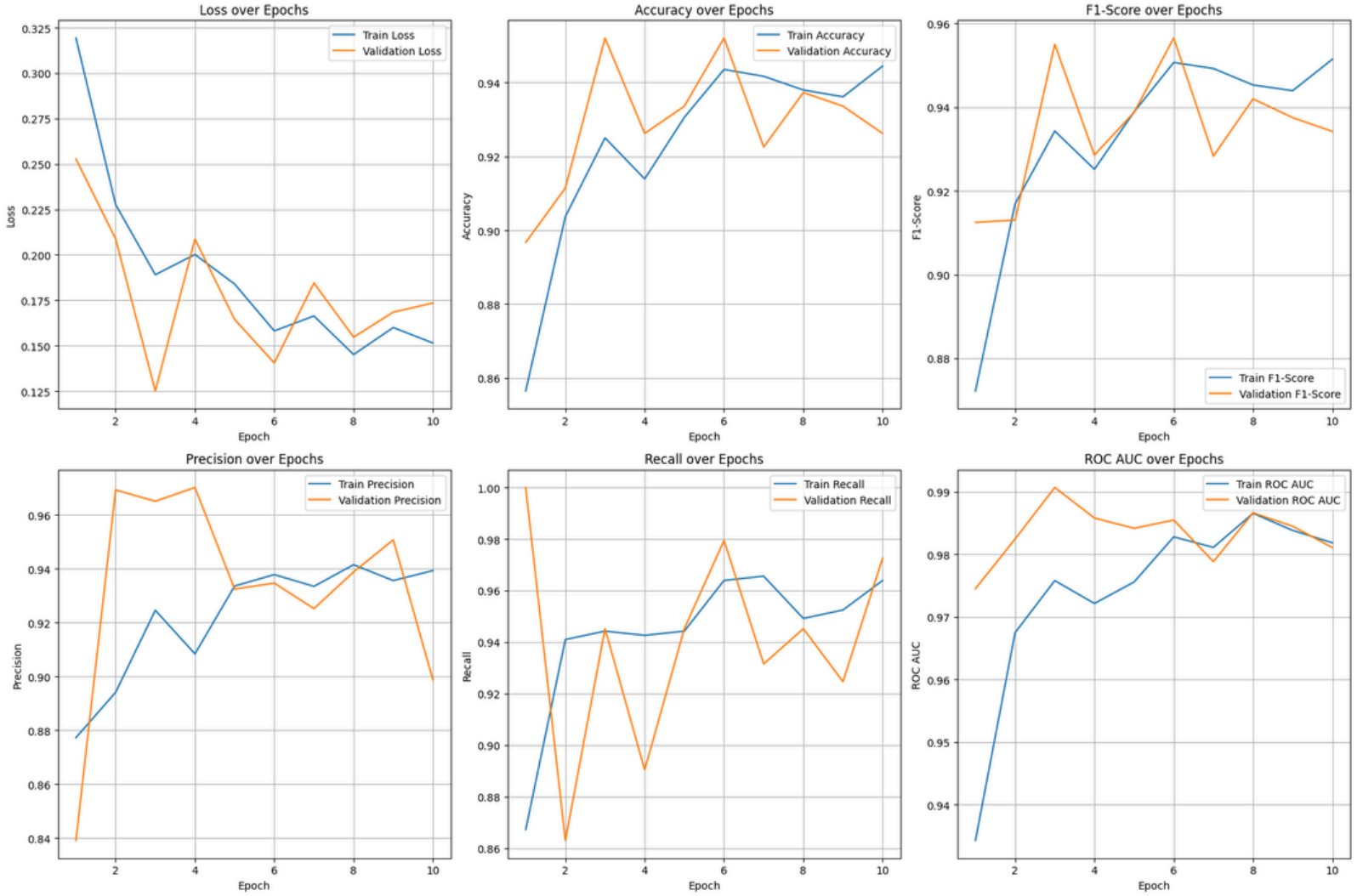
RepNeXt

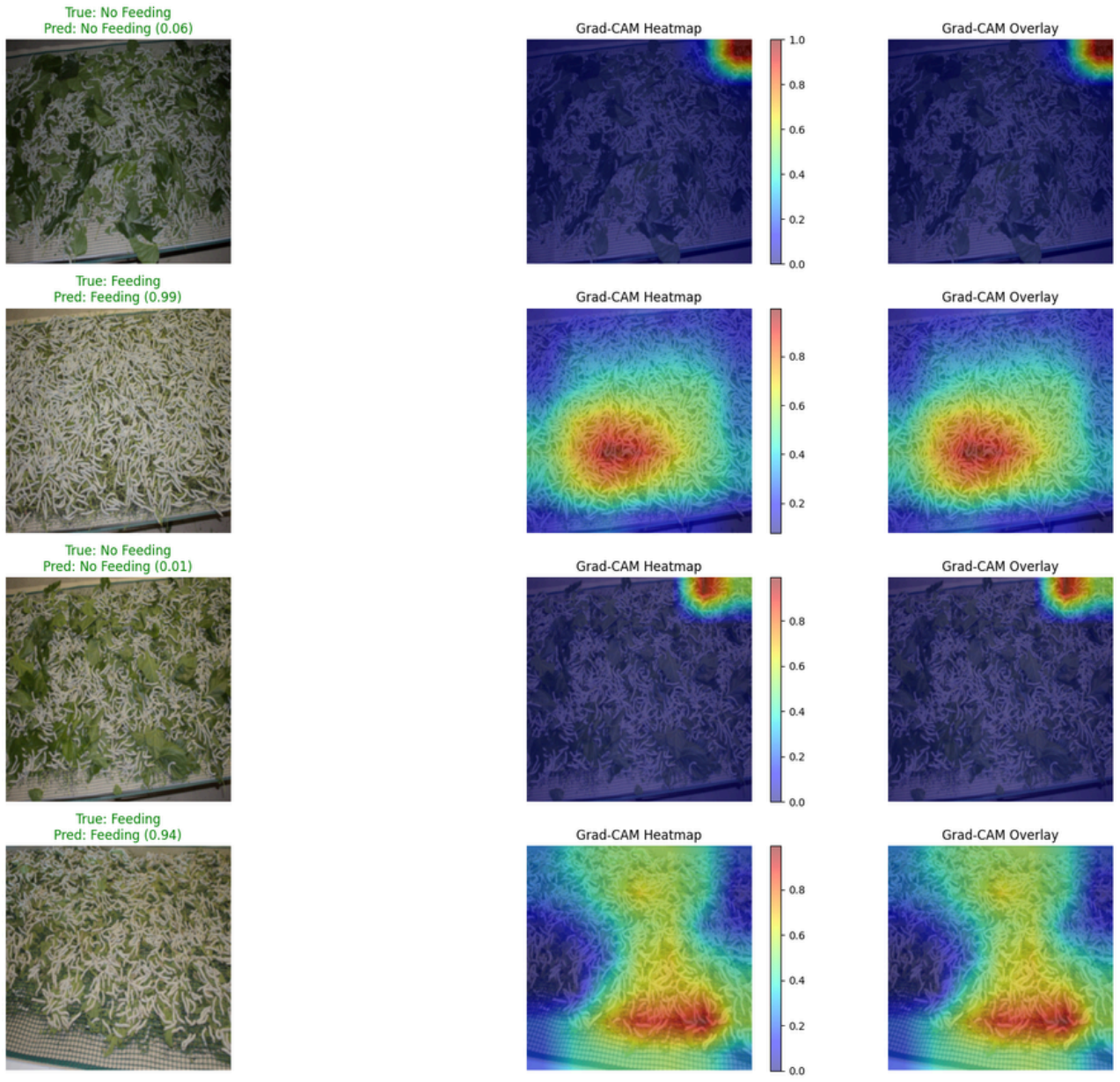


A Fast Multi-Scale CNN using Structural Reparameterization.

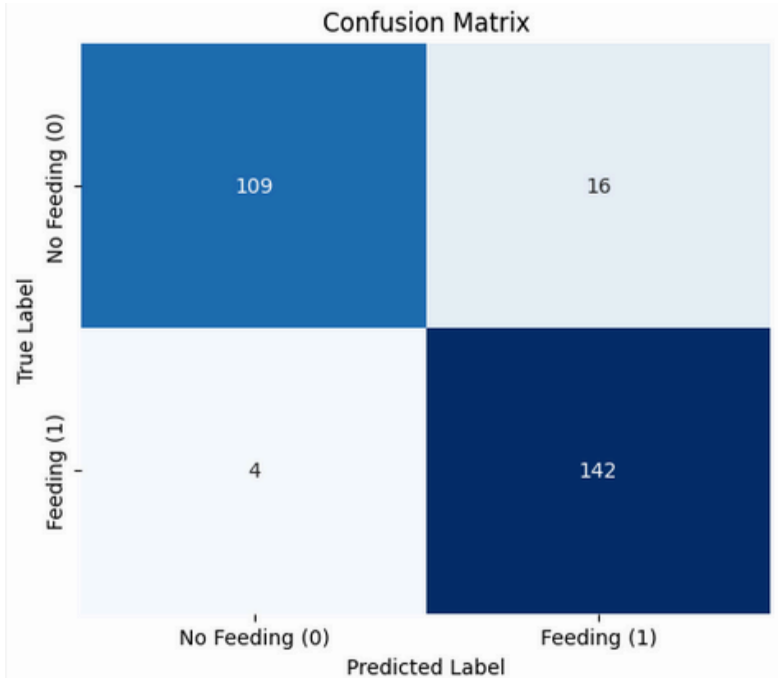
ANALYSIS PERFORMANCE

```
--- Final Classification Metrics (Validation Set) ---
    Metric    Value
  Accuracy 0.926199
  F1-Score 0.934211
 Precision 0.898734
    Recall 0.972603
   ROC AUC 0.981096
```

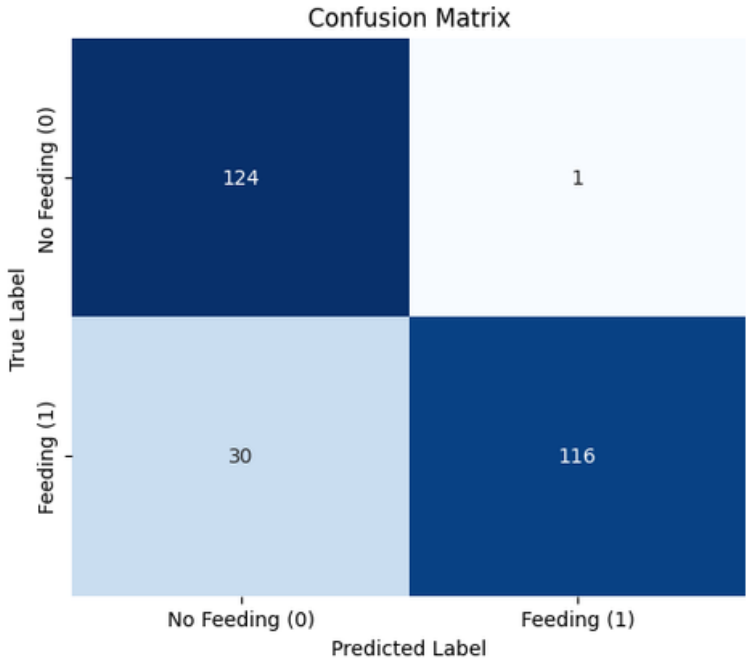



Classification Predictions and Relevance (Grad-CAM)
True: No Feeding
Pred: No Feeding (0.06)
Grad-CAM Heatmap
Grad-CAM Overlay
1.0
0.8
0.6
0.4
0.2
0.0
True: Feeding
Pred: Feeding (0.99)
Grad-CAM Heatmap
Grad-CAM Overlay
0.8
0.6
0.4
0.2
True: No Feeding
Pred: No Feeding (0.01)
Grad-CAM Heatmap
Grad-CAM Overlay
0.8
0.6
0.4
0.2
0.0
True: Feeding
Pred: Feeding (0.94)
Grad-CAM Heatmap
Grad-CAM Overlay
0.8
0.6
0.4
0.2
0.0

