Rethinking Self-supervised Correspondence Learning: A Video Frame-level Similarity Perspective

Jiarui Xu    Xiaolong Wang
UC San Diego

Abstract

Learning a good representation for space-time correspondence is the key for various computer vision tasks, including tracking object bounding boxes and performing video object pixel segmentation. To learn generalizable representation for correspondence in large-scale, a variety of self-supervised pretext tasks are proposed to explicitly perform object-level or patch-level similarity learning. Instead of following the previous literature, we propose to learn correspondence using Video Frame-level Similarity (VFS) learning, i.e., simply learning from comparing video frames. Our work is inspired by the recent success in image-level contrastive learning and similarity learning for visual recognition. Our hypothesis is that if the representation is good for recognition, it requires the convolutional features to find correspondence between similar objects or parts. Our experiments show surprising results that VFS surpasses state-of-the-art self-supervised approaches for both OTB visual object tracking [62, 60, 69] and DAVIS video object segmentation. We perform detailed analysis on what matters in VFS and reveals new properties on image and frame level similarity learning. Project page with code: https://jerryxu.net/VFS.

1. Introduction

Learning visual correspondence across space and time is one of the most fundamental problems in computer vision. It is widely applied in 3D reconstruction, scene understanding, and modeling object dynamics. The research of learning correspondence in videos can be cast into two categories: the first one is learning object-level correspondence for visual object tracking [62, 60, 69], relocating the object with bounding boxes along the video; the other one is learning fine-grained correspondence, which is commonly applied in optical flow estimation [31, 21] and video object segmentation [9, 80]. While both lines of research have been extensively explored, most approaches acquire training supervision from simulations or limited human annotations, which increases the difficulty for generalization across different data and tasks.

One way to tackle this problem is to learn representations for correspondence using free temporal supervision signals in videos. Recently, a lot of efforts have been made in self-supervised learning of space-time visual correspondence [70, 74, 33], where different pretext tasks are designed to learn to track pixels or objects in videos. For example, Wang et al. [74] propose to use the cycle-consistency of time (i.e., forward-backward object tracking) as a supervisory signal for learning. Building on this, Jabri et al. [33] combine the cycle-consistency of time with patch-level similarity learning and achieve a significant improvement in learning correspondence. Given this encouraging result, we take a step back and ask the questions: Do we really need to design self-supervised object (or patch) tracking task explicitly to learn correspondence? Can image-level similarity learning alone learn the correspondence?

Recent development in image-level similarity learning (e.g., contrastive learning) has shown the self-supervised
representation can be applied to different downstream semantic recognition tasks, and even surpassing the ImageNet pre-training networks [27, 12, 13, 10, 51, 77, 29, 66, 67]. Our hypothesis is that if the higher-level fully-connected layer feature encodes the object structure and semantic information, it needs to be supported by the ability of finding correspondence between similar object instances and between object parts [83]. This forces the convolutional representation to implicitly learn visual correspondence.

With this hypothesis, we propose to perform Video Frame-level Similarity (VFS) learning for space-time correspondence without any explicit tracking-based pretext task. As illustrated in Figure 1, we forward one pair or multiple pairs of frames from the same video into a siamese network, and compute the similarity between the frame-level features (fully-connected layer embeddings) for learning the network representation. We examine the learning with negative pairs as [12, 27] and without negative pairs as [26, 14] under our VFS framework. We build our model based on the ResNet architecture [28]. During inference, we use the res4 features for fine-grained correspondence task (e.g., DAVIS object segmentation [57]) and the res5 features for object-level correspondence task (e.g., OTB object tracking [76]). Surprisingly, we find VFS can surpass state-of-the-art self-supervised correspondence learning approaches [33, 43]. Based on our experiments, we observe the following key elements for VFS:

(i) Training with large frame gaps and multiple frame pairs improves correspondence. When sampling a pair of frames from the same video for similarity learning, we observe that increasing the time differences between the two frames can improve fine-grained correspondence noticeably on DAVIS (∼3%), and object-level correspondence significantly on OTB (>10%). Training with multiple frame pairs at the same time achieves further improvement.

(ii) Training with color augmentation is harmful for fine-grained correspondence, but beneficial for object-level correspondence. With color augmentation, the performance on the DAVIS dataset is decreased (∼3%) while it significantly improved performance on the OTB dataset (∼10%), which indicates the feature learns better object invariance.

(iii) Training without negative pairs improves both fine-grained and object-level correspondences. Recent literature has shown that similarity learning for visual representation is achievable even without negative pairs [26, 14]. While the results are surprising, it was still unclear how it can be beneficial for performance. In this paper, we show that VFS without negative training pairs can improve representations for different levels of correspondence.

(iv) Training with deeper networks gives significant improvements. While deeper networks generally improves recognition performance, it is not the case when training with self-supervised tracking pretext tasks for correspondence. We observe very small improvement or even worse correspondence results when using ResNet-50 compared to ResNet-18 with previous approaches [74, 33, 43]. When learning correspondence with VFS implicitly, we achieve much better performance when using a deeper model.

