LEGEND

Geo Embeddings 101

Models, image chips, and semantic search

Matt Hanson

FOSS4G 2026 · Hiroshima

Sep 2, 2026

You have likely heard all of these

- embedding
- vector embedding
- image embedding
- Earth observation embedding
- geo embedding
- geospatial embedding
- embedding field
- latent space
- foundation model
- geospatial foundation model
- encoder
- fine-tuning
- zero-shot
- semantic search

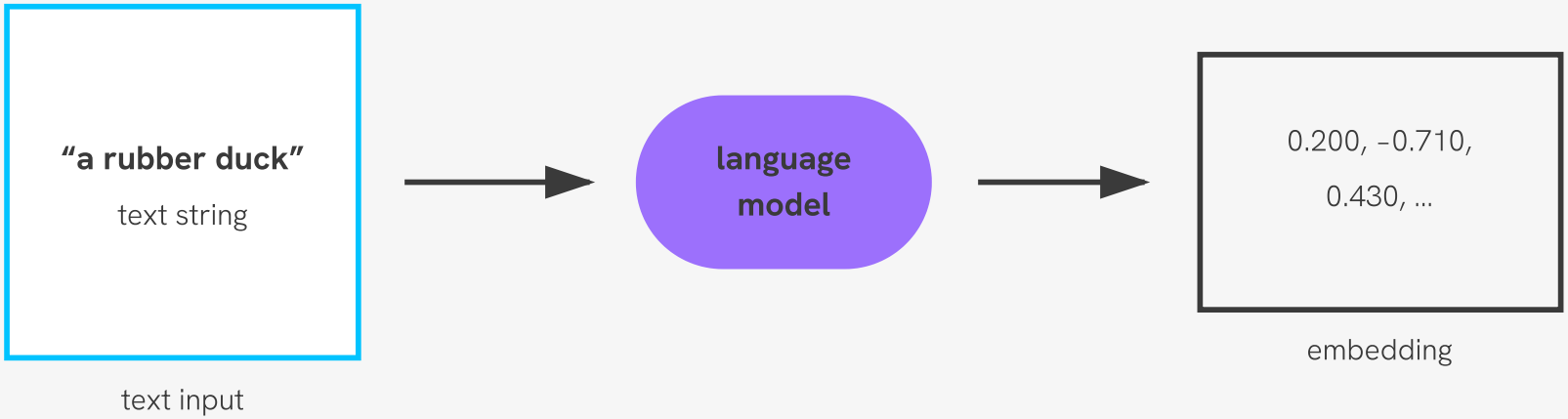
Now try defining any of them. Some of these are the same thing. Some are not.

01

What is an embedding?

Text, images, and other data can become vectors. Distance between vectors becomes useful.

Language models already do this



A text-embedding model produces one vector for the whole string.

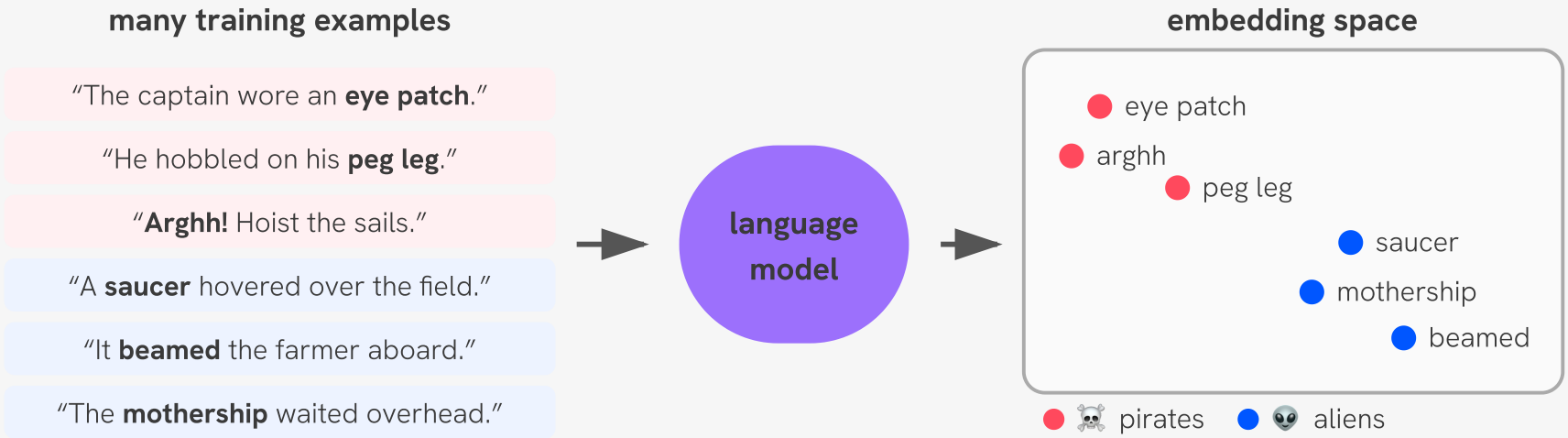
Every token reads the whole sentence

A flying **saucer** hovered over the field. → `0.31, −0.08, …`

He set the cup back on its **saucer**. → `−0.12, 0.44, …`

Same word, two different vectors. Every other token in the sentence feeds into it — not a fixed window of neighbours, and not the word on its own. Combine all of them and you get one vector for the whole sentence.

What did the model learn?



Each highlighted word gets its own vector, shaped by the sentence around it. That is what **semantic** similarity means. None of these words means *pirate* or *alien* on its own. “Hoist the sails” is about sailors — and pirates are sailors. The real model holds far more of these links than a picture can show, and the group names are ours.

