SiamFC++: Towards Robust and Accurate Visual Tracking with Target Estimation Guidelines

Yinda Xu,[1*†] **Zeyu Wang,**[2*†] **Zuoxin Li,**[2] **Ye Yuan,**[2] **Gang Yu**[2‡]
[1]College of Electrical Engineering, Zhejiang University
[2]Megvii Inc.
yinda_xu@zju.edu.cn, wangzeyu0408@outlook.com, {lizuoxin, yuanye, yugang}@megvii.com

Abstract

Visual tracking problem demands to efficiently perform robust classification and accurate target state estimation over a given target at the same time. Former methods have proposed various ways of target state estimation, yet few of them took the particularity of the visual tracking problem itself into consideration. Based on a careful analysis, we propose a set of practical guidelines of target state estimation for high-performance generic object tracker design. Following these guidelines, we design our Fully Convolutional Siamese tracker++ (SiamFC++) by introducing both classification and target state estimation branch (**G1**), classification score without ambiguity (**G2**), tracking without prior knowledge (**G3**), and estimation quality score (**G4**). Extensive analysis and ablation studies demonstrate the effectiveness of our proposed guidelines. Without bells and whistles, our SiamFC++ tracker achieves state-of-the-art performance on five challenging benchmarks(OTB2015, VOT2018, LaSOT, GOT-10k, TrackingNet), which proves both the tracking and generalization ability of the tracker. Particularly, on the large-scale TrackingNet dataset, SiamFC++ achieves a previously unseen AUC score of 75.4 while running at over 90 FPS, which is far above the real-time requirement.

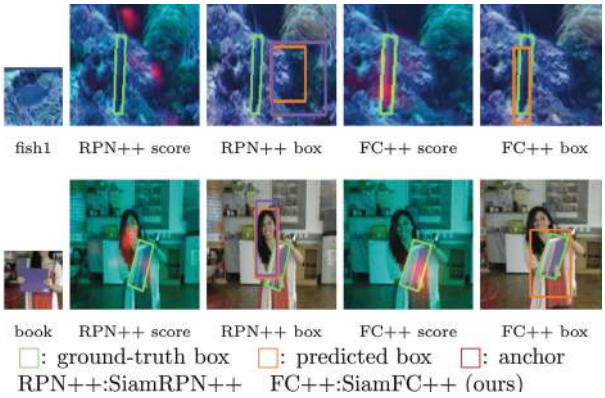

Figure 1: A comparison of our approach (following the guidelines) with state-of-the-art SiamRPN++ tracker (which violates some of the guidelines). Score maps are visualized by red color (i.e. red parts represent regions with high scores and vice versa). In the case of a significant change of target appearance, SiamRPN++ fails due to anchor-object mismatch while our SiamFC++ successes by directly matching between objects. See Section 4 for analysis in detail.

1 Introduction

Generic Visual Tracking aims at locating a moving object sequentially in a video, given very limited information, often only the annotation of the first frame. Being a fundamental build block in various areas of computer vision, the task comes with a variety of applications such as UAV-based monitoring (Mueller, Smith, and Ghanem 2016) and surveillance system (Kokkeby et al. 2015). One unique characteristic of generic object tracking is that no prior knowledge (e.g., the object class) about the object, as well as its surrounding environment, is allowed (Huang, Zhao, and Huang 2018).

Tracking problem can be treated as the combination of a *classification* task and an *estimation* task (Danelljan et al. 2019). The first task aims at providing a robust coarse location of the target via classification. The second task is then to estimate an accurate target state, often represented by a bounding box. While modern trackers have achieved significant progress, surprisingly their methods for the second task (i.e. target state estimation) largely differ. Based on this aspect, previous methods can be roughly divided into three categories. The first category, including Discriminative Correlation Filter (DCF) (Henriques et al. 2014; Bolme et al. 2010) and SiamFC (Bertinetto et al. 2016), employs brutal multi-scale test which is inaccurate (Danelljan et al. 2019) and inefficiency (Li et al. 2018a). Also, the prior assumption that target scale/ratio changes in a fixed rate in adjacent frames often does not hold in reality. For the second category, ATOM (Danelljan et al. 2019) iteratively refines multiple initial bounding boxes via gradient ascending to estimate the target bounding box (Jiang et al. 2018), which yields a significant improvement on accuracy.

[*]These authors contributed equally to this work.
[†]This work has been done when these authors were doing an intership at Megvii Inc.
[‡]Corresponding Author

However, this target estimation method brings not only a heavy computation burden but also many additional hyper-parameters (e.g. the number of initial boxes, distribution of initial boxes) that requires careful tuning. The third category is SiamRPN tracker family (Li et al. 2018a; Zhu et al. 2018; Li et al. 2019) that performs an accurate and efficient target state estimation by introducing the Region Proposal Network (RPN) (Ren et al. 2015). However, the pre-defined anchor settings not only introduce ambiguous similarity scoring that severely hinders the robustness (see Section 4) but also need access to prior information of data distribution, which is clearly against the spirit of generic object tracking (Huang, Zhao, and Huang 2018).

