This CVPR Workshop paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Learning Tracking Representations from Single Point Annotations

Qiangqiang Wu Antoni B. Chan
Department of Computer Science, City University of Hong Kong
qiangqwu2-c@my.cityu.edu.hk, abchan@cityu.edu.hk

Abstract

Existing deep trackers are typically trained with large-scale video frames with annotated bounding boxes. However, these bounding boxes are expensive and time-consuming to annotate, in particular for large scale datasets. In this paper, we propose to learn tracking representations from single point annotations (i.e., $4.5\times$ *faster to annotate than the traditional bounding box) in a weakly supervised manner. Specifically, we propose a soft contrastive learning (SoCL) framework that incorporates target objectness prior into end-to-end contrastive learning. Our SoCL consists of adaptive positive and negative sample generation, which is memory-efficient and effective for learning tracking representations. We apply the learned representation of SoCL to visual tracking and show that our method can 1) achieve better performance than the fully supervised baseline trained with box annotations under the same annotation time cost; 2) achieve comparable performance of the fully supervised baseline by using the same number of training frames and meanwhile reducing annotation time cost by 78% and total fees by 85%; 3) be robust to annotation noise.*

1. Introduction

Visual object tracking is a basic computer vision task with a long history spanning decades. In recent years, considerable progress [9, 26, 44] has been made in the tracking community with the development of deep learning techniques. Deep trackers have achieved strong performance on existing tracking benchmarks [17, 23, 32], and show great potential in various applications.

Existing deep trackers are mainly trained with large-scale datasets comprising bounding box annotations on video frames. In order to obtain high-quality bounding box annotations, one common practice is to employ large numbers of people on a crowd-sourcing platform (e.g., Amazon Mechanical Turk) for annotating. Usually, there are two typical steps to annotate a video frame: 1) draw a bounding box that tightly includes the object, and 2) verify the annotated bounding box for quality control. These two steps

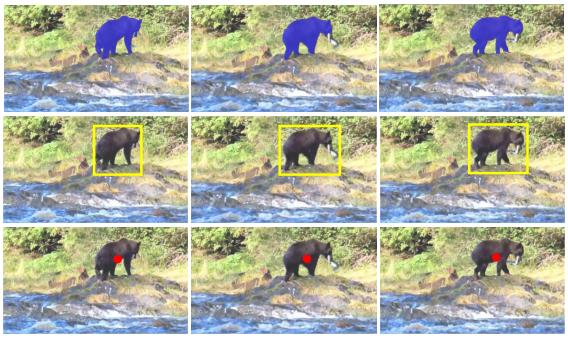


Figure 1. An illustration of video frame annotations using masks, bounding boxes and center points. The time for humans to label point annotations is $4.5\times$ and $34.4\times$ faster than the time for bounding boxes and mask annotations, respectively. In this paper, we propose a novel *soft contrastive learning framework* to learn tracking representations from point annotations in video frames so as to reduce annotation cost and total fees.

respectively take 10.2 and 5.7 seconds [37]. Considering that existing tracking datasets consist of millions of annotated bounding boxes, e.g., ILSVRC [36] (2.5M) and Got-10K [23] (1.4M), a conservative estimate of the time cost for annotating ILSVRC and Got-10K are 7,083 and 3,967 hours, even without accounting for the verification time. To ease the annotation cost in visual tracking, recent progress [42, 44] on unsupervised tracking generate pseudo labels for representation learning. However, these works still lag behind their fully supervised counterparts [9, 27, 55] due to noise in the pseudo labels.

Different from previous trackers that use expensive bounding box annotations for fully supervised training, in this paper, we propose to learn tracking representations from low-cost and efficient single point annotations (see Fig. 1) in a weakly supervised manner. For point annotations, annotators only need to click once at the object center, which takes about 2.27 seconds per frame (refer to Sec. 3.1 for more details) and is $4.5\times$ faster than bounding box annotation. Although point annotations have low time cost, learning effective tracking representations from them is challenging due to the following two reasons: 1) the point annotations naturally lack target scale information, whereas target scale is vital information needed for training traditional deep trackers; 2) the annotations may be noisy since

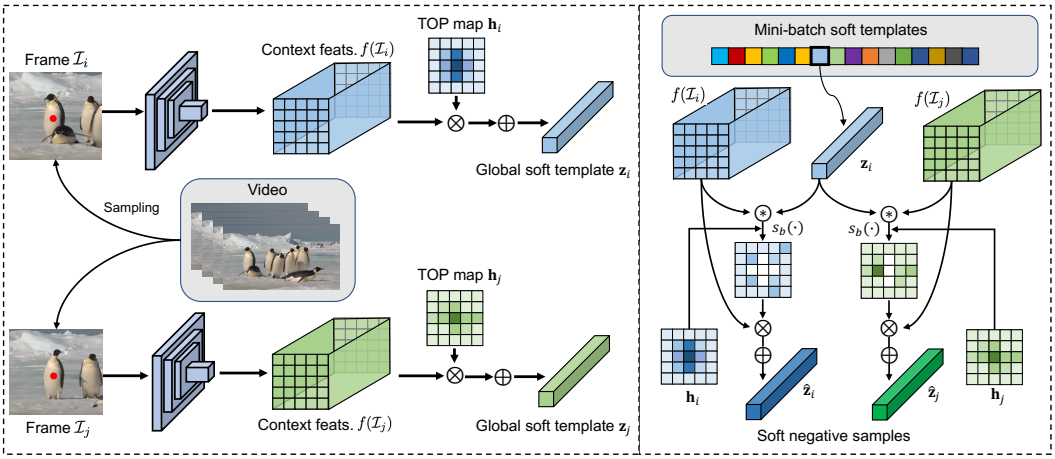


Figure 2. Overview of (a) global soft template generation (GST); and (b) soft negative sample (SNS) generation in the proposed SoCL framework. (a) Given two randomly selected frames $\mathcal{I}_i$ and $\mathcal{I}_j$ in a video, we firstly extract their context features $f(\mathcal{I}_i)$ and $f(\mathcal{I}_j)$, and then calculate GSTs $\mathbf{z}_i$ and $\mathbf{z}_j$ as the weighted sum over the spatial locations on $f(\mathcal{I}_i)$ and $f(\mathcal{I}_j)$, where each location weight is from the corresponding location in the target objectness prior (TOP) maps $\mathbf{h}_i$ and $\mathbf{h}_j$. (b) During the mini-batch training, for a specific GST (e.g., $\mathbf{z}_i$), we obtain two similarity maps between $\mathbf{z}_i$ and each location in the context features by using a cross-correlation operation (denoted as $\circledast$). We next use a background selection function $s_b(\cdot)$ to mask out target responses and select background counterparts with high responses in the similarity maps to generate the SNSs $\hat{\mathbf{z}}_i$ and $\hat{\mathbf{z}}_j$. The generation of both GST and SNS is memory-efficient. $\otimes$ is element-wise multiplication, while $\oplus$ is a sum over spatial locations.

the annotated target center does not always perfectly match the ground-truth target center.

