MAESTRO: Masked AutoEncoders for Multimodal, Multitemporal, and Multispectral Earth Observation Data

Supplementary Material

The appendix is structured as follows:

- Sec. 6 provides a high-level comparison of MAESTRO and related approaches from the litterature;
- Sec. 7 provides our full experimental details;
- Sec. 8 reports the detailed results for the experiments presented in Figs. 3 to 5;
- Sec. 9 reports additional ablation studies;
- Sec. 10 offers a qualitative analysis on inference results;
- Sec. 11 reports computational costs.

6. Comparison Table of MAESTRO versus Prior Work

We provide in Tab. 3 a high-level overview of the differences between our approach, MAESTRO, and prior SSL work in Earth observation.

Table 3. **Comparison of MAESTRO with prior SSL work in EO**. We indicate the characteristics of the original versions of the related SSL approaches (i.e., before the adaptations of Sec. 7.4). ⊕ means exclusive OR; 🔁 means parameter sharing; 🖼 means joint-token, 🧱 token-based; ⏳ means early fusion, ⌛ intermediate fusion.

	Model	Multimodal	Multitemporal	Multispectral	VHR	DEM/DSM	Remark
MAE/SimMIM style	ScaleMAE [66]	✗	✗	✗	✓	✗	-
	Cross-Scale MAE [75]	✗	✗	✗	✓	✗	-
	SatMAE [13]	✗	⊕✓ 🧱 ⌛	⊕✓ 🧱 ⌛	✗	✗	Limited to 3 dates
	Prithvi v1/v2 [41, 74]	✗	✓ 🧱 ⌛	✓ 🖼 ⌛	✗	✗	Limited to 3–4 dates (v1–v2)
	CROMA [25]	✓ 🧱 ⌛	✗	✓ 🖼 ⌛	✗	✗	-
	RingMo [73]	✓ 🔁	✗	✗	✓	✗	Frequency-enhanced MIM
	USat [38]	✓ 🧱 ⌛	✗	✓ 🧱 ⌛	✓	✗	-
	Billion-scale FM [11]	✓ 🔁	✗	✗	✓	✗	-
	GFM [57]	✓ 🔁	✗	✗	✓	✗	-
	msGFM [34]	✓ 🔁	✗	✓ 🖼 ⌛	✓	✓	-
	MMEarth [60]	✓ 🔁	✗	✓ 🖼 ⌛	✗	✓	ConvNeXt V2 encoder
	SenPa-MAE [64]	✓ 🔁	✗	✓ 🧱 ⌛	✗	✗	-
	SeaMo [47]	✓ 🧱 ⌛	✓ 🧱 ⌛	✓ 🖼 ⌛	✗	✗	Limited to 4 dates
	DOFA [102]	✓ 🔁	✗	✓ 🖼 ⌛	✓	✗	-
	EarthMAE [84]	✓ 🧱 ⌛	✓ 🧱 ⌛	✓ 🖼 ⌛	✓	✓	-
	FoMo-Net [9]	✓ 🧱 ⌛	✓ 🧱 ⌛	✓ 🧱 ⌛	✓	✓	-
	Copernicus-FM [97]	✓ 🔁	✗	✓ 🖼 ⌛	✗	✓	-
Other Approaches	CMID [59]	✗	✗	✗	✓	✗	Mix of contrastive and MIM
	DINO-v2 sat. [78]	✗	✗	✗	✓	✗	-
	DINO-v2 GeoSA [53]	✗	✗	✗	✓	✗	-
	OmniSat [4]	✓ 🧱 ⌛	✓ 🖼 ⌛	✓ 🖼 ⌛	✓	✓	-
	AnySat [5]	✓ 🧱 ⌛	✓ 🖼 ⌛	✓ 🖼 ⌛	✓	✓	JEPA-style
	SkySense v1/v2 [33, 105]	✓ 🧱 ⌛	✓ 🧱 ⌛	✓ 🖼 ⌛	✓	✗	-
	SkySense++ [100]	✓ 🧱 ⌛	✓ 🧱 ⌛	✓ 🖼 ⌛	✓	✗	Early parameter sharing
	Presto [81]	✓ 🧱 ⌛	✓ 🧱 ⌛	✓ 🧱 ⌛	✗	✓	Pixel level w/o spatial context
	Galileo [82]	✓ 🧱 ⌛	✓ 🧱 ⌛	✓ 🧱 ⌛	✗	✓	Mix of MAE and JEPA
	Terramind [42]	✓ 🧱 ⌛	✗	✓ 🖼 ⌛	✗	✓	-
Ours	MAESTRO	✓ 🧱 ⌛	✓ 🧱 ⌛	✓ 🖼 ⌛	✓	✓	-

7. Experimental Details

In this section, we provide our full experimental details. We provide details on the four evaluated datasets (Sec. 7.1), details shared across models (Sec. 7.2), details specific to MAEs/MAESTRO (Sec. 7.3), and details specific to the adapted baseline

FMs (Sec. 7.4). Additionally, we provide tables reporting the exhaustive list of hyperparameter values (Sec. 7.5).

7.1. Experimental Details on the Datasets

In this subsection, we provide additional details on the choice of input modalities and labels for the four evaluated datasets (Secs. 7.1.1 to 7.1.4) and the additional pre-training dataset S2-NAIP urban (Sec. 7.1.5). We also report the full details on the raw preprocessing steps applied to some input modalities (Sec. 7.1.6).

7.1.1. TreeSatAI-TS

TreeSat was introduced in [1] and comprises 50,381 tiles of 60 m × 60 m in Germany, featuring aerial imagery (RGB + NIR) at 0.2 m resolution, together with monotemporal Sentinel-1 and Sentinel-2 data (Sentinel-2 provided as seasonal medians). It was later extended as TreeSatAI-TS in [4], adding Sentinel-1 and Sentinel-2 time series covering the full year closest to the aerial acquisition. In this work, we retain four distinct modalities: aerial imagery, Sentinel-1 time series in both orbits (ascending and descending), and Sentinel-2 time series. We discard the original monotemporal Sentinel-1 and Sentinel-2 data.

The original labels introduced in [1] were formulated as regression targets, representing the spatial fraction of each tree species within a patch. Following [4, 5], we recast this as a multi-label classification task: a species class is considered present if its spatial fraction exceeds 0.07.

7.1.2. PASTIS-HD

PASTIS was introduced in [31] and comprises 433 tiles of 1280 m × 1280 m in France, featuring Sentinel-2 time series spanning approximately one year. It was later extended as PASTIS-R in [32] by adding Sentinel-1 time series covering about 70 dates in both ascending and descending orbits, and further expanded in [4] as PASTIS-HD with SPOT 6–7 imagery resampled to 1 m resolution. We retain four distinct modalities: SPOT 6–7 imagery, Sentinel-1 time series in both orbits (ascending and descending), and Sentinel-2 time series.

The crop segmentation labels in [31] covered both semantic and panoptic segmentation. Here, we retain only the semantic segmentation labels, omitting panoptic segmentation. Restricting to semantic segmentation allows us to consider a spatialized fine-tuning task without introducing additional complexities related to specialized heads or loss functions. We assume that the spatialized nature of the task (e.g., semantic segmentation vs. classification) plays a more decisive role than the precise segmentation type (e.g., panoptic vs. semantic) when benchmarking different SSL approaches.

7.1.3. FLAIR#2

FLAIR was introduced in [26] and comprises 77,762 tiles in France, with aerial and elevation imagery (RGB + NIR + DSM) at 0.2 m resolution. It was extended as FLAIR#2 in [27, 28] with Sentinel-2 time series spanning a full year in the form of superpatches, covering a larger spatial extent than the aerial imagery (400 m × 400 m vs. 102.4 m × 102.4 m) to provide additional spatial context. Following [5], we crop the Sentinel-2 time series to match the extent of the VHR imagery, discarding 93.5% of pixels. We also include elevation imagery at 0.2 m resolution, extracted from the FLAIR-HUB extension [29] on the 77,762 FLAIR#2 tiles. In total, we retain three distinct modalities: aerial imagery (RGB + NIR), elevation imagery (DEM + DSM), and Sentinel-2 time series.

We use the land cover semantic segmentation labels following the filtering procedure in [26–28]: we retain only 12 of the 18 original classes, excluding the remaining 6 classes from the loss and metric computations.

7.1.4. FLAIR-HUB

FLAIR-HUB [29] is an extension of FLAIR#2 with 241,100 tiles of 102.4 m × 102.4 m. Compared to FLAIR#2, FLAIR-HUB enriches the elevation imagery (DEM + DSM) and treats it as a separate modality, crops the Sentinel-2 imagery to the extent of the aerial imagery (as we do in our version of FLAIR#2), and adds Sentinel-1 time series spanning a full year, as well as SPOT 6–7 imagery and historical aerial imagery. We retain five distinct modalities: aerial imagery (RGB + NIR), elevation imagery (DEM + DSM), Sentinel-1 time series in both orbits, and Sentinel-2 time series. We discard the SPOT 6–7 imagery, which did not significantly improve fine-tuning performance—likely due to redundancy with the aerial modality—and the historical imagery, due to its lack of synchronicity with the labels.

We use the land cover semantic segmentation labels following the filtering procedure in [29]: we retain 15 of the 18 original classes, excluding the remaining 3 from the loss and metric computations. We omit the crop semantic segmentation labels.

7.1.5. S2-NAIP urban

S2-NAIP was introduced in [23, 99] and features National Agriculture Imagery Program (NAIP) aerial imagery at 1.25 m resolution, together with Sentinel-1, Sentinel-2, and Landsat time series at 10 m resolution.

We construct the urban subset following [5], i.e. we restrict the original S2-NAIP footprints to those overlapping with the urban set defined in [99], which includes only locations within a 5 km radius of U.S. cities with populations of at least 50,000. By preventing dominance by homogeneous rural landscapes, this filtering ensures that the dataset remains balanced [57, 99].

The resulting subset contains 167,397 tiles of size 640 m × 640 m, covering a total area of 68,565 km^2 across the continental United States.

