

PathML: Tools for computational pathology

downloads 36k

docs passing

 codecov 86%

code style black



pypi v2.1.1

☆ PathML objective is to lower the barrier to entry to digital pathology

Imaging datasets in cancer research are growing exponentially in both quantity and information density. These massive datasets may enable derivation of insights for cancer research and clinical care, but only if researchers are equipped with the tools to leverage advanced computational analysis approaches such as machine learning and artificial intelligence. In this work, we highlight three themes to guide development of such computational tools: scalability, standardization, and ease of use. We then apply these principles to develop PathML, a general-purpose research toolkit for computational pathology. We describe the design of the PathML framework and demonstrate applications in diverse use cases.

The fastest way to get started?

```
docker pull pathml/pathml && docker run -it -p 8888:8888 pathml/pathml
```

Branch	Test status
master	 Tests linux no status
dev	 Tests linux passing

Please note that the :construction: dev branch is under active development, with experimental features, bug fixes, and refactors that may happen at any time! Stable versions are available as tagged releases on GitHub, or as versioned releases on PyPI

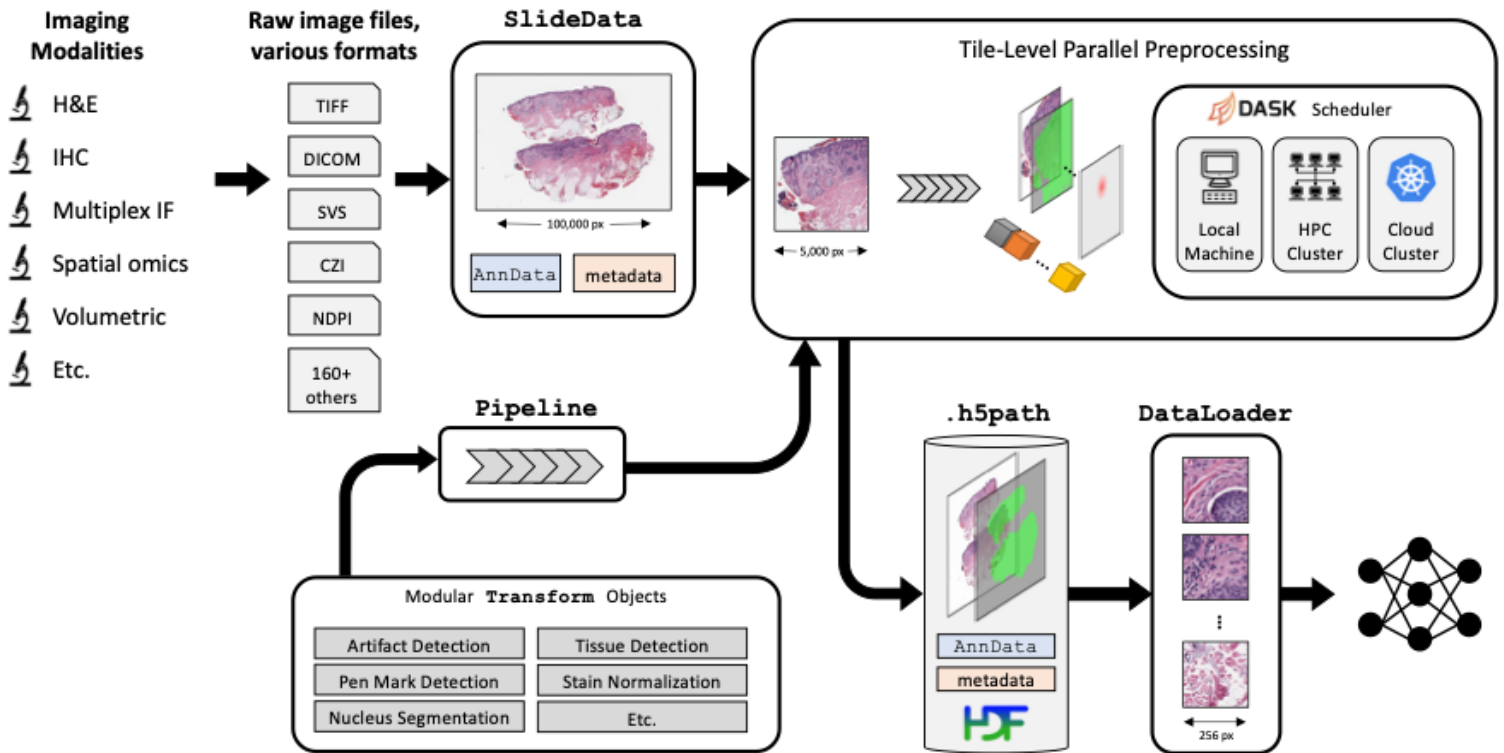
Official Documentation

View the official [PathML Documentation on readthedocs](#)

Examples! Examples! Examples!

