International Journal of Signal Processing, Image Processing and Pattern Recognition
Vol.8, No.12 (2015), pp.107-120
http://dx.doi.org/10.14257/ijsip.2015.8.12.12

Visual Tracking with Online Incremental Deep Learning and Particle Filter

Shuai Cheng [1], Yonggang Cao [3,1], Junxi Sun [2] and Guangwen Liu [1*]

[1]School of Electronic Information Engineering, Changchun University of Science and Technology, Changchun, China
[2]School of Computer Science and information Technology, Northeast Normal University, Changchun, China
[3]Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Sciences, Changchun, China

[Provider:earticle] Download by IP 159.226.165.28 at August 9, 2016 9:58 AM

Abstract

To solve the problem of tracking the trajectory of a moving object and learning a deep compact image representation in the complex environment, a novel robust incremental deep learning tracker is presented under the particle filter framework. The incremental deep classification neural network was composed of stacked denoising autoencoder, incremental feature learning and support vector machine to achieve the feature-extracting and classification of particle set. Deep learning is successfully taken to express the image representations obtained effectively. Unsupervised feature learning is used to learn generic image features and transfer learning transforms knowledge from offline training to the online tracking process. The incremental feature learning was consisted of adding features and merging features to online learn compact feature set. Linear support vector machine increases the discretion for target with similar appearance and is further tuned to adapt to appearance changes of the moving object. Compared with the state-of-the-art trackers in the complex environment, the results of experiments on variant challenging image sequences show that incremental deep learning tracker solves the problem of existent trackers more efficiently, it has better robust and more accurate, especially for occlusions, background clutter, illumination changes and appearance changes.

Keywords: *particle filter, deep learning, incremental feature learning, linear support vector machine, neural network*

1. Introduction

Object tracking is one of the most important components in a wide range of applications in computer vision, such as surveillance, behavioral recognition [1]. Although object tracking has been studied for several decades [2], it remains a very challenging problem. Numerous factors affect the performance of a tracking algorithm, such as appearance change, illumination variation, occlusion, as well as background clutters.

Numerous algorithms have been proposed with focus on effective appearance models, which can be categorized into generative [3-7] and discriminative [8-13] algorithm. A generative tracking method learns an appearance model to represent the target and search for image regions with best matching scores as the results. While it is critical to construct an effective appearance model in order to handle various challenging factors in tracking. However, generative methods discard useful information surrounding target regions that can be exploited to better separate objects from backgrounds. While some trackers simply

[*] Corresponding Author

[Provider:earticle] Download by IP 159.226.165.28 at August 9, 2016 9:58 AM

use raw pixels as features, some attempts have used more informative features such as Haar features. These features are all handcrafted offline but not tailor-made for the tracked object [14].

Discriminative methods treat tracking as a binary classification problem which learns to explicitly distinguish the object being tracked from its background. Discriminative trackers which usually put more emphasis on improving the classifiers rather than the image features used. When labeled training sample is scarce, the classification model is not accurate, lead to drift and even fail.

Deep learning architectures have been used successfully to give very promising results for learning features from complex, high-dimensional unlabeled and labeled data [15]. The key to success is to make use of deep network to learn richer features via multiple nonlinear transformations. Despite the promise, the number of features still remains a nontrivial question. When there are too many features, the model may over-fit the data or converge very slowly. When there are too few features, the model may under-fit due to the lack of relevant features. Further, finding an optimal feature set size becomes even more difficult for large-scale or online datasets whose distribution may change over time, since the cross-validation may be challenging given a limited amount of time or computational resources [16].

In this paper, we propose a novel incremental deep learning tracker (IDLT) for robust visual tracking based on particle filter framework. Firstly, it uses a stacked denoising autoencoder (SDAE) [17] to learn image features and to express the image representations obtained effectively and then transfers the features learned to the online tracking task. Second, linear SVM classifier replaces the sigmoid classifier to increase the discretion for targets with similar appearance. The support vector machine(SVM) classifier can be further tuned to adapt to specific objects during the online tracking process. Third, an incremental feature learning algorithm online choose an optimal number of features depending on the data and the existing features. It learns compact image representation and can be further tuned to adapt to appearance changes of the moving object.

2. Particle Filter

The particle filter approach [18] is commonly used for visual tracking. It is a sequential Monte Carlo importance sampling method for estimating the latent state variables of a dynamical system based on a sequence of observations. Suppose s^t and y^t denote the latent state and observation variables, respectively, at time *t*. Object tracking corresponds to the problem of finding the most probable state for each time step *t* based on the observations up to the previous time step:

$$s^t = \arg\max p(s^t \mid y^{1:t-1}) = \arg\max \int p(s^t \mid s^{t-1}) p(s^{t-1} \mid y^{1:t-1}) ds^{t-1} \tag{1}$$

When a new observation y^t arrives, the posterior distribution of the state variable is updated according to Bayes' rule:

$$p(s^t \mid y^{1:t}) = \frac{p(y^t \mid s^t) p(s^t \mid y^{1:t-1})}{p(y^t \mid y^{1:t-1})} \tag{2}$$

What is specific to the particle filter approach is that it approximates the true posterior state distribution $p(s^t \mid y^{1:t-1})$ by a set of n samples, called particles, $\{s_i^t\}_{i=1}^n$ with corresponding importance weights $\{\omega_i^t\}_{i=1}^n$ which sum to 1. The particles are drawn from an importance distribution $q(s^t \mid s^{1:t-1}, y^{1:t})$ and the weights are updated as follows:

$$\omega_i^t = \omega_i^{t-1} \cdot \frac{p(y^t \mid s_i^t)\, p(s_i^t \mid s_i^{t-1})}{q(s^t \mid s^{1:t-1}, y^{1:t})} \tag{3}$$