We retain three distinct modalities: NAIP imagery, Sentinel-1 time series from mixed ascending and descending orbits, and Sentinel-2 time series.

We exclude the OpenStreetMap and WorldCover labels, as S2-NAIP urban is used exclusively for MAESTRO pre-training in the cross-dataset evaluation of Sec. 4.5.

7.1.6. Raw Preprocessing

In TreeSatAI-TS, FLAIR#2, and FLAIR-HUB, the Sentinel-1 backscattering coefficient data is provided in linear scale. We apply a logarithmic transformation to express it in decibel (dB) scale (up to a multiplicative constant). For PASTIS-HD, the Sentinel-1 data is already in dB scale; however, we retain only the first two channels, corresponding to vertical (VV) and horizontal (VH) polarizations, discarding the third channel expressing their ratio (VV/VH).

In S2-NAIP urban, we retain only ten out of the twelve Sentinel-2 bands, keeping those at 10 m and 20 m resolutions but excluding the two at 60 m resolution.

In FLAIR#2 and FLAIR-HUB, we apply a simple preprocessing step to the elevation imagery (DEM + DSM). We recast it as the DEM and a rescaled elevation defined as $30 \times (\text{DSM} - \text{DEM})$, ensuring that both channels have comparable value ranges.

The TreeSatAI-TS aerial imagery tiles have a slightly larger spatial extent than the other modalities (304 × 304 pixels, corresponding to 60.8 m, instead of 300 × 300 pixels). We apply a centered crop to align them spatially with the other inputs.

7.2. Experimental Details Shared across Models

In this subsection, we provide details on the preprocessing pipeline (Sec. 7.2.1), positional/temporal encodings (Sec. 7.2.2), data augmentation (Sec. 7.2.3), regularization (Sec. 7.2.4) and optimizers (Sec. 7.2.5).

7.2.1. Preprocessing

Here we provide the full details on the preprocessing pipeline introduced in Sec. 3.1. The full implementation is available in `maestro/dataset/dataset.py` in our code.

Dataset-specific crop hyperparameter. Each dataset $\mathcal{D}$ has a specific hyperparameter controlling the *spatial extent of crops* within original tiles. For classification tasks, where labels apply to entire tiles, we set the crop extent equal to the original tile extent. For segmentation tasks, the crop extent can be smaller as long as the same crop is applied to the segmentation labels. Choosing the appropriate crop size involves balancing two competing effects:

- Smaller crops increase the spatial resolution of the token grid under a fixed token budget. For a fixed token budget $(I_m/P_m)^2 D_m$, a reduction of the image size I_m implies a reduction of the patch size P_m, mitigating the risk of an information bottleneck in the tokenizer (the model's entry block).
- Larger crops provide more spatial context for the model.

When cropping to a smaller extent than the original tile, we apply for each epoch a repetition factor $R_{\mathcal{D}}$ equal to the ratio of the original tile area to the cropped area. This ensures that the total spatial area seen by the model in each epoch matches that of the full dataset. Practically, this means the dataset length is scaled by $R_{\mathcal{D}}$: each tile, originally mapped to a single index, is now associated with $R_{\mathcal{D}}$ indices, effectively repeating it that many times with different crops.

The *crop sampling* strategy differs between training and validation/testing:

- *Training:* the crop is sampled randomly from all valid crops within the tile, subject only to the constraint that crop boundaries align with integer pixel indices for each modality m.
- *Validation and testing:* tiles are partitioned into non-overlapping crops, which are iterated over exhaustively. The same repetition factor is applied in the test epoch, ensuring test metrics cover the exact same spatial footprint as the original tiles.

In practice, we use the full tile extent for TreeSatAI-TS, FLAIR#2, and FLAIR-HUB. However, for PASTIS-HD and S2-NAIP urban, we found it beneficial to substantially reduce the crop size. This allows us to set the number of temporal bins

as $D_m = 16$ for Sentinel-2 and $D_m = 4$ for Sentinel-1, while setting the patch size as $P_m = 2$ for both modalities. Keeping the same token budget and number of temporal bins would have required to set $P_m = 16$ on PASTIS-HD and $P_m = 10$ on S2-NAIP urban for these modalities without cropping.

Modality-specific hyperparameters. Each modality m in the dataset $\mathcal{D}$ has a configurable set of preprocessing hyperparameters:

- The *image size* I_m, which by default corresponds to the number of pixels covering the spatial extent of the crop (however, this can be adjusted for cross-dataset MAESTRO and baseline FMs to yield a given token grid size while retaining a specific transferred patch size P_m);
- The *target number of temporal bins* D_m;
- Whether a *snow/cloud mask* mask is used to guide valid time step selection within temporal bins, and, if so, the probability threshold above which values are discarded;
- A *constant multiplicative normalization factor* applied to the modality.

Preprocessing steps. The `__getitem__` method of our datasets operates in three steps:

(i) Determine the sampled dataset tile based on the dataset index;
(ii) Sample a spatial crop shared across all modalities $m \in \mathcal{D}$;
(iii) Process each modality m based on the sampled crop.

Processing step (iii) for each modality m proceeds as follows:

(a) If the number of original time steps T_m is not a multiple of D_m, randomly truncate the sequence to $D_m \times \lfloor T_m / D_m \rfloor$ consecutive time steps.
(b) Read the corresponding tile section defined by the crop's spatial extent and the (potentially truncated) temporal extent. To minimize I/O overhead, we use lazy loading:
  - For `.tif` files: read with `rasterio` using window reading.
  - For `.npy` files: read with `numpy.load` using the argument `mmap_mode="r"`.
  - For `.h5` files: read with `h5py.File` in read mode, and index the required array sections.
(c) Reshape the array into D_m temporal bins.
(d) For each temporal bin, sample one time step inside the bin:
  - If a snow/cloud mask is used, filter out time steps that include any mask values above the configured threshold. If no valid time steps remain, all time steps in the bin are instead retained.
  - Select a single time step from the valid ones. During training, the selection is random to serve as data augmentation. During validation and testing, representativeness is maximized by: computing the pixel-wise median across valid time steps in the temporal bin; computing the mean absolute deviation to this median for each time step; and selecting the time step with the lowest mean absolute deviation.
(e) If the image size I_m does not match the number of pixels covering the spatial extent of the crop, reinterpolate the array spatially with `torch.nn.functional.interpolate` using `mode="nearest"`.
(f) Scale the array by the configured multiplicative normalization factor.

7.2.2. Positional/Temporal Encodings

As noted in Sec. 3.1, we do not include explicit modality encodings, since we use modality-specific tokenizers and learnable modality-specific `[mask]` tokens that implicitly encode the source or target modality for each token. However, we do include spatial and temporal positional encodings, as detailed below.

Spatial encodings. To encode spatial information, we follow the common practice of using two-dimensional sine–cosine positional encodings [35, 80, 83]. As in Scale-MAE [66], we scale these positional encodings according to the ground sampling distance of each modality. Concretely, we compute the *least common multiple* (LCM) of the token grid sizes across modalities, i.e., $\mathrm{lcm}_{m \in \mathcal{D}}\{I_m / P_m\}$, and generate positional encodings on the high-resolution grid defined by this LCM. For each modality m, we then obtain its positional encodings by downsampling the high-resolution grid, averaging over all high-resolution grid cells corresponding to each low-resolution grid cell.

Temporal encodings. To encode temporal information, we first extract the day of year and hour of day for the selected time step in each temporal bin (see Sec. 7.2.1). The day of year is normalized by 365.25 and the hour of day by 24. Applying sine and cosine transformations to these normalized values yields four temporal features per time step.

In addition, for each tile, we define a reference date shared across all modalities m to capture temporal differences across years, rather than only within a single year. For each time step, we compute the difference between its acquisition date and this reference date, obtaining a fifth temporal feature. This value is then duplicated four times, resulting in a total of eight

temporal features per time step.

Aggregation. Spatial and temporal encodings are aggregated by concatenation. In the encoder with latent dimension C_e, the spatial encodings occupy $C_e - 8$ dimensions, while the temporal encodings use 8 dimensions, leading to C_e dimensions after concatenation. The same approach is applied in the decoder with latent dimension C_d.

7.2.3. Data Augmentation

We use up to three types of data augmentation:

- *Random spatial cropping*: During training, a crop of the configured extent is randomly sampled within the original tile at the start of preprocessing. This is disabled for validation and testing, where tiles are instead partitioned into non-overlapping crops processed exhaustively in each epoch. When the crop extent matches the original tile extent, this augmentation is implicitly disabled, even during training.
- *Random time step selection*: During training, time steps are randomly sampled from the valid steps within each temporal bin. For validation and testing, we instead select time steps that maximize representativeness among valid steps.
- *D4 augmentation*: Throughout training, validation, and testing, synchronized D4 transformations are applied across all modalities and semantic segmentation labels just after the preprocessing pipeline. We retain this augmentation for validation and testing, as we assume the model has learned equivariance to these transformations during training.

7.2.4. Regularization

As regularization, we apply an Exponential Moving Average (EMA) of the model weights during fine-tuning [58], similar to Stochastic Weight Averaging [39]. Concretely, an EMA of the weights is updated at each epoch with a smoothing window equal to 20% of the total fine-tuning epochs. Denoting the model weights by θ and their EMA by θ_{EMA}, the EMA weights are initialized as θ and updated at each epoch as:

$$\theta_{\text{EMA}} = \alpha\theta_{\text{EMA}} + (1-\alpha)\theta, \qquad \alpha = 1 - (0.2 \times N_{\text{epochs}})^{-1}.$$

During validation and testing, we use the EMA weights instead of the regular model weights.

7.2.5. Optimizer

In all experiments, we use the AdamW optimizer [52]. The learning rate is scaled with the square root of the batch size [45, 55, 68, 103]; that is, we set the learning rate equal to the base learning rate multiplied by the square root of the batch size. However, for a given dataset and training phase (i.e., SSL pre-training, probing, or fine-tuning), we keep the batch size and learning rate fixed across experiments to avoid any confounding factors.

