Human-centric Computing and Information Sciences

February 2023 | Volume 13

www.hcisjournal.com

KIPS Korea Information Processing Society

KIPS CSWRG Korea Information Processing Society Computer Software Research Group

HCIS | Human-centric Computing and Information Sciences

RESEARCH (Research Manuscript) **Open Access**

Human-centric Computing and Information Sciences (2023) 13:07
DOI: https://doi.org/10.22967/HCIS.2023.13.007
Received: April 7, 2022; Accepted: June 5, 2022; Published: February 15, 2023

Fast Visual Tracking with Squeeze and Excitation Region Proposal Network

Dun Cao[1], Renhua Dai[1], Jin Wang[1], Baofeng Ji[2], Osama Alfarraj[3], Amr Tolba[3], Pradip Kumar Sharma[4], and Min Zhu[5,*]

Abstract

Siamese trackers have achieved significant progress over the past few years. However, the existing methods are either high speed or high performance, and it is difficult for previous Siamese trackers to balance both. In this work, we propose a high-performance yet effective tracker (SiamSERPN), which utilizes MobileNetV2 as the backbone and equips with the proposed squeeze and excitation region proposal network (SERPN). For the SERPN block, we introduce the distance-IoU (DIoU) into the classification and regression branches to remedy the weakness of traditional RPN. Benefiting from the structure of MobileNetV2, we propose a feature aggregation architecture of multi-SERPN blocks to improve performance further. Extensive experiments and comparisons on visual tracking benchmarks, including VOT2016, VOT2018, and GOT-10k, demonstrate that our SiamSERPN can balance speed and performance. Especially on GOT-10k benchmark, our tracker scores 0.604 while running at 75 frames per second (FPS), which is nearly 27 times that of the state-of-the-art tracker.

Keywords

Object Tracking, Siamese Network, MobileNet-V2, SERPN, Distance-IoU

1. Introduction

Visual object tracking is one of the most fundamental yet challenging topics in computer vision [1], and it has come into a wide range of applications [2–7]. Over the past few years, due to the neural network structure from shallow to deep, the Siamese network-based trackers that utilize the neural network as the backbone have achieved significant progress. But meanwhile, the Siamese tracking models are becoming increasingly heavy, which severely slows down the speed of the Siamese trackers and even below the minimum speed of real-time—25 frames per second (FPS), minimum rate for computer vison and industrial applications. For instance, the latest SiamRPN++ [8] and SiamRCNN [9] trackers, respectively, only run at 35 FPS and 2 FPS to achieve state-of-the-art performance, being much slower than the early SiamFC [10] and SiamRPN [11] method that adopts shallow networks as the backbone, as visualized in

***Corresponding Author:** Min Zhu (zhumin@zjsru.edu.cn)
[1]School of Computer and Communication Engineering, Changsha University of Science and Technology, Changsha, China
[2]College of Information Engineering, Henan University of Science and Technology, Luoyang 471000, China
[3]Department of Computer Science, Community College, King Saud University, Riyadh, Saudi Arabia
[4]Department of Computing Science, University of Aberdeen, Aberdeen, UK
[5]School of Information, Zhejiang Shuren University, Zhejiang, China

Fig. 1. Therefore, how to keep the balance between performance and speed is one of the main challenges for the Siamese trackers.

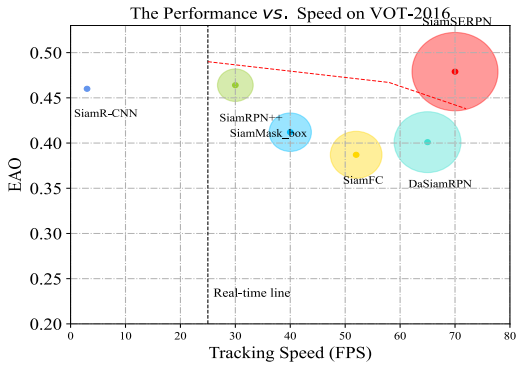


Fig. 1. Comparisons with the mainstream trackers in terms of except average overlap (EAO) performance, speed in the VOT2016 benchmark. The circle diameter is in proportion to the combination of EAO value and Speed value, and the upper right corner the better. Best viewed in color.

However, existing compression techniques such as model pruning and quantization can reduce trackers' complexity, while they inevitably bring non-negligible performance degradation due to information loss.

Different from traditional solutions, this paper proposes a simple yet effective visual tracking framework named Siamese-SERPN (SiamSERPN) to alleviate the problem of the incompatibility of speed and performance. Specifically, the SiamSERPN consists of a Siamese subnetwork for feature extraction and multiple SERPN blocks for region proposals. For the former, we utilize two identical lightweight deep networks MobileNetV2 [12], as the feature extraction backbone. Because the inverted residual block and linear bottlenecks block of MobileNetV2 not only drives it to be as efficient as the shallow network but also makes it can capture low-level information as the deep networks. Nevertheless, since the structure of lightweight networks is simpler than that of the very deep networks, performance degradation is still inevitable.

Inspired by this observation, to compensate for the performance loss caused by the lightweight network, we propose the SERPN block, which consists of a standard RPN and a squeeze excitation network [13], where the standard RPN includes the classification and regression branches. In the classification branch, we introduce the distance intersection-over-union (DIoU) [14] metric to distinguish the object from the background, and in the regression branch, we adopt the DIoU metric as the loss function for bounding box regression. The introduction of DIoU can tackle two nature disadvantages of traditional RPN at once. One is that the IoU metric used in the classification branch cannot focus on the distance between the predicted bounding box and the ground-truth box, leading the IoU metric to handle the relationship between two boxes in complex scenarios insufficiently, and the other is that the smooth L_n-norm adopted in the regression branch treats the coordinates of the predicted bounding box as independent variables for optimization, which goes against the intuition that those variables are correlated and should be regressed jointly. Additionally, benefiting from the MobileNetV2 architecture, we propose a layer-wise feature aggravation structure for the cross-correlation operation, which assists the tracker predicts the similarity map from features at multiple levels and further compensates for the performance loss.

Using the proposed SiamSERPN, we conduct extensive experiments on three large tracking benchmarks, in which it scores 0.479 and 0.401 on the VOT2016 [15] and VOT2018 [16] benchmarks. Especially the results on the VOT2016 benchmark, the proposed SiamSERPN outperforms state-of-the-art SiamRCNN by 4% and runs at nearly 70 FPS. Meanwhile, on the GOT-10k [17] benchmark, our tracker ranks second, yet runs at 75 FPS. The comprehensive experiments confirm that the proposed tracker is effective and efficient.

To summarize, this work makes the following contributions:

- We propose a fast Siamese tracker using a lightweight backbone that maintains competitive performance while running at 75 FPS.
- We design the SERPN block and propose the SERPN aggregation structure to compensate for the performance loss caused by the lightweight backbone.
- We introduce the DIoU metric into the classification branch and regression branch of the proposed SERPN to remedy the natural deficiencies of the standard RPN.

The structure of this paper is expressed as follows. Section 2 briefly presents the work related to this paper. Section 3 introduces the framework of SiamSERPN. Section 4 conducts comprehensive experiments on our proposed method with the latest trackers on three large benchmarks. Section 5 gives the conclusions and prospects of this paper.

2. Related Work

Visual tracking is one of the most active research topics in computer vision in recent decades. Many excellent methods have emerged [18–23], from correlation filter-based trackers to deep learning-based trackers. A comprehensive survey of the trackers is beyond the scope of this paper, so we only briefly review three aspects that are most relevant to our work: Siamese network-based visual trackers, deep architecture, and RPN in detection, in which the Siamese visual tracker is our major direction. Therefore, to clearly review the related work, the contributions of the mainstream anchor-based Siamese trackers are listed in Table 1 [8–11, 24, 25].

