UNIVERSITÄT BONN

Supervising software projects.

Moritz Wolter

February 3, 2025

HPCA-Lab, Universität Bonn

Overview

Existing best practices

Adpotion in Machine Learning reserach

Conclusion

Motivation

- Software engineering protects us from critical faliures and bugs.
- In 2024 for example, a failure to test software caused rougly 8.5 million Microsoft Windows operating systems to crash worldwide, causing global disruption of critical services during the CrowdStrike-related IT outages.
- We are in science, we need to be able to reproduce our and others results.

Motivation 2, example from science

Lets play with `https://simpleitk.org/`, a standard library for medical image processing, by opening a DICOM series from the ProstateX dataset.

```
>>> import SimpleITK as sitk
>>> import numpy as np
>>> reader = sitk.ImageSeriesReader()
>>> files = reader.GetGDCMSeriesFileNames(
  './1.3.6.1.4.1.14519.5.2.1.7311.5101.97258762436
>>> reader.SetFileNames(files)
>>> image = reader.Execute()
>>> image.GetSize()
(384, 384, 19)
>>> sitk.GetArrayFromImage(image).shape
(19, 384, 384)
```

Motivation

- Numerous projects rely on the current implementation of `sitk`.
- $\rightarrow$ Software engineering protects the sanity of ourselves and those around us.
- $\rightarrow$ Bugs in test code can invalidate the results of entire projects. They lead to wasted time and embarrassing retractions.

Why Python

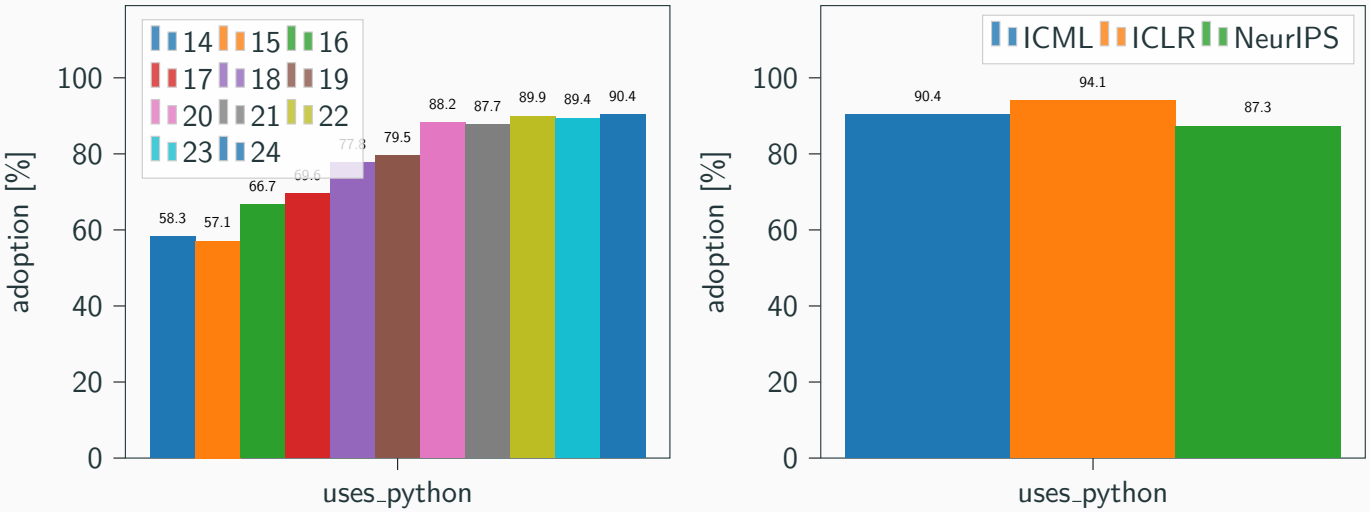


Figure: (left) Python use at the international conference of machine learning (ml) over time. (right) python use of works linked at major 2024 ml conferences.

Existing best practices

Software engineering best practices

- Version control
- Documenting requirements
- Systematic testing
- Packaging code
- Continuous integration

Version control

- Version control is a must for any software project.
- The university of Bonn for example provides a GitLab instance for students and employees at `https://gitlab.uni-bonn.de/`.
- Ask you students to share projects links with you.
- Next we will discuss what to look for in a repository.

Documenting requirements

- With PyTorch, for example, “reproducible results are not guaranteed across PyTorch releases, individual commits, or different platforms” [PyT24]. The same is true for most of the software-world, really. For reproducibility, it is important to document the software environment.
- With the package installer for Python (pip),

  ```
  pip freeze > requirements.txt
  ```

  does the trick,
- recommend

  ```
  conda env export > environment.yml
  ```

  to people using conda.

Systematic testing

- Ask your students to follow a systematic testing approach. Python projects should have a `src` and `tests`-folder [Pyt25c].
- We want tests to run automatically. Point people towards `pytest` [Pyt25a] or `unittest` [Pyt25b].
- Run tests in containers recommend Nox [Flo+25] or Tox [tox25].

Packaging

- Packaging allows others to install code automatically. Ultimately packaging is about reducing code duplication, by making code installable via a Python `import` statement.
- It requires us to structure our code and to create a `pyproject.toml` file.
- Asks you students to read: Packaging Python-Projects from the Python-Packaging-Authority [Pyt25c].

