GeoSANE: Learning Geospatial Representations from Models, Not Data

Supplementary Material

8. Effect of Remote Sensing Fine-tuning on GeoSANE Model Generation.

To assess the contribution of remote sensing–specific fine-tuning in GeoSANE's latent weight space, we compare models generated from two variants of the backbone: (1) `CV-only`, where GeoSANE is trained only on general computer vision models [12], and (2) `CV+RS`, where GeoSANE is additionally finetuned on our remote sensing (RS) model collection. For each setting, we generate weights for the same target architectures (ViT-L) and finetune them identically on downstream tasks. This allows us to isolate the effect of training GeoSANE on remote sensing weights on the quality of the generated models. Results are shown in Table 8.

When comparing Table 8 with the results in Table 5 (models prompt vs. GeoSANE-generated models), we observe that models generated without remote sensing fine-tuning (`CV-only`) perform similarly to their ImageNet-pretrained model prompts. Since these model prompts are themselves ImageNet models, the latent space trained only on computer vision models provides limited additional benefit. In contrast, with additional training on remote sensing model weights `CV+RS`, GeoSANE-generated models clearly outperform their model prompts. This demonstrates that training on remote sensing model weights is essential: it injects geospatial structure into the latent space and enables the generation of initializations that outperform ImageNet-pretrained prompts.

Table 8. Downstream performance of models generated by GeoSANE trained without (`CV-only`) and with (`CV+RS`) remote sensing model weights.

Dataset	CV-only	CV+RS	Δ
EuroSAT	97.8	**99.1**	+1.3
RESISC45	93.4	**96.5**	+3.1
fMoW	53.1	**58.9**	+5.8
Sen12Flood	82.9	**85.2**	+2.3
Cal. Wildfires	92.2	**94.9**	+2.7
BigEarthNet	83.0	**88.7**	+5.7
DFC2020	48.6	**54.3**	+5.7
Spacenet1	75.3	**78.2**	+2.9
Sen1Floods11	86.0	**89.6**	+3.6
DIOR	73.7	**79.0**	+5.3

9. Downstream Datasets Sizes

Table 9 provides an overview of the evaluation datasets used in this work, covering scene classification, segmentation, and object detection tasks. For each dataset, we report the number of classes, label type, input channels, and number of samples. We also include the spatial resolution at which we train our models; when datasets provide images at different native resolutions, we uniformly resize them to the sizes listed in the table to ensure consistent training.

10. Additional Details on Model Collection Retrieval and Automatic Loader

Listing 1. Tags and keywords used for models retrieval.

```
[remote sensing, remote-sensing, NDVI, DSM,
remotesensing, earth observation, enMAP,
earth-observation, EO, satellite, LiDAR,
satellites, satellite imagery, S1, S2,
satellite-imagery, aerial, aerial imagery,
aerial-imagery, Sentinel-1, Sentinel 1,
Sentinel-2, Sentinel 2, SAR, landsat, MODIS,
synthetic-aperture-radar, multispectral,
multi-spectral, hyperspectral, cloud-mask,
hyper-spectral, enmap, vegetation-index,
AVIRIS, VIIRS, Pleiades, PlanetScope,
WorldView, drone, UAV, CubeSat, rsfm, RSFM,
satMAE, scalemae, satmae, dofa, optical,
radar, landcover, land-cover, land cover,
land use, land-use, urban mapping,
urban land cover, deforestation, snow cover,
flood detection, flood mapping, wildfire,
fire detection, glacier monitoring, DFC2023,
hazard mapping, coastal erosion, crop
monitoring, precision agriculture, air
quality, natural disasters, marine debris,
disaster response, soil sealing, methane
detection, building extraction, road
extraction, ai4eo, ml4eo, torchgeo, eo-learn,
rasterio, geopandas, earthengine, EuroSAT,
BigEarthNet, xView, FloodNet, SpaceNet,
so2sat, bigearthnet, brick-kiln, forestnet,
pv4ger, RESISC-45, pv4ger-seg, chesapeake,
cashew-plant, Crop Types, NeonTree, Cattles,
SegMunich, UC Merced, AID, FAIR1M, DIOR,
iSAID, ISPRS Potsdam, LEVIR-CD, BurnScars,
MADOS, PASTIS, Sen1Floods11, DynamicEarthNet,
FiveBillionPixels, CTM-SS, SpaceNet7, NAIP,
AI4Farms, METER-ML, fMoW, MLRSNet, WHU-RS19,
Optimal-31, AiRound, CV-BrCT, SpaceNet2,
INRIA Aerial, GID-15, DFC2020, Dynamic World,
MARIDA, WHU Aerial, Vaihingen, OSCD, DSIFN]
```