Across all phases—SSL pre-training, probing, and fine-tuning—we use a cosine decay learning rate scheduler with a single cycle and a warm-up period. This approach is common in recent SSL studies with Transformers in computer vision [10, 35, 62, 101]. We implement this using `torch.optim.lr_scheduler.OneCycleLR` with its default annealing strategy and a warm-up for 20% of the total epochs.

During SSL pre-training and probing, the learning rate is annealed down to 1×10^{-4} times its maximum value. During fine-tuning, it is instead annealed only to half of the maximum learning rate. This helps fully leverage the EMA strategy as weight averaging (i) naturally reduces noise [58] and (ii) benefits from greater model diversity along the training trajectory when the learning rate is kept sufficiently high.

7.3. Experimental Details Specific to MAEs/MAESTRO

In this subsection, we provide experimental details specific to MAEs/MAESTRO. We detail the masking strategy (Sec. 7.3.1) and the selection of band groups for patch-group-wise normalization (Sec. 7.3.2). We also provide details on our adaptations of MAESTRO in the cross-dataset setting (Sec. 7.3.3).

7.3.1. Masking

Masking strategy. We adopt a masking strategy that proceeds in two stages:

(i) *Structured Masking:*
  (a) *Modality structure:* Mask each modality with a fixed probability of 0.25;
  (b) *Spatial structure:* Within each modality, mask each spatial position with a fixed probability of 0.25;
  (c) *Temporal structure:* Within each modality, mask each temporal position with a fixed probability of 0.25.

(ii) *Unstructured Masking:* Adjust the masking from step (i) to match an overall 75% masking ratio:
  - If step (i) results in too few masked tokens, randomly mask additional unmasked tokens;
  - If step (i) results in too many masked tokens, randomly unmask some of them.

Comparison with previous works. Our masking strategy has similarities with previous works.

Step (i-a), which introduces modality-structured masking, is conceptually similar to the band-masking strategy in Fomo-Net [9], although a direct equivalence would require treating each spectral band as a distinct modality.

Step (i-b), which introduces spatially-structured masking, aligns with the consistent masking in SatMAE [13] and the tube masking in VideoMAE [79]. It also resembles the approach in SeaMo [47], but with a key distinction: SeaMo enforces consistency across modalities at a single time step, whereas we enforce consistency across time steps within a single modality.

Step (i-c), which introduces temporally-structured masking, corresponds to the "Timesteps" strategy in Presto [81].

Our combined masking strategy shares similarities with those in AnySat [5], EarthMAE [84], and Galileo [82], while still differing in important ways. AnySat and Galileo apply only spatially- and temporally-structured masking, while EarthMAE uses only spatially-structured masking. Moreover, the way in which we integrate structured and unstructured masking is distinct from these methods.

7.3.2. Band Groups

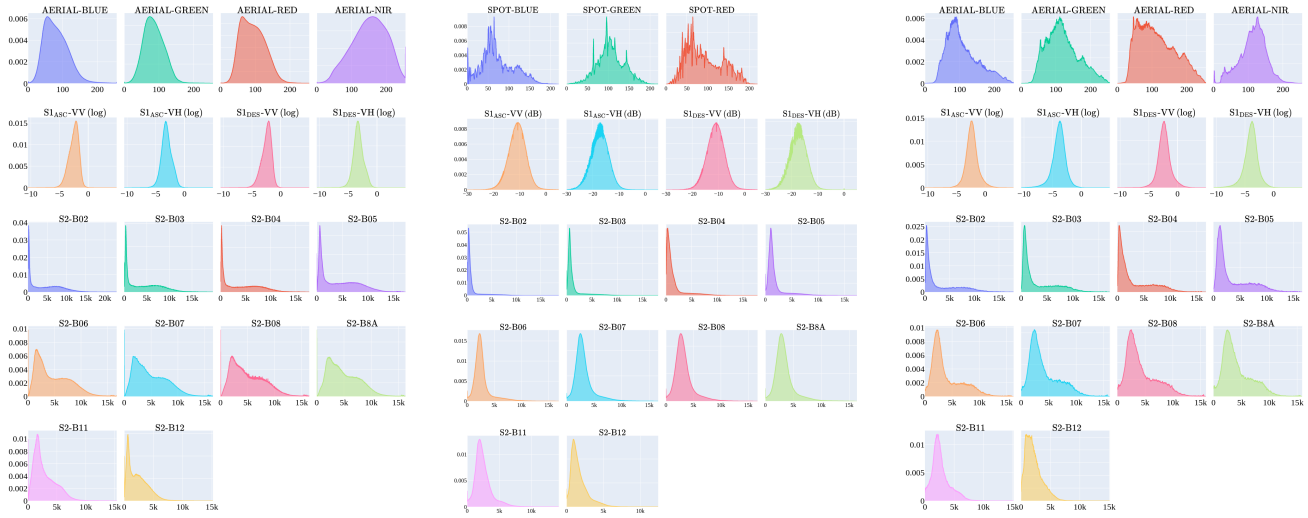
Here, we provide details on our selection of spectral band groups for patch-group-wise normalization.

We start by computing per-band histograms for the VHR, Sentinel-1, and Sentinel-2 modalities on TreeSatAI-TS, PASTIS-HD, and FLAIR-HUB. To reduce computational cost, a toy version of FLAIR-HUB containing 250 tiles was used for histogram computation. The resulting histograms are shown in Fig. 6a, Fig. 6b, and Fig. 6c.

For VHR aerial imagery, the RED, GREEN, and BLUE bands exhibit relatively similar histograms, whereas the NIR band shows a distinct distribution.

For Sentinel-1, the histograms of VV and VH differ markedly, in line with their differing polarization responses; VH generally exhibits lower backscatter than VV.

For Sentinel-2, the ten bands cluster into three natural groups with strong intra-group similarity but higher inter-group variation.



(a) TreeSatAI-TS (b) PASTIS-HD (c) FLAIR-HUB

Figure 6. **Per-band histograms for VHR, Sentinel-1, and Sentinel-2 modalities on the TreeSatAI-TS, PASTIS-HD, and FLAIR-HUB datasets.**

Table 4. **Spectral band groups selected for patch-group-wise normalization.**

Modality	Group 1	Group 2	Group 3
Aerial	RED, GREEN, BLUE	NIR	
SPOT	RED, GREEN, BLUE		
$S1_{ASC}$ / $S1_{DES}$	VV	VH	
S2	B02, B03, B04, B05	B06, B07, B08, B8A	B11, B12

Based on these insights, we define the spectral band groups detailed in Tab. 4 for patch-group-wise normalization.

Note that the Sentinel-2 band groups follow the natural wavelength order, which does not always coincide with the order of spatial resolutions (e.g., band B05 has 20 m resolution, while B08 has 10 m). This contrasts with some prior approaches to S2 band grouping [13, 38].

7.3.3. Cross-Dataset Transfer

Here, we detail the adaptations of MAESTRO made in the cross-dataset setting compared to the intra-dataset setting.

First, we ensure that the patch sizes P_m for each modality match between the pre-training and fine-tuning datasets. This allows the patchification layers to be fully transferred across datasets. During fine-tuning, we ensure a fair comparison by adjusting the image size I_m for each modality to use this fixed patch size P_m while maintaining the same token budget as in intra-dataset MAESTRO, ViTs, and adapted baseline FMs.

Second, *fixed cross-dataset grids* for positional encodings are defined based on the ground sampling distance, following the approach in Scale-MAE [66].

Finally, when pre-training on S2-NAIP urban, we generate surrogate modalities for aerial and SPOT imagery via resampling of NAIP imagery (see Tab. 10). These surrogate modalities are transferred to the corresponding modalities in the downstream tasks. Additionally, we do not distinguish the Sentinel-1 modality by orbit during pre-training, as this information is unavailable in S2-NAIP. During fine-tuning on TreeSatAI-TS, PASTIS-HD, and FLAIR-HUB, we retain the separate Sentinel-1 modalities by orbit, using a single patchification layer initialized from the pre-trained Sentinel-1 layer. The Sentinel-1 encoder is subsequently transferred to the grouped encoder for Sentinel-1 orbits, consistent with MAESTRO's *inter-group* fusion mode, where these modalities are grouped.

7.4. Experimental Details Specific to Adapted Baseline FMs

In this subsection, we describe the baseline FMs that we use in our comparative evaluation of MAESTRO, along with their extensive adaptations.

The selected models include DINO-v2 [62], which is pre-trained on natural images, as well as DINO-v2 sat. [78], CROMA [25], DOFA [102], SatMAE [13], and Prithvi-EO 2.0 [74], which are pre-trained on EO data.

7.4.1. Limitations in the Original Models

In their original configurations, the selected baseline FMs exhibit several limitations (see also Tab. 3):

- **CROMA/DOFA.** CROMA natively supports multiple modalities, but it is monotemporal. DOFA also supports different modalities, but only through parameter sharing and not through multimodal fusion. Additionally, DOFA is monotemporal.
- **DINO-v2/DINO-v2 sat.** DINO-v2 and DINO-v2 sat. are strictly monomodal and monotemporal. Additionally, DINO-v2 and DINO-v2 sat. are only pre-trained on RGB images.
- **Prithvi-EO 2.0/SatMAE.** Prithvi-EO 2.0 is natively multitemporal, but its pre-training is limited to four Sentinel-2 dates. SatMAE also supports multitemporality, but its reference implementation and its pre-training are limited to three Sentinel-2 dates. Both models are pre-trained on a limited subset of Sentinel-2 bands: six bands for Prithvi-EO 2.0 (RGB, NIR, SWIR1, SWIR2), and three bands for SatMAE (RGB).

7.4.2. Overview of Adaptations

To optimize the performance of these baseline FMs, we introduce several key adaptations:

- **Multimodal tokenization:**
  - Multiple modalities are processed via parallel, modality-specific tokenizers.
- **Incorporation of additional bands:**
  - Tokenizers are modified to allow the inclusion of bands not seen during pre-training, while retaining weights for previously available bands to minimize transfer disruption.
- **Multimodal/multitemporal fusion:**
  - For DINO-v2, DINO-v2 sat., DOFA, we add late fusion across modalities and time steps, using the *shared* and *monotemp* modes (see Sec. 3.1).
  - For CROMA, we adopt specialized multimodal and multitemporal fusion modes (see Sec. 7.4.6).
  - For SatMAE, Prithvi-EO 2.0, we apply early multitemporal fusion as in the original models, but we allow the handling of an arbitrary number of Sentinel-2 dates.
- **Temporal encoding:**
  - For DINO-v2, DINO-v2 sat., DOFA, CROMA, we inject the temporal encodings from Sec. 7.2.2 into the embedded tokens *after* the encoder. This minimizes the risk of transfer disruption.

