# OlmoEarth: Stable Latent Image Modeling for Multimodal Earth Observation

Team OlmoEarth★

Henry Herzog♥[1] Favyen Bastani♥[1] Yawen Zhang♥[1] Gabriel Tseng♥[1] Joseph Redmon♥[1]

Hadrien Sablon[1] Ryan Park[1] Jacob Morrison[1, 2] Alexandra Buraczynski[1] Karen Farley[1] Joshua Hansen[1] Andrew Howe[1] Patrick Alan Johnson[1] Mark Otterlee[1] Ted Schmitt[1] Hunter Pitelka[1] Stephen Daspit[1] Rachel Ratner[1] Christopher Wilhelm[1] Sebastian Wood[1] Mike Jacobi[1]

Hannah Kerner[3] Evan Shelhamer[4]

Ali Farhadi[1, 2] Ranjay Krishna[1, 2] Patrick Beukema♥[1]

[1] Allen Institute for AI [2] University of Washington [3] Arizona State University [4] University of British Columbia

★OlmoEarth was a team effort.
♥Equal contribution from modeling team. Authors are listed in reverse alphabetical order by last letter of first name.

**Platform:** `olmoearth.allenai.org`

**Training Code:** `olmoearth_pretrain` (pretraining) `olmoearth_projects` (fine-tuning)

**OlmoEarth Pre-trained Models:** `OlmoEarth-v1-Nano` `OlmoEarth-v1-Tiny` `OlmoEarth-v1-Base` `OlmoEarth-v1-Large`

**OlmoEarth Pre-training Dataset:** `olmoearth_pretrain_dataset`

**Contact:** `olmoearth@allenai.org`

## Abstract

Earth observation data presents a unique challenge: it is spatial like images, sequential like video or text, and highly multimodal. We present OlmoEarth: a spatio-temporal, multimodal foundation model that employs a novel self-supervised learning formulation, masking strategy, and loss all designed for the Earth observation domain. OlmoEarth achieves state-of-the-art performance compared to 12 other foundation models across a variety of research benchmarks and real-world tasks from external partners. When evaluating embeddings OlmoEarth achieves the best performance on 15 out of 24 tasks, and with full fine-tuning it is the best on 19 of 29 tasks. We deploy OlmoEarth as the backbone of an end-to-end platform for data collection, labeling, training, and inference of Earth observation models. The OlmoEarth Platform puts frontier foundation models and powerful data management tools into the hands of non-profits and NGOs working to solve the world's biggest problems. OlmoEarth source code, training data, and pre-trained weights are available at `https://github.com/allenai/olmoearth_pretrain`.

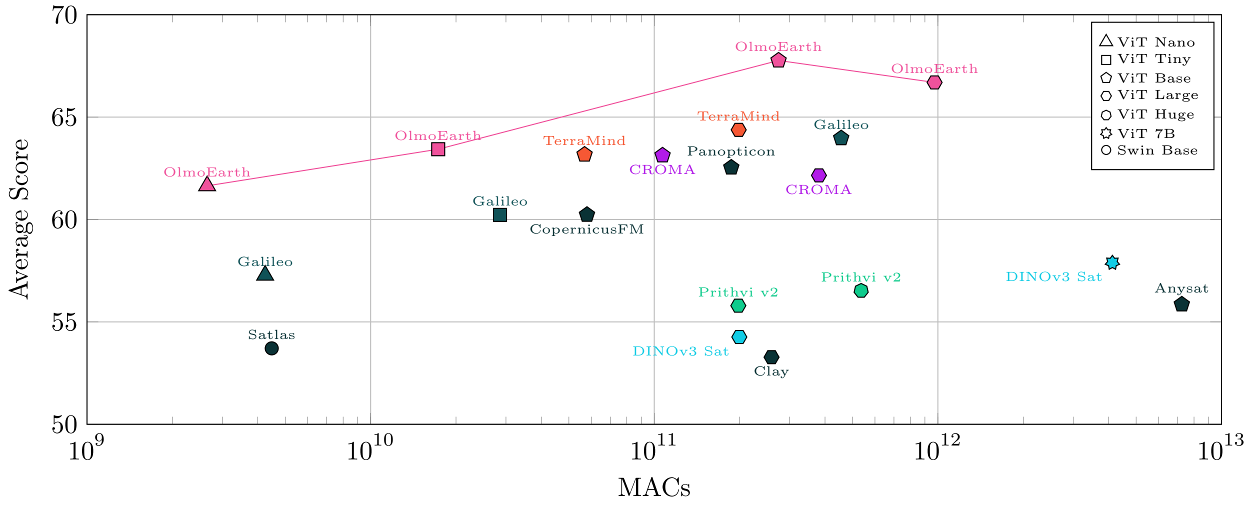


**Figure 1** OlmoEarth defines a Pareto optimum of performance vs. computational efficiency averaged across 13 embedding tasks (measured by kNN and linear probing)[1]. The chart shows average multiply-accumulate operations to encode one example across all tasks (input size varies by task). See Table 2 for full results.

# 1 Introduction

Earth observation foundation models show promising results in research settings [2, 15, 16, 25, 50, 54, 55]. However, adoption for real-world tasks lags behind, especially in the non-profit sector. Foundation models are large, complex to train, and expensive to deploy.

To help enable non-profit, humanitarian, and environmental organizations to use these powerful tools we train OlmoEarth, a new family of models, using a novel, stable training regime. We comprehensively evaluate OlmoEarth against 12 other foundation models on research benchmarks and real-world tasks from partner organizations. Finally we deploy these models in an open, end-to-end platform bringing frontier models directly to organizations who need it the most.

## 1.1 Stable Training

Foundation models are complex and expensive to train. When attempting to replicate existing work we frequently saw training instability, representation collapse, and models underperforming their stated potential. We introduce a stable training regime that models images in latent space but avoids instability and collapse.

Our approach strikes a middle ground between two common approaches in self-supervised learning. Masked autoencoders (**MAE**) predict pixel-level reconstructions of masked input while approaches like **I-JEPA** and Latent Masked Image Modeling (**Latent MIM**) predict reconstructions in feature space [1, 56]. MAE tends to be stable but limited in its feature representations while latent approaches are unstable but produce better features [34].

We present Latent Masked Image Modeling of Linear, Invariant Token Embeddings (**Latent MIM Lite**), a simplification of Latent MIM that leads to stable training and better performance. We replace the target encoder of Latent MIM with a linear projection from image patches to token space that is randomly initialized and never updated during training. This simple modification stabilizes training but maintains the representative power of modeling in latent space. It also unifies self-supervised and supervised learning as we project both observational data and labeled maps through the frozen random projection layer into token space and calculate loss the same for both.

[1] Average over all tasks every model can perform, specifically the Sentinel-2 versions of: m-bigearthnet, m-so2sat, m-brick-kiln, m-eurosat, BreizhCrops CropHarvest-Togo, CropHarvest-PRC, m-cashewplant, m-SA-crop-type, PASTIS, MADOS, AWF, Nandi.

### 1.1.1 Masking

In image or text modeling it is sufficient to randomly mask some portion of the input and have the model reconstruct the input from context. With multimodal remote sensing data, any token in the input will have many similar tokens either in space, time, or at a different aligned modality. Random masking is too easy of a task unless you use a very high masking ratio [50]. We introduce a modality-aware masking strategy that combines random token masking with full modality reconstruction. This makes the task challenging without resorting to skewed masking ratios.

### 1.1.2 Loss

Like other SSL approaches in latent space we use a contrastive loss instead of a reconstruction loss. However, contrasting a reconstructed token against all other tokens in a batch, or even in the same sample, leads to many easy negatives given the redundant nature of Earth observation data. Instead we contrast tokens only with other tokens in their respective bandset (a subdivision of modality explained in 2.1). This focuses the model training on a more challenging but more productive objective, as shown in our experiments.

## 1.2 Comprehensive Evaluation

There is no standard evaluation suite for remote sensing models. While there are some established standard practices [16, 39, 50], they are not always followed. To get a more complete picture of the state of foundation modeling we run a comprehensive evaluation effort of OlmoEarth compared to 12 other foundation models on 18 research benchmarks and 19 datasets from 7 partner organizations that are using Earth observation modeling in their work.

Following standard practice we evaluate all models using simple transfer learning techniques (kNN and linear probing) as well as full, end-to-end fine-tuning. We evaluate all models using a standard training recipe and sweeping over a variety of hyperparameters. OlmoEarth achieves the best performance in 15 of 24 tasks for the kNN/LP evaluation and 19 of 29 tasks for full fine-tuning. See Figure 1 for a summary.

## 1.3 Open Platform

Training and fine-tuning remain out of reach for most environmental and humanitarian non-profits. Applying a foundation model to a task requires data gathering, alignment, pre-processing, labeling, fine-tuning, and running inference. We deploy OlmoEarth as part of the OlmoEarth Platform to simplify and streamline this process.

The OlmoEarth Platform is an end-to-end solution for organizations who want to harness Earth observation data for the public good. Our partner organizations are already using the platform for things like mangrove conservation, ecosystem mapping, and food security. The OlmoEarth Platform solves the last-mile problem of putting frontier research into the hands of people who can use it to do the most good.

# 2 OlmoEarth

OlmoEarth is a Vision Transformer (ViT) based encoder-decoder style architecture. It processes a multimodal image timeseries of aligned satellite images and derived maps. A FlexiViT-style projection layer [5] converts the input data from pixels to tokens with a variable patch size. Positional, temporal, and modality encodings add additional context to the tokens. During training, some portion of the input tokens are masked. The encoder transformer layers attend across space, time, and between modalities to produce embeddings for the input tokens. The decoder predicts representations for the masked input tokens.

## 2.1 Data

OlmoEarth is designed to flexibly handle input Earth observation data across a range of spatial and temporal resolutions. During our pretraining experiments we train on three satellite modalities and six derived maps:

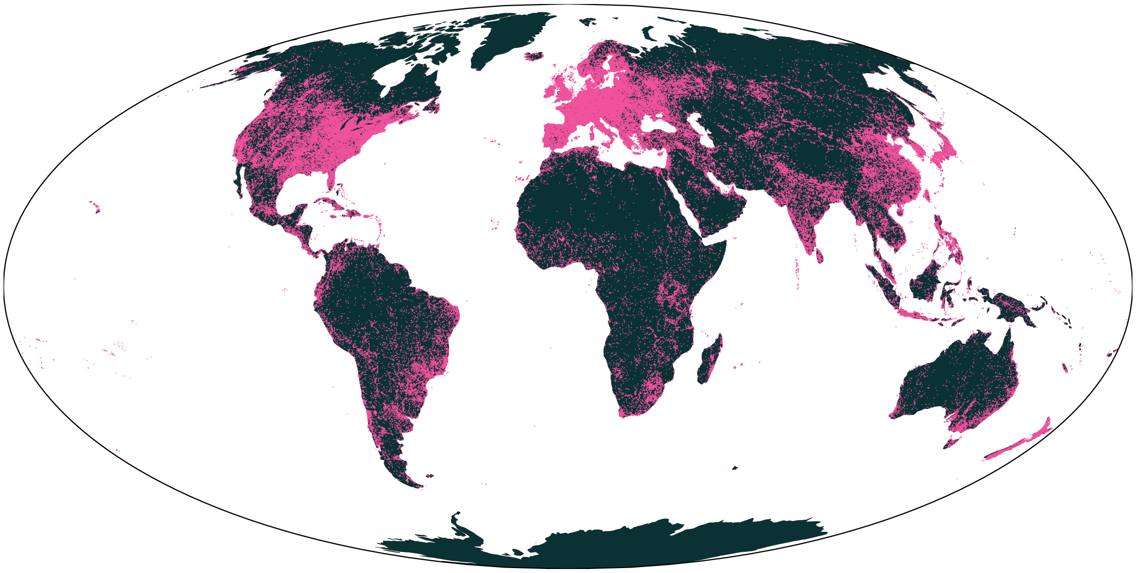


**Figure 2** Global distribution of OlmoEarth pretraining data. We sample 285,288 locations based on OpenStreetMap categories.

| Observations | Maps | |
|---|---|---|
| Sentinel-1 | WorldCereal [53] | OpenStreetMap [38] |
| Sentinel-2 | WorldCover [59] | Cropland Data Layer [51] |
| Landsat-8 | SRTM [36] | Canopy Height Map [47] |

Our pretraining dataset contains 285,288 samples from around the world. Each sample covers a 2.56km×2.56km spatial region and a one-year time range. For multi-temporal modalities, we use up to 12 timesteps sampled monthly over the course of the year, although many samples contain only a subset of the timesteps and modalities.

