End-to-end representation learning for Correlation Filter based tracking
Supplementary material

Jack Valmadre* Luca Bertinetto* João Henriques Andrea Vedaldi Philip H. S. Torr
University of Oxford
{name.surname}@eng.ox.ac.uk

C. Correlation Filter formulation

C.1. Kernel linear regression

First, consider the general linear regression problem of learning the weight vector w that best maps each of n example input vectors $x_i \in \mathbb{R}^d$ to their target $y_i \in \mathbb{R}$. The squared error can be expressed

$$\frac{1}{2n}\sum_{i=1}^{n}(x_i^T w - y_i)^2 = \frac{1}{2n}\|X^T w - y\|^2 \tag{1}$$

where $X \in \mathbb{R}^{d\times n}$ is a matrix whose columns are the example vectors and $y \in \mathbb{R}^n$ is a vector of the targets. Incorporating regularization, the problem is

$$\arg\min_w \quad \tfrac{1}{2n}\|X^T w - y\|^2 + \tfrac{\lambda}{2}\|w\|^2 \ . \tag{2}$$

Kernel linear regression can be developed by writing this as a constrained optimization problem

$$\begin{aligned} \arg\min_{w,r} \quad & \tfrac{1}{2n}\|r\|^2 + \tfrac{\lambda}{2}\|w\|^2 \\ \text{subject to} \quad & r = X^T w - y \end{aligned} \tag{3}$$

and then finding a saddle point of the Lagrangian

$$L(w, r, v) = \tfrac{1}{2n}\|r\|^2 + \tfrac{\lambda}{2}\|w\|^2 + v^T(r - X^T w + y) \ . \tag{4}$$

The final solution can be obtained from the dual variable

$$w = \tfrac{1}{\lambda} X v \tag{5}$$

and the solution to the dual problem is

$$v = \tfrac{\lambda}{n} K^{-1} y \tag{6}$$

where $K = \frac{1}{n}X^T X + \lambda I$ is the regularized kernel matrix. It is standard to introduce a scaled dual variable $\alpha = \frac{1}{\lambda}v$ that defines w as a weighted combination of examples

$$w = X\alpha = \sum_{i=1}^{n}\alpha_i x_i \quad \text{with} \quad \alpha = \frac{1}{n}K^{-1}y \ . \tag{7}$$

*Equal first authorship.

The kernel matrix is $n \times n$ and therefore the dual solution is more efficient than the primal solution, which requires inversion of a $d \times d$ matrix, when the number of features d exceeds the number of examples n.

C.2. Single-channel Correlation Filter

Given a scalar-valued example signal x with domain $\mathcal{U}$ and corresponding target signal y, the Correlation Filter w is the scalar-valued signal

$$\arg\min_w \quad \frac{1}{2n}\|w \star x - y\|^2 + \frac{\lambda}{2}\|w\|^2 \tag{8}$$

where signals are treated as vectors in $\mathbb{R}^{\mathcal{U}}$ and the circular cross-correlation of two signals $w \star x$ is defined

$$(w \star x)[u] = \sum_{t\in\mathcal{U}} w[t]x[u + t \bmod m] \quad \forall u \in \mathcal{U} \ . \tag{9}$$

The solution from the previous section can then be used by defining X to be the matrix in $\mathbb{R}^{\mathcal{U}\times\mathcal{U}}$ such that $X^T w = w \star x$. It follows that the kernel matrix K belongs to $\mathbb{R}^{\mathcal{U}\times\mathcal{U}}$ and the dual variable α is a signal in $\mathbb{R}^{\mathcal{U}}$.

