Vision Transformers Need Registers

Mechanistic Debugging of High-Norm Tokens in ViTs

How did we get here?

- Feature Learning (via general-use embeddings) key problem in computer vision <— text-image alignment (CLIP, VLMs), SSL (DINO), sup learning with ViTs (DeiT).

- DINO in particular gives strong segmentation performance and a semantically meaningful last attention layer.

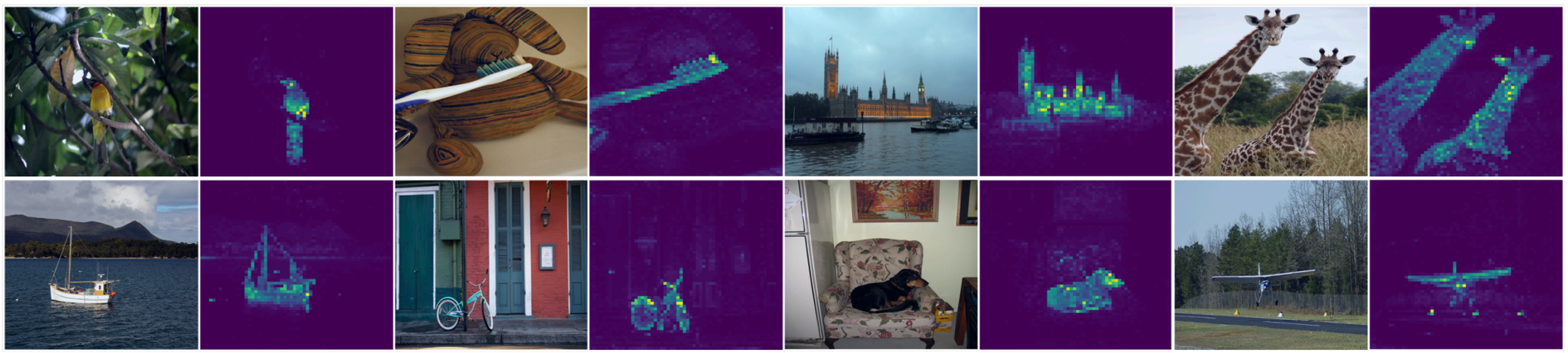


Figure 1: **Self-attention from a Vision Transformer with 8×8 patches trained with no supervision.** We look at the self-attention of the `[CLS]` token on the heads of the last layer. This token is not attached to any label nor supervision. These maps show that the model automatically learns class-specific features leading to unsupervised object segmentations.

- Object discovery: Category discovery + unsupervised localisation.

- LOST is a strong object discovery method built on top of DINO:

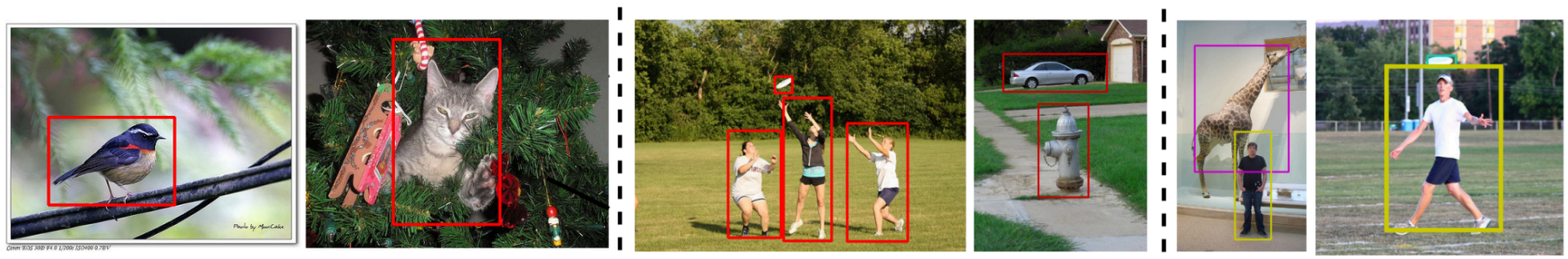


Figure 1: Three applications of LOST to unsupervised single-object discovery (left), multi-object discovery (middle) and object detection (right). In the latter case, objects discovered by LOST are clustered into categories, and cluster labels are used to train a classical object detector. Although large image collections are used to train the underlying image representation [13] and the detector [51], *no annotation* is ever used in the pipeline. See Figure 3 and Tables 1, 3 for more experiments.

- Authors of this paper (and DINOv2) wanted to improve results on object discovery using DINOv2.

- However, DINOv2 + LOST doesn't work due to **artifacts** in the feature maps. Upon further investigation, these also appear in CLIP and DeiT vision transformers, so DINO is an exception and these artifacts are the baseline (per the paper's claims).

