The Deep Learning Research Stack

Giorgio Strano - strano@di.uniroma1.it

How to follow this section

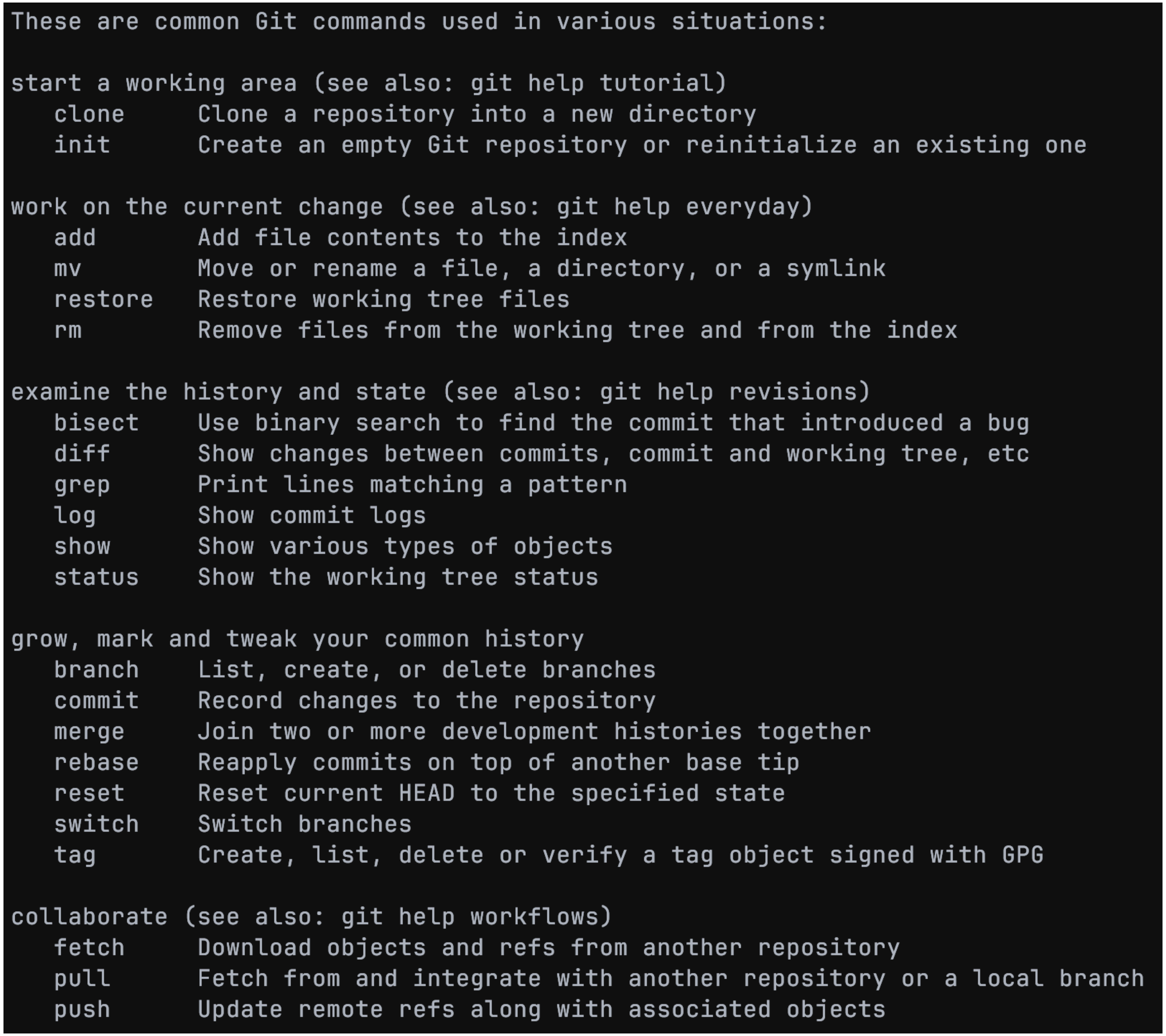
- Have a notebook ready
- If we are talking about something you know and use regularly:
    - **chill & relax**
- As soon as I mention something you don’t know:
    - **write it down**
- Have a list of new tools to use in your next project