To tackle the above problems and learn robust tracking representations from point annotations, we propose a soft contrastive learning framework (SoCL) that generates global and local soft templates (GSTs and LSTs) based on a target objectness prior (TOP) map, and then optimizes a pairwise contrastive loss between positive/negative soft samples. The TOP map contains the pixel-wise probabilities that each pixel location belongs to the target. The GSTs are generated by aggregating each location in the feature map based on the TOP maps (see Fig. 2). In order to facilitate discriminative feature learning and avoid large memory cost, we propose a memory-efficient method to adaptively generate soft negative samples using high-similarity regions of the cross-correlation map between the GST and the feature map. In addition, we also sample LSTs, which simulate partial occlusion or appearance variations, to augment the positive set, further boosting the representation learning.

The learned representations of SoCL can be directly applied to both Siamese and correlation filter tracking frameworks. In addition, we also successfully combine our framework with additional sparse bounding box annotations so as to generate pseudo bounding box labels, in order to train state-of-the-art scale regression-based trackers (e.g., TransT [9]). In summary, our main contributions are:

- We propose a soft contrastive learning (SoCL) framework, which incorporates a target objectiveness prior into end-to-end contrastive learning, in order to learn tracking representations from single point annotations.
- We propose a memory-efficient method for soft negative sample generation, which significantly increases the number of negative samples with low memory cost.
- We propose to generate a local soft template for each global soft template and facilitate the representation learning via global-to-local contrastive learning.
- Experiment results show that our tracker learned from single point annotations can: 1) achieve comparable performance to the fully supervised baseline trained with box annotations when using the same number of training frames, while reducing the annotation time cost by 78% and total fees by 85%; 2) obtain better performance by using the same annotation time cost, and 3) be robust to annotation noise.

2. Related Work

Deep Tracking Methods. Currently, visual tracking is dominated by deep learning-based trackers that are trained with large-scale annotated datasets. The deep CF trackers [10–12, 28, 28–30, 45, 46] employ deep features for CF tracking. SiamFC [4] and SINT [39] are two pioneering Siamese trackers, which convert visual tracking to a template matching problem. Follow-up works aim to more accurately regress target scale via anchor free [53, 56] or anchor based designs [16, 26, 27]. Although these methods can achieve favorable performance on several tracking benchmarks, they are still inferior to the recent state-of-the-art deep trackers, including online learning-based trackers (e.g., ATOM [13], DiMP [5], PrDiMP [14]), and transformer trackers (e.g., TransT [9], STARK [52], OSTrack [54] and DropTrack [47]). However, these deep trackers are trained with large-scale tracking datasets with expensive

bounding box annotations, which lead to both large annotation time and fee costs. To ease both of the costs, in this paper, we propose to use low-cost point annotations to train the above methods with the proposed SoCL framework.

Annotation Types. Various annotation types have been explored in computer vision. Bounding box annotations have been widely used in various tasks, e.g., object detection [35] and object tracking [23]. [37] shows that annotating a bounding box takes ∼10.2s. Mask annotations are also used to perform more fine-grained tasks [43, 48], but takes ∼78s per instance [3]. To ease the annotation time, [3, 34] propose to learn models from point annotations for object detection and semantic segmentation. However, these methods are designed for static images, which are not applicable for video representation learning. In contrast, our SoCL can effectively learn temporal correspondences from point annotations in videos and be directly applied for online tracking.

Contrastive Learning. Contrastive learning methods have achieved leading performance on unsupervised representation learning. Commonly, a memory bank is needed to store pre-computed image features for more efficient and effective learning [19, 31, 40, 49, 57]. Recent works [7, 41] propose to use a large batch size (e.g., 8192) to include large numbers of negative samples in each mini-batch, or even remove the negatives samples to only focus on target prediction [8, 18, 33]. In addition, video frame-level contrastive learning is proposed in [50]. However, these methods are designed to learn from ImageNet [36] images or Kinetics videos [22], which main contain target objects. In our case, no explicit target bounding boxes are provided and most of the video frames are noisy (i.e., containing cluttered backgrounds and distractor objects), which makes existing methods ineffective. To address these issues, we propose to incorporate target objectness prior (TOP) maps into end-to-end contrastive learning from noisy videos.

3. Proposed Method

Our goal is to learn effective visual tracking representations from low-cost annotations, i.e., point annotations. Moreover, the learned representations should be generic and effective for various deep trackers, including both Siamese [4, 55] and correlation filter [12, 21] trackers. An overview of our proposed soft contrastive learning (SoCL) framework is shown in Fig. 2. Previous tracking frameworks [4, 21, 26, 27] are trained using bounding box annotations, which contain scale information that defines the extent of the tracked object on the feature map. However, in our formulation, point-wise annotations do not provide explicit scale information. Thus, we first generate a *target objectness prior* (TOP) for each image using its point annotation, which estimates the likely extent of the target. To learn more discriminative features, we propose multiple sample

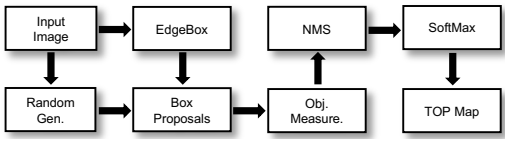


Figure 3. Target objectness prior (TOP) map generation for a given input image, which consists of proposal generation (including both EdgeBox and random proposal generation) and aggregation of objectness measurements.

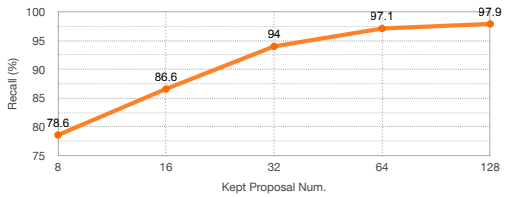


Figure 4. A plot of target recall at various numbers of kept proposals after NMS. The evaluation uses an overlap threshold of 0.5.

generation methods based on the TOP map probabilities, including global soft template generation (GST), soft negative sample (SNS) generation and local soft template (LST) generation. The generated samples can be used to learn an effective tracking model via our soft contrastive learning loss.

3.1. Target Objectness Prior (TOP) Map

The TOP map contains the probability that each pixel belongs to the target object, and is computed by aggregating over the objectness scores of many object proposal boxes sampled over the annotated point (see Fig. 3). Specifically, given video frame $\mathcal{I}_i$ and corresponding annotated target center $\mathcal{P}_i$, we generate 5000 random proposals centered at $\mathcal{P}_i$ with various scales and aspect ratios. Using only proposals centered on $\mathcal{P}_i$ may introduce a center bias to the TOP maps, but the point $\mathcal{P}_i$ might have spatial annotation noise (since it is difficult to click on the exact center of an object). Thus, to alleviate this bias, we also generate proposals using EdgeBox [58]. Specifically, for each frame, we use EdgeBox to generate 1,000 proposals and keep the proposals whose center locations are close to the annotated location (i.e., within 30 pixels). These EdgeBox proposals are then combined with the random proposals.

