Recasting Generic Pretrained Vision Transformers As Object-Centric Scene Encoders For Manipulation Policies

Jianing Qian[1], Anastasios Panagopoulos[1] and Dinesh Jayaraman[1]

Abstract— Generic re-usable pre-trained image representation encoders have become a standard component of methods for many computer vision tasks. As visual representations for robots however, their utility has been limited, leading to a recent wave of efforts to pre-train robotics-specific image encoders that are better suited to robotic tasks than their generic counterparts. We propose *Scene Objects From Transformers*, abbreviated as `SOFT`(·), a wrapper around pre-trained vision transformer (`PVT`) models that bridges this gap without any further training. Rather than construct representations out of only the final layer activations, `SOFT`(·) individuates and locates object-like entities from `PVT` attentions, and describes them with `PVT` activations, producing an object-centric embedding. Across standard choices of generic pre-trained vision transformers `PVT`, we demonstrate in each case that policies trained on `SOFT`(`PVT`) far outstrip standard `PVT` representations for manipulation tasks in simulated and real settings, approaching the state-of-the-art robotics-aware representations. Code, appendix and videos: https://sites.google.com/view/robot-soft/

I. INTRODUCTION

In the last decade, computer vision researchers have demonstrated consistently [1], [2] that visual features pre-trained with the right objectives on large image datasets offer large advantages for learning to perform new visual tasks. Consequently, it is the standard practice today for generic "foundation" image encoder models [3]–[12] to be reused for virtually all standard computer vision tasks, including image classification, object detection, segmentation, pose estimation, monocular depth estimation, and video understanding, sometimes even despite domain differences with the pre-training data [13]. Thanks to this robust pre-training advantage, vision algorithms have been able to leverage the benefits of ever-improving neural network backbones, training algorithms, and pre-training objectives to boost performance across many tasks.

However, advantages from using these generic pre-trained image encoders have remained notably elusive in robotics [14], [15]. In response, there has been a recent wave of image encoders pre-trained specifically for robotic tasks, especially manipulation, often exploiting the temporal structure of human videos of manipulation activities [16]–[20]. These representations already achieve much more promising results than their generic counterparts for enabling robot learning and control, even though the pre-training datasets are less diverse, and objectives and architectures are less mature.

Towards understanding and addressing the deficiencies of generic image encoders for robotics, we focus on the

inference procedure that extracts image representations from such encoders to provide to downstream policies. With no new training, we propose a procedure to extract a more robotics-relevant representation of an image from a simple forward pass through a generic pre-trained encoder. Our insight is that the computational trace of a forward pass through a modern transformer neural network includes not just key, value, and query feature activations but also attention weights at each layer. Such attention weights have been noted in prior works to contain meaningful information, such as about referring expressions and phrases in language [21], [22], and object regions in images [9]. However, to our knowledge, all prior attempts to apply pre-trained image encoders to downstream tasks have ignored this information, instead relying solely on activations.

At a high level, given an input image, our approach identifies *attention-based* patch token groupings corresponding to object-like entities, and then computes *activation-based* representations to describe each such identified object. We call our method Scene Objects From Transformers, or `SOFT`(·) for short. `SOFT`(·) acts as a wrapper around pre-trained vision transformer (`PVT`) models. Rather than producing the standard activation-based "scene vector" representations from a `PVT`, `SOFT`(`PVT`) produces an object-centric image representation that individuates, locates, and describes each object in the scene. This reflects the object-centric structure of the world [23]–[26], which is useful for robotic manipulation [27]–[29].

Our experiments across simulated and real robotic manipulation settings demonstrate that: (1) `SOFT`(`PVT`) representations, which involve no new training, easily outperform vanilla `PVT` activation features for manipulation for various choices of the backbone `PVT`, (2) `SOFT`(`PVT`) with the best generic `PVT` models largely bridge the gap to today's best robotics-specific image encoders, and (3) `SOFT`(·) out-of-the-box consistently mines object groupings out of generic pre-trained vision transformer (`PVT`) models in unseen robotics domains better than prior object-centric embedding approaches, even with domain-specific training.

II. RELATED WORK

a) Robotics-Specific Pre-Trained Image Representations: In response to the growing evidence that widely used generic pre-trained image representations in computer vision form poor inputs for robotic policies, research has recently turned to robotics-specific image representations. Nair et al [16] first showed performance gains on various simulated and

[1]All authors are with GRASP Lab, Computer and Information Science Department, University of Pennsylvania, 3330 Walnut St, USA `jianingq, anpans, dineshj@seas.upenn.edu`
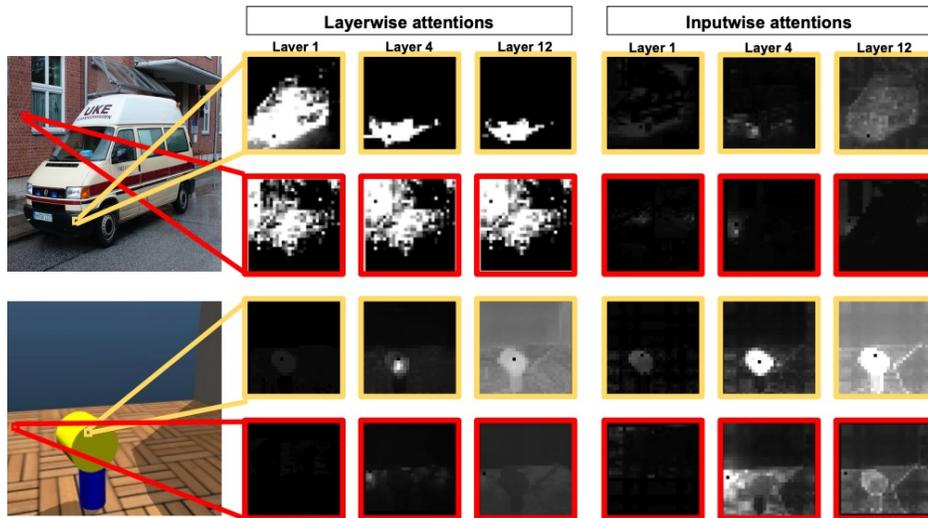
Fig. 1. Visualizing attentions for two images from ImageNet and Shapestacks. On each image, we consider a foreground patch (yellow) and a background patch (red). Attentions a_{ii} for each patch towards itself are zeroed out to identify the patch in attention images. Layerwise attentions at various DINO layers (left) reveal little information at higher layers, so we instead compute inputwise attentions (right) using attention flow.

