# Learn to Match: Automatic Matching Network Design for Visual Tracking

Zhipeng Zhang[1,2], Yihao Liu[2], Xiao Wang[3], Bing Li[1,2,†], and Weiming Hu[1,2,4,†]
[1] National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences
[2] School of AI, University of Chinese Academy of Sciences [3] Peng Cheng Laboratory
[4] CAS Center for Excellence in Brain Science and Intelligence Technology

## Abstract

*Siamese tracking has achieved groundbreaking performance in recent years, where the essence is the efficient matching operator cross-correlation and its variants. Besides the remarkable success, it is important to note that the heuristic matching network design relies heavily on expert experience. Moreover, we experimentally find that one sole matching operator is difficult to guarantee stable tracking in all challenging environments. Thus, in this work, we introduce six novel matching operators* ***from the perspective of feature fusion instead of explicit similarity learning****, namely Concatenation, Pointwise-Addition, Pairwise-Relation, FiLM, Simple-Transformer and Transductive-Guidance, to explore more feasibility on matching operator selection. The analyses reveal these operators' selective adaptability on different environment degradation types, which inspires us to combine them to explore complementary features. To this end, we propose binary channel manipulation (BCM) to search for the optimal combination of these operators. BCM determines to retrain or discard one operator by learning its contribution to other tracking steps. By inserting the learned matching networks to a strong baseline tracker Ocean [47], our model achieves favorable gains by* $67.2 \rightarrow 71.4$, $52.6 \rightarrow 58.3$, $70.3 \rightarrow 76.0$ *success on OTB100, LaSOT, and TrackingNet, respectively. Notably, Our tracker, dubbed* ***AutoMatch****, uses less than half of training data/time than the baseline tracker, and runs at 50 FPS using PyTorch. Code and model are released at https://github.com/JudasDie/SOTS.*

## 1. Introduction

Generic object tracking, aiming to infer the location and scale of an arbitrary object in a video sequence, is one of the fundamental problems in computer vision [16, 21, 25, 33]. The recent prevailing Siamese methods [3, 6, 11, 18, 41, 42, 47], decompose the tracking problem into a *relation learning* task and a *state estimation* task. In the former case, the goal is to measure the similarity between exemplar and candidate (search) images. The second task, which is normally comprised of foreground classification and scale regression [8, 18, 47], is followed to estimate the target state.

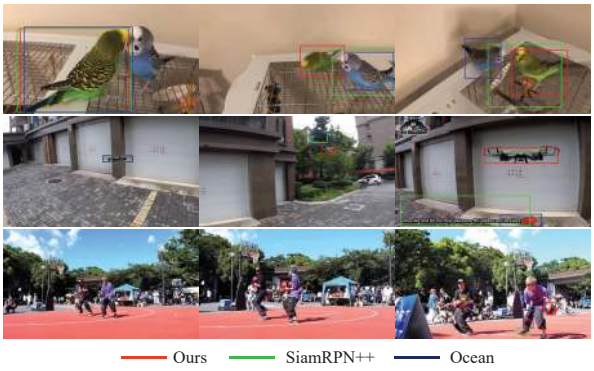



Figure 1: Comparisons of our approach with depthwise cross-correlation based trackers SiamRPN++ [18] and Ocean [47]. Our model, employing the automatically searched matching networks, can better handle different challenging factors, *e.g.,* distractor in the first video, occlusion and scale change of the second one, background clutter and fast motion of the third sequence.

Fuelled by the emergence of object detection that facilitates bounding box regression, the network design for state estimation has substantially advanced in recent years [6, 8, 19, 41, 47]. However, the advancements in relation learning have been limited. Previous works generally perform relation learning with heuristically designed matching operators. Concretely, the seminal work SiamFC [3] employs cross-correlation to model the relation between exemplar and candidate images. The follow-ups propose upchannel cross-correlation [19] and depthwise cross-correlation [18] to learn fine-grained feature similarities. Besides their great success, it is important to note that the heuristic

Email: zhangzhipeng2017@ia.ac.cn † Corresponding authors.

matching network design requires substantial effort of human experts, and it is extremely difficult to guarantee robustness in all challenging environments, as experimentally verified in Fig. 1 and Tab. 1. One straightforward solution is to find the optimal matching operator under various circumstances, which is however obviously tedious and impractical. Hence, it is natural to throw a question: *can we search for a general matching network for Siamese tracking?*

In this work, we show the answer is affirmative by proposing a search algorithm for automatic matching network design. Instead of adopting the conventional cross-correlation and its variants, we explore more feasibility of matching operator selection. Specifically, besides cross-correlation, we introduce six novel matching operators to Siamese tracking, namely Concatenation, Pointwise-Addition, Pairwise-Relation, FiLM, Simple-Transformer and Transductive-Guidance. We shed light on the intrinsic differences of these operators by comparing their performances under different environment degradation types. Surprisingly, by simply replacing the cross-correlation to concatenation, the strong baseline tracker Ocean [47] achieves 1.2 points gains on success score of OTB100 [40] (see Tab. 1). Moreover, we observed that the matching operators show different resilience on various challenging factors and image contents. This inspires us to combine them to exploit complementary informative features.

To this end, we propose a search algorithm, namely Binary Channel Manipulation (BCM), to automatically select and combine matching operators. Firstly, we construct a search space with the aforementioned seven operators. The exemplar and candidate images pass through all matching operators to generate the corresponding response maps. For each response channel, we assign it with a learnable manipulator to indicate its contribution for other tracking steps. Gumbel-Softmax [37] is applied to discretize the manipulators as binary decision, as well as guarantee the differentiable training. Then, we aggregate manipulators of all channels to identify the operator's potential for adapting to the baseline tracker. Our search algorithm aims to find the matching networks with better generalization on different tracking environments. Thus, the performance on the validation set is treated as the reward or fitness. Concretely, we solve the search algorithm using bilevel optimization, which finds the optimal manipulators on the validation set with the weight of other layers (*e.g.,* convolution kernels) learned on the training data. Notably, we simultaneously predict matching networks for both the classification and regression branches in state estimation. **The different search results for classification and regression demonstrate that our method is capable of finding task-dependent matching networks.** Finally, we integrate the learned matching networks into the baseline tracker [47] and train it following the standard Siamese procedure.

The effectiveness of the proposed framework is verified on OTB100 [40], LaSOT [10], GOT10K [14], TrackingNet [27] and TNL2K [39]. Our approach surpasses the baseline tracker [47] on all five benchmarks. It is worth noting that the proposed tracker also outperforms the recent online updating methods DiMP [4] and KYS [5] on all criteria of the evaluated datasets.