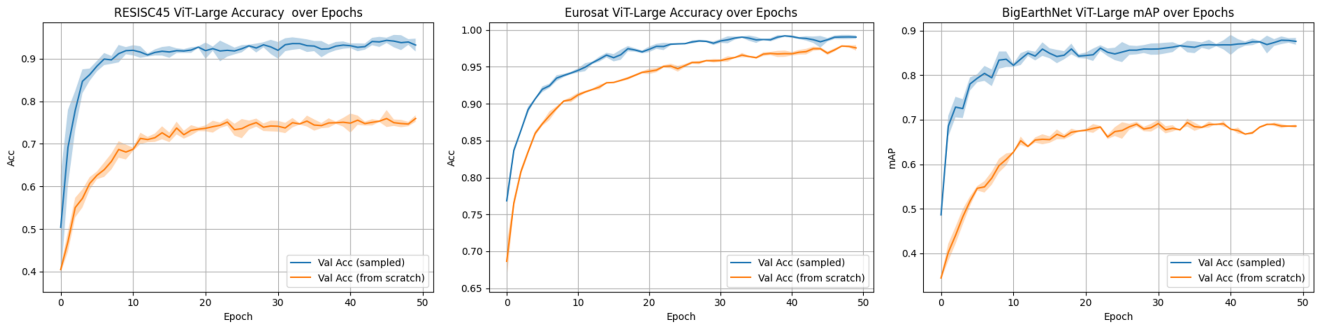

Figure 5. Convergence comparison between GeoSANE-initialized models and models trained from scratch.

Task	Dataset	# of classes	Labels (per image)	Size	Channels	# of samples
Classification	RESISC45 [5]	45	single	256×256	RGB	31.5K
	EuroSAT [22]	10	single	64×64	Multispectral	27K
	fMoW [6]	63	single	512×512	Multispectral	$> 1M$
	BigEarthNet [49]	19	multiple	120×120	Multispectral	590K
	Sen12Flood [36]	2 (binary)	single	256×256	SAR	16K
	California Wildfires [4]	2 (binary)	single	224×224	SWIR	20K
Segmentation	DFC2020 [40]	8	multiple	256×256	Multispectral	5K
	Sen1Floods11 [2]	2 (binary)	single	512×512	SAR	5K
	Spacenet1 [10]	2 (binary)	single	400×432	RGB	7K
Obj. Detection	DIOR [28]	20	multiple	800×800	RGB	23.5K

Table 9. Overview of datasets, tasks, label types, and channels.

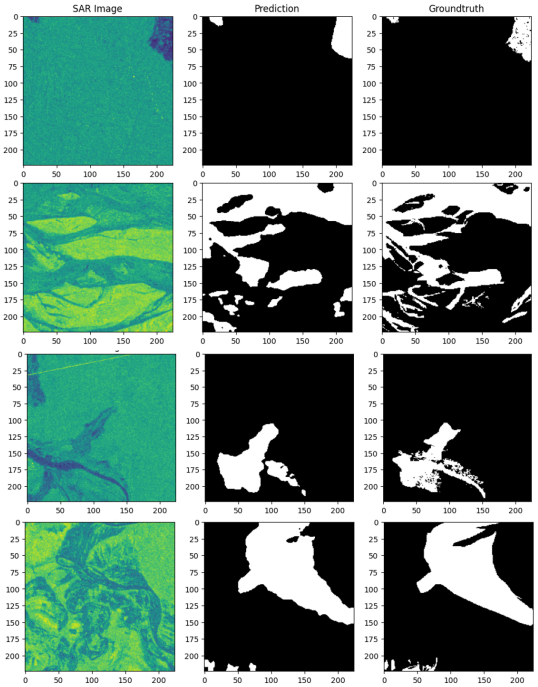

Figure 6. Qualitative Results of the Flood Segmentation task on the Sen1Floods11 dataset

As outlined in the main paper, we build the GeoSANE model collection by querying the HuggingFace Hub using a comprehensive set of modality-, task-, and dataset-specific keywords (see Listing 1). Here we provide the additional technical details required for full reproducibility. Because many remote sensing repositories on HuggingFace lack consistent metadata [24], we extend the search beyond tags by matching the same keyword vocabulary against repository names, model filenames, and model-card text. This procedure allows us to recover models even when authors provide incomplete or missing tags. The full list of retrieval keywords used in our search is included above.