real robotics tasks from training time-contrastive representations on human videos [30]. Since then, several other works [17]–[20] have followed suit, demonstrating gains from various training objectives on robotics-relevant data [30], [31], including objectives specifically motivated by usage for control tasks [17], [18]. Recent works [32], [33] come up with different benchmarks to study the effectiveness of pre-trained representations. However, the reason for the poor performance of generic models relative to these is not well-understood. We show that, at least for *vision transformer* models, this gap may be largely bridged through changes to the representation inference procedure.

b) Object-Centric Embeddings (OCE): An OCE summarizes the scene in terms of the objects, their locations and properties, and their relationships to each other. A common OCE format for an image x_i is $f_\theta(x_i) = (S = \{s_1, s_2, ..., s_{N_i}\}, C = \{c_1, c_2, ...c_{N_i}\})$. This is a tuple of two unordered discrete sets, the object slot vectors S and their coordinates C. Each slot vector $s_j \in \mathbb{R}^D$ describes one object or entity (such as an object part) in the scene, and the number N_i of such slots varies from scene to scene. Each location specifier c_i describes the location of its corresponding slot, such as through a segmentation mask, bounding box, or keypoint location. While the utility of acquiring such representations without manual supervision is widely recognized [23]–[26], including for robotic manipulation [27]–[29], current approaches face several key challenges, most notably architecture design and optimization difficulties, which we elaborate upon in Appendix. At a high level, coaxing neural networks to generate unordered discrete sets of objects without explicit supervision requires architectural choices that create hyperparameter-sensitivity and other difficulties during optimization [34]–[36]. The end result is that even the best of these approaches struggle to generalize beyond narrow domains [36]–[39]. In contrast, our

approach sidesteps all the above difficulties since it is only an inference procedure: it infers OCEs straight out of standard vision transformer architectures generating continuous scene-level activation maps and trained with mature and stable optimization procedures. As such, SOFT(PVT) can extract OCEs out-of-the-box on the same large natural image domains where the PVT scene-level image encoders can operate.

With the recent rise of pre-trained open-world object detectors [40], [41], several recent works have proposed to construct OCEs based on their output detections. MOO [42] uses OWL-ViT [41] to infer centroid locations of foreground objects at the start of task execution. FOCUS [43] uses SAM [40] outputs to learn object-centric world models. GROOT [44] uses SAM segmentation masks to build 3D representations of task-relevant objects. Concurrent with our work, POCR [45] builds OCEs by encoding SAM masks with various pre-trained vision encoders for robotics control.In contrast, our work is the first to construct useful OCEs from *generic* pre-trained vision transformers such as DINO-v2 [10] even when they are not explicitly trained for object recognition, by utilizing not only their outputs or final layer activations, but also their internal attentions.

c) Object Discovery from Image Encoders: Prior work has attempted to discover foreground objects from pre-trained convolutional or transformer image encoders by clustering pixel-wise or token-wise activations at one or more layers [46]–[49] using k-means or spectral clustering approaches. Recently, MaskCut [50] apply such an approach iteratively to discover multiple objects, still using the pixel-wise activations. We compare against using such activation-clustering approaches in our experiments but find that they perform worse than our approach that uses a complementary source of information generated in a transformer's forward pass, namely, internal attentions, rather than key, value, query activations.

DINO [9] first showed evidence that attention weights

inferred within vision transformers may be informative for object discovery: its last layer CLS attentions cleanly segment the foreground objects in single-object images, a feature that is also employed in [12]. MaskDistill [51] builds on this, using feature similarities between the query and key features of the last DINO layer and the CLS token to identify a single foreground object, which is thereafter distilled into a Mask R-CNN encoder that can extract multiple objects. We propose an improved approach to use attentions for multi-object discovery in robotic manipulation scenes that requires no new training: it employs full token-token attention matrices rather than just CLS attentions, aggregates them across multiple layers using an "attention rollout" procedure, and exploits a handful of other images from the same domain, easily available in robotics settings, for background removal.

III. MANIPULATION POLICIES USING REPRESENTATIONS INFERRED FROM GENERIC VISION TRANSFORMERS

Consider the following setup. For learning a new manipulation task, we are provided with a small number N of visual demonstration trajectories $\mathcal{D} = \{\tau_n\}_{n=1}^N$. The n-th trajectory is an image-action sequence $\tau_n = [x_{n,1}, u_{n,1}, x_{n,2}, u_{n,2}, ...]$. We would like to train a visual policy $\pi(x)$. For consistency with evaluations of prior pre-trained representations for robotics [16]–[18], we will train policies with simple behavior cloning, minimizing the objective: $\pi^* = \min_\pi \sum_{n,t} \ell^2(\pi(x_{n,t}), u_{n,t})$ where ℓ is the squared ℓ^2 error. We seek to develop a new pre-trained representation for robotics that would facilitate training such policies from limited demonstrations $\mathcal{D}$ to achieve higher rewards. Conceptually, we may compose a policy π from image representation (f_ψ) and action mapping (g_ϕ) modules: $\pi(x) = g_\phi(f_\psi(x))$. With no pre-training, policy learning involves training both f_ψ and g_ϕ from $\mathcal{D}$. With a pre-trained image encoder $f^*(\cdot)$, the policy $\pi(x) = g_\phi(f^*(x))$ has fewer learnable parameters, permitting learning from small $\mathcal{D}$. If the representations conveniently capture task-relevant information, the resulting policies would achieve high rewards. Good pre-trained representations must facilitate such sample-efficient, performant policy learning across a large range of unseen environments, objects, task, and robots.