[↴ Jump to the gallery of examples below](#)

PathML



1. Installation

PathML is an advanced tool for pathology image analysis. Below are simplified instructions to help you install PathML on your system. Whether you're a user or a developer, follow these steps to get started.

1.1 Prerequisites

We recommend using [Conda](#) for managing your environments.

Installing Conda

If you don't have Conda installed, you can download Miniconda [here](#).

Updating Conda and Using libmamba (Optional)

Recent versions of Conda have integrated `libmamba`, a faster dependency solver. To benefit from this improvement, first ensure your Conda is updated:

```
conda update -n base conda
```

Then, to install and set the new `libmamba` solver, run:

```
conda install -n base conda-libmamba-solver
conda config --set solver libmamba
```

Note: these instructions are for Linux. Commands may be different for other platforms.

Platform-Specific External Dependencies

For installation methods [1\)](#) and [2\)](#), you will need to install the following platform-specific packages.

- Linux: Install external dependencies with [Apt](#):

```
sudo apt-get install openslide-tools g++ gcc libblas-dev liblapack-dev
```

- MacOS: Install external dependencies with [Brew](#):

```
brew install openslide
```

- Windows:

1. Option A: Install with [vcpkg](#):

```
vcpkg install openslide
```

2. Option B: Using Pre-built OpenSlide Binaries (Alternative) For Windows users, an alternative to

using `vcpkg` is to download and use pre-built OpenSlide binaries. This method is recommended if you prefer a quicker setup.

- Download the OpenSlide Windows binaries from the [OpenSlide Downloads](#) page.
- Extract the archive to your desired location, e.g., `C:\OpenSlide\`.

1.2 PathML Installation Methods

1.2.1 Install with pip (Recommended for Users)

Create and Activate Conda Environment

```
conda create --name pathml python=3.9
conda activate pathml
```

Install OpenJDK

```
conda install -c conda-forge 'openjdk<=18.0'
```

Install PathML from PyPI

```
pip install pathml
```

1.2.2 Install from Source (Recommended for Developers)

Clone repository

```
git clone https://github.com/Dana-Farber-AIOS/pathml.git
cd pathml
```

Create conda environment

- Linux and Windows:

```
conda env create -f environment.yml
conda activate pathml
```

To use GPU acceleration for model training or other tasks, you must install CUDA. The default CUDA version in our environment file is 11.6. To install a different CUDA version, refer to the instructions [here](#)).

- MacOS:

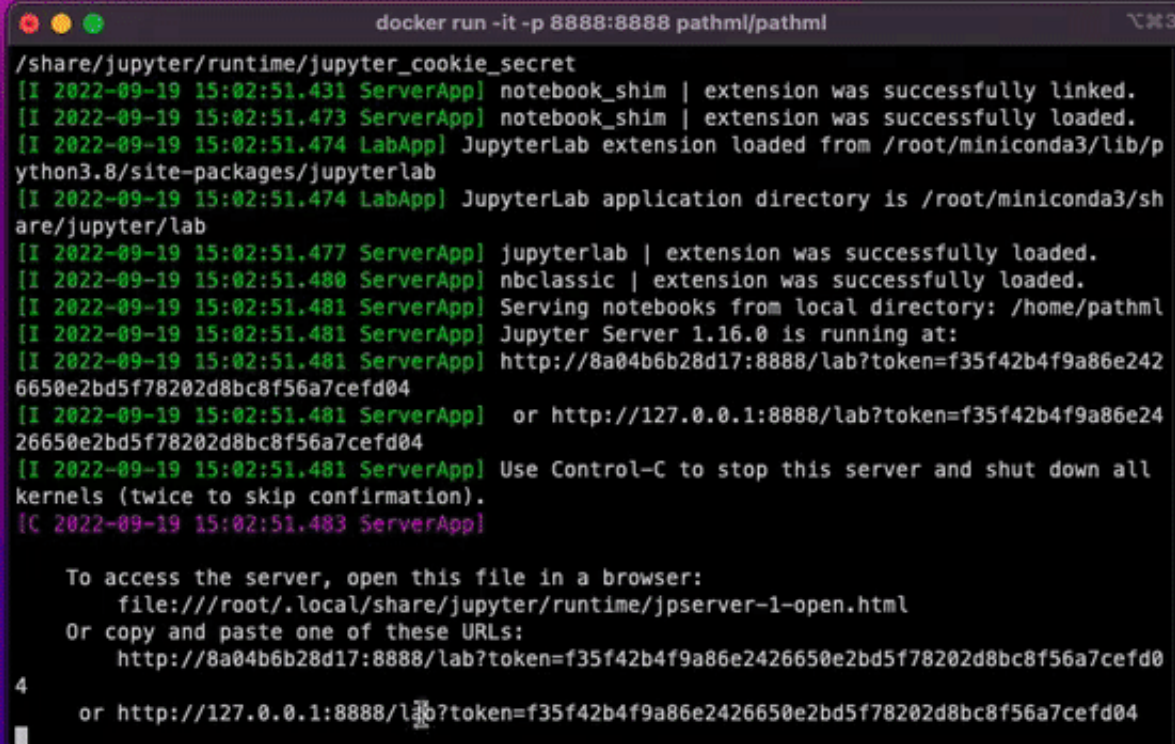
```
conda env create -f requirements/environment_mac.yml
conda activate pathml
```