  - For SatMAE, Prithvi-EO 2.0, we use the temporal encodings from the original models.
- **Input resizing:**
  - For all baseline FMs, input images are resized to match MAESTRO's token grid size while preserving the original patch size (see Tabs. 7 to 9).

All adapted baseline FMs natively use ViT backbones [19], as in MAESTRO. They also use the same classification and segmentation heads as in MAESTRO.

In Tab. 5, we report the model sizes along with the pre-training and fine-tuning modalities used for each adapted baseline FM.

Table 5. **Overview of the adapted baseline FMs.** We report model sizes, pre-training modalities, and modalities selected for fine-tuning.

Model	Model size	Pre-training modalities	Fine-tuning modalities			
			VHR	$S1_{ASC}/S1_{DES}$	S2	DEM/DSM
DINO-v2 [62]	Base	Natural RGB images (LVD-142M dataset)	✓	✗	✓	✗
DINO-v2 sat. [78]	Large	Maxar Vivid2 mosaic imagery	✓	✗	✓	✗
DOFA [102]	Base	Sentinel-1, Sentinel-2, Gaofen, NAIP, EnMaP	✓	✓	✓	✗
CROMA [25]	Base	Sentinel-1, Sentinel-2 (SSL4EO dataset)	✗	✓	✓	✗
Prithvi-EO-2.0 [74]	Large	Sentinel-2, Landsat	✗	✗	✓	✗
SatMAE [13]	Large	Sentinel-2 (fMoW RGB+Sentinel dataset)	✗	✗	✓	✗

7.4.3. DINO-v2

DINO-v2 [62] is a FM pre-trained on natural RGB images using a SSL teacher–student distillation framework, which combines global (image-level) and local (patch-level) objectives. Due to the domain gap between natural and EO imagery, transferring DINO-v2 to EO tasks is non-trivial.

To adapt it to our multimodal setting, we modify its patchification operation to support both VHR and Sentinel-2 inputs using two parallel patchification layers:

- On PASTIS-HD, FLAIR#2, and FLAIR-HUB, we retain the pre-trained RGB channel weights, while additional channels are initialized with near-zero values to minimize disruption, following [24].
- On TreeSatAI-TS, we found it beneficial to map the pre-trained RGB weights to the infrared colors (IRC) channels NIR, RED, and GREEN in the downstream task, while initializing the BLUE channel weights with near-zero values. We attribute this beneficial effect to two factors: (i) representations learned on natural RGB images transfer well to IRC aerial imagery, and (ii) the NIR channel plays a decisive role in TreeSatAI-TS, benefiting from being mapped to the transferred weights.

We experiment with two fusion modes for handling multimodality and multitemporality:

(i) *shared* (default), where a single encoder (initialized from DINO-v2's weights) processes all modalities and time steps independently, with shared weights across them;

(ii) *monotemp*, where modality-specific encoders (also initialized from DINO-v2) process each modality and time step independently.

Since DINO-v2 lacks native temporal handling, the temporal encodings from Sec. 7.2.2 are injected into the post-encoder token embeddings. The class token is kept in the model but discarded in its output.

7.4.4. DINO-v2 sat

DINO-v2 sat. [78] is based on the same architecture as DINO-v2 but pre-trained on Maxar's very high-resolution RGB satellite imagery. We apply the same adaptation protocol for this model as for DINO-v2.

7.4.5. DOFA

DOFA [102] is a FM pre-trained on multi-source EO data using the MAE framework and a dynamic channel handling via a hypernetwork that generates patch embedding weights based on input wavelengths (see Table 6). While DOFA is inherently flexible, adaptations are required to support multimodal and multitemporal fusion.

We implement four parallel patchification layers (initialized from the original patchification layer) to handle VHR, Sentinel-2, and Sentinel-1 in both ascending and descending orbits.

As with DINO-v2 and DINO-v2 sat., we explore both *shared* and *monotemp* fusion modes for handling multimodality and multitemporality.

Since the original model lacks native temporal handling, the temporal encodings from Sec. 7.2.2 are injected into the post-encoder token embeddings.

Table 6. **DOFA's wavelengths per modality.**

Modality	Bands	Wavelengths
Aerial	RED, GREEN, BLUE, NIR	0.64, 0.56, 0.48, 0.81
SPOT	RED, GREEN, BLUE	0.66, 0.56, 0.48
$S1_{ASC}$	VV, VH	5.405, 5.405
$S1_{DES}$	VV, VH	5.405, 5.405
S2	B02, B03, B04, B05, B06, B07, B08, B8A, B11, B12	0.490, 0.560, 0.665, 0.705, 0.740, 0.783, 0.842, 0.865, 1.610, 2.190

7.4.6. CROMA

CROMA [25] is a FM specifically designed for Sentinel imagery, featuring separate encoders for Sentinel-1 and Sentinel-2, coupled via a cross-encoder that fuses their intermediate representations. Since we fine-tune CROMA on its original modalities, only minimal adaptations are required.

We consider two fusion modes for handling multimodality and multitemporality:

(i) *late-croma* (cross-encoder disabled), where monotemporal Sentinel-1 and Sentinel-2 encoder outputs are concatenated across modalities and time steps before the classification/segmentation heads;

(ii) *inter-croma* (default), where monotemporal Sentinel-1 and Sentinel-2 encoder outputs are grouped into pairs, fused via the cross-encoder, and finally concatenated across pairs before the classification/segmentation heads.

In each case, the time series for Sentinel-1 in ascending and descending orbits are concatenated and passed as a single sequence to the Sentinel-1 encoder.

The temporal encodings from Sec. 7.2.2 are injected post-encoder.

Note that CROMA remains intrinsically monotemporal: *late-croma* performs only late multitemporal fusion, while *inter-croma* performs intermediate multitemporal fusion within individual Sentinel-1/Sentinel-2 pairs, but late fusion across pairs.

7.4.7. Prithvi-EO

Prithvi-EO 2.0 [74] is a FM pre-trained on a corpus of NASA's Harmonized Landsat Sentinel-2 (HLS) data at 30-meter resolution. The architecture is an adaptation of the MAE to satellite time series, employing token-based multitemporal fusion. The standard 2D patch and positional embeddings of the MAE architecture are replaced with 3D counterparts to natively handle spatiotemporal inputs.

In its original form, the model processes only six Sentinel-2 bands: RED, GREEN, BLUE, NIR, SWIR 1 & 2. To extend it to additional bands, we adapt the patchification layer to accept new bands. The pre-trained channel weights are retained, while the extra channels are initialized with near-zero values to minimize transfer disruption, following [24].

Although the original model was pre-trained on sequences of four Sentinel-2 dates, the official TerraTorch implementation does not impose a strict limitation on the number of dates. Using this implementation, we let the model process an arbitrary number of time steps.

We retain the same temporal encodings as in the original model.

7.4.8. SatMAE

Temporal SatMAE [13] is a pre-training framework that adapts the MAE to satellite time series, employing token-based multitemporal fusion.

In its original form, the model processes only three Sentinel-2 bands (RGB). To extend it to additional bands, we adapt the patchification layer to accept new bands. The pre-trained RGB channel weights are retained, while the extra channels are initialized with near-zero values to minimize transfer disruption, following [24].

A further limitation of the original model is its restriction to three Sentinel-2 dates. We remove this constraint and enable the model to process an arbitrary number of time steps.

We retain the same temporal encodings as in the original model.

7.5. Hyperparameter Tables

In this subsection, we provide tables reporting the exhaustive list of our hyperparameter values.

Table 7. **TreeSatAI-TS's hyperparameters.**

	Spatial extent of original tiles: 60 m *Spatial extent of crops: 60 m*			
	Aerial	$S1_{ASC}$	$S1_{DES}$	S2
Dataset's original resolution (m)	0.2	10	10	10
Image size (I_m)				
MAESTRO-intra/MAE/ViT	300	6	6	6
MAESTRO-cross	240	6	6	6
DINO-v2	210	✗	✗	42
DINO-v2 sat.	240	✗	✗	48
DOFA	240	48	48	48
CROMA	✗	24	24	24
Prithvi-EO-2.0/SatMAE	✗	✗	✗	48
Patch size (P_m)				
MAESTRO-intra/MAE/ViT	20	2	2	2
MAESTRO-cross	16	2	2	2
DINO-v2	14	✗	✗	14
DINO-v2 sat.	16	✗	✗	16
DOFA	16	16	16	16
CROMA	✗	8	8	8
Prithvi-EO-2.0/SatMAE	✗	✗	✗	16
Number of temporal bins (D_m)	1	4	4	16
Number of channels (C_m)	4	2	2	10
Multiplicative normalization factor	255	5	5	5×10^3
Use cloud/snow mask				✓(mask proba. > 0)
MAE/MAESTRO's band groups ($\mathcal{G}_m$)	RED, GREEN, BLUE NIR	VV VH	VV VH	B02, B03, B04, B05 B06, B07, B08, B8A B11, B12

