DS5007 Deep Learning Lab 2 - CNN

Date: 01/02/2025
Max Marks: 10
Deadline: 07/02/2025, 12:00 PM

Instructions

1. Provide well-commented, indented code with meaningful variable names.
2. Write the task description in separate text blocks before the corresponding code block.
3. Carefully follow the task requirements and use only the specified libraries or approaches.
4. Ensure all plots have appropriate axis labels, titles, and legends.
5. Submit a single Jupyter Notebook (.ipynb) file named YourName_YourRollNo_Assignment2.ipynb.

Task 1: Custom CNN Implementation for Image Classification (6 marks)

Objective: Implement a convolutional neural network (CNN) from scratch using TensorFlow to classify images from the **CIFAR-10 dataset**.

Tasks:

1. **Dataset Preparation**:
    - Load CIFAR-10 and split into training, validation, and test sets.
    - Apply data augmentation (e.g., random horizontal flip).
2. **Model Design**:
    - Define a CNN with the following layers:
        - 2 convolutional layers (number of kernels = 64, kernel size = 3x3, stride = 1, padding = 1).
        - 2 max-pooling layers (kernel size = 2x2).
        - 1 fully connected layer (hidden layer) with (units = 128, dropout = 0.5).
        - Model architecture will be conv layer > pooling > conv layer > pooling > dense layer > output layer
    - Use ReLU activation and batch normalization after each convolutional layer.
3. **Hyperparameter tuning:**
    - Fine-tune the above model to find the optimal parameters that give the best performance.
        - Try with different numbers of convolutional layers/kernels/max pooling layers/dense layers/dense layer units
        - Try changing the activation functions/dropout rate/optimizer

4. **Training**:
    - Train the model using the models before and after hyper-parameter tuning.
    - Report training/validation accuracy and loss curves.
5. **Evaluation**:
    - Compute test accuracy.
    - Compare the performance between models before and after tuning.
    - Plot confusion matrix and classification report.

Task 2: Transfer Learning with Pre-trained Architectures (2 marks)

Objective: Fine-tune any one pre-trained CNN (e.g., ResNet-50, VGG-16, or EfficientNet) on the **CIFAR 10** dataset**..**

Tasks:

1. **Data Preparation**:
    - Load the dataset and split into train/validation/test sets.
    - Resize images to match the input size of the pre-trained model.
2. **Model Setup**:
    - Load a pre-trained model from keras.
    - Replace the final fully connected layer accordingly.
    - Freeze all layers except the final classification layer.
3. **Training**:
    - Train only the unfrozen layers.
4. **Analysis**:
    - Compare test accuracy with your custom CNN from Task 1.
    - Discuss why the pre-trained model performs better/worse.

Task 3: Visualizing Model Decisions with Grad-CAM (2 marks)

Objective: Implement Grad-CAM (Gradient-weighted Class Activation Mapping) to interpret predictions of your **custom CNN from Task 1**.

Tasks:

1. **Grad-CAM Implementation**:
    - Extract feature maps from the last convolutional layer of your custom CNN.
    - Compute gradients of the predicted class score with respect to these feature maps.
    - Generate a heatmap by combining the feature maps and gradients.
2. **Visualization**:
    - Overlay the heatmap on 5 test images from the given dataset.
    - Compare regions highlighted by Grad-CAM with the actual objects in the images.