Confidence-Based Data Association and Discriminative Deep Appearance Learning for Robust Online Multi-Object Tracking

Seung-Hwan Bae and Kuk-Jin Yoon, *Member, IEEE*

Abstract—Online multi-object tracking aims at estimating the tracks of multiple objects instantly with each incoming frame and the information provided up to the moment. It still remains a difficult problem in complex scenes, because of the large ambiguity in associating multiple objects in consecutive frames and the low discriminability between objects appearances. In this paper, we propose a robust online multi-object tracking method that can handle these difficulties effectively. We first define the tracklet confidence using the detectability and continuity of a tracklet, and decompose a multi-object tracking problem into small subproblems based on the tracklet confidence. We then solve the online multi-object tracking problem by associating tracklets and detections in different ways according to their confidence values. Based on this strategy, tracklets sequentially grow with online-provided detections, and fragmented tracklets are linked up with others without any iterative and expensive association steps. For more reliable association between tracklets and detections, we also propose a deep appearance learning method to learn a discriminative appearance model from large training datasets, since the conventional appearance learning methods do not provide rich representation that can distinguish multiple objects with large appearance variations. In addition, we combine online transfer learning for improving appearance discriminability by adapting the pre-trained deep model during online tracking. Experiments with challenging public datasets show distinct performance improvement over other state-of-the-arts batch and online tracking methods, and prove the effect and usefulness of the proposed methods for online multi-object tracking.

Index Terms—Multi-object tracking, tracking-by-detection, tracklet confidence, confidence-based data association, deep appearance learning, online transfer learning, surveillance system

1 Introduction

The goal of multi-object tracking (MOT) is to estimate the states of multiple objects, such as locations, velocities, and sizes, while conserving their identifications under appearance and motion variations with time. In a complex scene, this problem is still challenging due to frequent occlusion of target objects by a clutter or other objects, similar appearances of target objects, and so on.

To solve this problem, many different methods have been proposed for decades. Among them, tracking-by-detection methods have shown impressive performance improvement in multi-object tracking thanks to the development of reliable object detectors [1], [2]. The tracking-by-detection methods generally build long tracks of objects by associating detections provided by detectors. Thus, these methods can recover tracking failures by finding the object hypothesis from the detections. In addition, by using detections, the search space of object hypothesis can be greatly reduced and new track initialization can be also achieved automatically.

The tracking-by-detection methods can be roughly categorized into batch and online methods. Batch methods [3], [4], [5], [6], [7] usually utilize the detections of all the frames of the sequence together to build long tracks robustly against occlusion and false detections. In general, given a set of detections, short tracklets are generated first by linking individual detections, and the tracklets are then globally associated to build longer tracklets. Therefore, the global association is very important in this approach, and many methods [3], [6], [7] for the global association have been proposed. However, the performance of the batch methods is still limited when tracking multiple objects with similar appearances. Moreover, since they usually require the detections for an entire sequence beforehand and also require expensive computation for the iterative associations to generate globally optimized tracks, it is hard to apply the batch methods to real-time applications.

On the other hand, online methods [8], [9], [10], [11], [12], [13], [14] can be applied to real-time applications because they sequentially build trajectories based on the frame-by-frame association using the information given up to the present frame. However, in return, the online methods tend to produce fragmented trajectories and to drift under occlusion and detection errors, because it is more difficult to handle inaccurate detections (e.g., false positive and false negative) compared to the batch methods.

- *S.-H. Bae is with Department of Computer Science and Engineering, Incheon National University, 119 Academy-ro, Yeonsu-gu, Incheon, 22012, South Korea. E-mail: shbae@inu.ac.kr.*
- *K.-J. Yoon is with the School of Information and Communications, Gwangju Institute of Science and Technology, 261 Cheomdan-Gwagiro, Buk-Gu, Gwangju 500-712, South Korea. E-mail: kjyoon@gist.ac.kr.*

Manuscript received 14 Mar. 2016; revised 4 Mar. 2017; accepted 30 Mar. 2017. Date of publication 5 Apr. 2017; date of current version 13 Feb. 2018. (Corresponding author: Kuk-Jin Yoon.)
Recommended for acceptance by I. Laptev.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TPAMI.2017.2691769

Fig. 1. Contaminated training samples in tracking sequences.

In this paper, we propose a robust *online* MOT method in consideration of the aforementioned limitations of previous methods. The proposed method is based on (1) *confidence-based data association* to handle track fragments due to occlusion or unreliable detections and (2) *discriminative deep appearance learning* to handle similar appearances of objects in tracklet association.

To handle track fragments due to occlusion or unreliable detections, we first propose the tracklet confidence based on the detectability and continuity of a tracklet. We then divide the online association problem into subproblems, which are high and low confidence (HC- & LC-) association problems based on the tracklet confidence and solve the online multi-object tracking problem by associating tracklets and detections in different ways according to their confidence values: reliable tracklets having high confidence values are locally associated with online-provided detections first and the tracklets having low confidence values are globally associated with other tracklets and detections later. Based on this strategy, tracklets sequentially grow with online-provided detections and fragmented tracklets can be linked with others without any iterative and expensive association steps.

Here, as described above, the core steps of the proposed method are the HC and LC associations. In both steps, appearance modeling is crucial for associating tracklets and detections of the same object while distinguishing different objects. Actually, in recent years, the appearance learning has been a key issue in MOT for handling appearance variations of target objects and improving appearance discriminability between objects. Previous methods typically learn object-specific [4], [9], [10], [15] or object-shared [12], [16] appearance models of target objects. However, the discriminability of learned appearance models is often limited due to the small number of samples and the samples contaminated by occlusions and inaccurate tracking as shown in Fig. 1

One possible solution is to learn a discriminative appearance model using well-managed large datasets beforehand, and apply this model to multi-object tracking. However, although a large number of datasets are available, previous learning methods using boosting [4], [7], [9], [16], multiple instance learning (MIL) [5], part models [10], and subspace learning [12] do not learn rich representations due to their limited capacity. This is because these methods build shallow architectures by linearly combining several classifiers or mapping features onto a linear projection matrix.

