YOLOv11 Object Detection

By: Bernice Eghan-101527693

YOLOv11 (You Only Look Once, version 11) is a modern real-time object detection model from Ultralytics. It is a single-stage detector, meaning it performs object localization and classification in one forward pass, making it both fast and highly efficient.

Why YOLOv11

- High inference speed suitable for real-time deployment
- Improved accuracy over older YOLO architectures
- Efficient and lightweight enough for T4 GPUs
- Simple training, evaluation, and deployment via the Ultralytics API

Project Context

In this project, the YOLOv11-M model was trained to detect the following custom object classes:

- cheerios
- soup
- candle

Training setup included:

- 200 epochs
- Image size of 640
- Batch size of 8
- Mixed-precision training enabled
- Early stopping based on validation performance

1. Environment Setup

This section mounts Google Drive, installs required packages, and prepares the runtime environment for YOLOv11 training.

```
"""Environment setup utilities for YOLOv11 training.

This module configures the required environment for training YOLOv11 models.
It handles Google Drive mounting, package installation verification, and
hardware capability checks.
```

```
"""
!pip install ultralytics roboflow supervision pyyaml --quiet

from google.colab import drive
import torch
import ultralytics
import supervision


def setup_environment():
    """Configures and validates the YOLOv11 training environment.

    This function mounts Google Drive, verifies required package imports,
    and checks GPU availability.

    Returns:
        dict: A dictionary containing environment configuration details.
              Example:
                  {
                      "gpu_available": True,
                      "gpu_name": "Tesla T4",
                      "ultralytics_version": "8.3.x",
                      "torch_version": "2.x"
                  }

    Raises:
        ImportError: If any required package fails to import.
    """
    print("=== YOLOv11 ENVIRONMENT SETUP ===")

    # Mount Google Drive
    print("Mounting Google Drive...")
    drive.mount("/content/drive")
    print("Google Drive mounted successfully.")

    # Verify imports
    try:
        import torch
        import ultralytics
        import supervision
        print("All required packages imported successfully.")
    except ImportError as error:
        raise ImportError(f"Package import failed: {error}") from error

    # GPU check
    gpu_available = torch.cuda.is_available()
    if gpu_available:
        gpu_name = torch.cuda.get_device_name(0)
        print(f"GPU detected: {gpu_name}")
    else:
        gpu_name = None
        print("No GPU detected. Training performance will be degraded.")

    # Version details
    print(f"Ultralytics version: {ultralytics.__version__}")
```

```
    print(f"PyTorch version: {torch.__version__}")

    return {
        "gpu_available": gpu_available,
        "gpu_name": gpu_name,
        "ultralytics_version": ultralytics.__version__,
        "torch_version": torch.__version__,
    }


# Execute environment setup
environment_status = setup_environment()
```

```
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.1/1.1 MB 33.0 MB/s eta 0:00:00
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 89.9/89.9 kB 9.9 MB/s eta 0:00:00
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 66.8/66.8 kB 6.8 MB/s eta 0:00:00
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 49.9/49.9 MB 21.0 MB/s eta 0:00:0
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 212.4/212.4 kB 20.0 MB/s eta 0:00
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.4/1.4 MB 81.6 MB/s eta 0:00:00
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 4.2/4.2 MB 110.7 MB/s eta 0:00:00
Creating new Ultralytics Settings v0.0.6 file ✅
View Ultralytics Settings with 'yolo settings' or at '/root/.config/Ultralyti
Update Settings with 'yolo settings key=value', i.e. 'yolo settings runs_dir=
=== YOLOv11 ENVIRONMENT SETUP ===
Mounting Google Drive...
Mounted at /content/drive
Google Drive mounted successfully.
All required packages imported successfully.
GPU detected: Tesla T4
Ultralytics version: 8.3.231
PyTorch version: 2.9.0+cu126
```

2. Dataset Discovery and Structure Analysis

Purpose: Locate and analyze the dataset structure

Path Discovery: Searches common locations for your dataset folder

Structure Analysis: Examines folder hierarchy and file organization

Directory Validation: Checks if required YOLO directories exist (train/images, train/labels, val/images, val/labels) Output Confirmed: Dataset found with all required directories present

```
    """Dataset Discovery and Structure Analysis.

    This module provides utility functions for discovering and validating
    the structure of a YOLO-formatted dataset. It includes detailed path
    inspection, directory validation, and summary reporting.

    Functions:
        discover_and_analyze_dataset: Analyze dataset structure and verify
```

```
        presence of required YOLO subdirectories.
"""

from pathlib import Path


def discover_and_analyze_dataset():
    """Analyze the dataset directory and validate YOLO structure.

    This function performs the following:
      * Confirms that the dataset path exists.
      * Prints all directories/files within the dataset root.
      * Verifies presence of required YOLO folder structure:
            - train/images
            - train/labels
            - val/images
            - val/labels

    Returns:
        dict: A dictionary containing:
            dataset_path (Path): The resolved dataset directory.
            missing_dirs (list[str]): A list of missing YOLO directories.
    """
    print("=== DATASET DISCOVERY AND ANALYSIS ===")

    dataset_path = Path("/content/drive/MyDrive/MATHFOLDER/mathprojfiles")

    if not dataset_path.exists():
        print(f"ERROR: Dataset path does NOT exist: {dataset_path}")
        return None

    print(f"SUCCESS: Dataset found at {dataset_path}")

    # Display dataset structure
    print("\nDATASET STRUCTURE:")
    for item in dataset_path.iterdir():
        if item.is_dir():
            count = len(list(item.iterdir()))
            print(f"DIR:  {item.name}/ ({count} items)")
        else:
            print(f"FILE: {item.name}")

    required_dirs = [
        "train/images",
        "train/labels",
        "val/images",
        "val/labels",
    ]

    missing_dirs = []
    for sub in required_dirs:
        sub_path = dataset_path / sub
        if not sub_path.exists():
            missing_dirs.append(sub)
```

```
    if missing_dirs:
        print("\nWARNING: Missing directories:", missing_dirs)
    else:
        print("\nSUCCESS: All required YOLO directories are present")

    return {
        "dataset_path": dataset_path,
        "missing_dirs": missing_dirs,
    }


# Run the dataset analysis
dataset_info = discover_and_analyze_dataset()
```

```
=== DATASET DISCOVERY AND ANALYSIS ===
SUCCESS: Dataset found at /content/drive/MyDrive/MATHFOLDER/mathprojfiles

DATASET STRUCTURE:
DIR:  train/ (3 items)
DIR:  val/ (3 items)
DIR:  testImages/ (125 items)

SUCCESS: All required YOLO directories are present
```