Now replace the text with an image



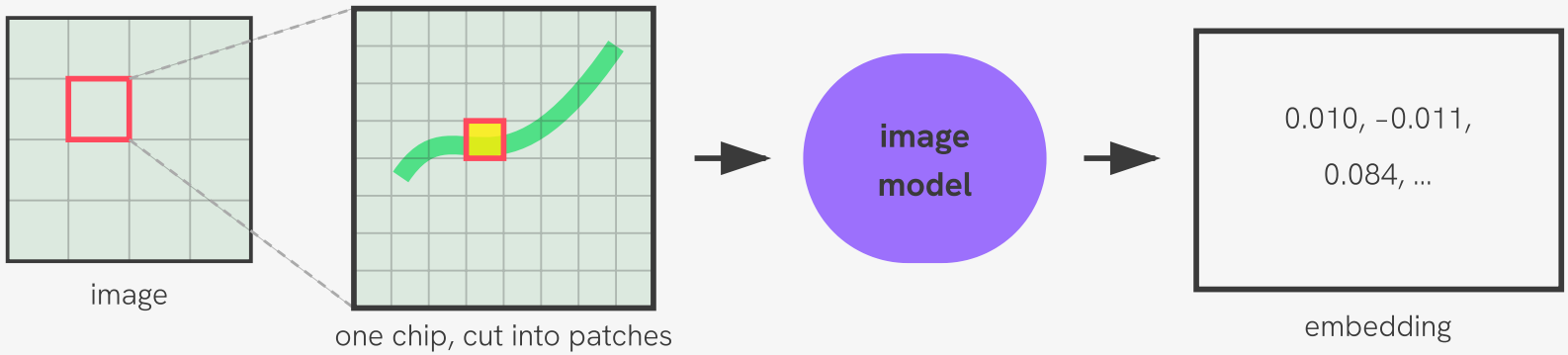
NAIP · Alabama, 2023 · 1,024 × 1,024 px

Same idea, different model

- The input is an **image chip** — a fixed-size crop of overhead imagery, not a sentence
- The model is a **geospatial model**, trained on overhead imagery: Sentinel-2, Landsat, NAIP
- Nothing in it was trained on text
- The output is the same shape: one vector

Both models turn an input into numbers that preserve useful meaning. They do not produce the same numbers, and they do not share a vector space.

Image → vector



The model cuts the chip into a grid of **patches**, then reduces them to one vector for the chip.

The same structure in text and imagery

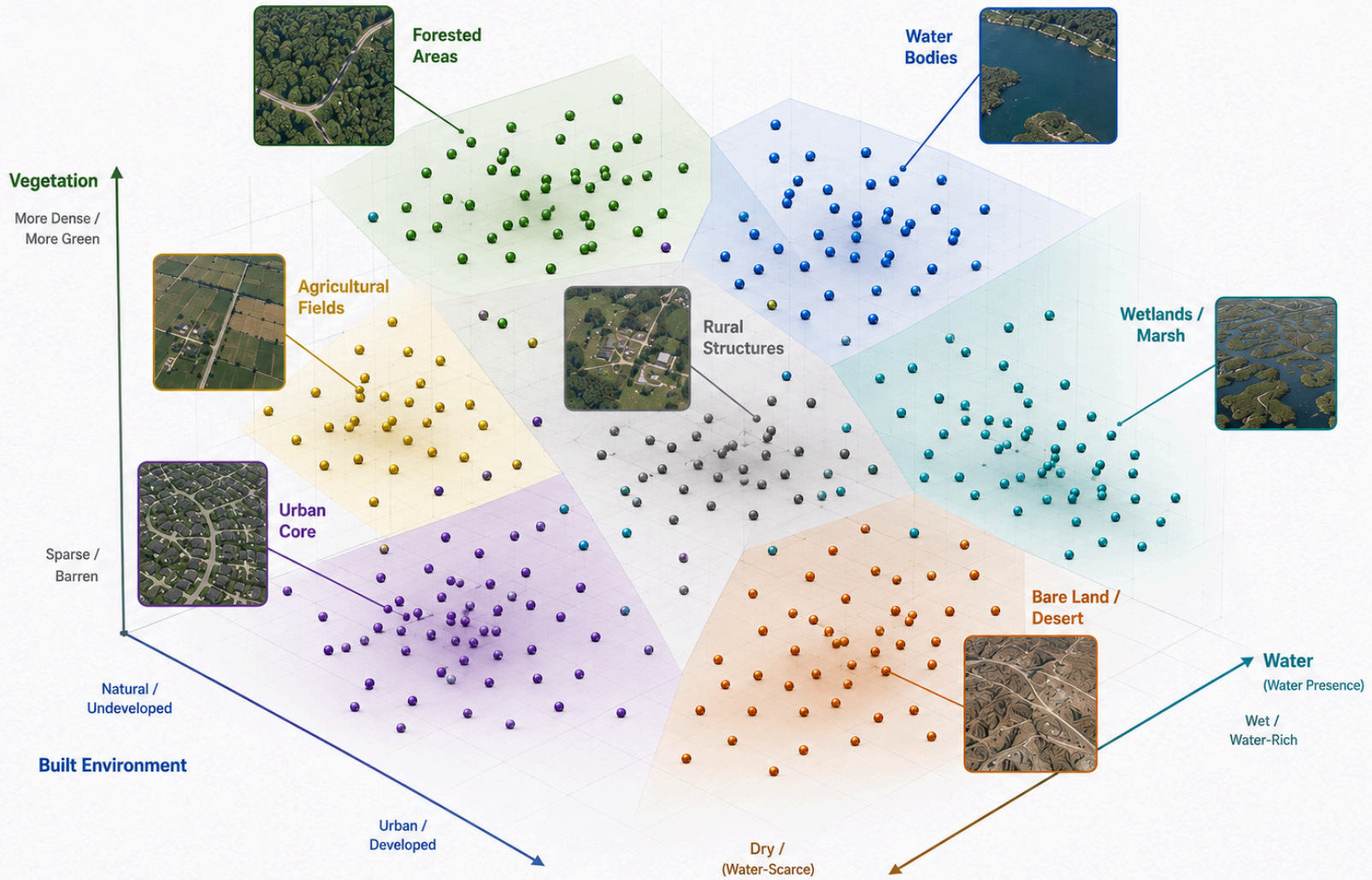
Text	Imagery
a sentence	a **chip**
a token	a **patch**
one vector per token	one vector per patch
one vector for the sentence	one vector for the chip