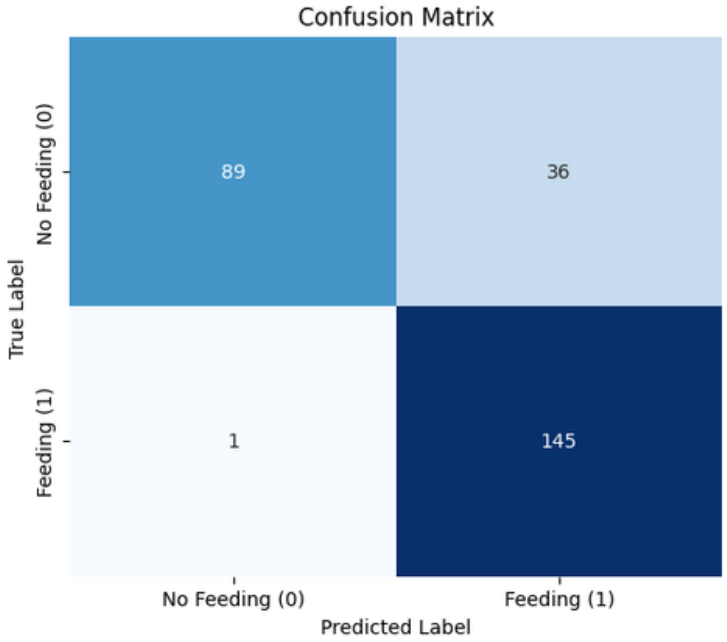
Results: baseline evaluation



RepNeXt



MobileNet



EfficientV2

Experimental SetUp

Pre-processing (mainly done in watershed, means, sam)

1. **Linear Color Space Conversion**: Images are converted from the sRGB color space to a linear space then back to sRGB. This ensures brightness and color adjustments.(Only for watershed and kmeans);

2. **Brightness Normalization**: Adjusts the overall image brightness;

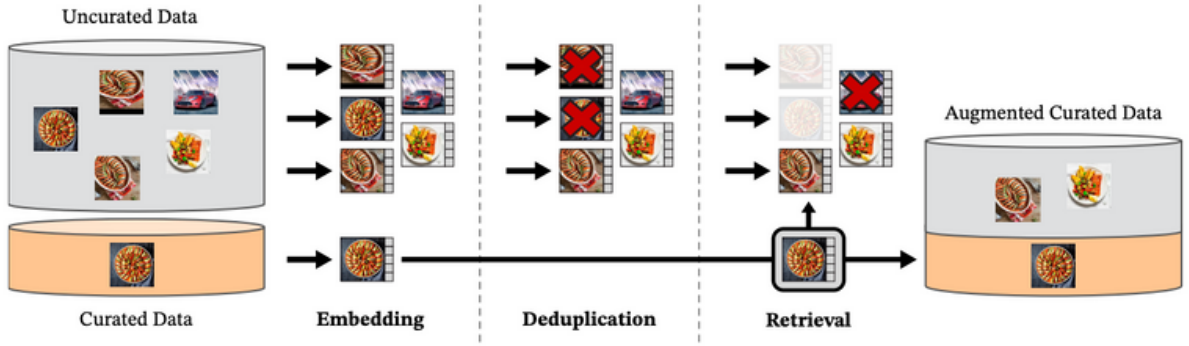
3. **Gray World Color Balancing**: This reveals the true colors of leaves and silkworms;

4. **LAB Color Space**: to mimic how humans perceive color;

5. **High-Pass Filtering**: sharpens edges and emphasize high-frequency components;

6. **HSV Color Space**: The image is converted to HSV (Hue, Saturation, Value). Saturation of green pixels (leaves) is boosted, and red pixels are desaturated;

Proposed method: DINOV2



Model: Pretrained from meta with millions of images;

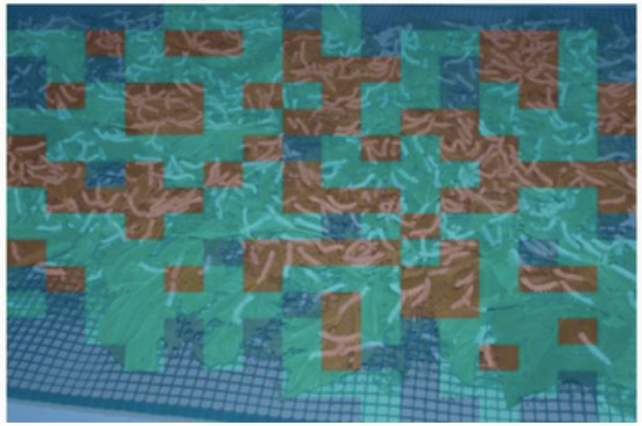
Process Extraction: DINOv2 analyzes all the images and extracts their visual "fingerprints";

Clustering: K-Means takes all these fingerprints and groups them together, finding common visual categories across the entire dataset.

Reconstruction: The code uses the clustering results to create a "mask" for each image, coloring each area according to the category to which it has been assigned.

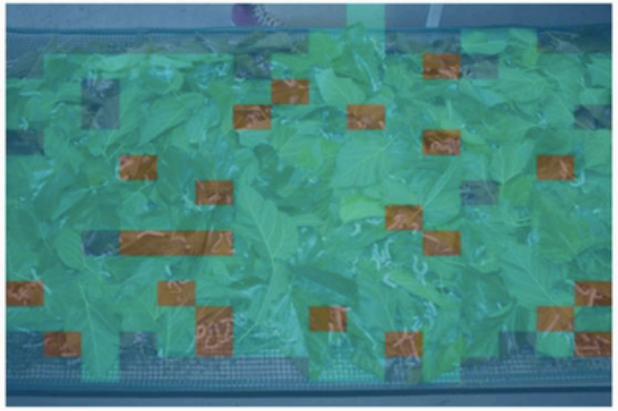
Originale (Indice Casuale: 70)

Segmentazione DINO (K=3)



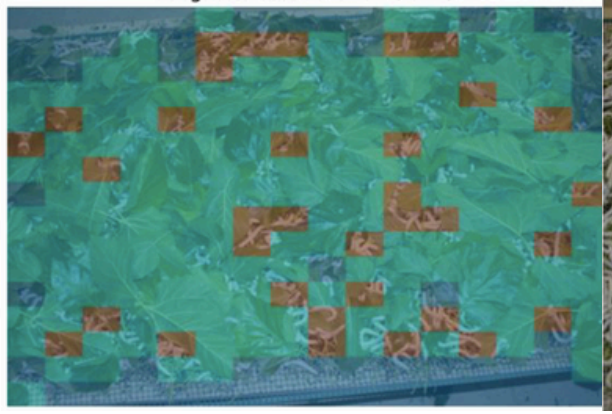
Originale (Indice Casuale: 289)

Segmentazione DINO (K=3)