For the above modalities we resample the data to be uniformly 10 meters per pixel. We experimented with adding NAIP data at 2.5 meter per pixel [52] and ERA5 data at 160 meters per pixel [23] but found no significant improvement on our evaluations.

We further subdivide Landsat and Sentinel-2 into **bandsets** based on the original resolution of their bands, grouping bands captured at the same resolution together. Landsat consists of 2 bandsets while Sentinel-2 consists of 3 bandsets. For the precise split see the OlmoEarth source code.

The locations of samples are chosen based on OpenStreetMap features. We select 120 categories of map features in OpenStreetMap, ranging from roads to geothermal power plants, and enumerate all 2.56km × 2.56km tiles containing each category. We then randomly sample up to 10,000 tiles per category to derive the 285,288 samples (many categories appear in fewer than 10,000 tiles). The one-year time range of each sample is sampled uniformly between January 2016 and December 2024.

## 2.2 Architecture

Similar to many Earth observation models, OlmoEarth is a transformer-based encoder-decoder style architecture. Inspired by Galileo, we use a flexible patch-embedding layer [5, 50]. However, instead of doing that confusing pseudo-inverse stuff from FlexiViT we keep the actual projection weights the same size and resize the input image to mimic changing the patch size. It's probably basically equivalent.

Once the input is in token space, OlmoEarth adds in a 2D sincos positional embedding, a sinusoidal temporal embedding, and a learnable modality embedding to each token. During training, some tokens are masked out of the input, otherwise all tokens are passed to the encoder transformer which performs full self-attention

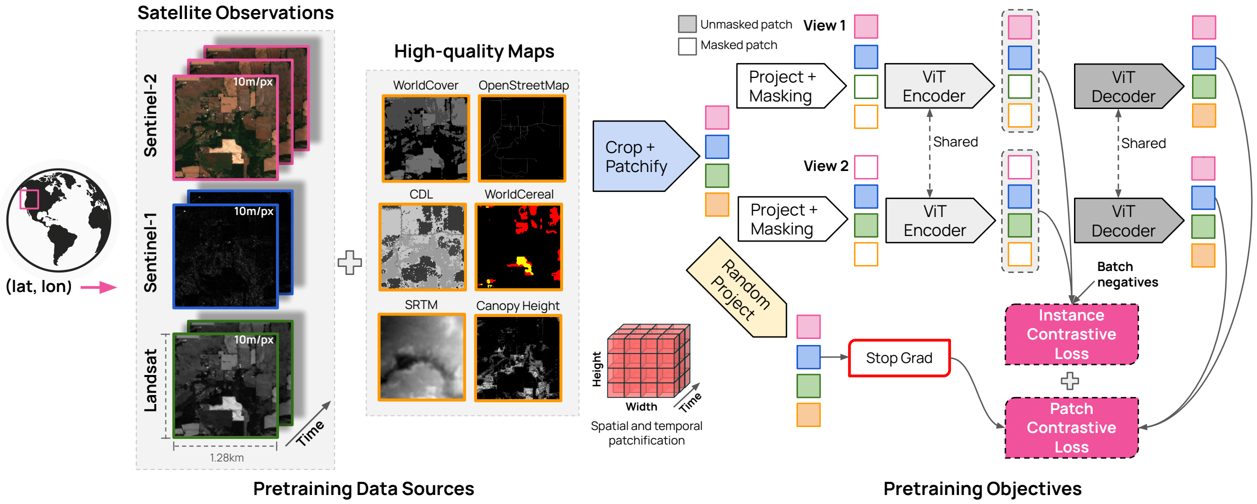


**Figure 3** We train OlmoEarth with a combination of satellite observations and high-quality maps. After tokenizing these inputs, we: (1) apply a modality-aware masking strategy to define which tokens are inputs vs. targets, (2) pass the target tokens through fixed random projections to construct targets, (3) pass the input tokens through our learned encoders, and then (4) through a decoder which predicts the target tokens and (5) apply a modality-aware patch discrimination loss between the predicted and target tokens. Steps 1-5 are applied twice on the same data to then (6) apply an instance contrastive loss over the aggregated tokens per instance.

across space, time, and between modalities.

| Architecture | Depth | Dim | Heads | Parameters |
|---|---|---|---|---|
| ViT Nano | 4 | 128 | 8 | 1.4M |
| ViT Tiny | 12 | 192 | 3 | 6.2M |
| ViT Base | 12 | 768 | 12 | 90M |
| ViT Large | 24 | 1024 | 16 | 300M |

**Table 1** ViT encoder architectures and number of parameters for the four OlmoEarth model sizes.

We train four different encoder sizes based on standard Vision Transformer sizes, see Table 1. For each model size, the decoder has the same feature dimension and number of heads but only a depth of 4. We design a smaller decoder so that the encoder does the majority of the modeling.

During training the decoder represents the masked portions of the input with a learned $<\mathrm{MASK}>$ token added to the appropriate positional, temporal, and modality embeddings. The decoder cross-attends to these tokens with the visible tokens from the encoder. It then predicts the latents for the masked tokens.

## 2.3 Masking

OlmoEarth uses a modality-aware masking strategy. For every example the masking strategy selects some bandsets to be encoded and also some to be decoded, non-exclusively. Thus every bandset falls into one of four categories:

- **Not selected**: Ignored for this example.
- **Encode only**: Randomly masked, input to encoder.
- **Decode only**: Used as target for decoder.
- **Encode and decode**: Randomly masked, input to encoder, masked tokens used as targets for decoder.

This masking strategy re-frames the problem slightly from reconstructing data that has been partially masked to reconstructing missing bandsets from partial views of other bandsets. When all bandsets are encoded and decoded we find the task is too easy. Masked tokens in a bandset will likely have other tokens in the same bandset that are highly correlated with them that are visible in the input, tokens nearby spatially or temporally. Training in this easier paradigm requires using very high masking ratios (i.e. masking out 90% of the input) to get decent results. Masking some bandsets entirely makes the problem harder and allows more balanced masking ratios.

OlmoEarth trains on both observations and maps but at inference time we only use observations. Maps can change over time–indeed downstream tasks are often detecting this kind of change–so we only rely on observations for inference. Thus during training our masking strategy never encodes map data, it only ever decodes it. While observations can fall into any of the above four categories, maps will only be "decode only" or "not selected".

## 2.4 Latent MIM Lite

During training OlmoEarth predicts reconstructions of the masked input in latent space. We use a randomly initialized, frozen projection layer for each modality to project masked patches in the input into token space. Thus OlmoEarth performs Latent Masked Image Modeling of Linear, Invariant Token Embeddings (Latent MIM Lite).

Randomly projecting raw input data extracts valuable features both from a theoretical and practical standpoint [6, 7, 43]. Thus our predictions are operating in a true latent space of our input data. However, because we use a fixed target encoder we avoid the representation collapse common in Latent MIM-style training. While it's possible this approach is too simplistic in more diverse domains like natural image processing, empirical results show a clear benefit in our domain of Earth observation data.

Latent MIM Lite allows us to unify supervised and self-supervised training under the same architecture. We project each modality, whether observations or maps, through a frozen random projection into token space. Loss is calculated the same for both types of modalities. We do not need to add on specific predictor heads for supervised data or adjust our training strategy or loss. In our ablations we see this approach gives strong results in a purely self-supervised setting and also benefits from additional supervised data.

Other models like Galileo and Terramind train on both supervised and unsupervised data however they treat supervised maps as a valid input to the model [25, 50]. This means their encoders must learn to model these map modalities as input and during training may use map modalities to predict observations or other map modalities. While this also unifies supervised and semi-supervised training, we theorize that our approach simplifies learning for the encoder while maintaining the benefits of training with supervised data. In our evaluations we see improved performance over these models on most tasks.

### 2.4.1 Modality Patch Discrimination

Masked image modeling in pixel space typically uses a reconstruction loss like Smooth L1. Latent MIM proposes using a contrastive loss (Patch Discrimination) instead of reconstruction loss to incentivize diversity in the latent space predictions. Patch discrimination loss frames token reconstruction as a classification task where we want the predicted token for a patch to be similar to the target token but dissimilar from other ground truth tokens for other patches. Patch discrimination uses cosine similarity to measure token similarity and cross entropy loss to contrast between positive and negative matches.

Typical patch discrimination contrasts a predicted token with all target tokens in the input. For image modeling, the target tokens from an image are encodings of different parts of the image so they are from the same distribution, making the contrastive task challenging. In OlmoEarth, different target tokens can come from different modalities or different time steps as well as different spatial locations.

Tokens from different modalities have very different distributions so distinguishing between them is easy. Yet there are so many tokens from other modalities that a significant amount of the loss comes from these "easy" negatives. We find eliminating easy negatives and only contrasting tokens with targets from the same modality gives a substantial performance increase.

### 2.4.2 Instance Contrastive Loss

Patch discrimination loss operates on the local representations generated by the encoder and decoder but many tasks (like classification) require a global understanding of the input region. Some foundation models use a single < CLASS > token to represent this global information. Instead we opt to pool information globally over all modalities, timesteps, and locations for an input. To generate a global representation for an input we run the OlmoEarth encoder and average pool the output tokens.

Tokens encoded from the same modality share semantics but tokens from different modalities may look very different from each other. We want to be able to average tokens from all modalities together and get a sensible global representation of an input. Thus we use a contrastive loss on the pooled representation from the encoder to encourage tokens to exist in a common representation space and behave well when pooled.

We want both positive and negative samples for our contrastive loss so we take an approach similar to SimCLR [10] and encode two versions of the same input, contrasting these two versions as positive examples with the rest of the batch as negative examples. However, instead of using different data augmentation to generate the two samples we use different random masking.

We run random masking twice, then encode both batches with our encoder, pool the resulting tokens, and apply contrastive loss to the pooled representations. We run the decoder twice, decoding masked portions for both images and calculate the modality patch discrimination loss. A scalar multiple controls the contribution of instance contrastive loss to modality patch discrimination loss. For experiments in this paper we scale the instance contrastive loss by 0.1.

# 3 Experiments

We extensively evaluate OlmoEarth on both standard research benchmarks and real-world downstream tasks from partner organizations. Following standard practice in remote sensing foundation models we evaluate both kNN/linear probe performance with a frozen encoder and full fine-tuning performance [16, 39, 50].

To get as comprehensive an evaluation as possible we import other top performing foundation models into our evaluation framework and evaluate them as well so they are directly comparable [2, 4, 11, 15, 16, 25, 44, 46, 49, 50, 54, 55]. We use the same training recipes for each foundation model but sweep a variety of hyperparameters to find the best performance for each model on each task. We leave evaluations blank for models that do not support particular modalities. We also do not fine-tune some large models on partner tasks due to compute and time limitations.

## 3.1 Pretraining

We pretrain OlmoEarth on our pretraining dataset described in 2.1 using Latent MIM Lite. We use AdamW optimization with a base learning rate of $1 \times 10^{-4}$, weight decay of 0.02, batch size of 512, linear learning rate warm-up of 8000 steps, cosine annealing of learning rate by 0.1 over a total of $667,200$ steps. Due to memory constraints we use a micro-batch size of 32 so the pooled contrastive loss is only applied over these 32 examples, not the full batch of 512.

During training OlmoEarth uses a random effective patch size in the range $\{1 \ldots 8\}$ and takes a random square crop from the input with side length in tokens in the range $\{1 \ldots 12\}$. Thus, along the spatial dimension the smallest input is a $1 \times 1$ pixel region in the input with a patch size of 1, and the largest input is $96 \times 96$ pixel region in the input with a patch size of 8. Along the temporal dimension, our model processes between 3 and 12 timesteps. During training our model processes around 100 billion tokens.

