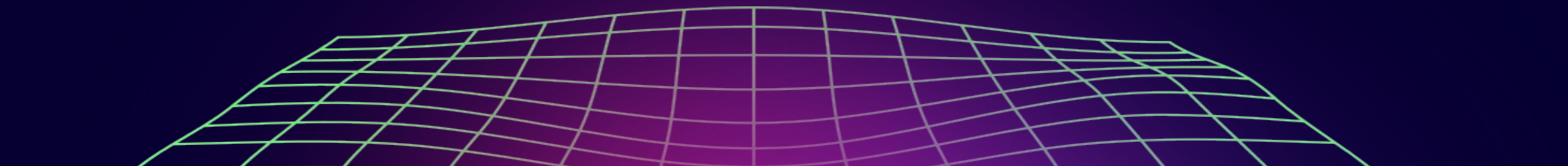
Image-Based Waste Classification with Visual Explanations

By: Maya Kusumakar, Shifanaaz Fazalur

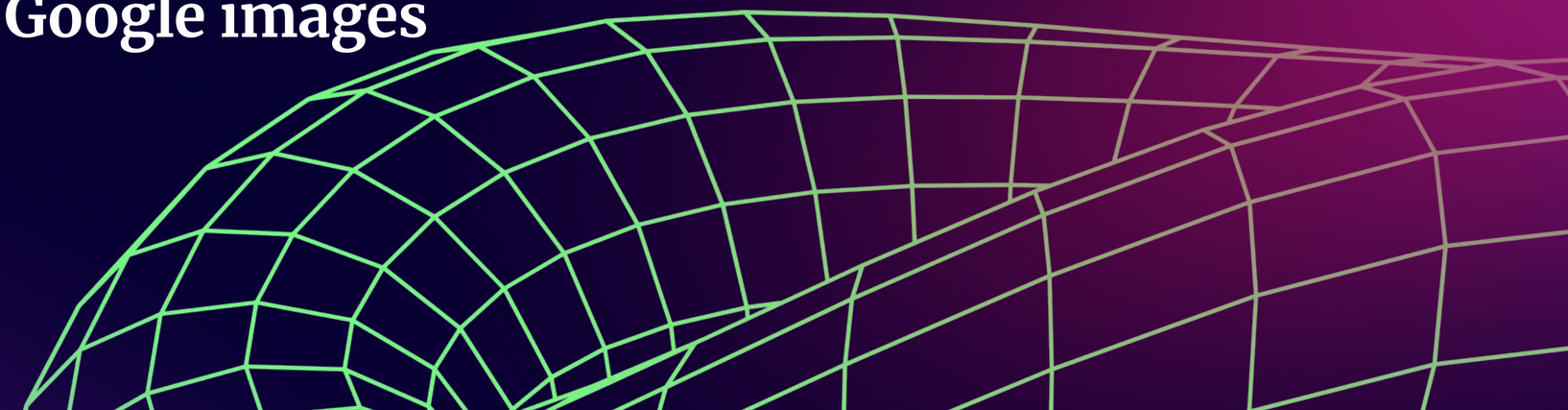
Motivation: Why this project on Waste Classification?

- These days, recycling contamination is a big problem
- People are often unsure how to sort items.
- We wanted a lightweight, interpretable tool

