Target-Aware Tracking with Long-Term Context Attention

Kaijie He[1]
Canlong Zhang[1,2*], Sheng Xie[1], Zhixin Li[1,2], Zhiwen Wang[3]
[1]School of Computer Science and Engineering, Guangxi Normal University, China
[2]Guangxi Key Lab of Multi-source Information Mining and Security, China
[3]School of Computer Science and Technology, Guangxi University of Science and Technology, China
hekaijie123@outlook.com, clzhang@gxnu.edu.cn

Abstract

Most deep trackers still follow the guidance of the siamese paradigms and use a template that contains only the target without any contextual information, which makes it difficult for the tracker to cope with large appearance changes, rapid target movement, and attraction from similar objects. To alleviate the above problem, we propose a long-term context attention (LCA) module that can perform extensive information fusion on the target and its context from long-term frames, and calculate the target correlation while enhancing target features. The complete contextual information contains the location of the target as well as the state around the target. LCA uses the target state from the previous frame to exclude the interference of similar objects and complex backgrounds, thus accurately locating the target and enabling the tracker to obtain higher robustness and regression accuracy. By embedding the LCA module in Transformer, we build a powerful online tracker with a target-aware backbone, termed as TATrack. In addition, we propose a dynamic online update algorithm based on the classification confidence of historical information without additional calculation burden. Our tracker achieves state-of-the-art performance on multiple benchmarks, with 71.1% AUC, 89.3% NP, and 73.0% AO on LaSOT, TrackingNet, and GOT-10k. The code and trained models are available on https://github.com/hekaijie123/TATrack.

Introduction

Visual target tracking is a fundamental computer vision task. Given an initial position of any target, the tracker is required to evaluate the target state in subsequent each frame of a video. Tracking task faces significant challenges such as the variable appearance, fast movement, attraction from similar objects, etc. Siamese structure based trackers have achieved considerable success, which realizes tracking by using twin networks to represent the target and search image and calculating their similarity. Although the existing trackers have become more and more complex, most of them still originate from Siamese paradigms.

After carefully investigating the existing Siamese trackers, we found that they have more or less inherited certain simple operation from SiamFC (Bertinetto et al. 2016), and have some drawback as follows: (1) Due to insufficient appearance information, using target template without any background to correlate with the search region will be diffcult to distinguish the real target from the background attractors similar to the target, and also has difficulty in coping with the severe appearance changes; (2) In tracking task, the correlation operation is performed after the backbone network has completely extracted the image features, but the backbone network is originally designed for classification task, so the applicability of the feature extraction to the tracking task is limited to a certain extent. (3) The tracker uses only the optimal model obtained by offline training to predict the target, and the model only knows the target appearance of the initial frame without online updated information. Such a tracker is static and it does not have any perception of the changes that occur in the target state throughout the video sequence. The static tracker lacks perception of the continuous changes in the target and loses robustness in long time tracking. Above drawbacks are not very prominent in simple tracking scenarios, but they will be rapidly enlarged in complex tracking scenarios, so it is necessary to overcome them. Inspired by above three problems, we propose a long-term context attention mechanism that can simultaneously accept a target template, a historical frame and a current search frame as input in an adaptive weighted fusion way. We embed an improved location encoder in the

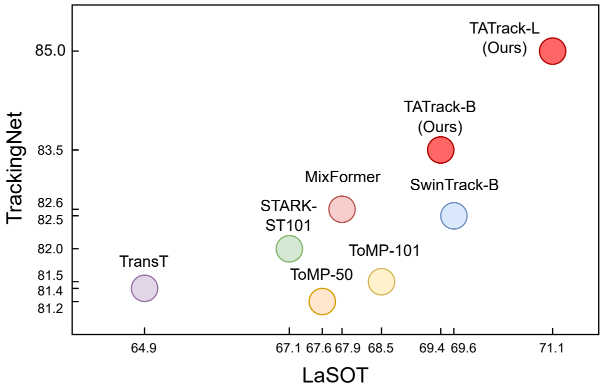

Figure 1: Comparison with other advanced trackers on TrackingNet and LaSOT benchmarks.

[*]Corresponding author

LCA, which enables the target template, the historical frame and the current search frame to perform self-attention calculation while perform cross-attention calculation with each other. The LCA extensively fuses target and background features of images spanning different times and can effectively extract the location information of the target and the state information around the target. Since the LCA module has correlation calculation and feature extraction functions, we alternately stack multiple LCA and SWA (Liu et al. 2021) modules to construct the backbone network suitable for tracking task. With the deepening of layers of the backbone network, the target-aware ability of LCA module will be stronger, that is, the real target will be highlighted while the features of other interfering objects will be weakened. The previous templates are filled with high quality historical frames from the inference process, so we need a reliable quality determination method to select the historical frames. Different from existing online update trackers that use a two-stage inference update approach (Mayer et al. 2022) or a two-stage training network structure (Cui et al. 2022; Yan et al. 2021), we propose a very concise and efficient algorithm that determines whether to update the template according to the classification confidence scores of historical frames, thus achieving high robustness and avoiding large calculational cost like aforementioned two-stage methods. TATracker achieves state-of-the-art performance, shown in Fig. 1.