We evaluate the objectness scores for all generated proposals using [2]. Note that we only use multi-scale saliency, color contrast and edge density cues for the objectness measurement, and exclude the superpixel cue used in [2]. This variant runs 3.3× faster than the original method and also achieves high recall in our case (see Fig. 4). Next, non-maximum suppression (NMS) is applied with an overlap threshold of 0.7 to keep the top-64 proposals for each frame, and filter out redundant proposals.

Finally, to calculate the target objectness score at each location in the frame, we sum over the scores of all the bounding boxes that cover that location, yielding a score map. The softmax function (over locations) is then applied to the score map to obtain TOP map $\mathbf{h}_i$ for the video frame

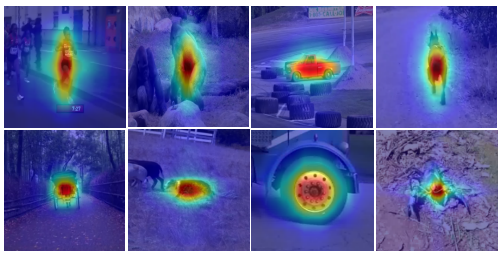


Figure 5. Examples of target objectness prior (TOP) maps generated by using the combination of random and EdgeBox proposal generation.

$\mathcal{I}_i$. Fig. 5 shows examples of TOP maps. Each score in $\mathbf{h}$ represents the probability that the corresponding location belongs to the target. Generally, the peak score is located near the annotated location, and the scores gradually decrease moving towards the background regions.

In our implementation, calculating the TOP map takes about 0.4 seconds for one video frame (512×512), which only needs to be performed once before the training. Since the point annotation in each frame takes about **1.87** seconds [34], the overall per-frame cost of the point annotation and TOP map is **2.27** seconds, which is **4.5**$\times$ faster than bounding box annotation (10.2s).

3.2. Soft Contrastive Learning

We next introduce how to use the generated TOP maps for soft contrastive learning. There are three types of soft samples generated in our SoCL framework, including global and local soft templates (GST and LST) and soft negative samples (SNS), which respectively aggregate global/local target and hard negative counterparts in context features for sample generation.

3.2.1 Global Soft Template (GST) Generation

The overall pipeline of global soft template (GST) generation is shown in Fig. 2a. Given a video frame image $\mathcal{I}_i$ and its generated TOP map $\mathbf{h}_i$, the corresponding GST $\mathbf{z}_i \in \mathbb{R}^C$ is calculated as a weighted sum over the spatial locations, with higher weight given to locations that are more likely to be part of the object (according to the TOP map),

$$\mathbf{z}_i = f(\mathcal{I}_i)^T \mathbf{h}_i, \; f(\mathcal{I}_i) \in \mathbb{R}^{HW \times C}, \mathbf{h}_i \in \mathbb{R}^{HW \times 1}, \quad (1)$$

where $f(\cdot)$ is an embedding function (feature extractor), which is implemented as a deep neural network. The above generation is efficient since it only relies on a single matrix multiplication operation between two HW-dim. vectors.

During the mini-batch training, for each training video, we randomly select two frames $(\mathcal{I}_i, \mathcal{I}_j)$ to construct a pair of GSTs $(\mathbf{z}_i, \mathbf{z}_j)$, which are a pair of positive samples (i.e., the same object). The GSTs in the other videos are considered as negative samples to $(\mathbf{z}_i, \mathbf{z}_j)$. However, these negative samples are usually easy negatives (since that object is completely different), and the number of negative samples are also limited. In the next subsection, we introduce our soft negative sample (SNS) generation for more effective representation learning.

3.2.2 Soft Negative Sample (SNS) Generation

Previous works [24, 44] show that negative samples play an essential role in contrastive representation learning. These methods commonly use a large mini-batch size or specifically design a hard negative selection strategy to include more hard negatives in one mini-batch. However, larger memory cost is also needed for additional negative samples. In this work, we propose to fully leverage the pre-computed context features for memory-efficient soft negative sample (SNS) generation in the feature space.

In the sampled video, $f(\mathcal{I}_i)$ and $f(\mathcal{I}_j)$ are regarded as two context feature vectors, which contain both target and background information. The pipeline of SNS generation is shown in Fig. 2b, and the goal is to aggregate hard negative features in the feature map to generate the SNSs $\hat{\mathbf{z}}_i, \hat{\mathbf{z}}_j \in \mathbb{R}^C$. First, the similarity maps $(\mathbf{g}_i, \mathbf{g}_j)$ between the GSTs and each location in the feature maps are computed,

$$\mathbf{g}_i = f(\mathcal{I}_i) * \mathbf{z}_i, \quad \mathbf{g}_j = f(\mathcal{I}_j) * \mathbf{z}_i, \quad (2)$$

where $\mathbf{g}_i, \mathbf{g}_j \in \mathbb{R}^{HW}$, and $*$ is the convolution operation. The similarity maps contain high responses for both the target and the hard negative background. Next, the high responses for the target are masked out using the TOP maps $(\mathbf{h}_i, \mathbf{h}_j)$, which yields the hard-negative maps

$$\hat{\mathbf{h}}_i = \sigma(s_b(\mathbf{g}_i, \mathbf{h}_i)), \quad \hat{\mathbf{h}}_j = \sigma(s_b(\mathbf{g}_j, \mathbf{h}_j)), \quad (3)$$

where $s_b(\cdot)$ is the background selection function (discussed later), and $\sigma(\cdot)$ is the softmax function. Finally, the SNS $(\hat{\mathbf{z}}_i, \hat{\mathbf{z}}_j)$ are generated as a weighted sum of the feature maps over spatial locations, weighted by the hard-negative maps,

$$\hat{\mathbf{z}}_i = f(\mathcal{I}_i)^T \hat{\mathbf{h}}_i, \quad \hat{\mathbf{z}}_j = f(\mathcal{I}_j)^T \hat{\mathbf{h}}_j, \quad (4)$$

Thus the generated SNS contain features from the background with high response to the GST, which helps contrastive learning to find background discriminative features.