The main contributions of this work are twofold.

- We introduce six novel matching operators for Siamese tracking. A systematic analysis reveals that the commonly-used (depthwise) cross-correlation is not a requisite, and an appropriate matching operator can further bring remarkable performance gains.

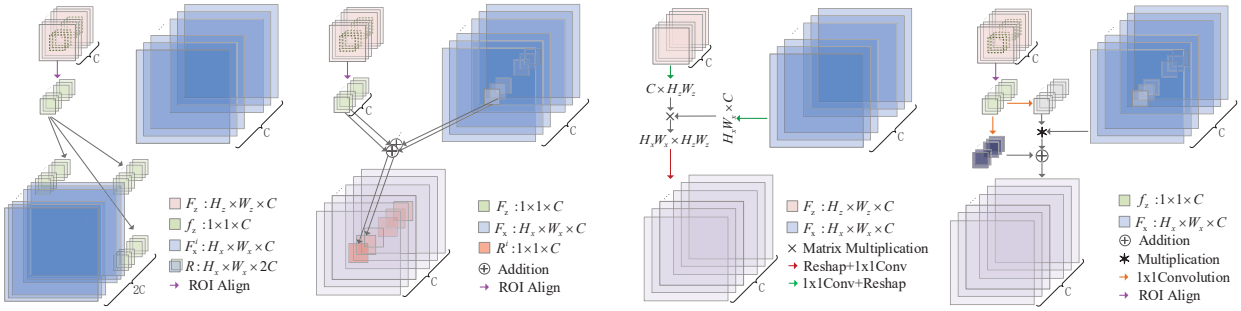
- A conceptually simple algorithm, namely Binary Channel Manipulation (BCM), is proposed for automatic matching networks design with the introduced operators. By integrating the learned matching networks into the baseline tracker, it achieves remarkable performance gains with neglectable overhead on tracking speed.

## 2. Related Work

In this section, we review the related work on matching based tracking, as well as briefly describe recent thriving Siamese trackers, where the baseline tracker belongs to.

### 2.1. Tracking via Heuristic Matching

In the context of visual tracking, it usually corresponds to the process of predicting foreground probability as a one-shot matching problem. SINT [36] proposes to learn a matching function to identify candidate image locations that match with the initial object appearance. The matching function is simply defined as *dot product* operation. Held et al. introduce GOTURN [13], which predicts target location by directly regressing the *concatenation* feature of the exemplar and candidate images. Global Track [15] and ATOM [8] inject the target information into the region proposal network by applying *hadamard product* to exemplar and candidate embeddings. Recent prominent Siamese trackers [3, 19, 18] achieve groundbreaking results on all benchmarks, which is mostly attributed to the effective *cross-correlation* module and its variants. We observed that when choosing matching functions for a tracking method, expertise and massive experiments are inevitably required. Moreover, the heuristic matching network may not be an optimal architecture design. In this work, we propose a differentiable search algorithm to automatically determine which matching functions to use and how to combine them in visual tracking. Since the proposed search algorithm is applied to the Siamese framework, in the following, we briefly retrospect the development of Siamese tracking.




(a) Concatenation (b) Pointwise-addition (c) Pairwise-Relation (d) FiLM

Figure 2: Matching operators: (a) Concatenation (b) Pointwise-Addition (c) Pairwise-Relation (d) FiLM (see Sec. 3.1).

### 2.2. Siamese Tracking

Siamese tracking has drawn attention because of its balanced accuracy and speed. The pioneering work of Siamese trackers, *i.e.*, SiamFC [3], introduces the *cross-correlation* layer as a similarity metric for target matching, which significantly boosts tracking efficiency. SiamRPN [19] ensues to improve SiamFC by advocating a region proposal network for scale estimation. The follow-up works unleash the capability of deeper backbone networks in Siamese tracking by alleviating position bias [18] and perceptual inconsistency [46]. The estimation network evolves from anchor-based to anchor-free mechanism recently [6, 11, 47, 41]. Whilst deeper backbone and advanced estimation network significantly enhance the transferability of tracking models, the feasibility of matching network design remains less investigated. In this work, we narrow this gap by introducing new matching operators and searching their optimal combination for Siamese tracking.

## 3. Analysis of Matching Operators

### 3.1. Instantiations

The standard Siamese tracker takes an exemplar image $\boldsymbol{z}$ and a candidate image $\boldsymbol{x}$ as input. The image $\boldsymbol{z}$ represents object of interest in the first frame, while $\boldsymbol{x}$ is typically larger and represents the search area in subsequent video frames. The two images are first fed into a shared backbone network to generate two corresponding feature maps $\boldsymbol{F}_z \in \mathbb{R}^{H_z \times W_z \times C}$ and $\boldsymbol{F}_x \in \mathbb{R}^{H_x \times W_x \times C}$. Then a matcing network $\varphi$ is applied to inject the information of exemplar $\boldsymbol{F}_z$ to $\boldsymbol{F}_x$, which outputs a correlation feature $\boldsymbol{R}$,

$$\boldsymbol{R} = \varphi(\boldsymbol{F}_z, \boldsymbol{F}_x). \tag{1}$$

Recent top-ranked Siamese trackers define $\varphi$ as *depth-wise cross-correlation* [18, 42, 6, 11, 41, 47]. Notably, when the spatial size of $\boldsymbol{F}_z$ is $1 \times 1$ ($\boldsymbol{f}_z$), the depthwise cross-correlation resembles hadamard product [15]. Besides depthwise cross-correlation, in this work, we explore other matching operators, namely *Concatenation*, *Pointwise-Addition*, *Pairwise-Relation*, *FiLM*, *Simple-Transformer* and *Transductive-Guidance*. The concatenation operator has been exploited in previous work [13], while others have not, to the best of our knowledge. We detail each of them in the following.

**Concatenation** is used by the pairwise function in Relation Networks [35] for visual reasoning. We also explore a concatenation form of $\varphi$, as shown in Fig. 2 (a):

$$\boldsymbol{R} = \mathrm{Conv}([\boldsymbol{f}_z, \boldsymbol{F}_x]), \tag{2}$$

here $\boldsymbol{f}_z \in \mathbb{R}^{1 \times 1 \times C}$ is the pooled features on $\boldsymbol{F}_z$ (inside the bounding box). $[\cdot, \cdot]$ denotes concatenation and Conv is a $1 \times 1$ convolution layer with output channel of $C$.

**Pointwise-Addition** is similar to the hadamard product, but changes "multiplication" to "addition" (see Fig. 2 (b)):

$$\boldsymbol{R} = \boldsymbol{f}_z + \boldsymbol{F}_x, \tag{3}$$

where $+$ denotes elementwise addition.