To overcome this limitation in appearance modeling, we propose the deep appearance learning that can capture discriminative deep representations from a large dataset and distinguish multiple objects even with large appearance variations. We first design a deep appearance model based on the Siamese network [17] for learning a dissimilarity metric between a pair of objects. Here, for improving the discriminability of the Siamese network, we define the new energy function based on a pairwise constraint with distances of object pairs instead of using the contrastive loss function [17] (In Table 4, both energy functions are compared). We then learn discriminative deep representations (*or* dissimilarity metric) by minimizing the energy function. To do this, we derive the gradients of the defined energy with respect to learnable parameters of each layer. As a result, the appearance model minimizes the distances for the same object pairs (i.e., positive pairs) while maximizing it for the different object pairs (i.e., negative pairs). In addition, we combine *online transfer learning (OTL)* with our deep learning for fine-tuning the learned deep model using online tracking data. As a result, we can transfer the learned deep representation to online MOT and make the deep appearance model more suitable for the specific tracking sequence, and also handle appearance variations of target objects during tracking.

To sum up, the main contributions of this paper can be summarized as follows:

(i) proposition of a tracklet confidence for evaluating tracklet's reliability, and confidence-based associations for building locally and globally optimal tracklets,
(ii) proposition of a deep learning method for discriminating different objects and adapting the learned appearances with ongoing tracking results, and (iii) proposition of a practical whole online tracking framework by effectively combining our methods, as given in Fig. 3.

2 Related Works

In this section, we introduce previous works on MOT and appearance learning for tracking, which are closely related to our work.

Given detections from a detector at each frame, online tracking methods locally associate detections frame-by-frame to build long trajectories in general. Based on the particle filtering [18], [19], [20] guides multiple trackers with detections to track objects under occlusions. An edgelet-based part model [8] is also exploited for describing appearances of objects. However, methods using the pre-deifmend appearance models [8], [20], [21], [22], [23] suffer from track drift when appearances dramatically change.

In order to solve the drift problem, online classifiers [9], [10], [11], [14], [24] trained during tracking are used to capture object appearance variations and find object hypotheses under occlusion. The confidence map [9] to combine the outputs of a pre-trained detector and an online-trained classifier is constructed. By applying the deformable part model, [10] handle partial occlusions. Recently, [24] learn a similarity function with reinforcement learning. Although they show improved performance in many scenarios, these local association-based tracking methods still tend to produce short fragmented trajectories under long-term occlusion because they only use the information in two consecutive frames.

For building long trajectories under occlusions, global association methods [3], [4], [5], [6], [7], [25], [26] build optimal tracklets with all detections for a sequence in general. A hierarchical association framework [21] is designed to produce longer tracklets at each level gradually. In [23] and [27], they solve a global data association problem using a min-cost flow algorithm in a network flow. In [3], short tracklets are merged into longer ones by finding maximum

weighted independent sets in a graph of detection pairs. [7] develop an online-learned CRF model and link tracklets by minimizing an energy function. [6] present an energy function and an optimization scheme for finding the optimal set of tracklets. In [26], the classical MHT algorithm is enhanced for MOT.

On the other hand, some methods have been proposed for discriminative appearance models for MOT. To handle appearance variations by updating appearance models, some methods [9], [11], [15], [28] employ target-specific appearance models with ensemble learning [15] and online boosting [28]. However, their appearance models are trained for distinguishing a target object from the background, rather than other objects. To learn appearance models discriminating multiple different objects, a few methods [4], [5], [7], [29] collect positive samples from the same tracklets and negative samples from other tracklets after low-level associations, and the models are simultaneously learned using standard AdaBoost [4], [7], [29] or multiple-instance learning (MIL) [5] methods. However, these learning methods are not appropriate for updating learned appearance models online because the appearance models are learned in a batch manner. In addition, the discriminability of the existing appearance learning methods [4], [7], [9], [11], [29] is often limited due to the small number of training samples and their shallow representation models.

Inspired by the recent advances in deep learning, several appearance models using deep learning have been proposed for visual object tracking in [30], [31]. These models have shown that generic features robust against appearance variations and cluttered background can be trained using deep learning. However, their models are learned for representing object appearances in different categories, while an appearance model for MOT should be learned for discriminating object appearances in the same category. Recently, discriminative deep models based on the Siamese network [17] have been developed for person re-identification [32] and face recognition [33]. For identifying two input images, they usually learn distance metrics [17], [32] and/or the correlation patterns between filters [33]. In this paper, we also design our deep appearance model based on the Siamese network for improving discriminability of objects in the same category. However, we leverage online transfer learning for handling appearance variations of target objects and make the learned model more suitable for the specific tracking sequence.

