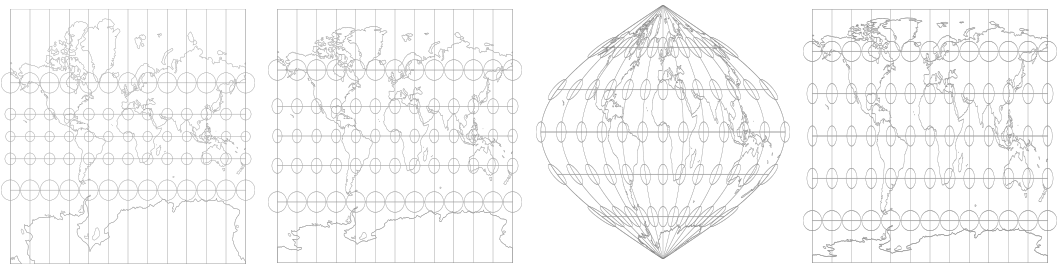


(a) Mercator (b) Miller (c) Sinusoidal (d) Equirectangular

Figure 6: An illustration for map projection distortion: (a)-(d): Tissot indicatrices for four projections. The equal area circles are putted in different locations to show how the map distortion affect its shape.

9 APPENDIX

9.1 THE MAP PROJECTION DISTORTION PROBLEM

In fact there is no map projection, which can preserve distances in all directions. The so-called equidistant projection can only preserve distance on one direction, e.g., the longitude direction for the equirectangular projection (See Figure 6d), while the conformal map projections (See Figure 6a) can preserve directions while resulting in a large distance distortion. For a comprehensive overview of map projections and their distortions, see Mulcahy & Clarke (2001). This is a well recognized problem in Cartography which shows the importance of *calculating on a round planet* Chrisman (2017). Due to the limitations of these 2D location encoders, there is an urgent need for *a location encoding method which preserves the spherical distance (e.g., great circle distance*[5]*) between two points*. The multi-scale encoding method utilizes Double Fourier Sphere basis ($O(S^2)$ terms) or a subset ($O(S)$ terms) of it while still being able to correctly measure the spherical distance. This inspires us to explore the most effective subset of Fourier bases on spheres.

9.2 SPHEREC,SPHERECP,SPHEREM,SPHEREMP

sphereC Inspired by the fact that any point (x, y, z) in 3D Cartesian coordinate can be expressed by sin and cos basis of spherical coordinates (λ, ϕ plus radius) [6], we define the basic form of *Sphere2Vec*, namely $sphereC$ encoder, for scale s as

$$G^{(sphereC)}(\mathbf{x}) = \bigcup_{s=0}^{S-1} [\sin \phi_s, \cos \phi_s \cos \lambda_s, \cos \phi_s \sin \lambda_s]. \tag{9}$$

It can be shown that when $S = 1$, $G^{(sphereC)}(\mathbf{x})$ directly satisfies our expectation in Equation 1 where $f(x) = \cos(\frac{x}{R})$. See more detailed analysis and comparison to the $grid$ encoder in Appendix 9.3.

sphereM Considering the fact that many geographical features are more sensitive to either latitude (e.g., temperature, sunshine duration) or longitude (e.g., timezones, geopolitical borderlines), we might want to focus on increasing the resolution of either ϕ or λ while the other is hold relatively at large scale. Therefore, we introduce a multi-scale position encoder $sphereM$, where interaction terms between ϕ and λ always have one of them fixed at top scale:

$$G^{(sphereM)}(\mathbf{x}) = \bigcup_{s=0}^{S-1} [\sin \phi_s, \cos \phi_s \cos \lambda, \cos \phi \cos \lambda_s, \cos \phi_s \sin \lambda, \cos \phi \sin \lambda_s]. \tag{10}$$

This new encoder ensures that the ϕ term interact with all the scales of λ terms and λ term interact with all the scales of ϕ terms. Note that when $S = 1$, $G^{(sphereM)}$ is equivalent to $G^{(sphereC)}$. Both $sphereC$ and $sphereM$ are multi-scale versions of a spherical distance-kept encoder (See Equation 12) and keep that as the main term in their multi-scale representation.

[5] https://en.wikipedia.org/wiki/Great-circle_distance

[6] https://en.wikipedia.org/wiki/Spherical_coordinate_system

sphereC+ and sphereM+ From the analysis of the two proposed encoders and the state-of-the-art $grid$ encoders (Appendix 9.3), we know that $grid$ pays more attention to the sum of cos difference of latitudes and longitudes, while our proposed encoders pay more attention to the spherical distances. In order to capture both information, we consider merging $grid$ with each proposed encoders to get more powerful models that encode geographical information from different angles.

$$\begin{aligned} G^{(sphereC+)}(\mathbf{x}) &= G^{(sphereC)}(\mathbf{x}) \cup G^{(grid)}(\mathbf{x}), \\ G^{(sphereM+)}(\mathbf{x}) &= G^{(sphereM)}(\mathbf{x}) \cup G^{(grid)}(\mathbf{x}). \end{aligned} \tag{11}$$