**Pairwise-Relation** is widely used in video object segmentation [44]. It is a variant of non-local attention [43], and is defined as,

$$\boldsymbol{R} = \mathrm{matmul}(S(\boldsymbol{F}_x), S(\boldsymbol{F}_z)), \tag{4}$$

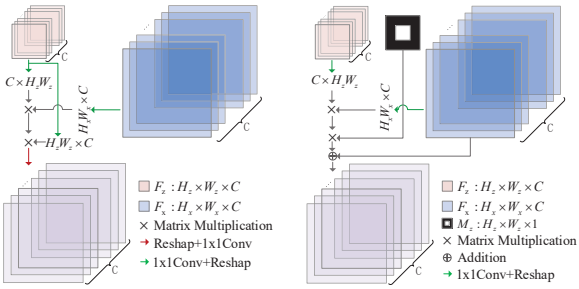
where $S$ reshapes $\boldsymbol{F}_x$ and $\boldsymbol{F}_z$ to the size of $H_x W_x \times C$ and $C \times H_z W_z$, respectively (see Fig. 2 (c)). Here, matmul denotes matrix multiplication. The pairwise-relation measures the affinity of each cell in the exemplar feature to all that in the candidate feature.

**FiLM** is firstly introduced in visual reasoning [30]. It learns to adaptively influence the output of a neural network by applying an affine transformation to the network's "intermediate features", based on some "input". For visual tracking, we consider the exemplar feature $\boldsymbol{f}_z$ as the "input", and the candidate feature $\boldsymbol{F}_x$ as "intermediate features". More formally,

$$\begin{aligned} \gamma &= \mathrm{Conv}(\boldsymbol{f}_z), \\ \beta &= \mathrm{Conv}(\boldsymbol{f}_z), \\ \boldsymbol{R} &= \gamma \boldsymbol{F}_x + \beta, \end{aligned} \tag{5}$$

Table 1: Performance (Success Rate) of different operators on OTB100 [40]. Illumination Variation (IV), Scale Variation (SV), Occlusion (OCC), Deformation (DEF), Motion Blur (MB), Fast Motion (FM), In-Plane Rotation (IPR), Out-of-Plane Rotation (OPR), Out-of-View (OV), Background Clutters (BC) and Low Resolution (LR) are 11 challenging attributes.

| # NUM | # Operators | Overall | IV | SV | OCC | DEF | MB | FM | IPR | OPR | OV | BC | LR |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ① | Depthwise Cross-correlation | 67.2 | 69.3 | 67.7 | 62.8 | 65.2 | 68.3 | 67.5 | 67.8 | 66.6 | **63.9** | 62.6 | 67.9 |
| ② | Concatenation | **68.4** | **71.5** | 67.3 | **65.2** | **66.5** | **70.0** | **69.0** | **69.8** | 67.2 | 62.7 | **65.3** | 65.6 |
| ③ | Pointwise-Addition | 67.1 | 66.6 | 66.2 | 61.5 | 61.8 | 65.6 | 66.8 | 67.7 | 65.9 | 52.2 | 58.1 | **69.7** |
| ④ | Pairwise-Relation | 67.8 | 67.0 | 66.5 | 63.7 | 65.1 | 68.0 | 66.6 | 66.7 | **68.2** | 57.2 | 63.6 | 59.8 |
| ⑤ | FiLM | 67.4 | 69.4 | 66.9 | 60.4 | 63.7 | 66.9 | 67.3 | 66.8 | 65.2 | 53.7 | 58.5 | 66.8 |
| ⑥ | Simple-Transformer | 65.8 | 67.3 | 65.8 | 60.1 | 62.1 | 65.9 | 65.7 | 66.8 | 66.0 | 55.4 | 60.7 | 64.8 |
| ⑦ | Transductive-Guidance | 65.0 | 64.8 | **68.3** | 61.6 | 61.2 | 67.2 | 65.0 | 64.9 | 65.0 | 57.6 | 56.0 | 64.2 |




(a) Simple-Transformer (b) Transductive-Guidance

Figure 3: Matching operators: (a) Simple-Transformer (b) Transductive-Guidance. Details are described in Sec. 3.1.

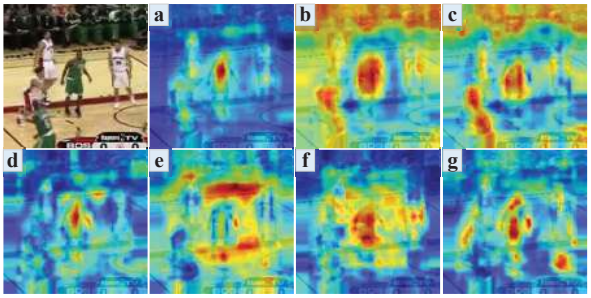



Figure 4: Activation maps of different matching operators. (a) Depthwise Cross-correlation (b) Concatenation (c) Pointwise-Addition (d) Pairwise-Relation (e) FiLM (f) Simple-Transformer (g) Transductive-Guidance.

where the coefficient $\gamma$ and bias $\beta$ are two tensors with size of $1 \times 1 \times C$, as shown in Fig. 2 (d).

**Simple-Transformer** is motivated by recent booming visual transformer [12],

$$\boldsymbol{R} = \mathrm{Att}(query, key, value), \tag{6}$$

where $query = \mathrm{Conv}(\boldsymbol{F}_x), key = \mathrm{Conv}(\boldsymbol{F}_z), value = \mathrm{Conv}(\boldsymbol{F}_z)$. Att is a multi-head attention layer in visual transformer [12], and is implemented by "nn.multiheadAttention" in PyTorch [29]. More details are presented in Fig. 3 (a).

**Transductive-Guidance** is originated from mask propagation mechanism in video object segmentation [44, 45], where the segmentation masks of previous frames guide the prediction of the current frame. In our work, we specifically modify it for Siamese tracking. First, the affinity between examplar and candidate feature is predicted by,

$$\boldsymbol{A} = \mathrm{matmul}(S(\boldsymbol{F}_x), S(\boldsymbol{F}_z)). \tag{7}$$

This step is the same as the computation of the pairwise-relation. With the affinity, the spatial guidance is learned by propagating the pseudo mask of the first frame,

$$\boldsymbol{G} = \mathrm{matmul}(\boldsymbol{A}, S(\boldsymbol{M}_z)), \tag{8}$$

where $\boldsymbol{M}_z$ is the pseudo mask of the first frame. Specifically, the pixels inside and outside the bounding box are set to 1 and 0, respectively, as shown in Fig. 3 (b). $\boldsymbol{G}$ serves as the spatial guidance for target localization, in which each pixel indicates the foreground probability of a location. Then the spatial guidance is fused with the visual feature by,

$$\boldsymbol{R} = \boldsymbol{G} + \boldsymbol{F}_x. \tag{9}$$

### 3.2. Analysis

In Sec. 3.1, we introduce six novel matching operators for Siamese tracking, besides the conventional depthwise cross-correlation. It is natural to ask: *How do these new operators perform, and could the conventional depthwise cross-correlation be replaced by these proposed operators?* We answer the questions in this section.

