Universidad de La Sabana Felix Mohr
Facultad de Ingenería 2025.1

Deep Learning

- Sheet 3-

Exercise 1 (TorchMonitor - 6 Points) Implement the `TorchMonitor` class that will maintain 3 dictionaries for layer weights, activation function net values, and activation function outputs, respectively. Keys are the respective layer names in the torch module, and values are lists with one entry per processed batch (except for weights, which should have one more, because of the initial weights).

Given a model, your monitor should create hooks to be informed about what happens during a forward pass (only training phase) for any `Linear` layer and `ReLU` layer.

Make sure that you then do the following:

1. extract the initial weights from all linear layers and initialize the respective list of each layer weight list with this matrix.
2. implement the method that retrieves the weights of all linear layers after the optimizer has updated them; add them to the respective lists initialized in the previous point.
3. for each ReLU activation function in the network, store the matrix of net values (first dimension is batch size, second is number of neurons) in the respective list whenever a forward pass is realized (use a forward hook for this). There must be one list for each ReLU unit, and the number matrices in it must be equal to the number of realized forward passes (in training mode).
4. for each ReLU activation function in the network, store the matrix of output values (first dimension is batch size, second is number of neurons) in the respective list whenever a forward pass is realized (use a forward hook for this). There must be one list for each ReLU unit, and the number matrices in it must be equal to the number of realized forward passes (in training mode).
5. implement the method to receive the loss on the train and validation data after each processed batch, respectively, and add them to the respective list.
6. implement the method to create a GIF movie that will contain one frame for every processed batch, and the image in the frame is defined by the input function `fig_fun`. This function must have 6 arguments (check dokstring) and return a Figure object from matplotlib. Make sure that you use the `imageio` library to create the GIF.

Exercise 2 (Training Curve vs Validation Curve) Write yourself a nice function for the visualization of what has happened up to a certain number of batches. The figure should include:

- training and validation loss in a single panel (different colors for both, use a legend)
- some summary (maybe boxplots) that shows, for each forward pass, the distribution of linear layer weights, per linear layer.
- some summary (maybe boxplots) that shows, for each forward pass, the net values of the neurons and the neuron outputs.

Exercise 3 (4 Points) Use you monitor to create GIF movies for 100 batches for these different setups on the CIFAR-10 dataset:

- initializations in He and Uniform
- with and without standardization of the input data
- using no regularization, once using L2 regularization with 3 different values of lambda, and once dropout with 3 different probabilities
- without batch norm, with batch norm before the activation, and once with batch norm after the activation

This makes: $2 \cdot 2 \cdot 7 \cdot 3 = 84$ movies in total; make sure to use reasonable filenames. In the zip, include a txt file where you mention two findings:

1. the configuration you find to have the best learning progress with an explanation of why that is the case
2. the observation that you found most interesting or surprising and why.