Teacher-Students Knowledge Distillation for Siamese Trackers

Yuanpei Liu 1, Xingping Dong 2, Wenguan Wang 2, and Jianbing Shen∗ 2,1

1Beijing Laboratory of Intelligent Information Technology, School of Computer Science, Beijing Institute of Technology, China, 2Inception Institute of Artificial Intelligence, Abu Dhabi, UAE

Abstract

With the development of Siamese network based trackers, a variety of techniques have been fused into this framework for real-time object tracking. However, Siamese trackers suffer from the dilemma between high memory cost and strict constraints on memory budget for practical applications. In this paper, we propose a novel distilled Siamese tracker framework to learn small, fast yet accurate trackers (students), which can capture critical knowledge from large Siamese trackers (teachers) by a teacher-students knowledge distillation model. This model is intuitively inspired by a one teacher vs multi-students learning mechanism, which is the most usual teaching method in the school. In particular, it contains a single teacher-student distillation model and a student-student knowledge sharing mechanism. The first one is designed by a tracking-specific distillation strategy to transfer knowledge from teacher to students. The second one is applied for mutual learning between students to enable more in-depth knowledge understanding. Moreover, to demonstrate its generality and effectiveness, we conduct a theoretical analysis and extensive empirical evaluations on two Siamese trackers, on several popular tracking benchmarks. The results show that the distilled trackers achieve compression rates of $13\times - 18\times$, while maintaining the same or even slightly improved tracking accuracy.

1. Introduction

Recent years have witnessed various approaches of Siamese network for visual tracking task because of their balance between accuracy and speed. The pioneering work SiamFC [4] proposed a simple yet effective tracking framework by designing a Siamese network for offline training to learn a metric function and convert the tracking task to template matching using the learned metric. This framework is an ideal baseline for real-time tracking since its simple architecture is easy to be combined with other techniques and the high speed of nearly 86 Frame-Per-Second (FPS) allows adding these skills to improve accuracy and simultaneously maintain real-time speed (30 FPS). Since then, many real-time trackers [38, 17, 20, 52, 42, 18, 14, 43, 13, 45, 49, 24, 51] have been proposed to improve its accuracy through various of techniques. Along with this line, the recent tracker SiamRPN [24, 22] (the champion of VOT-2018 [22] real-time challenge), achieved significant improvement of accuracy and high speed (nearly 90 FPS), by applying a Region Proposal Network (RPN) to directly regress the position and scale of objects. This method will likely become the next baseline to further promote real-time tracking, due to its high speed and impressive accuracy.

Despite being studied actively with remarkable progress, Siamese-network based visual trackers generally face a conflict between high memory cost and strict constraints on memory budget in real-world applications, especially for SiamRPN [24, 22], whose model size is up to 361.8 MB. Their high memory cost makes them undesirable for practical mobile visual tracking applications, such as accurate trackers running real-time on a drone, smartphone or sensor nodes. How to decrease the memory cost of Siamese
trackers without a remarkable loss of tracking accuracy is one of the key points to build the bridge between the academic algorithms and practical applications. In the other aspect, reducing model size will directly decrease the computational cost to produce a faster tracker. If the faster tracker achieves similar accuracy as the larger one, like SiamFC or SiamRPN, it will be another attractive baseline to facilitate real-time tracking.

To address the above points, we propose a novel Distilled Siamese Trackers (DST) framework built upon a Teacher-Students Knowledge Distillation (TSsKD) model, which is specially designed for learning a small, fast yet accurate Siamese tracker through Knowledge Distillation (KD) techniques. TSsKD essentially explores a one teacher vs multi-students learning mechanism inspired by the most usual teaching and learning methods in the schools, i.e., multiple students learn knowledge from a teacher and help each other to facilitate learning effect. In particular, TSsKD models two kinds of knowledge distillation styles. First, knowledge transfer from teacher to students, which is achieved by a tracking-specific distillation strategy. Second, mutual learning between students, working in a student-student knowledge sharing manner.

More specifically, to inspire more efficient and tracking-specific knowledge distillation within the same domain (without additional data or labels), the teacher-student knowledge transfer is equipped with a set of carefully designed losses, i.e., a teacher soft loss, adaptive hard loss, and Siamese attention transfer loss. The first two allow the student to mimic the high-level semantic information of the teacher and ground-truth while reducing over-fitting, and the last one incorporated with Siamese structure is applied to learn the middle-level semantic hints. To further enhance the performance of the student tracker, we introduce a knowledge sharing strategy with a conditional sharing loss that encourages sharing reliable knowledge between students. This provides extra guidance that facilitates small-size trackers (the “dull” students) to establish a more comprehensive understanding of the tracking knowledge and thus achieve higher accuracy.

As a summary, our key contributions include

- For the knowledge transfer model, we design a set of losses to tightly couple the Siamese structure and also decrease the over-fitting during training for better tracking performance. For the knowledge sharing mechanism, a conditional sharing loss is proposed to transfer reliable knowledge between students and further enhance the “dull” students.

Extensive empirical evaluations for famous SiamFC [4] and SiamRPN [24, 22] trackers on several tracking benchmarks clearly demonstrate the generality and impressive performance of the proposed framework. The distilled trackers achieve compression rates of more than $13 \times -18 \times$ and a speedup of nearly $2 \times -3 \times$, respectively, while maintaining the same or even slightly improved tracking accuracy. The distilled SiamRPN also obtains a state-of-the-art performance (as shown in Fig. 1) at an extremely high speed of 265 FPS.

