Deep Object Tracking with Shrinkage Loss

Xiankai Lu, Chao Ma, Jianbing Shen, *IEEE Senior Member*, Xiaokang Yang, *IEEE Fellow*, Ian Reid, Ming-Hsuan Yang, *IEEE Fellow*

Abstract—In this paper, we address the issue of data imbalance in learning deep models for visual object tracking. Although it is well known that data distribution plays a crucial role in learning and inference models, considerably less attention has been paid to data imbalance in visual tracking. For the deep regression trackers that directly learn a dense mapping from input images of target objects to soft response maps, we identify their performance is limited by the extremely imbalanced pixel-to-pixel differences when computing regression loss. This prevents existing end-to-end learnable deep regression trackers from performing as well as discriminative correlation filters (DCFs) trackers. For the deep classification trackers that draw positive and negative samples to learn discriminative classifiers, there exists heavy class imbalance due to a limited number of positive samples when compared to the number of negative samples. To balance training data, we propose a novel shrinkage loss to penalize the importance of easy training data mostly coming from the background, which facilitates both deep regression and classification trackers to better distinguish target objects from the background. We extensively validate the proposed shrinkage loss function on six benchmark datasets, including the OTB-2013, OTB-2015, UAV-123, VOT-2016, VOT-2018 and LaSOT. Equipped with our shrinkage loss, the proposed one-stage deep regression tracker achieves favorable results against state-of-the-art methods, especially in comparison with DCFs trackers. Meanwhile, our shrinkage loss generalizes well to deep classification trackers. When replacing the original binary cross entropy loss with our shrinkage loss, three representative baseline trackers achieve large performance gains, even setting new state-of-the-art results.

Index Terms—Data imbalance, shrinkage loss, regression tracking, classification tracking, Siamese tracking

1 Introduction

Recent years have witnessed a growing demand for visual object tracking algorithms in various vision applications such as robotics, augmented reality, autonomous driving, and human-computer interaction. Existing tracking-by-detection approaches usually construct the target state inference module with deep models. The data imbalance issue arises for deep trackers as a large number of easy training data contribute little to the learning process. For example, in regression learning, a large portion of pixel-to-pixel differences are quite small but their sum dominates the total regression loss. Similarly, in classification, there exists heavy class imbalance due to a limited number of positive samples in the tracking scenario. Although it is well-known that data distribution plays a crucial role in learning and inference models, considerably less attention has been paid to the problem of data imbalance for visual tracking, which heavily hinders the performance of existing deep trackers. In this paper, we aim to address the issue of data imbalance for developing more robust deep tracking algorithms.

The regression trackers [1]–[4] often use linear regression layers on top of convolutional neural networks to learn a pixel-to-pixel dense mapping from input images of target objects to soft response maps, which are usually generated by Gaussian functions. These one-stage regression trackers have recently received increasing attention due to their conveniences in both implementation and computation. However, state-of-the-art deep regression trackers [1]–[3], [5] do not perform as well as their regression counterparts, DCFs trackers [6]–[11], on the benchmark datasets [12], [13]. The end-to-end learnable deep regression trackers have much greater potential to take advantage of large-scale training data than the DCFs trackers, where learning and updating DCFs are independent of deep feature extraction.

We identify the main performance bottleneck of current deep regression trackers [1]–[3] as the issue of data imbalance [14] in regression learning. The majority pixels mostly coming from the background produce small training errors individually but dominate the overall regression loss. However, current regression trackers pay little attention to this issue. As evidenced by the effectiveness, state-of-the-art DCFs trackers improve tracking accuracy by re-weighting sample locations using Gaussian-like maps [15]. In this work, we revisit the shrinkage estimator [16] in regression learning. We propose a novel shrinkage loss to handle data imbalance during learning regression networks. Specifically, we use a Sigmoid-like function to penalize the importance of easy samples coming from the background (*e.g.*, points close to the boundary). This not only improves tracking accuracy but also accelerates the convergence of training regression networks. In addition, we observe that deep regression networks can be further improved by effectively exploiting multi-level semantic abstraction across multiple convolutional layers. We apply residual connections to integrate multiple convolutional layers as well as their output response maps. Since the residual connections and shrinkage loss are fully differentiable, allowing our regres-

- *X. Lu, C. Ma, and X. Yang are with the MoE Key Lab of Artificial Intelligence, AI Institute, Shanghai Jiao Tong University, Shanghai, 200240, China. X. Lu is also with School of Software, Shandong Univeristy, China. E-mail: carrierlxk@gmail.com, {chaoma, xkyang}@sjtu.edu.cn.*
- *J. Shen is with Inception Institute of Artificial Intelligence, Abu Dhabi, UAE. E-mail: shenjianbingcg@gmail.com*
- *I. Reid is with School of Computer Science, The University of Adelaide, Adelaide, 5005, Australia. E-mail: ian.reid@adelaide.edu.au*
- *M.-H. Yang is with School of Electronic Engieering and Computer Science, University of California at Merced, Merced, USA. E-mail: mhyang@ucmerced.edu*
- *The first two authors contribute equally. Corresponding author: Chao Ma*

Fig. 1. Qualitative results of different loss functions for learning one-stage regression trackers on the *Bolt2* sequence [12]. The proposed regression tracker (DSLT*) with shrinkage loss performs much better than that with the L2 and L3 losses.

sion network to be trained end-to-end. Our shrinkage loss helps the proposed deep regression tracker perform well against state-of-the-art methods especially in comparison with DCFs trackers on benchmark settings.

Deep classification trackers [17]–[20] often draw a large number of samples around the target position in the previous frame and then classify each sample as the target object or background. This leads to data imbalance as there is typically a limited number of positive samples compared to negative samples. The majority of negative samples belong to easy training data, which contributes little to classification learning. The way to handle extreme positive-negative class imbalance issue has been extensively studied. For example, hard negative mining [17] uses an empirical hard threshold to filter out easy negative samples. Meanwhile, the positive to negative class ratio can be used as weights to adjust the importance of training samples [21]. Despite the demonstrated successes, these two schemes both rely on empirical threshold values that cannot be optimized end-to-end. In contrast, our shrinkage loss can be seamlessly integrated into deep classification trackers for end-to-end learning. We take the state-of-the-art classification trackers SiameseFC [22] and SiamRPN [20] as baseline algorithms. SiameseFC resembles the proposed regression network in regressing input images into ground-truth labels generated by a Gaussian function. However, SiameseFC binarizes the soft ground-truth map first and hence employs the binary cross entropy (BCE) loss for classification learning. SiamRPN incorporates an additional region proposal network to generate samples based on the SiameseFC tracker. By replacing the original BCE loss with our shrinkage loss, both SiameseFC and SiamRPN achieve significant performance gains on large-scale benchmark datasets. Especially, our shrinkage loss helps SiamRPN advance the state-of-the-art on the VOT-2018 and LaSOT datasets.

Our shrinkage loss shares a similar motivation with the focal loss [23], which is used to improve one-stage object detectors by penalizing easily classified examples. We observe that focal loss also partially decreases the loss from valuable hard samples as well, resulting in minor location changes of detected bounding boxes. This does not affect the detection performance too much as detection results are computed in single image. For object tracking, slight inaccuracies of the estimated target positions will accumulate over time, leading trackers to drift soon. Fig. 1 compares the qualitative results of different loss functions for learning regression trackers on the *Bolt2* sequence [12]. The proposed regression tracker with our shrinkage loss handles data imbalance well and succeeds in tracking the target person with large appearance changes, whereas the L2 and L3 losses cause the regression tracker to fail in a short time.

In summary, the main contributions of this work lie in an effective approach that best alleviates the common data imbalance issue in learning deep models for robust visual tracking. We summarize the contributions as follows:

- We propose a novel shrinkage loss to handle the common issue of data imbalance in learning deep models for visual tracking. The proposed shrinkage loss helps accelerate the convergence of network training as well.
- Equipped with the proposed shrinkage loss, we learn one-stage deep regression models across multiple convolutional layers. We succeed in narrowing the gap between the deep regression trackers and DCFs trackers.
- We show that the proposed shrinkage loss generalizes well to deep classification trackers. Our shrinkage loss advances the performance of two baseline Siamese trackers and MDNet by large margins.
- We extensively evaluate the proposed methods on six benchmark datasets. Extensive experiments demonstrate the effectiveness and efficiency of the proposed shrinkage loss for robust object tracking in comparison with state-of-the-art trackers.

This paper builds upon our conference paper [5] and significantly extends it in various aspects. First, we perform in-depth analysis of the data imbalance issue in deep tracking under *regression learning* and *classification* scenarios. Second, we provide a more generic shrinkage loss and extend it on deep classification tracking. Experimental results on two representative trackers further validate the effectiveness of the proposed shrinkage loss. Third, we exploit a variant of the backbone network used in the deep regression tracking to verify the generalization ability of the proposed multiple-layer feature fusion strategy. Last but not least, we have incorporated two recent large scale datasets (*i.e.*, the VOT-2018 [24] and LaSOT [25]) for numerical evaluations and added more state-of-the-art trackers for comparisons.

2 Related Work

Visual tracking has been an active research topic with several comprehensive surveys [26], [27]. In this section, we first discuss the representative tracking frameworks that learn regression or classification models for inferring the target states. We then review the data imbalance issue in both *regression* and *classification* learning.

Regression Tracking. The one-stage regression tracking framework takes the whole search area as input and directly outputs a response map through a regression model, which

learns a mapping between input features and the target labels usually generated by a Gaussian function [28], [29]. One representative category of one-stage regression trackers are on the basis of discriminative correlation filters[28], [30]–[38], which regress all the circularly shifted versions of the input image into soft labels. By computing the correlation as an element-wise product in the Fourier domain, DCFs trackers achieve the fastest speed thus far. Numerous extensions include KCF [39], LCT [40], [41], MCF [42], MCPF [43] and BACF[44]. With the use of deep features, DCFs trackers, such as DeepSRDCF [6], HDT [7], HCFT [8], C-COT [9] and ECO [10], have demonstrated superior performance on a number of benchmark datasets. In [8], Ma *et al*. proposed to learn multiple DCFs over different convolutional layers and empirically fuse the output correlation maps to locate the target objects. A similar idea was exploited in [9] to integrate multiple response maps. In [10], Danelljan *et al*. proposed reducing feature channels to accelerate learning correlation filters. Despite the impressive performance, DCFs trackers learn and update correlation filters independently of deep feature extraction, benefiting little from end-to-end training.

The second category of one-stage regression trackers are typically on the basis of convolutional regression networks. The FCNT [1], STCT [2], GOTURN [45], and CREST [3] trackers belong to this category. The FCNT makes the first effort to learn regression networks over two convolutional layers. The output response maps from different layers are switched according to their confidence to locate target objects. The STCT algorithm exploits ensemble learning to select CNN feature channels. GOTURN [45] uses fully connected layers to directly regress input features to the apexes of the target bounding boxes rather than soft labels. Similar to GOTURN, Gao *et al*. [46] train an object part level regression network to predict the part center and translation. CREST [3] learns a base network as well as a residual network on the output of last convolutional layer. The output maps of the base and the residual networks are fused to infer target positions. We note that current deep regression trackers do not perform as well as DCFs trackers. We identify the main bottleneck as the data imbalance issue in regression learning. By balancing the importance of training data, the performance of one-stage deep regression trackers can be significantly improved over DCFs trackers.

Classification Tracking. In contrast to the above one-stage regression trackers using soft labels to represent the target states, the one-stage Siamese trackers [22], [48] first binarize the soft labels and then employ a binary cross entropy loss for classification learning. Bertinetto *et al*. [22] substituted the fully connected layer in GOTURN with a cross correlation layer and proposed a novel fully-convolutional Siamese network, namely SiameseFC, which consists of a template branch and a search branch. The CFNet [48] tracker adds a correlation filter to the template branch, yielding a shallow Siamese tracker. More recently, many works [49]–[51], [51]–[57] learn to update SiameseFC to adapt appearance changes, such as extra correlation filter [49], target-specific network [50], [51], [55] and graph neural network [56]. Note that the Siamese trackers perform template matching, which requires a large search region as input. This complicates learning classifiers by introducing imbalanced data from the background.

