Weights & Biases Tutorial Part 1: Basics

20% of features covering most cases

What is wandb?

- Performance tracker for Machine Learning experiments
- Cloud-based visual dashboard for your run data
- Replacement for tensorboard, printouts etc.
- Enables easy result sharing, filtering and collaboration
- Helps produce reproducible results (logged commits, data, …)

wandb - init

Docs can be found here.

- Starts a new run for this project and logs it using wandb
- Spawns a background service that (per default) logs your data to the cloud
- Syncs hyperparameters, code etc. of the experiment
- Must be done before any wandb functionality is used

wandb - init (Arguments)

- **entity**: Team or user where the project is logged. Defaults to "None", which is your account

- **project**: Project name. If it's the first run, the project will be created automatically

- **name**: Name of this *individual run*

- **config**: Python dictionary with hyperparameters of your run

- **mode**: "*online*", "*offline*" or "*disabled*". Can also be set via CLI

- **save_code**: Saves your training script to the cloud

wandb - init (Example)

```
run = wandb.init(
    project="Active Third Person IL Hopper",
    name=f"{timestamp}_{cfg.prefix}_{cfg.policy_strategy}_corr-{cfg.learn_corr}",
    config=vars(cfg),
    mode="online" if cfg.track_wandb else "disabled",
)
```

wandb - log

Docs can be found here.

- Adds values to the run's log
- Takes a dictionary as input where **key**=metric_name, **value**=metric_value
- With "*panel/metric_name*" you can log to distinct panels (see example later)
- Per default: Increments the step counter

wandb - log (Arguments)

- **data**: Your data in dictionary format

- **commit**: Allows summing up metrics and only logging at specific steps

- **step**: Global step (usually you don’t have to set this)

wandb - log (Example)

```
wandb.log(
    {
        "Results/Iteration": iter_step + 1,
        "Results/Reward": eval_result,
        "Results/Smoothed_Reward": moving_average(results, n=5),
        "Results/Data_Generating_Perspective": data_disc_id,
    }
)
```

Command Line Interface

Docs can be found here.

- **wandb login**: Connect your local wandb instance with your account
- **wandb enabled** / **wandb disabled**: Completely disables wandb from this project. Nothing will be logged
- **wandb online** / **wandb offline**: Determines whether data will be synchronized during the experiment or just logged offline
- **wandb sync**: Syncs data in the wandb folder (depending on the flag)

Debugging wandb

- Initializing wandb adds overhead to your code
- Don’t want to log incomplete debug runs
- **Solution 1**: Use **wandb disabled** to temporarily disable wandb (*Can be annoying if you forget to turn it on again*)
- **Solution 2**: Use a config flag and **mode**=”disabled” in **wandb init** (*What I usually do*)

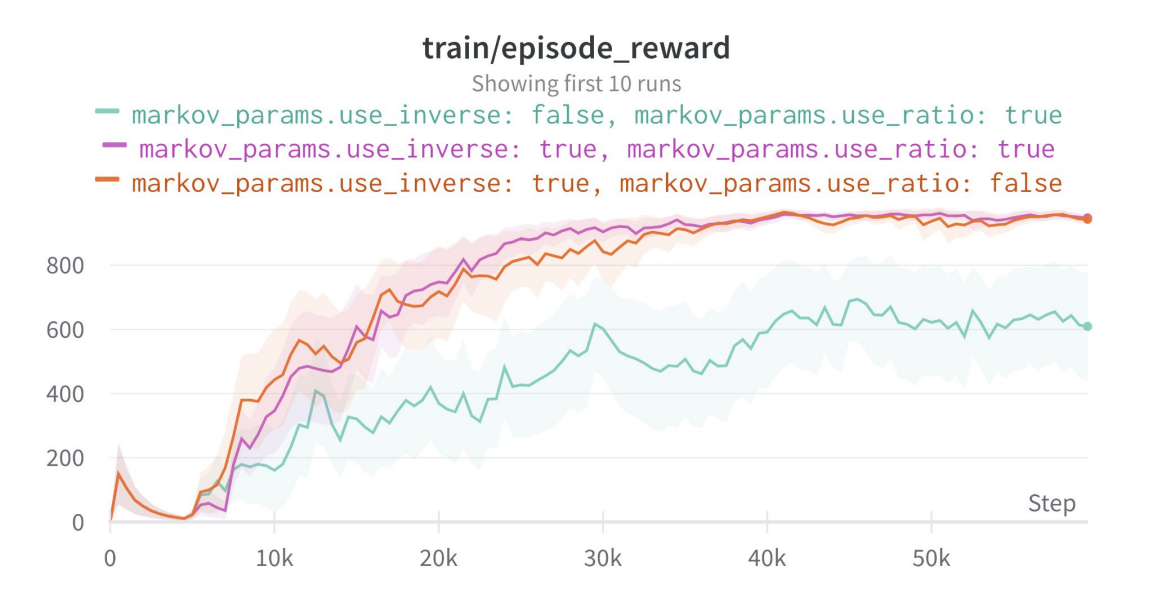
Dashboard basics

- wandb logs each individual run with parameters and config in a Table
- All values in the table can be used to sort runs
- Some logged values (e.g. Histogram) only visible for individual run
- Overview has git project name, current commit hash, conda env…

