Model Uncertainty Guides Visual Object Tracking[*]

Lijun Zhou[1,2,3,4], Antoine Ledent[4], Qintao Hu[2,3], Ting Liu[1†], Jianlin Zhang[2†], Marius Kloft[4]

[1]Alibaba Group, Hangzhou, China
[2] Institute of Optics and Electronics, Chinese Academy of Sciences, Chengdu, China
[3]University of Chinese Academy of Sciences, Beijing, China
[4]Department of Computer Science, TU Kaiserslautern, Kaiserslautern, Germany
{zhoulijun16,huqintao16}@mails.ucas.edu.cn, {ledent,kloft}@cs.uni-kl.de, brooks.lt@alibaba-inc.com, jlin@ioe.ac.cn

Abstract

Model object trackers largely rely on the online learning of a discriminative classifier from potentially diverse sample frames. However, noisy or insufficient amounts of samples can deteriorate the classifiers' performance and cause tracking drift. Furthermore, alterations such as occlusion and blurring can cause the target to be lost. In this paper, we make several improvements aimed at tackling uncertainty and improving robustness in object tracking. Our first and most important contribution is to propose a sampling method for the online learning of object trackers based on uncertainty adjustment: our method effectively selects representative sample frames to feed the discriminative branch of the tracker, while filtering out noise samples. Furthermore, to improve the robustness of the tracker to various challenging scenarios, we propose a novel data augmentation procedure, together with a specific improved backbone architecture. All our improvements fit together in one model, which we refer to as the Uncertainty Adjusted Tracker (UATracker), and can be trained in a joint and end-to-end fashion. Experiments on the LaSOT, UAV123, OTB100 and VOT2018 benchmarks demonstrate that our UATracker outperforms state-of-the-art real-time trackers by significant margins.[1]

Introduction

Visual tracking aims to estimate the trajectory of a target in a video sequence. It has wide applications ranging from human motion analysis, human-computer interaction, to autonomous driving. Modern CNN-based object trackers typically aim to learn a classifier that can quickly adapt to object and background variations. In this so-called online learning framework, earlier image frames together with the target location are fed to an online learning classifier branch of the tracker architecture. A serious issue is that due to limitations in computing and memory resources, a tracker can only include a small number of frames for learning the classifier. Thus, the success of a tracker heavily depends on the design of a sensible strategy to select the most relevant frames.

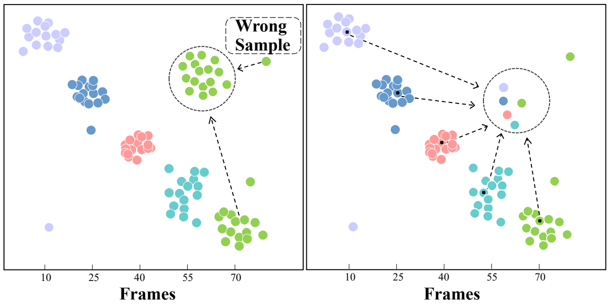

Figure 1: Illustrative comparison of the sample selection strategies by the state-of-the-art Dimp50 (left) and our proposed UATracker (right). The y axis is the feature dimension, which represents (a feature representation of) the image frames. The samples' colours represent the time intervals they belong to. The dashed line circles represent which frames are chosen by the tracker to be fed into the online learner. On the right, samples marked with black are selected.

Many classical trackers are based on correlation filters (Danelljan et al. 2015; Henriques et al. 2015; Danelljan et al. 2016, 2017; Liu, Wang, and Yang 2015) and update the learning model based on the previous frame. On the other hand, the more recent Siamese trackers (Bertinetto et al. 2016b; Guo et al. 2017; Tao, Gavves, and Smeulders 2016; Wang et al. 2018; Zhu et al. 2018; Li et al. 2018, 2019) typically use the first frame to provide a reasonable initialization to the model. Meanwhile, CFNet (Valmadre et al. 2017) uses a running average with a constant learning rate. This means the influence of past frames to the model is decreasing exponentially fast. Recently, trackers such as ATOM (Danelljan et al. 2019) and Dimp (Bhat et al. 2019) were able to combine features from earlier frames via an optimisation process to predict the location of the target. However, the sample selection is ad hoc and rudimentary: they simply include samples from the first frame and the last few frames.

We argue that these existing sampling strategies may not select compact and representative samples. For one thing, they tend to select clumps of samples from the same periods.

[*] This work was supported by the funding of "Leading Innovation Team of the Zhejiang Province"(2018R01017). AL and MK acknowledge support by the German Research Foundation (DFG) award KL 2698/2-1 and by the Federal Ministry of Science and Education (BMBF) awards 01IS18051A and 031B0770E.

[†] Corresponding author.

[1] The code is available at. github.com/TrackerLB/UATracker

But frames that are close to each other in time are likely to yield similar feature representations, which increases the redundancy of naively chosen samples.

To address these issues, we propose a sampling approach based on an uncertainty adjustment for online learning. Firstly, the video frames are partitioned into clusters depending on time stamp information [2]. Taking inspiration from work on linear regression for heteroskedastic data (Kendall and Gal 2017), we *jointly incorporate uncertainty estimation into our model*'s target location procedure in an online manner and retain this information for all historic frames. Using it, an uncertainty-adjusted center is calculated for each cluster, and the frame closest to the center is selected as the representative frame, as long as its response score is higher than a minimum threshold. Otherwise, the frame is considered not to be representative, and the next closest frame to the cluster centre is chosen instead. We refer to this technique by Significant Sample Selection (S3). The method is schematically illustrated in Fig. 1, which compares the sample selection strategies of Dimp (left) with that of our tracker UATracker (right). It is obvious that the sample selection strategy of the UATracker has covered much more representative samples and avoids anomalous samples, $i.e.$, frames corresponding to outliers in feature space, where the target is most likely blurred or occluded.