**Performance of Individual Operators.** To investigate the impact of each operator on Siamese tracking, we apply them to a recent tracker Ocean [47], and evaluate the performance on OTB100 [40]. As shown in Tab. 1, the vanilla Ocean [47] with depthwise cross-correlation (①) achieves overall success of 67.2. When replacing the depthwise cross-correlation by Simple-Transformer (⑥)

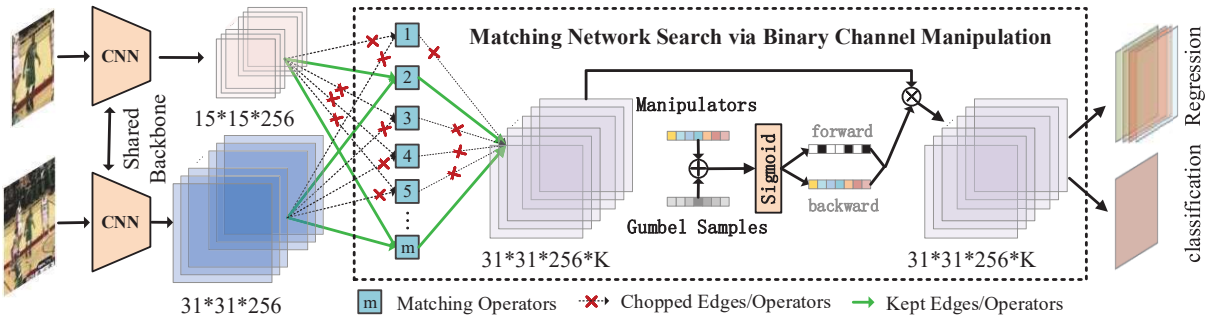



Figure 5: Overview of the proposed framework AutoMatch. The matching operators in search space explore the relation between exemplar and candidate features. The crosses and dashed arrows indicate the discarded operators after searching with binary channel manipulation. And operators linked with the green arrows constructs the searched matching network. The search algorithm is applied to both classification and regression, and only one of that is illustrated here for simplicity.

and Transductive-Guidance (⑦), the overall score drops to 65.8 and 65.0, respectively. The performance degradation illustrates that randomly choosing a matching operator may bring negative impacts to a tracking framework. But surprisingly, the results of all other four operators (②∼⑤) are favorably comparable to or even better than depthwise cross-correlation. The comparisons inspire us that the classical depthwise cross-correlation is not the optimal choice for Siamese tracking, and an appropriate matching operator can lead to better tracking accuracy.

**Potential of Complementarity.** Although one well-designed matching operator may surpass classical depthwise cross-correlation under certain circumstances, the improvements cannot be guaranteed for all challenging cases. As shown in Tab. 1, although the concatenation operator (②) exhibits superiority over most challenging factors, it is inferior to Transductive-Guidance (⑦) on Scale Variation (SV), Pairwise-Relation (④) on Out-of-Plane Rotation (OPR), Depthwise Cross-correlation (①) on Out-of-View (OV) and Point-Addition (③) on Low Resolution (LR). We further visualize the activation map of matching outputs in Fig. 4. It shows that the depthwise cross-correlation (a), Pairwise-relation (d), and Transductive-Guidance (g) tend to filter out the context features and focus on the target itself. Conversely, the concatenation (b), Pointwise-Addition (c), Simple-Transformer (e), and FiLM (e) exploit more context information. The possible reason is that the hard negative examples introduced by the context help prevent overfitting to the easy background.

In a nutshell, the quantitative comparison in Tab. 1 and qualitative analysis in Fig. 4 demonstrate that different matching operators show different resilience on various challenging factors and image contents. This inspires us to combine them to exploit complementary informative features. Instead of searching for the best matching operators under various circumstances, which is obviously impractical, we propose an automatic method that can adaptively learn to choose and combine the matching functions.

# 4. Methodology

## 4.1. Overview of AutoMatch

The proposed framework AutoMatch is illustrated in the Fig. 5. Typical Siamese tracking framework contains three main steps, *i.e.*, feature extraction, matching, and target localization. Given an exemplar image $\boldsymbol{z}$ and a candidate image $\boldsymbol{x}$, a backbone network is first applied to extract visual features $\boldsymbol{F}_z$ and $\boldsymbol{F}_x$. $\boldsymbol{F}_z$ and $\boldsymbol{F}_x$ then pass through a matching network $\varphi$ to learn their relation. $\varphi$ is generally defined as depthwise cross-correlation in recent works [18, 47]. In our study, the matching network design evolves from heuristic selection to automatic search. Concretely, $\boldsymbol{F}_z$ and $\boldsymbol{F}_x$ are fed to matching operators in the search space (see Sec. 3.1), which obtains $m$ multi-channel response features $\{\boldsymbol{r}_1, \boldsymbol{r}_2, ..., \boldsymbol{r}_m\}$. Each channel of a response feature is assigned with a learnable manipulator $w_i^j$, indicating a feature channel's contribution to other tracking steps. We introduce the binary Gumbel-Softmax [37] to discretize the manipulators for binary decision, as well as guarantee the differentiable training. The learning of manipulators is formulated as bilevel optimization (see Sec. 4.3). Two operators are finally retained based on the guidance of the learned manipulators, and their response maps are concatenated as the input of the following steps. With the learned matching networks, the classification and regression networks are followed to predict the target state (see Sec. 5.1).