In summary, our main contributions as follows:

- We propose a new cross-frame attention module suitable for fusion interaction of target and its context.
- Based on LCA, we build a powerful tracker that has a backbone better suited to the tracking task.
- We propose a concise and efficient online updating approach based on classification confidence to select high-quality templates with very low computation burden.
- We evaluate our tracker through comprehensive ablation and comparison experiments, and the experimental results verify its effectiveness and advancement.

Related Work

Tracker Backbone. Deep trackers rely heavily on offline training, and more powerful feature extraction networks can capture deeper semantic information about the target. This feature allows twin network architectures to easily gain more powerful performance from each backbone network upgrade. From the early days of siamFC (Bertinetto et al. 2016), SiamRPN (Li et al. 2018) used AlexNet, then SiamRPN++ (Li et al. 2019) pioneered the use of the more mature backbone network Resnet, to recent years when Transformer backbone networks started to be used in trackers (Lin et al. 2021). In these previous works, the backbone networks used by the tracker were derived from the upstream image classification task, and the direct use of the feature extraction network for the classification task is inefficient for the tracker. The backbone network for the classification task is used to determine the overall category of the image and has no perception of the target and background in the tracking task, which is contrary to the requirement of distinguishing interferers in the tracking task. We propose a target-aware backbone that focuses on the extraction of target features. In addition, we also add auxiliary positioning information as (Zhang, Li, and Wang 2018) fuses multi-feature information.

Online Update. In the Siamese paradigm, the tracker uses the first frame as a template and remains unchanged, and the performance of the tracker depends entirely on the ability to match the appearance of the target. However, the appearance of the target tends to change continuously over time, and models that are not updated have significant bottlenecks. Guided by this, much work has been done to experiment with online updates. The large network structure of the tracker requires long time and large amount of data for training, and the video history information obtained online alone can hardly be used to accurately update the model parameters of the subject. the ATOM (Danelljan et al. 2019) model is designed with mini-localization branches, and the localization branches are trained at each inference. However, mini-branches have significant performance limits and increase inference time, and have not become mainstream. UpdateNet (Zhang et al. 2019) uses a CNN to add the target template and the cumulative template to the current frame template with certain weights, resulting in a template with continuously changing information, but the template will accumulate contamination over time leading to failure. STMtrack (Fu et al. 2021) uses fused information from multiple templates, and updates are taken from the historical template at medium intervals. MixFormer (Cui et al. 2022; Yan et al. 2021)et al. take the training quality branch to score the history frames, which has the disadvantage of secondary training and requires restarting the training quality branch after the training of the main body of the network. ToMP (Mayer et al. 2022) uses two templates and takes the most recent frame that meets the conditions as the template, but the inference phase requires two repetitions to ensure performance. While each of these online update approaches possesses relatively obvious limitations, our approach uses historical information about the classification confidence generated during the tracker inference to achieve a way to update templates online without any additional cost.

Method

Long-Term Contextual Attention (LCA)

In this section, we first introduce the proposed Long-term Contextual Attention (LCA) module, which is designed for integrating the information of target and context from multi-frames. Fig. 2 shows the overview of the LCA module.

LCA is a powerful attention computation module, which can integrate the features from the target template, previous template, and searched image, perceive the real target from the previous template and search image based on the features of the target template, and reinforce features of the target while weakening interference information. At the same time, the LCA can implicitly find the changes in the search image based on the target state including the appearance and relative position in the previous template, to further exclude

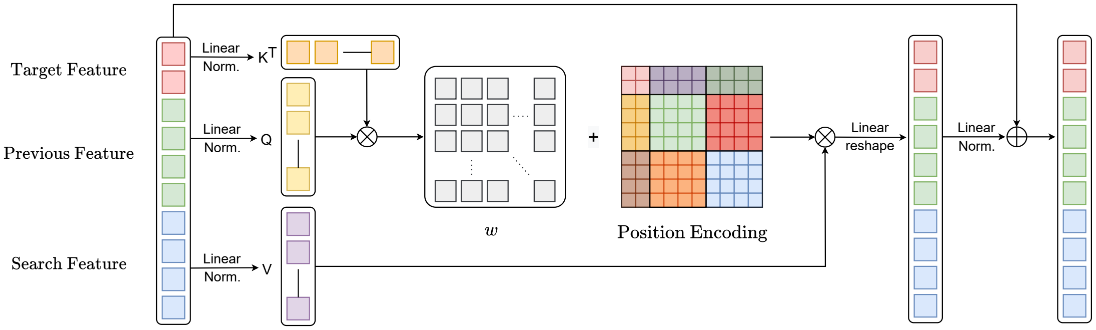

Figure 2: Long-term contextual attention module (LCA) is an efficient multi-image attention operation. It can simultaneously perform feature extracting for each image through self-attention and target searching through cross-attention among images. LCA uses inter-image independent location encoding to divide the attention weight map into TtoT, TtoP, TtoS, PtoT, PtoP, PtoS, StoT, StoP, and StoS from top to bottom and left to right, where T, P, and S represent the target template, the previous template, and the searched image, respectively.

the interfering objects similar to the target, thus more accurately capturing the current target state.