Our tracker is built upon the classic CNN framework with a target classification branch and a target localisation branch atop the convolutional layers. The classification network, which is an online learning branch equipped with the S3 modules, identifies the coarse locations (bounding boxes). These coarse locations are further fed into the target localisation branch to estimate the precise target location.

Furthermore, we also propose two strategies to enhance the tracker and improve its robustness. First of all, whilst existing data augmentation methods such as translation and flipping are widely used in in current trackers, problems such as occlusion, blurring and overlapping have not been properly addressed in existing research. We propose a ”mixed-features” data augmentation method which creates new samples by simulating occlusions and blurring. Whilst previous work such as Mixup (Zhang et al. 2017) fused samples directly at the image level, we choose to fuse at the feature level, which we experimentally demonstrate to be more effective. The proposal is aligned when added, which can better simulate occlusion and blur. Finally, we introduce deformable convolutions into the backbone network, which are trained in an end-to-end fashion together with our tracker.

Our main contributions can be summarized as follows:

- First, we improve the robustness by finding out the most beneficial input training samples for the tracker, and removing noisy samples as much as possible. Specifically, a simple-yet-efficient Significant Sample Selection (S3) strategy is proposed. The method relies on (aleatoric) uncertainty adjustment, with the uncertainty estimation embedded into the regression branch of the model through a carefully designed loss function.
- To further improve the robustness of the tracker, a ”mixed-feature” data augmentation method is proposed and applied to the classifier. The method consists in adding samples perturbed through simulated occlusions and blurring to the training data set, and achieves significantly increased performance in the event of *actual* target occlusion or blur. Furthermore, we apply deformable convolution to the backbone of the network and perform an end-to-end training.
- Our tracker achieves top performance on four benchmarks: VOT2018, OTB100, LaSOT and UAV123. In particular, we improve the state-of-the-arts on the LaSOT and UAV123 benchmarks by significant margins.

Related Work

In recent years, benefiting from the rapid development of CNN and object detectors (Zhang et al. 2019b), visual object tracking has achieved unprecedented progress. Existing visual object trackers can be roughly categorised into two classes: detection-based tracking and template-matching methods. In detection-based tracking, we treat tracking as an online classification problem, classifying the target and the background to locate the target, and capturing the change in the scale of the target by searching on multiple scales. Template-matching methods, in particular those based on Siamese networks, have attracted more and more researchers' attention due to the end-to-end training ability and high efficiency. The main component is a simple end-to-end symmetrical network, which learns the similarity measure between the template and the search area through offline fine-tuning. In the following, we introduce these two methodologies in details.

Siamese Tracker. The Siamese tracker is based on the Siamese network, which has the ability to perform offline pre-training and high efficiency tracking under a single simple framework. The network structure achieves high-speed running and has attracted much attention. It uses a symmetric Siamese network to learn the similarity measure between the object template and the search area. SiamFC (Bertinetto et al. 2016b) performs similarity prediction using a fully convolutional structure, obtaining a super high-speed tracker. It treats the deep convolutional network as a more general similarity learning problem in the initial offline phase, and then performs a simple online estimation of this problem during tracking. SiamRPN (Li et al. 2018) combines the Siamese network with a regional proposal network (RPN), which uses an end-to-end method for offline training on large-scale image pairs. Unlike standard RPN, SiamRPN uses the feature map of two branches (a template branch and a search area branch) to extract proposal regions. SiamRPN++ (Li et al. 2019) introduces a deeper feature network into the SiamRPN (Li et al. 2018), which successfully enables the Siamese network to perform end-to-end offline pre-training on ResNet (He et al. 2016).

The main weakness of the Siamese trackers is that they cannot integrate tracking samples of present frames into

[2] This choice of clustering is a quick and efficient way to use time stamp information as a similarity measure without the computational hassle of using a more complicated kernel.

templates: similarity measures trained purely offline cannot adapt to complex evolving tracking scenarios.

Detection-based Tracking. This line of tracking methods (Ma et al. 2015; Danelljan et al. 2016, 2017; Hong et al. 2015; Li, Li, and Porikli 2015) converts object tracking to classification problems by discriminating between targets and backgrounds online. For example, the trackers (Nam and Han 2015; Nam et al. 2016; Han, Sim, and Adam 2017) based on discriminant correlation filters use the target information to continuously update the tracking template.

In recent years, feature representations have mainly been extracted from pre-trained deep networks for image classification combined with online learning of correlation filters. CCOT (Danelljan et al. 2016) and ECO (Danelljan et al. 2017) propose implicit interpolation models to formulate learning problems in the continuous space domain and achieve effective integration of multi-resolution deep features. ATOM (Danelljan et al. 2019) attempts to bridge object classification and location estimation in target tracking by building a multi-task tracking model which consists of two parts: object classification and position estimation, with the latter typically achieved through end-to-end offline pre-training. Dimp (Bhat et al. 2019) designs a loss with discriminating ability and learns the key parameters of loss focus through end-to-end training. This combined weight prediction module can initialize the network well.

Although acceptable performance has been achieved by detection-based trackers, their template updating strategies are not constructed carefully. Early correlation filtering methods simply use samples from the previous frame of the tracking frame as training samples, which can be affected by inaccurate target prediction. The recent Dimp (Bhat et al. 2019) and ATOM (Danelljan et al. 2019) trackers use the first few frames of the tracking frame to construct a training set. Considering the large variations of appearance targets can exhibit in a long-term tracking procedure, a more sensible online sample selection strategy is needed.

