ÉCOLE CENTRALE DE LYON

MOS 4.4

New Technologies of Information and Communication

Deep Learning based techniques for object tracking in image sequences

---

Student

PEREZ MARTIN

Professor

LIMING CHEN

12 March 2019

Table des Matières

Introduction

Motivation

Tracking object image is a challenging problem, traditionally tackled using engineered features. With the recent surge of deep learning methods in image analysis competitions, new tracker architecture based on deep neural networks might be used for object tracking. In particular, ImageNet Large Scale Visual Recognition Challenge (ILSVRC) showed the improved performance of CNN for object localization, which is of great interest for object tracking.

The study purpose is to review recent object tracking method that are based on deep learning models, and to discuss its performance and advantages compared to traditional methods. This study has no ambition to analyze every deep learning tracker, but rather give an overview of the tracking techniques.

Limits of the study

Object tracking consist in detecting and localizing one or more targets in a frame, and to track these targets among frame. Most multi-target trackers usually have one tracker for each object, so mostly single target tracker will be reviewed here.

Object tracking differs from feature point tracking or optical flow in that the desired output is a bounding box [8] or a confidence map [7] from which a bounding box can be extracted. This is so far no characterization of a good bounding box for localization, so trackers are evaluated with respect to annotated datasets, i.e. where the bounding box are explicitly given for each frame of a video sequence. Popular object tracking datasets and trackers evaluation protocols are presented is the section 2.

Figure 1 : Bounding box samples [31]

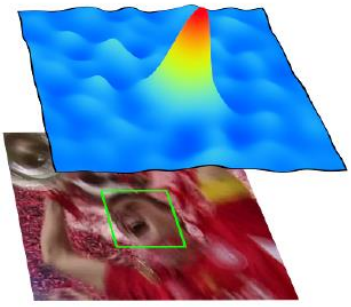


Figure 2 :Confidence map and corresponding bounding box [7]

The aim of this scientific monitoring is to review deep learning techniques that allow object tracking in sequences of images, and its options compared to traditional computer vision approaches. The study takes into account self-motion of the camera (robot, drone) as well as target objects moving in the field of view of a camera.

Also point feature tracking or other techniques such as generative models (e.g. 3D body mesh generation [14]) or instance segmentation might have interesting application for object tracking, the present study will focus on bound box trackers.

Tracking dataset and evaluation

The problem of object tracking has traditionally tackled by learning the appearance of objects to track online [2], which can be only applied to previously encountered objects.
In this study, we focus on more advanced models that allow to track that had not been seen before. By tracking arbitrary objects, a reference dataset can be used to evaluate and compare state-of-the-art trackers. Here, we present tree popular object tracking benchmark: the Visual Object Tracking (VOT) challenge, the Object Tracking Benchmark (OTB) and the Multi Object Tracking (MOT) challenge.

1.1. Visual Object Tracking challenge (VOT)

VOT short-term challenge

In the short-term challenge, tracker are evaluated on the VOT dataset. The VOT2018 dataset contains 60 sequences fully annotated with rotated bounding box [15] where the target may be occluded but does not disappear for a long time, in contrast with the long-term challenge. When the tracker fails to localize the target, i.e. when there is no overlap between predicted box and ground truth, the tracker is re-initialized. After re-initialization, the next 10 frames are ignored for evaluation, otherwise the overall performance would be severely biased. Tracking performances are evaluated by the overlap ratio between predicted bound box and ground truth, robustness or failing rate (counting re-initializations), and Expected Averaged Overlap (EAO) which is a combination of accuracy and robustness [15].

VOT long-term challenge

In the long term challenge, the target may disappear of the sequence for a long duration, so the tracker must be able to detect the absence and the reappearance of the object [15]. The dataset used in the VOT challenge is the LTB35 dataset [23] that contains 35 sequences, with a total of 14687 frames including 433 target disappearances. This video sequences contain on average 12 disappearances of at least 40 frames.

To address long-term tracking problem, tracker should not report the target when it is not visible (e.g. occlusion), and perform target re-detection after tracking failure. Trackers are evaluated with precision, recall and F-score measures, which are evaluated based on 50% overlap (correctly tracked object in one frame means the bounding box overlap ration with the ground truth is over 50%). This measurements allow to consider the localization accuracy as well as the ability of the tracker to predict the absence and re-detect the target [23].

1.2. Object Tracking Benchmark (OTB)

The object tracking benchmark (OTB) evaluates trackers on the OTB dataset, originally containing 30 video sequences and later extended to 80 video sequences, fully annotated with bounding box of the target [31].

The evaluation protocol is different from the one of the VOT challenge: for each video sequence, given the initial bounding box of target, the target is tracked until the end of the video without re-initialization, or one-pass (OP) evaluation.

Tracker are evaluated with the success plot, which measures how many predicted frames the predicted bounding box overlap with the ground truth more than a threshold. This value is calculated for a threshold varying from 0 to 1, and trackers are ranked according to their area under the curve (AUC) of the success plot.

Also, the robustness of the tracker is similarly to the VOT challenge robustness metric. It is calculated with OP evaluation, and is also declined in two other evaluations: a temporal and a spatial robustness, which measure the robustness of the tracker with perturbation with respect to the initialization [31]. The temporal robustness consist initialization the tracker with ground truth bounding box at different starting frames. For the spatial robustness, the tracker may be re-initialized with an object detector in case of failure, but the initializations (including the first one) are perturbed with random shift or scale change of the bounding box.

