OPERA x Google Earth Engine

Daniel Coelho
9/11/2025

New Additions to Earth Engine's Data Catalog

We are in the process of adding the following datasets:

- Dynamic Surface Water Extent (HLS & S1)
- Radiometric Terrain Corrected Sentinel 1

The datasets will be available directly through Earth Engine and directly through Cloud-Optimized-Geotiffs

Quick Demo of Dynamic Surface Water Extent

Preview

```
var dswx = ee.ImageCollection("OPERA/DSWX/L3_V1/hls");
```

https://code.earthengine.google.com/8d2067bb2b157038cf6bcc172f0ab8c7

Day 2 PM

AUG 26

You are here

1:00 PM	Lunch (and T-shirt distribution!)
2:15 PM 75 MINUTES	PITCH PINE Effective use of Google's Satellite Embedding dataset Overflow rooms in Sassafras and Hackberry
3:30 PM	Coffee Break
4:00 PM 60 MINUTES	PITCH PINE Tipping the Scale: Heavy Lifting with Earth Engine SASSAFRAS Co-create: The Future of Earth x Earth Engine Workflows

2025 GEO FOR GOOD — NEW YORK CITY —

goo.gle/g4g-nyc-satellite-embedding-dataset

Effective use of Google's Satellite Embedding dataset

Valerie Pasquarella & **Emily Schechter**

August 2025 | #GeoForGood25

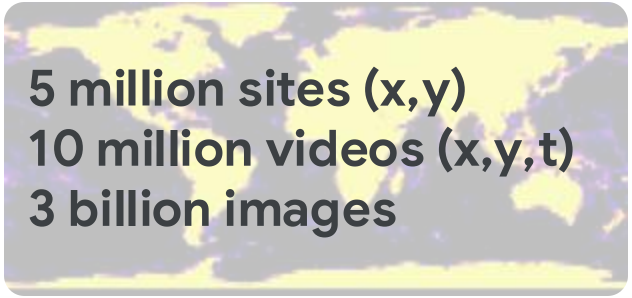
Biplov Bhandari, Woolpert Digital Innovations
Kyle Woodward, Spatial Informatics Group
Angela Tsao, Stanford University
Álvaro Moreno Martínez, University of Valencia/IPL
Emma Izquierdo, Institute of Geomatics at BOKU
Mina Burns, Oregon State University

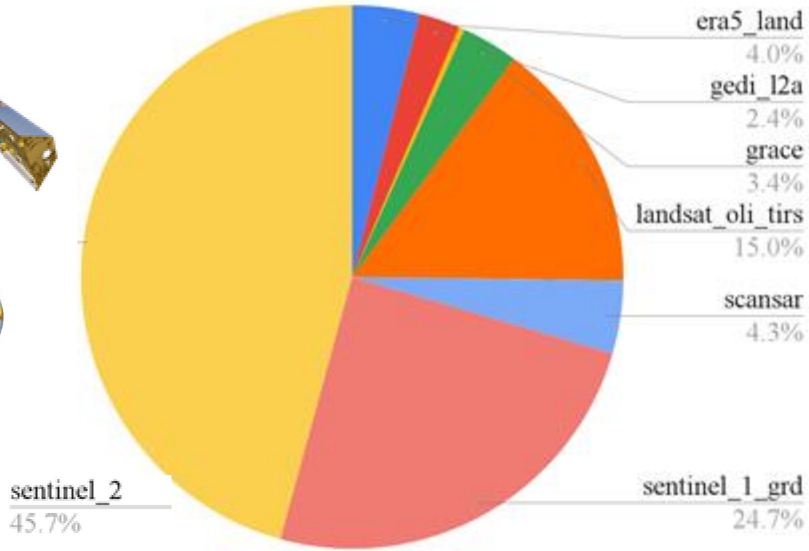
What's embedded in an embedding?

Training sources

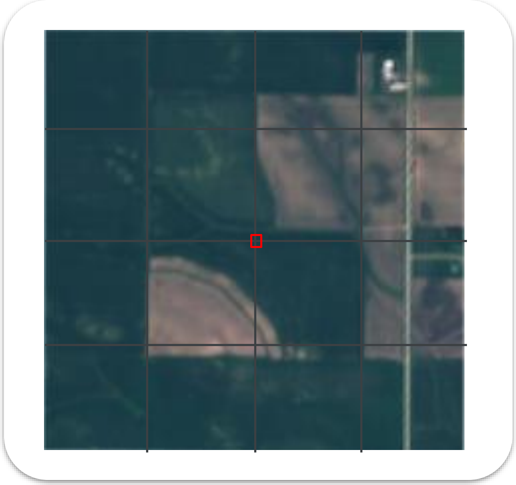
Type	Dataset	Product	Bands	Resolution (m)	Usage
Optical	Sentinel-2	L1C	B2 (Blue), B3 (Green), B4 (Red), B8 (NIR), B11 (SWIR)	10, 20, 60	input, target
Optical, Thermal	Landsat-8, Landsat-9	L1C	B2 (Blue), B3 (Green), B4 (Red), B5 (NIR), B6 (SWIR), B8 (Panchromatic), B10 (Thermal)	15, 30, 100	input, target
C-band SAR	Sentinel-1A, Sentinel-1B	GRD	VV, VH, HH, HV, angle	10	input, target
L-band SAR	ALOS PALSAR ScanSAR	Level 2.2	HH, HV, lin	25	target
Elevation	Copernicus DEM	GLO-30	DEM (elevation)	30	target
LiDAR	GEDI	L2A	Relative height metrics (rh*)	25	target
Climate	ERA5-Land	Monthly aggregates	total precipitation (sum, min, max), air temperature 2m (and min, max), dew-point temperature 2m (and min, max), surface pressure (and min, max)	11132	target
Gravity fields	GRACE	Monthly mass grids	equivalent liquid water thickness	11132	target (@50%)
Land cover	National Land Cover Database	NLCD 2019, 2021	landcover	N/A	target (@50%)
Text	Wikipedia	geocoded articles	text embeddings	N/A	target
Text	GBIF	Research-grade obs	text embeddings (class, genus, and species)	N/A	target

* **V2.1 updates:** Also includes NASS CDL landcover (target), NLCD & CDL weights reduced from 50% to 25%