We wish to develop a representation based on generic pre-trained vision transformers (PVT)[9]–[11], [52], [53]. As motivated above, standard feature activations at the output layers of such transformers have proven to be excellent pre-trained representations for vision tasks, but perform poorly for robotics; we verify this in Section IV. How might we generate a better robotics-ready representation from a PVT? To address this question, we will start by exploring PVT self-attentions, whose utility as representations is ignored in prior work.

A. The Information Held Within Transformer Self-Attentions

A vision transformer [8] takes as input a serialized sequence of small pxp image patches, e.g. $p = 16$. An image with spatial dimensions HxW is processed into $m = \frac{H}{p} \cdot \frac{W}{p}$ non-overlapping patch "tokens". Central to the computation in

each layer l of the transformer is the self-attention operator, which computes the next layer features for each token y_i as a weighted average of the "value" features v_j^l of other tokens $y_i^{l+1} \leftarrow \sum_j a_{ij}^l v_j^l$. These attention weights a_{ij}^l are assigned by a function with learned parameters, and determine how much each other patch j, out of all m patches, contributes to the new features of patch i. Information from all over the scene is thus mixed into the representations of each patch token. The "attention matrix" A^l is an mxm matrix, where the i-th row contains all m attention weights for patch i. A more detailed description of the basic vision transformer architecture is provided in [8].

In the layers of a well-trained transformer, the attention weights a_{ij}^l for a patch i, must be large for the patches j that most provide context for understanding patch i. In particular, in images, the meanings of small nxn tokens depend heavily on their context. For example, in the image in Fig 1 (top), no meaning can be assigned to the black patch token on the van bumper, without accounting for the context tokens, prominent among which is the rest of the van. The visualizations of attention weights from DINO [9] layers as $\frac{H}{n}$x$\frac{W}{n}$ grids, suggests that the weights might indeed capture information about object groupings.

How do these attention weights change from layer to layer, and does any one layer contain more information than others about the objects? Fig 1 shows examples of attention weights over multiple layers. The attentions in the first layer are based on low-level color and position similarities between patches and do not yet capture object groupings, except in very simple scenes where objects have a single color or texture, easily distinguished from their backgrounds. For example, the bumper patch's attentions extend far beyond the van on the bottom left, yet they do not quite capture the top-right regions of the van. On the other hand, interpreting layerwise attentions becomes more complex in higher layers because these attentions operate over patch representations that are no longer based on purely local information; instead each patch representation at these layers already contains information from all over the scene.

Several methods have been proposed to resolve this mixing problem to produce more interpretable visualizations of attentions in later transformer layers [22], [54], [55]. We build on perhaps the simplest approach, "attention rollout", first presented by Abnar et al. [22] in the context of language transformers. We seek to know: how much do the features y_i^l at the current layer l of the transformer depend on the *input* patch tokens j, which we will refer to as inputwise attentions $\tilde{a}_{ij}^l$. To answer this question, attention rollout prescribes matrix-multiplying the attention matrices A^l of all preceding layers, corresponding to a linear model of information mixing across layers. To account for skip connections between layers, as in our models, we modify this [56] to:

$$\tilde{A}^l = \prod_{l' \leq l}(A^{l'} + I). \tag{1}$$

To handle multi-head attentions, we simply average the attention matrices across the heads. Fig 1 visualizes inputwise
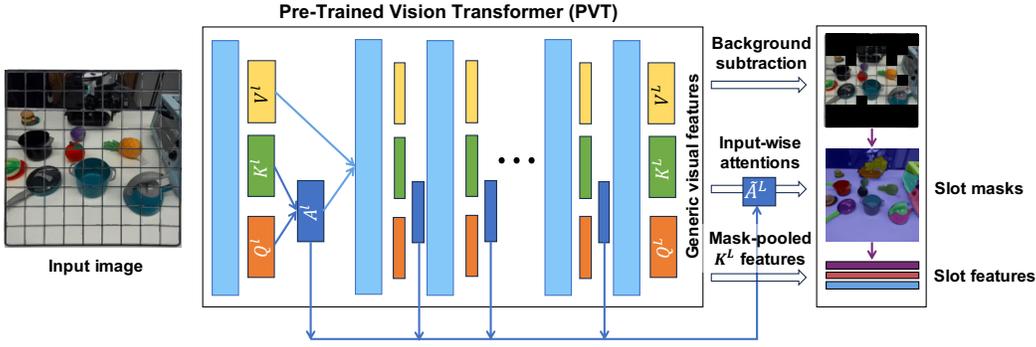
Fig. 2. SOFT($\cdot$) provides a wrapper around any pre-trained vision transformer model PVT. Relying on nothing other than the activations and attentions throughout PVT, SOFT($\cdot$) offers an alternative representation inference procedure to the standard last-layer activations. The resulting SOFT(PVT) representation is an object-centric image representation, suitable for off-the-shelf usage in robotic control tasks. V^l, K^l and Q^l represent the value, key and query of attention layers at layer l.

attentions across various layers in DINO. Foreground patch attentions (yellow) in later layers are now more interpretable, and higher layers contain more clearly semantic object or entity grouping information. The formation of these input-wise attention matrices is schematically depicted in Figure 2.