For the choice of the importance distribution $q(s^t \mid s^{1:t-1}, y^{1:t})$, it is often simplified to a first-order Markov process $q(s^t \mid s^{1:t-1})$ in which state transition is independent of the observation. Consequently, the weights are updated as $\omega_i^t = \omega_i^{t-1} p(y^t \mid s_i^t)$. Note that the sum of weights may no longer be equal to 1 after each weight update step. In case it is smaller than a threshold, re-sampling is applied to draw *n* particles from the current particle set in proportion to their weights and then resetting their weights to $1/n$. If the weight sum is above the threshold, linear normalization is applied to ensure that the weights sum to 1. For each frame, the tracking result is simply the particle with the largest weight.

3. Incremental Deep Classification Neural Network

Incremental deep classification neural network(IDCNN) consists of three modules in Figure 1. The first module is encoder of SDAE that extracts the feature vector from particle set through forward pass to express the image representations obtained effectively. The second module is incremental feature learning that optimize the number of feature set. The third module is binary class specific linear SVM that classifies the feature set to get the confidence of all particles in a frame.

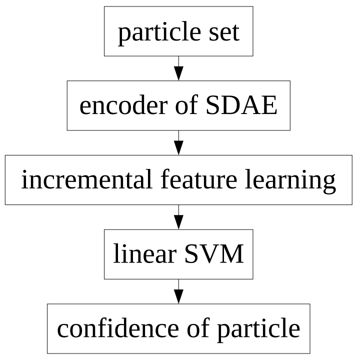


Figure 1. Incremental Deep Classification Neural Network

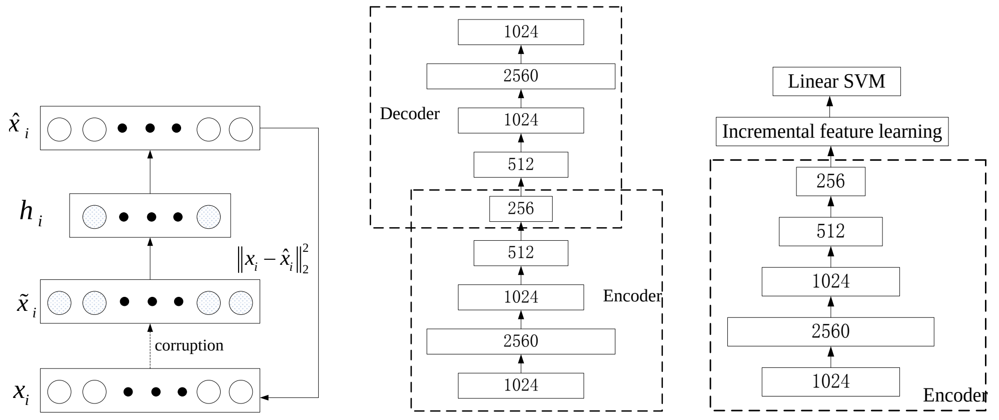
3.1. SDAE

The basic building block of an SDAE is a one-layer neural network called a denoising autoencoder(DAE). It learns to recover a data sample from its corrupted version. In so doing, robust features are learned since the neural network contains a “bottleneck” which is a hidden layer with fewer units than the input units. We show the architecture of DAE in Figure 2(a).

The layer-by-layer learning algorithm is a very effective way to pre-train the weights and the bias of a deep DAE by sparsity constraints[19]. After the pre-training multiple layers of feature detectors, the model is unrolled to produce encoder and decoder networks that initially use the same weights. the SDAE can be form a feed-forward neural network. The whole network is fine-tuned using the classical back-propagation algorithm.

For the network architecture, we use over-complete filters in the first layer. This is a deliberate choice since it has been found that an over-complete basis can usually capture the image structure better. Then the number of units is reduced by half whenever a new layer is added until there are only 256 hidden units. The whole structure of the SDAE is depicted in Figure 2(b).

[Provider:earticle] Download by IP 159.226.165.28 at August 9, 2016 9:58 AM



(a) denoising autoencoder (b) stacked denoising autoencoder (c) network for online tracking

Figure 2. The Network Architecture

3.2. Linear SVM Classifier

Linear SVM is originally formulated for binary classification. Given training data and its corresponding labels (x_n, y_n), $n = 1, \cdots, \mathbf{N}$, $x_n \in \Re^D$, $y_n \in \{-1, +1\}$, linear SVM learning consists of the following unconstrained optimization:

$$\min_{w} \frac{1}{2} \boldsymbol{w}^T \boldsymbol{w} + C \sum_{n=1}^{N} \max(1 - \boldsymbol{w}^T x_n y_n, 0)^2 \tag{4}$$

The objective of Eq. 4 is known as the primal form problem of L2-SVM, with the standard the squared hinge loss.

To predict the class label of a test data x:

$$\arg\max_{y} (w^T x) y \tag{5}$$

We propose using binary classification SVM's objective to train deep neural networks for classification tasks. Lower layer weights are learned by back-propagating the gradients from the SVM. To do this, we need to differentiate the SVM objective with respect to the activation of the penultimate layer. Let the objective in Eq. 4 be $l(w)$, and the input x is replaced with the penultimate activation h, for the L2-SVM, we have

$$\frac{\partial l(\boldsymbol{w})}{\partial h_n} = -2C y_n \boldsymbol{w} (\max(1 - w^T h_n y_n, 0)) \tag{6}$$

3.3. Incremental Feature Learning

Incremental feature learning algorithm[16] is composed of two key processes: adding new feature mappings to the existing feature set. Merging parts of the existing feature mappings that are considered redundant.

