
scanflow Documentation

Release 0.1.0

Gussepe Bravo

Jan 30, 2022

Contents:

1	Scanflow [WORK IN PROGRESS]	1
1.1	Workflow + Agents	2
1.2	Features	2
1.3	Getting started	2
1.4	Dashboard alpha	3
1.5	Tutorials	4
1.6	Installation	4
1.7	References	4
2	Installation	5
2.1	Stable release	5
2.2	From sources	5
3	Usage	7
4	Contributing	9
4.1	Types of Contributions	9
4.2	Get Started!	10
4.3	Pull Request Guidelines	11
4.4	Tips	11
4.5	Deploying	11
5	Credits	13
5.1	Contributors	13
6	History	15
6.1	0.1.0 (2021-04-27)	15
7	Indices and tables	17

Scanflow [WORK IN PROGRESS]

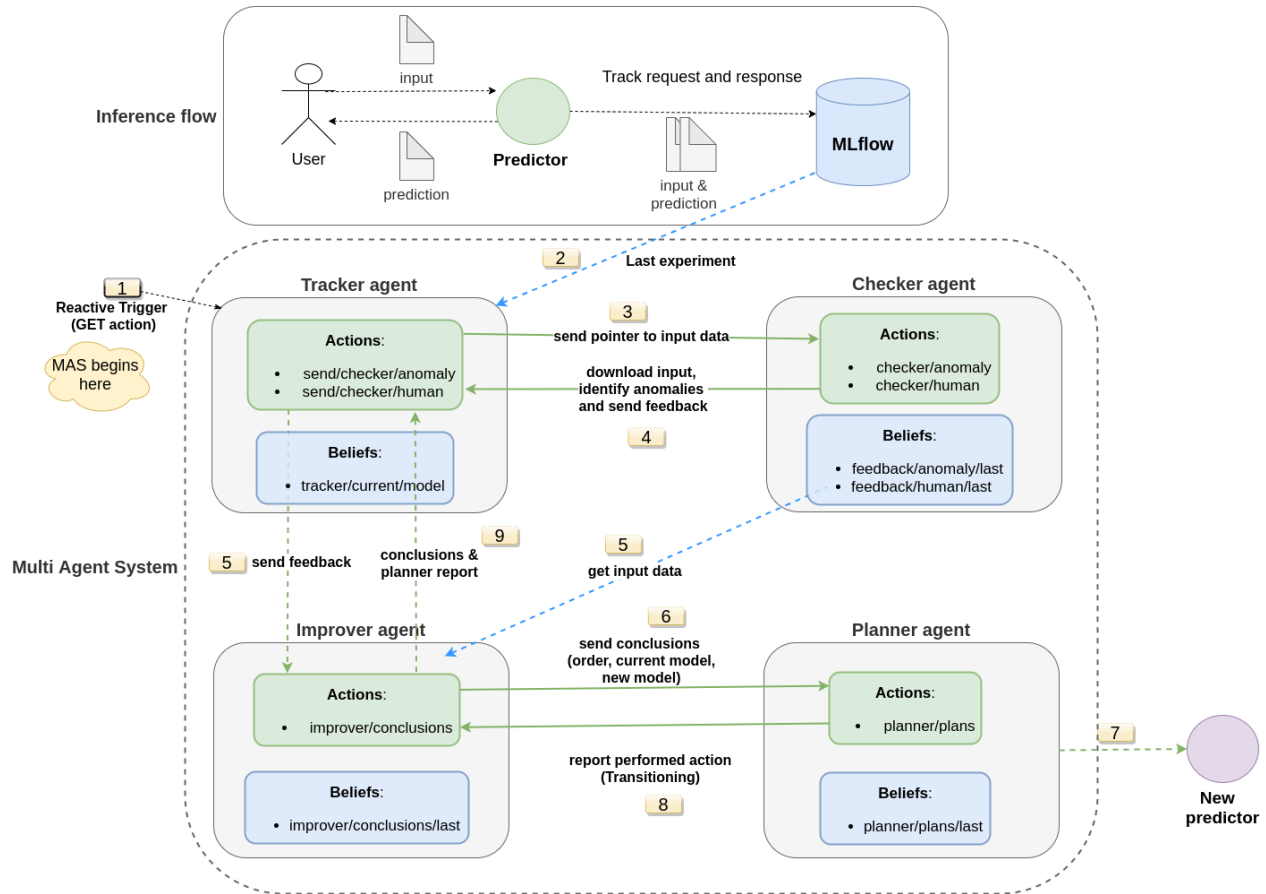
Machine Learning (ML) is more than just training models, the whole workflow must be considered. Once deployed, a ML model needs to be watched and constantly supervised and debugged to guarantee its validity and robustness in unexpected situations. Debugging in ML aims to identify (and address) the model weaknesses in not trivial contexts such as bias in classification, model decay, adversarial attacks, etc., yet there is not a generic framework that allows them to work in a collaborative, modular, portable, iterative way and, more importantly, flexible enough to allow both human and machine tasks to work in a multi-agent environment.

