Time Does Tell: Self-Supervised Time-Tuning of Dense Image Representations

Salehi, M.; Gavves, E.; Snoek, C.G.M.; Asano, Y.M.

DOI
10.1109/ICCV51070.2023.01516
10.48550/arXiv.2308.11796

Publication date
2023

Document Version
Final published version

Published in
2023 IEEE/CVF International Conference on Computer Vision

License
Article 25fa Dutch Copyright Act (https://www.openaccess.nl/en/in-the-netherlands/you-share-we-take-care)

Citation for published version (APA):

General rights
It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content license (like Creative Commons).

Disclaimer/Complaints regulations
If you believe that digital publication of certain material infringes any of your rights or (privacy) interests, please let the Library know, stating your reasons. In case of a legitimate complaint, the Library will make the material inaccessible and/or remove it from the website. Please Ask the Library: https://uba.uva.nl/en/contact, or a letter to: Library of the University of Amsterdam, Secretariat, Singel 426, 1012 WP Amsterdam, The Netherlands. You will be contacted as soon as possible.

UvA-DARE is a service provided by the library of the University of Amsterdam (https://dare.uva.nl)
**Time Does Tell: Self-Supervised Time-Tuning of Dense Image Representations**

Mohammadreza Salehi, Efstratios Gavves, Cees G. M. Snoek, Yuki M. Asano
QUVA Lab, University of Amsterdam
{s.salehidehnavi, e.gavves, c.g.m.snoek, y.m.asano}@uva.nl

**Abstract**

Spatially dense self-supervised learning is a rapidly growing problem domain with promising applications for unsupervised segmentation and pretraining for dense downstream tasks. Despite the abundance of temporal data in the form of videos, this information-rich source has been largely overlooked. Our paper aims to address this gap by proposing a novel approach that incorporates temporal consistency in dense self-supervised learning. While methods designed solely for images face difficulties in achieving even the same performance on videos, our method improves not only the representation quality for videos – but also images. Our approach, which we call time-tuning, starts from image-pretrained models and fine-tunes them with a novel self-supervised temporal-alignment clustering loss on unlabeled videos. This effectively facilitates the transfer of high-level information from videos to image representations. Time-tuning improves the state-of-the-art by 8-10% for unsupervised semantic segmentation and achieves 1-2% on semantically dense tasks. The method starts with a 2D encoder pretrained on images and fine-tunes it using unlabeled videos. This approach leads to improved performance not only for videos but also for images. The implementation can be found here: https://github.com/SMSD75/Timetuning

**1. Introduction**

Dense self-supervised learning, whereby meaningful deep features for each pixel or patch of input are learned in an unsupervised manner, has recently received increasing attention [74, 24, 57, 51]. By learning spatially consistent features for different views of an input, strong gains in unsupervised semantic segmentation have been achieved using unlabeled images. However, so far, an even more information-rich source for unsupervised training has been largely overlooked: videos. With their additional time dimension and being the most rapidly growing form of digital content, they are well-suited to scaling dense self-supervised learning even further.

Some efforts have already been made to learn from the video domain by using different frames from a video as augmentations [18, 63, 49] or by involving temporal correspondence [71, 58]; however, they mostly did it in a supervised way [29, 73, 19, 40, 65, 38, 31, 28, 41, 37, 54, 44], which is not scalable specifically for dense tasks where the number of targets can increase significantly as the number of pixels grows. To this end, self-supervised learning approaches offer a solution by reducing the need for supervision. However, these methods typically rely on the notion of “views”, which involves learning similar features for corresponding locations over time. This usually leads to a chicken-and-egg problem, where the correspondences are required for learning dense features – which in turn enable good correspondences [30].

Figure 1: **Time-tuning compared to previous methods.** Unlike existing methods that ignore or utilize expensive 3D models to implicitly model temporal information, the proposed method explicitly incorporates temporal consistency in dense feature representations using a temporal self-supervised loss. The method starts with a 2D encoder pretrained on images and fine-tunes it using unlabeled videos. This approach leads to improved performance not only for videos but also for images.
sections. As we demonstrate through a new evaluation protocol, using image models on frames alone is not nearly as effective. A similar finding has also been reported, albeit for non-dense works [20, 34], which assumed the passage of time as an image augmentation. These works have generally reported reduced performances, even when compared to simple image-based pretraining methods. Similarly, video-level tasks [17, 18, 55] assume sufficiently similar semantics between different frames. This is also not true for dense tasks, as static features can only be assumed where nothing is moving – which is rare due to possible object, camera, and background motion between the frames.