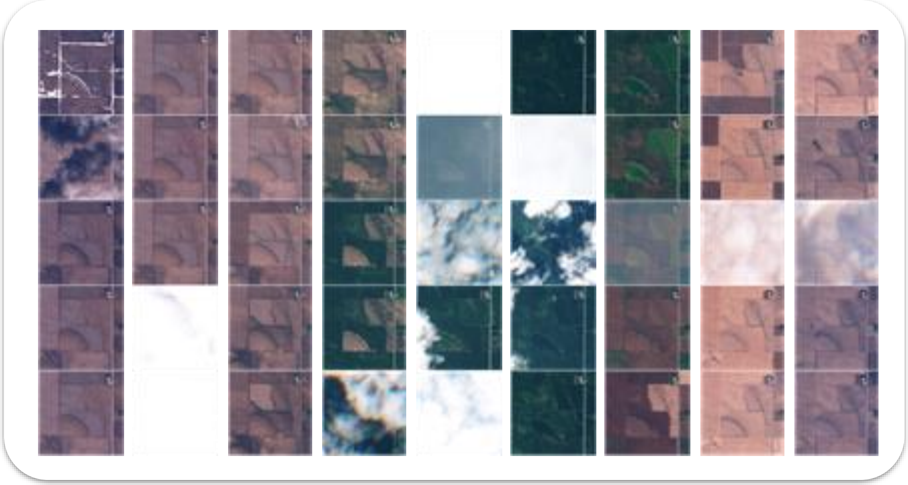




Spatial & temporal context



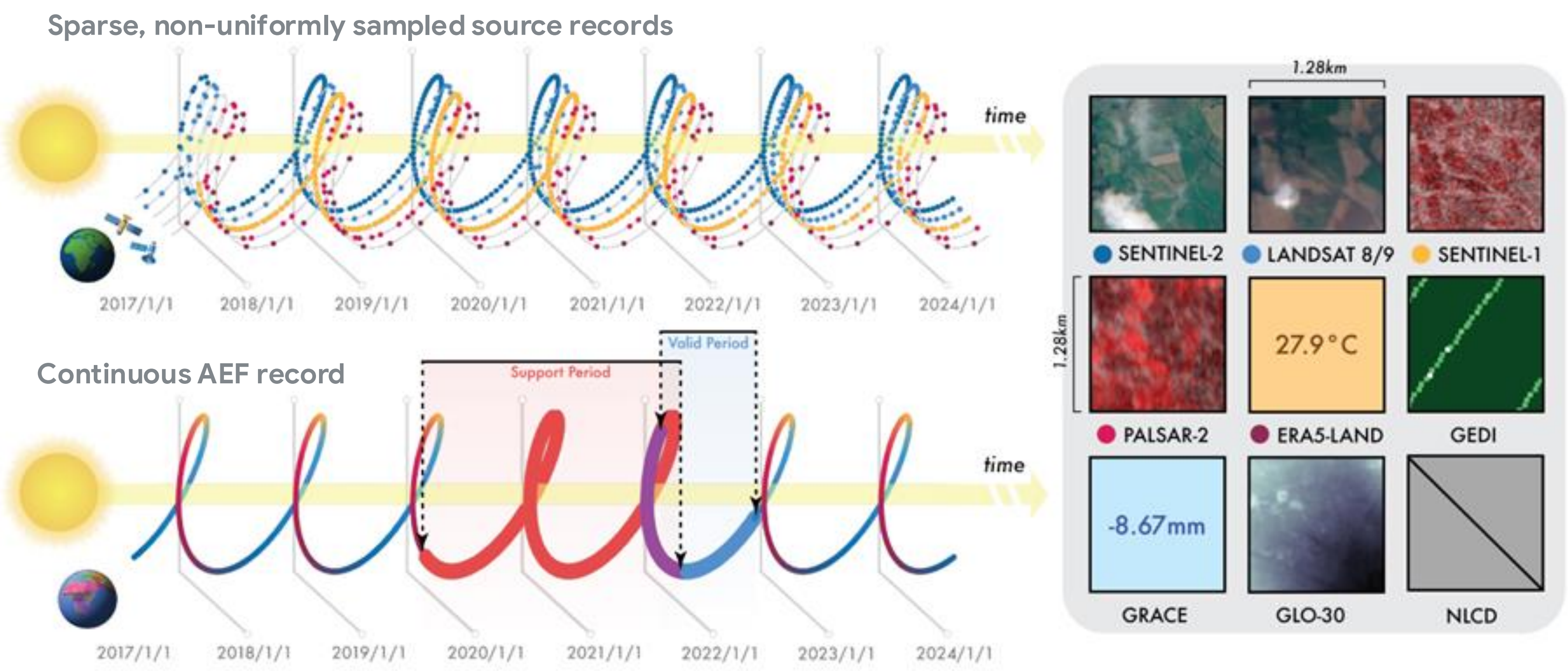
1.28 km x 1.28 km input image



Full calendar year of imagery (per sensor)

Model can use any information in this window when generating embedding for target pixel

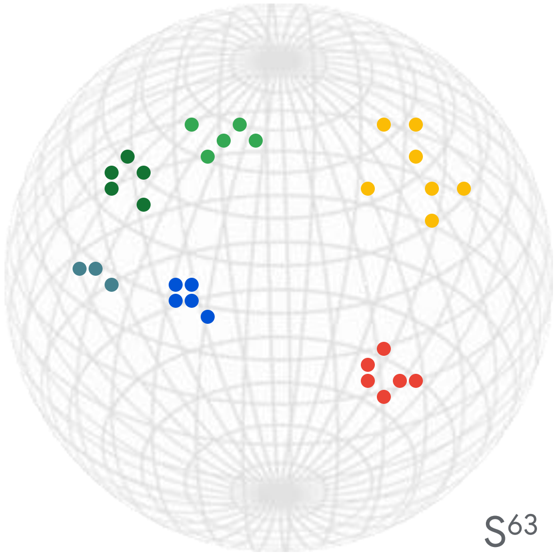
Model learns from a sequence of image frames (video) to encode a temporal trajectory



A virtual satellite that can image any place over any time

Building a better embedding...

- **Reconstruction objective**: Reconstruct randomly selected frames, using their corresponding metadata and timecode
- **Batch uniformity objective:** Encourage a uniform distribution over the training set of embeddings in a 64-dimensional unit sphere (S^{63})
- **Contrastive consistency objective**: Encourage forward passes with missing inputs to create embeddings identical to forward passes without missing inputs
- **Text contrastive objective**: Align embeddings derived from text descriptions with embeddings derived from the video sequence.



= well-structured 64-dimensional unit-length embedding fields

Google

Things we don't do...

- We don't **include location as a predictor** → location is more of an index than a signal
- We don't **require surface reflectance or cloud masks** to generate embeddings → we want to learn in a way that ignores sensor minutia and noise
- We don't **require re-training** → we want embedding outputs to be useful as-is
- We don't **assume reconstruction is enough** → we use additional constraints to help structure the feature space

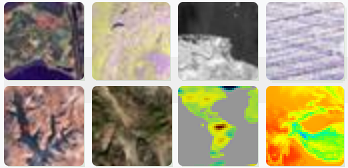
AlphaEarth Foundations

- A model designed to function as a **virtual satellite** whose images summarize Earth's surface dynamics over space and time
- Learn **meaningful patterns** across data streams and generate spatially and temporally precise embedding vectors for any place, any time
- Outputs designed for **practitioners** – like an AI-powered image composite
- **Embedding images** are a foundation on which many different EO applications can be built

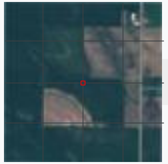
Read more in our preprint here: arxiv.org/abs/2507.22291

Satellite Embedding Dataset

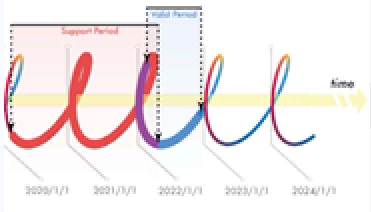
Annual summaries generated by AlphaEarth Foundations

