Interpreting Satellite Images With ViTs

Jesse Ward-Bond
University of Toronto
jesse.wardbond@mail.utoronto.ca

***Abstract*—Obtaining high-quality labels for remote sensing image analysis tasks is extremely difficult. This has lead to a growing interest in computer vision models that can be trained in an unsupervised or self-supervised fashion. In this work, I explore *how* a recent self-supervised vision transformer learns representations of satellite images. I show that the attention heads of this model attend to interpretable and semantically distinct regions within an image, and I show that the image features learned by this model share meaning across images. I further demonstrate how this can be used to segment images in a completely unsupervised fashion using a graph convolutional network. Code is available on Github.**

I. Introduction

The rise of deep learning technologies for remote sensing image (RSI) analysis has resulted in models that are increasingly hungry for well-labeled datasets [1]. These labels are often either unavailable, incomplete, or extremely hard to generate for RSI analysis: limitations which have naturally lead to a growing interest in *unsupervised* or *self-supervised* (SSL) techniques. However, what models are actually learning through these techniques can be inscrutable. Recent research in the broader field of computer vision (CV) has suggested that vision transformers (ViT) pre-trained using SSL are actually learning to generate semantic maps of input images (Fig. 1) — ones that are similar to how humans would understand an image — but this research has not been extended to the RSI analysis subdomain [2], [3].

In this work, I extend the ViT semantic interpretability work done in [3] to RSI analysis, using a backbone trained on either ground-level photographs or RSIs. In Section III I show that the attention heads of these models attend to interpretable and semantically distinct regions within an image, and I show that the generated deep image features share meaning *across* images. In Section IV I use a recently published graph convolutional network (GCN) unsupervised learning method to show how the deep features of a satellite image can be used for semantic segmentation. In the remainder of this section I present a brief literature review and relevant background information.

A. Remote Sensing

There are many tasks within the the field of RSI analysis which are instances of a broader set of traditional computer vision (CV) tasks. For instance, given a set of aerial or satellite images, researchers might be interested in: classifying images as urban or rural (image classification), dividing it up into roads/not-roads or crops/buildings/water (semantic segmentation), or drawing bounding boxes around automobiles

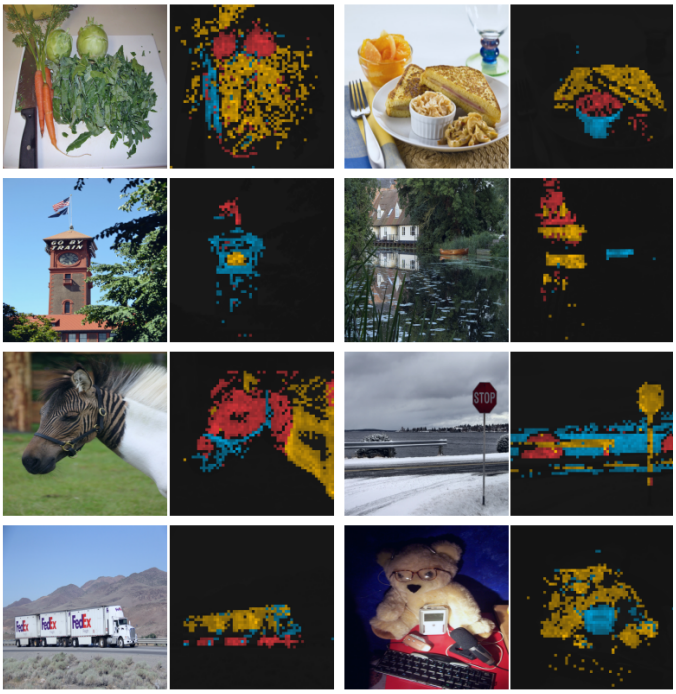

Fig. 1. Semantic maps generate using a ViT trained using in an SSL paradigm using contrastive learning. The different colors represent attention scores from different heads. From [2].

(object detection). Recent developments in deep learning techniques for CV have accordingly greatly advanced RSI analysis. However, the transferrability of these techniques can be limited by some unique characteristics of RSIs:

1) Image characteristics: In traditional photographs (e.g. those in ImageNet [4]) there is usually some notion of a "subject", or of a foreground-background relationship, and important information tends to be visually distinct, focused, and centered within the image. While this is obviously not always the case (e.g. CCTV, vehicle sensors, etc.) there is still an informative aspect of *depth* to ground level photography which the more planar RSIs lack.

2) Object density: It is common for RSIs to contain many objects, scattered from edge-to-edge across the image. For instance, a single image might contain agricultural fields, roads, water bodies, multiple buildings, etc. (Fig. 2). How distinguishable these information-rich regions are from one another obviously depends on their optical characteristics, but also depends on the resolution and technical features (e.g. available bands) of the sensor.

Fig. 2. An example of an RSI taken from the DeepGlobe Land Cover Classification dataset [5].

3) Data diversity: A single satellite can generate multiple terabytes of compressed image data *daily*. Satellites vary in how often they revisit the same location on the earth, what optical bands they are capable of capturing, and their resolution. As a prototypical example: The two satellites from the European Space Agency's *Sentinel* program revisit the same spot on the Earth every 5 days and capture images in 13 different bands ranging from Infrared to "Ultra Blue". The spatial resolution of different bands varies from 10m (visible) to 60m (Ultra Blue) [6].

