Journal of Intelligent & Robotic Systems (2021) 101: 11
https://doi.org/10.1007/s10846-020-01273-2

A Meta-Q-Learning Approach to Discriminative Correlation Filter based Visual Tracking

Akihiro Kubo[1] · Kourosh Meshgi[2] · Shin Ishii[1]

Received: 26 February 2020 / Accepted: 13 October 2020 / Published online: 11 December 2020

Abstract

Visual object tracking remains a challenging computer vision problem with numerous real-world applications. Discriminative correlation filter (DCF)-based methods are a recent state-of-the-art approach for dealing with this problem. The learning rate when applying a DCF is typically fixed, regardless of the situation. However, this rate is important for robust tracking, insofar as real-world video sequences include a variety of dynamical changes, such as occlusions, motion blur, and deformations. In this study, we propose Meta-Q-learning Correlation Filter (MQCF), a method for dynamically determining the learning rate of a baseline DCF-based tracker based on hand-crafted features of Histogram of Oriented Gradient (HOG), by means of reinforcement learning. The incorporation of reinforcement learning enables us to train a function for an image patch that outputs a situation-dependent learning rate of the baseline tracker in an autonomous fashion. We evaluated this method using two open benchmarks, namely, OTB-2015 and VOT-2105, and found our MQCF tracker outperformed a baseline state-of-the-art tracker by 1.8% in Area Under Curve on OTB-2015, and 8.4% relative gain in Expected Average Overlap in the VOT-2015 challenge. Our results demonstrate the advantages of the so-called meta-learning with DCF-based visual object tracking.

Keywords Visual tracking · Deep reinforcement learning · Meta-learning

1 Introduction

Visual object tracking is an important but challenging task in the field of computer vision. In a visual tracking task, we specify a target at a certain frame of a video and determine its location in the successive frames. This task is important owing to its wide range of real-world applications, such as visual surveillance, robot navigation, and sports analysis [1]. Visual tracking still includes difficulty, however, because such applications require online tracking in real-time, as well as high efficiency and performance.

A discriminative correlation filter (DCF) has been used in visual object tracking tasks [2] and has proven to have high computational efficiency and accuracy. A filter-based method like DCF obtains a heatmap that finds the target pattern by applying a convolution operation within the frequency domain, which is computationally efficient. In the operation a correlation filter is constructed to discriminate foreground or background patterns. To adapt to changes in the foreground and/or background, the correlation filter is updated in an incremental manner. The DCF has been extended for use with kernel tricks [3], adaptation to changes in the scale of the target [4–6], the handling of deformations and partial occlusions of the target [7, 8], a reduction in the level of confusion [9], and the application of boundary effects [10–12].

An incremental update of the correlation filter can decrease the robustness against changes in the target appearance, particularly when we consider natural video sequences (see Fig. 1). There is a trade-off between robustness and overfitting, which is controlled by tuning the learning rate for each dataset [2, 4, 11–13]. However, when the condition

✉ Akihiro Kubo
kubo-a@sys.i.kyoto-u.ac.jp

Kourosh Meshgi
kourosh.meshgi@riken.jp

Shin Ishii
ishii@i.kyoto-u.ac.jp

1 Department of Systems Science, Graduate School of Informatics, Kyoto University, Kyoto, Japan

2 The RIKEN Center for Advanced Intelligence Project, Tokyo, Japan

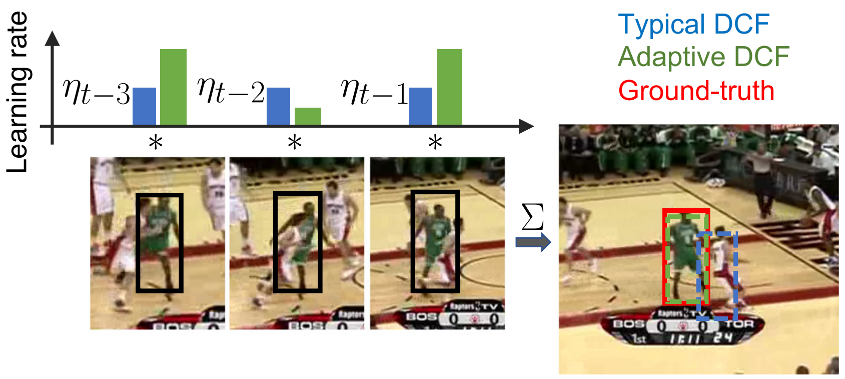

Fig. 1 DCF-based tracker's learning is self-contained. When a typical DCF-based tracker takes a new input patch at time $k = t - i$ $(i = 1, 2, 3)$, it assigns a prefixed learning rate η_{t-i}, shown with the blue bar. When the target has been occluded, however, this strategy can fail to track at time $k = t$ due to overfitting. By contrast, an adaptive learning rate (shown with the green bar) avoids such tracking failures. Here, each dashed bounding boxes is possible prediction

changes across frames in a video sequence, the optimal setting of the learning rates may differ among the frames.

In this study, we establish the advantages of adaptive learning rates for the sequential learning of a DCF-based tracker, especially which is based on the hand-crafted features of Histogram of Oriented Gradient (HOG) [14]. In effect, we propose a method to learn the learning, that is, a meta-learning algorithm which is performed offline by reinforcement learning (RL) [15]. We refer to this algorithm as meta-Q-learning because in the meta-learning we train a Q-function that determines the value of taking an action given a particular state, thus maximizing the total reward given in a sequence of actions and states. We adopted the RL framework to the visual object tracking problem by relating the state with the image patch and the action with the learning rate. Moreover, a reward is offered when a DCF-based tracker performs well. Based on the RL, we can then expect that the tracker will achieve good performance under similar dynamical situations in other video sequences. In addition, we used a deep neural network (DNN) to represent the Q-function of the RL agent. We compared our results with and without the use of a DNN, and the results demonstrate a significant improvement when a DNN is incorporated.

Our contributions provided in this study are (1) We show the benefits of a dynamical adaptation of the learning rate when training a DCF-based tracker on HOG features. (2) We propose an RL framework for training a controller that outputs an adaptive learning rate as a meta-learning of the DCF-based tracker, which is done outside of tracking tasks over test dataset. (3) We show the advantages of meta-Q-learning in combination with deep learning.

