LiFT: A Surprisingly Simple Lightweight Feature Transform for Dense ViT Descriptors (Supplemental Material)

Saksham Suri* Matthew Walmer* Kamal Gupta Abhinav Shrivastava

University of Maryland, College Park

A Ablation Study of LiFT Design Choices

We present a careful analysis of LiFT design configurations by varying different factors in both training and inference. To show the general applicability and benefits of LiFT, we include three different backbones in this study: DINO [1], MoCo v3 (MoCo for short) [4], and a Fully Supervised ViT (ViT for short). We standardize the architecture to a ViT S/16 backbone for this analysis. We use the SPair-71k dataset and the keypoint correspondence task as the main representative metric for this analysis. The results are summarized in Table 1.

A.1 Random LiFT

One might question if LiFT actually benefits from training, or if the simple act of increasing the feature resolution with any arbitrary function is sufficient to improve performance. To test this question, we take a random initialization of the LiFT model and measure its performance. We denote this model as 'Random LiFT' in Table 1. It can clearly be seen in Table 1 rows 2, 8, and 14 that a randomly initialized LiFT model does not do anything meaningful as it performs poorly on all metrics. These results validate the importance of LiFT's self-supervised training method.

A.2 Ablation of Image Input to LiFT

In our approach, we increase the feature resolution through LiFT by also using the image as a source of finer spatial information. It should be noted that we use the image at the same resolution as was used to generate the initial features, which means LiFT does not have or require any additional information beyond the original ViT's input. To show the importance of this image information, we present a version of LiFT with the image input ablated, denoted as 'LiFT No Img.' in Table 1. We can see from rows 3 *vs.* 6, 9 *vs.* 12, and 15 *vs.* 18, that providing the image input helps LiFT produce better quality features which give improved performance on the keypoint correspondence task. It appears that ablating the image input is less harmful for higher-resolution inputs like 448,

* Equal contributors.

Table 1: Ablation of different design decisions for LiFT training for three different ViT backbones. We report PCK@0.1 and PCK@0.05 on SPair-71k. For each backbone, we mark the best score for each metric and input resolution in **bold**.

		PCK@0.1				PCK@0.05			
Row	Method/Resolution	56	112	224	448	56	112	224	448
1	DINO	2.04	12.67	24.76	28.6	0.51	3.61	9.54	15.33
2	DINO + Random LiFT	1.45	2.37	4.21	6.16	0.35	0.7	1.41	2.35
3	DINO + LiFT No Img.	4.38	15.74	28.49	**31.42**	1.14	5.03	13.28	18.33
4	DINO + LiFT L1	4.48	16.64	27.77	31.03	1.01	5.93	13.88	18.09
5	DINO + LiFT L2	4.82	**17.72**	28.17	31.13	**1.29**	6.18	14.12	18.37
6	DINO + LiFT	**5.05**	**17.72**	**28.68**	31.38	1.19	**6.29**	**14.72**	**18.90**
7	MOCO	1.27	3.43	7.37	12.31	0.21	0.84	2.35	5.49
8	MOCO + Random LiFT	2.59	3.08	4.05	5.79	0.67	0.77	1.31	2.1
9	MOCO + LiFT No Img.	4.58	8.78	13.01	15.48	1.18	2.69	4.95	7.27
10	MOCO + LiFT L1	6.12	9.80	13.73	14.98	1.59	3.22	5.86	7.53
11	MOCO + LiFT L2	6.37	10.08	13.91	**16.41**	1.53	3.12	5.99	**8.34**
12	MOCO + LiFT	**6.48**	**10.51**	**14.13**	16.34	**1.74**	**3.36**	**6.42**	8.05
13	ViT	1.26	5.72	13.23	16.9	0.27	1.62	4.89	7.34
14	ViT + Random LiFT	2.36	3.29	7.15	8.21	0.58	1.09	2.33	3.13
15	ViT + LiFT No Img.	2.94	7.76	15.69	18.74	0.79	2.22	5.68	8.23
16	ViT + LiFT L1	3.27	8.32	16.04	18.29	0.79	2.74	6.77	8.45
17	ViT + LiFT L2	3.57	8.78	16.29	**18.80**	0.97	2.64	**6.82**	**8.87**
18	ViT + LiFT	**3.76**	**9.21**	**16.5**8	18.69	**1.02**	**2.71**	6.63	8.81

