High-Performance Discriminative Tracking with Transformers

Bin Yu[1,2], Ming Tang[2], Linyu Zheng[1,2], Guibo Zhu[1,2], Jinqiao Wang[1,2,3],
Hao Feng[4], Xuetao Feng[4], Hanqing Lu[1,2]
[1]School of Artificial Intelligence, UCAS, China
[2]National Laboratory of Pattern Recognition, Institute of Automation, CAS, China
[3]ObjectEye Inc. [4]Alibaba Group
{bin.yu,tangm,linyu.zheng,gbzhu,jqwang,luhq}@nlpr.ia.ac.cn
{yuanning.fh,xuetao.fxt}@alibaba-inc.com

Abstract

End-to-end discriminative trackers improve the state of the art significantly, yet the improvement in robustness and efficiency is restricted by the conventional discriminative model, i.e.*, least-squares based regression. In this paper, we present DTT, a novel single-object discriminative tracker, based on an encoder-decoder Transformer architecture. By self- and encoder-decoder attention mechanisms, our approach is able to exploit the rich scene information in an end-to-end manner, effectively removing the need for hand-designed discriminative models. In online tracking, given a new test frame, dense prediction is performed at all spatial positions. Not only location, but also bounding box of the target object is obtained in a robust fashion, streamlining the discriminative tracking pipeline. DTT is conceptually simple and easy to implement. It yields state-of-the-art performance on four popular benchmarks including GOT-10k, LaSOT, NfS, and TrackingNet while running at over 50 FPS, confirming its effectiveness and efficiency. We hope DTT may provide a new perspective for single-object visual tracking.*

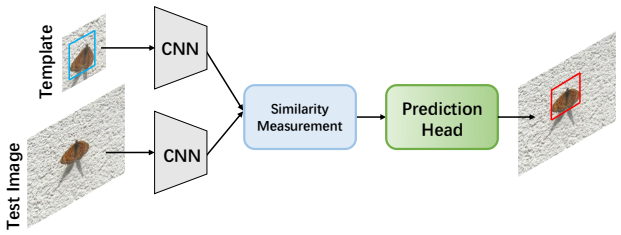
1. Introduction

Generic visual tracking is a long-standing topic in the field of computer vision and has attracted increasing attention over the last decades. Despite significant progress in recent years [2,6,7,19,23,26,33,34,41,42], visual tracking remains challenging due to numerous factors such as very limited online training samples, large appearance variation, and heavy background clutters.

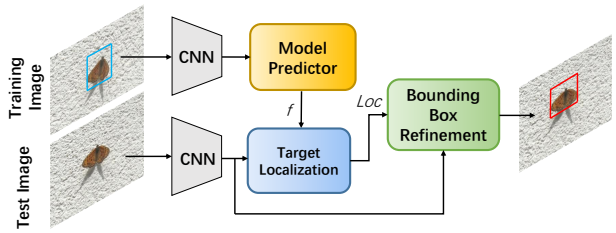
In recent years, Siamese network based trackers [1,5,11–13,19,20,36,37] have attracted great attention because of their balanced speed and accuracy. These methods formulate the visual tracking task as a target matching problem and aim to learn a general similarity metric between the target template and the search region (see Fig. 1(a)). Powerful backbone networks [19] and effective proposal networks [11] are proposed to achieve promising results. However, the Siamese learning framework cannot exploit the background information effectively to improve the discrimination.

On the contrary, modern discriminative trackers [2,3,42] are able to exploit the background information and typically learn an adaptive discriminative model by minimizing the regression loss (see Fig. 1(b)). Although they have achieved leading performance on serval benchmarks [15,16,35], we point out that such tracking scheme has the following three limitations. 1) The discrimination of the applied regression models (*i.e.*, least-squares based regression) is rough and insufficient for robust tracking because the conventional models often fail to reserve the detailed scene information and encode the relationships among the distractors in the background. 2) Modern discriminative models can only contribute to localization, and thus have to rely on other methods like ATOM [6] for final predicted bounding box, resulting in a separated tracking pipeline. 3) In online tracking, iterative solution is needed in both the model predictor [2] and the bounding box refinement module [6], which is not friendly to most embedded devices and may negatively affect the efficiency.

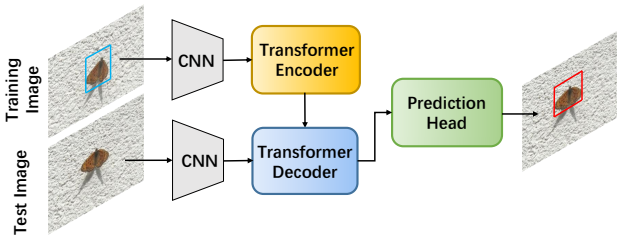
To this end, we present a novel discriminative tracker with Transformers, termed as DTT, which is a conceptually simple, efficient and robust tracking architecture. The core advantage of DTT is that it can effectively and efficiently exploit the rich scene information for both classification and bounding box regression. Specifically, DTT is built upon an encoder-decoder Transformer architecture [32] where the features of training image obtained by convolutional neural networks (CNNs) are fed to the encoder, as shown in Fig. 1(c). Due to the self-attention mechanism in the encoder, its outputs, called *discrimina-*



(a) A simple Siamese tracking pipeline.



(b) A discriminative tracking pipeline.



(c) The simplified architecture of the proposed DTT.

Figure 1: The simplified framework of our method along with two dominant tracking schemes. Different from the previous methods, DTT is a brand new tracking architecture. It enables utilizing the rich scene information in the features effectively for both localization and bounding box regression, simplifying the discriminative tracking pipeline.

tive feature embeddings, contain a wealth of global and local information about the scene, which are beneficial to discriminative tracking. Through end-to-end training on large-scale tracking datasets, the discriminative feature embeddings can highlight the most discriminative representation for visual tracking task, effectively removing the need for a hand-designed discriminative model predictor.