9.3 Analysis of $sphereC$ and its comparison to the $grid$ encoder

To illustrate that $sphereC$ is good at capturing spherical distance, we take a close look at its basic case $S = 1$ (define $s = 0$ and $f_s = 1$), where the multi-scale encoder degenerates to

$$\widehat{G}^{(sphereC)}(\mathbf{x}) = [\sin(\phi), \cos(\phi)\cos(\lambda), \cos(\phi)\sin(\lambda)]. \tag{12}$$

These three terms are included in the multi-scale version ($S > 1$) and serve as the main terms at the largest scale and also the lowest frequency (when $s = S - 1$). The high frequency terms are added to help the downstream neuron network to learn the point-feature more efficiently. Interestingly, $\widehat{G}^{(sphereC)}$ captures the spherical distance in a very explicit way:

Theorem 2. *Let $\mathbf{x}_1$, $\mathbf{x}_2$ be two points on the same sphere with radius R, then*

$$\langle \widehat{G}^{(sphereC)}(\mathbf{x}_1), \widehat{G}^{(sphereC)}(\mathbf{x}_2) \rangle = \cos(\frac{\Delta D}{R}), \tag{13}$$

where ΔD is the great circle distance between $\mathbf{x}_1$ and $\mathbf{x}_2$. Under this metric,

$$\|\widehat{G}^{(sphereC)}(\mathbf{x}_1) - \widehat{G}^{(sphereC)}(\mathbf{x}_2)\| = 2\sin(\frac{\Delta D}{2R}). \tag{14}$$

Moreover, $\|\widehat{G}^{(sphereC)}(\mathbf{x}_1) - \widehat{G}^{(sphereC)}(\mathbf{x}_2)\| \approx \frac{\Delta D}{R}$, when ΔD is small w.r.t. R.

See the proof in Appendix 9.4. Since the central angle $\Delta\delta = \frac{\Delta D}{R} \in [0, \pi]$ and $\cos(x)$ is strictly monotonically decrease for $x \in [0, \pi]$, Theorem 2 shows that $\widehat{G}^{(sphereC)}(\mathbf{x})$ directly satisfies our expectation in Equation 1 where $f(x) = \cos(\frac{x}{R})$. In comparison, when $S = 1$, the inner product in the output space of $grid$ encoder is

$$\langle \widehat{G}^{(grid)}(\mathbf{x}_1), \widehat{G}^{(grid)}(\mathbf{x}_2) \rangle = \cos(\phi_1 - \phi_2) + \cos(\lambda_1 - \lambda_2), \tag{15}$$

which models the latitude difference and longitude difference of $\mathbf{x}_1$ and $\mathbf{x}_2$ separately rather than spherical distance. This introduces problems in encoding. For instance, consider data pairs $\mathbf{x}_1 = (\lambda_1, \phi)$ and $\mathbf{x}_2 = (\lambda_2, \phi)$, the distance between them in output space of $grid$, $\|\widehat{G}^{(grid)}(\mathbf{x}_1) - \widehat{G}^{(grid)}(\mathbf{x}_2)\|^2 = 2 - 2\cos(\lambda_1 - \lambda_2)$ stays as a constant in terms of ϕ. However, when ϕ varies from $-\frac{\pi}{2}$ to $\frac{\pi}{2}$, the actual spherical distance changes in a wide range, e.g., the actual distance between the data pair at $\phi = -\frac{\pi}{2}$ (South Pole) is 0 while the distance between the data pair at $\phi = 0$ (Equator), gets the maximum value. This issue in measuring distances also has a negative impact on $grid$'s ability to model distributions in areas with sparse sample points because it is hard to learn the true spherical distance. We observe that $grid$ reaches peak performance at much smaller r_{min} than that of *Sphere2Vec* encodings (See Appendix 9.7). Moreover, $sphereC$ outperforms $grid$ near polar regions where $grid$ claims large distance though the spherical distance is small (A, B in Figure 1).

9.4 Proof of Theorem 2

Proof. Since $\widehat{G}^{(sphereC)}(\mathbf{x}_i) = [\sin(\phi_i), \cos(\phi_i)\cos(\lambda_i), \cos(\phi_i)\sin(\lambda_i)]$ for $i = 1, 2$, the inner product

$$\begin{aligned} &\langle \widehat{G}^{(sphereC)}(\mathbf{x}_1), \widehat{G}^{(sphereC)}(\mathbf{x}_2) \rangle \\ &= \sin(\phi_1)\sin(\phi_2) + \cos(\phi_1)\cos(\lambda_1)\cos(\phi_2)\cos(\lambda_2) + \cos(\phi_1)\sin(\lambda_1)\cos(\phi_2)\sin(\lambda_2) \\ &= \sin(\phi_1)\sin(\phi_2) + \cos(\phi_1)\cos(\phi_2)\cos(\lambda_1 - \lambda_2) \\ &= \cos(\Delta\delta) = \cos(\Delta D/R), \end{aligned} \tag{16}$$

where $\Delta\delta$ is the central angle between $\mathbf{x}_1$and $\mathbf{x}_2$, and the spherical law of cosines is applied to derive the second last equality. So,

$$\begin{aligned}&\|\widehat{G}(\mathbf{x}_1) - \widehat{G}(\mathbf{x}_2)\|^2 \\ &= \langle \widehat{G}(\mathbf{x}_1) - \widehat{G}(\mathbf{x}_2), \widehat{G}(\mathbf{x}_1) - \widehat{G}(\mathbf{x}_2) \rangle \\ &= 2 - 2\cos(\Delta D/R) \\ &= 4\sin^2(\Delta D/2R).\end{aligned} \tag{17}$$