2. Related Work

Trackers with Siamese Networks: Tao et al. [36] utilized a Siamese network with both convolutional and fully-connected layers for training, and achieved favorable accuracy, while maintaining a low speed of 2 FPS. To improve the speed, Bertinetto et al. [4] proposed “SiamFC” by only applying an end-to-end Siamese network with 5 fully-convolutional layers for offline training. Because of its high speed at nearly 86 FPS on GPU, favorable accuracy, and simple mechanism for online tracking, there has been a surge of interest around SiamFC. Various improved methods are proposed [38, 17, 20, 52, 42, 18, 14, 43, 13, 45, 49, 24, 51]. For instance, Li et al. [24] proposed a SiamRPN tracker by combining the Siamese network and RPN [32], which directly obtains the location and scale of objects by regression, to avoid multiple forward computations for scale estimation in common Siamese trackers. Thus, it can run at 160 FPS with a better tracking accuracy. Subsequently, Zhu et al. [51] proposed distractor-aware training to use more datasets and applied distractor-aware incremental learning to improve online tracking. In the recent VOT-2018 [22], a variant of SiamRPN with a larger model size won the real-time challenge in the EAO metric and ranked 3rd in the main challenge.

Knowledge Distillation for Compression: In network compression, the goal of KD is to generate a student network that obtains better performance than the one trained directly by transferring knowledge from the teacher network. In an early work, Bucilua et al. [5] compressed key information into a single neural network from an ensemble of networks. Recently, Ba et al. [2] demonstrated an approach to improve the performance of shallow neural networks, by mimicking deep networks in training. Romero et al. [33] approximated the mappings between student and teacher hidden layers to compress networks by training the
relatively narrower students with linear projection layers. Subsequently, Hinton et al. [19] proposed a dark knowledge extracted from the teacher network by matching the full soft distribution between the student and teacher networks during training. Following this work, KD has attracted more interest in this community and a variety of methods have been applied to it [35, 37, 47, 8, 6, 50, 16]. For example, Zagoruyko et al. [47] employed an attention map to KD by training student network with matching the attention map of the teacher at the end of each residual stage. In most existing works concerned with KD, the architecture of the student network is usually manually designed. Net-to-Net (N2N) [11] method focuses on generating optimal reduced architecture for KD automatically. We use it to obtain the “dull” student network with reduced architecture. Then, we propose new KD [19] and attention transfer [47] methods to be adaptive for Siamese tracking network, and a novel teacher-students learning mechanism to further improve performance.

3. Revisiting SiamFC and SiamRPN

Since we adopt SiamFC [4] and SiamRPN [24] as the base trackers for our distilled tracking framework, we first revisit their basic network structures and training losses.

SiamFC adopts a two-stream fully convolutional network architecture, which takes target patches (denoted as z) and current search regions (denoted as x) as inputs. After a no-padding feature extraction network ϕ modified from AlexNet [23], a cross-correlation operation ⋆ is conducted on the two extracted feature maps:

\[ S = \varphi(x) \ast \varphi(z). \]  

The location of the target in the current frame is then inferred according to the peak value on the correlation response map S. The logistic loss, i.e., a usual binary classification loss, is used to train SiamFC:

\[ L_{\text{FC}}(x, z, y) = \frac{1}{|S|} \sum_{u \in S} \log(1 + e^{-(y[u] - S[u])}), \]  

where S[u] is a real-valued score in the response map S and y[u] ∈ {+1, −1} is a ground-truth label.

SiamRPN, as an extension of SiamFC, has a Siamese feature extraction subnetwork (same as SiamFC) and an additional RPN subnetwork [32]. After feature extraction, the features are fetched into the RPN subnetwork. The final outputs are foreground-background classification score maps and regression vectors of predefined anchors. By applying a single convolution and cross-correlation operations ⋆ in RPN on two feature maps, the outputs are obtained by:

\[ S_{\text{cls}}^{\text{conv1}} = \text{conv1}_{\text{cls}}[\varphi(x)] \ast \text{conv2}_{\text{cls}}[\varphi(z)]; \]

\[ S_{\text{reg}}^{\text{conv1}} = \text{conv1}_{\text{reg}}[\varphi(x)] \ast \text{conv2}_{\text{reg}}[\varphi(z)], \]

where k is the predefined anchor number. The template feature maps conv2_{cls} [ϕ(z)] and conv2_{reg} [ϕ(z)] are then used as kernels in the cross-correlation operation to obtain the final classification and regression outputs, with size w × h.

During training, the following multi-task loss function is optimized:

\[ L_{\text{RPN}} = L_{\text{cls}}^RPN(S_{\text{cls}}^{\text{RPN}}, G_{\text{cls}}) + L_{\text{reg}}^RPN(S_{\text{reg}}^{\text{RPN}}, G_{\text{reg}}), \]

where G_{cls} and G_{reg} are the ground-truths of the classification and regression outputs, L_{cls}^RPN is a cross-entropy loss for classification, and \( L_{\text{reg}}^RPN \) is a smooth \( L_1 \) loss with normalized coordinates for regression.

4. Distilled Siamese Trackers

In this section, we detail the proposed framework of Distilled Siamese Trackers (DST) for high-performance tracking. As shown in Fig. 2, the proposed framework consists of two essential stages. First, in [4.1] for a given teacher network, such as SiamRPN, we obtain a “dull” student with a reduced network architecture via Deep Reinforcement Learning (DRL). Second, the “dull” student network is further trained simultaneously with an “intelligent” student via the proposed distillation model facilitated by a teacher-students learning mechanism (see [4.2]).

4.1. “Dull” Student Selection

Inspired by N2N [11] for compressing classification networks, we transfer selecting a student tracker with reduced network architecture to learning an agent with optimal compression strategy (policy) by DRL. Unlike N2N, we only conduct layer shrinkage because of Siamese trackers’ shallow network architecture. Layer removal will cause a sharp decline in accuracy and divergence of the policy network.