As discussed in [1], the OTB and VOT benchmarks propose two different evaluation protocols which lead to different ranking of tracking algorithms:

Table 1 : Tracking performances of different trackers with both OTB and VOT protocols [1] : In OTB, trackers are ranked based on their AUC scores; in VOT trackers are ranked according to a combination of their robustness and accuracy

Ranking	OTB				VOT			
	Original	Mirrored	Average rank	Overall rank	Original	Mirrored	Average rank	Overall rank
1	DFTo (0.405)	CSKm (0.408)	CSK (0.400)	CSKm	CSKo (2.29)	CTm (2.47)	CSK (2.58)	CSKo
				DFTo				DFTo
2	CSKo (0.392)	DFTm (0.369)	DFT (0.387)	CSKo	DFTo (2.35)	CSKm (2.86)	CT (2.63)	CTm
				DFTm				CTo
3	IVTo (0.369)	IVTm (0.360)	IVT (0.364)	IVTo	CTo (2.78)	DFTm (3.03)	DFT (2.69)	IVTo
				IVTm				CSKm
4	ORIAo (0.330)	ORIAm (0.331)	ORIA (0.330)	ORIAm	IVTo (2.78)	IVTm (3.20)	IVT (2.99)	DFTm
				ORIAo				IVTm
5	CTo (0.263)	CTm (0.269)	CT (0.266)	CTm	ORIAo (4.78)	ORIAm (3.44)	ORIA (4.11)	ORIAm
				CTo				ORIAo

1.3. Multi Object Tracking (MOT) benchmark

The MOT dataset consists of 22 video sequences, half used for training and half for test [21], with in average more than 500 frames per sequence. The video show pedestrians in different situations, with varying viewpoint and resolution, weather, and static or dynamic camera. Frames are fully annotated bounding box for every pedestrians in the scene.

For testing the trackers, the benchmarks proposes two evaluation set of metrics : CLEAR metrics and track quality measures [21]. The CLEAR metrics use the multi-object tracking accuracy (MOTA), the robustness as the standard deviation of the MOTA, and the multi-object precision (MOTP).

This metrics use the correct classification of a bounding box, namely if the predicted bounding box of a target overlap ratio with the ground truth is more than 50%. For each frame, predicted bounding box and ground truth give a set of correct classifications. Accuracy measure (MOTA) then take into account False Positive (FP), False Negative (FN) and Identity Switches of the target (IDSW) :

$$\text{MOTA} = 1 - \frac{\sum_t \text{FN}_t + \text{FP}_t + \text{IDSW}_t}{\sum_t \text{GT}_t}$$

With t the frame index and GT_t the number of ground truth object at frame t.

1. Discriminative Correlation Filters

1.1. Adaptative correlation filter

Discriminative correlation filters is one of the first data-driven tracking technique. It consists in modelling the appearance of the target object with a filter, and then tracking the object in subsequent frames by correlating the filter of the query template with templates in the search image: the location of the target is the new frame is assigned as the position which has the maximum correlation value. The filter is initialized in the first frame, and is trained online so the model can be applied to any object. The main purpose of that online training is to be robust to great variation in appearance of the objects (e.g. rotations, illumination, partial occlusion, etc.).

In [3], Bolme *et al.* use a spectral filter, denoted as MOSSE, which uses Fast Fourier Transform of the spatial to perform correlation in the spectral domain rather than the spatial domain, with speed up computation. The spectral filter is trained on single image sets to map the image in a 2D-gaussian centered on the center of the object. The 2D-gaussian are artificially created centered on the template bounds of the target object for the training.

The parameters of the FFT are learned by minimizing the sum of the distances between the FFT H^* of the filter h convoluted to the training images F_i with the reference 2D-gaussian g_i (where $\odot$ denotes element-wise multiplication) :

$$\min_{H^*} \sum_i \|F_i \odot H^* - G_i\|$$

The corresponding results is found by derivating the function with respect to H^* parameters, and is found to be the sum of the correlations between the images and the 2D-gaussian output normalized by the energy of the spectrum of the input :

$$H^* = \frac{\sum_i G_i \odot F_i^*}{\sum_i F_i \odot F_i^*}$$

The model were designed and applied to real-time application, with about 66 target update per second according to the author, on davidin300 video sequence (and others) and on a single CPU.

1.2. Continuous correlation filter

C-COT tracker

A more sophisticated version of the MOSSE filter [3], denoted as Continuous Correlation Filter, is introduced in [7]. This model consists in training a filter that maps the input image in a confidence map, similar to the 2D-gaussian of the previous method. However, the confidence map is continuous, meaning that sub-pixel information can be given in the training of the filter.

The approach does not train filters on images but on image features, which provide both semantic information while preserving spatial information. Standard technique would require to use features that have the same spatial resolution to apply a filter, however low-level features in CNN are as interesting as high-level features because they capture more spatial information [24]. To take the best of shallow features that provide accurate localization information, and deeper features that capture higher level information, the approached consists in spatially interpolating the features to continuous functions, to then combine features of different scales in the final model [7].

More specifically, let x be a D-channel feature with N_d values (spatial scale), extracted from a pre-trained Deep network. Let x^d be the d-th channel x, viewed as a discrete spatial function, which value of the n-th spatial location is $x^d[n]$. The interpolated channel feature is a continuous function of the spatial variable t defined by the interpolation operator J_d :

$$J_d\{x^d\}\ t\ = \sum_{n=0}^{N_d} x^d[n] \cdot \left(t - \frac{T}{N_d} n\right)$$

Where x^d is viewed as a discrete spatial function b_d is an interpolation function that is shifted for every channel d of the feature x, and is set to a periodic repetition and scaled form of the function $t \rightarrow b.\left(\frac{N_d}{T}\left(t - \frac{T}{2N_d}\right)\right)$.