3 Online Multi-Object Tracking with Tracklet Confidence

If object i appears at frame t, we denote it by using a binary function as $v^i(t) = 1$. Otherwise, $v^i(t) = 0$. When $v^i(t) = 1$, the state of object i is represented as $\mathbf{x}_t^i = (\mathbf{p}_t^i, \mathbf{s}_t^i, \mathbf{v}_t^i)$, where $\mathbf{p}_t^i$, $\mathbf{s}_t^i$, and $\mathbf{v}_t^i$ are the position, size, and velocity, respectively. The tracklet of an object i, T^i, is then defined as a set of states up to frame t, and denoted as $T^i = \{\mathbf{x}_k^i | v^i(k) = 1, 1 \le t_s^i \le k \le t_e^i \le t\}$, where t_s^i and t_e^i are the time stamps of the start- and end-frame of the tracklet. In addition, a set of tracklets of all objects up to frame t is denoted as $\mathbb{T}_{1:t}$. Similarly, we denote the detection of object i at frame t as $\mathbf{z}_t^i$, and a set of all detections up to frame t as $\mathbb{Z}_{1:t}$. Then, the online multi-object tracking problem can be formulated as to find the optimal $\mathbb{T}_{1:t}$ by maximizing the posterior probability for given $\mathbb{Z}_{1:t}$ as

$$\hat{\mathbb{T}}_{1:t}^{\text{MAP}} = \underset{\mathbb{T}_{1:t}}{\operatorname{argmax}}\, p(\mathbb{T}_{1:t} | \mathbb{Z}_{1:t}). \tag{1}$$

Note that directly solving Eq. (1) is not feasible in practice because the number of all possible combinations of $\mathbb{T}_{1:t}$ and $\mathbb{Z}_{1:t}$ is innumerable. In practice, one sequentially and iteratively associates detections rather than associating all detections at once. In this work, we divide the association problem Eq. (1) into the subproblems Eq. (5) using the tracklet confidence Eq. (2), and sequentially associate detections with tracklets.

3.1 Tracklet Confidence

Tracklet confidence can be intuitively interpreted as the reliability measure of the tracklet constructed during tracking with time. It can be measured based on the following factors:

- Length: while a short tracklet tends to be unreliable, a long tracklet is more likely to be a reliable tracklet of an object.
- Occlusion: a tracklet severely occluded by other tracklets may not be a reliable tracklet.
- Affinity: a reliable tracklet will have a high affinity score with an associated detection.

In this paper, we model the tracklet confidence $conf(T_i)$ based on the above factors as

$$\begin{aligned} conf(T_i) \models & \left(\tfrac{1}{L}\textstyle\sum_{k\in[t_s^i,t_e^i],v^i(k)=1} \Lambda\big(T^i, \mathbf{z}_k^i\big)\right) \\ & \times\left(1 - \exp^{-\beta\cdot\sqrt{(L-w)}}\right), \end{aligned} \tag{2}$$

where L is the cardinality of T^i (i.e., the length of a tracklet) as $L = |T^i|$, and w is the number of frames in which the object i is missing due to occlusion by other objects or unreliable detection as $w = t_e^i - t_s^i + 1 - L$. The first term in Eq. (2) is the average affinity score between the tracklet and associated observations (i.e., detections): a high affinity score increases the confidence. Here, the affinity can be defined by using several cues. We define the affinity in Section 3.2. The second term in Eq. (2) is also computed with L and w together, and decreases for short or heavily occluded tracklets. β is a control parameter relying on the performance of a detector. When a detector shows high accuracy, β should be set to a large value. The first and the second terms are all closely related to the detectability and the continuity of a tracklet. Fig. 2 shows the confidence variation of an object under occlusion.

3.2 Affinity Model

We describe tracklet T^i with three elements $\{A^i, S^i, M^i\}$, where A^i, S^i and M^i represent appearance, shape, and motion models, respectively. Then, an affinity measure to determine how well two tracklets (or a tracklet and a detection) are matched is defined as

$$\Lambda(X, Y) = \Lambda^A(X, Y)\Lambda^S(X, Y)\Lambda^M(X, Y), \tag{3}$$

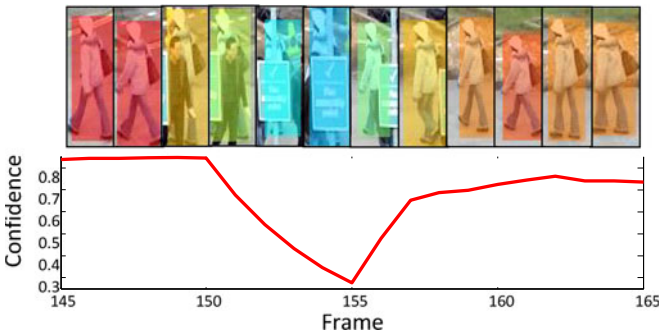


Fig. 2. Tracklet confidence variation of an object (in the PETS-L1 sequence) under occlusion. Under occlusion, the confidence decreases, but it then gradually increases by association with detections with time.

where X and Y can be tracklets or detections. Each affinity score is computed as follows:

$$\begin{aligned}\Lambda^A(X,Y) &= \exp(-1/c^2 \cdot (D_g^2(\mathbf{x}_p,\mathbf{x}_q))^2),\\ \Lambda^S(X,Y) &= \exp\Big(-\Big\{\tfrac{h_X-h_Y}{h_X+h_Y}+\tfrac{w_X-w_Y}{w_X+w_Y}\Big\}\Big),\\ \Lambda^M(X,Y) &= \mathcal{N}(\mathbf{p}_X^{tail}+\mathbf{v}_X^F\Theta;\mathbf{p}_Y^{head},O^F)\\ &\quad\times\mathcal{N}(\mathbf{p}_Y^{head}+\mathbf{v}_Y^B\Theta;\mathbf{p}_X^{tail},O^B).\end{aligned} \tag{4}$$

For the appearance affinity $\Lambda^A(X,Y)$, we exploit the proposed appearance model. We first extract output features for image pairs $\mathbf{x}_p$ and $\mathbf{x}_q$ using forward propagation and then compute their L2 distance $D_g^2(\mathbf{x}_p,\mathbf{x}_q)$ using Eq. (10). The shape affinity $\Lambda^S(X,Y)$ is calculated with their height h and width w. $\Lambda^M(X,Y)$ is the motion affinity between X tail (i.e., the last refined position) and Y head (i.e., the first refined position) with the frame gap Θ. The forward velocity $\mathbf{v}_X^F$ is evaluated from the head to the tail of X, while the backward velocity $\mathbf{v}_Y^B$ is evaluated from the tail to the head of Y. The difference between the predicted position computed with the velocity and the refined position is assumed to follow a Gaussian distribution. Note that only the forward motion is used when evaluating affinity between a tracklet and a detection.

