LiFT: A Surprisingly Simple Lightweight Feature Transform for Dense ViT Descriptors

Saksham Suri*   Matthew Walmer*   Kamal Gupta   Abhinav Shrivastava

University of Maryland, College Park

Abstract. We present a simple self-supervised method to enhance the performance of ViT features for dense downstream tasks. Our Lightweight Feature Transform (LiFT) is a straightforward and compact postprocessing network that can be applied to enhance the features of any pre-trained ViT backbone. LiFT is fast and easy to train with a self-supervised objective, and it boosts the density of ViT features for minimal extra inference cost. Furthermore, we demonstrate that LiFT can be applied with approaches that use additional task-specific downstream modules, as we integrate LiFT with ViTDet for COCO detection and segmentation. Despite the simplicity of LiFT, we find that it is not simply learning a more complex version of bilinear interpolation. Instead, our LiFT training protocol leads to several desirable emergent properties that benefit ViT features in dense downstream tasks. This includes greater scale invariance for features, and better object boundary maps. By simply training LiFT for a few epochs, we show improved performance on keypoint correspondence, detection, segmentation, and object discovery tasks. Overall, LiFT provides an easy way to unlock the benefits of denser feature arrays for a fraction of the computational cost. For more details, refer to our project page.

Keywords: Self-supervised Learning · ViTs · Feature Densification

1 Introduction

In recent years, Vision Transformers (ViTs) [18] have emerged as preferred architectures for many image and video recognition tasks in the Computer Vision community. They also represent a major design shift compared with the well-explored Convolutional Neural Networks (CNNs). ViTs typically convert images into a very coarse grid of image patches (or tokens) before applying transformer layers. This allows ViTs to learn increasingly powerful patch-wise representations in successive layers [54]. The expressive power of ViTs stems from their wide receptive field throughout all layers made possible by multi-headed self-attention operations [52]. The downside of this design is that despite being able to learn powerful representations, ViTs often lack spatial granularity in their features due to the low resolution of the token/patch grid. This hinders their off-the-shelf application to dense and local tasks such as object detection, segmentation, and
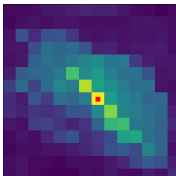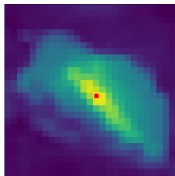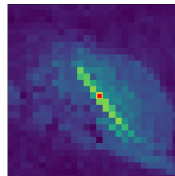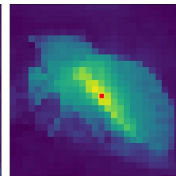
---

* Equal contributors.
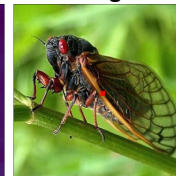
Baseline Model	Double Resolution	Increase Backbone Size	Apply **LiFT** (Ours)
DINO ViT-S/16 (224 x 224) Parameters: 21M FLOPs: 4.34G KP Performance: 24.76	**DINO ViT-S/16** (448 x 448) Parameters: 21M (+0%) FLOPs: 17.28 (+298%) KP Performance: 28.60 (+15.5%)	**DINO ViT-B/16** (224 x 224) Parameters: 85M (+304%) FLOPs: 17.21G (+296%) KP Performance: 24.90 (+0.6%)	**DINO ViT-S/16 + LiFT** (224 x 224) Parameters: 22.2M (+5.7%) FLOPs: 5.30G (+22.1%) KP Performance: 28.68 (+15.8%)

DINO 14×14	**DINO + Bilinear** 14×14 → 28×28	**DINO** 28×28	**DINO + LiFT** 14×14 → 28×28	**Original** Image

Fig. 1: (Top) Increasing the backbone size or doubling the input resolution can boost the effectiveness of self-supervised ViT features for dense tasks like keypoint (KP) correspondence. However, both of these options come at a significant cost in terms of parameter count, inference cost, or both. We present **LiFT**, a surprisingly simple **Li**ghtweight **F**eature **T**ransform that unlocks the benefits of dense self-supervised ViT representations for minimal extra cost. **(Bottom)** LiFT also has useful emergent properties, such as yielding cleaner object boundaries in feature similarity maps.

keypoint correspondence. Increasing the feature resolution of a ViT directly using a larger image size or smaller patch size leads to an increased number of patches. Self-attention, being a quadratic operation, grows in memory consumption as $\mathcal{O}(N^2)$ where N is the number of patches in the image. Prior works have proposed alterations to the ViT architecture to make it better suited for dense tasks, but their methods either involve expensive carefully designed training, task-specific loss functions or heuristics, or high inference costs [2,61,64].

In this work, we propose a simple **Li**ghtweight **F**eature **T**ransform or **LiFT** to generate dense ViT features that provide significant performance gains in downstream tasks such as detection, segmentation, keypoint correspondence, and object discovery. LiFT can unlock the benefits of dense feature representations for a fraction of the computational cost compared with other approaches. As illustrated in Figure 2, our proposed method fuses the coarse high-level information of ViT features with convolution-based image features derived from the original image to generate higher-density feature maps without incurring the high computational cost of extra tokens. We show that this approach does not require any complex training recipe and, once trained on a general purpose dataset, generalizes well to multiple downstream tasks. Our approach can be trained with a simple self-supervised loss and generalizes to input image resolutions not seen during training. LiFT can be readily plugged on top of any ViT backbone to enhance its features, and it can also be integrated into pipelines that use additional task-specific downstream modules, like the Mask-RCNN head [29] used by ViTDet [33]. Additionally, we show that we can apply LiFT in a recursive manner to increase feature resolution even further.

We demonstrate the effectiveness of LiFT quantitatively on 'local' tasks, which require features computed at precise locations, as well as on 'dense' tasks, which require features computed for the entire image. Specifically, we present results for LiFT applied to SPair-71k Keypoint Correspondence [40], COCO Detection and Segmentation [34], DAVIS Video Segmentation [42], and Unsupervised Object Discovery on Pascal VOC 2007 [20], Pascal VOC 2012 [21] and COCO20K [35]. For all of these tasks, LiFT is able to meet or exceed the performance of prior works for a fraction of the computational cost. As an example, in Figure 1 we compare three options for boosting performance in SPair-71k Keypoint Correspondence. LiFT provides a significant performance gain while increasing the total parameter count of the network by a mere 5.7%. This is compared to the 304% parameter count increase incurred by the step up from ViT-S/16 to ViT-B/16. Increasing the input resolution is a trivially easy way to boost the feature density, and it also gives improved performance. However, it increases the total inference FLOPs by almost 300% while LiFT only increases the cost by 22.1%, giving a far superior compute cost *vs.* performance trade off.