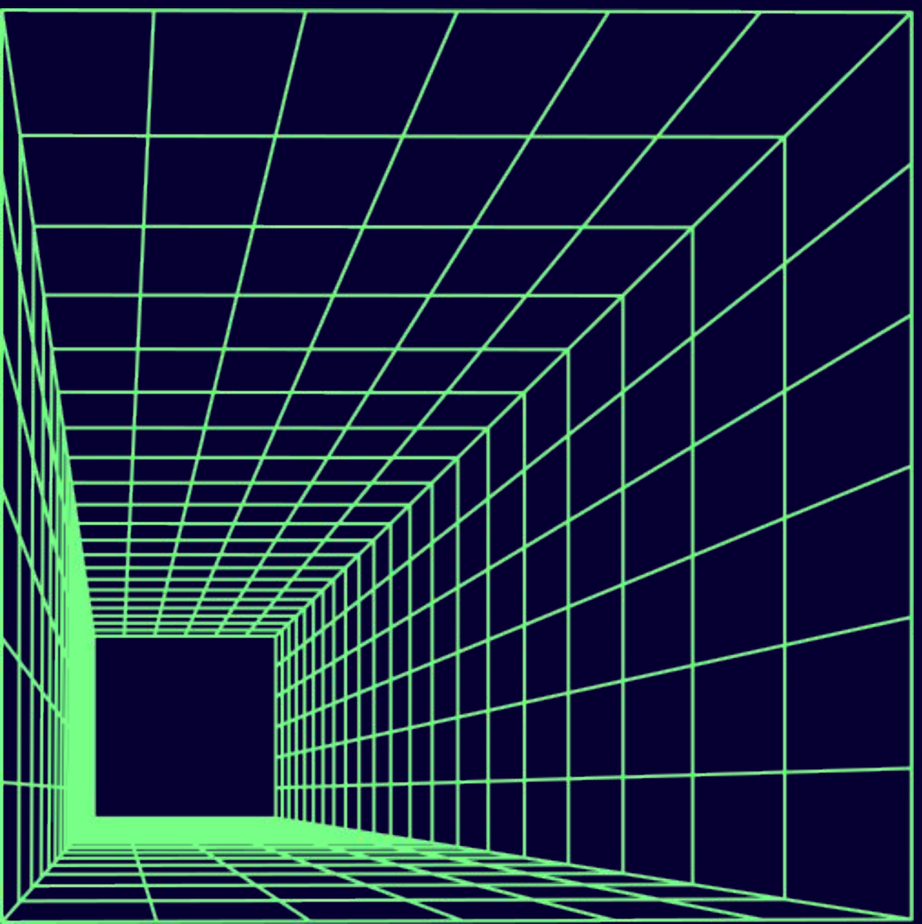


Dataset: Kaggle Waste Classification

- Kaggle “Waste Classification Data”
- 2 labels: Organic vs Recycle
  Train/Val/Test = 70% / 30% held-out evaluation
- Baseline CNN: evaluation split further divided into val + test
- Training (mixed organic/recycle)
- Test set
- Extra check on unseen Google images



Overview of Our Pipeline



- Input RGB Image
- Preprocessing & Data Augmentation
- CNN Classifier
  - Baseline CNN (Keras)
  - MobileNetV2 (PyTorch, transfer learning)
- Output: Recyclable vs Organic & Model Evaluation
  - Accuracy, macro-F1, ROC–AUC, confusion matrix on test set
- Computer Vision Extension: Bounding Box Highlight
  - Grayscale → blur → edges → contours → bounding box
  - Overlay box + predicted label + confidence on original image

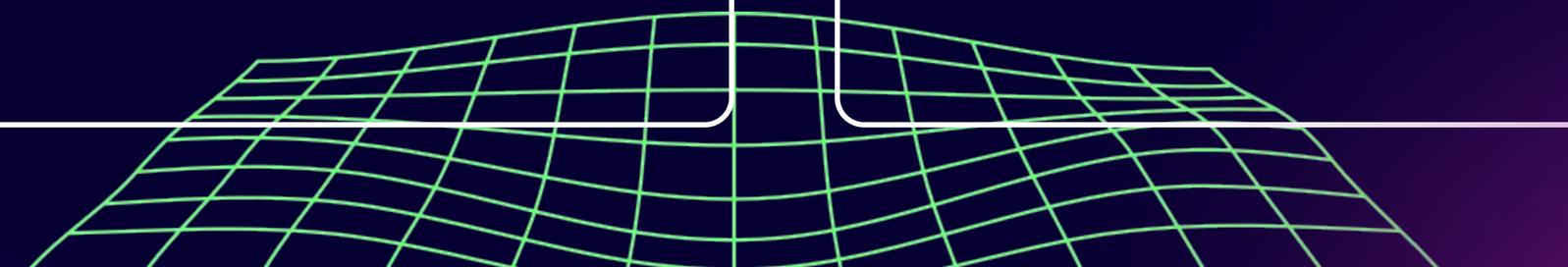
Models: Custom CNN and MobileNetV2

CNN (Keras)

- Input: 65×65×3
- 4 × conv blocks: Conv2D → BatchNorm → ReLU → MaxPool
- Flatten
- Dense(4096) → Dropout(0.5) → Dense(4096)
  Output: Dense(1, sigmoid)
- Code from Kaggle