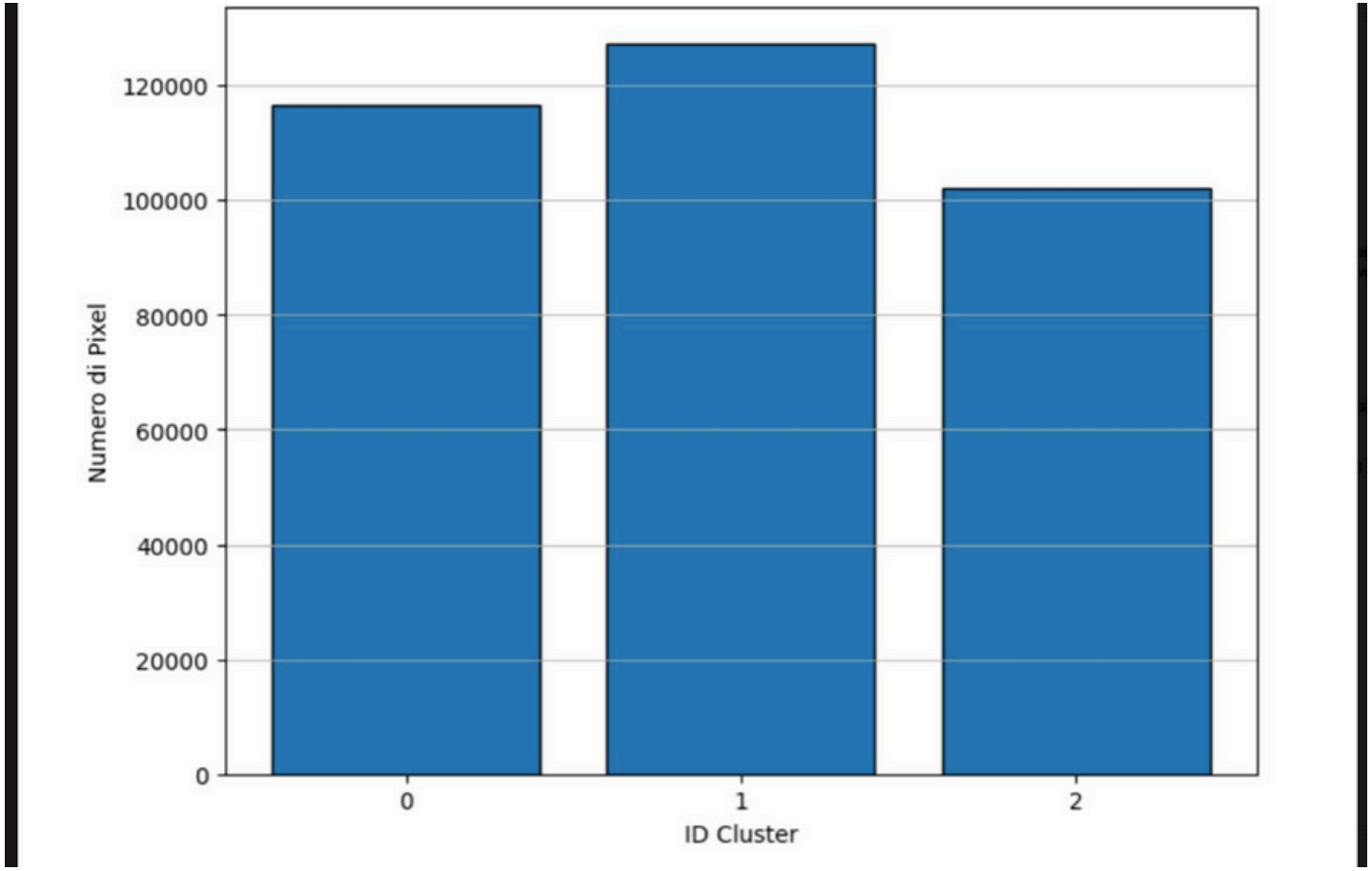
Originale (Indice Casuale: 1129)

Segmentazione DINO (K=3)



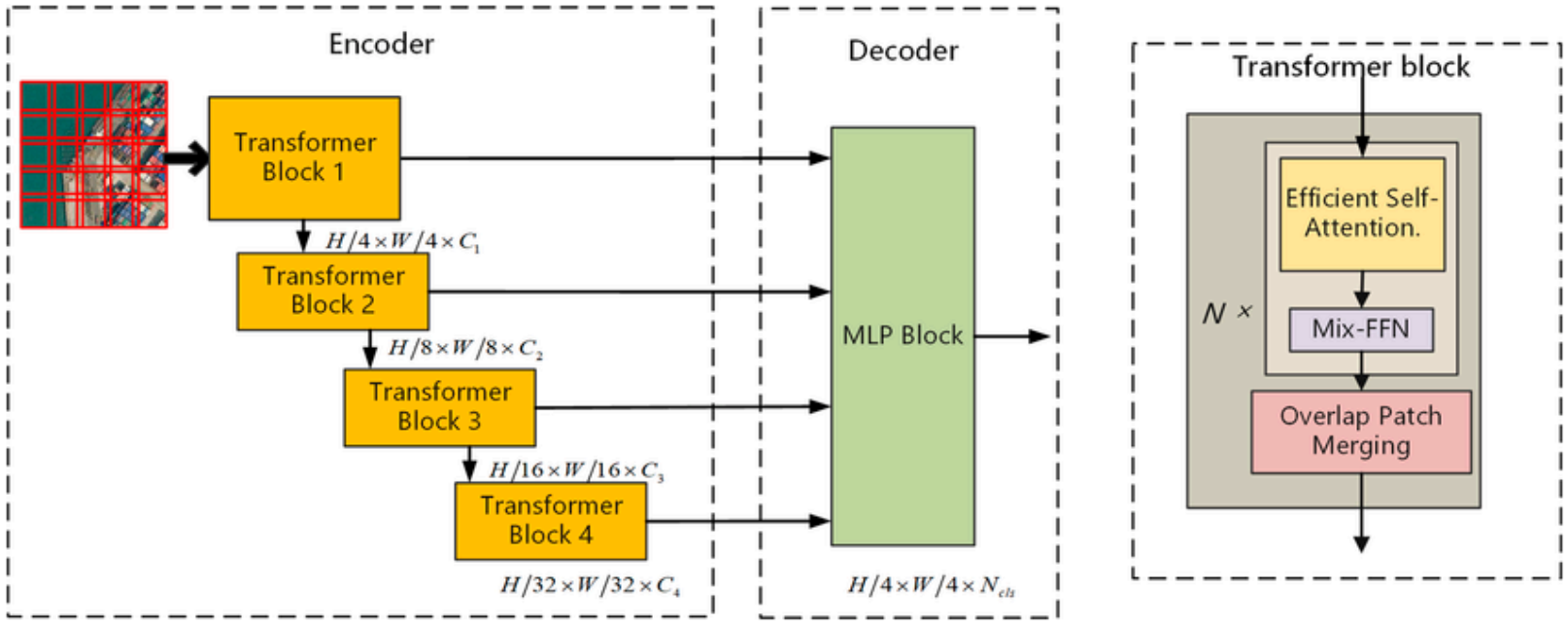
Originale (Indice Casuale: 779)