- This paper is about these artifacts.

What are these artifacts?

- A small fraction of tokens (~2% of input) with **10x higher norm at the output**.
- The distribution of feature norms over a dataset is bimodal.
- High-norm tokens = outliers/artifacts

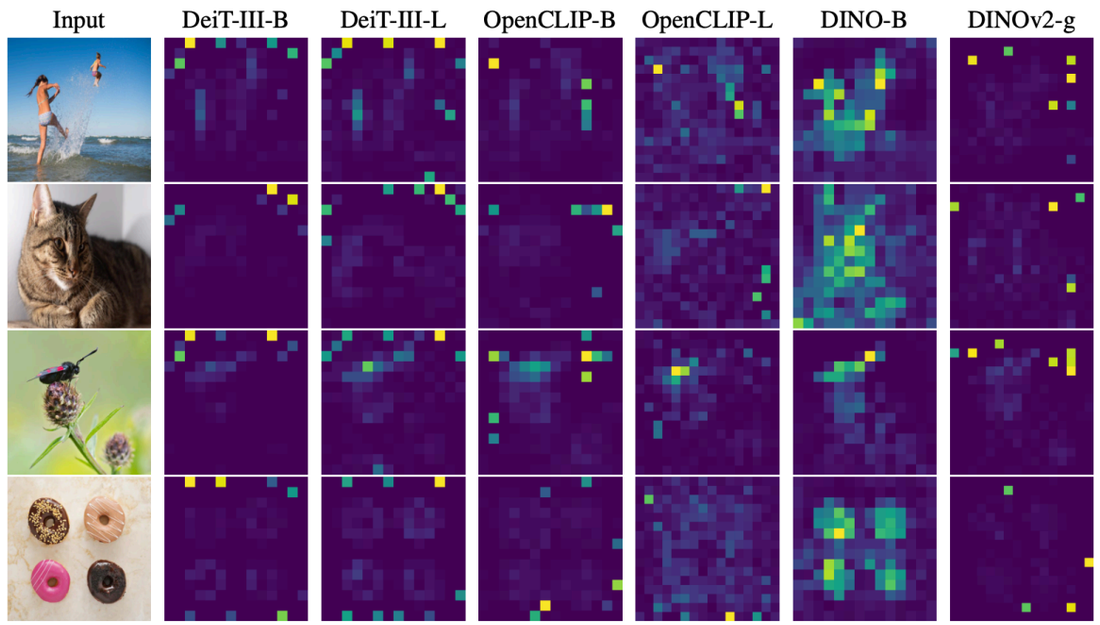


Figure 2: Illustration of artifacts observed in the attention maps of modern vision transformers. We consider ViTs trained with label supervision (DeiT-III), text-supervision (OpenCLIP) or self-supervision (DINO and DINOv2). Interestingly, all models but DINO exhibit peaky outlier values in the attention maps. The goal of this work is to understand and mitigate this phenomenon.

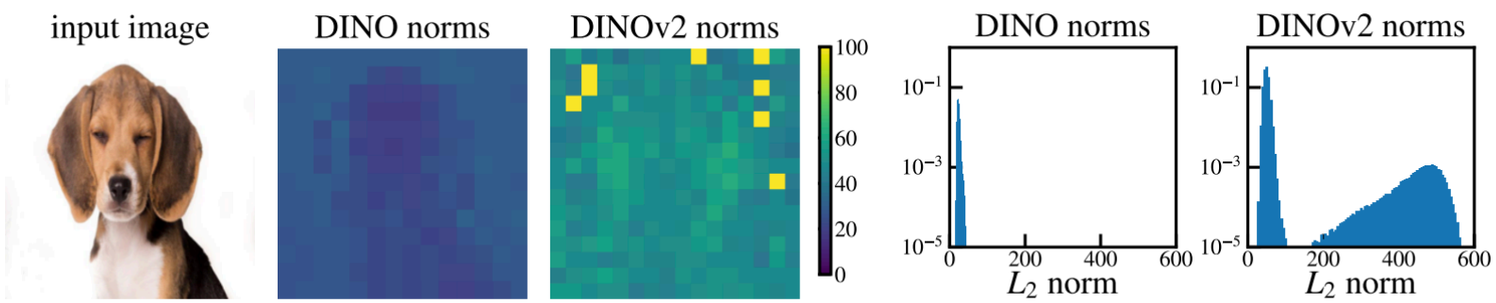
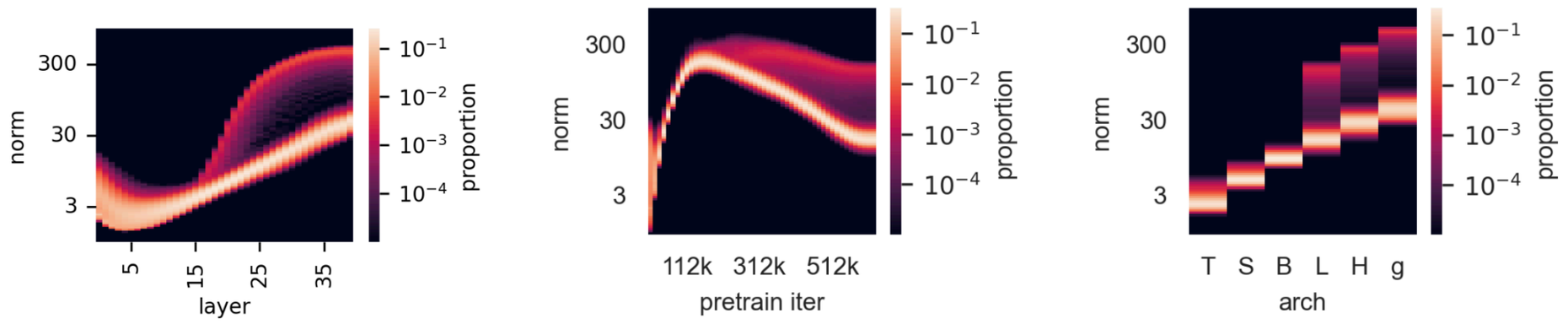