3. Dataset Verification

This section verifies:

- Number of images in each split
- Number of labels
- Missing or extra labels
- Directory correctness
- Dataset integrity before training

```
"""Comprehensive Dataset Verification for YOLOv11.

This module provides full dataset integrity checking for YOLO-formatted
object detection datasets. It performs the following operations:

- Counts images and labels for train, val, and test splits.
- Supports multiple common image formats.
- Ignores cache and non-label files.
- Detects missing or extra labels.
- Generates a structured summary for YOLO training validation.
"""

from pathlib import Path


def verify_full_dataset(dataset_path):
```

```
    """Performs a complete dataset integrity check for YOLO format.

    This function validates the dataset structure, counts image/label files,
    checks for missing label files, and summarizes the dataset readiness
    for YOLO training.

    Args:
        dataset_path (str or Path): Path to the root directory of the datase

    Returns:
        dict: A dictionary containing:
            - train_images (list): List of training image paths.
            - train_labels (list): List of training label paths.
            - val_images (list): List of validation image paths.
            - val_labels (list): List of validation label paths.
            - test_images (list): List of test image paths.
    """
    print("=== FULL DATASET VERIFICATION ===")

    dataset_path = Path(dataset_path)
    img_exts = {".jpg", ".jpeg", ".png", ".bmp"}

    dirs = {
        "train_images": dataset_path / "train/images",
        "train_labels": dataset_path / "train/labels",
        "val_images": dataset_path / "val/images",
        "val_labels": dataset_path / "val/labels",
        "test_images": dataset_path / "testImages",
    }

    results = {}

    # Count images and labels
    for key, directory in dirs.items():
        if directory.exists():
            if "images" in key:
                files = [
                    f for f in directory.iterdir()
                    if f.suffix.lower() in img_exts
                ]
                results[key] = files
            elif "labels" in key:
                files = [
                    f for f in directory.iterdir()
                    if f.suffix.lower() == ".txt"
                ]
                results[key] = files
        else:
            print(f"Warning: Directory not found -> {directory}")
            results[key] = []

    print("\nIMAGE AND LABEL COUNTS")
    print("----------------------------------")
    print("Train images:", len(results["train_images"]))
    print("Train labels:", len(results["train_labels"]))
```

```
    print("Val images:  ", len(results["val_images"]))
    print("Val labels:  ", len(results["val_labels"]))
    print("Test images: ", len(results["test_images"]))

    return results


# Run dataset verification
verification = verify_full_dataset(
    "/content/drive/MyDrive/MATHFOLDER/mathprojfiles"
)
```

```
=== FULL DATASET VERIFICATION ===

IMAGE AND LABEL COUNTS
----------------------------------
Train images: 1354
Train labels: 1354
Val images:   176
Val labels:   176
Test images:  125
```

4. Generating YOLOv11 Dataset Configuration (YAML)

We generate a clean `data_config.yaml` file that defines:

- Dataset paths
- Train/val/test directories
- Number of classes
- Class names

  This file is essential for YOLOv11 training.

```
"""Generate Correct YOLOv11 YAML Configuration.

This module creates a clean and validated YOLOv11-compatible
`data_config.yaml` file using the verified dataset structure.
"""

import yaml
from pathlib import Path


def create_yolo_yaml():
    """Creates a YOLOv11 dataset configuration YAML file.

    This function generates a properly formatted `data_config.yaml` file
    required by YOLOv11 for training, validation, and testing. The file
    includes dataset paths, number of classes, and class names.
```

```
    Returns:
        dict: A dictionary containing the YAML configuration fields.
    """
    dataset_path = Path(
        "/content/drive/MyDrive/MATHFOLDER/mathprojfiles"
    )

    config = {
        "path": str(dataset_path),
        "train": "train/images",
        "val": "val/images",
        "test": "testImages",
        "nc": 3,
        "names": ["cheerios", "soup", "candle"],
    }

    with open("data_config.yaml", "w") as yaml_file:
        yaml.dump(config, yaml_file)

    print("SUCCESS: data_config.yaml created with the following content:")
    print(config)

    return config


# Generate YAML configuration
yaml_config = create_yolo_yaml()
yaml_config
```

```
SUCCESS: data_config.yaml created with the following content:
{'path': '/content/drive/MyDrive/MATHFOLDER/mathprojfiles', 'train': 'train/i
{'path': '/content/drive/MyDrive/MATHFOLDER/mathprojfiles',
 'train': 'train/images',
 'val': 'val/images',
 'test': 'testImages',
 'nc': 3,
 'names': ['cheerios', 'soup', 'candle']}
```