Despite the simplicity of our LiFT approach, we show that it is not just learning a more complex version of bilinear upsampling. Instead, we demonstrate that LiFT has several desirable emergent properties that enhance ViT features to make them better suited for dense tasks. We find that LiFT improves the scale invariance of ViT features, as measured using Centered Kernel Alignment (CKA) [13, 32]. We also qualitatively show that LiFT yields better object boundary maps when computing feature similarity maps. Overall, LiFT represents an orthogonal avenue of improvement compared to prior dense ViT feature extraction strategies, and it can be combined with past methods to further advance self-supervised performance on dense prediction tasks. In summary, our contributions are as follows:

– We propose **LiFT**, a **Li**ghtweight **F**eature **T**ransform that boosts the performance of existing ViT features on dense and local downstream tasks using a simple, quick training and inference strategy.
– We show that LiFT boosts the performance of self-supervised ViT features for detection, segmentation, keypoint correspondence, and object discovery tasks.
– We demonstrate the adaptability of LiFT for any ViT backbone by showing improvements with DINO [6], MoCo [9] and Supervised ViT features. Additionally, LiFT even works on image resolutions not used during training.
– We show that LiFT features have desirable emergent properties like improved scale invariance and better feature alignment with object boundaries.

2   Related Work

2.1   Vision Transformers

Recently, Vision Transformers (ViTs) [18] have gained wide popularity as general-purpose models for multiple computer vision tasks such as image classification [1, 18, 51], object detection [33, 36, 63], segmentation [44, 48], video classification [3, 4], and more. Many of these methods adapt ViTs to different tasks by

making suitable changes to the output heads [33,60]. There have also been multiple variants of ViT like Swin [36], MViT [22], and PVT [56] which incorporate hierarchy and multiscale learning. Additionally, there are some works which try to bridge the gap between transformers and CNNs by incorporating convolutions into ViT architectures [19,59]. In this work, we focus on improving traditional or "Plain" ViT backbones, as they are the most general and widely adopted form of ViT. These models learn powerful representations, but they suffer in terms of feature resolution, an issue which we aim to address with LiFT. Many self-supervised ViT pretraining methods have been proposed to learn powerful off-the-shelf features, such as DINO [6], MoCo [8,10,28], and MAE [27]. Prior works [25,46,54] have shown that different pretraining leads to significant differences in the properties of the learned features. We show that LiFT can improve the quality and usefulness of ViT features for a range of pretraining methods.

2.2   Extracting Denser Features from ViTs

Works like [2] and [54] show the general benefits of denser feature maps for local tasks, and multiple strategies have been proposed to extract denser feature arrays from pretrained ViTs. ViT-Adapter [11] applies ViTs to dense tasks through the use of finetunable side-networks for feature pyramid extraction. Like ViT-Adapter, LiFT aims to enhance "Plain" ViT features for dense tasks, however, the ViT-Adapter method is not task-agnostic, and it is trained in a fully supervised way with full detection and segmentation labels for the downstream dataset. In contrast, LiFT is a task-agnostic general enhancement for ViT features, and it is trained with a completely self-supervised objective. LiFT is also faster and easier to train, as it does not require passing gradients through the ViT backbone, and it is also $\sim 4.8 \times$ smaller than the similar ViT-Adapter. [2] proposes a simpler technique by reducing the stride during initial image patch extraction. This does not require any training, but also becomes computationally expensive as the number of tokens increase quadratically requiring more GPU memory and FLOPs. Our LiFT approach can be thought of as a "shortcut" to achieve the benefits of denser features for a fraction of the cost of increased tokens.

2.3   Feature Upsampling

Over the years, many works have been proposed to convert lower resolution features into higher resolution ones. This includes basic methods like bilinear interpolation, and Joint Bilinear Upsampling (JBU) [31], which leverages high-frequency information from the input image to improve bilinear interpolation. Our LiFT module leverages the input image similarly. Resize-convolutions [41] are proposed as an alternative to deconvolution. Many more sophisticated upsampling operators have been proposed [15,37–39,55], however these methods are designed to be trained in the context of their particular architecture and downstream task, unlike our LiFT module, which is stand-alone and task-agnostic. Moreover, many of these methods are designed to be integrated with convolutional encoder/decoder architectures, and thus are not easily applied to ViT

backbones. [50] proposes a GAN-based approach for CNN feature densification which requires careful training and a mixture of adversarial and focal loss. In comparison, our LiFT approach is easy to train through a simple self-supervised objective. Other works, like [7] and [62], have applied student/teacher distillation with feature super-resolution to improve CNN classification performance on low-resolution images. While we also perform feature super-resolution, the aim of our work is not to distill a student network, but rather we aim to generate densified features while avoiding finetuning the ViT backbone.

Finally, we would like to acknowledge a concurrently published work, FeatUp [23], which presents two feature upsampling methods, one based on JBU and a second based on implicit network learning. Like LiFT, FeatUp trains general-purpose stand-alone feature upsamplers for multiple backbones and downstream tasks, though its learning process is more complex, as it requires training an additional downsampler module which is later discarded. LiFT and FeatUp$_{(JBU)}$ are both feedforward, making them easy to apply to any image, though we believe LiFT's feature maps are sharper and better matched to the object boundaries than the JBU-enhanced features. FeatUp$_{(Implicit)}$ produces sharp, impressive feature maps, though compared with LiFT and FeatUp$_{(JBU)}$, it is a much less scalable method, as it trains a new implicit network for every input image. Overall, we believe LiFT captures the best properties of both FeatUp variations.

2.4   Finetuning ViTs for Dense Tasks

Another line of research has focused on finetuning self-supervised ViT backbones to make their features better suited to dense, locally-focused tasks. SelfPatch [64] and Leopart [61] both use pretrained DINO models as a starting point and improve their patch-level representations using dense self-supervised tasks. SelfPatch learns by enforcing similarity to neighboring patch features, and Leopart uses spatially dense clustering to enforce similarity within clusters to learn better part representations. These approaches finetune the full backbone with specialized training strategies and losses, which can still be expensive to train and not easily extendable to other backbones. In contrast, our approach does not require backbone finetuning at all, and instead trains a lightweight post-processing module. LiFT can be easily trained on ImageNet [16] in a self-supervised manner in very few epochs and afterwards it can generalize to multiple downstream tasks.