This can be generalized to the 2-dimensions case, where features are extracted from images and are feature channels are 2-dimensions functions $x^d[n_1, n_2]$ where (n_1, n_2) corresponds to a part of the image. The operator is this generalized as follow:

$$J_d\{x^d\}\ t_1, t_2\ = \sum_{n_1=0}^{N_d} \sum_{n_2=0}^{N_d} x^d[n_1, n_2] \cdot b_d\left(t_1 - \frac{T}{N_d} n_1\right) \cdot b_d\left(t_2 - \frac{T}{N_d} n_2\right)$$

The final operator is a linear combination of filters $f^1, \ldots, f^D$ convolved with the interpolation functions of the feature channels $x^1, \ldots, x^D$, where the filters f^d, $d \in \{1, \ldots, D\}$ parametrize the model :

$$S_f\{x\} = \sum_{d=0}^{D} f^d \star J_d\{x^d\}$$

The filters are trained using features samples $x_1, \ldots, x_m$ and corresponding outputs values $y_1, \ldots, y_m$, that are sharp peaked 2D-gaussian centered on the target object estimated position (2D case). Then, the training aims at minimizing the cost function (f) :

$$E\ f\ = \sum_{j=1}^{m} \|S_f\{x_j\} - y_j\|^2$$

This cost function is slightly modified by Danelljan *et al.* [7] with a weighting of the different training instances and by adding a regularization term of the filters $f^1, \ldots, f^D$:

$$E\ f\ = \sum_{j=1}^{m} \alpha_j \cdot \|S_f\{x_j\} - y_j\|^2 + \sum_{d=1}^{D} \|w f^d\|^2$$

However, the filters are not trained with this equation, but is the Fourier domain: this equation can be reformulated using the Parseva's formula :

$$E\ f\ = \sum_{j=1}^{m} \alpha_j \cdot \left\| \sum_{d=1}^{D} f^d X_j^d \widehat{b_d} - y_j \right\|^2 + \sum_{d=1}^{D} \left\| w f^d \right\|^2$$

Where $\cdot$ denotes the Fourier transform, and X^d is the Fast Fourier Transform of the feature x^d :

$$X^d[k] = \sum_{n=0}^{N_d - 1} x^d[n] \cdot e^{-2i\pi k \frac{n}{N_d}}$$

In [7], the Fourier coefficients of the filters $f^d[k]$ are set to zero for $|k| > K_d$ where K_d controls the model size. Filters are then represented by its Fourier coefficients

$f^d = \left(f^d[-K_d], \ldots, f^d[K_d]\right)$ so that minimization problem can be used using the normal form of the least squares problem.

As a demonstration of the model performance, Danelljan *et al.* [7] applied their tracker to the VOT-2015 and OTB-2015 dataset, with VGG features of conv-1 and conv-5 layers as input of the filter. They show that their model (C-COT) outperform other methods, which are mostly correlation filter based trackers:

Table 2 : Results of the C-COT tracker on OTB-2015 [7] (AUC score)

	DSST	SAMF	TGPR	MEEM	LCT	HCF	Staple	SRDCF	SRDCFdecon	DeepSRDCF	**C-COT**
OTB-2015	60.6	64.7	54.0	63.4	70.1	65.5	69.9	72.9	76.7	*77.3*	**82.4**
Temple-Color	47.5	56.1	51.6	62.2	52.8	58.2	63.0	62.2	*65.8*	65.4	**70.4**

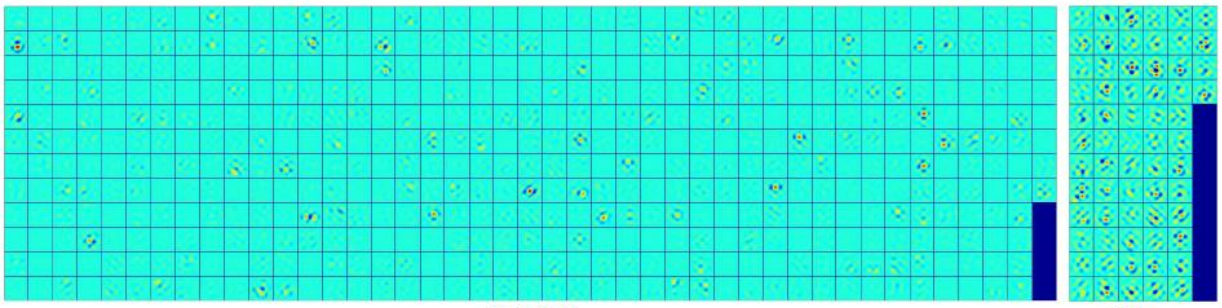
Table 3 : Results of the C-COT tracker on VOT-2015 [7] (robustness and accuracy)

	S3Tracker	RAJSSC	Struck	NSAMF	SC-EBT	sPST	LDP	SRDCF	EBT	DeepSRDCF	**C-COT**
Robustness	1.77	1.63	1.26	1.29	1.86	1.48	1.84	1.24	1.02	1.05	0.82
Accuracy	0.52	0.57	0.47	0.53	0.55	0.55	0.51	0.56	0.47	0.56	0.54

Although the accuracy of the C-COT tracker based on CNN features is twice as good as the one of MOSSE, as mentioned in [6], the model is about 1000 slower than the pioneer model MOSSE.

ECO tracker

Correlation filter based trackers have recently adopted deep network features to improve tracking performances (e.g. accuracy, robustness) with deterioration in computation time and model update speed (online update of the filter). Further improvement of C-COT [7] have been done by reducing tracking speed: Danelljan *et al.* [6] noticed that, with C-COT tracker, most defined filters (one per channel) almost did not contribute to the tracking, and deteriorate tracker computation speed as well as its robustness (overfit the training data). They expose that by representing the energy of the 512 convolution filters in the spatial domain (from Fourier coefficients) of the C-COT tracker (from last VGG layer) with are mostly empty :



(a) C-COT (b) Ours

Figure 3 : Spectral density of convolution filters [6]

To solve this problem, [6] proposes to reduce the number of filters by applying a factorized convolution, so the that operator S_f from C-COT becomes :

$$S_{Pf}\{x\} = \sum_{c=1}^{C}\sum_{d=1}^{D} p_{d,c} f^c \star J_d\{x^d\}$$

This factorized approach allow to reduce the number of filters from D to C, e.g. in the figure above $D = 512$ and $C = 64$.

Along with this improvement, [6] also proposes a novel approach for feeding input samples to the tracker, which consists in extracting different appearances of an object in a video sequence, while giving the full video sequence introduce redundancies due to the slow changes in consecutive frames and thus cause overfitting. Namely, the method consists in clustering the training images with a Gaussian Mixture Model (GMM). The GMM estimate the probability of a given input image, and has a pre-defined number of components L. After the GMM is trained, only one image is selected by cluster, resulting in using L training images instead of the whole dataset.