Table 1. Related studies of the mainstream anchor-based Siamese trackers

Study	Proposed methods	Main contributions	Performance (on VOT2016)	Speed (FPS)
Bertinetto et al. [10]	SiamFC	First Siamese tracker used shallow network	0.387	86
Li et al. [11]	SiamRPN	Introduce RPN into Siamese trackers	0.393	160
Zhu et al. [24]	DaSiamRPN	Expand more dataset to SiamRPN	0.411	160
Li et al. [8]	SiamRPN++	Introduce very deep network into Siamese tracker	0.464	35
Wang et al. [25]	SiamMask	Add Semantic segmentation	0.412	77
Voigtlaender et al. [9]	SiamRCNN	Proposed re-detection	0.460	4.7

2.1 Siamese Network-Based Visual Trackers

Recently, the Siamese network-based trackers attracted great attention from the visual tracking community due to their end-to-end training capabilities and high efficiency [10, 26]. SiamFC [10] adopts the Siamese network as a feature extractor and introduces the correlation layer to combine feature maps for the first time. Owing to its light structure and no need to model updates, SiamFC runs efficiently at 86 FPS. To get a more accurate object bounding box, SiamRPN [11] introduces the RPN [27] into the SiamFC, which improves performance but struggles to deal with distractors when facing similar appearances to the object. DaSiamRPN [24] tries to expand more datasets and improve the tracker's generalization by enhancing positive samples from large-scale datasets. Up to now, the framework has been modified a lot from SiamFC, and although they still are fast trackers, the performance cannot move on with a deeper network by using AlexNet [28] as the backbone. Aiming to address this problem, SiamRPN++ [8] successfully trains a tracker that uses the very deep network ResNet [29] as the backbone by randomly shifting the object location in the search region during model training to eliminate the center bias. Following SiamRPN++, SiamMask [25], and SiamRCNN [9] adopt ResNet as the backbone for feature extraction. Benefiting from very deep networks, though these anchor-based trackers can achieve state-of-the-art performance, their speed is severely degraded. A series of anchor-free trackers, such as

SiamCAR [30], SiamFC++ [31], and SiamBAN [32], are designed to solve this problem, but the performance is naturally weaker than anchor-based ones.

2.2 Deep Architecture

With the proposal of modern convolutional architecture AlexNet [28] in 2012, the research of networks is developing rapidly, and numerous sophisticated architectures are proposed, such as VGG [33], ResNet [29], and GhostNet [34], to name a few. These deep networks push the latest research on many computer vision tasks forward. But meanwhile, their excessively deep structure leads to these tasks becoming heavy and expensive. Inspired by the observation, MobileNet [12, 35, 36] proposes lightweight networks to alleviate the conflict between performance and efficiency. Following MobileNet's idea, we believe that the Siamese network-based trackers can also find a balance between performance and efficiency to address the current phenomenon that these trackers prefer to sacrifice speed for performance.

2.3 RPN in Detection

Before RPN, traditional methods are time-consuming and insufficiently quality in object detection. Fast R-CNN [27] first proposed RPN to extract more precise proposals using the supervision of both foreground-background classification and bounding box regression. Many variants based on Fast R-CNN have emerged since then. R-FCN [37] takes the component’s position information into account, and FPN [38] employs a feature pyramid network to improve the performance of tiny object detection. In contrast to two-stage detectors, the improved versions of RPN, such as DetectoRS [39] and YOLOv4 [40] are efficient detectors. However, the RPN in visual tracking is different from detection. Simply introducing RPN into visual tracking may lead to its struggle to process complex scenes, which not only increases the model's weight but also seriously degrades its robustness.

The speed of RPN-based Siamese network trackers gradually slows down with the performance improvement after utilizing the deep network as the backbone, which the previous researchers usually ignore. Some trackers run at nearly 30 FPS, and some even below 25 FPS, making it difficult to apply theoretical methods to industrial applications. The proposed method utilizes the lightweight yet deep network as the backbone to speed up, and inspired by RPN, SERPN is designed to compensate for the performance loss caused by the lightweight network. The proposed method will balance the two aspects of performance and speed by utilizing a light yet deep network, which is expected to narrow the gap between academic theory and industrial applications.

3. SiamSERPN Framework

In this section, we illustrate the proposed SiamSERPN framework. As shown in Fig. 2, the proposed SiamSERPN consists of a Siamese network backbone and multiple SERPN blocks. The Siamese network backbone is responsible for computing the convolutional feature maps of the template patch and the search patch, which uses a lightweight convolutional network. The SERPN block includes a classification branch and a regression branch. Specifically, the classification branch performs foreground-background classification on each point of the correlation layer, and the regression branch performs bounding box regression on the corresponding position.

3.1 MobileNet-driven Siamese Tracking

Modern deep neural networks [29] have proven to be effective as the feature extraction backbone in Siamese network-based trackers [8], which has led to an increasing number of trackers [9, 25] using the deep network as the backbones. Although the performance of trackers has improved, the resulting

inefficiencies have been neglected. In our work, we utilize MobileNetV2 [12] as the backbone network, whose parameters are listed in Table 2.

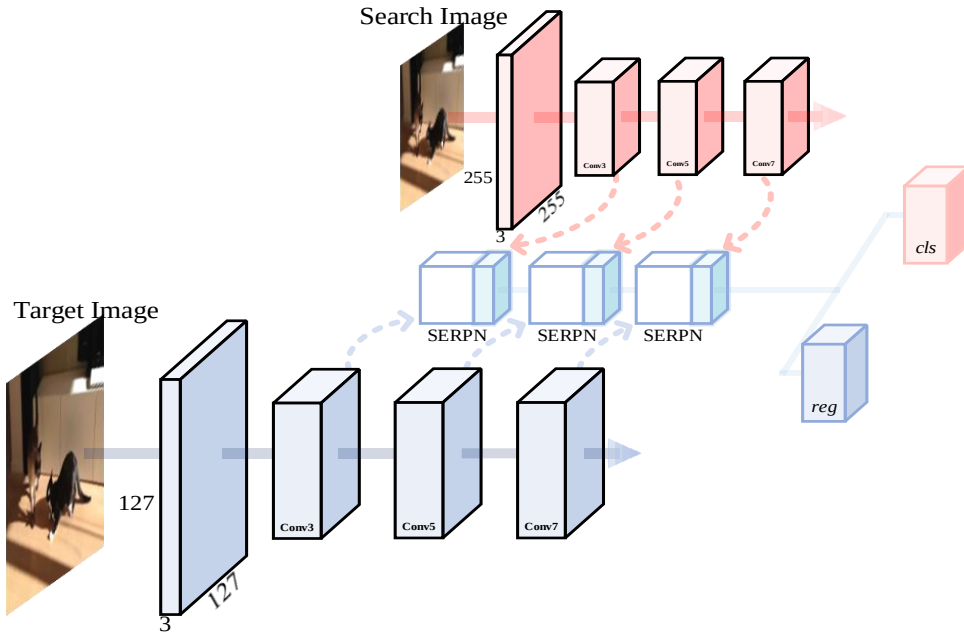


Fig. 2. Main diagram of our proposed framework. Given the target template image and search image, the proposed model outputs a dense prediction by fusing the outputs feature from multi SERPN blocks equipped on *conv3*, *conv5*, and *conv7*. *cls* denotes the classification branch, and reg represents the regression branch.

However, the original MobileNetV2 cannot be adopted directly for dense Siamese network prediction because it has a large stride of 32 pixels, which impacts the accuracy of the location of visual tracking. Therefore, we reduce the effective stride of *conv5* and *conv7* blocks to 8 pixels in order to have the unit spatial stride and increase their receptive field with dilated convolution [41], which leads to the captured information being naturally different. An extra 1×1 convolutional layer is appended to the output of each block to reduce the channel to 256. Besides, since the original purpose of MobileNetV2 is for image classification, we remove the last fully connected layer to accommodate visual tracking.