Figure 3: Comparison of local feature norms for DINO ViT-B/16 and DINOv2 ViT-g/14. We observe that DINOv2 has a few outlier patches, whereas DINO does not present these artifacts. For DINOv2, although most patch tokens have a norm between 0 and 100, a small proportion of tokens have a very high norm. We measure the proportion of tokens with norm larger than 150 at 2.37%.

When do they appear?

- Outliers appear from the **15th** layer downwards in g variant (middle of model).
- They start differentiating themselves from normal patches **1/3** of the way through training.
- They appear when training **L**, **H**, **g** variants (for DINOv2), <u>not</u> for smaller models.



(a) Norms along layers. (b) Norms along iterations. (c) Norms across model size.

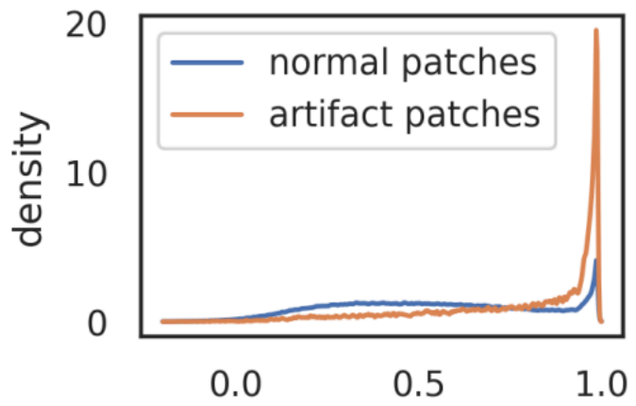
Figure 4: Illustration of several properties of outlier tokens in the 40-layer DINOv2 ViT-g model. **(a)**: Distribution of output token norms along layers. **(b)**: Distribution of norms along training iterations. **(c)**: Distribution of norms for different model sizes. The outliers appear around the middle of the model during training; they appear with models larger than and including ViT-Large.

Where do they appear?

- Outliers appear in patches where there is little useful signal (seems logical, plays into narrative*). But how do the authors substantiate this claim?

- Probe the output of the **patch embedding** layer (start of the model) -> measure cosine similarity between 4 neighbouring patch embeddings for normal vs outlier patches -> outliers are more similar to their neighbours more often.

- Bonus: Plot distribution of outliers along image (patch) locations -> we notice aliasing artifacts (stripes) -> apply anti-aliasing -> outlier position becomes fairly standardised near edges of image.



(a) Cosine similarity to neighbors.

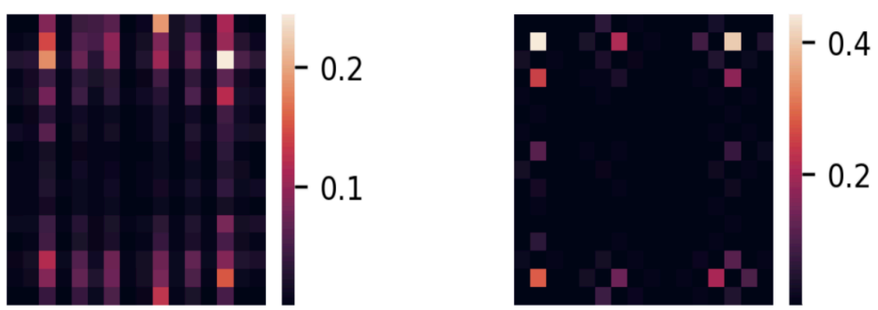


Figure 10: Feature norms along locations: proportion of tokens with norm larger than the cutoff value at a given location. Left: official DINOv2 model (no antialiasing), right: our models (with antialiasing). At some positions, more than 20% of tokens have a high norm.