Code version control: GIT

GIT *is not* **GitHub**

GIT *is a* **Command Line Interface** tool

Code version control: GIT



```
These are common Git commands used in various situations:

start a working area (see also: git help tutorial)
   clone      Clone a repository into a new directory
   init       Create an empty Git repository or reinitialize an existing one

work on the current change (see also: git help everyday)
   add        Add file contents to the index
   mv         Move or rename a file, a directory, or a symlink
   restore    Restore working tree files
   rm         Remove files from the working tree and from the index

examine the history and state (see also: git help revisions)
   bisect     Use binary search to find the commit that introduced a bug
   diff       Show changes between commits, commit and working tree, etc
   grep       Print lines matching a pattern
   log        Show commit logs
   show       Show various types of objects
   status     Show the working tree status

grow, mark and tweak your common history
   branch     List, create, or delete branches
   commit     Record changes to the repository
   merge      Join two or more development histories together
   rebase     Reapply commits on top of another base tip
   reset      Reset current HEAD to the specified state
   switch     Switch branches
   tag        Create, list, delete or verify a tag object signed with GPG

collaborate (see also: git help workflows)
   fetch      Download objects and refs from another repository
   pull       Fetch from and integrate with another repository or a local branch
   push       Update remote refs along with associated objects
```

Data version control: DVC

DVC tracks **large files**

DVC connects to **your own cloud storage**

PyTorch Lightning

A Deep Learning **framework** built **on top of** PyTorch

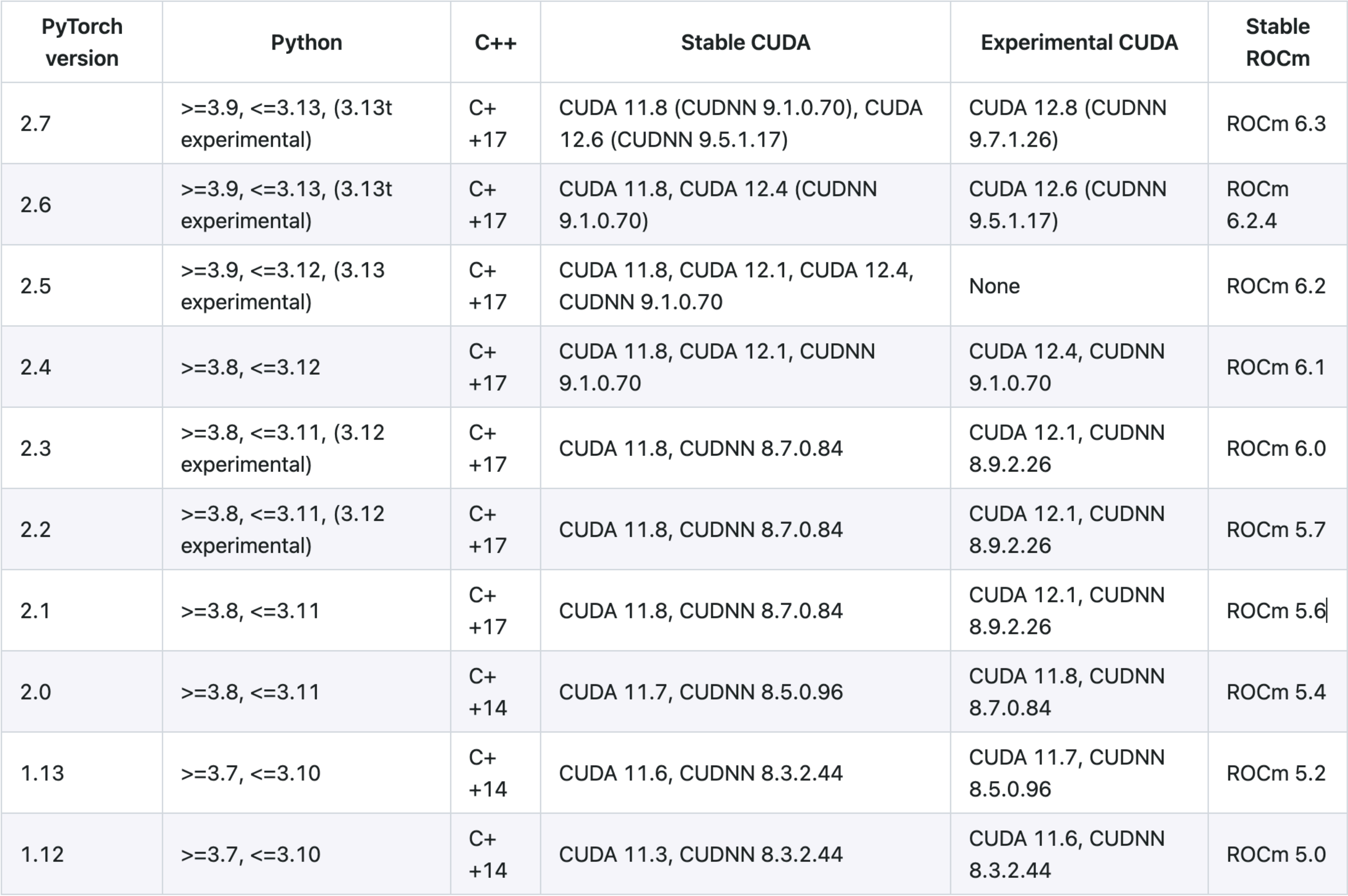
- Support for **distributed training**
- Automatic **logging** and **checkpointing**
- Automatic **optimization** and training loops
- And much more…