Motivated by the aforementioned analysis, we propose a set of guidelines for high-performance generic object tracker design:

- **G1: decomposition of classification and state estimation** The tracker should perform two sub-tasks: classification and state estimation. Without a powerful classifier, the tracker cannot discriminate the target from background or distractors, which severely hinders its robustness (Zhu et al. 2018). Without an accurate estimation result, the accuracy of the tracker is fundamentally limited (Danelljan et al. 2019). Those brutal multi-scale test approaches largely ignore the latter task, suffering from inefficiency and low accuracy.
- **G2: non-ambiguous scoring** The classification score should represent the confidence score of target existence directly, in the "field of view", i.e. sub-window of the corresponding pixel, rather than the pre-defined settings like anchor boxes. As a negative example, matching between objects and anchors (e.g. the anchor-based RPN branch) is prone to deliver a false positive result, leading to tracking failure (see Section 4 for more details).
- **G3: prior knowledge-free** Tracking approaches should be free of prior knowledge like scale/ratio distribution, as is proposed by the spirit of generic object tracking (Huang, Zhao, and Huang 2018). Dependency on prior knowledge of data distribution exists widely in existing methods, which hinders the generalization ability.
- **G4: estimation quality assessment** As is shown in previous researches (Jiang et al. 2018; Tian et al. 2019), using classification confidence for bounding box selection directly will result in degenerated performance. An estimation quality score independent of classification should be used, as in many previous pieces of research about both object detection and tracking (Jiang et al. 2018; Tian et al. 2019; Danelljan et al. 2019). The astonishing accuracy of the second branch (e.g. ATOM and DiMP) largely comes from this guideline. While the others still overlook it, leaving room for further estimation accuracy improvement.

Following the guidelines above, we design our SiamFC++ method based on fully-convolutional siamese trackers (Bertinetto et al. 2016), where each pixel of the feature map directly corresponds to each translated sub-window on the search image due to its fully convolutional nature. We add a regression head for accurate target estimation, in parallel with the classification head (**G1**). Since the pre-defined anchor settings is removed, the matching ambiguity (**G2**) and prior knowledge (**G3**) about target scale/ratio distribution is also removed. Finally, following **G4**, an estimation quality assessment branch is added to privilege bounding boxes with high quality.

Our contribution can be summarized in three-fold:

1. By identifying the unique characteristics of tracking, we devise a set of practical guidelines of target state estimation for modern tracker design.
2. We design a simple but powerful SiamFC++ tracker with the application of our proposed guidelines. Extensive experiments and comprehensive analyses demonstrate the effectiveness of our proposed guidelines.
3. Our approach achieves state-of-the-art results on five challenging benchmarks. To the best of our knowledge, our SiamFC++ is the first tracker that achieves an AUC score of 75.4 on the large-scale TrackingNet dataset (Muller et al. 2018) while running at over 90 FPS.

2 Related Works

Tracking Framework

Modern trackers can be roughly divided into three branches by their way of target state estimation.

Some of them, including DCF (Henriques et al. 2014; Bolme et al. 2010) and SiamFC (Bertinetto et al. 2016), use multi-scale test to estimate the target scale. Concretely, by rescaling the search patch into multiple scales and assembling a mini-batch of scaled images, the algorithm picks the scale corresponding to the highest classification score as the predicted target scale in the current frame. This strategy is fundamentally limited since bounding box estimation is inherently a challenging task, requiring a high-level understanding of the pose of objects (Danelljan et al. 2019).

Inspired by DCF and IoU-Net (Jiang et al. 2018), ATOM (Danelljan et al. 2019)tracks target by sequential classification and estimation. The coarse initial location of the target obtained by classification is iteratively refined for accurate box estimation. The Multiple random initializations of bounding boxes in each frame and multiple back propagations in iterative refinement greatly slows down the speed of ATOM. This approach yields a significant improvement on accuracy but also brings a heavy computation burden. What's more, ATOM introduces many additional hyper-parameters that require careful tuning.

Another branch, named SiamRPN and its succeeding works (Li et al. 2018a; Zhu et al. 2018; Li et al. 2019) append a Region Proposal Network after a siamese network, achieving a previously unseen accuracy. RPN regresses the location shift and size difference between pre-defined anchor boxes and target location. However, the RPN structure is much more fit for object detection, in which a high recall rate is required, while in visual tracking one and only one object should be tracked. Also, the ambiguous matching between anchor box and object severely hinders the robustness of tracker (see Section 4). Finally, the anchor setting does not comply with the spirit of generic object tracking, requiring pre-defined hyper-parameters describing its shape.