As the feature extractor, the backbone of the Siamese network consists of two identical branches with shared parameters, one of which is called the search branch and receives the search patch of size 255×255×3 (denoted as x). The other is called the template branch and receives the target template patch of size 127×127×3 (denoted as z). These two inputs are fed into the MobileNetV2 to generate the output feature maps. Then, the similarity response of the two different output feature maps, $\varphi(x)$ and $\varphi(z)$, is calculated by cross-correlation operation:

$$M(x,z) = \varphi(x) * \varphi(z) \tag{1}$$

where * denotes the cross-correlation operation.

In addition, to reduce the heavy computational burden, we crop the input images of the template branch, keeping the central region of 7×7 as the template feature and the average RGB values to fill the rest, which still captures the entire target region [10].

Table 2. Siamese network tracker utilizes two identical convolutional neural networks as the backbone network, so the Siamese network tracker is inherently heavier and more expensive than other computer

vision tasks (limited by objective experimental conditions, we give part of the parameters of our backbone network in the table)

Layer (type)	Output shape	Parameters	Connected to
Input_1 (input layer)	(None, 255, 255, 3)	0	
Conv1_pad (ZeroPadding2D)	(None, 255, 255, 3)	0	Input_1[0][0]
Conv1 (Conv2D)	(None, 112, 112, 32)	864	Conv1_pad [0][0]
Bn_Conv1(BatchNormalization)	(None, 112, 112, 32)	128	Conv1[0][0]
Conv1_relu (Relu)	(None, 112, 112, 32)	0	Bn_Conv1[0][0]
Expanded_conv_depthwise_Depthw	(None, 112, 112, 32)	288	Conv1_Relu [0][0]
Expanded_conv_depthwise_BN	(None, 112, 112, 32)	128	Expanded_conv_depthwise [0][0]
Expanded_conv_depthwise_Relu	(None, 112, 112, 32)	0	Expanded_conv_depthwise_BN [0][0]
Expanded_conv_project	(None, 112, 112, 16)	512	Expanded_conv_depthwise_Relu [0][0]
Expanded_conv_depthwise_BN	(None, 112, 112, 16)	64	Expanded_conv_project [0][0]
Block_1_expand (Conv2D)	(None, 112, 112, 96)	1536	Expanded_conv_depthwise_BN [0][0]
Block_1_expand_BN (BatchNormalization)	(None, 112, 112, 96)	384	Block_1_expand [0][0]
Block_1_expand_relu (Relu)	(None, 112, 112, 96)	0	Block_1_expand_BN [0][0]
Block_1_pad (ZeroPadding2D)	(None, 113, 113, 96)	0	Block_1_expand_relu [0][0]
Block_1_depthwise	(None, 56, 56, 96)	864	Block_1_pad [0][0]
Block_1_depthwise_BN (BatchNormalization)	(None, 56, 56, 96)	384	Block_1_depthwise [0][0]
Block_1_depthwise_relu (Relu)	(None, 56, 56, 96)	0	Block_1_depthwise_BN [0][0]
Block_1_project (Conv2D)	(None, 56, 56, 24)	2304	Block_1_depthwise_relu [0][0]
Block_1_project_BN	(None, 56, 56, 24)	96	Block_1_project [0][0]
Block_2_expand (Conv2D)	(None, 56, 56, 144)	3456	Block_1_project_BN [0][0]
⋮	⋮	⋮	⋮
Block_16_project_BN (BatchNormalization)	(None, 7, 7, 320)	1280	Block_16_project [0][0]

Total parameters (2,257,984), Trainable parameters (2,223,872), Non-trainable parameters (34,112), and Parameters in Siamese backbone network (4,515,968).

3.2 Region Proposal Network with Squeeze and Excitation

As shown in Fig. 3, the SERPN block consists of the pair-wise correlation, supervision, and squeeze excitation sections. In particular, the supervision section includes the classification and regression branches, in which the former is responsible for the foreground-background classification, and the latter is used for proposal regression. If there are k anchors, the block needs to output $2k$ channels for classification and $4k$ channels for regression. The rough features from the Siamese backbone are fed into the SERPN, where those from the search branch are denoted as $\varphi(x)$ and those from the template branch are denoted as $\varphi(z)$. Once into SERPN, $\varphi(x)$ is first split into two branches ($\varphi(x)_{cls}$ and $\varphi(x)_{reg}$) while keeping the channels unchanged, and then the pair-wise correlation section also increase the channels of $\varphi(z)$ to two branches $\varphi(z)_{cls}$ and $\varphi(z)_{reg}$ which have $2k$ and $4k$ times in channel respectively by two convolutional layers. In both two branches, $\varphi(z)$ is served as the convolutional kernel of $\varphi(x)$ and performs the convolutional operation with $\varphi(x)$, which means the channel number in $\varphi(z)$ is the same as the overall channel number of $\varphi(x)$. The computation process can be defined as follows in both classification and regression branches:

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

where the template features $\varphi(z)_{cls}$ and $\varphi(z)_{reg}$ are used as kernels, $\star$ represents the convolution operation. For convenience, $S^{cls}_{w\times h\times 2k}$ and $S^{reg}_{w\times h\times 4k}$ are uniformly denoted by S.

The squeeze excitation section includes the squeeze submodule and the excitation submodule, detailed in the right side of Fig. 3. Different from the traditional SENet [13], we remove the beginning transformation to reduce the computational complexity and let the rough features input the squeeze submodule directly. The squeeze submodule first replicates the rough features into two samples. One sample is the original input features (denoted as S_a), which are retained. In contrast the global spatial information of the other sample (denoted as S_b) is squeezed to a channel descriptor by global average pooling to generate channel-wise statics. Formally, a statistic $D \in \mathbb{R}^C$ is generated by shrinking the sample S_b through spatial dimensions $H \times W$, where the c-th elements of D_c is calculated by:

$$\begin{aligned} D_c &= F_{sq}(S_b) = \frac{1}{w \times H}\sum_{i=1}^{w}\sum_{j=1}^{h} S(i,j) \\ &= F_{sq}(S^{cls}_{w\times h\times 2k}) = \frac{1}{w \times H}\sum_{i=1}^{w}\sum_{j=1}^{h} S^{cls}_{w\times h\times 2k}(i,j) \\ &= F_{sq}(S^{reg}_{w\times h\times 4k}) = \frac{1}{w \times H}\sum_{i=1}^{w}\sum_{j=1}^{h} S^{reg}_{w\times h\times 4k}(i,j) \end{aligned} \tag{3}$$

where h and w represent the height and width of sample S_b in spatial dimensions.

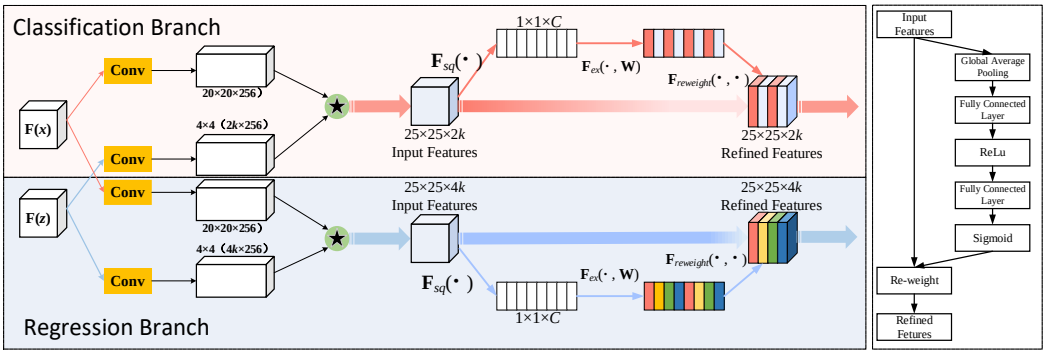


Fig. 3. The main framework diagram of SERPN, it consists of two branches, one for classification and the other for regression. Pair-wise correlation is adopted to obtain the output features of two branches. In the classification branch, the output feature map has 2k channels corresponding to the foreground and background of k anchors, and in the regression branch, the output feature maps have 4k channels corresponding to the four coordinates used for proposal refinement of k anchors. ⋆ denotes correlation operator.

