The Journal of Open Source Software

# geoai-datacubes: AI-ready multi-mission satellite data cubes for Earth-observation machine learning

**Joachim Moortgat** [1,2¶], **Bhavika Jain** [3], **Aswathnarayan Radhakrishnan** [3], **Satyaki Roy Chowdhury** [3], **and Hsiao Jou Hsu** [1]

**1** School of Earth Sciences, The Ohio State University, Columbus, OH, USA **2** BuckAI Observatory, College of Arts and Sciences, The Ohio State University, Columbus, OH, USA **3** Department of Computer Science and Engineering, The Ohio State University, Columbus, OH, USA ¶ Corresponding author

**DOI:** 10.21105/joss.11034

**Software**

- Review
- Repository
- Archive

**Editor:** Andrew Walker

**Reviewers:**

- @gmarupilla
- @betolink
- @wood-chris

**Submitted:** 04 July 2026
**Published:** 13 November 2026

**License**
Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License (CC BY 4.0).

## Summary

geoai-datacubes is an open-source Python pipeline that turns raw multi-mission Earth-observation imagery into AI-ready data cubes for machine-learning and deep-learning workflows. From a single configuration block — a region of interest, a time window, a set of bands — it fetches, co-registers, fuses, and tiles imagery from **26 missions**. The direct sensor observations are Sentinel-2 L1C and L2A (Drusch et al., 2012), Sentinel-1 RTC (Torres et al., 2012), Landsat 8/9 C2 L2 (Roy et al., 2014), NAIP, PlanetScope 4-band and 8-band imagery, MODIS reflectance and land-surface temperature, HLS, ALOS PALSAR, Copernicus DEM at 30 m and 90 m, and USGS 3DEP. The derived products are ESA WorldCover (Zanaga et al., 2022), ALOS Forest / Non-Forest, USDA CDL, USGS LCMAP, Impact Observatory annual LULC, JRC Global Surface Water, Hansen Global Forest Change, and Chloris biomass. Per-mission details are in `docs/data_layers.md`. Four data providers are unified behind one dispatcher — Element 84 Earth Search, Microsoft Planetary Computer, the commercial Planet Orders API, and a `direct_http` path for non-STAC datasets such as Hansen GFC on Google Cloud Storage. The output is a multi-band GeoTIFF or Zarr cube on a common Universal Transverse Mercator (UTM) grid at a user-specified resolution, ready to be consumed by a PyTorch `Dataset` that streams training tiles on the fly without ever materialising them to disk. Three self-bootstrapping Colab notebooks ship with the repository as reviewed examples: a *grand tour* of every mission, an *LULC classification* notebook that trains and compares logistic regression, random forest (Breiman, 2001), XGBoost (Chen & Guestrin, 2016) and a lightweight U-Net (Ronneberger et al., 2015) against a conditional NDWI (McFeeters, 1996) spectral-index baseline, and an *integration* notebook that composes geoai-datacubes with the opengeos/geoai modelling library (Wu, 2026) across a multi-AOI Cleveland / Cincinnati / Columbus experiment. The `smoke-tests/` folder includes Slurm and Bash scripts for every mission.

## Statement of need

As foundation models and pretrained wrappers for Earth observation become one `pip install` away, the bottleneck for machine-learning research on satellite imagery is more often the data preparation pipeline than the model itself. A typical workflow stitches together vendor SDKs, manual reprojection between coordinate systems, per-mission cloud masks, ad hoc nodata handling, band-specific normalisation, and thousands of tile files written to disk. Each step silently degrades the final model: cloud-smeared nodata biases predictions, integer QA bands resampled bilinearly become nonsense, unnormalised digital number (DN) values can cause CNN training to fail without an error message. `geoai-datacubes` consolidates these steps

into a single configuration-driven pipeline. The library targets graduate students and postdocs entering Earth-observation machine learning, researchers running cross-modal experiments (optical + SAR + DEM + LULC labels), HPC users moving training jobs from laptop to cluster, and instructors teaching applied remote sensing.