Segmentazione DINO (K=3)

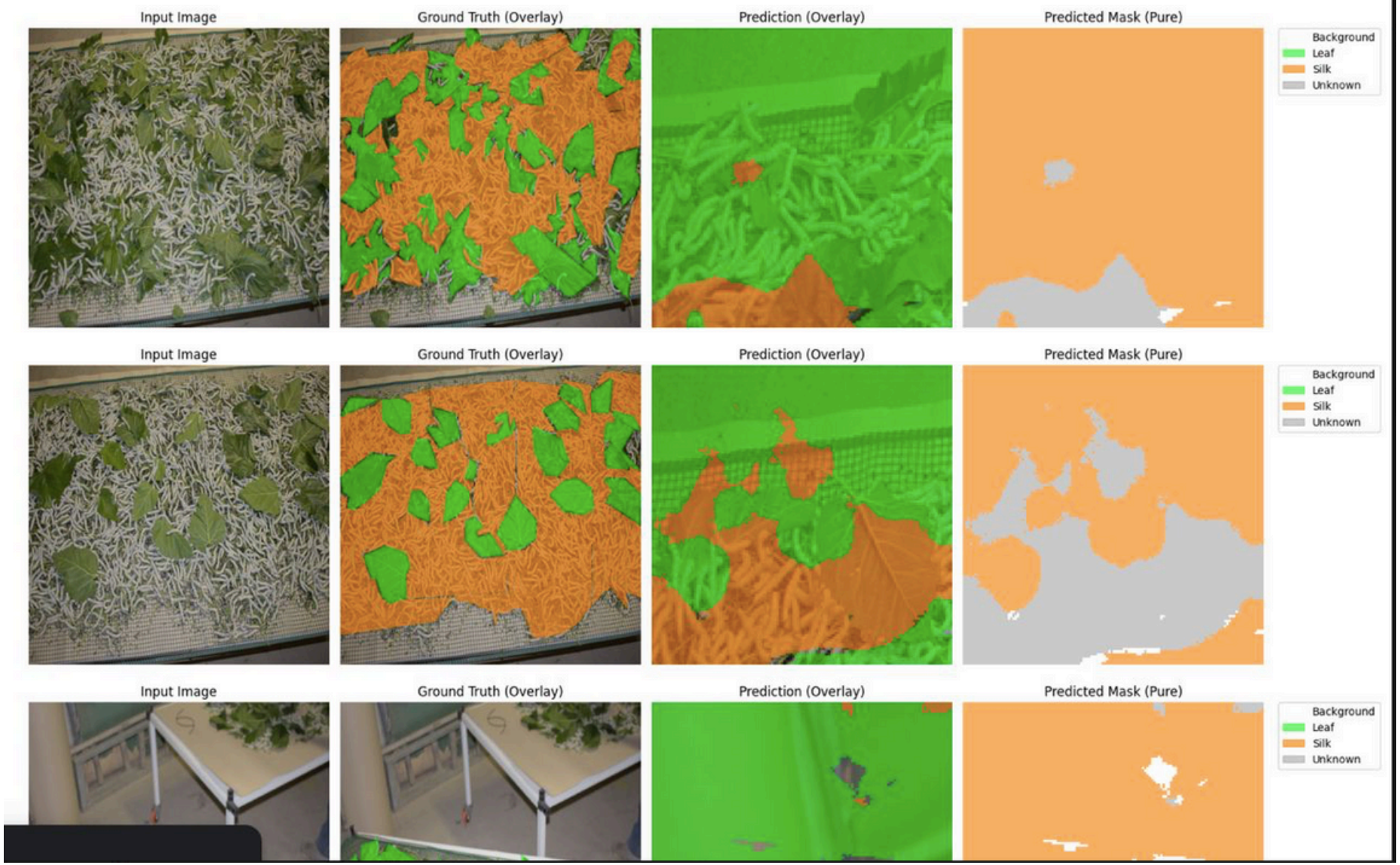


This graph shows how many pixels ended up in each group

Proposed method: SegFormer with manual annotation on LABELME



We took a small, carefully hand-labeled dataset of images and their corresponding masks, and used it to teach a powerful, pre-trained AI model how to specifically recognize and outline "leaves", “silkworms” and “background” in the images. Once that's done, we trained it, evaluated its performance, and then visualized its predictions to really see how well it learned.



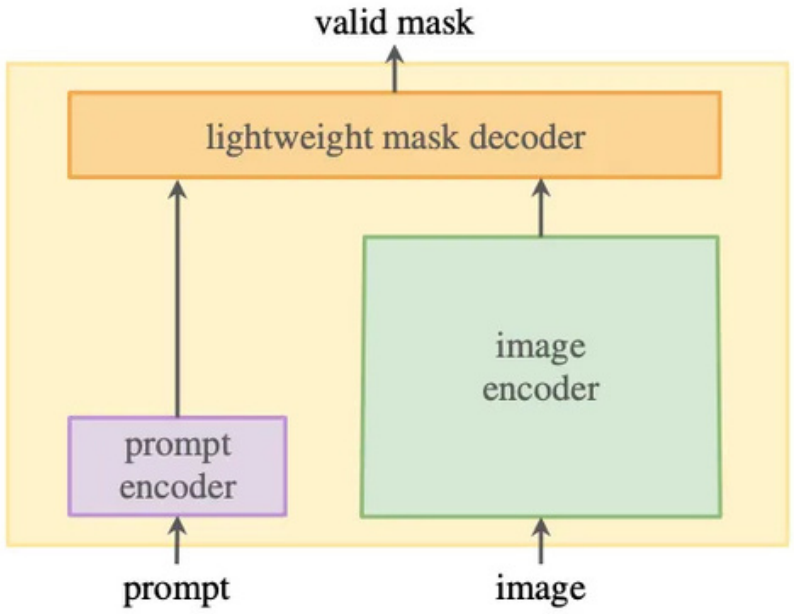
What the model saw.