Scanflow allows defining and deploying ML workflows in containers, tracking their metadata, checking their behavior in production, and improving the models by using both learned and human-provided knowledge.

Scanflow is a high-level library that is built on top of MLflow and Docker to manage and supervise workflows efficiently. Its main goals are usability, integration for deployment and real-time checking.

- Free software: MIT license
- Documentation: <https://scanflow.readthedocs.io>.

1.1 Workflow + Agents



1.2 Features

- Ease of use, fast prototyping. Write once use everywhere.
- Portability, scalability (based on docker containers).
- Dynamic nested and parallel workflow execution.
- Workflow tracking (e.g, logs, metrics, settings, results, etc.).
- Workflow checking (e.g, drift distribution, quality data, etc.).
- Orchestrator-agnostic.
- Model version control.

1.3 Getting started

Define your working folder and workflows.

```
import scanflow

from scanflow.setup import Setup, Executor, Workflow
from scanflow.special import Tracker, Checker, Improver, Planner
from scanflow.deploy import Deploy

# App folder
base_path = os.path.dirname(os.getcwd())
app_dir = os.path.join(base_path, "examples/demo_leaf/")

gathering = Executor(name='gathering',
                     file='gathering.py',
                     dockerfile='Dockerfile_gathering')

preprocessing = Executor(name='preprocessing',
                        file='preprocessing.py',
                        requirements='req_preprocessing.txt')

executors = [gathering, preprocessing]
```

Append your workflows and set a tracker. Besides, you can set how to run your workflows (sequentially or in parallel).

```
# Workflows
workflow1 = Workflow(name='workflow1',
                    executors=executors,
                    tracker=Tracker(port=8001))
```

Setup your configuration: build and start the containers. Then, run each workflow.

```
workflows = Setup(app_dir, workflows=[workflow1],
                  verbose=False)

# Define a deployer to build, start and run the workflows
deployer = Deploy(app_dir, setup, verbose=False)

# Build the docker images
deployer.build_workflows()

# Start the containers
deployer.start_workflows()

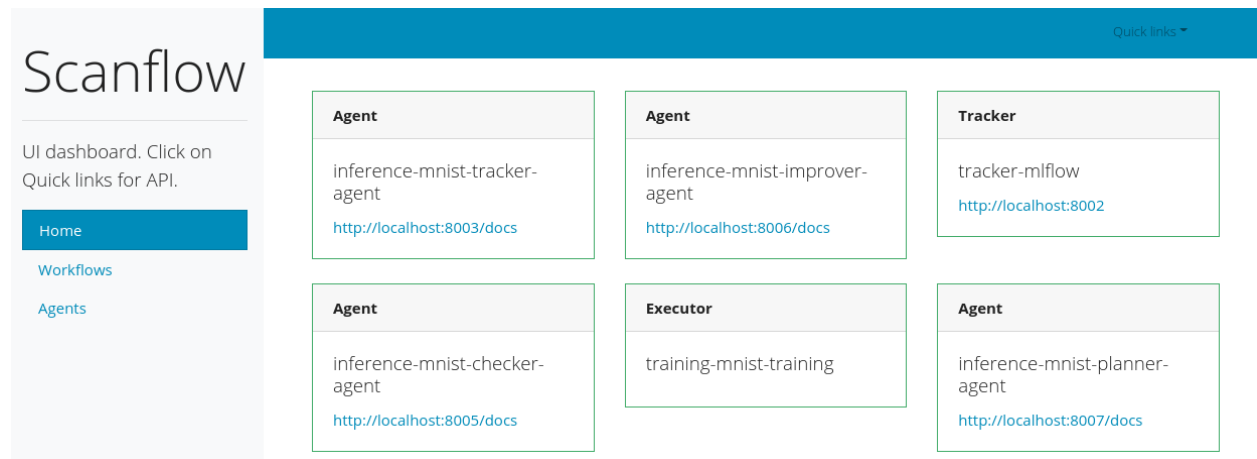
# Run the user's code on the containers
deployer.run_workflows()
```