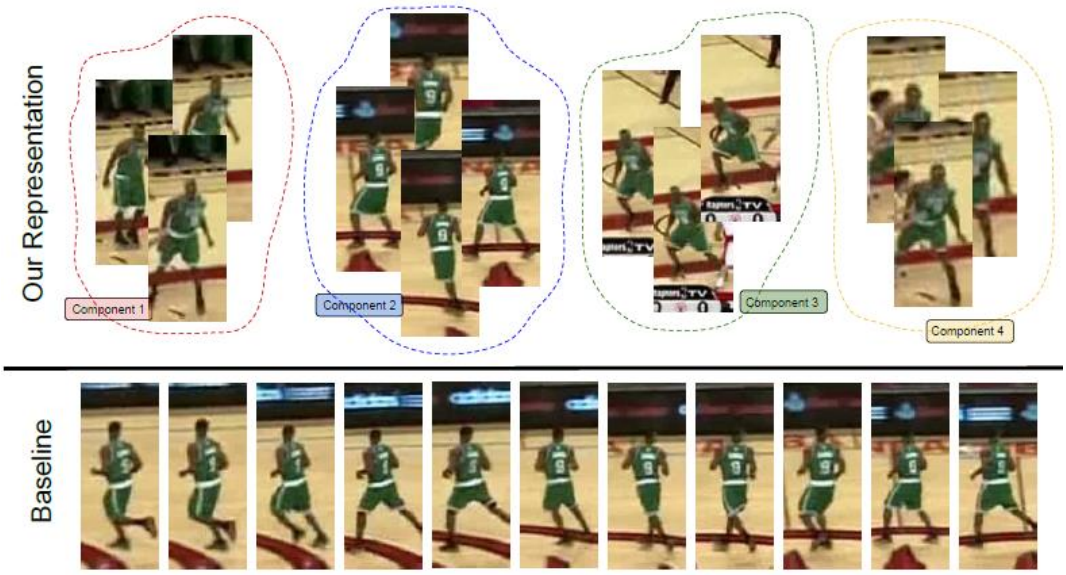


Figure 4 : Visualization of the training set : baseline uses all images whereas the clustering method only select images with different appearances of a same object [6]

The improved version of C-COT achieves better performances on VOT-2016 challenge and OTB-2016 dataset, while the computation cost is drastically decreased:

Table 4 : Results of the ECO tracker on VOT-2016 [6] compared to other state-of-the-art trackers

	SRBT [23]	EBT [39]	DDC [23]	Staple [1]	MLDF [23]	SSAT [23]	TCNN [30]	C-COT [12]	ECO-HC Ours	ECO Ours
EAO	0.290	0.291	0.293	0.295	0.311	0.321	0.325	*0.331*	0.322	**0.374**
Fail. rt.	1.25	0.90	1.23	1.35	*0.83*	1.04	0.96	0.85	1.08	**0.72**
Acc.	0.50	0.44	0.53	*0.54*	0.48	**0.57**	*0.54*	0.52	0.53	*0.54*
EFO	3.69	3.01	0.20	*11.14*	1.48	0.48	1.05	0.51	**15.13**	4.53

2. Siamese architecture tracker

2.1. Siamese architecture

A Siamese architecture aims at learning the similarity metric by learning the very high level layers [5], on top of CNN, so that same objects are very similar and different objects are highly separated. The architecture is composed of two convolutional networks: a query stream with query template as input, and a search stream with a search image as input. The distance between the outputs of the two streams gives the similarity of a templates in the search image.

The network is trained with a margin contrastive loss, so that the distance model parameters are updated so that the distance between same objects is decreased and increased between distinct objects. Siamese also used by [11].

As an example, [29] constructs its Siamese networks on top of a pre-trained CNN, namely AlexNet and VGG, and only conserving early maxpooling, as early maxpooling give robustness to noise induced by small distortions of the targets, but later maxpooling deteriorate spatial precision for tracking. Layers built on top of the CNN include *region-of-interest pooling* [10], which consist in extracting a set of region in the images which are likely to contain the target object. This regions could be selected ad hoc or be learned by the network [10]. Overall, the use of region-of-interest allow to drastically accelerate computation to localize the target in a wide search image.

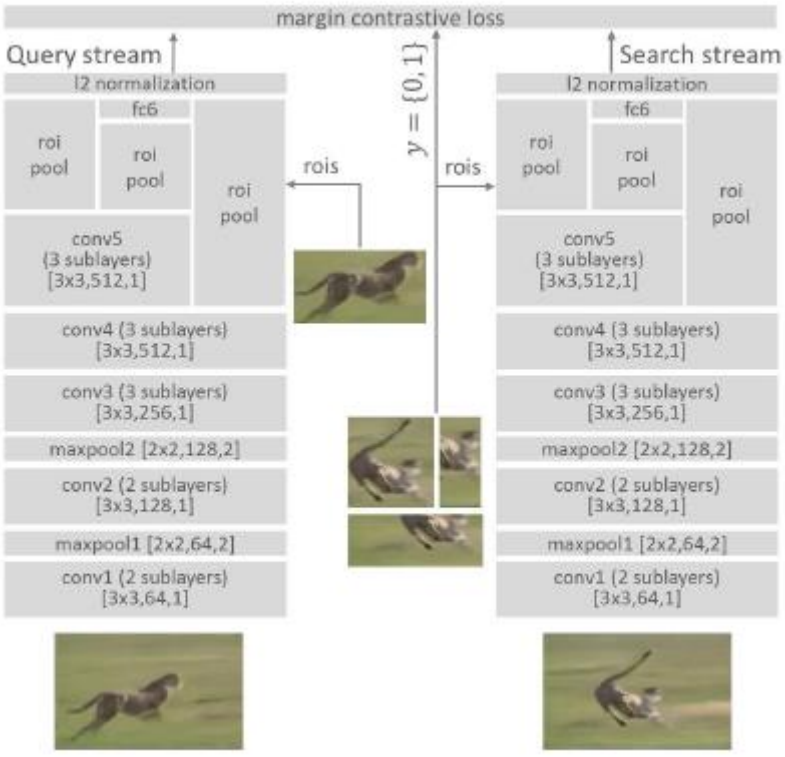


Figure 5 : example of Siamese architecture [29]

