Pretrained Model

1.Object Detection

Pretrained Faster R-CNN with PyTorch (from TorchVision)

🧠 **Logic Explained:**

- **Model**: A two-stage detector (`Faster R-CNN`) pretrained on COCO (80 common classes).
- **Stage 1** – *Region Proposal Network (RPN)*: Slides over feature maps to propose candidate bounding boxes with "objectness" scores.
- **Stage 2** – *Box classification & refinement*: Each proposed region is classified and its box coordinates fine-tuned.
- **Input pipeline**:
    1. Load image → convert to tensor `[C×H×W]`, batch dim `[1×C×H×W]`.
    2. Feed into model: `model = torchvision.models.detection.fasterrcnn_resnet50_fpn(weights="DEFAULT")`.
    3. Output: a dict with `boxes`, `labels`, `scores`.
- **Usage**: Loop through outputs, filter by score threshold (e.g. 0.5), then draw bounding boxes accordingly.

Link:
https://docs.pytorch.org/tutorials/intermediate/torchvision_tutorial.html?utm_source=chatgpt.com

2.Text Recognition (OCR)

EasyOCR for a simple OCR pipeline *(Adrian Rosebrock, PyImageSearch)*

🧠 **Logic Explained:**

- **Detection + Recognition pipeline all-in-one.**
  - **Initialization**: `reader = easyocr.Reader(['en'], gpu=False)`
  - **Read image**: `results = reader.readtext('image.jpg')`
  - **Outputs**: List of tuples `(bbox, text, confidence)`
    - `bbox`: 4 corner points of detected text region.
    - `text`: recognized string.
    - `confidence`: model's prediction score.
- **Internals breakdown**:
  - **CNN backbone** extracts features.
  - **LSTM** sequences spatial features.
  - **CTC decoding** maps sequences to characters.

Link: https://blog.roboflow.com/how-to-use-easyocr/?utm_source=chatgpt.com

3.Face Detection & Recognition

facenet-pytorch: MTCNN + FaceNet (InceptionResnetV1)

🧠 **Logic Explained:**

- **Face detection**:
  1. **MTCNN** – A 3-stage cascade (P-Net, R-Net, O-Net) to detect bounding boxes and landmarks.

    2. Output: Locations of faces, plus cropped face tensor images.

- **Embedding generation**:

    1. `InceptionResnetV1(pretrained='vggface2').eval()` produces a 512-dim vector per face.

- **Recognition**:

    1. Detect and align faces with MTCNN.

    2. Crop & normalize each face.

    3. Pass through the InceptionResnet model to get embeddings.

    4. Compare embeddings via cosine or euclidean distance.

        - If below a threshold (~0.8), they're the same person.

        - Uses FaceNet’s triplet-loss training principle

Link:
https://www.reddit.com/r/pytorch/comments/xfcrpn/tutorial_faster_rcnn_object_detection_with_pytorch/?utm_source=chatgpt.com