## 3.2 Research Benchmarks and Partner Tasks

We evaluate on a variety of common research benchmarks for classification and segmentation across single and multiple sensor modalities. Our evaluations include all seven Sentinel-2 and Landsat benchmarks from GEO-Bench [30]: m-bigearthnet, m-so2sat, m-brick-kiln, m-forestnet, m-eurosat, m-cashewplant, and m-SA-crop-type. We also evaluate on the classification benchmarks BreizhCrops [42] and CropHarvest [48] and the segmentation benchmarks PASTIS [17], MADOS [29], and Sen1Floods11 [8].

|  |  | m-bigearthnet | m-so2sat | m-brick-kiln | m-forestnet | m-eurosat | BreizhCrops | CropHarvest-PRC | CropHarvest-PRC | CropHarvest-PRC | CropHarvest-Togo | CropHarvest-Togo | CropHarvest-Togo | m-cashewplant | m-SA-crop-type | PASTIS | PASTIS | MADOS | Sen1Floods11 | AWF | AWF | AWF | Nandi | Nandi | Nandi |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
|  | Modalities | S2 | S2 | S2 | L8 | S2 | S2 | S1 | S2 | S1,S2 | S1 | S2 | S1,S2 | S2 | S2 | S1 | S2 | S2 | S1 | L8 | S1 | S2 | L8 | S1 | S2 |
|  | Time series | ✘ | ✘ | ✘ | ✘ | ✘ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✘ | ✘ | ✔ | ✔ | ✘ | ✘ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ |
|  | Method | kNN | kNN | kNN | kNN | kNN | LP | LP | LP | LP | LP | LP | LP | LP | LP | LP | LP | LP | LP | kNN | kNN | kNN | kNN | kNN | kNN |
| Model | Metric | $\mu$F1 | Acc. | Acc. | Acc. | Acc. | Acc. | Acc. | Acc. | Acc. | Acc. | Acc. | Acc. | mIOU | mIOU | mIOU | mIOU | mIOU | mIOU | Acc. | Acc. | Acc. | Acc. | Acc. | Acc. |
| Anysat | ViT Base | 54.5 | 36.5 | 84.5 | 34.0 | 80.4 | 62.7 | 57.0 | 74.1 | 72.3 | 69.6 | 78.4 | 74.5 | 24.6 | 27.2 | 24.2 | 41.9 | 41.3 | 77.8 | 60.0 | 60.0 | 64.0 | 47.4 | 20.5 | 55.8 |
| Clay | ViT Large | 48.8 | 38.7 | 90.8 | 40.3 | 86.3 | 57.0 | 56.7 | 66.5 | 63.4 | 78.4 | 67.0 | 67.3 | 30.8 | 23.1 | 19.9 | 22.6 | 47.5 | 78.9 | 59.5 | 61.5 | 56.5 | 52.2 | 23.4 | 57.0 |
| CopernicusFM | ViT Base | 64.6 | 50.3 | 85.9 | - | 84.7 | 65.5 | 55.1 | 72.8 | 74.4 | 77.5 | 75.8 | 70.6 | 32.2 | 28.4 | 15.9 | 32.1 | 63.9 | 77.6 | - | 59.0 | 67.0 | - | 24.4 | 59.8 |
| CROMA | ViT Base | 61.3 | 51.3 | 92.0 | - | 84.6 | 69.0 | 56.8 | 74.3 | 75.1 | 76.8 | 80.7 | 76.5 | 24.9 | 30.3 | 26.3 | 44.7 | 60.4 | 78.8 | - | 67.5 | **79.0** | - | 24.6 | 68.2 |
| CROMA | ViT Large | 59.2 | 48.2 | 91.7 | - | 85.8 | 68.2 | 56.2 | 72.7 | 71.0 | 79.4 | 76.5 | 81.0 | 27.0 | 30.4 | 25.9 | 42.7 | 66.4 | 78.8 | - | 62.5 | 71.0 | - | 26.1 | 68.2 |
| DINOv3 | ViT Base | 51.0 | 47.1 | 91.3 | 43.3 | 86.6 | 31.3 | - | 64.5 | - | - | 67.0 | - | 23.5 | 26.7 | - | 18.1 | 53.5 | - | 48.5 | - | 61.0 | 42.5 | - | 54.1 |
| DINOv3 | ViT Large | 55.8 | 45.3 | 89.9 | 46.0 | 84.0 | 31.3 | - | 66.1 | - | - | 68.3 | - | 24.5 | 26.1 | - | 17.4 | 52.4 | - | 43.5 | - | 60.5 | 39.9 | - | 49.8 |
| DINOv3 | ViT Huge+ | 57.0 | 45.7 | 88.2 | 46.9 | 86.1 | 31.3 | - | 68.7 | - | - | 68.3 | - | 25.1 | 26.7 | - | 17.4 | 48.1 | - | 47.5 | - | 54.0 | 43.5 | - | 55.4 |
| DINOv3 | ViT 7B | 60.8 | 46.6 | 91.3 | 48.0 | 85.2 | 31.3 | - | 68.7 | - | - | 70.9 | - | 34.3 | 27.9 | - | 21.1 | 52.2 | - | 47.5 | - | 57.0 | 44.7 | - | 56.7 |
| DINOv3 Sat | ViT Large | 60.2 | 44.0 | 91.4 | 44.2 | 89.2 | 31.3 | - | 70.1 | - | - | 68.6 | - | 32.4 | 28.5 | - | 22.5 | 57.5 | - | 42.5 | - | 69.5 | 35.5 | - | 48.4 |
| DINOv3 Sat | ViT 7B | 61.6 | 50.1 | 91.4 | 47.0 | 91.3 | 31.3 | - | 72.2 | - | - | 71.9 | - | **54.1** | **31.7** | - | 26.3 | 59.7 | - | 49.5 | - | 68.5 | 31.8 | - | 42.5 |
| Galileo | ViT Nano | 55.0 | 53.7 | 90.9 | - | 89.4 | 66.3 | 60.9 | 74.4 | 72.4 | 75.2 | 70.3 | 78.1 | 21.2 | 19.5 | 19.2 | 19.1 | 53.1 | 78.6 | - | 65.5 | 67.5 | - | 24.4 | 64.2 |
| Galileo | ViT Tiny | 55.8 | 53.1 | 87.5 | - | 89.1 | 66.7 | 55.7 | 80.3 | 79.3 | 69.9 | 78.8 | 77.1 | 23.6 | 21.5 | 23.4 | 27.7 | 61.1 | 78.6 | - | 65.5 | 71.0 | - | 25.0 | 66.7 |
| Galileo | ViT Base | 58.3 | 55.7 | 91.1 | - | 92.8 | 69.7 | **60.9** | **81.9** | 79.3 | 67.3 | 80.1 | 77.5 | 28.9 | 25.3 | 28.0 | 39.6 | **68.4** | 79.4 | - | 66.5 | 72.5 | - | **26.8** | 67.3 |
| Panopticon | ViT Base | **64.9** | 60.5 | 92.9 | **52.3** | 95.2 | 57.7 | 55.9 | 75.9 | 75.6 | 72.2 | 72.9 | 76.5 | 32.7 | 27.3 | 23.7 | 30.2 | 66.1 | 78.0 | 66.0 | 65.0 | 71.5 | 60.4 | 22.9 | 65.2 |
| Presto | ViT Nano | - | - | - | - | - | 60.9 | 59.3 | 74.3 | 76.6 | 78.4 | 81.4 | 72.5 | - | - | 16.3 | 28.2 | - | - | - | 60.5 | 53.5 | - | 25.2 | 57.6 |
| Prithvi v2 | ViT Large | 51.6 | 34.7 | 89.7 | 37.9 | 82.2 | 66.2 | - | 67.6 | - | - | 71.9 | - | 46.8 | 24.7 | - | 37.2 | 50.9 | - | 57.0 | - | 49.0 | 57.7 | - | 52.8 |
| Prithvi v2 | ViT Huge | 51.0 | 34.0 | 90.1 | 41.4 | 81.2 | 66.1 | - | 69.5 | - | - | 69.9 | - | 45.8 | 26.6 | - | 37.5 | 52.2 | - | 59.0 | - | 55.0 | 57.9 | - | 55.9 |
| Satlas | Swin Base | 52.3 | 44.5 | 83.0 | 36.9 | 82.2 | 64.6 | 57.4 | 71.3 | - | 76.8 | 75.8 | - | 30.6 | 24.0 | 10.5 | 14.4 | 30.2 | 72.9 | 57.5 | 52.0 | 62.0 | 61.5 | 25.1 | 60.2 |
| TerraMind | ViT Base | 63.9 | 46.7 | 91.9 | - | 85.6 | 66.4 | 57.0 | 75.1 | 75.6 | 74.2 | 74.8 | 77.5 | 46.0 | 30.4 | 22.7 | 40.9 | 66.0 | 78.7 | - | 66.0 | 69.5 | - | 24.7 | - |
| TerraMind | ViT Large | 63.9 | 47.4 | 92.2 | - | 90.0 | 68.2 | 56.2 | 74.7 | 72.0 | 75.2 | 77.8 | 75.2 | 50.4 | 31.2 | 22.3 | 41.3 | 67.5 | 78.4 | - | 62.0 | 67.0 | - | 23.3 | 65.3 |
| TESSERA |  | - | - | - | - | - | - | - | - | 72.2 | - | - | 81.0 | - | - | - | - | - | - | - | - | - | - | - | - |
| OlmoEarth | ViT Nano | 59.5 | 54.3 | **96.2** | 38.8 | 89.9 | 64.1 | 58.0 | 79.5 | 74.3 | 73.5 | 81.7 | **83.7** | 25.5 | 23.6 | 18.1 | 35.0 | 55.2 | 78.2 | 73.0 | 61.5 | 69.5 | 57.7 | 24.8 | 67.4 |
| OlmoEarth | ViT Tiny | 59.4 | 61.8 | 92.0 | 40.5 | 91.6 | 64.0 | 58.3 | 78.6 | **80.3** | 75.8 | 85.6 | 82.4 | 24.7 | 23.2 | 21.4 | 40.1 | 58.6 | 78.5 | 75.5 | 64.0 | 76.0 | 60.7 | 24.7 | 69.0 |
| OlmoEarth | ViT Base | 62.4 | 67.7 | 93.3 | 41.9 | 94.7 | **70.9** | 56.8 | 73.4 | 75.4 | **80.1** | **87.3** | 82.0 | 32.3 | 28.9 | 29.7 | 50.6 | 67.2 | 79.2 | **77.0** | **68.5** | 77.5 | **67.9** | 26.5 | **74.7** |
| OlmoEarth | VIT Large | 62.0 | **68.2** | 93.4 | 41.6 | **96.3** | 70.7 | 56.6 | 74.1 | 76.1 | 67.6 | 78.1 | 79.7 | 30.9 | 28.5 | **30.6** | **51.8** | 66.4 | **79.8** | 76.0 | 66.5 | 73.0 | 66.4 | 26.2 | 73.6 |

**Table 2** kNN/Linear probe results on research benchmarks and real-world tasks from our partners. We run kNN on single time-step classification tasks and linear probing on all other tasks. We sweep across data normalization strategies, feature pooling, and learning rate (for linear probing) and report the test set result for the best validation set performance. Not all models can run on all tasks due to incompatible input modalities. OlmoEarth has consistently strong performance and is the best on 15 out of 24 tasks.