Libraries and versions

PyTorch is a very **complex C++ / CUDA** library:

It's useful to know which **versions** we are using

Compatibility Matrix

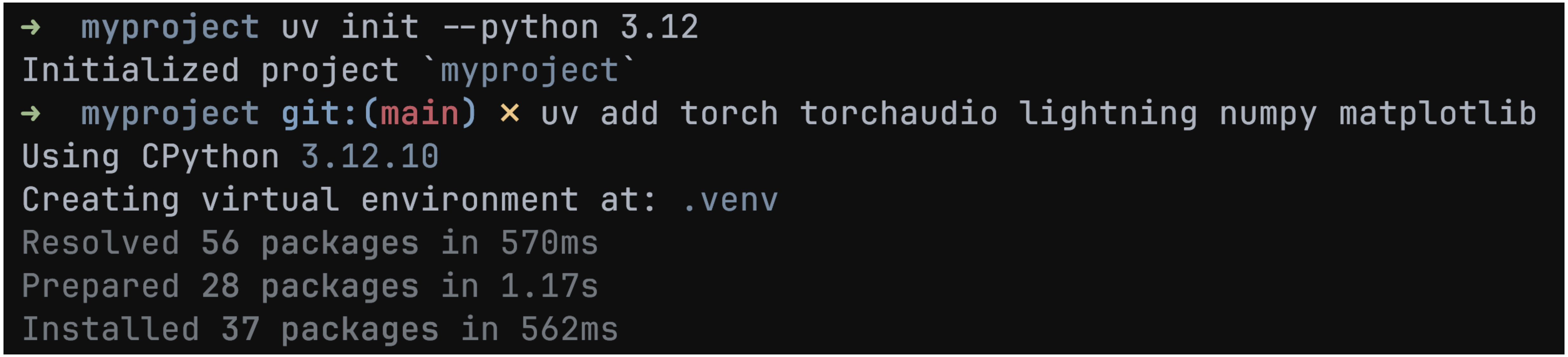


PyTorch version	Python	C++	Stable CUDA	Experimental CUDA	Stable ROCm
2.7	>=3.9, <=3.13, (3.13t experimental)	C++17	CUDA 11.8 (CUDNN 9.1.0.70), CUDA 12.6 (CUDNN 9.5.1.17)	CUDA 12.8 (CUDNN 9.7.1.26)	ROCm 6.3
2.6	>=3.9, <=3.13, (3.13t experimental)	C++17	CUDA 11.8, CUDA 12.4 (CUDNN 9.1.0.70)	CUDA 12.6 (CUDNN 9.5.1.17)	ROCm 6.2.4
2.5	>=3.9, <=3.12, (3.13 experimental)	C++17	CUDA 11.8, CUDA 12.1, CUDA 12.4, CUDNN 9.1.0.70	None	ROCm 6.2
2.4	>=3.8, <=3.12	C++17	CUDA 11.8, CUDA 12.1, CUDNN 9.1.0.70	CUDA 12.4, CUDNN 9.1.0.70	ROCm 6.1
2.3	>=3.8, <=3.11, (3.12 experimental)	C++17	CUDA 11.8, CUDNN 8.7.0.84	CUDA 12.1, CUDNN 8.9.2.26	ROCm 6.0
2.2	>=3.8, <=3.11, (3.12 experimental)	C++17	CUDA 11.8, CUDNN 8.7.0.84	CUDA 12.1, CUDNN 8.9.2.26	ROCm 5.7
2.1	>=3.8, <=3.11	C++17	CUDA 11.8, CUDNN 8.7.0.84	CUDA 12.1, CUDNN 8.9.2.26	ROCm 5.6
2.0	>=3.8, <=3.11	C++14	CUDA 11.7, CUDNN 8.5.0.96	CUDA 11.8, CUDNN 8.7.0.84	ROCm 5.4
1.13	>=3.7, <=3.10	C++14	CUDA 11.6, CUDNN 8.3.2.44	CUDA 11.7, CUDNN 8.5.0.96	ROCm 5.2
1.12	>=3.7, <=3.10	C++14	CUDA 11.3, CUDNN 8.3.2.44	CUDA 11.6, CUDNN 8.3.2.44	ROCm 5.0

Python environments: UV

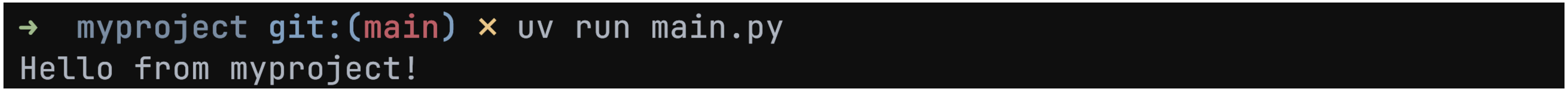
A single tool to replace **pip, pipx, virtualenv, conda**