## State of the field

`geoai-datacubes` is complementary to the modelling-breadth `geoai` package (Wu, 2026), which wraps foundation and task-specific models but handles raster missions one at a time — no multi-mission fusion, per-band normalisation, or spatially-aware splits. Our library fills exactly that data-engineering gap, and a bundled `select_bands` helper plus `BAND_PRESETS` dictionary lets `geoai`'s loaders consume our cubes without modification; notebook 03 demonstrates the end-to-end hand-off with honest in-distribution F1 $\approx 0.95$ and held-out-city F1 $\approx 0.05$. Beyond `geoai`, PySTAC (STAC Community, 2024) handles STAC catalogue queries, Rasterio (Gillies & others, 2023) and xarray (Hoyer & Hamman, 2017) handle raster I/O, and TorchGeo (Stewart et al., 2022) provides PyTorch datasets for pre-existing benchmark cubes and is the lingua franca for the raster-dataset abstraction. None address the per-band-metadata, multi-mission-fusion, spatially-aware-split, or HPC-integration concerns `geoai-datacubes` exists to solve.

## Software design

Machine learning on multi-mission satellite data needs every band to be normalised according to what it measures, all missions to share one pixel grid, and missing data and train / test splits to be handled without leaking information between tiles. Three architectural decisions follow from these requirements and distinguish `geoai-datacubes` from generic raster tooling:

1. **A declarative per-band metadata system (`band_meta`)** pairing every band with a `kind` (`spectral`, `sar`, `elevation`, `temperature`, `index`, `categorical`, `qa`) and a normalisation recipe (`linear`, `log_db`, `mean_subtract`, `kelvin_to_celsius_norm`, `divide`, `one_hot`, `passthrough`, …). Defaults are documented and inspectable; `apply_band_norm` and `get_band_norm` produce ML-ready features by applying those defaults explicitly. A regex fallback infers a sensible recipe for unregistered bands.
2. **Multi-mission fusion onto a common UTM grid (`fuse_response_tiffs`)** with band-appropriate resampling — bilinear for continuous reflectance and elevation, nearest-neighbour for categorical and QA bands — and provenance preserved in mission-prefixed band descriptions. A fused cube is a single multi-band cloud-optimised GeoTIFF that any TorchGeo-style PyTorch dataset can consume directly.
3. **Band-aware handling of missing data and spatially-aware data sampling.** With `nan_handling="auto"`, per-kind NaN fill (mean for spectral / SAR, biharmonic in-painting for elevation, nearest-neighbour fill rounded to integer class values for categorical bands, tile-drop on QA NaN) plus four split strategies (`random`, `block`, `stripes`, `regions`) reduce the risk of train/test data leakage associated with random tile splits in spatially-autocorrelated imagery.

A `LazyTileDataset` class lets researchers sweep tile size, stride, augmentation, NaN-handling, and split-method choices at training time without ever materialising a tile to disk.

## Research impact statement

The pipeline underpins ongoing research within the authors' group. Hsu & Moortgat (2026) use it to fuse Sentinel-2 reflectance with airborne LiDAR bathymetry and benchmark classical machine learning against deep learning for transferable satellite-derived bathymetry. A parallel interpretability line fine-tunes deep networks on Sentinel-2 cubes assembled by the same

pipeline and probes how those networks weight individual bands when regressing water depth (Chowdhury et al., 2026). The library is being adopted by additional graduate students in the BuckAI Observatory for water-resource remote sensing and cross-city land-cover-transfer experiments.

## Acknowledgements

This work was supported by The Ohio State University's BuckAI Observatory and School of Earth Sciences. The pipeline's foundational Sentinel-1 / Sentinel-2 acquisition and tiling layer was prototyped by Jain and Radhakrishnan in August–October 2025; the multi-mission expansion, ML/DL notebooks, and the present documentation are by Moortgat; and review and testing during 2026 were conducted by Chowdhury and Hsu. Open data are provided by ESA Copernicus, USGS, and the Copernicus DEM programme; cloud-hosted imagery by Element 84 and Microsoft Planetary Computer; commercial PlanetScope by Planet's Education and Research Program.

## AI usage disclosure

Code development since May 2026 was accelerated by Anthropic's Claude Code (Opus 4.5–4.7 and Sonnet 4.5–4.6 models), which assisted with code generation, refactoring, and drafting. All architectural decisions were made by the human authors, who reviewed, edited, and validated every AI-generated artefact before commit.

The repository's commit history is unevenly distributed in time, with insertions concentrated in a small number of multi-day windows in December 2025 and June 2026. This reflects a common academic-software pattern: the code was developed for several months on local workstations and HPC accounts before each round of consolidation and public push. The authors are transitioning toward more incremental public development as the project matures and its external user base grows.

