Abstract

Understanding the visual world from human perspectives has been a long-standing challenge in computer vision. Egocentric videos exhibit high scene complexity and irregular motion flows compared to typical video understanding tasks. With the egocentric domain in mind, we address the problem of self-supervised, class-agnostic object detection, aiming to locate all objects in a given view, without any annotations or pre-trained weights. Our method, self-supervised object detection from egocentric videos (DEVI), generalizes appearance-based methods to learn features end-to-end that are category-specific and invariant to viewing angle and illumination. Our approach leverages natural human behavior in egocentric perception to sample diverse views of objects for our multi-view and scale-regression losses, and our cluster residual module learns multi-category patches for complex scene understanding. DEVI results in gains up to 4.11% AP50, 0.11% AR1, 1.32% AR10, and 5.03% AR100 on recent egocentric datasets, while significantly reducing model complexity. We also demonstrate competitive performance on out-of-domain datasets without additional training or fine-tuning.

1. Introduction

The ability to detect objects in complex scenes is essential in smart applications and systems, such as autonomous vehicles [31], precision agriculture [3], 3D reconstruction and mapping [58], episodic memory [36], and remote sensing [4]. Broadly stated, the best performing object detection methods require large amounts of densely annotated data, providing bounding boxes for all or most objects in the scene [27, 54, 88]. Such annotations are costly, time consuming to produce, and difficult to scale over large or complex datasets [8]. Recent methods address the costly procedure by using either weak annotations [2, 44, 70, 71], or general self-supervision pre-training [12, 41]. However, such methods lack generalizability to complex scenes, often depending on image-wise features which lack feature granularity, leading to poor object localization and attention coverage. In this work, we aim to both maximize applicable scene complexity and minimize annotation costs by learning a class-agnostic object detector from highly diverse videos without using any annotations. Our method is able to distinguish different-category objects, while also remaining consistent for same-category objects. Best viewed in color; colors are random.
We are also interested in egocentric videos for their highly variable viewing directions and egocentric optical flow. While in many ways adding difficulty, these characteristics can provide unique advantages as well, and increased availability of egocentric data in recent years [18, 36, 65, 75] brings new opportunities. In particular, objects and the environment are mostly stationary in many egocentric videos, with head movements and locomotion being the primary sources of camera motion. We draw connections to classical computational appearance-based methods such as the bidirectional reflectance distribution function (BRDF) [62] and Bidirectional Texture Function (BTF) [20], whose computational appearance functions capture the distribution of reflectance measurements of a given opaque surface from all possible viewing angles and illumination conditions at well defined sampling structures and scales. We propose leveraging egocentric camera motion to naturally and unobtrusively sample instances from an object’s computational appearance distribution (see Fig. 2). 1 From these diverse views, we propose a novel, end-to-end, self-supervised method for learning features by matching multi-temporal patches covering the same surface or object, thus learning good features for class-agnostic object detection.

An inherent challenge of patch sampling and patch-wise representation learning is content ambiguity: A patch may be sampled from any object, group of objects, or empty surfaces, leading to ambiguous category association. For that reason, we utilize our object residual module to encode soft representation to patches, capturing the affinity of patches and all learnt clusters. The object residual module also allows us to define the number of expected categories and therefore learn category-specific features, unlike common self-supervised methods which learn image-wise, general features. To the best of our knowledge, we are the first to learn effective self-supervised features from egocentric videos. We qualitatively demonstrate the ability of our method to generate category-specific features in Fig. 1.

Our contributions in this work are as follows:

1. We present a self-supervised object detection model from egocentric videos (DEVI) that estimates locations of objects in complex scenes.
2. We propose loss functions inspired by computational appearance methods and tuned to egocentric perception named the multi-view and scale-regression losses.
3. Our object residual module extends existing work on patch representation learning and complex scene understanding to learn category-specific features and precise representation of ambiguous patches without hand-crafted assumptions.

2. Related work

2.1. Self-supervised representation learning

Increasing availability of unlabeled images and videos has inspired researchers to learn effective representations without manual annotations. These unsupervised methods learn invariance to color intensity [47], geometric and affine transformations [61], geometric and affine transformations [61], temporal ordering [28], relative sub-patch localization [24], and patch filling [82]. Traditionally, self-supervised learning (SSL) methods maximize similarity of global features of an image and its transform [13, 14] or learn through clustering features into pseudo-labels [10, 11]. Many of these concepts have been extended to self-supervised video representations as well [30, 37, 38, 39, 64, 68, 76]. Although DEVI and these SSL methods both aim to learn features from unlabeled video, the learning objectives and settings are very different: SSL methods generally learn generic visual features requiring labels to fine-tune on for downstream tasks, while DEVI does not use any annotations at any stage of training.

