1. References to at least Two Scientific Papers Related to Topic:

Paper 1: "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks" by Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun.

Summary: This paper introduces the Faster R-CNN model, which is a widely used method for object detection. The model uses region proposal networks to efficiently identify object locations, offering a fast and accurate detection system.

Link: https://arxiv.org/abs/1506.01497

Paper 2: "YOLO9000: Better, Faster, Stronger" by Joseph Redmon and Ali Farhadi.

Summary: This paper presents YOLO9000, a real-time object detection system capable of detecting over 9,000 categories. The model combines speed with accuracy and it is commonly used for high-performance object detection.

Link: https://arxiv.org/abs/1612.08242

2. Decision on a Topic:

The chosen topic for this project is Object Detection using the PASCAL VOC 2012 dataset. The objective is to detect and classify objects in images by building a deep learning model capable of real-time object detection.

3. Decision on the Type of Project:

The type of project selected is "Bring Your Own Method". In this project, I will use an existing dataset (PASCAL VOC 2012) and implement a state-of-the-art object detection model (such as YOLOv5 or Faster R-CNN). I will train the model and fine-tune it to improve its detection accuracy on the dataset.

4. Written Summary:

a. Short Description of the Project Idea and Approach:

The goal of this project is to implement an object detection model that identifies and classifies multiple objects in an image, localizing each object using bounding boxes. Using the PASCAL VOC 2012 dataset, I will train a deep learning model like YOLOv5 or Faster R-CNN to detect objects in images. The model will be evaluated based on its accuracy, speed, and mean Average Precision (mAP), which is the standard evaluation metric for object detection tasks. I will also fine-tune the model's hyperparameters and optimize its performance for detecting objects from the 20 classes present in the PASCAL VOC dataset.

b. Description of the Dataset:

Dataset Name: PASCAL VOC 2012

Source: Kaggle - PASCAL VOC 2012 Dataset

Size: Around 11,000 images with 20 object categories and 27,000 annotated objects.

Annotations: Each image has XML annotations that provide object class labels and bounding boxes indicating the object's location.

Categories: The dataset contains 20 object categories, including person, car, dog, bicycle, bus, and more.

Usage: This dataset is widely used in computer vision tasks like image classification, segmentation, and object detection. It has high-quality annotations and a diverse set of real-world images.

c. Work Breakdown Structure (WBS) and Time Estimates:

Dataset Collection and Preprocessing:

Task: Download the PASCAL VOC 2012 dataset from Kaggle and preprocess the data.

Steps:

Download and extract the dataset.

Explore the folder structure.

Convert XML annotations (if needed) to a format suitable for the model (YOLO or Faster R-CNN).

Estimated Time: 2-3 days.

Model Selection and Setup:

Choosing a deep learning model for object detection and set up the environment.

Steps:

Decide between YOLOv5 or Faster R-CNN for object detection.

Install necessary dependencies (e.g., PyTorch, TensorFlow, OpenCV).

Load a pre-trained version of the model.

Estimated Time: 3-4 days.

Design and Build the Network:

Implementation of the chosen model architecture.

Steps:

Modify the chosen model architecture to fit the PASCAL VOC dataset (if necessary).

Define the training parameters and hyperparameters (learning rate, batch size, etc.).

Build and compile the model for training.

Estimated Time: 5-6 days.

Training the Model:

Training the model on the PASCAL VOC dataset.

Steps:

Split the dataset into training, validation, and test sets.

Train the model using GPUs (if available) for faster computation.

Track the model’s performance using metrics like loss and mean Average Precision (mAP).

Estimated Time: 1-2 weeks.

Fine-Tuning the Model:

Task: Optimize the model’s hyperparameters and improve its accuracy.

Steps:

Adjust hyperparameters (e.g., learning rate, dropout, batch size).

Use techniques like data augmentation to improve model performance.

Estimated Time: 4-5 days.

Evaluation and Testing:

Evaluate the model on the test set and compute mAP and other metrics.

Steps:

Use the test set to evaluate the final model.

Compare results against baseline models or traditional algorithms.

Visualize predictions by displaying bounding boxes on images.

Estimated Time: 3-4 days.

Application Development:

Task: Build a simple application to showcase the model’s real-time object detection capabilities.

Steps:

Develop a user interface (web or desktop) to allow users to upload images or use a webcam for object detection.

Integrate the trained model into the application.

Estimated Time: 1 week.

Final Report and Documentation:

Task: Write the final report summarizing the project, results, and methodology.

Steps:

Document the project, including the dataset used, model selection, training process, evaluation metrics, and results.

Include visualizations, such as annotated images and precision-recall curves.

Estimated Time: 5-6 days.

Presentation Preparation:

Task: Prepare a presentation to showcase the project results.

Steps:

Create slides summarizing the project's goals, methodology, results, and key insights.

Include visuals, such as graphs and images with predicted bounding boxes.

Estimated Time: 3-4 days.

Summary of Time Estimates:

Dataset Collection & Preprocessing: 2-3 days

Model Selection & Setup: 3-4 days

Network Design & Build: 5-6 days

Training: 1-2 weeks

Fine-Tuning: 4-5 days

Evaluation & Testing: 3-4 days

Application Development: 1 week

Final Report: 5-6 days

Presentation: 3-4 days

Total Estimated Time: Around 5-6 weeks.