2 Related Studies

Because a tracker used in visual object tracking should be self-contained, too much adaptability to the new sample can be harmful, which is a phenomenon called overfitting. Such an adaptation, however, is necessary in certain cases of a sudden change, e.g., a change in illumination. Therefore, controlling the trade-off between adaptation and overfitting is one of the issues for achieving a more robust tracking.

One major approach to this issue is the regularization of model learning [2, 11, 12] in an incremental manner to reduce overfitting while saving memory. Such regularization has been shown to sufficiently alleviate overfitting to make the trackers more robust, even in an incremental manner, with greater memory efficiency.

Although the learning rate for a filter update is basically fixed, it can be controlled adaptively depending on the peak-to-sidelobe ratio (PSR) [7]. A higher PSR value denotes more confidence in the target detection. This idea of an adaptive learning rate is based on the intuition that the more the target changes, the more rapidly the model should adapt to the new sample even under the risk of overfitting.

Another way to mitigate the unwanted effects of overfitting is to learn at the lowest risk of such an occurrence to the samples applied [13]. Potentially corrupted samples are down-weighted while increasing the impact of correct samples. There is a relatively low risk of overfitting to the correct samples, and therefore these samples are stored in memory for use in tracking.

RL is a machine-learning framework that allows a system to conduct sequential decision-making, even without any knowledge of the system dynamics. Recently, the combination of a tracking algorithm and RL has been proposed, in which the incorporation of DNNs into the algorithm has also been discussed [16–21]. To achieve a better prediction of the target's location, in ADNets [17] actions were selected along the video sequence from a set of options, including translation moves, scale changes, and stopping, and the policy to select them was trained by RL. Choi et al. [19] proposed a visual tracking algorithm with a template selection policy trained by RL. The HP tracker [21] used RL for dynamically tuning five hyperparameters and demonstrated an improvement in terms of the tracking accuracy.

In this study, on the other hand, we propose the use of a meta-learned policy that solely controls the learning rate of a DCF-based tracker.

Unlike the method described in [13] used for controlling the unwanted effects of overfitting according to the samples in the current video sequence, our method reduces such effects by capturing patterns of changes in the target using a policy learned from the other video sequences.

3 Background

First, we introduce a DCF-based tracking framework on top of which we apply a learning rate controller. We then present the use of RL, which provides DCF-based trackers with such control.

3.1 Discriminative Correlation Filter

A discriminative model used in visual object tracking is a classifier that classifies a given image region or patch as either the target or background. The classifier consists of a discriminatively trained correlation filter f.

Provided that we have a set of training samples $\{x_k, y_k\}_{k=1}^t$ at time t. A label map $y_k \in \mathbb{R}^{M\times N}$ is the desired output of the convolution operation, which takes a peak positive value around the target region. In addition, x_k is a feature at time k consisting of $x_k^l \in \mathbb{R}^{M\times N}, l = 1, \cdots, L$ extracted from an image patch, where L denotes the number of channels. In this work, we use HOG features as x. By defining a correlation filter of channel l as a matrix $f^l \in \mathbb{R}^{M\times N}$, we derive an output that integrates the L filters as follows:

$$S(x; f) = \sum_{l=1}^{L} x^l * f^l, \tag{1}$$

where $*$ denotes a cyclic convolution operation. The filter f is optimized according to the regularized least squares method, whose error is the L_2 error between the filter heatmap matrix $S(x_k; f)$ and the desired output y:

$$\epsilon_t(f) = \sum_{k=1}^{t} \alpha_k ||S(x_k; f) - y_k||^2 + \lambda \sum_{l=1}^{L} ||f^l||^2, \tag{2}$$

where the weight $\alpha_k \geq 0$ adjusts the effect of each training sample, and $\lambda \geq 0$ is the strength of the ridge regularization.

According to the DCF-based tracking, the filter f_t predicts the next target's location using the index at which the filter heatmap matrix is maximal.

3.2 Reinforcement Learning

Reinforcement learning constitutes a set of machine learning techniques concerned with learning what to do—how to map situations to actions—so as to maximize cumulative reward [15]. The learner is not told which actions to take, but instead must discover which actions would be most rewarded by trying them. According to the formulation of RL, an agent takes an action $a \in \mathcal{A}$ in a state $s \in \mathcal{S}$, moves to a new state $s' \in \mathcal{S}$, and simultaneously receives a reward $r(s, a)$ from the environment. Here, a (possibly probabilistic) mapping from state s to action a to be taken in that state is called a policy, which is denoted as $\pi(a|s)$. In this study, we assume that the environmental dynamics and reward function are deterministic. That is, given action a_t and state s_t, the reward r_t and the next state s_{t+1} are uniquely determined. The goal of RL is to find a policy π that maximizes the expectation of cumulative rewards, or the Q-function:

$$Q^\pi(s, a) = E\left[\sum_{i=0}^{\infty} \gamma^i r(s_i, a_i) \middle| s_0 = s, a_0 = a\right], \tag{3}$$

where $0 < \gamma < 1$ is a discount factor emphasizing recent rewards. A policy is said optimal, when it is associated with the optimal Q-function: $Q^*(s, a) = r(s, a) + \gamma \max_{a'} Q^*(s', a')$. That is, $\pi^*(s) = \arg\max_a Q^*(s, a)$ for any state $s \in S$. When we sample an action a during RL, we do not use the optimal policy induced from the optimal Q-function, but use the following behavior policy:

$$\pi_\epsilon(a|s) = \frac{\exp(\epsilon Q(s, a))}{\sum_{a'} \exp(\epsilon Q(s, a'))} \tag{4}$$

where ϵ is a parameter that incorporates an exploration and exploitation into a policy by setting ϵ close to 0 and a relatively large value, respectively.

4 Proposed Tracking Framework

In this section, we first argue that in a DCF-based tracking it is desirable for its learning rate to be dynamically controlled according to the pattern of changes in the target image patches. A policy is therefore required to control this learning rate. In this study, we thus propose the use of RL to identify a policy for control of the situation-dependent learning rate for DCF-based trackers. In addition, we combine the RL using a data-driven feature extraction technique with deep learning.