A second practical challenge is that a substantial fraction of the collected models (particularly those originating from TorchGeo, SSL4EO, or various institutional releases) do not provide configuration files or explicit architectural specifications. To ensure that all such models can be loaded in a unified manner, we implement an automatic architecture reconstruction mechanism. Instead of relying on external configs, the loader infers the backbone type and input dimensionality directly from the structure of the `state_dict`. Convolutional backbones (e.g., ResNet-type models) are identified through their stem convolutions, whose tensor shapes reveal both the expected number of input channels and the architectural family. Transformer-based models (e.g., ViT, Swin) are de-

Figure 7. Qualitative Results of the Object Detection task on DIOR dataset

tected through their patch-embedding layers, from which the loader extracts both `in_chans` and the embedding dimension. When the checkpoint structure remains ambiguous, we fall back on controlled filename heuristics (e.g., detecting multispectral, SAR, RGB, or mission-specific Sentinel/Landsat naming patterns) to infer sensing modality and band count. Using these inferred attributes, the loader reconstructs the closest matching architecture from `timm` or `segmentation_models_pytorch`, and loads the checkpoint. This procedure enables GeoSANE to handle heterogeneous, partially documented checkpoints in a consistent and fully automated way. All loader code will be released alongside the final model collection.

11. Qualitative Results

Aside from quantitative results, we also provide qualitative examples. Following the same procedure described in the experimental setup, we first generate a Swin-B backbone with GeoSANE, then attach the appropriate task-specific head (a Faster R-CNN detection head for DIOR, or a segmentation head for Sen1Floods11). Figure 7 shows some object detection outputs on the DIOR dataset, while Figure 6 displays semantic flood segmentation results on Sen1Floods11.

12. UMAP Visualization of the Latent Weight Space

To better understand the structures learned by GeoSANE, we visualize (Figure 4) the latent representations of all models in our remote sensing collection using UMAP. For each model, we extract its full latent embedding sequence, sample 100 tokens randomly, and project these vectors to 2D using UMAP. When colored by architecture (top), the embedding shows clear structural organization: models sharing similar backbone types, such as ViTs, UNets, Swins or ResNets form well-separated clusters. This indicates that GeoSANE's latent space preserves meaningful distinctions between architectural families. When colored by sensing modality (bottom), the same embedding shows modality-related structure: models having similar inputs (e.g., SAR or Landsat) tend to appear in nearby regions of the space. Together, these observations show that GeoSANE organizes heterogeneous remote sensing models into meaningful latent groups.

13. Convergence Behavior of GeoSANE-generated Models

We study the convergence behavior of models initialized with GeoSANE compared to models trained from scratch. Figure 5 reports the validation performance over epochs for three representative datasets (RESISC45, EuroSAT, BigEarthNet), all using a ViT-L backbone. In every case, GeoSANE-generated initializations achieve strong accuracy within the first few epochs and maintain a consistent lead throughout training. This shows that GeoSANE does not only produce models competitive with remote sensing foundation models, but it also consistently accelerates optimization; which is an important advantage in settings with limited compute or training budgets.

14. Model Checkpoints Used for Training

Table 10 provides the complete list of model checkpoints used to train GeoSANE. For each model, we report the primary task (Table 12) and the sensing modalities (Table 11) used during training when this information is available from the original repository or documentation.