When scenes are complex, with many, diverse, and/or small objects, however, global features fail to capture fine-grain details. To address more complex scenes, recent methods propose patch-wise self-supervision approaches [5, 45, 48] learning local, surface-based feature representations. MATTER [5] does this with remote sensing imagery, though with significant data assumptions, requiring multi-spectral, multi-temporal, and spatially aligned inputs. In contrast to MATTER, we discard the inter-cluster residual weighted average for more concise residual representation, remove the need for spatial alignment by explicitly learning to match patches, and eliminate the multi-spectral constraint by generalizing the notion of material and texture to objects.

---

1 More traditional, non-egocentric video datasets such as DAVIS [66] and YT-8K [69] tend to have static or stable cameras following a specific object, and video motion is often due to object motion or actions, with accompanying changes in form or appearance. This makes such video a lesser fit for our BRDF and BTF inspired patching matching method.
2.2. Unsupervised class agnostic object detection

Because of the challenge of converting general self-supervised features to category-specific features, unsupervised object detection is still an open issue. Consequently, many methods [43, 73, 80, 81, 84, 85] follow a naive pipeline similar to the following: (1) self-supervised pre-training of a general purpose network (e.g., DINO [12]), (2) generate object discovery predictions for the entire dataset (single object detected, even if there are multiple in the scene), (3) cluster features of discovered objects into a predefined number of clusters equal to the number of categories in the dataset (e.g., foreground/background for class agnostic object detection), and (4) train an off-the-shelf detector using cluster labels and object discovery predictions.

In order for these methods to be effective, every module in the pipeline must be successful, and individual failures may affect the entire system. While such approaches may be relatively effective on simple datasets such as Pascal VOC [27] or COCO [54], their reliance on object-centric self-supervised pre-training produces sub-optimal features (e.g., object saliency) on datasets with increased complexity [36], leading to poor object discovery predictions and subsequent overall object detection performance, as shown in Tab. 1. In contrast, our method trains end-to-end, and is able to learn fine-grain, category-specific features.

2.3. Patch-based learning

Many computer vision methods are based on patch-wise learning, including SIFT [57], HOG [29], convolutional neural networks [50], and vision transformers (ViT) [25]. These approaches use patch-wise features (e.g., kernel weights) to obtain global, image-wise representations. Typically, such patch representations are transient or intermediate to some image-level objectives such as detection [86], image-deblurring [60], image-editing [7], or place recognition [40], where global, dense features are achieved. A more explicit utility of patch representations and local descriptors is to predict sparse features, such as keypoints, which are then used for global objectives such as depth estimation [55] and 3D reconstruction [32]. In contrast to these methods, where some patch operation or representation is transient and/or implicit for an image-wise task, we aim to explicitly learn both dense and local features through our multi-view and scale-regression patch-based loss functions.

2.4. Learning from egocentric data

The unique challenges presented in egocentric data have compelled egocentric-specific methods and data collection efforts for various tasks. The introduction of egocentric datasets such as Ego4D [36] and EpicKitchens [18] propelled work in egocentric action recognition [67], egocentric video-language pre-training [53], task understanding [46], and object discovery [9, 19, 51]. Due to the elevated complexity of egocentric settings, methods often require annotated data and/or specialized hardware. Instead, we utilize the innate properties of egocentric videos to implicitly learn high-level object features, without additional annotations, specialized hardware, or intermediate tasks.

3. DEVI

DEVI aims to learn fine-grain, category-specific features that are robust to varying viewing angles and illumination conditions from egocentric videos. We achieve this without any supervision, pre-trained weights, or hand-crafted assumptions about the data in an end-to-end manner for the task of class agnostic object detection. By using patches, we allow the model to detect local features from their global context and learn patch-level, local objectives, which increase feature granularity and enables isolation of regions in highly complex scenes. Our patch-wise objectives align with our computational appearance analogy: an objective function that operates in the temporal space, enforcing similarity of multi-temporal patches, and a function in the scale space, enforcing similarity of multi-scale patches. The former captures appearance variations in time such as viewing angles and illumination conditions, and the latter captures appearance variations in scale. The framework’s training and inference pipelines are illustrated in Fig. 3 and 6.