As another example, Bertino *et al.* [2] also use a fully-convolutional Siamese architecture (without any Fully-Connected layer), nevertheless designed and trained the network for the specific task of tracking rather than using a pre-trained one. The siamese stream computes a convolutional embedding with a structure similar to the convolutional stage of AlexNet [20]:

				Activation size		
Layer	Support	Chan. map	Stride	for exemplar	for search	chans.
				127×127	255×255	$\times 3$
conv1	11×11	96×3	2	59×59	123×123	$\times 96$
pool1	3×3		2	29×29	61×61	$\times 96$
conv2	5×5	256×48	1	25×25	57×57	$\times 256$
pool2	3×3		2	12×12	28×28	$\times 256$
conv3	3×3	384×256	1	10×10	26×26	$\times 192$
conv4	3×3	384×192	1	8×8	24×24	$\times 192$
conv5	3×3	256×192	1	6×6	22×22	$\times 128$

Figure 6 : CNN components of the Siamese streams designed for object tracking [2]

For a given location candidate in the image for the target position in the new frame, a template is extracted around the candidate location. Then, the corresponding embedding is computed and the similarity is calculated as the cross-correlation of the two embedding.

The similarity is calculated for every location in a search image and at different scales, to find the maximum of similarity with the query image embedding, while penalizing distant of at least 4 times the size of the query image. As Bertino *et al.* [2] point out, the model can achieve tracking on over 58 to 86 frames per second (depending on the number of scales that are explored) on NVIDIA GeForce GTX Titan X GPU, due to the fact that every embedding are simultaneously computed by the fully-convolutional network.

2.2. Region Proposal Network

Standard region proposal network

Li *et al.* applied Region Proposal Network (RPN) for tracking, which was first introduced in [28] and designed to accelerate convolution networks for real-time application. RPN takes an image as input and outputs a set of rectangular regions (anchors) for an object to localize, and the corresponding confidence score that the object is or is not in this regions. In tracking, RPN allows to look for the target in a given number of regions and not the entire image, which require much less computation.

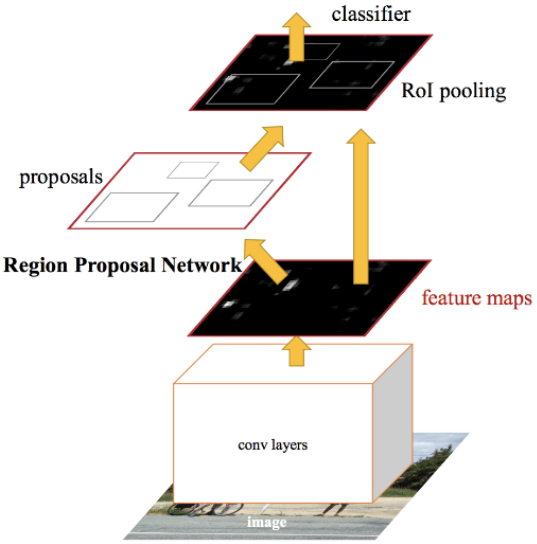


Figure 7 : Original RPN architecture for object detection [28]

Concretely, a Region Proposal Network, built with a regression branch and a classification branch, is built on top of the last convolution layer of a CNN. A sliding window on the features map of the last convolution layer map into a lower dimension vector than the feature map (e.g. dimension d-256 from d-512). Then, for each position of the sliding window, two branch consisting of 1x1 convolution layers, resp. regression and classification, outputs $4k$ coordinates encoding k anchors (regression branch), and $2k$ scores corresponding to the probability that the object is in the anchor (classification branch).

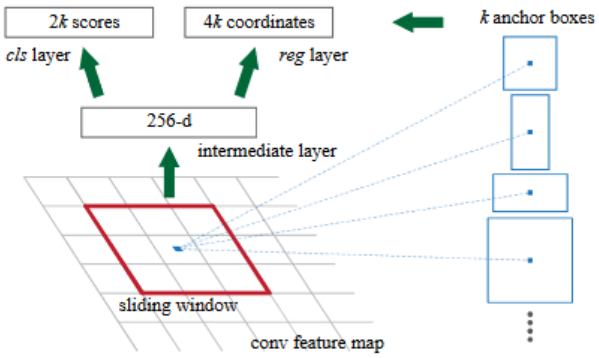


Figure 8 : Extraction anchors from map features with RPN [28]

The difference with [28] is that [22] uses a Siamese architecture: convolution layers are applied to the query template in the query stream, and its output is convolved with the regression and classification branch to output anchors and scores that take into account information of the target object and its localization in the query frame to.

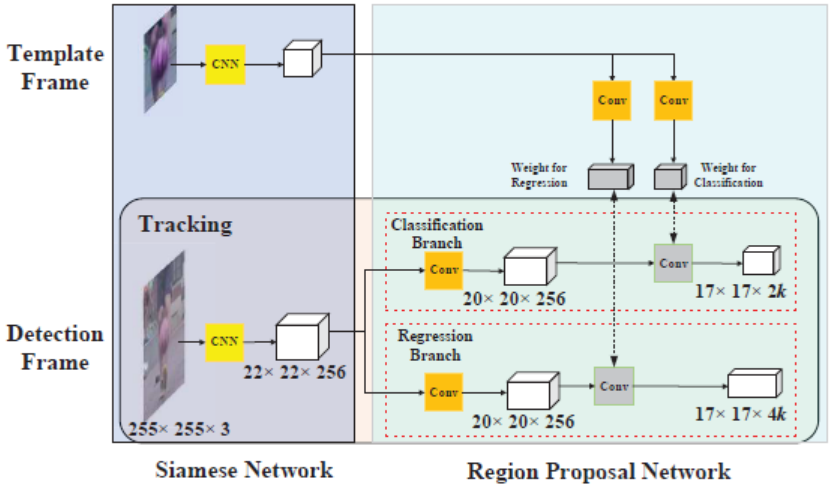


Figure 9 : Siamese-RPN pipeline [22] with a classification branch that outputs region proposal, and regression branch that output scores

Finally, to track the target in the new frame, the top K regions are extracted from the region proposal network. Then, the scores of the K proposed regions are passed to a consine window that penalize large displacement of the target with respect to the previous frame, and then multiplied by a function that penalize scale change:

$$\text{scale penalty} = e^{k \cdot \max\left(\frac{r}{r'}, \frac{r'}{r}\right) \cdot \left(\frac{s}{s'}, \frac{s'}{s}\right)}$$

Where r (resp. r') is the ratio height-width of the bounding box of the query frame (resp. of the target frame), s (resp. s') is the scale of the bounding box of the query frame (resp. of the target frame), and k is an hyper-parameter, along with the cosine window size.

