SRAI: Towards Standardization of Geospatial AI

Piotr Gramacki, Kacper Leśniara, Kamil Raczycki, Szymon Woźniak, Marcin Przymus, Piotr Szymański

Kraina.AI Research Team
Department of Artificial Intelligence
Wrocław University of Science and Technology

Wrocław University of Science and Technology

GeoAI@SIGSPATIAL 2023

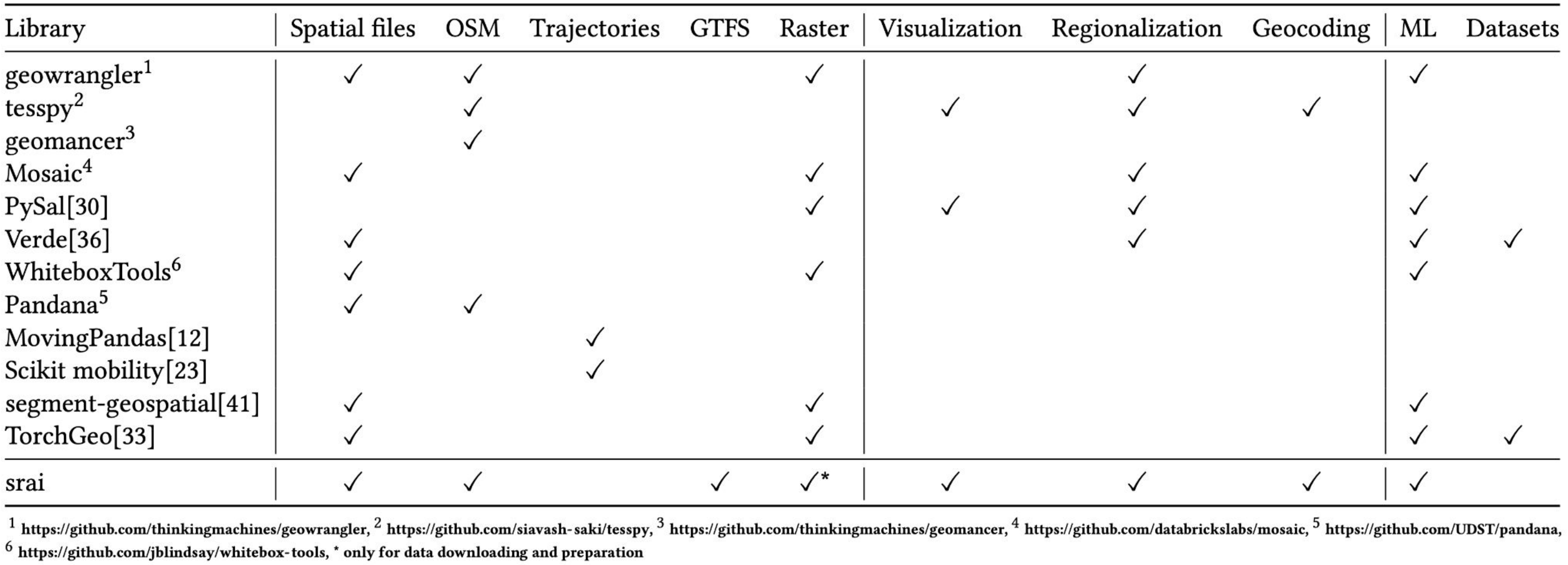
Kraina AI

srai

The need for Standardization in Geospatial AI

- a *Hugging Face*-like hub
- reproducibility in GeoAI
- encouraging sharing of data and code
- uniform interface/pipeline for GeoAI
- lack of established benchmarks
- easier access to openly available geospatial data