B. A Procedure To Infer Objects and Their Locations

With these insights, we now propose a simple clustering-based procedure to permit SOFT(PVT) to individuate and locate objects. Given the inputwise attentions $\tilde{a}_{ij}^l$ of a vision transformer PVT, SOFT(PVT) performs four steps:

- **Symmetrized inputwise attentions as similarity measures**: SOFT(PVT) first computes inputwise attentions $\tilde{A}^L$ at the final layer $l = L$ as prescribed by Eq (1). To explicitly account for object locations, we also compute a pairwise patch similarity matrix S. The symmetric matrix $A^s = \tilde{A}^L + \text{Transpose}(\tilde{A}^L) + S$ is treated as the measure of similarities between patch features. Both of the similarity matrices $(\tilde{A}^l + \text{Transpose}(\tilde{A}^L))$ and S are normalized between 0 and 1. In practice, we propagate only the top 10% of attention weights in each layer to highlight the key regions.
- **Background removal**: While there is no domain-specific training set for producing SOFT(PVT) representations, the demonstration dataset $\mathcal{D}$, available in advance, does have other images within the domain. We sample $m = 30$ reference frames per task from among the demonstrations $\mathcal{D}$ to identify and discard recurring, unchanging background regions. We follow the procedure described by Amir et al. [12]: each patch token in the reference frames is assigned to foreground or background based on its CLS attention score. Then, each patch token in the current frame is assigned a backgroundness score based on matching its key features to these reference tokens. Scores higher than a threshold are discarded. More details in the appendix.
- **Spectral clustering**: For the remaining tokens, we treat A^* as the affinity matrix between tokens in spectral clustering [57]. The number of clusters may be automatically determined by the eigengap heuristic, permitting handling a varying number of objects in the scene.
- **Mask Refinement**: As described above, SOFT($\cdot$) assigns

each nxn-pixel patch token to a cluster (8x8 in standard transformers), rather than each pixel. To generate smooth pixel-wise masks, we refine the patch-wise mask by applying a multi-label conditional random field [58]. Details in appendix.

Much of this procedure is visualized in Figure 2.

C. Activation-Based Object Descriptions

Having identified and located object-like regions as above, we now add slot vectors describing the contents of each region. For each object cluster "slot" i, its object slot vector s_i is the average-pooled PVT feature over the in-cluster patch tokens. At this stage, the combined object-centric scene descriptor SOFT(PVT)(x) is an unordered set of slot vectors $\{s_i\}_{i=1}^k$, each corresponding to an object-like entity in the scene, and capturing its location and contents. The cardinality k, i.e., the number of object slots, varies by image.

D. Policy Learning from Demonstrations

How might SOFT(PVT) be used as input to a policy, given that the slots are unordered and of variable number? While solutions based on set descriptors and graph neural networks are possible [59], we opt for a simple solution based on explicit slot binding to achieve an ordered, fixed-size representation. For each frame, we match SOFT(PVT) slots $\{s_{t,i}\}$ in the current frame to the averaged slot features from a randomly selected reference demonstration $\{\sum_j s_{ref,j}\}$. We use Hungarian matching [60] based on Euclidean distances between slot vectors, so that each reference slot is matched to at most one slot in the current frame. Unmatched slots $s_{t,i}$ corresponding to spurious detections or distractor objects are discarded, and the remaining slots are now ordered as per the reference frame.

Finally, we concatenate these newly ordered slot vectors $\{s_1, \ldots, s_{k^*}\}_t$ into a flat, fixed-size representation $(k^* \times D)$ and feed into a multi-layer perceptron g_ϕ.

$$\pi(x_t) := g_\phi(\text{concat}(s_1(x_t), \ldots, s_{k^*}(x_t))) \qquad (2)$$

This is now compatible with the behavior cloning policy training procedure described above. A more formal pseudo-code description of SOFT($\cdot$) is provided in the supplementary material, and we will release all code upon publication.

A) Shapestacks B) B&O C) RLB-PickUpCup

D) RLB-PutRubbishInBin E) MW-Hammer F) Real-Tabletop
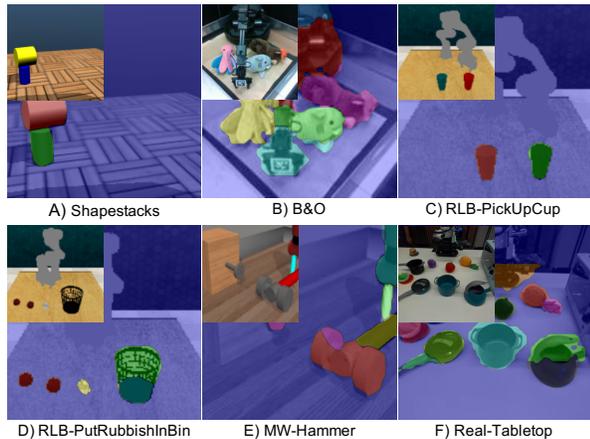
Fig. 3. Data from our datasets and simulation environments. We show the segmentation masks from SOFT(PVT) as well as the original images. Blue indicates background.

TABLE I
SEGMENTATION METRICS. HIGHER IS BETTER.