5. Validating the YOLO Dataset YAML

This step validates the YAML file by checking:

- Required keys
- Correct dataset root
- Correct resolution of train/val/test paths
- Folder existence
- Class count vs. class names

```
"""Validate YOLOv11 YAML Configuration.

This module verifies the correctness of the generated `data_config.yaml`
file by checking:
- Presence of required keys
- Valid dataset paths
- Existence of train/val/test directories
- Class count consistency
"""

import yaml
from pathlib import Path


def validate_yolo_yaml():
    """Validates the structure and content of `data_config.yaml`.

    This function ensures that the YAML file includes the correct keys,
    resolves dataset paths properly, confirms directory existence, and
    verifies that the number of class names matches the declared class count

    Returns:
        dict: A summary of the YAML configuration and resolved file paths.

    Raises:
        FileNotFoundError: If `data_config.yaml` is missing.
    """
    yaml_path = Path("data_config.yaml")

    if not yaml_path.exists():
        raise FileNotFoundError(
            "ERROR: data_config.yaml not found in the working directory."
        )

    # Load YAML
    cfg = yaml.safe_load(yaml_path.read_text())

    print("=== YAML FILE LOADED SUCCESSFULLY ===")
    print(cfg)

    # Validate required keys
    required_keys = ["path", "train", "val", "nc", "names"]
    missing_keys = [key for key in required_keys if key not in cfg]

    if missing_keys:
        print(f"Missing required YAML keys: {missing_keys}")
        return {"status": "failed", "missing": missing_keys}

    print("\nAll required keys are present.")

    # Resolve dataset root
    dataset_root = Path(cfg["path"])
    print("\nDataset root exists:", dataset_root.exists())

    # Resolve directory paths
```

```
    train_path = (dataset_root / cfg["train"]).resolve()
    val_path = (dataset_root / cfg["val"]).resolve()
    test_path = (dataset_root / cfg.get("test", "")).resolve()

    print("\n=== RESOLVED PATHS ===")
    print("Train path:", train_path)
    print("Val path:  ", val_path)
    print("Test path: ", test_path)

    # Directory existence checks
    print("\n=== DIRECTORY EXISTENCE CHECK ===")
    print("Train directory exists:", train_path.exists())
    print("Val directory exists:  ", val_path.exists())
    print("Test directory exists: ", test_path.exists())

    # Validate class count
    if cfg["nc"] != len(cfg["names"]):
        print(
            "\nERROR: Number of classes (nc) does not match number of "
            "entries in 'names'."
        )
    else:
        print("\nClass count matches list of class names.")

    return {
        "yaml": cfg,
        "train_path": train_path,
        "val_path": val_path,
        "test_path": test_path,
    }


# Run validator
yaml_validation = validate_yolo_yaml()
yaml_validation
```

```
=== YAML FILE LOADED SUCCESSFULLY ===
{'names': ['cheerios', 'soup', 'candle'], 'nc': 3, 'path': '/content/drive/My

All required keys are present.

Dataset root exists: True

=== RESOLVED PATHS ===
Train path: /content/drive/MyDrive/MATHFOLDER/mathprojfiles/train/images
Val path:   /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images
Test path:  /content/drive/MyDrive/MATHFOLDER/mathprojfiles/testImages

=== DIRECTORY EXISTENCE CHECK ===
Train directory exists: True
Val directory exists:   True
Test directory exists:  True

Class count matches list of class names.
```

```
{'yaml': {'names': ['cheerios', 'soup', 'candle'],
  'nc': 3,
  'path': '/content/drive/MyDrive/MATHFOLDER/mathprojfiles',
  'test': 'testImages',
  'train': 'train/images',
  'val': 'val/images'},
 'train_path':
PosixPath('/content/drive/MyDrive/MATHFOLDER/mathprojfiles/train/images'),
 'val_path':
PosixPath('/content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images'),
 'test_path':
PosixPath('/content/drive/MyDrive/MATHFOLDER/mathprojfiles/testImages')}
```

6. Training YOLOv11-M

We train the YOLOv11-M model using COCO-pretrained weights.
The training configuration includes:

