EBERHARD KARLS UNIVERSITÄT TÜBINGEN

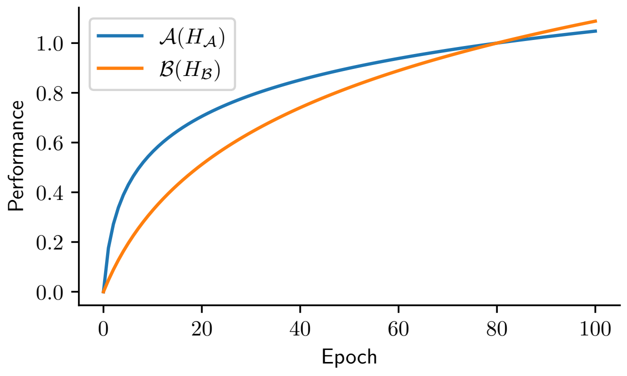
Numerics of Machine Learning

Exercise Sheet #11

Submission due on Jan 19th, 2023 at 2.15pm on Ilias

1. **EXAMple Question**
   Assume we have two update rules $\mathcal{A}$ and $\mathcal{B}$ which each have a set of hyperparameters $\mathcal{H}_{\mathcal{A}}$ and $\mathcal{H}_{\mathcal{B}}$. In the following, we consider the performance of both methods only on a single problem (e.g. a single neural network using a fixed dataset).

   Both methods are now tuned using random search with 20 tuning trials. The hyperparameters are tuned for $\mathcal{A}$ and $\mathcal{B}$ independently with the search spaces given by $\Phi(\mathcal{A})$ and $\Phi(\mathcal{B})$. The specific hyperparameter setting that reached the best-observed performance after 100 epochs are denoted by $H_{\mathcal{A}}$ and $H_{\mathcal{B}}$. The performance curves (higher means better) of both training algorithms $\mathcal{A}(H_{\mathcal{A}})$ and $\mathcal{B}(H_{\mathcal{B}})$ are shown in the figure below.



   What is wrong with the following statement:

   "On this problem, although $\mathcal{B}$ provides a slightly better final performance, $\mathcal{A}$ offers faster training, and should be preferred if trained for 80 epochs or less."

   Hint: Your task is to identify parts of the statement that cannot be confidently inferred from the provided data, rather than to identify conclusions that are definitely wrong. There may be more than one issue with the provided statement. Feel free to state multiple problematic aspects but it is sufficient to identify just one.

2. **Coding exercise**
   Assume your colleague is trying to train a neural network but it is clearly not getting the results they think are achievable (as measured by the accuracy on unseen data). They hand over their current code to you and you can find the code in the notebook file `Exercise_11.ipynb`. Your task is to debug the neural network training process in the provided notebook file, improve the network's performance as much as you can, and create a logbook detailing your process.

   Code: The entire training code is provided in `Exercise_11.ipynb`.You can play around with this file as you wish, modifying or extending it. You can assume that all code that is located in `utils.py` (compressed in the `utils.zip` file) is correct (but you are nevertheless allowed to change it if you want). The provided `yml` file (compressed in the `utils.zip` file) describes a `conda` environment with the necessary packages to run the notebook, to help you get started.

Task: Your task is to debug the training process, increase the model's performance, and accurately document your process similar to the OPT Logbook. Your logbook should contain a description of what you did, what you observed, and what you derived from this. You can format the logbook using the *observation*, *hypothesis*, and *results* structure. This could, for example, look like this:

- **Observation:** I observed that the loss continued to go down during the entire training time.
- **Hypothesis:** This could mean that the learning rate is too small (since the loss doesn't go down fast enough) or that the training time is too short.
- **Results:** Changing the learning rate from 0.01 to 0.1 was too much, but increasing it to 0.02 improved the performance from 70 % to 75 %. It appears that the loss plateaus now at the end of the training.

You can fill the logbook with screenshots from the notebook to visualize and explain your observations and results.

Be honest with your log! This exercise is not about getting it right and only documenting the successful approaches. We want to understand how you would train, debug, and tune a neural network, what observables and plots you look at, and what conclusions you draw from them.

Cockpit: The provided notebook (`Exercise_11.ipynb`) already contains the option to monitor the training process using COCKPIT. The provided plots and quantities can be a starting point for debugging and improving the training. COCKPIT's documentation provides a useful reference for what each instrument and quantity describes.

Submission: Submit a `PDF` file of your logbook. It should describe what you tried to improve the training process and what you observed as a result. Ideally, it would identify the (artificially introduced) "bugs" of the provided code and improve the network's accuracy. Optionally, you can also submit your notebook file with your final training process. It is not mandatory to improve the network's performance above a certain threshold or even to find any "bugs" in order to pass this exercise. Instead, to pass the exercise, you should have made an honest effort to debug and tune the network and document your process faithfully.

Hint: There are multiple "bugs" purposely introduced to the training process (and perhaps even a few more we introduced accidentally) that make proper training almost impossible. Try to find these bugs first, before you focus on tuning the network's performance.

Training time: A single training run can take a significant amount of time depending on your hardware setup (for me, it took roughly 10 minutes using a GPU and 25 minutes on a CPU). You have to play around with the number of training epochs, the tracking frequency of COCKPIT, or the amount of model evaluation to reduce the training time and allow faster debugging and tuning as needed.