3   Method

3.1   ViT Background

Consider a ViT that takes an image with dimensions $H \times W \times 3$ as input and outputs feature descriptors at the resolution $\frac{H}{P} \times \frac{W}{P}$, where P is the patch size. P is usually 8 or 16 depending on the ViT variant. We typically assume that the patch-extraction stride length, S, is equal to P, though this can be altered, as proposed by [2]. For now we will assume that $S = P$. Our goal is to transform the
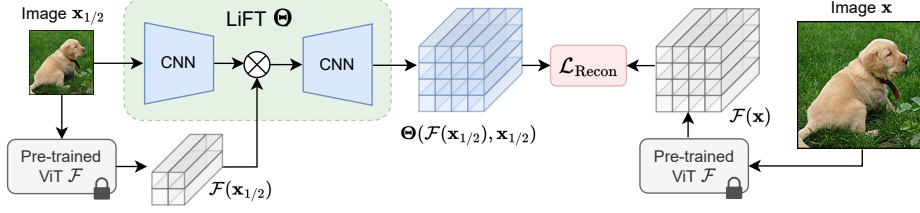
Fig. 2: Illustration of **LiFT**, our proposed **Li**ghtweight **F**eature **T**ransform for generating dense ViT descriptors. The frozen ViT backbone is used to extract features for both low- and high-resolution images. The low-resolution image and its corresponding features are passed through LiFT, which generates a dense version of the features. The LiFT Block first encodes fine-resolution image features using a small CNN. It then combines the CNN features with the ViT features at multiple phases in an upsampling CNN, which outputs dense features. The LiFT block is trained using a self-supervised reconstruction error with the corresponding high-resolution features.

coarse, low-resolution features of a pretrained ViT into dense feature descriptors without having to re-train or finetune the ViT. A naïve way to achieve this transformation could be to scale up the input image to $CH \times CW$ resolution to achieve a feature grid that is C times larger along both dimensions. This approach can result in a significant increase in memory consumption and can be prohibitively expensive since the memory of the ViT scales with $\mathcal{O}(H^2W^2)$. Another option could be to upscale the features directly by a factor of C using bilinear-interpolation. This approach computes sub-optimal pixel-level features as it simply bilinearly redistributes the features between the centers of patches. Such an approach fails to take advantage of the information that is readily available in the original image space. LiFT takes advantage of this information.

3.2   Lightweight Feature Transform (LiFT)

Our proposed approach, LiFT, builds on the hypothesis that, even though the ViT feature descriptors have a low spatial resolution, their high dimensionality allows them to store rich information about the image structure. This hypothesis is further supported by works on internal learning of images [26,47,65]. However, unlike internal learning, we propose to train a general-purpose, lightweight upsampling network using only self-supervision to double the resolution of feature descriptors obtained from a frozen, pretrained ViT. Furthermore, we can gain additional fine-level information directly from the original image at the same resolution used to generate the ViT features. We fuse these two information sources to create our final LiFT module, as illustrated in Figure 2. Given ViT descriptors of size $\frac{H}{P} \times \frac{W}{P}$, a single LiFT expansion block scales them to $\frac{2H}{P} \times \frac{2W}{P}$ in a single forward pass. Our LiFT block is built following a U-Net-style structure [45] with skip connections, where semantically rich but coarse ViT features are combined with shallow but dense image features derived from a second input with the original image. The image input is processed through a series of convolution blocks

and the resulting features are concatenated to the ViT features. Then, a single transpose convolution block is applied to generate the upscaled semantically rich features. Thanks to its fully convolutional nature, the LiFT block can be applied to any image size. One can even apply a LiFT block multiple times to further upscale the features, as shown in Section 6.

3.3   Training Objective

Given an image $\mathbf{x} \in \mathbb{R}^{H \times W \times 3}$ and a pretrained, frozen ViT model $\mathcal{F}$ of stride P, we extract last-layer features $\mathcal{F}(\mathbf{x}) \in \mathbb{R}^{\frac{H}{P} \times \frac{W}{P} \times D}$, where D is the feature dimension. The LiFT block $\boldsymbol{\Theta}$ upscales the features from $\frac{H}{P} \times \frac{W}{P}$ to $\frac{2H}{P} \times \frac{2W}{P}$. For training, we propose the following multi-scale reconstruction objective:

$$\mathcal{L}_{\text{Recon}} = \quad d\left(\mathcal{F}(\mathbf{x}), \boldsymbol{\Theta}(\mathcal{F}(\mathbf{x}_{1/2}), \mathbf{x}_{1/2}\right) + d\left(\mathcal{F}(\mathbf{x}_{1/2}), \boldsymbol{\Theta}(\mathcal{F}(\mathbf{x}_{1/4}), \mathbf{x}_{1/4}\right) \quad (1)$$

Where $\mathbf{x}_{1/2}$ and $\mathbf{x}_{1/4}$ are the images resized to 1/2 and 1/4 of the original image resolutions, and d is a distance function. For d we select the cosine distance metric, as it inherently normalizes the output and empirically achieves better performance. When processing $\mathbf{x}_{1/2}$ and $\mathbf{x}_{1/4}$, we follow the method of [6] to handle positional embeddings for images of different sizes. We optimize the parameters of the LiFT module $\boldsymbol{\Theta}$ to minimize Equation 1.

3.4   Training Details

Training LiFT is fast and efficient, as it is lightweight and does not require propagating gradients through the ViT backbone. The LiFT module, $\boldsymbol{\Theta}$, is a small network with $1.2M$ trainable parameters. We train LiFT for 5 epochs on the ImageNet dataset on a single GPU. We use a learning rate of 0.001 with a batch size of 256 for stride 16 training. We use DINO [6] ViT-S/16 as our base ViT, and we apply color jitter as the only augmentation. In total, training takes only ~8 hours on one RTX A6000 GPU. Once our LiFT module is trained, it is a general purpose feature enhancement module that can be directly applied to a range of downstream tasks without need for any further finetuning. We demonstrate the flexibility of LiFT by applying features from the same DINO+LiFT model to several tasks in Section 4. We present additional analysis of different design choices for training and inference in Appendix A.