To utilize the information aggregated in the squeeze operation, we follow the second that fully captures channel-wise dependencies. In the excitation submodule, we adopt a simple gating mechanism with a sigmoid activation, which ensures that the function can learn a nonlinear interaction between channels and learn a non-mutually exclusive relationship. Meanwhile, to limit model complexity, we parameterize the gating mechanism by forming a bottleneck with two fully connected layers around the nonlinear, one for dimensionality-reduction with parameters W_1 and reduction ratio r, and the other for dimensionality-increasing with parameters W_2. The operations in the excitation submodule are represented as follow:

$$R = F_{ex}(D_c, W) = \sigma\big(g(D_c, W)\big) = \sigma\big(W_2\, Relu(W_1\, D_c)\big) \tag{4}$$

where σ represents the sigmoid activation, $W_1 \in \mathbb{R}^{C/r\times C}$ and $W_2 \in \mathbb{R}^{C\times C/r}$, $Relu$ represents the ReLu function used in nonlinear. The refine outputs are performed by the channel-wise multiplication between the weight one-dimensional vector R and the original input features S_a, referred to as re-weight:

$$H = F_{reweight}(D_c, R) = D_c \otimes R \tag{5}$$

where H represents the refined features and $\otimes$ refers to channel-wise multiplication.

3.3 Multi-Level Aggregation

In the previous works, which only use shallow networks like AlexNet [27], multi-level features cannot provide very different representations. However, different layers in ResNet are much more meaningful considering that the receptive fields vary a lot. Therefore, SiamRPN++ [8] utilizes ResNet [29] as the backbone because it can capture lower-level information, which makes multi-level features aggregation possible. Similar, in our work, MobileNetV2, as the lightweight deep network that is intermediate between shallow and very deep ones, has a deeper architecture than shallow networks, which equally leads to aggregating different layers becoming possible. Benefiting from the dilated convolution mentioned in Section 3.1, the spatial resolution of the *conv3*, *conv5*, and *conv7* blocks of our backbone network are the same, the outputs of their channels are unified to 256.

In our work, multi-branch features are extracted into inferring the target localization collaboratively. As for MobileNetV2, we explore multi-level features extracted from the three convolutional layers for our multi-level aggregation. We denote these outputs as $F_3(z)$, $F_5(z)$, $F_7(z)$, respectively. As shown in Fig. 2, the outputs of *conv3*, *conv5* and *conv7* are fed into three SERPN blocks individually.

Due to the three SERPN blocks' output sizes having the same spatial resolution, the weighted sum is adopted directly on the SERPN output. All the three SERPN blocks linearly combine a final feature-fusion, which is defined as follows:

$$C_{all}^{cls} = \sum_{i=3,5,7} \alpha_i * C_l^{cls} \quad B_{all}^{reg} = \sum_{i=3,5,7} \beta_i * B_l^{reg} \tag{6}$$

where α_i and β_i are the weights corresponding to each map and are optimized together with network. By aggregating classification graphs and regression features separately, classification branches and regression branches can focus on the areas they need to.

3.4 SERPN with Classification and Regression

Classification and bounding box regression are two of the most critical tasks in anchor-based trackers. Still, the components adopted in the anchor-based tracker suffer from invisible defects that prevent further performance enhancements to the tracker.

3.4.1 Classification task

In the classification branch of the standard RPN, the IoU metric is widely used to evaluate the similarity between the predicted bounding box and the ground-truth box. Specifically, the IoU comes from the intersection area of two boxes divided by the union area of these two boxes, as defined by Equation (6). However, when multiple predicted bounding box (B_1^p, B_2^p, B_3^p, …, B_k^p) enumerated by RPN overlap with the ground-truth box (B^{gt}), although their IoU values may be equal, *i.e.*, IoU (B_1^p, B^{gt}) = IoU (B_2^p, B^{gt}) = IoU (B_3^p, B^{gt}) = IoU (B_k^p, B^{gt}), each group of boxes may overlap in different ways actually, as shown in Fig. 4, which leads to IoU metric misjudgment in complex scenes.

$$IoU = \frac{|B^P \cap B^{gt}|}{|B^P \cup B^{gt}|} \tag{7}$$

where B^p and B^{gt} represents the predicted bounding box and the ground-truth box, respectively.

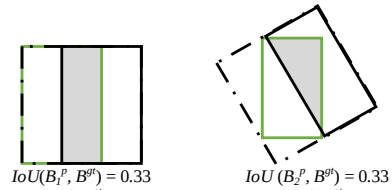


Fig. 4. The IoU metric does not reflect the overlap direction. When multiple predicted bounding boxes overlap with the ground-truth box, each group has the equal IoU value but may be completely different in the actual overlap direction. The green box is the ground-truth box, and the solid black box is the predicted bounding box.

In SERPN, we expect to introduce the DIoU metric used in object detection to solve this problem. Compared with the IoU metric, the DIoU metric is more concerned with the distance between the predicted bounding box and the ground-truth box. Based on the original IoU metric, it introduces a penalty term that is the center distance between the predicted bounding box and the ground-truth box divided by the diagonal length of the smallest enclosing shape. However, directly introducing the DIoU metric into visual tracking may adversely affect the core penalty term. Therefore, we normalize the smallest enclosing shape as the rectangle in the DIoU metric, making it more adaptive with the predicted bounding box and the ground-truth box, as described in Algorithm 1.

Algorithm 1. DIoU metric for classification tasks

Input: the predicted bounding box B^p and the ground-truth box B^{gt}

Output: DIoU metric

1. For B^p and B^{gt}, find the smallest enclosing shape E and fix it as a rectangle
2. Calculate the Euclidean distance d between the center points of B^p and B^{gt}
3. Calculate the diagonal length c of the smallest enclosing rectangle E
4. $IoU = \frac{|B^p \cap B^{gt}|}{|B^p \cup B^{gt}|}$
5. $DIoU = IoU + \frac{d^2}{c^2}$

3.4.2 Bounding box regression

Bounding box regression is the significant step for anchor-based trackers. In existing methods, while the smooth L_n-norm is widely adopted for bounding box regression, it is not matched to the evaluation method in the classification branch.

For the smooth L_n-norm loss used in the regression branch of the standard RPN, let $x_1^p, y_1^p, x_2^p, y_2^p$ denote the upper-left and lower-right coordinates of the predicted bounding box, and let $x_1^g, y_1^g, x_2^g, y_2^g$ represents the upper-left and lower-right coordinates of the ground-truth box, the offset between two boxes is calculated as follows:

$$\delta[0] = \frac{x_1^g - x_1^p}{x_2^p - x_1^p} \quad \delta[1] = \frac{y_1^g - y_1^p}{y_1^p - y_2^p}$$
$$\delta[2] = ln\frac{x_2^g - x_1^p}{x_2^p - x_1^p} \; \delta[3] = ln\frac{y_1^g - y_2^p}{y_1^p - y_2^p} \tag{8}$$

They pass through the smooth L_n-norm loss which can be written as follows:

$$smooth(x) = \begin{cases} 0.5\sigma^2 x^2 & |x| < \frac{1}{\sigma^2} \\ |x| - \frac{1}{\sigma^2} & |x| \geq \frac{1}{\sigma^2} \end{cases} \quad (9)$$

The total loss in the regression branch is obtained by directly adding up the losses for each of the individual coordinates. The following equation can define it:

$$L_{reg} = \sum_{i=0}^{3} smooth(\delta[i], \sigma) \quad (10)$$

From Equations (9) and (10), we can see that the smooth L_n-norm loss function used by the standard RPN treats the coordinates of the predicted bounding box as independent variables for optimization rather than training them as an entirety. It goes against the intuition that those variables are correlated and should be regressed jointly. Although some trackers [30–32] notice this problem and adopt the IoU loss [42] for regression task, the IoU loss only works when the predicted bounding box and the ground-truth box overlap and do not provide any moving gradient for non-overlapping cases.