To address this challenge, we propose to model the additional time-dimension explicitly to identify which pixels should retain similar embeddings and which should not. We propose two separate modules to tackle the correspondence and the dense learning, respectively. For the former, we introduce the Feature-Forwarder (FF) module, which breaks the mentioned chicken-and-egg loop by leveraging the good tracking performance of pretrained image models, and allows an approximate second “view” that can then be treated as a target for the further dense self-supervised loss. On top of this, we introduce a spatio-temporally dense clustering module, which learns unsupervised clusters across samples, locations, and time. Using these two components and starting from image-pretrained features, our proposed method allows time-tuning (TiMET) the dense representation in a self-supervised manner, see Figure 1.

Finally, we demonstrate that TiMET paves the way for further scaling of self-supervised learning by leveraging the abundant availability of video datasets and transferring their knowledge to the image domain. This results in consistently achieving state-of-the-art performances not only for the task of unsupervised semantic segmentation of videos, but also for unsupervised image semantic segmentation, a feat previously out of reach for methods trained on videos.

Overall, this paper makes the following contributions:

- We show that image-based unsupervised dense segmentation models applied to videos exhibit degraded performance and lack temporal consistency in their segmentation maps.

- Building on this observation, we propose a novel dense self-supervised learning method that utilizes temporal consistency as a learning signal.

- We demonstrate that our method enables the scaling of self-supervised learning by leveraging abundant video datasets and effectively transferring knowledge to the image domain. Our approach consistently achieves state-of-the-art performance for both images and videos, opening up new opportunities in the field.

2. Related Works

Dense self-supervised learning. These methods build upon image-level self-supervised representation learning by incorporating existing losses to enhance the spatial features, demonstrating a commendable advancement. DenseCL [61] works on spatial features by constructing dense correspondences across views using the contrastive objective given in MoCo [23], while PixPro [62] utilizes the augmentation wrapper to get the spatial correspondence of the pixel intersection between two views. Similarly, MaskContrast [57], Leopart [74], DetCon [24] and Odin [25] also ensure spatial feature similarities via contrastive learning and spatial correspondences. Self-Patch [67] treats the spatial neighbors of the patch as positive examples for learning more semantically meaningful relations among patches. Inspired by SelfPatch, ADCLR [69] proposes patch-level contrasting via query crop and cross-attention mechanism. Pursuing the same objective through none end-to-end approaches. Both [68] and [59] propose an unsupervised salient object segmentation pipeline that extracts noisy object masks from the inputs and fine-tunes a specifically designed object segmentation head with several self-training steps to make an unsupervised semantic segmentation model. Unlike the existing image-based approaches, we propose a method that improves the dense prediction performance of a pretrained encoder by explicitly modeling the temporal dimension and learning from diverse natural dynamics and variations found in videos.

Video to image knowledge transfer. Learning from videos instead of images has recently received attention since they contain far more information than still images and hold the potential for learning rich representations of the visual world. To this end, several non-dense works have been released [46, 4, 27]. VITO [46] shows that naïvely applying image domain self-supervised learning methods on videos can lead to a performance drop coming from the distribution shift, which by applying data processing techniques is relieved but not fully solved. Following the same way, [27] trains masked autoencoders [22] with contrastive learning [10] on video datasets and shows a decent performance on both image and video tasks. While video to image knowledge transfer is the end goal of such methods and our paper; our dense task is different and is as of yet unaddressed.

Unsupervised video object segmentation. Unsupervised video object segmentation methods do not require any manual annotations. However, they are only designed to tackle a foreground/background segmentation task, which refers to the segmentation of the most prominent, general objects in video sequences [66, 43, 50, 1, 39]. For instance, [66]
Our method works by densely clustering features in a manner that is consistent with time. At a high level, it works by transforming features from a past time manner that is consistent with time. At a high level, it works for a given video, and in detail.