which makes intuitive sense as the feature map resolution is higher and thus more detail about the object boundaries can be represented. For DINO and the supervised ViT (rows 3 and 15), the no-image LiFT actually does very slightly better at 448 input resolution for PCK@0.1, but for all other cases normal LiFT is better. For PCK@0.05, the standard LiFT with image input is consistently much better. We believe this happens because LiFT can take direct cues regarding scene and object boundaries from the image input and generate higher resolution features which better respect these contours.

A.3 Effect of Distance Function

In our final approach, we use cosine distance to compute the loss between the ViT-generated higher resolution features and the upscaled features from LiFT. In Table 1, we compare with two alternative options for this distance function, specifically the L1 and L2 distance metrics. We denote these as 'LiFT L1' and 'LiFT L2' respectively. Cosine distance gives the best performance in most cases, such as in rows 4 & 5 *vs*. row 6, rows 10 & 11 *vs*. row 12, and rows 16 & 17 *vs*. row 18. For higher-resolution inputs, L2 distance is sometimes slightly better than cosine distance, but in most cases cosine is preferable. We believe this occurs because of the inherent normalization that cosine distance provides before computing the final loss.

A.4 Ablation of Training Epochs

As an additional experiment, we train the LiFT module on ImageNet for an extended period up to 100 epochs on 4 GPUs in Table 2. We find that there are

Table 2: Ablation of LiFT training epochs on ImageNet, including longer training. Results are shown for DINO+LiFT on Keypoint Correspondence using PCK@0.1.

Res/Epochs	5	10	30	50	100
112×112	17.47	17.53	17.75	17.97	18.14
224×224	28.45	28.50	28.65	29.00	29.11

small performance gains from training to very long epochs, however performance mostly saturates by epoch 5. At resolution 224, DINO+LiFT at 5 epochs gives a $\sim$3.7 point gain over the base DINO model, while training 95 epochs further only gives an additional 0.66 point gain. We believe this early saturation is thanks to the LiFT network's small size.

B Additional Details for ViTDet+LiFT

For our experiments combining LiFT with ViTDet [5], we increase the size of our LiFT module to address the additional complexity of the task and backbone. To be consistent with ViTDet, we use an MAE-trained ViT-Base backbone instead of the ViT-Small used in our other primary experiments. Note that a standard ViT-Small model outputs feature maps with 384 channels, while ViT-Base outputs 768 channels. To handle the increased number of channels, we commensurately increase the number of channels in the layers of our LiFT module. We also add an additional convolutional block to the encoder segment. This larger LiFT module has a total of $7M$ parameters, as compared with the $1.2M$ parameter version used for smaller architectures. The ViTDet model used has $111M$ parameters, so our combined ViTDet+LiFT architecture has $118M$ parameters total. This is a 6.3% increase in total parameters, which is similar to the relative percentage increase of the smaller LiFT version for DINO ViT-S/16. Also, here we train LiFT on the COCO dataset in place of ImageNet. Because the COCO dataset is much smaller than ImageNet, we train on it for 100 epochs.

C Additional Backbones with LiFT

We further demonstrate the general utility of LiFT by applying it to several additional backbones, including Leopart [6] and several other DINO [1] ViTs, namely ViT-S/8, ViT-B/16, and ViT-B/8. The results are summarized in Table 3. LiFT shows consistent improvement for the various architectures and models across patch sizes (8 and 16), trainings (Leopart and DINO), and backbone sizes (Base and Small). We also extend the Performance *vs*. Compute Cost analysis curve to include both the MOCO and fully-supervised ViT backbones, as shown in Figure 1. We find that LiFT consistently boosts the performance of all three backbones at all FLOP allowances.

Table 3: Application of LiFT to various backbones for the Keypoint Correspondence task on SPair-71k for all metrics. LiFT gives consistent performance improvements.