The key to the correlation filter is that the circulant structure of X enables the solution to be computed efficiently in the Fourier domain. The matrix X has elements $X[u, t] = x[u + t \bmod m]$. Since the matrix X is symmetric, the template w is obtained as cross-correlation

$$w = X\alpha = \alpha \star x \ . \tag{10}$$

The linear map defined by the kernel matrix K is equivalent to convolution with a signal k

$$Kz = k * z \quad \forall z \tag{11}$$

which is defined $k = \frac{1}{n}x \star x + \lambda\delta$, since

$$\begin{aligned} \forall z : FX^T Xz &= F((z \star x) \star x) \\ &= \widehat{z} \circ \widehat{x}^* \circ \widehat{x} = F(z * (x \star x)) \ . \end{aligned} \tag{12}$$

Therefore the solution is defined by the equations

$$\begin{cases} k = \frac{1}{n} x \star x + \lambda \delta \\ k * \alpha = \frac{1}{n} y \\ w = \alpha \star x \end{cases} \tag{13}$$

and the template can be computed efficiently in the Fourier domain

$$\begin{cases} \widehat{k} = \frac{1}{n} \widehat{x}^* \circ \widehat{x} + \lambda \mathbb{1} \\ \widehat{\alpha} = \frac{1}{n} \widehat{k}^{-1} \circ \widehat{y} \\ \widehat{w} = \widehat{\alpha}^* \circ \widehat{x} \ . \end{cases} \tag{14}$$

C.3. Multi-channel Correlation Filter

There is little advantage to the dual solution when training a single-channel Correlation Filter from the circular shifts of a single base example. However, the dual formulation is much more efficient in the multi-channel case [2].

For signals with k channels, each multi-channel signal is a collection of scalar-valued signals $x = (x_1, \dots, x_k)$, and the data term becomes

$$\| \sum_p w_p \star x_p - y \|^2 = \| \sum_p X_p^T w_p - y \|^2 \tag{15}$$

and each channel of the template is obtained from the dual variables

$$w_p = X_p \alpha = \alpha \star x_p \tag{16}$$

The solution to the dual problem is still $\alpha = \frac{1}{n} K^{-1} y$, however the kernel matrix is now given

$$K = \tfrac{1}{n} \textstyle\sum_p X_p^T X_p + \lambda I \tag{17}$$

and the linear map defined by this matrix is equivalent to convolution with the signal

$$k = \tfrac{1}{n} \textstyle\sum_p x_p \star x_p + \lambda \delta \ . \tag{18}$$

Therefore the solution is defined by the equations

$$\begin{cases} k = \frac{1}{n} \sum_p x_p \star x_p + \lambda \delta \\ k * \alpha = \frac{1}{n} y \\ w_p = \alpha \star x_p \quad \forall p \end{cases} \tag{19}$$

and the template can be computed efficiently in the Fourier domain

$$\begin{cases} \widehat{k} = \frac{1}{n} \sum_p \widehat{x}_p^* \circ \widehat{x}_p + \lambda \mathbb{1} \\ \widehat{\alpha} = \frac{1}{n} \widehat{k}^{-1} \circ \widehat{y} \\ \widehat{w}_p = \widehat{\alpha}^* \circ \widehat{x}_p \quad \forall p \ . \end{cases} \tag{20}$$

It is critical that the computation scales only linearly with the number of channels.

D. Adjoint of the differential

Consider a computational graph that computes a scalar loss $\ell \in \mathbb{R}$. Within this network, consider an intermediate function that computes $y = f(x)$ where $x \in \mathcal{X} = \mathbb{R}^m$ and $y \in \mathcal{Y} = \mathbb{R}^n$. Back-propagation computes the gradient with respect to the input $\nabla_x \ell \in \mathcal{X}$ from the gradient with respect to the output $\nabla_y \ell \in \mathcal{Y}$.