So $\|\widehat{G}(\mathbf{x}_1) - \widehat{G}(\mathbf{x}_2)\| = 2\sin(\Delta D/2R)$ since $\Delta D/2R \in [0, \frac{\pi}{2}]$. By Taylor expansion, $\|\widehat{G}(\mathbf{x}_1) - \widehat{G}(\mathbf{x}_2)\| \approx \Delta D/R$ when ΔD is small w.r.t. R. □

9.5 Proof of Theorem 1

$\forall * \in \{sphereC, sphereC+, sphereM, sphereM+\}$, $G^{(*)}(\mathbf{x}_1) = G^{(*)}(\mathbf{x}_2)$ implies

$$\sin(\phi_1) = \sin(\phi_2), \tag{18}$$

$$\cos(\phi_1)\sin(\lambda_1) = \cos(\phi_2)\sin(\lambda_2), \tag{19}$$

$$\cos(\phi_1)\cos(\lambda_1) = \cos(\phi_2)\cos(\lambda_2), \tag{20}$$

from $s = 0$ terms. Equation 18 implies $\phi_1 = \phi_2$. If $\phi_1 = \phi_2 = \pi/2$, then both points are at North Pole, $\lambda_1 = \lambda_2$ equal to whatever longitude defined at North Pole. If $\phi_1 = \phi_2 = -\pi/2$, it is similar case at South Pole. When $\phi_1 = \phi_2 \in (-\frac{\pi}{2}, \frac{\pi}{2})$, $\cos(\phi_1) = \cos(\phi_2) \neq 0$. Then from Equation 19 and 20,

$$\sin\lambda_1 = \sin(\lambda_2), \cos(\lambda_1) = \cos(\lambda_2), \tag{21}$$

which shows that $\lambda_1 = \lambda_2$. In summary, $\mathbf{x}_1 = \mathbf{x}_2$, so $G^{(*)}$ is injective.
If $* = sphereDFS$, $G^{(*)}(\mathbf{x}_1) = G^{(*)}(\mathbf{x}_2)$ implies

$$\sin(\phi_1) = \sin(\phi_2), \cos(\phi_1) = \cos(\phi_2), \sin(\lambda_1) = \sin(\lambda_2), \cos(\lambda_1) = \cos(\lambda_2), \tag{22}$$

which proves $\mathbf{x}_1 = \mathbf{x}_2$ and $G^{(*)}$ is injective directly.

9.6 Baselines

In order to understand the advantage of sphereical-distance-kept location encoders, we compare different versions of *Sphere2Vec* with multiple baselines:

- $No\ Prior$ indicates a image classifier without using any location information, i.e., predicting image labels purely based on image information $P(y|\mathbf{I})$.
- $tile$ divides the study area A (e.g., the earth surface) into grids with equal intervals along the latitude and longitude direction. Each grid has an embedding to be used as the encoding for every location $\mathbf{x}$ fall into this grid. This is a common practice by many previous work when dealing with coordinate data (Berg et al., 2014; Adams et al., 2015; Tang et al., 2015).
- $wrap$ is a location encoder model introduced by Mac Aodha et al. (2019). Given a location $\mathbf{x} = (\lambda, \phi)$, it uses a coordinate wrap mechanism to convert each dimension of $\mathbf{x}$ into 2 numbers - $G^{(wrap)}(\mathbf{x}) = [\sin(\pi\frac{\lambda}{180^\circ}), \cos(\pi\frac{\lambda}{180^\circ}), \sin(\pi\frac{\phi}{90^\circ}), \cos(\pi\frac{\phi}{90^\circ})]$. Then the results are passed through a multi-layered fully connected neural network $\mathbf{NN}^{(wrap)}()$ which consists of an initial fully connected layer, followed by a series of h residual blocks, each consisting of two fully connected layers (k hidden neurons) with a dropout layer in between. We adopt the official code of Mac Aodha et al. (2019)[7] for this implementation.