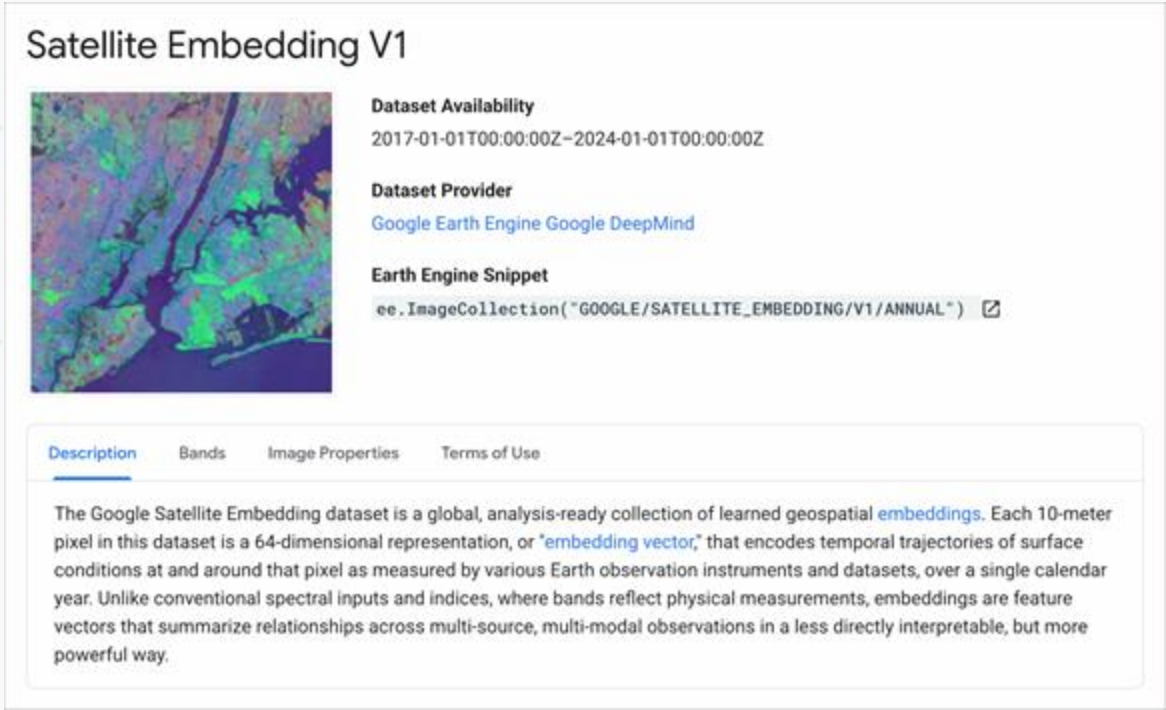


measurements



space





Satellite Embedding V1

Dataset Availability

2017-01-01T00:00:00Z–2024-01-01T00:00:00Z

Dataset Provider

Google Earth Engine Google DeepMind

Earth Engine Snippet

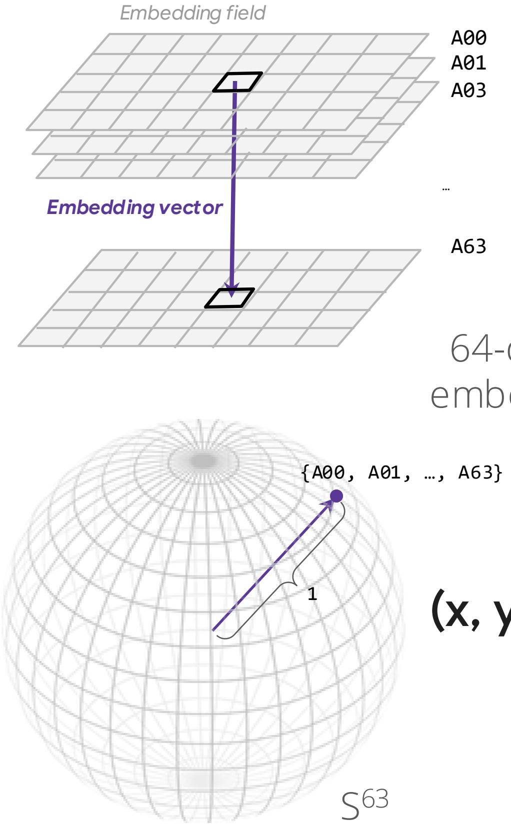
`ee.ImageCollection("GOOGLE/SATELLITE_EMBEDDING/V1/ANNUAL")`

Description | Bands | Image Properties | Terms of Use

The Google Satellite Embedding dataset is a global, analysis-ready collection of learned geospatial embeddings. Each 10-meter pixel in this dataset is a 64-dimensional representation, or "embedding vector," that encodes temporal trajectories of surface conditions at and around that pixel as measured by various Earth observation instruments and datasets, over a single calendar year. Unlike conventional spectral inputs and indices, where bands reflect physical measurements, embeddings are feature vectors that summarize relationships across multi-source, multi-modal observations in a less directly interpretable, but more powerful way.

```
ee.ImageCollection("GOOGLE/SATELLITE_EMBEDDING/V1/ANNUAL")
```

Embedding fields



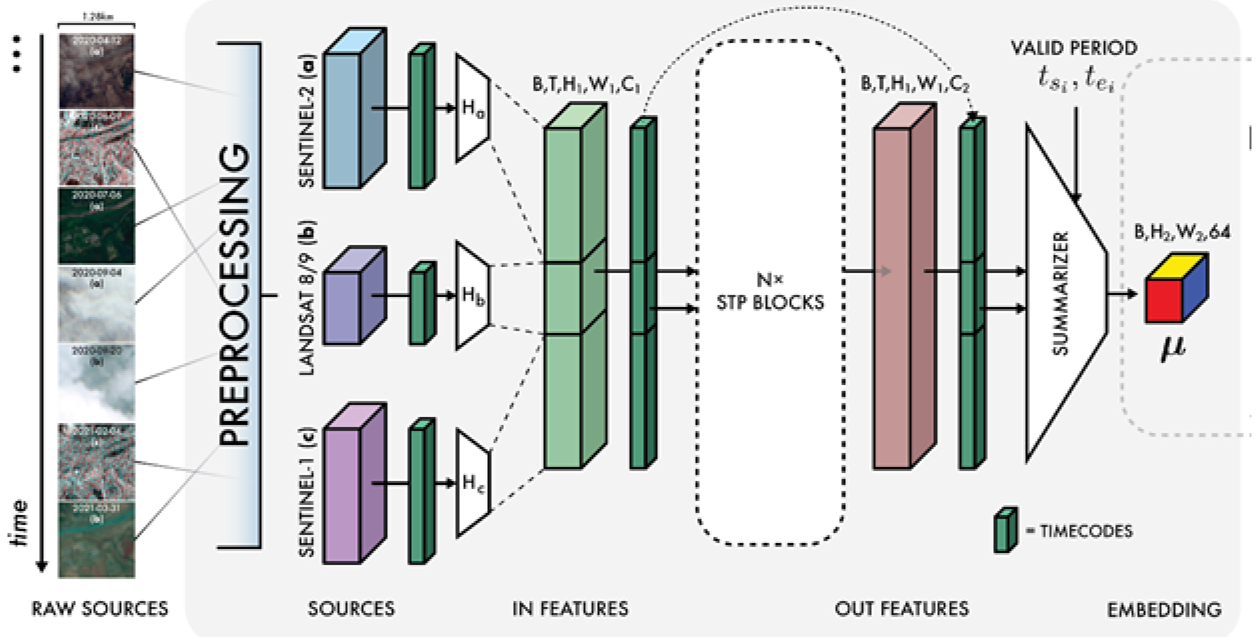
64-dimensional embedding space

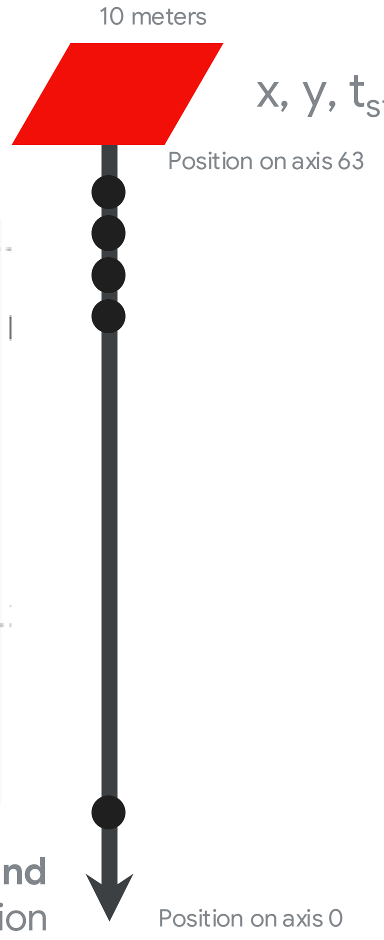
$(x, y, t_{start}, t_{end})$

10-meter resolution

Google

Learned axes != wavelengths





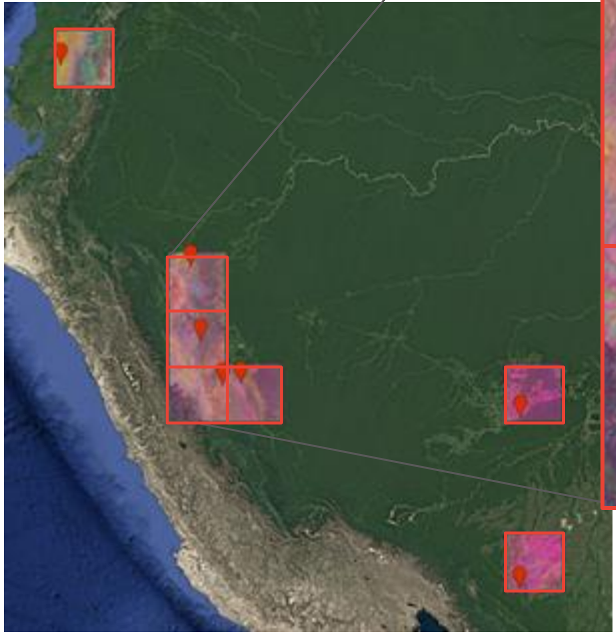
x, y, t_{start}, t_{end}

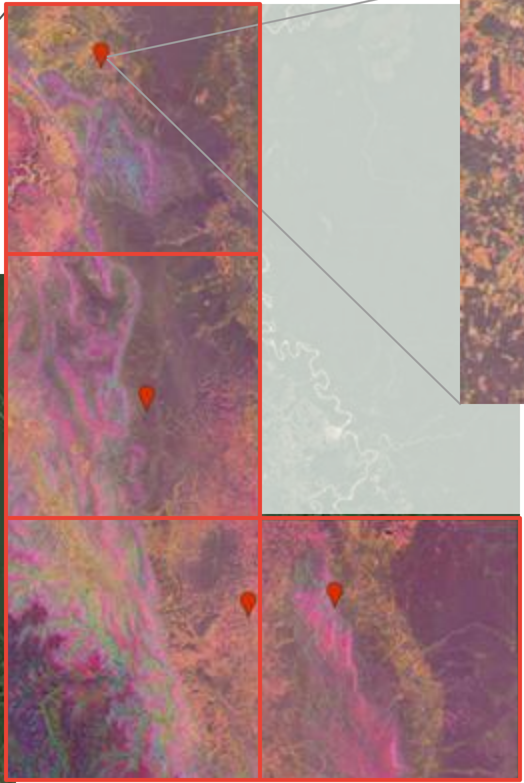
Annual summary of **surface condition and temporal variability** for given location

Google

Image Collection

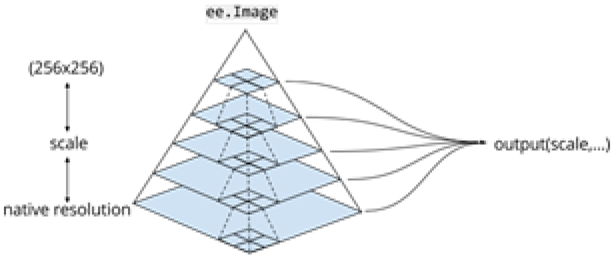
Satellite Embedding
Image Collection in EE





Tiled embedding fields (wall-to-wall terrestrial + shallow water coverage)

Per-pixel embeddings (10m x 10m)



Linearly composable = pyramiding

What can you do with the Satellite Embedding Dataset?

