Task 1: Fine-tune on previously published architectures

In this section, we choose three previously published models (`VGG-16`, `ResNet-50`, and `DenseNet-201`), which were pretrained on `ImageNet`, and perform a fine-tuning on `Oxford Flowers 102` dataset.

The intuition of fine-tuning

If a model is trained on a large and general enough dataset, this model will develop great ability to extract the features of an image, and will effectively serve as a generic model of the visual world. In this task, our dataset only contains about 8,000 samples in total, which is quite small compared with other larger datasets. Instead of training a model from scratch, we might reuse those learned feature maps to perform a task on our smaller dataset.

Data preparation

- **Source** The dataset is fetched using `tensorflow_datasets`.
- **Split** For simplicity, we split the whole dataset into (`train`, `val`, `test`) = (1020, 1020, 6149) formmat, which means we merged the validation set into train set.
- **Balance** Each category has 10 samples in `train` and `val` set, and at least 20 samples in `test` set. The `train` set is well balanced.
- **Resize** To be consist with original papers, we choose image size `img_sz` to be `224`, which is widely used in experiments on ImageNet. The images are resized into `224*224` with **scaling reserved** and **padded** with 0s.
- **Normalization** Since the `Oxford Flowers 102` is much smaller than `ImageNet` and our models were pretrained on `ImageNet`, the normalization is done with the statistics (`pixel_mean`, `pixel_std`) of `ImageNet`. Each image is subtracted by `pixel_mean` and divided by `pixel_std`.
- **Augmentation** We didn't perform any augmentation on `train` set.

Pretrained model and modification/methodology

All three pretrained models are downloaded using `tensorflow.keras.applications` and remove the classification layer. The model serves as `backbone` and is appended a `GlobalAveragePooling2D` layer, a `Dense` layer of 102 neurons with `ReLU` activation, and finally a `Softmax` layer. Which in general is

> `backbone`->`GlobalAveragePooling2D`->`Dense(ReLU)`->`Softmax`

The other possible solution is to replace the `GlobalAveragePooling2D` with a `Flatten` layer, however, it doesn't improve the performance in our experiments and increase the model complexity.

Training

In our experiment, we use `sparse_categorical_accuracy` as loss function and apply the `SGD` optimizer with learning rate equals `0.01`. We ran `100` epochs on each model with an `EarlyStop` and patience equals `20`.

Results

Task 2: Few-shot learning using part of the dataset

In this section, we use `VGG-16` as a backbone and perform `5-way-1-shot`, `5-way-5-shot`, `102-way-1-shot`, `102-way-5-shot` learning and analyse the result.

The intuition of few-shot learning

Some data are difficult and expensive to collect or it is impossible to collect enough samples. Machine learning alrothims (neural networks, for example) are often hampered when they are not "fed" with enough data. Few-shot learning is proposed to tackle this problem. Using prior knowledge, FSL can rapidly generalize to new tasks containing only a few samples with supervised information. (Wang, 2019).