Practical work 07 – 09/04/2023
Convolutional Neural Networks with Keras

Objectives

The first objective of this PW is to understand the principles of Convolutional Neural Networks (CNN). Another objective is to experiments with CNN using TensorFlow.
Hint : In this exercise you will exclusively use Keras API of TensorFlow because the second part of the course will be based on this framework.

Submission

— **Deadline** : Tuesday 16th April, 15pm

— **Format** :

  — Jupyter notebook or pdf with convolution results.

  — Completed Jupyter notebook with the blanks filled in (the sections to be completed marked as usual, `### START YOUR CODE ###, ### END YOUR CODE ###`) and the table at the end completed.

Submission of all files in a single zip-file using the naming convention (for team of two students #1, #2) :

`family name_given name #1- family name_given name #2.zip`

Exercise 1 Computation of convolutions

Revise the section on convolutions in the lecture notes and make sure you understand the principles and the different steps that were described.

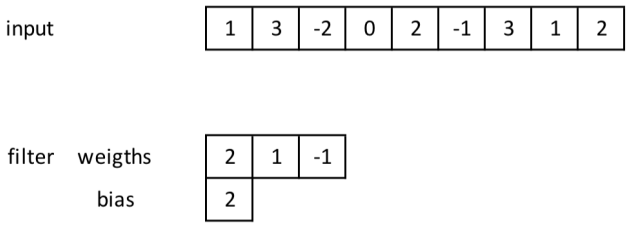

FIGURE 1 – 1D convolution

a) Given the 1D example in Figure 1 compute the output of the convolution by hand considering the following stride and padding values :
   - $S = 1$, $P = 0$
   - $S = 2$, $P = 0$
   - $S = 4$, $P = 0$
   - $S = 1$, $P = 1$
   - $S = 4$, $P = 1$

   In the previous examples, for which values of S and P do we get an output of same dimension as the input ?

b) Given the 2D example in Figure 2 :
   - How many activation maps will we obtain ?
   - With $S = 1$ and $P = 0$, what will be the shape of the output ?
   - With $S = 2$ and $P = 0$, what will be the shape of the output ?
   - Give a filter size, padding value and stride value that will preserve the shape of the input.
   - Compute the values of the output with $S = 1$ and $P = 0$ using an appropriate iPython notebook.