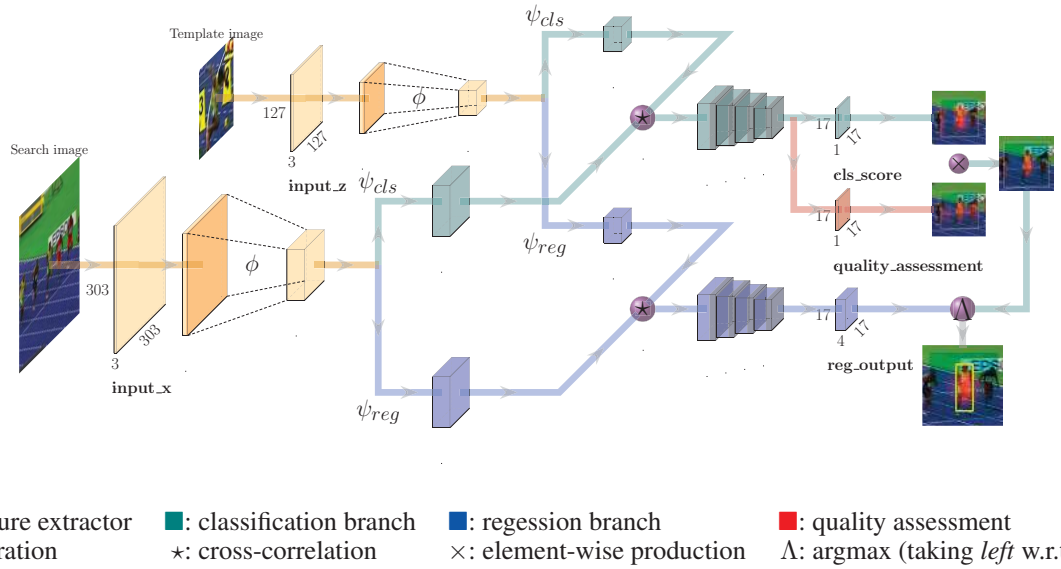

Figure 2: Our SiamFC++ pipeline (AlexNet version). Boxes denote feature maps. The intermediate layers in the common feature extractor have been omitted for clarity. For score visualization, deep green color represents the corresponding region on input image of score map, while the brightness of red color denotes the magnitude of scores (min-max normalized). Better viewed in color with zoom-in.

Detection Framework

With many unique characteristics, visual tracking task still has a lot in common with object detection, which makes each one task benefiting from each other possible. For example, the RPN structure first devised in Faster-RCNN (Ren et al. 2015) achieves astonishing accuracy in SiamRPN (Li et al. 2018a). Inheriting from Faster-RCNN (Ren et al. 2015), most state-of-the-art modern detectors, named anchor-based detectors, have adopted the RPN structure and the anchor boxes setting (Ren et al. 2015; Liu et al. 2016; Li et al. 2018b). The anchor-based detectors classifies pre-defined proposals called anchor as positive or negative patches, with an extra offsets regression to refine the prediction of bounding box locations. However, hyper-parameters introduced by anchor boxes (e.g. the scale/ratio of anchor boxes) have shown a great impact on the final accuracy, and require heuristic tuning (Cai and Vasconcelos 2018; Tian et al. 2019). Researchers have tried various ways to design anchor-free detectors, like predicting bounding boxes at points near the center of objects (Redmon et al. 2016; Huang et al. 2015), or detecting and grouping a pair of corners of a bounding box (Law and Deng 2018). In this paper, we show that a simple pipeline based on a carefully designed guidelines for target state estimation inspired by (Huang et al. 2015; Yu et al. 2016; Tian et al. 2019) can achieve state-of-the-art tracking performance.

3 SiamFC++: Fully Convolutional Siamese Tracker for Object Tracking

In this section, we describe our Fully Convolutional Siamese tracker++ framework in detail. Our SiamFC++ is based on SiamFC and progressively refined according to the proposed guidelines. As shown in Figure 2, the SiamFC++ framework consists of a siamese subnetwork for feature extraction and a region proposal subnetwork for both classification and regression.

Siamese-based Feature Extraction and Matching

Object tracking task can be viewed as a *similarity learning* problem (Li et al. 2018a). Concretely speaking, a siamese network is trained offline and evaluated online to locate a *template* image within a larger *search* image. A siamese network consists of two branches. The *template* branch takes target patch in the first frame as input (denoted as z), while the *search* branch takes the current frame as input (denoted as x). The siamese backbone, which shares parameters between two branches, performs the same transform on the input z and x to embed them into a common feature space for subsequent tasks. A cross-correlation between template patch and search patch is performed in the embedding space ϕ:

$$f_i(z,x) = \psi_i\left(\phi\left(z\right)\right) \star \psi_i\left(\phi\left(x\right)\right), i \in \{\text{cls}, \text{reg}\} \qquad (1)$$

where $\star$ denotes the cross-correlation operation, $\phi(.)$ denotes the siamese backbone for common feature extraction, $\psi_i(.)$ denotes the task-specific layer and i denotes the subtask type ("cls" for classification and "reg" for regression). In our implementation, We use two convolution layers for both ψ_{cls} and ψ_{reg} after common feature extraction to adjust the common features into task-specific feature space. Note that the extracted features of ψ_{cls} and ψ_{reg} are of the same size.

Application of Design Guidelines in Head Network

Based on SiamFC, we progressively refine each part of our trackers following our guidelines.

Following **G1**, we design both classification head and regression head after the cross-correlation in the embedding space. For each pixel in feature maps, the classification head takes ψ_{cls} as input and classifies the corresponding image patch as either one positive or negative patch, while the regression head takes ψ_{reg} as input and outputs an extra offsets regression to refine the prediction of bounding box locations. The structure of heads is presented after the cross-correlation operation of Figure 2.

