**Nine Tools I Wish I Mastered before My PhD in Machine Learning**

Whether you are building a start up or making scientific breakthroughs these tools will bring your ML pipeline to the next level



Image by Author.

Despite its monumental role in advancing technology, academia is often ignorant of industrial achievements. By the end of my PhD I realised that there is a myriad of great auxiliary tools, overlooked in academia, but widely adopted in industry.

From my personal experience I know that learning and integrating new tools can be boring, scary, could put back and demotivate, especially when the current set up is so familiar and works.

Dropping bad habits can be difficult. With every tool outlined below I had to accept that the way I did things was suboptimal. However, in the process I have also learnt that at times results not seen in the moment pay off ten fold at a later stage.

Below I talk about the tools that I have found very useful for researching and building machine learning applications, both as an academic and an AI engineer. I group the tools in four sections by their purpose: environment isolation, experiment tracking, collaboration and visualisation.

**Isolating Environments**

Machine learning is an extremly fast developing field and hence commonly used packages are updated very often. Despite developers efforts, newer versions are often not compatible with their predecessors. And that does cause a lot of pain!

Fortunately there are tools to solve this problem!

[**Docker**](https://www.docker.com/)

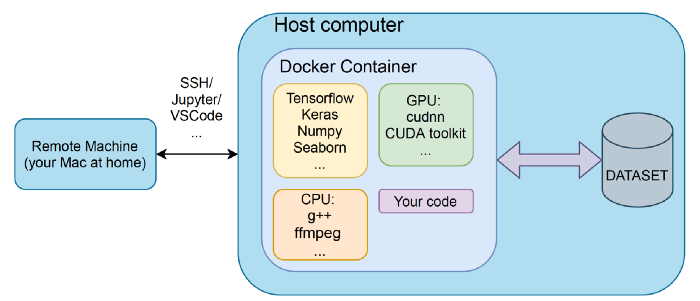


Image by Author.

How many times did those NVIDIA drivers caused you trouble? During my PhD I had a university managed machine that was regularly updated. Updated overnight and without any notice. Imagine my surprise when the morning after the update I find out that most of my work is now incompatible with the latest drivers.

Although not directly meant for that, docker saves you from these especially stressful before the deadline misfortunes.

Docker allows to wrap software in packages called containers. Containers are isolated units that have their own software, libraries and configuration files. In a simplified view a container is a separate, independent virtual operating system that has means to communicate with the outside world.

Docker has a plethora of ready made containers for you to use, without extensive knowledge of how to configure everything yourself it is very easy to get started with the basics.

For those wanting to have a quick start, check out this [tutorial](https://docs.docker.com/get-started/). Also Amazon AWS has done a great job explaining why and how to use docker for machine learning [here](https://aws.amazon.com/blogs/opensource/why-use-docker-containers-for-machine-learning-development/).

[**Conda**](https://docs.conda.io/en/latest/)

Reusing someones code became a new norm today. Someone creates a useful repository on github, you clone the code, install and get your solution without the need to write anything yourself.

There is a slight inconvenience though. When multiple projects are used together you run into package managing problem, where different projects require different versions of packages.

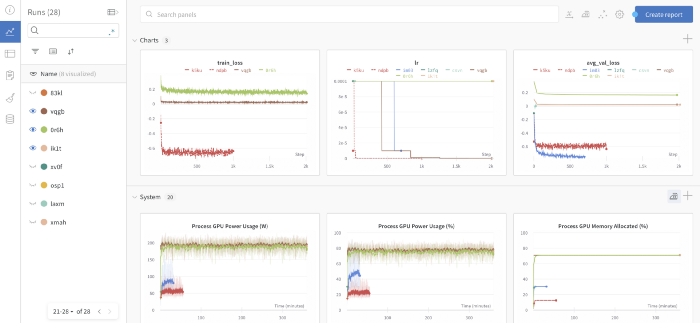
I am glad I discovered Conda not so late in my PhD. Conda is a package and environment management system. It allows to create multiple environments and quickly installs, run and update packages and their dependencies. You can quickly switch between isolated environments and always be sure that your project interacts only with the packages you expect.