Given our detailed analysis and state-of-the-art performance, we hope VFS can serve as a strong baseline for self-supervised correspondence learning. The study of intermediate representations for correspondence also provides a better understanding on what image-level self-supervised similarity learning has learned. Finally, VFS also reveals the new property of similarity learning without negative pairs: it improves both object-level and fine-grained correspondence.

2. Related Work

Temporal Correspondence. Learning correspondence from video frames is a long stand problem in computer vision. We can classify the temporal correspondence into two categories. The first one is the fine-grained correspondence which has been widely studied in optical flow and motion estimation [48, 31, 50, 7, 6, 64, 46]. For example, Brox and Malik [6] proposed a region hierarchy matching approach to perform dense and long-range flow estimation. Recently, deep learning based approaches have been applied to estimate optical flows by training on synthetic datasets [8, 19, 61, 32, 65]. While largely improving the efficiency, training on synthetic data largely restricts the network’s generalization ability to real world scenes. The second one is finding object-level correspondence which is meant to offer reliable and long-range visual object tracking [62, 60, 79, 75, 54, 37]. While tracking by training a detector to perform per-frame recognition offers promising results [60, 2, 38, 72, 4, 42], there is recent rise back to the classic tracking-by-matching methods [16, 79, 30, 45] using deep features [5, 69]. For example, Bertinetto et al. [5] propose a fully-convolutional siamese network and adopt similarity learning for tracking. However, these approaches are still heavily relying on human annotations for training. In this paper, we propose a self-supervised similarity learning approach which learns both fine-grained and object-level correspondence.

Self-supervised Learning from Videos. Self-supervised learning offers a way to learn generalizable visual representations with different pretext tasks [20, 18, 56, 52, 84, 22]. Beyond static images, temporal information from videos also offers rich supervision for representation learning [24, 1, 35, 49, 63, 73, 44, 55, 23, 58]. For example, Wang et al. [73] use off-the-shelf tracker to track objects in videos to provide supervisory signals. The learned representation has shown to be useful for multiple downstream recognition and geometry estimation tasks. Instead of learning a
general representation, there is a line of recent research on self-supervised learning specifically for finding correspondence [70, 74, 71, 41, 40, 43, 33, 59]. For example, Von Drick et al. [70] propose to propagate the current frame color to predict the future frame color as a pretext task to learn fine-grained correspondence between the current and future frame. Other tasks including tracking objects [74, 71] and patches [33, 59] are also designed to explicitly find different levels of correspondences. But is explicit tracking task the only way to learn correspondence? Our work introduces an alternative perspective to learn correspondence implicitly via image-level similarity learning.

**Image-level Similarity Learning.** The image-level similarity learning provides a way for visual representation learning. It uses a siamese network to enforce two different views or augmentations of the same image to have similar features. The recent proposed self-supervised contrastive learning is a one version of similarity learning [77, 53, 29, 82, 3, 66, 27, 13, 51, 12, 10, 23, 58, 15]. The idea is to learn representations via attracting the similar (positive) image pairs and repulsing a large number of dissimilar (negative) image pairs. For example, He et al. [27] propose to use a momentum network to encode the number of negatives for efficient contrastive learning. They have shown the learned representation achieves state-of-the-art performance when transferred to multiple downstream recognition tasks. Purushwalkam and Gupta [58] further extend contrastive learning in temporal domain and show using temporal augmentation can improve object viewpoint invariance. Recently, it has been shown that the negative pairs are not necessary to learn a good visual representation [25, 14]. While it provides a better understanding for similarity learning, it is unclear if learning without negatives gives better representations. In this paper, we extend image-level similarity learning to video frames. Instead of performing recognition tasks, our aim is to learn visual correspondence. We also reveal that learning without negatives is beneficial for finding space-time correspondence.

### 3. Method

We propose Video Frame-level Similarity (VFS) learning for different levels of space-time correspondence. In this section, we will first introduce two common practices for image-level similarity learning for visual representations. Then we will unify these two paradigms under VFS.

#### 3.1. Background: Image-level Similarity Learning

The image-level similarity learning [77, 27, 25, 12, 14] provides a way to learn visual representations in a self-supervised manner. It learns the representation by minimizing the distance between two different augmented views of the same image in feature space. We classify the similarity learning into two types based on whether it is using the negative examples during training.

**Similarity Learning with Negatives.** Contrastive learning [77, 27, 12] is similarity learning with negative sample pairs. In image-level contrastive learning, the positive pairs are different augmented views of the same image, and the negative pairs are from different images. The training objective is to push the representations of the positive (similar) pairs to be close to each other, while keep the representations of the negative (dissimilar) pairs to be far.