Backbone	Datasets→ Num-images →	Shapestacks 20k		RLB-PickUpCup 1.4k		B&O 300	
	Methods ↓	ARI	MSC	ARI	MSC	ARI	MSC
DINO-ViT	SOFT [Ours]	**0.80**	**0.86**	**0.93**	**0.50**	**0.55**	**0.58**
	ODIN-feature	0.39	0.44	0.20	0.40	0.54	0.54
	MaskCut[50]	0.08	0.24	0.447	0.326	0.04	0.12
DINOv2-ViT	SOFT [Ours]	**0.72**	**0.65**	**0.99**	**0.51**	**0.65**	**0.63**
	ODIN-feature	0.04	0.12	0.00	0.22	0.60	0.62
DeiT	SOFT* [Ours]	0.17	0.40	**0.82**	**0.32**	**0.69**	**0.52**
	ODIN-feature	0.18	0.41	0.08	0.32	0.47	0.46
ConvNext	SOFT* [Ours]	0.18	0.24	0.23	0.09	**0.42**	**0.30**
	ODIN-feature	0.12	0.24	0.23	0.09	0.15	0.19
MVP [67]	SOFT [Ours]	**0.69**	**0.55**	**0.80**	**0.43**	0.60	0.53
	ODIN-feature	0.67	0.44	0.48	0.26	0.68	0.55
	Genesis-v2[68]	0.73	0.71	-	-	0.25	0.20
	AST-SEG[38]	0.73	0.74	-	-	0.03	0.04

IV. EXPERIMENTS

SOFT(·) provides an object-centric wrapper around pre-trained vision transformers (PVT), aiming to improve the performance of generic pre-trained vision transformers as reusable representation encoders for robotic tasks. We first evaluate the extent to which SOFT(PVT) can successfully discover object-like regions in various robotic settings, out of the box. Next, we evaluate SOFT(PVT) object-centric embeddings to see: to what extent do they improve downstream policy performance compared to standard PVT representations extracted from the same models? How does this compare to robotics-specific pre-trained representations?

A. Discovering Object-Like Regions

While SOFT(·) is not a segmentation method, it discovers object-like segmentation masks en route to generating a representation for robotics. We use these segments to evaluate SOFT(·) under various conditions.

Datasets. We first evaluate this component of SOFT(·) on 3 offline datasets with increasing degrees of difficulty: (1) **Shapestacks** [61] is a synthetic dataset that has geometric shapes stacked on top of each other in various stable and unstable formations. (2) **RLB-PickUpCup** is a tabletop manipulation task in RLBench [62], that we will use for control evaluations. **RLB-PickUpCup** requires the robot to pick up the red cup when presented with another distractor cup. We use the frames from 100 demos. (3) **B&O** [63] contains real images of a robot pushing deformable toy objects in its workspace, with annotated object masks. All object locations are randomized at the start of each episode.

PVT backbones. We evaluate SOFT(·) with various choices of PVT backbones: 1) **DINO-ViT** [9], a popular pretrained ViT network that produces useful representations for downstream vision and robotics tasks. 2) **DINOv2-ViT** [10] is the followup work that shows improved performances compared to DINO-ViT. 3) **DeiT** [64], [65] is a supervised ViT network trained on ImageNet-21k. 4) **ConvNext** [66] is a modern convolutional network that matches vision transformers on

various vision benchmarks. Note that this is the only non-transformer based architecture, so we only apply the spectral clustering step and mask refinement step on the features of the last convolutional layer. 5) **MVP** [67] is a transformer-based pretrained visual encoder for robot manipulation tasks. Note that the background removal procedure only applies to ViT networks trained with unsupervised objectives, so we remove this step when performing experiments with DeiT and ConvNext and marked it as SOFT*.

Activation-clustering. With each backbone, we compare SOFT(·)'s attention-based object discovery to **ODIN** [46]-style k-means clustering on final layer key features, and also iterative spectral clustering on key features as proposed in **MaskCut** [50].

Unsupervised OCEs. Finally, we compare against prior unsupervised object-centric embeddings, which typically train *only* on in-domain images: **Genesis-V2** [68], and the current state-of-the-art, **AST-SEG** [38], unlike SOFT(·) which is *only* pre-trained on out-of-domain images.

We report the standard segmentation metrics: Adjusted Rand Index (ARI) [69], [70] and Mean Segmentation Covering (MSC) [71], [72], both in [0, 100], higher is better. See appendix for an explanation of these metrics. For each dataset, we compute the average metrics for each method over a test set of 320 images.

Table I shows these results, and Fig 3 shows examples of the segmentations masks on the evaluated datasets (more on website). First, with a fixed PVT backbone, SOFT(PVT) object masks produced using our attention rollout procedure consistently match or outperform activation-clustering with ODIN or MaskCut, validating this choice. Second, in-domain trained OCEs like Genesis-v2 or AST-SEG struggle to learn on smaller datasets, RLB-PickUpCup and B&O. Finally, comparing among different backbones, DINO-ViT and DINOv2-ViT segments best match the object annotations.

B. Robot Manipulation Experiments

We have thus far only evaluated SOFT(·) object masks, as a way to sanity-check our approach, and select the strongest PVT backbone through cheap offline evaluations. We now evaluate SOFT(DINOv2) as an object-centric embedding for

TABLE II
SUCCESS RATES FOR MANIPULATION TASKS.

	RLB-PickUpCup 100 demos	RLB-PutRubbishInBin 100 demos	MW-Hammer 10 demos
SOFT(DINOv2) [Ours]	45	15	73.4
DINOv2-Flat-CLS	10	0	67.6
DINOv2-Flat-Pool	0	0	69
LIV [18]	49.3	21.9	81.7
R3M [16]	52.3	20.7	81.3
ImageNet	14.0	6.7	53.7
SAM [73]	15.9	4.4	51.5

Fig. 4. Success rate of different methods as a function of the number of demonstrations.

Fig. 5. Example of a successful policy rollout using SOFT(DINOv2)

robot learning. We perform thorough experiments comparing various approaches quantitatively on two RLBench [62] and one Meta-World [74] tabletop manipulation environments: RLB-PickUpCup and RLB-PutRubbishInBin (see Figure 3). In **RLB-PickUpCup**, the goal is to pick up the red cup in the presence of a distractor cup. The locations of the two cups are randomized for each episode as well as the color of the distractor. This requires the learned scene representations to include precise information about the cup's location and identity. During the task, cups could be heavily occluded by each other or the robot arm, thus the representation should also be robust under such conditions. In **RLB-PutRubbishInBin**, the robot needs to locate and pick up irregular-shaped rubbish among two other apples and put it in a trash can, all with randomized starting location. This is one of the more challenging tasks in RLBench, stress testing the representations. In **MW-Hammer**, the goal is to pick up a hammer and then smash a nail into a wall. The location of the hammer is randomized.

