SGN-41007 Pattern Recognition and Machine Learning

Exercise Set 5: November 25–November 29, 2019

Exercises consist of both pen&paper and computer assignments. Pen&paper questions are solved at home before exercises, while computer assignments are solved during exercise hours. The computer assignments are marked by `python` and Pen&paper questions by `pen&paper`

1. `pen&paper` *Count the number of parameters in a **dense** neural network.*

   Consider the traditional shallow neural network architecture of Figure 1. Suppose our inputs are 224×224 RGB bitmaps of 10 categories.

   Let the network structure be the following:

   - The input is $224 \times 224 \times 3 = 150528$-dimensional
   - On the 1st layer there are 100 nodes (marked in blue)
   - On the 2nd layer there are 100 nodes (marked in blue)
   - On the 3rd (output) layer there are 10 nodes (marked in blue; one for each class)

   Compute the number of parameters (coefficients) in the net.

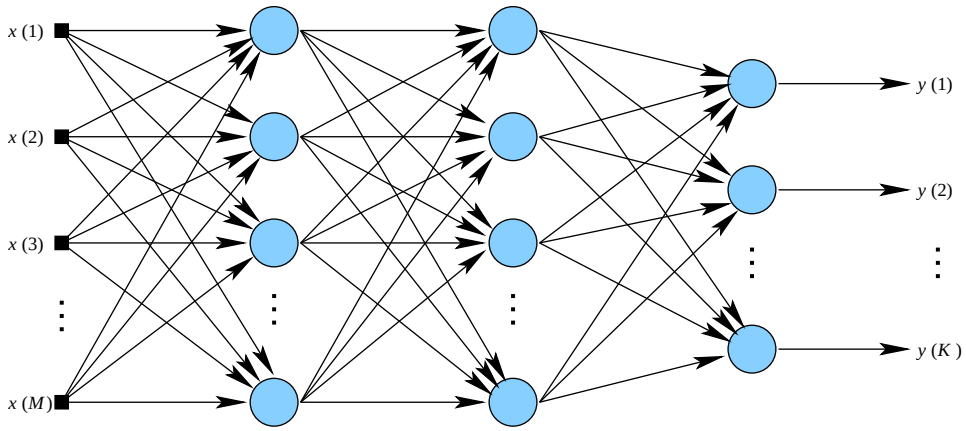

Figure 1: Vanilla neural network.

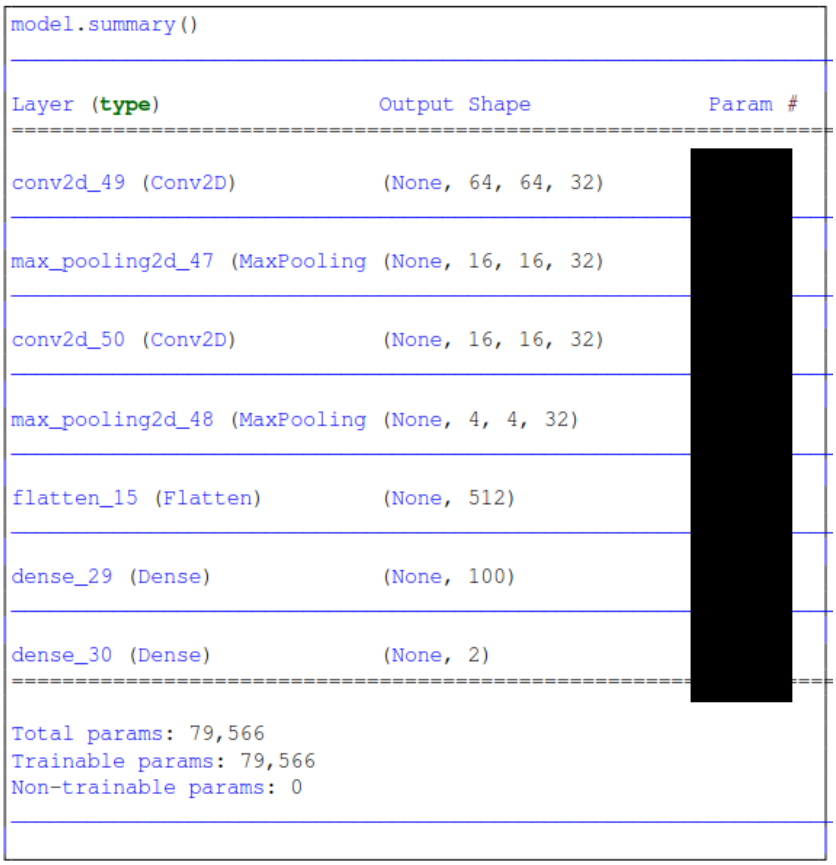

Figure 2: A CNN model summary as reported by Keras

2. **pen&paper** *Count the number of parameters in a **convolutional** neural network.*

Consider the Keras model defined in Listing 2. Inputs are 64×64 color images from two categories, and all convolutional kernels are of size $w \times h = 5 \times 5$.

a) Draw a diagram of the network.

b) Compute the number of parameters for each layer.

c) How many scalar multiplications take place on the first convolutional layer?

3. **python** *MNIST with custom network.*

Let us train two models for the MNIST dataset. Download a template code from

`http://www.cs.tut.fi/courses/SGN-41007/mnist.py`

Execute the code. The epochs should take roughly 5-10 seconds if running on the GPU. If it seems slow, try running the code directly from command line instead of Spyder. Thus; assuming the code is at `C:\Temp\`, you do the following:

```
>> c:
>> cd \Temp
>> python mnist.py
```

4. **python** *MNIST with pretrained network.*

Instead of using the custom ConvNet of question 4, initialize a mobilenet and attach 2 dense layers at the end. The mobilenet should have parameters `include_top = False`, `alpha = 0.25`, and `input_shape = (128,128,3)`. Check `model.summary()`.

Add dense layers after the convolutional pipeline such that `model.summary()` reports the following (top of listing omitted):

```
conv_pw_13_bn (BatchNormaliz (None, 4, 4, 256)         1024
_________________________________________________________________
conv_pw_13_relu (ReLU)       (None, 4, 4, 256)         0
_________________________________________________________________
flatten (Flatten)            (None, 4096)              0
_________________________________________________________________
dense (Dense)                (None, 100)               409700
_________________________________________________________________
dense_1 (Dense)              (None, 10)                1010
=================================================================

Total params: 629,254
Trainable params: 623,782
Non-trainable params: 5,472
```

5. **python** *Train the model.*

After designing the model, compile and train it. You will also need to resize your inputs to match the input size of the net (128 x 128). What accuracy do you get?