Background selection function. The background selection function $s_b(\cdot)$ masks out the target object locations in the similarity map $\mathbf{g}$ using the TOP map $\mathbf{h}$. First, the locations with high score in $\mathbf{h}$ are selected by thresholding its cumulative sum. Specifically, elements in $\mathbf{h}$ are sorted in descending order, yielding the sorted vector $\mathbf{a}$ and inverse mapping $\phi(k)$ such that $h_{\phi(k)} = a_k$. Next, the first location that gives a cumulative sum of at least θ_b is computed,

$$q^* = \min_{\sum_{k=1}^{q} a_k \geq \theta_b} q, \quad (5)$$

where $\theta_b \in [0, 1]$ is a fixed threshold, and thus the dimensions $\{1, \cdots, q^*\}$ in $\mathbf{a}$ correspond to the target locations

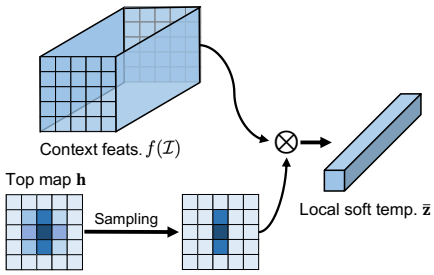


Figure 6. Overview for generating local soft templates (LST) by sampling high-score locations in the TOP map

with high score. Second, the corresponding locations in the similarity map $\mathbf{g}$ are masked out,

$$s_b(\mathbf{g}, \mathbf{h}) = \left[\begin{cases} -\infty, & i \in \{\phi(k)\}_{k=1}^{q^*}, \\ \mathbf{g}_i, & \text{otherwise.} \end{cases} \right]_i \tag{6}$$

Note that θ_b controls the target location selection, and it should be set to a relatively large value in order to include most of the target locations. Otherwise, too many target (positive) features are included in the generated SNS, which degrades the learning. In our implementation, $\theta_b = 0.8$.

3.2.3 Local Soft Template (LST) Generation

The TOP map gives the likely extent of the object, but there could be some errors in the map. To further enrich positive views and to make the learning robust to these errors, we propose to generate a LST $\bar{\mathbf{z}}$ for each GST $\mathbf{z}$ by aggregating over a subset of target locations based on the TOP map $\mathbf{h}$ (see Fig. 6). Similar to SNS, first the locations with high score in the TOP map $\mathbf{h}$ are selected by finding the top accumulated scores via the selection function

$$s_t(\mathbf{h}) = \left[\begin{cases} \mathbf{h}_i, & i \in \{\phi(k)\}_{k=1}^{q^*}, \\ 0, & \text{otherwise,} \end{cases} \right]_i, \tag{7}$$

where q^* is computed as in (6) but using threshold θ_p. Second, the LST is generated as a weighted sum over the feature map, aggregating features with high-probability to be the object,

$$\bar{\mathbf{z}} = f(\mathcal{I})^T \psi(s_t(\mathbf{h})), \tag{8}$$

where $\psi(\cdot)$ is a total sum normalization function.

Here θ_p controls how many high-scoring locations are selected for the LST, with $\theta_p \to 1$ selecting more more complete LSTs ($\theta_p = 1$ is equivalent to the GST). In order to make training robust, we randomly generate the LSTs based on the TOP map probabilities, by sampling θ_p from a uniform distribution over $[b_p, 1)$ each time we generate an LST. In our implementation, hyperparameter $b_p = 0.6$.

3.2.4 Soft Contrastive Learning Loss

Each mini-batch in our contrastive learning contains 2 sampled frames from N sampled training videos, totaling $2N$ frames. The GST, SNS, and LST are generated for each frame in the mini-batch, and then collected to form the positive and negative sample sets for contrastive learning.

Negative Sample Set. For a GST pair $(\mathbf{z}_i, \mathbf{z}_j)$ from the same video, its negative samples come from three sources in the mini-batch: 1) $2(N-1)$ GSTs that are generated from the other $N-1$ videos; 2) $4N$ SNS generated from all videos; 3) $2N$ additional hard negative samples created using a mix-up strategy [24]. Our mix-up strategy generates a novel hard negative example $\hat{\mathbf{z}}'_i$ for each $\mathbf{z}_i$ by interpolating its two hardest SNS $(\hat{\mathbf{z}}_1, \hat{\mathbf{z}}_2)$, i.e., $\hat{\mathbf{z}}'_i = \lambda\hat{\mathbf{z}}_1 + (1-\lambda)\hat{\mathbf{z}}_2$, where $\lambda \in (0.5, 1)$ is the interpolation factor.

In total there are $8N-2$ negative samples for each GST pair $(\mathbf{z}_i, \mathbf{z}_j)$, which is denoted as the negative sample set $\mathcal{N}_{ij} = \{\hat{\mathbf{z}}_k\}_{k=1}^{8N-2}$. Our negative sample set is $4\times$ larger than the baseline of only using the positive samples (the $2N-2$ GSTs) from the other videos. Meanwhile, the SNS generated from the pre-computed context features have no additional memory cost.

Global-to-global Contrastive Learning. Suppose that we treat the global soft template $\mathbf{z}_i$ as a query, based on its global positive view $\mathbf{z}_j$ and the negative sample set $\mathcal{N}_{ij}$, a global-to-global contrastive learning loss is computed as

$$\mathcal{L}(\mathbf{z}_i, \mathbf{z}_j, \mathcal{N}_{ij}) = -\log \frac{\exp(\mathbf{z}_i^T \mathbf{z}_j / \tau)}{\sum_{\hat{\mathbf{z}}_k \in \mathcal{N}_{ij}} \exp(\mathbf{z}_i^T \hat{\mathbf{z}}_k / \tau)}, \tag{9}$$

where τ is a temperature hyper-parameter.

Global-to-local Contrastive Learning. Global-to-local contrastive learning is also conducted to make the learned representations robust to scale variations and partial occlusion. The global-to-local contrastive loss uses the LSTs $\bar{\mathbf{z}}_i$ and $\bar{\mathbf{z}}_j$ in place of $\mathbf{z}_j$, which is given by $\mathcal{L}(\mathbf{z}_i, \bar{\mathbf{z}}_j, \mathcal{N}_{ij})$ and $\mathcal{L}(\mathbf{z}_i, \bar{\mathbf{z}}_i, \mathcal{N}_{ij})$.

Thus, the overall training loss is for $(\mathbf{z}_i, \mathbf{z}_j)$ is the sum of the global-to-global and global-to-local losses:

$$\mathcal{L}_{all} = \mathcal{L}(\mathbf{z}_i, \mathbf{z}_j, \mathcal{N}_{ij}) + \mathcal{L}(\mathbf{z}_i, \bar{\mathbf{z}}_j, \mathcal{N}_{ij}) + \mathcal{L}(\mathbf{z}_i, \bar{\mathbf{z}}_i, \mathcal{N}_{ij}). \tag{10}$$

3.3. Tracking Applications

After training the model with the overall loss in (10), we use the context feature extractor $f(\mathcal{I})$ as the backbone $\mathcal{M}$ for various tracking applications.