4.1 How to Weight Samples in a DCF

In many DCF-based trackers, α_i in Eq. 2, which indicates the weight of each sample in the objective function of learning, is incrementally set based on a fixed strategy [2–4, 6, 11]; that is, the more recent the observation is, the more it is emphasized in learning. In an online learning setting, we solve the ridge regression at time t as follows:

$$\hat{\epsilon}_t(f) = \|S(x_t;\, f) - y_t\|^2 + \lambda \sum_{l=1}^{d} \|f^l\|^2. \tag{5}$$

Therefore, $\epsilon_t(f) = \sum_{k=1}^{t} \alpha_k \hat{\epsilon}_k(f)$. As time step proceeds we incrementally apply an interpolation to the filter parameters $\dot{f}_t$ as follows:

$$\dot{f}_t = (1-\eta)\dot{f}_{t-1} + \eta f_t^*, \tag{6}$$

where $f_t^* := \arg\max_f \hat{\epsilon}_t(f)$ is a local model in terms of time; $\dot{f}_{t-1}$, the previous filter, is a global model at $t-1$; and η the fixed learning rate.

The learning rate is important for robust tracking because it controls the balance between newly observed features and past estimations, i.e., the local model and the global model. Thus, if the target changes its appearance rapidly, the tracker should focus on the more recent features of the target, whereas if the target changes only slightly, the tracker can retain the information with the global model. We explore this in greater detail below.

4.2 Necessity of Adaptation

Figure 2 shows snapshots of tracking with a baseline DCF-based tracker (in particular, we used BACF, whose tracking is shown as the blue bounding box, where AUC, a tracking performance indicator described in Section 5.4, was 0.485), with a plot for a typical profile of the learning rates over time.

Here, we describe the result of a simple experiment. For each frame of the video clip, we manually tuned the learning rate parameter that was used for the model update in Eq. 6; this manual tuning was based on the intuition discussed above. The plot under the snapshots shows a series of these manually tuned learning rates, with which the tracker achieved a much better performance (red bounding box in the snapshots, with AUC = 0.568). This observation encourages the idea that the adaptive tuning of η in Eq. 6 will improve the performance of the tracker.

4.3 Meta-Q-Correlation Filter

The observation above suggests that a controller, or a policy, is required to control the learning rate in a situation-dependent manner.

This requirement is difficult to be attained, because there is no systematic method to annotate an ideal learning rate to each state, due to the dynamic nature of general visual object tracking problems. The concept of RL is viable, however, because it has the potential to solve a credit assignment problem in dynamic and even unknown environments.

To incorporate RL into the tracking algorithms, we consider a policy $\pi(\eta|s)$ which outputs the learning rate η used for sequential learning in Eq. 6, according to an observed state s.

We assume that up to time $t = k$, we have a series of observed image patches $x_{k-i}, \cdots, x_k$, which are the tracking results. Note that x_{k-j} is dependent on $\dot{f}_{k-j-1}$. We obtain a state at time k : $s_k = g(x_{k-i}, \cdots, x_k)$, where $g(\cdot)$ is an arbitrary function, e.g., the Euclidian distance between x_k and x_{k-1} with $i = 1$, or a representation obtained through a data-driven feature extraction technique such as deep learning.

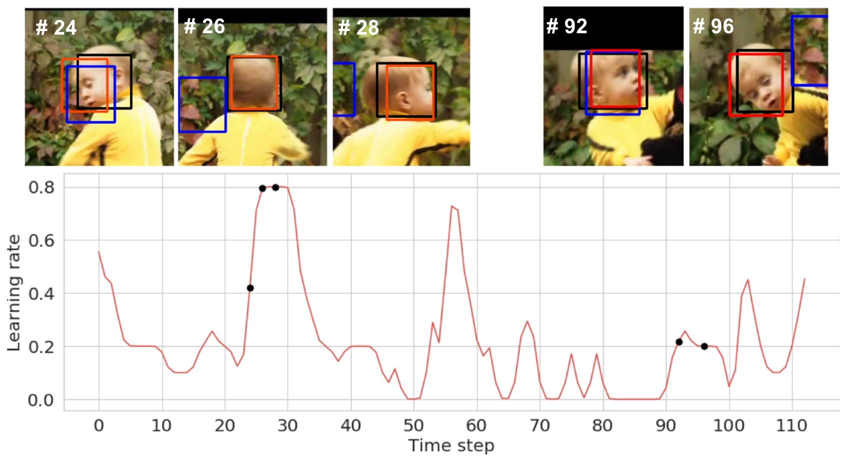

Fig. 2 Snapshots of tracking results (top) and a plot for a typical profile of the learning rate η over time (bottom). The tracking results are those using BACF with a fixed parameter (blue) and manually set parameter time-series (red). Here, black indicates a human-annotated bounding box, and is regarded as ground-truth. The bottom plot shows changes in the learning rate, which was annotated manually. With the rapid changes of the z-rotation of the target, shown in snapshots, higher learning rates were required near the time indicated by the black dots

Given a reward function and an environment with an embedded tracker, we can learn a policy of the learning rate by RL. The definition of a reward can be an overlap between the bounding box that has been output by the learner tracker and the ground-truth. Please refer to Section 5.1 for the definitions of a state and reward function.

For the RL algorithm we define the Q-function, and thus we call a DCF-based visual tracking algorithm by meta-Q-learning, Meta-Q-Correlation Filter (MQCF). Since we expect the generalization ability of the RL-based meta-Q-learning, we applied the meta-Q-learning in an offline manner, as outside of the target dataset; that is, we did not use the videos to be tested when meta-Q-learning the baseline correlation filter tracker. To investigate the applicability, we first introduce a simple implementation based on the SARSA algorithm [22]. Then, in the following section, we extend the algorithm to equip the policy with larger generalizability, with the help of deep neural networks (DNNs). The overall tracking and meta-Q-learning schemes are shown in Fig. 3.

4.4 Adaptive update through Deep Q-learning

Next, we discuss how to represent a state that becomes an input for the policy. One simple way to represent a state, as used with the MQCF (in Section 5.1.2), is to use the Euclidean distance between two consecutive image patches, reflecting the idea that the policy is dependent on changes in the target's appearance.