3.3 Online MOT Formulation with Tracklet Confidence

To effectively solve the online multi-object tracking problem, we reformulate the online multi-object problem Eq. (1) by using the tracklet confidence as

$$\begin{aligned}\hat{\mathbb{T}}_{1:t} &= \underset{\mathbb{T}_{1:t}}{\operatorname{argmax}} \iint p\Big(\mathbb{T}_{1:t}|\mathbb{T}_{1:t}^{(hi)},\mathbb{T}_{1:t}^{(lo)}\Big)\times p\Big(\mathbb{T}_{1:t}^{(hi)},\mathbb{T}_{1:t}^{(lo)}|\mathbb{Z}_{1:t}\Big)\\ &= \underset{\mathbb{T}_{1:t}}{\operatorname{argmax}} \iint p\Big(\mathbb{T}_{1:t}|\mathbb{T}_{1:t}^{(hi)},\mathbb{T}_{1:t}^{(lo)}\Big)\times\\ &\quad \underbrace{p\Big(\mathbb{T}_{1:t}^{(lo)}|\mathbb{T}_{1:t}^{(hi)},\mathbb{Z}_{1:t}\Big)}_{LC-association}\underbrace{p\Big(\mathbb{T}_{1:t}^{(hi)}|,\mathbb{Z}_{1:t}\Big)}_{HC-association} d\mathbb{T}_{1:t}^{(hi)}d\mathbb{T}_{1:t}^{(lo)},\end{aligned} \tag{5}$$

Here, $\mathbb{T}_{1:t}^{(hi)}$ and $\mathbb{T}_{1:t}^{(lo)}$ represent a set of tracklets with high confidence and a set of tracklets with low confidence. As shown in Eq. (5), the problem is solved in two phases: tracklets with high confidence are locally associated with online-provided detections in the HC-association phase, while tracklets with low confidence, which are more likely to be fragmented, are globally associated with other tracklets and detections in the LC-association phase. To be more concrete, the tracklets with high confidence are first considered to be locally associated with detections, because detections are more likely to be correctly associated with the reliable tracklets with high confidence than the tracklets with low confidence. The HC-association allows us to progressively grow locally optimal tracklets with online provided-detections. The target object being tracked, however, is frequently not detected (by a detector) due to occlusion or unreliable detectors. When a detection of the object is not available, the confidence of a tracklet decreases. Therefore, we consider the tracklets with low confidence as fragmented and unreliable tracklets, and globally associate them with other tracklets and detections in the LC-association. The overall framework of the proposed method is shown in Fig. 3. Here, since the tracklet confidence lies in $[0,1]$, we consider a tracklet as a reliable tracklet with high confidence when $conf(T^i) > 0.5$;

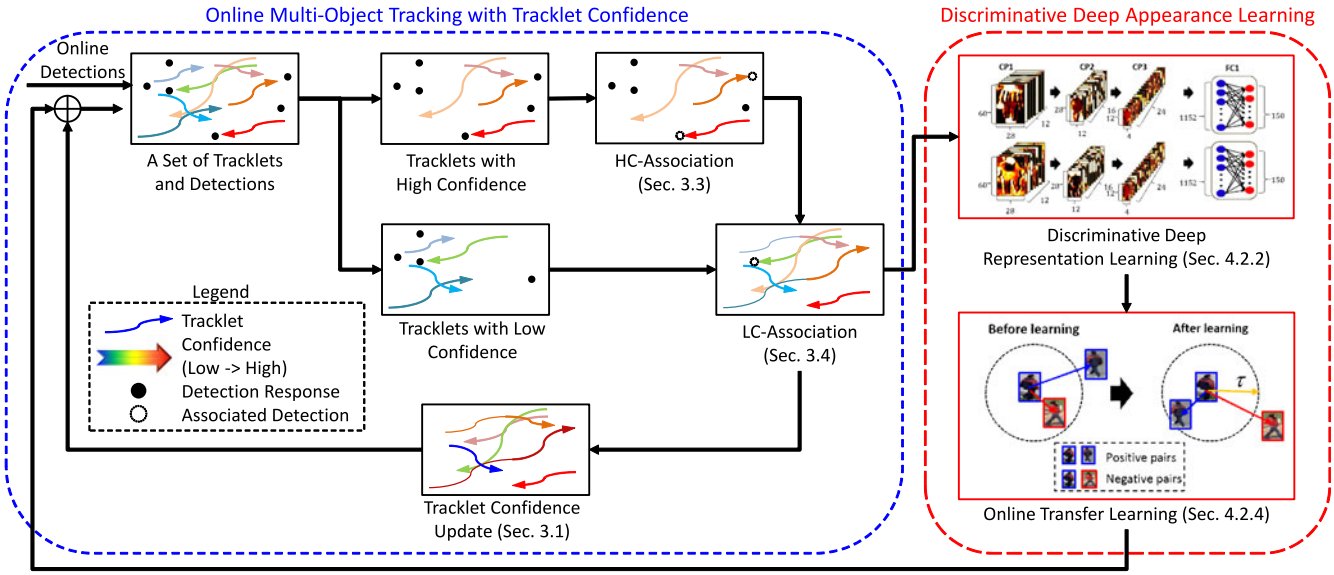


Fig. 3. Proposed framework for robust online multi-object tracking. Colors of tracklets indicate their confidence values.

otherwise it is considered as the unreliable tracklet with low confidence. In our experiment, the tracking performance, however, is not significantly affected by this threshold.