4) Labelling challenges: The three previous characteristics all contribute to one of the hallmark challenges of working with RSI data: getting appropriately labelled training data. Generating RSI data with labels specific to your task is extremely laborious, and publicly available datasets are typically only available for small fractions of the Earth's surface, potentially leading to biased models. For instance: imagine annotating all the buildings in Fig. 2, which is one scene in a dataset of 1146 RSIs that only covers some areas of Indonesia, India, and Thailand [5]. The complexity of this labelling challenge has spurred great interest in SSL [7].

B. Self Supervised Learning

In the SSL paradigm, models are first trained on large amounts of unlabelled, task-agnostic data before being fine-tuned for specific downstream applications. The training objective (*pretext task*) can have several forms. In *generative* methods, models learn to reconstruct the input data; in *predictive* methods, models learn to predict self-applied labels such as image color, seasonal contrast, or image rotation; in *contrastive* methods, models can learn to output similar predictions for pairs of semantically identical inputs (e.g. original image + flipped image) [7].

The choice of pretext task greatly effects the representations learned by the model, and it is thus important to select the proper pretext task for the desired downstream application [8]. Contrastive methods in particular allow models to learn high-level representations of input images, enabling better generalization in downstream tasks [7].

In [2], Caron *et al.* introduced a contrastive learning approach using self-**di**stillation with **no** labels (DINO). DINO is composed of a *teacher* network and a *student* network with identical architectures parameterized by θ_t and θ_s respectively. DINO is trained as follows: for every input image, a set of views $\mathcal{V}$ is generated, which contains distortions of the original image (crops, Gaussian blur, color jittering, and solarization). $\mathcal{V}$ contains random *local* crops which cover small areas ($< 50\%$) of the original input, and two *global* crops, x_1^g and x_2^g, which cover larger areas ($\geq 50\%$) of the input image. All crops in $\mathcal{V}$ are passed to the student network, and the teacher network only receives the global crops. The training task is

$$\min_{\theta_s} \sum_{x \in \{x_1^g, x_2^g\}} \sum_{\substack{x' \in \mathcal{V} \\ x' \neq x}} H(P_t(x), P_x(x')), \tag{1}$$

where $H(a, b) = -a \log b$. Note that only θ_s is being learned; θ_t is updated using an exponential moving average of student weights [2] (Fig. 3. By using (1) as the training objective, the student network is learning to predict the correspondence (or lack thereof) of all crops in $\mathcal{V}$ to the global crops.

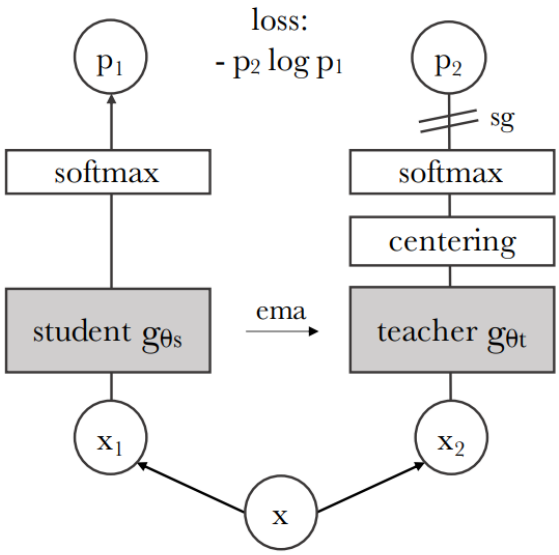

Fig. 3. DINO architecture from [2], showing crops of an input image x getting encoded by student and teacher networks. Note that **sg**stands for "stop gradient", indicating that teacher parameters are not updated by the contrastive loss function, but by an exponential moving average of the student network parameters **ema**.

The authors of [2] used ViTs as the backbones to the student and teacher networks (DINO). They demonstrated that, when trained on ImageNet using the SSL paradigm described above, the student network explicitly learns scene layout and object boundaries. They further show that this information is stored in the attention values of the final layer (Fig. 2). Both of these findings appear to be unique to ViTs trained using SSL [2]. These conclusions were extended in [3], which compared the image encodings generated by DINO between different images and found that image segments with similar semantic meanings — for instance, animal legs from two images of different animals — were encoded similarly. The authors then used these generalizable, semantic encodings for zero-shot semantic segmentation tasks, and showed that these semantic-rich features could perform similar to state of the art supervised and unsupervised techniques [3].

DINO was recently adapted to the RSI analysis domain in [9]. In this work, the authors used identical SSL training setups and ViT backbones to [2], but trained their ViTs with 100k images from the SeCo (seasonal contrast) dataset — a dataset of RSIs which has good global coverage [10] — and with the addition of multi-sized crops to $\mathcal{V}$. Similar to [3] and [2], [9] demonstrates the utility of their model (DINO-MC) on downstream segmentation and classification tasks. On these tasks, DINO-MC performs comparably to the state of the art supervised models but only shows a slight improvement over the original DINO. Unlike the work done in [2] on DINO, [9] did not investigate what, if any, semantic relationships are actually being learned by DINO-MC - a gap that I try to address herein.