3.5   Using LiFT with Downstream Modules

Our LiFT module can be easily applied to downstream tasks that directly operate on the output image features. However, LiFT can also be applied in circumstances where additional downstream modules follow the feature extracting backbone. To demonstrate this, we show how to apply LiFT to the ViTDet architecture [33], which combines a ViT backbone with a detector head to perform COCO Object Detection and Segmentation. To achieve this, we inject the LiFT module after the backbone and before the head in a pretrained ViTDet

Table 1: Comparison between LiFT and other baselines on the Keypoint Correspondence task on SPair-71k. We report PCK@0.1/0.05/0.01 at multiple input resolutions.

Method/Res.	PCK@0.1				PCK@0.05				PCK@0.01			
	56	112	224	448	56	112	224	448	56	112	224	448
DINO	2.04	12.67	24.76	28.6	0.51	3.61	9.54	15.33	0.01	0.2	0.54	1.4
Leopart	2.35	11.2	23.33	26.54	0.6	3.22	8.9	12.26	0.05	0.1	0.47	0.79
SelfPatch	2.13	12.18	23.03	27.34	0.44	3.61	9.32	14.44	0.02	0.17	0.42	1.12
DINO+BL	3.81	13.71	26.72	30.48	1.01	4.23	11.37	16.75	0.02	0.14	0.5	1.64
DINO+RC	3.87	14.02	26.09	30.21	0.94	4.41	11.51	16.76	0.03	0.16	0.51	1.7
DINO+JBU	3.08	12.26	24.87	29.11	0.82	3.82	10.6	16.01	0.03	0.17	0.47	1.42
DINO+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**

model. We train this LiFT module on COCO training data using the same self-supervised objective described in Section 3.3. We then briefly finetune the pretrained head on the LiFT features. We present these results in Section 4, and we provide additional details of our ViTDet+LiFT architecture in Appendix B.

3.6   Baseline Methods

In addition to the base DINO features, we compare to three types of baselines. The first are baselines which finetune DINO features to enhance them without increasing their density. This includes Leopart [64] and SelfPatch [61]. The second are basic feature upsampling methods, which includes bilinear interpolation (BL), resize-convolutions (RC), and Joint Bilinear Upsampling (JBU). For these methods, we double the feature resolution to match LiFT. We use the generalized JBU implementation of [23], which includes learnable parameters. We train both JBU and RC following the same optimization protocol as our LiFT module. The final type are dense ViT feature extraction methods, specifically increased image resolution and reduced stride [2]. Note that LiFT represents an orthogonal direction of ViT improvement that can be used in combination with methods like [2] and [64], which we show in the following sections and in Appendix C.

4   Performance Benefits of LiFT

SPair Keypoint Correspondence. This task involves matching keypoints between pairs of images in the SPair-71k dataset. We follow the evaluation protocol of Amir et al. [2] and report Percentage of Correct Keypoints (PCK) with three different distance thresholds. We extract dense features using the frozen DINO+LiFT combination trained in Section 3.4. For this task, the features of all methods are resized to match the original image resolution using Lanczos interpolation before feature matching begins. Table 1 presents the results. Compared to both the base DINO model and other baselines, LiFT performs the best across all resolutions for all metrics. For lower resolutions like 56×56, LiFT more than doubles the performance of DINO. Also note that under the PCK@0.1 metric,

Table 2: DAVIS Video Object Segmentation results for the J & F Mean metric.

Method/Res.	56	112	224	448
DINO	7.4	17.5	33.0	50.9
Leopart	6.9	16.1	30.3	45.1
SelfPatch	7.4	17.2	33.0	51.4
DINO+BL	10.8	23.7	37.0	53.0
DINO+RC	11.0	24.0	37.4	53.2
DINO+JBU	11.2	26.2	39.0	54.5
DINO+LiFT	**13.0**	**28.0**	**44.3**	**61.1**

Table 3: COCO20K Unsupervised Object Discovery results for CorLoc metric.

Method/Res.	56	112	224	448
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**

Table 4: COCO Detection and Segmentation results with ViTDet+LiFT for two different heads: Mask R-CNN (MR) and Cascade R-CNN (CR).

Method	Detection			Segmentation		
	AP	AP_{50}	AP_{75}	AP	AP_{50}	AP_{75}
ViTDet(MR)	39.50	61.56	42.23	37.81	60.47	39.97
ViTDet(MR)+LiFT	**45.98**	**66.41**	**49.87**	**40.78**	**63.34**	**43.57**
ViTDet(CR)	45.65	64.39	48.90	40.93	62.86	44.28
ViTDet(CR)+LiFT	**47.92**	**65.88**	**51.67**	**41.42**	**63.21**	**44.45**

DINO+LiFT at 224×224 resolution beats DINO run at 448×448, even though both configurations produce final features of the same density.

DAVIS Video Object Segmentation. This task involves propagating a video object segmentation across multiple frames where the first frame ground truth segmentation mask is provided. This is achieved through dense feature matching between frames. Again, we extract dense features with the same pre-trained DINO+LiFT. We follow the evaluation protocol of [30] and, for brevity, we report results for the J & F mean metric in the main paper, but we can see consistent improvements across the J mean and F mean individually as shown in Appendix D. In Table 2, it can be seen that across all resolutions LiFT outperforms all other approaches. At the lowest resolution of 56×56, the performance gain over the base DINO is ~1.75×. On average, we improve by 9.4 points over base DINO.

Unsupervised Object Discovery. We test the benefits of LiFT for Unsupervised Single Object Discovery on COCO20K [35]. We apply TokenCut [57], which performs Graph Cut on features, to LiFT and the other baseline methods. Similar to prior works [12, 17, 43, 53, 57, 58] we report the Correct Localization (CorLoc) metric, which is computed as the fraction of the images in which at least one object box prediction has an IoU greater than a threshold (0.5) with a ground truth box. As shown in Table 3, LiFT gives a good boost in CorLoc, with gains across all resolutions and with LiFT outperforming the other approaches. It should be noted that the performance gains at lower resolutions are especially large. For example, at the 56×56 resolution, we see an improvement of 11.44

Table 5: FLOPs comparison for a single forward pass across resolutions and strides for DINO and DINO+LiFT. Along with FLOPS, we report the performance on both SPair-71k and DAVIS.