Baselines. Recall that `SOFT(·)` is a wrapper around a generic `PVT`, in this case DINOv2. We compare this to using standard last layer activation-based features of the `PVT`: (1) **DINOv2-Flat-CLS** uses the CLS token of the last attention layer, which is widely adopted as a representations for downstream tasks [9], [10], and 2) **DINOv2-Flat-Pool** mean-pools the key features over the entire image. Next, we compare to a CNN-based generic image feature, **ImageNet**-pretrained ResNet [3], and to two state-of-the-art robotics-specific pre-trained features: LIV [18] and R3M [16] that also use the ResNet backbone. Finally, to evaluate whether good segmentation alone is sufficient, we compare against using the recent SOTA segmentation model, **SAM** [73]. For this, we train a CNN encoder to process SAM object masks and feed into the policy.

Table II shows the success rates. The `SOFT(DINOv2)` OCE easily outperforms both the flat DINOv2 baselines, though they use the same backbone, trained with the same data. These more standard feature extraction approaches perform very poorly, as does ImageNet, which represents another standard representation for computer vision tasks. It is precisely the poor performance of standard vision representations that prompted the recent development of more robotics-specific pre-trained representations such as LIV and R3M, which indeed perform much better. Here, `SOFT(·)` shines: using differently inferred representations from the same pre-trained vision transformer model DINOv2,
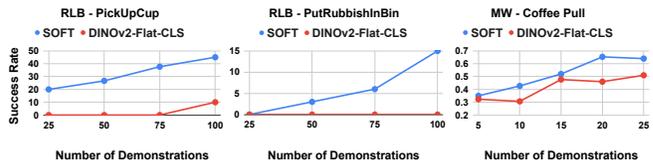
it performs approximately on par with these robotics-aware representations. Qualitatively, we observe that even when it fails, the learned `SOFT(DINOv2)` policy fails more gracefully than other methods, most often moving correctly to the right locations around the object of interest. See videos on website.

Finally, Figure 4 studies the performance of `SOFT(DINOv2)` and DINOv2-Flat-CLS with varying numbers of training demonstrations: object-centric structure in `SOFT` enables learning from very few demonstrations.

C. Real Robot Experiments

We also evaluated `SOFT(DINOv2)` policies in a cluttered kitchen countertop setting (see Figure 3 F)), comparing them to DINOv2-Flat-CLS and LIV representations for the task of picking and placing toy fruits into a pot. We collect 50 expert demonstrations by teleoperation, and extract SOFT(DINOv2) features for policy learning. We randomized the placement of fruits during demo and evaluation. Our observations here are similar: DINOv2-CLS performs very poorly, but `SOFT(DINOv2)` performs noticeably better. With 10 trials of randomized placement of various fruits on the table, `SOFT(DINOv2)` has a success rate of 40% which is as often as LIV policies. Figure 5 shows a successful rollout of the policy learned with SOFT(DINOv2). We show videos of our sim and real robot trials on the website.

V. CONCLUSIONS

By introducing a simple change to the representation inference procedure from a pre-trained vision transformer `PVT` backbone, `SOFT(PVT)` produces object-centric embeddings that yield dramatic performance gains for control tasks, to the extent that it approximately matches the performance of the best robotics-specific pre-trained representations. Our results validate the utility of transformer attentions to inform representations and the power of object-centric embeddings, and demonstrate a route to making generic visual representations, wildly successful in computer vision, compatible with the specific needs of robot learners.

REFERENCES

[1] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "Overfeat: Integrated recognition, localization and detection using convolutional networks," *arXiv preprint arXiv:1312.6229*, 2013.

[2] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587.

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[4] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *International Conference on Machine Learning*, PMLR, 2020, 1597–1607.

[5] X. Chen and K. He, "Exploring simple siamese representation learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, 15792–15801.

[6] J.-B. Grill, F. Strub, F. Altch'e, *et al.*, "Bootstrap your own latent: A new approach to self-supervised learning," *arXiv preprint arXiv:2006.07733*, 2020.

[7] X. Chen, H. Fan, R. Girshick, and K. He, "Improved baselines with momentum contrastive learning," *arXiv preprint arXiv:2003.04297*, 2020.

[8] A. Dosovitskiy, L. Beyer, A. Kolesnikov, *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[9] M. Caron, H. Touvron, I. Misra, *et al.*, "Emerging properties in self-supervised vision transformers," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 9650–9660.

[10] M. Oquab, T. Darcet, T. Moutakanni, *et al.*, "Dinov2: Learning robust visual features without supervision," *ArXiv*, vol. abs/2304.07193, 2023. [Online]. Available: https : / / api . semanticscholar . org / CorpusID:258170077.

[11] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, "Masked autoencoders are scalable vision learners," *arXiv preprint arXiv:2111.06377*, 2021.

[12] S. Amir, Y. Gandelsman, S. Bagon, and T. Dekel, "Deep vit features as dense visual descriptors," *ArXiv*, vol. abs/2112.05814, 2021.

[13] D. Kim, K. Wang, S. Sclaroff, and K. Saenko, "A broad study of pre-training for domain generalization and adaptation," in *European Conference on Computer Vision*, Springer, 2022, pp. 621–638.

[14] A. Zeng, S. Song, S. Welker, J. Lee, A. Rodriguez, and T. Funkhouser, "Learning synergies between pushing and grasping with self-supervised deep reinforcement learning," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2018, pp. 4238–4245.