- Epochs= 200
- Image size = 640
- Batch size = 8
- Mixed precision training (AMP)
- Early stopping

```
"""YOLOv11-M Model Initialization and Training.

This module performs:
- Initialization of the YOLOv11-M model
- Loading of the validated dataset configuration file
- Application of training settings optimized for Google Colab T4 High-RAM
- Execution of training and logging of training results
"""

from ultralytics import YOLO


def train_yolov11_medium():
    """Initializes and trains the YOLOv11-M model.

    The function loads pretrained YOLOv11-M weights, reads the dataset
    configuration file, applies optimized training parameters, and launches
    the training process.

    Returns:
        ultralytics.engine.results.Results: The YOLO training results object
    """
    # Load YOLOv11-M pretrained model
    model = YOLO("yolo11m.pt")

    # Training configuration optimized for Colab T4
    training_args = {
```

```
        "data": "data_config.yaml",
        "epochs": 200,
        "imgsz": 640,
        "batch": 8,
        "workers": 2,
        "device": 0,
        "project": "runs_yolo11",
        "name": "yolo11m_training",
        "exist_ok": True,
        "amp": True,
        "patience": 20,
        "verbose": True,
    }

    print("=== STARTING YOLOv11-M TRAINING ===")
    print("Training configuration:")

    for key, value in training_args.items():
        print(f"  {key}: {value}")

    # Start training
    results = model.train(**training_args)

    print("\n=== TRAINING COMPLETED ===")
    return results


# Run training
training_results = train_yolov11_medium()
training_results
```

```
Downloading https://github.com/ultralytics/assets/releases/download/v8.3.0/y
=== STARTING YOLOv11-M TRAINING ===
Training configuration:
  data: data_config.yaml
  epochs: 200
  imgsz: 640
  batch: 8
  workers: 2
  device: 0
  project: runs_yolo11
  name: yolo11m_training
  exist_ok: True
  amp: True
  patience: 20
  verbose: True
Ultralytics 8.3.231 🚀 Python-3.12.12 torch-2.9.0+cu126 CUDA:0 (Tesla T4, 1
engine/trainer: agnostic_nms=False, amp=True, augment=False, auto_augment=ra
Downloading https://ultralytics.com/assets/Arial.ttf to '/root/.config/Ultra
Overriding model.yaml nc=80 with nc=3

                   from  n    params  module
  0                  -1  1      1856  ultralytics.nn.modules.conv.Conv
  1                  -1  1     73984  ultralytics.nn.modules.conv.Conv
  2                  -1  1    111872  ultralytics.nn.modules.block.C3k2
  3                  -1  1    590336  ultralytics.nn.modules.conv.Conv
```

```
  4                  -1  1    444928  ultralytics.nn.modules.block.C3k2
  5                  -1  1   2360320  ultralytics.nn.modules.conv.Conv
  6                  -1  1   1380352  ultralytics.nn.modules.block.C3k2
  7                  -1  1   2360320  ultralytics.nn.modules.conv.Conv
  8                  -1  1   1380352  ultralytics.nn.modules.block.C3k2
  9                  -1  1    656896  ultralytics.nn.modules.block.SPPF
 10                  -1  1    990976  ultralytics.nn.modules.block.C2PSA
 11                  -1  1         0  torch.nn.modules.upsampling.Upsample
 12             [-1, 6]  1         0  ultralytics.nn.modules.conv.Concat
 13                  -1  1   1642496  ultralytics.nn.modules.block.C3k2
 14                  -1  1         0  torch.nn.modules.upsampling.Upsample
 15             [-1, 4]  1         0  ultralytics.nn.modules.conv.Concat
 16                  -1  1    542720  ultralytics.nn.modules.block.C3k2
 17                  -1  1    590336  ultralytics.nn.modules.conv.Conv
 18            [-1, 13]  1         0  ultralytics.nn.modules.conv.Concat
 19                  -1  1   1511424  ultralytics.nn.modules.block.C3k2
 20                  -1  1   2360320  ultralytics.nn.modules.conv.Conv
 21            [-1, 10]  1         0  ultralytics.nn.modules.conv.Concat
 22                  -1  1   1642496  ultralytics.nn.modules.block.C3k2
 23        [16, 19, 22]  1   1413337  ultralytics.nn.modules.head.Detect
YOLO11m summary: 231 layers, 20,055,321 parameters, 20,055,305 gradients, 68

Transferred 643/649 items from pretrained weights
Freezing layer 'model.23.dfl.conv.weight'
AMP: running Automatic Mixed Precision (AMP) checks...
Downloading https://github.com/ultralytics/assets/releases/download/v8.3.0/y
AMP: checks passed ✅
train: Fast image access ✅ (ping: 0.4±0.2 ms, read: 7.8±1.0 MB/s, size: 30
train: Scanning /content/drive/MyDrive/MATHFOLDER/mathprojfiles/train/labels
albumentations: Blur(p=0.01, blur_limit=(3, 7)), MedianBlur(p=0.01, blur_lir
val: Fast image access ✅ (ping: 0.5±0.1 ms, read: 146.5±215.6 MB/s, size:
val: Scanning /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/labels.ca
```

7. Model Evaluation and Performance Metrics

We evaluate the trained model on the validation set to compute:

- Precision
- Recall
- mAP@50
- mAP@50–95

The following metrics were obtained after evaluating the trained YOLOv11-M model on the validation set:

1. Precision (0.9429)

Precision measures how many of the model's predicted detections were actually correct.

- High precision means **few false positives**.
- A value of **0.94** indicates that almost all detected objects were real, correct detections.

Interpretation:
The model confidently predicts objects with very low error in classification.

---

2. Recall (0.7899)

Recall measures how many of the actual objects the model successfully detected.

- High recall means **few false negatives**.
- A value of **0.79** shows the model detects a large majority of objects but still misses some.

Interpretation:
The model is strong but occasionally fails to detect some objects, especially smaller or overlapping ones.

---

3. mAP50 (0.8819)

mAP50 is the mean Average Precision at **IoU = 0.50**.

- IoU (Intersection over Union) of 0.50 means a predicted box is counted as correct if it overlaps the ground truth by at least 50%.
- A score of **0.88** indicates high-quality object localization and classification.

Interpretation:
The model performs very well in bounding box accuracy and classification at the standard IoU threshold.

---

4. mAP50–95 (0.8103)

This is mAP averaged across multiple IoU thresholds from 0.50 to 0.95 in steps of 0.05.