In this study, as an extension of the MQCF above, we propose the use of a DNN to capture the patterns of appearance changes in a data-driven fashion, and such our proposal is called Deep MQCF (DMQCF). Among the family of methods that combine DNNs and RL, the Deep Q-network (DQN) [23] algorithm provides a suitable architecture for our purpose owing to its inclusion of a convolutional neural network (CNN) [24]; CNNs are known to be robust to image blur, rotations, shifts, and other factors.

Our DQN takes a sequence of i image patches as an input, and outputs a d-dimensional vector $\mathbf{a} \in R^d$. The output $\mathbf{a}$ is a vector of Q-values, each of which represents the goodness of the corresponding discretized learning rate.

The pseudo-code is shown in Algorithm 1. Here, for the purpose of illustration, annotation y and the tracker's estimation x have ambiguities in the image patch matrix or its location. The function *local_opt* returns a solution to the minimization problem in Eq. 5), *detect* outputs the location where the heatmap in Eq. 1 is maximal, *sample_action* obtains a learning rate by sampling an action along the distribution in Eq. 4, *global_opt* applies a new parameter by updating Eq. 6, *get_state* applies a new state by cutting off the oldest patch and adding the newest patch, and *get_reward* returns a reward. Lines 4 through 7 are the initialization of the RL environment. After that an execution of action from the agent and an observation of a new state follow each other.

5 Evaluation

We first compared our MQCF and DMQCF to a baseline DCF method, in particular, BACF [11]. For this baseline comparison we used a benchmark dataset OTB-2015 [25] and divided it into two datasets of OTB50 and OTB100. OTB50 is a set of such videos that share with OTB-2013 [26] and OTB100 is a set of such videos that are not shared with OTB-2013. Second, we compared DMQCF with the other state-of-the-art trackers on OTB-2015 and another dataset, VOT-2015 [27].

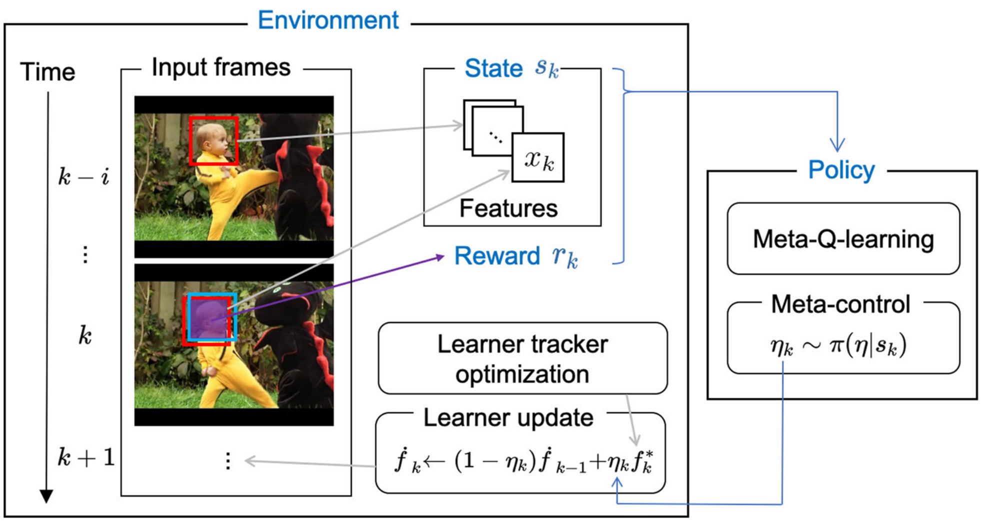

Fig. 3 Schematic of our proposal. At time $t = k$, we make a state s_k consisting of image patches $\{x_{k-i}, \cdots, x_k\}$. The red bounding box denotes the region from which these features are extracted and blue bounding box is the ground-truth. The policy π outputs a learning rate η_k as an action. Blue words correspond to the components of RL

Algorithm 1 DMQCF tracker.

```
Input:
 1: $y_1, I_1, ..., I_T$ and $y_2, ..., y_T$ (when meta-Q-learning)
Output:
 2: $x_1, \cdots, x_T$
 3: procedure QCF($\mathbf{y}, \mathbf{I}$)
 4:     $x_1 \leftarrow y_1$
 5:     $\dot{f}_1 \leftarrow local\_opt(x_1)$                ▷ Eq. 5
 6:     $x_2 \leftarrow detect(\dot{f}_1, I_2)$               ▷ Eq. 1
 7:     $s_1 \leftarrow [0, x_1, x_2]$
 8:     for $i \leftarrow 1$ to $T$ do
 9:         $\eta_i \leftarrow sample\_action(s_i)$           ▷ Eq. 4
10:         $s_{i+1} \leftarrow step(\eta_i)$

 1: function $step(\eta_i)$
 2:     $f_i^* \leftarrow local\_opt(x_i)$
 3:     $\dot{f}_{i+1} \leftarrow global\_opt(\eta_i, \dot{f}_i, f_i^*)$   ▷ Eq. 6
 4:     $x_{i+1} \leftarrow detect(\dot{f}_{i+1}, I_{i+1})$
 5:     $s_{i+1} \leftarrow get\_state(s_i, x_{i+1})$
 6:     if meta-Q-learning then
 7:         $r_i \leftarrow get\_reward(y_i)$                 ▷ Eq. 7
 8:         Store an experience tuple $(s_i, \eta_i, r_i, s_{i+1})$
 9:         Update Q-function
    return $s_{i+1}$
```

5.1 Definitions of the RL Environment

To apply the RL framework, we define the state and reward and clarify the environment used in our case.

5.1.1 Reward function

We assume that the tracker receives a reward r_k from the environment at each time k during meta-learning based on annotated video sequences.

As the immediate reward r_k, we define the intersection over union (IoU):

$$r_k = \frac{|Y_k \cap G_k|}{|Y_k \cup G_k|}, \tag{7}$$

where Y_k is the set of pixels included in the estimated bounding box; G_k is the pixel set belonging to the ground-truth bounding box; and $|\cdot|$ denotes the number of pixels. Note that, during the meta-learning stage, the tracker was trained based on annotated video sequences, and as such it can access the ground-truth bounding box in each image frame. By definition, the reward in Eq. 7 measures the closeness between the estimated target region Y_k and the ground-truth target region G_k.