Dashboard aggregation

- **Filter**: Only show runs with specific settings (bool selection in pandas)
- **Group**: Aggregate runs based on specific values (groupby in pandas)
- Default aggregation: Mean with min/max shading
- Often more useful to use standard deviation/standard error as bands

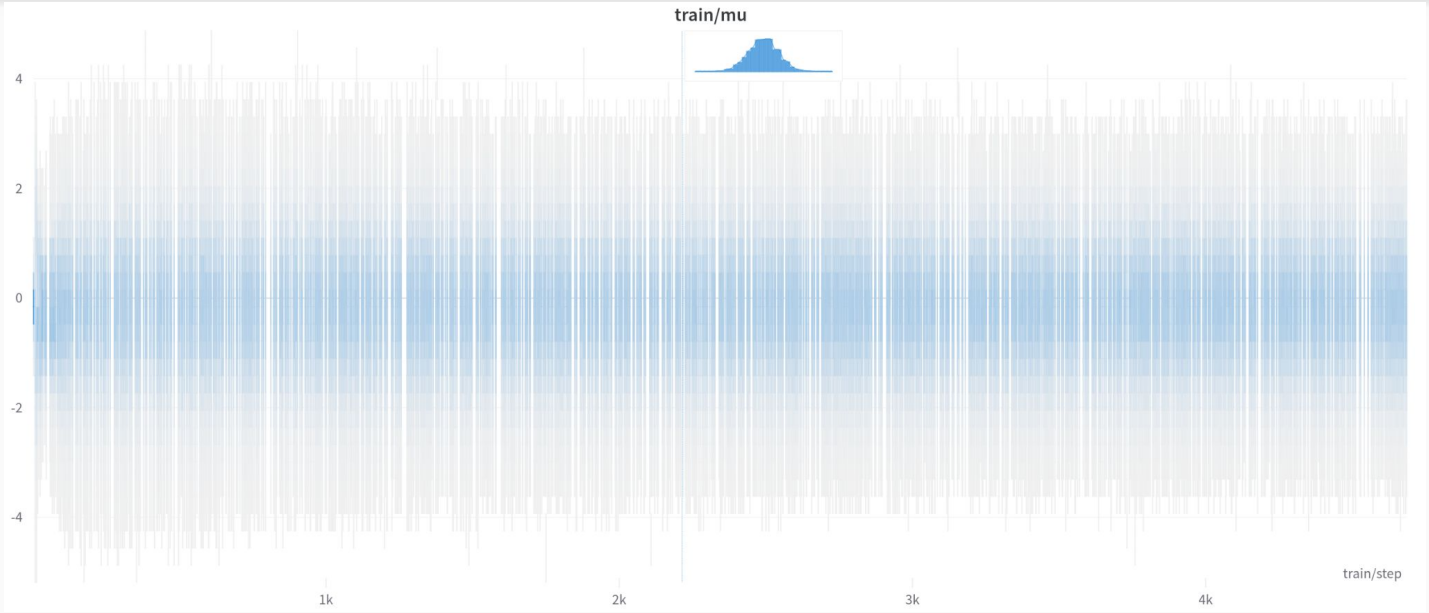
Aggregation example



Logging histograms

- Can use **wandb.Histogram** as drop-in replacement for np.histogram
- Not visible in the project's dashboard!
- Visualized as line plot over time
- Custom histograms are also possible, see here
  (*I never use this, too much hassle for non-publication plots*)

Histogram example



Logging images

- **wandb.Image** can be used directly with the RGB array
- Works similarly to plt.imshow
- Can even log segmentation masks
- *Warning: Logging lots of images or custom plots slows down the interface*

Logging custom plots

More info in the docs here.

- Matplotlib and Plotly figures can be logged directly to wandb
- Allows you to log very custom figures that are hard to do with wandb
- I rarely use this feature -> High quality plots are for the paper :)

Saving models

- wandb can save models to the cloud via **wandb.save**
- Argument is a *path to a file as string*. No pathlib.Path…
- Globbing is possible, e.g. with “*models/*.pt*”
- The **policy** flag determines when the file is saved
  “*now*” (Save immediately), “*live*” (sync and overwrite), “*end*” (sync at end)

“Advanced” topics: Outlook

- Logging metrics with different time steps
- Scraping your data from the cloud
- Multiprocessing & wandb
- Hyperparameter tuning
- Creating reports (not advanced, just no time)

Slides & Code

- https://github.com/timoklein/wandb_tutorial
- https://wandb.ai/timo_kk/Wandb%20Tutorial