|  |  | m-bigearthnet | m-so2sat | m-brick-kiln | m-forestnet | m-eurosat | m-cashewplant | m-SA-crop-type | PASTIS | MADOS | Sen1Floods11 | AWF | AWF | GEA North Africa | Forest Loss Driver | Live Fuel Moisture Content | Live Fuel Moisture Content | Mangrove | Mangrove | Marine Infrastructure | Marine Infrastructure | Nandi | Nandi | Vessel Detection | Vessel Detection | Vessel Detection | Vessel Length | Vessel Type | Solar Farm Detection | Solar Farm Detection |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
|  | Modalities | S2 | S2 | S2 | L8 | S2 | S2 | S2 | S2 | S2 | S1 | S2 | S2, S1 | S2 | S2 | S2 | S2, S1 | S2 | S2, S1 | S2 | S2, S1 | S2 | S2, S1 | L8 | S1 | S2 | S2 | S2 | S2 | S1, S2 |
|  | Time series | ✘ | ✘ | ✘ | ✘ | ✘ | ✘ | ✘ | ✔ | ✘ | ✘ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✘ | ✘ | ✘ | ✘ | ✘ | ✔ | ✔ |
| Model | Metric | $\mu$F1 | Acc. | Acc. | Acc. | Acc. | mIOU | mIOU | mIOU | mIOU | mIOU | Acc. | Acc. | Acc. | Acc. | L1 | L1 | Acc. | Acc. | F1 | F1 | Acc. | Acc. | F1 | F1 | F1 | L1 | Acc. | mIoU | mIoU |
| Anysat | ViT Base | 68.4 | 56.7 | 98.7 | 51.6 | 95.9 | 80.4 | 34.2 | 60.9 | 63.3 | 77.4 | 78.0 | 83.0 | 59.3 | 84.6 | 19.1 | 19.4 | 96.9 | 97.1 | 52.7 | 5.2 | 78.4 | 76.7 | - | - | - | 73.4 | 43.5 | 82.6 | 79.7 |
| Clay | ViT Large | 65.7 | 61.3 | 98.7 | 49.2 | 95.8 | 73.9 | 33.4 | 48.9 | 68.9 | 78.5 | 77.0 | - | 53.4 | 93.3 | 24.8 | - | 96.6 | - | 86.0 | - | 30.9 | - | 70.7 | **79.9** | 72.4 | 16.4 | 68.3 | 82.2 | - |
| CopernicusFM | ViT Base | 71.3 | 66.8 | 98.1 | - | 98.5 | 78.7 | 33.6 | 54.6 | 66.0 | 78.6 | 79.0 | 79.0 | 58.8 | 90.0 | 25.2 | 24.5 | 97.1 | 97.1 | 82.8 | 88.9 | 68.2 | 55.9 | - | 77.4 | 76.9 | 16.7 | 69.6 | 77.6 | 76.8 |
| CROMA | ViT Base | 69.5 | 59.1 | 98.7 | - | 95.6 | 46.4 | 34.8 | 56.3 | 66.6 | 79.4 | 76.5 | 75.5 | 57.5 | 93.2 | 24.3 | 24.0 | 96.4 | 96.3 | 86.1 | 84.3 | 62.2 | 67.4 | - | - | 70.0 | 19.8 | 64.4 | 79.5 | 79.5 |
| CROMA | ViT Large | 71.8 | 58.9 | 97.8 | - | 97.5 | 47.8 | 36.0 | 58.1 | 68.8 | 79.4 | 79.5 | - | 56.6 | 92.3 | 24.6 | 24.1 | 96.6 | 96.2 | - | - | 76.4 | - | - | - | - | - | - | 81.7 | - |
| DINOv3 Sat | ViT Large | 69.9 | 63.3 | 98.9 | **59.2** | 96.7 | 80.6 | 34.5 | 42.8 | 64.7 | - | 34.5 | - | 43.0 | 80.4 | 90.2 | - | 65.8 | - | 82.9 | - | 35.8 | - | - | - | 58.0 | 30.3 | 54.8 | 70.7 | - |
| Galileo | ViT Base | 69.2 | 64.7 | 98.3 | - | 97.8 | 78.8 | 35.7 | 61.2 | 71.9 | 79.7 | 81.0 | 81.5 | **62.9** | 95.1 | 20.1 | 18.7 | 97.3 | 97.5 | 85.5 | 88.0 | **81.9** | 81.9 | - | 78.7 | 75.4 | 16.4 | 73.0 | 83.1 | 85.1 |
| Panopticon | ViT Base | 69.3 | 65.4 | **99.0** | 56.0 | 98.2 | 79.7 | 33.4 | 54.4 | 72.8 | 79.1 | 75.5 | 78.5 | 54.3 | 96.4 | 24.5 | 23.7 | 97.1 | 97.4 | 86.4 | 88.6 | 65.2 | 69.5 | 74.9 | 76.7 | 76.8 | 17.7 | 69.4 | 81.8 | 79.3 |
| Prithvi v2 | ViT Huge | 70.6 | 64.7 | 98.2 | - | 96.8 | 81.1 | 38.8 | 58.6 | 69.3 | - | 80.0 | - | 60.6 | 92.4 | - | - | 97.2 | - | 84.8 | - | 77.1 | - | 71.1 | - | 74.8 | 17.4 | 68.2 | 84.1 | - |
| Satlas | Swin Base | 72.7 | 65.1 | 98.7 | 56.0 | 97.0 | 77.0 | 37.8 | 57.4 | 60.5 | 78.5 | 78.0 | - | 56.1 | 63.3 | 25.0 | 24.6 | 96.6 | - | **87.5** | - | 47.6 | - | - | - | 77.6 | 16.2 | 71.6 | 83.3 | - |
| TerraMind | ViT Base | 72.6 | 66.1 | 98.5 | - | 97.6 | 80.9 | 39.2 | 59.9 | 73.2 | 79.5 | 84.0 | 82.0 | 49.8 | 96.4 | 24.3 | 23.8 | **97.7** | 96.8 | 84.0 | 87.8 | 66.1 | 79.3 | - | 79.6 | 74.7 | 18.1 | - | 83.5 | 82.1 |
| TerraMind | ViT Large | **74.0** | 65.4 | 98.1 | - | 97.8 | **81.3** | **41.1** | 60.9 | 71.5 | 79.5 | 81.5 | - | 51.1 | 93.9 | 24.5 | 24.5 | 96.5 | 96.9 | - | - | 66.1 | - | - | - | - | - | - | 83.0 | - |
| OlmoEarth (Random Init) | ViT Base | 61.0 | 48.9 | 94.7 | 41.7 | 80.3 | 43.0 | 27.5 | 43.9 | 45.6 | 77.0 | 62.5 | - | 52.9 | 52.7 | 20.9 | 20.5 | 96.3 | 96.4 | - | - | 60.6 | 56.1 | - | - | - | - | - | 74.1 | 70.3 |
| OlmoEarth | ViT Nano | 66.8 | 61.5 | 98.0 | 50.3 | 95.3 | 39.5 | 35.4 | 53.0 | 60.6 | 78.8 | 82.5 | 82.5 | 61.1 | 96.0 | 20.4 | 19.7 | 97.4 | 97.4 | 86.8 | 87.8 | 75.6 | 74.8 | 70.2 | 75.5 | 75.0 | 17.1 | 72.0 | 82.1 | 79.6 |
| OlmoEarth | ViT Tiny | 69.6 | 63.5 | 98.7 | 53.2 | 97.1 | 72.5 | 38.5 | 60.3 | 71.5 | 79.7 | 85.0 | 85.5 | 60.6 | 97.7 | 19.8 | 19.2 | 97.6 | 97.7 | 85.6 | 89.3 | 78.2 | 76.4 | 74.4 | 76.9 | 77.6 | 15.8 | 73.5 | 85.2 | 85.2 |
| OlmoEarth | ViT Base | 72.0 | **68.6** | 98.6 | 51.2 | **98.7** | 79.8 | 39.6 | 64.3 | 77.8 | 79.8 | **87.0** | **86.0** | 62.4 | 97.1 | **18.5** | **17.9** | 97.6 | **97.9** | 86.3 | **89.6** | 81.8 | **82.2** | **75.4** | 79.2 | **78.8** | **15.4** | **74.6** | **85.4** | **86.7** |
| OlmoEarth | ViT Large | 72.4 | 68.1 | 98.6 | 52.7 | 98.5 | 80.6 | 40.8 | **66.3** | **81.8** | **79.8** | 84.5 | - | 58.8 | **97.9** | 19.9 | 18.5 | 97.6 | 97.6 | - | - | 81.0 | - | - | - | - | - | - | 84.2 | - |

**Table 3** Fine-tuning results on research benchmarks (left) and partner tasks (right). We train all models with the same recipe and report test set results for the model checkpoint with the best validation set performance. Some models are only compatible with a subset of tasks. Due to resource constraints, we do not fine-tune large models on all tasks. OlmoEarth is best on 19 out of 29 tasks.

While developing OlmoEarth, we partnered with several organizations who are already using or want to use remote sensing data for environmental, climate, or research tasks. These organizations provided labeled data across a variety of domains for our evaluations, offering critical insights into how models perform on real-world tasks. For example, we partnered with the African Wildlife Foundation (AWF) to map land use and land cover in southern Kenya. We pair these tasks with different combinations of Sentinel-1, Sentinel-2, and Landsat observations.

## 3.3 kNN and Linear Probing

For evaluations without fine-tuning we extract embeddings from the train, validation, and test set and apply either a kNN model for single time step classification or a linear probe model for segmentation and multi-temporal classification. For OlmoEarth we use a patch size of 4 except we sweep patch size for applicable models on m-Cashew Plant (See discussion in Appendix). For external models we use recommended settings for patch size and resize input data to that model's pretraining size following [13]. For models that do not support time series data we input each time step separately. We sweep pooling method for the resulting embeddings across time (mean vs max). We also sweep normalization statistics (computed during pretraining vs. on the evaluation set).

We run kNN with $k = 20$ using cosine similarity, and follow standard evaluation practices [20, 50]. For models that output a <CLASS> embedding token we use that as the embedding for the whole image, otherwise we average across resulting tokens.

We run linear probing on the output embeddings, training for 50 epochs. We sweep across a variety of learning rates for each model $\{1 \times 10^{-4}, 5 \times 10^{-4}, 1 \times 10^{-3}, 5 \times 10^{-3}, 1 \times 10^{-2}, 5 \times 10^{-2}, 1 \times 10^{-1}, 5 \times 10^{-1}\}$ and report the test results for the highest validation set performance.

## 3.4 Fine-Tuning

For fine-tuning evaluations, for each model, we take the encoder and add a decoder that makes classification, regression, semantic segmentation, or object detection predictions. Our fine-tuning recipe freezes encoder parameters for 20% of the epochs, only training the added decoder layers, and then unfreezes and fine-tunes the full model for the remaining epochs. We use AdamW optimization with a plateau scheduler that reduces the learning rate by a factor of 0.2 after 2 epochs without improvement on the validation set and a 10 epoch cooldown after reduction.

For fine-tuning on research benchmarks, the decoder is a single-layer linear probe; for classification tasks, it makes a prediction using embeddings pooled over the image, and for segmentation tasks, it makes a prediction using embeddings pooled temporally (when applicable) at each spatial patch. We sweep learning rates for each model over $\{1 \times 10^{-4}, 5 \times 10^{-4}, 1 \times 10^{-3}\}$.

For fine-tuning on partner tasks, the decoder is:

- **Classification:** 3-layer MLP.
- **Segmentation:** Transposed convolutional layers, or U-Net decoder for multi-scale encoders [41].
- **Object Detection:** Faster R-CNN head, with an FPN for multi-scale encoders [32, 40].

We use a learning rate of $10^{-4}$ for all tasks, except Nandi, for which some models exhibit unstable learning and we sweep over $\{10^{-4}, 10^{-5}\}$.

## 3.5 Results

For kNN/LP evaluations, OlmoEarth is the best performing on 11 of 18 research benchmarks and 4 of 6 partner tasks. For fine-tuning evaluations, OlmoEarth is the best performing on 5 of 10 research tasks and 14 of 19 partner tasks. OlmoEarth gets consistently high performance except in a couple instances.

OlmoEarth Large does not always outperform OlmoEarth Base, and for embedding-based pixel time series tasks it is significantly worse. This may reflect that we explore the training recipe for the Base model more

| | m-so2sat | m-eurosat | PASTIS |
|---|---|---|---|
| Full Latent MIM* | 32.2 | 68.4 | 7.9 |
| Latent MIM Lite | 42.2 | 87.2 | 35.2 |
| + Modality Masking | 53.6 | 90.2 | 46.6 |
| + Modality Patch Disc | 55.3 | 91.5 | 48.1 |
| + Contrastive Loss | 56.8 | 92.3 | 49.0 |
| + Maps | **62.4** | **92.9** | **50.7** |

**Table 4** Development path of the OlmoEarth base model showing effect of adding our various contributions starting from a Latent MIM approach. *Full Latent MIM collapsed during training.

than Large. Terramind and CROMA Base models often outperform Large models on many tasks so this may reflect the challenges of scaling Earth observation models.