Install PathML from source:

```
pip install -e .
```

1.2.3 Use Docker Container

First, download or build the PathML Docker container:



```

docker run -it -p 8888:8888 pathml/pathml

/share/jupyter/runtime/jupyter_cookie_secret
[I 2022-09-19 15:02:51.431 ServerApp] notebook_shim | extension was successfully linked.
[I 2022-09-19 15:02:51.473 ServerApp] notebook_shim | extension was successfully loaded.
[I 2022-09-19 15:02:51.474 LabApp] JupyterLab extension loaded from /root/miniconda3/lib/python3.8/site-packages/jupyterlab
[I 2022-09-19 15:02:51.474 LabApp] JupyterLab application directory is /root/miniconda3/share/jupyter/lab
[I 2022-09-19 15:02:51.477 ServerApp] jupyterlab | extension was successfully loaded.
[I 2022-09-19 15:02:51.480 ServerApp] nbclassic | extension was successfully loaded.
[I 2022-09-19 15:02:51.481 ServerApp] Serving notebooks from local directory: /home/pathml
[I 2022-09-19 15:02:51.481 ServerApp] Jupyter Server 1.16.0 is running at:
[I 2022-09-19 15:02:51.481 ServerApp] http://8a04b6b28d17:8888/lab?token=f35f42b4f9a86e2426650e2bd5f78202d8bc8f56a7cefd04
[I 2022-09-19 15:02:51.481 ServerApp] or http://127.0.0.1:8888/lab?token=f35f42b4f9a86e2426650e2bd5f78202d8bc8f56a7cefd04
[I 2022-09-19 15:02:51.481 ServerApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 2022-09-19 15:02:51.483 ServerApp]

To access the server, open this file in a browser:
file:///root/.local/share/jupyter/runtime/jpserver-1-open.html
Or copy and paste one of these URLs:
http://8a04b6b28d17:8888/lab?token=f35f42b4f9a86e2426650e2bd5f78202d8bc8f56a7cefd04
4
or http://127.0.0.1:8888/lab?token=f35f42b4f9a86e2426650e2bd5f78202d8bc8f56a7cefd04

```

- Option A: download PathML container from Docker Hub

```
docker pull pathml/pathml:latest
```

Optionally specify a tag for a particular version, e.g. `docker pull pathml/pathml:2.0.2`. To view possible tags, please refer to the [PathML DockerHub page](#).

- Option B: build docker container from source

```
git clone https://github.com/Dana-Farber-AIOS/pathml.git
cd pathml
docker build -t pathml/pathml .
```

Then connect to the container:

```
docker run -it -p 8888:8888 pathml/pathml
```

The above command runs the container, which is configured to spin up a jupyter lab session and expose it on port 8888. The terminal should display a URL to the jupyter lab session starting with `http://127.0.0.1:8888/lab?token=<.....>`. Navigate to that page and you should connect to the jupyter lab session running on the container with the pathml environment fully configured. If a password is requested, copy the string of characters following the `token=` in the url.