[15] L. Yen-Chen, A. Zeng, S. Song, P. Isola, and T.-Y. Lin, "Learning to see before learning to act: Visual pre-training for manipulation," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2020, pp. 7286–7293.

[16] S. Nair, A. Rajeswaran, V. Kumar, C. Finn, and A. Gupta, "R3m: A universal visual representation for robot manipulation," *arXiv preprint arXiv:2203.12601*, 2022.

[17] Y. J. Ma, S. Sodhani, D. Jayaraman, O. Bastani, V. Kumar, and A. Zhang, "Vip: Towards universal visual reward and representation via value-implicit pre-training," *arXiv preprint arXiv:2210.00030*, 2022.

[18] Y. J. Ma, W. Liang, V. Som, *et al.*, "Liv: Language-image representations and rewards for robotic control," *arXiv preprint arXiv:2306.00958*, 2023.

[19] I. Radosavovic, T. Xiao, S. James, P. Abbeel, J. Malik, and T. Darrell, "Real-world robot learning with masked visual pre-training," in *Conference on Robot Learning*, PMLR, 2023, pp. 416–426.

[20] A. Majumdar, K. Yadav, S. Arnaud, *et al.*, "Where are we in the search for an artificial visual cortex for embodied intelligence?" In *Workshop on Reincarnating Reinforcement Learning at ICLR 2023*, 2023.

[21] S. Vashishth, S. Upadhyay, G. S. Tomar, and M. Faruqui, "Attention interpretability across nlp tasks," *arXiv preprint arXiv:1909.11218*, 2019.

[22] S. Abnar and W. Zuidema, "Quantifying attention flow in transformers," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 4190–4197.

[23] A. Dittadi, S. S. Papa, M. De Vita, B. Schölkopf, O. Winther, and F. Locatello, "Generalization and robustness implications in object-centric learning," in *Proceedings of the 39th International Conference on Machine Learning*, K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, and S. Sabato, Eds., ser. Proceedings of Machine Learning Research, vol. 162, PMLR, 2022, pp. 5221–5285. [Online]. Available: https://proceedings.mlr.press/v162/dittadi22a.html.

[24] J. Yoon, Y.-F. Wu, H. Bae, and S. Ahn, "An investigation into pre-training object-centric representations for reinforcement learning," *arXiv preprint arXiv:2302.04419*, 2023.

[25] K. Greff, S. van Steenkiste, and J. Schmidhuber, "On the binding problem in artificial neural networks," Dec. 2020. arXiv: 2012.05208 [cs.NE].

[26] S. van Steenkiste, K. Greff, and J. Schmidhuber, "A perspective on objects and systematic generalization in model-based rl," *arXiv preprint arXiv:1906.01035*, 2019.

[27] W. Yuan, C. Paxton, K. Desingh, and D. Fox, "Sornet: Spatial object-centric representations for sequential

manipulation," in *Conference on Robot Learning*, PMLR, 2022, pp. 148–157.

[28] Y. Jiang, A. Gupta, Z. Zhang, *et al.*, "Vima: General robot manipulation with multimodal prompts," *arXiv preprint arXiv:2210.03094*, 2022.

[29] Y. Zhu, A. Joshi, P. Stone, and Y. Zhu, "Viola: Object-centric imitation learning for vision-based robot manipulation," in *Conference on Robot Learning*, PMLR, 2023, pp. 1199–1210.

[30] M. Xu, C. Zhao, M. Ramazanova, and B. Ghanem, "Ego4d: Around the world in 3,000 hours of egocentric video," in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2022, pp. 18 973–18 990.

[31] D. Damen, H. Doughty, G. M. Farinella, *et al.*, "Scaling egocentric vision: The epic-kitchens dataset," *arXiv preprint arXiv:1804.02748*, 2018.

[32] Y. Hu, R. Wang, L. E. Li, and Y. Gao, "For pre-trained vision models in motor control, not all policy learning methods are created equal," *arXiv preprint arXiv:2304.04591*, 2023.

[33] W. Pumacay, I. Singh, J. Duan, R. Krishna, J. Thomason, and D. Fox, "The colosseum: A benchmark for evaluating generalization for robotic manipulation," 2023.

[34] M. Engelcke, O. P. Jones, and I. Posner, "Reconstruction bottlenecks in Object-Centric generative models," Jul. 2020. arXiv: `2007.06245 [cs.LG]`.

[35] S. Papa, O. Winther, and A. Dittadi, "Inductive biases for object-centric representations in the presence of complex textures," in *UAI 2022 Workshop on Causal Representation Learning*, 2022.

[36] Y. Yang and B. Yang, "Promising or elusive? unsupervised object segmentation from real-world single images," *NeurIPS*, 2022.

[37] G. Singh, Y. Kim, and S. Ahn, "Neural block-slot representations," *arXiv preprint arXiv:2211.01177*, 2022.

[38] B. Sauvalle and A. de La Fortelle, "Unsupervised multi-object segmentation using attention and soft-argmax," *2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 3266–3275, 2022.

[39] O. Biza, S. van Steenkiste, M. S. M. Sajjadi, G. F. Elsayed, A. Mahendran, and T. Kipf, "Invariant slot attention: Object discovery with slot-centric reference frames," *ArXiv*, vol. abs/2302.04973, 2023.

[40] A. Kirillov, E. Mintun, N. Ravi, *et al.*, "Segment anything," *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 3992–4003, 2023. [Online]. Available: `https : / / api . semanticscholar . org / CorpusID : 257952310`.

[41] M. Minderer, A. A. Gritsenko, A. Stone, *et al.*, "Simple open-vocabulary object detection with vision transformers," *ArXiv*, vol. abs/2205.06230, 2022. [Online]. Available: `https://api.semanticscholar. org/CorpusID:248721818`.