Uncertainty Estimation for Computer Vision. In complex problems involving a large amount of data and variables such as computer vision, errors can come from a wide variety of potential sources, which means the need to quantify this uncertainty and weigh intermediate predictions accordingly is particularly marked. Accordingly, uncertainty estimation has a long history in the computer vision literature. In (Kendall and Gal 2017), the authors discuss different types of uncertainty and propose a Bayesian deep learning framework that models both the aleatoric and epistemic uncertainties. MonoPair (Chen et al. 2020) proposes an uncertainty perception prediction module in the context of 3D target detection. Gaussian-YOLO (Choi et al. 2019) learns the uncertainty of bounding box (bbox) prediction values through Gaussian modeling and loss function reconstruction. Monoloco (Feng, Rosenbaum, and Dietmayer 2018) evaluates and visualizes the uncertainty of azimuth prediction in pedestrian positioning.

We argue that it is also possible to use uncertainty to guide the components of the network. Our tracker models the uncertainty present in frame samples to be fed to the classification branch of the model in order to better screen out noisy samples and improve the robustness of the branch. This is to the best of our knowledge the first time that uncertainty estimation was embedded directly in the selection of intermediary samples, and trained in an end-to-end fashion.

Methodology

Our model is built upon the Dimp tracker (Bhat et al. 2019), which involves a target classification branch and a target localization branch. The classification network is an online learning branch and identifies the coarse locations. These coarse locations are further fed to the target localization branch to estimate the precise target location, as shown in Fig. 2. The target localization branch is not shown here, please refer to ATOM (Danelljan et al. 2019) for details. As mentioned above, we propose several improvements aimed at better keeping track of uncertainty in the model and improving the robustness of the model. Below, we explain in more details the architecture and training procedure for our full model, including all of our improvements.

General Architecture and Training Procedure

Architecture. The main architecture of our tracker is composed of two branches, as seen in Fig. 2. The middle branch takes as input a reference frame together with bounding boxes indicating the target position, and outputs a feature representation of the target appearance. The bottom branch takes as input a single test frame. The final feature representation of the test frame obtained by the bottom branch of the model is convolved with a filter ω to reach a score map (heat map) representing the probability of the target being in a certain region of the test frame. The filter ω is trained using the labels of the training frames via online learning.

Online Learning Loss. The online-learning tracker trains the classification branch by using the previous sampled frames as training samples, and minimizing the discrepancy between the tracking response and a Gaussian prior c centered at the centre of the bounding box label. Denote the training samples as x_m. Whenever a new sample frame is added, the parameters of the classification branch are updated by minimizing the following objective function:

$$L(\omega) = \frac{1}{M} \sum_{m=1}^{M} \left\| r(x_m * \omega, c_m) \right\|^2 + \lambda \left\| \omega \right\|^2 , \qquad (1)$$

where the samples $\{1, 2, \dots, M\}$ are chosen as described below, c_m is set to a Gaussian prior at the target location, $*$ denotes a multi-channel convolution. λ is a hyper-parameter, $x_m * \omega$ is the tracking response obtained by convolution, and r is a point-wise loss function.

By optimizing Eq. (1) with a conjugate gradient descent method, the model predicts the target response map to estimate the coarse locations of a tracked object, which is shown in Fig. 2. The Dimp (Bhat et al. 2019) tracker will feed some features of earlier frames in the model predictor to obtain a more robust model to estimate the score map of the test frame. As explained above, performance of this online learning branch is heavily dependent on the quality of the samples it is based on, which can be subject to errors since they rely

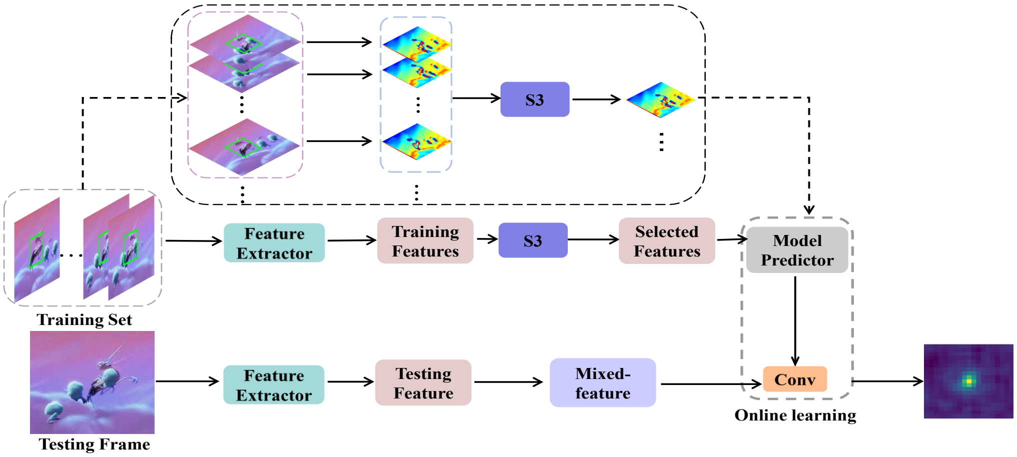

Figure 2: Architecture for the UATracker, which converts the feature map into a response map and provides the coarse locations of the target by online learning. Specifically, selected features are chosen by our S3 strategy to replace original training samples for online learning, and the classification branch finally obtains the response score map of the test frame. Note that the mixed-feature is used in the training of the classification branch. The top branch is a Visualization of the middle branch.

on previous iterations of the model. We propose to embed uncertainty estimation directly into the loss of the regression branch of the model, and further use it as a weight to average the samples in each interval and find the most reliable sample. The specific method is described below.

