Efficient Computer Vision Models for Silkworm Feeding Prediction and Habitat Analysis

Master's Degree in Artificial Intelligence and Robotics

SAPIENZA
UNIVERSITÀ DI ROMA

Venneri Roberta (2019902)
Ruggeri Riccardo (1613249)

Outline

- Problem Statement
- State Of The Art
- Proposed Method
- Dataset
- Experimental Setup
- Model Evaluation
- Conclusions
- References

PROBLEM STATEMENT

- Ensuring efficiency in silkworm rearing is a key factor for sustainable silk production, and it critically depends on accurately tracking feeding conditions.

- There are 2 current issues:
  - **manual monitoring**, which is slow, expensive and subject to human errors
  - **hard to distinguish** worms, leaves, and background

TWO CHALLENGES

Food status **Classification**

Automatic and **Unsupervised Segmentation** of elements in the image

STATE OF THE ART

- Current problems with the manual monitoring of silkworms has led to the need for automated visual analysis.

- However, traditional CNNs are effective, but capture only local spatial features.

- Transformers are effective but also heavy and difficult to implement.

- Hybrid Models like CNN + transformers improve accuracy but need large datasets and a complex training.

PROPOSED METHOD

1. BINARY CLASSIFICATION

GOAL: determine whether the silkworms need feeding.

- Application of a binary classification on images of silkworm farms, where each image is labelled as “feeding” or “no feeding”.
- The model chosen is a convolutional network known as **MobileNet_v2**, which due to its efficiency in embedded environments is particularly suitable for this task.
- It was decided to use a version of the model pre-trained on ImageNet using the torchvision library, so as to take advantage of general features already learned.
- The last layer was then replaced by a fully connected layer so as to adapt the network to our case, i.e. “feeding” or “no feeding”.

PROPOSED METHOD

2. UNSUPERVISED SEGMENTATION

GOAL: Implement unsupervised methods to segment silkworms, mulberry leaves, and backgrounds without ground truth.

It is possible to divide the methodology followed into 3 steps:

1. **Feature Extraction with MobileViT**

- Implementation of the **MobileViT** model, which exploits the combination of a CNN and Transformer network applied to feature map patches.
- From the model, features are extracted at an intermediate stage (stage2) and made ready for step 2.

PROPOSED METHOD

2. UNSUPERVISED SEGMENTATION

GOAL: Implement unsupervised methods to segment silkworms, mulberry leaves, and backgrounds without ground truth.

2. Feature clustering

- Features extracted from all images are then randomly sampled and dimensionally reduced with PCA.

- This is where segmentation takes place. Two methods are proposed**: K-means** and Gaussian Mixture Model (**GMM**).

PROPOSED METHOD

2. UNSUPERVISED SEGMENTATION

GOAL: Implement unsupervised methods to segment silkworms, mulberry leaves, and backgrounds without ground truth.

3. Local Segmentation + Augment with Sobel filter

- Finally, the proposed method is used on some images to test its effectiveness. A Sobel filter is also applied to highlight the edges more clearly.

- Features and edges are then concatenated to obtain richer features.

- After repeating PCA and clustering, the results obtained are analysed.

DATASET

- The Dataset used is the **Silkwork rearing data**, provided by the professor in .zip format.
- It contains 1352 images in .jpg format of silkworms, mulberry leaves and background.
- The dataset is loaded via a CSV with the names of the images and their labels, applying resize transformations and conversion to PyTorch tensors.
- A train/validation split (80%-20%) is performed for supervised training, while for unsupervised clustering the entire dataset is used without shuffling.