Other notable models include Panopticon for strong performance on embedding tasks and Terramind on fine-tuning tasks. DINOv3 shows good results for tasks that mainly require visual information but lags behind specialized models on tasks where temporal understanding is critical. Galileo shows strong performance on many benchmarks, especially agriculture-related tasks.

## 3.6 Ablations

We based OlmoEarth off of Latent MIM self-supervised training and iterated on various modifications, keeping the best. Table 4 shows our development process, starting from standard Latent MIM, random masking, patch discrimination loss only, and no maps data. Models in the table are trained according to training recipe in Subsection 3.1 but only for 140,000 steps. Results are shown for kNN and LP on the validation set of three benchmarks. During development we ran a subset of our evaluations in our “in-loop evals” but saw that improvements on a representative subset carried over to the full evaluation.

We see the Latent MIM model gets poor performance due to representation collapse. Switching to Latent MIM Lite substantially boosts performance. Further modifications show increased performance for all tasks. We conduct additional ablations in Appendix C.

## 3.7 Environmental Impact

Following recent work on environmental impact analysis of language modeling [18, 35, 37] we estimate total energy use, carbon emissions, and water consumption from training OlmoEarth in Table 5. Similar to other environmental impact estimates this should be viewed as a lower bound as it does not account for hardware manufacturing, transportation, etc.

| | | | | Energy | Carbon | Water |
|---|---|---|---|---|---|---|
| Model | Stage | Hardware | GPU Hrs | (kWh) | ($tCO_2eq$) | (kL) |
| OlmoEarth Nano | Pretraining | H100 | 1,149 | 195 | 0.08 | 0.30 |
| OlmoEarth Tiny | Pretraining | H100 | 1,149 | 205 | 0.08 | 0.32 |
| OlmoEarth Base | Pretraining | H100 | 2,989 | 803 | 0.32 | 1.24 |
| OlmoEarth Large | Pretraining | B200 | 5,240 | 1,933 | 0.77 | 2.99 |
| OlmoEarth Nano | Fine-tuning | – | 647 | 186 | 0.07 | 0.29 |
| OlmoEarth Tiny | Fine-tuning | – | 723 | 261 | 0.10 | 0.40 |
| OlmoEarth Base | Fine-tuning | – | 1,224 | 685 | 0.27 | 1.06 |
| OlmoEarth Large | Fine-tuning | – | 58 | 39 | 0.02 | 0.06 |
| **Total** | Overall | – | 13,179 | 4,307 | 1.72 | 6.67 |

**Table 5** Approximate environmental impact of pretraining and fine-tuning OlmoEarth. Metrics for fine-tuning OlmoEarth Nano, Tiny, and Base include research benchmarks and partner tasks. Metrics for fine-tuning OlmoEarth Large only include research benchmarks.

We train all of our models in a single data center on NVIDIA H100 and B200 GPUs. We calculate the total GPU power required for a training run by tracking actual GPU power utilization every ~25ms to calculate a weighted average of power consumption throughout training. We then multiply this by the power usage efficiency (PUE) factor for our data center, according to our provider, and then we multiply this final GPU power usage amount by either the carbon intensity of the grid or the water usage efficiency factor of the data center to calculate total carbon emissions and water consumption, respectively.

The total energy usage during training (4,307 kWh) could power the average U.S. household for 5 months. The total carbon emissions are equivalent to an economy ticket on a flight from Seattle to Portugal.

# 4 Related Work

**Pretraining** for remote sensing models initially focused on contrastive approaches [3, 26, 33]. Recently masked modeling has taken over as the dominant paradigm, similar to language and vision [14, 21]. Early approaches to remote sensing pretraining directly reconstructed the masked pixel values [12, 39, 49]. Following research in natural imagery [1, 44, 56], remote sensing focuses more on reconstruction in latent space. Latent approaches work well [2, 50, 54] but have documented instabilities [1, 34].

**TerraMind** avoids instability by using a frozen tokenizer during pretraining. For image modalities they train a quantized autoencoder and use the encoder as their frozen tokenizer during multimodal masked modeling.

**Precomputed embeddings** offer an alternative approach for accessibility [9, 15] but still require expertise to retrieve and use. Best results may still require training a decoder on top of the embeddings. Precomputed embeddings also limit flexibility; both AEF and TESSERA generate annualized embeddings making real-time or sub-annual predictions impossible. OlmoEarth embeddings match or outperform AEF embeddings on partner tasks, and full fine-tuning enables even better results (Appendix Table 7).

# 5 Discussion

We want OlmoEarth to have a positive impact on the world. Toward that end we release it as part of the OlmoEarth Platform, an end-to-end, open solution for Earth observation tasks. OlmoEarth Platform enables partner organizations to use the latest, best foundation models in their work on the environment, conservation, food security, and more. Organizations like Global Mangrove Watch, Global Ecosystem Atlas, and the International Food Policy Research Institute are using OlmoEarth Platform for data curation and labeling, model fine-tuning, and inference.

## 5.1 Case Studies

**Global Mangrove Watch** maps and tracks the extent and health of coastal mangrove forests. Mangrove forests sequester carbon, protect the coastline from erosion, and provide a habitat for little fishies. GMW uses a random forest model with a 95.3% F1 score to generate maps on a yearly cadence, and only covering about half of relevant coastal regions. Using OlmoEarth Platform we fine-tune a OlmoEarth model using their data up to an F1 score of 98.1%. The OlmoEarth Platform can run inference on a monthly cadence to generate new maps, or on a rolling basis to detect change faster.

**Global Ecosystem Atlas** is building a comprehensive map of the world's ecosystems [28]. For the last 3 months they have been using OlmoEarth Platform to label more than 15,000 data points. OlmoEarth Platform allows them to partition areas of interest, generate points to label, assign those points to labelers, review the results, and export the data or fine-tune a model directly in the platform. With a subset of the data from North Africa we fine-tune a model that achieves state-of-the-art accuracy and run inference to generate new ecosystem maps. Humans can review the results to feed better labels back into the training pipeline.

## 5.2 Downstream Risks

The power and versatility of OlmoEarth also bring risks. We release OlmoEarth under an open license designed to address some of these risks by allowing the free use, modification, and sharing of the model weights, datasets, and associated code while restricting use for military, defense-related, and extractive industry applications.

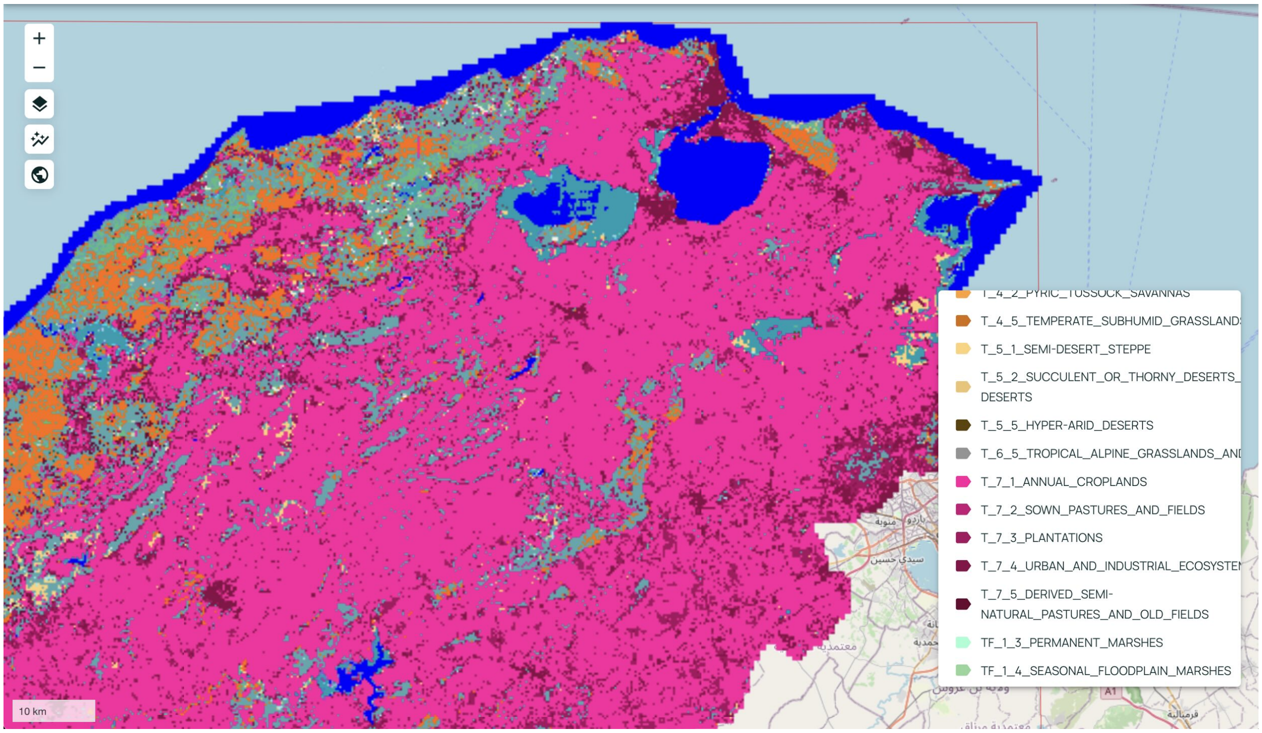

+
−
T_4_2_PYRIC_TUSSOCK_SAVANNAS
T_4_5_TEMPERATE_SUBHUMID_GRASSLAND
T_5_1_SEMI-DESERT_STEPPE
T_5_2_SUCCULENT_OR_THORNY_DESERTS_
DESERTS
T_5_5_HYPER-ARID_DESERTS
T_6_5_TROPICAL_ALPINE_GRASSLANDS_AN
T_7_1_ANNUAL_CROPLANDS
T_7_2_SOWN_PASTURES_AND_FIELDS
T_7_3_PLANTATIONS
T_7_4_URBAN_AND_INDUSTRIAL_ECOSYSTE
T_7_5_DERIVED_SEMI-
NATURAL_PASTURES_AND_OLD_FIELDS
TF_1_3_PERMANENT_MARSHES
TF_1_4_SEASONAL_FLOODPLAIN_MARSHES
10 km
A1

**Figure 4** Results of a fine-tuned ecosystem classification model in the OlmoEarth Platform. Users can label data, fine-tune models, and run inference to generate maps all in the OlmoEarth Platform.

### 5.3 The Future

We plan to add climate and weather data and forecasting to the OlmoEarth model to help with tasks like wildfire prediction and crop yield forecasting. Expanding to this kind of data will require handling a wider variety of input resolutions both spatially (from meters to kilometers) and temporally (from days to years).

We also plan to add non-geospatial data to the model. Often data labeling for tasks like crop type mapping requires actually going to a location in person and looking at stuff. We'd like the model to be able to do that too. The ability to process geolocated natural images would expand OlmoEarth's ability to handle these fine-grained recognition tasks.

Ultimately we want to support and grow the community of partner organizations who bring incredible knowledge, expertise, and passion to this work. We plan to learn from our partners about what tools and capabilities they need and then improve OlmoEarth Platform to better help them. We hope OlmoEarth Platform can become a hub for data, models, training, and inference across a wide range of organizations working to solve the world's biggest problems.

## Acknowledgments

We wish to express deep gratitude to our early collaborators who shared data, expertise, and time to make these models successful for real-world, mission-critical applications: Amazon Conservation Association, African Wildlife Foundation, CGIAR/International Food Policy Research Institute (IFPRI), Global Mangrove Watch, Global Ecosystem Atlas, ITC University of Twente, NASA Jet Propulsion Laboratory (JPL), and NASA Harvest.

We would also like to thank the OLMo-core, Beaker, Comms, and Legal teams at Ai2 for their support, especially Pete Walsh, Dirk Groeneveld, Sam Skjonsberg, Tara Wilkins, Caroline Wu, Johann Dahm, David Albright, Kyle Wiggers, Jordan Steward, Crystal Nam, Will Smith, and Janice Dow.