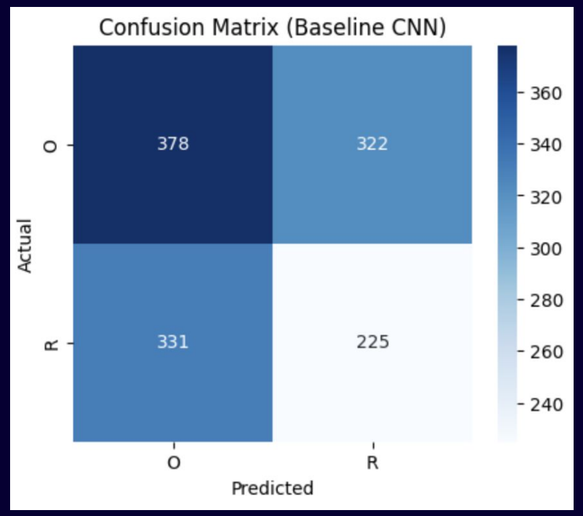
MobileNetV2 (PyTorch, Transfer Learning)

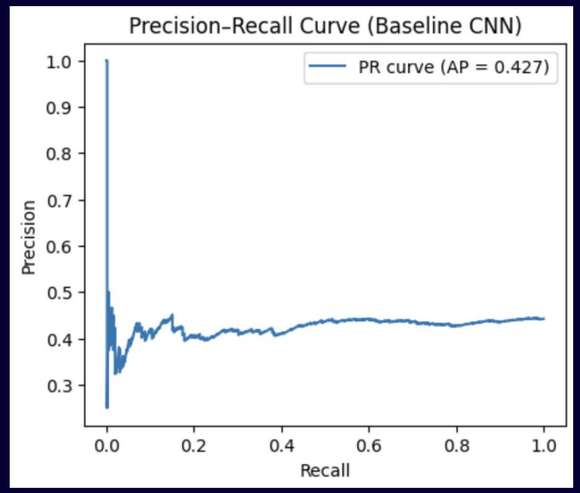
- Pretrained on ImageNet
- Pretrained Layers frozen
- Replace classifier with Linear
- Train layer with class-weighted cross-entropy.
- Code from Kaggle

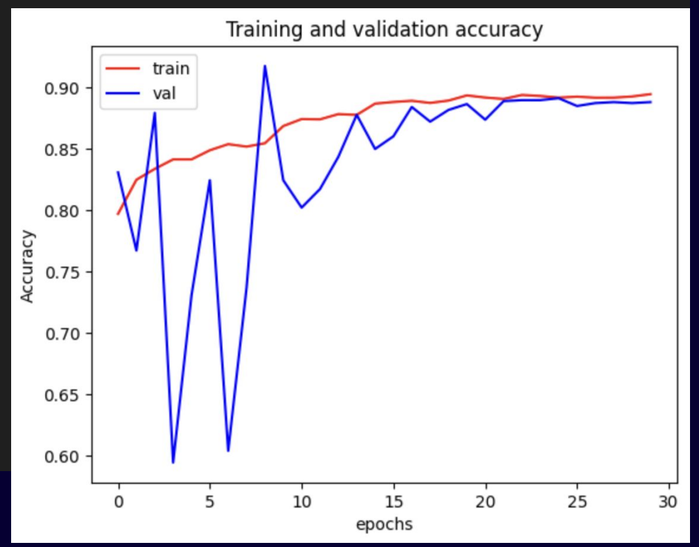


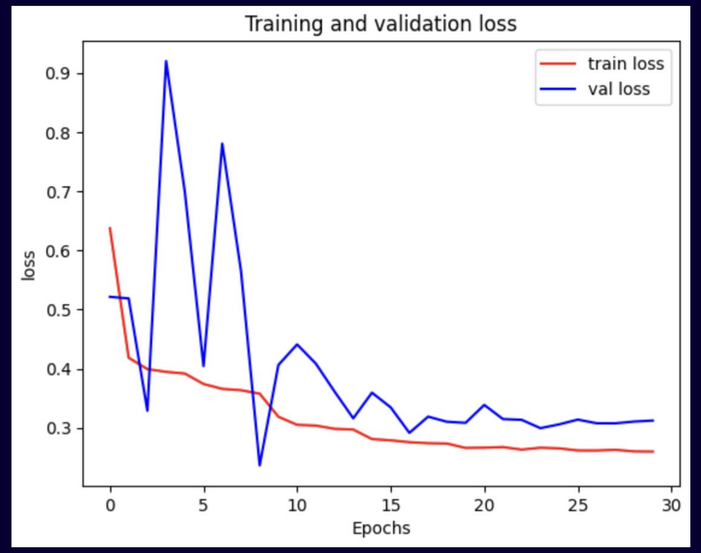
Classification Performance: Baseline CNN