The two-stage classification tracking performs tracking over two steps. The first stage generates a set of candidate target samples around the previously estimated location using random sampling, regularly dense sampling [29], [58]–[60], or region proposal [61]–[63]. The second stage classifies each candidate sample as the target object or as the background. Numerous efforts have been made to learn a discriminative boundary between positive and negative samples [64]. Examples include the multiple instance learning (MIL) [65] and Struck [66], [67] methods. Recent deep trackers, such as DeepTrack [18], SANet [68] and CNN-SVM [19], all belong to the two-stage classification framework. Recently, the SiamRPN [20] tracker extends the one-stage SiameseFC algorithm with an additional region proposal network (RPN). SiamRPN first takes the template and search images as input and outputs proposals, based on which SiamRPN outputs classification scores as well as the estimated bounding boxes in one forward pass. Despite the favorable performance on the challenging object tracking benchmarks [12], [13], we note that two-stage deep trackers suffer from heavy class imbalance due to a limited number of positive samples in the tracking scenario. Moreover, the positive samples are spatially correlated and redundant.

Data Imbalance. The data imbalance issue has been extensively studied in the learning community [14], [69], [70]. Helpful solutions involve data re-sampling [71]–[73], and cost-sensitive loss [23], [74]–[77]. Our shrinkage loss belongs to the latter strategy by re-weighting the contribution of each sample based on the observed loss. For object detection, hard negative mining [78] is widely used in training one-stage detectors, *e.g*., YOLO [79] and SSD [80]. For visual tracking, Li *et al*. [81] used a temporal sampling scheme to balance positive and negative samples to facilitate CNN training. Bertinetto *et al*. [22] balanced the loss of positive and negative examples in the score map when pre-training a fully convolutional Siamese network. The success of the MDNet [17] tracker shows that it is crucial to mining hard negative samples during training classification networks. Similar to MDNet, Chen *et al* [77] introduced clipped loss to suppress the importance of easy samples. The recent work [23] on dense object detection proposed focal loss to decrease the loss from imbalance samples. Despite the importance, current deep regression trackers[1]–[3] pay little attention to the issue of data imbalance. In this work, we propose a novel shrinkage loss to penalize easy samples which have little contribution to learning regression networks. The proposed shrinkage loss generalizes well to classification learning as well. Our shrinkage loss differs from focal loss [23] significantly in that we only penalize the loss from easy samples while maintaining the loss of hard samples unchanged, whereas focal loss decreases the loss of both easy and hard samples.

3 Proposed Algorithm

To demonstrate the effectiveness of the proposed shrinkage loss in handling data imbalance, we start by developing our tracker within the one-stage regression framework owing to the convenience in implementation. Fig. 2 shows an

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2020.3041332, IEEE Transactions on Pattern Analysis and Machine Intelligence

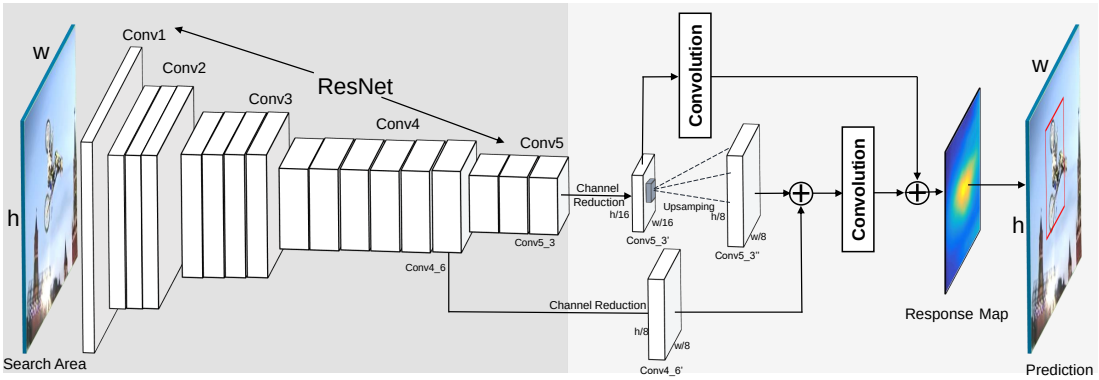


Fig. 2. Overview of the proposed deep regression network for tracking. Left: Fixed feature extractor (ResNet50 [47]). Right: Regression network trained in the first frame and updated frame-by-frame. We apply residual connections to both convolutional layers and output response maps, and use a bilinear interpolation layer for upsampling. The proposed network effectively exploits multi-level semantic abstraction across convolutional layers (§3.3). Our shrinkage loss helps to break the performance bottleneck of one-stage regression trackers caused by data imbalance and accelerates the convergence of network training (§3.2).

overview of the proposed regression network. In the following (§3.1), we first briefly revisit learning deep regression networks. We then present the proposed shrinkage loss in the context of regression learning in detail (§3.2). To facilitate regression learning with shrinkage loss, we exploit multi-level semantics across convolutional layers with residual connections in §3.3. Finally, we show the generalization ability of our shrinkage loss to classification learning based object tracking (§3.4).

3.1 Convolutional Regression

Convolutional regression networks regress a dense sampling of inputs to soft labels which are usually generated by a Gaussian function. Here, we formulate the regression network as one convolutional layer. Formally, learning the weights of the regression network involves solving the following minimization problem:

$$\arg\min_{\mathbf{W}} \|\mathbf{W} * \mathbf{X} - \mathbf{Y}\|^2 + \lambda \|\mathbf{W}\|^2, \tag{1}$$

where $*$ denotes the convolution operation and $\mathbf{W}$ denotes the kernel weight of the convolutional layer. Note that there is no bias term in Eq. 1 as we set the bias parameters to 0. $\mathbf{X}$ means the input features. $\mathbf{Y}$ is the matrix of soft labels, and each label $Y_{i,j} \in \mathbf{Y}$ ranges from 0 to 1. λ is the regularization weight. We estimate the target translation by searching for the location of the maximum value of the output response map. The size of the convolution kernel $\mathbf{W}$ is either fixed (*e.g.*, 5×5) or proportional to the size of the input features $\mathbf{X}$. Let η be the learning rate, we iteratively optimize $\mathbf{W}$ by minimizing the square loss:

$$\begin{aligned} \mathcal{L}(\mathbf{W}) &= \|\mathbf{W} * \mathbf{X} - \mathbf{Y}\|^2 + \lambda \|\mathbf{W}\|^2 \\ \mathbf{W}_t &= \mathbf{W}_{t-1} - \eta \frac{\partial L}{\partial \mathbf{W}}. \end{aligned} \tag{2}$$

3.2 Regression Tracking with Shrinkage Loss

In order to learn convolutional regression networks, the input search area has to contain a large body of background surrounding target objects (Fig. 3(a)). As the surrounding background contains valuable context information, a large area of the background helps strengthen the discriminative power of the target objects from the background. However, this increases the number of easy samples from the background as well. These easy samples produce a large loss in total to make the learning process unaware of the valuable samples close to targets. Formally, we denote the response map in every iteration by $\mathbf{P}$, which is a matrix of size $m \times n$. $P_{i,j} \in \mathbf{P}$ indicates the probability of the position $i \in [1, m], j \in [1, n]$ being the target object. Let l be the absolute difference between the estimated probability $P_{i,j}$ and its corresponding soft label y, *i.e.*, $l = |P_{i,j} - Y_{i,j}|$. Note that, when the absolute difference l is larger, the sample at the location (i, j) is more likely to be the hard sample and vice versa. Fig. 3(d) shows the histogram of the absolute differences. Note that easy samples with small absolute difference scores dominate the training data.

In terms of the absolute difference l, the square loss in regression learning can be formulated as:

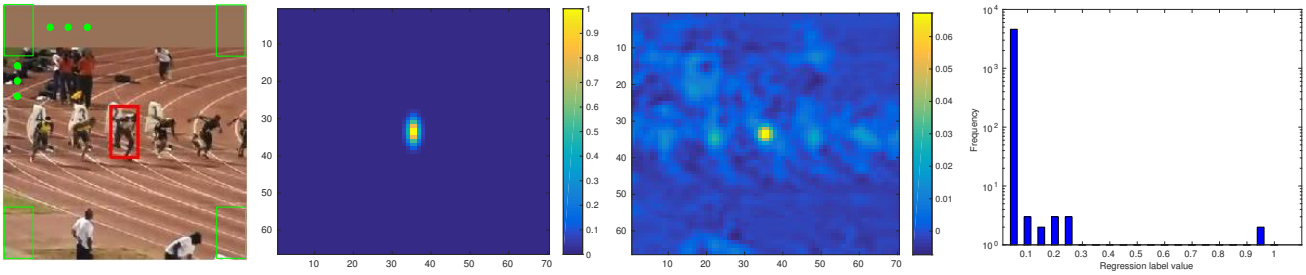
$$\mathcal{L}_2 = |P_{i,j} - Y_{i,j}|^2 = l^2. \tag{3}$$

It is worth noting that, as the output probability $P_{i,j}$ learns to regress the ground truth $Y_{i,j} \in [0, 1]$, $P_{i,j}$ almost ranges between 0 and 1 during the training process. Hence the absolute difference l almost ranges between 0 and 1 as well. The recent work on dense object detection [23] shows that adding a modulating factor to the cross entropy loss helps alleviate the data imbalance issue. The modulating factor is a function of the output probability with the goal to decrease the loss from easy samples. In regression learning, this amounts to re-weighting the square loss using an exponential form of the absolute difference term l as follows:

$$\mathcal{L}_F = l^\gamma \cdot \mathcal{L}_2 = l^{2+\gamma}. \tag{4}$$

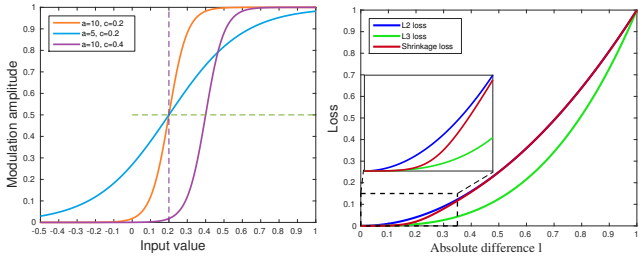
For simplicity, we set the parameter γ to 1 as we observe that the performance is not sensitive to this parameter, *i.e.*, $\mathcal{L}_F = l^3$. Note that, the weight not only penalizes easy samples (*i.e.*, $l < 0.5$) but also penalizes hard samples (*i.e.*, $l > 0.5$). By revisiting the shrinkage estimator [16] and the cost-sensitive weighting strategy [70] in learning regression networks, instead of using the absolute difference l as weight, we propose a modulating factor with respect to l

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2020.3041332, IEEE Transactions on Pattern Analysis and Machine Intelligence



(a) Input patch (b) Soft labels $\mathbf{Y}$ (c) Outputs $\mathbf{P}$ (d) Histogram of $|\mathbf{P}-\mathbf{Y}|$

Fig. 3. (a) Input patch. (b) The corresponding soft labels $\mathbf{Y}$ generated by Gaussian function for training. (c) The output regression map $\mathbf{P}$. (d) The histogram of the absolute difference $|\mathbf{P}-\mathbf{Y}|$. Note that easy samples with small absolute difference scores dominate the training data. See §3.2 for details.



(a) Modulating factor (b) Loss comparison

Fig. 4. (a) Modulating factors in Eq. 5 with different hyper-parameters. (b) Comparison between the square loss (L_2), L_3 loss and the proposed shrinkage loss for regression learning. The proposed shrinkage loss only decreases the loss from easy samples ($l < 0.5$) and keeps the loss from hard samples ($l > 0.5$) unchanged. See §3.2 for details

to re-weight the square loss to penalize easy samples only. The modulating function is with the shape of a Sigmoid-like function as:

$$f(l) = \frac{1}{1+\exp\left(a\cdot(c-l)\right)}, \tag{5}$$

where a and c are hyper-parameters controlling the shrinkage speed and the localization respectively. Fig. 4(a) shows the shapes of the modulating function with different hyper-parameters. When applying the modulating factor to weight the square loss, we obtain the proposed shrinkage loss as:

$$\mathcal{L}_S = \frac{l^2}{1+\exp\left(a\cdot(c-l)\right)}. \tag{6}$$

As shown in Fig. 4(b), the proposed shrinkage loss only penalizes the importance of easy samples ($l<0.5$) and keeps the loss of hard samples almost unchanged ($l > 0.5$) when compared to the square loss (L_2). In contrast, the L_3 loss penalizes both the easy and hard samples.