II. Methodology

A. Dataset

This work uses a randomly selected subset of 100 images taken from the DeepGlobe dataset. The DeepGlobe dataset consists of 1146 images of (mostly) rural areas in India, Thailand, and Indonesia, covering a total of 1716.9 km^2. Each image is 2448^2 pixels with 3×8-bit channels (RBG). These images have been completely semantically segmented into 7 classes including `agriculture_land`, `urban_land`, `rangeland`, `water`, `barren_land`, `forest_land`, and `unknown`. In total, these segments account for 3042 labelled objects, or ~ 3 segments per image [5].

B. ViT Models

Two ViT model checkpoints are used in this work: DINO as developed by Caron *et al.* and trained on ImageNet over 100 epochs [2], and DINO-MC as developed by Wanyan *et al.* and trained on SeCo over 300 epochs [9]. Both of these models were pre-trained using identical contrastive SSL techniques, as described in Section I-B and Figure 3. Both of these models use the same ViT-S/8 backbones from the seminal ViT paper by Dosovitsky *et al* [11]. This backbone consists of 12 stacked transformer blocks, each with 6 attention heads, a linear encoder, and a positional encoder for a total of 21M trainable parameters.

As input, each model takes a sequence of linear encodings corresponding to *patches* from the original image. Patches are 8^2 pixels (for the ViT-S/**8** backbone) with a stride length of 4 pixels. These patches are linearized, and then combined with a positional encoding before being passed through the models. The output of each model is a sequence of patch embeddings of shape $(N+1) \times d$ where $N+1$ is the length of the input sequence plus an additional `[CLS]` token and $d = 384$ is the embedding size.

Note that while both DINO models were pre-trained on sequences of $N = 197$ (corresponding to images of 224^2 pixels), they can generalize to limited extent to images that are not exactly the size of their training data. Images in this work are resized to 280^2 pixel and positional encodings are interpolated using bicubic interpolation, consistent with existing literature [3], [12].

C. Semantic Maps

1) Attention maps: Attention maps are generated by extracting the attention scores between the `[CLS]` token and every other token in a given layer. These attention scores are extracted for each of the 6 attention heads and normalized 0-1, consistent with what was done in [3]. Composite attention maps are obtained by coloring all individual attention maps and combining them such that only the brightest color in each pixel was kept in the final image [2].

2) Correspondence maps: Correspondence maps are generated by comparing a patch in a source image with all the patches in a target image. The patches are compared by taking the dot product of either the `query`, `key`, `value` embeddings of the patch at a given layer, or the patch (`token`) embeddings themselves. Most experiments within this work use the `key` facet, as this has been empirically demonstrated in [3] to result in semantic maps with lower noise, a result I independently validated (Supplemental Fig. S1).

D. Semantic Segmentation

In addition to the semantic maps, model performances are compared on a semantic segmentation task. The objective is to partition all pixels in an image into one of k clusters in a way such that pixels that are semantically similar are grouped together, typically with the goal of assisting some downstream task like object detection. In traditional segmentation, pixels can be clustered based on a number of factors (color, intensity, parent object shape, etc.) with the ultimate goal of (e.g.) classifying land use type within an image. In this work I segment an image according to the semantic information generated by DINO or DINO-MC.

To perform this semantic segmentation, I use the Graph Convolutional Network (GCN)-based methodology outlined in Aflalo *et al.* [12]. This is an unsupervised technique which uses the deep features predicted by the ViTs and does not assume any *a priori* knowledge of segmentation goals (other than the maximum value of k). This formulation allows for

a segmentation that is purely dependant on the semantic information learned by the ViTs.[1]

This GCN segmentation procedure is outlined briefly below. The reader can refer to [12] for a more complete explanation.

1) GCN formulation: Deep features ($f \in \mathbb{R}^{N \times d}$) are obtained by passing an input image through DINO or DINO-MC and extracting the facet values at a given layer. From these deep features, a patch-wise correlation matrix is constructed according to

$$W = ff^T \cdot (ff^T > 0) \in \mathbb{R}^{N \times N}, \tag{2}$$

where $\cdot(ff^T > 0)$ represents an element-wise thresholding operation. This thresholding operation is necessary as the clustering loss function (Section II-D2) only accepts positive edge weights, but this has the added benefit of preventing the graph from being fully-connected, thus limiting oversmoothing. The patches are then organized into a weighted graph ($G(V, E)$) with W as the weight/adjacency matrix.

2) Objective formulation: With the patches organized as a graph, the semantic segmentation problem thus becomes a node classification problem. Following the methodology outlined in [12], I find optimal node assignments using a single layer GCN and an MLP classifier parameterized by θ_{GCN} and θ_{MLP} respectively. Mainly:

$$\hat{f} = \text{GCN}(f, W; \theta_{GCN}) \tag{3}$$

$$S = \text{MLP}(\hat{f}, W; \theta_{MLP}), \tag{4}$$

where $\hat{f} \in \mathbb{R}^{N \times 64}$ is an intermediate feature matrix, and $S \in \mathbb{R}^{N \times k}$ is the cluster assignment matrix (k clusters). There are a total of 30k trainable parameters between θ_{GCN} and θ_{MLP}.

