Project 2
Machine Learning
Image Classification
Report

Net ID: CRR220000 Name: Chetan Rajendra Rahane

Introduction

The objective of this project was to build a machine learning model to predict the label of the image provide i.e. to classify the image into 13 classes provided in training dataset, based on extracted features from images. The dataset used consisted of 12847 images belonging to 13 classes. To achieve optimal predictive performance, I experimented with multiple machine learning algorithms as base layers and then added extra layers on top of it. The based models from Tensorflow were **MobileNetV2, Resnet50, EfficientNetB0, Inception v3, Densenet121, VGG16** classifiers. **GlobalAveragePooling2D** was added after the base model. Then, the **activation layer** was added using **Softmax**, so that the image will be classified in one of the 13 classes. Optimizer **Adam** was used, along with loss function of **Sparse Categorical Crossentropy** for compiling the model. Later on, after 10 epochs, model was unfreeze and retrained with different parameters for 5 more epochs and tested on data.

After thorough experimentation with different settings, and different unfreezing of layers depending on depths of each model setting, I concluded that the **Resnet50** provided superior performance compared to others. This report outlines the methodology used, the tuning process, and the reasoning behind selecting the final model.

Model Training and Tuning Process

1. Dataset loading

The training file flowers.zip was loaded into code. Tensorflow needs the image dataset to be in format: dataset folder => Classes folders(labels) => Images belonging to those classes(labels)

The provided zip file contained the same format, so it was simple to load the data.

2. Dataset Splitting

The given dataset then split into training and validation sets. 20% of dataset was used for validation and 80% for training. Further, the validation dataset was split, and 1/5th of it was used as test dataset.

3. Data Augmentation

Since, the images in dataset, are considered to be low in size, we needed to increase the size of dataset. For this, Augmentation process was used. For this project, Random flip of vertical and horizontal, Random Rotation, Zoom, Contrast, Brightness and Translation was used for augmenting.

4. Model Selection

I began by experimenting with multiple base models for the classification:

- **MobileNetV2:** A lightweight deep learning model designed for efficient performance on mobile and edge devices. It uses depthwise separable convolutions to reduce the number of parameters and computational cost, while maintaining accuracy for image classification tasks.

- **Resnet50:** A deep convolutional neural network known for its use of residual connections, which help mitigate the vanishing gradient problem and allow for the training of very deep networks. It is highly effective for image recognition tasks and offers strong performance in both accuracy and speed.

- **EfficientNetB0:** A highly efficient convolutional neural network that scales depth, width, and resolution systematically using a compound scaling method. It achieves state-of-the-art performance with fewer parameters and less computation compared to other models, making it well-suited for resource-constrained environments.

- **InceptionV3:** A deep convolutional neural network designed for high performance in image classification tasks. It utilizes an Inception module, which allows for more efficient use of computational resources by combining multiple types of convolutions within a single layer, leading to improved accuracy and computational efficiency.

- **Densenet121:** A deep neural network characterized by its dense connectivity pattern, where each layer receives input from all preceding layers. This design promotes feature reuse, reduces the number of parameters, and improves model performance, especially in tasks requiring complex pattern recognition.

- **VGG16:** A deep convolutional neural network with a simple, uniform architecture that uses small 3x3 convolution filters and a large number of layers. It is well-known for its high accuracy on image classification tasks, though it is relatively computationally expensive due to its large number of parameters.

5. Classification

For each base model, we need to add classification layers. For this, GlobalAveragePooling2D was used. An activation dense layer of Softmax was used to classify the image into 13 classes. Dropouts were introduced, to prevent overfitting. After this, the base setup for the Model is now ready.

Following were the parameters in each of the base models:

MobileNetV2:
Total params: 2,274,637 (8.68 MB)

Trainable params: 16,653 (65.05 KB)

Non-trainable params: 2,257,984 (8.61 MB)

Resnet50:
Total params: 23,614,349 (90.08 MB)

Trainable params: 26,637 (104.05 KB)

Non-trainable params: 23,587,712 (89.98 MB)

EfficientNetB0:
Total params: 4,066,224 (15.51 MB)

Trainable params: 16,653 (65.05 KB)

Non-trainable params: 4,049,571 (15.45 MB)

Inception v3:
Total params: 21,829,421 (83.27 MB)

Trainable params: 26,637 (104.05 KB)

Non-trainable params: 21,802,784 (83.17 MB)

Densenet121:

Total params: 7,050,829 (26.90 MB)

Trainable params: 13,325 (52.05 KB)

Non-trainable params: 7,037,504 (26.85 MB)

VGG16:
Total params: 14,721,357 (56.16 MB)

Trainable params: 6,669 (26.05 KB)

Non-trainable params: 14,714,688 (56.13 MB)

Now, the model needed to be compiled. At this time, I used **Adam** as **optimizer**, **Sparse Categorical Crossentropy** as **loss function** and **accuracy** metric.
Each setting was training with 10 epochs at the beginning. Using all above settings, model was ready for fitting.
After 10 epochs were completed, now the model was fine tuned. The base model was **unfreeze** at different layers, to view which setting gave better accuracy. Optimum setting was chosen. Unfreeze at: MobileNetV2 at 100th layer, Resnet50 at 100th layer, EfficientNetB0 at 180, Inception v3 at 250, Densenet121 at 300, VGG16 at 15. Note: The number of frozen layers were changed during execution and these gave the best result)
After unfreezing, model was trained for another 5 epochs, this time with **another set of parameters** such as Learning rate, Optimizer with new settings **RMSprop optimizer, and learning rate 1/10th of initial learning rate**. After this, final accuracy of model was observed on test dataset.
Results of each setting were as follows:

Base model	Accuracy
MobileNetV2	0.859375

Base model	Accuracy
Resnet50	0.861328125
EfficientNetB0	0.828125
Inception v3	0.78515625
Densenet121	0.845703125
VGG16	0.7070

Based on these result, Resnet50 gave me the best accuracy.

Conclusion

After conducting multiple rounds with different base models, unfreezing at different layers, the **Resnet50** was found to be the base model with highest accuracy for this dataset.

Thus, the **Resnet50** base model with the mentioned setting, and unfreeze at 100 layers for parameter turning was chosen as the final model for the task of classifying flowers. This model is both accurate and efficient, making it a solid choice for deployment in real-world predictive applications.