## 4.2. Binary Channel Manipulation

Let $\mathcal{O} = \{o_1, o_2, ..., o_m\}$ be the search space consisting of optional matching operators $o_i(\cdot)$ to be applied to exemplar and candidate features. The response set $\mathcal{R}$ is got by,

$$\mathcal{R} = \{o_1(\boldsymbol{z}, \boldsymbol{x}), ..., o_m(\boldsymbol{z}, \boldsymbol{x})\}. \tag{10}$$

The search algorithm aims to find the optimal combination of operators based on the response set $\mathcal{R}$. We propose binary channel manipulation (BCM) to decide the contribution of an operator for target state prediction. Each element $\boldsymbol{r}_i^j$ in $\mathcal{R}$ is a tensor with size of $H_x \times W_x \times C$. We assign each feature channel with a learnable manipulator $w_i^j$, and then aggregates the weighted maps in $\mathcal{R}$ by concatenation,

$$\boldsymbol{E} = [\sigma(w_1^1)\boldsymbol{r}_1^1, ..., \sigma(w_i^j)\boldsymbol{r}_i^j, ..., \sigma(w_m^C)\boldsymbol{r}_m^C], \tag{11}$$

where $\boldsymbol{r}_i^j$ indicates the $j_{th}$ channel of the $i_{th}$ response feature. $\sigma$ is sigmoid. $\boldsymbol{E} \in \mathbb{R}^{H_x \times W_x \times C|\mathcal{O}|}$ denotes the aggregated feature, which is used as the input of subsequent target estimation network. The manipulator defines the channel's contribution to target location. For each operator, we define the summation of channel manipulators as the potential $p_i$ of an operator for adapting to the baseline tracker,

$$p_i = \sum_{j=1}^{C} \sigma(w_i^j). \tag{12}$$

Inspired by channel pruning [1] and differentiable network architecture search [7, 23], we translate the continuous solution $w_i^j$ to discrete one for final decision. These discrete decisions are trained end-to-end using the Gumbel-Softmax [37]. Concretely, given a distribution with (two) class probabilities $\pi = \{\pi_1 = \sigma(w_i^j), \pi_2 = 1 - \sigma(w_i^j)\}$, the discrete samples $d$ can be drawn using,

$$d = \text{onehot}(\arg\min_k[\log(\pi_k) + g_k]), \tag{13}$$

where $g_k$ is noise sample drawn from Gumbel distribution. $k \in \{1, 2\}$ denotes binary classification. The Gumbel-Softmax defines a continuous, differentiable approximation by replacing the argmax with a softmax,

$$y_k = \frac{\exp((\log(\pi_k) + g_k)/\tau)}{\sum_{c=1}^{2} \exp((\log(\pi_c) + g_c)/\tau)}. \tag{14}$$

Substituting $\pi_1 = \sigma(w_i^j)$, $\pi_2 = 1 - \sigma(w_i^j)$, Eq. 14 is simplified to ($k = 1$ for binary case),

$$y_1 = \sigma(\frac{w_i^j + g_1 - g_2}{\tau}). \tag{15}$$

We attach the derivation in supplementary materials due to space limit. The $\tau$ is set to 1, $g_k$ to 0 following [2, 37]. For the discrete sample $d$, a hard value is used during the forward pass and gradients are obtained from soft value during the backward pass:

$$d = \begin{cases} y_1 > 0.5 \equiv \frac{w_i^j + g_1 - g_2}{\tau} = w_i^j > 0, forward \\ y_1, backward. \end{cases} \tag{16}$$

### 4.3. Bilevel Optimization

With binary channel manipulation, our goal is to jointly learn the manipulators $w$ and the weights $\theta$ of other layers (*e.g.,* convolution layers in operators). Analogous to differentiable architecture search [23], where the validation set performance is treated as the reward or fitness, we aim to optimize the validation loss. Let $\mathcal{L}_{train}$ and $\mathcal{L}_{val}$ denote the training and validation loss, respectively. The goal for matching network search is to find $w^*$ that minimizes the validation loss $\mathcal{L}_{val}(\theta^*; w^*)$, where the network parameters $\theta^*$ associated with the architecture are obtained by minimizing the training loss $\theta^* = \text{argmin}_w \ \mathcal{L}_{train}(\theta, w^*)$. This implies a bilevel optimization problem [23, 7] with $w$ as the upper-level variable and $\theta$ as the lower-level variable,

$$\min_w \ \mathcal{L}_{val}(\theta^*(w); w), \tag{17}$$

$$s.t. \quad \theta^*(w) = \text{argmin}_\theta \ \mathcal{L}_{train}(\theta, w). \tag{18}$$

To speed up the bilevel optimization during training, Liu et al. propose a simple approximation in [23],

$$\nabla_w \mathcal{L}_{val}(\theta^*(w); w) \tag{19}$$

$$\approx \nabla_w \mathcal{L}_{val}(\theta - \epsilon \nabla_\theta \mathcal{L}_{train}(\theta, w), w), \tag{20}$$

where $\epsilon$ is the learning rate for a step of inner optimization. The derivation is beyond the scope of this work. We refer the reader to [23] for more details about the approximation.

In summary, we propose binary channel manipulation to identify the contribution of a matching operator. Then we learn the manipulators by bilevel optimization. We simultaneously apply the search algorithm on the classification and regression branches in state estimation to learn task-dependent matching networks. After training, the first two operators with the maximum potential $p_i$ are retained (see green arrows in Fig. 5). Finally, we follow the procedure of the baseline tracker [47] to train the searched architecture.

## 5. Experiments

### 5.1. Implementation Details

**Network Architecture.** We adopt the recent Siamese tracker Ocean [47] as the baseline model. The backbone network is the modified ResNet50 [26]. The target localization network consists of a classification branch and a regression branch. Though the updating branch of Ocean [47] is not used in our work, our tracker remarkably outperforms its online updating version. We refer the readers to [47] for more details about the baseline tracker. In this work, we simultaneously search for the target-dependent matching networks for the classification and regression branches.

**Training Procedure.** The training procedure consists of two stages, *i.e.,* matching network search and new tracker

Table 2: Result comparisons on five tracking benchmarks. The red, green and blue indicate performances ranked at the first, second, and third places. Ocean [47] is our baseline model, and we apply the proposed search algorithm on it.