3.4 High Confidence (HC-) Association of Tracklets

In the HC-association, tracklets with high confidence, $T^{i(hi)}$, sequentially grow with a set of detections at frame t, $\mathbb{Z}_t$. Pairwise association is performed to associate detection responses with tracklets. When h tracklets with high confidence and n detections are given at frame t, we compute a score matrix $S_{h\times n}$ defined as

$$S = [s_{ij}]_{h\times n}, s_{ij} = -\log\Big(\Lambda(T^{i(hi)}, \mathbf{z}_t^j)\Big), \mathbf{z}_t^j \in \mathbb{Z}_t, \quad (6)$$

where the affinity $\Lambda(T^{i(hi)}, \mathbf{z}_t^j)$ is computed by Eq. (3). We then determine tracklet-detection pairs using the Hungarian algorithm [34] such that the total affinity in $S_{h\times n}$ is maximized. When the association cost of a pair is less than a predefined threshold, $-\log(\theta)$, $\mathbf{z}_t^j$ is associated with $T^{i(hi)}$. For the tracklet $T^{i(hi)}$ associated with detection $\mathbf{z}_t^j$, the following procedure is performed:

(i) The position and the velocity of a tracklet are updated with the associated $\mathbf{z}_t^j$. The size of the object is also updated by averaging the sizes of associated detections of recent past frames.
(ii) $conf(T_i)$ is updated using $\mathbf{z}_t^j$ by Eq. (2).

Here, it is possible to skip this HC-association and try to solve the problem via only the LC-association that will be described in the next section. However, in this case, much more computation is required and the performance is also degraded. This is because the HC-association greatly reduces the ambiguity in the LC-association as well as the complexity. This is proven in Section 6.

3.5 Low Confidence (LC-) Association of Tracklets

In the LC-association, tracklets with low confidence, which are more likely to be fragmented due to many reasons, are globally associated with other tracklets and detections. Suppose that there exist h and l tracklets with high and low confidence, respectively. Since association events are mutually exclusive, we only consider n detections, $\mathbb{Y}_t = \{\mathbf{y}_t^j\}_{j=1}^n \subset \mathbb{Z}_t$ in associating $T^{i(lo)}$, where $\mathbb{Y}_t$ is a set of detections not associated with any $T^{i(hi)}$ in the HC-association. The following association events are then considered:

- Event A: $T^{i(lo)}$ is associated with $T^{j(hi)}$,
- Event B: $T^{i(lo)}$ is terminated,
- Event C: $T^{i(lo)}$ is associated with $\mathbf{y}_t^j$.

We define the cost matrix for all events as follows:

$$G_{(l+n)\times(h+l)} = \begin{bmatrix} A_{l\times h} & B_{l\times l} \\ -\log(\theta)_{n\times h} & C_{n\times l} \end{bmatrix}, \quad (7)$$

Here, $A = [a_{ij}]$ represents the event A, where $a_{ij} = -\log(\Lambda(T^{i(lo)}, T^{j(hi)}))$ is the association cost computed by the affinity between them using Eq. (3). $B = \mathrm{diag}[b_1, \ldots, b_l]$ models the event B, where $b_i = -\log(1 - conf(T^{i(lo)}))$ is the cost to terminate $T^{i(lo)}$, and $C = [c_{ij}]$ represents the event C, where $c_{ij} = -\log(\Lambda(T^{i(lo)}, \mathbf{y}_t^j))$ is the association cost computed by Eq. (3). The same threshold θ used in the HC-association is also employed to select reliable association pairs having high affinity scores. Once the cost matrix is computed, the optimal association pairs, which minimize the LC-association cost in G, are determined using the Hungarian algorithm [34], and the tracklets and their confidence values are updated with the results.

When association pairs exist between $T^{i(lo)}$ and $T^{j(hi)}$ or $T^{i(lo)}$ and $\mathbf{y}_t^j$, in order to combine their appearance and motion models, we horizontally concatenate their output features extracted from the network, and generate the forward motion from the head of $T^{i(lo)}$ to the tail of $T^{j(hi)}$ or the position of $\mathbf{y}_t^j$.

4 Discriminative Deep Appearance Learning

As mentioned, the appearance modeling is very important in both the high and low confidence association for associating tracklets and detections of the same object while distinguishing different objects. To this end, we propose to learn discriminative deep appearance models in consideration of two main issues in multi-object tracking: (1) online learning to update appearance models according to ongoing tracking results, and (2) online training sample collection for discriminating appearances of multiple objects.

Unfortunately, most previous tracking methods with online appearance learning focus on only one of theses issues. For instance, some methods [9], [15], [28], [35] devise online learning methods for adapting learned appearances, but their sample collection strategies aim at distinguishing an object from the background rather than other objects. On the other hand, some methods [4], [5], [7], [29] collect training samples for discriminating different objects, but the discriminative appearance models are learned in a batch manner: once training samples are collected from the tracklets after low-level association, the models are simultaneously learned with all collected samples.

Unlike these previous works, the proposed appearance learning method is designed in consideration of two issues together to learn discriminative deep appearance models using online transfer learning. This allows us to distinguish each object using learned deep representations and also incrementally update the learned deep representations with online tracking results. By exploiting the proposed appearance learning, tracklet association can be successfully performed even under occlusion.

In the proposed learning method, from a large training dataset, discriminative deep representations (*or* dissimilarity metric) is built by minimizing the output feature distance of the same object pairs while maximizing it of the different object pairs. Here, it is worthy of notice that we also use the OTL method for adapting the learned deep network. The main reason of using OTL is that we can increase the discriminability of the appearance model for distinguishing objects in the specific tracking sequence. Furthermore, the benefit of our OTL lies in reducing the online learning complexity by retraining the high-level representation only while maintaining the low-level representation learned in advance. These allow us to accurately identify objects even under significant pose and appearance changes and long-term occlusion.