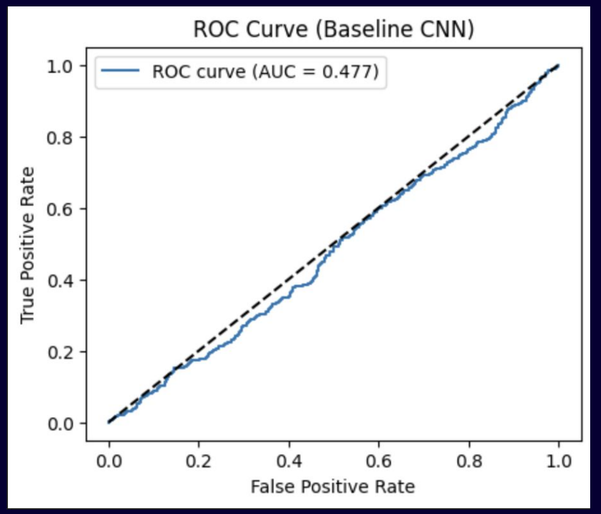
```
Final Model Performance Summary (Baseline CNN, Keras):
--------------------------------------------------------------
Best Validation Loss: 0.2364
Final Metrics:
accuracy : 91.4809
precision: 0.9104
recall: 0.8957
f1 : 0.4080
auc_roc: 0.4771

Per-class Performance Metrics:
--------------------------------------------------------------

Class: O
Precision: 0.5331
Recall : 0.5400
F1-Score : 0.5366

Class: R
Precision: 0.4113
Recall : 0.4047
F1-Score : 0.4080
```