Architecture. In the following, we describe each component to learn from videos. Figure 2 shows the proposed method’s final comes from the dense temporal clustering of forwarded.

3.1. Feature Forwarding

The goal of this component, the feature-forwarder (FF), is to propagate the dense features in time as accurately as possible, given some RGB input frames. More formally, let $f$ be the feature forwarding function, $I_t$ be a frame at time $t$ for a given video, and $z_t$ the features corresponding to $I_t$, then the task is to propagate $z_t$ to a future time $T$,

$$f([I_t, \ldots, I_T], z_t) \rightarrow \hat{z}_T.$$  \hfill (1)

Note that $\hat{z}_T$ and $z_T$ are not necessarily the same for $T > t$; since $\hat{z}_T$ represents an approximation of $z_T$ using the previous features. Obviously, the further $T$ is from $t$, the more variance is seen and therefore a better learning signal might be provided; however, for such larger intervals, the idea of feature forwarding encounters two main challenges: error accumulation and object non-permanence.

Challenge 1: Error accumulation. While computing how features have changed simply based on the start and end frames is the most straight-forward (i.e., $I_t$ and $I_T$), it does not fully leverage the knowledge contained in the intermediate frames. Instead, it is common [30, 60, 36] to forward features for every time-step $\delta t$, i.e.,

$$f([I_t, \ldots, I_T], z_t) = \bigcirc_{T=\delta t}^T f([I_{t+\delta t}, I_{t'}], z_{t'}),$$  \hfill (2)

where $\bigcirc$ is a composition operator such as element-wise multiplication. However, this results in the accumulation of errors over time.

Challenge 2: Object non-permanence. One difficulty that arises when using videos as training data is the fact that objects sometimes simply disappear from the screen. Compared to images, which carry a heavy photographer's bias, videos are more prone to natural variation due to camera and object motion resulting in temporary object non-permanence in a particular video clip. A method that simply assumes an object (or a feature) has to be present in any given frame simply because it was present in the previous

Figure 2: Time-Tuning overview. We conduct self-supervised video semantic segmentation by tuning image-pretrained models with temporal information from videos. (a): Our general pipeline of adapting an image-pretrained ViT transformer on video data using our dense clustering loss. The encoder is specified by $\Psi$ and kept frozen except for the last two layers. Also, $z_t$ shows the output of Sinkhorn-Knopp algorithm, which is forwarded to time-step $T$ to be compared with the features that are obtained using the last time-step. (b): Detailed view into our Feature-Forwarder module that is used for aligning cluster-maps from past frames to logit-features of future ones.
one might therefore easily fail when going towards videos that exist in-the-wild. A case in point are simple occlusions that arise from an object being fully- or partially-covered by another one further in the background. Taking longer training clips has been attempted to resolve the issue. However, this increases the training complexity, especially for dense-tasks that necessitate a higher number of predictions for each input.

We address these challenges using a novel, yet simple component, which we call the Stabilizing Feature-Forwarder. The first choice in designing this module is the function and data used for composing information across time. For instance, a simple approach would be to use pixel-wise optical flow and aggregate the flow at the feature level to predict how the features change over time. However, as we also show in our experiments, optical flow is prone to accumulating error with time and also cannot easily recover from occlusions [6]. Instead, we utilize the fact that we wish to pretrain the visual encoder and recycle part of it for the purpose of feature forwarding. This not only speeds up the process, as no additional encoder/modality is required, but also produces a feedback loop of better forwarding and better feature learning.

Concretely, we first compute the L2-normalised dense features of a pretrained visual encoder $\Phi$ to compute spatial similarities across time (the $t'$ in $t'$ left out for clarity):

$$F_{ij}^t = \langle \Phi(I_t)_i, \Phi(I_{t'})_j \rangle / \tau,$$

(3)

which yields matrices $F^{t \ldots T - \delta t} \in R^{N \times N}$ with values between [0,1] indicating semantic similarities between the $N$ spatial features that are sharpened by a temperature $\tau$. Note that because we compare each frame with the final frame, the effect of object non-permanence can be minimized: even if the object is only present in every other frame, there is enough signal to forward its features.