In this module, position encoding plays an important role. We know that in the self-attention formula Eq. 1, the self-attention formula without position encoding is Eq. 2 and the self-attention formula with absolute position encoding is Eq. 3. $x \in \mathbb{R}^{L \times d}$ is the input feature, p is the absolute position encoding, and $W \in \mathbb{R}^{d \times d}$ is the linear transformation matrix. Q, K and V represent the query, the key and value three mapping matrices, which have the same dimensionality. We did not just directly stitch together the target template, the previous template, and the search image as a whole to calculate the self-attention. Because the relationship between the target template, previous template, and search image will not be distinguished in the formula, the three features will be treated as one big image and the model's ability to construct connections between the three will be limited. Therefore, we must make improvements to the location coding.

$$\text{Attention } = \operatorname{Softmax}(w) x W^{V} \tag{1}$$

$$\text{where } w = \frac{1}{\sqrt{d}}(xW^{Q})(xW^{K})^{T} \tag{2}$$

$$w^{Abs} = \frac{\left((x+p)\, W^{Q}\right)\left((x+p)\, W^{K}\right)^{T}}{\sqrt{d}} \tag{3}$$

TUPE (Ke, He, and Liu 2020) proposes untied absolute positional encoding 4 as an alternative to the traditional positional encoding formulation, where U is a linear transformation matrix of learnable absolute positional encoding with dimensionality equal to W. This formulation decouples the feature x from the absolute positional encoding p. p extracts position information with a learnable independent transformation matrix, unlocking the potential of absolute position coding.

$$\begin{aligned} w^{Abs} = & \frac{1}{\sqrt{2d}}\left(xW^{Q}\right)\left(xW^{K}\right)^{T} \\ & + \frac{1}{\sqrt{2d}}\left(pU^{Q}\right)\left(pU^{K}\right)^{T} \end{aligned} \tag{4}$$

We adopt the relative position encoding $r \in \mathbb{R}^{L \times L}$ form used in SWA (Liu et al. 2021) Eq. 5, where the $\mathbf{P}_{i-j}$ is the learnable variable de-indexed according to $i - j$.

$$\mathbf{r}_{ij} = \mathbf{P}_{i-j} \tag{5}$$

We extend the positional encoding designed for a single image to the multi-image case. We first expand the input target template, the previous template, and the features of the search region into a dimension L along the W and H dimension, and concatenate the three together along the L dimension $x = Concat\,(\mathrm{z}, \mathrm{pre}, \mathrm{x})$, and do the same for the absolute position encoding $p = Concat\,(p_z, p_{pre}, p_x)$. Divide the relative position encoding r into $n \times m = 9$ independent regions to compute $\mathbf{P}_{\ i-j}$, as shown by the position encoding in Fig. 2. Finally, we obtain the LCA attention formula as follows Eq. 6.

$$\begin{aligned} \text{LCA } & = \operatorname{Softmax}(w + a + r) x W^{V} \\ w & = \frac{1}{\sqrt{2d}}(xW^{Q})(xW^{K})^{T} \\ a & = \frac{1}{\sqrt{2d}}\left(pU^{Q}\right)\left(pU^{K}\right)^{T} \\ r_{nm,ij} & = \mathbf{P}_{nm,i-j} \end{aligned} \tag{6}$$

Discussions: Why do we introduce additional prior templates? (1) We introduce rich background information through the previous templates to locate the target with the help of the relative position of the previous target in the background. (2) The addition of prior templates enables online updates, and prior templates contain the state of the target at more recent time points for more accurate regression of the target. Why did we not add the background to the target template? Because, the previous template is temporally closer to the current frame, and the initial target is not important to include the background in the target template, and including only the non-updated targets can emphasize the consistency of the tracker's tracking of the target.

Overall

The overall structure of the model is shown in Fig. 3.

