Coursework 2 – ACSE-8

Lluís Guasch, Deborah Pelacani, Oscar Bates

start date: 11th May, 9:30h BST
end date: 16th May, 23:59h BST

Instructions:

Follow the instructions below to complete the coursework and submit it:

1. Read the materials you will need to complete the coursework; you can find them in the following [github classroom] The provided materials are:
   - This document which contains instructions on how to complete the coursework and the points per question (100 points total).
   - The paper *Visualizing the Loss Landscape of Neural Nets* [paper link].
   - A Jupyter Notebook template to fill in your answers (it contains some code to assist you).
2. Complete your coursework using the provided Jupyter Notebook template (use Google Colab or your local machine if it has a GPU and/or sufficient computational power).
3. Once you have completed your answers, upload your final notebook in github classroom. Make sure to have all the answers in there:
   - **All the cells in your final Jupyter Notebook should be executed before saving and uploading to github in order to have the output of the cells available in the uploaded version** (images you plot, training graphs generated with `livelossplot`, etc). Add comments in the code to explaining what you are doing.
   - All answers requiring written answers should be in markdown blocks in the Jupyter Notebook. The provided Jupyter Notebook template has allocated blocks for the questions, but you can add any coding or markdown blocks that you need.
4. As stated in the title of the document, the coursework will be released on Tuesday 11th May at 9:30h BST, and the answers have to be submitted on 16th May, 23:59h BST. We will not accept late submissions.

BST stands for British Summer Time (local UK time)

The coursework consists of a number of questions/exercises you have to complete. You will also find them in the Jupyter Notebook template. They are:

1-Prepare your LeNet-5 network [10 points]

Use the code provided in the Jupyter Notebook template and modify it as you see fit to be able to perform a forward pass using the single dummy tensor input `x` provided. The lines of code that will do the forward pass and print the network are provided in the template.

2-Load CIFAR-10 [10 points]

Use `torchvision.datasets.CIFAR10` to load the CIFAR-10 dataset (training and test sets).

3-Plot data [5 points]

Plot 25 images of the training set together with their corresponding label names.

4-Create a training, validation split [5 points]

Split the data using `sklearn.model_selection.StratifiedShuffleSplit`:

- 90% of the data in the training set.
- 10% of the data in the validation set.

Prepare the downloaded datasets to be used with your modified network in **1-Prepare your LeNet-5 network**.

5-Grid search [20 points]

From the list below, select two hyperparameters and perform a 2D grid-search to find the optimal values for these two hyperparameters. The range of values to test are provided. Justify your choice of the two hyperparameters you want to tune (write a paragraph in a markdown cell explaining why you chose these two particular parameters). The list of hyperparameters to choose from is:

a) Random Number Seed: **42** [31, 42, 53]

b) Learning Rate: **1e-2** [1e-1, 1e-2, 1e-3]

c) Momentum: **0.5** [0.2, 0.5, 0.8]

d) Batch Size: **64** [64, 128, 640]

e) Number of epochs: **30** [10, 30, 50]

The **values in bold** next to each hyperparameter are the values you need to use if you are not tuning this particular hyperparameter. The values between square brackets [a,b,c,...] are the values to use if you choose to tune this particular hyperparameter. Fixed hyperparameters:

- Optimiser: **SGD+momentum**
- Test batch size: **1000**

Write the results in two tables (one for the loss and one for the accuracy) where the columns and rows are the first and second hyperparameter have selected. You can use markdown tables or create the table in python.

You don't need to plot all the `livelossplot` plots for each combination you try, as the results will be summarised in the table.

Select the best values for the two hyperparameters you have chosen to optimise and **justify your choice**

6-Train with best hyperparameters [5 points]

Once you have your two best hyperparameters, retrain the model by combining the validation, training, and test sets as you see fit. Report the final accuracy on the test set. Use `livelossplot` to plot the values of the training evolution.

7-Answer the following questions [1 point each]

Which of these data-augmentation transforms would be reasonable to apply to CIFAR10 and why?

1. Left-Right Flips
2. Random Rotations by up to 10 Degrees
3. Up-Down Flips
4. Shifting up-down, left-right by 5 pixels
5. Contrast Changes
6. Adding Gaussian Noise
7. Random Rotations by up to 90 Degrees

Justify each one of your answers.

8-Plot augmented data [13 points]

Select one of the data-augmentation transforms you chose as reasonable in the previous question, implement it, and apply it to 9 images of the CIFAR-10 dataset. Plot the 9 transformed images.

9-Visualising loss landscapes paper [10 points]

Read the provided paper *Visualising the Loss Landscape of Neural Nets*. This paper is contains a lot of advanced concepts, but you only need to read and understand it well up to, and including section 4, (Proposed Visualisation: Filter-Wise Normalisation) to answer the questions below. In section 4 you don't need to fully understand the rationale for doing Filter-Wise Normalisation, but you do need to understand what Filter-Wise Normalisation is.

Answer the following questions (in a markdown cell):

1. What are δ, η, α and β in equation (1)? [5 points]
2. What does Filter-Wise Normalisation do? Don't need to explain the reasons for doing it, just how it modifies the random directions δ and η. [2.5 points]

Explain well and justify your answers. **(Don't answer in 1. that δ is a random direction just because it says that in 2. Explain what is meant by a random direction well.)**

10-Visualise loss landscape [15 points]

Use the formula described in equation (1) in the paper in combination with the Filter-Wise Normalisation to generate landscape plots. For that use your final trained model (output of question **7**) and 25 values for α and 25 values for β to generate a 2D plot with a 100 points. Use the provided snippets of code in the Jupyter Notebook template to assist you in generating the plots and to guide you in the functions you will need to implement (not mandatory, you can implement everything from scratch if you prefer).

Note that in this question you will not be comparing the smoothness of different loss landscapes (as they do in the paper), you will only be plotting the loss function landscape around the loss value corresponding to your trained network.

Discuss your results, and justify the choices you make along the generation process of plotting the loss landscapes (for example the range your choose for your α and β values).