What do they contain?

- Narrative/Goal*: Show outliers contain 1) little local information and 2) disproportionate global information.

- 1: **position prediction** and **pixel value reconstruction** via linear probing (1 probe for each objective) -> normal patches outperform outliers -> outliers contain less local information.

- 2: Linear probe classification performance on multiple datasets for random normal and random outlier patches -> outlier patches outperform normal ones (sometimes even [CLS] token!) -> outliers contain more global information.

	position prediction		reconstruction
	top-1 acc	avg. distance ↓	L2 error ↓
normal	**41.7**	**0.79**	**18.38**
outlier	22.8	5.09	25.23

(b) Linear probing for local information.

	IN1k	P205	Airc.	CF10	CF100	CUB	Cal101	Cars	DTD	Flow.	Food	Pets	SUN	VOC
[CLS]	**86.0**	**66.4**	**87.3**	**99.4**	**94.5**	**91.3**	96.9	**91.5**	**85.2**	**99.7**	**94.7**	**96.9**	**78.6**	89.1
normal	65.8	53.1	17.1	97.1	81.3	18.6	73.2	10.8	63.1	59.5	74.2	47.8	37.7	70.8
outlier	69.0	55.1	79.1	99.3	93.7	84.9	**97.6**	85.2	84.9	99.6	93.5	94.1	78.5	**89.7**

Table 1: Image classification via linear probing on normal and outlier patch tokens. We also report the accuracy of classifiers learnt on the class token. We see that outlier tokens have a much higher accuracy than regular ones, suggesting they are effectively storing global image information.

Why do they appear?

- Only hypotheses: for all their work, they cannot establish a causal relationship

- H1: "**Large**, **sufficiently trained** models learn to recognise *redundant* tokens, and to use them as places to **store**, **process** and **retrieve** global information. (a.k.a. auxiliary [CLS] tokens)

- H2: This behaviour in actual image patches leads to discarding local patch information, possibly hurting dense prediction tasks.

Proposed Solution

- Add register tokens after patch embedding layer (just like the [CLS] token). Think of them like scratchpads. After training, bin them.

- Nothing forces a specific behaviour for [reg] tokens, model is free to use them as it wants.