# References

[1] Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15619–15629, 2023.

[2] Guillaume Astruc, Nicolas Gonthier, Clement Mallet, and Loic Landrieu. AnySat: An Earth observation model for any resolutions, scales, and modalities. *arXiv preprint arXiv:2412.14123*, 2024.

[3] Kumar Ayush, Burak Uzkent, Chenlin Meng, Kumar Tanmay, Marshall Burke, David Lobell, and Stefano Ermon. Geography-aware self-supervised learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10181–10190, 2021.

[4] Favyen Bastani, Piper Wolters, Ritwik Gupta, Joe Ferdinando, and Aniruddha Kembhavi. SatlasPretrain: A large-scale dataset for remote sensing image understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16772–16782, 2023.

[5] Lucas Beyer, Pavel Izmailov, Alexander Kolesnikov, Mathilde Caron, Simon Kornblith, Xiaohua Zhai, Matthias Minderer, Michael Tschannen, Ibrahim Alabdulmohsin, and Filip Pavetic. FlexiViT: One model for all patch sizes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14496–14506, 2023.

[6] Ella Bingham and Heikki Mannila. Random projection in dimensionality reduction: applications to image and text data. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 245–250, 2001.

[7] Avrim Blum. Random projection, margins, kernels, and feature-selection. In *International Statistical and Optimization Perspectives Workshop" Subspace, Latent Structure and Feature Selection"*, pages 52–68. Springer, 2005.

[8] Derrick Bonafilia, Beth Tellman, Tyler Anderson, and Erica Issenberg. Sen1Floods11: A georeferenced dataset to train and test deep learning flood algorithms for Sentinel-1. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 210–211, 2020.

[9] Christopher F Brown, Michal R Kazmierski, Valerie J Pasquarella, William J Rucklidge, Masha Samsikova, Chenhui Zhang, Evan Shelhamer, Estefania Lahera, Olivia Wiles, Simon Ilyushchenko, et al. Alphaearth foundations: An embedding field model for accurate and efficient global mapping from sparse label data. *arXiv preprint arXiv:2507.22291*, 2025.

[10] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[11] Clay. Clay Foundation Model - Clay Foundation Model. `https://clay-foundation.github.io/model/`, 2025.

[12] Yezhen Cong, Samar Khanna, Chenlin Meng, Patrick Liu, Erik Rozi, Yutong He, Marshall Burke, David Lobell, and Stefano Ermon. Satmae: Pre-training transformers for temporal and multi-spectral satellite imagery. *Advances in Neural Information Processing Systems*, 35:197–211, 2022.

[13] Isaac Corley, Caleb Robinson, Rahul Dodhia, Juan M. Lavista Ferres, and Peyman Najafirad. Revisiting pre-trained remote sensing model benchmarks: Resizing and normalization matters. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 3162–3172, 2024.

[14] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[15] Zhengpeng Feng, Sadiq Jaffer, Jovana Knezevic, Silja Sormunen, Robin Young, Madeline Lisaius, Markus Immitzer, James Ball, Clement Atzberger, David A Coomes, et al. Tessera: Temporal embeddings of surface spectra for earth representation and analysis. *arXiv preprint arXiv:2506.20380*, 2025.

[16] Anthony Fuller, Koreen Millard, and James Green. CROMA: Remote sensing representations with contrastive radar-optical masked autoencoders. *Advances in Neural Information Processing Systems*, 36, 2024.

[17] Vivien Sainte Fare Garnot and Loic Landrieu. Panoptic segmentation of satellite image time series with convolutional temporal attention networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4872–4881, 2021.

[18] Dirk Groeneveld, Iz Beltagy, Pete Walsh, Akshita Bhagia, Rodney Kinney, Oyvind Tafjord, Ananya Harsh Jha, Hamish Ivison, Ian Magnusson, Yizhong Wang, Shane Arora, David Atkinson, Russell Authur, Khyathi Raghavi Chandu, Arman Cohan, Jennifer Dumas, Yanai Elazar, Yuling Gu, Jack Hessel, Tushar Khot, William Merrill, Jacob Morrison, Niklas Muennighoff, Aakanksha Naik, Crystal Nam, Matthew E. Peters, Valentina Pyatkin, Abhilasha Ravichander, Dustin Schwenk, Saurabh Shah, Will Smith, Emma Strubell, Nishant Subramani, Mitchell Wortsman, Pradeep Dasigi, Nathan Lambert, Kyle Richardson, Luke Zettlemoyer, Jesse Dodge, Kyle Lo, Luca Soldaini, Noah A. Smith, and Hannaneh Hajishirzi. Olmo: Accelerating the science of language models, 2024.

[19] Group on Earth Observations (GEO). Global Ecosystems Atlas. https://globalecosystemsatlas.org, 2025.

[20] Matthew Gwilliam and Abhinav Shrivastava. Beyond supervised vs. unsupervised: Representative benchmarking and analysis of image representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9642–9652, 2022.

[21] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009, 2022.

[22] Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(7):2217–2226, 2019.

[23] Hans Hersbach, Bill Bell, Paul Berrisford, Shoji Hirahara, András Horányi, Joaquín Muñoz-Sabater, Julien Nicolas, Carole Peubey, Raluca Radu, Dinand Schepers, et al. The era5 global reanalysis. *Quarterly Journal of the Royal Meteorological Society*, 146(730):1999–2049, 2020.

[24] Jeremy Irvin, Hao Sheng, Neel Ramachandran, Sonja Johnson-Yu, Sharon Zhou, Kyle Story, Rose Rustowicz, Cooper Elsworth, Kemen Austin, and Andrew Y Ng. Forestnet: Classifying drivers of deforestation in indonesia using deep learning on satellite imagery. *arXiv preprint arXiv:2011.05479*, 2020.

[25] Johannes Jakubik, Felix Yang, Benedikt Blumenstiel, Erik Scheurer, Rocco Sedona, Stefano Maurogiovanni, Jente Bosmans, Nikolaos Dionelis, Valerio Marsocci, Niklas Kopp, Rahul Ramachandran, Paolo Fraccaro, Thomas Brunschwiler, Gabriele Cavallaro, Juan Bernabe-Moreno, and Nicolas Longépé. Terramind: Large-scale generative multimodality for earth observation, 2025.

[26] Neal Jean, Sherrie Wang, Anshul Samar, George Azzari, David Lobell, and Stefano Ermon. Tile2vec: Unsupervised representation learning for spatially distributed data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 3967–3974, 2019.

[27] Z. Jin, C. Lin, C. Weigl, J. Obarowski, and D. Hale. Smallholder cashew plantations in benin version 1.0. https://doi.org/10.34911/rdnt.hfv20i, 2021.

[28] David A. Keith, José R. Ferrer-Paris, Emily Nicholson, Melanie J. Bishop, Beth A. Polidoro, Eva Ramirez-Llodra, Mark G. Tozer, Jeanne L. Nel, Ralph Mac Nally, and Edward J. Gregr. A function-based typology for earth's ecosystems. *Nature*, 610:513–518, 2022.

[29] Katerina Kikaki, Ioannis Kakogeorgiou, Ibrahim Hoteit, and Konstantinos Karantzalos. Detecting marine pollutants and sea surface features with deep learning in Sentinel-2 imagery. *ISPRS Journal of Photogrammetry and Remote Sensing*, 210:39–54, 2024.

[30] Alexandre Lacoste, Nils Lehmann, Pau Rodriguez, Evan Sherwin, Hannah Kerner, Björn Lütjens, Jeremy Irvin, David Dao, Hamed Alemohammad, Alexandre Drouin, et al. GEO-Bench: Toward foundation models for earth monitoring. *Advances in Neural Information Processing Systems*, 36, 2024.

[31] Jihyeon Lee, Nina R Brooks, Fahim Tajwar, Marshall Burke, Stefano Ermon, David B Lobell, Debashish Biswas, and Stephen P Luby. Scalable deep learning to identify brick kilns and aid regulatory capacity. *Proceedings of the National Academy of Sciences*, 118(17):e2018863118, 2021.

[32] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017.

[33] Oscar Manas, Alexandre Lacoste, Xavier Giró-i Nieto, David Vazquez, and Pau Rodriguez. Seasonal contrast: Unsupervised pre-training from uncurated remote sensing data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9414–9423, 2021.

[34] Shentong Mo and Shengbang Tong. Connecting joint-embedding predictive architecture with contrastive self-supervised learning. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[35] Jacob Morrison, Clara Na, Jared Fernandez, Tim Dettmers, Emma Strubell, and Jesse Dodge. Holistically evaluating the environmental impact of creating language models, 2025.

[36] National Aeronautics and Space Administration (NASA) Earthdata. Shuttle Radar Topography Mission. `https://e4ftl01.cr.usgs.gov/MEASURES/SRTMGL1.003/`, 2018.

[37] Team OLMo, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, Nathan Lambert, Dustin Schwenk, Oyvind Tafjord, Taira Anderson, David Atkinson, Faeze Brahman, Christopher Clark, Pradeep Dasigi, Nouha Dziri, Allyson Ettinger, Michal Guerquin, David Heineman, Hamish Ivison, Pang Wei Koh, Jiacheng Liu, Saumya Malik, William Merrill, Lester James V. Miranda, Jacob Morrison, Tyler Murray, Crystal Nam, Jake Poznanski, Valentina Pyatkin, Aman Rangapur, Michael Schmitz, Sam Skjonsberg, David Wadden, Christopher Wilhelm, Michael Wilson, Luke Zettlemoyer, Ali Farhadi, Noah A. Smith, and Hannaneh Hajishirzi. 2 olmo 2 furious, 2025.

[38] OpenStreetMap contributors. Planet dump retrieved from https://planet.osm.org . `https://www.openstreetmap.org`, 2017.

[39] Colorado J Reed, Ritwik Gupta, Shufan Li, Sarah Brockman, Christopher Funk, Brian Clipp, Kurt Keutzer, Salvatore Candido, Matt Uyttendaele, and Trevor Darrell. Scale-MAE: A scale-aware masked autoencoder for multiscale geospatial representation learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4088–4099, 2023.

[40] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE transactions on pattern analysis and machine intelligence*, 39(6):1137–1149, 2016.

[41] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[42] Marc Rußwurm, Sébastien Lefèvre, and Marco Körner. BreizhCrops: A satellite time series dataset for crop type identification. In *Proceedings of the International Conference on Machine Learning Time Series Workshop*, 2019.

[43] R Siddharth and Gnanasekaran Aghila. Randpro-a practical implementation of random projection-based feature extraction for high dimensional multivariate data analysis in R. *SoftwareX*, 12:100629, 2020.

[44] Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, Francisco Massa, Daniel Haziza, Luca Wehrstedt, Jianyuan Wang, Timothée Darcet, Théo Moutakanni, Leonel Sentana, Claire Roberts, Andrea Vedaldi, Jamie Tolan, John Brandt, Camille Couprie, Julien Mairal, Hervé Jégou, Patrick Labatut, and Piotr Bojanowski. DINOv3, 2025.

[45] Gencer Sumbul, Marcela Charfuelan, Begüm Demir, and Volker Markl. Bigearthnet: A large-scale benchmark archive for remote sensing image understanding. In *IGARSS 2019-2019 IEEE International Geoscience and Remote Sensing Symposium*, pages 5901–5904. IEEE, 2019.

[46] Daniela Szwarcman, Sujit Roy, Paolo Fraccaro, Thorsteinn Elí Gíslason, Benedikt Blumenstiel, Rinki Ghosal, Pedro Henrique de Oliveira, Joao Lucas de Sousa Almeida, Rocco Sedona, Yanghui Kang, et al. Prithvi-eo-2.0: A versatile multi-temporal foundation model for earth observation applications. *arXiv preprint arXiv:2412.02732*, 2024.

[47] Jamie Tolan, Hung-I Yang, Benjamin Nosarzewski, Guillaume Couairon, Huy V Vo, John Brandt, Justine Spore, Sayantan Majumdar, Daniel Haziza, Janaki Vamaraju, et al. Very high resolution canopy height maps from rgb imagery using self-supervised vision transformer and convolutional decoder trained on aerial lidar. *Remote Sensing of Environment*, 300:113888, 2024.