5.1.2 State

Because the learning rate for the filter update should reflect changes in the target's appearance, we first employed a simple definition of a state used for tabular representation of a Q-function inside the meta-Q-learning.

At time k, based on two consecutive image patches, x_k and x_{k-1}, we calculate the Euclidian distance between the two patches, $||x_k - x_{k-1}||$, and divide it by MN, where M and N denote the height and width of the extracted patch, respectively, i.e., $s_k = ||x_k - x_{k-1}||/MN$ and $s_1 = 0$. This is because the size of patch $M \times N$ is variable over the sequences and should thus be normalized.

As a second option for the definition of a state used for the Deep MQCF method, we used a stack of I image patches $x_{k-I}, \cdots, x_k$. In other words, such image patches are defined as a state $\mathbf{s}_k \in R^{I\times M\times N\times L}$, where I is the time length and L is the number of channels for each image patch.

5.1.3 Tracker inside Environment

We regard the DCF-based tracking as a part of the environment for RL, which takes the learning rate as an input and then outputs a state and a reward defined above. In this study, as the baseline tracker, we employed the BACF [11], which has demonstrated a state-of-the-art tracking accuracy on several benchmark datasets.

5.2 Training and Evaluation

Meta-learning using RL is performed only offline using multiple annotated videos, whereas tracking coupled with learning the target appearance is performed online for each video sequence. Please note that the word "annotation" here means attaching a bounding box for a single target. We do not need to label a ground-truth learning rate for each image frame.

During the meta-Q-learning stage, we avoided situations in which the tracker apparently failed, because training based on such poorly rewarded situations will hinder the reward-based learning. Thus, we stopped the RL process when the tracker failed to track the target. We regarded the tracker as having failed when it had a moving average AUC of below 0.2 over the last 100 image frames. To avoid a sampling bias from the earliest frames for each training videos, a randomly selected frame was used as an initial image frame and the procedure started from the time. Moreover, to introduce robustness to DQN, we added pixel-wise independent Gaussian noise to each of the image patches during the meta-Q-learning stage.

5.3 Implementation Details

The DNN architecture used in this study is basically the same as the standard architecture proposed as DQN [23], where a CNN and a fully connected network are connected into a sequential (i.e., hierarchical) architecture. In our particular implementation, we used three consecutive patches, namely, the current, previous, and two-steps before, as a state to be input into the DQN. The DQN output was a vector whose element each corresponds to the Q-value for a discretized learning rate; the rate can take any of 20 discretized values, which were arranged at a log-linear scale of between 0 to 1. Regardless of the size of the image patch cut from each frame, the image patch was resized to 200 pixels $\times$ 200 pixels, and then multiplied with a Hann window, as described in [2].

The discount factor γ for the RL was set to 0.99. The learning rate for the stochastic gradient optimization was set to 2.5×10^{-4}. We utilized the RMSProp optimizer with a momentum of 0.95. For the training of the DQN, experience replay buffer and fixed target Q-network techniques were incorporated [23]. The parameters were synchronized with that of the target Q-network during a 1,000 step period, and the experience tuples were randomly sampled. The mini-batch size for DQN training was 32, and the hyperparameter ϵ in Eq. 4 was decayed exponentially, starting at 1.0 and eventually approaching 0.1. The number of learning episodes was 5,000, where an episode denotes DQN training with a single annotated video.

Other parameters for the tracker were the same as those in BACF. The image features used by BACF were 31-channel HOGs, in line with the original study [14]. We used the default settings for these hyperparameters, and did not tune them.

5.4 Quantitative Results

First, we evaluated our trackers on two datasets: OTB50 and OTB100, in a severe manner, i.e., leave-one-dataset-out cross-validation. Each dataset contains 50 sequences respectively. That is, the meta-learning was performed on OTB50 and tested on OTB100, and vice versa. Second, we compared our DMQCF tracker with the other state-of-the-art trackers on OTB-2015 and VOT-2015. Over all of these evaluations we used the common architecture for the DMQCF, including the size of an image patch for the input, regardless of the used database.

5.4.1 Meta-Q-Correlation Filter

Table 1 shows the results, average performance for the two datasets, in terms of the area-under-the-curve (AUC) [26]. The AUC measures an area under the success plot curve showing the percentage of bounding boxes whose IoU scores are larger than a given threshold. Therefore, the larger the AUC is, the better the tracking performance.

Although the MQCF utilized the simplest RL based on SARSA, incorporating the naive state representation of a difference between two consecutive image patches, our meta-trained tracker showed a better performance for OTB100 than the baseline. However, this improvement was not as significant as expected. One reason for this moderate improvement might be the state representation. Because we relied on RL, the trained policy should output an appropriate learning rate for the filter update, in which the target situations in the test videos are similar to those in the training videos. This similarity is solely attributed to the features extracted from the image patches. If these features cannot extract the similarity between the training videos and the test videos, the entire RL scenario will collapse. This observation motivated us to utilize more generalizable features, rather than patch difference-based features, to allow RL to be more feasible when meta-training the DCF-based trackers; thus, we proposed DMQCF.

5.4.2 Deep Q-Correlation Filter

In this section, we first show a comparison of DMQCF in the same experimental setting as in Table 1 except for the test dataset and the definition of the state, as explained above. Figure 4 shows a comparison on OTB-2015 between our DMQCF and five state-of-the-art DCF-based trackers, namely, DSST [4], SRDCF [12], KCF [3], STAPLE [28], and BACF [11]. Moreover, as an important related method [13], we included BACFdecon in this comparison, which jointly optimized the sample weights and the model filter

Table 1 Average AUCs for the DMQCF and BACF on two benchmark datasets

	OTB50	OTB100
baseline (BACF [11])	69.7	56.1
MQCF	67.3	58.0
DMQCF	71.5 ± 1.19	58.0 ± 2.10

Owing to the stochastic nature of our proposed trackers, MQCF and DMQCF, we conducted ten individual runs with different random seeds. The table shows the mean and standard deviation of the DMQCF performance for all videos over the ten runs. The standard deviation over the ten runs for the MQCF was omitted in the table because it was small (below 0.01) for both datasets