We define an objective function to determine when to add or merge features and how to learn the new features. Specifically, we form a set B which is composed of hard training examples whose objective function values are greater than a certain threshold and use these examples as input data to greedily initialize the new features. We only begin adding features when $|B| > \tau$.

The overall algorithm is schematically shown in Figure 3, and a pseudo-code is provided in Algorithm 1. During the training, our incremental feature learning algorithm optimizes the parameters of new features (the orange units and the red units on the right), while holding the other features (green units outside the dotted blue box on the left). The

[Provider:earticle] Download by IP 159.226.165.28 at August 9, 2016 9:58 AM

[Provider:earticle] Download by IP 159.226.165.28 at August 9, 2016 9:58 AM

orange units are incremental features, and the red unit is the merged feature from the similar existing features depicted as green units in the dotted red box.

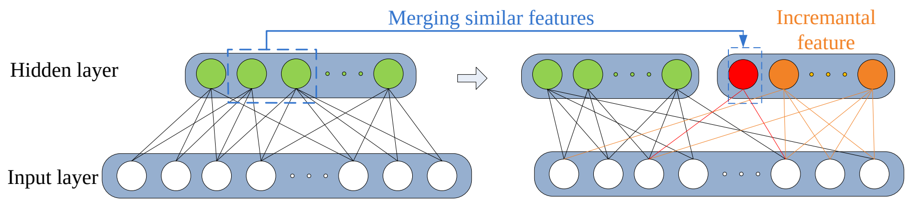


Figure 3. Illustration of Incremental Feature Learning

```
Algorithm 1 Incremental feature learning
repeat
    Compute the objective $L(x)$ for input $x$.
    Collect hard examples into a subset $B$ (i.e., $B \leftarrow B \cup \{x\}$ if $L(x) > \mu$).
    if $|B| > \tau$ then
        Select $2\Delta M$ candidate features and merge them into $\Delta M$ features.
        Add $\Delta N$ new features by greedily optimizing with respect to the subset $B$.
        Set $B = \theta$ and update $\Delta N_t$ and $\Delta M_t$.
    end if
    Fine-tune all the features jointly with in current batch of data.
until convergence
```

In this paper, we denote D for input dimension, M for the number of (existing) features, K for the binary class labels, $K \in \{0,1\}$. Based on these notations, the parameters for generative training are defined as the weight matrix $W \in \Re^{M \times D}$, the hidden bias $b \in \Re^{M}$, and the input bias $c \in \Re^{D}$. Similarly, the parameters for discriminative training are composed of the weight matrix $\Gamma \in \Re^{K \times M}$ between hidden and output units, the output bias $v \in R^{K}$, as well as the weight matrix W and the hidden bias b. We use θ to denote all parameters $\{W, b, c, \Gamma, v\}$.

For incremental feature learning, we use N to denote new features and O to denote existing or old features. For example, $f_{\mathrm{O}} \equiv h_{\mathrm{O}} \in [0,1]^{M}$ represents an encoding function with existing features, and $f_{\mathrm{N}} \equiv h_{\mathrm{N}} \in [0,1]^{N}$ denotes an encoding function with newly added features. A combined encoding function is then written as $f_{\mathrm{O} \cup \mathrm{N}}(\tilde{x}) = [h_{\mathrm{O}}; h_{\mathrm{N}}] \in [0,1]^{M+N}$. Likewise, θ_{O} refers to the existing parameters, *i.e.*, $\{W_{\mathrm{O}}, b_{\mathrm{O}}, c, \Gamma_{\mathrm{O}}, v\}$, and $\theta_{\mathrm{N}} = \{W_{\mathrm{N}}, b_{\mathrm{N}}, c, \Gamma_{\mathrm{N}}, v\}$ denotes the parameters for new features.

3.3.1. Adding Feature: We describe an efficient learning algorithm for adding features with different types of objective functions, such as generative, discriminative, and hybrid objectives. The basic idea for efficient training is that only the new features and corresponding parameters θ_{N} are trained to minimize the objective function while keeping θ_{O} fixed.

A generative objective function measures an average reconstruction error between the input x and the reconstruction $\hat{x}$. The cross-entropy function, assuming that the input and output values are both within interval [0, 1], is used as an objective function. The optimization problem is posed as:

$$\min_{W_{\mathrm{N}}, b_{\mathrm{N}}} \frac{1}{|B|} \sum_{i \in B} L_{gen}(x^{(i)}) \tag{7}$$

[Provider:earticle] Download by IP 159.226.165.28 at August 9, 2016 9:58 AM

where $L_{gen}(x)$ is cross-entropy function. For incremental feature learning, the SDAE optimizes the new features to reconstruct the data x from the corrupted data $\tilde{x}$. The encoding and decoding functions for the new features can be written as:

$$h_{\mathrm{N}} = sigm(W_{\mathrm{N}}\tilde{x} + b_{\mathrm{N}}) \tag{8}$$

$$\hat{x} = sigm(W_{\mathrm{N}}^{T}h_{\mathrm{N}} + W_{\mathrm{O}}^{T}h_{\mathrm{O}} + c) \tag{9}$$

The output $\hat{x}$ depends on both new features h_{N} and old features h_{O} as described in equation (9), the training of incremental feature θ_{N} that minimizes the residual of the objective function is still highly efficient because θ_{O} is fixed during the training. From another point of view, we can interpret $W_{\mathrm{O}}^{T}h_{\mathrm{O}}$ as a part of the bias. Thus, we can rewrite equation (9) as:

$$\hat{x} = sigm(W_{\mathrm{N}}^{T}h_{\mathrm{N}} + c_{d}(h_{\mathrm{O}})) \tag{10}$$

where $c_{d}(h_{\mathrm{O}}) \equiv W_{\mathrm{O}}^{T}h_{\mathrm{O}} + c$ is viewed as a dynamic decoding bias. Since the parameters for existing features and the corresponding activations are not changing during the training of new features, we compute h_{O} once and recall the $c_{d}(h_{\mathrm{O}})$ repeatedly for each training example without additional computational cost.