3.1. Pipeline overview

Given a video \( V = \{x^0, x^1, x^2, \ldots, x^{T−1}\} \) composed of \( T \) frames with \( x^t \in \mathbb{R}^{3 \times H \times W} \), where \( t \), \( H \), and \( W \) represent the time instance, height, and width of the frame. We sample two frames \( x^\tau \) and \( x^{\tau'} \), where \( 1 \leq \tau' - \tau \leq \delta \), and feed them to two architecturally identical (different weights) transformer-based networks, the patch matching network (Sec. 3.2) and patch feature extractor.

We denote the features of a frame at time \( t \) as \( x^t_{i,s} \in \mathbb{R}^{L_x \times D} \), representing \( L_x \) patches, each a \( 1 \times D \) vector denoted as \( x^t_{i,s} \) at scale \( s \in S \) and patch location \( i \). For each image patch location \( i \) and scale \( s \) in time \( \tau \), the patch matching network (Sec. 3.2) determines the set of positive indices \( P^{\tau \rightarrow \tau'}_{i,s} \) corresponding to the patch matches between \( x^\tau \) and \( x^{\tau'} \), where \( |P| \leq L \); we also form a negative set of the negative-matching indices \( N^{\tau \rightarrow \tau'}_{i,s} \) (illustrated in the supplementary material). Matched patches in \( x^\tau_s \) and \( x^{\tau'}_{s'} \) are fed to the object residual module (Sec. 3.3) to output residuals \( r^\tau_s \) and \( r^{\tau'}_{s'} \) used for the multi-view and scale-regression losses (Sec. 3.4). We note the sets of anchor, positive, and negative matched patch features as \( z^\tau, z^{\tau'} \), and \( z^{\tau - \tau'} \), and their residuals as \( r^\tau, r^{\tau'} \), and \( r^{\tau - \tau'} \), for all scales.

While DEVI requires video input for training, inference can be performed on individual frames (Sec. 3.5). The method assigns multi-scale features at each spatial location to cluster centers learned by the object residual module to
Figure 3. **DEVI training framework.** Multi-temporal frames $x^r$ and $x'^r$ are fed to patch-matching and the patch feature extractor to produce sets of anchor, positive, and negative patch features $z^{r, +}$, $z'^{r, +}$, and $z'^{r, -}$ at all scales $s \in S$, from which the object residual module generates anchor, positive, and negative residual representations $r^{+}$, $r'^{+}$, and $r'^{-}$, used in the multi-view and scale-regression losses. Positive examples are generated differently for the multi-view and scale-regression losses, as denoted with the $MV$ and $SR$ subscripts, respectively. $E$, $\phi$, and $K$ represent the feature extractor, learnt cluster, and total number of clusters respectively. Best viewed in color.

generate spatial cluster maps, from which we extract predicted bounding boxes and confidence scores.

### 3.2. Matching multi-temporal patches

The task of patch matching is closely related to the task of keypoint matching, which is commonly used for depth estimation [55], 3D reconstruction [32, 78], motion estimation [77], and more. While patch matching is too spatially sparse to be used for these downstream tasks, it provides a few notable advantages for our method: (1) patch matching has additional inter-patch, contextual information compared to a single pixel for keypoint matching, making it significantly easier to learn, and providing more stable predictions under significant view changes, and (2) patch-based architectures and direct patch matching (compared to bootstrapping keypoint matching) provide one-to-one patch correspondence with our patch-based feature extractor. These advantages allow us to train the method in an end-to-end manner by allowing fast convergence for the patch matching task, and simplifying model complexity and overhead operations by re-utilizing the same architecture for both the patch matching and patch feature extractor. The pipeline and integration of the module are illustrated in Fig. 3 and 4.

**Training.** Given input image $x$, we apply a random affine transformation, $T$, on $x$ to obtain $\tilde{x} = T(x)$. Both $x$ and $\tilde{x}$ are then fed to the patch matching network to obtain patch-wise features $z_s$ and $\tilde{z}_s$ for all scales $s \in S$. We apply the same affine transformation on $\tilde{z}_s$ to align anchor patches $T(z_s)$ with their corresponding positive patches $\tilde{z}'_s$, while all other, non-corresponding patches are considered negative. Lastly, we use a contrastive loss [63] to enforce feature similarity between anchor and positive patches, and dissimilarity between anchor and negative patches.