Specifically, for classification, location (x, y) on feature map ψ_{cls} is considered as a positive sample if its corresponding location $\left(\left\lfloor \frac{s}{2} \right\rfloor + xs, \left\lfloor \frac{s}{2} \right\rfloor + ys\right)$ on the input image falls into the ground-truth bounding box. Otherwise, it is a negative sample. Here s is the total stride of backbone ($s = 8$ in this paper). For the regression target of each positive location (x, y) on feature map ψ_{reg}, the final layer predicts the distances from the corresponding location $\left(\left\lfloor \frac{s}{2} \right\rfloor + xs, \left\lfloor \frac{s}{2} \right\rfloor + ys\right)$ to the four sides of the ground-truth bounding box, denoted as a 4D vector $\boldsymbol{t}^* = (l^*, t^*, r^*, b^*)$. Hence, the regression targets for location (x, y) can be formulated as

$$\begin{aligned} l^* &= \left(\left\lfloor \frac{s}{2} \right\rfloor + xs\right) - x_0, \quad t^* = \left(\left\lfloor \frac{s}{2} \right\rfloor + ys\right) - y_0 \\ r^* &= x_1 - \left(\left\lfloor \frac{s}{2} \right\rfloor + xs\right), \quad b^* = y_1 - \left(\left\lfloor \frac{s}{2} \right\rfloor + ys\right) \end{aligned} \tag{2}$$

where (x_0, y_0) and (x_1, y_1) denote the left-top and right-bottom corners of the ground-truth bounding box B^* associated with point (x, y).

Each location (x, y) on the feature map of both classification and regression head, corresponds to an image patch on the input image centered at location $\left(\left\lfloor \frac{s}{2} \right\rfloor + xs, \left\lfloor \frac{s}{2} \right\rfloor + ys\right)$. Following **G2**, we directly classify corresponding image patch and regress the target bounding box at the location, as in many previous tracker (Henriques et al. 2014; Bolme et al. 2010; Bertinetto et al. 2016). In other words, our SiamFC++ directly views locations as training samples. While the anchor-based counterparts (Li et al. 2018a; Zhu et al. 2018; Li et al. 2019), which consider the location on the input image as the center of multiple anchor boxes, output multiple classification score at the same location and regress the target bounding box with respect to these anchor boxes, leading to ambiguous matching between anchor and object. Although (Li et al. 2018a; Zhu et al. 2018; Li et al. 2019) have shown superior performance on various benmarks than (Henriques et al. 2014; Bolme et al. 2010; Bertinetto et al. 2016), we empirically show that the ambiguous matching could result in serious issues (see Section 4 for more details). In our per-pixel prediction fashion, only one prediction is made at each pixel on the final feature map. Hence it is clear that each classification score directly gives the confidence that the target is in the sub-window of the corresponding pixel and our design is free of ambiguity to this extent.

Since SiamFC++ does classification and regression w.r.t. the location, it is free of pre-defined anchor boxes, hence free of prior knowledge about target data distribution (e.g. scale/ratio), which comply with **G3**.

During the above sections, we do not take the target state estimation quality into consideration yet and directly use classification score to select the final box. That could cause the degradation of localization accuracy, as (Jiang et al. 2018) shows that classification confidence is not well correlated with the localization accuracy. According to the analysis in (Luo et al. 2016), input pixels around the center of a sub-window will have larger importance on the corresponding output feature pixel than the rest. Thus we hypothesize that feature pixels around the center of objects will have a better estimation quality than others. Following **G4**, we add a simple yet effective quality assessment branch similar to (Tian et al. 2019; Jiang et al. 2018) by appending a 1×1 convolution layer in parallel with the 1×1 convolution classification head, as shown in the right part of Figure 2. The output is supposed to estimate the Prior Spatial Score (PSS) which is defined as follows:

$$\text{PSS}^* = \sqrt{\frac{\min(l^*, r^*)}{\max(l^*, r^*)} \times \frac{\min(t^*, b^*)}{\max(t^*, b^*)}} \tag{3}$$

Note that PSS is not the only choice for quality assessment. As a variant, we can also predict the IoU score between predicted boxes and ground-truth boxes similar to (Jiang et al. 2018):

$$\text{IoU}^* = \frac{\text{Intersection}(B, B^*)}{\text{Union}(B, B^*)} \tag{4}$$

where B is the predicted bounding box and B^* is its corresponding ground-truth bounding box.

During inference, the score used for final box selection is computed by multiplying the PSS with the corresponding predicted classification score. In this way, those bounding boxes far from the center of objects will be downweighted seriously. Thus the tracking accuracy is improved.

Training Objective

We optimize a training objective as follows:

$$\begin{aligned} L\left(\{p_{x,y}\}, q_{x,y}, \{\boldsymbol{t}_{x,y}\}\right) = & \frac{1}{N_{\text{pos}}} \sum_{x,y} L_{\text{cls}}\left(p_{x,y}, c^*_{x,y}\right) \\ & + \frac{\lambda}{N_{\text{pos}}} \sum_{x,y} \mathbf{1}_{\{c^*_{x,y} > 0\}} L_{\text{quality}}\left(q_{x,y}, q^*_{x,y}\right) \\ & + \frac{\lambda}{N_{\text{pos}}} \sum_{x,y} \mathbf{1}_{\{c^*_{x,y} > 0\}} L_{\text{reg}}\left(\boldsymbol{t}_{x,y}, \boldsymbol{t}^*_{x,y}\right) \end{aligned} \tag{5}$$

where $\mathbf{1}_{\{\cdot\}}$ is the indicator function that takes 1 if the condition in subscribe holds and takes 0 if not, L_{cls} denote the focal loss (Lin et al. 2017) for classification result, L_{quality} denote the binary cross entropy (BCE) loss for quality assessment and L_{reg} denote the IoU loss (Yu et al. 2016) for bounding box result. We assign 1 to $c^*_{x,y}$ if (x, y) is considered as a positive sample, and 0 if as a negative sample.