Uncertainty-sensitive Online Learning

As mentioned in the introduction, on top of the main online learning classification branch, our tracker involves a separate regression branch which estimates the shape of the target and is trained to compute the Intersection Over Union (IOU) of the target with any bbox proposal. Here, we augment this branch with joint uncertainty prediction, and further use this measure of uncertainty to decide which frames are fed to the online learning branch at test time.

In (Kendall and Gal 2017), an ingenious technique is developed to perform linear regression on heteroscedastic data by predicting both the output and the uncertainty jointly. Taking inspiration from this, we develop a technique to incorporate such methods in our deep learning context. Specifically, we enhance the regression branch of our model with joint uncertainty prediction. Our regression branch takes as input a proposal bounding box, and outputs a real number estimating the IOU between the target and the bbox proposal. We further augment the network so that it includes a measure of uncertainty σ: let us write IOU_θ for the regression branch of our network (where θ represents the weights of the neural network). Our output is then two dimensional:

$$(y, \sigma) = \text{IOU}_\theta(B). \tag{2}$$

Here, σ will be an estimate of the (aleatoric) uncertainty associated with the IOU prediction y returned by the network IOU. To train the network based on the ground truth IOU calculated with the labels, we use the following loss:

$$L(\theta) = \sum_i \left[\frac{1}{2} \exp(-\sigma_i) \|y_i - \hat{y}_i\|^2 + \frac{1}{2}\sigma_i \right]. \tag{3}$$

Note that this loss function does not require explicit 'labels' for the uncertainty σ: the loss function simply directly encourages σ to take a high value whenever the error $\|y_i - \hat{y}_i\|^2$ is large. Thus, trained jointly with the predictions y_i and our special loss function, σ can *estimate the noise of the input data*, which will help us filter out unreliable samples from the training set we feed to the online learning branch.

We propose the Significant Sample Selection (S3) strategy to obtain high-quality representative samples.

We propose the Significant Sample Selection (S3) strategy to obtain high-quality representative samples. We divide the set of previous frames into intervals, and select a representative frame from each interval taking into account to the uncertainty of the output of each sample. For each interval J, we select a representative sample by assigning a weight to each sample based on the uncertainty computed by (2) and then calculating the average value:

$$\tilde{x} = \frac{1}{\sum_{j \in J} \exp(-\sigma_j)} \sum_{j \in J} \exp(-\sigma_j) x_j, \tag{4}$$

where x_j is the feature representations for $j \in J$. We select the $j \in J$ which minimises the distance $|x_j - \tilde{x}|$ as a preliminary choice. If the score returned by x_j is above a predetermined threshold, the sample is retained as the representative sample of the interval. Otherwise, the next nearest sample is chosen. This reduces the impact of poorly-tracked samples, (which may involve occlusions etc.): only samples which are known to contain high confidence information about the target can be part of the template training set. This ensures that the template is not contaminated by poor-quality frames.

Two Effective Ways to Improve the Model

If the training data lacks diversity, a well-trained model can generalize poorly. To further improve the generalization performance of the tracker, we propose two methods:(1) aug-

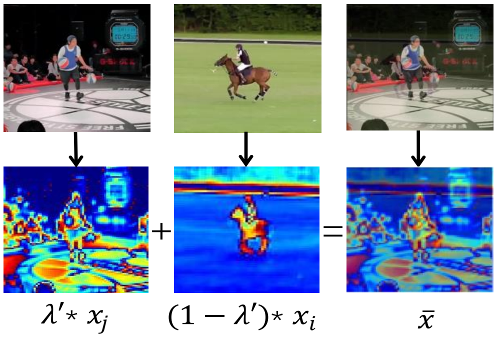

Figure 3: Visualization of the mixed-feature method.

menting the data with more diverse artificial samples,and (2) improving the robustness of the model architecture.

Mixed-feature Method for the Training. When training the classification branch we employ a data augmentation strategy which we refer to as the ”mixed-feature” method. We perturb each sample frame x_i by a small multiple of another sample x_j in feature space, and retain the label of the first sample. This induces smoothness of the branch, and improves robustness w.r.t. alterations of the feature maps. Since CNN feature maps retain image-like appearance, one can think of such perturbations as slightly more abstract analogues of occlusions or blur (as Figure 3 illustrates).

More precisely, our method is as follows. At each iteration of the SGD procedure, we randomly select two sample frames (x_i, y_i) and (x_j, y_j) in feature space. Then, we randomly select a parameter λ from a Beta distribution $\beta(0.1, 0.1)$ (this encourages choices of λ which are close to 0 and 1). We then form a virtual sample $(\bar{x}, \bar{y})$ as follows:

$$\bar{x} = \lambda x_j + (1-\lambda)x_i, \tag{5a}$$

$$\bar{y} = y_i 1_{\lambda<0.5} + y_j 1_{\lambda\geq 0.5}, \tag{5b}$$

where 1 stands for the indicator function. Thus, the label is simply set to that of y_i if $\lambda < 0.5$ (i.e. $\bar{x}$ is closer to x_i than x_j), and to that of y_j otherwise. Recall that the labels c_j are Gaussians centered around the presumed centre y_j of the target, so equivalently, the label centres satisfy $\bar{c} = c_i 1_{\lambda<0.5} + c_j 1_{\lambda\geq 0.5}$. We then feed the pair $(\bar{x}, \bar{y})$ (or equivalently $(\bar{x}, \bar{c})$) to the offline target classification branch.