**Inference.** After the training procedure is performed for a small number of epochs, the network weights are frozen, and the model is federated with the main task’s training pipeline. Multi-temporal input images $x^r$ and $x'^r$, are fed to the patch matching network, producing patch-wise $\tilde{Z}_s^r$ and $\tilde{Z}'_s^r$ respectively. We then select anchor, positive, and negative match indices as $P_{i,s}^{r, +}$ and $N_{i,s}^{r, +}$, $P_{i,s}^{r, -}$ and $N_{i,s}^{r, -}$, respectively. We note the positive matches in $P_{i,s}^{r, +}$, and negative match indices in $N_{i,s}^{r, +}$ and $N_{i,s}^{r, -}$. After the training procedure is performed for a small number of epochs, the network weights are frozen, and the model is federated with the main task’s training pipeline. Multi-temporal input images $x^r$ and $x'^r$, are fed to the patch matching network, producing patch-wise $\tilde{Z}_s^r$ and $\tilde{Z}'_s^r$ respectively. We then select anchor, positive, and negative match indices as $P_{i,s}^{r, +}$ and $N_{i,s}^{r, +}$, $P_{i,s}^{r, -}$ and $N_{i,s}^{r, -}$, respectively. We note the positive matches in $P_{i,s}^{r, +}$, and negative match indices in $N_{i,s}^{r, +}$ and $N_{i,s}^{r, -}$, respectively.

Since the patch matching and patch feature extractor networks are architecturally identical, positive and negative matches indices, $P_{i,s}^{r, +}$ and $N_{i,s}^{r, +}$, have direct correspondence with the patch feature extractor outputs, $z_s^r$ and $\tilde{z}_s^r$, for the multi-temporal patch matching.

### 3.3. Object residual module

An implicit goal of any deep learning method is to cluster features that belong to the same categories together. If we consider the task of classification, and scatter image-wise features, it can be observed that examples from a given category are mapped closely together, and are far away from examples belonging to other categories. In order to measure the relative similarity of a given example and all
other examples within its own or any other category, we can use residuals. The residual of a given feature vector is the distance or similarity metric from/of that feature vector to a specific cluster center. Recent methods have used residual representation as confidence measurements for classification predictions [33, 42], intermediate soft representation for data quantization [26, 34], and mixed-surface representation as confidence measurements for classification [33, 42], intermediate soft representation for data quantization [26, 34], and mixed-surface representation learning [5]. Here, we expand upon the work proposed by [5, 42] to learn effective patch representation in highly ambiguous and/or complex environments.

Consider the large-scale patch in \( x'_{s=2} \) illustrated in Fig. 5 (in magenta), where multiple objects of different categories are in view (2 bowls and a fruit box). Given our anchor, a small scale patch sampled from \( x'_{s=1} \), only depicts one of the bowls, it would be inaccurate to enforce strict equivalence (e.g., hard assignment where both patches are labeled 1 and used as ground truth to a cross entropy loss). By utilizing soft representation and residuals, we can represent a patch by its similarity to multiple categories (i.e., clusters), which allows us to enforce multi-category similarity and learn from category ambiguous patches.

Given output feature map \( z'_i \in \mathbb{R}^{L_i \times D} \), and learned cluster centers \( \Phi \in \mathbb{R}^{K \times D} \) with \( K \) clusters, each represented by a \( 1 \times D \) vector. Ideally, clusters centers learn association with specific categories in the dataset, allowing to directly distinguish between objects. The residual of the feature vector \( z \) and cluster center \( \phi \) is defined by the distance between them, using \( r = z - \phi \). We build a patch-wise residual table, \( r_i \in \mathbb{R}^{L_i \times K \times D} \), measuring the similarity of all patches in \( z'_i \) with learned cluster centers using

\[
r_i = \sigma(\|z'_i - \phi\|_2) * (z'_i - \phi) \quad \forall \phi \in \Phi,
\]

with learnable parameters \( \theta \in \mathbb{R}^{1 \times K} \) and \( \Phi \) corresponding to residual scales and cluster centers respectively, and softmax function \( \sigma \) applied on the normalized residual. We perform this operation on patches in the anchor, positive, and negative patch features sets, \( z^{r+}, z^{r-}, \) and \( z^{r'}, \) to obtain \( r^{r+}, r^{r+}, \) and \( r^{r'}, \) where \( \{(r^{r+}, r^{r+}) \in \mathbb{R}^{1 \times K \times D} \} \) and \( r^{r'} \) are fed to a transformer-based network to produce spatially aligned anchor and positive samples, with all other, non-aligned patches considered negative examples. \( E \) represents the feature extractor. Best viewed in color.