Fig. 3: Performance *vs.* Compute Cost trade-off curve for SPair-71k keypoint correspondence. For any given FLOP-budget, DINO+LiFT achieves far superior performance.

Method	Res.	Stride	FLOPs (G)	PCK@0.1	J&F mean
DINO		16	4.34	24.76	33.0
DINO+LiFT	224		5.30	28.68	44.3
DINO		8	16.07	29.92	43.9
DINO+LiFT			19.65	31.91	52.6
DINO		16	17.28	28.60	50.9
DINO+LiFT	448		21.12	31.38	61.1
DINO		8	66.60	31.92	61.9
DINO+LiFT			81.18	32.20	69.7
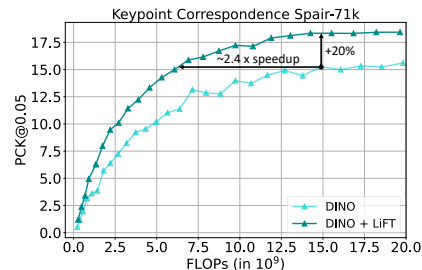
over DINO. In Appendix D, we present additional results on PASCAL VOC 2007 [20] and PASCAL VOC 2012 [21], and we observe similar overall trends.

COCO Detection and Segmentation. To further demonstrate the versatility of LiFT, we show how it can be applied to COCO Detection and Segmentation using a ViTDet+LiFT model as described in Section 3.5. We present results for two ViTDet variants using different heads: Mask R-CNN (MR) [29], and Cascade R-CNN (CR) [5]. In Table 4, ViTDet(MR)+LiFT gives a ∼6.5% boost for Detection AP and a ∼3% boost for Segmentation AP. Smaller but still consistent boosts are also achieved when combining LiFT with ViTDet(CR).

5   Computational Efficiency of LiFT

We have shown that feature densification with LiFT provides significant performance benefits in several tasks. We now show why LiFT is a *Lightweight* transform, as it is vastly more computationally efficient than other alternatives that increase the token density of the ViT backbone. A trivially easy way to boost the density of ViT features is to increase the resolution of the input image. Another option is the dense feature extraction strategy of [2] which increases the number of tokens in the network by reducing the stride during patch extraction. Both of these methods increase token density but at a high computational cost. We present a comprehensive compute cost *vs.* performance benefit analysis for LiFT against and in combination with these methods, and we show that LiFT acts as a "shortcut" to achieve higher resolution features for minimal extra compute.

LiFT, Resolution, and Stride. For this analysis, we take SPair Keypoint Correspondence and DAVIS Video Segmentation as the tasks, and we measure compute cost in FLOPs (G) against performance both with and without LiFT over a range of input resolutions and strides. As shown in rows 1 and 2 of Table 5,

Table 6: Comparison of parameters & FLOPs *vs.* performance at 224×224 resolution for Keypoint Correspondence. We report PCK@0.1 and PCK@0.05 on SPair-71k.

Method	Parameters	FLOPs (G)	PCK@0.1	PCK@0.05
DINO S/16	21M	4.34	24.76	9.54
DINO S/16+LiFT	22.2M	5.30	**28.68**	**14.72**
DINO B/16	85M	17.21	24.90	9.64

Table 7: LiFT when using different backbones for training, inference, or both. We report PCK@0.1 on SPair-71k.

Inference Model	Training Model			
	No LiFT	DINO	MoCo	ViT
DINO	28.6	**31.38**	16.02	20.71
MoCo	12.31	9.86	**16.34**	11.31
ViT	16.9	12.55	8.91	**18.69**

LiFT introduces a small increase in FLOPs from 4.34G to 5.30G (+22%) but also brings a significant performance improvement of 3.92 and 11.3 points on the PCK@0.1 and J & F Mean metrics respectively. For comparison, reducing the stride from 16 to 8 (row 1 *vs.* row 3) gives a similar level of improvement, but nearly quadruples the FLOPs from 4.34G to 16.07G (+270%). A similar trend of large improvements can be seen when comparing pairs of rows (3 *vs.* 4, 5 *vs.* 6) with DINO *vs.* DINO+LiFT. The best overall results are achieved by combining LiFT with increased resolution and reduced stride, but this also comes with the highest computation cost. It should also be noted that DINO+LiFT at 224×224 resolution and stride 8 (row 4) uses 3× less FLOPs than DINO at 448×448 resolution at stride 8 (row 7) while having similar PCK performance.

Performance Trade-Off Curve. We present a comprehensive Compute Cost *vs.* Performance trade-off curve for SPair Keypoint Correspondence in Figure 3. By incrementally increasing the input resolution, we can gradually increase both the performance and inference cost for DINO and DINO+LiFT features. We find that LiFT significantly outperforms the base DINO features at any given FLOP allowance, in most cases seeing a ∼20% performance gain. Alternatively, LiFT can be run at a lower input resolution to achieve equivalent performance at a fraction of the compute cost. For example, to achieve a score of 15.0 in PCK@0.05, DINO+LiFT only requires ∼6.25 Giga-FLOPs of compute power, while the standard DINO requires at least 15 Giga-FLOPs.

Parameter Count. We acknowlege that the addition of the LiFT module slightly increases the overall model size and parameter count as shown in Table 6. However, this addition is quite small and only represents a +5.7% change in total parameters. For comparison, the jump from ViT-S to ViT-B results in a +304% increase in parameters. Furthermore, for dense tasks like SPair Keypoint Correspondence, we find that the performance benefits provided by LiFT far exceed the benefits of a larger backbone. Note that the methods Leopart and SelfPatch do not increase the parameter count of the ViT, as they finetune the DINO backbone instead of introducing new modules. We believe the major performance benefits of LiFT justify the small extra costs in Parameter Count and Inference FLOPs for a given resolution. And, as shown in the previous section, for any fixed FLOP-budget, DINO+LiFT achieves far superior performance.
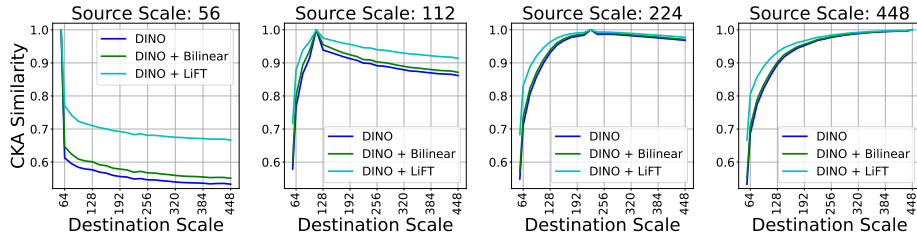
Fig. 4: CKA Similarity of ViT features extracted from SPair-71k images at different input image sizes, denoted by Source Scale and Destination Scale. LiFT produces features that are more scale-invariant, especially for smaller scale inputs and objects.