Deformable Convolutions for the Backbone. Traditional CNNs’ fixed convolution kernel size limits their ability to model effects such as geometric deformation (Dai et al. 2017). To improve robustness to geometric deformations, DCN (Dai et al. 2017) introduced ”deformable convolutions”, which dynamically adjust the receptive field. Similarly, inspired by DCN-V2 (Zhu et al. 2019), we replace all $3 * 3$ conv layers in the layer2-layer4 stage of our backbone (ResNet50) by deformable convolutions. We introduce this modification into our backbone and carry out the training with the other branches in an end-to-end fashion. Further training details are described in the experiments Section.

S3	Mixed-feature	DCN	AUC	Precision	FPS
			0.654	0.827	45
✓			0.671	0.872	45
	✓		0.663	0.861	45
		✓	0.664	0.862	45
✓	✓		0.673	0.875	45
✓		✓	0.671	0.873	45
	✓	✓	0.669	0.870	45
✓	✓	✓	**0.676**	**0.879**	45

Table 1: Ablation study of our UATracker on the UAV123 benchmark. S3 denotes our significant sample selection. ”DCN” and ”Mixed-feature” refer to the use of deformable convolutions in the backbone and our mixed-feature data augmentation technique respectively. The baseline performance is reported by the state-of-the-art Dimp50 tracker.

Experiments

Implementation Details

All the experiments were carried out with Pytorch on an Intel i5-8600k 3.4GHz CPU and a single Nvidia GTX 1080ti GPU with 24GB memory. The UATracker was implemented based on the Dimp architecture (Bhat et al. 2019), by using $ResNet50 + DCN_V2$ (He et al. 2016; Zhu et al. 2019) as backbone. We choose 10 as the size of the time intervals. All experiments reported are the average of multiple runs: VOT is the average of 15 runs, whilst OTB, UAV123 and LaSoT are the average of 5 runs. When the maximum score in the response score map of the current target is less than 0.2 times the maximum score in the first frame, we conclude that the target may have been lost and accordingly expand the search area to retrieve it.

Ablation Study

We verify the effectiveness of our methods and compare the effects of all three modules proposed in this paper. Ablation studies were performed on the UAV123 (Mueller, Smith, and Ghanem 2016) and LaSoT (Fan et al. 2019) datasets.

Significant Sample Selection. We compare our sample selection method with the following situations: 1. Randomly selecting one of several samples as the updated sample. 2. The GMM (Gaussian Mixture Model) from ECO applied to the baseline. 3. Using the standard mean when selecting the memory sample. 4. Directly using IoU prediction as the sample selection method. The results are shown in Table 2. Our method clearly outperforms the other four selection methods above. Compared with the baseline, it increased the AUC by 1.7% (from 0.654 to 0.671). As shown in Table 1, when combined with mixed-feature and DCN in backbone, our method performs best by a significant margin: it increased the AUC by 2.2% as compared with baseline. Note also that each of the three techniques (S3, mixed-feature and DCN) provides noticeable improvements in all combinations, with our S3 sampling strategy providing the greatest benefits.

Mixed-feature in the Training. From the results in Table 1, we see that the mixed-feature data augmentation in the training has a positive effect on the results. The AUC

	RS	GMM	SM	IOU-pre	Ours
AUC	0.662	0.656	0.659	0.655	**0.671**
Precision	0.859	0.852	0.854	0.849	**0.872**

Table 2: Comparison of different sample selection methods on the UAV123 dataset. 'RS' refers to random selection, whilst 'GMM' refers to the Gaussian Mixture Model from ECO. 'SM' is the standard (non-weighted) mean strategy and IOU-pre means directly using IoU prediction as the sample selection method.

Tracker	EAO	Accuracy	Robustness
Ours	0.458	0.614	0.159
Dimp50	0.440	0.597	0.153
SiamRPN++	0.414	0.600	0.234
ATOM	0.401	0.590	0.204
SiamMask	0.380	0.609	0.276
LADCF	0.389	0.503	0.159
MFT	0.385	0.505	0.140
DaSiamRPN	0.383	0.544	0.276
UPDT	0.378	0.536	0.184
RCO	0.376	0.507	0.155

Table 3: Performance comparison on VOT-2018.

score when it works alone is from 0.654 to 0.663. The best result is when it works in combination with S3 and DCN.

DCN in the Backbone. From Table 1, we see that adding DCNs into the backbone has a positive effect. This alone bumps up the AUC from 0.654 to 0.664, but works best in combination with the S3 and mixed-feature strategies.

Analysis of Occlusion and Blurring. We choose videos with obvious occlusion and blur properties from the OTB dataset (Wu, Lim, and Yang 2015), and calculate the mean of the average overlap rate obtained in each video. As shown in Fig. 4, our tracker's performance on these videos is significantly better than the baseline(Dimp50), which further proves the effectiveness of our method at managing situations involving occlusion or blur. In particular, for some videos, the low overlap score exhibited by the baseline means that the target may be lost, whilst our tracker still performs well. This implies that our tracker is demonstrably less likely to lose the target due to temporary occlusions or blur.

Effectiveness of S3 in Different Trackers. We further verified the role of the S3 modules on different trackers including ECO (Danelljan et al. 2017), ATOM (Danelljan et al. 2019), Dimp50 (Bhat et al. 2019) and UpdateNet (Zhang et al. 2019a). We applied the S3 module on the basis of GMM. S3 is directly applied to ECO instead of replacing the original sample selection strategy. Our conclusion is that the introduction of this sample update strategy is clearly effective, not just on the Dimp50 tracker but on all other trackers we tried. The experiments were conducted on the LaSOT testing dataset. Table 4 shows the statistical results of different trackers with and without their S3 update module. Especially, when only the S3 module is used, the OTB performance is 0.701 (Dimp: 0.684).

