RANGE: Retrieval Augmented Neural Fields for Multi-Resolution Geo-Embeddings

Supplementary Material

8. Implementation Details

In this section, we describe the specifics of our experiments.

Quantitative evaluation on downstream tasks: For Section 5.1, we conducted a linear probe for each model using a ridge classifier. We swept over different regularization weights and selected the optimal one using cross-validation on the training set. For our RANGE model, we used $\tau = 1/15$ for all tasks. This value was selected by using the cross-validation scores on only Biome and Temperature data. For RANGE$^+$, we used $\tau_1 = 1/12$, and $\tau_2 = 1/40$. These were selected using the same procedure as we described for RANGE. For all the experiments with RANGE$^+$, we set $\beta = 0.5$, giving equal weight to the semantic and spatial similarity of visual features.

Evaluation on iNaturalist data: For Section 5.2, we conducted a linear probe for each model using the training split of iNaturalist data. However, following prior work [33], we used the "assume negative" loss function, proposed by Cole *et al* [5]. For RANGE and RANGE$^+$, we use the same hyperparameters as we used for our experiments in Section 5.1. We used the pre-trained "full high-resolution" model by Mac Aodha *et al.* [25] to get the image-only predictions $P(y \mid I)$ for the iNaturalist test set.

Ablation of database size: For the ablation on database sizes, we used a stratified sampling strategy to create smaller databases with 75%, 50%, 25%, and 10% of the original data. The original data contained around 82,000 locations uniformly distributed across the landmass. The 82k locations are a subset of the SatCLIP [18] dataset after removing corrupted downloads. We use fixed hyperparameters for the models across all tasks while varying the database size.

9. Quantitative Evaluation of β parameter

In this section, we show how the β parameter can be tuned to solve geospatial tasks at different resolutions. To show this, we use the checkerboard experiment, which was used by Rußwurm *et al.* [33]. We choose k points in the sphere using Fibonacci-lattice; the surface area represented by each point is almost identical [12, 33]. Each of these points is assigned one out of 16 categories in a regular order. For the train and test set, we sample 10,000 points on the sphere and assign each point the label of the closest labeled point. The task is to learn a linear model to classify each point (we use the same strategy described in Section 8).

	β	100 19.21°	500 8.72°	1000 6.11°	1500 5.06°	2000 4.34°
SatCLIP		36.0	26.4	25.5	25.5	21.7
GeoCLIP		64.1	22.8	15.7	14.4	13.8
CSP-INat		44.9	27.1	24.0	21.9	18.5
CSP		67.8	33.8	26.8	23.9	21.1
SINR		87.6	58.4	34.3	22.6	19.0
RANGE$^+$	0	**94.0**	**73.2**	50.0	43.1	37.8
	0.25	<u>93.3</u>	<u>72.6</u>	<u>55.8</u>	50.7	45.3
	0.5	92.3	70.0	**56.4**	**52.3**	**47.2**
	0.75	89.4	65.1	54.5	<u>50.6</u>	<u>46.3</u>
	1	65.7	53.6	50.5	46.6	42.6

Table 5. We quantitatively show that controlling the beta parameter allows us to generate optimal embeddings depending on the resolution of the task. We evaluate on the checkerboard task [33] and change the number of grid cells in the Fibonacci lattice to simulate tasks with different spatial resolutions. Lower β yields better embeddings for low-resolution tasks, whereas higher β yields better embeddings for high-resolution tasks. We see that we outperform all the baselines across all spatial resolutions.

Changing k allows us to change the spatial scale of the task. Higher k creates more grid cells and, therefore, requires higher resolvable resolution. We use different β values to solve the checkerboard task with different k's. The results are shown in Table 5. The columns in the table shows the different values of k and the average distance between the checkerboard centers in degrees. We see that as we increase the resolution of the task, increasing the value of β (reducing spatial smoothness) achieves better performance. Similarly, lower β (adding spatial smoothness) performs better for low-resolution tasks.

We outperform the existing baselines across all resolutions. Within the baselines, SINR performs the best at lower resolutions, whereas SatCLIP performs better at higher resolutions. The quantitative results validate the qualitative results from Section 5.4. RANGE$^+$ outperforms all the baselines across all spatial resolutions. At $\beta = 0.5$, we get the most stable performance across different resolutions. We also see that the gap between the state-of-the-art baseline and RANGE$^+$ increases more dramatically for higher resolutions.

Models	temp_mean	temp_min	temp_max	dew_temp	precipitation	pressure	u_wind	v_wind	Avg
CSP	0.944	0.933	0.940	0.918	0.610	0.427	0.499	0.550	0.727
CSP-INat	0.987	0.897	0.886	0.857	0.534	0.307	0.413	0.386	0.658
SINR	0.982	0.975	0.976	0.977	0.758	0.706	0.726	0.694	0.849
GeoCLIP	0.960	0.953	0.948	0.954	0.591	0.651	0.502	0.529	0.761
SatCLIP	0.904	0.900	0.887	0.894	0.497	0.743	0.488	0.455	0.721
RANGE	0.975	0.972	0.966	0.972	0.759	0.888	0.741	0.717	0.873
RANGE$^+$	**0.990**	**0.985**	**0.984**	**0.988**	**0.815**	**0.896**	**0.742**	**0.772**	**0.896**

Table 6. We show the linear probe results on real-world climate data from ERA5. We predict 8 different climate variables using different location encoders. The results show that RANGE and RANGE$^+$ achieve the two highest average R^2 across all variables, with RANGE$^+$ consistently achieving the best performance for each task.