When applying the shrinkage loss to Eq. 1, we employ the cost-sensitive weighting strategy [70] and utilize the values of soft labels as an importance factor, *e.g.*, $\exp(\mathbf{Y})$, to highlight the valuable rare samples. In summary, we rewrite Eq. 1 with the shrinkage loss for learning regression networks as:

$$\mathcal{L}_S(\mathbf{W}) = \frac{\exp(\mathbf{Y})\cdot\|\mathbf{W}*\mathbf{X}-\mathbf{Y}\|^2}{1+\exp(a\cdot(c-|\mathbf{W}*\mathbf{X}-\mathbf{Y}|))} + \lambda\|\mathbf{W}\|^2. \tag{7}$$

We set the value of a to be 10 to shrink the weight function quickly and the value of c to be 0.2 to adapt to the distribution of l, which ranges from 0 to 1. Extensive comparison with the other losses shows that the proposed shrinkage loss not only improves the tracking accuracy but also accelerates the training speed (see §5.4).

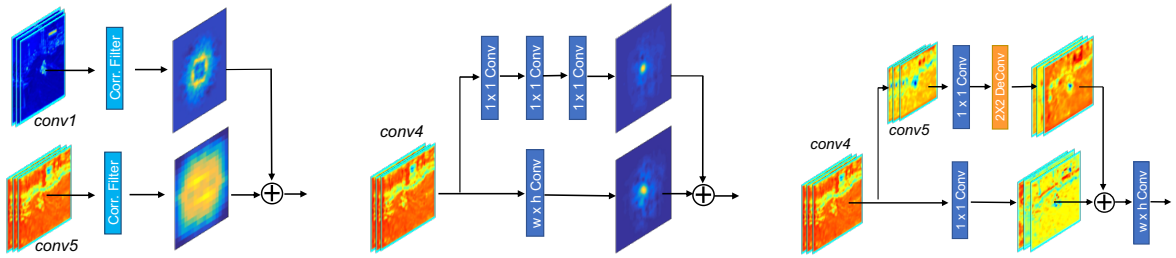
3.3 Convolutional Layer Connection

It is well known that CNN models consist of multiple convolutional layers emphasizing different levels of semantic abstraction. For visual tracking, early layers with fine-grained spatial details are helpful in precisely locating target objects; while the later layers maintain semantic abstraction that are robust to significant appearance changes. To exploit both merits, existing deep trackers [1], [8], [10] develop independent models over multiple convolutional layers and integrate the corresponding output response maps with empirical weights. When learning regression networks, we observe that semantic abstraction plays a more important role than spatial detail in dealing with appearance changes. The FCNT exploits both the *conv4* and *conv5* layers and CREST [3] merely uses the *conv4* layer. Our studies in §5.4 also suggest that regression trackers perform well when using the *conv4* and *conv5* layers as the feature backbone. To integrate the response maps generated over convolutional layers, we use a residual connection block to make full use of multiple-level semantic abstraction of target objects. In Fig. 5, we compare our scheme with the ECO [10] and CREST [3] methods. The DCFs tracker ECO [10] *independently* learns correlation filters over the *conv1* and *conv5* layers. CREST [3] learns a base and a residual regression network over the *conv4* layer. The proposed method in Fig. 5(c) fuses the *conv4* and *conv5* layers before learning the regression networks. Here we use the deconvolution operation to upsample the $conv5$ layer before connection. We reduce feature channels to ease the computational load as in [47], [82]. Our connection scheme resembles the Option C of constructing the residual network [47]. Ablation studies affirm the effectiveness of this scheme to facilitate regression learning (see §5.4).

3.4 Classification Tracking with Shrinkage Loss

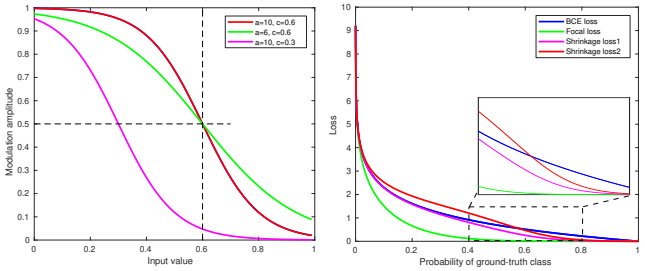
To further demonstrate the effectiveness of the proposed shrinkage loss for classification learning, we apply our shrinkage loss to the well-known Siamese tracking framework [83]. We take the representative SiameseFC [22], SiamRPN [20], SiamRPN++ [84] and MDNet [85] trackers as baseline due to their clear architectures and state-of-the-art performances.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2020.3041332, IEEE Transactions on Pattern Analysis and Machine Intelligence



(a) ECO (b) CREST (c) Ours

Fig. 5. Different schemes to fuse convolutional layers (§3.3). ECO [10] independently learns *correlation filters* over multiple convolutional layers. CREST[3] learns a base and a residual regression network over a single convolutional layer. We first fuse multiple convolutional layers using residual connection and then perform regression learning. Our regression network makes full use of multi-level semantics across multiple convolutional layers rather than merely integrating response maps as ECO and CREST.



(a) Modulating factor (b) Loss comparison

Fig. 6. (a) Modulating factors in Eq. 12 with different hyper-parameters (§3.4). (b) Comparison between the BCE loss, focal loss and the proposed shrinkage loss for Siamese network learning. The proposed shrinkage loss only decreases the loss from easy samples ($l < 0.5$) and keeps the loss from hard samples ($l > 0.5$) almost unchanged (pink) or even greater (red) for hard samples.

SiameseFC. The SiameseFC tracker formulates object tracking as a frame-by-frame matching problem. Let z' denote the template (i. e., target to be tracked) given in the first image and x' denote the search area in the subsequent frames. The matching similarity score can be computed by a cross correlation operation between two streams with the feature embedding $\phi(\cdot)$:

$$g(z', x') = \phi(z') \star \phi(x') + b, \tag{8}$$

where $\star$ means cross correlation and b means the bias. As ϕ is implemented via fully convolution networks, the output $g(z', x')$ preserves the spatial information in which each element reflects the similarity score between the target image and the search region. As a result, the position of the maximum score relative to the center of the score map reflects the displacement of the target frame-by-frame. To determine whether the target matches with the search region, SiameseFC learns a binary classifier for each element. The binary cross entropy (BCE) loss between the label $y \in [0, 1]$ and the prediction score p is formulated as:

$$\mathcal{L}_\phi(\mathbf{Y}, \mathbf{P}) = \sum_{i \in S} -\left(y_i \log(p_i) + (1 - y_i) \log(1 - p_i)\right), \tag{9}$$

where $p = \text{sigmoid}(\cdot)$ is the network final prediction.

Fig. 7 shows that there exists class imbalance between the positive and negative labels. To alleviate this issue, SiameseFC uses the class ratio as weights to balance the loss:

$$\mathcal{L}_\phi(\mathbf{Y}, \mathbf{P}) = \sum_{i \in S} -\left(y_i \beta_1 \log(p_i) + (1 - y_i) \beta_2 \log(1 - p_i)\right), \tag{10}$$

where β_1 and β_2 denote the negative and positive sample ratio in each training batch. To further handle data imbalance, we first apply the recent focal loss [23] to Eq. 10. We reformulated the weighted BCE loss (Eq. 10) in the focal loss type as follows:

$$\mathcal{L}_F(y, p) = -\alpha(1-p_t)^\gamma \log(p_t) = \begin{cases} -\beta_1(1-p)^\gamma \log(p) & y=1 \\ -\beta_2 p^\gamma \log(1-p) & y=0. \end{cases} \tag{11}$$

To avoid setting an empirical value to α for controlling the positive loss and negative as the original focal loss does, we use the values of β_1 and β_2 in Eq. 10.

Similar to Eq. 6, we apply the proposed shrinkage loss to the weighted BCE loss (Eq. 10) by multiplying the modulating function:

$$f(p_t) = \frac{1}{1 + \exp(a \cdot (p_t - c))}. \tag{12}$$

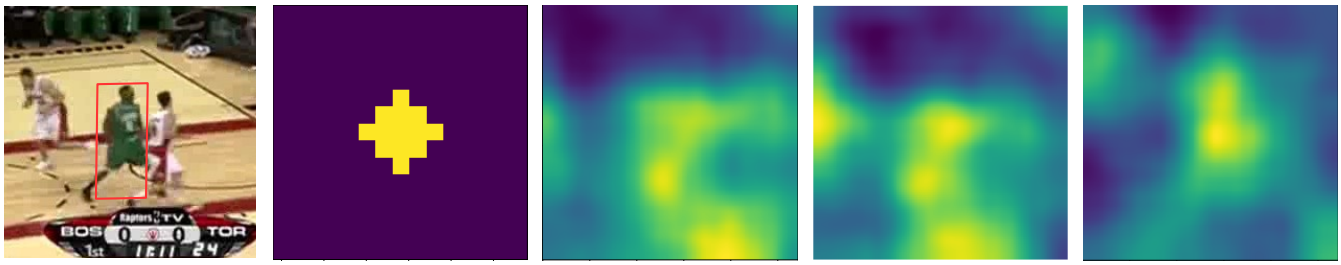
Fig. 6(d) demonstrates that the proposed shrinkage loss helps penalize the loss of easy samples while preserving the loss of hard samples. In comparison, focal loss partially penalizes the loss of hard samples. Furthermore, we take the cost-sensitive weighting strategy and apply a weight factor to Eq. 12 as:

$$f(p_t) = \frac{\exp(p_t)}{1 + \exp(a \cdot (p_t - c))}. \tag{13}$$

In this way, the loss of easy samples can be suppressed while the loss for hard samples can be enhanced. Overall, the shrinkage loss for the BCE loss can be written as:

$$\mathcal{L}_S(y, p) = \begin{cases} -\frac{\beta_1 \exp(p)}{(1+\exp(a \cdot (p-c)))} \log(p) & y = 1 \\ -\frac{\beta_2 \exp(1-p)}{(1+\exp(a \cdot ((1-p)-c)))} \log(1-p) & y = 0. \end{cases} \tag{14}$$

The values of a and c are set to 10 and 0.6, respectively. From Fig. 7, we can see that, compared to the response map of BCE loss in (c), focal loss penalizes partial contribution of hard samples as well, resulting in false large response for the distractor (d). However, equipped with the proposed shrinkage loss, SiamFC_sl can separate the targets from distractors and the background (e).



(a) Input (b) Binary labels (c) SiameseFC (d) SiamFC_fl (e) SiamFC_sl

Fig. 7. Response maps with different loss functions. (a) The region of interest. (b) Ground truth map. (c), (d) and (e) show the response maps of SiameseFC with the original loss, focal loss and the proposed shrinkage loss. See §3.4 for details.

SiamRPN. The SiamRPN [20] tracker extends SiameseFC by incorporating an additional region proposal network (RPN) from the FastRCNN detector [86]. The outputs of SiamRPN include one classification branch $[\phi(x)]_{cls}$ for scoring each proposal and one regression branch $[\phi(x)]_{reg}$ for locating each proposal:

$$\begin{aligned} A^{cls}_{w\times h\times 2k} &= [\phi(x)]_{cls} \star [\phi(z)]_{cls} \\ A^{reg}_{w\times h\times 4k} &= [\phi(x)]_{reg} \star [\phi(z)]_{reg}, \end{aligned} \tag{15}$$

where k is the number of anchors in the RPN. Similar to deep regression trackers in Fig. 3, each element on the feature map corresponds to one anchor. The only difference lies in that the bounding boxes in deep regression tracking are generated with a fixed scale and aspect ratio while the anchors in the RPN are generated with varying scales and aspect ratios. As a result, for the classification branch, there exists heavy class imbalance between positive and negative samples. We apply the proposed shrinkage loss to train the classification branch as in Eq. 14. For the regression branch, we keep the original smooth $\mathcal{L}_1$ loss. Finally, the overall loss is composed of the classification loss $\mathcal{L}_{cls}$ and the regression loss $\mathcal{L}_{reg}$ for training the whole network:

$$\mathcal{L}_{all} = \mathcal{L}_{cls} + \eta\mathcal{L}_{reg}, \tag{16}$$

where $\mathcal{L}_{cls}$ denotes the proposed shrinkage loss in Eq. 14 and $\mathcal{L}_{reg}$ denotes the smooth $\mathcal{L}_1$ loss, and η is a hyper-parameter to balance these two losses. We name the new network as SiamRPN_sl.

