Benchopt: Reproducible, efficient and collaborative optimization benchmarks

Journées "Recherche Reproductible"
Thomas Moreau
Reproducible research

Different goals:

▶ Reproduce the exact same results?
▶ Run with new parameters with robust results?
▶ Run with a new dataset?
▶ Extend the results with a new method?
▶ Provide tools for other to use?

Does not require the same set of tools!
Reproducible research

Different goals:

▶ Reproduce the exact same results?
▶ Run with new parameters with robust results?
▶ Run with a new dataset?
▶ Extend the results with a new method?
▶ Provide tools for other to use?

Does not require the same set of tools!

Here is my take.
Extending the results?

Current process in ML to extend results:

▶ Hard to extend existing code.
▶ Re-code methods and tools to integrate a new method.
▶ Competitors’ methods do not work out of the box.

All of this started from scratch by every new method!

Also very cumbersome to add a new dataset, or metric.
Extending the results?

Current process in ML to extend results:

▶ Hard to extend existing code.
▶ Re-code methods and tools to integrate a new method.
▶ Competitors’ methods do not work out of the box.

All of this started from scratch by every new method!

Also very cumbersome to add a new dataset, or metric.

⇒ We need to come up with tools to make it easy!
Extending the results?

Current process in ML to extend results:
▶ Hard to extend existing code.
▶ Re-code methods and tools to integrate a new method.
▶ Competitors’ methods do not work out of the box.

All of this started from scratch by every new method!

Also very cumbersome to add a new dataset, or metric.

⇒ We need to come up with tools to make it easy!

Benchopt produces open, reproducible, extendable benchmarks
How does Benchopt do it?

Benchopt is a framework to organize and run benchmarks:

- one repository per benchmark
- one base open source Python CLI to run them

3 components: Objective, Dataset, Solver

Start yours with https://github.com/benchopt/template_benchmark!
Structure of a benchmark

benchmark/
├── objective.py
├── datasets/
│   ├── dataset1.py
│   └── dataset2.py
├── solvers/
│   ├── solver1.py
│   └── solver2.py

Modular & extendable

New solver? add a file
New dataset? add a file
New metric? modify objective
Components of a benchmark

Dependency relation between Dataset - Objective - Solver

**Dataset**
- `.get_data()`
  - Customizable dictionary of the collected data

**Objective**
- `.set_data()`
  - Arguments from `get_data` to specify the data
- `.get_objective()`
  - Inputs for solvers to solve the optimization problem
- `.compute()`
  - Compute the objective from `get_result` output

**Solver**
- `.set_objective()`
  - Arguments from `get_objective` to compute the objective function
- `.get_result()`
  - Returns the estimated solution computed by the solver

Flexible API so that each component is standalone.
Benchopt makes your life easy

- build on previous benchmarks
- use solvers in Python, R, Julia, binaries...
- monitor any metric you want altogether (test/train loss, ...)
- add parameters to solvers
- share and publish HTML results
- run all benchmarks in parallel
- cache results
- and much more!
Interactive results exploration

Result on resnet classif benchmark

ConvNet classification fitting[model_size=18, model_type=resnet]

Data: CIFAR[framework=pytorch]
Example: Resnet benchmark

- image classification with resnet18
- various optimization strategies
- compare pytorch and tensorflow
- publish reproducible SOTA for baselines

![Graphs showing test error vs time for CIFAR-10, SVHN, and MNIST datasets with different optimization strategies.

Other examples

- Resnet18
- Lasso
- ICA
- Logistic regression
- Federated Learning
- Total Variation
- Ordinary Least Squares
- Non convex sparse regression
- linear SVM
- Bi-level optimization

https://benchopt.github.io/results/
Other examples

- Resnet18
- Lasso
- ICA
- Logistic regression
- Federated Learning
- Total Variation
- Ordinary Least Squares
- Non convex sparse regression
- linear SVM
- Bi-level optimization

https://benchopt.github.io/results/

You can easily add yours! :}
Conclusion

Reproducible research needs more than just releasing code:

- Clean and Documented.
- Reusable.
- Extendable.

Use proper tools to make it possible!

Research is also collaborative: don’t hesitate to report your issues and give feedback :)
Contributors from...