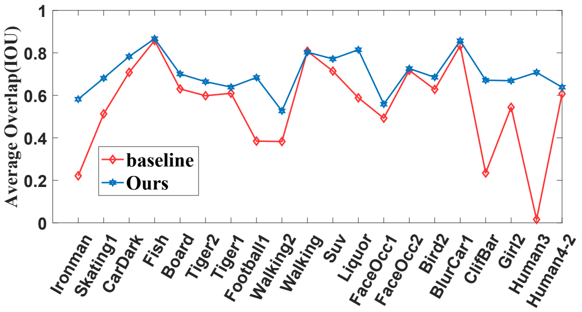

Figure 4: Comparison of the average overlap rate of our method and the baseline(Dimp50) in videos featuring occlusion or blur in the OTB dataset.

Tracker	Success Rate	Normalized Precision
ECO	0.324	0.338
ECO+S3	0.391	0.493
UpdateNet	0.475	0.560
UpdateNet+S3	0.490	0.583
ATOM	0.515	0.576
ATOM+S3	0.532	0.612
Dimp50	0.569	0.643
Dimp50+S3	0.584	0.667

Table 4: Effectiveness of our S3 for different trackers.

State-of-the-art Comparison

OTB Dataset. The object tracking benchmark (OTB100) (Wu, Lim, and Yang 2015) consist of 100 fully annotated videos. All OTB sequences are manually tagged with one or more of 11 typical tracking interference properties making tracking more challenging. Two evaluation metrics of success rate and precision are included. Trackers were ranked using the area under the curve (AUC) for each success plot. We use the success rate and precision plot in the one-pass evaluation (OPE) as the evaluation metrics for the results reported in the paper.

We compared the UATracker on OTB100 and the subset of this dataset consisting of videos tagged as featuring occlusions with state-of-the-art trackers including Dimp50 (Bhat et al. 2019), ATOM (Danelljan et al. 2019) and DaSiamRPN (Zhu et al. 2018), ECO-HC (Danelljan et al. 2017), SiamRPN (Li et al. 2018), SRDCF (Danelljan et al. 2015), Staple (Bertinetto et al. 2016a), CF2 (Valmadre et al. 2017) and CNN-SVM (Hong et al. 2015). Fig. 5 shows that our tracker achieves the best performance as measured by the AUC score. Our UATracker further improves the results with an AUC score of 70.9%. Specifically, compared with Dimp50, UATracker improved the AUC score by 2.5%, and increased the AUC score by 5.2% in the occlusion attribute.

LaSOT Testing Set. LaSOT (Fan et al. 2019) is a long-term tracking dataset, composed of 1400 video sequences, each with an average of 2512 frames. The shortest and

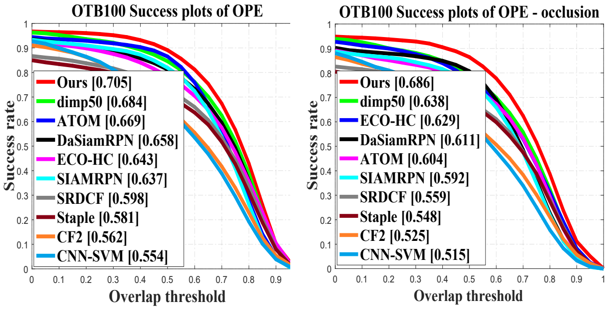

Figure 5: Success plot on the OTB100 dataset benchmark.

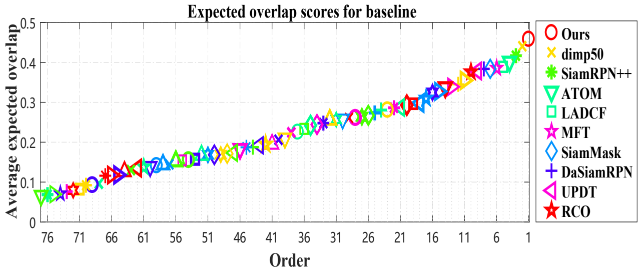

Figure 6: EAO ranking of the tested trackers on VOT2018.

longest videos have 1000 and 11397 frames respectively. They are divided into 70 categories, and each category contains twenty video sequences. Each video sequence presents different challenges. As shown in Fig. 7, we compare the UATracker with state-of-the-art trackers including Dimp50, Dimp18, ATOM, SiamRPN++ (Li et al. 2019), VITA (Song et al. 2018), SiamFC, ECO, ECO-HC and CFNet. Our tracker obtains the best AUC score: 0.590. Compared to Dimp50, the UATracker improves the normalized precision and AUC scores by 2.5% and 3.7%, respectively.

UAV123 Dataset. The UAV123 dataset (Mueller, Smith, and Ghanem 2016) contains a total of 123 video sequences and more than 110K frames. Data sets can be easily integrated with visual tracker benchmarks. It includes all bounding boxes and attribute annotations of the UAV dataset. As shown in Fig. 8, we compare the UATracker with state-of-the-art trackers including Dimp50, Dimp18, ATOM, SiamRPN++, DaSiamRPN, SiamRPN, ECO, ECO-HC and SRDCF. Specifically, compared to Dimp50, our UATracker improved the AUC score by 2.2%, and increased the AUC score by 2.4% in the occlusion category.

Experiments on the VOT2018 Dataset. In the visual object tracking (VOT) benchmark, we choose VOT2018 (Kristan et al. 2018) to evaluate our tracker. VOT2018 includes 60 public sequences with different challenging factors. The VOT benchmark evaluates trackers by using a reset-based approach. When the tracker does not overlap with the ground truth, the tracker is reinitialized after five frames. Trackers are evaluated by expected average overlap (EAO), which is the inner product of empirically estimated average overlap and typical sequence length distribution. In addition, accuracy (A) and robustness (R) are reported.