The classification loss function is

$$\mathbb{L} = \frac{Tr(S^T W S)}{Tr(S^T D S} + \left\| \frac{S^T S}{||S^T S||_F} - \frac{\mathbb{I}_k}{\sqrt{k}} \right\|_F, \tag{5}$$

where D is the sum of the diagonal elements of W, $\mathbb{I}_k$ is the identity matrix, and F denotes the Frobenius norm. Equation (5) is is a differentiable approximation of traditional spectral loss functions proposed in [13] and which was the best performing clustering objective used in [12]. Importantly, this formulation requires that k is pre-specified. $k = 7$ in this work: the total number of labels in the DeepGlobe dataset [5].

3) Training: The GCN and MLP weights are only learned at test time: a new model is loaded and trained from scratch to minimize Equation (5) *for each image in the dataset*. For each image, the clustering pipeline is trained for 100 epochs using the `AdamW` optimizer in PyTorch with a starting learning rate of 0.001 and a `weight_decay` parameter of 0.01.

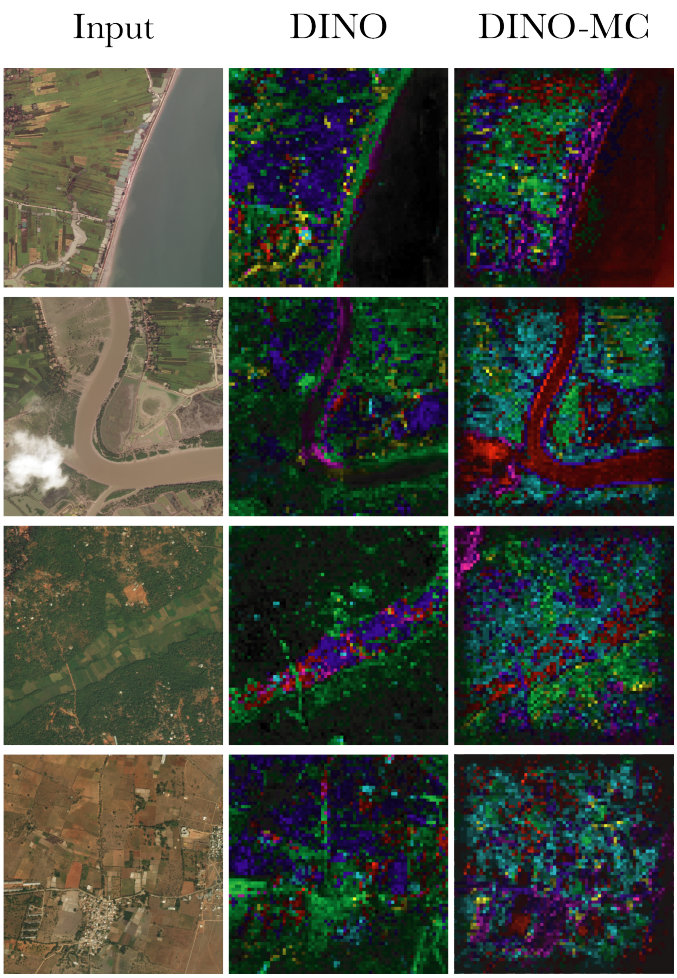

Fig. 4. Attention maps of DINO and DINO-MC on different RSIs from the DeepGlobe dataset. Each color represents separate attention heads in the final (12^{th}) layer of the models. Only the highest attention score (brightest color) is kept for each pixel.

III. Semantic Maps

A. Attention Maps

Attention maps are extracted from the final layer of both DINO and DINO-MC (Fig. 4). While these are qualitative results, and only cover a small fraction of the dataset, it is immediately clear from Fig. 4 that the DINO and DINO-MC attend to the input images in different ways. Similar to the behaviour of DINO on photographs that was observed in [2], both DINO-MC attention heads appear to be attending to semantically different regions within an RSI. This behaviour is muted but still present when DINO is used on RSIs. Interestingly, certain attention heads of DINO-MC appear to attend to similar regions across images; heads 3 (blue) and 4 (magenta) appear to attend mostly to *edges*, and head 1 (red) appears to attend mostly to large, visually contiguous regions (e.g. clouds, water, large fields).