| Methods | Year | OTB100 | | LaSOT | | TrackingNet | | TNL2K | | GOT10K | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Succ. | Prec. | Succ. | Prec. | Succ. | Prec. | Succ. | Prec. | AO | $SR_{0.5}$ | $SR_{0.75}$ |
| SiamFC [3] | 2016 | 58.7 | 77.2 | 33.6 | 33.9 | 57.1 | 66.3 | 29.5 | 28.6 | 34.8 | 35.3 | 9.8 |
| MDNet [28] | 2016 | 67.8 | 90.9 | 39.7 | 37.3 | 60.6 | 56.5 | 31.0 | 32.2 | 29.9 | 30.3 | 9.9 |
| ECO [9] | 2017 | 69.1 | 91.0 | 32.4 | 30.1 | 55.4 | 49.2 | 32.6 | 31.7 | 31.6 | 30.9 | 11.1 |
| VITAL [34] | 2018 | 69.1 | 91.7 | 39.0 | 36.0 | - | - | 36.6 | 35.3 | 35.0 | 36.0 | 9.0 |
| GradNet [20] | 2019 | 63.9 | 86.1 | 36.5 | 35.1 | - | - | 31.7 | 31.8 | - | - | - |
| SiamDW [46] | 2019 | 67.4 | 90.5 | 38.4 | 35.6 | - | - | 32.3 | 32.6 | 41.6 | 47.5 | 14.4 |
| SiamRPN++ [18] | 2019 | 69.6 | 92.3 | 49.6 | 49.1 | 73.3 | 69.4 | 41.3 | 41.2 | 51.7 | 61.6 | 32.5 |
| ATOM [8] | 2019 | 66.7 | 87.9 | 51.5 | 50.5 | 70.3 | 64.8 | 40.1 | 39.2 | 55.6 | 63.4 | 40.2 |
| DiMP [4] | 2019 | 68.6 | 89.9 | 56.9 | 56.7 | 74.0 | 68.7 | 44.7 | 43.4 | 61.1 | 71.7 | 49.2 |
| SiamFC++ [41] | 2020 | 68.3 | 91.2 | 54.3 | 54.7 | 75.4 | 70.5 | 38.6 | 36.9 | 59.5 | 69.5 | 47.9 |
| D3S [24] | 2020 | - | - | - | - | 72.8 | 66.4 | 38.8 | 39.3 | 59.7 | 67.6 | 46.2 |
| MAMLTrack [38] | 2020 | 71.2 | **92.6** | 52.3 | 53.1 | 75.7 | 72.5 | 28.4 | 29.5 | - | - | - |
| SiamAttn [42] | 2020 | 71.2 | **92.6** | 56.0 | - | 75.2 | - | - | - | - | - | - |
| SiamCAR [11] | 2020 | - | - | 50.7 | 51.0 | - | - | 35.3 | 38.4 | 56.9 | 67.0 | 41.5 |
| SiamBAN [6] | 2020 | 69.6 | 91.0 | 51.4 | 52.1 | - | - | 41.0 | 41.7 | - | - | - |
| KYS [5] | 2020 | 69.5 | 91.0 | 55.4 | 55.8 | 74.0 | 68.8 | 44.9 | **43.5** | 63.6 | 75.1 | 51.5 |
| Ocean [47] | 2020 | 67.2 | 90.2 | 52.6 | 52.6 | 70.3 | 68.8 | 38.4 | 37.7 | 59.2 | 69.5 | 46.5 |
| **AutoMatch** | Ours | **71.4** | **92.6** | **58.3** | **59.9** | **76.0** | **72.6** | **47.2** | **43.5** | **65.2** | **76.6** | **54.3** |

training. In the first stage, we search for the matching networks using methods in Sec. 4 and determine the best cell based on the validation performance. In the second stage, we use the optimized matching networks to construct a new tracker on the baseline approach Ocean [47]. Both stages are trained with Youtube-BB [31], ImageNet-VID [32], ImageNet-DET [32], GOT10K [14] and COCO [22] (including training and validation sets). The search algorithm's training takes 5 epochs, with each containing $6\times10^5$ pairs. The learning rate exponentially decays from $10^{-3}$ to $10^{-4}$. The training of the new tracker follows the baseline model [47]. **Notably, we simplify Ocean [47] by reducing the training epochs from 50 to 20 to expedite the learning process.** For the first 5 epochs, we start with a warmup learning rate of $10^{-3}$. For the remaining epochs, the learning rate exponentially decays from $5\times10^{-3}$ to $5\times10^{-5}$. Both stages are trained with synchronized SGD [17] on 4 GTX2080 Ti GPUs, with each hosting 32 images.

### 5.2. State-of-the-art Comparison

The search algorithm determines different matching networks for the classification and regression branches. **After the first stage training, Simple-Transformer and FiLM are retrained for the classification branch, meanwhile, FiLM and Pairwise-Relation are preserved for the regression branch.** We compare the new tracker with state-of-the-art models on five benchmarks. Our tracker achieves compelling performance while running at over 50 FPS. Notably, it only takes less than 24 hours for the second stage training (with 4 GTX2080Ti GPUs), which provides a strong but efficient baseline for further research.

**OTB100 [40].** OTB100 is a classical tracking benchmark consisting of 100 sequences. Methods are ranked by the area under the success curve (AUC) and precision (Prec.). As shown in Tab. 2, our model achieves the top-ranked AUC score, which outperforms the previous best result by SiamAttn [42], *i.e.*, 71.4 *vs* 71.2. When equipping the baseline tracker Ocean [47] with our searched matching network, it brings favorabale 4.2 points gains, *i.e.*, 71.4 *vs* 67.2. The proposed model also surpasses online updating models ATOM [8]/DiMP[4] for 4.5/2.6 points, respectively.

**LaSOT [10].** LaSOT is a tracking benchmark designed for long-term tracking. Tab. 2 shows the comparison results on 280 testing videos. Our method achieves the best AUC and precision score, outperforming Ocean [47] for 5.7 and 7.3 points, respectively. Compared with DiMP [4], our method achieves improvements of 1.4 points on success score. Notably, the proposed tracker runs at 50 FPS, which is comparable to 58 FPS of Ocean, and faster than 43 FPS of DiMP. The comparisons demonstrate that the proposed method brings significant performance gains with small overhead.

**TrackingNet [27].** TrackingNet is a large-scale tracking dataset consisting of 511 sequences for testing. The evaluation is performed on the online server. We report the results in Tab. 2. Compared with the baseline tracker Ocean [47], it achieves 5.7 points gains on success score. Our model also surpasses the meta-learning based MAMLTrack [38] on TrackingNet, *i.e.,* success score of 76.0 *vs* 75.7.

**GOT10K [14].** The evaluation of GOT10K is on the online server. We report the average overlap (AO), success rate ($SR_{0.5}$, $SR_{0.75}$) in Tab. 2. Comparing the proposed model

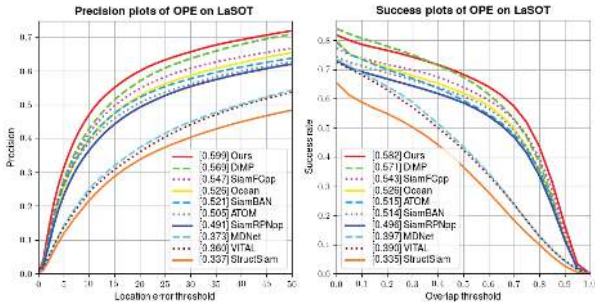



Figure 6: Visualization of results comparison on LaSOT.

with the baseline Ocean [47], we achieve gains of 6 points, 7.1 points, and 7.8 points on AO, $SR_{0.5}$, and $SR_{0.75}$, respectively. Notably, our model outperforms SiamBAN [6] for 1.6 points on AO, while running faster (50FPS *vs.* 40FPS).