Fast, **easy** and **reproducible** environments



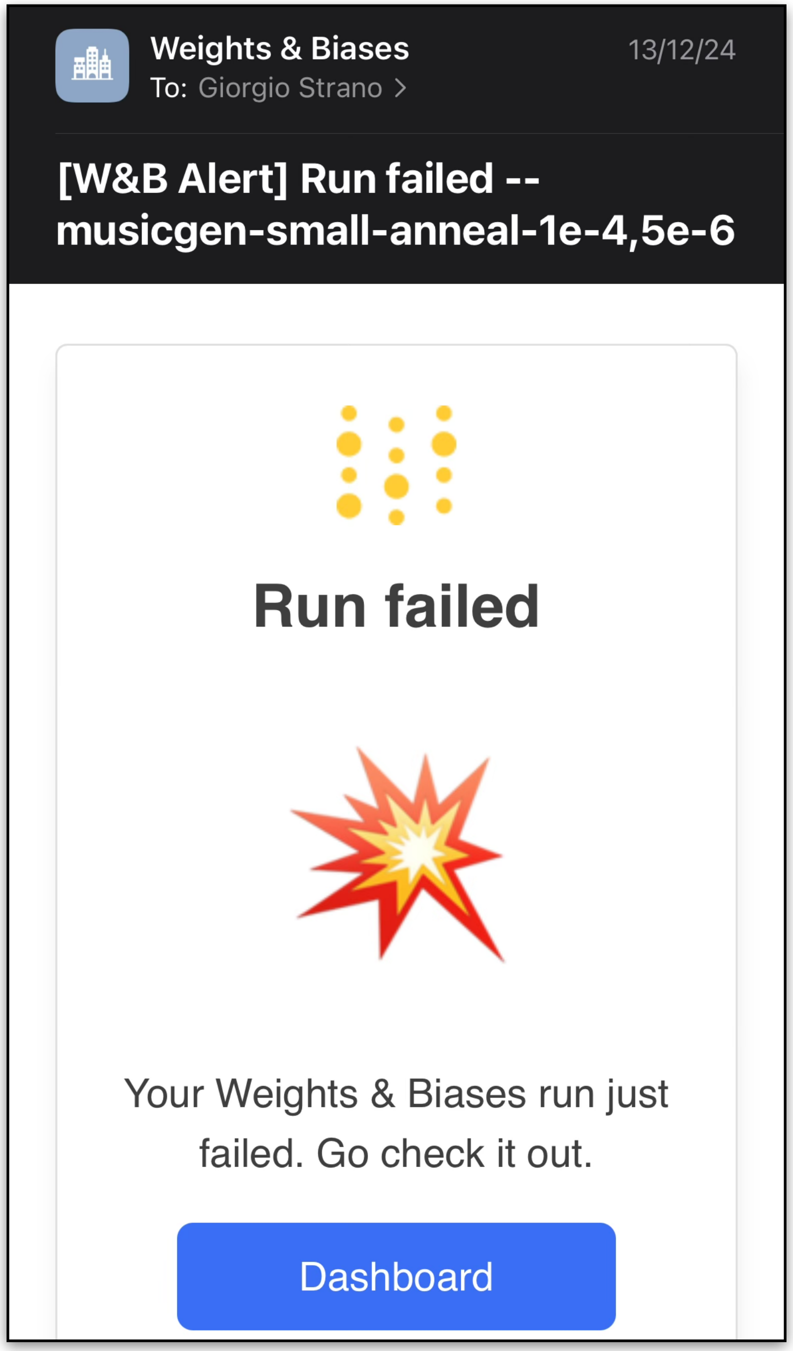
```
➜  myproject uv init --python 3.12
Initialized project `myproject`
➜  myproject git:(main) ✗ uv add torch torchaudio lightning numpy matplotlib
Using CPython 3.12.10
Creating virtual environment at: .venv
Resolved 56 packages in 570ms
Prepared 28 packages in 1.17s
Installed 37 packages in 562ms
```

No need to **activate** the environment: just use **uv run**



```
➜  myproject git:(main) ✗ uv run main.py
Hello from myproject!
```