3.4. Learning similarity across time and scale

Our loss functions aim to leverage the natural egocentric perception of human agents to sample diverse views of the same objects. As humans operate in a given environment, they often either advance towards or circumnavigate objects and elements in their surroundings. The action of circumnavigation allows us to samples multi-temporal patch matches, as described in Sec. 3.2, viewing the same objects from different view points and illumination conditions. These multi-temporal samples are then used for the multi-view loss function, \( L_{multi-view} \), maximizing similarity of features of corresponding patches. Ideally, this means that objects viewed from different viewing angles, even if visually different (as in Fig. 2), expect to generate highly similar features. We illustrate our loss functions in Fig. 5.

We also address the direct advancement action by proposing the scale-regression loss, \( L_{scale-regress} \). This loss maximizes feature similarity of a given patch and its overlapping higher-scale patch, increasing model robustness to local viewing scale. How we define positive ex-
amples varies between our proposed loss functions. For the multi-view loss, we utilize the multi-temporal patch matching predictions, while for the scale-regression loss, we use the higher-scale patch. Anchor and negative patches are defined, both loss functions (MV=multi-view, SR=scale-regression) are formulated as

\[ \mathcal{L}_{MV/SR} = \mathbb{E}_{0 \leq i \leq |\varepsilon|} \left[ \log \frac{\exp(r_i^+ \cdot r_i^-)}{\sum_{j=0}^{\varepsilon - 1} \exp(r_i^+ \cdot r_j^-)} \right] . \] (2)

3.5. Inference from learned clusters

During inference, we first perform per-batch cluster smoothing (batch can be 1 or more frames) on our learned clusters, \( \Phi \), using output features \( z' \) at time \( t \) to obtain smooth cluster centers \( t \). We use the Expectation-Maximization algorithm [21, 59] initialized with \( \Phi \) to iteratively find maximum likelihood cluster assignments for all features in \( z' \). We optimize this for \( r \) iterations (not necessarily until convergence) to produce per-batch smooth cluster centers \( \Phi \) (qualitative examples in supplementary material). This operation allows us to reduce overall noise when assigning features in \( z' \) to cluster centers \( \Phi \) to obtain cluster map \( m_t \). We separate \( m_t \) into a set of blobs using the connected components algorithm [23] and generate bounding boxes around blobs and their confidence scores. The inference pipeline is illustrated in Fig. 6.

Bounding box scoring. Object detection methods traditionally use confidence scores to rank predictions [35]. As we do not use any supervision, usual confidence scores are unavailable: we don’t define what the method should be confident in. Instead, we define confidence scores of boxes by the convexity of their corresponding blobs, following a common assumption that objects tend to have convex shapes [74]. Given cluster map \( m_t \) of frame at time instance \( t \), we use the connected component algorithm [23] on \( m_t \) to obtain a set of blobs, \( b \), and their bounding boxes. We define the confidence score of a bounding box, \( S(b_i) \) by the harmonic mean of the convexity measurement and average objectness prior of their corresponding blob \( b_i \), using

\[ S(b_i) = \frac{1 + \beta^2}{\frac{\text{Area}(b_i)}{\text{ConvexHull}(b_i)} \cdot O(b_i)} \] (3)

where \( \beta \) is a scaling factor and \( O(b_i) \) represents the mean objectness prior of blob \( b \) obtained from an off-the-shelf, self-supervised model [12].

Filtering bounding boxes. We employ classical and unsupervised methods to filter probable false positive predicted boxes, including cluster pruning [16], an objectness prior [49], and convexity thresholding. Cluster pruning is used during the cluster smoothing operation to prune clusters with less than \( \gamma \) mapped pixels. The objectness prior is obtained from an off-the-shelf self-supervised model estimating coarse foreground regions, from which predicted boxes are filtered. Lastly, we employ a threshold \( \psi \), under which bounding boxes are not considered.