The decoder receives the test image features as one of the three input (see Fig. 2). Through the self-attention module, the local and global information about the test image is extracted and contained in the feature embeddings of each pixel, enabling dense prediction at all spatial positions for more accurate and robust tracking. Then in the following cross-attention module, the discriminative embeddings are utilized to produce the fused feature embeddings for prediction. Finally, similar to Siamese trackers [5, 13], our prediction head consists of a classification branch and a bounding box regression branch for robust and accurate tracking. Besides, considering the importance of model update along with its efficiency in visual tracking, we employ a simple yet effective update method to fit our DTT to the variation of the scene and the target in online tracking. Without bells and whistles, the overall pipeline is neat, straightforward, and easy to implement. Extensive experiments on four popular benchmarks, GOT-10k [15], LaSOT [10], NfS [16], and TrackingNet [24], show that DTT achieves the state-of-the-art results on all datasets, while running at over 50 FPS. Code shall be released.

In summary, our contributions are in four folds.

1. We propose a novel and conceptually simple discriminative tracker, called DTT, which is based on an encoder-decoder transformer architecture.

2. DTT is able to exploit rich scene information and generate discriminative feature embeddings in an end-to-end learning pipeline, removing the need of integrating conventional discriminative models.

3. DTT allows dense prediction to obtain both the location and bounding box of the target object in an robust way, simplifying the pipeline of discriminative tracking framework.

4. Experimental results show that DTT is comparable with state-of-the-art trackers without bells and whistles. We hope this effective and efficient method could provide a new perspective for visual tracking.

2. Related Work

Siamese Network Based Trackers Recently, Siamese network based trackers have attracted great attention from the visual tracking community due to their end-to-end training capabilities and high efficiency [1, 5, 12, 13, 19, 20, 36, 37, 39]. SiamFC [1] employs a fully-convolutional Siamese network to extract the feature maps of target. It can run at high speed because of its light-weight structure and no need to update. In order to get a more accurate target bounding box, SiamRPN [20] introduces the region proposal subnetwork into the SiamFC instead of the multi-levels scale search strategy. SiamRPN++ [19] and SiamDW [37] alleviate the negative influence such as padding by different methods, and introduce modern deep neural networks like ResNet [14] into the Siamese network based trackers. More recently, SiamBAN [5] and SiamCAR [13] employ fully convolutional network to directly classify objects and regress their bounding boxes at each spatial location, removing the tricky hyper-parameter tuning of anchors. However, Siamese network based trackers are typically limited to the mechanism of cross correlation and thus cannot

exploit background information effectively to discriminate target from distractors or cluttered scenes. Differently, our tracker is able to encode rich scene information to generate the discriminative feature embeddings.

Online Discriminative Trackers Online discriminative trackers [2, 6, 7, 23, 28, 29, 31, 40, 42] are prevalent over the last decades, because they can effectively exploit the background information and achieve the state-of-the-art results on multiple challenging benchmarks [15, 17, 18, 35]. Recently, DiMP [2] and DCFST [42] present two end-to-end trainable architectures which integrate the discriminative model predictor into the offline training to learn optimal features for the discriminative model. Our approach follows the ideas of utilizing the background information and the end-to-end framework. However, we do not assume the discriminative embeddings learned via a least-squares based regression is ideal and discriminative enough for visual tracking task. Differently, we propose to directly generate discriminative feature embeddings and make them adapt to the visual tracking task in an end-to-end learning pipeline. Moreover, our discriminative feature embeddings can not only contribute to robust localization but also be used for bounding box regression. More related to our work, KYS [3] also exploits the scene information by building state vectors and its encoding is also trained end-to-end by minimizing a tracking loss. However, the receptive field of each state vector and the computation of dense correspondence are local and limited, causing that the accuracy drops severely without an extra appearance model. Different from KYS, our DTT is neat, straightforward and efficiency.

Attention for Image Recognition. The self-attention mechanism used in Transformers [32] correlates information for each element of the input with respect to the others. Recently, Transformer based architectures have been applied to various tasks such as object detection [4]. We successfully adapt the transformer architecture to single-object visual tracking scheme and use it to learn strong discriminative feature embeddings for both classification and bounding box regression .

3. Proposed Method

3.1. Transformer Encoder for Discriminative Tracking

Given training samples and their corresponding labels, conventional discriminative methods such as least-squares-based regression aim to learn a discriminative model which can be used to discriminate between target and background appearance in the feature space, by means of minimizing a discriminative loss. However, due to the limitation of conventional discriminative methods, the solved models may not capture all discriminative representations of data that may be beneficial to visual tracking, *e.g.*, detailed texture features and relationships among the distractors in the scene. Differently, in this work, we point out that the discriminative tracking task can be accomplished in a straightforward way without optimizing a discriminative model, if the feature embeddings of training image themselves contain enough discriminative representations for robust tracking and can be utilized effectively in online tracking. To implement this, we utilize the Transformer architecture [32].

To be specific, we employ multi-head self-attention mechanism to refine the feature embeddings of each element with consideration to all the other elements. The powerful relation modeling capability of Transformers enables the output feature embeddings to contain much more and stronger discriminative representations compared with the original convolutional features. Formally, we let query $\mathbf{Q} \in \mathcal{R}^{HW \times C}$, key $\mathbf{K} \in \mathcal{R}^{HW \times C}$, and value $\mathbf{V} \in \mathcal{R}^{HW \times C}$ denote the input triplets of the self-attention module, where H, W and C denote height, width and channel number of the input convolutional features, respectively. We additionally introduce the fixed positional embeddings $\mathbf{Y} \in \mathcal{R}^{HW \times C}$ as done in DETR [4] to disambiguate different spatial positions. Then, as shown in Fig. 2, given the reshaped training image features $\mathbf{X} \in \mathcal{R}^{HW \times C}$, we have $\mathbf{Q} = \mathbf{X} + \mathbf{Y}$, $\mathbf{K} = \mathbf{X} + \mathbf{Y}$, and $\mathbf{V} = \mathbf{X}$. Finally, the discriminative feature embeddings $\mathbf{F} \in \mathcal{R}^{HW \times C}$ are obtained through the standard multi-head self-attention layer and the feed-forward networks (FFNs) in the encoder, with the whole process defined as $\mathbf{F} = \text{Enc}(\mathbf{X})$. Through end-to-end training on large-scale tracking datasets, the discriminative feature embeddings are tightly coupled with the tracking task and able to highlight the most discriminative representation for visual tracking. Note that all the training images are cropped and centered at the target object to make the networks recognize the location of target object and benefit the training process.