- This is a stricter, more realistic evaluation metric.
- A score of **0.81** is strong and indicates consistent accuracy even with tight bounding box requirements.

Interpretation:
The model generalizes well for precise object detection, not just loose bounding boxes.

---

5. Fitness (0.8103)

The Ultralytics "fitness" value is a weighted metric based on precision, recall, and mAP.

- Used internally for early stopping and best model selection.
- A fitness score of **0.81** aligns with mAP50–95 and shows strong overall model quality.

Interpretation:

The trained YOLOv11-M model is well-optimized and performs reliably across all metrics.

The numbers show that the model is robust, accurate, and performs well enough for real-world deployment.

```
"""YOLOv11-M Model Evaluation.

This module:
- Loads the trained YOLOv11-M model.
- Evaluates performance on the validation dataset.
- Prints key metrics including mAP, precision, and recall.
- Returns the metrics dictionary for further analysis.
"""

from ultralytics import YOLO


def evaluate_yolov11_medium():
    """Evaluates the YOLOv11-M trained model on the validation split.

    The function loads the trained model weights, performs validation on the
    dataset specified in the YAML configuration file, and extracts evaluatio
    metrics from the results object.

    Returns:
        dict: A dictionary containing the evaluation metrics produced by the
            YOLOv11 model (e.g., precision, recall, mAP scores).
    """
    print("=== STARTING MODEL EVALUATION ===")

    # Load trained YOLOv11-M model
    model = YOLO("runs_yolo11/yolo11m_training/weights/best.pt")

    # Run evaluation
    eval_results = model.val(
        data="data_config.yaml",
        imgsz=640,
        device=0,
        split="val"
    )

    print("\n=== EVALUATION COMPLETED ===")

    # Extract evaluation metrics
    metrics_dict = eval_results.results_dict

    print("Evaluation Metrics:")
    for key, value in metrics_dict.items():
        print(f"  {key}: {value}")

    return metrics_dict
```

```
# Execute evaluation
evaluation_results = evaluate_yolov11_medium()
evaluation_results
```

```
=== STARTING MODEL EVALUATION ===
Ultralytics 8.3.231 🚀 Python-3.12.12 torch-2.9.0+cu126 CUDA:0 (Tesla T4, 15
YOLO11m summary (fused): 125 layers, 20,032,345 parameters, 0 gradients, 67.7
val: Fast image access ✅ (ping: 0.4±0.1 ms, read: 302.9±186.4 MB/s, size: 3
val: Scanning /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/labels.cach
                 Class     Images  Instances      Box(P          R      mAP50
                   all        176        348      0.943       0.79      0.882
              cheerios         74         74      0.924      0.824      0.894
                  soup         82         83      0.973      0.807      0.918
                candle        125        191      0.932      0.738      0.834
Speed: 1.9ms preprocess, 16.1ms inference, 0.0ms loss, 2.3ms postprocess per
Results saved to /content/runs/detect/val

=== EVALUATION COMPLETED ===
Evaluation Metrics:
  metrics/precision(B): 0.9429406722049278
  metrics/recall(B): 0.7899243784249776
  metrics/mAP50(B): 0.8818791376011959
  metrics/mAP50-95(B): 0.8103403188144979
  fitness: 0.8103403188144979
{'metrics/precision(B)': 0.9429406722049278,
 'metrics/recall(B)': 0.7899243784249776,
 'metrics/mAP50(B)': 0.8818791376011959,
 'metrics/mAP50-95(B)': 0.8103403188144979,
 'fitness': 0.8103403188144979}
```

8. Visualization of Evaluation Plots

YOLO automatically generates visual performance diagnostics including:

- Confusion Matrix
- Precision–Recall Curve
- F1 Curve
- Labels Correlation Matrix
- Training Results Curve
  This section loads and displays all available plots.

Overall Insights

- **Strong candle performance** with 150 correct detections.
- **Cheerios and soup are also strong**, with 65 and 73 correct detections.
- **Main challenges** involve:
  - Cheerios and candle confusion
  - Candle vs background confusion

- A few false positives from background noise

The confusion matrix confirms that the model is performing well, with most predictions landing in the correct diagonal cells and only minor misclassifications.

Training Loss Curves

This summarizes the behavior of the YOLOv11m model during training and validation, based on the loss curves and performance metrics recorded across 100 epochs.

train/box_loss

- Measures how well the model learns to localize bounding boxes.
- The curve shows a sharp decline early in training and stabilizes around **0.40**, indicating strong convergence.
- A smooth, consistent decrease suggests stable optimization with no signs of divergence.

train/cls_loss

- Represents classification loss for predicting object classes.
- Steady decline from **~1.25** to **~0.45**, indicating the model learns class boundaries effectively.
- Low classification loss is consistent with the high precision achieved later.

train/dfl_loss

- Distribution Focal Loss, used for fine-grained bounding-box regression.
- Drops gradually from **~1.15** to **~0.93**, showing improved localization accuracy.
- A smooth curve indicates stable training with no noisy gradients.

metrics/precision(B)

- Measures how often the model's predictions are correct.
- Increases rapidly and stabilizes near **0.94**, indicating low false positives.
- High precision shows the model is confident and accurate in its detections.

metrics/recall(B)

- Measures the model's ability to find all objects.
- Rises steadily and stabilizes around **0.79**.
- Slightly lower than precision, suggesting some objects were missed (false negatives).