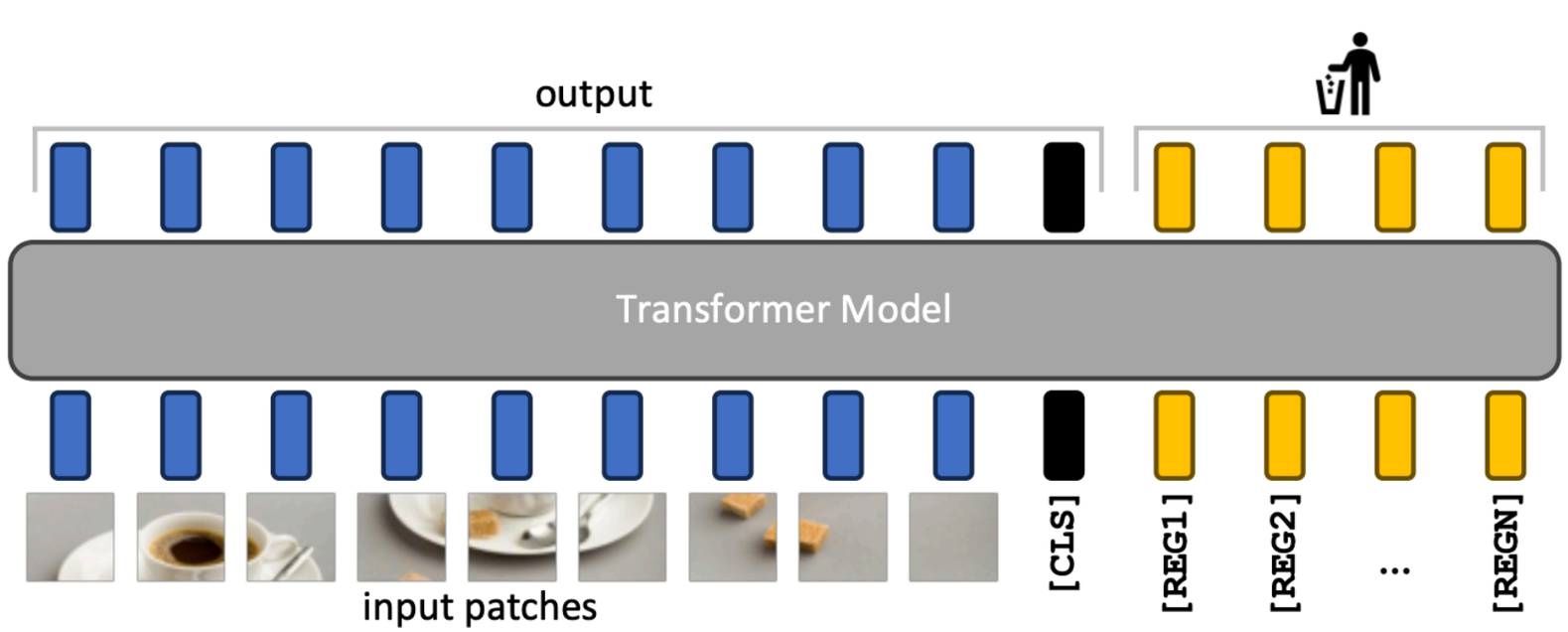


Figure 6: Illustration of the proposed remediation and resulting model. We add N additional learnable input tokens (depicted in yellow), that the model can use as *registers*. At the output of the model, only the patch tokens and `[CLS]` tokens are used, both during training and inference.

How many registers are needed?

- Investigation only done for DINOv2.
- “Step behaviour” between 0 and 1 registers
- FLOP increase of ~2% per 4 registers
- Authors choose 4 for experiments based on segmentation performance as it related more to object discovery*

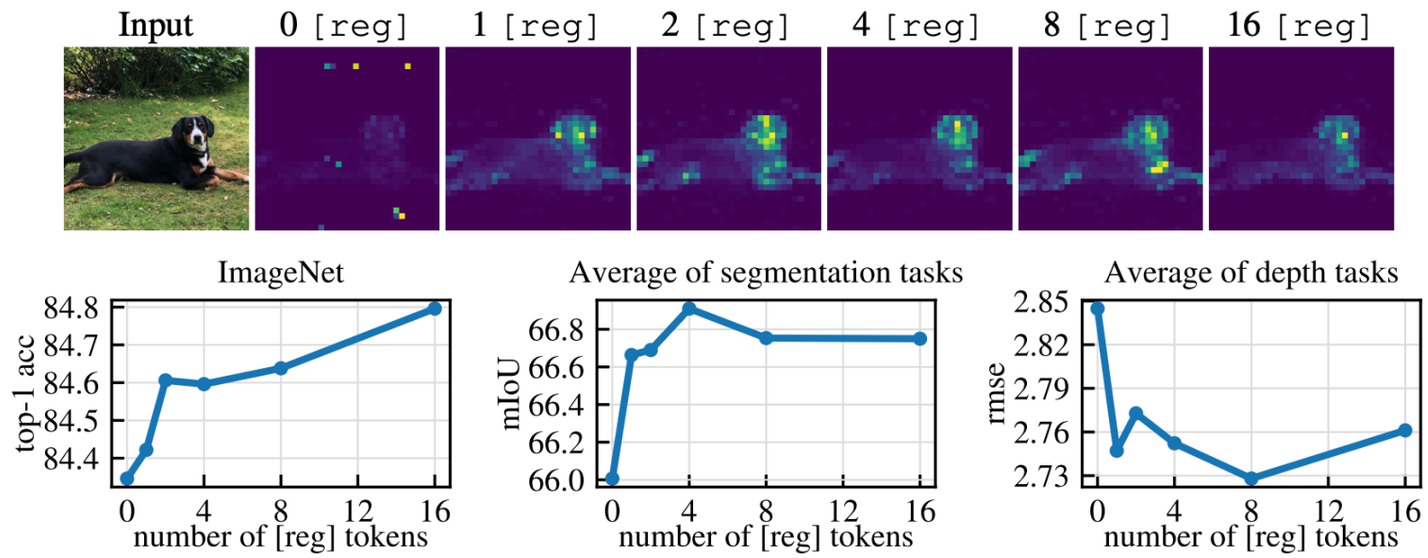


Figure 8: Ablation of the the number of register tokens used with a DINOv2 model. **(top)**: qualitative visualization of artifacts appearing as a function of number of registers. **(bottom)**: performance on three tasks (ImageNet, ADE-20k and NYUd) as a function of number of registers used. While one register is sufficient to remove artefacts, using more leads to improved downstream performance.