3.2. Transformer Decoder for Discriminative Tracking

The decoder in DTT also has a standard Transformer architecture. Each layer consists of a multi-head self-attention module, a multi-head cross-attention one and FFNs. Different from the common settings in detection task [4], our approach receives the test image features $\mathbf{Z} \in \mathcal{R}^{HW \times C}$ as one input instead of the learned query embeddings [4]. In order for the Transformer model to make use of the positional information of the test image, as done in the encoder, fixed positional encodings are added to the test image features for the input key and query elements of the self-attention module. As such, through the self-attention module, the local and global information about the test image is learned and contained in the feature embeddings of each pixel, enabling dense prediction at all spatial positions of the test image

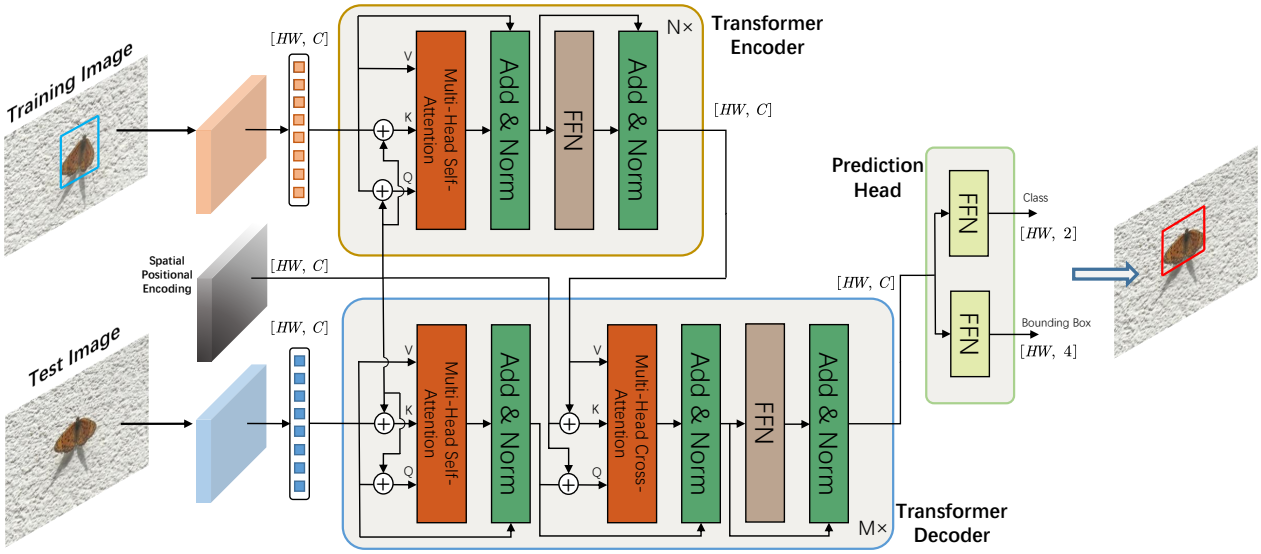


Figure 2: The detailed architecture of the proposed DTT. Features of training image and test image are fed to the encoder and the decoder, respectively. The prediction head consists of a classification branch and a bounding box regression branch.

features for more accurate and robust tracking. Note that in the dense prediction of previous trackers [5, 13], the global scene information about the test image can not contribute to each prediction, resulting in inferior generalization, especially in cluttered scenes.

The following cross-attention module provides an effective manner to utilize the discriminative embeddings $\mathbf{F}$, where key and query are $\mathbf{F}$ and the output of the ahead self-attention module, respectively, and supplemented with the positional embeddings as well (see Fig. 2). Formally, the output of the decoder, called cross-attention feature embeddings $\mathbf{R} \in \mathcal{R}^{HW \times C}$, is obtained by $\text{Dec}(\mathbf{Z}, \mathbf{F})$. Compared with the response map with only one channel in previous discriminative methods [2, 6, 31], the cross-attention feature embeddings contain more discriminative representation about the training and test images and can be used for both target localization and bounding box regression.

3.3. Bounding Box Prediction

Since the cross-attention feature embeddings have the same spatial size as the test image features, each spatial location (i, j) in $\mathbf{R}$ can be mapped to the corresponding location (m, n) in test image by $(m, n) = (i \times s - \lfloor s/2 \rfloor, j \times s - \lfloor s/2 \rfloor)$ directly, where s denotes the stride of the networks. Similar to previous works [13, 30], our prediction head consists of a classification branch to predict the category (foreground or background) for each location, and a regression branch to compute the target bounding box at this location. Formally, the final prediction is obtained by

$$\begin{aligned} \mathbf{P}^{\text{cls}}_{W \times H \times 2} &= \varphi^{\text{cls}}(\mathbf{R}), \\ \mathbf{P}^{\text{reg}}_{W \times H \times 4} &= \varphi^{\text{reg}}(\mathbf{R}), \end{aligned} \tag{1}$$

where $\varphi^{\text{cls}}(\cdot)$ and $\varphi^{\text{reg}}(\cdot)$ denote the FFNs for classification and bounding box regression, respectively. The 2-D vector at each spatial position, *i.e.*, $\mathbf{P}^{\text{cls}}_{i,j} = (p_f, p_b)$, represents the foreground and background scores of the corresponding location in the test image. The 4-D vector at each spatial position, *i.e.*, $\mathbf{P}^{\text{reg}}_{i,j} = (l, t, r, b)$, represents the distances from the corresponding location to the four sides of the bounding box in the test image. Then, the predicted bounding box $\hat{\mathbf{b}}_{(i,j)} = (\hat{x}, \hat{y}, \hat{w}, \hat{h})$ is given by

$$\begin{aligned} \hat{x} &= m + (r - l)/2, \quad & \hat{y} &= n + (b - t)/2, \\ \hat{w} &= l + r, & \hat{h} &= t + b, \end{aligned} \tag{2}$$

where $(\hat{x}, \hat{y})$, $\hat{w}$ and $\hat{h}$ denote the center, the width and the height of the predicted bounding box.