4. Experiments

4.1. Datasets

We report performance on in-domain (egocentric) and out-of-domain (internet images) datasets of varying complexity. For the egocentric domain, DEV1 is trained and evaluated on Ego4D [36] and EgoObjects [65]. Ego4D provides \( \sim 3.600 \) hours of egocentric videos and is highly complex. For training, we temporally down-sample unannotated Ego4D videos to produce \( \sim 27M \) frames overall, out of which we use \( \sim 1M \). For evaluation, we use the episodic memory validation set, which provides \( \sim 9.9k \) sparsely annotated frames. EgoObjects [65] provides \( \sim 110 \) hours of egocentric videos, resulting in \( \sim 66k \) training frames, and \( \sim 7.7k \) sparsely annotated validation frames, and is largely object-centric, lower complexity dataset. We emphasize that while we train on video, inference is performed on an image level, making comparisons with other frame-based methods fair. For our out-of-domain (internet images) study (Sec. 5.2), we show our performance on the COCO [54] validation set which provides \( \sim 5k \) annotated images. Note that we do not train on COCO at any stage.

4.2. Evaluation protocol

We evaluate our method for the task of class agnostic object detection using average precision (AP) and average recall (AR), though we tend to prefer AR (particularly with more proposals) due to the non-exhaustive nature of most object detection datasets [87] making precision measurements less reliable. We use non-maximum suppression with Intersection-over-Union (IoU) threshold of 0.5. Since we use static validation datasets, we run the entire training and evaluation pipeline 5 times and report the mean performance. We note that the difference between
multi-object discovery and class-agnostic object detection depends on the evaluated dataset. Multi-object discovery generally refers to when the task is performed on the train set, while class-agnostic object detection refers to when the task is performed on the validation or test set. In this work we only consider the task of class agnostic object detection.

5. Results

Tab. 1 reports the average precision at 0.5 IoU and average recall at 1, 10, and 100 boxes per image for the class agnostic object detection task (additional discussion on metric interpretation is found in the supplemental). We compare with state-of-the-art self-supervised detection methods, LOST [73] and FreeSOLO [84], on the egocentric domain. We train and evaluate these baselines according to reported procedures on Ego4D and EgoObjects, showing the best results. We also compare with recent generic image and video representation learning methods MoCo V3 [15] and VideoMAE [76]. These self-supervised works require fine-tuning on bounding box labels for detection, which we do not assume in our setting; instead, we compare by dropping these pre-trained models in as a replacement to our self-supervised learned patch feature extractor. Our approach outperforms our baselines by up to 4.11% AP\(_50\), 0.11% AR\(_1\), 1.32% AR\(_{10}\), and 5.03% AR\(_{100}\), despite their extensive multi-stage and complex pipeline. FreeSOLO requires 3 separate training stages: self-supervised pre-training for object discovery generation (FreeMasks), training on the generated FreeMasks for pseudo-label generation, and then training on the generated pseudo labels for final predictions. This lengthy training process takes \(\sim 72\) hours of training with substantial computing resources (we use 8 Tesla V100-32GB GPUs), not including any intermediate inference or evaluation steps. In contrast, our method trains end-to-end in \(\sim 36\) hours with the same computational resources, without any pre-training or multi-training stages, achieving state-of-the-art performance in a single-stage.

Both LOST and FreeSOLO depend on a global, image-wise self-supervised pre-training procedure followed by object discovery (expanded seeded patch for LOST and FreeMask for FreeSOLO). Generic self-supervised methods like MoCo V3 and VideoMAE also tend to have global objectives. When considering dense and complex scenes, as typical in egocentric data, such pre-training strategies result in coarse features, leading to sub-optimal object discovery and class-agnostic object detection. This is supported quantitatively: As scene complexity increases, with COCO and EgoObjects on the lower end of complexity and Ego4D on the higher end, our baselines’ performances suffer significantly. In particular, generic self-supervised visual features are not fine-grained enough for detection when bounding box annotations for fine-tuning are unavailable. In contrast, by utilizing patches and residuals, our method has the feature granularity and scene ambiguity robustness to achieve state-of-the-art performance.

In Fig. 7 we present the qualitative results of our method on the EgoObjects and Ego4D datasets. Our method produces bounding boxes that align well with objects in the scene, even when scenes are highly complex. We include implementation details, ablation study, and additional qualitative of results and challenging cases in the supplemental.