Note that the docker container requires extra configurations to use with GPU.

Note that these instructions assume that there are no other processes using port 8888.

Please refer to the `Docker run` [documentation](#) for further instructions on accessing the container, e.g. for mounting volumes to access files on a local machine from within the container.

1.2.4 Use Google Colab

To get PathML running in a Colab environment:

```
import os
!pip install openslide-python
!apt-get install openslide-tools
!apt-get install openjdk-17-jdk-headless -qq > /dev/null
os.environ["JAVA_HOME"] = "/usr/lib/jvm/java-17-openjdk-amd64"
!update-alternatives --set java /usr/lib/jvm/java-17-openjdk-amd64/bin/java
!java -version
!pip install pathml
```

Thanks to all of our open-source collaborators for helping maintain these installation instructions!

Please open an issue for any bugs or other problems during installation process.

1.3. Import PathML

After you have installed all necessary dependencies and PathML itself, import it using the following command:

```
import pathml
```

For Windows users, insert the following code snippet at the beginning of your Python script or Jupyter notebook before importing PathML. This code sets up the DLL directory for OpenSlide, ensuring that the library is properly loaded:

```
# The path can also be read from a config file, etc.
OPENSIDE_PATH = r'c:\path\to\openside-win64\bin'

import os
if hasattr(os, 'add_dll_directory'):
    # Windows-specific setup
    with os.add_dll_directory(OPENSIDE_PATH):
        import openside
else:
    # For other OSes, this step is not needed
    import openside

# Now you can proceed with using PathML
import pathml
```

This code snippet ensures that the OpenSlide DLLs are correctly found by Python on Windows systems. Replace `c:\path\to\openside-win64\bin` with the actual path where you extracted the OpenSlide binaries.

If you encounter any DLL load failures, verify that the OpenSlide `bin` directory is correctly added to your `PATH` .

1.4 CUDA

To use GPU acceleration for model training or other tasks, you must install CUDA. This guide should work, but for the most up-to-date instructions, refer to the [official PyTorch installation instructions](#).

Check the version of CUDA:

```
nvidia-smi
```

Replace both instances of 'cu116' in `requirements/requirements_torch.txt` with the CUDA

version you see. For example, for CUDA 11.7, 'cu116' becomes 'cu117'.

Then create the environment:

```
conda env create -f environment.yml
conda activate pathml
```

After installing PyTorch, optionally verify successful PyTorch installation with CUDA support:

```
python -c "import torch; print(torch.cuda.is_available())"
```

2. Using with Jupyter (optional)

Jupyter notebooks are a convenient way to work interactively. To use `PathML` in Jupyter notebooks:

2.1 Set `JAVA_HOME` environment variable

`PathML` relies on Java to enable support for reading a wide range of file formats. Before using `PathML` in Jupyter, you may need to manually set the `JAVA_HOME` environment variable specifying the path to Java. To do so:

1. Get the path to Java by running `echo $JAVA_HOME` in the terminal in your `pathml` conda environment (outside of Jupyter)
2. Set that path as the `JAVA_HOME` environment variable in Jupyter:

```
import os
os.environ["JAVA_HOME"] = "/opt/conda/envs/pathml" # change path as needed
```

2.2 Register environment as an IPython kernel

```
conda activate pathml
conda install ipykernel
```

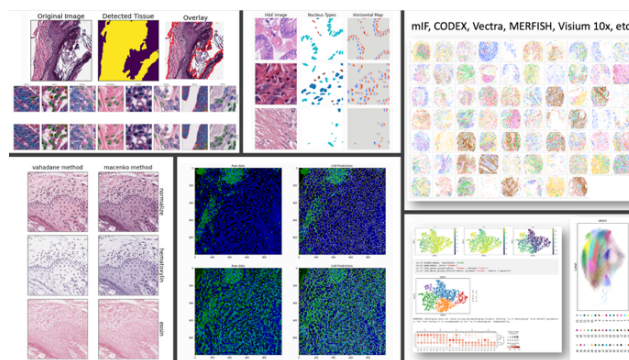
```
python -m ipykernel install --user --name=pathml
```

This makes the pathml environment available as a kernel in jupyter lab or notebook.