The derivative $\partial f(x) / \partial x$ is a matrix in $\mathbb{R}^{n \times m}$ whose ij-th element is the partial derivative $\partial f_i(x) / \partial x_j$. This matrix relates the gradients according to

$$(\nabla_x \ell)^T = \frac{\partial \ell}{\partial x} = \frac{\partial \ell}{\partial y} \frac{\partial y}{\partial x} = (\nabla_y \ell)^T \frac{\partial f(x)}{\partial x} \tag{21}$$

From this it is evident that the back-propagation map is the linear map which is the adjoint of that defined by the derivative. That is, if the derivative defines the linear map

$$J(u) = \frac{\partial f(x)}{\partial x} u \tag{22}$$

then the back-propagation map is the unique linear map J^* that satisfies

$$\langle J^*(v), u \rangle = \langle v, J(u) \rangle \quad \forall u \in \mathcal{X}, v \in \mathcal{Y} \tag{23}$$

and the gradient with respect to the input is obtained $\nabla_x \ell = J^*(\nabla_y \ell)$. This is the core of reverse-mode differentiation [1].

An alternative way to obtain the linear map defined by the derivative is to use differential calculus. Whereas the *derivative* represents this linear map as a matrix with respect to the standard bases, the *differential* represents the linear map as an expression $df(x; dx)$. This is valuable for working with variables that possess more interesting structure than simple vectors. This technique has previously been used for matrix structured back-propagation [3]. In this paper, we use it for circulant structured back-propagation.

E. Back-propagation for multi-channel case

The differentials of the equations that define the multi-channel CF in eq. 19 are

$$\begin{cases} dk = \frac{1}{n} \sum_p (dx_p \star x_p + x_p \star dx_p) \\ dk * \alpha + k * d\alpha = \frac{1}{n} dy \\ dw_p = d\alpha \star x_p + \alpha \star dx_p \quad \forall p \ , \end{cases} \tag{24}$$

and taking the Fourier transforms of these equations gives

$$\begin{cases} \widehat{dk} = \frac{1}{n} \sum_p \left(\widehat{dx}_p^* \circ \widehat{x}_p + \widehat{x}_p^* \circ \widehat{dx}_p \right) \\ \widehat{d\alpha} = \widehat{k}^{-1} \circ \left[\frac{1}{n} \widehat{dy} - \widehat{dk} \circ \widehat{\alpha} \right] \\ \widehat{dw}_p = \widehat{d\alpha}^* \circ \widehat{x}_p + \widehat{\alpha}^* \circ \widehat{dx}_p \quad \forall p \ . \end{cases} \tag{25}$$

Now, to find the adjoint of the map $dx \mapsto dk$, we rearrange the inner product

$$\begin{aligned}\langle Fdk, FJ_1(dx)\rangle &= \left\langle \widehat{dk}, \tfrac{1}{n}\textstyle\sum_p \left(\widehat{dx}_p^* \circ \widehat{x}_p + \widehat{x}_p^* \circ \widehat{dx}_p\right)\right\rangle \\ &= \tfrac{1}{n}\textstyle\sum_p \left[\langle \widehat{dx}_p, \widehat{dk}^* \circ \widehat{x}_p\rangle + \langle \widehat{dk} \circ \widehat{x}_p, \widehat{dx}_p\rangle\right] \\ &= \textstyle\sum_p \langle \widehat{dx}_p, \frac{2}{n}\,\mathrm{Re}\{\widehat{dk}\} \circ \widehat{x}_p\rangle \end{aligned} \tag{26}$$

to give the back-propagation map

$$\widehat{\nabla_{x_p}\ell} = \tfrac{2}{n}\widehat{x}_p \circ \mathrm{Re}\{\widehat{\nabla_k \ell}\} \quad \forall p \ . \tag{27}$$

The linear map $dk, dy \mapsto d\alpha$ is identical to the single-channel case. To find the adjoint of the map $dx, d\alpha \mapsto dw$, we examine the inner-product

$$\begin{aligned}\langle dw, J_3(dx, d\alpha)\rangle &= \textstyle\sum_p \langle \widehat{dw}_p, \widehat{d\alpha}^* \circ \widehat{x}_p + \widehat{\alpha}^* \circ \widehat{dx}_p\rangle \\ &= \left\langle \widehat{d\alpha}, \textstyle\sum_p \widehat{dw}_p^* \circ \widehat{x}_p \right\rangle + \textstyle\sum_p \langle \widehat{dw}_p \circ \widehat{\alpha}, \widehat{dx}_p\rangle \ , \end{aligned} \tag{28}$$

giving the back-propagation maps

$$\widehat{\nabla_\alpha \ell} = \textstyle\sum_p \widehat{x}_p \circ (\widehat{\nabla_{w_p}\ell})^* \ , \tag{29}$$

$$\widehat{\nabla_{x_p}\ell} = \widehat{\alpha} \circ \widehat{\nabla_{w_p}\ell} \quad \forall p \ . \tag{30}$$