Next to propagate from time $t$ to $T$, these similarities are stacked and normalised along time, as follows:

$$F_{ij}^{t'} = \exp(N(F_{ij}^t)) / \sum_{i', t''} \exp(N(F_{i'j}^{t''})),$$

(4)

where $t \leq t' \leq T$ and $N$ is a neighborhood thresholding method which forces short-term spatial-smoothness, i.e., $N(F_{ij}^t)$=0 if $i$ is not within a local window around $j$ with the size $k$. Finally, the propagated feature is computed $\tilde{z}_T$ as:

$$\tilde{z}_T(j) = \sum_{t', i} F_{ij}^{t'} \tilde{z}_{t'}(i), \quad j \in \{1, \ldots, N\}.$$  

(5)

This means that for arriving at the target feature $\tilde{z}_T$, we not only use the previous frame as the source, but instead use the past $(T - t)/\delta t$ frames and aggregate these.

### Relation to mask-propagation methods.

While previous methods such as DINO [9] and STC [30] have utilized a similar technique for propagating ground-truth masks for evaluating, for example on the DAVIS dataset, there are three key differences to our forwarding method. First, instead of propagating binary maps of foreground-vs-background, our feature-forwarder has as inputs soft, noisy and multi-label self-supervised segmentation maps, which require the forwarder to tolerate overlapping of different object probabilities throughout training. Second, previous methods have used this approach mainly for inference; we, however, use this module as a trainable component and show how our loss improves this forwarder with training time. Finally, we do not follow a typical re-normalizing step across feature dimensions (typically done after Eq. (5)) as this harms the scale of logits that are being propagated and leads to heavily diluted target distributions.

### 3.2. Self-supervised dense clustering

While the Feature-Forwarder produces target features that include information about the dynamics with time, its computation utilizes $\Phi(I_T)$ and could lead to trivial solutions, i.e., $f([I_t, \ldots, I_T], z_t) = \Phi(I_T)$. To counteract this, we propose a self-supervised clustering task across views in time. For this, we utilize the basic online clustering algorithm based optimal-transport [12], utilized in works such as SeLa [2], SwAV [8] and DINOv2 [45]. In particular, let $\Psi$ be a visual encoder with a clustering head $g$ that yields a $K$ dimensional output, then the clustering loss is given by:

$$\mathcal{L}(x_i) = -\hat{y}_i \log(g(\Psi(x_i))),$$

(6)

which is a standard cross-entropy loss with regards to self-supervised pseudo-label $\hat{y}_i$. These labels, in turn, are generated by solving an entropy-regularised optimal transport problem on the batch $B$ [2]:

$$\min_{\hat{y}} (\hat{y}, -\log g(\Psi(x))) + \frac{1}{\lambda} \text{KL}(\hat{y}||rC^T),$$

(7)

with $r = \frac{1}{K} \mathbb{1}, \quad c = \frac{1}{|B|} \mathbb{1}.$

(8)

Here $\lambda$ controls the entropy regularisation and $r,c$ are the marginals for the prototypes and the batch, respectively. Note that solving this problem can be done extremely quickly on the GPU and yields soft pseudo-labels $\hat{Y}$, such that $\argmax(\hat{Y}) = \hat{y}$. These might yield sufficiently similar semantics [17] this is not true for dense tasks, as static features can only be assumed for videos where nothing is moving, which is rare.

### 3.3. Overall TIME loss

We combine the previous two modules to arrive at our full method (as shown in Fig. 2): First, self-supervised
Sinkhorn-Knopp clustering is conducted on early features \( g(\Psi(I_t)) \), yielding soft pseudo-labels \( SK(g(\Psi(I_t))) = \tilde{Y}_t \). These are then forwarded in time using our Feature-Forwarder to arrive at dense targets \( FF(\tilde{Y}_t) \), which are used in the final loss. Compactly:

\[
L_{\text{TIME}}(I_T) = - \sum_{i,j} FF(\tilde{Y}_t) \log(g(\Psi(I_T))). \tag{9}
\]

4. Experiments

4.1. Setup

Datasets. We train our method and baselines on YTVOS [64], one of the largest video segmentation datasets available, and evaluate on DAVIS17 [48] and YTVOS. For YTVOS the ground truth masks are only available for the first frames of the test and validation sets, and therefore, a fixed random 20% of the training set is used for testing. For transfer learning experiments, we use the validation set of Pascal VOC 2012 [16]. As the dataset has been commonly used as a main reference for recent works in dense self-supervised image segmentation [74, 57, 61]. For completeness, we also report the performance on egocentric datasets that have less object-centric bias and are prevalent in real-world scenarios. For egocentric experiments, we train on EPIC-KITCHENS-100 [13] and evaluate on VISOR [14]. Further details are provided in Appendix A.

