Image Classification using Transfer Learning

Business Objective

Image classification - Image classification refers to a process in computer vision that can classify an image according to its visual content. Image classification applications include recognizing various objects, such as vehicles, people, moving objects, etc., on the road to enable autonomous driving.

Transfer Learning - The reuse of a pre-trained model on a new problem is known as transfer learning in machine learning. A machine uses the knowledge learned from a prior assignment to increase prediction about a new task in transfer learning.

In this project, we will be dealing with a binary classification dataset. A model is to be built that can correctly classify a t-shirt image into plain or topographic. We will be building transfer learning models like InceptionV3 and Resnet.

Data Description

We have a dataset of t-shirt images categorized into 2 types, namely plain(solid) and topography (contains text)

The training, testing and validation folders contain two subfolders (plain and topography) each.

- Training - around600 images
- Testing - around 100 images
- Validation - around 150 images

Aim

To build a transfer learning model that can perform binary classification on the given dataset of images.

Tech stack

- Language - Python
- Libraries - pandas, numpy, seaborn, matplotlib, sklearn, tensorflow

Approach

1. Importing the required libraries.
2. Load and read the data images
3. Data Visualization
4. Model Training
    - Create a function for initiating Inception and Resnet models
    - Create a function that defines the optimizers
    - Add layers to the pretrained model
    - Function to compile the model
    - Fit and train the model
    - Plot the results
    - Save the model (.h5 file)
5. Create train and validation image data generators
6. Model Testing
    - Create testing images data generator
    - Load the saved model
    - Perform predictions by plotting the confusion matrix
    - Analysing the incorrect predictions

Modular code overview

```
input
  |_images
     |_train
     |_test
     |_valid


src
  |_Engine.py
  |_ML_Pipeline
            |_data.py
            |_evaluate_model.py
            |_model.py
            |_plots.py
            |_predict.py
  |_config
  |_README
  |_requirements.txt


lib
  |_image_classification.ipynb


output
  |_models
  |_plots
```

Once you unzip the modular_code.zip file, you can find the following folders within it.

1. Input
2. Source Folder
3. Output
4. Lib

1. Input folder **-** It contains all the data that we have for analysis. We have three folders. Train, test and valid. The following two subfolders are present;
    - Plain
    - Topography

2. Source folder - This is the most important folder of the project. This folder contains all the modularized code for all the above steps in a modularized manner. This folder consists of:
    - Engine.py
    - ML_Pipeline
      The ML_Pipeline is a folder that contains all the functions put into different python files, which are appropriately named. These python functions are then called inside the engine.py file.

3. Output folder **-** The output folder contains the fitted model that we trained for this data. This model can be easily loaded and used for future use, and the user need not have to train all the models from the beginning.

4. Lib - This folder contains two notebooks.
    - image_classification.ipynb

Project Takeaways

1. Introduction to Deep learning
2. Introduction to CNN
3. The architecture of CNN
4. What is data augmentation?
5. How to perform data visualization
6. Understanding Transfer learning
7. What is Inception and Resnet models?
8. Understanding Hyperparameters
9. What are optimizers?
10. How to build the model?
11. How to fit and train the model over the dataset?
12. How to save the model?
13. Predictions on the test data
14. Evaluating the results with metrics like precision, recall, and accuracy