Finally, combining these results gives the procedure for back-propagation in the multi-channel case

$$\begin{cases} \widehat{\nabla_\alpha \ell} = \sum_p \widehat{x}_p \circ (\widehat{\nabla_{w_p}\ell})^* \\ \widehat{\nabla_y \ell} = \frac{1}{n}\widehat{k}^{-*} \circ \widehat{\nabla_\alpha \ell} \\ \widehat{\nabla_k \ell} = -\widehat{k}^{-*} \circ \widehat{\alpha}^* \circ \widehat{\nabla_\alpha \ell} \\ \widehat{\nabla_{x_p}\ell} = \widehat{\alpha} \circ \widehat{\nabla_{w_p}\ell} + \frac{2}{n}\widehat{x}_p \circ \mathrm{Re}\{\widehat{\nabla_k \ell}\} \quad \forall p \ . \end{cases} \tag{31}$$

Again, it is important that the computation scales only linearly with the number of channels.

F. Hyperparameter optimization

The hyperparameters that define the simplistic tracking algorithm have a significant impact on the tracking accuracy. These include parameters such as the penalty for changes in scale and position and the learning rate of the template average. Choosing hyperparameters is a difficult optimization problem: we cannot use gradient descent because the function is highly discontinuous, and each function evaluation is expensive because it involves running a tracker on every sequence from multiple starting points.

For the experiments of the main paper, where we sought to make a fair comparison of different architectures, we therefore used a *natural* choice of hyperparameters that were not optimized for any particular architecture. Ideally, we would use the optimal hyperparameters for each variant, except it would have been computationally prohibitive to perform this optimization for every point in every graph in the main paper (multiple times for the points with error bars).

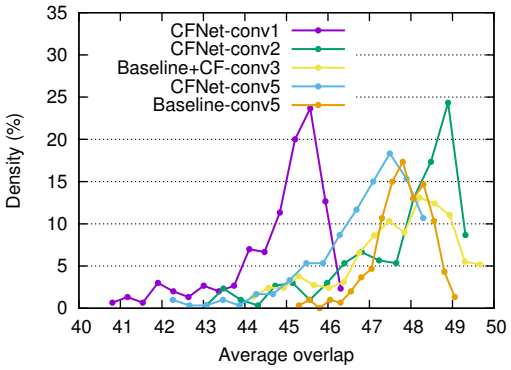

Figure 1: Empirical distribution of the average overlap for the hyperparameter search.

To achieve results that are competitive with the state-of-the-art, however, it is necessary to optimize the parameters of the tracking algorithm (on a held-out validation set).

To find optimal hyperparameters, we use random search with a uniform distribution on a reasonable range for each parameter. Specifically, we sample 300 random vectors of hyperparameters and run the evaluation described in Section 4.1 on the 129 videos of our validation set. Each method is then evaluated once on the test sets (OTB-2013, OTB-50 and OTB-100) using the hyperparameter vector which gave the best results on the validation set (specified in Table 1). We emphasize that, even though the ground-truth labels are available for the videos in the benchmarks, we do not choose hyperparameters to optimize the results on the benchmarks, as this would not give a meaningful estimate of the generalization ability of the method.

Note that this random search is performed after training and is only used to choose parameters for the online tracking algorithm. The same network is used for all random samples. The training epoch with the best tracking results on the validation set (with natural tracking parameters) is chosen.

Figure 1 shows, for each method, the empirical distribution of results (in terms of average overlap) that is induced by the distribution of tracking parameters in random search.

