CHRIST (Deemed to be University)
Department of Computer Science
Master of Artificial Intelligence and Machine Learning

Course: MAI371 – Deep Learning
Exercise No: LAB Exercise – 8
Date: 10 – 04 – 2024
Duration: 2 Hrs

Question (10 Marks)

Imagine you are working for a smart city project where monitoring traffic flow is crucial for urban planning and management. Your task is to develop an object detection system using YOLO to detect vehicles on city streets from surveillance camera footage. You will utilize a customized dataset consisting of images collected from various traffic intersections.

1. Dataset Preparation:
- Collect images from traffic surveillance cameras capturing various traffic scenarios, including intersections, highways, and urban streets.
- Annotate the images with bounding boxes around vehicles using labeling tools, ensuring accurate annotations.

2. Data Augmentation:
- Apply data augmentation techniques such as random rotation, scaling, and flipping to increase the diversity of the dataset.
- Implement techniques to handle imbalanced classes if present in the dataset.

3. Preprocessing:
- Resize the images to a standardized input size suitable for YOLO.
- Convert annotations to YOLO format, i.e., normalized bounding box coordinates (x_center, y_center, width, height) relative to image dimensions.
- Split the dataset into training, validation, and test sets.

4. Model Configuration:
- Download the pre-trained YOLO weights (YOLO8 or 9).
- Customize the YOLO architecture according to the classes in your dataset and configure anchor boxes suitable for vehicle detection.

5. Training:
- Train the YOLO model on the custom dataset using a deep learning framework like TensorFlow or PyTorch.
- Monitor training progress, track loss metrics, and visualize training/validation performance.

6. Evaluation:
- Evaluate the trained model on the test set to assess its performance.

- Calculate metrics such as mean Average Precision (mAP) ,Intersection over Union (IoU) and Generalized Intersection over Union (GIoU) to measure detection accuracy.
- Analyze detection results, including precision, recall, and F1 score, for each class.

7. Post-processing:

- Implement post-processing techniques such as non-maximum suppression (NMS) to remove redundant bounding boxes and improve detection precision.
- Visualize detection results overlaid on test images to validate the model's performance qualitatively.

Optional:

Fine-tuning and Optimization:(Not Compulsory)

- Explore transfer learning by fine-tuning the pre-trained YOLO model on the custom dataset to improve convergence speed and generalization.
- Experiment with hyperparameter tuning, including learning rate scheduling, batch size adjustment, and optimizer selection.
- Investigate techniques for model optimization, such as quantization, pruning, and model compression, to reduce inference time and memory footprint.

Evaluation Rubrics:

- Dataset Preparation **(3 Marks)**
- YOLO Implementation**(4 Marks)**
- Evaluation**(3 Marks)**

Total:10 Marks

General Instruction:

1. Ensure that your code includes relevant comments to enhance readability and understanding. Subsequently, upload your code to GitHub for version control and collaborative access.
2. Include descriptive comments within the code, explaining its functionality and logic.
3. In the Google Classroom submission, include the GitHub URL where your code is hosted.
4. Attach a PDF document named "your_register_number_exercise_No.pdf" to the submission.The PDF document should include screenshots of the code and the output screen.
5. Upload the answer document&GitHub URL in Google Classroom on or before the deadline mentioned.Evaluation will not be considered for late submission