6   Properties of LiFT

LiFT and Scale Invariance of Features. In this experiment, we demonstrate that LiFT intrinsically learns to generate features that are more scale-invariant. We use the Centered Kernel Alignment (CKA) metric [14, 32, 49], which can measure the similarity of a pair of feature maps even when they are different sizes. Using images in SPair-71k, we re-scale each image to a range of different sizes and then extract features with DINO or DINO+LiFT and measure the CKA similarity between all input size pairings. As a baseline, we also compare with bilinearly upsampled DINO features. In Figure 4, we take four source scales and plot the CKA similarity with the features at all other scales. We see that LiFT greatly improves the inter-scale feature similarity for small input scales, and moderately improves the similarity for medium and larger scales. This shows that LiFT produces representations that are more scale-invariant than the base DINO features. Bilinear upsampling does provide a small improvement in CKA similarity across scales, though the benefit is far smaller. This scale invariance property of LiFT is learned automatically, and likely comes from its multi-scale reconstruction objective. LiFT must learn to counteract the effect of input scale to map features from low resolution inputs to those of high resolution inputs. This property is desirable for dense tasks where objects appear at different scales.

Enhanced Self-Similarity Maps with LiFT. To further improve our understanding of the dense features generated by LiFT, we visualize the self-similarity of the features in Figure 5. Base DINO S/16 yields 14×14 features for a 224×224 image. In our comparisons, we visualize these features alongside: a bilinearly interpolated upsampling of these same features to 28×28 (DINO+Bilinear), DINO features for a 448×448 input image, and finally DINO+LiFT features generated for a 224×224 input image. The last three configurations all yield 28×28 feature grids. To visualize these features, we select the center-most token feature for each of the maps and compute its similarity with all other features and visualize the similarity scores. We show six feature similarity maps in Figure 5, and additional sample visualizations can be found in Appendix E. We find that, qualitatively,
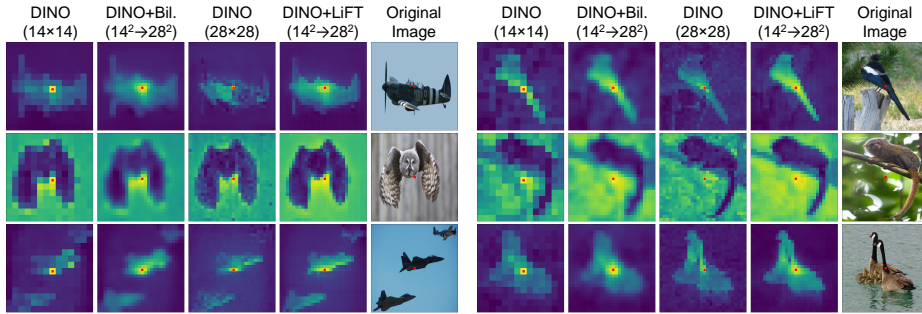
Fig. 5: Visualization of the self-similarity of features for DINO, DINO+Bilinear interpolation, DINO with higher resolution image, and DINO+LiFT. To generate this visualization, the self-similarity is computed using the feature corresponding to the center of the grid (marked in red) and all other features from each spatial location. Brighter map shows a higher similarity. Best viewed digitally in color.

the LiFT output gives a cleaner boundary and better highlights the content corresponding to the central patch. Having a high similarity to relevant regions and a clear boundary is beneficial for multiple localized downstream tasks. It also indicates that the DINO+LiFT features have better spatial awareness of object boundaries. Note that for row 2, where the central pixel corresponds to the background region, the similarity map highlights the background in the image, though it still appears that DINO+LiFT produces better similarity maps with sharper edges. In row 3, when there are multiple objects of the same type, DINO+LiFT better highlights the separate object instances. These results are qualitative, but they suggest that the LiFT-enhanced features have better content and boundary information than the base DINO features, which likely contributes to their improved performance in correspondence and segmentation tasks.

Variations in Backbone. One of the requirements of a feature densifying approach is that it should be easy to train on any backbone. To this end, we show that our approach consistently gives a performance gain with multiple different backbones: DINO (Table 8 row 1 *vs*. row 2), MoCo (Table 8 row 4 *vs*. row 5) and a fully-supervised ViT (Table 8 row 7 *vs*. row 8). This consistent improvement shows that LiFT can be trained in the exact same manner on differently trained ViTs without need for careful hyperparameter tuning. To verify that LiFT does not simply learn a bilinear interpolation, we apply LiFT modules trained on one backbone to the output of a different backbone. We show these results in Table 7 using the PCK@0.1 metric and 448×448 images. It can be seen that when LiFT is applied to a different model than what was used to train it, the performance drops and is lower than not applying LiFT. This shows that LiFT learns a model-specific feature-densifying transform and not a simple interpolation.

Repeated Application of the LiFT Module. In our base approach, the LiFT module is applied once to the ViT features to double their resolution. In this section, we also test the potential benefits of applying the LiFT block

Table 8: Performance for LiFT with different backbones and for repeated application of LiFT. We report PCK@0.1 on SPair-71k.

Fig. 6: With no extra training, repeated application of LiFT produces pixel-dense feature maps.