In our task, the agent for selecting a small and reasonable network is learned from a sequential decision-making process by policy gradient DRL. The whole decision process can be modeled as a Markov Decision Process (MDP), which is defined as the tuple \( M = (S, A, T, r, \gamma) \). The state space S is a set of all possible reduced network architectures derived from the teacher network. A is the set of all actions to transform one network into another compressed one. Here, we use layer shrinkage \( A_t \) actions \( a_t \in [0, 1, 0.2, \cdots, 1] \) by changing the configurations of each layer, such as kernel size, padding, and number of output filters. \( T : S \times A \rightarrow S \) is the state transition function. \( \gamma \) is the discount factor in MDP. To maintain the equal contribution for each reward, we set \( \gamma \) to 1. r is the reward function. The reward of final state in [11] achieves a balance between tracking accuracy and compression rate, which is defined as follows:

\[ R = C(2 - C) \cdot \frac{\text{acc}_s}{\text{acc}_t}, \]

where \( C = 1 - \frac{S_s}{S_t} \) is the relative compression rate of a student network with size \( S_s \) compared to a teacher with size \( S_t \), \( \text{acc}_s \) and \( \text{acc}_t \) are the validation accuracies of the student and teacher networks. We propose to define a new metric of
accuracy for tracking by selecting the top-\(N\) proposals with the highest confidence and calculating their overlaps with the ground-truth boxes for \(M\) image pairs in validation set:

\[
\text{acc} = \sum_{i=1}^{M} \sum_{j=1}^{N} o(g_i, p_{ij}),
\]

where \(p_{ij}(j \in [1, 2, \ldots, N])\) denotes the \(j\)-th proposal of the \(i\)-th image pair, \(g_i\) is the corresponding ground-truth, and \(o\) is the overlap function. At each step, the policy network outputs \(N_a\) actions and the reward is defined as the average reward of generated students:

\[
R_t = \frac{1}{N_a} \sum_{i=1}^{N_a} R_{t_i}.
\]

Given a policy network \(\theta\) and the predefined MDP, we use the REINFORCE method [44] to optimize the policy and finally obtain the optimal policy \(\pi_0: S \rightarrow A\) and reduced student network. All the training processes in this section are based on a small dataset selected from the whole dataset, considering the time cost.

### 4.2. Teacher-Student Knowledge Distillation

After the network selection, we obtain a “dull” student network with poor comprehension due to small model size. To pursue more intensive knowledge distillation and promising tracking performance, we propose a Teacher-Students Knowledge Distillation (TSsKD) model. It encourages teacher-student knowledge transfer as well as mutual learning between students that serves as more flexible and appropriate guidance. In [4.2.1], we elaborate the teacher-student knowledge transfer (distillation) model. Then, in [4.2.2] we describe the student-student knowledge sharing strategy. Finally, in [4.2.3] we provide a theoretical analysis to prove the effectiveness of our TSsKD model.

#### 4.2.1 Teacher-Student Knowledge Transfer

In the teacher-student knowledge transfer model, we propose a novel transfer loss to capture the knowledge in teacher networks. It contains three components: Teacher Soft (TS) loss, Adaptive Hard (AH) loss, and Siamese Attention Transfer (SAT) loss. The first two allow the student to mimic the outputs of the teacher network, such as the logits [19] in the classification model. These two losses can be seen as a variant of KD methods [19] [6], which are used to extract dark knowledge from teacher networks. The last loss is for the middle feature maps and leads students to pay attention to the same regions of interest as the teacher. This provides middle-level semantic hints to the student. Our knowledge transfer loss includes both classification and regression parts and can be incorporated into other networks by deleting the corresponding part.

**Teacher Soft (TS) Loss**: We set \(C_s\) and \(B_s\) as the student’s classification and bounding box regression outputs, respectively. In order to incorporate the dark knowledge that regularizes students by placing emphasis on the relationships learned by the teacher network across all the outputs, we need to ‘soften’ the output of classification. We set \(P_t = \text{softmax}(C_t/\text{temp})\), where temp is a temperature parameter to obtain a soft distribution [19]. Similarly, \(P_s = \text{softmax}(C_s/\text{temp})\). Then, we give the TS loss for knowledge distillation as follows:

\[
L_{\text{TS}} = L_{\text{TS}}(P_s, P_t) + L_{\text{reg}}(B_s, B_t),
\]

where \(L_{\text{TS}} = \text{KL}(P_s, P_t)\) is a Kullback Leibler (KL) divergence loss on soft outputs of the teacher and student. \(L_{\text{reg}}\) is the original regression loss of the tracking network.
Adaptive Hard (AH) Loss: To make full use of the ground truth $G$, we combine the outputs of the teacher network with the original hard loss of the student network. For regression loss, we employ a modified teacher bounded regression loss \[^6\] , which is defined as:

$$L_{\text{reg}}^\text{AH}(B_s, B_t, G_{\text{reg}}) = \begin{cases} L_r(B_s, g), & \text{if } \text{gap} < m, \\ 0, & \text{otherwise}. \end{cases} \quad (10)$$

where $\text{gap} = L_r(B_t, G_{\text{reg}}) - L_r(B_s, G_{\text{reg}})$ is the gap between the student’s and the teacher’s loss (here it’s $L_r$, the regression loss of the tracking network) with the ground-truth. $m$ is a margin.