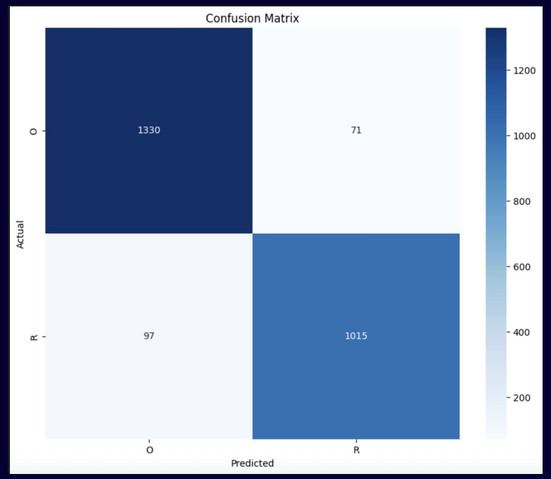


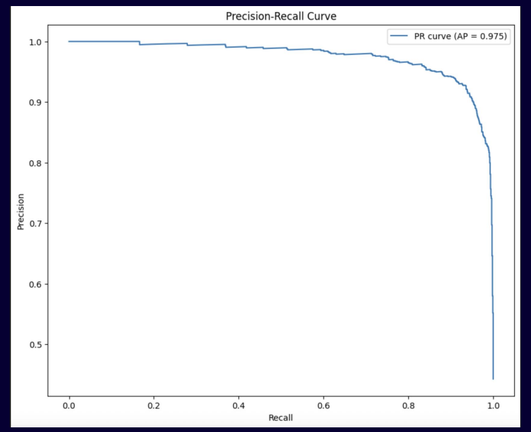


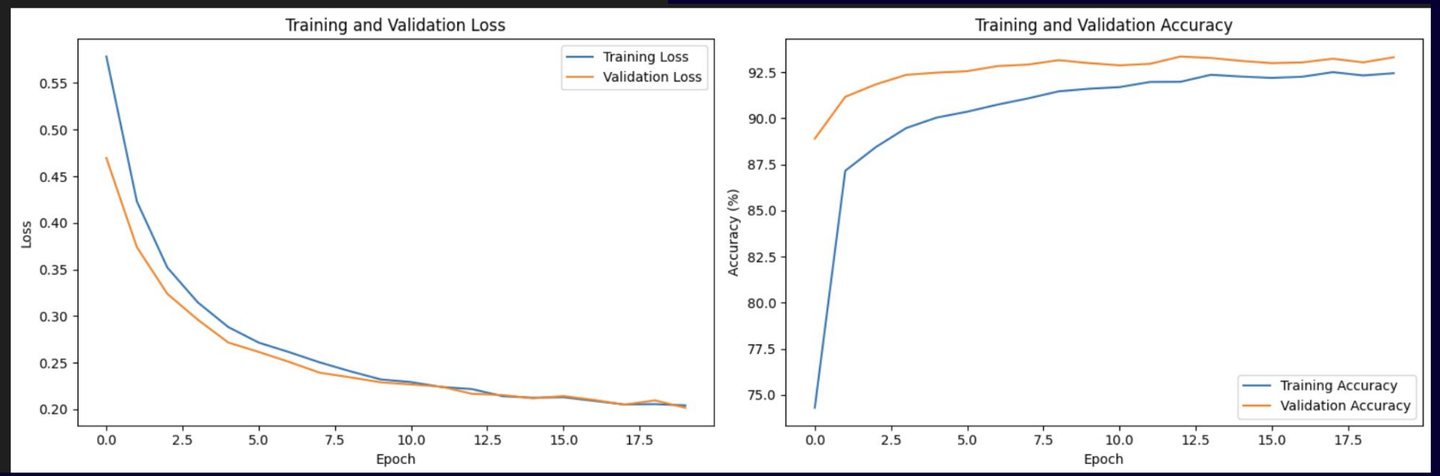


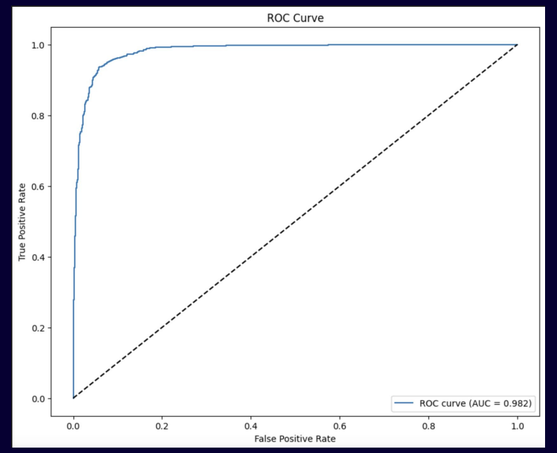
Classification Performance (MobileNetV2)

```
Final Model Performance Summary:
--------------------------------------------------
Best Validation Loss: 0.1973
Final Metrics:
accuracy: 92.8770
precision: 0.9292
recall: 0.9288
f1: 0.9286
auc_roc: 0.9816

Per-class Performance Metrics:
--------------------------------------------------

Class: O
Precision: 0.9208
Recall: 0.9543
F1-Score: 0.9373

Class: R
Precision: 0.9397
Recall: 0.8966
F1-Score: 0.9176
```