Weights & Biases



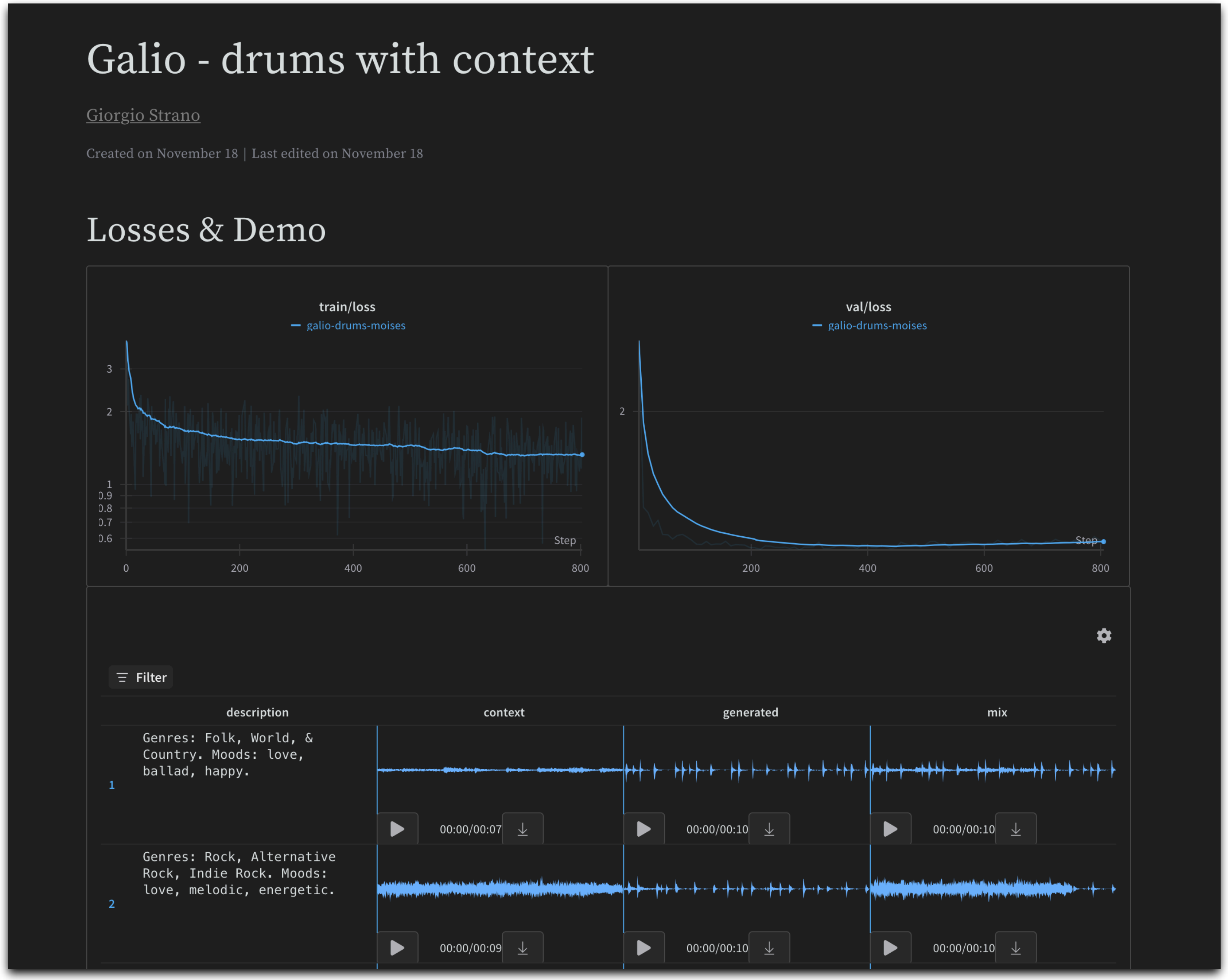
Weights & Biases
To: Giorgio Strano >
13/12/24

[W&B Alert] Run failed -- musicgen-small-anneal-1e-4,5e-6

Run failed

Your Weights & Biases run just failed. Go check it out.