[1]There are surely simpler downstream tasks, but this one gave me nice exposure to GCNs so I picked this one :)

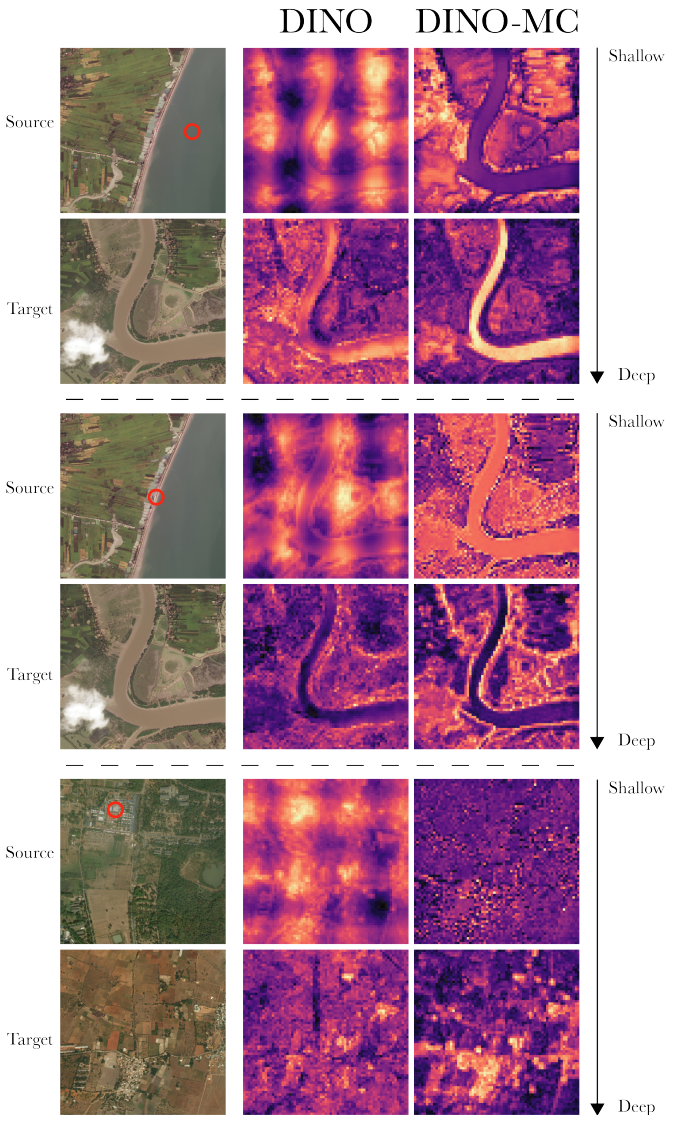

Fig. 5. Correspondence maps generated by taking the dot product of **the key facet** of a source patch embedding (circled in red) with all the patch embeddings in a target image. Results are shown after layers 2 and 12.

There are some notable differences in the attention maps between DINO and DINO-MC. Image brightness in Fig. 4 — which represents the maximum attention score in a given pixel — indicates DINO-MC does a better job attending to the whole image, but the attention drops off as you move radially outward from the center of an image. No clear patterns in total attention are observable when using DINO. The attention maps generated using DINO-MC also appear to be more interpretable. For example: dense forests are attended by different heads than open fields in the third row of Fig. 4, and a similar trend appears for buildings in the fourth row.

B. Correspondence Maps

To investigate what semantic information is being encoded by the model, I repeat a portion of point correspondence analysis from [3]. Fig. 5 shows a comparison between the key embeddings of a source patch to the key embeddings of every patch in a target image. This analysis is repeated for ViT layers 2 and 12.

DINO-MC is clearly better than DINO at encoding semantic information from RSI's. In all images tested, the correspondence maps produced using DINO-MC deep features are less noisy than their DINO counterparts. For the `water` and `shoreline` source patches in Fig. 5, DINO generates similar (but noisier) correspondence maps. However, for the `buildings` source patch, DINO does not generate a correspondence map that shows any (interpretable) semantics, whereas DINO-MC clearly highlights regions in the target image that contain buildings.

[3] reported that "shallow features mostly contain positional information" which gets exchanged for semantic information in deeper layers of the network. As evidence, they highlight the grid generated by the deep features in the earlier layers. This gridded nature is clear in the shallow DINO layers in Fig. 5 but *not* in the shallow DINO-MC layers, suggesting that the behaviours observed in [2] might be a result of training rather than an emergent feature of ViTs in general. Specifically:

1) There could be artefacts (e.g. from compression) in the training data for DINO that isn't present in the RSIs.
2) DINO and DINO-MC use slightly different data augmentation procedures during their contrastive SSL. In particular DINO uses solarization and a constant crop-size, although there is no obvious explanation for why this would lead to the grid-pattern seen in Fig. 5.

It would be worth exploring whether or not these grid artefacts exist in other ViT architectures like the Swin transformer [14] or the recently published DINOv2 [15], and whether or not they exist on DINO models trained on different datasets.

There are a few limitations to this semantic map analysis, the largest of which is that it is purely qualitative. The interpretation of the attention and correspondence maps cannot be assumed to hold for all datasets, or even for all images within a dataset. Adding RSIs from additional datasets may help strengthen these conclusions. Another natural extension would be to generate point correspondence maps, similar to [3] (Supplementary Fig. S2), although neither of these extensions truly solve the lack of a quantitative measurement.

IV. Semantic Segmentation

To give a quantitative description of the ability of DINO and DINO-MC to learn semantic representations of images, I compare the ability of a downstream segmentation task to segment images based on their respective features. After segmenting on the image features from DINO and comparing with the DeepGlobe ground-truth segmentations [5] the average ARI is 0.71. With DINO-MC this value decreases to 0.58, indicating a worse overall segmentation. ARI score distributions are available in Supplemental Fig. S4.