Models and baselines. Currently, there is a lack of available unsupervised semantic video semantic segmentation methods. Nevertheless, we have included a comprehensive comparison of our method with state-of-the-art techniques in both image and video domains for unsupervised image semantic segmentation and unsupervised video object segmentation. To evaluate the image-based models, we utilized either official reported numbers or provided pretrained models of STEGO [21] and Leopart [74]. We have taken every measure to ensure fairness in our comparison. To do so, all the used pretrained models have the same pretraining dataset (ImageNet-1k), and the number of their backbone parameters is roughly similar. For those models that use extra datasets, for instance, Leopart, we select the pre-trained backbones that closely match the specification of YTVOS training dataset. Additionally, we trained image-based models on the same video datasets, where we converted video data into their corresponding image data and reported their performance wherever possible. To ensure comprehensive analysis, we have included the recent concurrent work Flowdino [71], which utilizes optical flow to refine DINO features on unlabelled videos, in our comparison benchmarks as well. This is done only in cases where there is a shared experimental setup.

Evaluation procedure. Despite unsupervised object segmentation being a well-established evaluation in the image domain [57, 74], evaluating unsupervised video multi-label object segmentation is challenging due to the absence of an established evaluation protocol for video object semantic segmentation without supervision. In this regard, we propose a set of evaluation protocols for unsupervised video multi-label object segmentation, which exploits existing video object segmentation datasets for evaluation purposes (see details in Appendix B). In our experiments, we discard any projection head used during training and evaluate ViT’s spatial tokens directly, similar to [57, 32, 61], using four methods: classification with a linear head, classification with an FCN head, over-clustering, and clustering with as many clusters as the number of ground truth objects. To fine-tune different heads on top of the frozen spatial tokens, we follow [57]. For unsupervised semantic segmentation, we apply K-Means on all spatial tokens, where \( K \) is chosen to be higher or equal to the ground-truth number of classes, following the common protocol in image clustering [32, 56]. For grouping clusters to ground-truth classes, we match them either by pixel-wise precision or Hungarian matching on merged cluster maps [35]. For video datasets, we allow the matching to be per-frame, per-video, and per-dataset. The evaluation metric is permutation-invariant, following [32], and the results are averaged over five different seeds. Clustering evaluations are preferred as they require less supervision and fewer hyperparameters than training a linear classifier and operate directly on the learned embedding spaces. We report our results in mean Intersection over Union (mIoU) unless otherwise specified.

Model training. We train a ViT-Small with patch size 16 and initialized from ImageNet-pretrained DINO weights [9] using the proposed self-supervised loss function. For the ablations and the main experiments, our models are trained for 12 and 30 epochs respectively. Further training details are provided in Appendix A. Code will be made available.

4.2. Ablations

We first examine the essential parameters of our method by training \( \text{TIME} \) on YTVOS and assessing its ability to perform semantic segmentation on Pascal VOC. In addition to presenting the results through clustering and over-clustering, we also demonstrate linear classification outcomes.

Number of prototypes. We first ablate the influence of the number of prototypes on downstream segmentation performance. Results in Table 1a indicate a sharp increase in performance with a rise in the number of prototypes, but once a sufficiently large number is reached, performance stabilizes. We observe peak performance with a moderate number of 200 prototypes. The stability of our method over
a range of prototypes (50-300) suggests that our approach is robust to the change of this parameter.

Understanding FF. Next, we ablate the temporal dimensions that influence the working of the Feature-Forwarder. In Table 1b, we vary the time interval $\delta T$ between frames whilst keeping a fixed number of four frames. We generally find an increase in performance when increasing the time-interval, with performance peaking at 0.5s. One of the most critical observations is the clear difference in clustering performance from 26.2% to 42.8% between training on single frames (first row) vs. training with multiple frames. This clearly indicates the significant benefit of utilizing temporal information from the Feature-Forwarder during training.