No.	VID	Youtube	COCO&Det& LaSOT&GOT	Backbone	Head type	Head structure	Quality assessment	A	R	EAO	ΔEAO
1	✓	×	×	AlexNet	cls	0×conv3 × 3	None	0.506	0.566	0.213	0
2	✓	✓	×	AlexNet	cls	0×conv3 × 3	None	0.532	0.407	0.276	+0.063
3	✓	✓	×	AlexNet	cls	3×conv3 × 3	None	0.539	0.337	0.296	+0.083
4	✓	✓	✓	AlexNet	cls	3×conv3 × 3	None	0.536	0.323	0.306	+0.093
5	✓	✓	✓	AlexNet	cls+reg	3×conv3 × 3	PSS	0.556	**0.183**	0.400	+0.187
6	✓	✓	✓	GoogLeNet	cls+reg	2×conv3 × 3	PSS	**0.587**	**0.183**	**0.426**	**+0.213**

Table 1: Ablation study: from SiamFC towards SiamFC++. Experiments have been conducted on VOT-2018 (A/R/EAO). ΔEAO denotes the augmentation of EAO w.r.t. the baseline (Line 1).

4 Experiments

Implementation Details

Model settings In this work, we implement two versions of trackers with different backbone architectures: the one that adopts the modified version of AlexNet in the previous literature (Bertinetto et al. 2016), denoted as SiamFC++-AlexNet, and another one that uses GoogLeNet (Szegedy et al. 2015), denoted as SiamFC++-GoogLeNet. With lower computation cost, the later achieves the same or even better performance(see Section 4) on tracking benchmark than same previous methods using ResNet-50 (He et al. 2016). Both networks are pretrained on ImageNet (Krizhevsky, Sutskever, and Hinton 2012), which has been proven practical for tracking task (Li et al. 2018a; Zhu et al. 2018). We will release the code to facilitate further researches.

Training data We adopt ILSVRC-VID/DET (Russakovsky et al. 2015), COCO (Lin et al. 2014) , Youtube-BB (Real et al. 2017), LaSOT (Fan et al. 2019) and GOT-10k (Huang, Zhao, and Huang 2018) as our basic training set. Exceptions w.r.t. to specific benchmarks are detailed in the following subsections. For video datasets, we extract image pairs from VID, LaSOT, and GOT-10k by choosing frame pairs within an interval of less than 100 (5 for Youtube-BB). For image datasets (COCO/Imagenet-DET), we generate training samples by involving negative pairs (Zhu et al. 2018) as part of training samples to enhance the capacity to distinguish distractors of our model. We perform random shifting and scaling following a uniform distribution on the search image as data augmentation techniques.

Training phase For the AlexNet version, we freeze the parameters from conv1 to conv3 and fine-tune conv4 and conv5. For those layers without pretraining, we adopt a zero-centered Gaussian distribution with a standard deviation of 0.01 for initialization. We first train our model with for 5 warm up epochs with learning rate linearly increased from 10^{-7} to 2×10^{-3}, then use a cosine annealing learning rate schedule for the rest of 45 epochs, with 600k image pairs for each epoch. We choose stochastic gradient descent (SGD) with a momentum of 0.9 as the optimizer.

For the version implemented with GoogLeNet, we freeze stage 1 and 2, fine-tune stage 3 and 4, augment the base learning rate to 2×10^{-2}, and multiply the learning rate of parameters in the backbone by 0.1 w.r.t the global learning rate. We also reduce the number of image pairs per epoch to 300k, reduce the total epoch to 20 (thus 5 for warming-up, and 15 for training) and unfreeze the parameters in backbone at the 10th epoch to avoid overfitting. For the experiment on LaSOT benchmark (Fan et al. 2019) (protocol II), we freeze the parameters in the backbone and further reduce the number of image pairs per epoch to 150k so that the training with the relatively smaller amount of training data could be stabilized.

The proposed tracker with AlexNet backbone runs at 160 FPS on the VOT2018 short-term benchmark, while the one with GoogleNet backbone runs at about 90 FPS on the VOT2018 short-term benchmark, both evaluated on an NVIDIA RTX 2080Ti GPU.

Test phase The output of our model is a set of bounding boxes with their corresponding confidence scores s. Scores are penalized based on the scale/ratio change of corresponding boxes and distance away from the target position predicted in the last frame. Then the box with the highest penalized score is chosen and is used to update the target state.

From SiamFC towards SiamFC++

While both employing a per-pixel prediction fashion, there exists a significant performance gap between SiamFC and our SiamFC++. In this subsection we perform an ablation study on VOT2018 dataset, with SiamFC as the baseline, aiming at identifying the key component for the improvement of tracking performance.

Results are shown in Table 1. Concretely, in the SiamFC baseline, the tracker only performs classification tasks in its network and the target state estimation is done with multi-scale test. We gradually update SiamFC tracker by using extra training data (Line 2/4), applying a better head structure (Line 3), and adding the regression branch for accurate estimation to yield our proposed SiamFC++ tracker (Line 5). We further replace the AlexNet backbone with GoogLeNet which is more powerful to extract visual feature (Line 6).