Formally, we denote $x$ and $x'$ as two different augmented views of an input image. The contrastive learning utilizes a siamese network architecture with a predictor encoder $P$ and a target encoder $T$. We use these two networks to extract features for both inputs respectively, and normalize the outputs with $l_2$-normalization. The output embeddings for $x$ and $x'$ can be represented as $p \triangleq P(x)/\|P(x)\|_2$ and $z \triangleq T(x')/\|T(x')\|_2$. Let $\mathcal{U} = \{u_1, u_2, \ldots, u_K\}$ be the negative bank that stores the features of negative samples. The optimization objective is minimizing InfoNCE loss [53], defined as

\[
\mathcal{L}_{p,z,\mathcal{U}} = -\log \frac{\exp(p \cdot z/\tau)}{\exp(p \cdot z/\tau) + \sum_{k=1}^{K} \exp(p \cdot u_k/\tau)}
\]

where $\tau$ is a temperature hyper-parameter [77].

There are multiple ways to construct the negative sample bank including directly storing all the features [77] or sampling the negatives online [12]. In this paper, we adopt a recent practice proposed by He et al. [27], which uses a momentum updated queue as the negative bank. To achieve this, the target encoder network $T$ is updated as a moving average of the predictor encoder $P$ as,

\[
\xi \leftarrow m \xi + (1-m) \theta, \ m \in [0, 1),
\]

where $\theta, \xi$ are parameters of $P$ and $T$. Thus the encoders $P$ and $T$ share the same network architecture in [27].

**Similarity Learning without Negatives.** Recently, researchers discover that similarity learning without negative sample pairs can achieve comparable performance as contrastive learning in visual representation learning [25, 14]. Without the negative samples, the optimization objective can be simplified as the minimizing the cosine feature distance between two views

\[
\mathcal{L}_{p,z} = \|p - z\|_2^2 = 2 - 2 \cdot p \cdot z.
\]

However, optimizing this objective will easily lead to degenerated solution and yields a collapsed representation [25, 14]. To resolve this issue, Chen et al. [14] propose two techniques: (i) Share most of the parameters between the predictor encoder $P$ and the target encoder $T$, except adding one additional MLP for the predictor encoder; (ii)
Stop the gradients back-propagated from the loss to the target network. In this paper, we adopt this approach for similarity learning without negative data. Please refer to [14] for more details.

### 3.2. Video Frame-level Similarity Learning

Building on image-level similarity learning, we propose to perform similarity learning between video frames, i.e., Video Frame-level Similarity learning (VFS). Our approach considers frames at different timestamps as different views for similarity learning. In a video clip with length $L$, there are $\frac{L^2}{2}$ possible positive sample pairs for learning. In VFS, each video frame is pulled towards a global video feature, resulting in a representation invariant to natural object deformation and viewpoint changes over time. In our experiments, we show that this learning objective can enforce the emergence of visual correspondence from the convolutional layers. Our hypothesis is that, like applying other augmentations in image-level similarity learning, sampling different temporal views also help the representation to learn object structure and even semantic information. This implicitly requires the convolutional features to learn about object and fine-grained correspondence. We will introduce our approach in details as follows.

**Learning Objectives.** Given a video with $L$ frames $\{f_1, f_2, \ldots, f_L\}$, we sample two random frames $f_i, f_j$ and apply data augmentation on them. We then forward both frames to the predictor encoder $P$ and target encoder $T$ to extract their features as $p_i \triangleq P(f_i)/\|P(f_i)\|_2$, $z_j \triangleq T(f_j)/\|T(f_j)\|_2$. We provide two options for VFS learning with and without the negative pairs. Following Eq. 1, the objective for VFS with negative pairs is represented as,

$$
    \mathcal{L}_{p_i,z_j,\mathcal{U}} = -\log \frac{\exp(p_i^T z_j/\tau)}{\exp(p_i^T z_j/\tau) + \sum_{k=1}^{K} \exp(p_i^T u_k/\tau)}.
$$

(4)

On the other hand, VFS can also be trained without negative pairs by following Eq. 3, and the objective is,

$$
    \mathcal{L}_{p_i,z_j} = \|p_i - z_j\|_2^2 = 2 - 2p_i^T z_j.
$$

(5)

While we unify learning with and without negatives under VFS, we adopt the implementation details in [13, 14] to adjust the architectures and training schemes accordingly as introduced in Section 3.1.

We illustrate our learning pipeline as Figure 2. Beyond sampling one pair of data, learning from videos allows to sample multi-paired positive samples. To be more specific, we can sample $n$ frames from the videos, and divide them into two splits for predictor and target encoders. We then compute the feature similarity between $\frac{n}{2}$ predictor features and $\frac{n}{2}$ target features which yields a affinity matrix is of shape $\frac{n}{2} \times \frac{n}{2}$. When training without negatives, we can apply Eq. 5 to maximize each element in the affinity matrix, i.e., minimize the distance between each video frame pair. When training with $K$ negative examples, the shape of the affinity matrix becomes $\frac{n}{2} \times (\frac{n}{2} + K)$. We apply Eq. 4 for learning, where the negative bank $\mathcal{U} = \{u_1, u_2, \ldots, u_K\}$ are sampled from other videos. We illustrate the sampling method for the $n$ frames as follows.