Extension of the RPN architecture

The winner of the VOT2018 long-term challenge [33] adopts a region proposal network as presented in the previous section, which they refer as the *regression network* which is extended with a *verification network*.

The regression network outputs region proposals or anchors, and corresponding confidence score that an anchor contains the target. The feature extraction network is based on Mobile Net architecture [12], but search and query streams have different parameters as the query frame takes 127x127 template images and the search stream takes 300x300 input search images. Features from both streams are then merged by the protocol in figure 10:

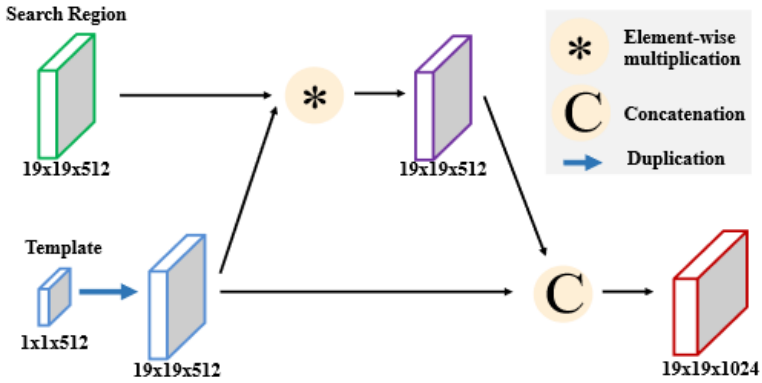


Figure 10 : features fusion in [33] merging 19x19x512 image features and 1x1x512 template features into a single 19x19x1024 map

The merged features are fed to the region proposal network that outputs anchors and corresponding confidence scores that the region contains the target, and this is passed to the verification network.

The verification network outputs a single region and a confidence score that the candidate proposal is foreground or background and is domain-specific network. Namely, the verification network is updated online to specializes in tracking that particular target, motivated by the performance of MDNet [25] that used an offline trained CNN extended with domain-specific online trained networks.

In particular, because the verification network is specific of the target, it is used as an absence detector: if the confidence score of the template output by the verification network is under a threshold, then the target is detected as absent of the frame, which makes the model suitable for long-term tracking [15].

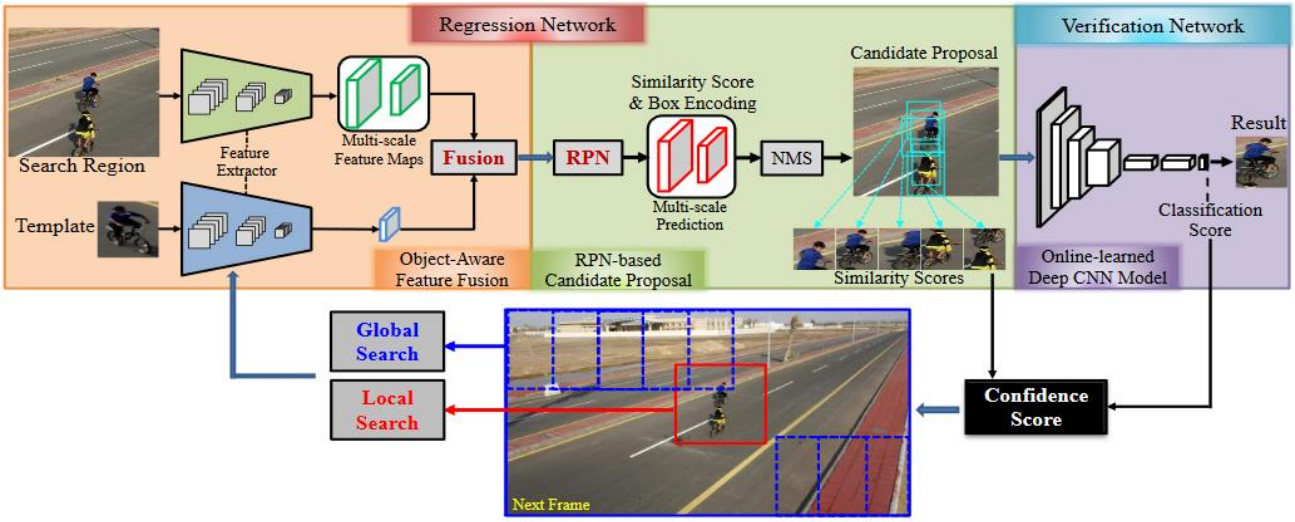


Figure 11 : Overview of the regression-verification pipeline [33]

3. Recurrent network based tracker

In parallel of the Siamese architecture, which aims at localizing the target in the search image given a previous image, agent-based models trained with deep reinforcement learning [13] have been applied to object tracking. The decision consist in determining how the bounding box of the target will change in the search image: the change is a combination of elementary changes, namely shift to the left, right, up and down, scale up or down or stop.