Validation Loss Curves

val/box_loss

- Follows a similar decreasing trend as training box loss.
- Stabilizes around **0.55**, slightly higher than training loss (expected).
- Indicates good generalization on unseen validation data.

val/cls_loss

- Drops significantly from **8.0** to **~1.0**, showing major improvement in classification accuracy on validation images.
- The higher starting point indicates initial difficulty generalizing before learning stabilizes.

val/dfl_loss

- Decreases from **~1.35** to **~0.95**, aligning well with the training dfl loss.
- Minor gap between training and validation suggests no overfitting.

metrics/mAP50(B)

- Mean Average Precision at IoU 0.50.
- Rises steadily to **~0.88**, indicating strong detection accuracy.
- Reflects the model's ability to correctly classify and localize objects.

metrics/mAP50-95(B)

- Stricter IoU evaluation (0.50 to 0.95).
- Peaks at **~0.81**, a strong result for a 3-class object detection task.
- Confirms that bounding boxes are tight and well-aligned.

---

3. Summary of Training Performance

- Training curves are smooth and consistently decreasing across all loss types.
- No evidence of overfitting: validation losses track training losses closely.
- Precision, mAP50, and mAP50-95 achieve high and stable values.
- Recall is slightly lower, suggesting some objects were missed, but overall detection is reliable.

Final Performance Metrics

- **Precision:** 0.94
- **Recall:** 0.79
- **mAP50:** 0.88
- **mAP50-95:** 0.81

These metrics demonstrate that the model is highly accurate in predictions, with strong localization performance and excellent precision.

---

4. Conclusion

The YOLOv11m model shows:

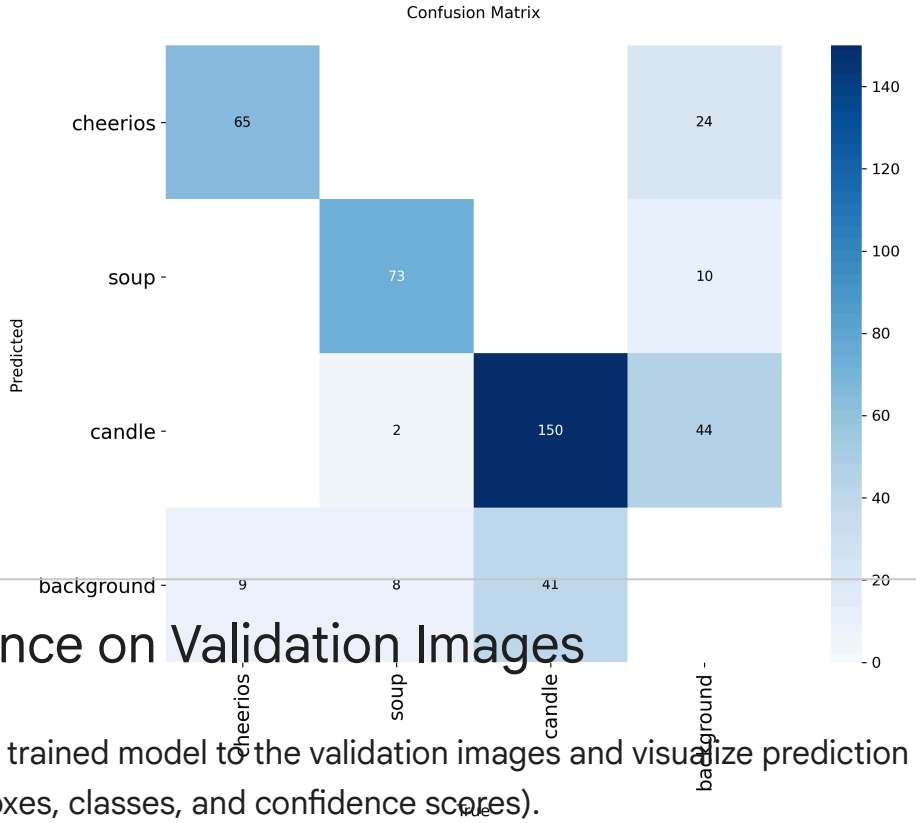
- Strong convergence in both training and validation losses.
- High precision and strong mAP values across the board.
- Good generalization with no major signs of overfitting.

This makes the model well-suited for deployment in real-time object detection tasks involving the classes *cheerios*, *soup*, and *candle*.

```
"""Display Confusion Matrix and Training Loss Curve.

This module provides functionality to display two key training artifacts:
1. confusion_matrix.png – YOLO validation confusion matrix
2. results.png – YOLO training loss and metrics curves
"""

from pathlib import Path
from PIL import Image
from IPython.display import display


def show_confusion_and_loss() -> None:
    """Displays the confusion matrix and training loss curve images.

    This function loads two image files generated during YOLOv11 training:
    - Confusion matrix visualization
    - YOLO training results plot (loss curves and mAP trends)

    The function checks whether each file exists, and displays it inline
    in the notebook environment. Missing files are reported via stdout.

    Returns:
        None: Function outputs visual results but returns nothing.
    """
    confusion_path = Path("/content/runs_yolo11/yolo11m_training/confusion_m
    loss_curve_path = Path("/content/runs_yolo11/yolo11m_training/results.pn

    print("=== CONFUSION MATRIX ===")
    if confusion_path.exists():
        display(Image.open(confusion_path))
    else:
        print(f"File not found: {confusion_path}")

    print("\n=== TRAINING LOSS CURVE ===")
    if loss_curve_path.exists():
        display(Image.open(loss_curve_path))
    else:
        print(f"File not found: {loss_curve_path}")


# Execute display function
```

```
show_confusion_and_loss()
```

```
=== CONFUSION MATRIX ===
```