Conda provides their own [tutorial](https://conda.io/projects/conda/en/latest/user-guide/getting-started.html#starting-conda) on how to create your first environment.

**Running, tracking and logging experiments**

Two essential pillars, without which getting a PhD in an applied field is close to impossible are rigour and consistency. And if you have ever tried to work with machine learning models you probably know how easy it is to loose track of the tested parameters. Back in the day parameter tracking was done in lab notebooks, I am certain these are still very useful in other fields, but in Computer Science we now have tools much more powerful than that.

[**Weights and biases**](https://wandb.ai/site)



Snapshot of the wandb panel for a set of simple metrics — train loss, learning rate and average validation loss. Notice that you can also track system parameters! Image by Author.

*experiment\_res\_1.csv  
experiment\_res\_1\_v2.csv  
experiment\_res\_learning\_rate\_pt\_5\_v1.csv  
….*

Do these names look familiar? If so, then your model tracking skills should be stepped up. This was me in the first year of my PhD. As an excuse, I should say that I had a spreadsheet where I would log the details of every experiment and all associated files. However, it is still very convoluted and every change in parameters logging would inevitably impact the post-processing scripts.

Weights and biases (W&B/wandb) is one of the gems that I found quite late, but now use in every project. It lets you track, compare, visualize and optimize machine learning experiments with just a few lines of code. It also lets you track your datasets. Despite a large number of options I found W&B easy to set up and use with a very friendly web interface.

For those interested check out their quick set up tutorial [here](https://docs.wandb.ai/quickstart).

[**MLflow**](https://mlflow.org/)

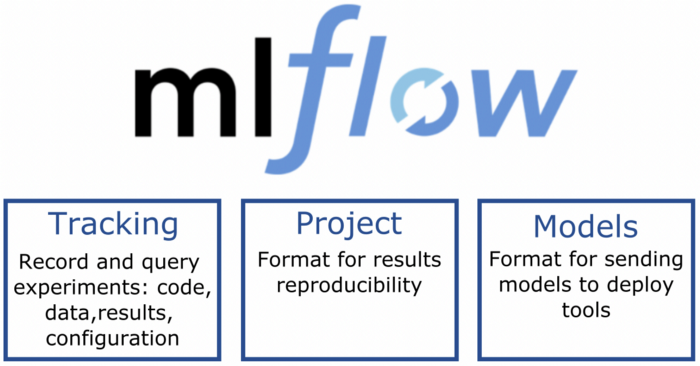


Image by Author.

Similar to W&B, MLFlow provides functionality for logging code, models and datasets on which your model has been trained. Although I have used it solely for the purpose of logging data, models and code, it provides functionality well beyond that. It allows to manage the whole ML lifecycle, including experimentation, reproducibility and deployment.

If you want to quickly integrate it into your models, check out this [tutorial](https://www.mlflow.org/docs/latest/tutorials-and-examples/tutorial.html). Databricks have also shared a very nice [explanation](https://databricks.com/blog/2018/06/05/introducing-mlflow-an-open-source-machine-learning-platform.html) of MLflow.

[**Screen**](https://linuxize.com/post/how-to-use-linux-screen/)

Leaving the experiments running overnight and hoping that your machine won't go to sleep was my go to option in the first half a year of my PhD. When the work moved to remote I used to worry about the ssh session breaking — the code was running for several hours and almost converged.

I learnt about *screen* function rather late, and so couldn’t save myself from half backed results in the mornings. But in this case it is indeed better late than never.

*Screen* lets you launch and use multiple shell sessions from a single ssh session. The process started with *screen* can be detached from session and then reattached at a later time. So your experiments can be run in the background, without the need to worry about session closing, or terminal crashing.

The functionality is summarised [here](https://www.geeksforgeeks.org/screen-command-in-linux-with-examples/).

**Collaboration**

Academia is notorious for not having proper mechanisms for effective team management. To an extent this is justified by very strict requirements for personal contribution. Nevertheless the pace at which machine learning is progressing needs joint effort. Below are two rather basic tools that would be handy for effective communication, especially in the new realm of remote work.

[**GitHub**](https://github.com/)

Pretty basic, huh? After seeing all the horror of how people track their code in accademia I cannot stress how important it is to be well versed in version control. No more folders named code\_v1, code\_v2.