Next, in Table 1c we vary the number of frames given a fixed clip duration of 2s. Again, we find that a moderate number of frames between the source and target time is most favorable. While additional frames do not degrade the performance much, they do add computational complexity, so we prefer using 4 frames in our main method.

Choice of propagation feature. Finally, in Table 2, we vary the type of feature that we forward to a future time. In the first row, we report the performance when not using any feature forwarding – thus solely training on single-frame inputs (i.e., the same as row 1 in Table 1c). In the second row, “Identity” shows the performance that is obtained when the feature map from the source frame is simply forwarded to the future without any changes, which shows an increase in the performance compared to the first row. This shows that often, training videos are not very dynamic, and a static assumption can already lead to some gains. However, compared to forwarding the Sinkhorn-Knopp (SK) clustered features, these gains are small (+4.6% vs. +17%). Importantly, our Feature-Forwarder relies on forwarding SK-sharpened feature maps and does not work when simply forwarding the network’s output logits $\Phi(x)$, followed by a clustering step. The reason for obtaining considerably lower numbers here can be traced back to using an un-entropy-regularized clustering algorithm. In this case, the logits tend to cause a few prototypes to dominate the cluster centers, exacerbated after the propagation, resulting in highly noisy and uninformative propagated logits. This shows that careful design of the Feature-Forwarder is indeed required.

Applying TIME to different models. As shown in Table 3, our method is generalisable to different backbones and self-supervised learning initializations, enabling it to be an effective method to transfer the knowledge of the videos to images in an unsupervised way, which reduces the cost of labeling for different downstream tasks.
Figure 3: **TImET unsupervised segmentations.** As we regularize DINO’s backbone to be consistent across time on YTVOS, it obtains strong performance on both image and video segmentation datasets, yielding high class consistencies (indicated by the segmentation colors) and tight borders. We provide more qualitative results in Appendix D.

<table>
<thead>
<tr>
<th>Clustering</th>
<th>Overclustering</th>
</tr>
</thead>
<tbody>
<tr>
<td>YTVOS</td>
<td>DAVIS</td>
</tr>
<tr>
<td>F</td>
<td>C</td>
</tr>
<tr>
<td>--------------</td>
<td>--------</td>
</tr>
<tr>
<td>Resnet50</td>
<td>44.0</td>
</tr>
<tr>
<td>SwAV [8]</td>
<td>39.5</td>
</tr>
<tr>
<td>DINO [9]</td>
<td>39.1</td>
</tr>
<tr>
<td>Leopart [74]</td>
<td>39.2</td>
</tr>
</tbody>
</table>

Table 4: **Unsupervised video semantic segmentation results** in mIoU for clustering (K=GT) and overclustering. TImET not only gets better numbers on YTVOS, but also achieves considerably better results on DA VIS. This shows the better quality of the learned features at transferability to other video datasets. The clip-length is set to 16 and 4 for DA VIS and YTVOS, respectively. For clustering, the Hungarian algorithm [35] matches the unsupervised segmentation clusters (K) with the ground truth (GT) per frame (F), clip (C) or across the whole dataset (D). For overclustering, we use K=10 for the frame-wise (F) and clip-wise (C) evaluations, and for dataset-wise (D) evaluation, K=200 for DA VIS and K=500 for YTVOS. *denotes finetuning pretrained models on the same video frames as input as our method. The matching protocol is greedy many-to-one, see Appendix B for details.

### 4.3. Large-scale experiments

**Unsupervised video semantic segmentation.** In this section, we train our TImET method on the YTVOS dataset and evaluate it for unsupervised video object segmentation on both DA VIS and YTVOS. The results are shown in Table 4. It should be noted that the number of prototypes we analyzed in Table 1a is not the number of clusters used to report accuracies for Table 4, as the actual number is usually unknown in the case of unsupervised learning. For the clustering experiment, we set K to the number of ground truth objects, which makes evaluation metrics easier to interpret and is commonly done in image clustering and segmentation. This evaluation is repeated but for larger numbers of clusters, in the “overclustering” scenario. Note that in dense self-supervised learning, overclustering can be particularly important because the learned representations are used as a feature extractor for downstream tasks such as semantic segmentation or object detection. By using more fine-grained representations, the network may be able to extract more discriminative features such as object parts [74], which can lead to better performances.

