Tutorial workflow trainers.py

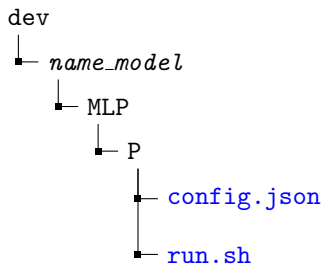
Luka Vranckx

June 28, 2025

This is a tutorial on how to use the closure repository by G. Miloshevich. More specifically here is explained how to train a neural network using trainers.py on Particle-in-cell (PIC) data.

0 Example directory

To run trainers.py, multiple files need to be prepared, as explained in the following sections. An example directory structure for one training routine is shown below. This is the initial structure you should always prepare. The `dev` folder in this example should be in the same folder as the `src` folder containing the source code of trainers.py. In black are folders, in blue are files. The folder structure is arbitrary and can be adjusted as you'd like.



1 Splitting PIC data in train/val/test set

Choose multiple PIC frames or multiple PIC data sets and split them in a training, validation and test set. You can choose to use all time frames of the simulation or only the ones where turbulence is fully developed.

Next, make three `csv` files, named `train.csv`, `val.csv`, `test.csv` as follows. The first line should be 'filenames', the line below is the location of the pickle-file of the first frame to include, e.g. 'T2D12_filter2/T2D-Fields_007500.h5.pkl', which is a relative path and should be consistent with the `data_folder` keyword specified in the `config.json` file. The lines below, you add all the other pickle-files, possibly from different PIC runs.

! Beware: the very first PIC frame of each run doesn't have off-diagonal pressure components, so should be excluded.

2 Setting up the configuration file

Here is explained how to correctly adjust the `config.json` file. It's a nested Python dictionary using keys and values that configures the trainer. Use a pre-existing config file, where you can change the values for keywords as needed. Generally, the things you have to check to correctly run the trainer are listed here:

- work_dir: absolute path to the location of the config.json file
- dataset_kwargs
  - data_folder: first part of absolute path to the PIC data (pickle files). This path is concatenated with the ones inside the `csv` files as explained in section 1.
  - train_sample: absolute path to `train.csv`
  - val_sample: idem for `val.csv`
  - test_sample: idem for `test.csv`
- Remove the 'run' key and its value if it is present (always last key).

Besides, you can adjust other values such as the architecture of the neural network and its hyperparameters.

3 Running trainers.py using a batch script

In order to run trainers.py, it's good practice to use a batch script, which we call `run.sh`. Also here, use a pre-existing file to easily change it to your needs.

- Change SBATCH lines to change job name and resources to use as well as the .out and .err file locations
- Change `REPO_DIR` to the direction of the `dev` folder, where also the `src` folder with the source code is located.
- Change the name of Python environment to activate after 'conda activate'.
- The line starting with 'srun' is where the trainer is executed. Here, you can force to change values in the config.json file. Importantly, you should specify the name of the run by adding e.g. `--config run='run0'`.

When the training is finished, new files have been added to the `P` folder (in the example above). Next, a new folder `run0` will be created in `P` where a copy of the config file will be stored, a log file called `run.log` and the model (`model.pth`) with the optimal weigh parameters for the neural network.

! TIP: You can train multiple models at ones with slightly different configurations using one batch file. To do so, copy the 'srun' line a few times and add syntax to change the config file. E.g. to change the number of epochs to 2000, add: `--config model_kwargs.scheduler_kwargs.epochs=2000` at the end of the line.