|  |  |  |  |
| --- | --- | --- | --- |
| Model | Speed | Accuracy | Notes |
| R-CNN | Slow | High | Region proposals + CNN |
| Fast R-CNN | ⚡ Faster | High | Uses shared CNN features |
| Faster R-CNN | ⚡ Faster | High | Introduces Region Proposal Network (RPN) |
| SSD | ⚡ Fast | Medium | Detects objects in one shot |
| YOLO | Super Fast | Good | Uses a single neural network for detection |
|  |  |  |  |

📌 **Object Detection Cheat Sheet**

**Low Accuracy?** ➝ Improve annotations & training data.

**False Positives?** ➝ Adjust confidence threshold.

**Slow Processing?** ➝ Use GPU acceleration.

* **Bad Bounding Boxes?** ➝ Tune IoU threshold.

Extra Tools:

📖 **COCO Dataset** – [cocodataset.org](https://cocodataset.org)  
📖 **YOLO Docs** – [pjreddie.com](https://pjreddie.com)  
📖 **TensorFlow Object Detection** – tensorflow.org

**TensorFlow/Keras** → pip install tensorflow keras

**PyTorch** → pip install torch torchvision

**OpenCV** → pip install opencv-python

**COMMON TOOLS/LIBRARIES**

## **Steps in Object Detection**

1️. **Data Collection** – Gather & annotate images.  
2️. **Preprocessing** – Resize, normalize, and augment data.  
3️. **Model Selection** – Choose YOLO, SSD, Faster R-CNN, etc.  
4️. T**raining** – Feed labeled data, optimize loss function.  
5️. I**nference** – Run model on new images & evaluate performance.

**Key Concepts**

* **Bounding Box:** Rectangle around detected objects.
* **Annotations:** Labeled data used for training models.
* **Confidence Score:** Probability of correct detection.
* **Intersection over Union (IoU):** Measures overlap between predicted & actual boxes.