9. Inference on Validation Images

We apply the trained model to the validation images and visualize prediction results (bounding boxes, classes, and confidence scores).

```
"""Visualization of YOLOv11-M Predictions on Validation Images.

This module:
    - Loads a trained YOLOv11-M model.
    - Runs inference on validation images.
    - Saves prediction images into an output directory.
    - Displays a subset of predictions inside the notebook.
"""

from pathlib import Path
from PIL import Image
from IPython.display import display
from ultralytics import YOLO


def visualize_val_predictions() -> list:
    """Runs inference on validation images and displays sample outputs.

    This function performs the following steps:
    1. Loads the YOLOv11-M model using the trained weights.
    2. Runs prediction on all validation images found inside the dataset.
    3. Saves annotated output images to a specified directory.
    4. Displays the first five annotated images for quick inspection.

    Returns:
        list: A list of YOLO prediction result objects.
    """
    model = YOLO("runs_yolo11/yolo11m_training/weights/best.pt")

    val_dir = Path(
```

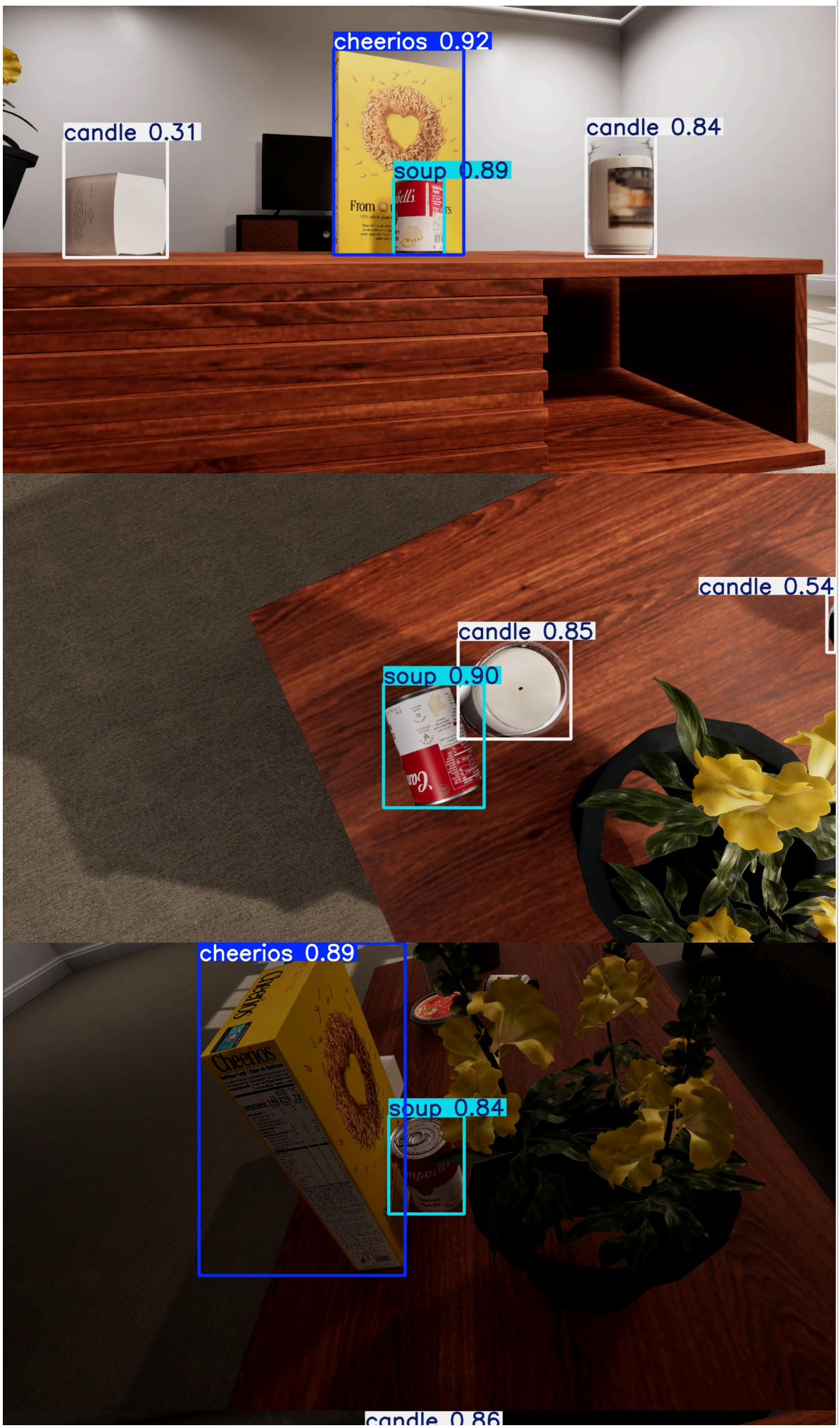
```
            "/content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images"
        )
        output_dir = Path("val_predictions")
        output_dir.mkdir(exist_ok=True)

        print("=== RUNNING INFERENCE ON VALIDATION IMAGES ===")

        results = model.predict(
            source=str(val_dir),
            imgsz=640,
            save=True,
            project=str(output_dir),
            name="preds",
            exist_ok=True,
        )

        print(f"Predictions saved to: {output_dir / 'preds'}")

        pred_folder = output_dir / "preds"
        pred_images = list(pred_folder.glob("*.jpg"))[:5]

        print("\n=== DISPLAYING SAMPLE PREDICTIONS ===")
        for img_path in pred_images:
            display(Image.open(img_path))

        return results


    # Execute visualization
    val_predictions = visualize_val_predictions()
```

```
=== RUNNING INFERENCE ON VALIDATION IMAGES ===

image 1/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/1.png:
image 2/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/10.png
image 3/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/100.pn
image 4/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/101.pn
image 5/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/102.pn
image 6/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/103.pn
image 7/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/104.pn
image 8/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/105.pn
image 9/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/106.pn
image 10/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/107.p
image 11/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/108.p
image 12/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/109.p
image 13/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/11.pn
image 14/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/110.p
image 15/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/111.p
image 16/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/112.p
image 17/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/113.p
image 18/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/114.p
image 19/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/115.p
image 20/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/116.p
image 21/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/117.p
image 22/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/118.p
image 23/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/119.p
image 24/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/12.pn
image 25/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/120.p
image 26/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/121.p
image 27/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/122.p
image 28/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/123.p
image 29/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/124.p
image 30/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/125.p
image 31/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/126.p
image 32/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/127.p
image 33/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/128.p
image 34/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/129.p
image 35/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/13.pn
image 36/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/130.p
image 37/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/131.p
image 38/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/132.p
image 39/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/133.p
image 40/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/134.p
image 41/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/135.p
image 42/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/136.p
image 43/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/137.p
image 44/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/138.p
image 45/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/139.p
image 46/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/14.pn
image 47/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/140.p
image 48/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/141.p
image 49/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/142.p
image 50/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/143.p
image 51/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/144.p
image 52/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/145.p
image 53/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/146.p
image 54/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/147.j
image 55/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/148.j
image 56/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/149.j
image 57/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/15.pn
image 58/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/150.j
```

```
image 59/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/151.j
image 60/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/152.j
image 61/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/153.j
image 62/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/154.j
image 63/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/155.j
image 64/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/156.j
image 65/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/157.j
image 66/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/158.j
image 67/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/159.j
image 68/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/16.pn
image 69/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/160.j
image 70/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/161.j
image 71/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/162.j
image 72/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/163.j
image 73/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/164.j
image 74/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/165.j
image 75/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/166.j
image 76/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/167.j