What the model should have produced.

What the model actually produced.

The raw output of the model, shown with a helpful legend.

Proposed method: SAM



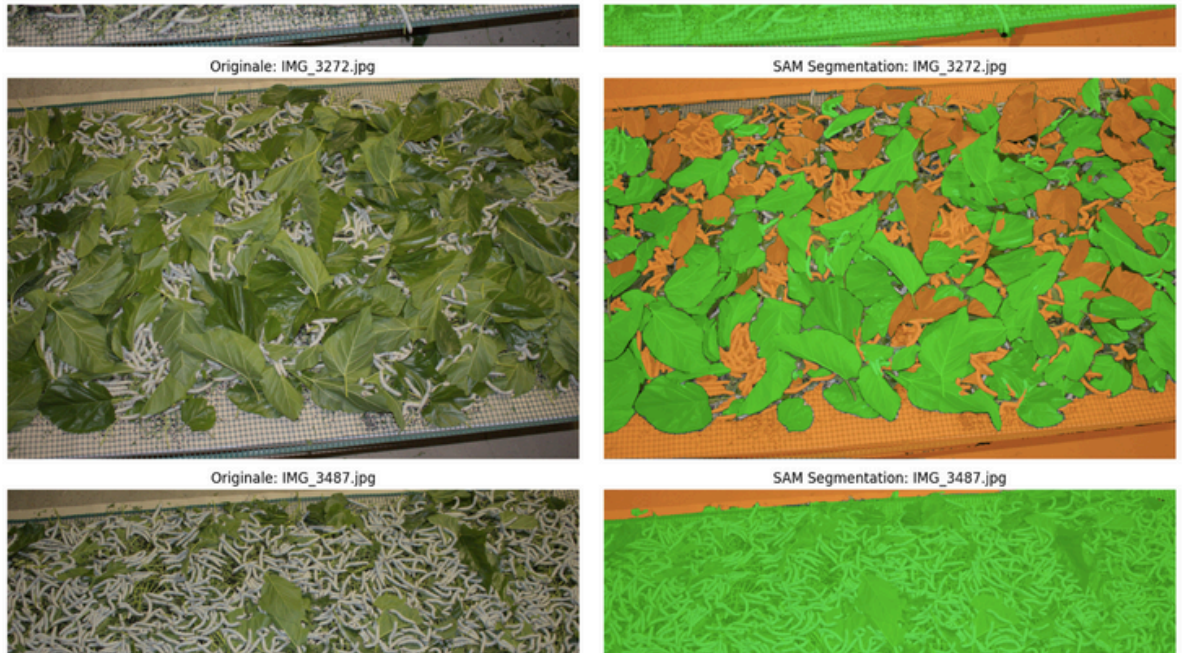
We applied SAM to 150 immages

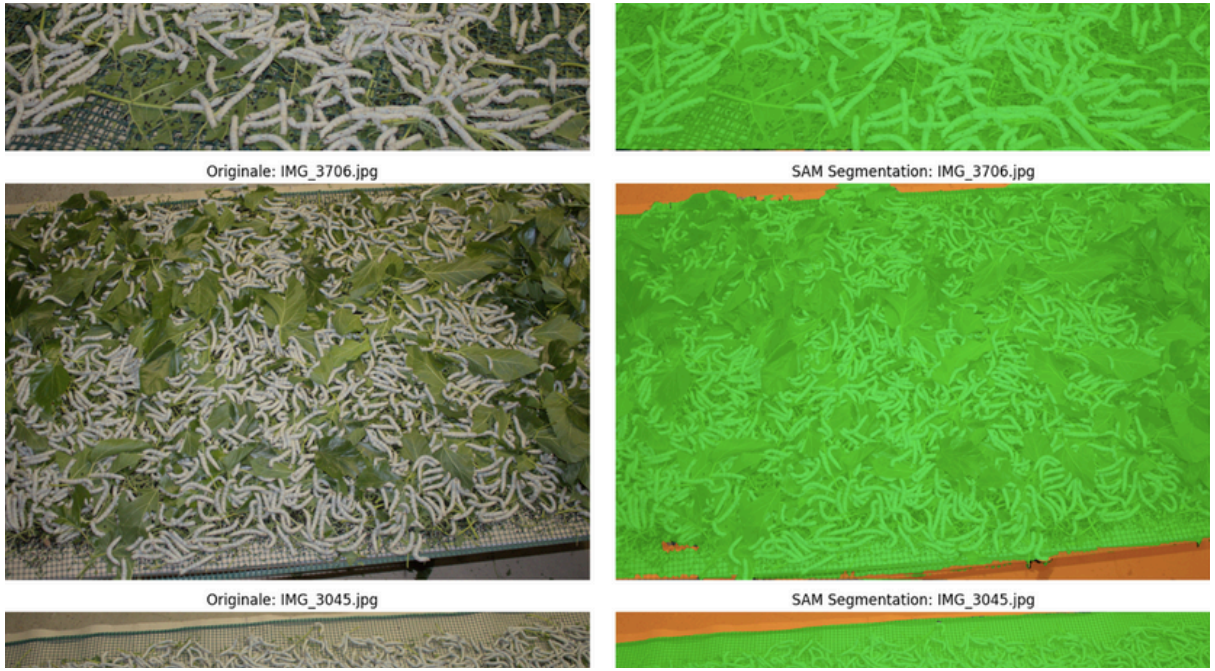
This code's main purpose is to **automate the segmentation** of silkworms and leaves in images **using Meta's Segment Anything Model (SAM)**.

For each generated mask the classification is based on the value of the HUE. If a mask could be both, we prioritized leaf.