MDNet. The MDNet [17], [85] tracker also adopts the binary cross entropy loss (Eq. 9) to optimize the whole network:

$$\mathcal{L}_{cls} = -\frac{1}{N}\sum_{i=1}^{N}(y_i \log(p_i) + (1-y_i)\log(1-p_i)), \tag{17}$$

where N is the total number of positive and negative samples, p denotes the prediction and y_i indicates a binary label. Similar to Eq. 14, we apply our shrinkage loss to the real-time version of MDNet [85] to handle data imbalance more effectively. Formally, we formulate our shrinkage loss as:

$$\begin{aligned} \mathcal{L}_{cls} = -\frac{1}{N}\sum_{i=1}^{N}\Big(&\frac{\exp(p_i)}{1+\exp(a\cdot(p_i-c))} y_i \log(p_i) + \\ &\frac{\exp(1-p_i)}{1+\exp(a\cdot(1-p_i)-c)}(1-y_i)\log(1-p_i)\Big). \end{aligned} \tag{18}$$

4 Tracking Framework

Deep Regression Tracking. We first detail the pipeline of the proposed regression tracker. In Fig. 2, we show an overview of the proposed deep regression network, which consists of model initialization, target object localization, scale estimation and model update. For training, we crop a patch centered at the estimated location in the previous frame. We use the ResNet-50 [47] model as the backbone feature extractor. Specifically, we take the activation from the third and fourth convolutional blocks as features. We fuse features via residual connection and then feed them into the proposed regression network. During tracking, given a new frame, we crop a search patch centered at the estimated position in the last frame. The regression network takes this search patch as input and outputs a response map, where the location of the maximum value indicates the position of target objects. Once obtaining the estimated position, we carry out scale estimation using the scale pyramid strategy from [87]. To adapt to appearance variations, we incrementally update our regression network frame-by-frame. We use the tracked results and soft labels in the last T frames for model updating.

Deep Classification Tracking. We follow the original implementation of SiameseFC [22] and take the AlexNet [88] model to build the two-stream fully convolutional network as feature embedding. During offline training, we use the ImageNet Video dataset [89] to train the whole network. In each iteration, we crop one training pair from the same video. Each pair consists of an exemplar patch (*i.e.*, template) and the corresponding search region. We feed the image pair into the Siamese network and optimize the feature embedding $\phi(\cdot)$ by minimizing the proposed shrinkage loss (Eq. 14). The exemplar patch is cropped in a fixed size of 127×127 pixels and the search region is of 255×255 pixels including partial surrounding context. During the tracking process, given the cropped target object in the initial frame, we crop a search region in the current frame and feed this image pair into the SiameseFC tracker. The maximum response of the score map indicates the location of the target object. We employ a multi-scale strategy to handle scale changes. For fair comparison, we do not use online fine-tuning as in [22]. For the SiamRPN tracker, we use AlexNet [88] as the backbone feature network. Following the protocol of focal loss [23], we use all the proposals to train the network rather than selecting the proposals with an IOU score smaller than 0.3 or larger than 0.6 as in [20].

5 Experiments

In this section, we first elaborate the implementation details in §5.1. We then evaluate the proposed methods on six benchmark datasets including OTB-2013 [90], OTB-2015 [12], UAV-123 [91], VOT-2016 [13], VOT-2018 [24] and LaSOT [25] in comparison with state-of-the-art trackers in §5.2 and §5.3. Finally, in §5.4, we present extensive ablation studies on different types of losses as well as their effect on the convergence speed of network training.

5.1 Implementation Details

We implement the proposed deep regression tracker DSLT* in Matlab using the Caffe toolbox [92]. All experiments are performed on a PC with an Intel i7 4.0GHz CPU and an NVIDIA TITAN X GPU. We apply a 1×1 convolution layer to reduce the channels of $Res3d$ and $Res4f$ from 512 and 1024 to 128, respectively. We train the regression networks with the Adam [93] algorithm. Considering the large gap between the maximum values of the output regression maps over different layers, we set the learning rate η in the $Res4f$ and $Res3d$ layers to $8e$-7 and $2e$-8. During online updating, we decrease the learning rates to $2e$-7 and $5e$-9, respectively. The length of frames T for model updating is set to 7. The soft labels are generated by a two-dimensional Gaussian function with a kernel width proportional to (0.1) the target size. For scale estimation, we set the ratio of scale changes to 1.03 and the levels of scale pyramid to 3. The average tracking speed including all training process is 6.3 frames per second. We implement the Siamese tracking (*i.e.*, SiamFC_sl, SiamRPN_sl and SiamRPN++) based on the open PySOT toolkit [1] and RT-MDNet on Pytorch. All the training setting is the same as the original SiameseFC and SiamRPN trackers. Specifically, we use the ILSVRC VID [89] to train SiameseFC and RT-MDNet. For SiamRPN, we use the ILSVRC VID [89], MS-COCO [94], ILSVRC Det [89] and Youtube-bounding box [95] as training data. The average running speeds of the improved SiameseFC, SiamRPN and RT-MDNet trackers are 45.7, 80.3 and 42.7 in FPS. All the source code of this project is available at https://github.com/chaoma99/DSLT.

5.2 Overall Performance of Regression Tracking

We extensively evaluate the proposed one-stage regression tracker on five challenging tracking benchmarks. We follow the protocol of the benchmarks for fair comparison with state-of-the-art trackers. For the OTB [12], [90] datasets, we report the results of one-pass evaluation (OPE) with distance precision (DP) and overlap success (OS) plots. The legend of distance precision plots contains the thresholded scores at 20 pixels, while the legend of overlap success plots contains area-under-the-curve (AUC) scores for each tracker.

OTB Dataset. There are two versions of this dataset. The OTB-2013[90] dataset contains 50 challenging sequences and the OTB-2015[12] dataset extends the OTB-2013 dataset with additional 50 video sequences. All the sequences cover a wide range of challenges including occlusion, illumination variation, rotation, motion blur, fast motion, in-plane rotation, out-of-plane rotation, out-of-view, background clutter and low resolution. We fairly compare the proposed DSLT* method with the state-of-the-art trackers, which mainly fall into three categories: (i) one-stage regression trackers including DSLT[5], CREST[3], FCNT[1], GOTURN[96], SiameseFC [22]; (ii) one-stage DCFs trackers including ECO [10], C-COT [9], BACF [44], DeepSRDCF [6], HCFT [8], HDT [7], SRDCF [15], KCF [39], and MUSTer [97]; and (iii) two-stage trackers including SiamRPN [20], MEEM [98], TGPR [99], SINT [83], and CNN-SVM [19]. For experimental completeness, we also report the tracking performance of the proposed SiamRPN_sl. As shown in Fig. 8, the proposed DSLT* achieves the best distance precision (94.1%) and the second best overlap success (67.6%) on OTB-2013. Our method outperforms the state-of-the-art deep regression trackers (CREST [3] and FCNT [1]) by a large margin. We attribute the favorable performance of our DSLT* to two reasons. First, the proposed shrinkage loss effectively alleviates the data imbalance issue in regression learning. As a result, the proposed method can automatically mine the most discriminative samples and eliminate the distraction caused by easy samples. Second, we exploit the residual connection scheme to fuse multiple convolutional layers to further facilitate regression learning as multi-level semantics across convolutional layers are fully exploited. As well, our DSLT* performs favorably against all DCFs trackers such as C-COT, HCFT and DeepSRDCF. Note that ECO achieves the best results by exploring both deep features and hand-crafted features. On OTB-2015, our method ranks second in both distance precision and overlap success. Meanwhile, our SiamRPN_sl outperforms SiamRPN significantly in terms of precision (+2.3%) and AUC scores (+1.6%).

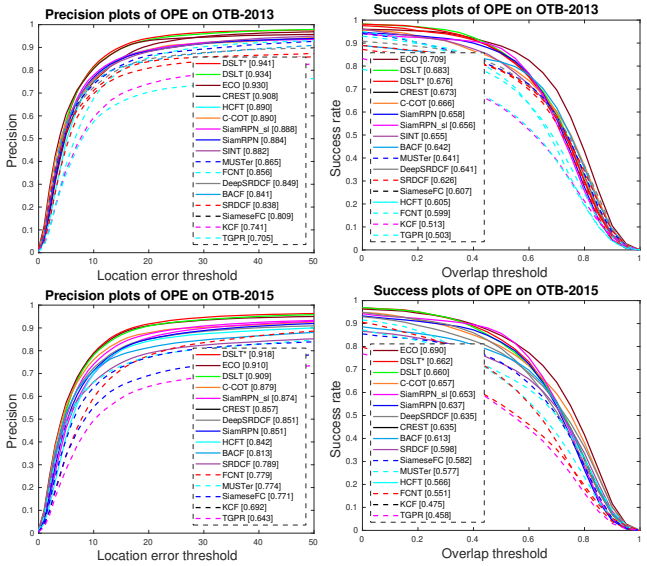


Fig. 8. Overall performance on the OTB-2013 [90] and OTB-2015 [12] datasets using one-pass evaluation (OPE). The legend of distance precision contains the threshold scores at 20 pixels while the legend of overlap success contains area-under-the-curve score for each tracker. Our tracker performs well against state-of-the-art methods.

UAV-123 Dataset. This dataset [91] contains 123 video sequences obtained by unmanned aerial vehicles (UAVs).

1. https://github.com/STVIR/pysot/

TABLE 1
Overall performance on the VOT-2016 [13] in comparison with the top 10 trackers. EAO: Expected average overlap. The three best scores are indicated in red, green and blue, respectively.

Tracker	EAO ↑	Accuracy ↑	Failure ↓
DSLT++	0.364	0.551	9.253
DSLT [5]	0.332	0.541	15.48
SiamRPN_sl	0.356	0.581	18.09
SiamRPN [20]	0.340	0.579	20.13
ECO [10]	0.374	0.546	11.67
C-COT [9]	0.331	0.539	16.58
TCNN [100]	0.325	0.554	17.93
SSAT [13]	0.321	0.577	19.27
MLDF [13]	0.311	0.490	15.04
Staple [101]	0.295	0.544	23.89
SiamRN [13]	0.277	0.547	23.99
CREST [3]	0.283	0.550	25.10
DeepSRDCF [6]	0.276	0.529	20.34
MDNet [17]	0.257	0.544	21.08
SRDCF [15]	0.247	0.544	28.31

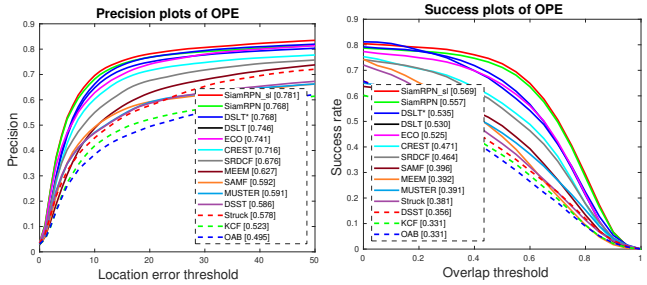


Fig. 9. Overall performance on the UAV-123 [91] dataset using one-pass evaluation (OPE). The legend of distance precision contains the threshold scores at 20 pixels while the legend of overlap success contains area-under-the-curve score for each tracker. The proposed DSLT++ method ranks first among the regression based methods.