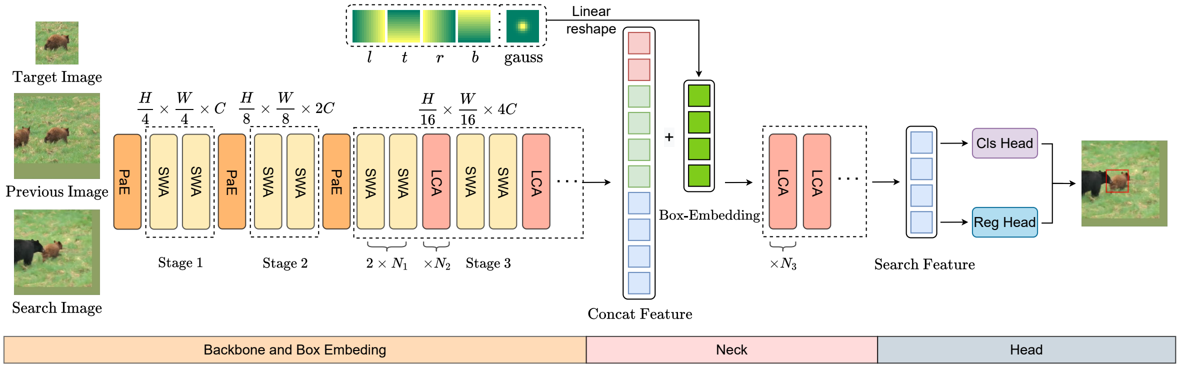

Figure 3: We construct a target-aware backbone network based on alternating stacks of LCA and shift window attention (SWA) module of Swin-Transformer. The target-aware features extracted by the backbone network are further refined by a neck consisting of multiple layers of LCAs to refine the state information of the target. Finally, only the features of the search image part are taken for the feature maps of the regression and classification head.

Backbone. The backbone network accepts the target template $z \in \mathbb{R}^{H_z \times W_z \times 3}$, the previous template $pre \in \mathbb{R}^{H_{pre} \times W_{pre} \times 3}$ and the search image $x \in \mathbb{R}^{H_x \times W_x \times 3}$ of the input. To better match the LCA, we choose the SWA with the same transformer structure to build the feature extraction network. In the first stage of the feature extraction network, we patch and embed the image features into $\frac{H}{4} \times \frac{W}{4} \times C$ feature tokens using a convolution kernel of size 4. The token expands both H, W dimensions into length L in the transformer operation. and then passed through two SWA modules. PaE is used before each subsequent stage, and H, W is halved while C is doubled. In the stage3, we use two SWA modules in a group with LCA modules stacked alternately. We use two SWA modules as a group because two SWA modules complete a window shift and recovery. The target features, previous features and search features are individually passed through PaE and SWA modules in turn, and the three are input into LCA after stitching along the L dimension, and the output of LCA is reduced to three tokens in turn by SWA to extract features. Finally, the sequence of $\frac{H}{16} \times \frac{W}{16} \times 4C$ tokens is obtained from the backbone network.

Neck. In this stage, we add the encoded information about the target location and target size of the previous template. We increase the robustness in training by randomly dithering regions of the previous template, where the target is not always in the center of the image. The previous template contains larger regions other than the target, and the inclusion of Gaussian localization information is necessary to avoid ambiguities induced by similar objects. Also, we adopt a similar approach by adding the ltrb representation (Mayer et al. 2022) of the length and width information to help the model prediction. For the bounding box of the previous frame $b_{pre} = \{b^{x_1}, b^{y_1}, b^{x_2}, b^{y_2}\}$, we first denote each position of the feature map by (k^x, k^y) . Then we get the formula Eq. 7 for the bounding box information $d_{pre} = (l, t, r, b)$ represented by the four sides.

$$\begin{aligned} l = k^x - b^{x_1}/s, \quad r = b^{x_2}/s - k^x, \\ t = k^y - b^{y_1}/s, \quad b = b^{y_2}/s - k^y, \end{aligned} \tag{7}$$

where $s = 16$ is the multiple of image mapping to feature map reduction. $d_{pre} \in \mathbb{R}^{H \times W \times 4}$, ψ is the multilayer perceptron that maps the dimension of d from 4 to C to get the size information embedding of the target. The Gaussian position map $y_{pre} \in \mathbb{R}^{H \times W \times 1}$ mentioned earlier is multiplied by the learnable weight $w \in \mathbb{R}^{1 \times 1 \times C}$ using the broadcast mechanism to get the position information embedding of the previous target. Box-embedding is given by the target size and the target location embedding are directly summed by the following equation:

$$\text{Box-embedding} = w \cdot y_{pre} + \text{MLP}(d_{pre}) \tag{8}$$

Then we successively overlap with multiple LCA modules to further process the information of the target. In the last layer of LCA, we keep only the attention computation of the search features to the target template and the previous template, then pass the search feature map to the classification head and regression head.