G. Detailed results on the OTB benchmarks

Figures 2 to 7 show the curves produced by the OTB toolkit for OTB-2013/50/100, of which we presented a summary in the main paper.

	avg. overlap	**best overlap**	scale step	scale penalty	scale l.r.	win. weight	template l.r.
CFNet-conv1	44.8	46.5	1.0355	0.9825	0.700	0.2375	0.0058
CFNet-conv2	**47.8**	49.5	1.0575	0.9780	0.520	0.2625	0.0050
Baseline+CF-conv3	47.7	**49.9**	1.0340	0.9820	0.660	0.2700	0.0080
CFNet-conv5	46.9	48.5	1.0310	0.9815	0.525	0.2000	0.0110
Baseline-conv5	**47.8**	49.2	1.0470	0.9825	0.680	0.1750	0.0102

Table 1: Average and best overlap scores over 300 random sets of hyperparameters. Values of hyperparameters associated to the best performance are also reported. These parameters describe: the geometric step to use in scale search, the multiplicative penalty to apply for changing scale, the learning rate for updating the scale, the weight of an additive cosine window that penalizes translation, and the learning rate for the template average.

References

[1] A. Griewank and A. Walther. *Evaluating derivatives: Principles and techniques of algorithmic differentiation*. SIAM, 2008. 2

[2] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista. High-speed tracking with kernelized correlation filters. *IEEE TPAMI*, 37(3):583–596, 2015. 2

[3] C. Ionescu, O. Vantzos, and C. Sminchisescu. Matrix backpropagation for deep networks with structured layers. In *ICCV 2015*, pages 2965–2973, 2015. 2

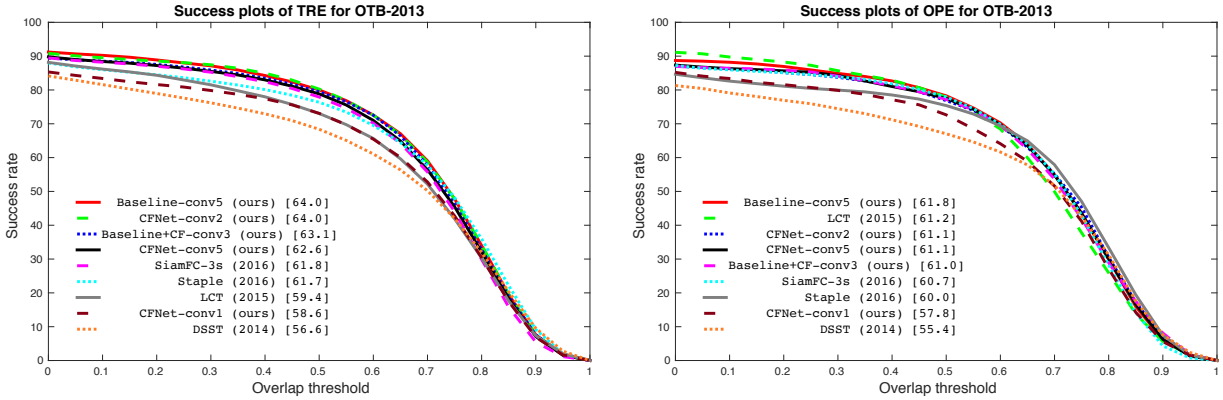

Figure 2: OTB-2013 success rate.