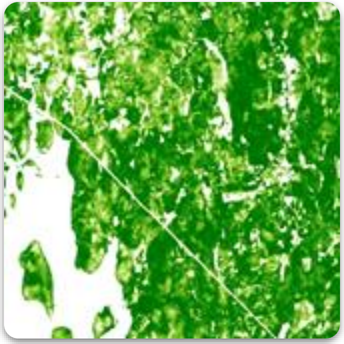
Similarity Search

(40.723, -74.000)

Classification

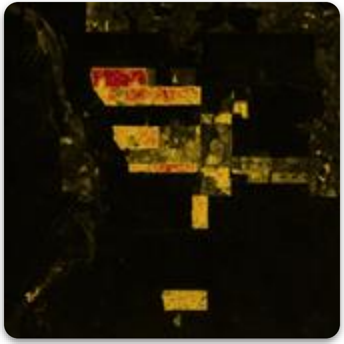
Crop type

Regression



Above ground biomass

Change detection



Forest clearing

Instantly find other places like this

Accurate maps with less training data

Scale biophysical measurements

Easily spot changes & track processes

What can you do with the Satellite Embedding Dataset?

Similarity Search

(40.723, -74.000)

Instantly find other places like this

Classification

Crop type

Accurate maps with less training data

Regression

Above ground biomass

Scale biophysical measurements

Change detection

Forest clearing

Easily spot changes & track processes

Google

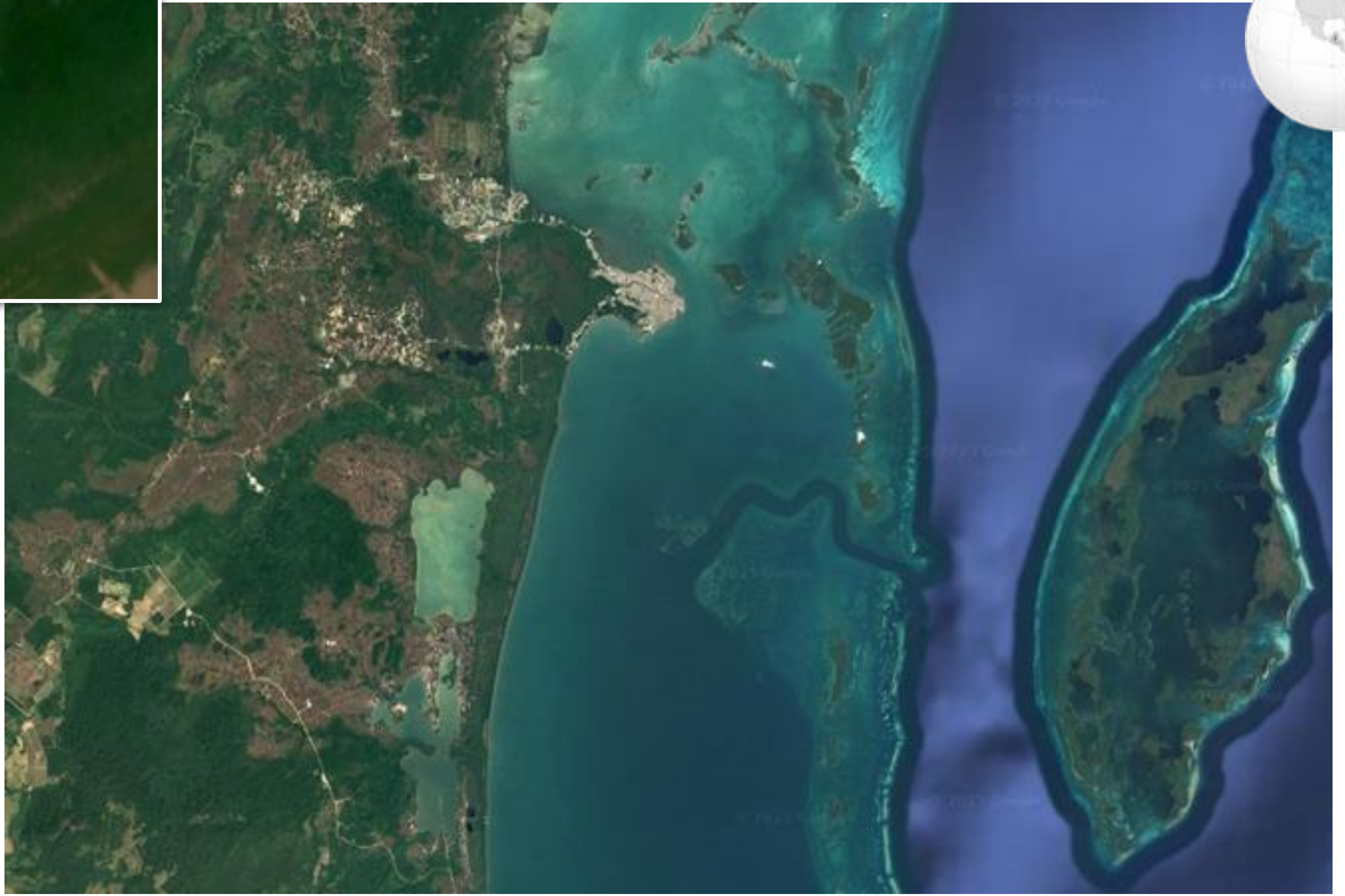
lon=106.2539
lat=9.5009

Mekong delta, Vietnam

Mekong delta, Vietnam

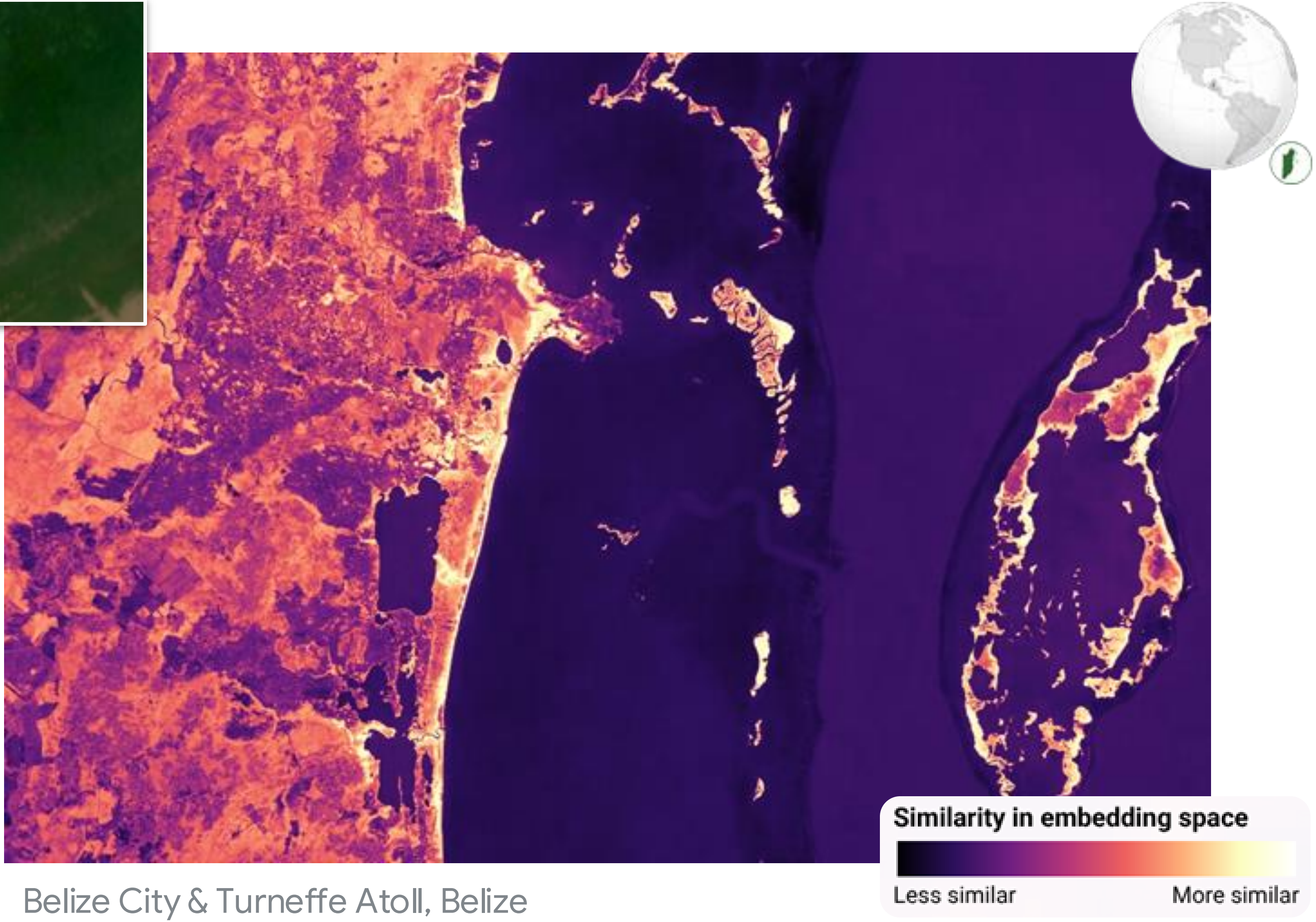
Google

lon=106.2539
lat=9.5009



Belize City & Turneffe Atoll, Belize

lon=106.2539
lat=9.5009

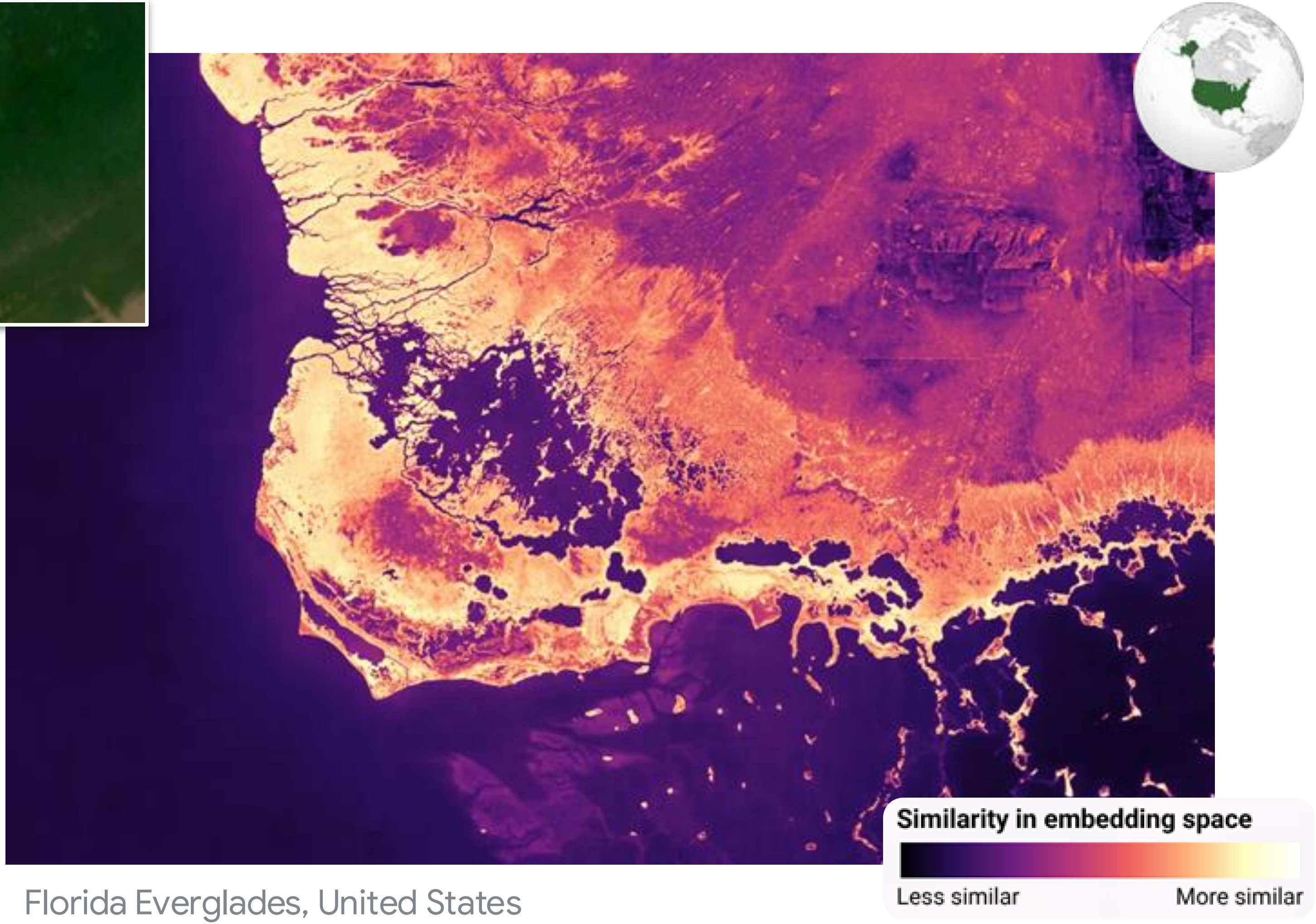


Belize City & Turneffe Atoll, Belize

lon=106.2539
lat=9.5009

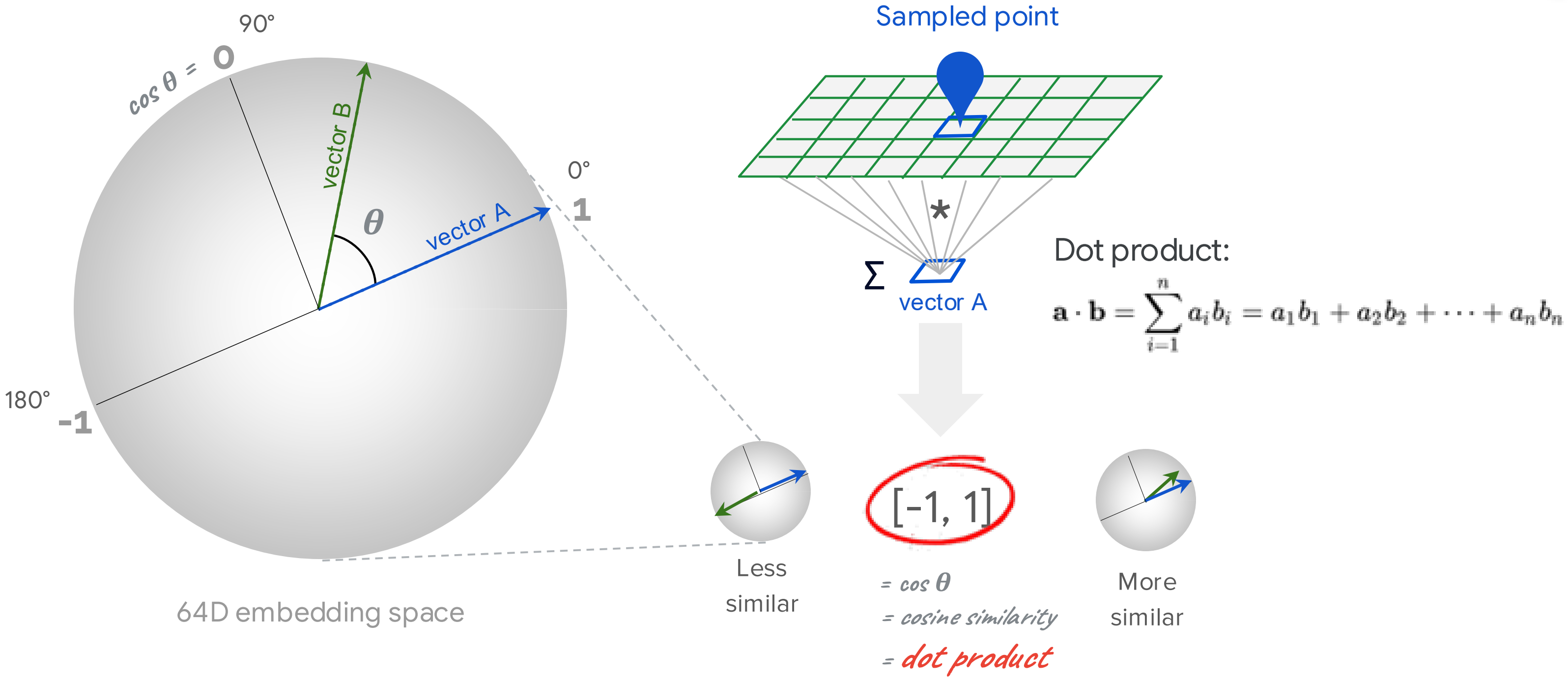
Florida Everglades, United States

lon=106.2539
lat=9.5009



Florida Everglades, United States

Under the hood: Dot product similarity

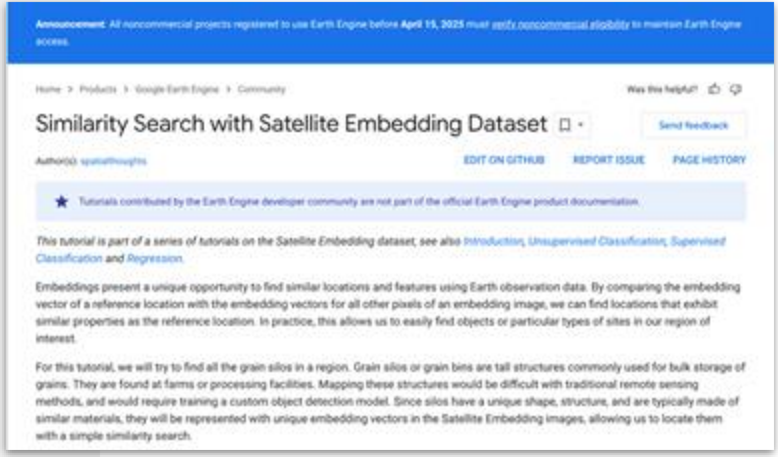


Under the hood: Dot product similarity

```