[48] Gabriel Tseng, Ivan Zvonkov, Catherine Lilian Nakalembe, and Hannah Kerner. CropHarvest: A global dataset for crop-type classification. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.

[49] Gabriel Tseng, Ruben Cartuyvels, Ivan Zvonkov, Mirali Purohit, David Rolnick, and Hannah Kerner. Lightweight, pre-trained transformers for remote sensing timeseries. *arXiv preprint arXiv:2304.14065*, 2023.

[50] Gabriel Tseng, Anthony Fuller, Marlena Reil, Henry Herzog, Patrick Beukema, Favyen Bastani, James R Green, Evan Shelhamer, Hannah Kerner, and David Rolnick. Galileo: Learning global & local features of many remote sensing modalities. In *Forty-second International Conference on Machine Learning*, 2025.

[51] United States Department of Agriculture (USDA) National Agricultural Statistics Service (NASS). Cropland Data Layer: USDA NASS, 2024. National Agricultural Statistics Service Marketing and Information Services Office, Washington, D.C. Retrieved from Link: https://croplandcros.scinet.usda.gov/.

[52] U.S. Geological Survey. National agriculture imagery program: 2003 - present. https://doi.org/10.5066/F7QN651G, 2023.

[53] Kristof Van Tricht, Jeroen Degerickx, Sven Gilliams, Daniele Zanaga, Marjorie Battude, Alex Grosu, Joost Brombacher, Myroslava Lesiv, Juan Carlos Laso Bayas, Santosh Karanam, et al. WorldCereal: a dynamic open-source system for global-scale, seasonal, and reproducible crop and irrigation mapping. *Earth System Science Data Discussions*, 2023:1–36, 2023.

[54] Leonard Waldmann, Ando Shah, Yi Wang, Nils Lehmann, Adam Stewart, Zhitong Xiong, Xiao Xiang Zhu, Stefan Bauer, and John Chuang. Panopticon: Advancing any-sensor foundation models for earth observation. In *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR) Workshops*, 2025.

[55] Yi Wang, Zhitong Xiong, Chenying Liu, Adam J Stewart, Thomas Dujardin, Nikolaos Ioannis Bountos, Angelos Zavras, Franziska Gerken, Ioannis Papoutsis, Laura Leal-Taixé, et al. Towards a unified copernicus foundation model for earth vision. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025.

[56] Yibing Wei, Abhinav Gupta, and Pedro Morgado. Towards latent masked image modeling for self-supervised visual representation learning. In *ECCV*, 2024.

[57] Western Cape Department of Agriculture. Crop type classification dataset for western cape, south africa. https://staging.source.coop/radiantearth/south-africa-crops-competition, 2021.

[58] Marta Yebra, Gianluca Scortechini, Karine Adeline, Nursema Aktepe, Turkia Almoustafa, Avi Bar-Massada, María Eugenia Beget, Matthias Boer, Ross Bradstock, Tegan Brown, et al. Globe-LFMC 2.0, an enhanced and updated dataset for live fuel moisture content research. *Scientific data*, 11(1):332, 2024.

[59] Daniele Zanaga, Ruben Van De Kerchove, Dirk Daems, Wanda De Keersmaecker, Carsten Brockmann, Grit Kirches, Jan Wevers, Oliver Cartus, Maurizio Santoro, Steffen Fritz, et al. ESA WorldCover 10 m 2021 v200. *ESA WorldCover Project*, 2022.

[60] Xiao Xiang Zhu, Jingliang Hu, Chunping Qiu, Yilei Shi, Jian Kang, Lichao Mou, Hossein Bagheri, Matthias Haberle, Yuansheng Hua, Rong Huang, et al. So2sat lcz42: A benchmark data set for the classification of global local climate zones [software and data sets]. *IEEE Geoscience and Remote Sensing Magazine*, 8(3):76–89, 2020.

# A Research Benchmarks

We describe the research benchmarks introduced in Section 3.2 in more detail below. We also share our observations on limitations of certain benchmarks.

GEO-Bench modifies benchmarks to form a unified and consistent collection of datasets:

**m-bigearthnet** is modified from BigEarthNet [45], which involves multi-label land cover classification of 120 × 120 Sentinel-2 image crops. It consists of 19 classes, such as arable land, inland wetlands, and urban fabric. The original dataset contains 549,488 examples, but the modified subset in GEO-Bench contains only 22,000, with 20,000 for training, 1,000 for validation, and 1,000 for testing.

**m-so2sat** is modified from So2Sat LCZ42 [60], which involves image-level classification of local climate zones from co-registered Sentinel-1 and Sentinel-2 crops. It consists of 17 classes, such as high-rise, industrial, and water bodies. The original dataset contains 400,673 examples, but the modified subset in GEO-Bench contains only 21,964, with 19,992 for training, 986 for validation, and 986 for testing.

**m-brick-kiln** is modified from the Brick Kiln Classification Dataset in Bangladesh [31]. The original dataset involves image-level classification of whether or not high-resolution 224 × 224 satellite image crops from DigitalGlobe contain at least one kiln, and contains 6,329 positive examples and 67,284 negative examples. The modified dataset in GEO-Bench performs the same task on corresponding 64 × 64 Sentinel-2 crops, and contains only 17,061 examples, with 15,063 for training, 999 for validation, and 999 for testing. While finding kilns in Sentinel-2 images is a challenging task, we find that the nature of the negatives in the GEO-Bench version of the dataset make the classification task too easy; for example, many negatives seem to have only dark pixels, making it easy to distinguish them.

**m-forestnet** is modified from ForestNet [24], which involves image-level classification of deforestation drivers from a composite 332 × 332 Landsat 8 satellite image captured within five years after each forest loss event. There are four driver categories: plantation, smallholder agriculture, grassland/shrubland, and other. The original dataset contains 2,756 examples. The modified subset in GEO-Bench contains 6,464 examples for training, 989 examples for validation, and 993 examples for testing; we could not determine where the additional examples came from.

**m-eurosat** is modified from EuroSat [22], which involves image-level land use and land cover classification from 64 × 64 Sentinel-2 image crops. It consists of 10 classes, such as annual crop, river, and highway. The original dataset contains 27,000 examples, but the modified subset in GEO-Bench contains only 4,000, with 2,000 for training, 1,000 for validation, and 1,000 for testing.

**m-cashewplant** is modified from the Smallholder Cashew Plantations in Benin Dataset [27], which involves segmentation of 256 × 256 Sentinel-2 image crops. It consists of six classes relating to cashew plantations: well-managed plantation, poorly-managed plantation, non-plantation, uncertain, residential, and background. The modified dataset contains 1,800 examples, with 1,350 for training, 400 for validation, and 50 for testing. Multiple models were sensitive to input patch size on this dataset, so for models that had a variable patch size, we swept input patch size and report the best result. Ultimately this is likely an effect of the labels being large polygons instead of per-pixel labels.

**m-SA-crop-type** is modified from the South Africa Crop Type Competition dataset [57], which involves crop type segmentation of 256 × 256 Sentinel-2 and Sentinel-1 image crops. It consists of 10 classes, such as fallow, wine grapes, and wheat. The modified dataset in GEO-Bench only uses the Sentinel-2 images, and contains 5,000 examples, with 3,000 for training, 1,000 for validation, and 1,000 for testing.

All of the GEO-Bench datasets share a significant limitation: although the tasks involve labels that do not change rapidly over time, the input consists of a single satellite image or image pair. We find that remote sensing models generally perform much better with multiple input images, and argue that single-image inputs should only be used for tasks like vessel detection where the labels are only valid for one timestep.

We compare on five additional datasets outside of GEO-Bench:

**BreizhCrops** [42] involves crop type classification from single-pixel Sentinel-2 time series. It consists of nine classes, such as wheat, corn, and permanent meadows. It contains 610K examples.

**CropHarvest** [48] involves binary cropland classification from single-pixel time series. The provided time series include Sentinel-2 and Sentinel-1 satellite image observations, as well as elevation from SRTM and weather data from ERA-5. It contains 95,186 examples.

**PASTIS** [17] involves crop type segmentation from Sentinel-1 and Sentinel-2 image time series, with $128 \times 128$ image crops. It consists of 19 classes, such as grapevine, spring barley, and soybeans. It contains 2,433 examples.

**MADOS** [29] involves marine debris segmentation in $80 \times 80$ Sentinel-2 image crops. It consists of 15 classes, such as oil spills, dense sargassum, and foam. It contains 2,803 examples. A key limitation with MADOS is that it provides custom-processed images, making it difficult to apply foundation models with their intended normalization statistics. Additionally, the dataset includes a lot of rare classes that greatly affect mIoU in the test set, making metrics highly variable across runs of the same model with different seeds.

**Sen1Floods11** involves binary water segmentation in $512 \times 512$ Sentinel-2 image crops that focus on flooded areas. It contains 4,831 examples. All of the remote sensing models we tested get between 78-80% accuracy, and we find that the accuracy is not well correlated with other benchmarks. However, Sen1Floods11 is one of the few Sentinel-1 benchmarks.

# B Partner Tasks

We describe the partner tasks introduced in Section 3.2 in more detail below.

**AWF - *African Wildlife Foundation (AWF)*** Land cover classification in southern Kenya. The dataset contains 1,459 examples with 9 classes, which range from lava forest and agriculture to urban development. The AWF team used Planet imagery as the main reference to annotate these examples.

**Live Fuel Moisture Content - *NASA JPL*** Regression dataset of 41,214 examples from Globe-LFMC-2.0 [58] labeled with the LFMC value. We partner with NASA JPL to deploy a model trained on this data. LFMC predictions are used to understand wildfire risk.

**Mangrove - *Global Mangrove Watch*** Classification dataset of 100,000 coastal areas into 3 classes: mangrove forest, water, or other. Mangrove maps across different years are used to understand mangrove growth and loss.

**Nandi - *CGIAR*** Crop-type classification in Nandi County, Kenya. The dataset contains 6,924 examples with 6 categories (coffee, maize, sugarcane, etc.). The ground-truth labels were collected through field surveys.

Ecosystem type mapping is similar, but only uses six timesteps of input images:

**GEA North Africa - *Global Ecosystem Atlas*** Ecosystem type classification of 2,361 examples in a region of North Africa, and labels correspond to the 110 categories in level 3 of the IUCN Global Ecosystem Typology [19].

The other tasks are more unique:

**Forest Loss Driver - *Amazon Conservation*** Classification dataset for the cause of forest loss in the Amazon rainforest into 10 classes (mining, logging, agriculture, etc.). The input consists of 4 Sentinel-2 images captured before the forest loss and 4 images captured after the forest loss. Driver predictions are used to prioritize enforcement and litigation efforts to deter further human-caused forest loss.

**Marine Infrastructure - *Skylight*** Global marine infrastructure detection dataset containing 7,197 examples labeled as offshore platform or wind turbine. The input consists of a time series of 4 Sentinel-2 or Sentinel-2 + Sentinel-1 images.

**Vessel Detection, Type, Length - *Skylight*** Three object detection tasks to detect vessels in Landsat (8,000 examples), Sentinel-1 (1,776 examples), and Sentinel-2 (45,545 examples) images, one classification task to predict the vessel type in Sentinel-2 images centered at detected vessels (584,432 examples), and one regression task to estimate the vessel length in Sentinel-2 images (584,432 examples). For all of these tasks, the input is a single image.

**Solar Farm Detection**: Binary segmentation dataset containing 3,561 examples densely labeled with solar farm polygons. The input consists of 4 timesteps, either Sentinel-2 or Sentinel-2 + Sentinel-1. Solar farm maps are used to understand the global rate of renewable energy deployment over time.