This loss will keep the student regression vector close to the ground-truth when its quality is worse than the teacher. However, once it outperforms the teacher network, we stop offering loss for the student to avoid over-fitting. Added with the student’s original classification loss, our AH loss is defined as follows:

$$L^\text{AH} = L^\text{AH}_{\text{cls}}(C_s, C_t) + L_{\text{reg}}^\text{AH}(B_s, B_t, G_{\text{reg}}). \quad (11)$$

Siamese Attention Transfer (SAT) Loss: To lead a student to concentrate on the same regions of interest a teacher during tracking, we introduce attention transfer \[^47\] into our framework. Based on the assumption that the activation of a hidden neuron can indicate its importance for a specific input, we can transfer the semantic features of a teacher onto a student by forcing it to mimic the teacher’s attention map. We use activation-based maps calculated by a mapping function $F: \mathbb{R}^{C \times H \times W} \rightarrow \mathbb{R}^{H \times W}$, which outputs 2D activation map with 3D feature maps provided. Here we use:

$$F(U) = \sum_{i=1}^{C} |U_i|, \quad (12)$$

where $U_i \in \mathbb{R}^{H \times W}$ is a spacial feature map of the $i$th channel, and $| \cdot |$ represents the absolute values of a matrix.

Siamese network has two weight-sharing branches with different inputs: a target patch and a larger search region. The feature network needs to learn the attention in the feature maps of both branches. It has been found that the surrounding noise in the search region’s feature will disturb the attention transfer of the target’s features due to the existence of distractors. To solve the problem, we set a weight on the search region’s feature maps. Then, we define the following multi-layer Siamese AT loss:

$$L^\text{SAT} = L^\text{SAT}_r + L^\text{SAT}_s, \quad (13)$$

$$L^\text{SAT}_r = \sum_{j \in \tau} ||F(W^r_T Q^j_s) - F(W^r_s Q^j_T)||_2, \quad (14)$$

$$L^\text{SAT}_s = \sum_{j \in \tau} ||F(Q^j_s) - F(Q^j_T)||_2, \quad (15)$$

where $\tau$ is the set of attention transfer layers’ indices. $Q^j_T$ and $Q^j_s$ denote the teacher’s feature map of layer $j$ on the search and target branch, respectively. $W^r_T = Q^j_T \ast Q^j_T$ is the weight on the teachers’ $j$th feature map. Student variables are defined in the same way. By introducing this weight, which rearranges the importance of different patches in the search region according to their similarities with the target, more attention is paid to the target. This keeps the attention maps of the two branches consistent and enhances the effect of attention transfer. An example of our multi-layer Siamese attention transfer is shown in Fig. 3. The comparison of activation maps with and without weights shows that the surrounding noise is suppressed effectively.

By combining the above three types of losses, the overall loss for transferring knowledge from a teacher to a student is defined as follows:

$$L^\text{KT} = L^\text{TS} + \lambda L^\text{AH} + \omega L^\text{SAT}. \quad (16)$$

4.2.2 Student-Student Knowledge Sharing

Based on our teacher-student distillation model, we propose a student-student knowledge sharing mechanism to further narrow the gap between the teacher and the “dull” student. As an “intelligent” student with a larger model size usually learns and performs better (due to its better comprehension), sharing its knowledge is able to inspire the “dull” one to develop a more in-depth understanding. On the other side, the “dull” one can do better in some cases and provide some helpful knowledge too.

We take two students as an example and denote them as a “dull” student $s_1$ and an “intelligent” student $s_2$. For a proposal $d_i$ in Siamese trackers, assume that the predicted probabilities of being target by $s_1$ and $s_2$ are $p_1(d_i)$ and $p_2(d_i)$, respectively. The predicted bounding-box regression values are $r_1(d_i)$ and $r_2(d_i)$. To improve the learning effect of $s_1$, we obtain the knowledge shared from $s_2$ by using its prediction as prior knowledge. The KL Divergence is used to quantify the consistency of $N$ proposals’ classification probabilities:

$$L^\text{KS}_{\text{cls}}(s_1 || s_2) = \sum_{i=1}^{N} \left( p_1(d_i) \log \frac{p_1(d_i)}{q_1(d_i)} + q_1(d_i) \log \frac{q_1(d_i)}{p_1(d_i)} \right), \quad (17)$$
where \( q_1(d_i) \) and \( q_2(d_i) \) are probabilities of background. For regression, we use smooth \( L_1 \) loss:

\[
L_{\text{reg}}^{KS}(s1||s2) = \sum_{i=1}^{N} L_1(r_1(d_i) - r_2(d_i)).
\]

The knowledge sharing loss for \( s1 \) can be defined as:

\[
L^{KS}(s1||s2) = L_{\text{ch}}^{KS}(s1||s2) + L_{\text{reg}}^{KS}(s1||s2).
\]

Combined with the knowledge transfer loss, our final objective functions for \( s1 \) and \( s2 \) are as follows:

\[
L_{s1}^{KD} = L_{s1}^{KT} + \sigma(s1) L^{KS}(s1||s2),
\]

where \( \beta \) is a discount factor on account of the two students’ different reliability, and \( \sigma \) denotes the weight of knowledge sharing. Considering the “dull” student’s worse performance, we set \( \beta \in (0, 1) \). To filter the reliable knowledge edge for \( s1 \), we set a condition on \( \sigma \):

\[
\sigma(s1) = \begin{cases} f(e) & \text{if } L^{GT}(s1) - L^{GT}(t) < h, \\ 0 & \text{otherwise}. \end{cases}
\]

Here, \( L^{GT}(s1) \), \( L^{GT}(t) \) are the losses for \( s1 \) and teacher, with ground-truth. \( h \) is their gap constraint. \( f(e) \) is a function decreasing geometrically with current epoch \( e \). For \( s2 \), a similar condition exists.