Checkpoint	Checkpoint	Checkpoint
jaychempan__EarthSynth	azdin__llava-onevision-weather-dora	mayrajeo__marine-vessel-yolo
torchgeo__presto	azdin__llava-onevision-weather-qlora	torchgeo__core-dino
DevPanda004__PrithviFlood	azdin__qwen2-vl-weather-adalora	torchgeo__delineate-anything
Mahadih534__YoloV8-VisDrone	azdin__qwen2-vl-weather-dora	torchgeo__delineate-anything-s
Mahadih534__yolov8_ship_det_satellite	banghyunmin__Thermal_Video_Detection	torchgeo__yolo11s_marine_vessel_detection
RedbeardNZ__LatentSync-1.6	banghyunmin__thermal-people-yolov11n	wangyi111__Copernicus-FM
azdin__llava-onevision-weather-adalora	mayrajeo__marine-vessel-detection-yolov8	IGNF__FLAIR-HUB_LC-A_IR_convnextv2base-unet
IGNF__FLAIR-HUB_LC-A_IR_convnextv2base-upernet	IGNF__FLAIR-HUB_LC-A_IR_convnextv2tiny-upernet	IGNF__FLAIR-HUB_LC-A_IR_swinbase-unet
IGNF__FLAIR-HUB_LC-A_IR_swinbase-upernet	IGNF__FLAIR-HUB_LC-A_IR_swinlarge-upernet	IGNF__FLAIR-HUB_LC-A_IR_swinsmall-upernet
IGNF__FLAIR-HUB_LC-A_IR_swintiny-upernet	IGNF__FLAIR-HUB_LC-A_RGB_swinbase-upernet	IGNF__FLAIR-HUB_LC-A_RGB_swinlarge-upernet
IGNF__FLAIR-HUB_LC-A_RGB_swinsmall-upernet	IGNF__FLAIR-HUB_LC-A_RGB_swintiny-upernet	IGNF__FLAIR-HUB_LC-D_swinbase-upernet
IGNF__FLAIR-HUB_LC-F_swinbase-upernet	IGNF__FLAIR-HUB_LC-I_swinbase-upernet	IGNF__FLAIR-HUB_LC-L_swinbase-upernet
IGNF__FLAIR-HUB_LPIS-A_swinbase-upernet	IGNF__FLAIR-HUB_LPIS-I_swinbase-upernet	IGNF__FLAIR-HUB_LPIS-J_swinbase-upernet
Jabasingh__VCTI	torchgeo__core-dino	torchgeo__satlas
chrimerss__flood-foundation-prithvi-100m	torchgeo__croma	torchgeo__seco-eco
chrimerss__flood-foundation-prithvi-300m	torchgeo__decur	torchgeo__seco-eco-ndvi
chrimerss__flood-foundation-prithvi-600m	torchgeo__delineate-anything	torchgeo__sentinel1_unet_effb4_openearthmap_sar
chrimerss__flood-foundation-prithvi-tiny	torchgeo__delineate-anything-s	torchgeo__ssl4eo_landsat
chrimerss__flood-foundation-resnet101-unet	torchgeo__dofa	torchgeo__swin_v2_b_naip_rgb_satlas
chrimerss__flood-foundation-resnet152-unet	torchgeo__earthloc	torchgeo__swin_v2_b_sentinel2_rgb_satlas
chrimerss__flood-foundation-resnet50-unet	torchgeo__fields-of-the-world	torchgeo__unet_resnet34_oam_rgb_tcd
galeio-research__OceanSAR-1	torchgeo__ftw	torchgeo__unet_resnet50_oam_rgb_tcd
galeio-research__OceanSAR-1-tengeop	torchgeo__resnet18_sentinel2_all_moco	torchgeo__vit_base_patch32_224_skyclip_50pct
galeio-research__OceanSAR-1-wave	torchgeo__resnet18_sentinel2_rgb_moco	torchgeo__vit_large_patch14_224_clip_laionrs
galeio-research__OceanSAR-1-wind	torchgeo__resnet18_sentinel2_rgb_seco	torchgeo__vit_large_patch14_224_skyclip_30pct
mrm8488__convnext-tiny-finetuned-eurosat	torchgeo__resnet50_fmow_rgb_gassl	torchgeo__vit_large_patch14_224_skyclip_50pct
openclimatefix__power_perceiver	torchgeo__resnet50_landsat7_l2_all_moco	torchgeo__vit_large_patch16_224_fmow_rgb_scalemae
pszemraj__convnextv2-nano-22k-384-boulderspot	torchgeo__resnet50_sentinel1_all_moco	torchgeo__vit_small_patch16_224_sentinel2_all_dino
quantum-leap-vcti__VCTI-RoBERTa-Fiber	torchgeo__resnet50_sentinel2_all_dino	torchgeo__vit_small_patch16_224_sentinel2_all_moco
swardiantara__drone-term-extractor	torchgeo__resnet50_sentinel2_all_moco	torchgeo__yolo11s_marine_vessel_detection
torchgeo__ai4g_flood	torchgeo__resnet50_sentinel2_rgb_moco	Burdenthrive__cloud-detection-segformer-mit_b4-RGB
torchgeo__copernicus-fm	torchgeo__resnet50_sentinel2_rgb_seco	Burdenthrive__cloud-detection-unet-regnetzd8
ingmarnitze__thaw-slump-segmentation	isaaccorley__unet_resnet50_oam_rgb_tcd	Kaludi__CSGO-Minimap-Layout-Generation
gsumbul__SMARTIES-v1-finetuned-models	truthdotphd__cloud-detection	DarthReca__actu-magnitude-regression
DimitrisMantas__RoofSense		

Table 10. Remote Sensing Model checkpoints used to train GeoSANE.

Modality	Number of Models
Multispectral	41
RGB	15
Multimodal	13
SAR	6
DEM	2
Unknown	26

Table 11. Distribution of modalities for the models used to train GeoSANE.

Task	Number of Models
Self-supervised representation learning	36
Semantic segmentation	19
Object detection	7
Classification	4
Generative models	3
Regression	2
Unknown	32

Table 12. Distribution of primary tasks for the models used to train GeoSANE.