3.4. Training Loss

The training loss consists of the bounding box regression loss and the classification loss. For the regression branch, since the locations far away from the center of the target object tend to produce low-quality predicted bounding boxes [13], the corresponding predictions do not contribute to

Table 1: Comparisons with the state-of-the-art trackers on GOT-10k [15].

Tracker	Venue	Backbone	AO	$SR_{0.5}$	$SR_{0.75}$	FPS	Hardware
MDNet [25]	CVPR'16	VGG-m	0.299	0.303	0.099	1.52	Titan X
CCOT [8]	ECCV'16	VGG-m	0.325	0.328	0.107	0.68	CPU
SiamFC [1]	CVPR'16	AlexNet	0.374	0.404	0.144	25.81	Titan X
CFNet [31]	CVPR'17	AlexNet	0.293	0.265	0.087	35.62	Titan X
ECO [7]	CVPR'17	VGG-m	0.316	0.309	0.111	2.62	CPU
ATOM [6]	CVPR'19	ResNet-18	0.556	0.634	0.402	20.71	GTX-1050
SiamRPN++ [19]	CVPR'19	ResNet-50	0.517	0.616	0.325	49.83	RTX-2080
DiMP [2]	ICCV'19	ResNet-50	0.611	0.717	0.492	34.05	GTX-1050
D3S [22]	CVPR'20	ResNet-50	0.597	0.676	0.462	25	GTX-1080
SiamCAR [13]	CVPR'20	ResNet-50	0.579	0.677	0.437	52.27	RTX-2080
Ocean [38]	ECCV'20	ResNet-50	0.611	0.721	-	25	V100
KYS [3]	ECCV'20	ResNet-50	0.636	0.751	0.515	20	RTX-2080
DCFST [42]	ECCV'20	ResNet-50	0.638	0.753	0.498	25	Titan Xp
DTT	ours	ResNet-50	0.634	0.749	0.514	54.5	Titan Xp
DTT*	ours	ResNet-50	0.689	0.798	0.622	54.5	Titan Xp

the regression loss if the locations fall outside the elliptic area centered at the target object, that is the locations fall inside/outside the elliptic area are regarded as foreground/background. Formally, similar to [13], we compute the regression loss by using

$$\mathcal{L}_{\text{reg}} = \frac{1}{\sum \mathbb{I}(\mathbf{b}_{(i,j)})} \sum_{i,j} \mathbb{I}(\mathbf{b}_{(i,j)})(1 - \text{IoU}(\mathbf{b}_{(i,j)}, \hat{\mathbf{b}}_{(i,j)})), \tag{3}$$

where $\mathbf{b}_{(i,j)} = (x, y, w, h)$ denotes the center, width and height of the ground-truth bounding box and $\text{IoU}(\cdot)$ is a function which computes the area ratio of intersection to union of the predicted bounding box and the ground-truth bounding box. The function $\mathbb{I}(\cdot)$ is defined by

$$\mathbb{I}(\mathbf{b}_{(i,j)}) = \begin{cases} 1, & \frac{4(x-\hat{x})^2}{w^2} + \frac{4(y-\hat{y})^2}{h^2} < 1 \\ 0, & otherwise. \end{cases}, \tag{4}$$

For another branch, we apply the cross-entropy loss [13, 30] $\mathcal{L}_{\text{cls}}$ for classification. The overall training loss of DTT is

$$\mathcal{L} = \lambda_1 \mathcal{L}_{\text{cls}} + \lambda_2 \mathcal{L}_{\text{reg}}, \tag{5}$$

where λ_1 and λ_2 are the tradeoff hyper-parameter. During training, we empirically set $\lambda_1 = \lambda_2 = 1$ in our experiments.

3.5. Online Tracking

Initialization Given the first frame with annotation, the image patch is cropped and centered at the ground-truth target, with an area of 4^2 times the target area, and then fed to the encoder to obtain the initial discriminative embeddings $\mathbf{F}_0$.

Prediction Given a new test image, we first calculate the predictions $\mathbf{P}^{\text{cls}}$ and $\mathbf{P}^{\text{reg}}$ by Eq. (1). Then, the location with the highest foreground score is selected as (i^*, j^*). Finally, the predicted bounding box $\hat{\mathbf{b}}_{(i^*,j^*)}$ is computed by Eq. (2).

Updating In online tracking, to make our tracking model robust to the variations of the target and the background, we update the discriminative feature embeddings every 10 frames in a moving average method. First, we crop the image patch centered at the predicted target and obtain the new discriminative embeddings in the current frame $\mathbf{F}_t$ by the encoder. Then, we empirically update the discriminative feature embeddings $\tilde{\mathbf{F}}$ as follows,

$$\tilde{\mathbf{F}} = (1 - \gamma)\tilde{\mathbf{F}} + \gamma \mathbf{F}_t, \tag{6}$$

where γ is a weight parameter.