To train the two students simultaneously, the final loss for our TSsKD is:

\[
L^{KD} = L_{s1}^{KD} + L_{s2}^{KD}.
\]

### 4.2.3 Why Does TSsKD Work?

According to the VC theory \([39, 27]\), the learning process can be regarded as a statistical procedure. Given \( n \) data, a student function \( f_{st} \) belonging to a function class \( \mathcal{F}_{st} \), and a real (ground-truth) target function \( f \in \mathcal{F} \), the task (classification or regression) error of learning from scratch without KD (NOKD) can be decomposed as:

\[
\mathcal{R}(f_{st}) - \mathcal{R}(f_{st}) \leq O \left( \frac{|\mathcal{F}||\mathcal{C}|}{n^{\alpha_{st}}} \right) + \epsilon_{st},
\]

where the \( \mathcal{R}(\cdot) \), \( O(\cdot) \) and \( \epsilon_{st} \) are the expected, estimation and approximation error, respectively. \( |\cdot|_{C} \) presents an appropriate function class measure. \( \alpha_{st} \in (0, 1) \) is the learning rate measuring the difficulty of a learning problem i.e. small values present difficult cases while large values indicate easy problems. Setting \( f_{t} \in \mathcal{F} \) as the teacher function, the error of \( f_{s} \) with KD is defined as:

\[
\mathcal{R}(f_{s}) - \mathcal{R}(f_{s}) \leq O \left( \frac{|\mathcal{F}||\mathcal{C}|}{n^{\alpha_{s}}} \right) + \epsilon_{sr} + O \left( \frac{|\mathcal{F}||\mathcal{C}|}{n^{\alpha_{st}}} \right) + \epsilon_{tr},
\]

where \((25) \leq (26)\) because \( \alpha_{st} \leq \alpha_{sr} \), which is an assumption in \([27]\). In addition, Lopez et al. \([27]\) made more reasonable assumptions for \((26) \leq (24)\) to prove that KD outperforms NOKD. For instance, \( |\mathcal{F}_{i}|_{C} \) is small, \( \alpha_{sr} < \alpha_{st} \), and \( \epsilon_{sr} \gg \epsilon_{st} + \epsilon_{tr} \).

To analyze our TSsKD model, we first focus on the “dull” student denoted as \( f_{s'} \). Assume \( f_{s'} \) also belongs to \( \mathcal{F}_{s} \) (the same network as \( f_{s} \)) but is selected (trained) by different methods from \( f_{s} \). Then, we can obtain the error upper bound of \( f_{s'} \) with our TSsKD:

\[
O \left( \frac{|\mathcal{F}_{s}||\mathcal{C}| + |\mathcal{F}_{i}||\mathcal{C}|}{n^{\alpha_{s}}} \right) + \epsilon_{s'tr} + \epsilon_{tr}.
\]

To prove that our TSsKD outperforms KD, \((28) \leq (26)\) should hold. Thus, we also make two reasonable assumptions: \( \alpha_{s't} \geq \alpha_{st} \), and \( \epsilon_{s'tr} \leq \epsilon_{st} \). Recalling Eq. \((20)\), the objective function of one student, we can find that the first item is a KD loss offering the same information, while the second item provides additional information. We use a condition function \( \sigma \) to filter out noisy information to capture reliable information. We believe that the \( \alpha_{s't} \geq \alpha_{st} \) is a general situation since more reliable information should allow for faster learning. In addition, more information also enhances the generalization of the network to decrease the approximation error, i.e. \( \epsilon_{s't} \leq \epsilon_{st} \). The above analysis is also suitable for other students since even the “dull” student can do better in some cases. Thus, our TSsKD can improve the performance of all students.

5. Experiments

To demonstrate the effectiveness of our framework, we employ it on two representative Siamese trackers: SiamFC \([4]\) and SiamRPN \([24]\) (VOT version). We evaluate the distilled trackers on several benchmarks, including OTB-100 \([40]\), DTB \([25]\), VOT-2016 \([21]\), LaSOT \([15]\), and TrackingNet \([28]\), and give an ablation study. All the experiments are implemented using PyTorch with an Intel i7 CPU and an Nvidia GTX 1080ti.

5.1. Implementation Details

N2N Setting. In the “dull” student selection experiment, an LSTM is employed as the policy network. A small representative dataset (about 10,000 image pairs) is created by
selecting images uniformly for several classes in the whole dataset of the corresponding tracker. The policy network is updated for 50 steps. In each step, three reduced networks are generated and trained from scratch for 10 epochs on the small dataset. We observe heuristically that this is sufficient to compare performance. Both SiamRPN and SiamFC use the same settings.

**Training Datasets.** For SiamRPN, same with teacher in [22], we pre-process four datasets: ImageNet VID [34], YouTube-BoundingBoxes [31], COCO [26] and ImageNet Detection [34], to generate about two million image pairs with 127 pixels for target patches and 271 pixels for search regions. SiamFC is trained on ImageNet VID [34] with 127 pixels and 255 pixels for two inputs, respectively, which is consistent with [4].

**Optimization.** During the teacher-students knowledge distillation, “intelligent” students are generated by halving the convolutional channels of both teachers. SiamRPN’s student networks are warm-up trained by exploring the ground-truth for 10 epochs and then trained for 50 epochs with the learning rate exponentially decreasing from $10^{-2}$ to $10^{-4}$. Same with the teacher, SiamFC’s student networks are trained for 30 epochs with a learning rate of $10^{-2}$. All the losses used in the experiments are reported in Table 1.

| $L_1$ | $L_2$ | $L_3$ | $L_4$ |
|-------|-------|-------|-------|
| $L_1^m$ | logistic | cross-entropy+bounded | KL |
| $L_2^1$ | MSE | $KL + L_1$ | MSE |
| $L_3^1$ | KL | $KL + L_1$ | $KL + L_1$ |

Table 1. Losses used in the knowledge transfer stage. MSE, $L_1$ and KL represent Mean-Square-Error loss, smooth $L_1$ loss and Kullback-Leibler Divergence loss, respectively.

results show that our method is able to generate an optimal architecture regardless of the teacher’s complexity. Finally, two reduced models of size 19.7 MB and 0.7 MB for SiamRPN (361.8 MB) and SiamFC (9.4 MB) are generated.