Positioning of SRAI

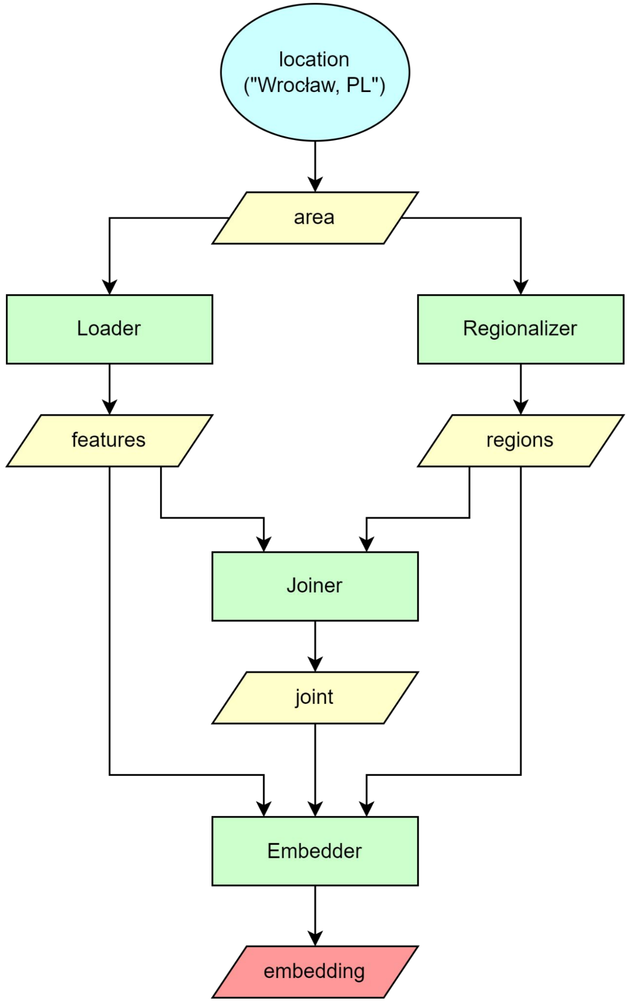


Library	Spatial files	OSM	Trajectories	GTFS	Raster	Visualization	Regionalization	Geocoding	ML	Datasets
geowrangler[1]	✓	✓			✓		✓		✓	
tesspy[2]		✓				✓	✓	✓		
geomancer[3]		✓								
Mosaic[4]	✓				✓		✓		✓	
PySal[30]					✓	✓	✓		✓	
Verde[36]	✓						✓		✓	✓
WhiteboxTools[6]	✓				✓				✓	
Pandana[5]	✓	✓								
MovingPandas[12]			✓							
Scikit mobility[23]			✓							
segment-geospatial[41]	✓				✓				✓	
TorchGeo[33]	✓				✓				✓	✓
srai	✓	✓		✓	✓*	✓	✓	✓	✓	

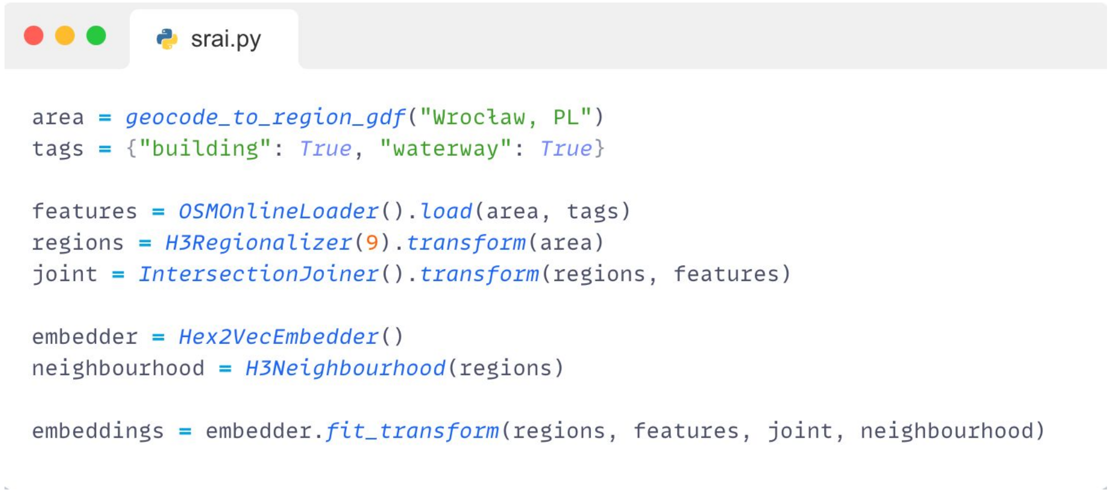
[1] https://github.com/thinkingmachines/geowrangler, [2] https://github.com/siavash-saki/tesspy, [3] https://github.com/thinkingmachines/geomancer, [4] https://github.com/databrickslabs/mosaic, [5] https://github.com/UDST/pandana, [6] https://github.com/jblindsay/whitebox-tools, * only for data downloading and preparation