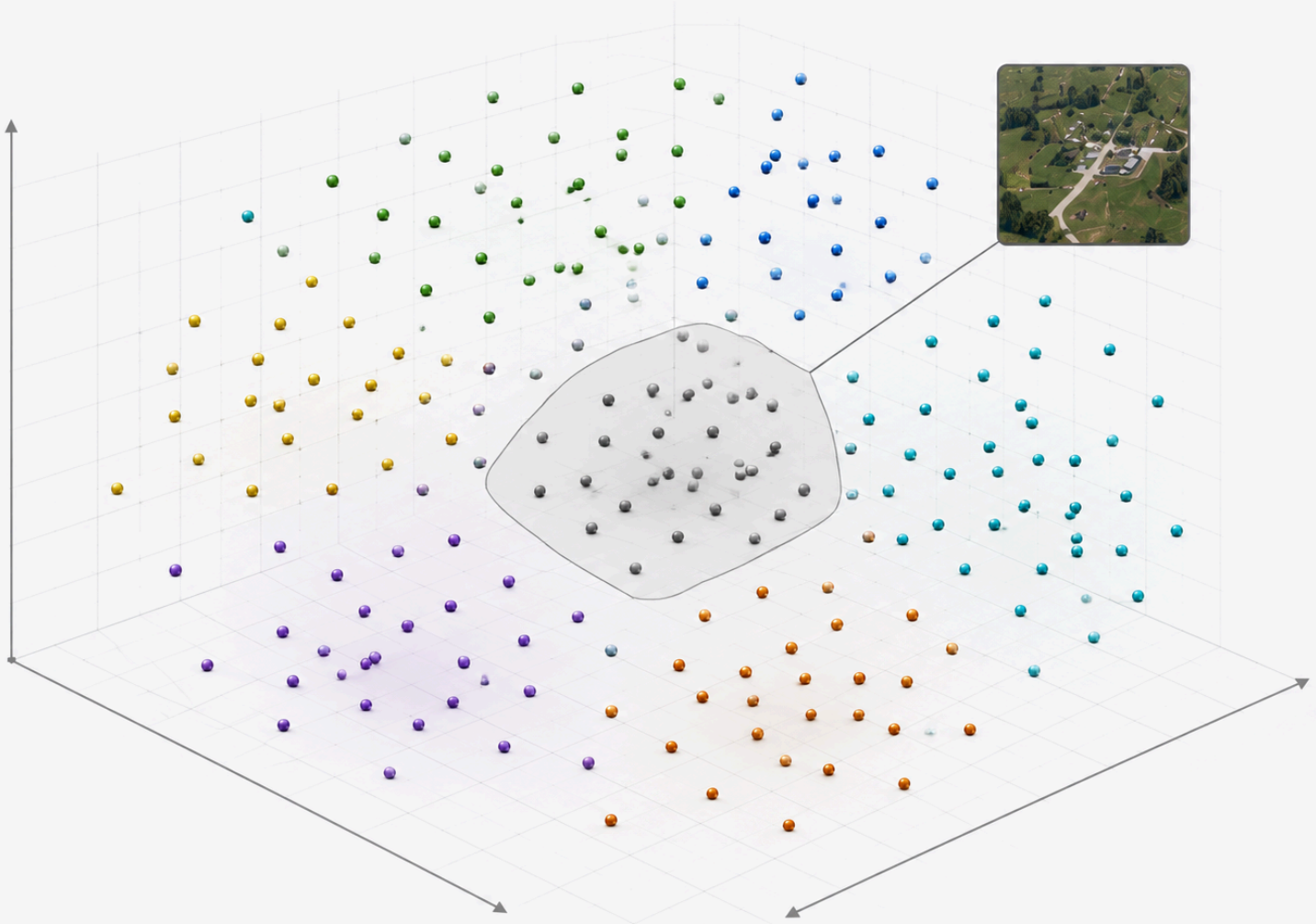
Most models hand you the chip vector. Some hand you the patch grid instead, and some skip chips entirely and give one vector per pixel. (“Patch” is an imagery word — text only ever says token.)

What kind of embedding?

Term	What it means
Embedding · vector embedding	The same thing. Every embedding is a vector of numbers, and "embedding" is the shorthand
Image embedding	Input: pixels
Earth observation embedding	Input: pixels, spectral bands, resolution, and sometimes time
Geospatial embedding	Input: Earth observation data plus explicit location or place context
Embedding field	Measurements anchored across space and time
Latent space · embedding space	Not a vector — the space the vectors live in

These are useful distinctions, not settled vocabulary. Models increasingly blur the boundaries.


Forested Areas
Water Bodies
Vegetation
More Dense / More Green
Agricultural Fields
Rural Structures
Wetlands / Marsh
Urban Core
Sparse / Barren
Bare Land / Desert
Water
(Water Presence)
Wet / Water-Rich
Natural / Undeveloped
Built Environment
Urban / Developed
Dry / (Water-Scarce)



What is an embedding?