	PCK@0.1				PCK@0.05				PCK@0.01			
Method/Resolution	56	112	224	448	56	112	224	448	56	112	224	448
DINO S/16	2.04	12.67	24.76	28.60	0.51	3.61	9.54	15.33	0.01	0.20	0.54	1.40
DINO S/16 + LiFT	5.05	17.72	28.68	31.38	1.19	6.29	14.72	18.90	0.06	0.29	0.91	2.52
DINO B/16	1.98	12.20	24.90	28.22	0.46	3.61	9.64	15.04	0.01	0.17	0.52	1.15
DINO B/16 + LiFT	5.43	17.74	29.35	31.27	1.29	6.56	14.80	18.10	0.04	0.37	0.92	2.43
DINO S/8	9.39	21.30	31.05	32.15	2.35	8.44	16.74	18.96	0.15	0.39	1.19	2.32
DINO S/8 + LiFT	12.90	26.73	34.54	34.58	4.35	11.99	20.61	21.01	0.18	0.75	2.21	3.77
DINO B/8	8.88	20.40	30.08	30.89	2.83	7.70	15.81	17.84	0.12	0.39	1.09	1.95
DINO B/8 + LiFT	12.21	25.17	33.23	33.17	4.22	11.73	19.39	20.18	0.13	0.69	2.27	3.32
MOCO S/16	1.27	3.43	7.37	12.31	0.21	0.84	2.35	5.49	0.00	0.03	0.10	0.31
MOCO S/16 + LiFT	6.48	10.51	14.13	16.34	1.74	3.36	6.42	8.05	0.04	0.16	0.42	0.73
ViT S/16	1.26	5.72	13.23	16.90	0.27	1.62	4.89	7.34	0.02	0.06	0.30	0.50
ViT S/16 + LiFT	3.76	9.21	16.58	18.69	1.02	2.71	6.63	8.81	0.02	0.13	0.45	0.72
Leopart S/16	2.35	11.20	23.33	26.54	0.60	3.22	8.90	12.26	0.05	0.10	0.47	0.79
Leopart S/16 + LiFT	4.24	15.61	27.77	30.06	1.22	5.16	12.81	15.66	0.02	0.25	0.74	1.39

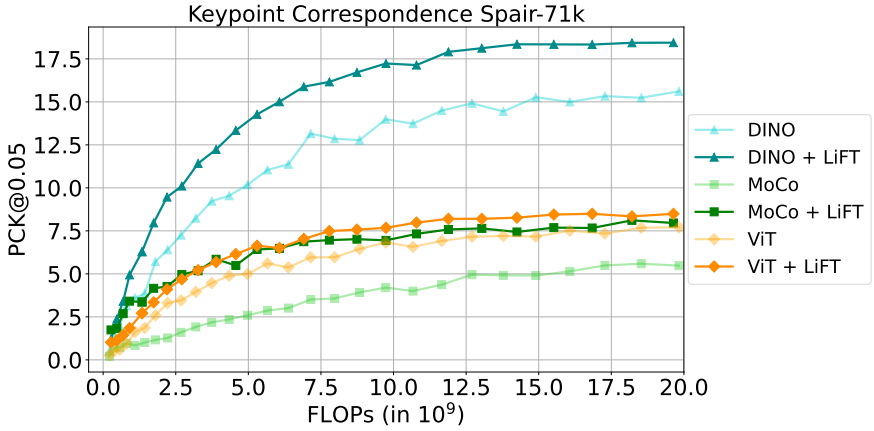

Fig. 1: Performance *vs*. Compute Cost trade-off curve for LiFT when combined with different ViT backbones. Results are presented for SPair-71k Keypoint Correspondence. LiFT provides a performance boost for all three backbones at any FLOP budget.

Table 4: Comparison between LiFT and other baselines on the DAVIS Video Object Segmentation task with additional metrics J Mean and F Mean.