Github provides a very useful framework for code tracking, merging and reviewing. Whenever a team is building a [deep image quality metric](https://towardsdatascience.com/deep-image-quality-assessment-30ad71641fac) each member could have its own branch of the code, working in parallel. Different parts of the solution can then be merged together. Whenever someone introduces a bug, it is dead easy to revert to the working version. Overall I rank git as the most important of all the tools I have mentioned in this article.

Check out this step by step [guide](https://guides.github.com/activities/hello-world/) on how to quickly start up.

[**Lucidchart**](https://www.lucidchart.com/pages/)

Lucidchart was introduced to me recently, before that I was using [draw.io](https://app.diagrams.net/)— a very simple interface for creating diagrams. Lucidchart is thousand times more powerful and has a much more versatile functionality. Its major strength is the shared space for collaboration and ability to make notes next to diagrams. Imagine a giant online whiteboard with a huge set of templates.

For a quick start check this [tutorial](https://www.lucidchart.com/pages/tour) page by Lucidchart.

**Visualisation**

Numerous paper submissions, especially unsuccessful ones, have taught me that presentation is often as important as the results. If the reviewer, who usually does not have much time, does not understand the text, the work is straightaway rejected. Images made in haste make a poor impression. Someone once told me: “If you cannot make a chart, how can I trust your results?”. I disagree with this statement, however, I do agree that the impression does matter.

[**Inkscape**](https://inkscape.org/)

A picture is worth a thousand words (in fact, [correction 84.1](https://www.cl.cam.ac.uk/~afb21/publications/Student-ESP.html) word).

Inkscape is a FREE software for vector graphics. In fact I was taught how to use it in my web-development course in my undergrad. However, I learnt how to enjoy it in full only during my PhD — working on those pretty pictures for the papers.

Of all the functionality that Inkscape provides especially valuable was [TexText](https://inkscape.org/~jcwinkler/%E2%98%85textext) extension. With this package you can integrate your [latex](https://www.latex-project.org/) formulas seamlesly into an image.

There is a myriad of tutorials, however for the basic functionality I would recommend the ones provided by Inkscape team [here](https://inkscape.org/learn/tutorials/).

[**Streamlit**](https://streamlit.io/)

Did you ever need to create a simple website to showcase you results or a simple machine learning application? In just few lines of python code it’s possible with Streamlit.

I found it particularly useful for paper supplementary materials, however it can be even more useful for easy deployment and showcasing project demos to clients.

For a quick start up check out this [tutorial](https://builtin.com/machine-learning/streamlit-tutorial).

**Summary and beyond**

Finishing my PhD while positioning myself in industry was not easy. But it taught me several important lessons I wish I had at an earlier stage of my PhD.

The most important lesson is that curiosity and readiness to learn and change can greatly impact the quality of your work.

Below is the summary of the tutorials I have mentioned in each section:

[Docker](https://www.docker.com/): [Tutorial](https://aws.amazon.com/blogs/opensource/why-use-docker-containers-for-machine-learning-development/)  
[Conda](https://docs.conda.io/en/latest/): [Tutorial](https://conda.io/projects/conda/en/latest/user-guide/getting-started.html#starting-conda)  
[Weights and biases](https://wandb.ai/site): [Tutorial](https://docs.wandb.ai/quickstart)  
[MLflow](https://mlflow.org/): [Tutorial](https://www.mlflow.org/docs/latest/tutorials-and-examples/tutorial.html)  
[GitHub](https://github.com/): [Tutorial](https://guides.github.com/activities/hello-world/)  
[Screen](https://linuxize.com/post/how-to-use-linux-screen/): [Tutorial](https://www.geeksforgeeks.org/screen-command-in-linux-with-examples/)  
[Inkscape](https://inkscape.org/): [Tutorial](https://inkscape.org/learn/tutorials/)  
[Streamlit](https://streamlit.io/): [Tutorial](https://builtin.com/machine-learning/streamlit-tutorial)  
[Lucidchart](https://www.lucidchart.com/pages/): [Tutorial](https://www.lucidchart.com/pages/tour)

If you liked this article share it with a friend! To read more on machine learning and image processing topics press subscribe!