Assignment 07 : YOLO

UTA027 : Artificial Intelligence

Raghav B. Venkataramaiyer

Mar '25

A07 : Object Detection with YOLO 5-May/12-May

1 Introduction

1.1 Background

Object detection is a computer vision technique that identifies and locates objects within images or videos. It goes beyond image classification by drawing bounding boxes around detected objects, enabling machines to "see" and understand their environment. This is crucial for applications like autonomous driving, surveillance, robotics, and medical imaging, where identifying object locations is essential for decision-making and automation.

The YOLO (You Only Look Once) family is a popular series of object detection models known for their speed and efficiency. YOLOv8 is a cutting-edge version that offers state-of-the-art performance.

YOLO models are particularly well-suited for this task because they process the entire image in a single pass through the neural network. This design choice makes them significantly faster than other object detection methods, enabling real-time performance. Despite their speed, YOLO models, especially YOLOv8, achieve high accuracy, making them ideal for applications requiring both speed and precision.

1.2 Learning Objectives

1. Understand the YOLO architecture.
2. Set up a development environment for YOLO.

3. Prepare a dataset for YOLO training.
4. Train a pre-trained YOLO model on a custom dataset.
5. Evaluate the performance of a trained YOLO model.
6. Perform object detection on images/videos using the trained model.

1.3 Dataset

Use the Pascal VOC Dataset which is available off the shelf with both PyTorch and TensorFlow ecosystem.

The PASCAL VOC dataset is a standard dataset for object detection tasks. It defines a specific set of object classes that models are trained to detect. The classes in the PASCAL VOC dataset are:

Person `person`

Animal `bird`, `cat`, `cow`, `dog`, `horse`, `sheep`

Vehicle `aeroplane`, `bicycle`, `boat`, `bus`, `car`, `motorbike`, `train`

Indoor `bottle`, `chair`, `dining table`, `potted plant`, `sofa`, `tv/monitor`

Dataset preparation and annotation are crucial for the PASCAL VOC dataset's effectiveness in training robust object detection models. High-quality annotations, including accurate bounding boxes and class labels, enable models to learn precise object locations and categories. Consistent annotation across the dataset ensures that models generalize well to unseen images.

Proper dataset preparation, such as splitting the data into training, validation, and test sets, is essential for evaluating model performance and preventing over-fitting.

1.4 Software and Libraries

1. Python 3.x
2. PyTorch or TensorFlow (specify which framework to use)
3. `torchvision` (if using PyTorch) or `tensorflow-datasets` (if using TensorFlow)
4. `ultralytics` (if using YOLOv5 or v8) or other relevant libraries.

5. Suggest using a GPU for training (and how to set up in Google Colab, if needed).

2 Tasks

2.1 Task 1: Setting up the Environment (10%)

1. Provide detailed instructions on setting up the environment. This should include:
   - Installing Python and pip.
   - Installing PyTorch/TensorFlow (with GPU support if available).
   - Installing the necessary libraries (e.g., ultralytics).
2. If using Colab, provide a link to a starter notebook with the basic setup.
3. Include a small test to verify that the environment is set up correctly (e.g., running a simple YOLO command).

2.2 Task 2: Data Preparation (25%)

1. Explain the YOLO dataset format. This is crucial! YOLO requires specific formatting of the annotation files.
2. Provide instructions on how to:
   - Download the dataset.
   - Convert the dataset to the YOLO format (if necessary). Provide code snippets or scripts if possible.
   - Organize the dataset into training and validation sets.
3. If students are using a dataset that needs conversion, point them to tools or scripts that can automate this process.
4. Explain data augmentation techniques (e.g., random flips, rotations, scaling) and how to apply them using the chosen library. (Emphasize why augmentation is important).

2.3 Task 3: Training the YOLO Model (40%)

1. Provide starter code or commands to train a pre-trained YOLO model on the prepared dataset.
2. Explain the key training parameters:
    - epochs
    - batch size
    - learning rate
    - Optimizer (e.g., Adam, SGD)
    - Weight decay
3. Explain how to monitor the training process (loss curves, mAP).
4. Encourage students to experiment with different hyperparameters to improve performance. Ask them to document their experiments.
5. Include instructions on how to save the trained model weights.

2.4 Task 4: Model Evaluation (15%)

1. Explain the evaluation metrics for object detection:
    - Precision
    - Recall
    - Intersection over Union (IoU)
    - Mean Average Precision (mAP)
2. Provide code or instructions on how to calculate these metrics on the validation set.
3. Ask students to analyze the results and discuss the model's strengths and weaknesses.
4. Require students to include a confusion matrix, if applicable, and explain what it shows.

2.5 Task 5: Inference and Visualization (10%)

1. Provide instructions on how to use the trained model to detect objects in new images or videos.

2. Explain how to visualize the results:
    - Drawing bounding boxes around detected objects.
    - Displaying class labels and confidence scores.
3. Ask students to provide examples of the model's output on test images or videos.

3 Submission

- ☐ Specify the files to be submitted:
    - ☐ Code files (Python scripts or Jupyter Notebooks)
    - ☐ A report (in PDF format) that includes:
        - ☐ Introduction
        - ☐ Methodology (explaining the data preparation, training process, and evaluation methods)
        - ☐ Results (including tables and figures showing the evaluation metrics)
        - ☐ Discussion (analyzing the results, discussing challenges, and suggesting future improvements)
- ☐ Trained model weights.
- ☐ Specify the submission deadline and any formatting requirements.

4 Grading Rubric

- ☐ Provide a detailed grading rubric. For example:
    - ☐ Task 1: Setting up the Environment: 10%
    - ☐ Task 2: Data Preparation: 25%
    - ☐ Task 3: Training the YOLO Model: 40%
    - ☐ Task 4: Evaluation: 15%
    - ☐ Task 5: Inference and Visualization: 10%
    - ☐ Report Quality (clarity, organization, completeness): 10%

5 Additional Tips for Students

- Provide links to relevant documentation and tutorials.
- Encourage them to use version control (Git) to manage their code.
- Suggest using a consistent coding style.
- Remind them to comment their code clearly.