**Temporal Sampling.** Given a video with $L$ frames, sampling $n$ frames yields a set of indices $\{I_1, I_2, \ldots, I_n\}$, where $I_i \in [1, L]$. In this paper, we investigate two strategies for temporal sampling: (i) Continuous sampling, which first selects a starting frame index and then continuously samples with a fixed frame interval $\delta$; (ii) Distant sampling, which splits the video into $n$ disjoint segments, then randomly selects 1 frame from each segment as
Data Augmentation. Besides using temporal signals to provide different views of training data, we also adopt the common data augmentation practices in image-level similarity learning [77, 13, 12]. We apply the spatial augmentation (e.g., random cropping and flipping) and color augmentation (e.g., grayscale and color jitter) in our experiments. Specifically, we observe the color augmentation plays an important role in correspondence learning and we will report our findings in the experiment section.

4. Experiments

We perform experiments on representation learned by VFS for fine-grained correspondence tasks [57, 36, 85] and object-level correspondence task [76]. We will first introduce our training details and evaluation metrics, then we will perform extensive ablations on different elements for VFS and provides a better understanding on how VFS learns correspondence. Based on these observations, we finally report VFS surpasses all previous self-supervised learning approaches on both correspondence tasks.

4.1. Self-Supervised Pre-Training

Architectures. We use standard ResNet [28] as the backbone network. We introduce the architecture for training without negative pairs (following [14]) and the architecture for training with negative pairs (following [13]) as below.

- Architecture without negative pairs. The predictor encoder consists of a backbone network and a projector followed by a predictor. The target encoder is composed of the backbone and the projector. The parameters of the backbone and the projector are shared between the two encoders. The projector is a 3-layer MLP and the predictor is a 2-layer MLP as [14]. All batch normalization layers in the backbone, the projector and the predictor are synchronized across devices (SyncBN) as in [12, 25, 14].

- Architecture with negative pairs. The predictor encoder and target encoder share the same architecture, including a backbone followed by a 2-layer projector MLP. The parameters of the target encoder are updated with momentum $m = 0.999$ with Eq. 2. We set the temperature $\tau = 0.2$ and the negative bank size $K = 65536$.

Pre-training. We adopt the Kinetics[39] dataset for self-supervised training. It consists of $\sim 240k$ training videos. The batch size is 256. The learning rate is initialized to 0.05, and decays in the cosine schedule [12, 47, 13]. We use SGD optimizer with momentum 0.9 and weight decay 0.0001. We found that training for 200 epochs is sufficient for ResNet-18 models, and ResNet-50 models need 500 epochs to converge (roughly the same number of iterations as 100 epochs training [14] on ImageNet).

4.2. Evaluation

We evaluate the pre-trained representation on both fine-grained and object-level correspondence downstream tasks.

Fine-grained Correspondence. To evaluate the quality of fine-grained correspondence, we follow the same testing protocol and downstream tasks in [33, 74]. We reimplement the inference approach based on [34]. Without any fine-tuning, we directly use unsupervised pre-trained model as the feature extractor. The fine-grained similarity is measured on the res$^3$ feature map, with its stride reduced to 1 during inference. As in [74, 33, 43], the recurrent inference strategy is applied: The first frame ground truth labels as well the prediction results in the latest 20 frames are propagated to the current frame. We evaluate the fine-grained correspondence over three downstream tasks and datasets: video object segmentation in DAVIS-2017 [57], human pose tracking in JHMDB [36] and human part tracking in VIP [85]. We perform most of our ablations with DAVIS-2017 and report the comparisons with state-of-the-art approaches in all datasets.

Object-level Correspondence. We evaluate object-level correspondence with visual object tracking in the OTB-100 [76] dataset. We adopt the SiamFC [5] tracking algorithm with our representation from the res$^3$ block. We follow the same evaluation protocol and hyperparameters in VINCE [23] and SeCo [81]: Given a pre-trained ResNet, the strides in res$^3$ and res$^2$ layers are removed, and the dilation rate of 3x3 convolution blocks in res$^4$ and res$^3$ layers are set to 2 and 4 respectively. This commonly used modification makes the res$^2$ block resolution compatible with the original setting in SiamFC. The network modification does not affect the pre-trained weights. We perform most of our analysis and ablation for VFS using the frozen representation after pre-training. We compare to the state-of-the-art results using fine-tuning in the end following [23, 81].

4.3. Results and Ablative Analysis

We first perform analysis on different elements of VFS. The experiments are under the without negative pairs setting with ResNet-18 backbone unless specified otherwise.