Alerts on run failure



Galio - drums with context

Giorgio Strano

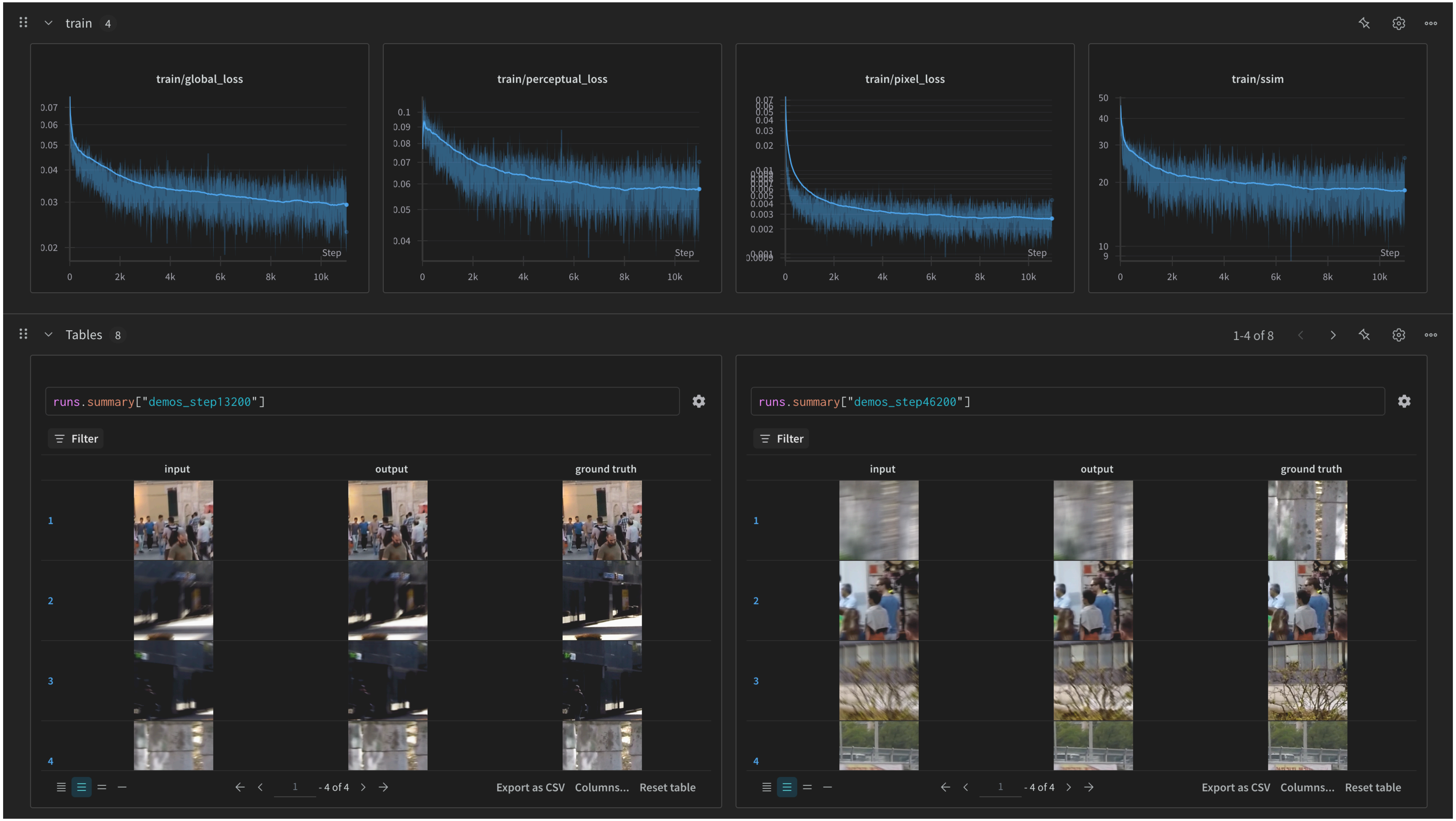
Created on November 18 | Last edited on November 18

Losses & Demo

	description	context	generated	mix
1	Genres: Folk, World, & Country. Moods: love, ballad, happy.	00:00/00:07	00:00/00:10	00:00/00:10
2	Genres: Rock, Alternative Rock, Indie Rock. Moods: love, melodic, energetic.	00:00/00:09	00:00/00:10	00:00/00:10

Training reports

Weights & Biases



```
runs.summary["demos_step13200"]
```

```
runs.summary["demos_step46200"]
```

Real-time logging