Siamese Tracking. We directly integrate the learned $\mathcal{M}$ into a basic SiamFC [4] tracking framework without further modifications. The online tracking steps are also same as [4], where the template and search features are extracted using our $\mathcal{M}$ and a cross-correlation operation is applied for target localization. The obtained basic tracker is denoted as "SoCL-SiamFC", which can show the effectiveness of our learned backbone $\mathcal{M}$.

Correlation Filter Tracking. We also validate our learned backbone $\mathcal{M}$ in an online updating-based correlation filter (CF) tracking framework. CF trackers learn a target appearance model by solving a ridge regression problem

in the frequency domain, and the appearance model is continuously updated during online tracking. ImageNet pre-trained models (e.g., VGGNet [38]) are usually employed as feature extractors by CF trackers, which can obtain strong performance. Here we employ our learned $\mathcal{M}$ as the feature extractor for the efficient convolution operator-based CF framework [12], and denote it as "SoCL-CF".

Scale Regression-based Tracking. State-of-the-art scale regression-based trackers [9, 26, 27] commonly use a bounding box regression branch to estimate the target scale. The training of this regression branch requires bounding box annotations. Here, we propose to combine the point annotations with a set of sparse box annotations, in order to train these scale regression trackers with low annotation cost. Specifically, bounding box annotations are provided in every T frames, such that each training video is divided into multiple short snippets with frame length of T. In each snippet, a bounding box defines the tracked object in the first frame and center point annotations are given in the following frames. We then run our SoCL-CF to generate pseudo bounding boxes for each snippet. Note that when the estimated target location by SoCL-CF is far away from the corresponding annotated center point (i.e., > 20 pixels), we treat it as a tracking failure and use the annotated point as the estimated bounding box center. Meanwhile, the tracker is also corrected to move to the annotated locations.

Our new annotation scheme is also quite efficient and the breakdown of its overall per-frame annotation cost is as follows: 1) overall per-frame point annotation cost $2.27(1-\frac{1}{T})$ (excluding the point annotation for the first frame); 2) running speed of SoCL-CF (0.1s per frame); 3) sparse bounding box annotation cost ($\frac{10.2}{T}$ per frame). We set $T = 10$, so the overall per-frame annotation cost for this new scheme is 3.16s, which is about $3.2\times$ faster than dense bounding box annotation. We use this new schema to generate pseudo bounding boxes on GOT-10k, and then train a scale regression-based tracker, TransT [9]). The obtained tracker is denoted as "SoCL-TransT". Note that we use the same training configurations in [9].

4. Experiments

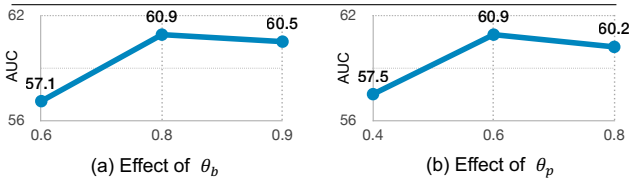
We present tracking experiments showing the efficacy of our backbone learned from point annotations. We first present ablation study, and then compare our method versus baselines at different operating points, such using the same annotation cost and using the same training videos.

4.1. Implementation Details

For SiamFC[4], $\mathcal{M}$ is an AlexNet-like network that is randomly initialized, while for SoCL-Siam and SoCL-CF, $\mathcal{M}$ is ResNet-18 [20] pre-trained on ImageNet. For SoCL-CF, we use the features extracted from *Conv1* and *Layer3* of ResNet-18. Different from [12], there are no color

Table 1. Consistent improvements of AUCs/EAOs (on OTB-13 and VOT-16, respectively) achieved by using global soft template (GST) generation, soft negative sample (SNS) generation, negative mixup and local soft template (LST) generation.

GST Gen.	SNS Gen.	Neg. Mix.	LST Gen.	AUC / EAO
✓				55.1 / 0.215
✓	✓			57.8 / 0.228
✓	✓	✓		58.1 / 0.231
✓	✓	✓	✓	**60.9 / 0.244**



(a) Effect of θ_b (b) Effect of θ_p

Figure 7. Ablation study of θ_b and θ_p that control SNS and LST sample generation respectively on OTB-13.

names features used in SoCL-CF, and the feature extractor in SoCL-CF[1] is ResNet-18 rather than VGG-M [38]. The baseline of SoCL-CF is denoted as Res18-CF, which uses the ImageNet pre-trained ResNet-18 as feature extractor. We use mini-batches of 192, and Adam optimizer [25] with learning rates of 3e-4 for training SoCL-Siam and SoCL-CF. SoCL-Siam and SoCL-CF are trained for 1000 and 500 epochs with a learning rate decay of 0.1 at 500 and 50 epochs, respectively. In addition, $\tau = 0.5$, and we experimentally set $\theta_b = 0.8$ and $b_p = 0.6$ (see Sec. 4.2).

Point annotations in tracking datasets. For fair comparison with box annotations and to better validate the effectiveness of our SoCL , we assume that the point annotation noise is similar to the noise in box annotations. Based on this assumption, we generate center point annotations from the original box annotations in GOT-10k [23], and then train SoCL-Siam and SoCL-CF.

4.2. Ablation Study

Positive/negative sample generation. In this experiment, we show that there is consistent improvement for each module that generates positive/negative samples for SoCL (see AUC on OTB-13 and EAO on VOT-16 in Table 1). First, we only use the GSTs for contrastive learning, where GSTs from one video form the positive set, and those from other videos form the negative set. The learned model is integrated to SiamFC for evaluation, and its AUC is better than the original method learned with a pixel-wise binary-cross entropy (BCE) loss (see Fig. 8), which shows that our SoCL can learn more effective representations from the generated noisy TOP maps than the traditional learning.

Second, adding the SNS into the negative set for CL improves AUC by 2.7% and EAO by 1.3% – the generated negative samples are hard negative samples that are similar to the positive GSTs, which improves the discrimination power of the tracking model. Moreover, the increased num-

[1]SoCL-CF is based on a 3rd party implementation of ECO: https://github.com/fengyang95/pyCFTrackers.git

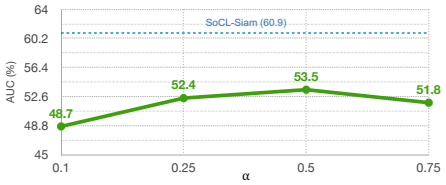


Figure 8. Comparison of SoCL-Siam, which is trained with contrastive learning on point annotations, and BCE-Siam, which is trained with BCE loss on pseudo bounding boxes. The pseudo bounding boxes are generated from the TOP maps using threshold α.

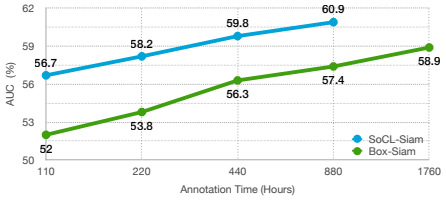


Figure 9. Comparison of Box-Siam and SoCL-Siam trained using the same annotation time costs (AUC on OTB-13).