Head and Loss. The head network contains two branches: the bounding box regression head and the classification head. The regression map $\mathbb{R}^{H \times W \times 4}$,and the corresponding map $\mathbb{R}^{H \times W \times 1}$ for the prediction are obtained from the search feature map through the three-layer perceptron, respectively. The regression loss uses the common generalized IoU loss (Rezatofighi et al. 2019), and the classification loss we use varifocal loss (Zhang et al. 2021), which uses the currently more popular IoU-aware design. the core idea of IoU-aware is to replace the IoU score with the positive label value of the classification, the traditional positive sample value of the classification is labeled as 1, the improvement relates the regression task to the classification task. varifocal loss is formulated as follows:

$$\text{VFL(p, q)} = \begin{cases} -q(q\log(p) + (1-q)\log(1-p)) & q > 0 \\ -\alpha p^{\gamma}\log(1-p) & q = 0 \end{cases} \tag{9}$$

p is the classification prediction while q is the IoU score of the target. Our total loss function is as follows:

$$\mathcal{L} = \lambda_{\text{cls}}\mathcal{L}_{\text{cls}}\big(p, \text{IoU}(\hat{b}, b)\big) + \lambda_{\text{giou}}\mathcal{L}_{\text{giou}}\left(\hat{b}, b\right) \tag{10}$$

For the weights λ_{cls} is set to 1.5 and λ_{giou} is set to 1.5.

Online Update Strategy

A common way to pick an update template is to retrain a quality judgment branch, but such an approach requires training the model twice. And almost all update approaches set a fixed hyperparameter as a threshold, and update only when the quality confidence is higher than this static threshold. We believe that updating the template should follow two principles, (1) The updated template should be as close to the current frame as possible in time to ensure that the updated template has the most similar state to the current frame. (2) The updated template should be as high quality as possible, with good recognition and accuracy. In this regard, we propose a dynamic thresholding algorithm with a simple classification confidence level to select the ideal historical template. The first algorithm uses the classification confidence historical average as the threshold value, Eq. 11 as follows:

$$mean = \sum_{i}^{n} s_i/n \tag{11}$$

n is the current frame serial number n and s_i is the classification confidence of the ith frame. The mean value as a threshold has a low update criterion and may be selected as close as possible to the historical template of the current frame. This approach shows relatively reliable performance in the got-10k and TrackingNet datasets, but underperforms in the long sequence tracking dataset Lasot. We analyzed the reasons for this; as the target deformation and the environmental changes it faces tend to get more and more complex with increasing time, the model keeps accumulating errors during the tracking process, and even loses the target forever after losing it midway. This is not obvious on got-10k and TrackingNet short series datasets, but long series benchmarks like LaSOT are more likely to encounter prolonged occlusion or target disappearance, and using only lower thresholds will tend to update to the wrong template. Wrong updates will continuously reduce the classification confidence in subsequent tracking, making the threshold of the mean formula invalid. Therefore, we further propose an improved calculation method by proposing a threshold formula with penalty Eq. 12.

The results of the threshold formula with penalty are more dependent on the prior classification confidence scores, and the results are more stable compared to the mean formula. Keeping the threshold higher allows the model to resist erroneous template updates when encountering long periods of target disappearance and occlusion.

$$\begin{aligned} p\_mean &= \sum_{m}^{n} \big(\sum_{i}^{m} s_i/m\big)/n \\ &= \big[\big(s_1 + \frac{s_1}{2} + \dots \frac{s_1}{n}\big) \\ &+ \big(\frac{s_2}{2} + \dots \frac{s_2}{n}\big) + \dots \big(\frac{s_n}{n}\big)\big]/n \end{aligned} \tag{12}$$

The results of the threshold formula with penalty are more dependent on the prior classification confidence scores, and the results are more stable compared to the mean formula. Keeping the threshold higher allows the model to resist erroneous template updates when encountering long periods of target disappearance and occlusion.

	TATrack-S	TATrack-B	TATrack-L
Target Image	112×112	112×112	192×192
Previous Image	224×224	224×224	384×384
Search Image	224×224	224×224	384×384
Backbone	$\begin{bmatrix} N_1 = 3 \\ N_2 = 2 \\ C = 96 \end{bmatrix}$	$\begin{bmatrix} N_1 = 9 \\ N_2 = 8 \\ C = 128 \end{bmatrix}$	$\begin{bmatrix} N_1 = 9 \\ N_2 = 8 \\ C = 128 \end{bmatrix}$
Neck	$N_3 = 4$	$N_3 = 8$	$N_3 = 8$
MACs	13.1 G	45.1 G	162.4 G
Param	24.5 M	112.8 M	112.8 M
Speed(V100)	29.6 FPS	14.1 FPS	6.6 FPS

Table 1: The network structure parameters of TATrack-S, TATrack-B, and TATrack-L. The number of N_1, N_2, N_3, C corresponds to Fig. 3

Experiments

Implementation Details

Our tracker was implemented on Python 3.9 and pytorch 1.11.0, trained on 2 Tesla A100 GPUs. The different sizes of TATrack are shown in Tab. 1, on PaE and SWA modules, TATrack-S, TATrack-B, and TATrack-L are loaded with pre-training weights of Swin-Tiny, Swin-Base, and Swin-Base384, respectively. We used TrackingNet, LaSOT, COCO and GOT-10k multiple training sets for joint training.