[7] http://www.vision.caltech.edu/~macaodha/projects/geopriors/

- $wrap + ffn$ is similar to $wrap$ except that it replaces $\mathbf{NN}^{(wrap)}()$ with $\mathbf{NN}_{ffn}()$, a simple learnable multi-layer perceptron with h hidden layers and k neurons per layer as that *Sphere2Vec* has. $wrap + ffn$ is used to exclude the effect of different $\mathbf{NN}()$ on the performance of location encoders. In the following, all location encoder baselines use $\mathbf{NN}_{ffn}()$ as the learnable neural network component so that we can directly compare the effect of different position encoding $G(\cdot)$ on the model performance.
- rbf randomly samples M points from the training dataset as RBF anchor points $\{\mathbf{x}_m^{anchor}, m = 1...M\}$, and use gaussian kernels $\exp\big(-\frac{\| \mathbf{x}_i - \mathbf{x}_m^{anchor} \|^2}{2\sigma^2}\big)$ on each anchor points, where σ is the kernel size. Each point p_i has a M-dimension RBF feature vector which is fed into $\mathbf{NN}_{ffn}()$ to obtain the location embedding. This is a strong baseline for representing floating number features in machine learning models.
- rff, i.e., *Random Fourier Features* (Rahimi & Recht, 2008; Nguyen et al., 2017), first encodes location $\mathbf{x}$ into a D dimension vector - $G^{(rff)}(\mathbf{x}) = \varphi(\mathbf{x}) = \frac{\sqrt{2}}{\sqrt{D}}[\cos(\omega_i^T\mathbf{x} + b_i)]_{i=1}^D$ where $\omega_i \overset{i.i.d}{\sim} \mathcal{N}(\mathbf{0}, \delta^2 I)$ and b_i is uniformly sampled from $[0, 2\pi]$. I is an identity matrix. Each component of $\varphi(\mathbf{x})$ first projects $\mathbf{x}$ into a random direction ω_i and makes a shift by b_i. Then it wraps this line onto the unit cirle in $\mathbb{R}^2$ with the cosine function. Rahimi & Recht (2008) show that $\varphi(\mathbf{x})^T\varphi(\mathbf{x}')$ is an unbiased estimate of the Gaussian kernal $K(\mathbf{x}, \mathbf{x}')$. $\varphi(\mathbf{x})$ is consist of D different estimates to produce a further lower variance approximation. To make rff comparable to other baselines, we feed $\varphi(\mathbf{x})$ into $\mathbf{NN}_{ffn}()$ to produce the final location embedding.
- $grid$ is a multi-scale location encoder on 2D Euclidean space proposed by Mai et al. (2020b). Here, we simply treat $\mathbf{x} = (\lambda, \phi)$ as 2D coordinate. It first use $G^{(grid)}(\mathbf{x})$ shown in Equation 3 to encode location $\mathbf{x}$ into multi-scale representation and then feed it into $\mathbf{NN}_{ffn}()$ to produce the final location embedding.
- $theory$ is another multi-scale location encoder on 2D Euclidean space proposed by Mai et al. (2020b). It use a position encoder $G^{(theory)}(\mathbf{x})$ shown in Equation 23.. Here, $\mathbf{x}^s = [\lambda_s, \phi_s] = [\frac{\lambda}{f_s}, \frac{\phi}{f_s}]$ and $\mathbf{a}_1 = [1, 0]^T, \mathbf{a}_2 = [-1/2, \sqrt{3}/2]^T, \mathbf{a}_3 = [-1/2, -\sqrt{3}/2]^T \in \mathbb{R}^2$ are three unit vectors which orient $2\pi/3$ apart from each other. The encoding results are feeinto $\mathbf{NN}_{ffn}()$ to produce the final location embedding.

$$G^{(theory)}(\mathbf{x}) = \bigcup_{s=0}^{S-1} \bigcup_{j=1}^{3} [\sin(\langle \mathbf{x}^s, \mathbf{a}_j \rangle), \cos(\langle \mathbf{x}^s, \mathbf{a}_j \rangle)]. \tag{23}$$

9.7 *Sphere2Vec* HYPERPARAMETERS

Table 2 shows the best hyperparameter combinations of different *Sphere2Vec* models on different image classification dataset. We use a smaller S for $sphereDFS$ since it has $O(S^2)$ terms while the other models have $O(S)$ terms. $sphereDFS$ with $S = 8$ yield a similar number of terms to the other models wth $S = 32$ (See Table 3). Interestingly, all first four *Sphere2Vec* models ($sphereC$, $sphereC+$, $sphereM$, and $sphereM+$) shows the best performance on all five datasets with the same hyperparamter combinations. Note that compared with other datasets, iNat2017 and iNat2018 are more up-to-date datasets with more training samples and better geographic coverage. This indicates that the proposed 4 *Sphere2Vec* models show similar performance over different hyperparameter combinations.

9.8 IMPACT OF MRR BY THE NUMBER OF SAMPLES AT DIFFERENT LATITUDE BANDS

See Figure 7

9.9 PREDICTED DISTRIBUTIONS INAT2018

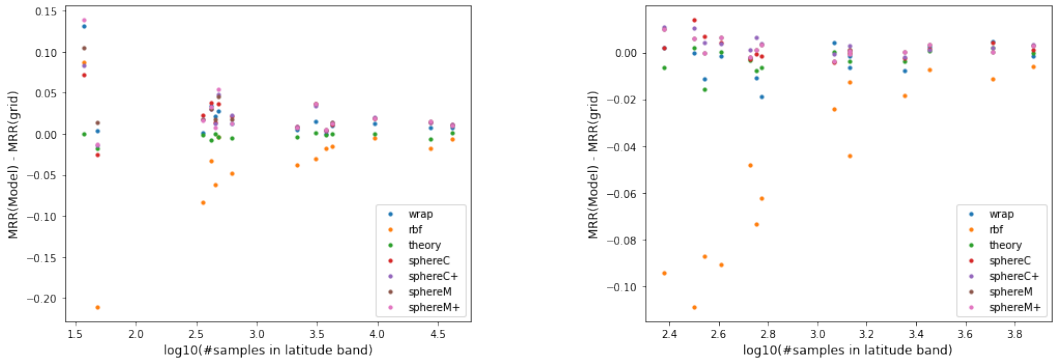
We plot the predicted species distributions from different models at different geographic regions, and compare them with the training sample locations of the corresponding species, see Figure 8. We can see that compared with $wrap*$ and $grid$, in each geographic region with sparse training samples and

Table 2: The best hyperparameter combinations of *Sphere2Vec* models on different image classification datasets. The learning rate α tends to be smaller for larger datasets; We fix the total number of frequencies S to be 8 for $sphereDFS$ and 32 for all others; the maximum scale $r_{max} = 1$; r_{min}: the minimum scale; the number of hidden layers $\mathbf{NN}()$ is fixed to $h = 1$; the number of neurons in $\mathbf{NN}()$ is fixed to $k = 1024$ except for the smallest dataset.