Augmentation. Our first discovery is that color augmentation plays an important role in VFS and affect fine-grained and object-level correspondence in an opposite way. We report our results on augmentations in Table 2.

\[ I_i = \frac{L}{K}(i - 1) + \text{unif}(0, \frac{L}{K}). \]
Data augmentation, the performance on DAVIS decrease by $>3\%$ and the OTB precision improves over $10\%$. We conjecture that while color augmentation helps learning invariance to object appearance changes, the per-pixel distortion confuses the lower-level convolution features to find fine-grained correspondence. From Table 2, we also observe sampling different frames in a video (with distant sampling) indeed help learning both object-level and fine-grained correspondence, and adding spatial augmentations (random crop and flip) can further improve the results. In the following experiments, we adopt different frame inputs and spatial augmentation by default. We will report results with and without color augmentations in the different ablations.

**Temporal Sampling.** Does sampling with larger frame interval improve correspondence learning? To answer this question, we study the temporal sampling strategy and report our results in Table 3. We perform the study without using the color augmentation. Recall we have two temporal sampling strategies in VFS (Section 3.2). With **continuous sampling**, we elaborate the frame interval $\delta$ from 0 to 32. We observe as $\delta$ increases, both fine-grained and object-level correspondence improves consistently. With **distant sampling** using $n = 2$ frame inputs (labeled as "D" in Table 3), we achieve the best results. We conjecture increasing the frame intervals offers better augmentation effect which leads to better correspondence. We will adopt the distant sampling in the following subsections.

### Table 3. Ablation on frame interval. 0 means sample two identical frames. D stands for the distant sampling. Others use continuous sampling with fixed frame sampling interval.

<table>
<thead>
<tr>
<th>frame interval</th>
<th>DAVIS $J_{ hypotheses} F_{ hypotheses}$</th>
<th>OTB Precision Success</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>62.2 60.8 63.6</td>
<td>30.6 26.1</td>
</tr>
<tr>
<td>2</td>
<td>63.5 61.7 65.4</td>
<td>38.1 31.2</td>
</tr>
<tr>
<td>4</td>
<td>62.9 61.4 64.3</td>
<td>35.5 29.5</td>
</tr>
<tr>
<td>8</td>
<td>63.5 61.7 65.3</td>
<td>38.5 31.8</td>
</tr>
<tr>
<td>16</td>
<td>63.9 61.9 65.8</td>
<td>44.1 33.9</td>
</tr>
<tr>
<td>32</td>
<td>64.5 62.4 66.7</td>
<td>46.9 36.1</td>
</tr>
<tr>
<td>D</td>
<td>65.0 62.6 67.4</td>
<td>48.1 37.9</td>
</tr>
</tbody>
</table>

### Table 4. Ablation on multiple frames. “frame num” is the number of sampled frame.

<table>
<thead>
<tr>
<th>negative pairs aug</th>
<th>DAVIS $J_{ hypotheses} F_{ hypotheses}$</th>
<th>OTB Precision Success</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>65.0 62.6 67.4</td>
<td>48.1 37.9</td>
</tr>
<tr>
<td></td>
<td>64.7 62.2 67.3</td>
<td>39.0 31.7</td>
</tr>
<tr>
<td></td>
<td>61.9 59.5 64.3</td>
<td>57.3 43.0</td>
</tr>
<tr>
<td></td>
<td>61.5 59.3 63.7</td>
<td>53.7 40.8</td>
</tr>
</tbody>
</table>

### Table 5. Ablation on negative pairs. For with negative pairs setting, the learning objective is Eq 4.

<table>
<thead>
<tr>
<th>negative pairs aug</th>
<th>DAVIS $J_{ hypotheses} F_{ hypotheses}$</th>
<th>OTB Precision Success</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>65.0 62.6 67.4</td>
<td>48.1 37.9</td>
</tr>
<tr>
<td></td>
<td>64.7 62.2 67.3</td>
<td>39.0 31.7</td>
</tr>
<tr>
<td></td>
<td>61.9 59.5 64.3</td>
<td>57.3 43.0</td>
</tr>
<tr>
<td></td>
<td>61.5 59.3 63.7</td>
<td>53.7 40.8</td>
</tr>
</tbody>
</table>

### Table 6. Ablation on deeper models. ResNet-18 and ResNet-50 are compared.

**Multiple Frames.** We investigate the effect of training with multiple pairs of frames from one video. With distant sampling, we ablate the number of segments $n = 2, 4, 8$, which indicates using 1, 2, 4 pairs of frames from one video during training. We report our results in Table 4. With or without using color augmentation, we observe training with more pairs of frames generally improves the correspondence representation. Note the first row in Table 4 corresponds to the last row in Table 3.

**Negative Pairs.** So far we conduct our experiments under the without negative pairs setting. But will adding the negative pairs (i.e., contrastive learning) help correspondence learning? To our surprise, adding negative pairs in training hurts learning both fine-grained and object-level correspondence. We report the comparisons in Table 5. While the fine-grained correspondence result on DAVIS drops slightly with negative pairs, the performance of object tracking in OTB degenerates significantly. When training without color augmentations, the object tracking precision drops $\sim 9\%$.