	J Mean				F Mean				J & F Mean			
Method/Res.	56	112	224	448	56	112	224	448	56	112	224	448
DINO	9.50	21.90	37.80	52.10	5.20	13.10	28.10	49.70	7.40	17.50	33.00	50.90
Leopart	9.00	20.20	34.90	47.30	4.80	12.10	25.70	42.80	6.92	16.12	30.33	45.08
SelfPatch	9.70	21.60	38.10	52.50	5.10	12.90	27.90	50.30	7.40	17.23	33.01	51.37
DINO+BL	13.70	29.40	42.80	54.00	7.90	17.90	31.20	52.00	10.78	23.66	37.01	53.02
DINO+RC	13.80	29.60	43.00	53.90	8.20	18.40	31.90	52.50	11.00	24.00	37.40	53.20
DINO+JBU	13.90	30.60	42.80	54.90	8.60	21.70	35.10	54.10	11.20	26.20	39.00	54.50
DINO+LiFT	**16.27**	**33.04**	**48.07**	**59.43**	**9.72**	**23.00**	**40.56**	**62.79**	**13.00**	**28.02**	**44.32**	**61.11**

D Additional Results

We present results for additional metrics on DAVIS, reporting the J Mean and F Mean in Table 4. We present these results alongside the previously reported J & F mean for completeness. We find that both the J Mean and F Mean are also consistently improved by adding LiFT, and that DINO+LiFT surpasses all other baselines. We also present additional Unsupervised Object Discovery results for PASCAL VOC 2007 [2] and PASCAL VOC 2012 [3] in Table 5 alongside the results for COCO20K. We again find that LiFT gives the best CorLoc boost over all baselines for both datasets.

E Additional Similarity Map Samples

We have found that the feature self-similarity maps for DINO+LiFT more clearly and sharply outline the central object in an image. To further highlight this, we provide a zoomed-in comparison of the difference between DINO+LiFT and DINO+Bilinear upscaling in Figure 2. We can see that DINO with Bilinear upsampling highlights the main object, but the outline is hazier and less precise due to the smoothing of the features. Meanwhile, the upscaled feature map produced by LiFT better respects object contours and produces a much sharper feature self-similarity map. Finally, we provide additional samples further showing the benefits of LiFT for self-similarity maps, as shown in Figure 3. In rows 1 to 8 (left), we show samples with single central objects of differing shapes and sizes. We see that the feature self-similarity maps for DINO+LiFT more uniformly fill the foreground object region, and have less noisy correlations with background regions. In rows 1 to 3 (right), we show samples where the central feature vector, shown by the red marker, lies on a background region. In these cases, we still see sharp contours around the foreground objects, or around the body of water in row 1 (right). In cases like rows 4 to 8 (right), when there are multiple overlapping instances of the same object class, we see a uniform highlighting of the multiple object instances. We also see that DINO+LiFT better highlights thin structures in objects, like the teapot handle and tripod legs in rows 7 and 8 (left). For comparison, when DINO (without LiFT) is given a doubled input size, these details are sometimes lost to noisy background regions.

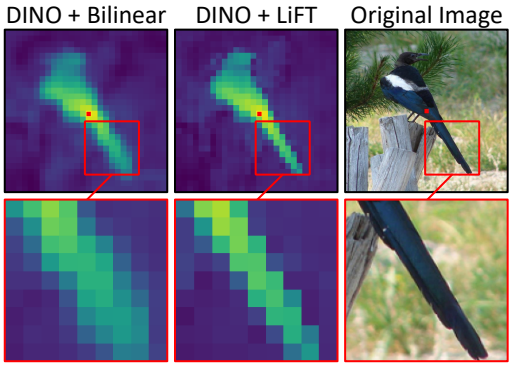

Fig. 2: Compared with DINO+Bilinear, DINO+LiFT gives feature self-similarity maps with much sharper object boundaries, especially when zoomed in.