Dataset	α	r_{min}	k
BirdSnap	0.001	10^{-6}	512
BirdSnap†	0.001	10^{-4}	1024
NABirds†	0.001	10^{-4}	1024
iNat2017	0.0001	10^{-2}	1024
iNat2018	0.0005	10^{-3}	1024

Table 3: Dimension of position encoding for different models in terms of total scales S

Model	$sphereC$	$sphereC+$	$sphereM$	$sphereM+$	$sphereDFS$
Dimension	$3S$	$6S$	$5S$	$8S$	$4S^2 + 4S$



(a) iNat2017 (b) iNat2018

Figure 7: Impact of MRR by The Number of Samples at Different Latitude Bands.

the North Pole area, the spatial distributions produced by $sphereC+$ are more compact while the other two have over-generalization issue.

9.10 Embedding Clustering

We use the location encoder trained on iNat2017 or iNat2018 dataset to produce a location embedding for the center of each small latitude-longitude cell. Then we do agglomerative clustering[8] on all these embeddings to produce a clustering map. Figure 9 and 10 show the clustering results for different models with different hyperparameters on iNat2017 and iNat2018 dataset.

9.11 Ablation Study on Different Self-Supervised Loss

Both the BI (Equation 6) and MC (Equation 7) unsupervised loss have three loss component: in-batch, negative location, SimCSE loss component. How does each of them contribute to the overall unsupervised training? In order to answering this question, we do ablation studies on these two objectives by using iNat2018 dataset as an example. Figure 11a and 11b illustrate the ablation study results.

From Figure 11a, we see that as for BI, adding $\mathcal{L}_{\text{BI}}^{negloc}$ will significantly increase the model performance especially when Γ is large. Adding $\mathcal{L}_{\text{BI}}^{simcse}$ also improves the model performance when Γ is small. As for the ablation study on MC loss as shown in Figure 11b, adding $\mathcal{L}_{\text{MC}}^{negloc}$ can significantly increase the model performance and adding $\mathcal{L}_{\text{MC}}^{simcse}$ can also make the performance slightly better.

[8] https://scikit-learn.org/stable/modules/generated/sklearn.cluster.AgglomerativeClustering.html