4. Experiments

4.1. Implementation Details

Feature Extraction The search regions of our tracker is set experientially 4 times larger than the object bounding box [13, 19]. The cropped training images and test ones are both first resized to 255×255, and then fed to the feature extraction network where ImageNet [9] pre-trained ResNet-50 [14] is adopted as the backbone network. To obtain detailed spatial information, we use atrous convolution with the stride of 1 and the atrous rate of 2 in the $conv4$ block. Then, for efficiency, we *only use* the $conv4$ block features and add a 1×1 convolution to reduce the output features channel to 256 without utilizing the $conv3$ block and $conv5$ block features which may improve the performance further. Finally, to fit the format of input for the encoder and decoder, both training image features and test image features are reshaped to the size of $HW \times C$ and then denoted as $\mathbf{X}$ and $\mathbf{Z}$, respectively.

Training Details The layers of the encoder and decoder are set to 2, *i.e.*, $M = N = 2$. Other hyper-parameter

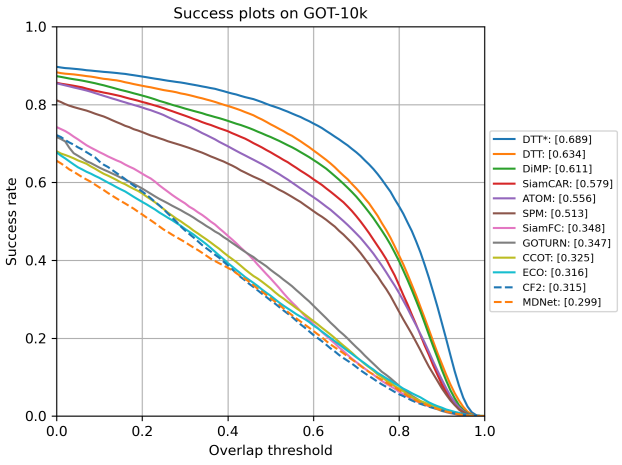


Figure 3: Success plots on GOT-10k [15]. Our DTT outperforms the other state-of-the-art methods.

settings follow DETR [4]. For efficiency, we train DTT only with the training splits of the GOT-10k [15] dataset for 20 epochs. DTT is trained with stochastic gradient descent (SGD) with a minibatch of 20 image pairs following the settings in [13]. Without considering the training efficiency too much, we also train DTT with large-scale training datasets (GOT-10k [15], TrackingNet [24], LaSOT [10] and COCO [21]) for much larger epochs (400), denoted as DTT*. Both DTT and DTT* are implemented in Python using PyTorch on TITAN X (Pascal) GPUs. Code will be released soon.

Tracking Details The weight parameter γ in Eq. (6) is set as 0.01. The running speed of DTT is over 50 FPS, tested on a single TITAN X (Pascal) on GOT-10k test dataset.

4.2. Results on GOT-10K

GOT-10K [15] is a large-scale and high-diversity benchmark for generic object tracking in the wild. Fair comparisons are ensured with the protocol because all approaches use the same training and testing data provided by the dataset. The evaluation metrics include success plots, average overlap (AO), success rate exceeding 0.5 ($SR_{0.75}$) and success rate exceeding 0.75 ($SR_{0.75}$).

We compare DTT and DTT* with state-of-the-art trackers. All the results are provided by the official website of GOT-10K. The quantitative results on different metrics are listed in Table 1 and the success plots are shown in Fig. 3. It can be seen that DTT outperforms all the Siamese Network based trackers in terms of all metrics. Note that though DTT has a similar prediction head to that of SiamCAR [13], it surpasses SiamCAR significantly by 5.5%, 7.2% and 7.7% in terms of AO, $SR_{0.5}$ and $SR_{0.75}$, respectively, which confirms that DTT is able to exploit scene information effectively for visual tracking. Compared with DiMP, DTT improves the scores by 2.3%, 3.2%, and 2.2%, respectively for AO, $SR_{0.5}$ and $SR_{0.75}$, showing the stronger discrimination of our method. Note that KYS outperforms DTT because KYS employs an extra appearance model. Compared with DTT, DTT* improves the scores by 5.5%, 4.9%, and 10.8% respectively for AO, $SR_{0.5}$ and $SR_{0.75}$ with the help of more training data and epochs. Moreover, both DTT and DTT* can run at the highest speed of 54.5 FPS among these trackers, 2× the speed of DCFST and KYS.

Table 2: Comparisons with the state-of-the-art trackers on LaSOT [10].

	MDNet [25]	ATOM [6]	Ocean [38]	DiMP50 [2]	SiamCAR [13]	DTT ours	DTT* ours
AUC	0.422	0.537	0.560	**0.568**	0.516	0.538	**0.601**

Table 3: Comparisons with the state-of-the-art trackers on NfS [16].

	KYS [3]	SiamRCNN [33]	DCFST [42]	DiMP [2]	SiamBAN [5]	DTT ours	DTT* ours
AUC	0.635	0.639	**0.641**	0.620	0.594	0.608	**0.659**

4.3. Results on LaSOT

LaSOT [10] is a large-scale benchmark for long-term single-object tracking. The test set consists of 280 high-quality sequences. The results are presented in Table 2. DTT and DTT* obtain AUC scores of 0.538 and 0.601, respectively. DTT* outperforms the previous trackers by over 3%, showing effectiveness of our way to utilize the rich scene information with a brand new approach.

4.4. Results on NfS

We evaluate our approach on the 30 FPS version of NfS dataset [16] which consists of 100 challenging videos. We compare DTT and DTT* with other five state-of-the-art trackers, including KYS [3], SiamRCNN [33], DCFST [42], DiMP [2], and SiamBAN [5]. Area-under-the-curve (AUC) scores are shown in Table 4. It can be seen that DTT achieves an AUC score of 0.608, outperforming the recent Siamese Network based tracker SiamBAN [5] by 1.4%. With multiple training datasets and more training epochs, DTT* obtains the highest AUC score of 0.659, leading KYS by 2.4%.