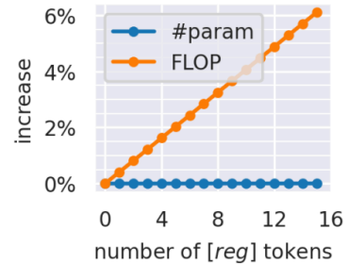


Figure 12: Increase in model parameter and FLOP count when adding different numbers of registers. Adding registers can increase model FLOP count by up to 6% for 16 registers. However, in the more common case of using 4 registers, that we use in most of our experiments, this increase is below 2%. In all cases, the increase in model parameters is negligible.

Effect on norm distribution

1. Registers remove high-norm mode from distribution:

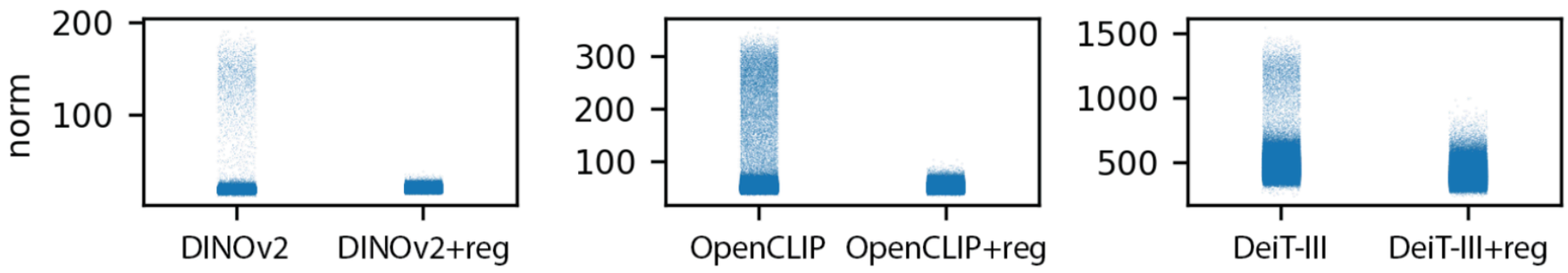
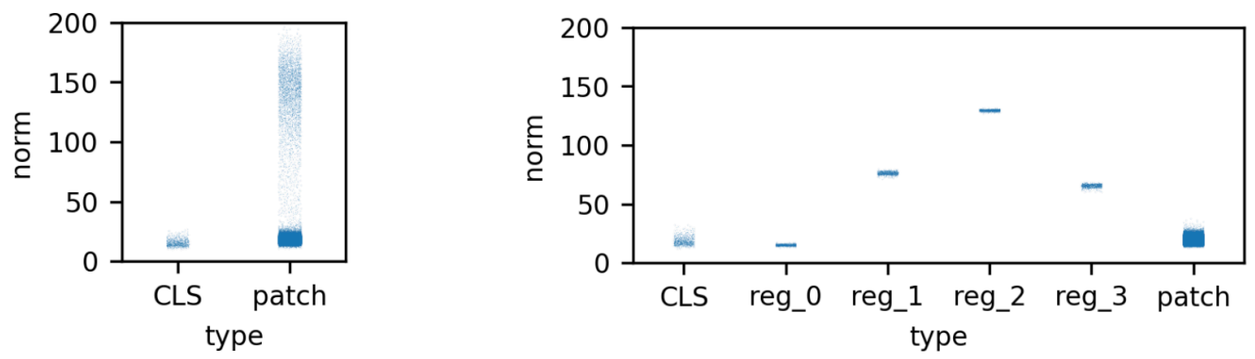


Figure 7: Effect of register tokens on the distribution of output norms on DINOv2, OpenCLIP and DeiT-III. Using register tokens effectively removes the norm outliers that were present previously.

2. Register norms appear quantised, the “why” remains unclear:



(a) DINOv2 - no register

(b) DINOv2 - 4 registers

Do registers affect downstream performance?

- IN1K performance increased for DINOv2 (not for CLIP, DeiT)
- ADE20k performance increased across the board

	ImageNet Top-1	ADE20k mIoU	NYUd rmse ↓
DeiT-III	84.7	38.9	0.511
DeiT-III+reg	84.7	39.1	0.512
OpenCLIP	78.2	26.6	0.702
OpenCLIP+reg	78.1	26.7	0.661
DINOv2	84.3	46.6	0.378
DINOv2+reg	84.8	47.9	0.366

(a) Linear evaluation with frozen features.