Table 8. **PASTIS-HD's hyperparameters.**

	Spatial extent of original tiles: 1280 m *Spatial extent of crops: 160 m* *Spatial resolution of token grid of reference: 20 m*			
	SPOT	$S1_{ASC}$	$S1_{DES}$	S2
Dataset's original resolution (m)	1	10	10	10
Image size (I_m)				
MAESTRO-intra/MAE/ViT	160	16	16	16
MAESTRO-cross	160	16	16	16
DINO-v2	140	✗	✗	112
DINO-v2 sat.	160	✗	✗	128
DOFA	160	128	128	128
CROMA	✗	64	64	64
Prithvi-EO-2.0/SatMAE	✗	✗	✗	128
Patch size (P_m)				
MAESTRO-intra/MAE/ViT	16	2	2	2
MAESTRO-cross	16	2	2	2
DINO-v2	14	✗	✗	14
DINO-v2 sat.	16	✗	✗	16
DOFA	16	16	16	16
CROMA	✗	8	8	8
Prithvi-EO-2.0/SatMAE	✗	✗	✗	16
Number of temporal bins (D_m)	1	4	4	16
Number of channels (C_m)	3	2	2	10
Multiplicative normalization factor	255	20	20	1×10^4
Use cloud/snow mask				✗ (not avail.)
MAE/MAESTRO's band groups ($\mathcal{G}_m$)	RED, GREEN, BLUE	VV VH	VV VH	B02, B03, B04, B05 B06, B07, B08, B8A B11, B12

Table 9. **FLAIR#2's and FLAIR-HUB's hyperparameters.**

	Spatial extent of original tiles: 102.4 m *Spatial extent of crops: 102.4 m* *Spatial resolution of token grid of reference: 3.2 m*				
	Aerial	DEM/DSM	$S1_{ASC}$	$S1_{DES}$	S2
Present in FLAIR#2	✓	✓	✗	✗	✓
Dataset's original resolution (m)	0.2	0.2	10.24	10.24	10.24
Image size (I_m)					
MAESTRO-intra/MAE/ViT	512	512	10	10	10
MAESTRO-cross	512	✗	10	10	10
DINO-v2	448	✗	✗	✗	70
DINO-v2 sat.	512	✗	✗	✗	80
DOFA	512	✗	80	80	80
CROMA	✗	✗	40	40	40
Prithvi-EO-2.0/SatMAE	✗	✗	✗	✗	80
Patch size (P_m)					
MAESTRO-intra/MAE/ViT	16	32	2	2	2
MAESTRO-cross	16	✗	2	2	2
DINO-v2	14	✗	✗	✗	14
DINO-v2 sat.	16	✗	✗	✗	16
DOFA	16	✗	16	16	16
CROMA	✗	✗	8	8	8
Prithvi-EO-2.0/SatMAE	✗	✗	✗	✗	16
Number of temporal bins (D_m)	1	1	4	4	16
Number of channels (C_m)	4	2	2	2	10
Multiplicative normalization factor	255	1×10^3	5	5	5×10^3
Use cloud/snow mask					✓(mask proba. > 0)
MAE/MAESTRO's band groups ($\mathcal{G}_m$)	RED, GREEN, BLUE NIR	DEM, DSM	VV VH	VV VH	B02, B03, B04, B05 B06, B07, B08, B8A B11, B12

Table 10. **S2-NAIP urban's hyperparameters.**

	Spatial extent of original tiles: 640 m *Spatial extent of crops: 120 m*			
	NAIP – Aerial surrogate	NAIP – SPOT surrogate	S1	S2
Dataset's original resolution (m)	1.25	1.25	10	10
Image size (I_m)				
MAESTRO-cross	384	128	12	12
Patch size (P_m)				
MAESTRO-cross	16	16	2	2
Number of temporal bins (D_m)	1	1	4	16
Number of channels (C_m)	4	3	2	10
Multiplicative normalization factor	255	255	20	5×10^3
Use cloud/snow mask				✗ (not avail.)
MAE/MAESTRO's band groups ($\mathcal{G}_m$)	RED, GREEN, BLUE NIR	RED, GREEN, BLUE	VV VH	B02, B03, B04, B05 B06, B07, B08, B8A B11, B12

Table 11. **Data augmentation/regularization hyperparameters.**

Augmentation/Regularization	Phase used	Hyperparameter
Random spatial cropping	pre-train/probe/fine-tune	
Random time step selection	pre-train/probe/fine-tune	
D4 augmentation	pre-train/probe/fine-tune	
EMA [58]	fine-tune	$\alpha = 1 - (0.2 \times N_{\text{epochs}})^{-1}$

Table 12. **Optimizer hyperparameters.**

Phase	Hyperparameter	TreeSatAI-TS	PASTIS-HD	FLAIR#2	FLAIR-HUB	S2-NAIP urban
	Optimizer	AdamW	AdamW	AdamW	AdamW	AdamW
	LR schedule	cosine decay	cosine decay	cosine decay	cosine decay	cosine decay
	Warmup fraction	20%	20%	20%	20%	20%
	Weight decay	1×10^{-2}	1×10^{-2}	1×10^{-2}	1×10^{-2}	1×10^{-2}
	β_1	0.9	0.9	0.9	0.9	0.9
	β_2	0.99	0.99	0.99	0.99	0.99
Pre-train	Base learning rate ($\times\sqrt{3}$)	3×10^{-5}	3×10^{-5}	3×10^{-5}	3×10^{-5}	1×10^{-5}
	Final div. factor	1×10^{4}	1×10^{4}	1×10^{4}	1×10^{4}	1×10^{4}
	Batch size					
	Dataset frac. 100%	96	72	72	144	512
	Dataset frac. 20%	96	72	–	72	–
	Dataset frac. 5%	96	72	–	72	–
	Epochs					
	Dataset frac. 100%	100	100	100	100	15
	Dataset frac. 20%	200	200	–	100	–
	Dataset frac. 5%	400	400	–	200	–
Probe	Base learning rate ($\times\sqrt{3}$)	1×10^{-5}	1×10^{-5}	1×10^{-5}	1×10^{-5}	–
	Final div. factor	1×10^{4}	1×10^{4}	1×10^{4}	1×10^{4}	–
	Batch size					
	Dataset frac. 100%	96	48	48	96	–
	Dataset frac. 20%	96	48	–	48	–
	Dataset frac. 5%	96	48	–	48	–
	Epochs					
	Dataset frac. 100%	10	10	15	15	–
	Dataset frac. 20%	20	20	–	15	–
	Dataset frac. 5%	40	40	–	30	–
Fine-tune	Base learning rate ($\times\sqrt{3}$)	1×10^{-5}	1×10^{-5}	1×10^{-5}	1×10^{-5}	–
	Final div. factor	2	2	2	2	–
	Batch size					
	Dataset frac. 100%	96	48	48	96	–
	Dataset frac. 20%	96	48	–	48	–
	Dataset frac. 5%	96	48	–	48	–
	Epochs					
	Dataset frac. 100%	50	50	100	100	–
	Dataset frac. 20%	100	100	–	100	–
	Dataset frac. 5%	200	200	–	200	–

Table 13. **SSL hyperparameters.**

Hyperparameter	
Reconstruction loss	L_1 w/ patch-group-wise normalization (for some ablations: L_1 w/o normalization or L_1 w/ patch-wise normalization)
Masking ratio	75%
Probability modality-structured masking	0.25
Probability temporally-structured masking	0.25
Probability spatially-structured masking	0.25

8. Detailed Results

In this section, we report the detailed results on multimodal and multitemporal fusion (Sec. 8.1), multispectral fusion and target normalization (Sec. 8.2), and scaling with respect to pre-training and fine-tuning dataset size (Sec. 8.3).

8.1. Multimodal/Multitemporal Fusion

The detailed numbers of the evaluation of multimodal and multitemporal fusion modes with MAEs, ViTs, and baseline FMs can be found in Tab. 14, in addition to Fig. 3.

Table 14. **Evaluation of multimodal and multitemporal fusion modes for MAEs, ViTs, and baseline FMs**. We report the weighted F1 score (%) on TreeSatAI-TS and the mIoU (%) on PASTIS-HD and FLAIR-HUB 20%.

Model	Model size	Fusion mode	Modality groups	**TreeSatAI-TS**	**PASTIS-HD** fold I	**FLAIR-HUB** filt. 20% / split 1
MAE	Base	shared		76.1	66.1	62.5
MAE	Base	monotemp		77.0	66.0	62.4
MAE	Base	mod		78.4	68.9	63.3
MAE	Base	group	$S1_{ASC}$, $S1_{DES}$	78.5	68.8	**63.6**
MAE	Base	group	$S1_{ASC}$, $S1_{DES}$, S2	78.1	**69.1**	63.5
MAE	Base	group	$S1_{ASC}$, $S1_{DES}$, S2, VHR	78.2	68.1	61.4
MAE	Base	inter-group	$S1_{ASC}$, $S1_{DES}$	**78.8**	68.6	63.5
ViT	Base	shared		72.6	64.1	59.3
ViT	Base	monotemp		73.0	64.4	58.9
ViT	Base	mod		75.4	**64.6**	59.0
ViT	Base	group	$S1_{ASC}$, $S1_{DES}$	75.7	64.6	**59.5**
ViT	Base	group	$S1_{ASC}$, $S1_{DES}$, S2	**76.0**	64.2	59.1
ViT	Base	group	$S1_{ASC}$, $S1_{DES}$, S2, VHR	75.9	64.0	56.4
ViT	Base	inter-group	$S1_{ASC}$, $S1_{DES}$	75.6	64.5	58.8
DINO-v2 [62]	Base	shared		76.7	64.4	64.2
DINO-v2 [62]	Base	monotemp		76.4	63.7	63.9
DINO-v2 sat. [78]	Large	shared		76.3	64.0	**64.5**
DINO-v2 sat. [78]	Large	monotemp		76.6	63.1	64.0
DOFA [102]	Base	shared		76.1	62.9	62.8
DOFA [102]	Base	monotemp		75.9	63.2	64.1
CROMA [25]	Base	late-croma		69.8	64.9	41.9
CROMA [25]	Base	inter-croma		70.5	65.0	42.5
Prithvi-EO-2.0 [74]	Large	mod		75.6	66.2	42.5
SatMAE [13]	Large	mod		**76.9**	**66.6**	42.8

8.2. Multispectral Fusion/Target Normalization

The detailed numbers of the evaluation of multispectral fusion and target normalization choices with MAEs can be found in Tabs. 15 and 16, in addition to Fig. 4. We also report results after probing evaluation in Tabs. 17 and 18.