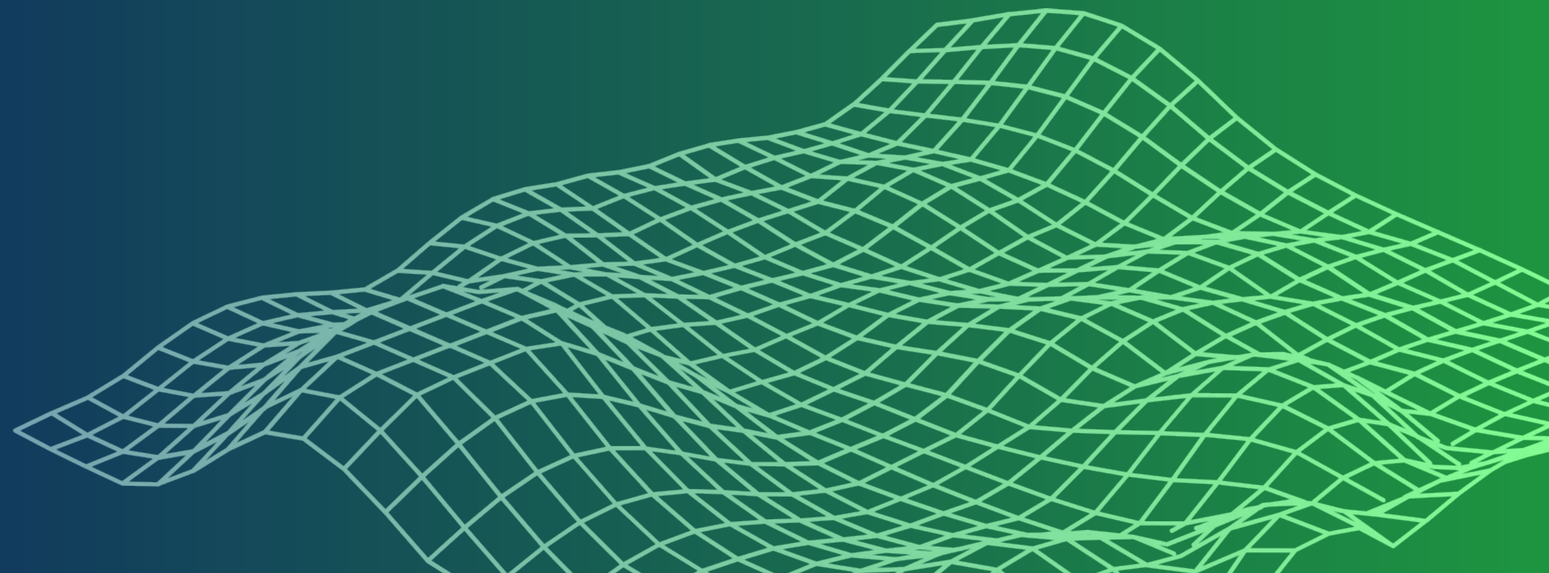






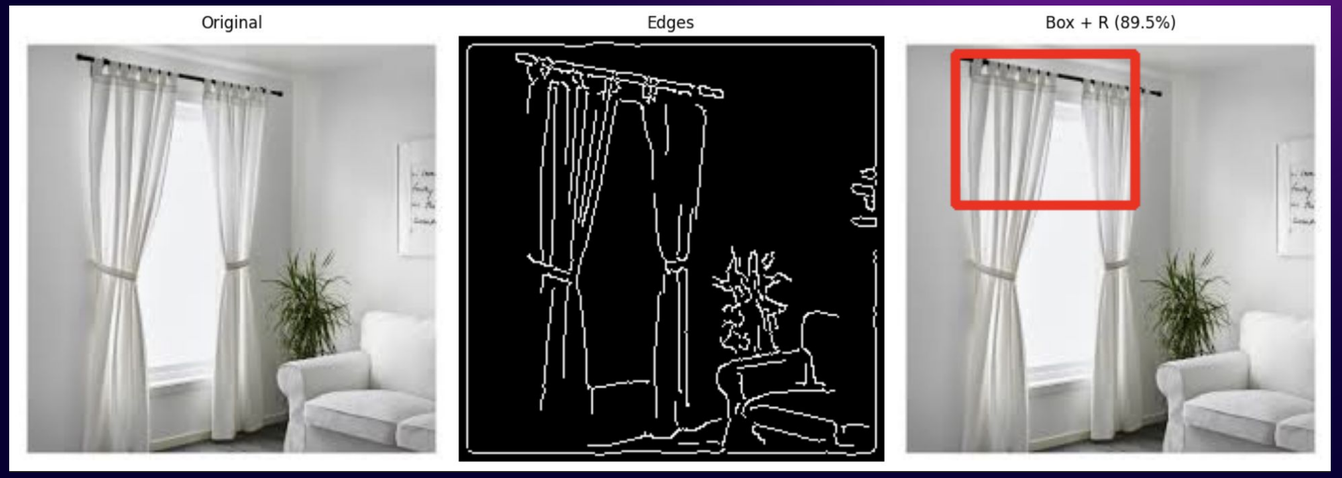


Computer Vision Extension:
Object Highlighting

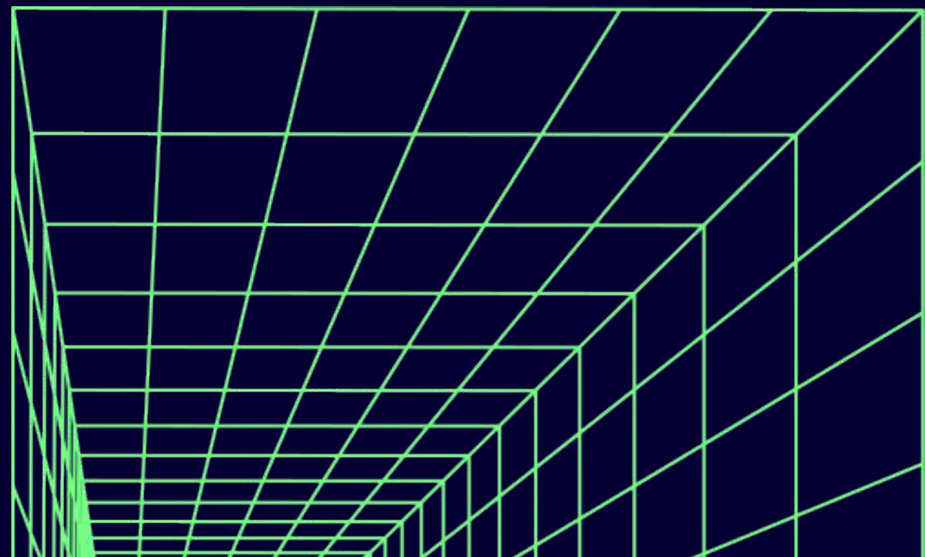


Overview of Our Pipeline

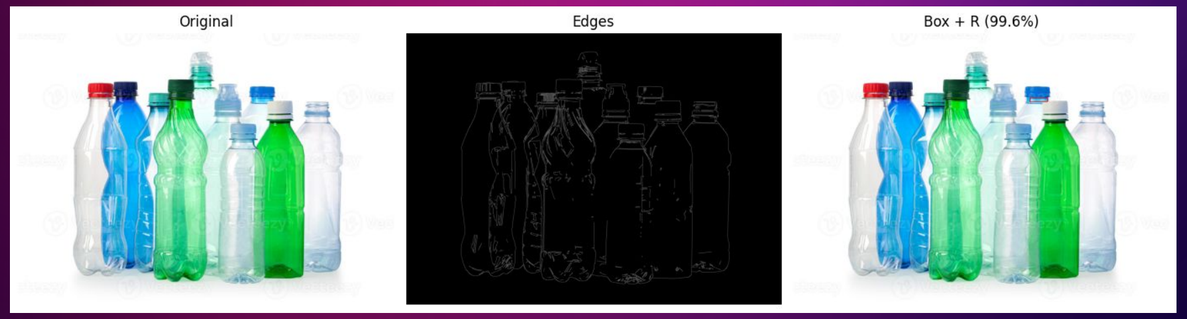
1. RGB → grayscale
2. Gaussian blur
3. Canny edge detection