4.5. Results on TrackingNet

TrackingNet is another large-scale dataset for training and testing trackers, where the the test set contains 511 sequences. We compare DTT and DTT* against recent state-of-the-art trackers, including KYS [25], SiamRCNN [33], DCFST [42], DiMP [2], and SiamRPN++ [19] on the test set of TrackingNet. Though DTT is only trained on GOT-

Table 4: Comparisons with the state-of-the-art trackers on TrackingNet [24]. Trackers are evaluated by using the area under the success rate curve (AUC), precision and normalized precision (Norm. Prec).

	KYS [3]	SiamRCNN [33]	DCFST [42]	DiMP [2]	SiamRPN++ [19]	DTT ours	DTT* ours
Precision	0.688	**0.800**	0.700	0.687	0.694	0.688	**0.789**
Norm. Prec	0.800	**0.854**	0.809	0.801	0.800	0.803	**0.850**
AUC	0.740	**0.812**	0.752	0.740	0.733	0.740	**0.796**

Table 5: Analysis of different architectures for integrating Transformers into the tracking scheme on GOT-10k test dataset.

	Baseline	Concatenation	Siamese	Query-key	DTT
AO	0.362	0.498	0.582	0.556	**0.634**
$SR_{0.50}$	0.393	0.598	0.689	0.678	**0.749**
$SR_{0.75}$	0.065	0.236	0.433	0.376	**0.514**

10k [15], it achieves the best normalized precision of 0.803 and the AUC score of 0.740, similar to those of DiMP, KYS and SiamRPN++. DTT* obtains the second best results, only inferior to SiamRCNN because the re-detection module of SiamRCNN, which is not employed in other methods including ours, can work better when the class number is small, *e.g.*, 27 on TrackingNet..

4.6. Analysis of Integration of Transformers

Aiming at finding out an effective way to integrate transformers into online discriminative tracking, we investigate different network architectures for integrating Transformers in single-object visual tracking. All the experiments are based on their respective retrained networks (including the ResNet backbone) and the training strategy is the same as DTT. The results of this investigation are shown in table 5. The simplified architectures of the other four architectures are presented in Fig. 4.

Baseline Motivated by the original structure in DETR [4], the baseline structure only uses the test image which is fed to the encoder. The decoder in this baseline receives 100 learned object queries as input as done in DETR [4]. Since the target to be tracked is arbitrary, such architecture cannot determine what object is the target, and thus the results are poor, confirming that the detection framework cannot be employed in visual tracking directly.

Concatenation We naively add the training image to the architecture by concatenating the features of training image and test image along the channel dimension in the baseline architecture. This leads to better results, with an improvement of 13.6% and 17.1% in terms of AO and $SR_{0.75}$, respectively, showing the importance of scene information in the training image.

Siamese In this architecture, the features of training image and test image are fed to the same encoder, respectively. Then the two output embeddings are fed to the cross-attention module in the decoder which does not contain self-attention module. The results are further improved by 8.4% and 9.1% in terms of AO and $SR_{0.75}$, respectively. Compared with DTT, the Siamese architecture cannot exploit the discriminative representation in the scene thoroughly due to shared parameters in the encoder.

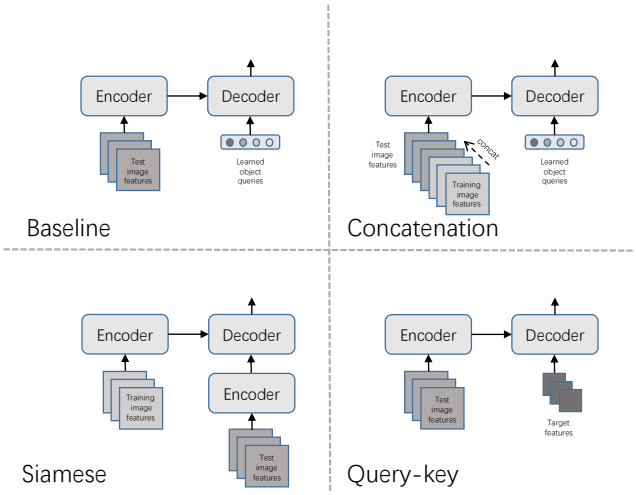


Figure 4: The four different simplified architectures for integrating Transformers in single-object visual tracking. Details can be found in Sec. 4.6.

Table 6: Comparisons of the effectiveness of training datasets and training epochs on GOT-10k test dataset.

	DTT	DTT-L	DTT*
Multiple training datasets			✓
Training epochs	20	400	400
AO	0.634	0.659	**0.689**

Table 7: Component-wise analysis of DTT on GOT-10k test dataset. The results prove that each component is important in our method. (BI: background information. PE: positional embeddings. OU: online update.)

	w/o OU	w/o BI	w/o PE in encoder	w/o PE in decoder	DTT
AO	0.621	0.559	0.576	0.616	**0.634**
$SR_{0.50}$	0.735	0.657	0.696	0.728	**0.749**
$SR_{0.75}$	0.495	0.402	0.398	0.490	**0.514**

Query-key This architecture is also mentioned in [27], where the test image features are fed to the encoder and the target features are regarded as the input object query of the decoder. The results are worse than the Siamese method and DTT, verifying that only the target information cannot help achieve satisfactory results during online tracking.

Different from the above heuristic ways, inspired by the modern discriminative tracking pipeline, DTT is able to exploit rich scene information effectively and highlight the discriminative representation in the training image, achieving the best results among these architectures.

4.7. Component-wise Analysis

To verify the effectiveness of the proposed method, we perform a component-wise analysis on the GOT-10k test dataset.

Training Strategy To verify the effectiveness of multiple training datasets and more training epochs respectively, we also train DTT which is only trained on GOT-10k with 400 epochs, denoted as DTT-L. The comparisons are shown in Table 6. Trained with more epochs, DTT-L obtains a gain of 2.5% in AO. Compared with DTT-L, DTT* is trained with more training datasets including GOT-10k, LaSOT, TrackingNet and COCO, and obtains a significant gain of 3% in AO.