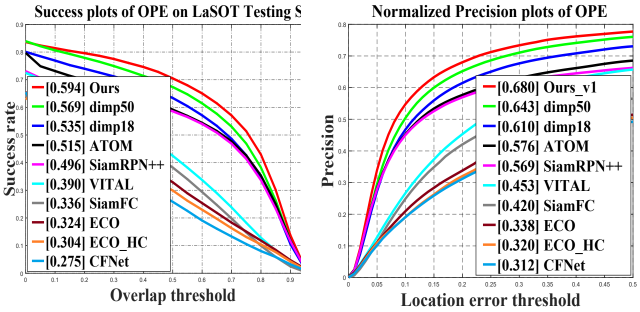

Figure 7: Normalized precision and success plots on the LaSOT testing set.

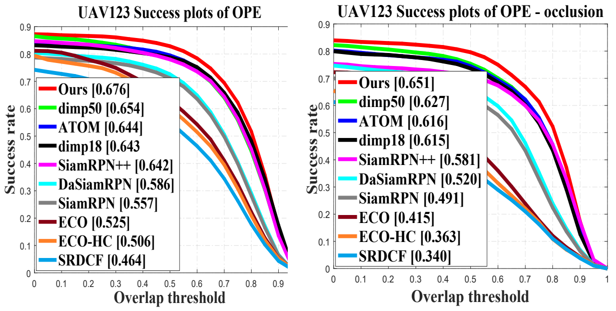

Figure 8: Success plot on the UAV123 benchmark.

We compare our UATracker with the 9 state-of-the-art trackers on VOT-2018 in Fig. 6. Table 3 reports the details of the comparison with Dimp50 (Bhat et al. 2019), SiamRPN++ (Li et al. 2019), ATOM (Danelljan et al. 2019), LADCF (Xu et al. 2018), MFT (Kristan et al. 2018), SiamMask (Wang et al. 2019), DaSiamRPN (Zhu et al. 2018), UPDT (Bhat et al. 2018) and RCO (Kristan et al. 2018). Our UATracker achieves the best accuracy and EAO. Our EAO score of 0.458 is significantly better than Dimp50, SiamRPN++ and other state-of-the-art trackers. In particular, our tracker outperforms the state-of-the-art Dimp50 by 1.8%, SiamRPN++ by 4.4% and ATOM by 5.7%, significant margins for object tracking on this challenging benchmark.

Conclusion

In this paper, we proposed the significant sample selection (S3) approach, which incorporates uncertainty estimation into the tracking framework and later relies on it to select more representative frame samples as a training set for the classifier branch of the network. This strategy better filters out noisy samples and makes the tracker demonstrably more robust. Moreover, we introduced two further improvements including a novel data augmentation procedure to increase robustness, and all our improvements fit together in one model, which we refer to as the UATracker and is trainable in a joint and end-to-end fashion. Experiments on the OTB100, LaSOT, UAV123 and VOT2018 benchmarks demonstrate that the proposed tracker improves object tracking performance, with striking contrast with the state-of-the-arts.

References

Bertinetto, L.; Valmadre, J.; Golodetz, S.; Miksik, O.; and Torr, P. H. S. 2016a. Staple: Complementary Learners for Real-Time Tracking. In *CVPR*.

Bertinetto, L.; Valmadre, J.; Henriques, J. F.; Vedaldi, A.; and Torr, P. H. S. 2016b. Fully-Convolutional Siamese Networks for Object Tracking. In Hua, G.; and Jégou, H., eds., *ECCVw*, 850–865. Cham: Springer International Publishing.

Bhat, G.; Danelljan, M.; Gool, L. V.; and Timofte, R. 2019. Learning Discriminative Model Prediction for Tracking. In *ICCV*.

Bhat, G.; Johnander, J.; Danelljan, M.; Khan, F. S.; and Felsberg, M. 2018. Unveiling the Power of Deep Tracking. In *CVPR*.

Chen, Y.; Tai, L.; Sun, K.; and Li, M. 2020. MonoPair: Monocular 3D Object Detection Using Pairwise Spatial Relationships. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Choi, J.; Chun, D.; Kim, H.; and Lee, H.-J. 2019. Gaussian YOLOv3: An Accurate and Fast Object Detector Using Localization Uncertainty for Autonomous Driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

Dai, J.; Qi, H.; Xiong, Y.; Li, Y.; Zhang, G.; Hu, H.; and Wei, Y. 2017. Deformable Convolutional Networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

Danelljan, M.; Bhat, G.; Khan, F. S.; and Felsberg, M. 2019. ATOM: Accurate Tracking by Overlap Maximization. In *CVPR*.

Danelljan, M.; Bhat, G.; Shahbaz Khan, F.; and Felsberg, M. 2017. ECO: Efficient Convolution Operators for Tracking. In *CVPR*.

Danelljan, M.; Hager, G.; Shahbaz Khan, F.; and Felsberg, M. 2015. Learning Spatially Regularized Correlation Filters for Visual Tracking. In *ICCV*.

Danelljan, M.; Robinson, A.; Shahbaz Khan, F.; and Felsberg, M. 2016. Beyond Correlation Filters: Learning Continuous Convolution Operators for Visual Tracking. In Leibe, B.; Matas, J.; Sebe, N.; and Welling, M., eds., *ECCV*, 472–488. Cham: Springer International Publishing.

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 *CVPR*.

Feng, D.; Rosenbaum, L.; and Dietmayer, K. 2018. Towards Safe Autonomous Driving: Capture Uncertainty in the Deep Neural Network For Lidar 3D Vehicle Detection. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, 3266–3273.