Comparison with the State-of-the-Art Trackers

GOT-10k. GOT10k (Huang, Zhao, and Huang 2019) is a large benchmark containing 560 classes of motion objects and 87 classes of motion patterns, and places more emphasis on the regression accuracy of the tracker on the target.GOT-10k officially requires that the tracker be trained based only on the training set of GOT-10k, and we followed this guidance.GOT-10k provides 180 test sequences with an average sequence length of 150, and officially does not disclose the true annotation of the test sequences, and we obtained the tracking metrics by submitting the raw tracking results for online evaluation Tab. 2.

TrackingNet. TrackingNet (Muller et al. 2018) contains 30312 video sequences, videos captured from real-life filmed YouTube content. trackingNet provides 511 test videos with an average sequence length of 441 frames, and no real annotation of the test sequences is publicly available. We submit the raw data to an official online evaluation service Tab. 2.

LaSOT. LaSOT (Fan et al. 2019) is a large benchmark for long sequences, it contains 280 test sequences averaging 2500 frames, and the challenge of LaSOT is the robustness

Method	Published	GOT-10k			TrackingNet			LaSOT		
		AO	SR_{50}	SR_{75}	AUC	P_{norm}	P	AUC	P_{norm}	P
TATrack-L	Ours	79.2	88.6	78.3	**85.0**	**89.3**	**84.5**	**71.1**	**79.1**	**76.1**
TATrack-B	Ours	77.3	87.8	74.1	83.5	88.3	81.8	69.4	78.2	74.1
TATrack-S	Ours	74.3	84.5	70.6	81.8	86.9	79.7	68.1	77.2	72.2
TATrack-B*	Ours	**73.0**	**83.3**	**68.5**	-	-	-	-	-	-
MixFormer(Cui et al. 2022)	CVPR22	71.2	79.9	65.8	82.6	87.7	81.2	67.9	77.3	73.9
ToMP(Mayer et al. 2022)	CVPR22	-	-	-	81.2	86.2	78.6	67.6	78.0	72.2
SBT-B(Xie et al. 2022)	CVPR22	69.9	80.4	63.6	-	-	-	65.9	-	70.0
SwinTrack-B(Lin et al. 2021)	arXiv21	69.4	78.0	64.3	82.5	87.0	80.4	69.6	78.6	blue74.1
KeepTrack(Mayer et al. 2021)	ICCV21	-	-	-	-	-	-	67.1	77.0	70.2
STARK(Yan et al. 2021)	ICCV21	68.8	78.1	64.1	82.0	86.9	-	67.1	77.0	-
TransT(Chen et al. 2021)	CVPR21	67.1	76.8	60.9	81.4	86.7	80.3	64.9	73.8	69.0
Ocean(Zhang et al. 2020)	ECCV20	61.1	72.1	47.3	-	-	-	56.0	65.1	56.6
SiamPRN++(Li et al. 2019)	CVPR19	51.7	61.6	32.5	73.3	80.0	69.4	49.6	56.9	49.1
SiamFC(Bertinetto et al. 2016)	ECCV16	34.8	35.3	9.8	57.1	66.3	53.3	33.6	42.0	33.9

Table 2: Comparison with the state of the art on the GOT-10k,TrackingNet and LaSOT. The underlined results in GOT-10k are not involved in the comparison because the models are trained based on multiple datasets. TATrack-B* is trained on the GOT-10k training set only.

Modification	GOT-10k	TrackingNet		LaSOT	
	AO	AUC	P	AUC	P
TATrack-S	**74.3**	81.8	**79.7**	**68.1**	**72.2**
Swin Bac.	72.9	81.5	79.3	67.1	71.4
No Pos.	71.3	81.1	78.8	66.7	70.5
No ltrb.	73.9	**82.0**	**79.7**	67.8	72.0
No gauss.	73.7	81.7	79.2	66.9	70.7

Table 3: Ablation studies on TATrack-S.

of long-term tracking. We applied the p-mean algorithm on LaSOT to calculate the threshold values and achieved state-of-the-art performance Tab. 2.

Ablation Study and Analysis

We did ablation experiments on the components of TATrack and we analyzed the contribution of each separable component in the model. We designed different combinations of templates to verify that templates with background are necessary for the tracker, and we demonstrated the effectiveness of the dynamic thresholding algorithm by comparing multiple update methods. Both the ablation experiments and the comparison experiments were performed under the TATrack-S model.

LCA Ablation. Swin backbone means we remove the LCA module in backbone and use the first three stages of swin transformer to extract features, so that the image loses target perception during the feature extraction. We can see that Tab. 3 all the metrics in TrackingNet, LaSOT, and GOT-10k are degraded, and we only need to add two layers of LCA in the TATrack-S backbone to get a direct performance improvement. For the third line, we remove all the location codes extended in the LCA module in backbone and neck. The metrics of the three datasets show significant degradation, demonstrating that the absence of location encoding divided independently by image relationships poses significant difficulties for the model to construct associations between multiple images.