Usually, the action model is combined with recurrent neural network (RNN) : [9],[13], or Long-Short-Term-Memory (LSTM) [8] to have some memory of the target appearance and bounding box position and predict the bounding box in the search image. A typical tracker pipeline looks like figure 12 :

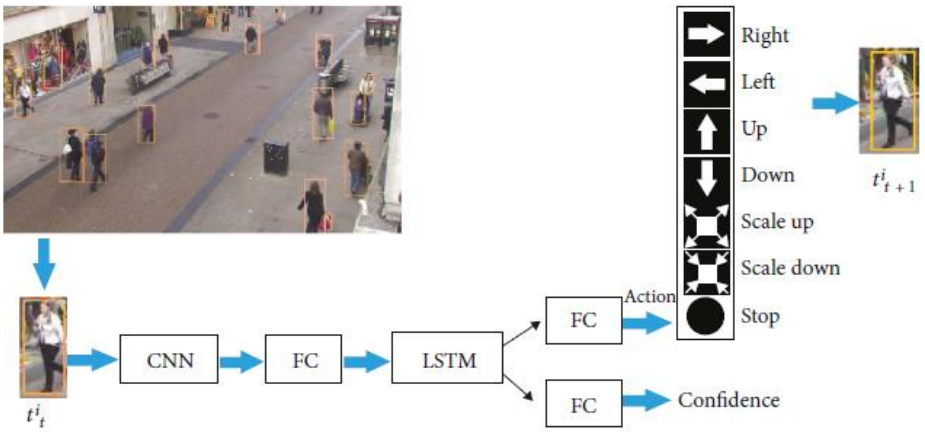


Figure 12 : Single object tracker pipeline based on decision agent |8]

The action model forces object to perform elementary displacements, with make it particularly suitable to occlusion compared to other methods which imply templates association.

The LSTM-based model in [8] achieve state-of-the-art performance in accuracy and robustness on the MOT benchmark [21]. Such agent-based models have mostly applied to multi-object tracking so far, because recurrent networks enable to different trackers which share the same pipeline, while the information of the target is capture by the hidden state of the LSTM. In contrast, online updated models, such as verification network in [33] require to build and update a network for every single object.

Also, because objects could be anywhere in the image, YOLO detector is used in [8] for deep features computation: YOLO detector split the images in a sparse set of regions, in contrast to the popular Region Proposal Network approach in single-object tracking.

Agent based trackers have one major issue however, in that it require very large dataset to be trained with reinforcement learning. Models are usually trained in synthetic dataset, e.g. trained in Unreal Engine or ViZDoom simulators.

Discussion

The present report reviews the two most popular object tracking benchmark for evaluating trackers, namely OTB and VOT challenges. Without the ambition to exhaustively review every method used in object tracking, this report presents a few state-of-the-art methods and the previous works on which it is based, and discusses the tracking problem this methods addresses.

Overall, use of Convolutional Neural Networks (CNN) features in tracking techniques have gradually increased over the past 3 years ([15],[16],[17],[18]). Traditional techniques, namely discriminative correlation filters and engineered features such as HOG and histograms of colors are still used in the VOT challenge [references], however the top performance trackers of the VOT challenge nearly all use deep features. Convolution networks are either fully trained for the tracking purpose or pre-trained networks, in which case image features are extracted from low-level layer, that capture more spatial information [24], [29].

In particular, the Siamese architecture [29], presented in section 2.2, have been widely used in the VOT long term and real-time challenges. Indeed, CNN-based models trade good performance for computation cost, but the use of GPU make it suitable for real-time tracking.

Recent tracker models seem to gradually make the model more complex by extending features CNN with further networks: Region Proposal Network (RPN) [22] and online trained network [25] have shown great tracking performances recently. The two top ranked long-term trackers of the VOT2018 challenge ([34],[33]) use a Siamese architecture extended with a RPN and an online trained Verification Network to better identify the target and detect its absence. With the success of the verification network [33], many online trained network on top of deep convolutional networks might be applied to object tracking in the next object tracking challenges, as the only training phase made the success made of success of correlation filters.

In addition, more and more methods based LSTM and Markov Decision Process (MDP) models trained with deep reinforcement learning seem to be applied to the tracking task. In particular, these models have an additional advantage that the tracker can easily be adapted to track multiple objects: the trackers share the same architecture and parameters, whereas the state of the LSTM is specific to a target. In [36], agent based tracker coupled with camera control was applied to single object tracking on VOT benchmark; [35] proposed to operate online model update with a meta-learning pipeline (based on RNN) that learns a model (e.g. correlation filter tracker) how to learn in the online tracking phase.

References

[1] Bei S, Zhen Z, Wusheng L, Liebo D, Qin L. Visual object tracking challenges revisited: VOT vs. OTB. *PLoS One*. 2018;13(9):e0203188. Published 2018 Sep 27. doi:10.1371/journal.pone.0203188

[2] Bertinetto, L., Valmadre, J., Henriques, J., Torr, P.H.S., Vedaldi, A.: Fully convolutional siamese networks for object tracking. In: ECCV Workshops. pp. 850-865 (2016)

[3] Bolme, D.S., Beveridge, J.R., Draper, B.A., Lui, Y.M.: Visual object tracking using adaptive correlation filters. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2010)

[4] Bromley, J., Bentz, J.W., Bottou, L., Guyon, I., LeCun, Y., Moore, C., Säckinger, E., Shah, R.: Signature verification using a siamese time delay neural network. International Journal of Pattern Recognition and Artificial Intelligence, 7(04):669–688, 1993. 2, 3

[5] Chopra, S., Hadsell, R., LeCun, Y.: Learning a similarity metric discriminatively, with application to face verification. In CVPR (2005)

[6] Danelljan, M., Bhat, G., Khan, F.S., Felsberg, M.: ECO: efficient convolution operators for tracking. In: CVPR (2017)

[7] Danelljan, M., Robinson, A., Shahbaz Khan, F., Felsberg, M.: Beyond correlation filters: Learning continuous convolution operators for visual tracking. In: ECCV. pp. 472-488 (2016)

[8] Jiang, M.X., Deng, C., Pan, Z.G., Wang, L.F., Sun, X.: Multiobject Tracking in Videos Based on LSTM and Deep Reinforcement Learning. Complexity, vol. 2018, Article ID 4695890 (2018) https://doi.org/10.1155/2018/4695890

