Applied Machine Learning

ASSIGNMENT - 1

HRITIK ANAND

104185477

Methodology

- This assignment requires students to create and evaluate different models for a multi-class image classification challenge. The dataset obtained from Kaggle's "Deep Learning Practice - Image Classification" contains 10,000 color images which are distributed across 10 different classes. The dataset includes 1,000 images for each class and these images are organized into individual folders.
- The original training set was divided into **80% training and 20% validation** using stratified sampling to maintain class balance because there was no test data provided. To enhance model generalization while reducing the risk of overfitting multiple preprocessing methods were employed. The pixel values of all images were standardized to fall within a range of **0 to 1**. The training phase included image augmentation procedures that applied random horizontal flips combined with zoom adjustments, rotations, and brightness modifications to each image.
- Two models were implemented for evaluation: Two evaluation models were used which included a baseline **CNN structure** and a **transfer learning** approach based on MobileNetV2. The baseline CNN structure included three convolutional layers activated by ReLU functions with max pooling operations followed by a flattening layer before reaching two fully connected layers. To prevent overfitting dense layers received dropout layers with a rate of **0.3**. The training process utilized the Adam optimizer and categorical cross-entropy loss for 10 epochs with batches of size 32.
- The MobileNetV2 transfer learning model utilized an ImageNet pretrained backbone after removing its classification head. The initial training only involved adjusting the top layers while keeping the base layers frozen. The top **30% of layers** underwent selective fine-tuning following their initial convergence. The architecture integrated a global average pooling layer which preceded the final softmax output. The learning rate adjustment strategy ReduceLROnPlateau was implemented to decrease the learning rate automatically whenever validation loss stopped decreasing.
- The training process spanned 35 epochs with batch size 32 and utilized mixed precision to utilize Mac GPU capabilities. The model's overfitting was addressed by combining data augmentation with dropout while also fine-tuning and adjusting learning rates. The MobileNetV2 model reached excellent generalization capabilities while maintaining minimal training-validation divergence through these strategies.

Results and Metrics

Transfer learning proved superior to building a CNN from scratch according to the results. The CNN model achieved **23.60% Top-1 Accuracy** while its Average Accuracy per **Class stood at 8.90%.** The model showed poor feature representation learning abilities while demonstrating underfitting characteristics because both training and validation accuracy stayed low and unchanged throughout epochs.

The fine-tuned MobileNetV2 model achieved **Top-1 Accuracy of 68.65%** and **Average Accuracy per Class of 68.80%.** Throughout all training phases the model demonstrated steady improvement while validation accuracy remained close to training accuracy which showed minimal overfitting. The model achieved superior results through its use of pretrained weights along with effective fine-tuning and learning rate scheduling.

Comparative analysis reveals that the CNN model did not possess adequate complexity for a 10-class image classification task with variability, but MobileNetV2 used its pretrained ImageNet features and generalization ability to perform better.

The fine-tuned transfer learning model achieved the highest validation accuracy along with equal performance across different classes making it the most suitable choice for this task.