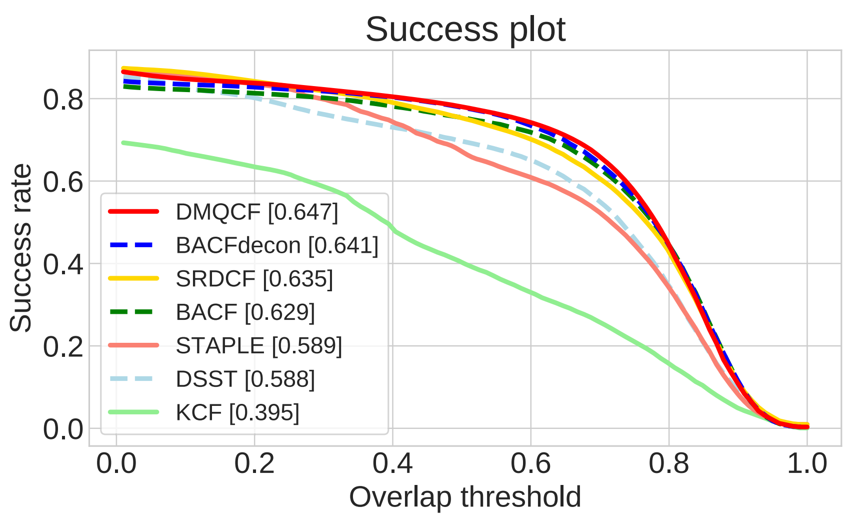

Fig. 4 Success plots comparing our DMQCF with five state-of-the-art DCF-based trackers and a BACF modified with a dynamical setting of the learning rate (i.e., BACFdecon), on OTB-2015. The AUCs are in brackets. Please note that BACF was the common baseline tracker of DMQCF and BACFdecon

of the BACF tracker. Although the original study [13] used the SRDCF, instead of BACF, as the baseline DCF-based tracker, we implemented BACFdecon for making the comparison with our RL-based meta-learning (which also used BACF as the baseline) fair. Note that our RL-based method and the adaptive sample weighting scheme, called decon in [13], are basically modifications of DCF-based trackers. We here report the results of the DMQCF, BACF, and BACFdecon, which were implemented by ourselves, while the other trackers' results are publicly available ones. When meta-Q-training DMQCF, we used a subset of video sequences registered in Temple-Color [29]; this subset did not include the video sequences that are shared with OTB-2015.

In this configuration, our DMQCF tracker outperformed all of these state-of-the-art DCF-based trackers including BACFdecon.

Moreover, we conducted experiments on VOT-2015 [27] and observed that our DMQCF outperforms BACF in terms of accuracy, robustness, and the expected average overlap (EAO); the accuracy of DMQCF was 0.546, with robustness of 41.2 and EAO 0.194, whereas values of 0.547, 53.7, and 0.179 were obtained by BACF, respectively. Here, the EAO measure is based on an empirically estimated average overlap (as a function of the sequence length) and the typical-sequence-length distribution. The larger the EAO is, the better the tracking performance.

These results demonstrate the effectiveness of meta-Q-learning for an adaptive learning rate of a DCF-based tracker, and the achieved robustness is generalizable to a wide range of video sequences registered in these benchmark datasets.

5.5 Qualitative Results

To demonstrate that the DMQCF could acquire a reasonable policy for controlling the learning rate, we examined the similarity of the learning rates output from DMQCF to those attached by human annotator (Section 4.2). Figure 5 shows typical results; the adaptive learning rate attained by DMQCF was effective especially when the target appearance was drastically altered. For example, from frames #24 to #28, the target exhibited rapid changes in the z-rotation, during which the baseline failed to track the target whereas our DMQCF could track the target by increasing the learning rate for those frames. This tendency was consistent with human strategy, although our DMQCF seemed more sensitive to the dynamics in the video frames than human annotations. The correlation between the learning rate adaptively predicted by DMQCF and that attached by a human annotator over all time steps was 0.357, which statistically rejects the null hypothesis of no correlation ($p < 0.0001$). Although this value may seem insufficient, it would have the case that human annotations were not sufficiently good; indeed, DMQCF's AUC score for this sequence was 0.616, which is higher than that of the manual adjustment (0.568) and that of a baseline tracker (0.485).

These results suggest that our DMQCF was able to reproduce human annotation of the adjustable learning rate, and moreover, to attach better annotations than by the human annotator.

5.6 Attribute Based Evaluation

Each video sequence in the OTB-2015 dataset is provided 11 labels to describe different challenges in the tracking problem, such as variations in illumination, out-of-plane rotation, and fast motion.

Table 2 lists an attribute-dependent comparison between our DMQCF and other DCF-based trackers on the OTB-2015 dataset. Like in Section 5.4.2, our DMQCF did not use the video sequences in OTB-2015 for its meta-Q-learning.

As a result, our DMQCF improved the tracking performance over the baseline BACF for all attributes, except

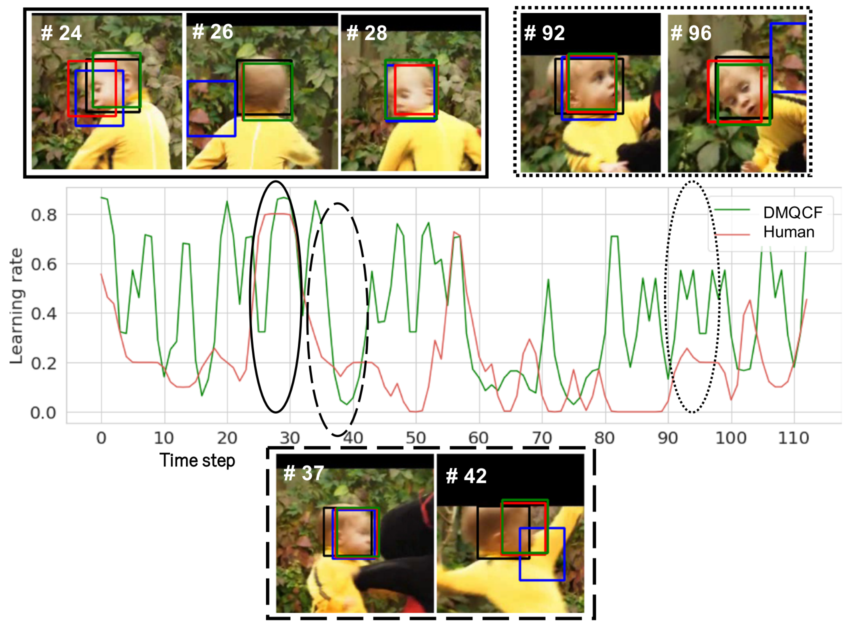

Fig. 5 Snapshots of tracking results (top and bottom) and a plot of a typical learning rate profile (middle). The green plots and bounding boxes correspond to our DMQCF, red indicates a human annotation described in Section 4.2, and blue bounding boxes are by the baseline tracker with the fixed learning rate. The black bounding boxes are the ground-truth for the target

for the in-plane rotation, out-of-view and low resolution attributes. For some attributes such as occlusions, deformations, and motion blur, our tracker showed a substantial improvement because the target changes are not negligible in such cases.