All the containers will be shown in the scanflow UI, run the following to start the server.

```
python cli.py server --server_port 8050
```

1.4 Dashboard alpha

- Go to: <http://localhost:8050>



1.5 Tutorials

Please check the jupyter notebooks for more examples:

```
tutorials/
```

1.6 Installation

- Install docker.
- `sudo usermod -aG docker <your-user>` (on Linux)

Using conda

```
conda create -n scanflow python=3.6
source activate scanflow
git clone https://github.com/gussepe/scanflow
cd scanflow
pip install -r requirements.txt
```

Using pip (not yet available)

```
pip install scanflow
```

1.7 References

Colab experiments

- MNIST: https://colab.research.google.com/drive/1t0EgpPk5_mEMNb_AvN7yJ_Tz88bqgS9A?usp=sharing
- FashionMNIST: <https://colab.research.google.com/drive/1tdGqg5WAGGhdZEd0HI5ZYQtRFAkrOvJO?usp=sharing>
- CIFAR-10: <https://colab.research.google.com/drive/147NDiwsY86xpUIsM-btMPMSzTktsNd6?usp=sharing>

2.1 Stable release

To install scanflow, run this command in your terminal:

```
$ pip install scanflow
```

This is the preferred method to install scanflow, as it will always install the most recent stable release.

If you don't have [pip](#) installed, this [Python installation guide](#) can guide you through the process.

2.2 From sources

The sources for scanflow can be downloaded from the [Github repo](#).

You can either clone the public repository:

```
$ git clone git://github.com/gussepe/scanflow
```

Or download the [tarball](#):

```
$ curl -OL https://github.com/gussepe/scanflow/tarball/master
```

Once you have a copy of the source, you can install it with:

```
$ python setup.py install
```


CHAPTER 3

Usage

To use scanflow in a project:

```
import scanflow
```


Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given. You can contribute in many ways:

4.1 Types of Contributions

4.1.1 Report Bugs

Report bugs at <https://github.com/gussepe/scanflow/issues>.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

4.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with “bug” and “help wanted” is open to whoever wants to implement it.

4.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with “enhancement” and “help wanted” is open to whoever wants to implement it.

4.1.4 Write Documentation

scanflow could always use more documentation, whether as part of the official scanflow docs, in docstrings, or even on the web in blog posts, articles, and such.

4.1.5 Submit Feedback

The best way to send feedback is to file an issue at <https://github.com/gussepe/scanflow/issues>.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

4.2 Get Started!

Ready to contribute? Here's how to set up *scanflow* for local development.

1. Fork the *scanflow* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:gussepe/scanflow.git
```

3. Install your local copy into a virtualenv. Assuming you have virtualenvwrapper installed, this is how you set up your fork for local development:

```
$ virtualenv scanflow
$ cd scanflow/
$ python setup.py develop
```

4. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

Now you can make your changes locally.

5. When you're done making changes, check that your changes pass flake8 and the tests, including testing other Python versions with tox:

```
$ flake8 scanflow tests
$ python setup.py test or py.test
$ tox
```

To get flake8 and tox, just pip install them into your virtualenv.

6. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

7. Submit a pull request through the GitHub website.

4.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 2.7, 3.4, 3.5 and 3.6, and for PyPy. Check https://travis-ci.org/gussepe/scanflow/pull_requests and make sure that the tests pass for all supported Python versions.

4.4 Tips

To run a subset of tests:

```
$ python -m unittest tests.test_scanflow
```

4.5 Deploying

A reminder for the maintainers on how to deploy. Make sure all your changes are committed (including an entry in HISTORY.rst). Then run:

```
$ bumpversion patch # possible: major / minor / patch
$ git push
$ git push --tags
```

Travis will then deploy to PyPI if tests pass.

5.1 Contributors

- Gusseppe Bravo <gusseppibravo@bsc.es>
- Peini Liu <peini.liu@bsc.es>
- Jordi Guitart <jordi.guitart@bsc.es>

6.1 0.1.0 (2021-04-27)

- First alpha release.

CHAPTER 7

Indices and tables

- `genindex`
- `modindex`
- `search`