Spatial Representations for Artificial Intelligence

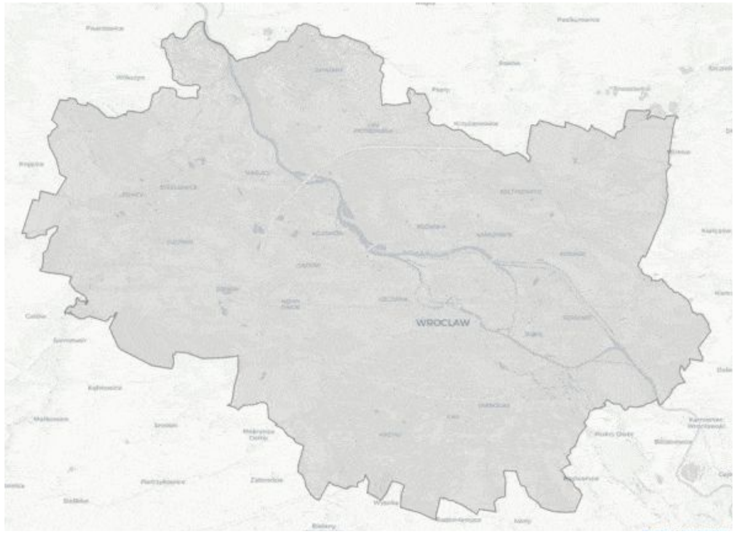
1. Geospatial data loading
   - OSM
   - other sources
2. Regionalization / tessellation
   - Spatial indices
   - Voronoi
   - Administrative
3. Embedding
   - Feature counts
   - Generic OSM based
   - Road segments
   - Public transport



Usage example

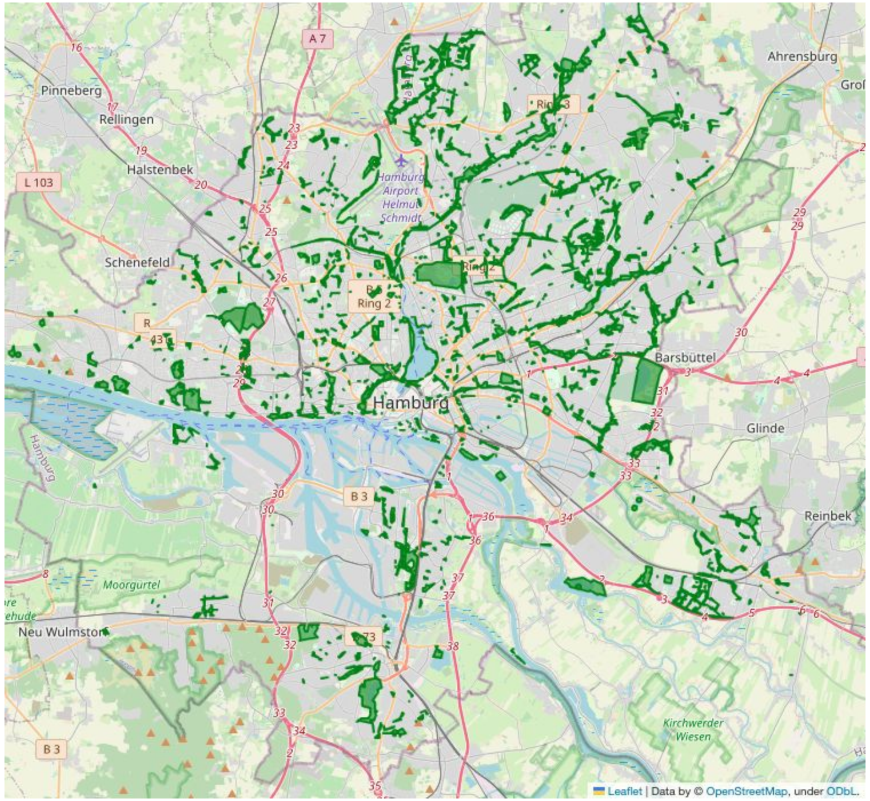

srai.py

```python
area = geocode_to_region_gdf("Wrocław, PL")
tags = {"building": True, "waterway": True}

features = OSMOnlineLoader().load(area, tags)
regions = H3Regionalizer(9).transform(area)
joint = IntersectionJoiner().transform(regions, features)

embedder = Hex2VecEmbedder()
neighbourhood = H3Neighbourhood(regions)

embeddings = embedder.fit_transform(regions, features, joint, neighbourhood)
```



