PART_2

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

Ans: I would use **object detection** because it can find **where the label is** on the product. If the label is detected, the product is OK; if not, the label is missing.
If detection doesn't work well (for example, the label is very small or blends with the product), I would use **segmentation** to check the label at the **pixel level**.
Classification only says *what the image is*, not *where the label is*, so it's less reliable here.

You trained a model on 1000 images, but it performs poorly on new images from the factory.
Q 2.: Design a small experiment or checklist to debug the issue. What would you test or visualize?

Ans: Look at factory images and see how they are different from training images.
Check where the model is making mistakes by looking at its predictions.
Make sure the training labels are correct.
Test the model on the same images it was trained on.
Add a few real factory images and retrain.

Your model has 98% accuracy but still misses 1 out of 10 defective products.
Q 3.: Is accuracy the right metric in this case? What would you look at instead and why?

Ans: Accuracy isn't the right metric for detecting defective products because it can be misleading when most products are non-defective. Even with 98% accuracy, the model still misses 1 out of 10 defective products, which is critical in a quality control setting. Instead of accuracy, we should focus on **recall**, which measures the percentage of actual defective products that the model correctly identifies, ensuring fewer defects are missed. **Precision** and **F1 score** are also useful to understand the balance between catching defects and avoiding false alarms, but recall is the most important metric here to ensure safety and product quality.

You're labeling data, but many images contain blurry or partially visible objects.
Q 4.: Should these be kept in the dataset? Why or why not? What trade-offs are you considering?

Ans: Blurry or partially visible images can be tricky. Keeping them helps the model **learn to handle real-world, imperfect cases** and become more robust. But if they're too unclear, they can **confuse the model** and reduce accuracy. The trade-off is between **robustness** and **clean data quality**—so keep some imperfect images, but remove the worst ones.