A discriminative objective function computes an average classification loss between the actual label $y \in [0,1]$ and the predicted label by $\hat{y} \in [0,1]$. More precisely, we pose an optimization problem of linear SVM as follows:

$$\min_{W_{\mathrm{N}}, b_{\mathrm{N}}, \Gamma_{\mathrm{N}}} \frac{1}{|B|}\sum_{i \in B} L_{disc}(x^{(i)}, y^{(i)}) \tag{11}$$

where $L_{disc}(x, y) = \max(1 - y\hat{y}(x), 0)^{2}$. Note that the label y is a binary vector. Formally speaking, the discriminative model predicts class labels via the L2-SVM function.

$$\hat{y} = SVM\,(v + \Gamma_{\mathrm{O}} f_{\mathrm{O}}(\tilde{x}) + \Gamma_{\mathrm{N}} f_{\mathrm{N}}(\tilde{x})) \tag{12}$$

A similar interpretation for $v + \Gamma_{\mathrm{O}} f_{\mathrm{O}}(\tilde{x})$, as in the generative training, is possible, and therefore, we can efficiently train the new parameters $\{W_{\mathrm{N}}, b_{\mathrm{N}}, \Gamma_{\mathrm{N}}\}$ using gradient descent.

Considering the discriminative model as a single objective function has a risk of over-fitting. As a remedy, we can use a hybrid objective function that combines the discriminative and generative loss function as follows:

$$L_{hybrid}(x, y) = L_{disc}(x, y) + \eta L_{gen}(x) \tag{13}$$

3.3.2. Merging Features: As described in the previous section, adding features could potentially result in many redundant features and over-fitting. To deal with this problem, we consider merging similar features to produce more compact feature representations.

Our merging process is done in two steps: we select a pair of candidate features and merge them to a single feature. Detailed descriptions are given below:

(1) Find a pair of features whose distance is minimal: $\hat{E} = \arg\min_{\{e_1, e_2\}} d(W_{e_1}, W_{e_2})$

(2) Add new features to $\theta_{\mathrm{O}\setminus E}$ by solving by $\hat{\theta}_{\mathrm{N}} = \arg\min_{\theta_{\mathrm{N}}} \frac{1}{|B|}\sum_{i \in B} L_{hybird}(x^{(i)}, y^{(i)})$.

Given the candidate features to merge, a newly added feature can be trained via gradient descent as described in the previous section.

4. Implementation Details

Flowchart of IDLT algorithm in Figure 4.

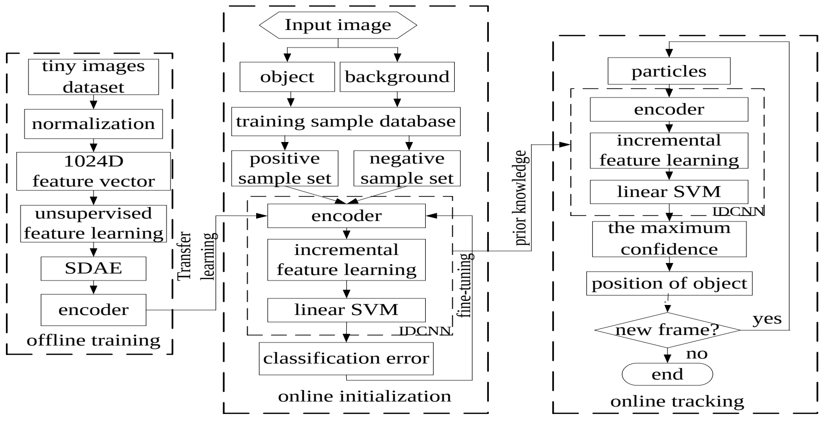


Figure 4. Flowchart of the IDLT

[Provider:earticle] Download by IP 159.226.165.28 at August 9, 2016 9:58 AM

For offline phase, we use the Tiny Images dataset [21] as auxiliary data for offline training processing. From the almost 80 million tiny images each of size 32×32, we randomly sample 3 million images for offline training. Consequently, each image is represented by a vector of 1024 dimensions corresponding to 1024 pixels. The feature value of each dimension is linearly scaled to the range [0-1].

To further speed up pre-training in the first layer to learn localized features, we divide each 32×32 tiny image into five 16×16 patches (upper left, upper right, lower left, lower right, and the center one which overlaps with the other four), and then train five DAEs each of which has 512 hidden units[14]. After that, we initialize a large DAE with the weights of the five small DAEs and then train the large DAE normally. Some randomly selected filters in the first layer are shown in Figure 5. As expected, most of the filters play the role of highly localized edge detectors.

