<u>Part 2: Reasoning-Based Questions</u>

Q1: Choosing the Right Approach

You are tasked with identifying whether a product is missing its label on an assembly line. The products are visually similar except for the label.

Q: Would you use classification, detection, or segmentation? Why? What would be your fallback if the first approach doesn't work?

Answer:

I would employ Object Detection (e.g., YOLOv8) rather than classification because it provides spatial verification ensuring the label is not just present in the frame, but correctly positioned on the specific Region of Interest (RoI). Classification is prone to "background bias" where the model might learn the conveyor belt texture instead of the label, whereas Semantic Segmentation introduces unnecessary inference latency for a simple binary check. If detection struggles due to subtle label variations, my fallback would be Unsupervised Anomaly Detection (e.g., PaDiM or PatchCore). This approach learns the manifold of "Golden Master" (perfect) samples and identifies any deviation (missing label) as a high-energy anomaly map, effectively bypassing the need to learn specific label features.

Q2: Debugging a Poorly Performing Model

You trained a model on 1000 images, but it performs poorly on new images from the factory.

Q: Design a small experiment or checklist to debug the issue. What would you test or visualize?

Answer:

Poor performance on production data typically indicates Covariate Shift a discrepancy between the training and inference data distributions. My debugging checklist would start by performing a Data Drift Analysis, comparing pixel intensity histograms and sensor metadata (ISO, exposure time) to quantify lighting or motion blur differences. I would also visualize Saliency Maps (Grad-CAM) on false positives to determine if the model is overfitting to spurious background correlations (e.g., specific factory markings) rather than the object features. Finally, I would execute a "Human-in-the-loop" experiment: identify high-uncertainty predictions, annotate a small batch (n=50), and measure if fine-tuning resolves the issue, indicating a need for Domain Adaptation.

Q3: Accuracy vs Real Risk

Your model has 98% accuracy but still misses 1 out of 10 defective products.

Q: Is accuracy the right metric in this case? What would you look at instead and why?

Answer:

Accuracy is a deceptive metric in this context due to inherent Class Imbalance; the model is likely biasing towards the majority class (non-defective) while failing on the critical minority class. In manufacturing, a False Negative (Type II error) represents a critical safety/liability risk, whereas a False Positive (Type I error) is merely an operational yield cost. Therefore, I would disregard accuracy and optimize for Recall (Sensitivity) to ensure near-zero defect escapes. Specifically, I would monitor the F2-Score (which weights Recall higher than Precision) and adjust the confidence threshold to push the decision boundary towards capturing all defects, even at the cost of slightly higher false rejections.

Q4: Annotation Edge Cases

You're labeling data, but many images contain blurry or partially visible objects.

Q: Should these be kept in the dataset? Why or why not? What trade-offs are you considering?

Answer:

These edge cases must be included because "clean" datasets fail to model the aleatoric uncertainty (inherent noise) of real-world production environments. Removing blurry or occluded instances creates a distribution mismatch, causing the model to fail catastrophically when these inevitable conditions occur in deployment. The trade-off is slightly slower convergence and lower validation mAP during training, but significantly higher inference robustness. To manage this technically, I would label these samples but tag them with a "Diffcult" flag, allowing me to potentially down-weight their contribution to the Loss Function or use them specifically for Hard Negative Mining to robustify the feature extractor.