The key components for tracking performance can be listed in descending order as follows: the regression branch (0.094), data source diversity (0.063/0.010), stronger backbone (0.026), and better head structure (0.020), where the ΔEAO brought by each part is noted in parentheses. Note that these are the extra components of SiamRPN++ over SiamFC. After adding all the extra components into SiamFC, Our SiamFC++ achieves superior performance with less computation budget. Also, there are two things worth to mention: 1). the robustness (**R**) of Line 2 surpasses SiamRPN tracker (0.46 (Li et al. 2018a)); 2). the **R** of Line 3

is at the same level of DaSiamRPN (0.337 (Zhu et al. 2018)) while using less data (without COCO and DET) than the latter. These results indicate that, while the introduction of the RPN module and anchor boxes setting undoubtedly gives better accuracy, its robustness is not improved and even hindered. We owe this to its violation of our proposed guidelines.

Quality Assessment Choice On GOT-10k *val* subset, we obtain an AO of 77.8 for the tracker predicting PSS and an AO of 78.0 for the tracker predicting IoU. Experiments have been conducted with SiamFC++-GoogLeNet. We finally choose PSS in this paper as an implementation of our approach for its stability empirically observed across datasets during our experiment.

Results on Several Benchmarks

We test our tracker on several benchmarks and results are gathered in Table 2.

Results on OTB2015 Benchmark As one of the most classical benchmarks for the object tracking task, the OTB benchmark (Wu, Lim, and Yang 2013) provides a fair test for all families of trackers. We conduct experiments on OTB2015 (Wu, Lim, and Yang 2013) which contains 100 videos for tracker performance evaluation. With a success score of 0.682, our tracker reaches the state-of-the-art level w.r.t. other trackers in comparison.

Results on VOT Benchmark VOT2018 (Kristan et al. 2018) contains 60 video sequences with several challenging topics including fast motion, occlusion, etc. We test our tracker on this benchmark and present the results in Table 2. Both versions of our trackers reaching comparable scores w.r.t. current state-of-the-art trackers, the tracker with AlexNet backbone outperforms other trackers with the same tracking speed and while the tracker with GoogLeNet backbone yields a comparable score. Besides, our tracker has a significant advantage in the robustness among the trackers in comparison. To the best of our knowledge, this is the first tracker that achieves an EAO of 0.400 on VOT2018 (Kristan et al. 2018) benchmark while running at a speed over 100 FPS, which demonstrate its potential of being applied in real production cases.

Results on LaSOT Benchmark With a large number of video sequences (1400 sequences under Protocol I while 280 under Protocol II), LaSOT (Fan et al. 2019) (Large-scale Single Object Tracking) benchmark makes it impossible for trackers to overfit the benchmark, which achieves the purpose of testing the real performance of object tracking. Following Protocol II under which trackers are trained on LaSOT *train* subset and evaluated on LaSOT *test* subset, the proposed SiamFC++ tracker achieves better performance, even w.r.t. those who have better performance than ours on the VOT2018 benchmark. This reveals the fact that the scale of the benchmark influences the rank of trackers.

Results on GOT-10k Benchmark For target class generalization testing, we train and test our SiamFC++ model on GOT-10k (Huang, Zhao, and Huang 2018) (Generic Object Tracking-10k) benchmark. Not only as a large-scale dataset (10,000 videos in train subset and 180 in both *val* and *test* subset), it also gives challenges in terms of the requirement of category-agnostic for generic object trackers as there is no class intersection between *train* and *test* subsets. We follow the protocol of GOT-10k and only trained our tracker on the *train* subset. Our tracker with AlexNet backbone reaches an AO of 53.5 surpassing SiamRPN++ by 1.7, while our tracker with GoogLeNet backbone yields 59.5 which is even superior to ATOM that uses online updating method. This result shows the ability of our tracker to generalize even the target classes are unseen during the training phase, which matches the demand of the generic tracking.

Trackers		SiamFC (2016)	ECO (2017)	MDNet (2016)	SiamRPN++ (2019)	ATOM (2019)	**SiamFC++-AlexNet**	**SiamFC++-GoogLeNet**
OTB-15	Success	58.2	**70.0**	67.8	*69.6*	66.9	65.6	<u>68.3</u>
VOT-18	A	0.503	0.484	-	**0.600**	*0.590*	0.556	<u>0.587</u>
	R	0.585	0.276	-	0.234	<u>0.204</u>	**0.183**	**0.183**
	EAO	0.188	0.280	-	*0.414*	<u>0.401</u>	0.400	**0.426**
LaSOT	Success	33.6	32.4	39.7	49.6	*51.5*	<u>50.1</u>	**54.4**
GOT	$SR_{.5}$	35.3	30.9	30.3	<u>61.8</u>	*63.4*	57.7	**69.5**
	$SR_{.75}$	9.8	11.1	9.9	<u>32.5</u>	*40.2*	32.3	**47.9**
	AO	34.8	31.6	29.9	<u>51.8</u>	*55.6*	49.3	**59.5**
T-Net	Prec.	51.8	49.2	56.5	*69.4*	<u>64.8</u>	64.6	**70.5**
	Norm. Prec.	65.2	61.8	70.5	**80.0**	<u>77.1</u>	75.8	**80.0**
	Succ.	55.9	55.4	60.6	*73.3*	70.3	<u>71.2</u>	**75.4**
FPS		<u>86</u>	8	1	35	30	**160**	*90*

Table 2: Results on several benchmarks. T-Net denotes TrackingNet. Top-3 results of each dimension (row) are boldfaced, italicized, and underlined, respectively.