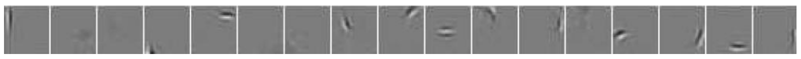


Figure 5. Some Filters in the First Layer of the Learned SDAE

For online initialization phase, the object to track is specified by the location of its bounding box in the first frame by selecting manually. For robust tracking, the full view database is established by distorting, rotating, scaling the target as positive sample. Some negative examples are collected from the background at a short distance from the object. A incremental feature learning layer and linear SVM classification layer is then added to the encoder part of the SDAE obtained from offline training. The overall network architecture is shown in Figure 2(c). Incremental deep classification neural network is trained by the positive and negative sample set using the supervised training. The SDAE encoder extracts the feature vector of the positive and negative sample set, incremental feature learning layer optimization the number of features, trains linear SVM classifier by using the feature set and the label of class, and fine tune the encoder.

In online tracking process, when a new video frame arrives, we first draw particles according to the particle filter approach. The confidence c_i of each particle is then determined by making a simple forward pass through the incremental deep classification neural network. If the confidence of all particles in a frame is above a predefined threshold τ, the tracking result is the particle with the maximum confidence. If the maximum confidence is below τ, it may indicate significant appearance change of the object being tracked due to the change of the environment. The maximum confidence of all particles is considered as the final result of tracking. The current tracking result and the background at a short distance from the object are rotating and scaling to update the positive set and the negative set. The whole network can be tuned by the positive set and the negative set to update the parameter of the whole network.

We note that the threshold τ should be set by maintaining a tradeoff. If τ is too small, the tracker cannot adapt well to appearance changes. On the other hand, if τ is too large, even an occluding object or the background may be mistreated as the tracked object and hence leads to drifting of the target.

5. Experiments

We empirically compare IDLT with some state-of-the-art trackers in this section using 8 challenging benchmark video sequences. These trackers are: IVT [3], MIL [12], OAB [8], SBT [9], TLD [22], DLT [14], CT [13], CXT [23], Frag [24], KMS [25]. We use the original implementations of these trackers provided by their authors.

5.1. Quantitative Comparison

We use two common performance metrics for quantitative comparison: success rate and f-measure. Let BB_T denote the bounding box produced by a tracker and BB_G the ground-truth bounding box. For each video frame, a tracker is considered successful if the overlap percentage $area(BB_T \cap BB_G)/area(BB_T \cup BB_G) > 50\%$. As for f-measure, it is defined as $F = 2PR/(P+R)$, where precision P, recall R. The quantitative comparison results are summarized in Table 1, 2. For each row which corresponds to one of 8 video sequences, the best result is shown in red and second best in blue. On average, IDLT is the best according to both performance metrics. As for the central-pixel error, it is defined as the Euclidean distance (in pixels) between the centers of BB_T and BB_G. We report the central-pixel errors over all frames for each video sequence in Figure 6. The central-pixel errors of IDLT is lowest than others overall.

Table 1. Comparison of 11 Trackers on 8 Video Sequences in the Success Rate

Seq.	IDLT	CT	CXT	DLT	Frag	IVT	KMS	TLD	MIL	OAB	SBT
animal	0.98	0.03	0.20	0.15	0.15	0.03	0.51	0.73	0.13	0.96	0.80
car11	1.00	0.00	0.60	0.80	0.28	0.69	0.39	0.54	0.18	0.95	0.93
football	0.84	0.19	0.66	0.34	0.92	0.75	0.02	0.43	0.74	0.36	0.23
girl	0.94	0.06	0.69	0.69	0.58	0.21	0.40	0.78	0.30	0.95	0.56
moubike	0.91	0.53	0.28	0.31	0.14	1.00	0.57	0.26	0.58	0.92	0.59
shaking	0.87	0.03	0.12	0.02	0.09	0.01	0.15	0.40	0.23	0.01	0.08
subway	0.99	0.63	0.23	0.32	0.51	0.06	0.25	0.23	0.81	0.22	0.38
woman	0.93	0.16	0.26	0.09	0.18	0.19	0.12	0.17	0.19	0.62	0.37
mean	0.92	0.20	0.38	0.34	0.36	0.37	0.30	0.44	0.40	0.62	0.49

Table 2. Comparison of 11 Trackers on 8 Video Sequences in the f-Measure

Seq.	IDLT	CT	CXT	DLT	Frag	IVT	KMS	TLD	MIL	OAB	SBT
animal	0.74	0.2	0.35	0.20	0.12	0.03	0.41	0.60	0.13	0.72	0.67
car11	0.82	0.03	0.56	0.69	0.32	0.62	0.36	0.45	0.20	0.79	0.84
footBall	0.63	0.34	0.55	0.31	0.70	0.57	0.07	0.50	0.59	0.34	0.22
girl	0.73	0.31	0.59	0.54	0.47	0.17	0.34	0.58	0.41	0.73	0.49
mountbike	0.74	0.41	0.23	0.49	0.13	0.73	0.47	0.20	0.46	0.64	0.45
shaking	0.70	0.14	0.11	0.04	0.09	0.03	0.21	0.39	0.43	0.01	0.26
subway	0.65	0.52	0.18	0.42	0.45	0.07	0.19	0.19	0.66	0.17	0.29
woman	0.72	0.13	0.24	0.25	0.14	0.15	0.10	0.13	0.16	0.49	0.35
mean	0.74	0.24	0.35	0.36	0.30	0.30	0.27	0.38	0.38	0.49	0.45