They are summaries

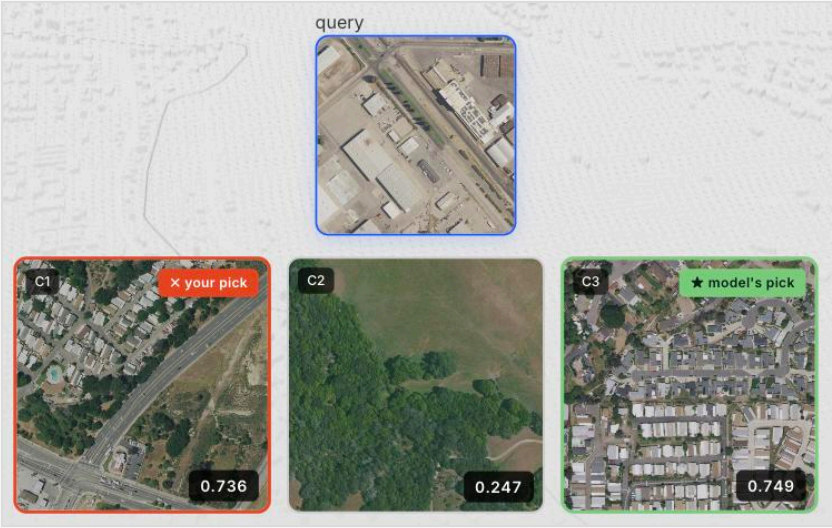
They are imperfect summaries

They are imperfect but detailed summaries

They are imperfect but detailed mathematical summaries

They are not classifications

Guess what Clay thinks



One query chip, six candidates. Pick the one you think sits closest in Clay's embedding space, then see the cosine similarity for every option.

Here the model and the player disagreed — 0.749 against 0.736. Both suburbs. The forest scored 0.247.

chipmatch.apps.lgnd.hosted.unionai.cloud

Sam Barrett, LGND

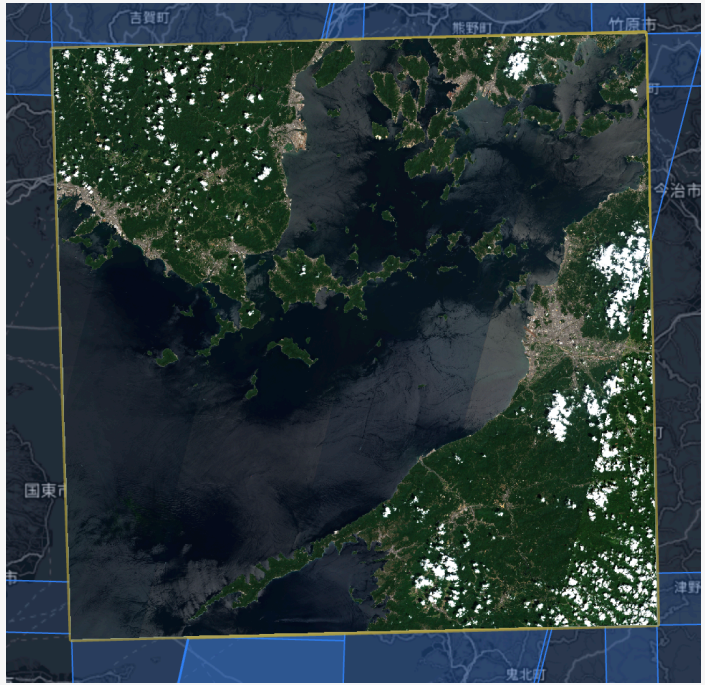
02

Chip size

Footprints, overlap

Why not embed a whole Sentinel-2 image?

- One Sentinel-2 tile is about **110 km across** — roughly 12,000 km^2
- A single vector for all of it would average a city, a forest, the sea and clouds into one summary
- At 256 px and 10 m, that same tile is about **1,800 chips**



Sentinel-2 · 10 m · Seto Inland Sea · 23 Aug 2026

The chip is the

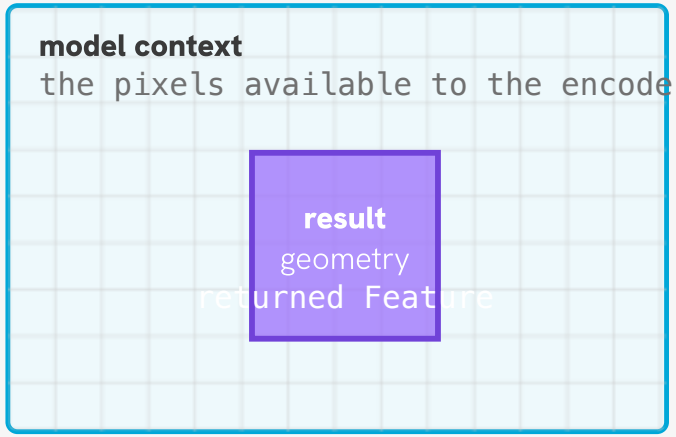
unit of search

You can only find something at the scale you embedded it.

Embedding footprint and model context

Suppose the search result is one **10 m pixel**.
The model may have produced that vector from a **2.56 km image chip** around it.