Box-Embedding. We examined the impact of the box embedding in the previous template Tab. 3. When we removed ltrb, there was a small decrease in the tracking metrics of GOT-10k and LaSOT and a small improvement in the AUC of TrackingNet. The ltrb embedding has a more limited improvement on the model, probably due to the fact that LCA already has a sufficiently accurate regression on the target. And there is an error in the prediction of the box during inference, and ltrb may introduce the error of the previous template into the prediction of the current frame. When gauss with localization function is removed, the performance degradation is more significant than removing ltrb. It indicates that the importance of localization information is higher in templates that contain background.

With or without Background. In Tab. 4 we compare the two experimental setups using TATrack. the Only target scheme removes the background from the previous template and does not use box-embedding, which results in a significant drop in performance metrics for all datasets, but has a faster speedup. Because the background is removed and box-embedding information cannot be introduced, this scenario is not a fair comparison. In the scenario where both templates contain background, we add background to the target and the target template does not use box-embedding information. We can see that the template with the background is higher in all performance metrics than the template

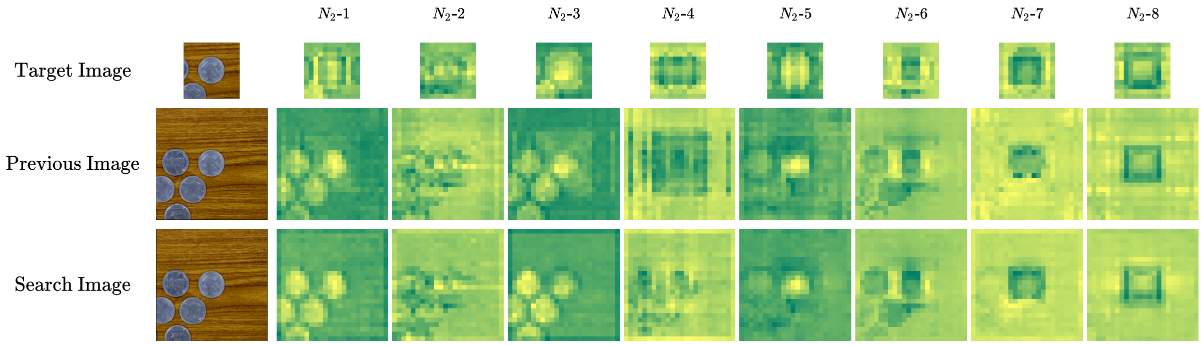

Figure 4: Visualization of the feature maps output from each layer of LCA in the Target-aware backbone network.

Method	GOT-10k	TrackingNet		LaSOT	
	AO	AUC	P	AUC	P
TATrack-S	74.3	81.8	79.7	68.1	72.2
Only target.	72.1	80.8	78.0	65.7	68.2
Both background.	**74.7**	**82.2**	**80.1**	**68.5**	**72.5**

Table 4: Comparison with the dual-template scheme with only the target and the dual-template scheme with both including the background.

Method	GOT-10k			LaSOT		
	AO	SR_{50}	SR_{75}	AUC	P_{norm}	P
No Update	71.4	81.3	66.3	66.7	75.8	70.8
Update Last	68.3	77.7	63.8	61.0	68.5	64.2
Mean.	**74.3**	**84.5**	**70.6**	66.1	75.0	69.8
P-mean.	74.0	84.3	70.0	**68.1**	**77.2**	**72.2**

Table 5: Effects of different update strategies on long and short series datasets.

without the background, even for the target template that is not updated. Considering the performance flatness, we finally chose the compromise between the target-only template and the previous template with the background, which still shows the huge potential of introducing the background in the template for performance improvement.

Update Experiment. We use a typical short sequence dataset GOT-10k and a long sequence dataset LaSOT to validate our experiments. As shown in Tab. 5, (1) no update strategy is adopted and the initial frame is used as the previous template, and this scheme achieves an ordinary performance. (2) is to take a fixed update of the previous frame as the previous template, the performance shows a significant drop, indicating that a low-quality template will be disastrous to the tracker. (3) We adopt the historical confidence mean as the dynamic threshold method to update the template, and we can see that better performance can be obtained in GOT-10k which requires higher regression accuracy. This is because the mean value method as a threshold can achieve a good balance of appropriately skipping low quality templates and keeping the previous templates close to the current frame. However, the threshold of the mean value is too low to resist the long-time target disappearance for long sequence datasets LaSOT. (4) Using the historical mean with penalty will appropriately raise the criteria to resist prolonged low-quality templates and the results are more stable. P-mean shows good performance on LaSOT and has no significant negative impact on short series datasets.

Visualization of LCA. To explore how LCA plays a target-aware role in the backbone network, we visualize the output feature maps of the LCA module in the TATrack-L backbone network. We calculate the average of the target features, previous features, and search features of the LCA output on the C channel and adjust them into a response map of $\mathbb{R}^{H\times W}$. Through the visualization in Fig. 4, we can conclude that (1) LCA can determine the target location in the previous template and search image layer by layer. (2) LCA can exclude interfering objects layer by layer.