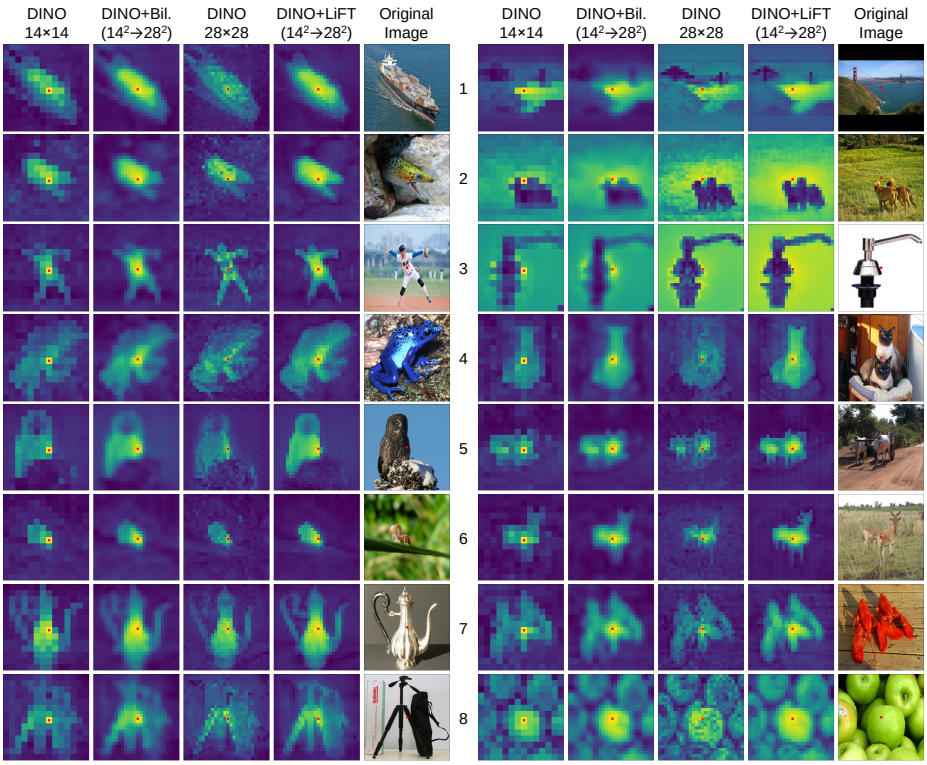

Fig. 3: Additional visualizations of the self-similarity of features extracted from DINO, DINO+Bilinear interpolation, DINO with higher resolution image, and DINO+LiFT. The input image is shown for comparison. The self-similarity is computed using the feature corresponding to the center of the grid (marked in red) and all other features from each spatial location. Brighter pixels show a higher similarity.

Table 5: Unsupervised Object Discovery comparison on PASCAL VOC 2007, PASCAL VOC 2012, and COCO20K. We report results for the CorLoc metric.

Dataset	Method	Resolution			
		56	112	224	448
VOC07	DINO	20.74	50.07	65.60	68.27
	Leopart	18.92	32.59	51.59	48.79
	SelfPatch	18.04	41.99	62.40	63.62
	DINO+BL	21.27	46.96	64.70	68.37
	DINO+RC	28.96	55.00	66.85	68.87
	DINO+JBU	26.16	56.60	66.75	69.03
	DINO+LiFT	**36.54**	**62.02**	**68.79**	**69.65**
VOC12	DINO	23.27	55.33	69.01	71.64
	Leopart	22.44	37.41	55.74	54.40
	SelfPatch	20.19	47.32	68.02	66.48
	DINO+BL	23.46	52.64	68.53	71.55
	DINO+RC	31.63	59.96	68.87	71.47
	DINO+JBU	28.97	61.67	69.13	71.54
	DINO+LiFT	**40.56**	**66.21**	**70.91**	**71.71**
COCO20K	DINO	16.28	40.08	53.98	57.99
	Leopart	16.14	26.78	43.89	44.08
	SelfPatch	14.15	35.76	52.18	55.47
	DINO+BL	17.78	35.62	51.53	56.84
	DINO+RC	22.92	42.53	54.52	58.40
	DINO+JBU	21.36	43.87	55.45	58.82
	DINO+LiFT	**27.72**	**50.20**	**58.03**	**60.50**

References

1. Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., Joulin, A.: Emerging properties in self-supervised vision transformers. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 9650–9660 (2021)
2. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html
3. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html
4. He, K., Fan, H., Wu, Y., Xie, S., Girshick, R.: Momentum contrast for unsupervised visual representation learning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 9729–9738 (2020)
5. Li, Y., Mao, H., Girshick, R., He, K.: Exploring plain vision transformer backbones for object detection. In: Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part IX. pp. 280–296. Springer (2022)
6. Ziegler, A., Asano, Y.M.: Self-supervised learning of object parts for semantic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 14502–14511 (2022)