What is the reason causing this performance drop when training with negatives? Our hypothesis is that training with negative pairs may sacrifice the performance on mod-
VFS with ResNet-50 backbone brings 5% precision gain over ResNet-18, yields 73.9% precision and 52.5% success score. Our simple VFS surpasses ImageNet supervised pretraining as well as previous self-supervised state-of-the-art SeCo [81], which involves joint training of 3 pretext tasks.

Different Blocks of Layers. We plot the VFS model performance on DAVIS throughout different epochs of similarity learning with ResNet-18 (Figure 4) and ResNet-50 (Figure 5). We investigate how each feature block in res$_4$ and res$_5$ layers performs. For example, res$_4$.b$_1$ indicates the feature from the first block in res$_4$. In Figure 4, all blocks begin with $\sim 60\%$ on DAVIS in early stages of training. However, the gap between res$_4$ and res$_5$ becomes larger as the model is trained longer. The blocks in res$_5$ ends up with similarly scores as beginning, while the results of res$_4$ continuously improve. This supports our previous results that res$_4$ learns fine-grained correspondence over time and res$_5$ focuses more on object-level features. Similarly for ResNet-50 in Figure 5, res$_4$ also outperforms res$_5$ by a noticeable margin for finding fine-grained correspondence.

4.4. Comparison with State-Of-The-Art

We compare fine-grained correspondence results of VFS against previous self-supervised methods in Table 7. The results are all reported with the last block in res$_4$ across all methods. Our method achieves state-of-the-art performance using ResNet-50. With ResNet-50, we observe UVC [43] does not benefit from using a deeper networks. For CRW [33] we could not make it work on ResNet-50. Note we try to adjust the hyper-parameters for CRW training (e.g., learning rate, learning schedule, warm up for training, temperature adjustment for softmax layer) for multiple times, but we cannot achieve better features than random initialization. Learning with VFS, the deeper network with ResNet-50 improves 2.2% on DAVIS and 3.3% on VIP over ResNet-18, which is significant. We observe consistent results across the JHMDB [36] human pose and VIP [85] human part tracking tasks. With ResNet-18, VFS achieves comparable performance with CRW [33]. As shown in Figure 4 and 5, the last block in res$_4$ may not achieve the optimal performance, thus we also report the result of the best block with gray color for reference.

The comparison on learning object-level correspondence is reported in Table 8. We report the fine-tuning performance of our best setting, namely distant sampling with $n = 8$, with color augmentation and without negative pairs. VFS with ResNet-50 backbone brings 5% precision gain over ResNet-18, yields 73.9% precision and 52.5% success score. Our simple VFS surpasses ImageNet supervised pretraining as well as previous self-supervised state-of-the-art SeCo [81], which involves joint training of 3 pretext tasks.
Table 7. Comparison with state-of-the-art on fine-grained correspondence. * indicates localization is involved during label propagation. † denotes strides of last two layers are removed. ‡ denotes max pooling of stem layer is also removed. Stride is the output stride (downsample ratio) of ResNet during training.