From Table 4, we observe a clear trend: our method, trained on YTVOS not only achieves superior performances on YTVOS, but also beats existing image-trained models on DA VIS by a large margin. In particular, the state-of-the-art self-supervised clustering method, Leopart, has 4% lower performance on per dataset DA VIS evaluation for K=GT and 1.6% lower performance on YTVOS. The gap even becomes larger when Leopart is trained on the same video dataset with 8% to 16% lower numbers in per dataset and per clip numbers across different datasets. This means that when grouping objects of the same class over time and
the whole dataset, the image-based self-supervised methods have trouble generalising to videos, where objects are not centered in the frame, can appear in varying poses, and are more difficult in general. A strong contender is Leopart, which matches the performance of DINO for per frame and per clip clustering, however, improves considerably with the per dataset clustering. We attribute this to the fact that their dense learning objective improves mainly the generalisation capabilities of the learned representation, however falling short of generalising to temporal variations. We make similar observations for overclustering, improving the DINO baseline which we utilize as initialisation by 12% to 14% across different datasets in the per dataset metric.

We conclude that the proposed method outperforms all other methods in learning robust and discriminative representations for dense video understanding, and image-based self-supervised learning may not have sufficient generalisation capacity. This shows that if used right, time is a particularly strong inductive bias for self-supervised learning. Figure 3 shows segmentations returned by the proposed model trained on YTVOS and tested on YTVOS, DA VIS, and VISOR respectively. The proposed method groups objects accurately, and importantly, the frame segmentations are considerably consistent over time (see more examples in Appendix D). This highlights the importance and relevance of temporal fine-tuning, not just for higher accuracy, but crucially for consistency and robustness.

Transfer from video training to images. Despite the common belief that features learned from videos perform worse when transferred to images [34, 20], the results in Table 5 demonstrate our method achieves high performance that match state-of-the-art methods directly trained on images, specifically for the K=500, FCN, and LC metrics. We also compare our performance gains in videos against models designed for unsupervised image segmentation. The results show that models highly biased towards image datasets cannot provide the same performance when trained on videos, lagging behind our approach by 7% to 30%. Our results demonstrate that achieving high transfer performance on challenging tasks through video-based self-supervised learning is not only feasible but can also maintain high performance across modalities. These findings suggest that our method can drive further advances in self-supervised learning and inspire new directions for research in this field. Figure 3 shows the qualitative results on Pascal VOC. For more visualisations we refer to Appendix D.

Salient object segmentation. In Table 6, we compare foreground masks obtained with various DINO ViT based methods. We use the cluster-based foreground extraction protocol from [74] (details provided in Appendix C). First, we find that our method outperforms the original DINO at-

<table>
<thead>
<tr>
<th></th>
<th>Pascal VOC</th>
<th>DAVIS</th>
<th>YTVOS</th>
</tr>
</thead>
<tbody>
<tr>
<td>DINO [9]</td>
<td>52.1</td>
<td>34.5</td>
<td>32.1</td>
</tr>
<tr>
<td>Leopart [74]</td>
<td>59.6</td>
<td>37.3</td>
<td>38.6</td>
</tr>
<tr>
<td>STEGO [21]</td>
<td>49.1</td>
<td>30.4</td>
<td>32.1</td>
</tr>
<tr>
<td>TIME T (ours)</td>
<td><strong>63.9</strong></td>
<td><strong>44.5</strong></td>
<td><strong>43.5</strong></td>
</tr>
</tbody>
</table>

Table 6: Salient object segmentation. We report performance using the Jaccard score [9] and use the official pretrained models for evaluation.
Table 7: **Generalisation to egocentric datasets.** Unsupervised semantic segmentation results in mIoU for clustering (K=GT) on VISOR [14] after training on EPIC-Kitchens-100 [13]. The clip-length is set to 4. Our method gets better results on egocentric datasets as well.