Some tracking targets belong to the small object and undergo long term occlusion. We evaluate the proposed DSLT* with several representative methods including SiamRPN [20], ECO [10], DSLT [5], SRDCF [15], KCF [39], MUSTer[97], MEEM[98], TGPR[99], SAMF[102], DSST [83], CSK [103], Struck [66], and TLD [104]. Fig. 9 shows that the performance of the proposed DSLT* is slightly superior to ECO in terms of distance precision and overlap success rate. Furthermore, equipped with the proposed shrinkage loss, our implemented SiamRPN_sl method achieves a new state-of-the-art in location precision (78.1%) and AUC (56.9%).

VOT-2016 Dataset. The VOT-2016 [13] dataset contains 60 challenging videos, which are annotated by the following attributes: occlusion, illumination change, motion change, size change, and camera motion. The overall performance is measured by the expected average overlap (EAO), accuracy (A) and failure (F). We compare our method with state-of-the-art trackers from the VOT-2016 benchmark including SiamRPN [20], ECO [10], DSLT [5], C-COT [9], CREST [3], Staple [101], SRDCF [15], DeepSRDCF [6], MDNet [17]. Table 1 shows that our method performs better than most compared methods such as C-COT and CREST. The VOT-2016 report [13] suggests a strict state-of-the-art bound as 0.251 with the EAO metric. The proposed DSLT* achieves a much higher EAO of 0.364. Meanwhile, our implemented SiamRPN_sl method achieves a slight performance gain when comparing to original SiamRPN. According to the definition of the VOT report, all these trackers are state-of-the-art. Detailed comparison can be seen in Fig. 10.

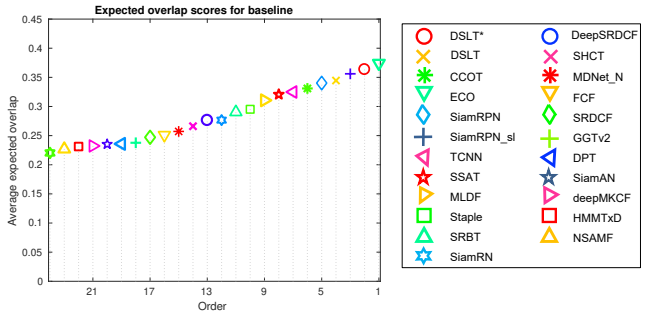


Fig. 10. Overall performance on the VOT-2016 [13] using expected average overlap graph. The proposed DSLT++ method ranks second.

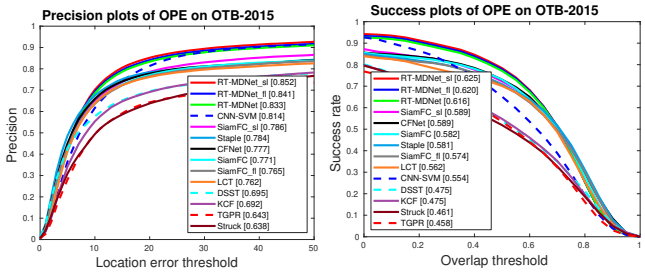


Fig. 11. Overall performance on the OTB-2015 [12] dataset using one-pass evaluation (OPE). Equipped with the proposed shrinkage loss, RT-MDNet_sl ranks first among the compared methods.

5.3 Overall Performance of Classification Tracking

To evaluate the generalization ability of the proposed shrinkage loss, we replace the original BCE loss of SiameseFC [22] and RT-MDNet [85] with focal loss (SiamFC_fl, RT-MDnet_fl) and our shrinkage loss (SiamFC_sl, RT-MDNet_sl). We fairly compare these four approaches with the representative Siamese tracking methods: SiameseFC [22], CFNet [48]; correlation filter based trackers: Staple [101], LCT [41], DSST [18], KCF [39] and other popular classification based trackers: TGPR [99], Struck [66], CNN-SVM [19]. Fig. 11 shows the overall tracking results on the OTB-2015 dataset. The RT-MDNet_sl method performs well among all the compared methods. Compared to original RT-MDNet, RT-MDNet_sl achieves large performance gains of distance precision (+1.9%) and overlap success (+0.9%). Meanwhile, we observe our SiamFC_sl surpasses SiameseFC across distance precision (+2.5%) and overlap success (+0.7%). As these methods adopt the same training and test protocols, we attribute the performance improvement solely to the proposed shrinkage loss which helps handle data imbalance during network learning. In contrast, focal loss causes SiamFC_fl to perform worse than the origin SiameseFC tracker with losses of −0.6% in distance precision and −0.8% in overlap success. The reason is that focal

loss penalizes not only easy samples but also hard samples which are crucial to learning Siamese networks.

TABLE 2
Overall performance on the VOT-2018[24] dataset in comparison with the state-of-the-art trackers. EAO: expected average overlap. Best result for each item is labeled in **bold**.

Tracker	EAO ↑	Accuracy ↑	Robustness ↓
SiamRPN++_sl	**0.440**	0.586	0.179
SiamRPN++_fl	0.419	**0.597**	0.198
SiamRPN++[84]	0.412	0.592	0.234
SiamRPN_sl	0.361	0.584	0.183
SiamRPN_fl	0.358	0.582	0.194
SiamRPN[20]	0.352	0.566	0.276
DSLTpp[105]	0.325	0.543	0.224
SA_Siam_R[106]	0.337	0.566	0.258
CPT[24]	0.339	0.506	0.239
ECO[10]	0.280	0.484	0.276
DSLT*	0.274	0.500	0.279
DSLT[5]	0.263	0.489	0.281
DeepSTRCF[107]	0.345	0.523	0.290
LSART[108]	0.323	0.495	0.258
SiamFC_sl	0.190	0.502	0.582
SiamFC_fl	0.188	0.501	0.584
SiamFC[22]	0.188	0.500	0.585
CSRDCF[109]	0.256	0.491	0.378
SRCT[24]	0.310	0.520	**0.159**
RT-MDNet_sl	0.182	0.525	0.561
RT-MDNet_fl	0.179	0.522	0.566
RT-MDNet[85]	0.178	0.522	0.567
CFTR[110]	0.300	0.505	0.184

To further demonstrate the effectiveness of the proposed shrinkage loss, we replace the original loss function of the SiamRPN++ [84], SiamRPN [20] tracker with our shrinkage loss (SiamRPN++_sl and SiamRPN_sl) and focal loss (SiamRPN++_fl and SiamRPN_fl) respectively. We mainly evaluate these two approaches on both the VOT-2018 [24] and LaSOT [25] datasets.

VOT-2018 Dataset. The VOT-2018[24] is a recent dataset for evaluating online tracking methods. It contains 60 video sequences with different challenging attributes: (1) full occlusion, (2) out-of-view motion, (3) partial occlusion, (4) camera motion, (5) fast motion, (6) scale change, (7) aspect ratio change, (8) viewpoint change, (9) similar objects. Following the evaluation protocol of VOT-2018, we adopt the Expected Average Overlap (EAO), Accuracy (A) and Robustness (R) as the criteria. Table 2 shows that SiamRPN_sl implementation significantly outperforms SiamRPN, achieving an EAO of 0.361 versus 0.352, accuracy of 0.584 versus 0.566, and robustness of 0.183 versus 0.224. Since we adopt the same network architecture, the performance gains are solely achieved by the proposed shrinkage loss, which helps penalize easy negative proposals during network training. When using focal loss, SiamRPN_fl performs better than SiamRPN, but not as well as SiamRPN_sl. This affirms that focal loss can alleviate the data imbalance issue to some extent by penalizing easy training samples, but our shrinkage loss succeeds in maintaining the loss of hard samples almost unchanged and achieves better results. Our shrinkage loss

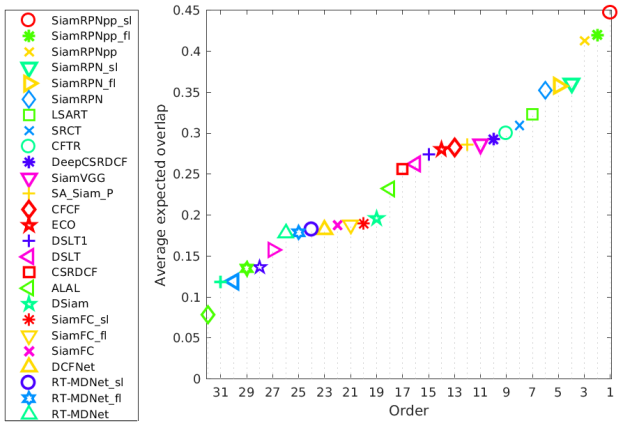


Fig. 12. Expected average overlap results on the VOT-2018[24] dataset. SiamRPNpp denotes SiamRPN++.

helps SiamRPN++ to achieve larger performance gains than focal loss does. With our shrinkage loss, SiamRPN++_sl sets a new state-of-the-art result on the VOT-2018 dataset. Fig. 12 presents the EAO rank plots of all the compared trackers as well as the overlap curves. Note that the VOT-2018 dataset is much challenging than VOT-2016, the performance of most regression based methods, such as ECO[10], CSRDCF[109] as well as our DSLT*, is inferior to the classification based methods.

LaSOT Dataset. The LaSOT [25] dataset is a recently released large-scale dataset for training and testing single object trackers. There are 1,400 videos in total containing 69 categories of objects including *airplane*, *bicycle* and *zebra*, *etc*. The average video length is about 2500 frames with challenging factors such as occlusion, deformation, out-of-view and motion blur. Following the OTB-2015[12] protocol, LaSOT uses One-Pass Evaluation (OPE) with the distance precision, and success rate as the evaluation metrics. We evaluate the proposed methods on LaSOT in comparison with 30 trackers, including SiamRPN++[84], SiamRPN[20], VITAL[111], ECO[10], DSLT* (ours), DSLT[5], MDNet[17], SiameseFC[22], CFNet[48], BACF[44], SRDCF[6], HCFT[8], *etc*. Fig. 13 illustrates the overall tracking results. With the use of shrinkage loss, SiamRPN++_sl advances the performance of the original SiamRPN++ tracker by a significant margin (+1.0% in DP and +0.9% in AUC) and achieves the new state-of-the-art results on the LaSOT dataset (49.7% and 50.3%). Equipped with focal loss, SiamRPN++_fl also achieve performance gains when compared to origin ones. However, the performance gap between SiamRPN++_fl and Siam-RPN++_sl clearly demonstrates our shrinkage loss is more effective in handling data imbalance. Benefiting from the offline training with large-scale annotated data in LaSOT, most offline based methods, such as SiamRPN++[84], SimaRPN[20] and VITAL[111], surpass methods without offline training (*e.g.*, DSLT* (ours), DSLT[5] and BACF[44]).

5.4 Ablation Studies

In this section, we first analyze the contributions of the loss function and the effectiveness of the residual connection scheme. We then discuss the convergence speed of different losses in regression learning.

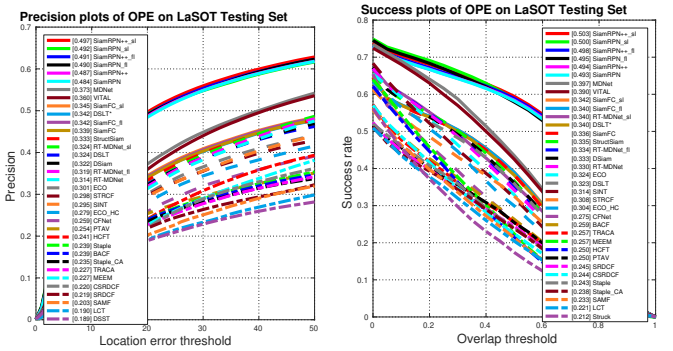


Fig. 13. Comparisons with the state-of-the-art methods on the LaSOT [25] dataset using one-pass evaluation (OPE). Equipped with the proposed shrinkage loss, the SiamRPN++_sl method ranks first.

TABLE 3
Ablation studies on the combined pool of the OTB-2015 and UAV-123 datasets under the regression and classification cases. We report distance precision (DP) and AUC scores.