<table>
<thead>
<tr>
<th>Method</th>
<th>Backbone</th>
<th>Stride</th>
<th>Dataset</th>
<th>$\mathcal{J}_R$, $F_m$</th>
<th>$F_m$</th>
<th>VIP mIoU</th>
<th>JHMDB PCK@0.1</th>
<th>PCK@0.2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supervised [28]</td>
<td>ResNet-18</td>
<td>32</td>
<td>ImageNet</td>
<td>62.9 60.6 65.2</td>
<td>31.9</td>
<td>53.8</td>
<td>74.6</td>
<td></td>
</tr>
<tr>
<td>SimSiam [14]</td>
<td>ResNet-18</td>
<td>32</td>
<td>ImageNet</td>
<td>62.0 60.6 64.0</td>
<td>30.3</td>
<td>55.2</td>
<td>74.0</td>
<td></td>
</tr>
<tr>
<td>MoCo [27]</td>
<td>ResNet-18</td>
<td>32</td>
<td>ImageNet</td>
<td>60.8 58.6 63.1</td>
<td>29.2</td>
<td>55.0</td>
<td>72.7</td>
<td></td>
</tr>
<tr>
<td>VINE [23]</td>
<td>ResNet-18</td>
<td>32</td>
<td>Kinetics</td>
<td>60.4 57.9 62.8</td>
<td>30.5</td>
<td>55.6</td>
<td>73.4</td>
<td></td>
</tr>
<tr>
<td>CorrFlow [41]†</td>
<td>ResNet-18</td>
<td>4</td>
<td>OtXUVa</td>
<td>50.3 48.4 52.2</td>
<td>-</td>
<td>58.5</td>
<td>78.8</td>
<td></td>
</tr>
<tr>
<td>MAST [40]†</td>
<td>ResNet-18</td>
<td>4</td>
<td>OtXUVa</td>
<td>63.7 61.2 66.3</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MAST [40]‡</td>
<td>ResNet-18</td>
<td>4</td>
<td>YT-VOS</td>
<td>65.5 63.3 67.6</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Vid. Color. [70]</td>
<td>ResNet-18</td>
<td>8</td>
<td>Kinetics</td>
<td>34.0 34.6 32.7</td>
<td>-</td>
<td>45.2</td>
<td>69.6</td>
<td></td>
</tr>
<tr>
<td>TimeCycle [74]</td>
<td>ResNet-18</td>
<td>8</td>
<td>VLOG</td>
<td>39.2 40.1 38.3</td>
<td>28.9</td>
<td>57.3</td>
<td>78.1</td>
<td></td>
</tr>
<tr>
<td>UVC [43]</td>
<td>ResNet-18</td>
<td>8</td>
<td>Kinetics</td>
<td>57.8 56.3 59.2</td>
<td>34.1</td>
<td>58.6</td>
<td>79.6</td>
<td></td>
</tr>
<tr>
<td>UVC+track [43]†</td>
<td>ResNet-18</td>
<td>8</td>
<td>Kinetics</td>
<td>59.5 57.7 61.3</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRW [33]</td>
<td>ResNet-18</td>
<td>8</td>
<td>Kinetics</td>
<td>67.6 64.8 70.2</td>
<td>38.6</td>
<td>59.3</td>
<td>80.3</td>
<td></td>
</tr>
<tr>
<td>VFS</td>
<td>ResNet-18</td>
<td>32</td>
<td>Kinetics</td>
<td>66.7 64.0 69.4</td>
<td>39.9</td>
<td>60.5</td>
<td>79.5</td>
<td></td>
</tr>
<tr>
<td>VFS (best block)</td>
<td>ResNet-18</td>
<td>32</td>
<td>Kinetics</td>
<td>67.9 65.0 70.8</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method</th>
<th>Backbone</th>
<th>Stride</th>
<th>Dataset</th>
<th>$\mathcal{J}_R$, $F_m$</th>
<th>$F_m$</th>
<th>VIP mIoU</th>
<th>JHMDB PCK@0.1</th>
<th>PCK@0.2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supervised [28]</td>
<td>ResNet-50</td>
<td>32</td>
<td>ImageNet</td>
<td>66.0 63.7 68.4</td>
<td>39.5</td>
<td>59.2</td>
<td>78.3</td>
<td></td>
</tr>
<tr>
<td>SimSiam [14]</td>
<td>ResNet-50</td>
<td>32</td>
<td>ImageNet</td>
<td>66.3 64.5 68.2</td>
<td>35.0</td>
<td>58.4</td>
<td>77.5</td>
<td></td>
</tr>
<tr>
<td>MoCo [27]</td>
<td>ResNet-50</td>
<td>32</td>
<td>ImageNet</td>
<td>65.4 63.2 67.6</td>
<td>36.1</td>
<td>60.4</td>
<td>79.3</td>
<td></td>
</tr>
<tr>
<td>VINE [23]</td>
<td>ResNet-50</td>
<td>32</td>
<td>Kinetics</td>
<td>65.6 63.4 67.8</td>
<td>36.0</td>
<td>58.2</td>
<td>76.3</td>
<td></td>
</tr>
<tr>
<td>RegionTracker [58]</td>
<td>ResNet-50</td>
<td>32</td>
<td>TrackingNet</td>
<td>63.4 61.5 65.4</td>
<td>33.9</td>
<td>57.5</td>
<td>74.6</td>
<td></td>
</tr>
<tr>
<td>TimeCycle [74]</td>
<td>ResNet-50</td>
<td>8</td>
<td>VLOG</td>
<td>40.7 41.9 39.4</td>
<td>28.9</td>
<td>57.7</td>
<td>78.5</td>
<td></td>
</tr>
<tr>
<td>UVC [43]</td>
<td>ResNet-50</td>
<td>8</td>
<td>Kinetics</td>
<td>56.3 54.5 58.1</td>
<td>34.2</td>
<td>56.0</td>
<td>76.6</td>
<td></td>
</tr>
<tr>
<td>VFS</td>
<td>ResNet-50</td>
<td>32</td>
<td>Kinetics</td>
<td>68.9 66.5 71.3</td>
<td>43.2</td>
<td>60.9</td>
<td>80.7</td>
<td></td>
</tr>
<tr>
<td>VFS (best block)</td>
<td>ResNet-50</td>
<td>32</td>
<td>Kinetics</td>
<td>69.4 66.7 72.0</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 8. Comparison with state-of-the-art on object-level correspondence. † denotes strides of last two layers are removed.