- But original goal was object discovery!
- Massive increase for DeiT, DINOv2 👍
- Slight regression for CLIP ☹️
- Even best results (DINOv2) a lot behind DINO 😐

Method	VOC07_trainval	VOC12_trainval	COCO_20k
LOST (ours)	**61.9**	**64.0**	**50.7**

	VOC 2007	VOC 2012	COCO 20k
DeiT-III	11.7	13.1	10.7
DeiT-III+reg	27.1	32.7	25.1
OpenCLIP	38.8	44.3	31.0
OpenCLIP+reg	37.1	42.0	27.9
DINOv2	35.3	40.2	26.9
DINOv2+reg	55.4	60.0	42.0

Table 3: Unsupervised Object Discovery using LOST (Siméoni et al., 2021) on models with and without registers. We evaluated three types of models trained with various amounts of supervision on VOC 2007, 2012 and COCO. We measure performance using corloc. We observe that adding register tokens makes all models significantly more viable for usage in object discovery.

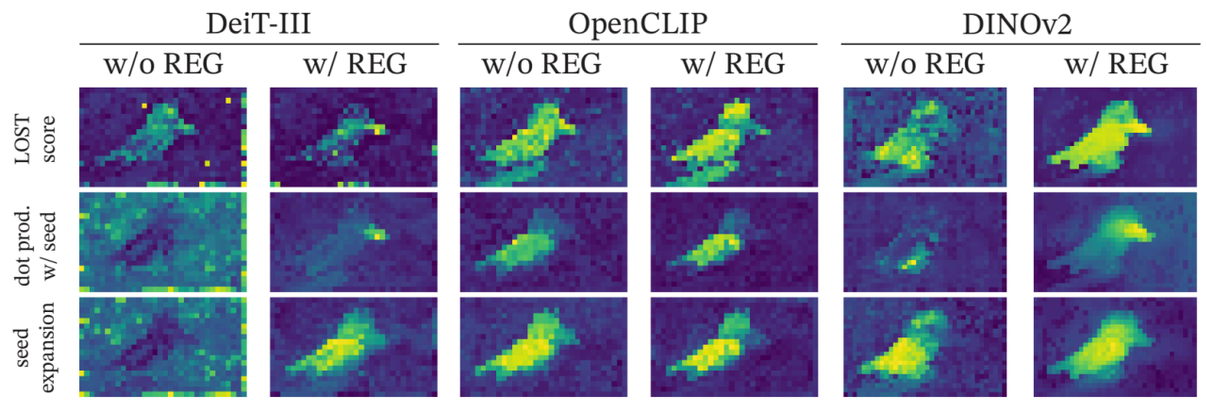


Figure 13: Illustration of the intermediate computations in the LOST algorithm for all models. Adding registers drastically improves the look of all intermediate steps for DeiT-III and DINOv2. The difference is less striking for the OpenCLIP model.

Qualitatively the attention maps look a lot better!