Ablation Cases of Shrinkage Loss		DP(%)↑	AUC(%)↑
Regression case (DSLT*)	w/o $\exp(Y)$ in Eq. 7	84.0	59.7
	Only $\exp(Y)$ in Eq. 7	80.6	58.1
	Different values of a and c in Eq. 7		
	a=10,c=0.2	**84.3**	**59.9**
	a=6,c=0.2	84.0	59.7
	a=10,c=0.6	83.7	59.6
	a=6,c=0.6	83.1	59.0
Classification case (SiamRPN_sl)	w/o $\exp(p_t)$ in Eq. 14	82.4	60.9
	Only $\exp(p_t)$ in Eq. 14	81.1	59.8
	Different values of a and c in Eq. 14		
	a=10,c=0.6	**82.7**	**61.1**
	a=6,c=0.2	82.4	60.7
	a=10,c=0.2	82.2	60.6
	a=6,c=0.6	81.9	60.4
Ablation Cases of Focal Loss		DP(%) ↑	AUC(%) ↑
Classification case (SiamRPN_fl)	Different values of α and γ in Eq. 11		
	α=0.5,γ=0.5	81.1	59.8
	α=1,γ=0.5	81.4	59.9
	α=0.5,γ=1	81.2	59.8
	α=1,γ=1	81.7	60.2
	α=0.5,γ=2	81.8	60.4
	α=1,γ=2	**82.0**	**60.5**

Shrinkage Loss Parameters Analysis. We first offer more in-depth performance analysis for our shrinkage loss. We report the comprehensive performances with different hyper-parameters on the combined pool of the OTB-2015 [12] and UAV-123 [91] datasets. Table 3 summarizes the ablation results. We can see that removing the importance factor $\exp(Y)$ in Eq. 7 and Eq. 14 yields a slight performance drop (84.3→84.0, 59.9→59.7). While disabling the modulating function leads to large performance degradation (84.3→80.6, 59.9→58.1). We can draw the similar conclusion from the results of SiamRPN_sl. In addition, we study the impact of hyper-parameters (a and c) of our shrinkage loss. We observe that the parameter c, which controls the importance of hard samples, is more sensitive to the final performance than the parameter a, which controls the shrinkage speed. We present more hyper-parameter analysis of focal loss. We observe that the best performance can be obtained when $\alpha = 0.5, \gamma = 2$.

Fig. 14. Quantitative results on the Motorcycle-3 sequence [25]. The predictions of SiamRPN_sl, SiamRPN_fl and SiamRPN are in red, pink and yellow, respectively. SiamRPN_sl method with the proposed shrinkage loss can locate the targets more robustly than that with focal loss and BCE loss.

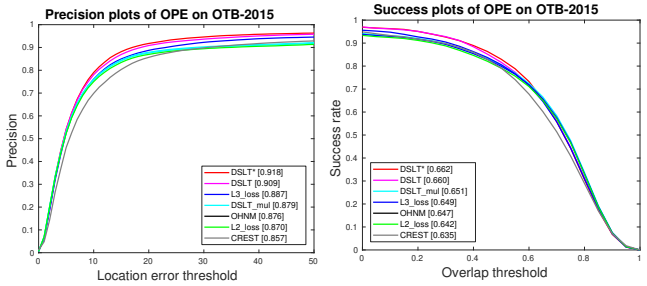


Fig. 15. Ablation studies with different losses and different layer connections on the OTB-2015 [12] dataset.

Loss Function Analysis. Next, we replace the proposed shrinkage loss with square loss ($\mathcal{L}_2$) or $\mathcal{L}_3$ loss. We evaluate the alternative implementations on the OTB-2015 [12] dataset. Overall, the proposed DSLT* method with shrinkage loss significantly advances the square loss ($\mathcal{L}_2$) and $\mathcal{L}_3$ loss by a large margin. Fig. 15 presents the quantitative results on the OTB-2015 dataset. Note that the baseline tracker with $\mathcal{L}_2$ loss performs much better than CREST [3] in both distance precision (87.0% vs. 83.8%) and overlap success (64.2% vs. 63.2%). This clearly proves the effectiveness of the convolutional layer connection scheme, which applies residual connection to both convolutional layers and output regression maps rather than only to the output regression maps as CREST does. In addition, we implement an alternative approach using online hard negative mining (OHNM) [17], [78] to completely exclude the loss from easy samples. We empirically set the mining threshold to 0.01. Our DSLT* outperforms the OHNM method significantly. Our observation is thus well aligned to [23] that easy samples still contribute to regression learning but they should not dominate the whole gradient. In addition, the OHNM method manually sets a threshold, which is hardly applicable to all videos. Moreover, we present the qualitative results with prediction scores on the challenging Motorcycle-3 sequence [25] in Fig. 14. Specifically, in the 631^{st} frame, our SiamRPN_sl yields a more accurate prediction than SiamRPN_fl and SiamRPN. In the 702^{nd} frame, we can see that both SiamRPN with binary cross entropy loss and SiamTPN_fl fail to track the target undergoing large appearance changes, whereas the proposed SiamRPN_sl can locate the targets robustly.

Feature Analysis. We report the influence of backbone feature networks: ResNet50 [47] and VGG16 [112]. With the use of ResNet50, DSLT* achieves better performance than its early version DSLT [5]. We further evaluate the effectiveness

of convolutional layers. For fair comparison, we use the VGG16 [112] as the backbone network. We first remove the connections between convolutional layers. The resulted DSLT_mul algorithm resembles the CREST [3]. Fig. 15 shows that DSLT_mul has performance drops of around 0.3% (DP) and 0.1% (OS) when compared to DSLT. This affirms the importance of fusing features before regression learning.

Convergence Speed. Following the protocol in previous works [113], [114] about loss functions, Fig. 16 compares the convergence plots and the required training iterations using different losses on the OTB-2015 [12] dataset. We also report the AUC scores on the validation set of LaSOT [25] over the fair training with our shrinkage loss, focal loss and the origin BCE loss. Overall, the training loss using the shrinkage loss descends quickly and stably. The shrinkage loss thus requires the least iterations to converge during tracking.

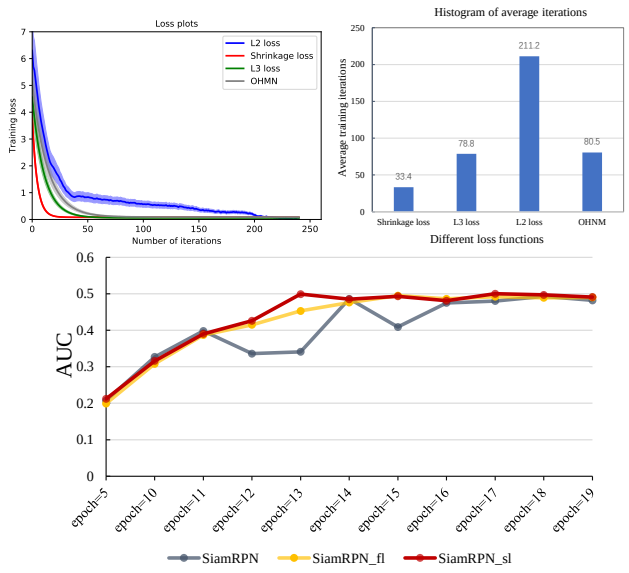


Fig. 16. Training loss plots in terms of average curves with deviations (top left), average training iterations per sequence (top right) and AUC plots (bottom).

5.5 Qualitative Evaluation

Fig. 17 visualizes the tracking results of the proposed DSLT* method on six challenging sequences in comparison with the top performing trackers including ECO [10], C-COT [9], CREST [3] and HCFT [8]. The HCFT does not perform well in most presented sequences. It is because HCFT empirically weights the response maps of multiple layers and does not incorporate a sample re-weighting strategy as in ECO and C-COT. For CREST, drift occurs on both the *Lemming* and *Bolt2* sequences. Despite a similar residual scheme to fuse multiple response maps, CREST cannot handle the background distractions well as the used square loss is unaware of data imbalance. Moreover, a single convolution layer in CREST is insufficient to capture the rich semantics. On the other hand, both the ECO and C-COT trackers use multiple convolutional layers as well as hand-crafted features. They integrate multiple correlation response maps and rely on a post-processing location refinement. Despite the overall favorable performance, ECO and C-COT do not perform well in the presence of heavy motion blur. These methods both drift in the 374^{th} frame of the *Lemming* sequence. The proposed DSLT* performs well on all these sequences. The proposed shrinkage loss can effectively suppress the target-similar samples from the background. This helps our DSLT* handle heavy occlusion (*Lemming*, *Skating1*) and background clutter (*Soccer*) well, let alone the motion blur (*Human9*) and deformation (*Bolt2*).

5.6 Discussions

Fig. 18 shows that the proposed DSLT* method does not handle large scale variations well (*e.g.*, *Bird1* and *Jump*). It is because our approach outputs axis-aligned rectangles as tracking results, which do not work well when target objects undergo severe rotation changes. In this case, the size of the target objects changes significantly in a short span of time. In the extreme scale variations scenarios (*e.g.*, *Gymnastic3*), the pre-defined axis-aligned bounding boxes in a pyramid cannot cover the potential objects well. Consequently, the regression tracker is not able to locate the target locations precisely. In future work, we plan to incorporate another branch in the regression network to automatically learn the scale change. Moreover, the proposed one-stage regression tracker runs more slowly than the Siamese trackers. This suggests our future work using a better optimization scheme to accelerate regression learning.

6 Conclusion

In this paper, we addressed the data imbalance issue for learning deep models for robust visual object tracking. We first revisited one-stage trackers based on deep regression networks and identify the performance bottleneck of one-stage regression trackers as the data imbalance issue in regression learning, which impedes one-stage regression trackers from achieving state-of-the-art results, especially when compared to DCFs trackers. To break the performance bottleneck, we proposed the novel shrinkage loss to facilitate learning regression networks with better accuracy and faster convergence speed. To further improve regression learning, we exploited multi-level semantic abstraction of target objects across multiple convolutional layers as features. We applied the residual connections to both convolutional layers and their output response maps. We succeed in narrowing the performance gap between one-stage deep regression trackers and DCFs trackers. Moreover, we showed that the proposed shrinkage loss is effective in addressing the class imbalance issue in classification learning as well. We took the Siamese trackers and RT-MDNet as baseline algorithms and investigated the class imbalance issue during the off-line learning process. We applied the proposed shrinkage loss to facilitate learning classification tracking networks. With the use of our shrinkage loss, the baseline Siamese trackers (SiameseFC and SiamRPN) and RT-MDNet both achieved large performance gains. Extensive experiments on six benchmark datasets: OTB-2013, OTB-2015, UAV-123, VOT-2016 as well as recent VOT-2018 and large-scale LaSOT demonstrated the effectiveness and efficiency of the proposed shrinkage loss in alleviating the data imbalance issue in deep object tracking.

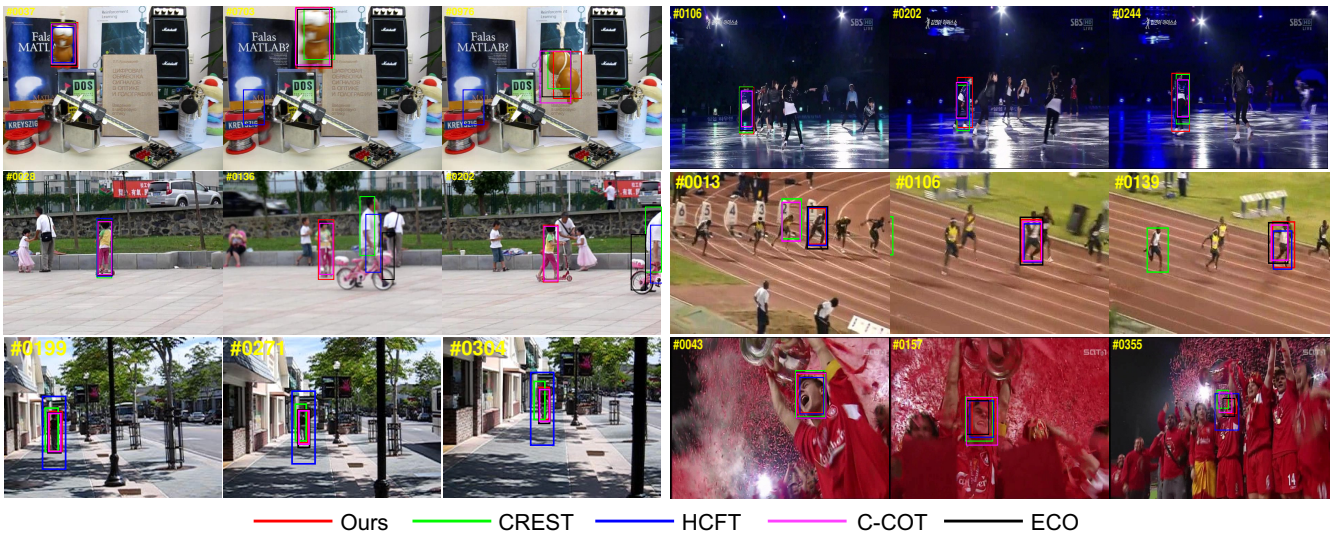


Fig. 17. Qualitative evaluation. We show tracking results of the CREST [3], HCFT [8], C-COT [9], ECO [10] and our method on six challenging video sequences (from left to right and top to bottom are *Lemming*, *Skating1*, *Girl2*, *Bolt2*, *Human9*, and *Soccer*, respectively).