[Provider:earticle] Download by IP 159.226.165.28 at August 9, 2016 9:58 AM

[Provider:earticle] Download by IP 159.226.165.28 at August 9, 2016 9:58 AM

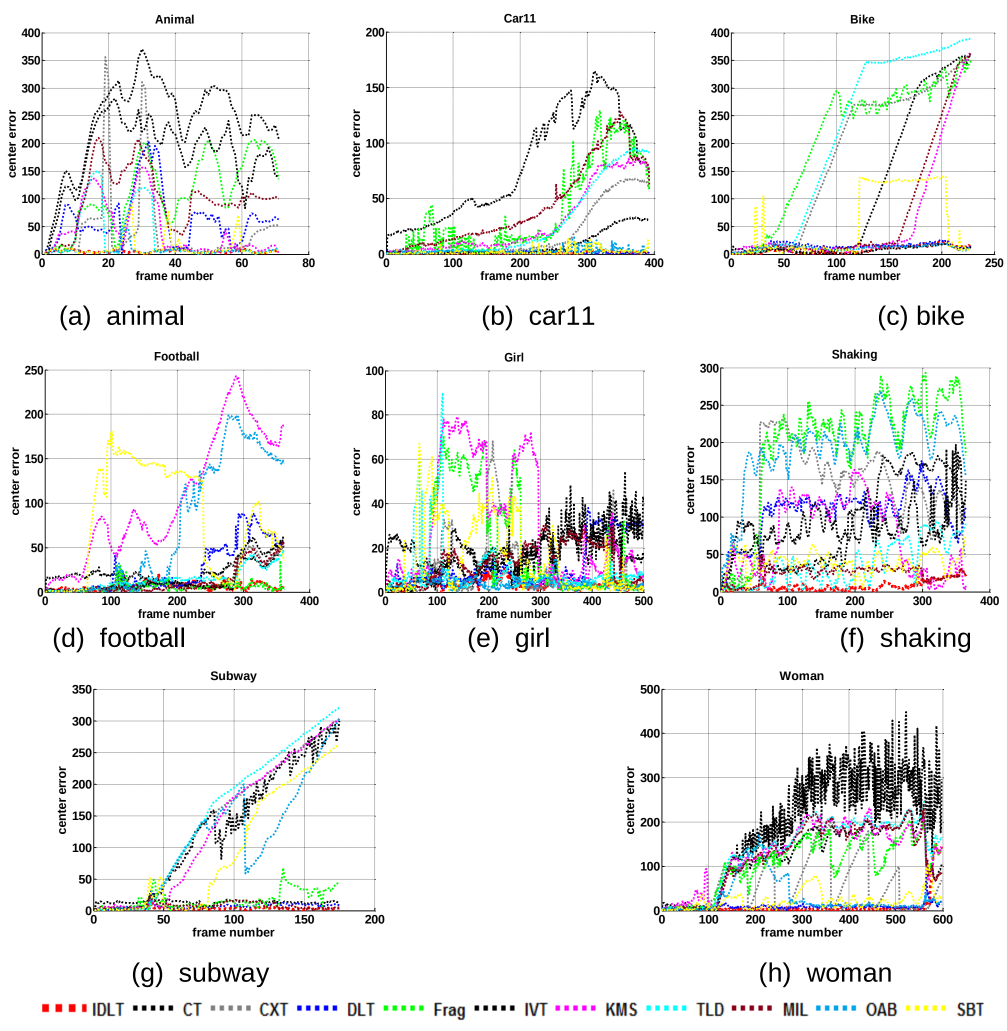


Figure 6. The Comparison of Central-Pixel Error

5.2. Qualitative Comparison

Figure 7 shows some key frames with bounding boxes reported by all 11 trackers for each of the 8 video sequences.

In the animal sequence, the target is a fast moving animal with motion blur. IDLT and OAB can merely track the target in the cluttered background. When the object move rapidly, other tracker drift and even fail during tracking.

In car11 sequences, the tracked object is car moving on an open road. At frame 114, CT, MIL and Frag drift to different degrees due to illumination variation. The environment is very dark with illumination in the cluttered background during tracking. IDLT and DLT can also track the car accurately, other tracker drift and fail.

The football sequence is very challenging since the complex environment of pose variation, occlusion, as well as rotation. All methods can track the target to the end, except OAB、KMS, but drift to some extends.

In the girl sequence, each tracker has to track the head of girl. The sequence is challenging because pose vary and rotate drastically along the video, *e.g.*, at frame 86, 132, 229, 239, 384, almost tracker drift. When the object is occluded at frame 441, DLT drift seriously and Frag, MIL, CT even fail to different degrees. IDLT yield the best results.

In the bike sequence, the goal is to track biker while its pose changes along the video sequence. As a consequence, all trackers can merely track it except that CT, CXT, Frag, SBT show an incorrect tracking.

The shaking sequence is recordings on the stage with illumination changes. For shaking, the pose of the head being tracked also changes. OAB, IVT and CT totally fail before frame 49, while SBT and MIL show some drifting effects then at frame 180. KMS and IDLT satisfactory results.

In the subway sequence, the target is to track a man walking. Some trackers fail or drift when it is occluded. IDLT, CT, DLT can track the target to the end, but DLT drift.

In the woman sequence, we track a woman walking in the street. The woman is severely occluded several times by the parked cars. TLD first fails at frame 138 because of the pose change. All other trackers compared fail when the woman walks close to the car at about frame 220. DLT can follow the target accurately.