5.3. Benchmark Results for Visual Object Tracking

**Results on OTB-100.** On the OTB-100 benchmark [46], we compare our DSTrpn (SiamRPN as teacher) and DSTfc (SiamFC as teacher) trackers with various recent fast trackers (more than 50 FPS), including the teacher networks SiamRPN [24] and SiamFC [4], Siam-tri [13]. TRACA [27], HP [13], Cfnnet2 [38], and fDSST [10]. The evaluation metrics include both precision and success plots in one pass evaluation (OPE) [46], where ranks are sorted using precision scores with center error less than 20 pixels and Area-Under-the-Curve (AUC), respectively.

In Fig. 5 our DSTrpn outperforms all the other trackers in terms of precision and success plots. We also report the running speed of all trackers for comparison. Notice that our DSTrpn runs at an extremely high speed of 265 FPS, which is nearly $3 \times$ faster than the teacher network SiamRPN (90 FPS) and obtains the same (even slightly better) precision and AUC scores. Our DSTfc runs more than $2 \times$ faster than SiamFC with comparable performance.

**Results on DTB.** We benchmark our method on the Drone Tracking Benchmark (DTB) [25] including 70 videos captured by drone cameras. We compare SiamRPN, recent Siamese works such as HP [13], and the trackers evaluated in DTB, including DAT [30], HOGLR [41], SODLT [40], DSST [9], MDNet [29], MEEM [48], and SRDCF [11]. The evaluation metrics include Distance Precision (DP) at a threshold of 20 pixels, Overlap Precision (OP) at an overlap threshold of 0.5, and the Area-Under-the-Curve (AUC).

As shown in Table 2 DSTrpn achieves the best performance in terms of DP and OP. For AUC, DSTrpn ranks first (0.5557) and significantly outperforms SiamFC (0.4797). Compared with the teacher SiamRPN, our student network surpasses it in terms of both AUC and OP, while even achieving a 3.6% improvement on DP, considering that our model size is just 1/18 of the teacher SiamRPN. DSTfc outperforms SiamFC in terms of all three criteria too.

Figure 5. Precision and success plots for all trackers on the OTB-100 benchmark [46].

As shown in Table 2 DSTrpn achieves the best performance in terms of DP and OP. For AUC, DSTrpn ranks first (0.5557) and significantly outperforms SiamFC (0.4797). Compared with the teacher SiamRPN, our student network surpasses it in terms of both AUC and OP, while even achieving a 3.6% improvement on DP, considering that our model size is just 1/18 of the teacher SiamRPN. DSTfc outperforms SiamFC in terms of all three criteria too.
Results on VOT-2016. The benchmark of VOT-2016 [21] differs from OTB-100, a tracker is evaluated using a reset-based mechanism. Whenever a tracker loses the object after five frames, it will be re-initialized. The major evaluation metric is the Expected Average Overlap (EAO) which combines accuracy and robustness. We evaluate the distilled trackers and various SOTA trackers, including CCOT [12], Staple [3], and other trackers in VOT-2016. Fig. 6 shows that our tracker achieves the second highest EAO among all trackers. Notice that our model is much smaller than the first rank SiamRPN while only losing a few EAO (0.023).

Results on LaSOT and TrackingNet. We also did extensive experiments on large-scale datasets such as LaSOT [15] and TrackingNet [28] to evaluate the generalization of our method. We compare DaSiamRPN [51], MDNet [29], and our baselines: SiamRPN [22] and SiamFC [4]. As shown in Table 3, the model size of our DSTrpn (or DSTfc) is further smaller than its teacher model SiamRPN (or SiamFC), while the AUC scores on two large-scale datasets are very close to the teacher model. Notice that, our DSTrpn achieves better performance than DaSiamRPN and MDNet in both two datasets.

5.4. Ablation Study

Knowledge Transfer Components. The teacher-student knowledge transfer consists of three major components: (i) Adaptive Hard (AH) loss, (ii) Teacher Soft (TS) loss, and (iii) Siamese Attention Transfer (SAT) loss. We conduct an extensive ablation study by implementing a number of variants using different combinations, including (1) GT: simply using the ground-truth without any of the three losses, (2) TS, (3) GT+TS, (4) AH+TS, (5) TS+SAT, (6) GT+TS+SAT, and (7) AH+TS+SAT (the full knowledge transfer method). Table 4 shows our results on SiamFC and SiamRPN.

For SiamRPN, we can see that the GT without any proposed loss degrades dramatically compared with the teacher network, due to the large model size gap. When using the Teacher Soft (TS) loss to distill knowledge from the teacher network, we observe a significant improvement in terms of precision (15.8%) and AUC (15.7%). However, directly combining GT and TS (GT+TS) could be suboptimal due to over-fitting. By replacing GT with AH, AH+TS further boosts the performance for two metrics. Finally, by adding the Siamese Attention Transfer (SAT) loss, the model (AH+TS+SAT) is able to close the gap between the teacher and student networks, outperforming other variants (TS+SAT or GT+TS+SAT). SiamFC only employs classification loss, so GT is equal to AH and we use GT here. Results show that the gaps are narrower than SiamRPN but performance improvements can still be seen. The results clearly demonstrate the effectiveness of each component.