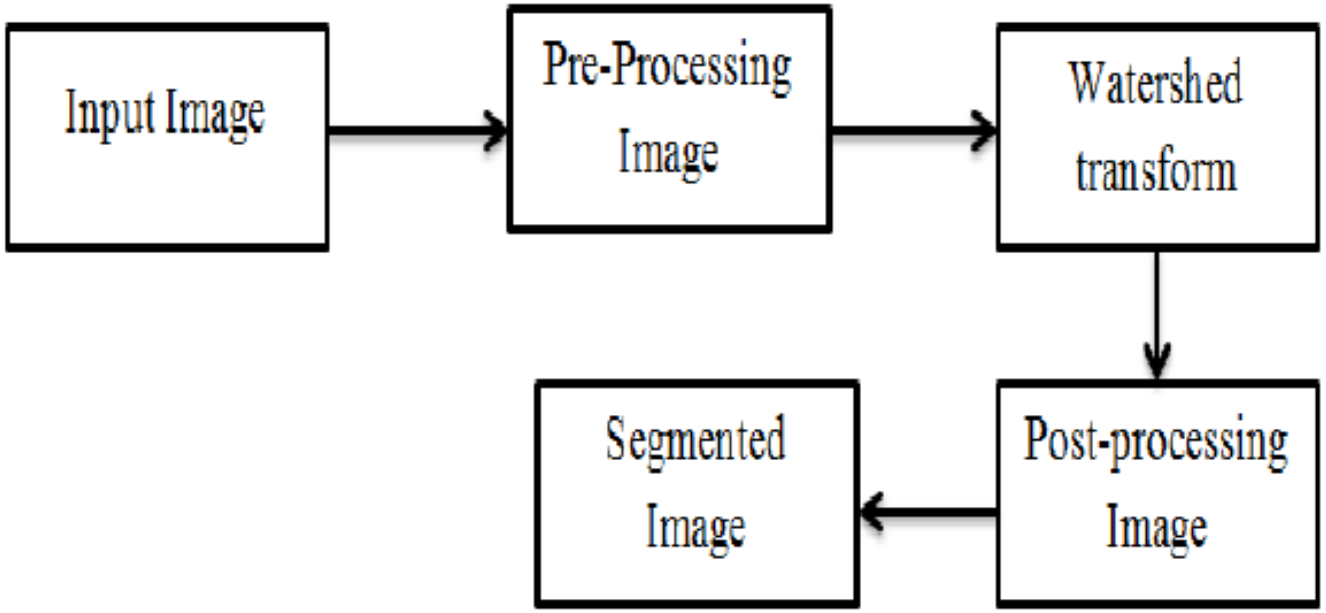
Finally, it **visualizes** the generated masks.

Proposed method: SAM



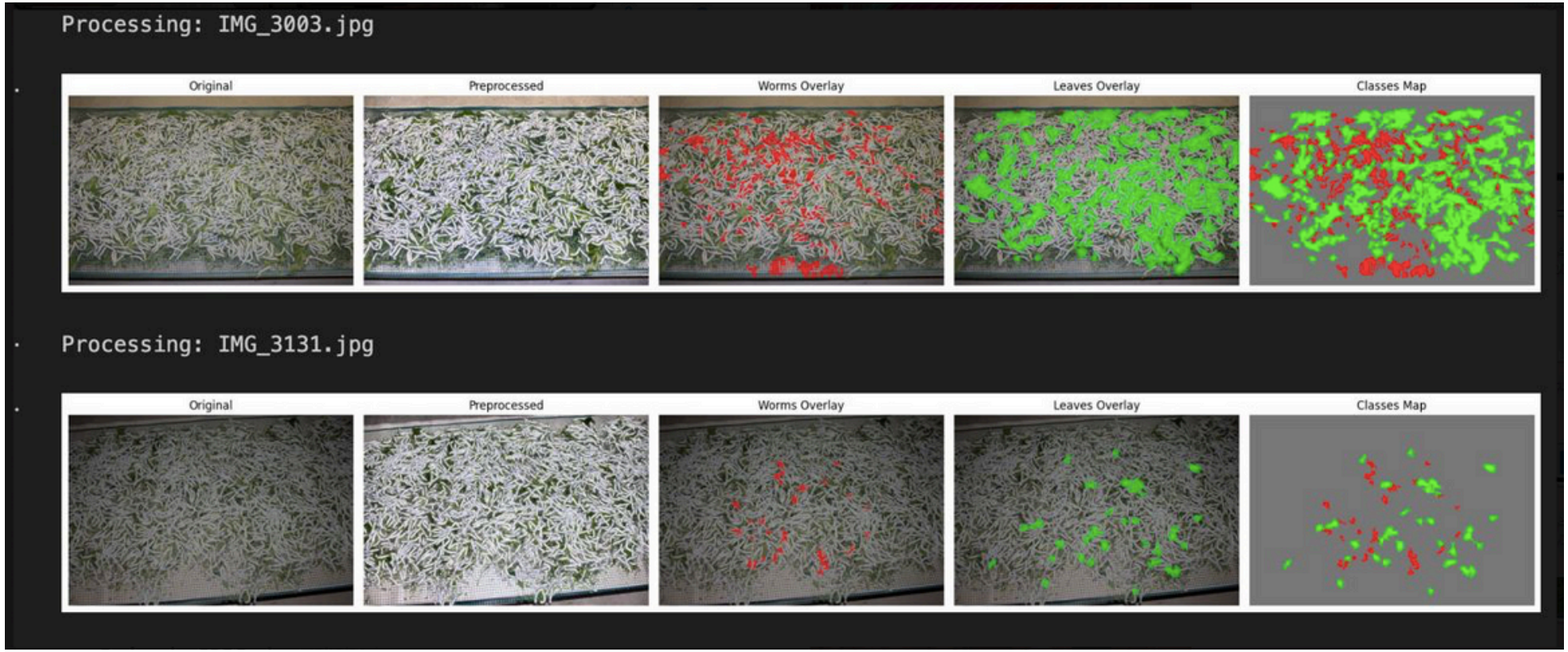


Proposed method: Watershed



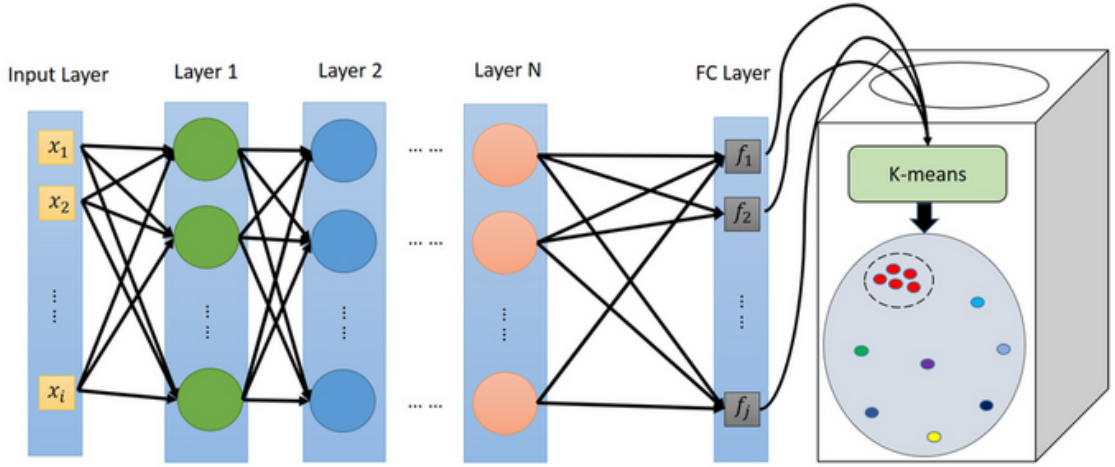
It is based on the topography and color properties of the image.
The Watershed Algorithm is used when segmenting images with touching or overlapping objects. It excels in scenarios with irregular object shapes, gradient-based segmentation requirements, and when marker-guided segmentation is feasible.

