Flower images

Machine Learning

Claudio Cusano

A.A. 2019/2020

Image recognition is a very important application of machine learning. There are two main approaches:

- with hand-crafted image features processed by a classification model;
- with deep-learning, with a neural network that directly processes the image pixels. Convolutional neural networks (CNN) are the most suitable architecture for this application.

1 Lab activity

In this exercise we will build classifiers for the classification of images of flowers. We will use a subset of the "Oxford Flower Datasets" collected by Maria-Elena Nilsback and Andrew Zisserman. The original data set, which is much larger, is available at the address `https://www.robots.ox.ac.uk/~vgg/data/flowers/`.

Before starting the exercise be sure to read the "Preliminaries" section here below, download the data set and review the relevant topics.

1.0 Preliminaries

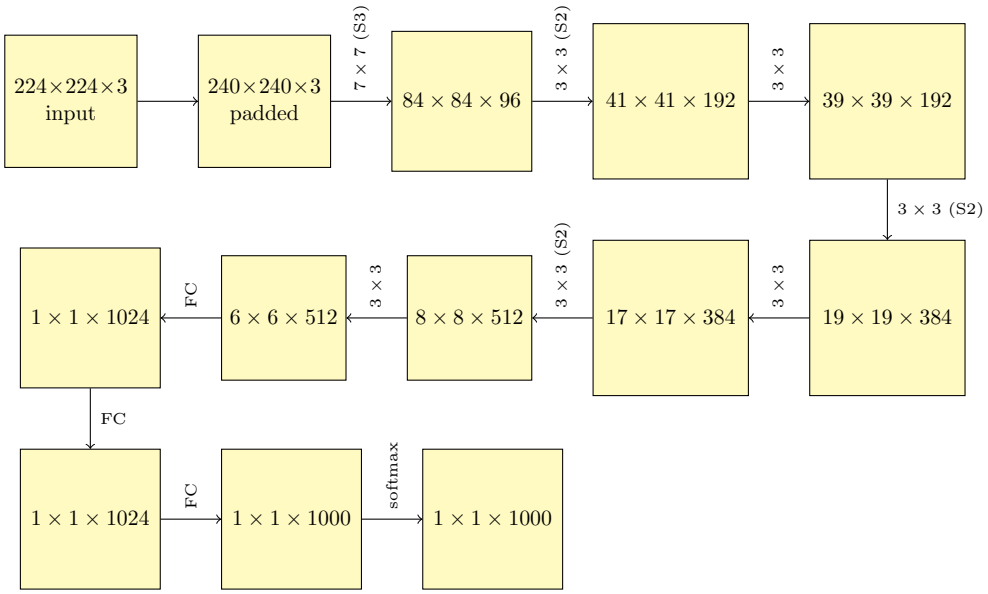
The data set contains 80 images for each of 12 species of flowers. For each class 60 images are in the training set and 20 form the test set. All the images have been resized to 224×224 pixels.

For feature extraction we will use low-level features. Some of them have been implemented in the `image_features.py` file.

We will also use neural features, computed py using the 'PVMLNet' CNN. The definition of the network is in the `pvml` library, while the pretrained weights are downloadable from the address `https://drive.google.com/open?id=1VmpvQkBk_dLsU54muiwsGcRRT_cSiYwa`.

1.0.1 PVMLNet

This CNN has been designed as a slight simplification over the AlexNet architecture. It takes as input color images of 224×224 pixels. Images that are mean-var normalized and padded with 16 zeros in each direction. After that a sequence of 7 valid convolutions followed by three fully-connected layers (also implemented as convolutions) computes the class scores as follows:



All layers are followed by ReLU activations. Some convolutions (S) are strided. A final softmax computes the class probabilities.

The network has been trained on the ILSVRC-12 subset of ImageNet. It achieves a top-1 accuracy of 56.5% and a top-5 accuracy of 80.3%.

1.1 Low-level features

Write a script that computes one of the low-level feature vector implemented in the `image_features.py` file. Train a classifier and evaluate the test accuracy.

1.2 Neural features

Use the pretrained PVMLNet to extract as features the activations of the last hidden layer. Train a perceptron without hidden layers and evaluate the test accuracy.

1.3 Transfer learning

Build a new network by replacing the last layer of PVMLNet with the weights of the trained perceptron.

2 Assignment

As homework, review and refine the scripts programmed in the lab activity. In addition, perform the following exercises.

2.1 Combining features

Try different combinations of low-level features (concatenate two or more feature vectors with `np.concatenate`).

2.2 Data augmentation

The '`train_augmented`' directory contains the training set with data augmentation, with 18 variations for each training image. Retrain the best models on the augmented training set and measure how much it improves the test accuracy.

2.3 Analysis

Identify the pairs of classes that are more likely to be confused. Also identify the hardest test images to be correctly classified.

2.4 CNN (Optional)

Try to fine-tune the CNN after transfer learning (i.e. continue training with as very small learning rate like 10^{-5}).

Try transfer learning by chopping the network at a different layer.

2.5 Report

Prepare a report of one or two pages with the answers to document all the experiments and their results. The report must be in the PDF format. Include your name in the report and conclude the document with the following statement: "I affirm that this report is the result of my own work and that I did not share any part of it with anyone else except the teacher."

Make a ZIP archive with the report and the python scripts you used and send it by e-mail before the next Friday.