To solve this problem, we explore using the DIoU metric as the bounding box regression loss because it has the advantages of IoU loss. DIoU loss also provides the moving gradient for optimization when there is no overlap between the predicted bounding box and the ground-truth box. In particular, the DIoU loss focuses on the central distance between the predicted bounding box and the ground-truth box, which pushes it to converge faster than the IoU loss to increase the tracker's speed further. Similar to IoU loss, we define DIoU loss as:

$$L_{DIoU} = 1 - DIoU = 1 - IoU + \frac{d^2}{c^2} \quad (11)$$

Specially, when the predicted bounding box and the ground-truth box perfectly match, $L_{DIoU} = 0$, and when two boxes are far away, $L_{DIoU} \to 2$.

4. Experimental Results and Analysis

4.1 Training Dataset

The backbone network of our tracker is pre-trained on ImageNet [43] for image labelling because the pre-trained network converges faster, which has been proven in other works [8]. We train the network on training dataset of ImageNet-DET [43], ImageNet-VID, COCO [44], LaSOT [45] and GOT-10k (For training) [17] to learn a generic notion of how to measure the similarities between general objects for visual tracking. In both training and testing, we use single scale images with 127 pixels for template images and 255 pixels for search images.

4.2 Implementation Details

In our experiments, we follow SiamRPN++ [8] for the training and inference settings. Our proposed method SiamSERPN is trained with stochastic gradient descent (SGD) and sets the batch size to 64. We trained a total of 50 epochs, with a warm-up learning rate of 0.001 for the first 10 epochs to train the SERPN. The last 40 epochs train the entire network end-to-end with a learning rate that exponentially decreases from 0.005 to 0.0005. Weight decay is 0.0005, and momentum is 0.9. In SiamSERPN, the loss function in the classification branch uses the cross-entropy loss, and the regression branch uses the DIoU loss from Section 3.4. The training loss is the sum of classification loss and the DIoU loss for regression. Our approach is implemented in Python using PyTorch on a PC with Intel Xeon E5-2667 v3 3.20 GHz

CPU, 32 G RAM, and Nvidia RTX 2080ti. The training time for the model lasts about a week, depending on the PC's GPU specifications. Implementation Details about hardware configurations and hyperparameters of our experiment are given in Table 3.

Table 3. Implementation details of the hardware configurations and hyper-parameters in experiments

Hardware		Software	
CPU	Intel Xeon E5-2667 v3	Training epoch	50
GPU	Nvidia RTX 2080ti	Warm-up learning rate	0.001
RAM	32G	Learning rate	0.005 to 0.0005
		Momentum	0.9
		Classification loss	Cross-entropy loss
		Regression loss	DIoU loss

4.3 Comparisons and Analyses

We compare the proposed SiamSERPN tracker with other current mainstream trackers in three extensive tracking benchmarks, including VOT2016, VOT2018, and GOT-10k (for testing). It is worth noting that the performance evaluation methods adopted in our comparisons and analysis are given by the official papers of the tracking benchmarks.

4.3.1 VOT2016 dataset

We test our SiamSERPN tracker on the VOT-2016 benchmark [15] in comparison with the current mainstream trackers. The VOT2016 public dataset is one of the most common benchmarks for evaluating single object trackers and includes 60 public video sequences with different challenge factors. Following the evaluation protocol of VOT-2016, we adopt the except average overlap (EAO), accuracy (average overlap during successful tracking periods) and robustness (failure rate) to compare different trackers. Accuracy expresses the overlap between the predicted bounding box and the ground-truth box, and it can be calculated as follows:

$$Accuracy = \frac{1}{N_{valid}} \sum_{t=1}^{N_{valid}} OS_t \tag{11}$$

where OS_t denotes the overlap rate at t frame, N_{valid} denotes the number of video sequences that are successfully tracked. As previously mentioned, the robustness represents the stability of the tracker, where the larger value indicates the poorer stability, and it calculated as:

$$Robustness = \frac{1}{N_{rep}} \sum_{k=1}^{N_{rep}} F(k) \tag{12}$$

where $F(k)$ represents the initial number of times after tracker failed. EAO combines accuracy and robustness. Firstly, all sequences in the benchmark are classified by the total number of video sequences N_s. Next, calculate the number of video frames that are tracked accurately. The EAO value of video with number N_s, can be obtained as:

$$EAO = \frac{1}{N_s} \sum_{v=1}^{N_s} Accuracy \tag{13}$$

The EAO value used for ranking trackers is the average EAO value for each frame. The detailed comparisons are listed in Table 4. Besides, we also count the average FPS while testing.

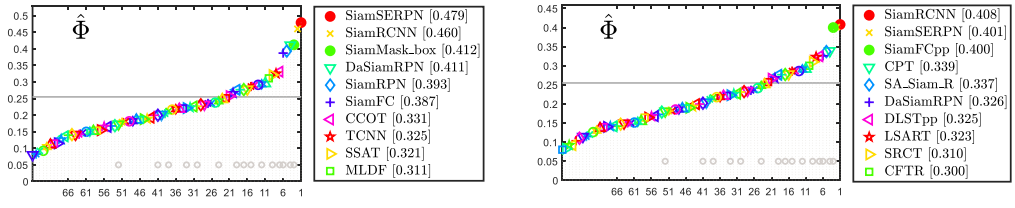
Table 4 shows the comparison results of our tracker with the current mainstream trackers, including the deep learning tracker TCNN [46], the correlation filtering tracker CCOT [47], and the mainstream Siamese network trackers, where Siamese network-based trackers include SiamRPN [11], DaSiamRPN [24], SiamMask [25] and SiamRCNN [9]. We observe that the proposed tracker SiamSERPN is the highest score of 0.479 in EAO but ranks second in accuracy and robustness, which are 0.641 and 0.191, respectively. Compared with SiamRCNN, our tracker lags 1% and 10% behind in accuracy and robustness. We think the core reason is that SiamRCNN, as a two-stage tracker, adopts the bounding box regression and the re-detection strategies, which improve the performance and stability of SiamRCNN. Still, a coin has two sides. These two strategies severely slow down SiamRCNN due to heavy hyperparameters. To the best of our knowledge, SiamRCNN's average speed does not exceed 25 FPS, while our tracker is close to 70 FPS on VOT2016 due to the lightweight backbone (Fig. 5).

Our tracker leads in all three metrics compared to SiamMask, which use standard RPN. Especially, our tracker achieves substantial gains of 14% in EAO, due to the fact that SiamMask introduces RPN from detection without any modification. In contrast, benefiting from SERPN, our tracker SiamSERPN scored 0.191 on robustness, 19% ahead of SiamMask. We believe the core reason is that the DIoU metric and DIoU loss are utilized in SERPN, both of which have a better ability than the original metric and loss function used in standard RPN to handle complex scenes in visual tracking. It is worth noting that SiamMask runs at 55 FPS [25] on the VOT2016 benchmark, while our SiamSERPN runs at close to 70 FPS, which means that the lightweight network as the backbone can provide substantial advancements in speed.

Table 4. Comparisons on VOT2016

Trackers	Accuracy	Robustness	EAO
MLDF	0.490	0.233	0.311
SSAT	0.577	0.291	0.321
TCNN	0.554	0.268	0.325
CCOT	0.539	0.238	0.331
SiamFC	0.568	0.262	0.387
SiamRPN	0.618	0.238	0.393
DaSiamRPN	0.612	0.221	0.411
SiamMask	0.623	0.233	0.412
SiamRCNN	0.645	0.172	0.460
SiamSERPN	0.641	0.191	0.479

The best results are highlighted in red, and the second-best results are blue fonts. SiamMask is the RPN-based version.



(a) (b)

Fig. 5. Excepted average overlap performance on (a) VOT2016 and (b) VOT2018 datasets. Specially, SiamFCpp is SiamFC++ and SiamMask_box is the RPN-based SiamMask.