3. Examples

Now that you are all set with PathML installation, let's get started with some analyses you can easily replicate:

1. Load over 160+ different types of pathology images using PathML
2. H&E Stain Deconvolution and Color Normalization
3. Brightfield imaging pipeline: load an image, preprocess it on a local cluster, and get it read for machine learning analyses in PyTorch
4. Multiparametric Imaging: Quickstart & single-cell quantification
5. Multiparametric Imaging: CODEX & nuclei quantization
6. Train HoVer-Net model to perform nucleus detection and classification, using data from PanNuke dataset
7. Gallery of PathML preprocessing and transformations
8. Use the new Graph API to construct cell and tissue graphs from pathology images
9. Train HACTNet model to perform cancer subtyping using graphs constructed from the BRACS dataset
10. Perform reconstruction of tiles obtained from pathology images using Tile Stitching
11. Create an ONNX model in HaloAI or similar



software, export it, and run it at scale using PathML

12. Step-by-step process used to analyze the Whole Slide Images (WSIs) of Non-Small Cell Lung Cancer (NSCLC) samples as published in the Journal of Clinical Oncology

4. Citing & known uses

If you use PathML please cite:

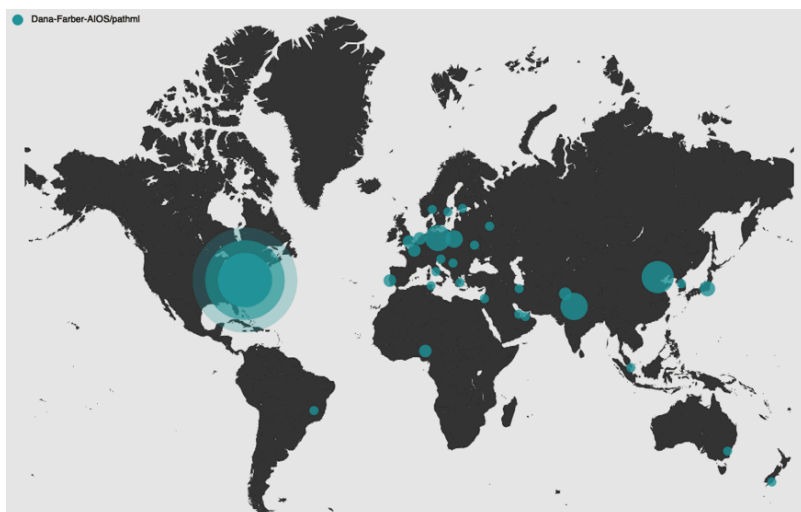
- [J. Rosenthal et al., "Building tools for machine learning and artificial intelligence in cancer research: best practices and a case study with the PathML toolkit for computational pathology." Molecular Cancer Research, 2022.](#)

So far, **PathML** was referenced in 20+ manuscripts:

- [A. Song et al. Nature Reviews Bioengineering 2023](#)
- [I. Virshup et al. Nature Bioengineering 2023](#)
- [A. Karargyris et al. Nature Machine Intelligence 2023](#)
- [S. Pati et al. Nature Communications Engineering 2023](#)
- [C. Gorman et al. Nature Communications 2023](#)
- [J. Nyman et al. Cell Reports Medicine 2023](#)
- [A. Shmatko et al. Nature Cancer 2022](#)
- [J. Pocock et al. Nature Communications Medicine 2022](#)
- [S. Orsulic et al. Frontiers in Oncology 2022](#)
- [D. Brundage et al. arXiv 2022](#)
- [C. Lama et al. bioRxiv 2022](#)
- [J. Linares et al. Molecular Cell 2021](#)
- the list continues [here](#) 

5. Users

This is where in the world our most enthusiastic supporters are located:



and this is where they work:



Source: <https://ossinsight.io/analyze/Dana-Farber-AIOS/pathml#people>

6. Contributing

PathML is an open source project. Consider contributing to benefit the entire community!

There are many ways to contribute to PathML , including:

- Submitting bug reports
- Submitting feature requests
- Writing documentation and examples
- Fixing bugs
- Writing code for new features
- Sharing workflows
- Sharing trained model parameters
- Sharing PathML with colleagues, students, etc.

See [contributing](#) for more details.

7. License

The GNU GPL v2 version of PathML is made available via Open Source licensing. The user is free to use, modify, and distribute under the terms of the GNU General Public License version 2.

Commercial license options are available also.

8. Contact

Questions? Comments? Suggestions? Get in touch!

pathml@dfci.harvard.edu



Dana-Farber
Cancer Institute



Weill Cornell
Medicine