We observe a slight performance drop when using token-based fusion with patch-wise/patch-group-wise normalization compared to joint-token fusion with patch-group-wise normalization. This may be attributed to the combination of two factors: (i) certain Sentinel-2 band groups may be less relevant to the downstream tasks, and (ii) our classification/segmentation heads, which aggregate multimodal and multitemporal features through a single attentive pooling layer, are shallow and may struggle to filter out irrelevant tokens—thereby introducing noise into the final predictions.

Table 15. **Evaluation of multispectral fusion and target normalization choices for MAE-B models across pre-training dataset fraction**. We report the weighted F1 score (%) on TreeSatAI-TS and the mIoU (%) on PASTIS-HD.

Multispectral fusion	Target normalization	**TreeSatAI-TS** Pre-training dataset frac. 5%	20%	100%	**PASTIS-HD** fold I Pre-training dataset frac. 5%	20%	100%
Joint-token	No normalization	72.7	72.8	74.6	64.9	65.4	65.1
Joint-token	Patch-wise	74.4	75.5	77.4	66.7	67.1	67.5
Joint-token	Patch-group-wise	**76.3**	**77.4**	**78.5**	**67.3**	**68.1**	**68.8**
Token-based	Patch-wise/ Patch-group-wise	76.6	77.0	78.9	66.7	67.2	68.6

8.3. Scaling by Dataset Size

The detailed numbers of the performance with different pre-training/fine-tuning dataset fractions for MAE-B and ViT-B models are provided in Tab. 19, in addition to Fig. 5.

Table 16. **Evaluation of multispectral fusion and target normalization choices for MAE models across model size**. We report the weighted F1 score (%) on TreeSatAI-TS and the mIoU (%) on PASTIS-HD.

Multispectral fusion	Target normalization	**TreeSatAI-TS** Model size			**PASTIS-HD** fold I Model size		
		Small	Base	Large	Small	Base	Large
Joint-token	No normalization	74.9	74.6	74.9	64.9	65.1	65.0
Joint-token	Patch-wise	77.2	77.4	77.1	66.8	67.5	67.7
Joint-token	Patch-group-wise	**77.3**	**78.5**	**78.5**	**67.8**	**68.8**	**69.2**
Token-based	Patch-wise/ Patch-group-wise	76.9	78.9	79.0	67.0	68.6	69.5

Table 17. **Probing evaluation of multispectral fusion and target normalization choices for MAE-B models across pre-training dataset fraction**. We report the weighted F1 score (%) on TreeSatAI-TS and the mIoU (%) on PASTIS-HD.

Multispectral fusion	Target normalization	**TreeSatAI-TS** Pre-training dataset frac.			**PASTIS-HD** fold I Pre-training dataset frac.		
		5%	20%	100%	5%	20%	100%
Joint-token	No normalization	57.2	61.1	63.2	52.5	56.9	57.9
Joint-token	Patch-wise	62.9	65.1	67.7	58.2	59.7	61.2
Joint-token	Patch-group-wise	**64.8**	**68.6**	**69.3**	**59.0**	**60.2**	**61.2**
Token-based	Patch-wise/ Patch-group-wise	61.9	66.2	69.3	55.9	57.5	61.0

Table 18. **Probing evaluation of multispectral fusion and target normalization choices for MAE models across model size**. We report the weighted F1 score (%) on TreeSatAI-TS and the mIoU (%) on PASTIS-HD.

Multispectral fusion	Target normalization	**TreeSatAI-TS** Model size			**PASTIS-HD** fold I Model size		
		Small	Base	Large	Small	Base	Large
Joint-token	No normalization	57.3	63.2	64.1	54.1	57.9	58.3
Joint-token	Patch-wise	63.1	67.7	68.4	**57.6**	61.2	62.1
Joint-token	Patch-group-wise	**64.3**	**69.3**	**71.4**	55.1	**61.2**	**63.6**
Token-based	Patch-wise/ Patch-group-wise	61.7	69.3	69.4	53.8	61.0	63.9

Table 19. **Scaling of MAE-B and ViT-B models with different pre-training/fine-tuning dataset fractions**. We report the weighted F1 score (%) on TreeSatAI-TS and the mIoU (%) on PASTIS-HD and FLAIR-HUB for three fine-tuning dataset fractions: 5%, 20%, and 100%. For each fine-tuning fraction, we compare three pre-training settings: no pre-training, pre-training on the same fraction as fine-tuning, and pre-training on 100% of the data. Note that pre-training on the same fraction as fine-tuning and pre-training on 100% of the data become equivalent when fine-tuning on 100% of the data, hence the identical performance.

	TreeSatAI-TS Fine-tuning dataset frac.			**PASTIS-HD** fold I Fine-tuning dataset frac.			**FLAIR-HUB** split 1 Fine-tuning dataset frac.		
	5%	20%	100%	5%	20%	100%	5%	20%	100%
No pre-training	61.8	69.0	75.7	38.8	52.2	64.6	55.3	59.5	61.6
Pre-training fraction = Fine-tuning fraction	65.5	71.8	**78.5**	46.6	57.1	**68.8**	60.1	63.6	**64.9**
Pre-training fraction = 100%	**67.8**	**73.8**	**78.5**	**52.5**	**59.2**	**68.8**	**61.5**	**64.6**	**64.9**

9. Additional Results

In this section, we report additional ablations on our masking strategy (Sec. 9.1), our choices of encodings (Sec. 9.2), the impact of multitemporal components (Sec. 9.3), and the importance of each modality by dataset (Sec. 9.4).

In all these ablations, we consider MAE-B models and ViT-B models with the *group* fusion mode, grouping together the Sentinel-1 ascending and descending modalities. For multispectral data, we use *joint-token* fusion combined with *patch-group-wise* target normalization during reconstruction.

9.1. Masking Strategy

We examine the impact of various components of our masking strategy. Specifically, we assess the effect of including the structured masking steps (i-a), (i-b), and (i-c) detailed in Sec. 7.3.1 in the SSL pretext task.

We report results after fine-tuning in Tab. 20 and after probing evaluation in Tab. 21.

We find that structured masking yields only marginal improvements in fine-tuning performance. However, temporally structured masking leads to a notable gain in probing performance on PASTIS-HD. This suggests that structured masking may produce more useful representations when evaluated directly (i.e., without fine-tuning), although these benefits do not necessarily transfer after fine-tuning.

Table 20. **Evaluation of different masking choices for MAE-B models**. We report the weighted F1 score (%) on TreeSatAI-TS and the mIoU (%) on PASTIS-HD and FLAIR-HUB 20%. Arrows (↑ / ↓) indicate the difference compared to our default masking strategy with modality, spatial and temporal structure.

Masking structure			**TreeSatAI-TS**	**PASTIS-HD** fold I	**FLAIR-HUB** filt. 20% / split 1
Modality	Spatial	Temporal			
✓	✓	✓	**78.5**	**68.8**	**63.6**
✗	✓	✓	78.5 ↑0.0	68.4 ↓0.4	63.3 ↓0.3
✓	✗	✓	78.5 ↑0.0	68.6 ↓0.2	63.0 ↓0.6
✓	✓	✗	78.3 ↓0.2	68.5 ↓0.3	63.4 ↓0.2
✗	✗	✗	78.2 ↓0.3	68.4 ↓0.4	63.6 ↑0.0

Table 21. **Probing evaluation of different masking choices for MAE-B models**. We report the weighted F1 score (%) on TreeSatAI-TS and the mIoU (%) on PASTIS-HD and FLAIR-HUB 20%. Arrows (↑ / ↓) indicate the difference compared to our default masking strategy with modality, spatial and temporal structure.

Masking structure			**TreeSatAI-TS**	**PASTIS-HD** fold I	**FLAIR-HUB** filt. 20% / split 1
Modality	Spatial	Temporal			
✓	✓	✓	69.3	61.2	56.2
✗	✓	✓	69.8 ↑0.5	61.4 ↑0.2	56.3 ↑0.1
✓	✗	✓	69.6 ↑0.3	**61.5** ↑0.3	56.3 ↑0.1
✓	✓	✗	69.2 ↓0.1	57.7 ↓3.5	56.2 ↑0.0
✗	✗	✗	**69.9** ↑0.6	57.6 ↓3.6	**56.3** ↑0.1

9.2. Ablation of Temporal/Modality Encodings

We evaluate the effectiveness of the encoding strategies described in Sec. 7.2.2.

First, we evaluate the impact of removing temporal encodings. In this case, positional encodings occupy the full C_e dimensions, and aggregated encodings are formed solely from them.

As shown in Tab. 22, including temporal encodings is critical on tasks with strong multitemporal components, such as TreeSatAI-TS and PASTIS-HD. Such temporal encodings constitute the only mechanism that contextualizes tokens across time steps. Without them, the model "sees" multiple dates but must infer their temporal positions and ordering. This ambiguity is especially detrimental for tasks that are strongly tied to multitemporal signatures, e.g. through phenological discrimination of tree species in TreeSatAI-TS or crop types in PASTIS-HD.

Next, we investigate the addition of modality encodings, implemented as learnable modality-specific parameters of dimension 8. When including these modality encodings, we reduce the positional encoding dimensionality to $C_e - 16$ and form

the aggregated encodings by concatenating the modality, temporal, and positional encodings of dimensions 8, 8, and $C_e - 16$, respectively.

As shown in Tab. 22, including explicit modality encodings has only a slight and inconsistent effect. This confirms that modality-specific tokenizers and learnable modality-specific `[mask]` tokens already provide sufficient modality differentiation. The small performance drop observed on FLAIR-HUB 20% may stem from the resulting reduction in positional encoding dimensionality.

Table 22. **Ablation of temporal and modality encodings for MAE-B and ViT-B models**. We report the weighted F1 score (%) on TreeSatAI-TS and the mIoU (%) on PASTIS-HD and FLAIR-HUB 20%. Arrows (↑ / ↓) indicate the difference compared to using temporal encodings but not modality encodings.