5.1. Interpreting metrics

The nature of egocentric data presents challenges not only in network design and increased scene complexity, but also during pre-processing and evaluation steps such as annotations and performance analysis. Due to the high complexity of the data, distinguished by largely varying object scales, diversity, and density, annotation of egocentric videos is often sparse, only considering specific categories at specific scenes. For example, brooms might be annotated when videos are captured in a kitchen, but not annotated when captured in a parking lot. This results in sparsely annotated datasets, which may alter the traditional view on performance metrics. While both recall and precision are affected by the sparse annotation problem, we note that pre-

<table>
<thead>
<tr>
<th>Method</th>
<th># stages</th>
<th>AP(_50)</th>
<th>AR(_1)</th>
<th>AR(_{10})</th>
<th>AR(_{100})</th>
<th>AP(_50)</th>
<th>AR(_1)</th>
<th>AR(_{10})</th>
<th>AR(_{100})</th>
</tr>
</thead>
<tbody>
<tr>
<td>Selective Search [79]</td>
<td>1</td>
<td>0.12</td>
<td>0.00</td>
<td>0.86</td>
<td>4.15</td>
<td>0.09</td>
<td>0.00</td>
<td>0.88</td>
<td>3.98</td>
</tr>
<tr>
<td>LOST [73]</td>
<td>4</td>
<td>2.13</td>
<td>0.89</td>
<td>6.99</td>
<td>9.41</td>
<td>0.82</td>
<td>0.43</td>
<td>1.42</td>
<td>6.67</td>
</tr>
<tr>
<td>FreeSOLO [84]</td>
<td>5</td>
<td>15.70</td>
<td>30.30</td>
<td>20.70</td>
<td>32.90</td>
<td>2.40</td>
<td>2.80</td>
<td>12.80</td>
<td>17.00</td>
</tr>
<tr>
<td>TimeCycle [83]</td>
<td>1</td>
<td>6.94</td>
<td>4.24</td>
<td>11.86</td>
<td>12.32</td>
<td>2.95</td>
<td>1.52</td>
<td>6.29</td>
<td>6.97</td>
</tr>
<tr>
<td>VideoMAE [76]</td>
<td>1</td>
<td>9.25</td>
<td>5.16</td>
<td>16.08</td>
<td>16.47</td>
<td>4.05</td>
<td>2.19</td>
<td>8.42</td>
<td>8.96</td>
</tr>
<tr>
<td>DEVI (Ours)</td>
<td>1</td>
<td>14.96</td>
<td>6.47</td>
<td>29.61</td>
<td>39.43</td>
<td>6.51</td>
<td>2.91</td>
<td>14.12</td>
<td>22.03</td>
</tr>
</tbody>
</table>

Table 1. Quantitative results on the egocentric domain. Average precision (AP) and average recall (AR) on EgoObjects [65] and Ego4D [36] validation sets. DEVI outperforms other self-supervised methods for the task of class agnostic object detection, despite the baselines’ increased model complexity and multi-stage procedure. We note the number of stages methods require before final inference.
Figure 7. Qualitative results of DEVI on EgoObjects (top three rows) and Ego4D (bottom three rows) validation sets. It can be observed that our method has a strong notion of objectness and is able to detect most objects in scenes, even when they are highly complex.

Precision is more noticeably affected. That is due to the high number of un-annotated objects in frames, leading to many false positives. As the number of false positives increases, the precision decreases \((P = \frac{TP}{TP + FP})\). For that reason, in this work, we place higher importance on recall performance.

### 5.2. Ablation studies

**DEVI components.** To understand what makes DEVI effective, we investigate the impact of various model design choices to overall performance. We study all possible combinations of the loss functions, \(L_{MV}\) and \(L_{SR}\), and the Object Residual Module (ORM), and report performance in Table 2. Note that when the ORM is not used, we use K-Means clustering [56] on the raw features instead. Note that at least one loss function is required for training.

We observe worse overall performance without the ORM, which implies its increased utility in ambiguous scenes compared to classical clustering methods such as K-Means. Then, just by incorporating our multi-view loss, \(L_{MV}\), with the Object Residual Module, we already outperform our baseline by 1.62% AP\(_{50}\). We then further improve our performance by adding the scale-regression loss, \(L_{SR}\), component. All combinations were trained for the same number of iterations, and with the same hyperparameters.