Row	Backbone	Method	Resolution			
			56	112	224	448
1		-	2.04	12.67	24.76	28.6
2	DINO	LiFT	5.05	17.72	28.68	**31.38**
3		2×LiFT	**7.42**	**20.12**	**29.45**	31.35
4		-	1.27	3.43	7.37	12.31
5	MOCO	LiFT	6.48	10.51	14.13	16.34
6		2×LiFT	**8.72**	**13.21**	**16.12**	**17.08**
7		-	1.26	5.72	13.23	16.9
8	ViT	LiFT	3.76	9.21	**16.5**8	**18.69**
9		2×LiFT	**5.17**	**9.89**	16.49	18.18

DINO   DINO+LiFT(4x)   Original Image
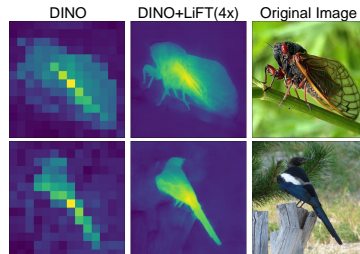
multiple times. To check this, we take the LiFT module and super-resolute the features twice by passing the output of first super-resolution again through the same LiFT network. We denote this approach as '2×LiFT' in Table 8. We can see that in most cases the performance increases after recursively applying LiFT. This is especially true for the 56×56, 112×112, and 224×224 resolutions. For 448×448, there are a few places where it negligibly drops performance, but it still shows improvement for most cases. Repeated application of LiFT can also be used to create detailed, pixel-dense feature self-similarity maps, as shown in Figure 6 for a 4× repeated application of the same pretrained LiFT module. Such maps could be useful for tasks like Unsupervised Semantic Segmentation [24].

7   Conclusion and Discussion

We have presented **LiFT**, a simple yet effective self-supervised **Li**ghtweight **F**eature **T**ransform to boost the density of features of pretrained ViT backbones. This approach allows us to extract higher resolution spatial features from ViTs which can then be used for multiple dense downstream tasks. LiFT is task-agnostic and gives significant boosts in SPair Keypoint Correspondence, DAVIS Video Object Segmentation, Unsupervised Object Discovery, and COCO Detection and Segmentation. This benefit comes for a fraction of the computational cost compared with other densification methods. LiFT is a lightweight module that is easily trained with a self-supervised objective, and it is far cheaper than full backbone finetuning, as is done by other prior works. Through extensive experiments, we have shown that LiFT can be trained easily on any backbone and consistently leads to improved performance by generating better quality, higher-density features. We also show that LiFT can be applied to its own output in a recursive manner enabling good performance with even lower image resolutions. Finally, we show that our surprisingly simple method has several desirable emergent properties including scale-invariant features, and better object boundary maps. This makes LiFT a useful multipurpose tool for many potential downstream applications.

References

1. Ali, A., Touvron, H., Caron, M., Bojanowski, P., Douze, M., Joulin, A., Laptev, I., Neverova, N., Synnaeve, G., Verbeek, J., et al.: Xcit: Cross-covariance image transformers. Advances in neural information processing systems **34**, 20014–20027 (2021)
2. Amir, S., Gandelsman, Y., Bagon, S., Dekel, T.: Deep vit features as dense visual descriptors. arXiv preprint arXiv:2112.05814 (2021)
3. Arnab, A., Dehghani, M., Heigold, G., Sun, C., Lučić, M., Schmid, C.: Vivit: A video vision transformer. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 6836–6846 (2021)
4. Bertasius, G., Wang, H., Torresani, L.: Is space-time attention all you need for video understanding? In: ICML. vol. 2, p. 4 (2021)
5. Cai, Z., Vasconcelos, N.: Cascade r-cnn: Delving into high quality object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 6154–6162 (2018)
6. 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)
7. Chen, H., Pei, Y., Zhao, H., Huang, Y.: Super-resolution guided knowledge distillation for low-resolution image classification. Pattern Recognition Letters **155**, 62–68 (2022)
8. Chen, X., Fan, H., Girshick, R., He, K.: Improved baselines with momentum contrastive learning. arXiv preprint arXiv:2003.04297 (2020)
9. Chen*, X., Xie*, S., He, K.: An empirical study of training self-supervised vision transformers. arXiv preprint arXiv:2104.02057 (2021)
10. Chen, X., Xie, S., He, K.: An empirical study of training self-supervised vision transformers. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 9640–9649 (2021)
11. Chen, Z., Duan, Y., Wang, W., He, J., Lu, T., Dai, J., Qiao, Y.: Vision transformer adapter for dense predictions. In: The Eleventh International Conference on Learning Representations (2022)
12. Cho, M., Kwak, S., Schmid, C., Ponce, J.: Unsupervised object discovery and localization in the wild: Part-based matching with bottom-up region proposals. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1201–1210 (2015)

13. Cortes, C., Mohri, M., Rostamizadeh, A.: Algorithms for learning kernels based on centered alignment. ArXiv **abs/1203.0550** (2012), `https://api.semanticscholar.org/CorpusID:9137763`
14. Cortes, C., Mohri, M., Rostamizadeh, A.: Algorithms for learning kernels based on centered alignment. The Journal of Machine Learning Research **13**(1), 795–828 (2012)
15. Dai, Y., Lu, H., Shen, C.: Learning affinity-aware upsampling for deep image matting. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6841–6850 (2021)
16. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255. Ieee (2009)
17. Deselaers, T., Alexe, B., Ferrari, V.: Localizing objects while learning their appearance. In: Computer Vision–ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part IV 11. pp. 452–466. Springer (2010)
18. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N.: An image is worth 16x16 words: Transformers for image recognition at scale. ICLR (2021)
19. d'Ascoli, S., Touvron, H., Leavitt, M.L., Morcos, A.S., Biroli, G., Sagun, L.: Convit: Improving vision transformers with soft convolutional inductive biases. In: International Conference on Machine Learning. pp. 2286–2296. PMLR (2021)
20. 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
21. 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
22. Fan, H., Xiong, B., Mangalam, K., Li, Y., Yan, Z., Malik, J., Feichtenhofer, C.: Multiscale vision transformers. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 6824–6835 (2021)
23. Fu, S., Hamilton, M., Brandt, L.E., Feldmann, A., Zhang, Z., Freeman, W.T.: Featup: A model-agnostic framework for features at any resolution. In: The Twelfth International Conference on Learning Representations
24. Gao, S., Li, Z.Y., Yang, M.H., Cheng, M.M., Han, J., Torr, P.: Large-scale unsupervised semantic segmentation (2022)
25. Ghiasi, A., Kazemi, H., Borgnia, E., Reich, S., Shu, M., Goldblum, M., Wilson, A.G., Goldstein, T.: What do vision transformers learn? a visual exploration. arXiv preprint arXiv:2212.06727 (2022)
26. Glasner, D., Bagon, S., Irani, M.: Super-resolution from a single image. In: 2009 IEEE 12th international conference on computer vision. pp. 349–356. IEEE (2009)
27. He, K., Chen, X., Xie, S., Li, Y., Dollár, P., Girshick, R.: Masked autoencoders are scalable vision learners. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 16000–16009 (2022)
28. 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)
29. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: Proceedings of the IEEE international conference on computer vision. pp. 2961–2969 (2017)