Model	Temporal enc.	Modality enc.	**TreeSatAI-TS**	**PASTIS-HD** fold I	**FLAIR-HUB** filt. 20% / split 1
MAE	✓	✗	78.5	68.8	**63.6**
MAE	✗	✗	77.8 ↓0.7	67.7 ↓1.1	63.3 ↓0.3
MAE	✓	✓	**78.7** ↑0.2	**69.0** ↑0.2	63.2 ↓0.4
ViT	✓	✗	**75.7**	**64.6**	**59.5**
ViT	✗	✗	74.7 ↓1.0	63.6 ↓1.0	59.2 ↓0.3
ViT	✓	✓	75.6 ↓0.1	64.6 ↑0.0	58.9 ↓0.6

9.3. Importance of Multitemporal Components

We examine the impact of various multitemporal components.

First, we evaluate the effect of varying the number of temporal bins D_m for the Sentinel-2 and Sentinel-1 ascending and descending modalities. As shown in Tab. 23, the number of temporal bins D_m for Sentinel-2 is critical when this modality drives performance, as in TreeSatAI-TS and PASTIS-HD (see Sec. 9.4). Notably, the performance gap between $D_m = 16$ and $D_m = 1$ for Sentinel-2 is much larger than that between $D_m = 1$ and $D_m = 0$. This indicates that it is the multitemporal dynamics of Sentinel-2, rather than its monotemporal component, that is essential. In contrast, varying D_m for Sentinel-1 has only a minor effect, reflecting its smaller role (see Sec. 9.4).

Table 23. **Importance of the number of temporal bins for MAE-B and ViT-B models**. We report the weighted F1 score (%) on TreeSatAI-TS and the mIoU (%) on PASTIS-HD and FLAIR-HUB 20%. Arrows (↑ / ↓) indicate the difference compared to using 16 temporal bins for the Sentinel-2 modality and 4 temporal bins for the Sentinel-1 ascending and descending modalities.

Model	Number of temporal bins			**TreeSatAI-TS**	**PASTIS-HD** fold I	**FLAIR-HUB** filt. 20% / split 1
	$S1_{ASC}$	$S1_{DES}$	S2			
MAE	4	4	16	**78.5**	68.8	**63.6**
MAE	4	4	4	75.3 ↓3.2	64.3 ↓4.5	62.4 ↓1.2
MAE	4	4	1	73.1 ↓5.4	49.3 ↓19.5	61.8 ↓1.8
MAE	4	4	0	72.6 **↓5.9**	44.9 **↓23.9**	61.6 **↓2.0**
MAE	1	1	16	78.3 ↓0.2	68.8 ↑0.0	63.5 ↓0.1
MAE	0	0	16	78.2 ↓0.3	**69.2** ↑0.4	63.1 ↓0.5
ViT	4	4	16	75.7	**64.6**	59.5
ViT	4	4	4	70.6 ↓5.1	61.2 ↓3.4	58.4 ↓1.1
ViT	4	4	1	66.1 ↓9.6	46.5 ↓18.1	57.6 ↓1.9
ViT	4	4	0	65.8 **↓9.9**	41.5 **↓23.1**	57.4 **↓2.1**
ViT	1	1	16	**75.8** ↑0.1	64.6 ↑0.0	**59.8** ↑0.3
ViT	0	0	16	75.7 ↑0.0	64.4 ↓0.2	59.2 ↓0.3

Next, we assess the impact of disabling the data augmentation associated with random time step selection within temporal bins. As shown in Tab. 24, disabling this data augmentation generally degrades performance, although the magnitude of the degradation varies across datasets.

Table 24. **Importance of random time step selection for MAE-B and ViT-B models**. We report the weighted F1 score (%) on TreeSatAI-TS and the mIoU (%) on PASTIS-HD and FLAIR-HUB 20%. Arrows (↑ / ↓) indicate the difference compared to enabling random time step selection.

Model	Number of temporal bins			Random time step selection	**TreeSatAI-TS**	**PASTIS-HD** fold I	**FLAIR-HUB** filt. 20% / split 1
	$S1_{ASC}$	$S1_{DES}$	S2				
MAE	4	4	16	✓	78.5	68.8	63.6
MAE	4	4	16	✗	**78.8** ↑0.3	68.2 ↓0.6	**63.7** ↑0.1
ViT	4	4	16	✓	**75.7**	**64.6**	**59.5**
ViT	4	4	16	✗	75.4 ↓0.3	63.9 ↓0.7	58.7 ↓0.8

9.4. Importance of each Modality by Dataset

We present an ablation study on modality removal for each dataset.

As shown in Tab. 25, the best performance is generally obtained when all modalities are included. The Sentinel-2 modality has a strong influence on the TreeSatAI-TS and PASTIS-HD tasks, which are heavily tied to multitemporal dynamics. Additionally, on FLAIR-HUB, the aerial modality from which the ground truth is derived markedly affects performance.

The slight performance drop observed when omitting Sentinel-1 on PASTIS-HD may stem from the combination of two factors: (i) its lower task relevance compared to Sentinel-2, and (ii) the relative shallowness of segmentation heads, which may struggle to filter out irrelevant tokens—thereby introducing noise into the final predictions.

Table 25. **Ablation study on modality removal for MAE-B and ViT-B models**. We report the weighted F1 score (%) on TreeSatAI-TS and the mIoU (%) on PASTIS-HD and FLAIR-HUB 20%. Arrows (↑ / ↓) indicate the difference compared to including all modalities.

Model	**TreeSatAI-TS**				**PASTIS-HD** fold I				**FLAIR-HUB** filt. 20% / split 1				
	Aerial	S1	S2	wF1 (%)	Spot	S1	S2	mIoU (%)	Aerial	S1	S2	DEM/DSM	mIoU (%)
MAE	✓	✓	✓	**78.5**	✓	✓	✓	68.8	✓	✓	✓	✓	**63.6**
MAE	✗	✓	✓	76.8 ↓1.7	✗	✓	✓	68.6 ↓0.2	✗	✓	✓	✓	52.2 ↓**11.4**
MAE	✓	✗	✓	78.2 ↓0.3	✓	✗	✓	**69.2** ↑0.4	✓	✗	✓	✓	63.1 ↓0.5
MAE	✓	✓	✗	72.6 ↓**5.9**	✓	✓	✗	44.9 ↓**23.9**	✓	✓	✗	✓	61.6 ↓2.0
MAE									✓	✓	✓	✗	62.7 ↓0.9
ViT	✓	✓	✓	**75.7**	✓	✓	✓	**64.6**	✓	✓	✓	✓	**59.5**
ViT	✗	✓	✓	75.3 ↓0.4	✗	✓	✓	64.5 ↓0.1	✗	✓	✓	✓	47.9 ↓**11.6**
ViT	✓	✗	✓	75.7 ↑0.0	✓	✗	✓	64.4 ↓0.2	✓	✗	✓	✓	59.2 ↓0.3
ViT	✓	✓	✗	65.8 ↓**9.9**	✓	✓	✗	41.5 ↓**23.1**	✓	✓	✗	✓	57.4 ↓2.1
ViT									✓	✓	✓	✗	58.3 ↓1.2

10. Inference Results

In this section, we report inference results from *intra-dataset* MAESTRO and supervised ViTs on the segmentation tasks of PASTIS-HD and FLAIR-HUB.

We consider *intra-dataset* MAESTRO-B and ViT-B models with the *group* fusion mode, grouping together the Sentinel-1 ascending and descending modalities. For multispectral data, we use *joint-token* fusion along with *patch-group-wise* target normalization during reconstruction.

Results are reported in Fig. 7 and Fig. 8 for PASTIS-HD and FLAIR-HUB, respectively. For each tile, we display the VHR imagery, the Sentinel-2 imagery, the MAESTRO prediction, the ViT prediction, and the corresponding ground truth. Examples are randomly sampled from the test set.

In Fig. 7, MAESTRO produces precise segmentation masks on PASTIS-HD, closely matching the boundaries of parcel and tree-covered areas. Its predictions are more spatially coherent than those of ViTs and better differentiate crop types.

In Fig. 8, although the complex scenes from FLAIR-HUB introduce prediction ambiguities, MAESTRO still delivers sharper and more accurate object delineations than ViTs. It also performs better on classes such as brushwood and water, and more effectively separates impervious from pervious surfaces.

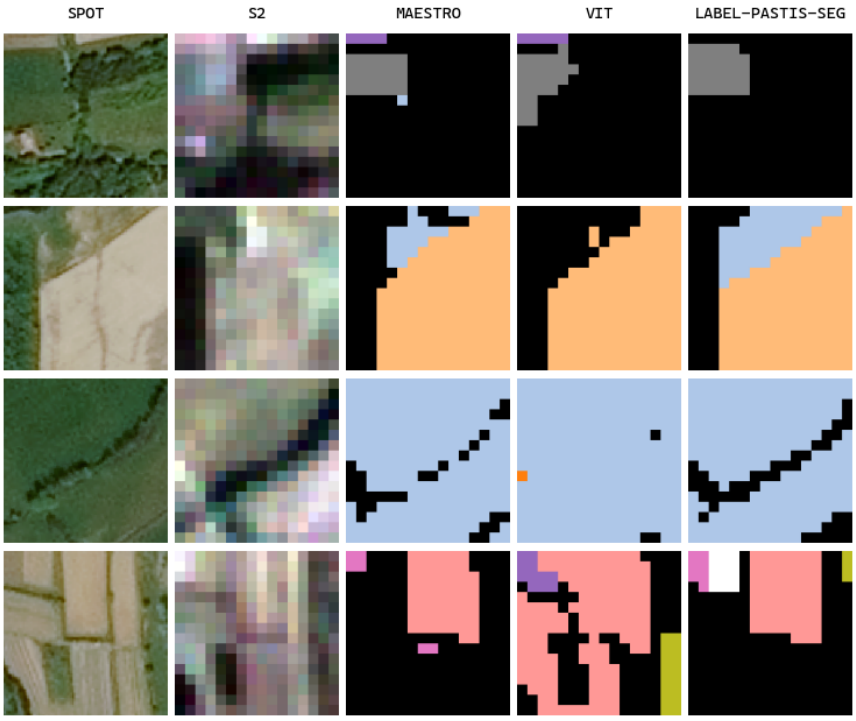


Figure 7. **Inference results from intra-dataset MAESTRO-B and ViT-B models on PASTIS-HD**. For Sentinel-2 imagery, we report the pixel-wise median across temporal bins. White parcels correspond to areas with missing annotations (void labels).