## References

Breiman, L. (2001). Random forests. *Machine Learning*, *45*(1), 5–32. https://doi.org/10.1023/A:1010933404324

Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 785–794. https://doi.org/10.1145/2939672.2939785

Chowdhury, S. R., Radhakrishnan, A., & Subramoni, H. (2026). From bands to depth: Understanding bathymetry decisions on Sentinel-2. *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2063–2072. https://doi.org/10.1109/WACV61042.2026.00205

Drusch, M., Del Bello, U., Carlier, S., Colin, O., Fernandez, V., Gascon, F., Hoersch, B., Isola, C., Laberinti, P., Martimort, P., Meygret, A., Spoto, F., Sy, O., Marchese, F., & Bargellini, P. (2012). Sentinel-2: ESA's optical high-resolution mission for GMES operational services. *Remote Sensing of Environment*, *120*, 25–36. https://doi.org/10.1016/j.rse.2011.11.026

Gillies, S., & others. (2023). *Rasterio: geospatial raster I/O for Python programmers*. https://github.com/rasterio/rasterio.

Hoyer, S., & Hamman, J. (2017). xarray: N-D labeled arrays and datasets in Python. *Journal of Open Research Software*, *5*(1), 10. https://doi.org/10.5334/jors.148

Hsu, H.-J., & Moortgat, J. (2026). From local training to large-scale mapping: A comparative

assessment of machine learning and deep learning for transferable satellite-derived bathymetry. *Remote Sensing*, *18*(11), 1768. https://doi.org/10.3390/rs18111768

McFeeters, S. K. (1996). The use of the Normalized Difference Water Index (NDWI) in the delineation of open water features. *International Journal of Remote Sensing*, *17*(7), 1425–1432. https://doi.org/10.1080/01431169608948714

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional networks for biomedical image segmentation. *Medical Image Computing and Computer-Assisted Intervention – MICCAI*, 234–241. https://doi.org/10.1007/978-3-319-24574-4_28

Roy, D. P., Wulder, M. A., Loveland, T. R., Woodcock, C. E., Allen, R. G., Anderson, M. C., Helder, D., Irons, J. R., Johnson, D. M., Kennedy, R., Scambos, T. A., Schaaf, C. B., Schott, J. R., Sheng, Y., Vermote, E. F., Belward, A. S., Bindschadler, R., Cohen, W. B., Gao, F., … Zhu, Z. (2014). Landsat-8: Science and product vision for terrestrial global change research. *Remote Sensing of Environment*, *145*, 154–172. https://doi.org/10.1016/j.rse.2014.02.001

STAC Community. (2024). *SpatioTemporal Asset Catalog (STAC) Specification*. https://stacspec.org/.

Stewart, A. J., Robinson, C., Corley, I. A., Ortiz, A., Lavista Ferres, J. M., & Banerjee, A. (2022). TorchGeo: Deep Learning With Geospatial Data. *Proceedings of the 30th International Conference on Advances in Geographic Information Systems*. https://doi.org/10.1145/3557915.3560953

Torres, R., Snoeij, P., Geudtner, D., Bibby, D., Davidson, M., Attema, E., Potin, P., Rommen, B., Floury, N., Brown, M., Traver, I. N., Deghaye, P., Duesmann, B., Rosich, B., Miranda, N., Bruno, C., L'Abbate, M., Croci, R., Pietropaolo, A., … Rostan, F. (2012). GMES Sentinel-1 mission. *Remote Sensing of Environment*, *120*, 9–24. https://doi.org/10.1016/j.rse.2011.05.028

Wu, Q. (2026). GeoAI: A Python package for integrating artificial intelligence with geospatial data analysis and visualization. *Journal of Open Source Software*, *11*(118), 9605. https://doi.org/10.21105/joss.09605

Zanaga, D., Van De Kerchove, R., Daems, D., De Keersmaecker, W., Brockmann, C., Kirches, G., Wevers, J., Cartus, O., Santoro, M., Fritz, S., Lesiv, M., Herold, M., Tsendbazar, N.-E., Xu, P., Ramoino, F., & Arino, O. (2022). *ESA WorldCover 10 m 2021 v200*. https://doi.org/10.5281/zenodo.7254221