4.1 Training Sample Collection

For training a deep model, we use the CUHK02 [36] dataset containing 7,262 image patches for 1,816 persons captured

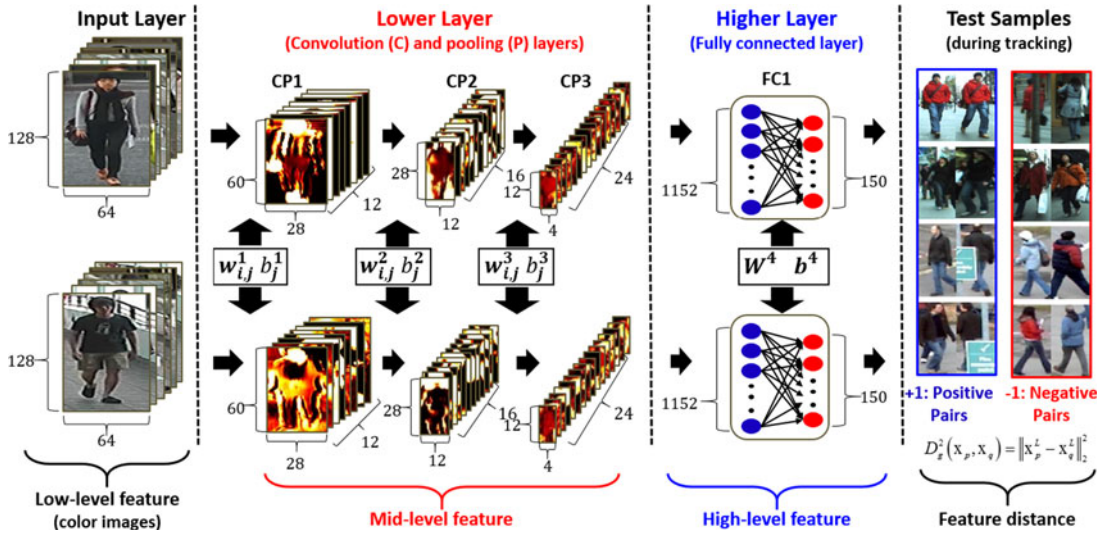


Fig. 4. Proposed discriminative deep appearance model: Our model is learned by minimizing the distances between high-level features of positive pairs while maximizing the distances between the high-level features of negative pairs. Here, $\mathbf{W}^l, \mathbf{b}^l, \mathbf{w}_{ij}^l$ and b_j^l are learned and shared parameters. For the image pairs, the feature maps extracted from each layer with the parameters are depicted in each row.

from 10 camera views. From the dataset, we collect 10,928 positive sample pairs. On the other hand, we can obtain a huge number of negative pairs from all possible combinations of persons. However, to avoid the data imbalance problem, we set the ratio of positive to negative pairs to 1:2, which gives the best performance shown in Fig. 9c, and randomly select negative pairs.

Given image patches (i.e., samples) with labels, we can collect positive (i.e., belonging to the same object) and negative sample pairs (i.e., belonging to different objects). Then, we construct a set $\mathbb{B} = \{(\mathbf{x}_p, \mathbf{x}_q, y_{pq})\}$ consisting of sample pairs $(\mathbf{x}_p, \mathbf{x}_q)$ and their labels y_{pq}. Here, if $\mathbf{x}_p$ and $\mathbf{x}_q$ are a positive pair, then $y_{pq} = 1$. Otherwise, $y_{pq} = -1$.

In our implementation, the patch is resized to $128 \times 64 \times 3$, and the dimension of the feature is 24,576. Since the feature dimension is very high, we project the high-dimensional feature onto a low-dimensional subspace, and compare the distance of the projected features using discriminative deep learning.

4.2 Discriminative Deep Appearance Model

As in Fig. 4, our deep network consists of 8 layers: 3 convolutional (C), 3 pooling (P), and 2 fully-connected (FC) layers. For reducing learning complexity, the lower layers CP(1-3) are designed by stacking several convolutional and max-pooling layers, whereas the higher layers (FC1) are based on the fully-connected neural layers for learning dependencies between the extracted mid-level features. As a result, we can extract meaningful features from the lower layers and extend the space of feature representations by combining the mid-level features with the higher layers.[1]

4.2.1 Feature Extraction

We denote a feature map i of size $h^l \times w^l$ at layer l as $\mathbf{x}_i^l$, where $l = 1, 2, \ldots, L$, and h^l and w^l are the height and the width of the map $\mathbf{x}_i^l$, respectively.

1. For notational convenience, we use $*$, $\bullet$ and $\|\|_F^2$ to denote the convolution, element-wise product, and Frobenius matrix norm. $\tilde{\mathbf{A}}$ is the horizontally and vertically flipped matrix of $\mathbf{A}$.

Input Layer. We take the image of an object of resolution 128×64 as an input. The image has three feature maps from RGB channels, and each map $\mathbf{x}_i^0$ is normalized by subtracting its mean and dividing by its variance. In this paper, this simple normalization yields better performance than other normalization methods, such as local contrast normalization [37] and ZCA whitening [38] as shown in Fig. 9d.

Convolution and Pooling (CP) Layer. To extract mid-level features, an input feature map $\mathbf{x}_i^{l-1}$ is convolved with different kernels $\mathbf{w}_{i,j}^l$ of size $m^l \times m^l$ at layer l, and then passed through the nonlinear activation function $f(\cdot)$ to obtain output feature maps $\mathbf{x}_j^l$ of size $(h^{l-1} - m^l + 1) \times (w^{l-1} - m^l + 1)$ as

$$\mathbf{x}_j^l = f\Big(\textstyle\sum_{i=1}^{k^l} \mathbf{x}_i^{l-1} * \mathbf{w}_{ij}^l + b_j^l\Big), \tag{8}$$

where b_j^l is a bias factor, and k^l is the number of kernels used at layer l. We use a hyperbolic tangent function as $f(\cdot)$ for scaling the linear output because it improves convergence [39] during training for normalized data with a zero mean and unit variance.