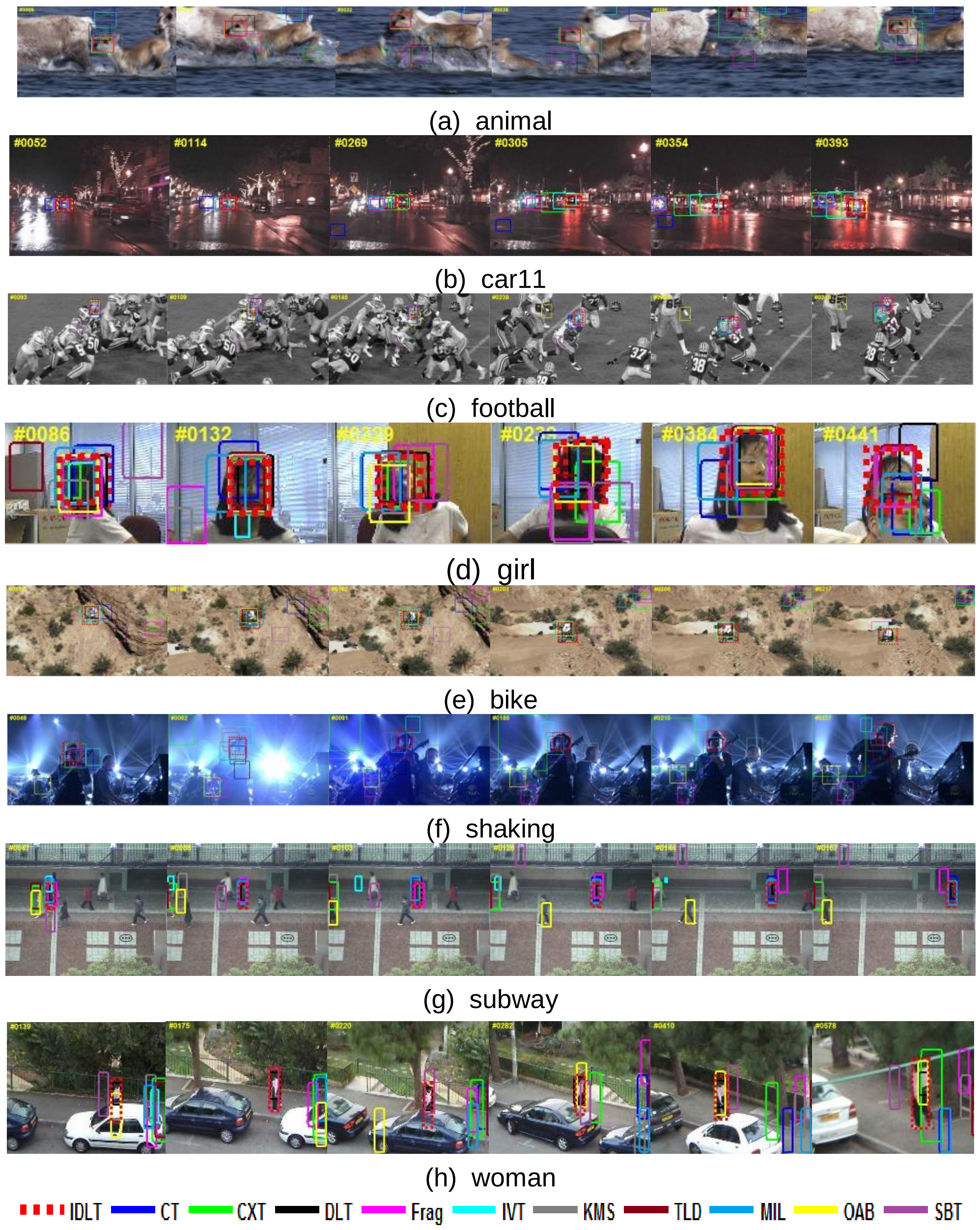


Figure 7. Comparison of 11 Trackers on 8 Video Sequences

[Provider:earticle] Download by IP 159.226.165.28 at August 9, 2016 9:58 AM

[Provider:earticle] Download by IP 159.226.165.28 at August 9, 2016 9:58 AM

6. Conclusions and Future Work

This paper studies the visual tracking problem and presents a novel robust incremental deep learning tracker under the particle filter framework. We have successfully taken deep learning, incremental feature learning and linear SVM to a tracking territory. A stacked denoising autoencoder learns useful features using unsupervised feature learning. Transfer learning transfers the encoder part of the SDAE from offline training to online tracking to alleviates the problem of not having much labeled data in visual tracking. During the online tracking process, train a incremental deep classification neural network to distinguish the tracked object from the background and optimize feature set. Since further fine-tuning is allowed during the online tracking process, both the feature extractor and the classifier can adapt to appearance changes of the moving object. Through comparison with state-of-the-art trackers on some challenging benchmark sequences, we demonstrate that our incremental deep learning tracker gives better robust and higher accuracy in the complex environment.

It would be an interesting direction to investigate a deep convolution neural network model. Also, the classification layer in our current tracker is just a linear SVM classifier for simplicity. Extending it to more powerful classifiers may provide more room for further performance improvement.

Acknowledgments

This research has been supported by the National Natural Science Foundation of China (Grant No. 61172111), and Science and Technology Development of Jilin province (20090512, 20100312).