Table 2. Comparison of Box-Siam and SoCL-Siam trained using the same annotation time costs (i.e., hours) in terms of AUC on OTB-15. The best results are highlighted.

Annotation time cost	110h	220h	440h	880h
Box-Siam	51.7	54.0	55.2	57.0
SoCL-Siam	**53.3**	**54.9**	**56.5**	**57.8**

ber of negative samples also improves the lower bound (on target mutual information), thus leading to better results.

Third, using negative mix-up to further augment the negative set slightly improves the AUC. Since we perform the negative mix-up in the feature space, there is no additional memory cost, and our SoCL can gain additional benefits.

Finally, augmenting the positive set using LSTs improves the AUC by 2.8% to 60.9% and EAO by 1.3% to 0.244, showing that the sampled LSTs effectively mimic targets with partial occlusion or large appearance variations.

Effect of θ_b and θ_p. The ablation study for the effect of θ_b and θ_p is in Fig. 7. Using small values for θ_b degrades the AUC, since the SNSs will contain too many target features. Also, using small values of θ_p also degrades AUC, since the LSTs will not encode enough target features

Learning from pseudo bounding boxes. An alternative training framework could use the TOP maps to generate pseudo bounding boxes for standard BCE training. Specifically, we use an adaptive threshold α on each frame to generate a pseudo bounding box, and then follow previous methods [4, 55] by training SiamFC with a BCE loss. The results are shown in Fig. 8 using different thresholds (α). Using the pseudo bounding boxes and standard BCE loss has degraded performance compared to our SoCL, which is due to the noise in the TOP maps being transferred to the pseudo bounding boxes. In contrast, our soft representations are more robust to the noisy TOP maps.

4.3. Comparison with Same Annotation Time Cost

We next compare our SoCL to a fully-supervised baseline trained with bounding-boxes (denoted as Box-Siam) under

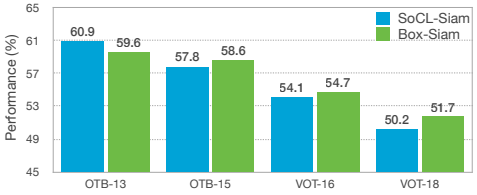


Figure 10. Comparison of Box-Siam and SoCL-Siam trained on the whole GOT-10k dataset. The annotation time costs of Box-Siam and SoCL-Siam are respectively 3,967 and 880 hours. The evaluation is conducted on OTB13/15 (AUC) and VOT16/18 (Accuracy).

Table 3. Comparison of Res18-CF and SoCL-CF.

Method	OTB13 DPR/AUC	OTB15 DPR/AUC	UAV123 DPR/AUC	VOT18 EAO	LaSOT AUC
Res18-CF	91.3/68.5	87.8/65.9	70.1/51.8	0.207	31.0
SoCL-CF	**92.7/69.6**	**89.7/67.1**	**72.3/52.6**	**0.216**	**32.2**

Table 4. Comparison of total fees for BCE learning schema from box annotations and the proposed SoCL schema w/ point annotations on GOT-10k. Amounts are in US dollars.

Schema	Anno. Fee	Training Fee	Total
BCE w/ Boxes	$50,400	$7.2	$50,407.2
SoCL w/ Points	$7,000	$21.6	$7,021.6

the same time cost of annotation. We randomly sample videos from GOT-10k to meet a specific total annotation time requirement for both Box-Siam and SoCL-Siam.

The results on OTB-13 and OTB-15 are presented in Fig. 9 and Table 2. The proposed SoCL-Siam outperforms Box-Siam by large margins for each annotation time cost on OTB-13, indicating that our SoCL effectively learns tracking representations from low-cost point annotations. One interesting phenomena is that the performance gap is larger (56.7 vs. 52.0) for very limited annotation cost (e.g., 110 hours). This is mainly because: 1) SoCL learns features by comparing objects from the same video and other videos, while Box-Siam treats each video independently, thus Box-Siam is more likely to overfit on small number of training videos; 2) relatively more video frames can be used by SoCL due to the less per-frame annotation time cost, even though the total annotation hours are limited. Our SoCL-Siam also achieves better performance than Box-Siam under each total annotation time costs on OTB-15 (Table 2).

We next compare SoCL-Siam and Box-Siam trained on the whole GOT-10k. Note that the total annotation time cost for Box-Siam is much larger than that of SoCL-Siam (3,967 versus 880 hours). Fig. 10 shows the performance on various test datasets. Although SoCL-Siam only uses weak supervision and much less total annotation time, SoCL-Siam still achieves comparable results to Box-Siam, especially on the OTB datasets. Despite the weak supervision of point annotations that do not contain scale information, the proposed SoCL effectively learns discriminative features by exploring the relationships between targets and hard negative samples in each training mini-batch.

4.4. Improving correlation-filter trackers

We next show that our SoCL also improves online cor-

Table 5. Comparison of our SoCL-TransT and state-of-the-art deep trackers on GOT-10k [23], TrackingNet [32] and LaSOT [17]. ATC denotes the annotation time cost (hours) of the training set for the tracker.

Trackers	ATC	GOT-10k		TrackingNet		LaSOT	
		AO	$SR_{0.5}$	AUC	P_{Norm}	AUC	P
TransT-GOT [9]	4.0K	67.1	76.8	-	-	-	-
KYS [6]	11.9K	63.6	75.1	74.0	80.0	55.4	-
Ocean [56]	34.6K	61.1	72.1	-	-	56.0	56.5
SiamFC++ [51]	42.5K	59.5	69.5	75.4	80.0	54.4	54.7
SiamRPN++ [27]	30.6K	51.7	61.6	73.3	80.0	49.6	49.1
DiMP [5]	19.0K	61.1	71.7	74.0	80.1	56.9	56.7
ATOM [13]	15.0K	55.6	63.4	70.3	77.1	51.5	50.5
ROAM++ [53]	26.6K	46.5	53.2	67.0	75.4	44.7	44.5
CGACD [15]	34.6K	-	-	71.1	80.0	51.8	-
D3S [1]	4.3K	59.7	67.6	72.8	76.8	-	-
SoCL-TransT	**1.2K**	62.2	72.4	75.0	80.5	56.0	56.9

relation filter trackers by learning better feature representations. We show the comparison between the proposed SoCL-CF and the baseline Res18-CF in Table 3. SoCL-CF achieves consistent improvements over Res18-CF on all the five test datasets. This demonstrates that the proposed SoCL are beneficial for both offline-learning Siamese and online-learning CF trackers. In addition, the total annotation time cost (880h) for gaining these improvements is acceptable.