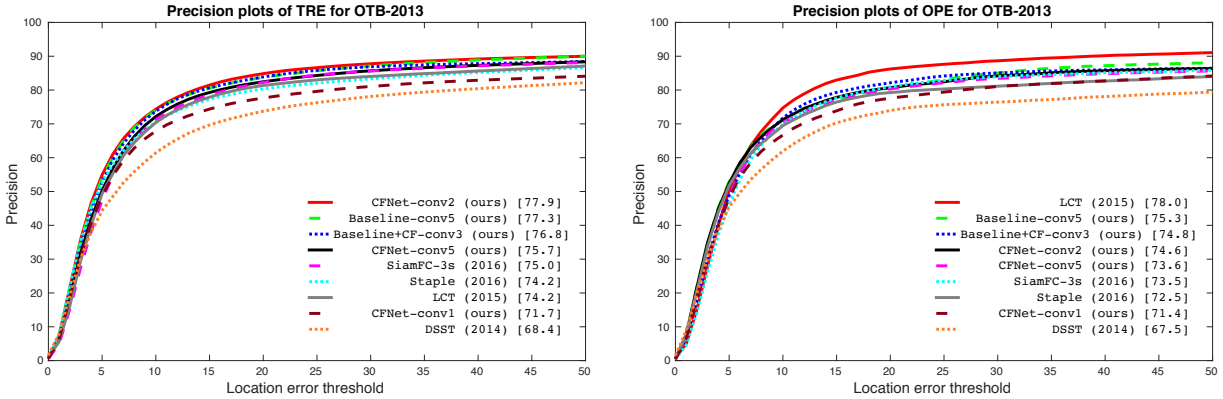

Figure 3: OTB-2013 precision.

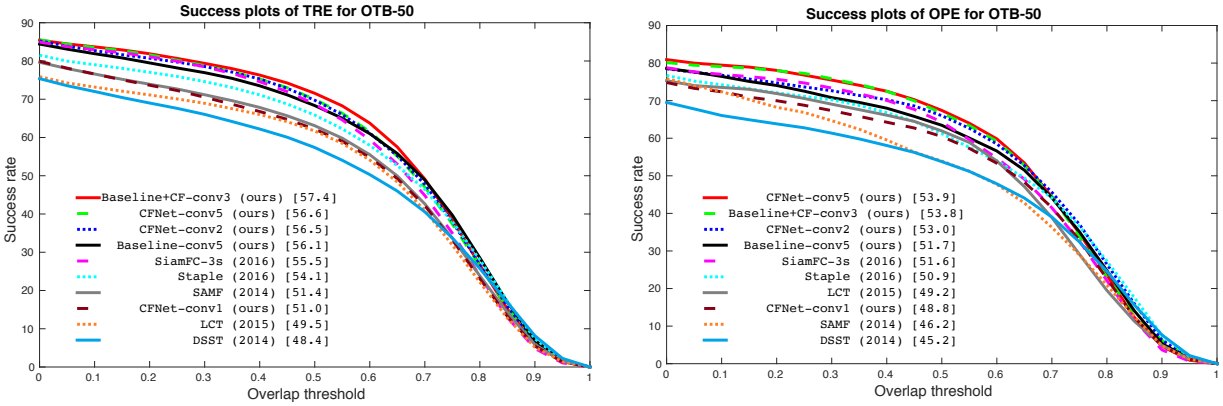

Figure 4: OTB-50 success rate.

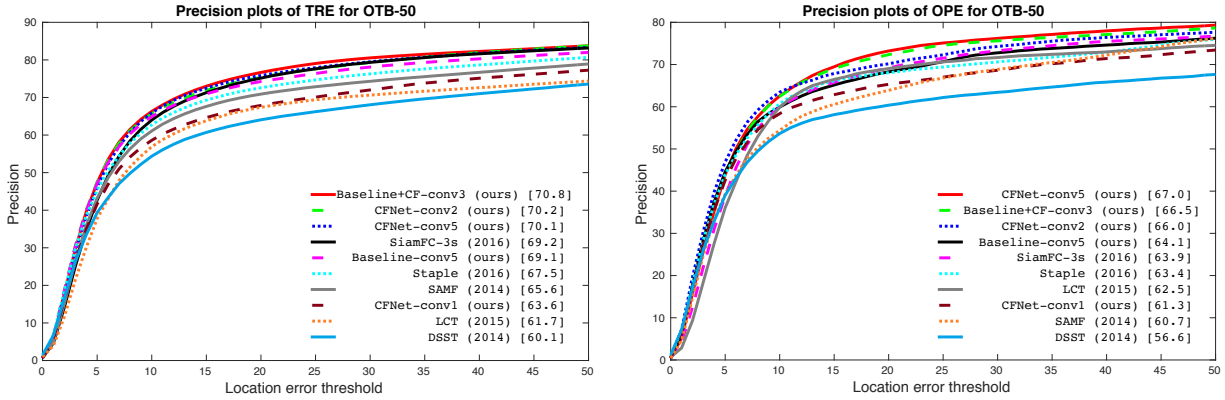

Figure 5: OTB-50 precision.