[42] A. Stone, T. Xiao, Y. Lu, *et al.*, "Open-world object manipulation using pre-trained vision-language models."

[43] S. Ferraro, P. Mazzaglia, T. Verbelen, and B. Dhoedt, "Focus: Object-centric world models for robotics manipulation," *ArXiv*, vol. abs/2307.02427, 2023. [Online]. Available: `https://api.semanticscholar. org/CorpusID:259342267`.

[44] Y. Zhu, Z. Jiang, P. Stone, and Y. Zhu, "Learning generalizable manipulation policies with object-centric 3d representations," in *Conference on Robot Learning*, 2023. [Online]. Available: `https : / / api . semanticscholar . org / CorpusID : 264426640`.

[45] J. Shi*, J. Qian*, Y. J. Ma, and D. Jayaraman, "Plug-and-play object-centric representations from "what" and "where" foundation models," *ICRA*, 2024. [Online]. Available: `https : / / sites . google . com / view/pocr`.

[46] O. J. Hénaff, S Koppula, E Shelhamer, D Zoran, and others, "Object discovery and representation networks," *ECCV*, 2022.

[47] O. Siméoni, G. Puy, H. V. Vo, *et al.*, "Localizing objects with self-supervised transformers and no labels," in *British Machine Vision Conference*, 2021.

[48] Y. Wang, X. Shen, S. X. Hu, Y. Yuan, J. L. Crowley, and D. Vaufreydaz, "Self-supervised transformers for unsupervised object discovery using normalized cut," *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14 523–14 533, 2022.

[49] Z. Lin, Z. Yang, and Y. Wang, "Foreground guidance and multi-layer feature fusion for unsupervised object discovery with transformers," *2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 4032–4042, 2022.

[50] X. Wang, R. Girdhar, S. X. Yu, and I. Misra, "Cut and learn for unsupervised object detection and instance segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 3124–3134.

[51] W. Van Gansbeke, S. Vandenhende, and L. Van Gool, "Discovering object masks with transformers for unsupervised semantic segmentation," *arXiv preprint arXiv:2206.06363*, 2022.

[52] J. Zhou, C. Wei, H. Wang, *et al.*, "Ibot: Image bert pre-training with online tokenizer," *arXiv preprint arXiv:2111.07832*, 2021.

[53] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, "Training data-efficient image transformers distillation through attention," *arXiv preprint arXiv:2012.12877*, 2020.

[54] H. Chefer, S. Gur, and L. Wolf, "Transformer interpretability beyond attention visualization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 782–791.

[55] J. Ma, Y. Bai, B. Zhong, W. Zhang, T. Yao, and T. Mei, "Visualizing and understanding patch interactions in

vision transformer," *arXiv preprint arXiv:2203.05922*, 2022.

[56] J. Gil, *Exploring explainability in vision transformers*, `https : / / jacobgil . github . io / deeplearning / vision – transformer – explainability`, Accessed: 2023-03-06.

[57] U. Von Luxburg, "A tutorial on spectral clustering," *Statistics and computing*, vol. 17, pp. 395–416, 2007.

[58] P. Krähenbühl and V. Koltun, "Efficient inference in fully connected crfs with gaussian edge potentials," *ArXiv*, vol. abs/1210.5644, 2011. [Online]. Available: `https : / / api . semanticscholar . org / CorpusID:5574079`.

[59] J. Kim, S. Oh, and S. Hong, "Transformers generalize deepsets and can be extended to graphs & hypergraphs," *Advances in Neural Information Processing Systems*, vol. 34, pp. 28 016–28 028, 2021.

[60] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval Research Logistics (NRL)*, vol. 52, 1955. [Online]. Available: `https://api. semanticscholar.org/CorpusID:9426884`.

[61] O. Groth, F. B. Fuchs, I. Posner, and A. Vedaldi, "Shapestacks: Learning vision-based physical intuition for generalised object stacking," in *Proceedings of the european conference on computer vision (eccv)*, 2018, pp. 702–717.

[62] S. James, Z. Ma, D. R. Arrojo, and A. J. Davison, "Rlbench: The robot learning benchmark & learning environment," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3019–3026, 2020.

[63] J. Qian, A. Panagopoulos, and D. Jayaraman, "Discovering deformable keypoint pyramids," in *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXVI*, Springer, 2022, pp. 545–561.

[64] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jegou, "Training data-efficient image transformers amp; distillation through attention," in *International Conference on Machine Learning*, vol. 139, 2021, pp. 10 347–10 357.

[65] H. Touvron, M. Cord, and H. Jegou, "Deit iii: Revenge of the vit," *arXiv preprint arXiv:2204.07118*, 2022.

[66] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie, "A convnet for the 2020s," *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[67] .

[68] M. Engelcke, O. P. Jones, and I. Posner, "Genesis-v2: Inferring unordered object representations without iterative refinement," in *Neural Information Processing Systems*, 2021.

[69] W. Rand, "Objective criteria for the evaluation of clustering methods," *Journal of the American Statistical Association*, vol. 66, pp. 846–850, 1971.

[70] L. Hubert and P. Arabie, "Comparing partitions," *Journal of Classification*, vol. 2, pp. 193–218, 1985.

[71] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 33, no. 5, pp. 898–916, 2010.

[72] M. Engelcke, A. R. Kosiorek, O. P. Jones, and I. Posner, "Genesis: Generative scene inference and sampling with object-centric latent representations," *arXiv preprint arXiv:1907.13052*, 2019.

[73] A. Kirillov, E. Mintun, N. Ravi, *et al.*, "Segment anything," *arXiv:2304.02643*, 2023.

[74] T. Yu, D. Quillen, Z. He, *et al.*, "Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning," *ArXiv*, vol. abs/1910.10897, 2019. [Online]. Available: `https : / / api . semanticscholar . org / CorpusID : 204852201`.