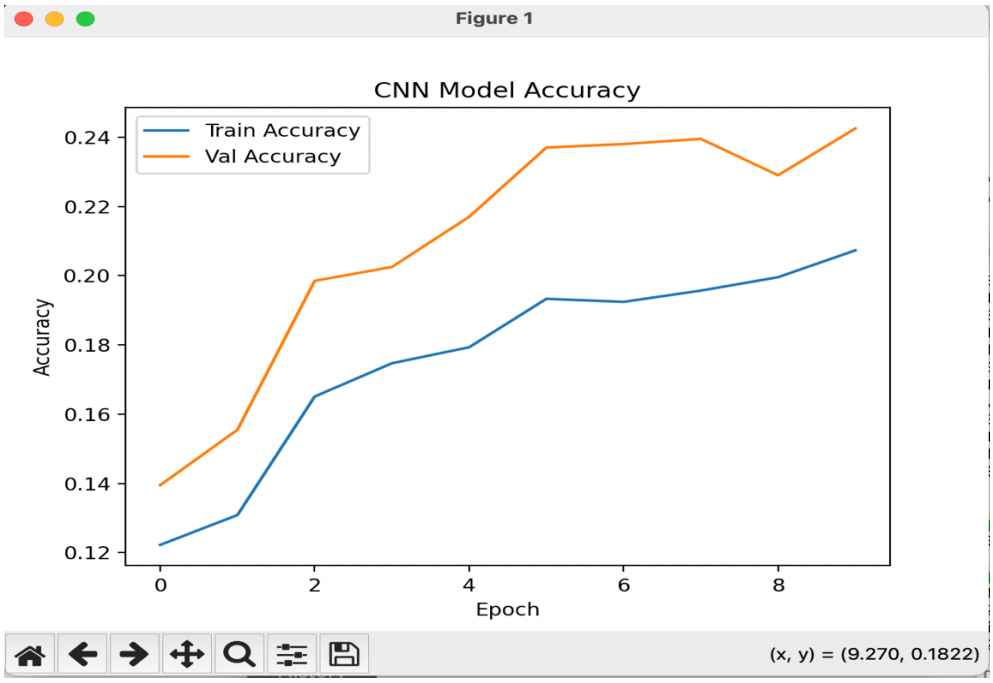
Outputs:

```
Assignment 1 — -zsh — 80×24
126/126 ━━━━━━━━━━━━━━━━━━━━ 406s 3s/step - accuracy: 0.6727 - loss: 0.9949 - va
l_accuracy: 0.6900 - val_loss: 0.9293 - learning_rate: 1.0000e-05
Epoch 21/30
126/126 ━━━━━━━━━━━━━━━━━━━━ 3498s 28s/step - accuracy: 0.6713 - loss: 1.0188 -
val_accuracy: 0.6950 - val_loss: 0.9201 - learning_rate: 5.0000e-06
Epoch 22/30
126/126 ━━━━━━━━━━━━━━━━━━━━ 142s 1s/step - accuracy: 0.6799 - loss: 0.9978 - va
l_accuracy: 0.6955 - val_loss: 0.9276 - learning_rate: 5.0000e-06
32/32 ━━━━━━━━━━━━━━━━━━━━ 24s 734ms/step

📊 CNN Model - Top-1 Accuracy: 0.2360, Avg Accuracy per Class: 0.0890
32/32 ━━━━━━━━━━━━━━━━━━━━ 31s 897ms/step

📊 MobileNetV2 (Fine-Tuned) - Top-1 Accuracy: 0.6855, Avg Accuracy per Class: 0.
1045
2025-05-02 21:12:08.919 python[33693:2019029] +[IMKClient subclass]: chose IMKCl
ient_Modern
2025-05-02 21:12:08.920 python[33693:2019029] +[IMKInputSession subclass]: chose
 IMKInputSession_Modern

✅ Num GPUs Available: 1
(venv) (base) hritikanand@Hritiks-MacBook-Air Assignment 1 %
(venv) (base) hritikanand@Hritiks-MacBook-Air Assignment 1 %
```



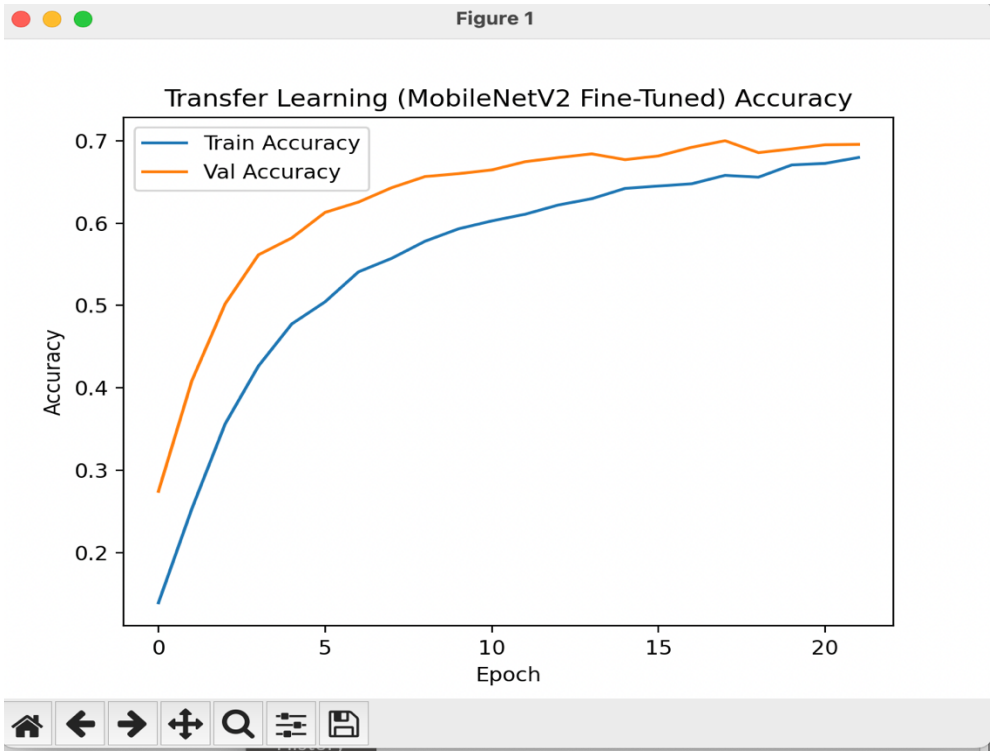


Figure 1
Transfer Learning (MobileNetV2 Fine-Tuned) Accuracy
Train Accuracy
Val Accuracy
Accuracy
Epoch
0.2
0.3
0.4
0.5
0.6
0.7
0
5
10
15
20