# C Additional Ablations

| | m-bigearthnet | m-so2sat | m-brick-kiln | m-forestnet | m-eurosat | BreizhCrops | PASTIS | PASTIS | MADOS | Sen1Floods11 | Average | Average Rank |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | S2 | S2 | S2 | L8 | S2 | S2 | S1 | S2 | S2 | S1 | | |
| | Acc. | Acc. | Acc. | Acc. | Acc. | F1 | F1 | F1 | F1 | F1 | | |
| MAE | 60.6 | 48.1 | 96.2 | 42.0 | 89.3 | 71.5 | **31.1** | 46.6 | 68.7 | 78.3 | 63.2 | 5.1 |
| Only S2 Data | 53.7 | 45.9 | 91.3 | - | 89.2 | **71.7** | - | 42.4 | 69.5 | - | 46.4 | - |
| No Maps | 59.5 | 58.6 | 95.2 | **46.0** | 92.6 | 71.4 | 29.1 | 48.0 | 70.2 | 77.9 | 64.9 | 4.7 |
| No Agricultural Maps | 60.9 | 66.5 | 94.3 | **46.0** | 93.9 | 71.4 | 29.0 | 48.5 | 71.4 | 78.8 | 66.1 | 3.6 |
| Random Masking | 60.7 | **67.4** | 94.7 | 43.5 | 91.8 | 70.3 | 24.7 | 51.1 | 71.9 | 77.8 | 65.4 | 4.7 |
| No Inst. Contrastive Loss | 60.5 | 65.6 | 93.6 | 44.9 | 93.6 | 70.2 | 28.5 | 51.4 | 72.1 | 78.4 | 65.9 | 4.7 |
| Patch Disc Loss | 62.0 | 62.1 | **96.3** | 44.8 | 94.0 | 70.3 | 29.6 | 50.0 | **74.1** | **79.3** | 66.2 | 3.0 |
| Final Recipe | **62.3** | 65.9 | 94.2 | 45.8 | **94.6** | 71.4 | 29.4 | **52.2** | 71.7 | 78.8 | **66.6** | **2.9** |

**Table 6** Ablation experiment selectively removing components of OlmoEarth base model.

In addition to the ablations in Section 3.6, we conduct a second set of ablations in Table 6. Our second set of ablations evaluates the contributions of components of our final model and training recipe by removing them individually, with the exception of the top row which is a MAE baseline. These models are trained for 300,000 steps. In the data ablation section we see the Sentinel-2 only model perform relatively poorly, however the “No Maps" run (only observational data) maintains relatively high performance. While our model can benefit from labeled data we still see good performance with pure self-supervised training.

Building remote sensing foundation models necessitates some tradeoffs. While our final model is not the best in every metric it retains high performance across the board and has the best average score and lowest average per-task rank.

# D Comparison to AlphaEarth Foundations

The AlphaEarth foundation model [9] is comparable to OlmoEarth in that both draw on similar data sources and were designed to support similar downstream tasks. Rather than releasing the model, Google released only the global, annualized embeddings computed by AlphaEarth. We compare OlmoEarth both as a frozen feature extractor (where, like AlphaEarth, only embeddings are used) and as an end-to-end finetune-able model.

It is expensive to export and download AlphaEarth embeddings from Google Earth Engine: our export jobs for $32 \times 32$ crops took 26 EECU-seconds on average, or $290 for a dataset with 100K crops. Thus, we were only able to evaluate AlphaEarth on five tasks: three classification tasks (Nandi, AWF, and Ecosystem), one per-pixel regression task (LFMC), and one segmentation task (Solar Farm).

Since the AlphaEarth model has not been released, we can’t evaluate AlphaEarth under a finetuning regime. We assess the performance of the annualized AlphaEarth embeddings compared to the OlmoEarth embeddings from the ViT Base encoder using a simple KNN classifier. We use the timestep of AlphaEarth embeddings that has the highest overlap with the time range of the labels. To assess the benefits of more complex decoders, we use the partner task decoders described in Section 3.4, while sweeping over the input size (AlphaEarth embeddings already capture spatial context, so we find that a smaller input size performs better).

With a KNN-classifier, OlmoEarth outperforms AlphaEarth on the Nandi and AWF tasks, while AEF outperforms OlmoEarth on the Ecosystem mapping task. However, OlmoEarth benefits significantly from full fine-tuning, with the fine-tuned models outperforming the best possible with AlphaEarth on all five tasks. This underscores the value of an open model that makes per-task fine-tuning possible.

|  |  | Nandi | AWF | Ecosystem | LFMC | Solar Farm |
|---|---|---|---|---|---|---|
| Model | Training | Acc. | Acc. | Acc. | L1 | mIOU |
| AEF | kNN | 55.6 | 81 | 60.6 | - | - |
| AEF | Frozen + Decoder | 66.0 | 75.9 | 61.2 | 23.1 | 77.5 |
| AEF | Full Fine-tuning | Not Possible |  |  |  |  |
| OlmoEarth | kNN | 66.2 | 82 | 59.3 | - | - |
| OlmoEarth | Frozen + Decoder | 62.9 | 84.0 | 61.1 | 19.9 | 84.8 |
| OlmoEarth | Full Fine-tuning | **82.2** | **86.0** | **62.4** | **17.9** | **86.7** |

**Table 7** Comparing AlphaEarth Foundation (AEF) embeddings with OlmoEarth ViT Base model using three different training strategies: kNN, frozen backbone + decoder, and decoder with full fine-tuning. For these evaluations, we use the "partner task" decoders described in Section 3.4.

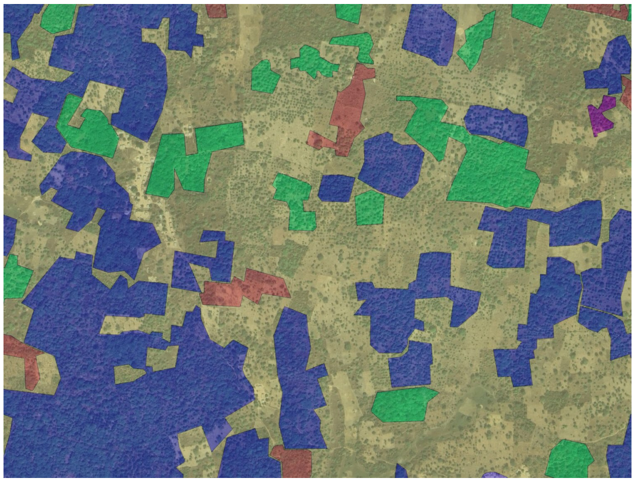


**Figure 5** An example instance from the `m_cashew_plant` dataset: note the coarse, polygonal labels

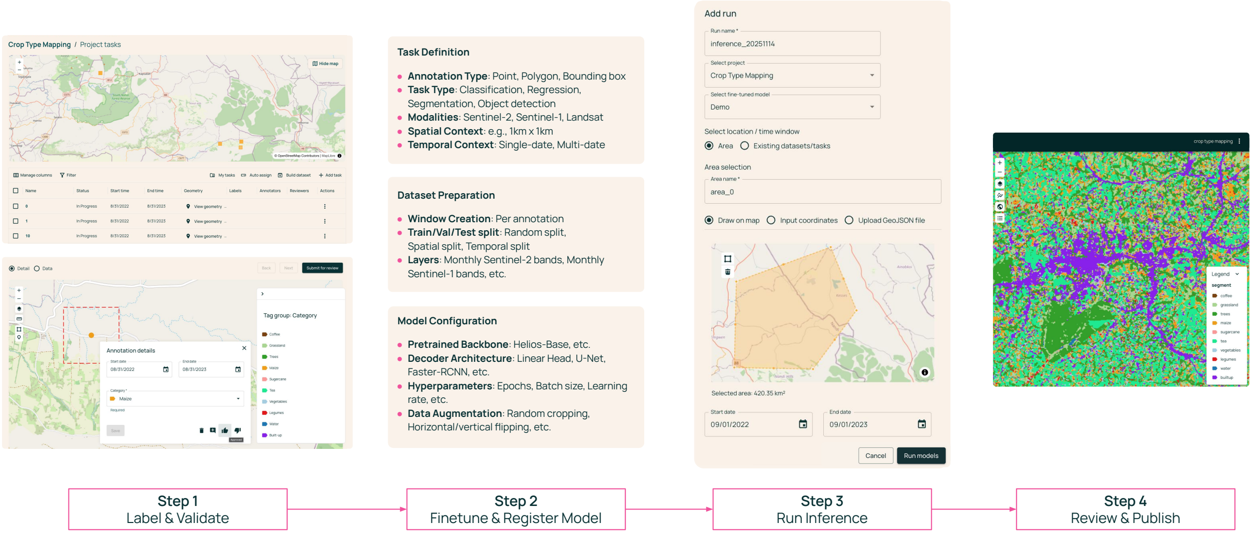
# E Patch Size Analysis for `m_cashew_plant`

We observe that for the `m_cashew_plant` evaluation task, larger patch sizes lead to better performance for models that support variable patch sizes, such as OlmoEarth and Galileo. Table 8 summarizes the linear probing and fine-tuning results for `m_cashew_plant` across different patch sizes.

This effect is unusual: a smaller patch size typically improves performance (e.g. Figure 4 of [50]). We hypothesize that this is due to the spatially coarse labels in the dataset, which are polygons instead of pixels (Figure 5).

| Model | Patch 4×4 |  | Patch 8×8 |  | Patch 16×16 |  |
|---|---|---|---|---|---|---|
|  | LP | FT | LP | FT | LP | FT |
| OlmoEarth-Base | 27.7 | 71.9 | 27.9 | 76.2 | 32.3 | 79.8 |
| Galileo | 24.3 | 73.0 | 25.6 | 76.9 | 28.9 | 78.8 |

**Table 8** Performance (mIoU) comparison (LP = Linear Probing, FT = Fine-tuning) across patch sizes.



Crop Type Mapping / Project tasks

| ☐ | Name | Status | Start time | End time | Geometry | Labels | Annotators | Reviewers | Actions |
|---|---|---|---|---|---|---|---|---|---|
| ☐ | 0 | In Progress | 8/31/2022 | 8/31/2023 | View geometry | | | | ⋮ |
| ☐ | 1 | In Progress | 8/31/2022 | 8/31/2023 | View geometry | | | | ⋮ |
| ☐ | 10 | In Progress | 8/31/2022 | 8/31/2023 | View geometry | | | | ⋮ |

● Detail ○ Data

Annotation details

Start date: 08/31/2022

End date: 08/31/2023

Category *: Maize

Required

Tag group: Category

**Task Definition**

- **Annotation Type**: Point, Polygon, Bounding box
- **Task Type**: Classification, Regression, Segmentation, Object detection
- **Modalities**: Sentinel-2, Sentinel-1, Landsat
- **Spatial Context**: e.g., 1km x 1km
- **Temporal Context**: Single-date, Multi-date

**Dataset Preparation**

- **Window Creation**: Per annotation
- **Train/Val/Test split**: Random split, Spatial split, Temporal split
- **Layers**: Monthly Sentinel-2 bands, Monthly Sentinel-1 bands, etc.

**Model Configuration**

- **Pretrained Backbone**: Helios-Base, etc.
- **Decoder Architecture**: Linear Head, U-Net, Faster-RCNN, etc.
- **Hyperparameters**: Epochs, Batch size, Learning rate, etc.
- **Data Augmentation**: Random cropping, Horizontal/vertical flipping, etc.

Add run

Run name *: inference_20251114

Select project: Crop Type Mapping

Select fine-tuned model: Demo

Select location / time window

● Area ○ Existing datasets/tasks

Area selection

Area name *: area_0

● Draw on map ○ Input coordinates ○ Upload GeoJSON file

Selected area: 420.35 km²

Start date: 09/01/2022

End date: 09/01/2023

**Figure 6 OlmoEarth Platform: End-to-End Workflow (using crop type mapping as an example).** The platform enables users to complete the full process from data labeling to map publishing: **Step 1:** Label and review annotations, **Step 2:** Fine-tune and register models for specific tasks, **Step 3:** Run inference on selected areas and time ranges, and **Step 4:** Review and publish the final maps.

# F OlmoEarth Platform

OlmoEarth Platform is an end-to-end solution that combines our foundation models with data management tools designed for organizations working on environmental challenges. The platform handles the complete workflow from satellite data collection through labeling, model fine-tuning, and inference, eliminating the need for organizations to manage GPU infrastructure or deep learning expertise. By making our models accessible, OlmoEarth Platform solves the last-mile problem of translating research into practical tools for applications including conservation, climate action, and food security.