Different Learning Mechanisms. To evaluate our TSsKD model, we also conduct an ablation study on different learning mechanisms: (i) NOKD: train from scratch, (ii) TSKD: our tracking-specific teacher-student knowledge distillation (transfer) and (iii) TSsKD. “Student1” and “Student2” represent “dull” and “intelligent” student, respectively. Students are trained following different paradigms and results on SiamRPN and SiamFC can be seen in Table 5.

| Experiment | GT | AH | TS | SAT | Precision | AUC |
|------------|----|----|----|-----|-----------|-----|
| SiamRPN    | ✓  | ✓  | ✓  | ✓   | 0.638     | 0.429|
|            | ✓  | ✓  | ✓  | ✓   | 0.796     | 0.586|
|            | ✓  | ✓  | ✓  | ✓   | 0.795     | 0.579|
|            | ✓  | ✓  | ✓  | ✓   | 0.800     | 0.591|
|            | ✓  | ✓  | ✓  | ✓   | 0.811     | 0.608|
|            | ✓  | ✓  | ✓  | ✓   | 0.812     | 0.606|
|            | ✓  | ✓  | ✓  | ✓   | 0.825     | 0.624|
| SiamFC     | ✓  | ✓  | ✓  | ✓   | 0.707     | 0.523|
|            | ✓  | ✓  | ✓  | ✓   | 0.711     | 0.535|
|            | ✓  | ✓  | ✓  | ✓   | 0.710     | 0.531|
|            | ✓  | ✓  | ✓  | ✓   | 0.742     | 0.548|
|            | ✓  | ✓  | ✓  | ✓   | 0.741     | 0.557|

Table 4. Ablation study: results for different combinations of GT, TS, AH and SAT in terms of precision and AUC on OTB-100 [46].
knowledge distillation, all students are improved. Moreover, with the knowledge sharing in our TSsKD, the “dull” SiamRPN student gets a performance improvement of 2.2% in terms of AUC. The “dull” SiamFC student gets a 1.6% improvement. On the other side, the “intelligent” SiamRPN and SiamFC students get slight improvements (0.3%) as well. Fusing the knowledge from teacher, ground-truth and “intelligent” student, the “dull” SiamRPN student obtains the best performance.

Loss comparison. We also compare the losses of different student networks in our experiments in the paper. As shown in Fig. 7, the “intelligent” students (denoted as Student1) have a lower loss than the “dull” ones (denoted as Student2) in the whole training and validation process, and maintain a better understanding of the training dataset. They provide additional reliable knowledge to the “dull” students which further inspires more intensive knowledge distillation and better tracking performance.

6. Conclusion

This paper proposed a new framework of Distilled Siamese Trackers (DST) to learn small, fast yet accuracy trackers from larger Siamese Trackers. This framework is built upon a teacher-students knowledge distillation model including two kinds of knowledge transfer styles: 1) knowledge transfer from teacher to students by a tracking-specific distillation strategy; 2) mutual learning between students in a knowledge sharing manner. The theoretical analysis and extensive empirical evaluations on two Siamese trackers have clearly demonstrated the generality and effectiveness of the proposed DST. Specifically, for the state-of-the-art (SOTA) SiamRPN, the distilled tracker also achieved a high compression rate, ran at an extremely high speed, and obtained similar performance as the teacher. Thus, we believe such a distillation method can be used for improving many SOTA deep trackers towards practical tracking tasks.

7. Appendix

7.1. Details of DRL

In the “dull” student selection stage, we use a policy gradient algorithm to optimize our policy network step by step. With the parameters of the policy network denoted as $\theta$, our objective function is the expected reward over all the action sequences $a_{1:T}$:

$$J(\theta) = E_{a_{1:T} \sim \rho_\theta}(R).$$  

(29)

To calculate the gradient of our policy network, we use REINFORCE in our experiment. Given the hidden state $h_t$, the gradient is formulated as:

$$\nabla_{\theta} J(\theta) = \nabla_{\theta} E_{a_{1:T} \sim \rho_\theta}(R) = \sum_{t=1}^{T} E_{a_{1:T} \sim \rho_\theta} [\nabla_{\theta} \log \rho_\theta(a_t|a_{1:(t-1)}) R_t]$$  

(30)

$$\approx \sum_{t=1}^{T} [\nabla_{\theta} \log \rho_\theta(a_t|h_t)] \frac{1}{N_a} \sum_{s=1}^{N_a} R_{ts},$$

where $\rho_\theta(a_t|h_t)$ is the probability of actions controlled by the current policy network with hidden state $h_t$. $R_{ts}$ is the reward of the current $k$-th student model at step $t$. Furthermore, in order to reduce the high variance of estimated gradients, a state-independent baseline $b$ is introduced:

$$b = \frac{1}{N_a \cdot T} \sum_{t=1}^{T} \sum_{s=1}^{N_a} R_{ts}. $$  

(31)

It denotes an exponential moving average of previous rewards. Finally, our policy gradient is calculated as:

$$\nabla_{\theta} J(\theta) \approx \sum_{t=1}^{T} [\nabla_{\theta} \log \rho_\theta(a_t|h_t)] \frac{1}{N_a} \sum_{s=1}^{N_a} R_{ts} - b].$$  

(32)

7.2. Extension to More Students

Our TSsKD model can be naturally extended to more students. Given $n$ students $s_1, s_2, ..., s_n$, the objective function for $s_i$ is as follows:

$$L_{s_i}^{KD} = L_{s_i}^{KT} + \frac{1}{n} \sum_{j=1}^{n} b_{ij} \sigma(s_i L_{s_j}^{KS} | s_i sj).$$  

(33)
Here $\beta_{ij}$ is the discount factor between si and sj considering their different reliability. For example, in our case with two students in the paper, $\beta_{12} = 1$ and $\beta_{21} = 0.5$. We conduct an experiment on different student numbers and obtain a result reported in Fig. 8. Students are generated by reducing the number of convolutional channels to a scale (0.4, 0.45, 0.5, 0.55). In our case, since our “dull” students achieve performance close to the teacher with one “intelligent” student, more students don’t bring significant improvements.