Online Update We investigate the impact of the online update (detailed in Sec. 3.5). It can be seen from Table 7 that AO, $SR_{0.5}$ and $SR_{0.75}$ drop by 1.3%, 1.4% and 1.9% respectively without employing online update. This verifies the effectiveness of our simple updating method. More complex updating way has the potential ability to exploit the temporal information for more robust performance but is not the focus of this work.

Background Information We remove the background information by cropping the target image patch with an area of 2^2 times the target area in the training image, as done in Siamese Network based trackers [1, 13, 19]. Table 7 shows the results drop severely since only the target appearance cannot help discriminative between target and distractors.

Positional Embeddings In this work, we use the fixed positional embeddings following the settings in DETR [4] and it is well-known that positional embeddings can disambiguate different spatial positions in Transformers. The representation features of training image and test image are added by the same positional embeddings for the input query and key element in Transformers. We can see from Table 7 that removing those in encoder yields drop of 5.8% in AO while removing those in decoder yields drop of only 1.8% in AO, showing that the positional embeddings in encoder are more important than those in decoder. We consider this is because the positional embeddings in encoder also contribute to recognizing the localization of the target object in the training image during the training phase.

4.8. Qualitative Results

To visualize the localization and bounding box regression quality of DTT in online tracking, we show the tracking results of DTT, DiMP [2] and SiamCAR [13] on the challenging sequences from GOT-10k [15] in Fig. 5. Three frames of *GOT-Test-004*, *GOT-Test-018*, *GOT-Test-037*, and *GOT-Test-063* sequences are shown in the figures. It can be seen that the target objects are able to be predicted robustly by DTT when undergoing radical variations, *e.g.*, rotation in *GOT-Test-004* and *GOT-Test-018*. Note that in cluttered scenes such as *GOT-Test-037* and *GOT-Test-063*, DTT will not be impacted negatively by the distractors with the help of scene information while the other two representative tracking methods, DiMP and SiamCAR, tend to drift in these scenes.

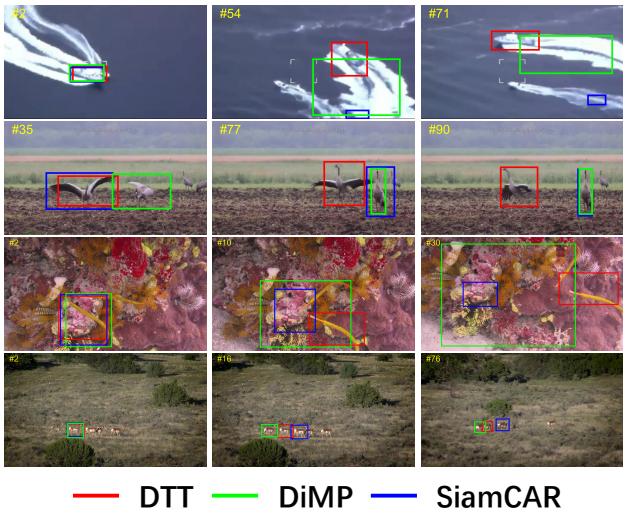


Figure 5: Visualization tracking results of DTT, DiMP [2] and SiamCAR [13] on the challenging sequences from GOT-10k [15]. We can see that DTT shows stronger generalization ability and better accuracy throughout tracking. Best viewed with zooming in.

5. Conclusion and Future Work

In this work, we depart from the two popular tracking schemes, and present a brand new discriminative tracking approach, namely DTT, which is based on an encoder-decoder Transformer architecture. DTT is able to exploit the rich scene information for robust tracking. We use a learning pipeline of achieving the discriminative embeddings which are able to highlight the most discriminative representation for visual tracking, removing the need for conventional discriminative models. Besides, the discriminative embeddings can be used for both localization and bounding box regression, simplifying the previous discriminative tracking pipeline. Without any tricks, our method achieves state-of-the-art performance on four benchmarks at a high speed of over 50 FPS, showing the potential ability of this novel tracking approach. Moreover, we believe that our approach is complementary to more sophisticated online updating methodologies, and expect future work to explore spatio-temporal scene information more thoroughly.

Acknowledgements. This work was supported by the Key-Areas Research and Development Program of Guangdong Province (No. 2020B010165001). This work was also supported by National Natural Science Foundation of China under Grants 61772527, 61976210, 62076235, and 62002356.

References

[1] Luca Bertinetto, Jack Valmadre, Joao F Henriques, Andrea Vedaldi, and Philip HS Torr. Fully-convolutional siamese networks for object tracking. In *European conference on computer vision*, pages 850–865. Springer, 2016. 1, 2, 5, 8

[2] Goutam Bhat, Martin Danelljan, Luc Van Gool, and Radu Timofte. Learning discriminative model prediction for tracking. In *ICCV*, pages 6182–6191, 2019. 1, 3, 4, 5, 6, 7, 8

[3] Goutam Bhat, Martin Danelljan, Luc Van Gool, and Radu Timofte. Know your surroundings: Exploiting scene information for object tracking. In *ECCV*, 2020. 1, 3, 5, 6, 7

[4] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020. 3, 6, 7, 8

[5] Zedu Chen, Bineng Zhong, Guorong Li, Shengping Zhang, and Rongrong Ji. Siamese box adaptive network for visual tracking. *arXiv preprint arXiv:2003.06761*, 2020. 1, 2, 4, 6

[6] Martin Danelljan, Goutam Bhat, Fahad Shahbaz Khan, and Michael Felsberg. Atom: Accurate tracking by overlap maximization. In *CVPR*, pages 4660–4669, 2019. 1, 3, 4, 5, 6

[7] Martin Danelljan, Goutam Bhat, Fahad Shahbaz Khan, and Michael Felsberg. Eco: Efficient convolution operators for tracking. In *CVPR*, pages 6638–6646, 2017. 1, 3, 5