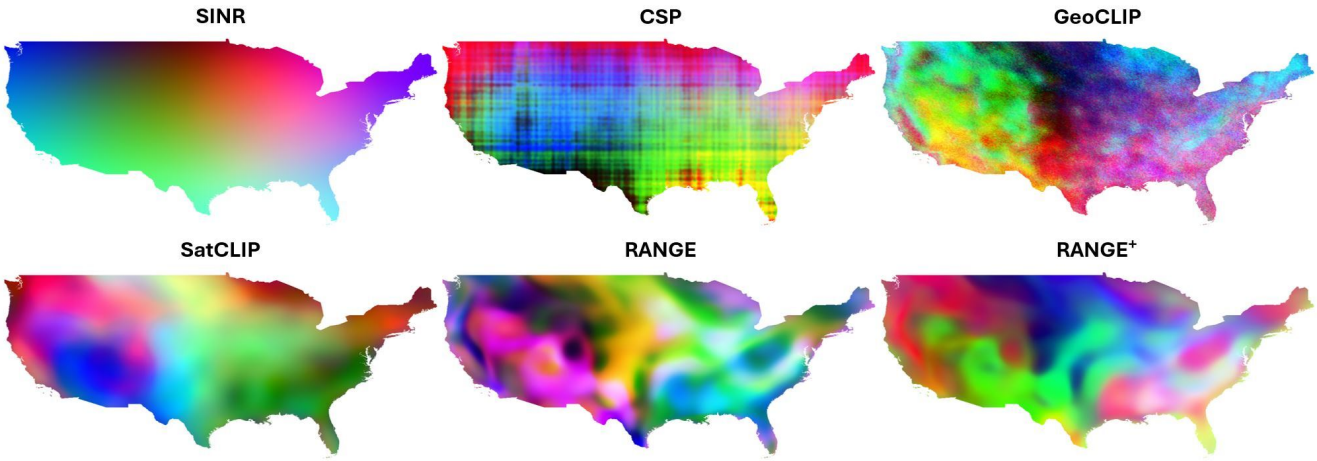

Figure 6. We visualize the geo-embeddings from different models on a country scale (USA) by projecting them into a 3-dimensional vector using Independent Component Analysis (ICA).

10. Evaluation on ERA5 data

We also evaluate our models on climate data from ERA5. We use 8 climate variables, namely, mean air temperature, maximum air temperature, minimum air temperature, dew-point temperature, precipitation, surface pressure, u component of the wind and v component of the wind. We fit a linear model to predict each of these variables using the location embeddings from different location encoders. We use the same hyperparameters for RANGE and RANGE$^+$ that are described in Section 8. Table 6 shows the R^2 values for each task from each model. We show that RANGE and RANGE$^+$ achieve the two highest average R^2 across all tasks. Furthermore, RANGE$^+$ also achieves the highest R^2 value for each task separately.

11. Visualizing Geo-Embeddings at Country Scale

In Section 5.4, we visualized the location embeddings on a global scale. Here, we visualize the location embeddings on a country scale. We densely sample points across the United States and use them to compute the location embeddings. We use Independent Component Analysis to project each embedding to a 3-dimensional vector and use it to represent the RGB channels. For different models, the same colors do not necessarily indicate similar information. We can see the visualizations in Figure 6. Visually, it appears that the RANGE embeddings can capture local variations relatively well.

12. Geoprior Evaluation with Training-free Baselines

In Section 5.2, we evaluated different training-based location encoding methods on geoprior task using iNaturalist data. Here, we show the results of using training-free location encoding methods. Table 7 shows that RANGE models outperform the training-free baselines.

13. Computational Cost

The retrieval process incurs some added computational cost. However, our setup makes this process highly ef-

	top-1	top-3	top-5	top-10
Direct	63.5	81.7	87.0	91.8
Cartesian	65.0	82.7	87.7	92.3
Wrap	65.4	83.1	87.9	92.5
SphereC+	69.5	85.5	89.9	93.5
SphereM+	70.8	86.3	90.5	93.9
RANGE	**75.2**	**89.6**	**92.9**	**95.5**
RANGE+	75.1	89.5	92.8	**95.5**

Table 7. **Top-k classification accuracy on INat-2018 test split:** Comparing our model with training free baselines.

ficient. Generating the feature bank is a one-time operation, which is inexpensive for a few thousand images (Section 5.3). Second, the retrieval process is completely vectorized and highly efficient. For reference, when using the 10k database, computing the RANGE embeddings for 1 million input locations takes less than 65 seconds on our CPU and less than 10 seconds on our H100 GPU, making our method efficient for any practical usage.

14. Limitations and Future Work

In our work, we argued the limitations of learning geo-embeddings by contrastively aligning location and images from the perspective of multi-view redundancy. While the aforementioned problems exist for any location-image alignment, we propose a solution for improving location and ***satellite-image*** alignment. In this paper, we exploit specific properties of satellite data to circumvent the existing issues. In the future, we would like to extend this work to all location-image alignment settings.