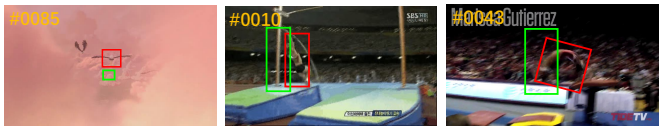


Fig. 18. Tracking failure cases on the *Bird1* (OTB-2013[90]), *Jump* (OTB-2015 [12]) and *Gymnastic3* sequences (VOT-2016 [13]). The proposed DSLT* tracker is not effective for handling significant scale variations of the target objects. Red boxes show the ground truth and green boxes are our tracking results.

ACKNOWLEDGMENTS

This work is supported in part by the National Key Research and Development Program of China (2016YFB1001003), NSFC (61906119, 61527804), STCSM(18DZ1112300), and Shanghai Pujiang Program.

REFERENCES

[1] L. Wang, W. Ouyang, X. Wang, and H. Lu, "Visual tracking with fully convolutional networks," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 3119–3127.

[2] W. Lijun, O. Wanli, W. Xiaogang, and L. Huchuan, "STCT: sequentially training convolutional networks for visual tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 1373–1381.

[3] Y. Song, C. Ma, L. Gong, J. Zhang, R. W. H. Lau, and M.-H. Yang, "Crest: Convolutional residual learning for visual tracking," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2574–2583.

[4] N. Wang, Y. Song, C. Ma, W. Zhou, W. Liu, and H. Li, "Unsupervised deep tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 1308–1317.

[5] X. Lu, C. Ma, B. Ni, X. Yang, I. Reid, and M.-H. Yang, "Deep regression tracking with shrinkage loss," in *European Conference on Computer Vision*, 2018, pp. 369–386.

[6] M. Danelljan, G. Häger, F. S. Khan, and M. Felsberg, "Convolutional features for correlation filter based visual tracking," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2015.

[7] Y. Qi, S. Zhang, L. Qin, H. Yao, Q. Huang, J. Lim, and M. Yang, "Hedged deep tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4303–4311.

[8] C. Ma, J.-B. Huang, X. Yang, and M.-H. Yang, "Hierarchical convolutional features for visual tracking," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 3074–3082.

[9] M. Danelljan, A. Robinson, F. Shahbaz Khan, and M. Felsberg, "Beyond correlation filters: Learning continuous convolution operators for visual tracking," in *European Conference on Computer Vision*, 2016, pp. 472–488.

[10] M. Danelljan, G. Bhat, F. Shahbaz Khan, and M. Felsberg, "Eco: Efficient convolution operators for tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 6931–6939.

[11] T. Zhang, C. Xu, and M.-H. Yang, "Learning multi-task correlation particle filters for visual tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 2, pp. 365–378, 2018.

[12] Y. Wu, J. Lim, and M. Yang, "Object tracking benchmark," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 9, pp. 1834–1848, 2015.

[13] M. Kristan, A. Leonardis, J. Matas, and et al., "The visual object tracking VOT2016 challenge results," in *European Conference on Computer Vision Workshops*, 2016.

[14] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 9, pp. 1263–1284, 2009.

[15] M. Danelljan, G. Häger, F. S. Khan, and M. Felsberg, "Learning spatially regularized correlation filters for visual tracking," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 4310–4318.

[16] J. B. Copas, "Regression, prediction and shrinkage," *Journal of the Royal Statistical Society*, vol. 45, 1983.

[17] H. Nam and B. Han, "Learning multi-domain convolutional neural networks for visual tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4293–4302.

[18] H. Li, Y. Li, and F. Porikli, "Deeptrack: Learning discriminative feature representations by convolutional neural networks for visual tracking," in *British Machine Vision Conference*, 2014, pp. 1420–1429.

[19] S. Hong, T. You, S. Kwak, and B. Han, "Online tracking by learning discriminative saliency map with convolutional neural network," in *Proc. ACM Int. Conf. Mach. Learn.*, 2015, pp. 597–606.

[20] B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu, "High performance visual tracking with siamese region proposal network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8971–8980.

[21] N. Wang and D. Yeung, "Learning a deep compact image representation for visual tracking," in *Proceedings of Advances in Neural Information Processing Systems*, 2013, pp. 809–817.

[22] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. S. Torr, "Fully-convolutional siamese networks for object tracking," in *European Conference on Computer Vision Workshops*, 2016.

[23] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollr, "Focal loss for dense object detection," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2999–3007.

[24] M. Kristan, A. Leonardis, J. Matas, M. Felsberg, R. Pflugfelder, L. Cehovin Zajc, T. Vojir, G. Bhat, A. Lukezic, A. Eldesokey *et al.*, "The sixth visual object tracking vot2018 challenge results," in *European Conference on Computer Vision*, 2018, pp. 3–53.

[25] H. Fan, L. Lin, F. Yang, P. Chu, G. Deng, S. Yu, H. Bai, Y. Xu, C. Liao, and H. Ling, "Lasot: A high-quality benchmark for large-scale single object tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5374–5383.

[26] S. Salti, A. Cavallaro, and L. di Stefano, "Adaptive appearance modeling for video tracking: Survey and evaluation," *IEEE Transactions on Image Process.*, vol. 21, no. 10, pp. 4334–4348, 2012.

[27] A. W. M. Smeulders, D. M. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah, "Visual tracking: An experimental survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 7, 2014.

[28] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui, "Visual object tracking using adaptive correlation filters," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2010, pp. 2544–2550.

[29] K. Zhang, L. Zhang, Q. Liu, D. Zhang, and M.-H. Yang, "Fast visual tracking via dense spatio-temporal context learning," in *European Conference on Computer Vision*, 2014, pp. 127–141.

[30] Y. Sun, C. Sun, D. Wang, Y. He, and H. Lu, "Roi pooled correlation filters for visual tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5783–5791.

[31] K. Dai, D. Wang, H. Lu, C. Sun, and J. Li, "Visual tracking via adaptive spatially-regularized correlation filters," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4670–4679.

[32] S. Liu, T. Zhang, X. Cao, and C. Xu, "Structural correlation filter for robust visual tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4312–4320.

[33] X. Dong, J. Shen, D. Yu, W. Wang, J. Liu, and H. Huang, "Occlusion-aware real-time object tracking," *IEEE Transactions on Multimedia*, vol. 19, no. 4, pp. 763–771, 2016.

[34] J. Shen, D. Yu, L. Deng, and X. Dong, "Fast online tracking with detection refinement," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 1, pp. 162–173, 2017.

[35] T. Zhang, A. Bibi, and B. Ghanem, "In defense of sparse tracking: Circulant sparse tracker," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 3880–3888.

[36] M. Danelljan, G. Bhat, F. S. Khan, and M. Felsberg, "Atom: Accurate tracking by overlap maximization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4660–4669.

[37] T. Xu, Z.-H. Feng, X.-J. Wu, and J. Kittler, "Joint group feature selection and discriminative filter learning for robust visual object tracking," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 7950–7960.

[38] N. Wang, W. Zhou, Q. Tian, R. Hong, M. Wang, and H. Li, "Multi-cue correlation filters for robust visual tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4844–4853.

[39] 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*, vol. 37, no. 3, 2015.

[40] C. Ma, X. Yang, C. Zhang, and M.-H. Yang, "Long-term correlation tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 5388–5396.

[41] C. Ma, J.-B. Huang, X. Yang, and M.-H. Yang, "Adaptive correlation filters with long-term and short-term memory for object tracking," *International Journal of Computer Vision*, pp. 1–26, 2018.

[42] M. Wang, Y. Liu, and Z. Huang, "Large margin object tracking with circulant feature maps," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4800–4808.

[43] T. Zhang, C. Xu, and M.-H. Yang, "Multi-task correlation particle filter for robust object tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4819–4827.

[44] H. Kiani Galoogahi, A. Fagg, and S. Lucey, "Learning background-aware correlation filters for visual tracking," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 1144–1152.

[45] D. Held, S. Thrun, and S. Savarese, "Learning to track at 100 fps with deep regression networks," in *European Conference on Computer Vision*, 2016, pp. 749–765.

[46] J. Gao, T. Zhang, X. Yang, and C. Xu, "P2t: Part-to-target tracking via deep regression learning," *IEEE Transactions on Image Process.*, vol. 27, no. 6, pp. 3074–3086, 2018.

[47] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[48] J. Valmadre, L. Bertinetto, J. Henriques, A. Vedaldi, and P. H. Torr, "End-to-end representation learning for correlation filter based tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2805–2813.

[49] Q. Guo, W. Feng, C. Zhou, R. Huang, L. Wan, and S. Wang, "Learning dynamic siamese network for visual object tracking," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 1763–1771.

[50] P. Li, B. Chen, W. Ouyang, D. Wang, X. Yang, and H. Lu, "Gradnet: Gradient-guided network for visual object tracking," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 6162–6171.

[51] X. Li, C. Ma, B. Wu, Z. He, and M.-H. Yang, "Target-aware deep tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 1369–1378.

[52] Y. Zhang, L. Wang, J. Qi, D. Wang, M. Feng, and H. Lu, "Structured siamese network for real-time visual tracking," in *European Conference on Computer Vision*, 2018, pp. 351–366.

[53] X. Dong and J. Shen, "Triplet loss in siamese network for object tracking," in *European Conference on Computer Vision*, 2018, pp. 459–474.

[54] X. Dong, J. Shen, W. Wang, L. Shao, H. Ling, and F. Porikli, "Dynamical hyperparameter optimization via deep reinforcement learning in tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

[55] L. Zhang, A. Gonzalez-Garcia, J. v. d. Weijer, M. Danelljan, and F. S. Khan, "Learning the model update for siamese trackers," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 4010–4019.

[56] J. Gao, T. Zhang, and C. Xu, "Graph convolutional tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4649–4659.

[57] A. He, C. Luo, X. Tian, and W. Zeng, "A twofold siamese network for real-time object tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4834–4843.

[58] X. Mei and H. Ling, "Robust visual tracking and vehicle classification via sparse representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 11, pp. 2259–2272, 2011.

[59] N. Wang, J. Shi, D.-Y. Yeung, and J. Jia, "Understanding and diagnosing visual tracking systems," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 3101–3109.

[60] H. Fan and H. Ling, "Parallel tracking and verifying: A framework for real-time and high accuracy visual tracking," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 5486–5494.

[61] Y. Hua, K. Alahari, and C. Schmid, "Online object tracking with proposal selection," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 3092–3100.

[62] G. Zhu, F. Porikli, and H. Li, "Beyond local search: Tracking objects everywhere with instance-specific proposals," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 943–951.

[63] S. Zhang, J. Huang, J. Lim, Y. Gong, J. Wang, N. Ahuja, and M. Yang, "Tracking persons-of-interest via unsupervised representation adaptation," *International Journal of Computer Vision*, vol. 128, no. 1, pp. 96–120, 2020.