var point = ee.Geometry.Point([-73.98119128411304, 40.76277998772318]);

var embeddings = ee.ImageCollection('GOOGLE/SATELLITE_EMBEDDING/V1/ANNUAL');
var year = 2024;

var mosaic = embeddings
    .filterDate(year + '-01-01', (year + 1) + '-01-01')
    .mosaic();

var bandNames = mosaic.bandNames();

var similarity = ee.ImageCollection(
        mosaic.sample({region: point, scale: 10}).map(function(f) {
          return ee.Image(f.toArray(bandNames))
              .arrayFlatten(ee.List([bandNames]))
              .multiply(mosaic)
              .reduce('sum');
        })).first();
```

https://code.earthengine.google.com/9b83b11c269ad0a9f0590fd9e2a43dfc

Check out our new tutorial for more!



Similarity Search with Satellite Embedding Dataset

EDIT ON GITHUB REPORT ISSUE PAGE HISTORY

Tutorials contributed by the Earth Engine developer community are not part of the official Earth Engine product documentation.

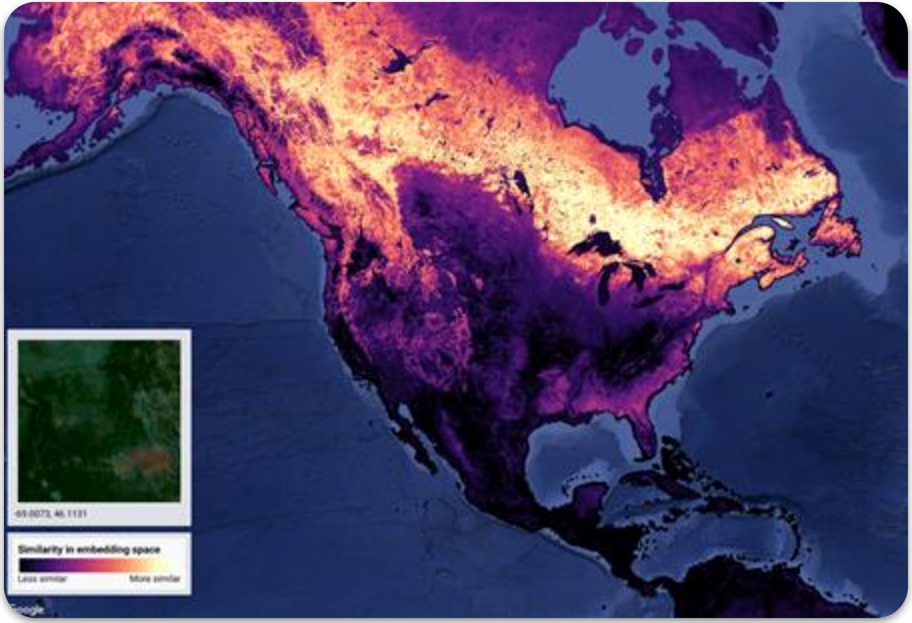
This tutorial is part of a series of tutorials on the Satellite Embedding dataset, see also Introduction, Unsupervised Classification, Supervised Classification *and* Regression.

Embeddings present a unique opportunity to find similar locations and features using Earth observation data. By comparing the embedding vector of a reference location with the embedding vectors for all other pixels of an embedding image, we can find locations that exhibit similar properties as the reference location. In practice, this allows us to easily find objects or particular types of sites in our region of interest.

For this tutorial, we will try to find all the grain silos in a region. Grain silos or grain bins are tall structures commonly used for bulk storage of grains. They are found at farms or processing facilities. Mapping these structures would be difficult with traditional remote sensing methods, and would require training a custom object detection model. Since silos have a unique shape, structure, and are typically made of similar materials, they will be represented with unique embedding vectors in the Satellite Embedding images, allowing us to locate them with a simple similarity search.

Summary: Similarity Search

- Compare embedding vector for a sampled location and every other location @ 10-meter resolution
- Generate a map of similarity scores on-demand
- Sample multiple points to define semantics
- Resample (average) embeddings to aggregate to patch scales (using any rules you like!)



Similar to protected forest in Maine, USA

Try it yourself!

goo.gle/satellite-embedding-similarity-demo

What can you do with the Satellite Embedding Dataset?

Similarity Search

(40.723, -74.000)