[9] Gan, Q., Guo, Q., Zhang, Z., Cho, K.: First step toward model-free, anonymous object tracking with recurrent neural networks. arXiv CoRR (2015)

[10] Girshick, R.: Fast R-CNN, In: 2015 IEEE International Conference on Computer Vision (ICCV), pp. 1440-1448 (2015)

[11] Han, X., Leung, T., Jia, Y., Sukthankar, R., Berg, A. C.: MatchNet: Unifying feature and metric learning for patch-based matching. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3279-3286 (2015)

[12] Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H.: Mobilenets: Efficient convolutional neural networks for mobile vision applications. In: arXiv preprint arXiv:1704.04861 (2017)

[13] Kahou, S.E., Michalski, V., Memisevic, R.: RATM: Recurrent Attentive Tracking Model. arXiv CoRR (2015)

[14] Kanazawa, A., Black, M.J., Jacobs, D.W., Malik, J.: End-to-end recovery of human shape and pose. CoRR abs/1712.06584 (2018) https://arxiv.org/abs/1712.06584

[15] Kristan, M., Leonardis, A., Matas, J., Felsberg, M., Pugfelder, R., Cehovin, L., Vojír, T., Häger, G., Lukežič, A., Eldesokey, A., Fernández, G., et al.: vot2018 challenge results. In: ECCV2018 Workshops, Workshop on visual object tracking challenge (2018)

[16] Kristan, M., Leonardis, A., Matas, J., Felsberg, M., Pugfelder, R., Cehovin, L., Vojír, T., Häger, G., Lukežič, A., Eldesokey, A., Fernández, G., et al.: The visual object tracking vot2017 challenge results. In: ECCV2017 Workshops, Workshop on visual object tracking challenge (2017)

[17] Kristan, M., Leonardis, A., Matas, J., Felsberg, M., Pugfelder, R., Cehovin, L., Vojír, T., Häger, G., Lukežič, A., Eldesokey, A., Fernández, G., et al.: vot2016 challenge results. In: ECCV2016 Workshops, Workshop on visual object tracking challenge (2016)

[18] Kristan, M., Leonardis, A., Matas, J., Felsberg, M., Pugfelder, R., Cehovin, L., Vojír, T., Häger, G., Lukežič, A., Eldesokey, A., Fernández, G., et al.: vot2016 challenge results. In: ECCV2015 Workshops, Workshop on visual object tracking challenge (2015)

[19] Kristan, M., Matas, J., Leonardis, A., Vojír, T., Pugfelder, R., Fernández, G., Nebehay, G., Porikli, F., Cehovin, L.: A novel performance evaluation methodology for single-target trackers. IEEE Transactions on Pattern Analysis and Machine Intelligence 38(11), 2137-2155 (2016)

[20] Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classication with deep convolutional neural networks. In: NIPS 2012

[21] Leal-Taixé, L., Milan, A., Reid, I., Roth, S., Schindler, K.: MOTChallenge 2015: Towards a benchmark for multi-target tracking. CoRR abs/1504.01942 (2015), http://arxiv.org/abs/1504.01942

[22] Li, B., Yan, J., Wu, W., Zhu, Z., Hu, X.: High performance visual tracking with siamese region proposal network. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018)

[23] Lukezic, A., Zajc, L.C., Vojr, T., Matas, J., Kristan, M.: Now you see me: evaluating performance in long-term visual tracking. CoRR abs/1804.07056 (2018), http://arxiv.org/abs/1804.07056

[24] Ma, C., Huang, J.B., Yang, X., Zhang, C., Yang, M.H.: Hierarchical convolution features for visual tracking. In: 2015 EEE International Conference on Computer Vision (2015)

[25] Nam, H., Han, B.: Learning multi-domain convolutional neural networks for visual tracking. In: CVPR. pp. 4293-4302 (2016)

[26] Ondrúška, P., Posner, I. : Deep tracking : seeing beyond seeing using recurrent neural networks. In: AAA-16 conference (2016)

[27] Qi, Y., Zhang, S., Qin, L., Yao, H., Huang, Q., Lim, J., Yang, M.H. et al.: Hedged deep tracking. In: CVPR. pp4303-4311 (2016)

[28] Ren., S., He, K., Girshick, R., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks, CoRR abs/1506.01497 (2015) https://arxiv.org/abs/1506.01497

[29] Tao, R., Gavves, E., Smeulders, A.W.M.: Siamese instance search for tracking, CoRR abs/1605.05863 (2016) https://arxiv.org/abs/1605.05863

[30] Vondrick, C., Shrivastava, A., Fathi, A., Guadarrama, S., Murphy, K.: Tracking emerges by coloziring videos. CoRR abs/1806.09594 (2018) https://arxiv.org/abs/1806.09594

[31] Wu, Y., Lim, J., Yang, M.H.: Object tracking benchmark. TPAMI 37(9), 1834-1848 (2015)

[32] Yun, D., Choi, J., Yoo, Y., Yun, K., Choi, J.Y.: Action-decision networks for visual tracking with deep reinforcement learning. In: CVPR (2017)

[33] Zhang, Yunhua & Wang, Dong & Wang, Lijun & Qi, Jinqing & Lu, Huchuan. Learning regression and verification networks for long-term visual tracking. CoRR abs/1809.04320 (2018), https://arxiv.org/pdf/1809.04320.pdf

[34] Zhu, Z., Wang, Q., Li, B., Wu, W., Yan, J., Hu, W.: Distractor-aware siamese networks for visual object tracking. CoRR abs/1808.06048 (2018) https://arxiv.org/abs/1808.06048

[35] Li, B., Xie, W., Zeng W., Liu, W.: Learning to Update for Object Tracking with Recurrent Meta-learner. In: IEEE Transactions on Image Processing (2019) https://doi: 10.1109/TIP.2019/2900577

[36] Luo, W. , Sun, P., Zhong, F., Liu, W., Zhang T., Wang, Y.: End-to-end Active Object Tracking and Its Real-world Deployment via Reinforcement Learning. In IEEE Transactions on Pattern Analysis and Machine Intelligence (2019) https://doi: 10.1109/TPAMI.2019.2899570