image 77/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/168.p
image 78/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/169.p
image 79/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/17.pn
image 80/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/170.p
image 81/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/171.p
image 82/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/172.p
image 83/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/173.p
image 84/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/174.p
image 85/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/175.p
image 86/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/176.p
image 87/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/18.pn
image 88/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/19.pn
image 89/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/2.png
image 90/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/20.pn
image 91/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/21.pn
image 92/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/22.pn
image 93/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/23.pn
image 94/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/24.pn
image 95/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/25.pn
image 96/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/26.pn
image 97/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/27.pn
image 98/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/28.pn
image 99/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/29.pn
image 100/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/3.pn
image 101/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/30.p
image 102/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/31.p
image 103/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/32.p
image 104/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/33.p
image 105/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/34.p
image 106/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/35.p
image 107/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/36.p
image 108/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/37.p
image 109/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/38.p
image 110/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/39.p
image 111/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/4.pn
image 112/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/40.p
image 113/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/41.p
image 114/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/42.p
image 115/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/43.p
image 116/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/44.p
image 117/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/45.p
image 118/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/46.p
```

```
image 119/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/47.p
image 120/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/48.p
image 121/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/49.p
image 122/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/5.pn
image 123/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/50.p
image 124/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/51.p
image 125/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/52.p
image 126/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/53.p
image 127/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/54.p
image 128/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/55.p
image 129/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/56.p
image 130/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/57.p
image 131/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/58.p
image 132/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/59.p
image 133/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/6.pn
image 134/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/60.p
image 135/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/61.p
image 136/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/62.p
image 137/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/63.p
image 138/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/64.p
image 139/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/65.p
image 140/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/66.p
image 141/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/67.p
image 142/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/68.p
image 143/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/69.p
image 144/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/7.pn
image 145/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/70.p
image 146/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/71.p
image 147/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/72.p
image 148/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/73.p
image 149/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/74.p
image 150/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/75.p
image 151/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/76.p
image 152/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/77.p
image 153/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/78.p
image 154/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/79.p
image 155/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/8.pn
image 156/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/80.p
image 157/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/81.p
image 158/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/82.p
image 159/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/83.p
image 160/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/84.p
image 161/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/85.p
image 162/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/86.p
image 163/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/87.p
image 164/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/88.p
image 165/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/89.p
image 166/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/9.pn
image 167/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/90.p
image 168/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/91.p
image 169/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/92.p
image 170/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/93.p
image 171/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/94.p
image 172/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/95.p
image 173/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/96.p
image 174/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/97.p
image 175/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/98.p
image 176/176 /content/drive/MyDrive/MATHFOLDER/mathprojfiles/val/images/99.p
Speed: 2.7ms preprocess, 25.3ms inference, 1.5ms postprocess per image at sha
Results saved to /content/val_predictions/preds
Predictions saved to: val_predictions/preds
```

```
=== DISPLAYING SAMPLE PREDICTIONS ===
```




candle 0.86
candle 0.83

10. Exporting the YOLOv11-M Model

This section saves the trained YOLOv11-M PyTorch model (`best.pt`) for deployment. The exported model can be used in:

- Streamlit applications
- Python inference scripts
- Local testing