CS5330 CS 5330 Pattern Recognition and Computer Vision

**Mini-project : MobileNet Image Classfication**

**Project Overview**:

This mini-project involves creating a Python application that utilizes MobileNet, a lightweight convolutional neural network architecture, for efficient image classification on mobile and embedded devices. Students will implement a real-time image classification system using a webcam feed, demonstrating the practical application of MobileNet in resource-constrained environments.

Assignment:

1. **Setup the Environment**
   * Create a folder named "mini-projectX" within your group project GitHub repository.
   * Set up the development environment by installing necessary libraries, including TensorFlow, OpenCV, and NumPy.
   * Familiarize yourself with the WebCamSave.py starter code for capturing video frames from a webcam.
2. **Implement MobileNet for Image Classification**
   * Utilize a pre-trained MobileNet model available in TensorFlow/Keras.
   * Load the pre-trained weights and architecture.
   * Implement the preprocessing steps required for MobileNet input.
3. **Develop the Webcam Application**
   * Modify the WebCamSave.py code to integrate MobileNet for real-time image classification.
   * Ensure the application can capture frames from the webcam and process them through MobileNet.
   * Display the top-k predictions (e.g., top 3) along with their confidence scores on each frame.
4. **Optimize for Real-time Performance**
   * Implement techniques to improve the frame processing speed, such as frame skipping or resizing.
   * Experiment with different MobileNet versions (e.g., MobileNetV1, MobileNetV2) and alpha values to find an optimal balance between accuracy and speed.
5. **Implement a Custom Classification Task**
   * Choose a specific domain (e.g., types of fruits, dog breeds) and fine-tune the MobileNet model on a custom dataset.
   * Document the fine-tuning process and any changes made to the model architecture.
6. **Testing and Video Recording**
   * Record a short video (less than 1 minute) demonstrating the real-time classification of various objects using your application.
   * Upload the video to a cloud service like YouTube or Google Drive and share the link.
7. **Project Documentation**
   * Document the code and provide a README file explaining how to set up and use the application.
   * Include information on the MobileNet version used, any optimizations implemented, and the custom classification task.

**Submission:**

* **GitHub Repo**: Upload the complete project code to the GitHub repository within the "mini-projectX" folder, including a README file with setup instructions and usage guide.
* **Code Submission**:
  + Submit the completed Python code, including modifications to WebCamSave.py and any additional scripts created for the project.
  + Provide clear instructions on how to run the code and any dependencies required.
* **Video Demonstration**:
  + Submit a short video (<1 minute) demonstrating the real-time image classification.
  + Upload the video to a cloud service and include the link in the README file.
* **Project Documentation**:
  + Include a comprehensive README file in your GitHub repository covering:
    - Project description and objectives
    - MobileNet version and configuration used
    - Custom classification task details and dataset used for fine-tuning
    - Performance optimizations implemented
    - Instructions for running the application
    - Link to the video demonstration

**Grading Criteria:**

The project will be graded based on the following (total 30 points):

* MobileNet Implementation (10 points): Correct integration and performance of MobileNet for image classification.
* Real-time Performance (5 points): Efficiency of the application in processing frames and displaying results in real-time.
* Custom Classification Task (5 points): Quality and relevance of the fine-tuned model for a specific domain.
* Video Demonstration (5 points): Clarity and effectiveness of the video in showcasing the application's capabilities.
* Documentation (5 points): Comprehensiveness of the documentation, including setup instructions and explanation of implemented techniques.