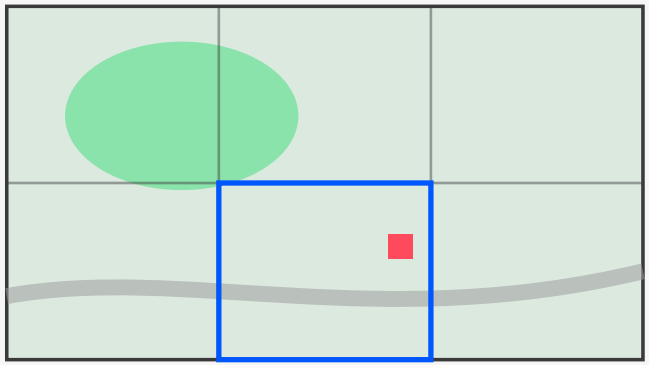
- `geometry` is the footprint the embedding represents
- `context_geometry` is the area the model looked at



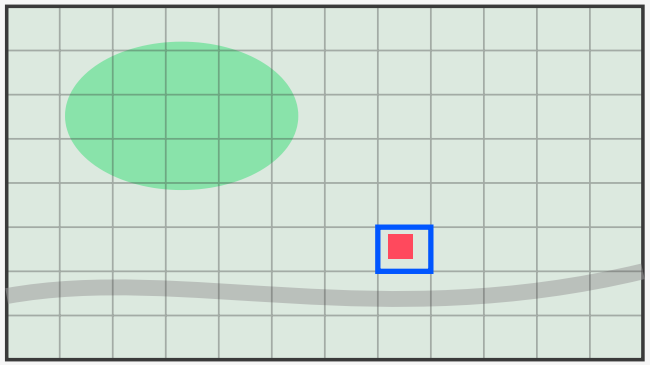
For chip embeddings, these may be the same. For pixel or patch embeddings, they may be very different.

IF THEY ARE DIFFERENT, THE RESPONSE SHOULD INCLUDE BOTH.

The smallest thing you can find



large chips
lost in the average



small chips
most of its own vector

Same place, same imagery, two chip sizes. The red square is the thing you are looking for; the blue outline is the chip whose vector has to carry it.

So pick the chip size closest to the size of the thing you care about — and if you do not know it yet, pick smaller.

Chip size changes the question



NAIP

~1 m resolution · ~256 m across

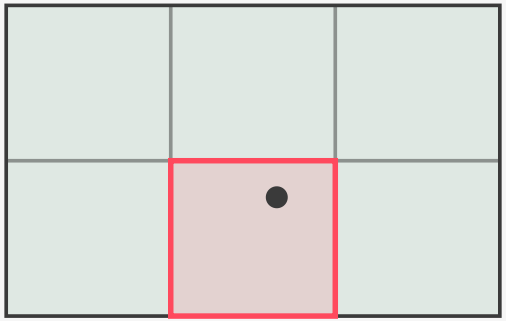


Sentinel-2

10 m resolution · 2.56 km across

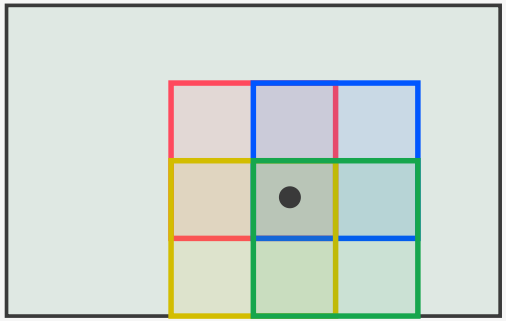
Both are 256 × 256 pixels.

Three ways to cover the ground



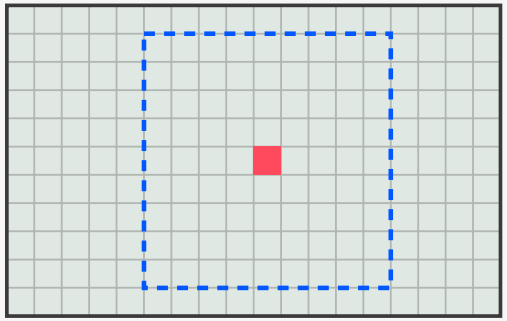
no overlap

1 vector for this place



50% overlap

4 vectors for this place



per pixel

1 vector, read from wider

The dot is one place; the outlines are the chips that have to describe it.

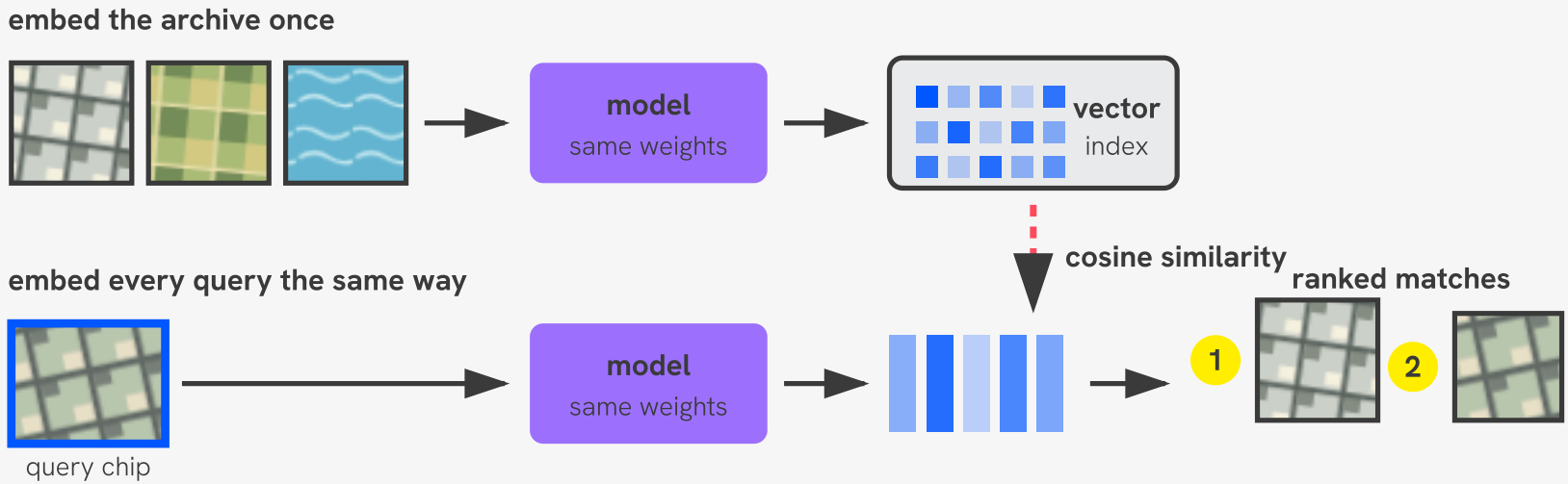
Overlap is an inference choice, not a property of the model — record chip size **and** stride. Per-pixel goes furthest: what a vector **describes** shrinks to one pixel while what it **reads** stays wide.