4. External contours + filtering
5. Bounding box around dominant object

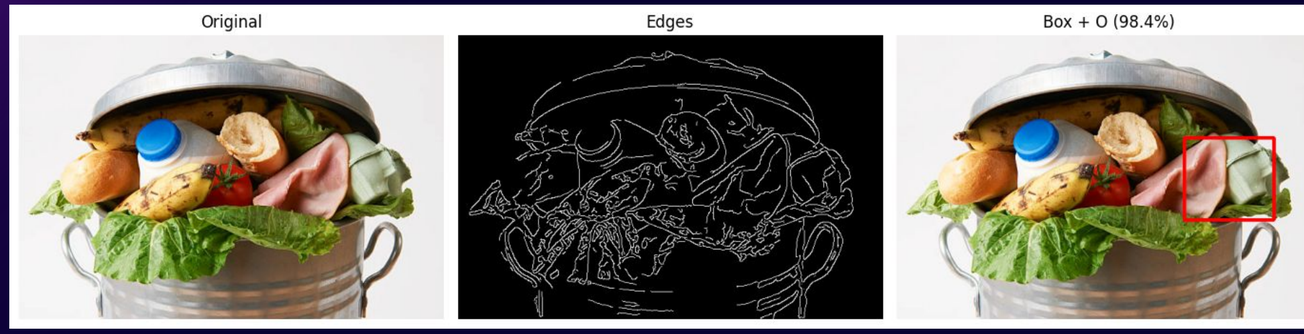


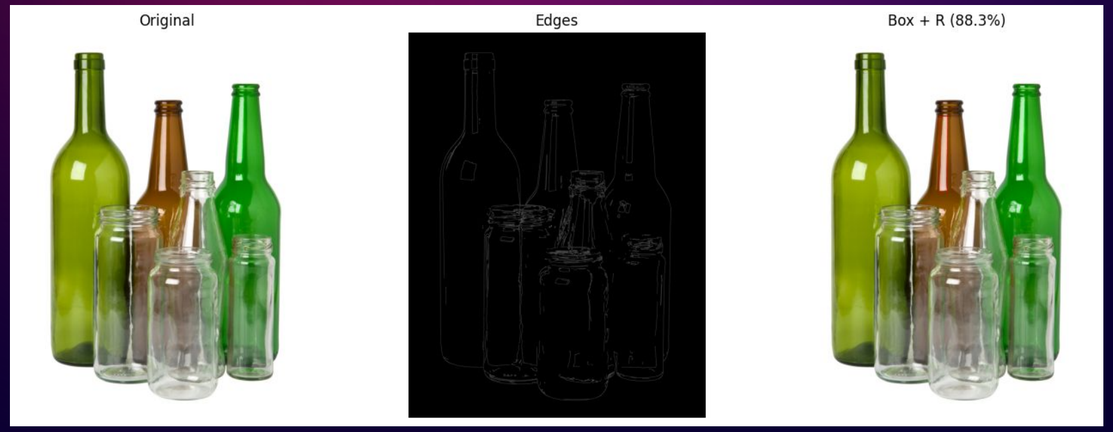
- 3 images side-by-side from the same test image:
  - Original
  - Edges
  - Original + bounding box + “Recyclable (94%)” or similar