<table>
<thead>
<tr>
<th>Method</th>
<th>Backbone</th>
<th>Dataset</th>
<th>OTB</th>
<th>Precision</th>
<th>Success</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supervised [28]</td>
<td>ResNet-18</td>
<td>ImageNet</td>
<td>61.4</td>
<td>43.0</td>
<td></td>
</tr>
<tr>
<td>SimSiam [14]</td>
<td>ResNet-18</td>
<td>ImageNet</td>
<td>58.8</td>
<td>42.9</td>
<td></td>
</tr>
<tr>
<td>MoCo [27]</td>
<td>ResNet-18</td>
<td>ImageNet</td>
<td>62.0</td>
<td>47.0</td>
<td></td>
</tr>
<tr>
<td>VINE [23]</td>
<td>ResNet-18</td>
<td>Kinetics</td>
<td>62.9</td>
<td>46.5</td>
<td></td>
</tr>
<tr>
<td>CRW [33]</td>
<td>ResNet-18</td>
<td>8 Kinetics</td>
<td>52.6</td>
<td>40.1</td>
<td></td>
</tr>
<tr>
<td>VFS</td>
<td>ResNet-18</td>
<td>Kinetics</td>
<td>68.9</td>
<td>52.2</td>
<td></td>
</tr>
<tr>
<td>Supervised [28]</td>
<td>ResNet-50</td>
<td>ImageNet</td>
<td>65.8</td>
<td>45.5</td>
<td></td>
</tr>
<tr>
<td>SimSiam [14]</td>
<td>ResNet-50</td>
<td>ImageNet</td>
<td>61.0</td>
<td>43.2</td>
<td></td>
</tr>
<tr>
<td>MoCo [27]</td>
<td>ResNet-50</td>
<td>ImageNet</td>
<td>63.7</td>
<td>46.5</td>
<td></td>
</tr>
<tr>
<td>VINE [23]</td>
<td>ResNet-50</td>
<td>Kinetics</td>
<td>66.0</td>
<td>47.6</td>
<td></td>
</tr>
<tr>
<td>RegionTracker [58]</td>
<td>ResNet-50</td>
<td>TrackingNet</td>
<td>57.4</td>
<td>43.4</td>
<td></td>
</tr>
<tr>
<td>SeCo [81]</td>
<td>ResNet-50</td>
<td>Kinetics</td>
<td>71.9</td>
<td>51.8</td>
<td></td>
</tr>
<tr>
<td>VFS</td>
<td>ResNet-50</td>
<td>Kinetics</td>
<td>73.9</td>
<td>52.5</td>
<td></td>
</tr>
</tbody>
</table>

5. Discussion and Conclusion

We propose a simple yet effective approach for self-supervised correspondence learning. We demonstrate that both fine-grained and object-level correspondence can emerge in different layers of the ConvNets with video frame-level similarity learning. In addition to the state-of-the-art performance, we provide the following insights.

Is designing a tracking-based pretext task a necessity for self-supervised correspondence learning? It might not be necessary. While tracking-based pretext tasks still have potentials, it is limited by small backbone models and is now surpassed by our simple frame-level similarity learning. To make the tracking-based pretext tasks useful, we need to first make its learning scalable and generalizable in model size and network architectures.

Does color augmentation improve the correspondence? Yes and no. We show that color augmentation is beneficial for correspondence in object-level but jeopardizes the fine-grained correspondence. While color augmentation brings object appearance invariance, it also confuses the lower-layer convolution features.

How to sample video frames? Sample multiple frames and sample with a large gap. The large temporal gap provides more aggressive temporal transform, which boosts correspondence significantly. Comparing multiple pairs of frame further improves the results.

Is negative pairs helpful? No. We observe inferior performance when training with negative samples, specifically for object-level correspondence. We also shed light on the reason why without negative pairs is more helpful, which has not been studied before.

In conclusion, we hope VFS can serve as a strong baseline for self-supervised correspondence learning. We provide a new perspective on studying similarity learning for visual representations beyond recognition tasks.

Acknowledgements. This work was supported, in part, by grants from DARPA LwLL, NSF 1730158 CI-New: Cognitive Hardware and Software Ecosystem Community Infrastructure (CHASE-CI), NSF ACI-1541349 CC*DNI Pacific Research Platform, and gifts from Qualcomm, TuSimple and Picsart.
References


[38] Zdenek Kalal, Krystian Mikolajczyk, and Jiri Matas. Tracking-learning-detection. *TPAMI*, 2012. 2


[62] Ishwar K Sethi and Ramesh Jain. Finding trajectories of feature points in a monocular image sequence. *IEEE Transac-
tions on pattern analysis and machine intelligence, (1):56–73, 1987. 1, 2
[64] Deqing Sun, Stefan Roth, and Michael J Black. Secrets of optical flow estimation and their principles. In CVPR, 2010. 2
[80] Linjie Yang, Yanran Wang, Xuehan Xiong, Jianchao Yang, and Aggelos K Katsaggelos. Efficient video object segmentation via network modulation. 2018. 1