Instantly find other places like this

Classification

Crop type

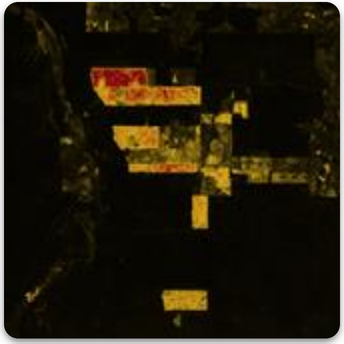
Accurate maps with less training data

Regression

Above ground biomass

Scale biophysical measurements

Change detection

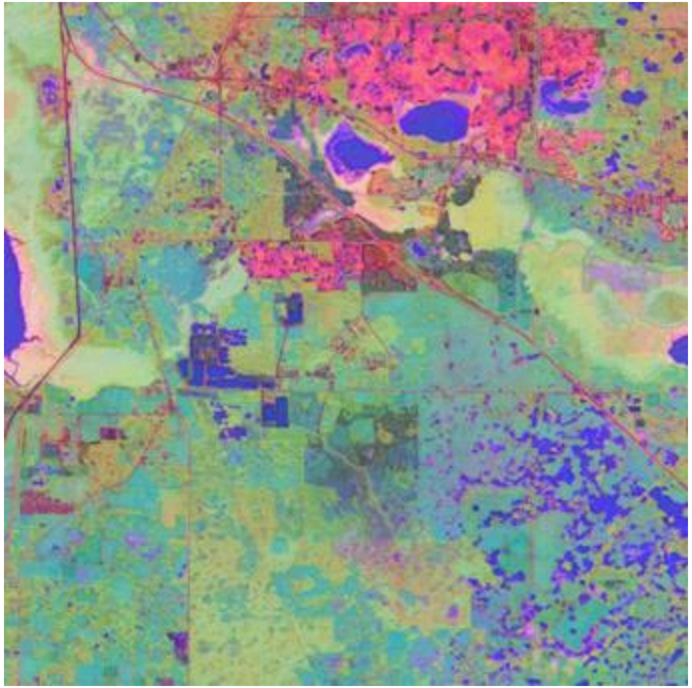


Forest clearing

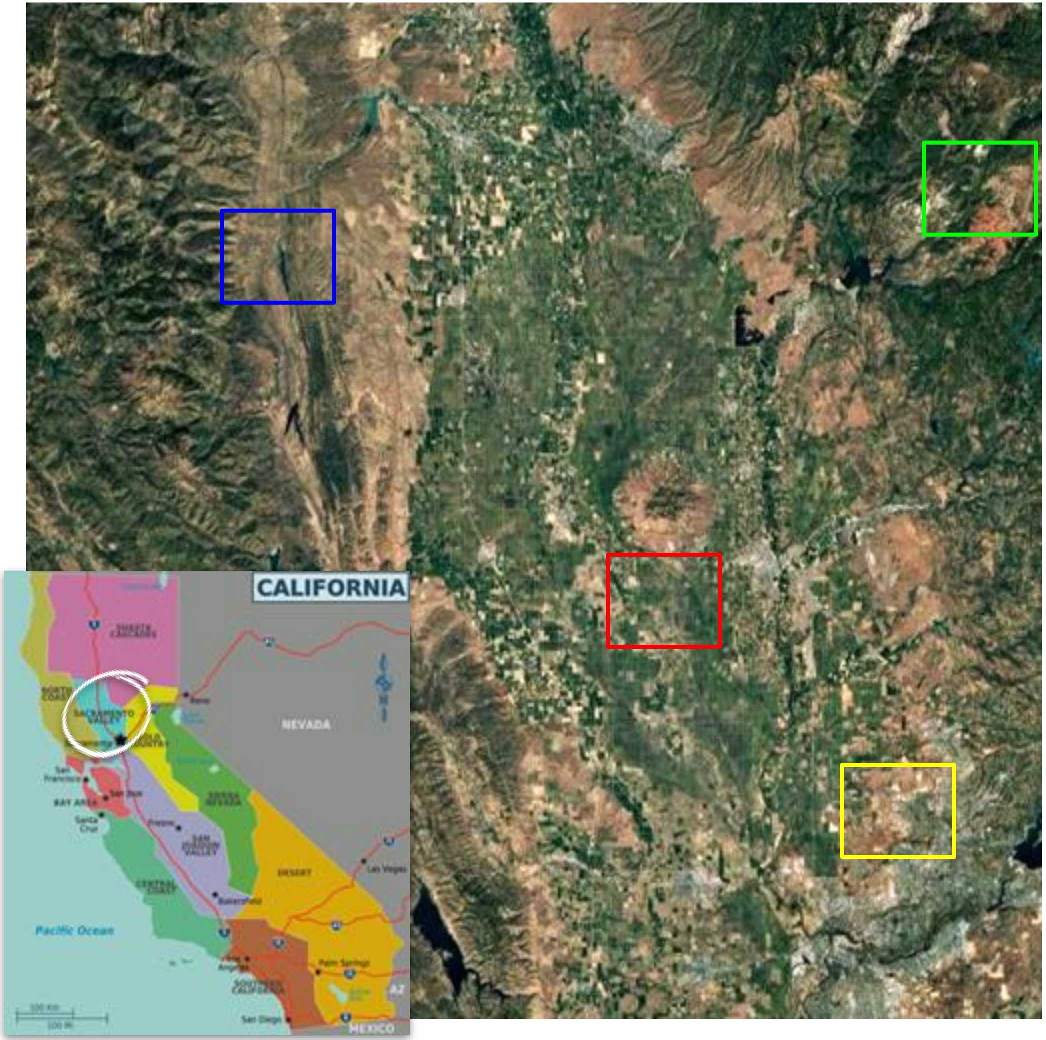
Easily spot changes & track processes

Similarity in time = stability

- **Compare** embedding vector for a sampled location with vector for that location for a different year
- Use **similarity** as a measure of relative **stability/instability**
- **Consistency objective** incentivizes overall stability, however **individual axes** may (jointly) encode responses in different properties
- Change can mean "**surface change**" or it could be "**environmental change**", i.e., wet year versus dry year