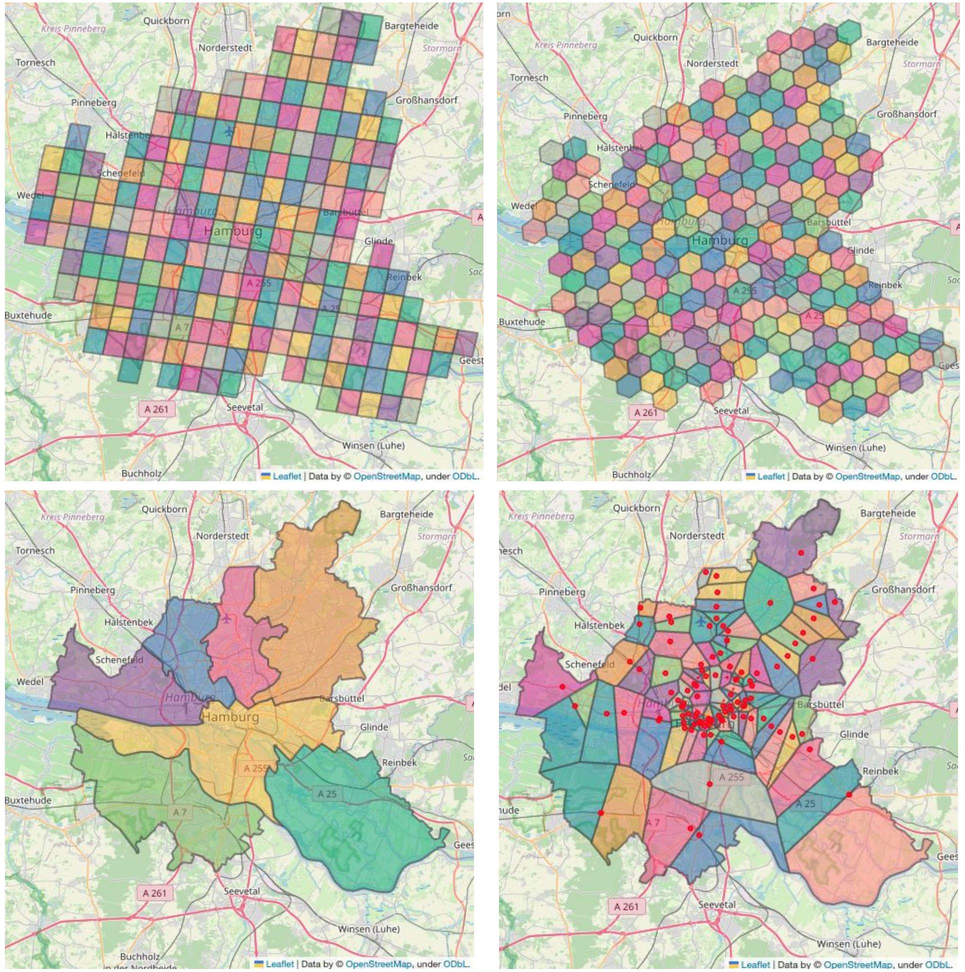
Loaders

1. OpenStreetMap
   - Tags & geometries
     - Overpass
     - Protomaps
     - Geofabrik & OSM.fr
   - Road networks
   - Map tiles
2. GTFS
   - public transport availability



Regionalizers

1. Spatial indices
   - Uber’s H3
   - Google’s S2
2. Data driven regions
   - Voronoi cells
3. Existing regions
   - OSM based
   - Admin levels



Embedders

1. Baseline embedders
   - Feature counts
   - Contextualized feature counts - ARIC’21
2. Trainable models
   - Hex2Vec - GeoAI’21
   - GTFS2Vec - GeoSearch’21
   - Highway2Vex - GeoAI’22
   - GeoVex - GeoAI’23 (by Daniele Donghi & Anne Morvan, impl. Max Schrader)

Future works

1. Pre-trained models and pre-calculated embeddings hosting
2. Datasets and Benchmarks
3. Fine-tuning interface
4. Multimodal approach - Computer Vision and Graph Embeddings
5. Next-gen Geospatial Representation Learning Models
6. Support for out-of-core scenarios (DuckDB or Dask-GeoPandas)
7. ...

Q&A

Library: github.com/kraina-ai/srai

Website: kraina.ai