4.3.2 VOT2018 dataset

We likewise evaluated our tracker on the VOT2018 benchmark [16], containing 60 public video sequences with several challenging topics, including fast motion, occlusion, etc. As the following

generation to VOT2016, VOT2018 also uses the three previous official metrics: accuracy, robustness, and EAO as evaluation methods.

As shown in Table 5, we compare the current mainstream trackers, including the deep learning tracker LSART [48], CPT [49] and the anchor-free method SiamFC++ [31]. Our tracker achieves the raking second only to SiamRCNN in VOT2018, where we are slightly behind SiamRCNN by 1.5% in accuracy and 1.8% in EAO, but we are the same as SiamRCNN in robustness. Meanwhile, we lag the anchor-free tracker SiamFC++ by 17% in robustness, we think the reason is that quality evaluation branch proposed by SiamFC++ greatly helps it maintain stability. The lack of a quality evaluation branch is a weakness of our tracker, which results in a less robustness score for our tracker than SiamFC++. Nevertheless, our SiamSERPN still achieves a 7.4% improvement in accuracy over it.

We set DaSiamRPN which use the standard RPN as the benchmark method to explore the differences between the proposed SERPN and RPN when testing. In Table 5, our method significantly improves more than 5.2% and 19% in accuracy and EAO, respectively. The core reason is that our proposed SERPN is more capable of performing in the face of complex scenes than RPN used in previous trackers. Besides, our tracker yields substantial gains of nearly 35% on robustness, which is the layer-wise feature aggregation to single RPN advantage.

Table 5. Comparisons with the mainstream trackers in terms of EAO, robustness (failure rate), and accuracy on the VOT2018 benchmarks

Trackers	Accuracy	Robustness	EAO
CFCT	0.505	0.258	0.300
SRCT	0.520	0.290	0.310
LSART	0.495	0.218	0.323
DLSTpp	0.543	0.224	0.325
DaSiamRPN	0.569	0.337	0.326
SA_Siam	0.566	0.258	0.337
CPT	0.506	0.239	0.339
SiamFC++	0.556	0.183	0.400
SiamRCNN	0.609	0.220	0.408
SiamSERPN	0.598	0.220	0.401

The best results are highlighted in red, and the second-best results are blue fonts. DaSiamRPN is our baseline tracker in the competition.

4.3.3 GOT-10k dataset

GOT-10k is a large diversity dataset recently released by CAS for the same object tracking in the field. It contains more than 10,000 video sequences of real-world moving objects. At the same time, the protocol guarantees the fairness of the trackers, and all methods use the same training data provided by the dataset. In particular, there are zero overlaps between the training and testing datasets classes. The tracker authors need to train on the officially given dataset, test the methods and upload the results to the official website. The evaluation is performed automatically through the official website. The AO represents the average overlaps between the estimated bounding box and the ground-truth boxes. The SR0.5 denotes the rate of successfully tracked frames whose overlap exceeds 0.5, while SR0.75 represents the rate of successfully tracked frames whose overlap exceeds 0.75.

As the details are listed in Table 6, we focus on the Siamese network trackers and deep learning ones [50, 51]. We achieve the second-best score in the GOT-10k benchmark. Compared to the two-stage tracker SiamRCNN, our tracker is weaker than it except for the speed metric. We believe the core reason is that the GOT-10k benchmark contains a large number of wild scenes, which are more complex than other visual benchmarks. Since our proposed SERPN block uses a lightweight network SENet, the proposed tracker is not strong enough to deal with complex scenes in the wild. Although we introduce DIoU to compensate for this phenomenon, it is still a weakness of our tracker. Despite this, our tracker

still achieves the second-best result compared to other trackers while running almost 27 times faster than SiamRCNN.

Table 6. Comparisons our tracker with mainstream trackers on GOT-10k

Trackers	mAO	$mSR_{0.5}$	$mSR_{0.75}$	FPS
DSiam	0.417	0.461	0.149	3.78
SiamFCv2	0.434	0.481	0.19	19.6
SA_Siam_P	0.445	0.491	0.165	25.4
DeepSTRCF	0.449	0.481	0.169	1.07
DaSiamRPN	0.444	0.536	0.22	134.4
THOR	0.447	0.538	0.204	1
SiamSERPN	0.604	0.726	0.472	75.45
SiamRCNN	0.649	0.728	0.593	2.79

The best results are highlighted in red, and the second-best results are blue fonts.

Moreover, compared to the benchmark tracker DaSiamRPN, our SERPN significantly improves the scores by 27%, 26%, and 54%, relatively for AO, SR0.5, and SR0.75. We believe that GOT-10k contains a large number of general field objects and that these scenarios are more complex and challenging. The RPN used by DaSiamRPN is incapable of handling a large number of complex cases, which results in much weaker than our SiamSERPN. However, it is worth noting that DaSiamRPN's speed of 134 FPS is the highest-ranked tracker. Although the proposed tracker is only half as fast as DaSiamRPN, it still achieves 75 FPS and far exceeds the 25 FPS. We observe that our SiamSERPN is competitive in both performance and speed types of metrics, which shows that our tracker is able to find a balance between speed and performance.

4.3.4 Summary of comparison experiments

After experiments on three large visual tracking benchmarks, the proposed tracker achieved competitive scores while running at approximately 70–75 FPS. On the VOT series benchmark, our trackers achieved competitive scores of 0.479 and 0.401, respectively. Especially on the VOT2016 benchmark, the proposed tracker SiamSERPN obtains the best performance, which adequately demonstrates that the proposed tracker equipped with SERPN blocks can achieve higher performance than the mainstream trackers that utilize the standard RPN. On the GOT-10k benchmark, SiamSERPN achieves second place in all indicators, which means that it performs better than high-speed trackers and is more efficient than high-performance trackers. One of the FPS scores is 75, substantially more than 25 FPS, which indicates that the proposed method is the real-time tracker. Finally, by combining the results of the VOT series benchmark with the GOT-10k benchmark, our tracker can achieve a balance of both performance and speed.

4.4 Ablation Study

We conduct ablation experiments on the VOT-2016 benchmark. We first explore multi-level aggregation, in which variants all use the standard RPN. After that, we test the changes carry by IoU, DIoU, and SERPN blocks, respectively.

4.4.1 Multi-level aggregation

We conduct ablation experiments on multi-layer aggregation to explore the role of different level features and the effect of multi-layer feature aggregation. We design multiple variants of the proposed SiamSERPN. At first, we do not output features at any level, similar to SiamFC, which outputs features directly through the convolution of two identical and shared parameter networks for object tracking.

However, compared to the benchmark tracker SiamRPN, the lack of RPN assistance leads to severe performance loss, despite the stronger feature extraction capability of the deep network MobileNetV2. For the variant that uses the single SERPN block, we adopt the original IoU metric in the classification branch and a standard smooth-ln loss function in the regression branch for bounding box regression. When the Siamese subnetwork feeds the extracted features to the single SERPN block, first, the standard RPN generates five scales of anchor boxes. Then the classification branch estimates whether the anchor boxes contain the object based on the IoU value (following SiamRPN, the IoU values greater than 0.6 are positive samples). Meanwhile, the regression branch performs bounding box regression on these positive samples using the smooth Ln-norm loss to obtain object position. Finally, the features are fed into the squeeze and excitation section for channel reweighting to improve the performance of the tracker. Although the use of single-layer features can lead to some performance gains, we observe that the performance of these variants is basically equivalent whether the SERPN blocks are placed on *conv3*, *conv5*, or *conv7*. Therefore, the performance gain from a single SERPN block is limited. Compared to single-layer features, performance is improved when two-layer features are aggregated, with *conv3* and *conv7* aggregation performing the best, improving by 1% over the baseline tracker SiamRPN. We believe this is because the latter SERPN block can further refine the output of the features from the previous block. As a result, after aggregating three layers of features, our tracker gradually achieves the best results.