Automatic workflows

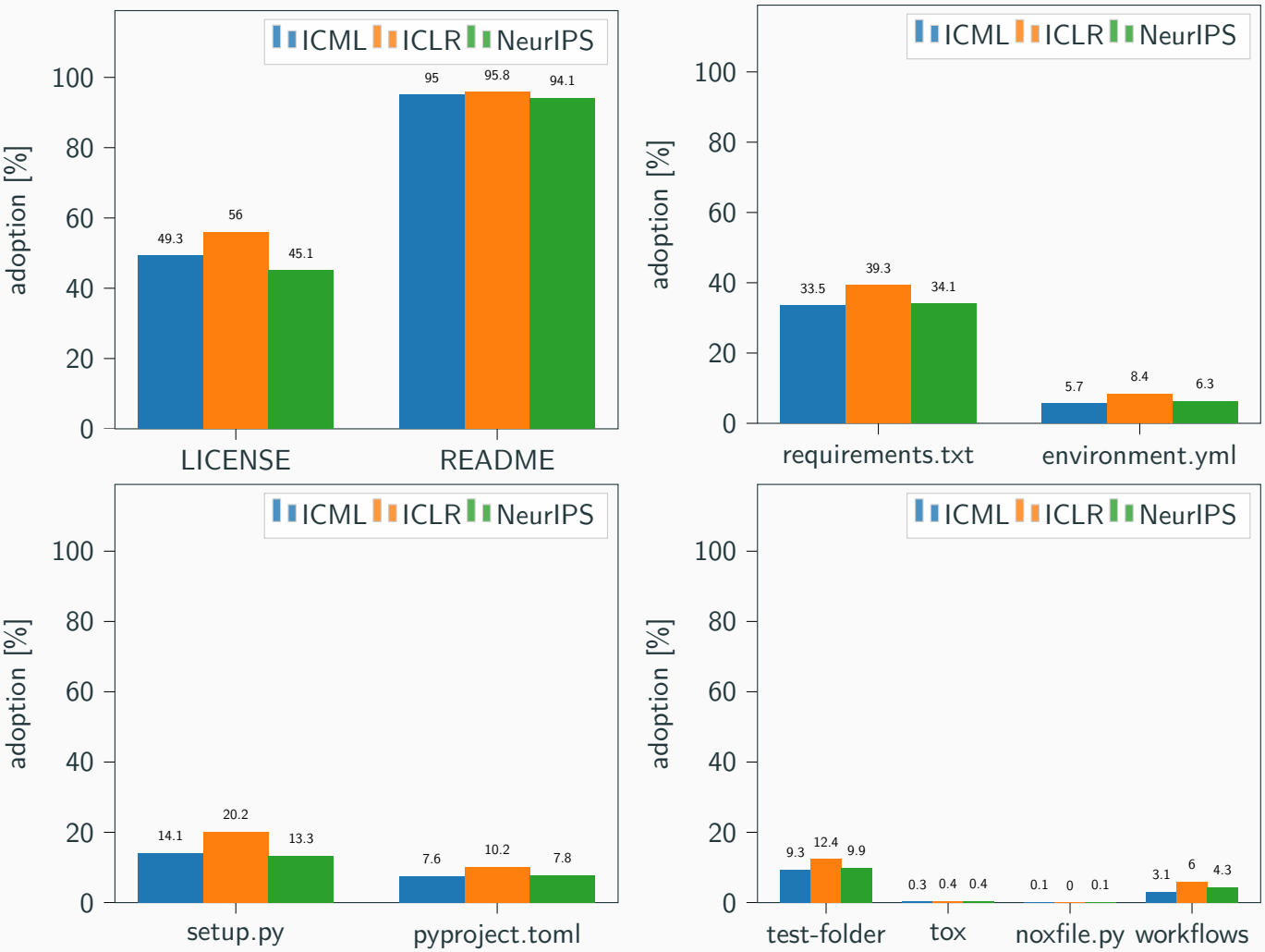
- Programm a server to run tests automatically.
- This is a part of what people describe as continuous integration.
- It helps teams by getting the computer to tell members when they break something.

It typically requires us to write a single configuration file.

Adpotion in Machine Learning reserach

Best Practice adoption in Machine Learning research

Most of the aforementioned best practices are required by the Neurips-Code guide [Sto+20].



Conclusion

Conclusion

- When done well software engineering makes everyones lives easer.
- We get to reuse our own code more easily.
- Others get to use our code, we reach more people and create more impact.
- We protect ourselfs from emberrasing retractions.
- By training networks on tested preprocessing pipelines we reduce waste.
- Continuous integration protects team members from bugs created elsewhere.

References

[Flo+25] Alethea Katherine Flowers, Chris Wilcox, Claudio Jolowicz, Danny Hermes, Diego Ramirez, Henry Schreiner, Luke Sneeringer, Santos Gallegos, and Tom Fleet. ***Nox documentation.*** 2025. URL: https://nox.thea.codes/en/stable/.

[Pyt25a] Pytest-developers. ***Pytest documentation.*** 2025. URL: https://docs.pytest.org/en/stable/.

Literature ii

[Pyt25b] Python-developers. ***unittest - Unit testing framework***. 2025. URL: https://docs.python.org/3/library/unittest.html.

[Pyt25c] Python-Packaging-Authority. ***Packaging Python-Projects***. 2025. URL: https://packaging.python.org/tutorials/packaging-projects/.

[PyT24] PyTorch-Contributors. ***Reproducibility***. 2024. URL: https://pytorch.org/docs/stable/notes/randomness.html.

Literature iii

[Sto+20] Robert Stojnic, Ross Taylor, Sarthak Pati, Fabian-Robert Stöter, Viktor Kerkez andShagun Sodhani, Hamel Husain, Amit Chaudhary, and Rishabh Jain. ***Tips for releasing research code in Machine Learning (with official NeurIPS 2020 recommendations)***. Accessed: 2025-01-28. 2020. URL: https://github.com/paperswithcode/releasing-research-code.

[tox25] tox-devs. ***tox - automation project***. 2025. URL: https://tox.wiki/en/latest/.