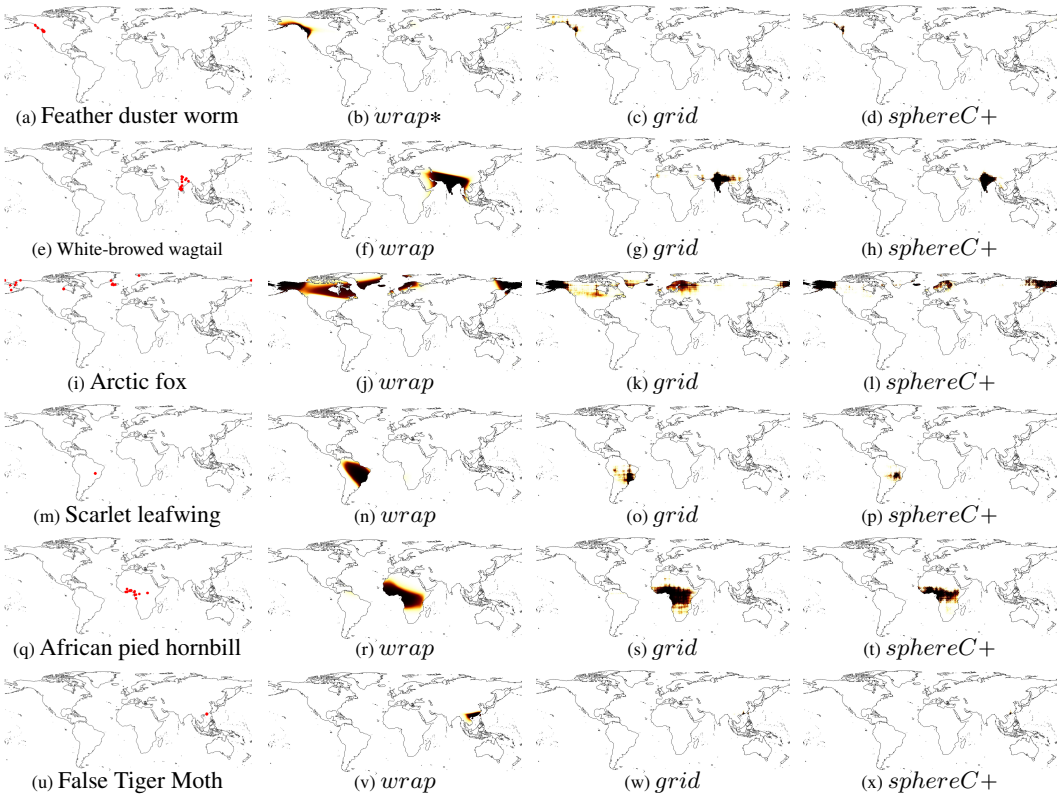
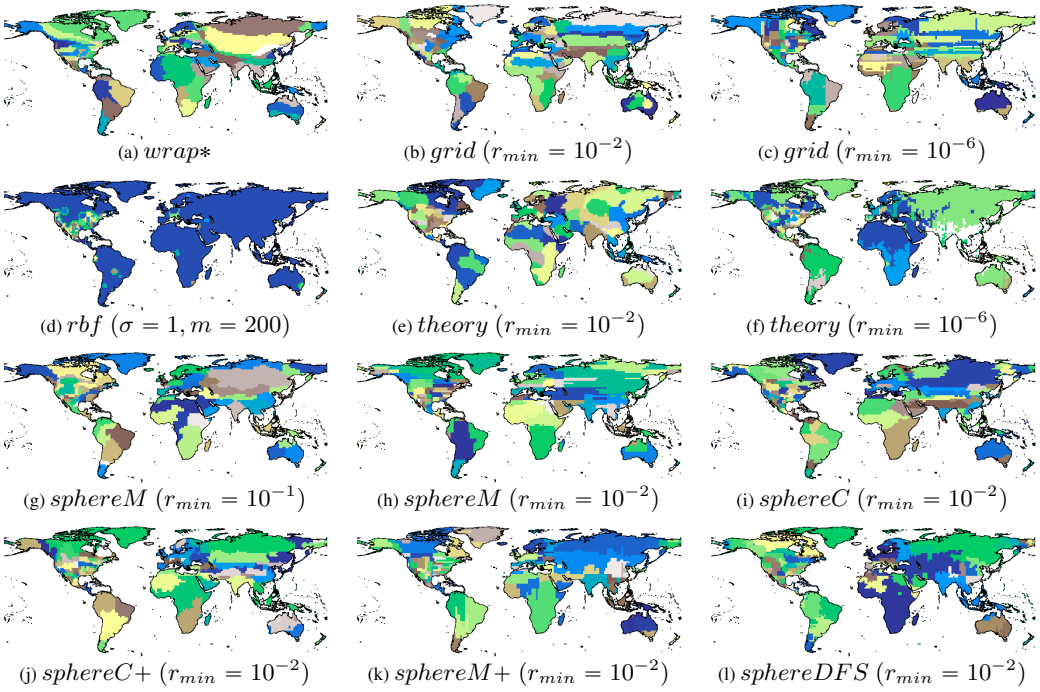


Figure 8: Compare the predicted distributions of example species from different models. The first figure of each row marks the data points from iNat2018 training data.

9.12 Effectiveness of Unsupervised Pretraining on fMoW dataset

.



(a) $wrap*$ (b) $grid$ $(r_{min} = 10^{-2})$ (c) $grid$ $(r_{min} = 10^{-6})$

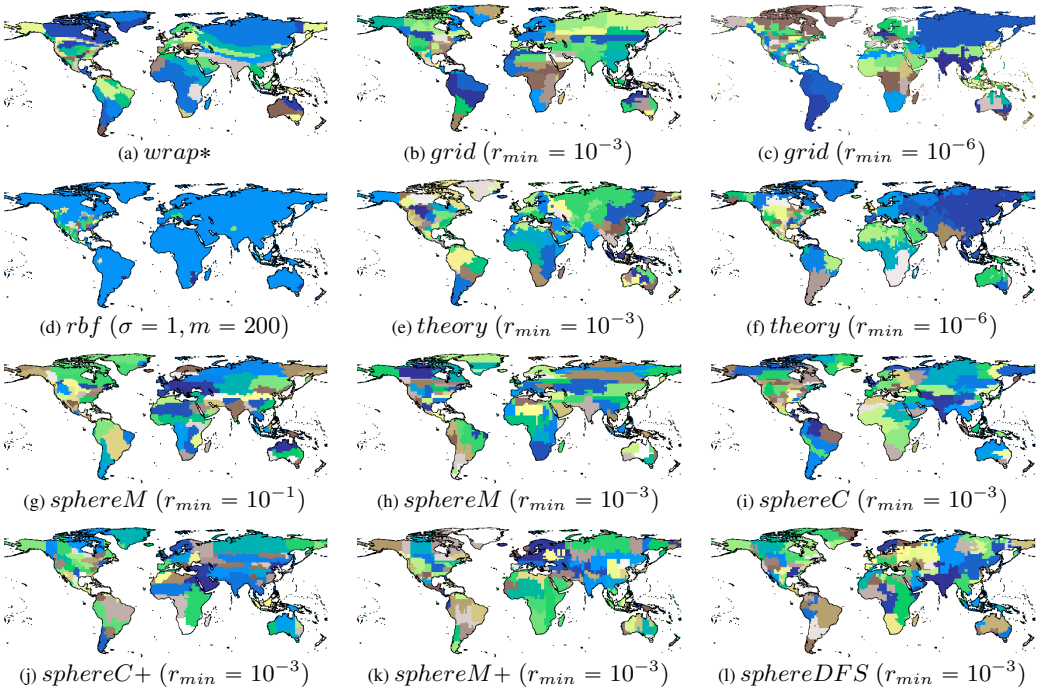
(d) rbf $(\sigma = 1, m = 200)$ (e) $theory$ $(r_{min} = 10^{-2})$ (f) $theory$ $(r_{min} = 10^{-6})$