30. Jabri, A., Owens, A., Efros, A.: Space-time correspondence as a contrastive random walk. Advances in neural information processing systems **33**, 19545–19560 (2020)
31. Kopf, J., Cohen, M.F., Lischinski, D., Uyttendaele, M.: Joint bilateral upsampling. ACM Transactions on Graphics (ToG) **26**(3), 96–es (2007)
32. Kornblith, S., Norouzi, M., Lee, H., Hinton, G.: Similarity of neural network representations revisited. In: International Conference on Machine Learning. pp. 3519–3529. PMLR (2019)
33. 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)
34. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13. pp. 740–755. Springer (2014)
35. Lin, T.Y., Maire, M., Belongie, S.J., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: European Conference on Computer Vision (2014), `https://api.semanticscholar.org/CorpusID:14113767`
36. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 10012–10022 (2021)
37. Lu, H., Dai, Y., Shen, C., Xu, S.: Index networks. IEEE Transactions on Pattern Analysis and Machine Intelligence **44**(1), 242–255 (2020)
38. Lu, H., Liu, W., Fu, H., Cao, Z.: Fade: Fusing the assets of decoder and encoder for task-agnostic upsampling. In: European Conference on Computer Vision. pp. 231–247. Springer (2022)
39. Lu, H., Liu, W., Ye, Z., Fu, H., Liu, Y., Cao, Z.: Sapa: Similarity-aware point affiliation for feature upsampling. Advances in Neural Information Processing Systems **35**, 20889–20901 (2022)
40. Min, J., Lee, J., Ponce, J., Cho, M.: Spair-71k: A large-scale benchmark for semantic correspondence. arXiv preprint arXiv:1908.10543 (2019)
41. Odena, A., Dumoulin, V., Olah, C.: Deconvolution and checkerboard artifacts. Distill **1**(10), e3 (2016)
42. Pont-Tuset, J., Perazzi, F., Caelles, S., Arbeláez, P., Sorkine-Hornung, A., Van Gool, L.: The 2017 davis challenge on video object segmentation. arXiv preprint arXiv:1704.00675 (2017)
43. Rambhatla, S.S., Chellappa, R., Shrivastava, A.: The pursuit of knowledge: Discovering and localizing novel categories using dual memory. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 9153–9163 (2021)
44. Ranftl, R., Bochkovskiy, A., Koltun, V.: Vision transformers for dense prediction. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 12179–12188 (2021)
45. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18. pp. 234–241. Springer (2015)
46. Shekhar, S., Bordes, F., Vincent, P., Morcos, A.: Objectives matter: Understanding the impact of self-supervised objectives on vision transformer representations. arXiv preprint arXiv:2304.13089 (2023)

47. Shocher, A., Cohen, N., Irani, M.: "zero-shot" super-resolution using deep internal learning. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3118–3126 (2018)
48. Strudel, R., Garcia, R., Laptev, I., Schmid, C.: Segmenter: Transformer for semantic segmentation. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 7262–7272 (2021)
49. Subramanian, A.: Torch cka. In: https://github.com/AntixK/PyTorch-Model-Compare. Github (2021)
50. Tan, W., Yan, B., Bare, B.: Feature super-resolution: Make machine see more clearly. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3994–4002 (2018)
51. Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., Jégou, H.: Training data-efficient image transformers & distillation through attention. In: International conference on machine learning. pp. 10347–10357. PMLR (2021)
52. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems **30** (2017)
53. Vo, V.H., Sizikova, E., Schmid, C., Pérez, P., Ponce, J.: Large-scale unsupervised object discovery. Advances in Neural Information Processing Systems **34**, 16764–16778 (2021)
54. Walmer, M., Suri, S., Gupta, K., Shrivastava, A.: Teaching matters: Investigating the role of supervision in vision transformers. arXiv preprint arXiv:2212.03862 (2022)
55. Wang, J., Chen, K., Xu, R., Liu, Z., Loy, C.C., Lin, D.: Carafe: Content-aware reassembly of features. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 3007–3016 (2019)
56. Wang, W., Xie, E., Li, X., Fan, D.P., Song, K., Liang, D., Lu, T., Luo, P., Shao, L.: Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 568–578 (2021)
57. Wang, Y., Shen, X., Yuan, Y., Du, Y., Li, M., Hu, S.X., Crowley, J.L., Vaufreydaz, D.: Tokencut: Segmenting objects in images and videos with self-supervised transformer and normalized cut. IEEE Transactions on Pattern Analysis and Machine Intelligence (2023)
58. Wei, X.S., Zhang, C.L., Wu, J., Shen, C., Zhou, Z.H.: Unsupervised object discovery and co-localization by deep descriptor transformation. Pattern Recognition **88**, 113–126 (2019)
59. Wu, H., Xiao, B., Codella, N., Liu, M., Dai, X., Yuan, L., Zhang, L.: Cvt: Introducing convolutions to vision transformers. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 22–31 (2021)
60. Xu, Y., Zhang, J., Zhang, Q., Tao, D.: Vitpose: Simple vision transformer baselines for human pose estimation. Advances in Neural Information Processing Systems **35**, 38571–38584 (2022)
61. Yun, S., Lee, H., Kim, J., Shin, J.: Patch-level representation learning for self-supervised vision transformers. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 8354–8363 (June 2022)
62. Zhu, M., Han, K., Zhang, C., Lin, J., Wang, Y.: Low-resolution visual recognition via deep feature distillation. In: ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 3762–3766. IEEE (2019)

63. Zhu, X., Su, W., Lu, L., Li, B., Wang, X., Dai, J.: Deformable detr: Deformable transformers for end-to-end object detection. arXiv preprint arXiv:2010.04159 (2020)
64. 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)
65. Zontak, M., Irani, M.: Internal statistics of a single natural image. In: CVPR 2011. pp. 977–984. IEEE (2011)