EXPERIMENTAL SETUP

Dataset

•Silkworm Feeding Dataset (~1300 images)
•Train/Validation split: 80 % / 20 % (≈ 2,161 / 541 samples)
•Resized to 224 × 224 pixels

Preprocessing

•transforms.Resize((224,224))
•transforms.ToTensor()
•ImageNet normalization (mean & std)

EXPERIMENTAL SETUP

Model Architecture

•MobileNet_v2 pre-trained on ImageNet for binary classification
•Final classification layer replaced with Linear(1280 → 2)
•MobileViT for unsupervised clustering

Training

•Optimizer: Adam (lr = 1e-4, weight_decay = 1e-4)
•Loss: Cross-Entropy
•Batch size: 16
•Max epochs: 10 with **early stopping** (patience = 2 on val loss)

EXPERIMENTAL SETUP

Pruning & Fine-Tuning

- Global unstructured L_1 pruning on all Conv2d and Linear weights
- Pruning levels tested: 0 %, 10 %, 20 %
- Post-pruning fine-tuning: 3 epochs at reduced lr (1e-5)

Metrics & Environment

- Metrics: validation loss, accuracy, “Prune Amount vs. Accuracy” curve
- Hardware: Google Colab GPU (Tesla T4), inference time ≈ 15 s per epoch on CPU

MODEL EVALUATION

Accuracy & Loss

- 97.2 % peak val accuracy, ~0.10 val loss
- Early stopping prevented overfitting

Pruning Analysis

- 10 % sparsity → accuracy ↑ to 95.9 %, inference time ↓
- 20 % sparsity → accuracy drop

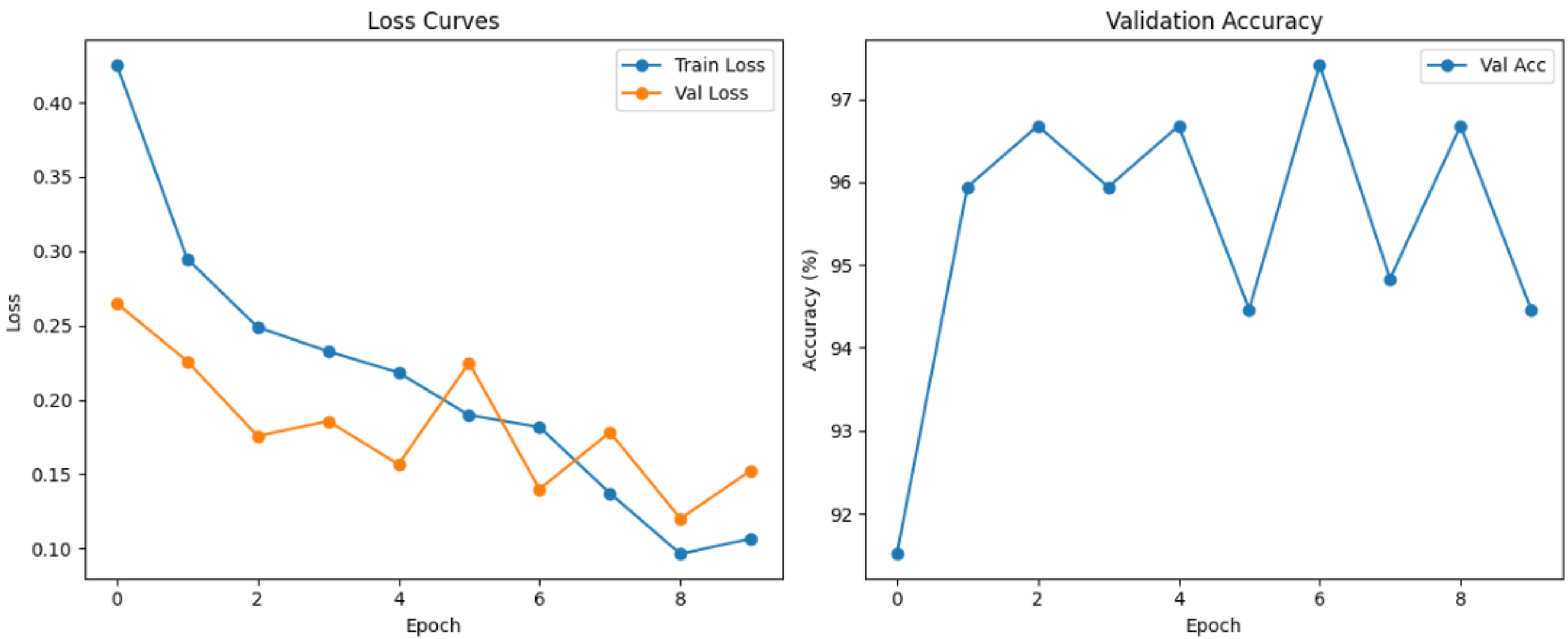
Unsupervised Segmentation

- **GMM vs. K-Means**: cleaner class masks with GMM
- **Clustering metrics**: Silhouette score & Davies–Bouldin index used to quantify segmentation quality