(g) $sphereM$ $(r_{min} = 10^{-1})$ (h) $sphereM$ $(r_{min} = 10^{-2})$ (i) $sphereC$ $(r_{min} = 10^{-2})$

(j) $sphereC+$ $(r_{min} = 10^{-2})$ (k) $sphereM+$ $(r_{min} = 10^{-2})$ (l) $sphereDFS$ $(r_{min} = 10^{-2})$

Figure 9: Embedding clusterings of iNat2017 models. (a) $wrap*$ with 4 hidden ReLU layers of 256 neurons; (d) rbf with the best kernel size $\sigma = 1$ and number of anchor points $m = 200$; (b)(c)(e)(f) are *Space2Vec* models Mai et al. (2020b) with different min scale $r_{min} = \{10^{-6}, 10^{-2}\}$.[a] (g)-(l) are different *Sphere2Vec* models.[b]

[a] They share the same best hyperparameters: $S = 64$, $r_{max} = 1$, and 1 hidden ReLU layers of 512 neurons.
[b] They share the same best hyperparameters: $S = 32$, $r_{max} = 1$, and 1 hidden ReLU layers of 1024 neurons.



(a) $wrap*$ (b) $grid$ $(r_{min} = 10^{-3})$ (c) $grid$ $(r_{min} = 10^{-6})$

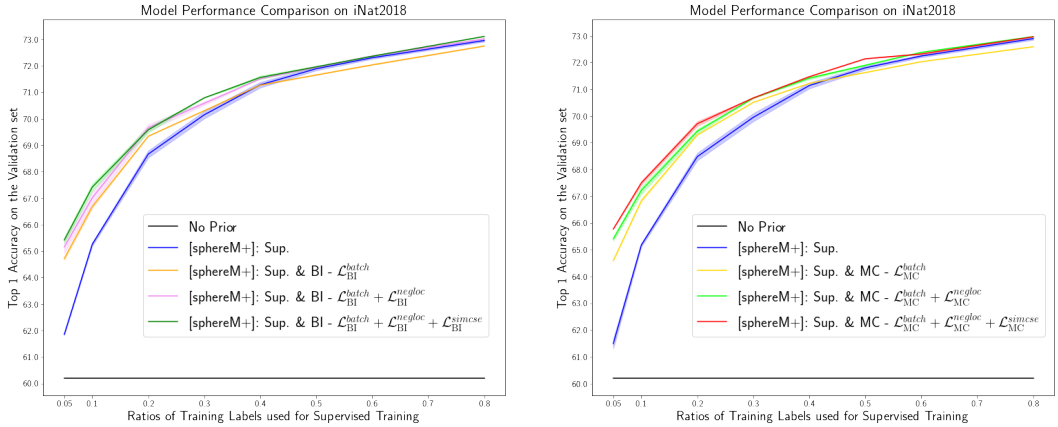
(d) rbf $(\sigma = 1, m = 200)$ (e) $theory$ $(r_{min} = 10^{-3})$ (f) $theory$ $(r_{min} = 10^{-6})$

(g) $sphereM$ $(r_{min} = 10^{-1})$ (h) $sphereM$ $(r_{min} = 10^{-3})$ (i) $sphereC$ $(r_{min} = 10^{-3})$

(j) $sphereC+$ $(r_{min} = 10^{-3})$ (k) $sphereM+$ $(r_{min} = 10^{-3})$ (l) $sphereDFS$ $(r_{min} = 10^{-3})$

Figure 10: Embedding clusterings of iNat2018 models. (a) $wrap*$ with 4 hidden ReLU layers of 256 neurons; (d) rbf with the best kernel size $\sigma = 1$ and number of anchor points $m = 200$; (b)(c)(e)(f) are *Space2Vec* models (Mai et al., 2020b) with different min scale $r_{min} = \{10^{-6}, 10^{-3}\}$.[a] (g)-(l) are *Sphere2Vec* models with different min scale.[b]

[a] They share the same best hyperparameters: $S = 64$, $r_{max} = 1$, and 1 hidden ReLU layers of 512 neurons.
[b] They share the same best hyperparameters: $S = 32$, $r_{max} = 1$, and 1 hidden ReLU layers of 1024 neurons.



(a) Ablation Study on BI

(b) Ablation Study on MC

Figure 11: Ablation study on different unsupervised learning setting on iNat2018 dataset. (a) Alblation study on BI loss. $\mathrm{BI} - \mathcal{L}_{\mathrm{BI}}^{batch} + \mathcal{L}_{\mathrm{BI}}^{negloc} + \mathcal{L}_{\mathrm{BI}}^{simcse}$ indicates the full BI loss while $\mathrm{BI} - \mathcal{L}_{\mathrm{BI}}^{batch} + \mathcal{L}_{\mathrm{BI}}^{negloc}$ deletes the SimCSE component. $\mathrm{BI} - \mathcal{L}_{\mathrm{BI}}^{batch}$ additionally deletes the negative location loss component. Comparing among those three model settings can help us understand the effect of different loss components. Similarly, we do the same ablation study on MC loss and show in (b).