To reduce the complexity for training the network and to achieve spatial and configural invariance, we then generate the down-sampled feature map of $\mathbf{x}_j^l$ using max-pooling with a scaling factor r. We select the maximum response over each $r \times r$ subregion of the feature map. As a result, the output feature map becomes r-times smaller along each spatial dimension.

Fully-Connected (FC) Layer. In order to learn the dependencies of the mid-level features, we concatenate all the feature maps of the previous layer CP3 as $\mathbf{o}^{l-1} = \Big[\mathbf{x}_{j=1}^{l-1}, \ldots, \mathbf{x}_{k^{l-1}}^{l-1}\Big]$, and fully connect them to the neurons of the output layer. The final outputs are also re-scaled by the sigmoid activation function $\sigma(\cdot)$ as

$$\mathbf{x}^l = \sigma\big(\mathbf{W}^l \mathbf{o}^{l-1} + \mathbf{b}^l\big), l = L. \tag{9}$$

4.2.2 Discriminative Deep Representation Learning

For learning the discriminative deep appearance model, we define an energy function with high-level feature distances

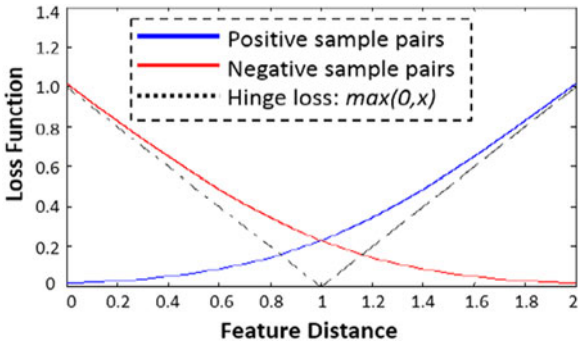


Fig. 5. A loss function using the pairwise constraint. The loss decreases when the feature distance of a positive pair is decreased but the distance of a negative pair is increased.

of sample pairs, and derive the gradient of the energy function with respect to the parameters $\left\{\mathbf{W}^l, \mathbf{w}_{ij}^l, \mathbf{b}^l, b_j^l\right\}$ of CP(1-3) and FC1 layers.

Given image pairs of objects, $\mathbf{x}_p$ and $\mathbf{x}_q$, we can extract their high-level features $\mathbf{x}_p^L$ and $\mathbf{x}_q^L$ by passing them to the network. We then define the similarity between $\mathbf{x}_p$ and $\mathbf{x}_q$ as the distance of their output features using the square of the L2 norm as

$$D_g^2(\mathbf{x}_p, \mathbf{x}_q) = \left\|\mathbf{x}_p^L - \mathbf{x}_q^L\right\|_2^2. \tag{10}$$

Since our goal is to build a deep network that can discriminate the appearances of object image pairs, we impose $D_g^2(\mathbf{x}_p, \mathbf{x}_q) \leq \tau$ with a threshold τ for positive pairs whereas $D_g^2(\mathbf{x}_p, \mathbf{x}_q) > \tau$ for negative pairs, when training the network. Based on this pairwise constraint, we define an energy function as in [40] and learn parameters by minimizing the function defined as

$$\begin{aligned}\underset{\{\mathbf{W}^l, \mathbf{b}^l\}}{\operatorname{argmin}} \quad & E = E_1 + E_2 = \frac{1}{2}\sum_{p,q} \psi\left(y_{pq}\left(D_g^2(\mathbf{x}_p, \mathbf{x}_q) - \tau\right)\right) \\ & + \frac{\lambda}{2}\sum_{l=1}^{L}\left(\left\|\mathbf{W}^l\right\|_F^2 + \left\|\mathbf{b}^l\right\|_2^2\right), l = 1, \ldots, L,\end{aligned} \tag{11}$$

where $\mathbf{W}^l = \{\mathbf{w}_{ij}^l\}$ and $\mathbf{b}^l = \{b_j^l\}$ for $l = 1, 2, 3$. $\psi(e) = \frac{1}{\beta}\log(1 + \exp(\beta e))$ is the generalized logistic loss function, which is the smooth approximation of the hinge loss function. If $\mathbf{x}_p$ and $\mathbf{x}_q$ are a positive pair, $y_{pq} = 1$. Otherwise, $y_{pq} = -1$. The first term therefore penalizes the pair that does not follow the pairwise constraint. On the other hand, the second term penalizes the large weighs and biases with the parameter λ for preventing overfitting to the training data. Fig. 5 shows the energy change for the distance between an object pair.

Using the stochastic gradient descent, we learn the parameters of all layers with a learning rate μ as

$$\begin{aligned}\mathbf{W}^l &= \mathbf{W}^l - \mu \cdot \frac{\partial E}{\partial \mathbf{W}^l}, \mathbf{b}^l = \mathbf{b}^l - \mu \cdot \frac{\partial E}{\partial \mathbf{b}^l}, \\ \mathbf{w}_{ij}^l &= \mathbf{w}_{ij}^l - \mu \cdot \frac{\partial E}{\partial \mathbf{w}_{ij}^l}, b_j^l = b_j^l - \mu \cdot \frac{\partial E}{\partial b_j^l}.\end{aligned} \tag{12}$$

Then, the learning problem is to find the gradients of the energy function with respect to the network parameters. We can efficiently derive the gradients by propagating the sensitives (δ^l, γ^l), the derivatives of the error (*or* energy) with respect to the total input map ($\mathbf{z}_p^l$, $\mathbf{z}_q^l$), from higher to lower layers. The complete derivation of the gradients can be found in the next Section 4.2.3.