Guo, Q.; Wei, F.; Zhou, C.; Rui, H.; and Song, W. 2017. Learning Dynamic Siamese Network for Visual Object Tracking. In *ICCV*.

Han, B.; Sim, J.; and Adam, H. 2017. BranchOut: Regularization for Online Ensemble Tracking with Convolutional Neural Networks. In *CVPR*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep Residual Learning for Image Recognition. In *CVPR*.

Henriques, J. F.; Caseiro, R.; Martins, P.; and Batista, J. 2015. High-Speed Tracking with Kernelized Correlation Filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37(3): 583–596. ISSN 1939-3539. doi:10.1109/TPAMI.2014.2345390.

Hong, S.; You, T.; Kwak, S.; and Han, B. 2015. Online Tracking by Learning Discriminative Saliency Map with Convolutional Neural Network. In *ICML*.

Kendall, A.; and Gal, Y. 2017. What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision? In Guyon, I.; Luxburg, U. V.; Bengio, S.; Wallach, H.; Fergus, R.; Vishwanathan, S.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 30*, 5574–5584. Curran Associates, Inc. URL http://papers.nips.cc/paper/7141-what-uncertainties-do-we-need-in-bayesian-deep-learning-for-computer-vision.pdf.

Kristan, M.; Leonardis, A.; Matas, J.; Felsberg, M.; Pflugfelder, R. P.; Zajc, L. C.; Vojir, T.; Bhat, G.; Lukežic, A.; Eldesokey, A.; et al. 2018. The Sixth Visual Object Tracking VOT2018 Challenge Results. In *ECCV*, 3–53.

Li, B.; Wu, W.; Wang, Q.; Zhang, F.; Xing, J.; and Yan, J. 2019. SiamRPN++: Evolution of Siamese Visual Tracking With Very Deep Networks. In *CVPR*.

Li, B.; Yan, J.; Wu, W.; Zhu, Z.; and Hu, X. 2018. High Performance Visual Tracking With Siamese Region Proposal Network. In *CVPR*.

Li, H.; Li, Y.; and Porikli, F. 2015. DeepTrack: Learning Discriminative Feature Representations Online for Robust Visual Tracking. *IEEE Transactions on Image Processing* 25(4): 1834–1848.

Liu, T.; Wang, G.; and Yang, Q. 2015. Real-time part-based visual tracking via adaptive correlation filters. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 4902–4912. doi:10.1109/CVPR.2015.7299124.

Ma, C.; Huang, J.-B.; Yang, X.; and Yang, M.-H. 2015. Hierarchical Convolutional Features for Visual Tracking. In *ICCV*.

Mueller, M.; Smith, N.; and Ghanem, B. 2016. A Benchmark and Simulator for UAV Tracking. In Leibe, B.; Matas, J.; Sebe, N.; and Welling, M., eds., *Computer Vision – ECCV 2016*, 445–461. Cham: Springer International Publishing. ISBN 978-3-319-46448-0.

Nam, H.; Baek; Mooyeol; and Han, B. 2016. Modeling and Propagating CNNs in a Tree Structure for Visual Tracking. In *CVPR*.

Nam, H.; and Han, B. 2015. Learning Multi-Domain Convolutional Neural Networks for Visual Tracking. In *CVPR*.

Song, Y.; Ma, C.; Wu, X.; Gong, L.; Bao, L.; Zuo, W.; Shen, C.; Lau, R. W.; and Yang, M.-H. 2018. VITAL: VIsual Tracking via Adversarial Learning. In *CVPR*.

Tao, R.; Gavves, E.; and Smeulders, A. W. M. 2016. Siamese Instance Search for Tracking. In *CVPR*.

Valmadre, J.; Bertinetto, L.; Henriques, J.; Vedaldi, A.; and Torr, P. H. S. 2017. End-To-End Representation Learning for Correlation Filter Based Tracking. In *CVPR*.

Wang, Q.; Teng, Z.; Xing, J.; Gao, J.; Hu, W.; and Maybank, S. 2018. Learning Attentions: Residual Attentional Siamese Network for High Performance Online Visual Tracking. In *CVPR*.

Wang, Q.; Zhang, L.; Bertinetto, L.; Hu, W.; and Torr, P. H. 2019. Fast Online Object Tracking and Segmentation: A Unifying Approach. In *CVPR*.

Wu, Y.; Lim, J.; and Yang, M.-H. 2015. Object Tracking Benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37(9): 1834–1848.

Xu, T.; Feng, Z. H.; Wu, X. J.; and Kittler, J. 2018. Learning Adaptive Discriminative Correlation Filters via Temporal Consistency Preserving Spatial Feature Selection for Robust Visual Tracking. *IEEE Transactions on Image Processing* .

Zhang, H.; Cissé, M.; Dauphin, Y. N.; and Lopez-Paz, D. 2017. mixup: Beyond Empirical Risk Minimization. *CoRR* abs/1710.09412. URL http://arxiv.org/abs/1710.09412.

Zhang, L.; Gonzalez-Garcia, A.; Weijer, J. v. d.; Danelljan, M.; and Khan, F. S. 2019a. Learning the Model Update for Siamese Trackers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

Zhang, X.; Wan, F.; Liu, C.; Ji, R.; and Ye, Q. 2019b. FreeAnchor: Learning to Match Anchors for Visual Object Detection. In *NeurIPS*, 147–155.

Zhu, X.; Hu, H.; Lin, S.; and Dai, J. 2019. Deformable ConvNets V2: More Deformable, Better Results. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Zhu, Z.; Wang, Q.; Li, B.; Wu, W.; Yan, J.; and Hu, W. 2018. Distractor-aware Siamese Networks for Visual Object Tracking. In *ECCV*.