[8] Martin Danelljan, Andreas Robinson, Fahad Shahbaz Khan, and Michael Felsberg. Beyond correlation filters: Learning continuous convolution operators for visual tracking. In *ECCV*, pages 472–488. Springer, 2016. 5

[9] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 5

[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 *CVPR*, pages 5374–5383, 2019. 2, 6

[11] Heng Fan and Haibin Ling. Siamese cascaded region proposal networks for real-time visual tracking. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 1

[12] Junyu Gao, Tianzhu Zhang, and Changsheng Xu. Graph convolutional tracking. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 1, 2

[13] D. Guo, J. Wang, Y. Cui, Z. Wang, and S. Chen. Siamcar: Siamese fully convolutional classification and regression for visual tracking. In *CVPR*, 2020. 1, 2, 4, 5, 6, 8

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 2, 5

[15] Lianghua Huang, Xin Zhao, and Kaiqi Huang. Got-10k: A large high-diversity benchmark for generic object tracking in the wild. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019. 1, 2, 3, 5, 6, 7, 8

[16] Hamed Kiani Galoogahi, Ashton Fagg, Chen Huang, Deva Ramanan, and Simon Lucey. Need for speed: A benchmark for higher frame rate object tracking. In *ICCV*, pages 1125–1134, 2017. 1, 2, 6

[17] Matej Kristan, Ales Leonardis, Jiri Matas, Michael Felsberg, Roman Pflugfelder, Luka Cehovin Zajc, Tomas Vojir, Gustav Hager, Alan Lukezic, Abdelrahman Eldesokey, et al. The visual object tracking vot2017 challenge results. In *ICCV*, pages 1949–1972, 2017. 3

[18] Matej Kristan, Ales Leonardis, Jiri Matas, Michael Felsberg, Roman Pfugfelder, Luka Cehovin Zajc, Tomas Vojir, Goutam Bhat, Alan Lukezic, Abdelrahman Eldesokey, Gustavo Fernandez, and et al. The visual object tracking vot2018 challenge results. In *ECCV*, 2018. 3

[19] 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, 5, 6, 7, 8

[20] 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

[21] 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 *European conference on computer vision*, pages 740–755. Springer, 2014. 6

[22] Alan Lukezic, Jiri Matas, and Matej Kristan. D3s-a discriminative single shot segmentation tracker. In *CVPR*, pages 7133–7142, 2020. 5

[23] Chao Ma, Jia-Bin Huang, Xiaokang Yang, and Ming-Hsuan Yang. Hierarchical convolutional features for visual tracking. In *ICCV*, pages 3074–3082, 2015. 1, 3

[24] 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 *ECCV*, pages 300–317, 2018. 2, 6, 7

[25] Hyeonseob Nam and Bohyung Han. Learning multi-domain convolutional neural networks for visual tracking. In *CVPR*, pages 4293–4302, 2016. 5, 6

[26] Chong Sun, Dong Wang, Huchuan Lu, and Ming-Hsuan Yang. Learning spatial-aware regressions for visual tracking. In *CVPR*, pages 8962–8970, 2018. 1

[27] Peize Sun, Yi Jiang, Rufeng Zhang, Enze Xie, Jinkun Cao, Xinting Hu, Tao Kong, Zehuan Yuan, Changhu Wang, and Ping Luo. Transtrack: Multiple-object tracking with transformer. In *arXiv preprint arXiv:2012.15460*, 2020. 7

[28] Ming Tang, Bin Yu, Fan Zhang, and Jinqiao Wang. High-speed tracking with multi-kernel correlation filters. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4874–4883, 2018. 3

[29] Ming Tang, Linyu Zheng, Bin Yu, and Jinqiao Wang. Fast kernelized correlation filter without boundary effect. In *WACV*, pages 2999–3008. 3

[30] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *ICCV*, 2019. 4, 5

[31] Jack Valmadre, Luca Bertinetto, João Henriques, Andrea Vedaldi, and Philip HS Torr. End-to-end representation learning for correlation filter based tracking. In *CVPR*, pages 2805–2813, 2017. 3, 4, 5

[32] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, 2017. 1, 3

[33] Paul Voigtlaender, Jonathon Luiten, Philip HS Torr, and Bastian Leibe. Siam r-cnn: Visual tracking by re-detection. In *CVPR*, pages 6578–6588, 2020. 1, 6, 7

[34] 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. 1

[35] Yi Wu, Jongwoo Lim, and Ming-Hsuan Yang. Object tracking benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9):1834–1848, 2015. 1, 3

[36] Yuechen Yu, Yilei Xiong, Weilin Huang, and Matthew R Scott. Deformable siamese attention networks for visual object tracking. In *CVPR*, 2020. 1, 2

[37] Zhipeng Zhang and Houwen Peng. Deeper and wider siamese networks for real-time visual tracking. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 1, 2

[38] Zhipeng Zhang and Houwen Peng. Ocean: Object-aware anchor-free tracking. In *ECCV*, 2020. 5, 6

[39] Linyu Zheng, Yingying Chen, Ming Tang, Jinqiao Wang, and Hanqing Lu. Siamese deformable cross-correlation network for real-time visual tracking. *Neurocomputing*, pages 36–47, 2020. 2

[40] Linyu Zheng, Linyu Tang, and Jinqiao Wang. Learning robust gaussian process regression for visual tracking. In *IJCAI*, pages 1219–1225, 2018. 3

[41] Linyu Zheng, Ming Tang, Yingying Chen, Jinqiao Wang, and Hanqing Lu. Fast-deepkcf without boundary effect. In *ICCV*, October 2019. 1

[42] Linyu Zheng, Ming Tang, Yingying Chen, Jinqiao Wang, and Hanqing Lu. Learning feature embeddings for discriminant model based tracking. In *ECCV*, 2020. 1, 3, 5, 6, 7