Limitations

The use of multiple templates, especially the previous templates containing background, makes our model run low. TATrack-Small becomes less tiny after using Swin-Tiny pretraining weights. we can see from the experiments that templates containing background have a greater potential to improve the tracker performance. Therefore, in future work we will consider using a transformer module that optimizes the computational effort to better mine templates with background to achieve more powerful performance. Also, we will continue to explore low-cost in new update methods.

Conclusion

We propose the Long-Term Contextual Attention module, a fusion module for fusing target and background information over multiple time frames. Taking advantage of LCA's ability to simultaneously extract features and compute correlations across images, we propose TATrack with target-awareness. to allow simple and efficient updating of templates, we propose an online update algorithm for dynamic thresholding.

Acknowledgements

This work is supported by National Natural Science Foundation of China (Nos. 62266009, 61866004, 62276073, 61966004, 61962007), Guangxi Natural Science Foundation (Nos. 2018GXNSFDA281009, 2019GXNSFDA245018, 2018GXNSFDA294001), Guangxi Collaborative Innovation Center of Multi-source Information Integration and Intelligent Processing, and Guangxi "Bagui Scholar" Teams for Innovation and Research Project.

References

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

Chen, X.; Yan, B.; Zhu, J.; Wang, D.; Yang, X.; and Lu, H. 2021. Transformer tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8126–8135.

Cui, Y.; Jiang, C.; Wang, L.; and Wu, G. 2022. MixFormer: End-to-End Tracking with Iterative Mixed Attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 13608–13618.

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

Fan, H.; Lin, L.; Yang, F.; Chu, P.; Deng, G.; Yu, S.; Bai, H.; Xu, Y.; Liao, C.; and Ling, H. 2019. Lasot: A high-quality benchmark for large-scale single object tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 5374–5383.

Fu, Z.; Liu, Q.; Fu, Z.; and Wang, Y. 2021. Stmtrack: Template-free visual tracking with space-time memory networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 13774–13783.

Huang, L.; Zhao, X.; and Huang, K. 2019. Got-10k: A large high-diversity benchmark for generic object tracking in the wild. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(5): 1562–1577.

Ke, G.; He, D.; and Liu, T.-Y. 2020. Rethinking positional encoding in language pre-training. *arXiv preprint arXiv:2006.15595*.

Li, B.; Wu, W.; Wang, Q.; Zhang, F.; Xing, J.; and Yan, J. S. 2019. Evolution of siamese visual tracking with very deep networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA*, 16–20.

Li, B.; Yan, J.; Wu, W.; Zhu, Z.; and Hu, X. 2018. High performance visual tracking with siamese region proposal network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 8971–8980.

Lin, L.; Fan, H.; Xu, Y.; and Ling, H. 2021. Swintrack: A simple and strong baseline for transformer tracking. *arXiv preprint arXiv:2112.00995*.

Liu, Z.; Lin, Y.; Cao, Y.; Hu, H.; Wei, Y.; Zhang, Z.; Lin, S.; and Guo, B. 2021. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 10012–10022.

Mayer, C.; Danelljan, M.; Bhat, G.; Paul, M.; Paudel, D. P.; Yu, F.; and Van Gool, L. 2022. Transforming model prediction for tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8731–8740.

Mayer, C.; Danelljan, M.; Paudel, D. P.; and Van Gool, L. 2021. Learning target candidate association to keep track of what not to track. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 13444–13454.

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

Rezatofighi, H.; Tsoi, N.; Gwak, J.; Sadeghian, A.; Reid, I.; and Savarese, S. 2019. Generalized intersection over union: A metric and a loss for bounding box regression. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 658–666.

Xie, F.; Wang, C.; Wang, G.; Cao, Y.; Yang, W.; and Zeng, W. 2022. Correlation-aware deep tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8751–8760.

Yan, B.; Peng, H.; Fu, J.; Wang, D.; and Lu, H. 2021. Learning spatio-temporal transformer for visual tracking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 10448–10457.

Zhang, C.; Li, Z.; and Wang, Z. 2018. Joint compressive representation for multi-feature tracking. *Neurocomputing*, 299: 32–41.

Zhang, H.; Wang, Y.; Dayoub, F.; and Sunderhauf, N. 2021. Varifocalnet: An iou-aware dense object detector. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8514–8523.

Zhang, L.; Gonzalez-Garcia, A.; Weijer, J. v. d.; Danelljan, M.; and Khan, F. S. 2019. Learning the model update for siamese trackers. In *Proceedings of the IEEE/CVF international conference on computer vision*, 4010–4019.

Zhang, Z.; Peng, H.; Fu, J.; Li, B.; and Hu, W. 2020. Ocean: Object-aware anchor-free tracking. In *European Conference on Computer Vision*, 771–787. Springer.