4.4.2 Classification and regression

The classification task and the regression task play a key role in the performance of the tracker, but previous Siamese network trackers do not pay sufficient attention to them. We adopt the DIoU metric and DIoU loss in the classification branch and regression branch of RPN to distinguish foreground-background and bounding box regression, respectively. In Table 7, DIoU, as metric and loss function, leads over IoU metric and the smooth Ln-norm loss in the context of three-layer features aggregation. However, we observe that the improvement is not significant. We believe the core reason is that the single improvement is too slight to impact the tracker, which lacks the ability to handle challenging scenarios.

Table 7. Ablation study of the proposed tracker on VOT2016

Backbone	L3	L5	L7	I/S	D/DL	SERPN	EAO
AlexNet				✓			0.397
MobileNet				✓			0.377
	✓			✓			0.383
		✓		✓			0.384
			✓	✓			0.384
	✓	✓		✓			0.397
		✓	✓	✓			0.399
	✓		✓	✓			0.400
MobileNet	✓	✓	✓	✓			0.403
	✓	✓	✓		✓		0.407
	✓	✓	✓	✓		✓	0.411
MobileNet	✓	✓	✓		✓	✓	0.479

L3, L5, and L7 represent *conv3*, *conv5*, and *conv7*, respectively. I/S and D/DL represent IoU metric/smooth L_n-norm loss and DIoU/DIoU loss. SERPN denotes the RPN using squeeze and excitation operations. We set SiamRPN as the benchmark tracker, which uses AlexNet as the backbone. Besides, MobileNet represents MobileNetV2.

4.4.3 RPN with squeeze and excitation

We simply operate the squeeze and excitation operation for the RPN in this variant and do not use the DIoU metric and DIoU loss to explore the effects of SERPN. As can be seen in Table 7, although feature aggregation using SERPN blocks alone can further improve the performance of the tracker, the degree

of performance improvements is not significant. We believe the core reason is similar to that described previously: the limited performance increasing from a single improvement.

4.4.4 Summary of ablation study

Finally, we compare the proposed method SiamSERPN with all variants and the benchmark tracker, and we find a great performance improvement. Because the complete SERPN block adopts the advanced DIoU for foreground-background classification and bounding box regression. Specifically, the DIoU metric can handle the relationship between the predicted bounding box and the ground-truth box in the classification task, which is easily ignored by the IoU metric. DIoU as the loss function can jointly optimize the coordinates of the predicted bounding box during bounding box regression, resulting in more accurate location information. Based on the complete SERPN block, utilizing the deep architecture of MobileNetV2, multi-layers allow features to be aggregated and output final refined features for peak performance. After combining all the improvements, our tracker achieves the best possible performance that demonstrate proposed improvements are effective and synergistic.

5. Conclusion

In this paper, we propose a visual tracking framework that can balance performance and speed named SiamSERPN. It consists of a Siamese subnetwork and multiple proposed SERPN blocks. The former utilizes two identical lightweight MobileNetV2 as the backbone to achieve efficiency. The latter consists of the standard RPN and squeeze-excitation section to compensate for the performance loss caused by the lightweight backbone. Specifically, the proposed SERPN block improves the performance via two main strategies. One is to reweight the rough features extracted from the backbone by squeezing and excitation to retain the valuable features and filter the unnecessary ones. The other is to introduce DIoU for foreground-background classification and bounding box regression to fix the deficiencies of traditional classification metric and regression loss function adopted in standard RPN. Extensive experiments on multiple tracking benchmarks show that our trackers achieve competitive performance while operating efficiently. It scores 0.479 on the VOT2016 benchmark, which is 4% ahead of the second place. Despite being second on both the VOT2018 and GOT-10k benchmarks, it runs at 70 FPS and 75 FPS, respectively, which significantly exceeds the minimum speed requirement of real-time performance (25 FPS). Being more efficient than other anchor-based trackers is the advantage of our SiamSERPN. Still, the proposed method is also essentially an anchor-based tracker, which inherently introduces many hyperparameters and complexity. Therefore, SiamSERPN still has the potential for further speedup, which leads to a gap between SiamSERPN and practical applications. In the future, we will focus on applying lightweight networks to anchor-free trackers, which only use the expressive power of fully convolutional networks to achieve visual tracking, leading this type of trackers more efficient and taking up fewer resources than anchor-based ones. We expect future work might be able to narrow the gap between academic methods and practical applications in object tracking field.

Author's Contributions

Conceptualization, DC, MZ. Investigation and methodology, RD. Formal analysis, JW. BJ. Supervision, QA, AT. Writing of the original draft, RD. Writing of the review and editing, SSB, AM.

Funding

This work was funded by the National Natural Science Foundation of China (Grant No. 62272063, 62072056, 61902041 and 61801170), Open research fund of Key Lab of Broadband Wireless Communication and Sensor Network Technology (Nanjing University of Posts and Telecommunications), Ministry of Education, project of Education Department Cooperation Cultivation (Grant No. 201602011005 and No. 201702135098), China Postdoctoral Science Foundation (Grant No.

2018M633351), the National 13th Five National Defense Fund (Grant No. 6140311030207). Researchers Supporting Project No. RSP2023R102 King Saud University, Riyadh, Saudi Arabia.

Competing Interests

The authors declare that they have no competing interests.

References

[1] N. Qadeer, J. H. Shah, M. Sharif, M. A. Khan, G. Muhammad, and Y. D. Zhang, "Intelligent tracking of mechanically thrown objects by industrial catching robot for automated in-plant logistics 4.0," *Sensors*, vol. 22, no. 6, article no. 2113. 2022. https://doi.org/10.3390/s22062113

[2] D. Cao, B. Zheng, B. Ji, Z. Lei, and C. Feng, "A robust distance-based relay selection for message dissemination in vehicular network," *Wireless Networks*, vol. 26, pp. 1755-1771, 2020.

[3] J. Wang, H. Han, H. Li, S. He, P. K. Sharma, and L. Chen, "Multiple strategies differential privacy on sparse tensor factorization for network traffic analysis in 5G," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 3, pp. 1939-1948, 2021.

[4] D. Cao, K. Zeng, J. Wang, P. K. Sharma, X. Ma, Y. Liu, and S. Zhou, "BERT-based deep spatial-temporal network for taxi demand prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, pp. 9442-9454, 2022.

[5] N. Hussain, M. A. Khan, S. Kadry, U. Tariq, R. R. Mostafa, J. I. Choi, and Y. Nam, "Intelligent deep learning and improved whale optimization algorithm based framework for object recognition," *Human-centric Computing and Information Sciences*, vol. 11, article no. 34, 2021. https://doi.org/10.1109/HCIS.2021.11.034

[6] M. T. N. Truong and S. Kim, "A study on visual saliency detection in infrared images using Boolean map approach," *Journal of Information Processing Systems*, vol. 16, no. 5, pp. 1183-1195, 2020.

[7] J. Yu and K. M. Lee, "Multi-person tracking using SURF and background subtraction for surveillance," *Journal of Information Processing Systems*, vol. 15, no. 2, pp. 344-358, 2019.

[8] 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/CVF Conference on Computer Vision and Pattern Recognition*, Long Beach, CA, 2019, pp. 4282-4291.

[9] P. Voigtlaender, J. Luiten, P. H Torr, and B. Leibe, "Siam R-CNN: visual tracking by re-detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Seattle, WA, 2020, pp. 6577-6587.

[10] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. Torr, "Fully-convolutional Siamese networks for object tracking," in *Computer Vision – ECCV 2016 Workshops*. Cham, Switzerland: Springer International Publishing, 2016, pp. 850-865.

[11] B. Li, J. Yan, W. Wu, Z. Zhu, and H. Hu, "High performance visual tracking with Siamese region proposal network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, 2018, pp. 8971-8980.

[12] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. C. Chen, "MobileNetV2: inverted residuals and linear bottlenecks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, 2018, pp. 4510-4520.