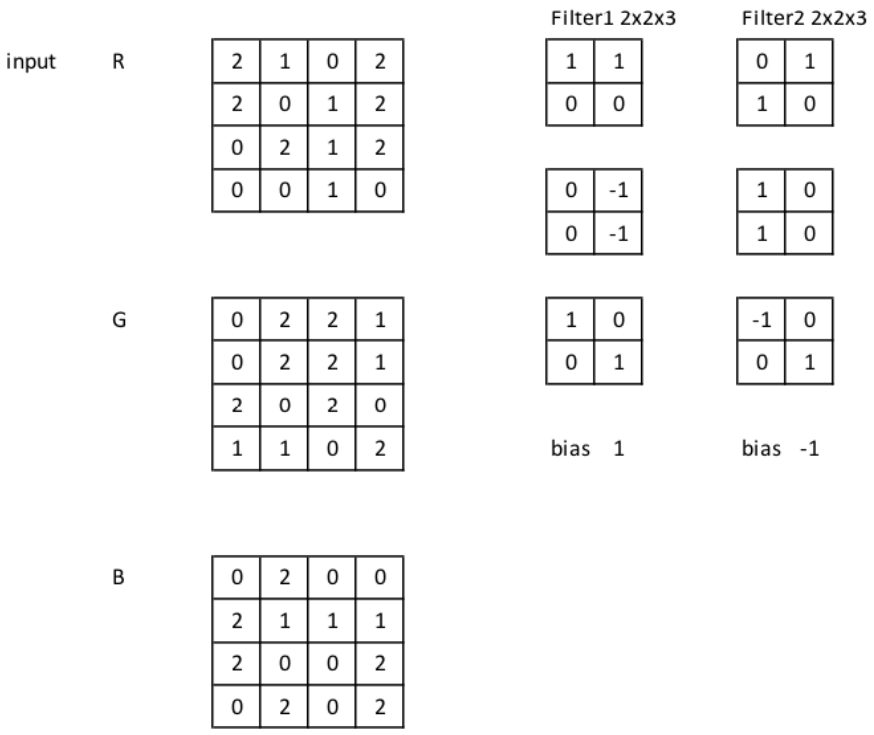

FIGURE 2 – 2D convolution

Exercise 2 CNN with Keras on CIFAR10

Use the Jupyter notebook `CIFAR10-CNN_from_raw_data_stud.ipynb` provided and complete the required sections. It is entirely based on the Keras API of TensorFlow (including image set download) in order to become more familiar with this framework. The main part consists of the definition of the different CNN architectures in the cell `Define the network`. You are supposed to implement and test different model complexities.
The following four architectures are possible inspirations but fell free to explore other option. For the convolutional kernel size 3 x 3 and stride 1 is typical but you may also choose other values. The same applies to the pooling layers, where a size of 2 and a step size of 2 occur most frequently :

— cnn1 :
- `Conv2D()` with 16 filters
- `relu()` activation
- `MaxPooling2D()`
- `Conv2D()` with 32 filters
- `relu()` activation
- `MaxPooling2D()`
- `Flatten2D()`
- `Dense(200)` layer with `relu` activation

- `Dense()` layer with `softmax` activation

— cnn2 :
- `Conv2D()` with 16 filters
- `relu()` activation
- `MaxPooling2D()`
- `Conv2D()` with 32 filters
- `relu()` activation
- `MaxPooling2D()`
- `Flatten2D()`
- `Dropout(0.3)`
- `Dense(300)` layer with `relu` activation
- `Dropout(0.3)`
- `Dense()` layer with `softmax` activation

— cnn3 :
- `Conv2D()` with 32 filters
- `relu()` activation
- `Conv2D()` with 32 filters
- `relu()` activation
- `MaxPooling2D()`
- `Conv2D()` with 64 filters
- `relu()` activation
- `Conv2D()` with 64 filters
- `relu()` activation
- `MaxPooling2D()`
- `Flatten2D()`
- `Dropout(0.3)`
- `Dense(300)` layer with `relu` activation
- `Dropout(0.3)`
- `Dense()` layer with `softmax` activation

— cnn4 :
- `Conv2D()` with 32 filters
- `relu()` activation
- `Conv2D()` with 32 filters
- `relu()` activation
- `MaxPooling2D()`
- `Conv2D()` with 64 filters
- `relu()` activation
- `Conv2D()` with 64 filters
- `relu()` activation
- `MaxPooling2D()`
- `Conv2D()` with 128 filters

- `relu()` activation
- `Conv2D()` with 128 filters
- `relu()` activation
- `MaxPooling2D()`
- `Flatten2D()`
- `Dropout(0.3)`
- `Dense(300)` layer with `relu` activation
- `Dropout(0.3)`
- `Dense()` layer with `softmax` activation

Report on your experiments and describe your best configuration through experimenting with at least four different architectures. With the more complex architecture (ccn3/4) the test accuracy should approach 80%. Use a table similar to the example given below and provide the hyper-parameters used for your configurations.

— epochs = ...

— batch size = ..

— all activations = 'relu'

— ...

CNN	Architecture	Acc. validation %	Acc. test %
1	Conv2D(16, 3) ; MaxPool2D(2) ; Conv2D(32, 3) ; MaxPool2D(2) ; Dense(200) ; Dense(10, 'softmax')	69.1%	68.6%
2	...	...	...
3	...	...	...
4	...	...	...
5	...	...	...

TABLE 1 – Performances on CIFAR10 with dierent CNN congurations.