<table>
<thead>
<tr>
<th></th>
<th>Encoder</th>
<th>Params</th>
<th>mIoU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trained on Images</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Supervised</td>
<td>ViT-S16</td>
<td>21M</td>
<td>35.1</td>
</tr>
<tr>
<td>DINO* [9]</td>
<td>ViT-S16</td>
<td>21M</td>
<td>47.9</td>
</tr>
<tr>
<td>CrOC* [52]</td>
<td>ViT-S16</td>
<td>21M</td>
<td>50.0</td>
</tr>
<tr>
<td>Leopart* [74]</td>
<td>ViT-S16</td>
<td>21M</td>
<td>63.6</td>
</tr>
<tr>
<td>DINO [9]</td>
<td>ViT-B16</td>
<td>86M</td>
<td>55.9</td>
</tr>
<tr>
<td>MAE [22]</td>
<td>ViT-B16</td>
<td>86M</td>
<td>6.6</td>
</tr>
<tr>
<td>Hummingbird [5]</td>
<td>ViT-B16</td>
<td>86M</td>
<td>70.5</td>
</tr>
<tr>
<td>Trained on Videos</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME T (ours)</td>
<td>ViT-S16</td>
<td>21M</td>
<td>61.6</td>
</tr>
<tr>
<td>TIME T (ours)</td>
<td>ViT-B16</td>
<td>86M</td>
<td>65.5</td>
</tr>
</tbody>
</table>

Table 8: **Visual In-Context Learning evaluation.** Numbers are taken from [5]. *: numbers are produced by this paper. This approach is equivalent to a non-parametric nearest-neighbor based evaluation on Pascal VOC. The details of the evaluation benchmark can be found in the provided implementation codes.

Visual In-Context Learning evaluation. Here, we contrast our approach with a recently introduced benchmark, which assesses the in-context reasoning capabilities of models within the vision domain. Unlike linear or FCN classification methods, visual in-context learning evaluation obviates the need for fine-tuning or end-to-end training. Instead, it constructs validation segmentation maps using patch-level, feature-wise nearest neighbor similarities between validation images (referred to as "queries") and training samples (termed "keys"). This approach mirrors strategies in the NLP domain, aiming to evaluate the proficiency of models in learning tasks from limited examples presented as demonstrations. The results are shown by Table 8. Given that the vast majority of extant models in the domain utilize ViT-S16 as their backbone, we conducted a re-evaluation of their checkpoints to furnish a consistent and directly comparable evaluation table. Subsequently, we re-trained our model employing ViT-B16 and benchmarked it against other baselines as presented by [5]. The results, as depicted in the table, indicate that even though our model was exclusively trained on videos, it registers performance metrics in line with Leopart [74]. Furthermore, it surpasses the results of the leading method, CrOC [52], which was trained on images. Contrary to Hummingbird [5], TIME T is not custom-fitted to the specific evaluation setup and boasts superior computational efficiency. Notably, it requires only a single GPU for training, in contrast to the 16 TPUs demanded by Hummingbird.

5. Conclusion

This paper has aimed to learn dense representations that are learned from videos; yet can be generalised to the image domain as well. As video content is growing rapidly and contains more information compared to images, learning generalisable knowledge from them facilitates further scaling of self-supervised learning methods. To this effect, we have proposed a self-supervised clustering loss to encourage temporally consistent features between different frames of a video. To efficiently find corresponding views between clip frames, we have proposed to recycle pretrained transformer features to leverage their natural tracking ability at each clip. Our empirical results indicate that this method achieves significant gains on the challenging task of video object segmentation across three different evaluation protocols and three datasets. Moreover, by transferring the learned model to the image domain, we have demonstrated the generalisability of the learned features by surpassing or matching the state-of-the-art for unsupervised image segmentation.

6. Acknowledgement

This work is financially supported by Qualcomm Technologies Inc., the University of Amsterdam and the allowance Top consortia for Knowledge and Innovation (TKIs) from the Netherlands Ministry of Economic Affairs and Climate Policy.

References


[22] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 16000–16009, 2022.


[28] Ping Hu, Fabian Cabra, Oliver Wang, Zhe Lin, Stan Sclaroff, and Federico Perazzi. Temporally distributed networks


computer vision and pattern recognition, pages 8934–8943, 2018. 15


16501

Authorized licensed use limited to: Universiteit van Amsterdam. Downloaded on March 21,2024 at 17:13:28 UTC from IEEE Xplore. Restrictions apply.