Results on TrackingNet Benchmark We evaluate our approach with 511 videos provided in the *test* split of TrackingNet (Muller et al. 2018). We exclude the Youtube-BB dataset from our training data in order to avoid data leak. As is described in (Muller et al. 2018), the evaluation server calculates the following three indexes based on tracking results: success rate, precision, and normalized precision. Our SiamFC++-GoogLeNet outperforms the current state-of-the-art methods (including online-update methods like (Danelljan et al. 2019)) in both precision and success rate dimensions, while our lightweight version SiamFC++ strikes a balance between performance and the speed. This result is achieved even without Youtube-BB containing a large portion of training data, which shows that the potential of our approach to be independent of large offline training data.

Comparison with Trackers that Do not Apply Our Guidelines

The family of SiamRPN (Li et al. 2018a; Zhu et al. 2018; Li et al. 2019) has achieved great success in visual tracking these years and drawn much attention from tracking community. Here we use state-of-the-art SiamRPN++ tracker as an example. Despite recent successes of the SiamRPN family, we have found that the SiamRPN tracker and its family do not follow our proposed guidelines entirely.

- **(G2)** the classification score of SiamRPN represents the similarity between anchor and object, rather than template object and objects in search image, which may cause

matching ambiguity;

- **(G3)** the design of pre-set anchor boxes needs prior knowledge of the distribution of size and ratio of target;
- **(G4)** the choice of target state estimation does not take estimation quality into consideration.

Note that the SiamRPN family adopts proposal refinement by the regression branch instead of the multi-scale test, and thus achieves astonishing tracking accuracy, which complies with our guideline **G1**.

As a consequence of the violation of guideline **G2**, we empirically find that the SiamRPN family is prone to deliver a false-positive result. In other words, SiamRPN will produce an unreasonable high score for nearby objects or background under large appearance variation of target object. As shown in Figure 1, we can see that SiamRPN++ fails to track the target object by giving very high scores for nearby objects (i.e. a rock or a face) under challenging scenarios like out-of-plane rotation and deformation. We hypothesize that SiamRPN matches objects and anchors rather than the object itself, which may deliver drifts and thus hinders its robustness. On the contrary, our proposed SiamFC++, which matches between template objects and objects in search image directly, gives accurate score predictions and successfully tracks the target.

To verify our hypothesis, we record the max score produced by SiamRPN++ and our proposed SiamFC++ on VOT2018 dataset. We then split them according to the tracking result, e.g., successful or failed. On VOT2018, a tracking result is considered failed if its overlap with the ground-truth box is zero. Otherwise, it is considered successful. The result is visualized in the first row in Figure 3. Comparing SiamRPN++ and SiamFC++ scores, we can see that most classification score of SiamRPN++ follows similar and highly overlapped distributions, successful or not, while the classification score of our SiamFC++ of failure state exhibit very different pattern with that of a successful state.

Another factor contributing to the ambiguity in SiamRPN++ is that the feature matching process is done with patches of fixed aspect ratio (multiple patches with different ratios will bring non-negligible computation cost), while each pixel of the feature after matching is assigned anchors whose aspect ratio varies.

As for the violation of **G3**, the performance of SiamRPN varies as the scales and ratios of anchors vary. As is shown in Table 3 from (Li et al. 2018a), three different ratio settings are tried and the performance of SiamRPN varies when using different anchor settings. Thus the best performance is achieved only by accessing prior knowledge of data distribution, which is against the spirit of generic object tracking (Huang, Zhao, and Huang 2018).

Besides, in the second row of Figure 3, we also plot the histogram of SiamRPN++ statistics of IoU between output bounding box and ground truth and the histogram between anchor and ground truth, in both success and failure state. As is shown from the IoU distribution, the prior knowledge given by anchor settings (violation of **G3**) leads to a bias in target state estimation. Concretely, the predicted box of SiamRPN++ tends to overlap more with the anchor box than

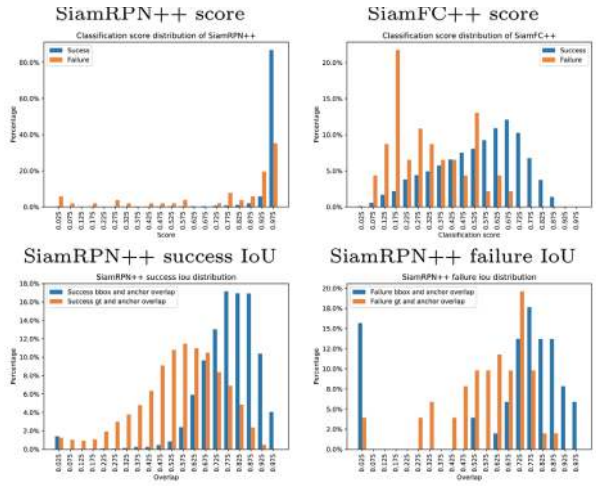

Figure 3: The first row: score distribution of SiamRPN++ and SiamFC++. The second row: IoU distribution of SiamRPN++ in both success/failure state. Better visualized when zoomed in.

with the ground truth box which can lead to performance degradation.

As for the violation of **G4**, we can see that the $SR_{.5}$ and $SR_{.75}$ of SiamRPN++ on GOT-10k benchmark are 7.7 and 15.4 points lower than those of SiamFC++, respectively. In GOT-10k, the Success Rate (SR) measures the percentage of successfully tracked frames where the overlaps exceed a predefined threshold (i.e., 0.5 or 0.75). The higher the threshold, the more accurate the tracking result. Hence SR is a solid indicator for estimation quality. The $SR_{.75}$ of SiamRPN++ is much lower than that of SiamFC++, indicating the lower estimation quality of SiamRPN++ caused by the violation of guideline **G4**.