[64] T. Zhang, S. Liu, N. Ahuja, M.-H. Yang, and B. Ghanem, "Robust visual tracking via consistent low-rank sparse learning," *International Journal of Computer Vision*, vol. 111, no. 2, pp. 171–190, 2015.

[65] B. Babenko, M. Yang, and S. J. Belongie, "Robust object tracking with online multiple instance learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 8, pp. 1619–1632, 2011.

[66] S. Hare, A. Saffari, and P. H. Torr, "Struck: Structured output tracking with kernels," in *Proceedings of the IEEE International Conference on Computer Vision*, 2011, pp. 263–270.

[67] J. Ning, J. Yang, S. Jiang, L. Zhang, and M. Yang, "Object tracking via dual linear structured SVM and explicit feature map," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4266–4274.

[68] H. Fan and H. Ling, "Sanet: Structure-aware network for visual tracking," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2017, pp. 42–49.

[69] B. Zadrozny, J. Langford, and N. Abe, "Cost-sensitive learning by cost-proportionate example weighting," in *ICDM*, 2003.

[70] M. Kukar and I. Kononenko, "Cost-sensitive learning with neural networks," in *ECAI*, 1998, pp. 445–449.

[71] C. Huang, Y. Li, C. C. Loy, and X. Tang, "Learning deep representation for imbalanced classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 5375–5384.

[72] Q. Dong, S. Gong, and X. Zhu, "Class rectification hard mining for imbalanced deep learning," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 1869–1878.

[73] T. Maciejewski and J. Stefanowski, "Local neighbourhood extension of SMOTE for mining imbalanced data," in *Proceedings of the IEEE Symposium on Computational Intelligence and Data Mining*, 2011, pp. 104–111.

[74] S. H. Khan, M. Hayat, M. Bennamoun, F. A. Sohel, and R. Togneri, "Cost-sensitive learning of deep feature representations from imbalanced data," *IEEE Transactions on Neural Network and Learning Systems*, vol. 29, no. 8, pp. 3573–3587, 2017.

[75] Y. Tang, Y. Zhang, N. V. Chawla, and S. Krasser, "Svms modeling for highly imbalanced classification," *IEEE Transactions on Cybernetics*, vol. 39, no. 1, 2009.

[76] K. M. Ting, "A comparative study of cost-sensitive boosting algorithms," in *Proc. ACM Int. Conf. Mach. Learn.*, 2000, pp. 435–446.

[77] K. Chen and W. Tao, "Convolutional regression for visual tracking," *IEEE Transactions on Image Process.*, vol. 27, no. 7, pp. 3611–3620, 2018.

[78] A. Shrivastava, A. Gupta, and R. Girshick, "Training region-based object detectors with online hard example mining," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 761–769.

[79] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 779–788.

[80] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *European Conference on Computer Vision*, 2016, pp. 21–37.

[81] H. Li, Y. Li, and F. M. Porikli, "Robust online visual tracking with a single convolutional neural network," in *Asian Conference on Computer Vision*, 2014, pp. 194–209.

[82] M. Danelljan, F. S. Khan, M. Felsberg, and J. van de Weijer, "Adaptive color attributes for real-time visual tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1090–1097.

[83] R. Tao, E. Gavves, and A. W. M. Smeulders, "Siamese instance search for tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 1420–1429.

[84] B. Li, W. Wu, Q. Wang, F. Zhang, J. Xing, and J. Yan, "Siamrpn++: Evolution of siamese visual tracking with very deep networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4282–4291.

[85] I. Jung, J. Son, M. Baek, and B. Han, "Real-time mdnet," in *European Conference on Computer Vision*, 2018, pp. 83–98.

[86] S. Ren, K. He, R. B. Girshick, and J. Sun, "Faster R-CNN: towards real-time object detection with region proposal networks," in *Proceedings of Advances in Neural Information Processing Systems*, 2015, pp. 91–99.

[87] M. Danelljan, G. Häger, F. S. Khan, and M. Felsberg, "Accurate scale estimation for robust visual tracking," in *British Machine Vision Conference*, 2014.

[88] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proceedings of Advances in Neural Information Processing Systems*, 2012, pp. 1097–1105.

[89] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[90] Y. Wu, J. Lim, and M.-H. Yang, "Online object tracking: A benchmark," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 2411–2418.

[91] M. Mueller, N. Smith, and B. Ghanem, "A benchmark and simulator for UAV tracking," in *European Conference on Computer Vision*, 2016, pp. 445–461.

[92] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *ACM International conference on Multimedia*, 2014, pp. 675–678.

[93] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proceedings of International Conference on Learning Representations*, 2015, pp. 749–765.

[94] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European Conference on Computer Vision*, 2014, pp. 740–755.

[95] E. Real, J. Shlens, S. Mazzocchi, X. Pan, and V. Vanhoucke, "Youtube-boundingboxes: A large high-precision human-annotated data set for object detection in video," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 7464–7473.

[96] D. Held, S. Thrun, and S. Savarese, "Learning to track at 100 FPS with deep regression networks," in *European Conference on Computer Vision*, 2016, pp. 749–765.

[97] Z. Hong, Z. Chen, C. Wang, X. Mei, D. V. Prokhorov, and D. Tao, "Multi-store tracker (muster): A cognitive psychology inspired approach to object tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 749–758.

[98] J. Zhang, S. Ma, and S. Sclaroff, "MEEM: robust tracking via multiple experts using entropy minimization," in *European Conference on Computer Vision*, 2014, pp. 188–203.

[99] J. Gao, H. Ling, W. Hu, and J. Xing, "Transfer learning based visual tracking with gaussian processes regression," in *European Conference on Computer Vision*, 2014, pp. 188–203.

[100] H. Nam, M. Baek, and B. Han, "Modeling and propagating cnns in a tree structure for visual tracking," *arXiv preprint arXiv:1608.07242*, 2016.

[101] L. Bertinetto, J. Valmadre, S. Golodetz, O. Miksik, and P. H. S. Torr, "Staple: Complementary learners for real-time tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 1401–1409.

[102] Y. Li and J. Zhu, "A scale adaptive kernel correlation filter tracker with feature integration," in *European Conference on Computer Vision Workshops*, 2014, pp. 254–265.

[103] J. F. Henriques, R. Caseiro, P. Martins, and J. P. Batista, "Exploiting the circulant structure of tracking-by-detection with kernels," in *European Conference on Computer Vision*, 2012, pp. 702–715.

[104] Z. Kalal, K. Mikolajczyk, and J. Matas, "Tracking-learning-detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 7, pp. 1409–1422, 2012.

[105] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool, "Temporal segment networks: Towards good practices for deep action recognition," in *European Conference on Computer Vision*, 2016, pp. 20–36.

[106] A. He, C. Luo, X. Tian, and W. Zeng, "Towards a better match in siamese network based visual object tracker," in *European Conference on Computer Vision Workshops*, 2018, pp. 132–147.

[107] F. Li, C. Tian, W. Zuo, L. Zhang, and M. H. Yang, "Learning spatial-temporal regularized correlation filters for visual tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4904–4913.

[108] C. Sun, D. Wang, H. Lu, and M.-H. Yang, "Learning spatial-aware regressions for visual tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8962–8970.

[109] A. Lukezic, T. Vojír, L. C. Zajc, J. Matas, and M. Kristan, "Discriminative correlation filter tracker with channel and spatial reliability," *International Journal of Computer Vision*, vol. 126, no. 7, pp. 671–688, 2018.

[110] G. Bhat, J. Johnander, M. Danelljan, F. Shahbaz Khan, and M. Felsberg, "Unveiling the power of deep tracking," in *European Conference on Computer Vision*, 2018, pp. 493–509.

[111] Y. Song, C. Ma, X. Wu, L. Gong, L. Bao, W. Zuo, C. Shen, L. Rynson, and M.-H. Yang, "Vital: Visual tracking via adversarial learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8990–8999.
[112] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proceedings of International Conference on Learning Representations*, 2015, pp. 744–752.
[113] M. Berman, A. Rannen Triki, and M. B. Blaschko, "The lovász-softmax loss: a tractable surrogate for the optimization of the intersection-over-union measure in neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4413–4421.
[114] W. Liu, Y. Wen, Z. Yu, and M. Yang, "Large-margin softmax loss for convolutional neural networks." in *Proc. ACM Int. Conf. Mach. Learn.*, vol. 2, no. 3, 2016, p. 7.

Xiankai Lu is a Tenure-track Professor in the School of Software, Shandong University. From 2018 to 2020, he was a research associate with Inception Institute of Artificial Intelligence, Abu Dhabi, United Arab Emirates. He received the Ph.D. degree from Shanghai Jiao Tong University in 2018. His research interests include computer vision, object tracking, video object segmentation and deep learning.

Chao Ma is a tenure-track assistant professor in the AI Institute, Shanghai Jiao Tong University, Shanghai, 200240, P. R. China. His research interests include computer vision and machine learning. He was a research associate with School of Computer Science at The University of Adelaide. He received the Ph.D. degree from Shanghai Jiao Tong University in 2016. He was sponsored by China Scholarship Council as a visiting Ph.D. student in University of California at Merced from the fall of 2013 to the fall of 2015. He is a member of the IEEE.

Jianbing Shen (M'11-SM'12) is currently acting as the Lead Scientist with the Inception Institute of Artificial Intelligence, Abu Dhabi, United Arab Emirates. He is also a Full Professor with the School of Computer Science, Beijing Institute of Technology. He published more than 100 top journal and conference papers, and his Google scholar citations are more than 9500 times with H-index 48, where eighteen papers are selected as the ESI Hightly Cited Papers and ESI Hot Papers. He was rewarded as the receptions of Highly Cited Researcher by the Web of Science in 2020. His research interests include computer vision, deep learning, autonomous driving, medical image analysis and intelligent systems. He is an Associate Editor of *IEEE TIP*, *IEEE TNNLS* and other journals.

Xiaokang Yang received the B.S. degree from Xiamen University, Xiamen, China, in 1994; the M.S. degree from Chinese Academy of Sciences, Shanghai, China, in 1997; and the Ph.D. degree from Shanghai Jiao Tong University, Shanghai, in 2000. He was a Research Fellow with the Center for Signal Processing, Nanyang Technological University, Singapore, from 2000 to 2002 and a Research Scientist with the Institute for Infocomm Research, Singapore, from 2002 to 2004. He is currently a full professor and the deputy director of the AI Institute, Shanghai Jiao Tong University. He is a fellow of the IEE.

Ian Reid is a professor of computer science at the University of Adelaide. He joined the School in September 2012. He is part of the Australian Centre for Visual Technologies a University research centre within the School. He was formerly a professor of engineering science at the University of Oxford where he rans the Active Vision Group which is part of the wider Robotics Research Group. His research interests span a wide range of topics in Computer Vision. He is concerned with algorithms for visual control of active head/eye robotic platforms (for surveillance and navigation), visual geometry and camera self-calibration (applications of these to measurement, AR and VR, including sporting events), visual SLAM, human motion capture, activity analysis, and novel view synthesis.

Ming-Hsuan Yang is a professor in Electrical Engineering and Computer Science at the University of California, Merced. He received the PhD degree in computer science from the University of Illinois at Urbana-Champaign in 2000. Prior to joining UC Merced in 2008, he was a senior research scientist at the Honda Research Institute working on vision problems related to humanoid robots. He coauthored the book Face Detection and Gesture Recognition for Human-Computer Interaction (Kluwer Academic 2001) and edited special issue on face recognition for Computer Vision and Image Understanding in 2003, and a special issue on real-world face recognition for IEEE Transactions on Pattern Analysis and Machine Intelligence. Yang served as an associate editor of the IEEE Transactions on Pattern Analysis and Machine Intelligence from 2007 to 2011 and is an associate editor of the International Journal of Computer Vision, Computer Vision and Image Understanding, Image and Vision Computing and Journal of Artificial Intelligence Research. He received the NSF CAREER award in 2012, and Google Faculty Award in 2009. He is a fellow of the IEEE, and a senior member of the ACM.