These analyses demonstrate empirically that RL-based meta-learning for a filter learning renders DCF-based tracking more robust under such difficult situations that include unpredictable changes in the target.

6 Discussion

In this study, we restricted the baseline DCF-based trackers as those utilizing HOG features as their inputs. For evaluations, in particular, we chose the BACF tracker as the baseline DCF-based tracker. As another option, we applied DMQCF to the KCF tracker [3] but could not obtain good generalizability of the meta-policy over a variety of video sequences. The reason is as follows; Both of BACF and KCF use HOG features x. The BACF tracker constructs a ridge regressor to discriminate foreground and background, which is linear with respect to both of x and its filter f in the Fourier domain. On the other hand, the KCF tracker constructs a kernel ridge regressor (in the nonlinear space transformed from the HOG feature space) and optimizes the filter α in the dual space with the kernel trick. The optimization is linear in the dual space, and thus the KCF's regressor is linear with respect to the nonlinear transformation of HOG features x, while the BACF's regressor is linear with respect to x. So there is not much difference between these two trackers' optimization.

When combined with our MQCF, however, there is a big difference. The current meta-Q-learning is performed by

Table 2 Comparison of AUC scores (%) for each attribute of illumination variation (IV), scale variation (SV), occlusion (OCC), deformation (DEF), motion blur (MB), fast motion (FM), in-plane rotation (IPR), out-of-plane rotation (OPR), out-of-view (OV), background clutter (BC), and low resolution (LR) on OTB-2015

	IV	SV	OCC	DEF	MB	FM	IPR	OPR	OV	BC	LR
KCF	39	34	36	35	30	37	42	38	32	42	29
DSST	66	57	54	48	52	56	59	55	49	57	52
STAPLE	63	55	57	52	56	57	61	57	53	55	49
BACF	**72**	60	55	51	64	63	**62**	58	56	63	61
SRDCF	71	62	57	49	62	65	**62**	58	55	**65**	60
BACFdecon	71	**63**	57	54	**69**	64	**62**	59	**57**	64	**62**
DMQCF	**72**	**63**	**59**	**57**	**69**	**66**	61	**61**	55	**65**	61

Bold font implies the largest value between the trackers for each attributs taking an image patch sequence as its input, when applied to both of the BACF and KCF trackers. That means there is no explicit way for the meta-Q-learner to know what kind of nonlinear transformation was used in the KCF tracker, while HOG or similar features themselves could be extracted by its CNN encoder in a data-driven fashion. To clarify this difference in the applicability of our meta-Q-learning between BACF and KCF, one possible way is to directly implement nonlinear transformation underlying the kernel function used in the KCF tracker in our meta-Q-learning. We left the clarification for future work.

Another point we want to discuss is a configuration of the reward function. In Section 5, we demonstrated that our RL-based meta-learning improves the baseline tracker, when immediate rewards were defined based on ground-truth annotations of the target location and given for every frame of the videos used for the meta-Q-learning. Note that our meta-Q-learned policy could capture the general strategy for dynamically adjusting the online learning rate for the DCF-based trackers. Indeed, the good tracking performance obtained by our very severe meta-Q-learning method, leave-one-dataset-out cross-validation, provides the evidence. We consider our method to be applicable to sequences even without annotations if the reward can be well defined. For example, the mimicking of some majority-of-vote results by multiple existing trackers can be a reward.

By contrast, the peak-to-sidelobe ratio (PSR) in the DCF-based trackers can be an alternative because a higher PSR value denotes more confidence in the target detection, and high confidence correlates with better tracking. In these cases, we do not require any ground-truth annotations at all. Moreover, our method may be able to provide a good performance improvement when some portion of an extremely large database is annotated, including a weak or semi-supervised annotation [30]. These extensions of our meta-Q-learning idea remain for future studies.

7 Conclusion

In this work we proposed a meta-Q-learning process applied to the filters in a variety of DCF-based visual object trackers, to obtain a policy that controls the online learning rate for the sequential update of the filter. We also presented Deep Meta-Q-learning (called DMQCF) to better acquire image features in a data-driven manner. We evaluated DMQCF on several open benchmark video datasets and observed that it outperforms state-of-the-art DCF-based trackers, including an alternative method that dynamically adjusts the learning rate. The results show the effectiveness of the use of dynamic control of learning parameters with DCF-based trackers, making them robust to various situations in a generalized manner.

We found that, on the other hand, DMQCF applied to the baseline KCF tracker could not obtain generalizability over different video sequences while DMQCF applied to the BACF tracker could. One of the remaining issues of the current study is to extend the applicability of our meta reinforcement learning scheme to nonlinear filter optimization problems underlying the kernel-based filter learning like in the KCF tracker. We remark that reinforcement learning is in principle applicable also to optimization of hyperparameters of the tracker, such as those of the kernels.

Acknowledgments The authors would like to thank the associate editor and anonymous reviewers for their time and constructive comments towards the improvement of our work. This research was partly supported by JSPS-KAKENHI (No. 17H06310 and 19H04180).