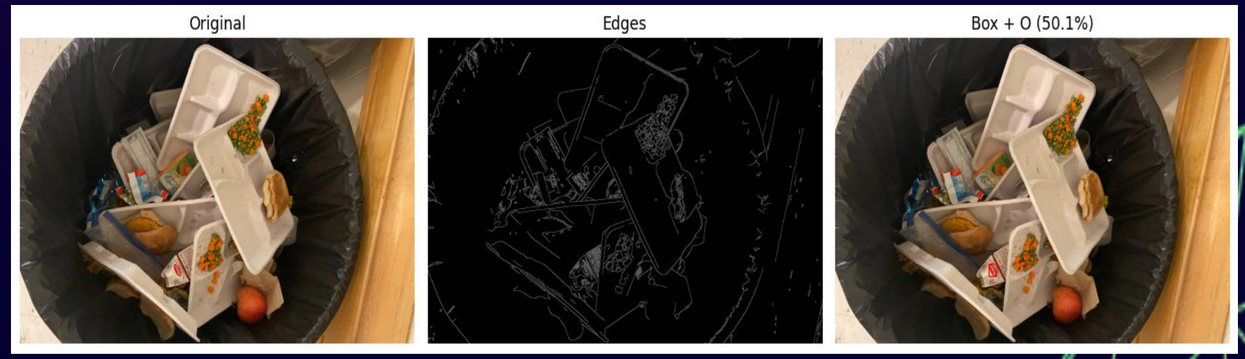


Demo of Google Image





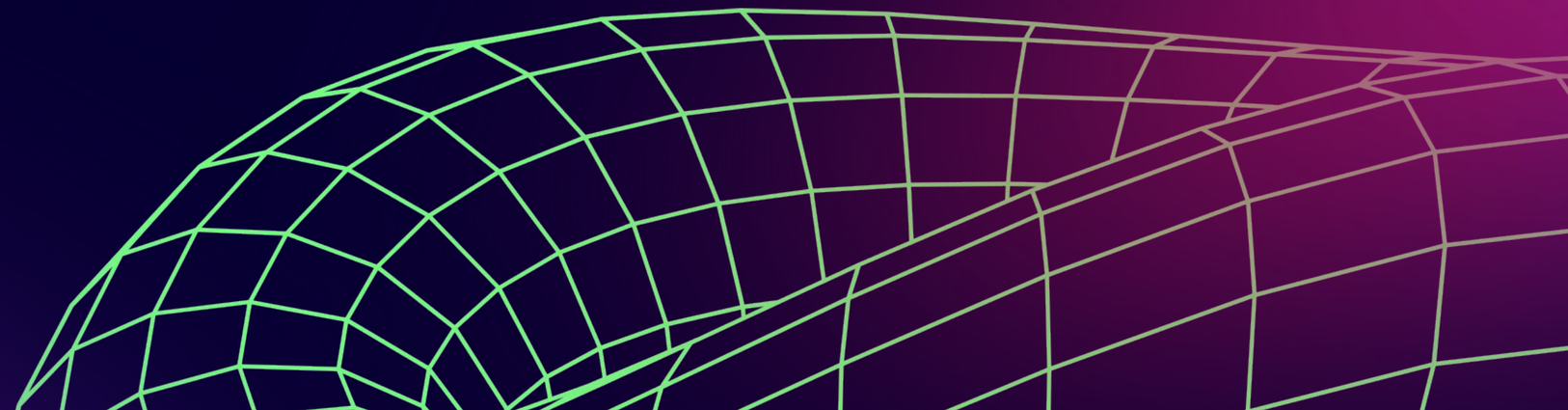




Future Work, and Takeaways

- Future Work:
  - Deploy as a simple web/mobile app.
  - Collect and test on more Images.

- Takeaways:
  - Lightweight CNNs + transfer learning work well for this task
  - Simple bounding boxes give useful visual explanations



Thank You!