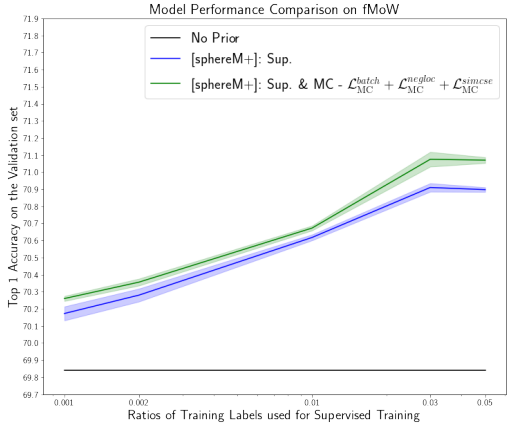


Figure 12: Comparison among two models with identical model architecture but different training objectives on fMoW dataset. Each point on each curve indicates a specific training process. We repeat each of them for five times and show the standard deviations as shaded areas along the line. We can see that with statistically significance, MC unsupervised loss is also effective in the few shot learning setting on the fMoW dataset.

Table 4: Ablation Study on unsupervised loss $\mathcal{L}_{\mathrm{MC}}$ over iNat2018 dataset. We show the effect of different hyperparematers of $\mathcal{L}_{\mathrm{MC}}$ on the performance of location encoders. We use $sphereM+$ as an representative location encoder ad use supervised training dataset ratio Γ as 0.5. lr indicates the learning rate used for unsupervised $\mathcal{L}_{\mathrm{MC}}$ training. $dropout$ indicates the dropout rate used by location encoder which will affect the SimCSE loss as Gao et al. (2021) shows.

Γ	$e(\mathbf{x})$	$\mathcal{L}$	lr	α_1	$\lvert\mathcal{N}^s\rvert$	α_2	τ_0	τ_1	τ_2	dropout	Top1
0.5	$sphereM+$	$\mathcal{L}_{\mathrm{MC}}$	0.001	1	1	1	1	1	1	0.5	71.31
			0.0005								**71.86**
			0.0002								71.78
			0.00005								71.76
			0.00001								71.83
			0.000005								71.7
			0.000001								71.69
			0.0005	2	1	1	1	1	1	0.5	71.8
				1							**71.86**
				0.5							71.71
				0							71.42
			0.0005	1	1	2	1	1	1	0.5	71.71
						1					71.86
						0.5					71.81
						0					71.82
			0.0005	1	1	1	64	1	1	0.5	71.85
							32				71.95
							24				71.74
							22				71.81
							20				**72.08**
							18				71.77
							16				72.02
							12				71.87
							8				71.93
							4				71.69
							2				71.89
							1				71.86
							0.1				71.67
							0.01				71.63
			0.0005	1	1	1	20	2	1	0.5	71.85
								1			**72.08**
								0.5			71.76
			0.0005	1	1	1	20	1	2	0.5	71.84
									1		**72.08**
									0.5		71.69
			0.0005	1	1	1	20	1	1	0.5	**72.08**
					4						71.84
					8						71.69
			0.0005	1	1	1	20	1	1	0.7	71.38
										0.6	71.91
										0.5	**72.08**
										0.4	71.55
										0.3	71.08
										0.2	69.96
										0.1	68.17.

Table 5: Ablation Study on unsupervised loss $\mathcal{L}_{\text{BI}}$ over iNat2018 dataset. We show the effect of different hyperparematers of $\mathcal{L}_{\text{BI}}$ on the performance of location encoders. We use $sphereM+$ as an representative location encoder ad use supervised training dataset ratio Γ as 0.5. lr indicates the learning rate used for unsupervised $\mathcal{L}_{\text{MC}}$ training.

Γ	$e(\mathbf{x})$	$\mathcal{L}$	lr	β_1	$\lvert\mathcal{N}^s\rvert$	β_2	dropout	Top1
0.5	$sphereM+$	$\mathcal{L}_{\text{BI}}$	0.001	1	1	0	0.5	71.76
			0.0008					71.87
			0.0005					71.85
			0.0002					**72.02**
			0.0001					71.67
			0.00001					71.62
			0.000005					71.48
			0.000002					71.56
			0.0002	2	1		0.5	71.98
				1				**72.02**
				0.5				71.85
				0				71.65
			0.0002	1	1		0.5	**72.02**
					4			71.82
					8			71.94
					16			71.84
			0.0002	1	1	0.00	0.5	**72.02**
						0.10		72.06
						0.20		71.93
						0.50		71.83
						1.00		71.84.

Table 6: Ablation Study on unsupervised loss MSE over iNat2018 dataset. We show the effect of different hyperparematers of MSE on the performance of location encoders. We use $sphereM+$ as an representative location encoder ad use supervised training dataset ratio Γ as 0.5. lr indicates the learning rate used for unsupervised MSE training.

Γ	$e(\mathbf{x})$	$\mathcal{L}$	lr	dropout	Top1
0.5	$sphereM+$	MSE	0.001	0.5	71.05
			0.0005		71.41
			0.0001		71.55
			0.00001		71.58
			0.000005		71.69
			0.000002		**71.71**
			0.000001		71.6