4.5. Comparison of Total Fees

We compare the total fee (dollar cost) between the two learning schemas: our SoCL using point annotations (SoCL w/ Points) and traditional BCE using bounding box annotations (BCE w/ Boxes) [4, 55]. Each bounding box annotation costs \$0.036 using Amazon Mechanical Turk, while each center point annotation costs \$0.005 [34]. Since GOT-10k contains about 1.4M instances, the overall annotation fees are \$50,400 and \$7,000 for BCE and SoCL. For training time, SiamFC takes about 8 hours for training on GOT-10k with a single GPU card, while ours SoCL takes about 24 hours. A single V100 GPU (p2.xlarge) instance on Amazon EC2 costs \$0.90 per hour, thus yielding training costs of \$7.2 and \$21.6 for BCE and SOCL. As shown in Table 4, despite having longer training times, our SoCL w/ Points schema is over 7$\times$ less expensive than BCE w/ Boxes.

4.6. Comparison with State-of-the-art Trackers

We compare our SoCL-TransT with state-of-the-art deep trackers on GOT-10k [23], TrackingNet [32] and LaSOT [17] in Table 5. Our SoCL-TransT only requires 1.2K annotation hours on GOT-10k, which is significantly lower than those of other trackers. Compared to the fully supervised baseline TransT-GOT, our SoCL-TransT achieves 92.7% and 94.3% of the AO and $SR_{0.5}$ performances of the fully-supervised baseline, while reducing annotation cost by 70%. In addition, SoCL-TransT performs favorably against state-of-the-art deep trackers on LaSOT and TrackingNet, even thought it hass low annotation cost.

Table 6. Comparison of SoCL trained on a subset of GOT-10k w/ and w/o added annotation noise.

Method	OTB15	VOT16	VOT18	UAV123	GOT-10k val
	AUC	Acc	Acc	AUC	AO
SoCL	53.8	51.5	46.8	45.0	47.2
SoCL w/ added noise	53.6	51.0	46.2	44.2	46.6
Difference	0.2	0.5	0.6	0.8	0.6

4.7. Learning from Noisy Point Annotations

In practice, the employed annotators may not be well trained or careful enough during the annotation process, which may lead to noisy point annotations. In [34], the average error of point annotations, i.e., the distance between the annotated location and the GT, was found to be 19.5 pixels for object detection tasks. In order to mimic such noisy annotations, we randomly pick 1000 videos from GOT-10K as our dataset, and add a 20 pixel shift with random direction to each point annotation. We use this noisy dataset to train SoCL-Siam, and compare it with the model trained using the dataset without adding noise.

Table 6 shows the results. The performance differences between SoCL with and without annotation noise is not large, with drops ranging from 0.2% to 0.8%. This indicates that our SoCL can learn robust representations from noisy annotations. Specifically, SoCL uses soft representations instead of performing strict pixel-wise matching (e.g., binary-cross entropy loss) like previous methods [4, 26, 27], which enables SoCL to be more robust to noisy data.

Furthermore, the EdgeBox proposal generation also makes SoCL robust to annotation noise. We calculate the average error distance between the GT target center and the mean of the generated proposals' centers in each frame. The average error distance is 14.1 pixels, which is less than the original 20 pixels of the annotation noise. Thus the generated TOP maps are also robust to noisy annotations.

5. Conclusion

This paper proposes a novel soft contrastive learning (SoCL) framework to learn tracking representations from low-cost single point annotations. To facilitate the learning, we propose several memory-efficient sample generation strategies including the generation of global and local soft templates and soft negative samples. Although a large number of samples can be included in SoCL for one mini-batch training, the whole training is memory-efficient and can be conducted on a single GPU (e.g., RTX-3090). We successfully apply the learned representations of SoCL to both Siamese and correlation filter tracking frameworks. Moreover, we design a new framework to train bounding box regression-based trackers.

6. Acknowledgment

This research was funded by a Strategic Research Grant (Project No. 7005665) from City University of Hong Kong.

References

[1] W. Yuan adn Y. Micheal and Q. Chen. Self-supervised object tracking with cycle-consistent siamese networks. In *arxiv:2008.00637*, 2020. 8

[2] B. Alexe, D. Thomas, and F. Vittorio. Measuring the objectness of image windows. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2189–2202, 2012. 3

[3] A. Bearman, O. Russakovsky, and V. Ferrari. What's the point: Semantic segmentation with point supervision. In *ECCV*, pages 549–565, 2016. 3

[4] L. Bertinetto, J. Valmadre, J.F. Henriques, A. Vedaldi, and P.H.S. Vedaldi. Fully-convolutional siamese networks for object tracking. In *ECCVW*, pages 850–865, 2016. 2, 3, 5, 6, 7, 8

[5] G. Bhat, M. Danelljan, L. V. Gool, and R. Timofte. Leanring discriminative model prediction for tracking. In *ICCV*, pages 6182–6191, 2019. 2, 8

[6] G. Bhat, M. Danelljan, and L. Van Gool. Exploiting scene information for object tracking. In *ECCV*, pages 205–221, 2020. 8

[7] T. Chen, S. Kornblith, and M. Norouzi. A simple framework for contrastive learning of visual representations. In *arXiv:2002.05709*, 2020. 3

[8] X. Chen and K. He. Exploring simple siamese representation learning. In *CVPR*, pages 15750–15758, 2021. 3

[9] X. Chen, B. Yan, and J. Zhu. Transformer tracking. In *CVPR*, 2021. 1, 2, 6, 8

[10] M. Danelljan, G. Hager, F. S. Khan, and M. Felsberg. Convolutional features for correlation filter based visual tracking. In *ICCVW*, pages 58–66, 2015. 2

[11] M. Danelljan, A. Robinson, F. S. Khan, and M. Felsberg. Beyond correlation filters: learning continuous convolution operators for visual tracking. In *ECCV*, pages 472–488, 2016.

[12] M. Danelljan, G. Bhat, F. S. Khan, and M. Felsberg. Eco: Efficient convolution operators for tracking. In *CVPR*, pages 21–26, 2017. 2, 3, 6

[13] M. Danelljan, G. Bhat, F. S. Khan, and M. Felsberg. Atom: Accurate tracking by overlap maximization. In *CVPR*, pages 4660–4669, 2019. 2, 8

[14] M. Danelljan, L. Gool, and R. Timofte. Probabilistic regression for visual tracking. In *CVPR*, pages 7183–7192, 2020. 2

[15] F. Du, P. Liu, W. Zhao, and X. Tang. Correlation-guided attention for corner detection based visual tracking. In *CVPR*, 2020. 8

[16] H. Fan and H. Ling. Siamese cascaded region proposal networks for real-time visual tracking. In *ICCV*, 2019. 2

[17] H. Fan, L. Lin, and F. Yang. Lasot: A high-quality benchmark for large-scale single object tracking. In *CVPR*, pages 5374–5383, 2019. 1, 8

[18] J. Grill, F. Strub, and F. Altche. Bootstrap your own latent: A new approach to self-supervised learning. In *arXiv:2006.07733*, 2020. 3