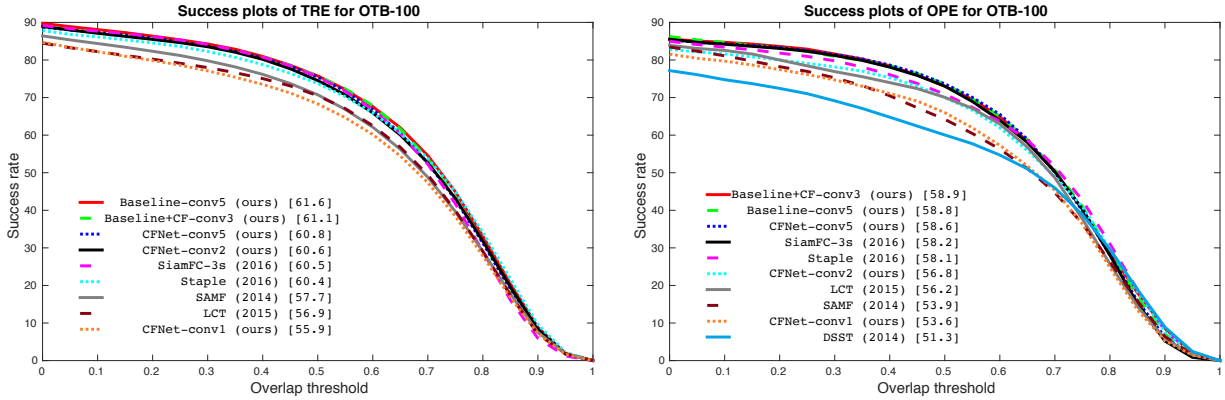

Figure 6: OTB-100 success rate.

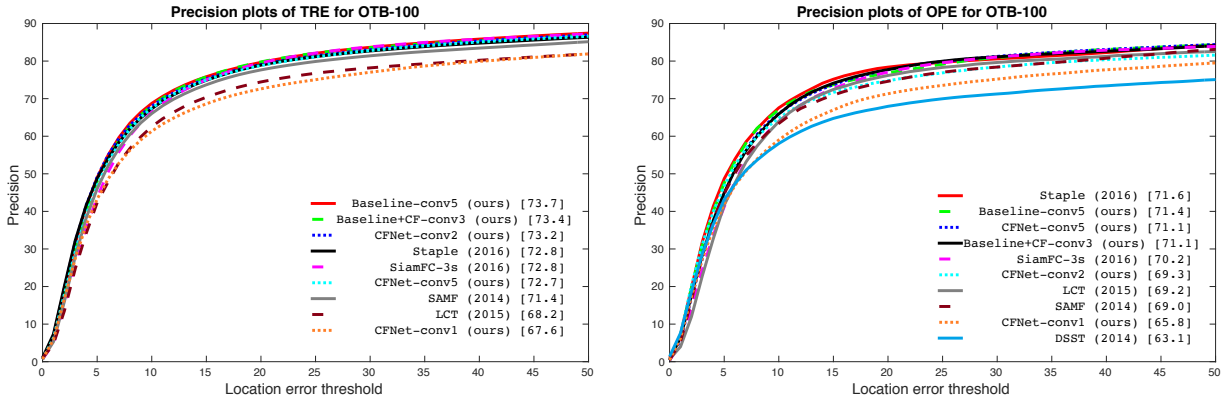

Figure 7: OTB-100 precision.