Technical Proposal: Dual-Stage Architecture

Tyler Venner

December 15, 2025

1 Component Architectures

We propose a split-architecture approach that decouples *localization* from *identification* and we optimize for each, one stage at a time.

1.1 The Locator: YOLOv8/v11 (You Only Look Once)

Role: High-Recall Object Detection
Objective: Maximize the probability of detecting an insect instance b given an image I, denoted as $P(b \mid I)$.

- **Mechanism:** YOLO treats detection as a single regression problem, simultaneously predicting bounding box coordinates and objectness scores across a grid.
- **Advantages:**
  - **Global Context:** It observes the entire image at once, allowing it to effectively distinguish "insect" from "background noise" (leaves, dirt).
  - **Inference Speed:** The single-shot architecture ensures minimal latency during the scanning phase.
- **Disadvantages:**
  - **Aggressive Downsampling:** To achieve speed, YOLO layers aggressively pool spatial dimensions. This effectively "blurs" high-frequency features necessary for fine-grained family differentiation. Thus, YOLO may not be best for deciphering small differences in insects.

1.2 The Classifier: EfficientNet (B0-B3)

Role: High-Precision Fine-Grained Classification
Objective: Maximize the probability of the correct class c given a specific image crop b, denoted as $P(c \mid b)$.

- **Mechanism:** Uses compound scaling to balance network depth, width, and resolution. Crucially, it maintains higher resolution feature maps deeper into the network compared to standard detectors.
- **Advantages:**
  - **Texture Preservation:** Ideal for Fine-Grained Visual Categorization (FGVC). It can distinguish between families based on micro-textures that YOLO suppresses.
  - **Transfer Learning:** We potentially could use weights pretrained on iNaturalist.
- **Disadvantages:**
  - **No Localization:** It cannot determine *where* the insect is; it assumes the input is purely the object of interest. So EfficientNet will struggle when dealing with images that frame the insect from far away.

2 The Mathematical Framework

We combine the best of both worlds: YOLO for detection and creating bounding boxes and EfficientNet for fine grain classification. We use YOLO for detecting an Insect, and creating a "sub picture" which contains *only* the insect. Then, EfficientNet takes this sub picture, and classifies it. The probability of a correct identification $P(y_{\text{correct}})$ for a given image I is the product of the detection probability and the conditional classification probability:

$$P(y_{\text{correct}} \mid I) = \underbrace{P(C_{\text{correct}} \mid B_{\text{optimal}})}_{\text{EfficientNet Precision}} \times \underbrace{P(B_{\text{optimal}} \mid I)}_{\text{YOLO Recall}} \tag{1}$$

Where:

- B_{optimal} is the event of generating a bounding box that encapsulates the insect with sufficient margin (Recall).
- C_{correct} is the event of assigning the correct taxonomic family given that perfect crop.

Note: While $P(C_{correct} \mid B_{optimal})$ depends on crop quality, enforcing high recall and margin padding makes this approximation valid in practice. In any case, this is simply an abstraction.

2.1 Optimization Strategy

This formula dictates our training strategy for each component, prioritizing different statistical properties at each stage.

2.1.1 Phase 1: Maximize $P(B_{\text{optimal}} \mid I)$ via Recall

For the detector (YOLO), we prioritize **Recall** over Precision. A False Negative (missing an insect) is a "fatal error" for the pipeline; the data is lost forever. A False Positive (cropping a leaf shadow) is a "recoverable error," as the downstream classifier can be trained to reject background noise (assigning it a "None" class or low probability).

We adjust the loss weights to penalize missed objects significantly more than false alarms.

$$\mathcal{L}_{\text{detect}} = \lambda_{\text{coord}} \sum_{i=0}^{S^2} \mathbf{1}_{\text{obj}}^{i} (b_i - \hat{b}_i)^2 + \lambda_{\text{obj}} \sum_{i=0}^{S^2} \mathbf{1}_{\text{obj}}^{i} (C_i - \hat{C}_i)^2 \tag{2}$$

For clarity, we present a simplified detection loss; modern YOLO variants (v8/v11) employ a slightly different losses, but the optimization principle remains identical.

Variable Definitions:

- S^2: The number of grid cells in the YOLO detection layer (e.g., 13×13).
- $\mathbf{1}_{\text{obj}}^{i}$: An indicator function that equals 1 if an object appears in cell i, and 0 otherwise. This ensures we only penalize coordinate errors when an insect is actually present.
- $b_i, \hat{b}_i$: The ground truth and predicted bounding box vectors (x, y, w, h).
- $C_i, \hat{C}_i$: The confidence score that an object exists in the box.
- λ_{coord}: A hyperparameter weight (e.g., 5.0) heavily penalizing poor localization.
- λ_{obj}: A hyperparameter weight balancing the objectness score.

2.1.2 Phase 2: Maximize $P(C_{\text{correct}} \mid B_{\text{optimal}})$ via Spatial Resolution

For the classifier (EfficientNet), we optimize for **Fine-Grained Feature Extraction**.

- **Rationale:** Insect taxonomy often depends on micro-features (e.g., wing venation patterns, tarsal segmentation) that are indistinguishable at low resolution. By cropping and resizing the image, we artificially increase the *effective resolution* of the insect relative to the neural network.
- **Context Preservation:** We can apply a dilation factor to the crop to prevent "clipping" peripheral features like antennae.

$$b_{\text{input}} = b_{\text{YOLO}} \cdot (1 + \delta) \tag{3}$$

Variable Definitions:

- b_{YOLO}: The tight bounding box coordinates returned by the detection phase.
- δ: The padding coefficient (maybe set to ≈ 0.15). This adds a 15% contextual margin around the insect, ensuring that elongated features (legs, antennae) are not severed by the crop boundary.

Although EfficientNet requires fixed-size inputs, variability in insect size is handled through cropping and resizing, which normalizes scale and increases effective resolution for fine-grained features. To mitigate artifacts from extreme upsampling or downsampling, we could employ scale-aware augmentation, enforce minimum crop sizes, and preserve contextual margins.