Declaration The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Poppe, R.: A survey on vision-based human action recognition. Image Vis. Comput. **28**(6), 976–990 (2010)
2. Bolme, D.S., Beveridge, J.R., Draper, B.A., Lui, Y.M.: Visual object tracking using adaptive correlation filters. In: Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on, pp. 2544–2550. IEEE (2010)
3. Henriques, J.F., Caseiro, R., Martins, P., Batista, J.: High-speed tracking with kernelized correlation filters. IEEE Trans. Pattern Anal. Mach. Intell. **37**(3), 583–596 (2015)
4. Danelljan, M., Häger, G., Khan, F., Felsberg, M.: Accurate scale estimation for robust visual tracking. In: British Machine Vision Conference. BMVA Press, Nottingham (2014)
5. Danelljan, M., Häger, G., Khan, F.S., Felsberg, M.: Discriminative scale space tracking. IEEE Trans. Pattern Anal. Mach. Intell. **39**(8), 1561–1575 (2017)
6. Li, Y., Zhu, J.: A scale adaptive kernel correlation filter tracker with feature integration. In: European Conference on Computer Vision, pp. 254–265. Springer (2014)
7. Liu, T., Wang, G., Yang, Q.: Real-time part-based visual tracking via adaptive correlation filters. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4902–4912 (2015)
8. Bibi, A., Mueller, M., Ghanem, B.: Target response adaptation for correlation filter tracking. In: European conference on computer vision, pp. 419–433. Springer (2016)
9. Possegger, H., Mauthner, T., Bischof, H.: In defense of color-based model-free tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2113–2120 (2015)
10. Galoogahi, H.K., Sim, T., Lucey, S.: Correlation filters with limited boundaries. In: Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on, pp. 4630–4638. IEEE (2015)
11. Kiani Galoogahi, H., Fagg, A., Lucey, S.: Learning background-aware correlation filters for visual tracking. In: Proceedings of the IEEE international conference on computer vision, pp. 1135–1143 (2017)
12. Danelljan, M., Hager, G., Shahbaz Khan, F., Felsberg, M.: Learning spatially regularized correlation filters for visual

tracking. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 4310–4318 (2015)
13. Danelljan, M., Hager, G., Shahbaz Khan, F., Felsberg, M.: Adaptive decontamination of the training set: A unified formulation for discriminative visual tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1430–1438 (2016)
14. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: 2005. CVPR 2005. IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 1, pp. 886–893. IEEE (2005)
15. Sutton, R.S., Barto, A.G., et al.: Introduction to reinforcement learning, vol. 135. MIT press Cambridge (1998)
16. Zhang, D., Maei, H., Wang, X., Wang, Y.-F.: Deep reinforcement learning for visual object tracking in videos. arXiv:1701.08936 (2017)
17. Yoo, S.Y.J.C.Y., Yun, K., Choi, J.Y., Yun, K., Choi, J.Y.: Action-decision networks for visual tracking with deep reinforcement learning. CVPR (2017)
18. Supancic III, J.S., Ramanan, D.: Tracking as online decision-making: Learning a policy from streaming videos with reinforcement learning. In: ICCV, pp. 322–331 (2017)
19. Choi, J., Kwon, J., Lee, K.M.: Visual tracking by reinforced decision making. arXiv:1702.06291 (2017)
20. Huang, C., Lucey, S., Ramanan, D.: Learning policies for adaptive tracking with deep feature cascades. In: IEEE Int. Conf. on Computer Vision (ICCV), pp. 105–114 (2017)
21. Dong, X., Shen, J., Wang, W., Liu, Y., Shao, L., Porikli, F.: Hyperparameter optimization for tracking with continuous deep q-learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 518–527 (2018)
22. Rummery, G.A., Niranjan, M.: On-line q-learning using connectionist systems, vol. 37. University of Cambridge, Department of Engineering (1994)
23. Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G., et al.: Human-level control through deep reinforcement learning. Nature **518**(7540), 529–533 (2015)
24. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proc. IEEE **86**(11), 2278–2324 (1998)
25. Wu, Y., Lim, J., Yang, M.-H.: Object tracking benchmark. IEEE Trans. Pattern Anal. Mach. Intell. **37**(9), 1834–1848 (2015)
26. Wu, Y., Lim, J., Yang, M.-H.: Online object tracking: A benchmark. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 2411–2418 (2013)
27. Kristan, M., Matas, J., Leonardis, A., Felsberg, M., Cehovin, L., Fernandez, G., Vojir, T., Hager, G., Nebehay, G., Pflugfelder, R.: The visual object tracking vot2015 challenge results. In: Proceedings of the IEEE international conference on computer vision workshops, pp. 1–23 (2015)
28. Bertinetto, L., Valmadre, J., Golodetz, S., Miksik, O., Torr, P.H.: Staple: Complementary learners for real-time tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1401–1409 (2016)
29. Liang, P., Blasch, E., Ling, H.: Encoding color information for visual tracking: Algorithms and benchmark. IEEE Trans. Image Process. **24**(12), 5630–5644 (2015)
30. Real, E., Shlens, J., Mazzocchi, S., Pan, X., Vanhoucke, V.: Youtube-boundingboxes: A large high-precision human-annotated data set for object detection in video. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5296–5305 (2017)

Akihiro Kubo received the B.E. from the Undergraduate School of Electrical and Electronic Engineering, Faculty of Engineering, Kyoto University in 2016 and the M.E. from the Graduate School of Informatics, Kyoto University in 2019. He is currently a Ph.D. student at the Graduate School of Informatics, Kyoto University. His research interests include reinforcement learning and optimal control, machine learning and computer vision.

Kourosh Meshgi received his Ph.D. in Informatics in 2015 from Kyoto University. He is currently a research scientist at RIKEN AIP and a guest researcher at Kyoto University. His research interests include machine learning, computer vision, and natural language processing. Dr. Meshgi's current research focuses on visual tracking and interpreting/explaining deep learning models, as well as active, reinforcement, multitask, and ensemble learning.

Shin Ishii received his B.E., M.E. and Ph.D. from University of Tokyo in 1986, 1988 and 2001, respectively. He was affiliated with the R& D Center, Ricoh Co. Ltd., ATR Human Information Processing Research Laboratories, and Nara Institute of Science and Technology, before becoming a professor at the Graduate School of Informatics, Kyoto University, in 2007. He has also served as a director of ATR Neural Information Processing Laboratories, Kyoto, Japan since 2018. He is engaged in integrated systems biology, with particular interests in modeling of information processing by the brain and cellular dynamics by means of informatics such as statistical science, machine learning and large-scale computation.