MACHINE LEARNING

The Machine Learning Lessons I’ve Learned This Month

October 2025: READMEs, MIGs, and movements

Pascal Janetzky

Oct 27, 2025 6 min read

Photo by Anthony Tori via Unsplash

Most days in machine learning are the same.

Coding, waiting for results, interpreting them, returning back to coding. Plus, some intermediate presentations of one’s progress. But, things mostly being the same does not mean that there’s nothing to learn. Quite the contrary! Two to three years ago, I started a daily habit of writing down lessons that I learned from my ML work. In looking back through some of the lessons from this month, I found three practical lessons that stand out:

1. Using README documents for yourself
2. Requesting MIG slices instead of full GPUs
3. Sprinkling movements throughout the day

Keep a README — for your future self

Most READMEs are written with other people in mind. They're there to onboard new collaborators, or to make an open-source repo usable for strangers.

Write one for yourself, too — especially for your **future** self.

When you're in the middle of a project, all the paths, commands, and subtle settings live in your head. After a pause, not so much. I ran into this recently when I had to prepare an update for a paper. In ML research, reviews often take months. In that time, you move on to the next project with new datasets, new code, new conventions. When the review finally lands, you return to the old project and… *spend half a day reconstructing which script produced which figure.*

That’s a lot of stress if the deadline is approaching soon.

Preparing for your own forgetting is part of the job. A small README saves a big headache.

What to capture (for you)

Keep it practical. Your future self doesn’t want prose; they want the “how”:

- **Project quickstart.** Environment setup, exact Python version, env file or `conda` / `pip` commands.
- **Data locations.** Where raw and processed data live; how to download, cache, and checksum. Note any **gotchas** (e.g., “NDVI tiles are flipped south–north after resampling”, as given in my older posts).
- **Reproduce results.** One command per artifact: figures, tables, and checkpoints.
- **Training & eval.** The exact commands to run the main experiments; how to resume; how to set seeds.
- **Hyperparameter search.** The command you actually used, with ranges; where results are logged.

- **Common pitfalls.** Anything you know you'll forget (required env vars, GPU flags, file naming conventions).
- **Changelog.** One-line bullets of meaningful changes.

A minimal, generic template you can adapt for all projects:

```
# <Project Title>

## Quickstart
# env
conda create -n proj python=3.10 -y
conda activate proj
pip install -r requirements.txt

## Data
# download & preprocess
python tools/download_data.py --out data/raw
python tools/preprocess.py --in data/raw --out data/processed

## Train
python train.py --cfg cfgs/base.yaml --seed 42

## Evaluate
python eval.py --ckpt runs/exp123/best.ckpt --split test

## Reproduce Figures
python scripts/fig_1.py  # outputs to figs/fig_1.png
python scripts/tab_2.py  # writes tables/tab_2.csv
```

```
## Hyperparameter Search
python sweep.py --study local.pkl --n-trials 100

## Notes / Pitfalls
- Requires CUDA 12.1
- Set `WANDB_MODE=offline` if no internet
```

MIG slices for quick scheduling

Training the current generation of large language models needs hundreds (or thousands) of high-end GPUs. But most day-to-day ML work doesn't require LLM-scale models. A lot of problems are solved with compact CNNs or small MLPs — and those do not need a full A100/H100 GPU.

Requesting a whole GPU for a tiny model wastes resources and gets you to the back of the queue. I relearned this the hard way this month: I was training a *4-layer MLP* and waiting ages to be scheduled. In my scheduling requests, I requested a full high end GPU. Of course, these are in high demand by those jobs that actually need them (such as fine-tuning LLMs).

Once I switched to a MIG slice, jobs started immediately and my iteration speed jumped.

What’s MIG and why use it?

MIG (Multi-Instance GPU) lets you split a recent NVIDIA GPU into several isolated “slices.” One big GPU becomes up to seven smaller virtual GPUs, and the full VRAM is split across these slices. In essence, each slice is a smaller GPU. And for many workloads, those small slices are more than enough.

There is an additional benefit: **fewer people request slices** (because they are unaware of the opportunity), so schedulers can fit your job in right away. This allows you to iterate over your models faster, reducing the time to get good results.

To apply it in practice

- **Check availability.** Ask your cluster admins or look at the scheduler docs for MIG partition names (e.g., `1g.10gb`, `2g.20gb`).
- **Right-size your request.** Start small. If VRAM OOMs, step up one size. Don’t default to full GPUs.
- **Profile memory early.** Run a small batch to read peak VRAM; pick the smallest slice that leaves ~10–20% headroom.

- **Template your job.** Keep a job script for MIG and one for full GPUs; switch by flag.

Combatting prolonged sitting with movement

Most computer jobs are done in front of a, well, computer. We all have noticed that the more time we spend, the more slouchy our posture becomes. We’re rounded forward; all work is done in front of us*.

That’s not a good position to hold for a long time, but it’s quite common these days. For ML people, computers are our toolset and we (need to) spend considerable time with this toolset.

Thankfully, we don’t need to spend time in bad posture when we are working on a screen (no, those two are not linked *per definitionem*).

I realized this (again, because I seem to fall back into old habits) this month: hours of reading, coding, and meetings pulled my shoulders forward and lock the upper back. Then, after a few days of heavy paper reading, my shoulders reminded me to change something.

The fix, it turned out, was easy and doesn't require a gym or a full workout. It just requires **alternating positions** and **brief movement snacks**.

I collected together a small program, of which I used parts throughout the day (check YouTube for any of those exercises if you don't know them. That's meta-knowledge that might help you do your job better and longer — and healthier):

- **Audio-only meetings:** stand up and walk around. If you must stay near the desk, shift to a **split stance** (one foot forward) to open the hips.
- **Two-minute resets** (e.g., after a coffee break or coming from the printer, etc)**:**
  - Band pull-aparts or face pulls (10–15 reps)
  - Wall pec stretch (30–45s each side)
  - Hip flexor stretch / couch stretch (30–45s each side)
- **Standing reading blocks:** print a paper or read on a tablet while standing; alternate standing and sitting blocks.

Beyond, I've found that a short morning session makes my shoulders feel better and improves focus throughout the day — a welcome bonus, since ML work demands concentration:

- 5 min easy cardio (walk or climb stairs)
- 5 min mobility (thoracic rotations, shoulder circles, deep squat hold)
- 5 min light strength (lunges, push-ups against desk, band rows)

• • •

* An often-raised argument: aren't all jobs done in front of us? Or, put differently: which job requires us to work with hands behind us? Right, except for maybe gymnasts, virtually all jobs happen in front of us. After all, that's where our eyes are! But: the non-computer jobs get diversified movements along the day: grabbing something from a shelf, dragging something, etc. It's the alternation that counts.

• • •

WRITTEN BY