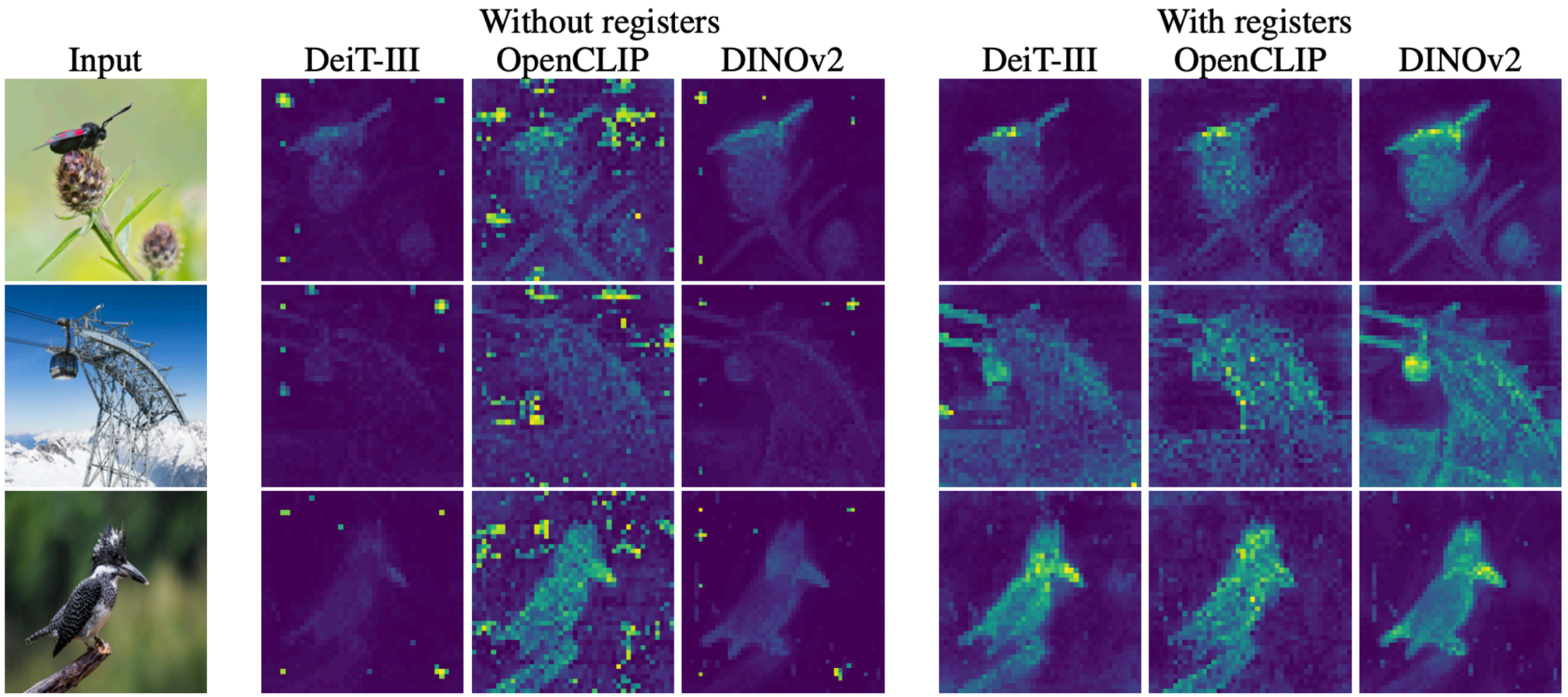


Figure 1: Register tokens enable interpretable attention maps in all vision transformers, similar to the original DINO method (Caron et al., 2021). Attention maps are calculated in high resolution for better visualisation. More qualitative results are available in appendix H.

What do registers learn to do?

Inconclusive. Here's an example of how they can occasionally learn to attend to actual image objects:

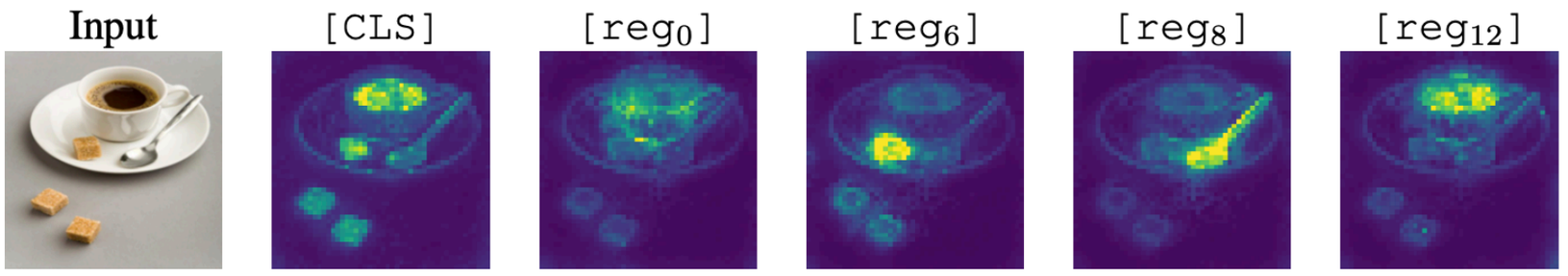


Figure 9: Comparison of the attention maps of the [CLS] and register tokens. Register tokens sometimes attend to different parts of the feature map, similarly to slot attention (Locatello et al., 2020). This behaviour was never required from the model, and emerged naturally from training.

A limited experiment in the appendix offers some evidence registers **absorb** outlier probing attributes:

#registers	top-1 accuracy [CLS]	normal patch	outlier patch	register
0	84.6	15.5	73.3	N/A
1	85.2	14.5	N/A	71.1

Table 4: Linear probing of models with and without registers on the Aircraft dataset, using various tokens as representation. We observe that the behavior of the outlier tokens, aggregating global information, is absorbed into the register.

Notes

- Extremely well-paced and laid out paper. They make you completely forget there a dataset component, the whole narrative is written independent of it as if we a priori accept these are general observations regardless of the dataset and settings used to train the ViTs. (NB: DINO was trained on IN1k, everything in this paper on IN21k).

- Mild acknowledgement of the above: “We note that we have not been able to fully determine which aspects of the training led to the appearance of artifacts in different models. The pre-training paradigm seems to play a role, as OpenCLIP and DeiT-III exhibit outliers both at size B and L (Fig. 2). However, the model size and training length also play important parts, as observed in Fig. 4.”

- Despite this being one of the clearest, crispest mechanistic interpretability works you’ll find, there are still bizarre behaviours that we cannot account for like register norms being quantized, and registers attending to specific parts of the image, sometimes.