**References**

[1] A. Ashok, N. Rhinehart, F. Beainy, and K. M. Kitani. N2n learning: network to network compression via policy gradient reinforcement learning. In ICLR, 2018.

[2] J. Ba and R. Caruana. Do deep nets really need to be deep? In NIPS, 2014.

[3] L. Bertinetto, J. Valmadre, S. Golodetz, O. Miksik, and P. H. Torr. Staple: Complementary learners for real-time tracking. In CVPR, 2016.

[4] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. Torr. Fully-convolutional siamese networks for object tracking. In ECCV Workshop, 2016.

[5] C. Bucilu, R. Caruana, and A. Niculescu-Mizil. Model compression. In SIGKDD, 2006.

[6] G. Chen, W. Choi, X. Yu, T. Han, and M. Chandraker. Learning efficient object detection models with knowledge distillation. In NIPS, 2017.

[7] J. Choi, H. J. Chang, T. Fischer, S. Yun, K. Lee, J. Jeong, Y. Demirir, and J. Y. Choi. Context-aware deep feature compression for high-speed visual tracking. In CVPR, 2018.

[8] W. M. Czarnecki, S. Osindero, M. Jaderberg, G. Swirszcz, and R. Pascanu. Sobolev training for neural networks. In NIPS, 2017.

[9] M. Danelljan, G. Häger, F. Khan, and M. Felsberg. Accurate scale estimation for robust visual tracking. In BMVC, 2014.

[10] M. Danelljan, G. Häger, F. S. Khan, and M. Felsberg. Discriminative scale space tracking. IEEE TPAMI, 39(8):1561–1575, 2017.

[11] M. Danelljan, G. Hager, F. Shahbaz Khan, and M. Felsberg. Learning spatially regularized correlation filters for visual tracking. In ICCV, 2015.

[12] M. Danelljan, A. Robinson, F. Shahbaz Khan, and M. Felsberg. Beyond correlation filters: Learning continuous convolution operators for visual tracking. In ECCV, 2016.

[13] X. Dong and J. Shen. Triplet loss in siamese network for object tracking. In ECCV, 2018.

[14] X. Dong, J. Shen, W. Wang, Y. Liu, L. Shao, and F. Porikli. Hyperparameter optimization for tracking with continuous deep q-learning. In CVPR, 2018.

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

[16] T. Furlanello, Z. C. Lipton, M. Tschannen, L. Itti, and A. Anandkumar. Born again neural networks. In ICML, 2018.

[17] Q. Guo, W. Feng, C. Zhou, R. Huang, L. Wan, and S. Wang. Learning dynamic siamese network for visual object tracking. In ICCV, 2017.

[18] A. He, C. Luo, X. Tian, and W. Zeng. A twofold siamese network for real-time object tracking. In CVPR, 2018.

[19] G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. In NIPS Workshop, 2014.

[20] C. Huang, S. Lucey, and D. Ramanan. Learning policies for adaptive tracking with deep feature cascades. In ICCV, 2017.

[21] M. Kristan, A. Leonardis, J. Matas, M. Felsberg, R. Pfugfelder, L. Čehovin, T. Vojir, G. Häger, A. Lukežič, and G. Fernandez. The visual object tracking vot2016 challenge results. In ECCV workshop, 2016.

[22] M. Kristan, A. Leonardis, J. Matas, M. Felsberg, R. Pfugfelder, L. C. Zajc, T. Vojir, G. Bhat, A. Lukezic, A. Eldesokey, G. Fernandez, and et al. The sixth visual object tracking vot2018 challenge results. In ECCV workshop, 2018.

[23] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In NIPS, 2012.
B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu. High performance visual tracking with siamese region proposal network. In CVPR, 2018.

S. Li and D.-Y. Yeung. Visual object tracking for unmanned aerial vehicles: A benchmark and new motion models. In AAAI, 2017.

T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In ECCV, 2014.

D. Lopez-Paz, L. Bottou, B. Scholkopf, and V. Vapnik. Unifying distillation and privileged information. In ICLR, 2016.

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

H. Nam and B. Han. Learning multi-domain convolutional neural networks for visual tracking. In CVPR, 2016.

H. Possegger, T. Mauthner, and H. Bischof. In defense of color-based model-free tracking. In CVPR, 2015.

E. Real, J. Shlens, S. Mazzocchi, X. Pan, and V. Vanhoucke. Youtube-boundingboxes: A large high-precision human-annotated data set for object detection in video. In CVPR, 2017.

S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In NIPS, 2015.

A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio. Fitnets: Hints for thin deep nets. arXiv preprint arXiv:1412.6550, 2014.

O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. IJCV, 115(3):211–252, 2015.

P. Sadowski, J. Collado, D. Whiteson, and P. Baldi. Deep learning, dark knowledge, and dark matter. In NIPS Workshop, 2015.

R. Tao, E. Gavves, and A. W. Smeulders. Siamese instance search for tracking. In CVPR, 2016.

G. Urban, K. J. Geras, S. E. Kahou, O. Aslan, S. Wang, R. Caruana, A. Mohamed, M. Philipose, and M. Richardson. Do deep convolutional nets really need to be deep and convolutional? In ICLR, 2017.

J. Valmadre, L. Bertinetto, J. F. Henriques, A. Vedaldi, and P. H. Torr. End-to-end representation learning for correlation filter based tracking. In CVPR, 2017.

V. Vapnik. Statistical learning theory, 1998.