This would seem to contradict the results found in Section III, where DINO-MC appears to perform better than DINO on RSIs. However, the segmentation analysis is very sensitive to the ground truth labels; there is no reason that the unsupervised

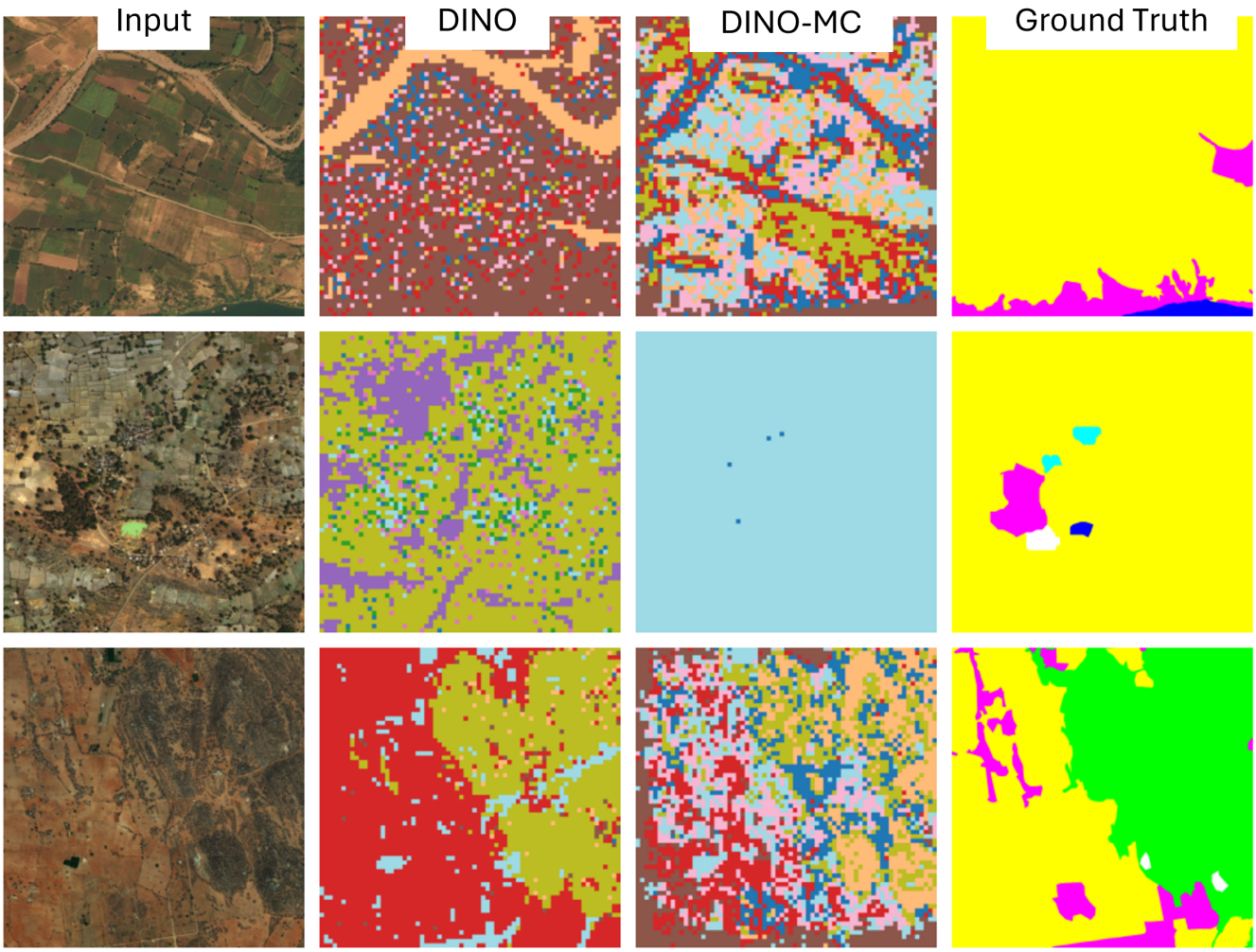

Fig. 6. Semantic segmentation of three different DeepGlobe images after processing with DINO and DINO-MC following the methodology in [12]. The images are ranked by their individual ARI scores (for the DINO model). Rank 100 (worst performing, top), 50, and 1 (best performing, bottom) are shown. Note that colors are randomly assigned to classes, except for `Ground Truth`, whose color-mapping is as follows: ■ `urban_land`, ■ `agricultural_land`, ■ `rangeland`, ■ `forest_land`, ■ `water`, ■ `barren_land`, ■ `unknown`,

segmentation procedure used in this study should necessarily result in classes that alight with the class definitions used in the DeepGlobe semantic segmentation ground truth. It is easy to imagine a situation where a model might be *more* expressive than the ground truth segmentation, yet perform worse than a comparably less-expressive model. An analysis of the worst- and best-performing segmentations confirms the sensitivity of this style of comparison to the ground truth results (Fig. 6). Additionally, image features from DINO-MC achieve lower segmentation losses, faster, than those from DINO (Supplemental Fig. S3). For a lack of other appropriate tools, comparing model performances against potentially misaligned ground-truth labels remains the best option for quantitative analysis. Using additional semantic segmentation datasets (of which there are many [16]) may help this analysis.