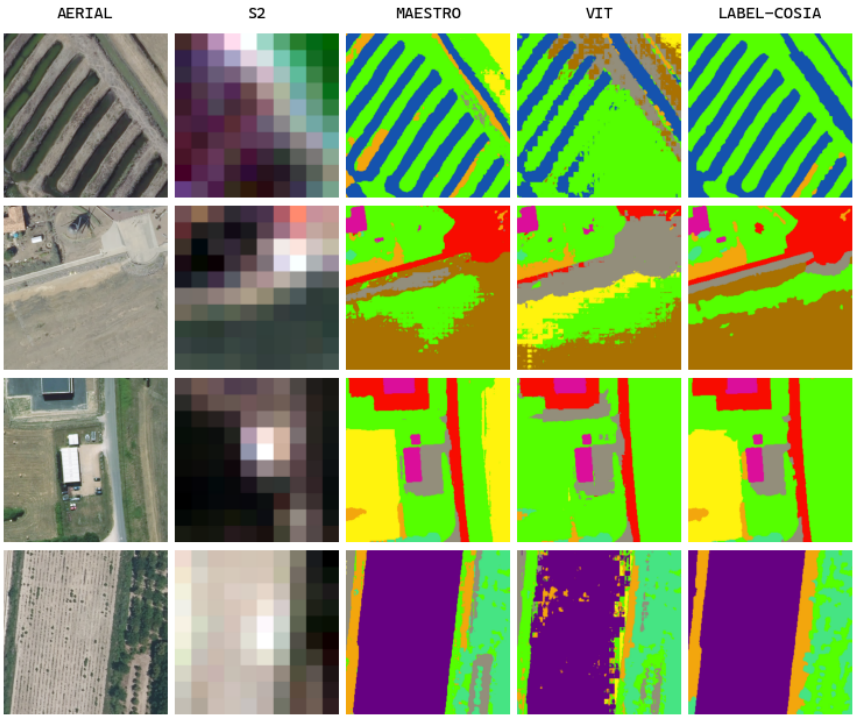


Figure 8. **Inference results from intra-dataset MAESTRO-B and ViT-B models on FLAIR-HUB**. For Sentinel-2 imagery, we report the pixel-wise median across temporal bins.

11. Computational Costs

In this section, we report our overall computational costs in floating-point operations (FLOPs) (Sec. 11.1) and GPU hours (Sec. 11.2)

11.1. Computational Costs in FLOPs

We report FLOPs per forward pass (for a single batch element) separately for SSL pre-training (Sec. 11.1.1) and probing/fine-tuning (Sec. 11.1.2). While forward FLOPs are identical between probing and fine-tuning, probing is substantially more efficient when backward FLOPs are taken into account.

We first calculate the number of multiply-accumulate operations (MACs), then convert to FLOPs using the relation FLOPs = $2 \times$ MACs. Following standard practice, we include only operations from model components with significant MAC contributions, excluding element-wise operations such as normalizations, nonlinearities, biases, and component-wise summations.

We consider the *group* fusion mode, grouping the Sentinel-1 ascending and descending modalities together. For multi-spectral data, we evaluate both *joint-token* and *token-based* multispectral fusion.

11.1.1. Pre-training FLOPs

For each modality m, let L_m denote its sequence length. With *joint-token* multispectral fusion, the sequence length is $L_m = (I_m/P_m)^2 D_m$, where I_m is the image size, P_m the patch size, and D_m the number of temporal bins. With *token-based* multispectral fusion, the sequence length becomes $L_m = (I_m/P_m)^2 D_m |\mathcal{G}_m|$, where $|\mathcal{G}_m|$ is the cardinality of the set of band groups mapped to different tokens.

For a modality m that is not grouped with others via early multimodal fusion, the MACs in the *encoder* and *decoder* are:

$$\text{MACs}^{\text{enc}} = \left(12\lfloor(1-M)L_m\rceil C_e^2 + 2\lfloor(1-M)L_m\rceil^2 C_e\right) N_e,$$
$$\text{MACs}^{\text{dec}} = \left(12 L_m C_d^2 + 2 L_m^2 C_d\right) N_d,$$

where M is the fraction of masked tokens, N_e and N_d are the number of layers in the encoder and decoder, while C_e and C_d are the latent dimensions in the encoder and decoder.

For the grouped Sentinel-1 ascending and descending modalities, MACs^{enc} and MACs^{dec} are computed by replacing L_m with the *sum of sequence lengths* of the two modalities.

The *dense* layers that project from the encoder space to the decoder space contribute:

$$\text{MACs}^{\text{enc-to-dec}} = \sum_m \lfloor(1-M)L_m\rceil C_e C_d.$$

The *patchify* and *unpatchify* operations contribute:

$$\text{MACs}^{\text{patchify}} = \sum_m I_m^2 D_m C_m C_e,$$
$$\text{MACs}^{\text{unpatchify}} = \sum_m I_m^2 D_m C_m C_d,$$

where C_m is the number of channels for each modality m.

Putting everything together, we compute in Tab. 26 the forward FLOPs (FLOPs = $2 \times$ MACs) for MAE models during pre-training on the five considered datasets. FLOPs are computed for three model sizes with both *joint-token* and *token-based* multispectral fusion.

The computation is based on the default configurations in Tab. 7, Tab. 8, Tab. 9, and Tab. 10 and the values $M = 0.75$ for the fraction of masked tokens, and $N_e = \{12, 12, 24\}$, $C_e = \{384, 768, 1024\}$, $N_d = \{2, 3, 4\}$, and $C_d = \{512, 512, 512\}$ for model sizes $\{\text{Small}, \text{Base}, \text{Large}\}$.

Table 26. **FLOPs for MAE models during pre-training on the five considered datasets**. We report the forward GFLOPs (for a single batch element) for three model sizes with both joint-token and token-based multispectral fusion. Multipliers (x) indicate the GFLOPs increase with token-based multispectral fusion compared to joint-token multispectral fusion.

Multispectral fusion	TreeSatAI-TS			PASTIS-HD			FLAIR#2			FLAIR-HUB			S2-NAIP urban		
	Small	Base	Large	Small	Base	Large	Small	Base	Large	Small	Base	Large	Small	Base	Large
Joint-token	11	29	80	45	112	308	49	118	320	54	131	355	36	91	252
Token-based	27	67	188	153	347	891	114	268	705	125	294	778	100	236	635
× FLOPS w/ token-based	×2.4	×2.4	×2.4	×3.4	×3.1	×2.9	×2.4	×2.3	×2.2	×2.3	×2.2	×2.2	×2.7	×2.6	×2.5

11.1.2. Probing/Fine-tuning FLOPs

We reuse the same notation as in Sec. 11.1.1. As before, the sequence length is given by $L_m = (I_m/P_m)^2 D_m$ for *joint-token* multispectral fusion, and by $L_m = (I_m/P_m)^2 D_m|\mathcal{G}_m|$ for *token-based* multispectral fusion.

For a modality m that is not grouped with others via early multimodal fusion, the MACs in the *encoder* are:

$$\text{MACs}^{\text{enc}} = \left(12 L_m C_e^2 + 2 L_m^2 C_e\right) N_e.$$

For the grouped Sentinel-1 ascending and descending modalities, MACs^{enc} is computed by replacing L_m with the *sum of sequence lengths* of the two modalities.

As before, the *patchify* operations contribute:

$$\text{MACs}^{\text{patchify}} = \sum_m I_m^2 D_m C_m C_e.$$

The attentive pooling operations in the classification and segmentation heads contribute:

$$\text{MACs}^{\text{attn-pool}} = 2 \sum_m L_m C_e^2.$$

The final *dense* projections in the classification and segmentation heads contribute:

$$\text{MACs}^{\text{proj-cls}} = C_e C_{\text{cls}},$$
$$\text{MACs}^{\text{proj-seg}} = L_{\text{ref}} C_e C_{\text{seg}},$$

where L_{ref} is the sequence length of the spatial *token grid of reference* (see Sec. 3.3), while C_{cls} and C_{seg} are the number of semantic classes for the classification and segmentation tasks, respectively—including classes ignored during loss and metric computations (see Sec. 7.1).

Putting everything together, we compute in Tab. 27 the forward FLOPs (FLOPs = 2 × MACs) for MAE/ViT models during probing/fine-tuning on the four evaluated datasets. FLOPs are computed for three model sizes with both *joint-token* and *token-based* multispectral fusion.

The computation is based on the default configurations in Tab. 7, Tab. 8, and Tab. 9 and the values $N_e = \{12, 12, 24\}$ and $C_e = \{384, 768, 1024\}$ for model sizes $\{\text{Small}, \text{Base}, \text{Large}\}$.

Table 27. **FLOPs for MAE/ViT models during probing/fine-tuning on the four evaluated datasets**. We report the forward GFLOPs (for a single batch element) for three model sizes with both joint-token and token-based multispectral fusion. Multipliers (x) indicate the GFLOPs increase with token-based multispectral fusion compared to joint-token multispectral fusion.

Multispectral fusion	TreeSatAI-TS			PASTIS-HD			FLAIR#2			FLAIR-HUB		
	Small	Base	Large	Small	Base	Large	Small	Base	Large	Small	Base	Large
Joint-token	21	79	276	95	331	1125	97	339	1150	106	375	1276
Token-based	52	192	665	374	1110	3584	257	816	2695	277	891	2954
× FLOPS w/ token-based	×2.5	×2.4	×2.4	×3.9	×3.4	×3.2	×2.6	×2.4	×2.3	×2.6	×2.4	×2.3

11.2. Computational Costs in GPU hours

The overall GPU hours required to run all the experiments shown in this paper are reported in Tab. 28.

All our experiments were run on SLURM clusters, with either V100, A40, A100, or H100 GPU nodes. Training was performed in mixed precision.

Table 28. **Overall computational costs in GPU hours**.

	V100	**A40**	**A100**	**H100**
Hours	19,682	8,102	2,714	268