Time series of embeddings showing new construction in Florida, USA

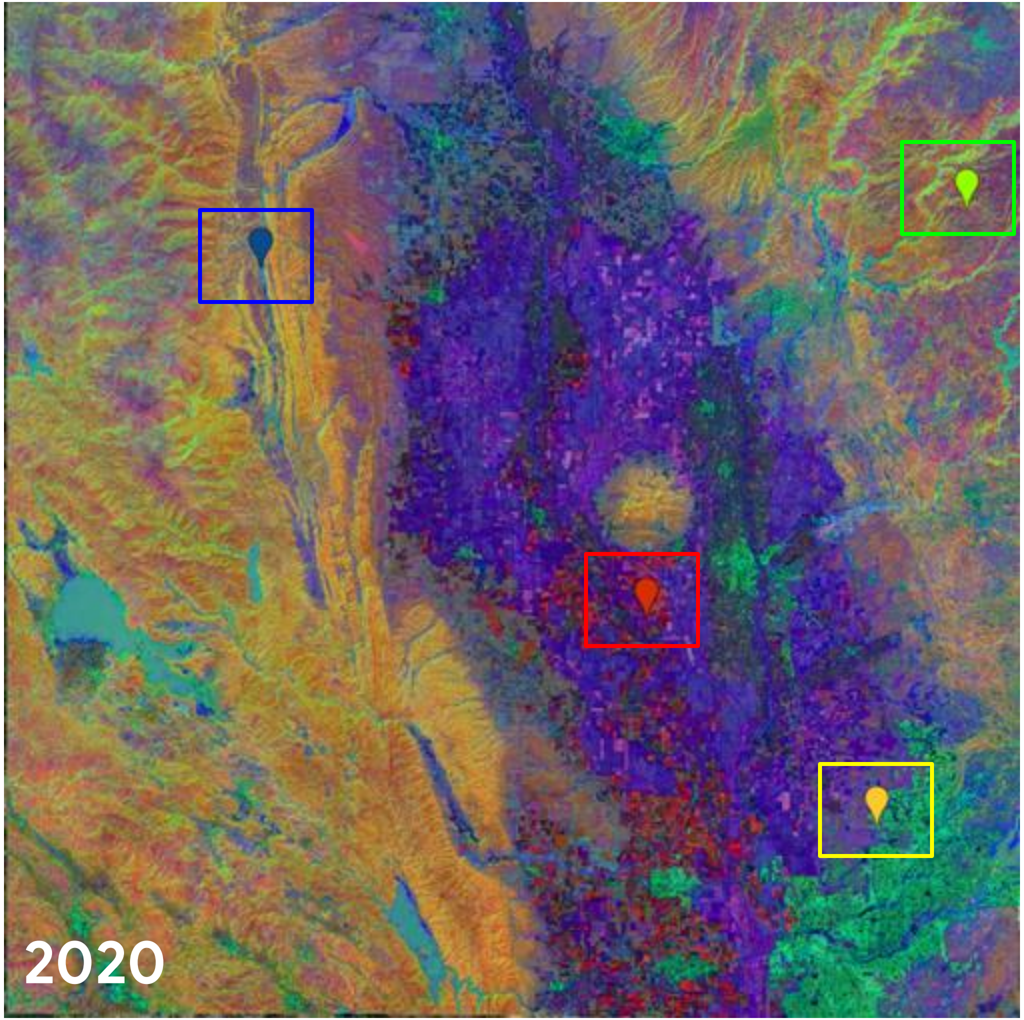


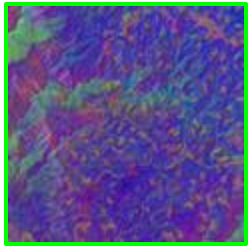
Forests fires
& Timber

Water
resources

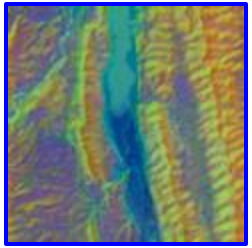
Agricultural
monitoring

Suburban
development

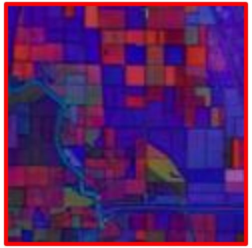




Forests fires
& Timber

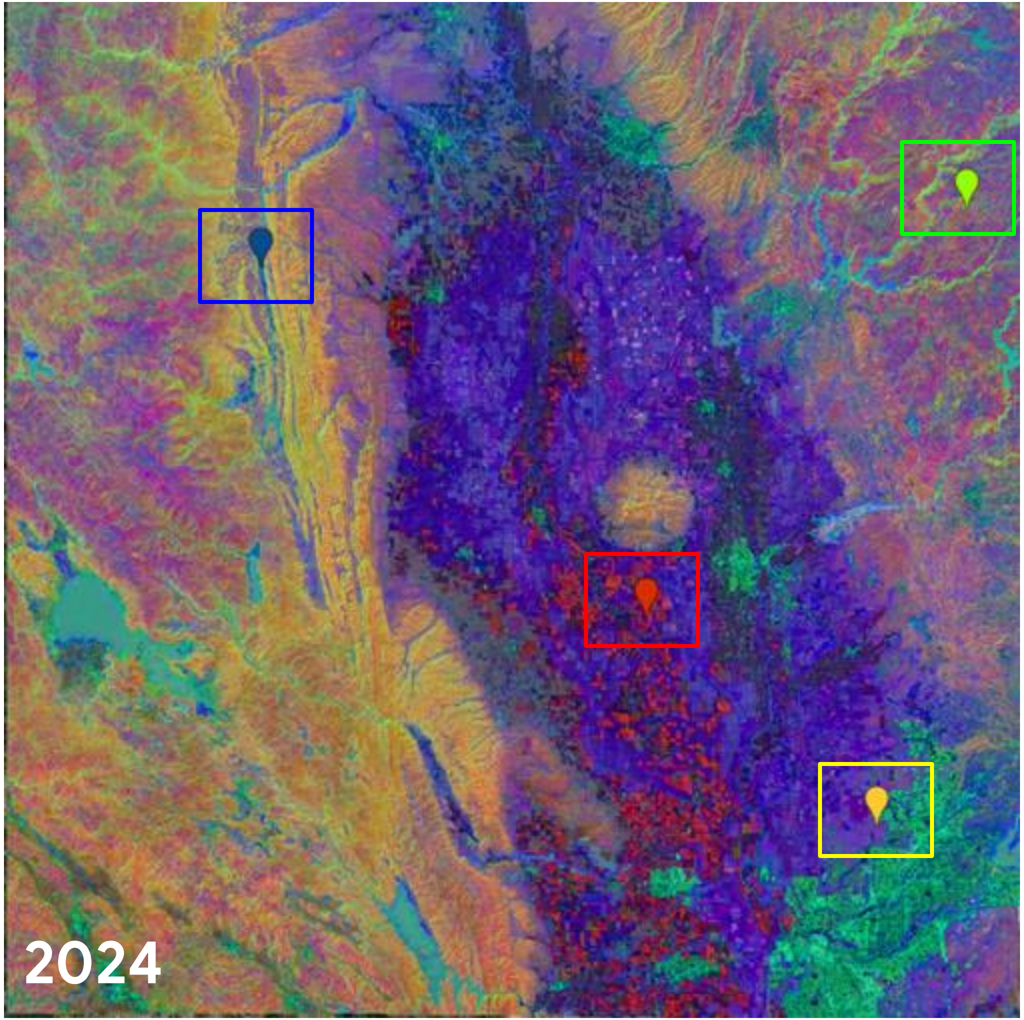


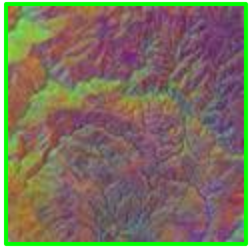
Water
resources



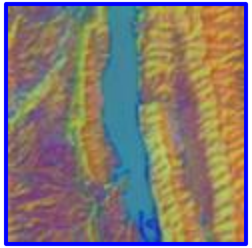
Agricultural
monitoring

Suburban
development





Forests fires
& Timber

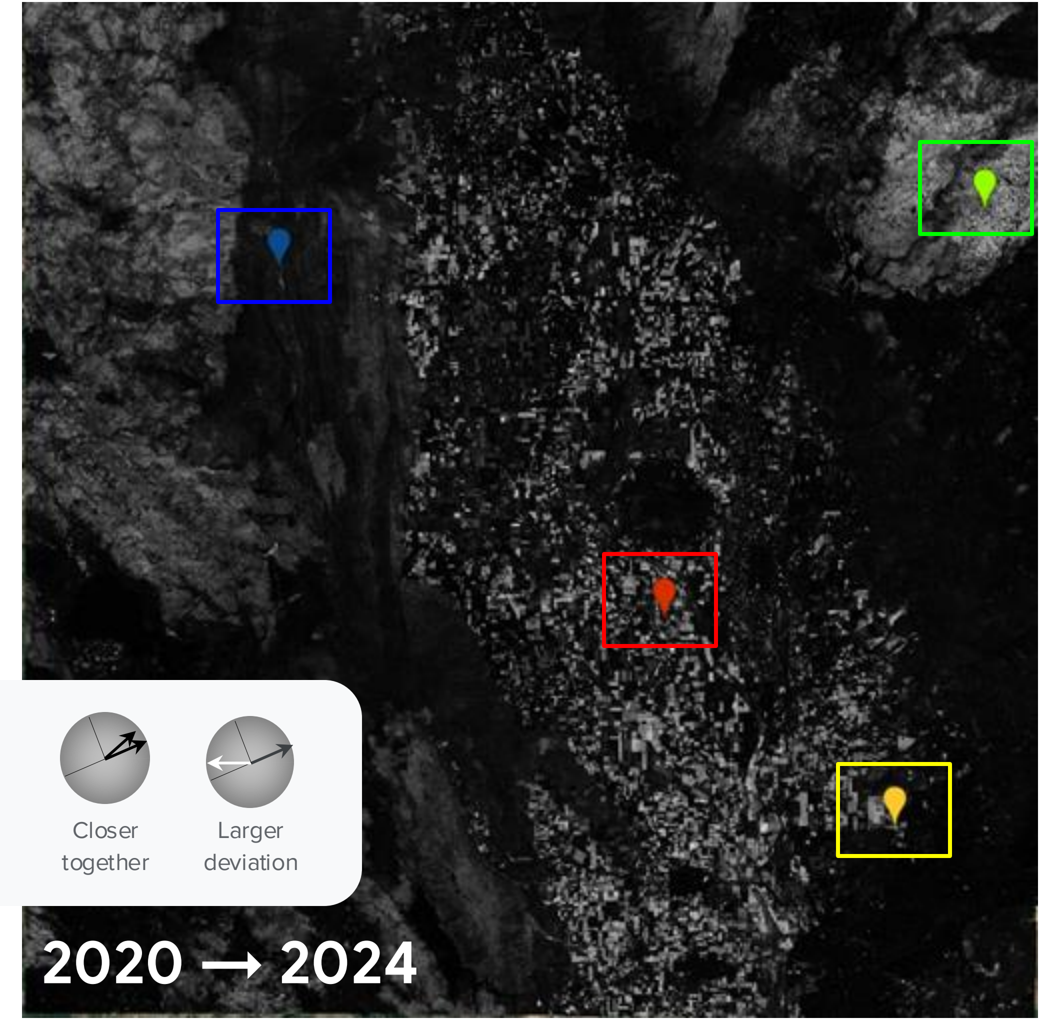


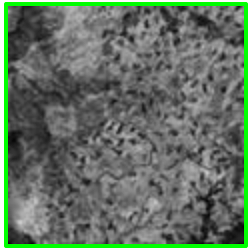
Water
resources

Agricultural
monitoring

Suburban
development

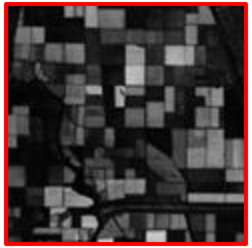
Google





Forests fires
& Timber

Water
resources



Agricultural
monitoring

Suburban
development

Under the hood: Dot product similarity for identifying change in embedding space

```
// Load collection.
var dataset = ee.ImageCollection('GOOGLE/SATELLITE_EMBEDDING/V1/ANNUAL');

// Point of interest.
var point = ee.Geometry.Point(-121.8036, 39.0372);

// Get embedding images for two years.
var image1 = dataset
      .filterDate('2023-01-01', '2024-01-01')
      .filterBounds(point)
      .first();

var image2 = dataset
      .filterDate('2024-01-01', '2025-01-01')
      .filterBounds(point)
      .first();

// Visualize three axes of the embedding space as an RGB.
var visParams = {min: -0.3, max: 0.3, bands: ['A01', 'A16', 'A09']};

Map.addLayer(image1, visParams, '2023 embeddings');
Map.addLayer(image2, visParams, '2024 embeddings');

// Calculate dot product as a measure of similarity between embedding vectors.
// Note for vectors with a magnitude of 1, this simplifies to the cosine of the
// angle between embedding vectors.
var dotProd = image1.multiply(image2).reduce(ee.Reducer.sum());

// Add dot product to the map.
Map.addLayer(dotProd, {min: 0, max: 1, palette: ['white', 'black']},
'Similarity between years (brighter = less similar)'
);
```

https://code.earthengine.google.com/ad76b608c58da436e3b902f468b6d168

Summary: Change detection

- Relatively stable overall
- Expect phenological and climatic variability
- Summaries of numerical representations still require interpreted meanings ⟶ semantics matter!