V. Conclusion

Computer vision models that can be trained in an unsupervised or self-supervised fashion are an attractive way of overcoming data limitations in the remote sensing field. In this work I explored the extent to which a recent ViT + SSL model is able to learn semantic representations of RSIs. I qualitatively demonstrated that this model learns to generate semantically-rich representations of images — even when trained only on ground-level photographs — and that these representations improve significantly when the model is trained specifically on RSI data. Additionally, I discovered that recent claims about the ability of this model to explicitly learn positional information in early layers is likely an artifact from training, rather than a general feature of this model + training paradigm.

References

[1] X. X. Zhu, D. Tuia, L. Mou, G.-S. Xia, L. Zhang, F. Xu, and F. Fraundorfer, "Deep learning in remote sensing: A comprehensive review and list of resources," *IEEE geoscience and remote sensing magazine*, vol. 5, no. 4, pp. 8–36, 2017.

[2] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin, "Emerging properties in self-supervised vision transformers," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 9650–9660.

[3] S. Amir, Y. Gandelsman, S. Bagon, and T. Dekel, "Deep vit features as dense visual descriptors," *arXiv preprint arXiv:2112.05814*, vol. 2, no. 3, p. 4, 2021.

[4] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[5] I. Demir, K. Koperski, D. Lindenbaum, G. Pang, J. Huang, S. Basu, F. Hughes, D. Tuia, and R. Raskar, "Deepglobe 2018: A challenge to parse the earth through satellite images," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2018.

[6] ESA. (2012, Jun.) Copernicus: Sentinel-2. [Online]. Available: https://www.eoportal.org/satellite-missions/copernicus-sentinel-2#ground-segment

[7] Y. Wang, C. M. Albrecht, N. A. A. Braham, L. Mou, and X. X. Zhu, "Self-supervised learning in remote sensing: A review," *IEEE Geoscience and Remote Sensing Magazine*, vol. 10, no. 4, pp. 213–247, 2022.

[8] A. Jaiswal, A. R. Babu, M. Z. Zadeh, D. Banerjee, and F. Makedon, "A survey on contrastive self-supervised learning," *Technologies*, vol. 9, no. 1, p. 2, 2020.

[9] X. Wanyan, S. Seneviratne, S. Shen, and M. Kirley, "Dino-mc: Self-supervised contrastive learning for remote sensing imagery with multi-sized local crops," *arXiv preprint arXiv:2303.06670*, 2023.

[10] O. Manas, A. Lacoste, X. Giró-i Nieto, D. Vazquez, and P. Rodriguez, "Seasonal contrast: Unsupervised pre-training from uncurated remote sensing data," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 9414–9423.

[11] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[12] A. Aflalo, S. Bagon, T. Kashti, and Y. Eldar, "Deepcut: Unsupervised segmentation using graph neural networks clustering," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 32–41.

[13] F. M. Bianchi, D. Grattarola, and C. Alippi, "Spectral clustering with graph neural networks for graph pooling," in *International conference on machine learning*. PMLR, 2020, pp. 874–883.

[14] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 10 012–10 022.

[15] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby *et al.*, "Dinov2: Learning robust visual features without supervision," *arXiv preprint arXiv:2304.07193*, 2023.

[16] M. Schmitt, S. A. Ahmadi, Y. Xu, G. Taşkın, U. Verma, F. Sica, and R. Hänsch, "There are no data like more data: Datasets for deep learning in earth observation," *IEEE Geoscience and Remote Sensing Magazine*, 2023.

Supplemental Information

A. Facet analysis

Figure S1 shows the effects of generating correspondence maps with the different facets (key, value, query, or the token embedding itself). In line with the findings in [3] I find that the key facet consistently has the best representations and the lowest noise.

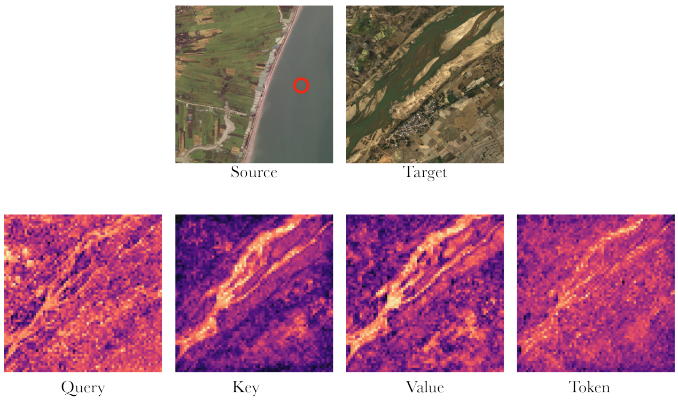

Fig. S1. Correspondence maps using different model facets, all from the final (12^{th}) layer.