Proposed method: Watershed



Aim: separate leaves (colored green), silkworms (colored red) and background (gray).
CLASSIFICATION RULE: For leaves we used the LAB color space by using a threshold to mark point as potential leaves. For silkworms we focused on the valuse of HSV (low HSW saturation , high HSV value, bightness).
The watershed algorithm used this data to create a RGB image, were pixels identified as leaves were colored green and pixels identified as worms were colored red.

Proposed method: K-MEANS



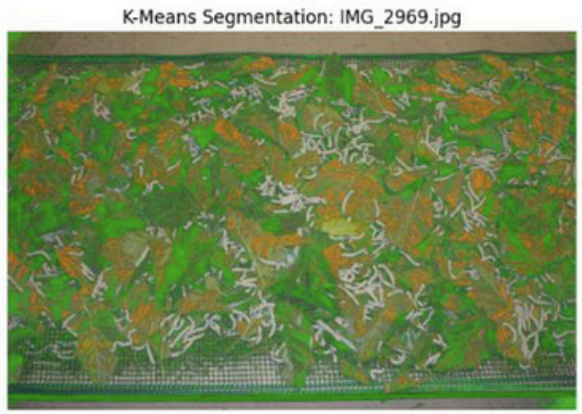
This code performs **unsupervised image segmentation using K-Means clustering**, specifically tailored to identify "silkworm" and "leaf" regions within images.

We found the main colors in the image and assigned each pixel to the group with the most similar color.

Finally, it visualizes the results.

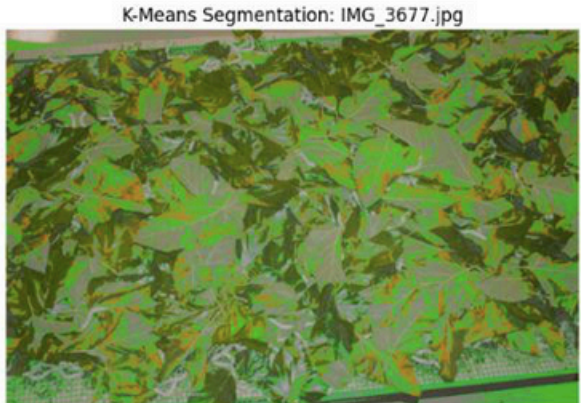

Originale: IMG_2969.jpg


Preprocessed: IMG_2969.jpg


K-Means Segmentation: IMG_2969.jpg

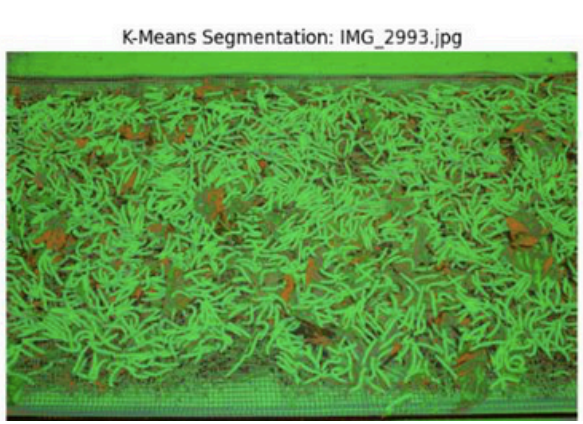

Originale: IMG_3677.jpg


Preprocessed: IMG_3677.jpg


K-Means Segmentation: IMG_3677.jpg


Originale: IMG_2993.jpg

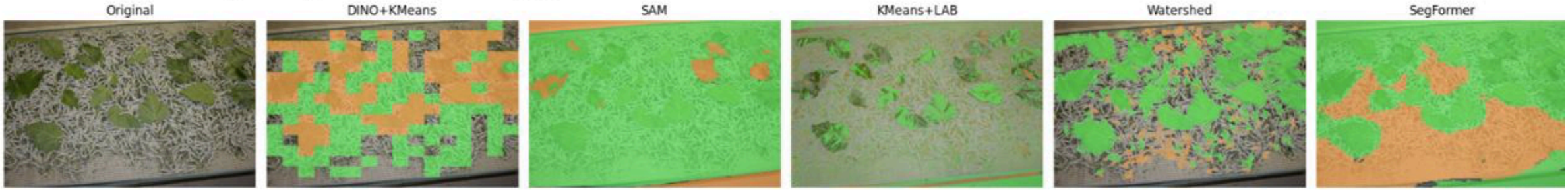

Preprocessed: IMG_2993.jpg


K-Means Segmentation: IMG_2993.jpg

Results: state of the art

```
================================================================================
          COMPARING SEGMENTATION METHODS
================================================================================

--- Processing Image: IMG_3200.jpg ---
```




```
--- Comparison Visualization Complete ---
```

Conclusion and future works

We used different models for classification and unsupervised segmentation.
We also tried to experiment by using a method for supervised segmentation thanks to the annotations made on LabelMe.

References

1. Zhao, M., Luo, Y., and Ouyang, Y. (2024). RepNeXt: A Fast Multi-Scale CNN using Structural Reparam-
eterization. arXiv.

2. Tan, M., and Le, Q. V. (2021). EfficientNetV2: Smaller Models and Faster Training. arXiv.

3. Mehta, S., and Rastegari, M. (2022). MobileViT: Light-weight, General-purpose, and Mobile-friendly Vision Transformer. arXiv.

4. Rossetti, S., Sam`a, N., and Pirri, F. (2023). Removing supervision in semantic segmentation with local-global matching and area balancing. arXiv.

5. Niu, D., Wang, X., Han, X., Lian, L., Herzig, R., and Darrell, T. (2023). Unsupervised Universal Image Segmentation. arXiv.

Thank you for the attention