Visualization

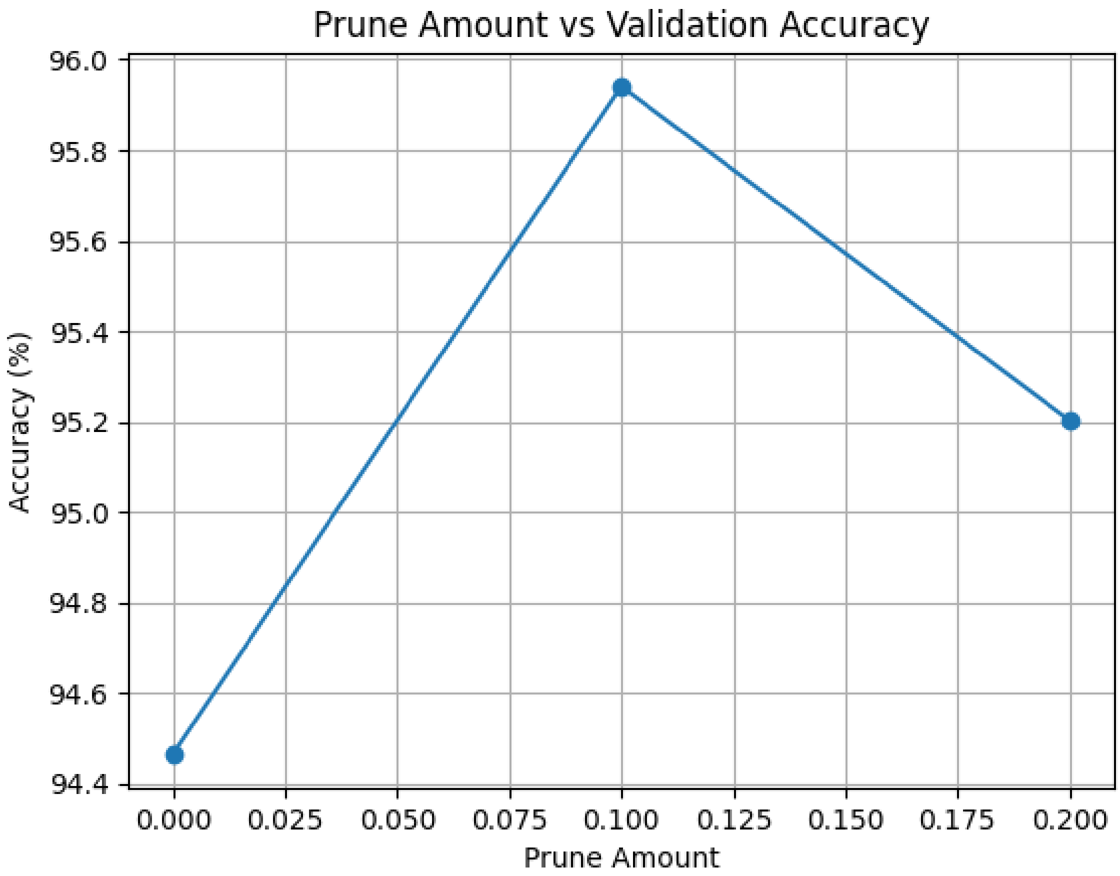
- Training/validation curves
- Accuracy vs. pruning level
- Silhouette vs. number of clusters