[19] K. He and H. Fan adn Y. Wu. Momentum contrast for unsupervised visual representation learning. In *CVPR*, pages 9729–9738, 2020. 3

[20] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 6

[21] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista. High-speed tracking with kernelized correlation filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(3):583–596, 2015. 3

[22] C. Hillier, S. Vijayanarasimhan, and F. Viola. The kinetics human action video dataset. In *arXiv:1705.06950*, 2017. 3

[23] L. Huang, X. Zhao, and K. Huang. Got-10k: A large high-diversity benchmark for generic object tracking in the wild. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019. 1, 3, 6, 8

[24] Y. Kalantidis, M.B. Sariyildiz, and N. Pion. Hard negative mixing for contrastive learning. In *arXiv:2010.01028*, 2020. 4, 5

[25] Diederik P. Kingma and Jimmy Lei Ba. Adam: A method for stochastic optimization. In *arXiv:1412.6980*, 2014. 6

[26] B. Li, W. Wu, Z. Zhu, and J. Yan. High performance visual tracking with siamese region proposal network. In *CVPR*, pages 8971–8980, 2018. 1, 2, 3, 6, 8

[27] B. Li, W. Wu, Q. Wang, F. Zhang, J. Xing, and J. Yan. Siamrpn++: Evolution of siamese visual tracking with very deep networks. In *CVPR*, 2019. 1, 2, 3, 6, 8

[28] Yanjie Liang, Qiangqiang Wu, Yi Liu, Yan Yan, and Hanzi Wang. Robust correlation filter tracking with shepherded instance-aware proposals. In *Proceedings of the 26th ACM International Conference on Multimedia*, page 420–428, 2018. 2

[29] Yanjie Liang, Haosheng Chen, Qiangqiang Wu, Changqun Xia, and Jia Li. Joint spatio-temporal similarity and discrimination learning for visual tracking. *IEEE Transactions on Circuits and Systems for Video Technology*, 2024.

[30] Yi Liu, Yanjie Liang, Qiangqiang Wu, Liming Zhang, and Hanzi Wang. A new framework for multiple deep correlation filters based object tracking. In *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 1670–1674, 2022. 2

[31] G. Lorre, J. Rabarisoa, and A. Orcesi. Temporal contrastive pretraining for video action recognition. In *WACV*, 2020. 3

[32] M. Muller, A. Bibi, and Giancola S. Trackingnet: A large-scale dataset and benchmark for object tracking in the wild. In *ECCV*, pages 300–317, 2018. 1, 8

[33] D. Niizumi, D. Takeuchi, and Y. Ohishi. Byol for audio: Self-supervised learning for general-purpose audio representation. In *arXiv:2103.06695*, 2021. 3

[34] D. Papadopoulos, J. Uijlings, and F. Keller. Training object class detectors with click supervision. In *CVPR*, pages 6374–6383, 2017. 3, 4, 8

[35] E. Real, J. Shlens, and S. Mazzocchi. Youtube-boundingboxes: A large high-precision human-annotated data set for object detection in video. In *CVPR*, pages 5296–5305, 2017. 3

[36] O. Russakovsky, J. Deng, and et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015. 1, 3

[37] O. Russakovsky, L.J. Li, and F.-F. Li. Best of both worlds: human-machine collaboration for object annotation. In *CVPR*, pages 2121–2131, 2015. 1, 3

[38] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image. In *International Conference on Learning Representations*, 2015. 6

[39] R. Tao, E. Gavves, and A. W.M. Smeulders. Siamese instance search for tracking. In *CVPR*, pages 1420–1429, 2016. 2

[40] Y. Tian, D. Krishnan, and P. Isola. Contrastive multiview coding. In *arXiv:1906.05849*, 2019. 3

[41] Y. Tian, S. Kornblity, and K. Swersky. Big self-supervised models are strong semi-supervised learners. In *Neurips*, 2020. 3

[42] N. Wang, Y. Song, and C. Ma. Unsupervised deep representation learning for real-time tracking. *IJCV*, pages 1–19, 2020. 1

[43] Q. Wang, L. Zhang, and L. Bertinetto. Fast online object tracking and segmentation: A unifying approach. In *CVPR*, 2019. 3

[44] Q. Wu, W. Jia, and A.B. Chan. Progressive unsupervised learning for visual object tracking. In *CVPR*, pages 2993–3002, 2017. 1, 4

[45] Q. WU, Y. Yan, and Y. Liang. Dsnet: deep and shallow feature learning for efficient visual tracking. In *ACCV*, pages 119–134, 2018. 2

[46] Q. Wu, H. Wang, Y. Liu, L. Zhang, and X. Gao. Sat: Single-shot adversarial tracker. *IEEE Transactions on Industrial Electronics*, 67(11):9882–9892, 2019. 2

[47] Q. Wu, T. Yang, Z. Liu, B. Wu, Y. Shan, and A. B. Chan. Dropmae: Masked autoencoders with spatial-attention dropout for tracking tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023. 2

[48] Qiangqiang Wu, Tianyu Yang, Wei Wu, and Antoni B. Chan. Scalable video object segmentation with simplified framework. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 13879–13889, 2023. 3

[49] Z. Wu, Y. Xiong, and S. Yu. Unsupervised feature learning via non-parametric instance discrimination. In *CVPR*, pages 3733–3742, 2018. 3

[50] J. Xu and X. Wang. Rethinking self-supervised correspondence learning: A video frame-level similarity perspective. In *arXiv:2103.17263*, 2021. 3

[51] Y. Xu and Z. Wang adn Z. Li. Siamfc++: Towards robust and accurate visual tracking with target estimation guidelines. In *AAAI*, pages 12549–12556, 2020. 8

[52] B. Yan, H. Peng, and J. Hu. Learning spatio-temporal transformer for visual tracking. In *arXiv:2103.17154*, 2021. 2

[53] T. Yang, P. Xu, and R. Hu. Roam: Recurrently optimizing tracking model. In *CVPR*, pages 6718–6727, 2020. 2, 8

[54] B. Ye, H. Chang, B. Ma, and S. Shan. Joint feature learning and relation modeling for tracking: A one-stream framework. In *ECCV*, pages 341–357, 2022. 2

[55] Z. Zhang and H. Peng. Deeper and wider siamese networks for raal-time visual tracking. In *CVPR*, 2017. 1, 3, 7, 8

[56] Z. Zhang, H. Peng, and Fu J. Ocean: Object-aware anchor-free tracking. In *ECCV*, pages 771–787, 2020. 2, 8

[57] C. Zhuang, A. Zhai, and D. Yamins. Local aggregation for unsupervised learning of visual embeddings. In *ICCV*, 2019. 3

[58] C. Zitnick and P. Dollar. Edge boxes: Locating object proposals from edges. In *ECCV*, pages 391–405, 2014. 3