LCPB 23-24 exercise 1 (Convolutional Neural Network, CNN)

One member of the group will submit the Jupyter notebook compiled, with all figures, by using the interface in moodle.

The name of the notebook should be

group23XX_exerciseY.ipynb

where XX is the code of the group and Y is the number of the exercise.

The notebook should start with the list of students (Name, Family Name, matriculation number).

Analyze the data file provided at the lesson with CNNs. Each of the three classes contains a part of a signal with N values, in which there could be the addition of a deterministic pattern of length P<N. All patterns have the same length P (in the notebook), but their shape depends on the class, and their position varies from sample to sample.

1.
The lesson discussed two CNN versions: one ending with a global max pooling layer (G), and one ending with a dense layer (D).
Choose one of the two G and D versions for the remaining exercise, motivating your choice.

2.
Try different optimizers, including at least Adam, RMSprop, and Nesterov. Each one should be tested for at least five values of the learning rate η (for example, including 10^{-5}, 10^{-4}, 10^{-3}). The range could be different for different algorithms. Moreover, at least five fits per model should be collected to obtain statistics with an average and standard deviation of the model's validation accuracy. Plot all results in a single panel of a figure as a function of η.

3.
Choose one among the best CNNs in step 2 (there could be equivalent ones within error bars). Using that CNN, try to understand the hidden patterns. Introduce regularization and study the kernels of the first layer, also by varying their length and their number (respectively, KS and NF in the notebook). Report your findings, including eventually improved validation accuracy, thanks to the regularization. However, note that this point's scope differs from optimizing the validation accuracy: it focuses on interpreting the kernel shapes.

The regularization is tuned by varying the parameter λ (in magnitude: 0, 10^{-5}, 10^{-4}, 10^{-3}, etc.) of the L2 (Ridge) regularization (see lambda in eqs.(43) and (52) in the review.) or the L1 (LASSO) regularization. One can also try a mixed version (l1_l2, optional). Does the regularization provide any significant improvement in understanding the weights in the filters?
Note that our regularization acts on the w's, not on the biases. One can also try the equivalent procedure for biases or the output of the ReLU units (see Keras doc.) if there is any reason for suspecting that it may help. In our case, the logic was to let the weights of the filters go to zero if not needed; hence that kind of regularization was selected.

4. OPTIONAL
Try to improve the CNN's performance by a random search in the hyperparameters' space. The performance is quantified again by the accuracy of the validation dataset.

To implement a "random search," one can use the **keras_tuner** package:
https://keras.io/guides/keras_tuner/getting_started/
Hyperparameters that could be considered for tuning include:
• minibatch size
• activation units (sigmoid, ReLU, ELU, etc.)
• Minimizer (ADAM, RMSprop, Nesterov, etc.)
• dropout values
• number of layers
• rescaling of data
(above, we already varied the learning rate, KS, and NF; it is allowed to reconsider their variation).