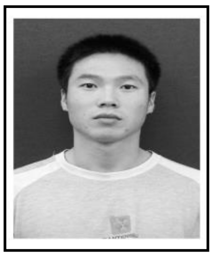
References

[1] K. Cannons, "A Review of Visual Tracking", Technical Report CSE-2008-07, York University, Canada, **(2008)**.
[2] Y. Wu, J. Lim and M. Yang, "Online object tracking: A benchmark", Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Portland, USA, **(2013)** June 25-27, pp. 2411-2418.
[3] D. Ross, J. Lim, R. Lin and M.-H. Yang, "Incremental learning for robust visual tracking", International Journal of Computer Vision, vol. 77, no. 1, **(2008)**, pp. 125-141.
[4] J. Kwon and K. M. Lee, "Visual tracking decomposition", Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, San Francisco, USA, **(2010)** June 13-18, pp. 1269-1276.
[5] J. Kwon and K. M. Lee, "Tracking by sampling trackers", Proceedings of IEEE International Conference on Computer Vision, Barcelona, Spain, **(2011)** November 6-13, pp. 1195–1202.
[6] H. Li, C. Shen and Q. Shi, "Real-time visual tracking using compressive sensing", Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Colorado, Canada, **(2011)** June 20-25, pp. 1305-1312.
[7] L. Sevilla-Lara and E. Learned-Miller, "Distribution fields for tracking", Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Providence, USA, (**2012)** June 16-21, pp. 1910-1917.
[8] H. Grabner, M. Grabner and H. Bischof, "Real-time tracking via on-line boosting", Proceedings of 17th British Machine Vision Association, Edinburgh, Britain, **(2006)** September 4-7, pp. 47-56.
[9] H. Grabner, C. Leistner and H. Bischof, "Semi-supervised on-line boosting for robust tracking", Proceedings of the 10th European Conference on Computer Vision, Marseille, France, **(2008)** October 12-18, pp. 234–247.
[10] Z. Kalal, J. Matas and K. Mikolajczyk, "P-N learning: Bootstrapping binary classifiers by structural constraints", Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, San Francisco, USA, **(2010)** June 13-18, pp. 49–56.
[11] S. Hare, A. Saffari and P. H. Torr, "Struck: Structured output tracking with kernels", Proceedings of IEEE International Conference on Computer Vision, Barcelona, Spain, **(2011)** November 6-13, pp. 263-270.
[12] B. Babenko, M.-H. Yang and S. Belongie, "Robust object tracking with online multiple instance learning", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 33, no. 8, **(2011)**, pp. 1619-1632.
[13] K. Zhang, L. Zhang and M.-H. Yang, "Real-time compressive tracking", Proceedings of the 12th European Conference on Computer Vision, Firenze, Italy **(2012)** October7-13, pp. 864-877.

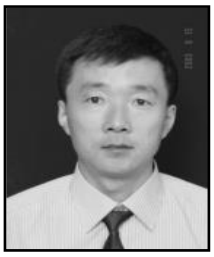
[14] N. Wang and D.Y Yeung, "Learning a Deep Compact Image Representation for Visual Tracking", Proceedings of Twenty 7th Annual Conference on Neural Information Processing Systems, Lake Tahoe, USA, (**2013)** December 5-9, pp. 5-10.
[15] G. E. Hinton and R. R. Salakhutdinov, "Reducing the Dimensionality of Data with Neural Networks", Science, vol. 313, no. 5786, **(2006)**, pp. 504-507.
[16] G. Y. Zhou, Kihyuk Sohn and Honglak Lee, "Online Incremental Feature Learning with Denoising Autoencoders", Journal of Machine Learning Research-Proceedings Track, **(2012)**, pp.1453-1461.
[17] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion", Journal of Machine Learning Research, vol.11, **(2010)**, pp.3371-3408.
[18] A. Doucet, D. N. Freitas and N. Gordon, "Sequential Monte Carlo Methods In Practice", Springer, New York, **(2001)**.
[19] G. Hinton, "A practical guide to training restricted Boltzmann machines", In Neural Networks: Tricks of the Trade, **(2012)**, pp.599–619.
[20] Y. Tang, "Deep Learning using Linear Support Vector Machines", ICML 2013 Challenges in Representation Learning Workshop, **(2013)**.
[21] A. Torralba, R. Fergus and W. Freeman, "80 million tiny images: A large data set for nonparametric object and scene recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 30, no.11, **(2008)**, pp.1958-1970.
[22] Z. Kalal, K. Mikolajczyk and J. Matas, "Tracking-learning-detection", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 34, no. 7, **(2012)**, pp. 1409-1422.
[23] T. B. Dinh, N. Vo and G. Medioni, "Context Tracker: Exploring supporters and distracters in unconstrained environments", Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Colorado, Canada, **(2011)** June 20-25, pp. 1305-1312.
[24] A. Adam, E. Rivlin and I. Shimshoni, "Robust Fragments-based Tracking using the Integral Histogram", Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, New York, USA, **(2006)** June, pp.798-805.
[25] D. Comaniciu, V. Ramesh and P. Meer, "Kernel-Based Object Tracking", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 25, no. 5, **(2003)**, pp. 564-577.

[Provider:earticle] Download by IP 159.226.165.28 at August 9, 2016 9:58 AM

Authors



Shuai Cheng, studies at Changchun University of Science and Technology, is completing a Ph.D. at present. His research interests are image processing and object tracking and deep learing.



Yonggang Cao, received his B.Sc. degree from Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Sciences. He is now pursuiting Ph.D degree in Changchun University of Science and Technology. His research intersets are image processing and pattern recognition.

Junxi Sun, received his B.Sc. and M.Sc. degrees from Department of Electronic Engineering, Changchun Institute of Optics and Fine Mechanics, Changchun, China, and the Ph.D. degree from Department of Biomedical Engineering, Shanghai Jiaotong University, Shanghai, China. He is now working as a professor in Northeast Normal University. His research interests are image processing and pattern recognition and intelligent system.

Correspondence author:

Guangwen Liu: received his the Ph.D. degree from Changchun University of Science and Technology at 2009. He is now working as a associate professor in Changchun University of Science and Technology. His research interests are image processing and intelligent system.

[Provider:earticle] Download by IP 159.226.165.28 at August 9, 2016 9:58 AM

[Provider:earticle] Download by IP 159.226.165.28 at August 9, 2016 9:58 AM