Fine-Grained Visual Classification of Aircraft (FGVC-Aircraft) Using CNNs and PyTorch

Project Description:

The goal of this project is to use deep learning to classify aircraft photos into fine-grained categories. The FGVC-Aircraft Benchmark dataset, which comprises 102 airplane variants, will be used to train a convolutional neural network (CNN). This task offers a realistic and demanding environment for image classification because of the high visual similarity between classes (e.g., different Boeing 737 variants).

Since the dataset contains 10,200 images of aircraft, with 100 images for each of 102 different aircraft model variants, we will use a subset of 10 randomly selected classes.

The objective is to use PyTorch to create a deep learning model that can correctly classify aircraft into a number of predetermined categories. In order to understand the model's behaviour, we will assess it using common classification metrics and examine misclassifications.

What Makes This Project Good?

- Addresses a fine-grained visual classification problem that arises in the real world.
- Makes use of a publicly available benchmark dataset that is frequently used in studies.
- Uses transfer learning, data augmentation, and model tuning—all contemporary deep learning techniques.
- Closely corresponds with the goals of the course, which include training, analyzing, evaluating, and implementing the model.
- Provides useful experience handling complex and unbalanced visual data.

How is it going to be completed?

- Dataset: FGVC-Aircraft Benchmark dataset (subset of 10 aircraft classes with approximately 100 photos per class).
- Framework: Torchvision, PyTorch, and Python.
- Preprocessing: includes image resizing, normalization, and augmentation (colour jitter, flipping, and rotation).
- Fine Tuning: ResNet-18 pre-trained model (transfer learning with fine-tuning).
- Training: Make use of the Adam optimizer with early stopping and cross-entropy loss.
- Evaluation metrics: include confusion matrix, F1-score, recall, accuracy, and precision.

Plan for Performance Evaluation

- Test the model using 20% of the training data as a held-out validation set.
- Use a different test set to evaluate the trained model.
- Report the accuracy of the validation step and comparing it to the test set accuracy.
- Visualize the top misclassified images and the confusion matrix.
- Adjust the hyperparameters and report the performance impact.