B. Point correspondence maps

Figure S2 shows two point correspondence maps taken from [3]. Similar to the correspondence maps generated in my work, it compares a source patch embedding with all patch embeddings from a target image, but then *only returns the most similar target patch*, allowing for a slightly richer interpretation of how well a model captures semantic similarity.

Fig. S2. Point correspondence maps from [3]. Source images are on the left, and target images on the right. Colored dots in the target image represent the patch that is most similar to the same colored patch in the source image.

C. Semantic segmentation results

Figure S3 shows when image features are generated with DINO-MC, GCN clustering loss is lower, faster than with DINO. This gives some evidence that the DINO-MC features are more useful than DINO features, but this is largely obscured by the poor ARI results in Section IV.

Fig. S3. Semantic segmentation training loss after training a GCN 100 separate times (100 different DeepCut images) over 100 epochs.

Figure S4 shows the ARI distribution after segmentations with features predicted by both models. DINO-MC does not segment as well as DINO, but this is likely do to a Ground Truth that is poorly aligned with the unsupervised segmentation, as discussed in Section IV.

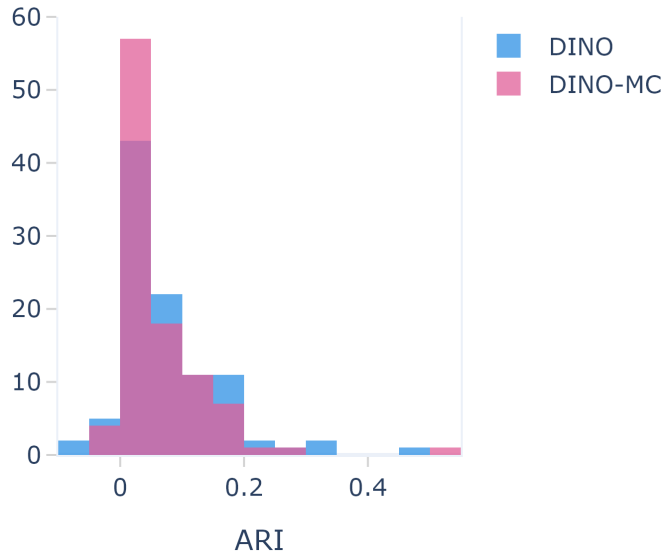

Fig. S4. ARI distributions after segmenting 100 different DeepCut images and comparing them to the ground truth labels provided by DeepCut.

D. Points I didn't get to make in the main body

1) Smoothing: In [12], the authors use smoothing to simplify the segmentation maps they generate. The high-resolution and scattered, small information (objects) in satellite images means these techniques are a little less appropriate when working with RSI data.

2) GCN clustering: [12] also develops a k-less clustering loss function, but found that Equation (5) worked better. While this requires defining a maximum number of clusters, it didn't end up being that limiting. Additionally, [12] says that their k-less clustering takes advantage of negative edge weights, but this is in fact only during the loss calculation; negative edges are pruned before passing the graph through the GCN, as there is no universally accepted way to treat negative edges in a GCN.

3) Pretext tasks: I am unsure if the cropping/augmenting strategy that DINO used for SSL on traditional photographs is the best strategy for SSL on RSI data. Particularly the idea that predicting local-to-global correspondence (whether a local crop is part of a global crop) has as much meaning for satellite images: is a local crop of a satellite image not just another, smaller satellite image? I don't have any sources to back this up, and clearly this pretext task still allows for improved performance of DINO-MC over DINO, but it is something I have been wondering.

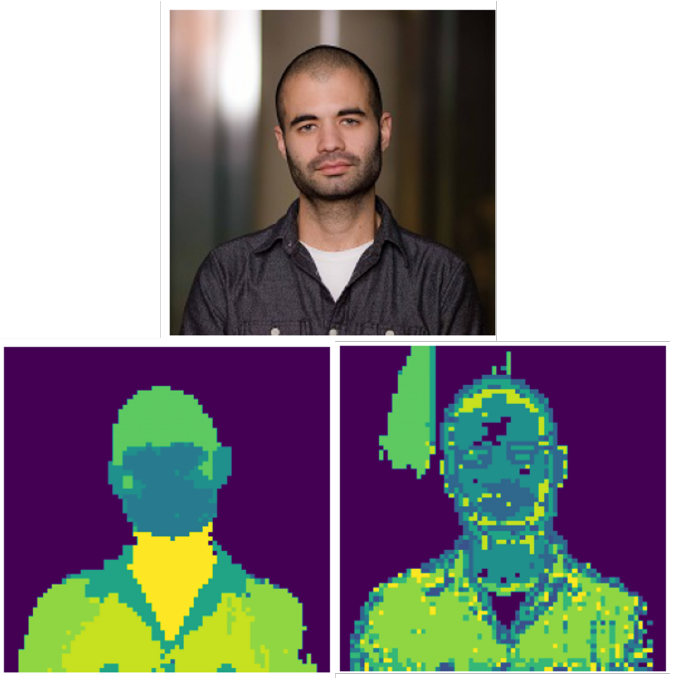

Fig. S5. Segmentation results of Dr. Eldan Cohen after image encoding with DINO (left) and DINO-MC (right), clearly illustrating that DINO-MC has learned to prioritize edges in an image.