03

Similarity and search

Queries, nearest neighbours, and cosine similarity.

Similarity search



The query must be embedded with the **same model** as everything already stored, then compared against every stored vector.

Similar imagery



query



nearest matches in the embedding space

No labels. No classifier. One example, ranked by similarity.

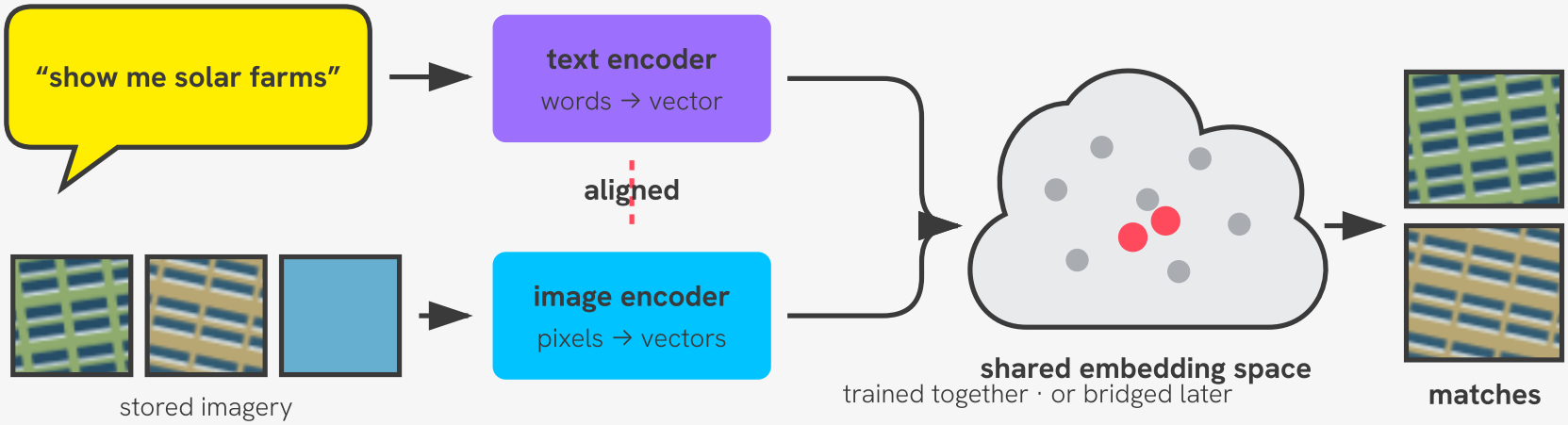
LEGEND

What are embeddings useful for?

YOU NEED TO…	START WITH…
Find places that look like this one	EMBEDDINGS
Find sites where something is happening	EMBEDDINGS
Detect semantic change	EMBEDDINGS
Count instances of something	EMBEDDINGS + SEGMENTATION
Search a large archive quickly	EMBEDDINGS
Measure reflectance, temperature, or biomass	ARD
Quantify change: 20% biomass loss	ARD
Produce an authoritative or regulatory measurement	ARD
Maintain a calibrated long-term record	ARD

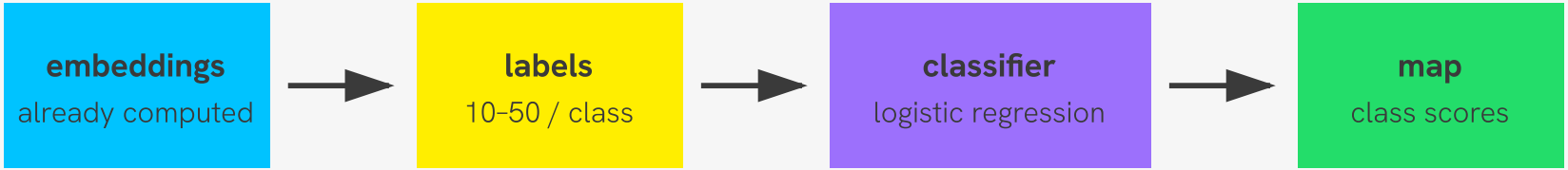
ARD = analysis-ready data. Embeddings find patterns; ARD preserves measurements.

What if my query is text?



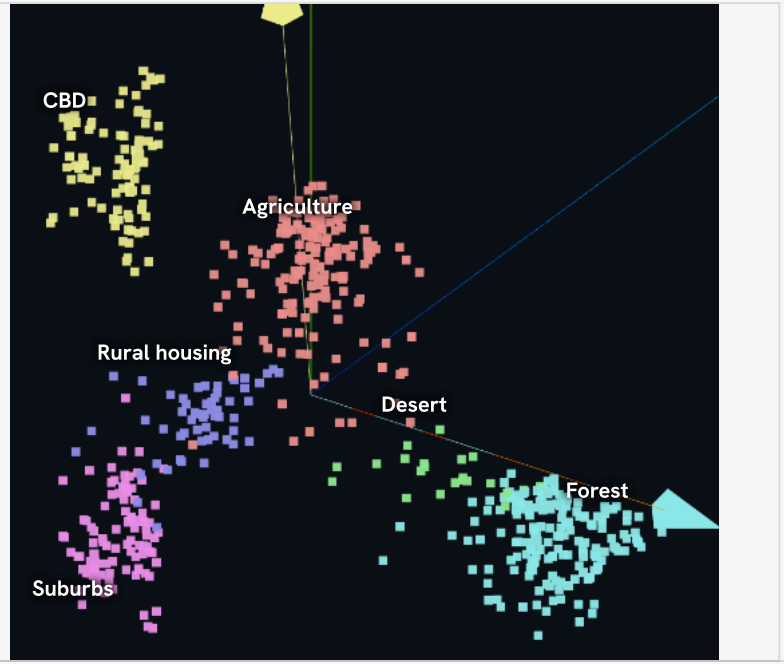
Text search works only when the text encoder and image encoder share a vector space.