**TNL2K [39].** TNL2K is a new dataset which consists of 2000 high diversity videos for natural language guided tracking. Adversarial samples and thermal images are introduced to improve the generality of tracking evaluation. Besides tracking by natural language, it also provides the results of tracking by bounding boxes. In Tab. 2, we present the results on 700 testing sequences. It shows that our model achieves the best success and precision scores among the compared trackers.

### 5.3. Ablation and Analysis

**One or Many Manipulators.** We link each channel in an operator with a manipulator. Differently, in differentiable neural network search [23], an operator is identified by a scalar. We also try this strategy, *i.e.,* assigning a matching operator with a scalar during the search. We achieves a final success score of 69.5 on OTB100 [40] and 54.7 on LaSOT [10]. The results are inferior to our model, which demonstrates the superiority of our search algorithm. We conjecture that the aggregation of channel information can provide finer guidance for operator selecting.

**Random Search.** To demonstrate the efficacy of the search algorithm, we evaluate the performance of random search. Two operators are randomly retained for classification and regression branches, respectively. We report the average performance of the three-time random search and training. The average success score on OTB100 and LaSOT are 69.1 and 53.2. The results manifest that the introduced search method is effective in finding better operators combination.

**NAS-like Matching Cell.** In differentiable neural network search [23], it represents the basic operating cell as a Directed Acyclic Graph (DAG). Each cell contains multiple nodes, and each node aggregates the outputs of multiple basic operators (*e.g.,* $3 \times 3$ convolution layer). One intuitive idea is directly replacing the operators in NAS with our designed matching functions and then searching a matching network. As shown in Fig. 7, we use DARTS [23] to

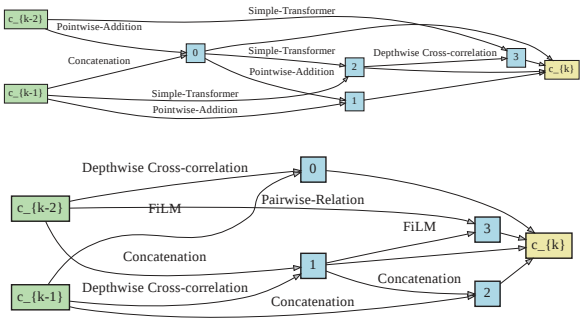



Figure 7: **top:** NAS-like Matching Network for classification. **bottom:** NAS-like Matching Network for regression.

search a matching cell, which looks like that in NAS. Surprisingly, though the searched cell is much complex than ours, it does not show superiority. Concretely, the NAS-like cell achieves an success score of 55.7 on LaSOT and runs at 35 FPS. Both the performance and inference speed is inferior to the proposed model. The comparison proclaims that directly borrowing NAS to matching network search may not be an optimal choice. We present more details about the DARTS-like structure search and the related work in supplementary materials, due to space limit.

## 6. Conclusion

In this work, we introduce six novel operations to explore more feasibility on matching operator selection in Siamese tracking. Quantitative and quantitative analyses demonstrate that the classical (depthwise) cross-correlation is not the optimal choice for Siamese tracking. We simultaneously find the optimal matching networks for both classification and regression branches in state estimation with the proposed binary channel manipulation (BCM). The learned matching networks are applied to a baseline tracker, and the experimental result shows the robustness of our approach on both short-term and long-term benchmarks. In the future work, we will apply our method to other matching based frameworks, *e.g.,* ATOM.

**Acknowledgements.** We thank Heng Fan for his help during ICCV2021 rebuttal. This work was supported by the National Key Research and Development Program of China (Grant No. 2020AAA0106800), the Natural Science Foundation of China (Grant No. 61902401, No. 61972071, No. 61906052, No. 62036011, No. 61721004, No. 61972394, and No. U2033210), the CAS Key Research Program of Frontier Sciences (Grant No. QYZDJ-SSWJSC040), the Postdoctoral Innovative Talent Support Program BX20200174, China Postdoctoral Science Foundation Funded Project 2020M682828. The work of Bing Li was also supported by the Youth Innovation Promotion Association, CAS.

## References


[1] Babak Ehteshami Bejnordi, Tijmen Blankevoort, and Max Welling. Batch-shaping for learning conditional channel gated networks. *arXiv preprint arXiv:1907.06627*, 2019. 6

[2] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013. 6

[3] Luca Bertinetto, Jack Valmadre, Joao F Henriques, Andrea Vedaldi, and Philip HS Torr. Fully-convolutional siamese networks for object tracking. In *ECCV Workshop*, pages 850–865. Springer, 2016. 1, 2, 3, 7

[4] Goutam Bhat, Martin Danelljan, Luc Van Gool, and Radu Timofte. Learning discriminative model prediction for tracking. In *ICCV*, pages 6182–6291, 2019. 2, 7

[5] Goutam Bhat, Martin Danelljan, Luc Van Gool, and Radu Timofte. Know your surroundings: Exploiting scene information for object tracking. In *European Conference on Computer Vision*, pages 205–221. Springer, 2020. 2, 7

[6] Zedu Chen, Bineng Zhong, Guorong Li, Shengping Zhang, and Rongrong Ji. Siamese box adaptive network for visual tracking. In *CVPR*, pages 6668–6677, 2020. 1, 3, 7, 8

[7] Xiangxiang Chu, Tianbao Zhou, Bo Zhang, and Jixiang Li. Fair darts: Eliminating unfair advantages in differentiable architecture search. In *European Conference on Computer Vision*, pages 465–480. Springer, 2020. 6

[8] Martin Danelljan, Goutam Bhat, Fahad Shahbaz Khan, and Michael Felsberg. Atom: Accurate tracking by overlap maximization. In *CVPR*, pages 4660–4669, 2019. 1, 2, 7

[9] Martin Danelljan, Goutam Bhat, Fahad Shahbaz Khan, Michael Felsberg, et al. Eco: Efficient convolution operators for tracking. In *CVPR*, pages 6931–6939, 2017. 7

[10] Heng Fan, Liting Lin, Fan Yang, Peng Chu, Ge Deng, Sijia Yu, Hexin Bai, Yong Xu, Chunyuan Liao, and Haibin 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. 2, 7, 8

[11] Dongyan Guo, Jun Wang, Ying Cui, Zhenhua Wang, and Shengyong Chen. Siamcar: Siamese fully convolutional classification and regression for visual tracking. In *CVPR*, pages 6269–6277, 2020. 1, 3, 7