MODEL EVALUATION

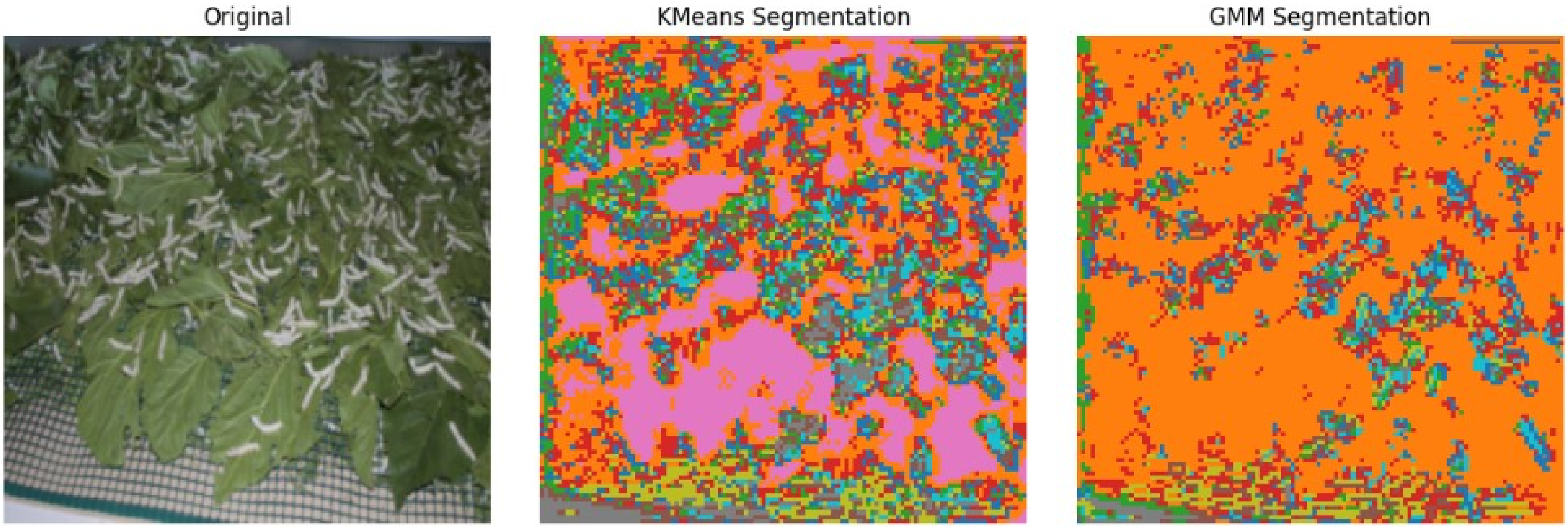


MODEL EVALUATION

```
Prune 0% → Val Acc: 94.46%
Prune 10% → Val Acc: 95.94%
Prune 20% → Val Acc: 95.20%
```



MODEL EVALUATION



CONCLUSION

Strong Baseline
•MobileNet-V2 → 97.2 % val accuracy, smooth train/val curves

Effective Pruning
•10 % sparsity → accuracy ↑ to 95.9 %, inference time ↓
•20 % sparsity → clear performance drop

Segmentation Findings
•GMM delivers higher Silhouette and lower Davies–Bouldin scores vs K-Means
•Cleaner feeding masks, with some over-segmentation remaining

Next Steps
•Integrate segmentation maps into the classifier
•Apply 8-bit quantization & distillation
•Deploy and benchmark on edge hardware

REFERENCES

- Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L.-C. (2018). *MobileNetV2: Inverted Residuals and Linear Bottlenecks*. CVPR.
- Kingma, D. P., & Ba, J. (2015). *Adam: A Method for Stochastic Optimization*. ICLR.
- Frankle, J., & Carbin, M. (2019). *The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks*. ICLR.
- McLachlan, G., & Peel, D. (2000). *Finite Mixture Models*. Wiley.
- Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. (2011). *Scikit-learn: Machine Learning in Python*. JMLR, 12, 2825–2830.
- Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., … & Chintala, S. (2019). *PyTorch: An Imperative Style, High-Performance Deep Learning Library*. NeurIPS.
- Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer. (for GMM foundations)
- Rehurek, R., & Sojka, P. (2010). *Software Framework for Topic Modelling with Large Corpora*. LREC. (for K-Means)

THANK YOU FOR YOUR ATTENTION!