From embeddings to a classifier



Most of the model training has already happened.

The classifier creates these axes



Clay gives each chip 1,024 numbers. None of those dimensions means “forest,” “CBD,” or “agriculture.” We labelled 642 chips, then fitted a linear classifier for each class. Each classifier found a **direction through the existing embedding space**; three of those fitted directions become the axes in this picture.

The classes separating along those directions is the point: the useful structure was already in Clay’s embeddings. The classifier only learned where to draw the boundaries.

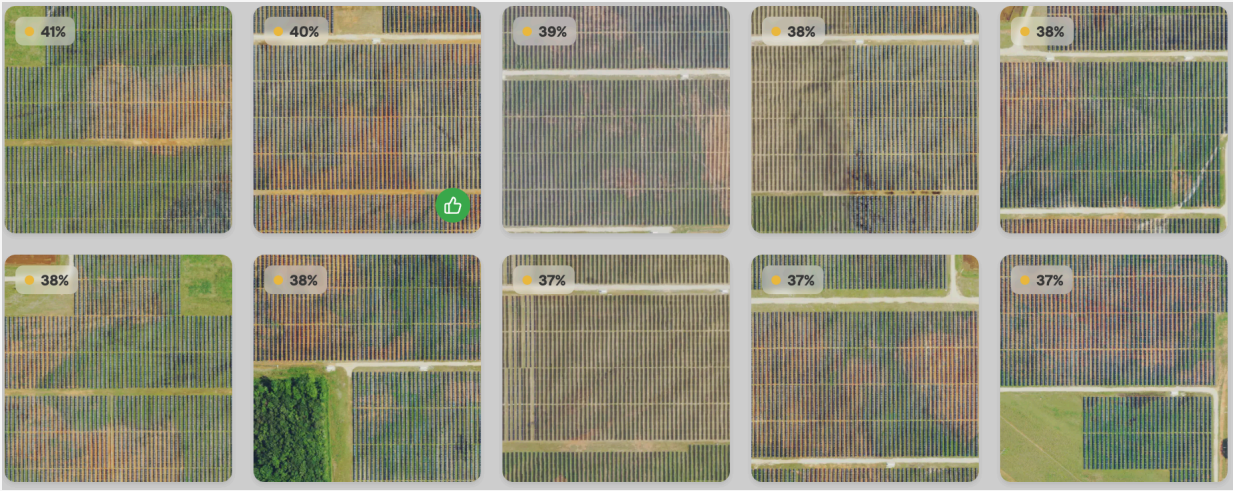
Embedding Geometry Toy · Sam Barrett. The concept directions are linear SVMs fitted on human-labelled chips, not something Clay ships.

How many labels do you need?

Approach	Labels needed	What gets trained
Zero-shot	none	Nothing. You compare vectors directly — similarity search is this
Few-shot	10–50 per class	A small classifier on top of frozen embeddings
Fine-tuning	many more	The encoder itself

Fine-tuning changes the encoder, and a changed encoder means a changed vector space. Every vector you already stored becomes invalid.

Ten labels, every solar farm



Five positive and five negative chips — the whole training set, and fewer per class than the table above suggests. The ranking is a **classifier score, not a distance and not a percentage**. Raw scores run −0.34 to 0.44: read the order, not the number.

04

Geospatial Foundation Models

Input, training data, chip contract, output, time, and openness.

What is a foundation model?

Term	What it means
Encoder	The part that turns an input into learned features: chip, patch, or pixel vectors
Foundation model	Trained once, on a large amount of data, for no particular task — then reused for many tasks
Geospatial foundation model	A foundation model trained on Earth observation and other spatial data

The expensive training already happened. You are using the result.

Text search needs a text encoder aligned to the image features; imagery training alone is not enough.

Inputs and training

Model	Input	Training data	Chips
Clay 1.5	S2, Landsat, S1, NAIP, LINZ, MODIS	70M chips; corpus not packaged	trained at 256 px; other sizes accepted
AlphaEarth	optical, radar, LiDAR/elevation, climate, land cover, text	many public sources; corpus not released	closed inference
TESSERA v2	S1 + S2	annual radar + optical sequences	product is already per pixel
Prithvi 2.0	six HLS bands	4.2M HLS v2 samples	224 px; four frames
TerraMind Base	S1, S2, DEM, RGB	9M published TerraMesh samples	224 px; missing modalities allowed
OlmoEarth 1.2 Base	S2, S1, Landsat, plus six derived maps	285,288 published samples	single image or time series
DINOv3 SAT	RGB	undisclosed SAT-493M corpus	16 px patches; flexible image size

What might cluster in each space?

Model	One embedding represents	Nearby vectors might share…
Clay 1.5	One chip or patch, one observation	land cover, built form, field pattern
AlphaEarth	One 10 m location over a year	annual land use, ecosystem or surface state
TESSERA v2	One 10 m location over an S1 + S2 year	crop cycles, vegetation regimes, flooding
Prithvi 2.0	One chip or patch across four HLS dates	surface type, phenology, disturbance
TerraMind Base	One patch in each supplied modality	terrain and structures visible across sensors
OlmoEarth 1.2 Base	One image or time series after pooling	land cover, land use, temporal behaviour
DINOv3 SAT	One RGB image patch	visual texture and geometry: roofs, roads, fields

The model does not name these groups. Dataset, scale, pooling and distance metric all shape what appears.