[12] Kai Han, Yunhe Wang, Hanting Chen, Xinghao Chen, Jianyuan Guo, Zhenhua Liu, Yehui Tang, An Xiao, Chunjing Xu, Yixing Xu, et al. A survey on visual transformer. *arXiv preprint arXiv:2012.12556*, 2020. 4

[13] David Held, Sebastian Thrun, and Silvio Savarese. Learning to track at 100 fps with deep regression networks. In *European conference on computer vision*, pages 749–765. Springer, 2016. 2, 3

[14] Lianghua Huang, Xin Zhao, and Kaiqi Huang. Got-10k: A large high-diversity benchmark for generic object tracking in the wild. *arXiv preprint arXiv:1810.11981*, 2018. 2, 7

[15] Lianghua Huang, Xin Zhao, and Kaiqi Huang. Globaltrack: A simple and strong baseline for long-term tracking. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 11037–11044, 2020. 2, 3

[16] Hamed Kiani Galoogahi, Ashton Fagg, and Simon Lucey. Learning background-aware correlation filters for visual tracking. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1135–1143, 2017. 1

[17] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989. 7

[18] Bo Li, Wei Wu, Qiang Wang, Fangyi Zhang, Junliang Xing, and Junjie Yan. Siamrpn++: Evolution of siamese visual tracking with very deep networks. In *CVPR*, pages 4282–4291, 2019. 1, 2, 3, 5, 7

[19] Bo Li, Junjie Yan, Wei Wu, Zheng Zhu, and Xiaolin Hu. High performance visual tracking with siamese region proposal network. In *CVPR*, pages 8971–8980, 2018. 1, 2, 3

[20] Peixia Li, Boyu Chen, Wanli Ouyang, Dong Wang, Xiaoyun Yang, and Huchuan Lu. Gradnet: Gradient-guided network for visual object tracking. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6162–6171, 2019. 7

[21] Xi Li, Weiming Hu, Chunhua Shen, Zhongfei Zhang, Anthony Dick, and Anton Van Den Hengel. A survey of appearance models in visual object tracking. *ACM transactions on Intelligent Systems and Technology (TIST)*, 4(4):58, 2013. 1

[22] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, pages 740–755, 2014. 7

[23] Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*, 2018. 6, 8

[24] Matas Jiri Lukezic, Alan and Matej Kristan. D3s-a discriminative single shot segmentation tracker. In *CVPR*, pages 7133–7142, 2020. 7

[25] Seyed Mojtaba Marvasti-Zadeh, Li Cheng, Hossein Ghanei-Yakhdan, and Shohreh Kasaei. Deep learning for visual tracking: A comprehensive survey. *arXiv preprint arXiv:1912.00535*, 2019. 1

[26] Henrique Morimitsu. Multiple context features in siamese networks for visual object tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 0–0, 2018. 6

[27] Matthias Muller, Adel Bibi, Silvio Giancola, Salman Alsubaihi, and Bernard 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. 2, 7

[28] Hyeonseob Nam and Bohyung Han. Learning multi-domain convolutional neural networks for visual tracking. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4293–4302, 2016. 7

[29] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017. 4

[30] Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a

general conditioning layer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018. 3

[31] Esteban Real, Jonathon Shlens, Stefano Mazzocchi, Xin Pan, and Vincent Vanhoucke. Youtube-boundingboxes: A large high-precision human-annotated data set for object detection in video. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pages 7464–7473. IEEE, 2017. 7

[32] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015. 7

[33] Arnold WM Smeulders, Dung M Chu, Rita Cucchiara, Simone Calderara, Afshin Dehghan, and Mubarak Shah. Visual tracking: An experimental survey. *IEEE transactions on pattern analysis and machine intelligence*, 36(7):1442–1468, 2013. 1

[34] Yibing Song, Chao Ma, Xiaohe Wu, Lijun Gong, Linchao Bao, Wangmeng Zuo, Chunhua Shen, Rynson WH Lau, and Ming-Hsuan Yang. Vital: Visual tracking via adversarial learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8990–8999, 2018. 7

[35] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales. Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1199–1208, 2018. 3

[36] Ran Tao, Efstratios Gavves, and Arnold WM Smeulders. Siamese instance search for tracking. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1420–1429, 2016. 2

[37] Thomas Verelst and Tinne Tuytelaars. Dynamic convolutions: Exploiting spatial sparsity for faster inference. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2320–2329, 2020. 2, 5, 6

[38] Guangting Wang, Chong Luo, Xiaoyan Sun, Zhiwei Xiong, and Wenjun Zeng. Tracking by instance detection: A meta-learning approach. In *CVPR*, pages 6288–6297, 2020. 7

[39] Xiao Wang, Xiujun shu, Zhipeng Zhang, Bo Jiang, Yaowei Wang, Yonghong Tian, and Feng Wu. Towards more flexible and accurate object tracking with natural language: Algorithms and benchmark. In *CVPR*, 2021. https://sites.google.com/view/langtrackbenchmark/. 2, 8

[40] Yi Wu, Jongwoo Lim, and Ming-Hsuan Yang. Object tracking benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9):1834–1848, 2015. 2, 4, 7, 8

[41] Yinda Xu, Zeyu Wang, Zuoxin Li, Ye Yuan, and Gang Yu. Siamfc++: Towards robust and accurate visual tracking with target estimation guidelines. In *AAAI*, pages 12549–12556, 2020. 1, 3, 7

[42] Kai Yang, Zhenyu He, Zikun Zhou, and Nana Fan. Siamatt: Siamese attention network for visual tracking. *Knowledge-Based Systems*, page 106079, 2020. 1, 3, 7

[43] Yulun Zhang, Kunpeng Li, Kai Li, Bineng Zhong, and Yun Fu. Residual non-local attention networks for image restoration. *arXiv preprint arXiv:1903.10082*, 2019. 3

[44] Yizhuo Zhang, Zhirong Wu, Houwen Peng, and Stephen Lin. A transductive approach for video object segmentation. In *CVPR*, June 2020. 3, 4

[45] Zhipeng Zhang, Bing Li, Weiming Hu, and Houweng Peng. Towards accurate pixel-wise object tracking by attention retrieval. *arXiv preprint arXiv:2008.02745*, 2020. 4

[46] Zhipeng Zhang and Houwen Peng. Deeper and wider siamese networks for real-time visual tracking. In *CVPR*, pages 4591–4600, 2019. 3, 7

[47] Zhipeng Zhang, Houwen Peng, Jianlong Fu, Bing Li, and Weiming Hu. Ocean: Object-aware anchor-free tracking. In *ECCV*, 2020. 1, 2, 3, 4, 5, 6, 7, 8