<table>
<thead>
<tr>
<th>ORM</th>
<th>(L_{MV})</th>
<th>(L_{SR})</th>
<th>AP(_{50}) (%)</th>
<th>AR(_{10}) (%)</th>
<th>AR(_{100}) (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>✓</td>
<td>1.12</td>
<td>2.46</td>
<td>8.58</td>
<td></td>
</tr>
<tr>
<td></td>
<td>✓</td>
<td>1.96</td>
<td>4.51</td>
<td>11.87</td>
<td></td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
<td>2.00</td>
<td>4.10</td>
<td>15.40</td>
<td></td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
<td>2.39</td>
<td>3.31</td>
<td>16.08</td>
<td></td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
<td>4.02</td>
<td>7.92</td>
<td>21.34</td>
<td></td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
<td>6.51</td>
<td>14.12</td>
<td>22.03</td>
<td></td>
</tr>
</tbody>
</table>

**Egocentric vs. exocentric training data.** While DEVI is designed with egocentric properties in mind, we also investigate the utility of our method on exocentric videos. This aims at verifying our computational appearance-based approach which utilizes the varying viewing angles and illumination conditions of objects in egocentric videos, which may not exist in exocentric videos. Due to the commonly stationary viewing angles captured in exocentric videos, we expect reduced efficacy of output features when trained on exocentric data. For this experiment, we train our model on \(\sim 1M\) frames from the YouTubeBB-8M video dataset [1] and evaluate on Ego4D validation set. We validate our hypothesis by showing significantly increased performance when trained on egocentric data than on exocentric data,
Figure 8. Varying viewing angle and illumination conditions. We observe consecutive frames and their corresponding pre-smoothing cluster masks. As the viewing direction changes, we observe that objects retain their cluster assignments, indicating feature consistency regardless of viewing direction and illumination conditions. Colors are random. Best viewed in color.

Table 3. Out of domain study. We test our method on unseen, out of domain data to study its generalizability. Note that the baseline methods are trained on COCO while we are not. † corresponds to performance achieved through independent experiments.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>COCO Validation Set [54]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method</td>
<td>AP50</td>
</tr>
<tr>
<td>UP-DETR [17] CVPR21</td>
<td>0.00</td>
</tr>
<tr>
<td>Selective Search [79] ICCV13</td>
<td>0.50</td>
</tr>
<tr>
<td>DETReg [6] CVPR22</td>
<td>3.10</td>
</tr>
<tr>
<td>LOST† [73] BMVC21</td>
<td>4.73</td>
</tr>
<tr>
<td>FreeSOLO† [84] CVPR22</td>
<td>9.60</td>
</tr>
<tr>
<td>FreeSOLO† [84] CVPR22</td>
<td><strong>12.20</strong></td>
</tr>
<tr>
<td>DEVI (Ours)</td>
<td><strong>8.03</strong></td>
</tr>
</tbody>
</table>

improving AP50 by +3.86%, AR1 by +1.42%, AR10 by +8.80%, and AR100 by +16.17%.

Out of domain study. We investigate generalizability to out-of-domain datasets by training DEVI on the Ego4D dataset and evaluating on the COCO validation set [54]. We compare DEVI to recent, state-of-the-art class agnostic object detection methods. Note that while our baselines are trained on the COCO training set, our method is not exposed to any COCO data at any stage. We include reported performance, if available, and performance obtained through our independent experiments (indicated by †). We note that LOST [73] only officially reports performance on the train set, while here we report on the validation set. Despite the domain misalignment, our method is able to achieve competitive performance, outperforming LOST by 3.30%, 1.32%, 11.77%, and 17.70% on the AP50, AR1, AR10, and AR100 metrics respectively (Tab. 3). We also demonstrate competitive performance compared to our independent experiments of FreeSOLO. Qualitative results on the COCO validation set are shown in Fig. 9 and supplementary material.

Varying viewing angles and illumination. We qualitatively visualize our robustness to changes in viewing angle and illumination conditions in Fig. 8; we expect similar cluster assignments for objects across frames. To verify, we visualize the pre-smoothing cluster masks, as it bypasses random elements from smoothing. It can be seen that despite the viewing direction of objects changes (also affecting illumination), the method is still able to retain consistent cluster assignments. Additional examples are provided in the supplementary material.

6. Conclusion

We have introduced DEVI, a self-supervised class agnostic object detection method for the egocentric domain. We utilize natural human movement patterns to sample views of objects for our computational appearance inspired method, demonstrating that our proposed multi-view and scale-regression losses enable our method to learn robust invariance to viewing angles and illumination conditions. We also show that our object residual module allows learning of effective features in highly complex and ambiguous scenes. Lastly, we achieve state-of-the-art performance on class agnostic object detection on egocentric datasets in a single, end-to-end stage, eliminating lengthy, multi-stage, and computationally expensive procedures.
References


[25] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner,


[76] Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. Videomae: Masked autoencoders are data-efficient learners