5 Conclusion

In this paper, we propose a set of guidelines for target state estimation in tracker design, by analyzing the unique characteristics of visual tracking tasks and the flaws of former trackers. Following these guidelines, we propose our approach that provides effective methods for both classification and target state estimation (**G1**), giving classification score without ambiguity (**G2**), tracking without prior knowledge (**G3**), and being aware of estimation quality (**G4**). We verify the effectiveness of proposed guidelines by extensive ablation study. And we show that our tracker based on these guidelines reaches state-of-the-art performance on five challenging benchmarks, while still running at 90 FPS.

References

Bertinetto, L.; Valmadre, J.; Henriques, J. F.; Vedaldi, A.; and Torr, P. H. 2016. Fully-convolutional siamese networks for object tracking. In *European conference on computer vision*, 850–865. Springer.

Bolme, D. S.; Beveridge, J. R.; Draper, B. A.; and Lui, Y. M. 2010. Visual object tracking using adaptive correlation filters. In *2010*

IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2544–2550. IEEE.

Cai, Z., and Vasconcelos, N. 2018. Cascade r-cnn: Delving into high quality object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 6154–6162.

Danelljan, M.; Bhat, G.; Shahbaz Khan, F.; and Felsberg, M. 2017. Eco: efficient convolution operators for tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 6638–6646.

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

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

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Henriques, J.; Caseiro, R.; Martins, P.; and Batista, J. 2014. High-speed tracking with kernelized correlation filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37.

Huang, L.; Yang, Y.; Deng, Y.; and Yu, Y. 2015. Densebox: Unifying landmark localization with end to end object detection. *arXiv preprint arXiv:1509.04874*.

Huang, L.; Zhao, X.; and Huang, K. 2018. Got-10k: A large high-diversity benchmark for generic object tracking in the wild. *arXiv preprint arXiv:1810.11981*.

Jiang, B.; Luo, R.; Mao, J.; Xiao, T.; and Jiang, Y. 2018. Acquisition of localization confidence for accurate object detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 784–799.

Kokkeby, K. L.; Lutter, R. P.; Munoz, M. L.; Cathey, F. W.; Hilliard, D. J.; and Olson, T. L. 2015. Methods for autonomous tracking and surveillance. US Patent 9,026,272.

Kristan, M.; Leonardis, A.; Matas, J.; Felsberg, M.; Pflugfelder, R.; Cehovin Zajc, L.; Vojir, T.; Bhat, G.; Lukezic, A.; Eldesokey, A.; et al. 2018. The sixth visual object tracking vot2018 challenge results. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 0–0.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, 1097–1105.

Law, H., and Deng, J. 2018. Cornernet: Detecting objects as paired keypoints. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 734–750.

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

Li, Z.; Peng, C.; Yu, G.; Zhang, X.; Deng, Y.; and Sun, J. 2018b. Detnet: Design backbone for object detection. In *The European Conference on Computer Vision (ECCV)*.

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

Lin, T.-Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; and Zitnick, C. L. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*, 740–755. Springer.

Lin, T.-Y.; Goyal, P.; Girshick, R.; He, K.; and Dollár, P. 2017. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, 2980–2988.

Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.-Y.; and Berg, A. C. 2016. Ssd: Single shot multibox detector. In *European conference on computer vision*, 21–37. Springer.

Luo, W.; Li, Y.; Urtasun, R.; and Zemel, R. 2016. Understanding the effective receptive field in deep convolutional neural networks. In *Advances in neural information processing systems*, 4898–4906.

Mueller, M.; Smith, N.; and Ghanem, B. 2016. A benchmark and simulator for uav tracking. In *European conference on computer vision*, 445–461. Springer.

Muller, M.; Bibi, A.; Giancola, S.; Alsubaihi, S.; and Ghanem, B. 2018. Trackingnet: A large-scale dataset and benchmark for object tracking in the wild. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 300–317.

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

Real, E.; Shlens, J.; Mazzocchi, S.; Pan, X.; and Vanhoucke, V. 2017. 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*, 5296–5305.

Redmon, J.; Divvala, S.; Girshick, R.; and Farhadi, A. 2016. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 779–788.

Ren, S.; He, K.; Girshick, R.; and Sun, J. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, 91–99.

Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. 2015. Imagenet large scale visual recognition challenge. *International journal of computer vision* 115(3):211–252.

Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; and Rabinovich, A. 2015. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1–9.

Tian, Z.; Shen, C.; Chen, H.; and He, T. 2019. Fcos: Fully convolutional one-stage object detection. *arXiv preprint arXiv:1904.01355*.

Wu, Y.; Lim, J.; and Yang, M.-H. 2013. Online object tracking: A benchmark. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2411–2418.

Yu, J.; Jiang, Y.; Wang, Z.; Cao, Z.; and Huang, T. 2016. Unitbox: An advanced object detection network. In *Proceedings of the 24th ACM international conference on Multimedia*, 516–520. ACM.

Zhu, Z.; Wang, Q.; Li, B.; Wu, W.; Yan, J.; and Hu, W. 2018. Distractor-aware siamese networks for visual object tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 101–117.