[13] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, 2018, pp. 7132-7141.

[14] Z. Zheng, P. Wang, W. Liu, J. Li, R. Ye, and D. Ren, "Distance-IoU loss: faster and better learning for bounding box regression," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 7, pp. 12993-13000, 2020. https://doi.org/10.1609/aaai.v34i07.6999

[15] M. Kristan, A. Leonardis, and J. Matas, et al., "The visual object tracking VOT2017 challenge results," in *Proceedings of 2017 IEEE International Conference on Computer Vision Workshops*, Venice, Italy, 2017, pp. 1949-1972.

[16] M. Kristan, A. Leonardis, J. Matas, M. Felsberg, R. Pflugfelder, L. Cehovin Zajc, et al., "The sixth visual object tracking VOT2018 challenge results," in *Computer Vision – ECCV 2018 Workshops*. Cham, Switzerland: Springer, 2018, pp. 3-53.

[17] L. Huang, X. Zhao, and K. Huang, "GOT-10k: a large high-diversity benchmark for generic object tracking in the wild," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 5, pp. 1562-1577, 2021.

[18] H. Masood, A. Zafar, M. U. Ali, T. Hussain, M. A. Khan, U. Tariq, and R. Damasevicius, "Tracking of a fixed-shape moving object based on the gradient descent method," *Sensors*, vol. 22, no. 3, article no. 1098, 2022. https://doi.org/10.3390/s22031098

[19] H. Masood, A. Zafar, M. Umair Ali, M. Attique Khan, S. Ahmed, U. Tariq, B. G. Kang, and Y. Nam, "Recognition and tracking of objects in a clustered remote scene environment," *Computers, Materials & Continua*, vol. 70, no. 1, pp. 1699-1719, 2022.

[20] L. Abualigah, A. Diabat, S. Mirjalili, M. Abd Elaziz, and A. H. Gandomi, "The arithmetic optimization algorithm," *Computer Methods in Applied Mechanics and Engineering*, vol. 376, article no. 113609, 2021. https://doi.org/10.1016/j.cma.2020.113609

[21] L. Abualigah, D. Yousri, M. Abd Elaziz, A. A. Ewees, M. A. Al-Qaness, and A. H. Gandomi, "Aquila optimizer: a novel meta-heuristic optimization algorithm," *Computers & Industrial Engineering*, vol. 157, article no. 107250, 2021. https://doi.org/10.1016/j.cie.2021.107250

[22] L. Abualigah, M. Abd Elaziz, P. Sumari, Z. W. Geem, and A. H. Gandomi, "Reptile search algorithm (RSA): a nature-inspired meta-heuristic optimizer," *Expert Systems with Applications*, vol. 191, article no. 116158, 2022. https://doi.org/10.1016/j.eswa.2021.116158

[23] L. M. Q. Abualigah, *Feature Selection and Enhanced Krill Herd Algorithm for Text Document Clustering*. Cham, Switzerland: Springer, 2019.

[24] Z. Zhu, Q. Wang, B. Li, W. Wu, J. Yan, and W. Hu, "Distractor-aware Siamese networks for visual object tracking," in *Proceedings of the European Conference on Computer Vision (ECCV)*, Munich, Germany, 2018, pp. 103-119.

[25] Q. Wang, L. Zhang, L. Bertinetto, W. Hu, and P. H. Torr, "Fast online object tracking and segmentation: a unifying approach," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Long Beach, CA, 2019, pp. 1328-1338.

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

[27] R. Girshick, "Fast R-CNN," in *Proceedings of 2015 IEEE International Conference on Computer Vision*, Santiago, Chile, 2015, pp. 1440-1448.

[28] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems*, vol. 25, pp. 1106-1114, 2012.

[29] 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*, Las Vegas, NV, 2016, pp. 770-778.

[30] D. Guo, J. Wang, Y. Cui, Z. Wang, and S. Chen, "SiamCAR: Siamese fully convolutional classification and regression for visual tracking," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Seattle, WA, 2020, pp. 6268-6276.

[31] Y. Xu, Z. Wang, Z. Li, Y. Yuan, and G. Yu, "SiamFC++: towards robust and accurate visual tracking with target estimation guidelines," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 7, pp. 12549-12556, 2020. https://doi.org/10.1609/aaai.v34i07.6944

[32] Z. Chen, B. Zhong, G. Li, S. Zhang, and R. Ji, "Siamese box adaptive network for visual tracking," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Seattle, WA, 2020, pp. 6667-6676.

[33] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, San Diego, CA, 2015.

[34] K. Han, Y. Wang, Q. Tian, J. Guo, C. Xu, and C. Xu, "GhostNet: more features from cheap operations," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Seattle, WA, 2020, pp. 1577-1586.

[35] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: efficient convolutional neural networks for mobile vision applications," 2017 [Online]. Available: https://arxiv.org/abs/1704.04861.

[36] A. Howard, M. Sandler, G. Chu, L. C. Chen, B. Chen, M. Tan, et al., "Searching for MobileNetv3," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, Seoul, South Korea, 2019, pp. 1314-1324.

[37] J. Dai, Y. Li, K. He, and J. Sun, "R-FCN: object detection via region-based fully convolutional networks," *Advances in Neural Information Processing Systems*, vol. 29, pp. 379-387, 2016.

[38] T. Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Honolulu, HI, 2017, pp. 936-944.

[39] S. Qiao, L. C. Chen, and A. Yuille, "DetectoRS: detecting objects with recursive feature pyramid and switchable Atrous convolution," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Virtual Event, 2021, pp. 10213-10224.

[40] A. Bochkovskiy, C. Y. Wang, and H. Y. M. Liao, "YOLOv4: optimal speed and accuracy of object detection," 2020 [Online]. Available: https://arxiv.org/abs/2004.10934.

[41] E. Shelhamer, J. Long, and T. Darrell, "Fully convolutional networks for semantic segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 4, pp. 640-651, 2017. https://doi.org/10.1109/TPAMI.2016.2572683

[42] J. Yu, Y. Jiang, Z. Wang, Z. Cao, and T. Huang, "UnitBox: an advanced object detection network," in *Proceedings of the 24th ACM International Conference on Multimedia*, Amsterdam, The Netherlands, 2016, pp. 516-520.

[43] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, et al., "ImageNet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, pp. 211-252, 2015.

[44] T. Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollar, and C. Lawrence Zitnick, "Microsoft COCO: common objects in context," in *Computer Vision–ECCV 2014*. Cham, Switzerland: Springer International Publishing, 2014, pp. 740-755.

[45] H. Fan, L. Lin, F. Yang, P. Chu, G. Deng, S. Yu, et al., "LaSOT: a high-quality benchmark for large-scale single object tracking," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Long Beach, CA, 2019, pp. 5374-5383.

[46] H. Nam, M. Baek, and B. Han, "Modeling and propagating CNNs in a tree structure for visual tracking," 2016 [Online]. Available: https://arxiv.org/abs/1608.07242.

[47] M. Danelljan, A. Robinson, F. Shahbaz Khan, and M. Felsberg, "Beyond correlation filters: learning continuous convolution operators for visual tracking," in *Computer Vision–ECCV 2016*. Cham, Switzerland: Springer International Publishing, 2016, pp. 472-488.

[48] 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*, Salt Lake City, UT, 2018, pp. 8962-8970.

[49] M. Che, R. Wang, Y. Lu, Y. Li, H. Zhi, and C. Xiong, "Channel pruning for visual tracking," in *Computer Vision–ECCV 2018 Workshops*. Cham, Switzerland: Springer, 2018, pp. 70-82.

[50] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, Montreal, Canada, 2021, pp. 9992-10002.

[51] A. Sauer, E. Aljalbout, and S. Haddadin, "Tracking holistic object representations," in *Proceedings of the 30th British Machine Vision Conference (BMVC)*, Cardiff, UK, 2019.