Outputs and openness

Model	Output	Time	What is open?
Clay 1.5	1,024-d CLS and patch tokens	single observation	Apache-2.0 weights + code
AlphaEarth	64-d per 10 m pixel	annual	CC BY outputs; model closed
TESSERA v2	128-d per 10 m pixel	annual sequence	CC0 weights; MIT code
Prithvi 2.0	768–1,280-d tokens, by model size	four dates	Apache-2.0
TerraMind Base	196 × 768 patch tokens per modality	supplied observation	Apache-2.0; corpus CC BY-SA
OlmoEarth 1.2 Base	768-d features; pooling is downstream	single image or series	weights, code and data; custom licence
DINOv3 SAT	class and patch tokens	single image	gated, custom terms

“Open” is several questions

- Are the **weights** available?
- Is the inference **code** available?
- Can you inspect the **training data**?
- Do the **licences** permit your use?
- Can you run **inference** yourself?

AlphaEarth has open outputs from a closed model. TerraMind publishes weights, code, and its training corpus. OlmoEarth publishes all three as well, but its licence forbids military and extractive use, so it is not open source. Clean examples in this survey include **CROMA, SeCo, and UniverSat**. Among the models above, **TerraMind** comes closest: open weights, code, inference, and the actual training corpus.

05

How do I use them?

Start with precomputed vectors, then decide whether you need to run a model.

Where should I start?

If you want to…	Start with…	Why
explore search or classification	**Source Cooperative**	precomputed; no GPU
compare models on the same chips	**Major TOM 249k suite**	seven models, one grid
use annual per-pixel vectors	**AlphaEarth or TESSERA**	the published output is already a map
embed your own multispectral imagery	**Clay, TerraMind, Prithvi, OlmoEarth or DOFA**	control imagery, chip size and pooling
embed RGB imagery	**Clay or a satellite vision model**	simpler input contract

Precomputed embeddings are the fastest start. Run inference when you need control or new coverage.

Embeddings on Source Cooperative

No account. No token. Every object has an HTTPS URL and an anonymous S3 URI.

Dataset	What you get	Format / open with
Clay 1.5 · Sentinel-2	global monthly chip vectors; 1,280 m / 256-d or 2,560 m / 1,024-d	Hive-partitioned **GeoParquet** · DuckDB, GeoPandas
AlphaEarth · annual	64-d per 10 m pixel, 2017–2025, tiled by year and UTM zone	quantized **COG + correcting VRT** · GDAL, Rasterio, QGIS
AlphaEarth · mosaic	the same annual 64-d pixels as one global `(time, band, y, x)` array	sharded **Zarr v3** · Zarr, Xarray

`https://data.source.coop/<organization>/<repository>/<path>`

For Clay, filter Hive partitions and bounding boxes. For AEF COGs, use the supplied spatial index and open the **VRT**, not the TIFF directly. Both AEF products store quantized `int8` values.

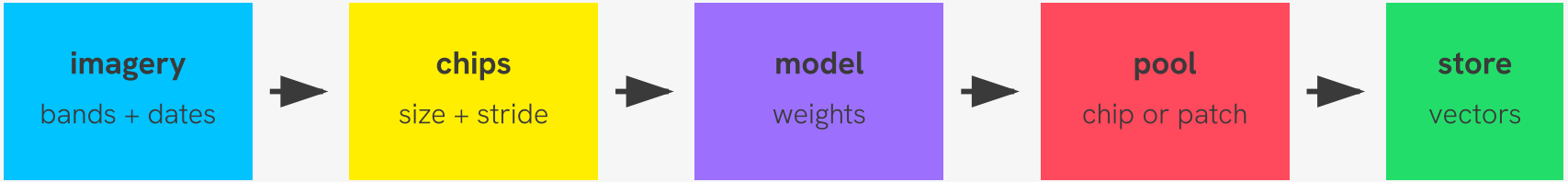
Similarity from an existing Clay chip

```sql
WITH chips AS MATERIALIZED (
  SELECT chips_id, embedding
  FROM read_parquet('s3://us-west-2.opendata.source.coop/clay/lgnd-embeddings/' ||
    'monthly-aggregated/model_version=v1.5/collection=sentinel-2-l2a/' ||
    'chip_size=1280m/dims=256/geohash=9y/year=2025/month=01/*.parquet')
),
seed AS (SELECT * FROM chips ORDER BY chips_id LIMIT 1)
SELECT c.chips_id,
       list_cosine_similarity(c.embedding, s.embedding) AS score
FROM chips c CROSS JOIN seed s
WHERE c.chips_id <> s.chips_id
ORDER BY score DESC
LIMIT 12;
```

Here the query is one existing Clay vector. For your own chip, run **Clay 1.5 with the same Sentinel-2 configuration**, then supply its 256 values instead. An unrelated image or text vector will not work.

Alex Leith's Auspacious workshop at this conference has notebooks that run this end to end.

Running your own inference



Pooling is how a model reduces its internal patch vectors to one vector for the chip — usually a mean, or a **CLS** token added for the purpose. Most tools hand you the chip vector and you never see a patch.
Every choice becomes part of the resulting dataset.

What to do next

- **Try several models:** Use the Major TOM 249k suite—seven models over the same 248,719 grid cells and the same 384 × 384 crops—so the comparison is fair.
- **Try change detection:** Compare embeddings for the same footprint at different dates. Temporal resolution sets the limit: an annual embedding cannot show a change that appeared and disappeared within the year.

Thank you

That was GeoAI. The word appears only once in this deck - here.

What will you use embeddings for?

LGND

matt@lgnd.ai
lgnd.ai