4.2.3 Gradient Derivation for Deep Appearance Learning

We provide the derivation of the gradients of each layer used for the deep appearance learning Eq. (12). The back-propagated errors through layers of a network can be considered as sensitivities, which are the derivatives of the error (*or* energy) with respect to the total input map, $\mathbf{z}_p^l$ and $\mathbf{z}_q^l$. Let then define the sensitivities for $\mathbf{z}_p^l$ and $\mathbf{z}_q^l$ as δ^l and γ^l. By propagating these sensitivities from higher to lower layers, we can efficiently derive the gradients of the energy with respect to parameters of each layer.

Fully-Connected (FC1) Layer. Let define the error of the distance of each positive and negative sample pair as

$$e = y_{pq}\left(D_g^2(\mathbf{x}_p, \mathbf{x}_q) - \tau\right). \tag{13}$$

As mentioned before, we first derive the sensitivities of the output layer ($l = L$) using $\frac{\partial\|\mathbf{x}\|_2}{\partial \mathbf{x}} = \frac{\mathbf{x}}{\|\mathbf{x}\|_2}$ as

$$\begin{aligned}\delta^l &= \frac{\partial E_1}{\partial \mathbf{z}_p^l} = \frac{\partial E_1}{\partial e}\frac{\partial e}{\partial D_g}\frac{\partial D_g}{\partial \mathbf{x}_p^l}\frac{\partial \mathbf{x}_p^l}{\partial \mathbf{z}_p^l}, \mathbf{x}_p^l = \sigma(\mathbf{z}_p^l), \\ \gamma^l &= \frac{\partial E_1}{\partial \mathbf{z}_q^l} = \frac{\partial E_1}{\partial e}\frac{\partial e}{\partial D_g}\frac{\partial D_g}{\partial \mathbf{x}_q^l}\frac{\partial \mathbf{x}_q^l}{\partial \mathbf{z}_q^l}, \mathbf{x}_q^l = \sigma(\mathbf{z}_q^l), \\ \frac{\partial E_1}{\partial e} &= \frac{1}{2}\psi'(e), \frac{\partial e}{\partial D_g} = y_{pq}, \frac{\partial D_g}{\partial \mathbf{x}_p^l} = 2\left(\mathbf{x}_p^l - \mathbf{x}_q^l\right), \\ \frac{\partial D_g}{\partial \mathbf{x}_q^l} &= 2\left(\mathbf{x}_p^l - \mathbf{x}_q^l\right), \frac{\partial \mathbf{x}_p^l}{\mathbf{z}_p^l} = \sigma'\left(\mathbf{z}_p^l\right), \frac{\partial \mathbf{x}_q^l}{\mathbf{z}_q^l} = \sigma'\left(\mathbf{z}_q^l\right),\end{aligned} \tag{14}$$

where $\mathbf{z}_p^l = \mathbf{W}^l\mathbf{o}_p^{l-1} + \mathbf{b}^l$ and $\mathbf{z}_q^l = \mathbf{W}^l\mathbf{o}_q^{l-1} + \mathbf{b}^l$ are total input maps or weighted sums of the outputs of the previous layer. By substituting them, we can derive the following δ^l and γ^l as

$$\begin{aligned}\delta^l &= \psi'(e)y_{pq}\left(\mathbf{x}_p^l - \mathbf{x}_q^l\right) \bullet \sigma'\left(\mathbf{z}_p^l\right), \\ \gamma^l &= \psi'(e)y_{pq}\left(\mathbf{x}_p^l - \mathbf{x}_q^l\right) \bullet \sigma'\left(\mathbf{z}_q^l\right).\end{aligned} \tag{15}$$

With the sensitivities δ^l and γ^l, the gradients of the energy function with respect to $\left\{\mathbf{W}^l, \mathbf{b}^l\right\}$ can be represented as

$$\begin{aligned}\frac{\partial E}{\partial \mathbf{W}^l} &= \frac{\partial E_1}{\partial e}\frac{\partial e}{\partial D_g}\left(\frac{\partial D_g}{\partial \mathbf{x}_p^l}\frac{\partial \mathbf{x}_p^l}{\partial \mathbf{z}_p^l}\frac{\partial \mathbf{z}_p^l}{\partial \mathbf{W}^l} - \frac{\partial D_g}{\partial \mathbf{x}_q^l}\frac{\partial \mathbf{x}_q^l}{\partial \mathbf{z}_q^l}\frac{\partial \mathbf{z}_q^l}{\partial \mathbf{W}^l}\right) + \frac{\partial E_2}{\partial \mathbf{W}^l} \\ &= \sum_{p,q}\left(\delta^l(\mathbf{o}_p^{l-1})^{\mathrm{T}} - \gamma^l(\mathbf{o}_q^{l-1})^{\mathrm{T}}\right) + \lambda\mathbf{W}^l, \\ \frac{\partial E}{\partial \mathbf{b}^l} &= 2\frac{\partial E_1}{\partial e}\frac{\partial e}{\partial D_g}\left(\frac{\partial D_g}{\partial \mathbf{x}_p^l}\frac{\partial \mathbf{x}_p^l}{\partial \mathbf{z}_p^l}\frac{\partial \mathbf{z}_p^l}{\partial \mathbf{b}^l} - \frac{\partial D_g}{\partial \mathbf{x}_q^l}\frac{\partial \mathbf{x}_q^l}{\partial \mathbf{z}_q^l}\frac{\partial \mathbf{z}_q^l}{\partial \mathbf{b}^l}\right) + \frac{\partial E_2}{\partial \mathbf{b}^l} \\ &= \sum_{p,q}\left(\delta^l - \gamma^l\right) + \lambda\mathbf{b}^l.\end{aligned} \tag{16}$$

Pooling Layers. In the similar manner, we can derive the sensitivities δ_j^l and γ_j^l of each map j of the pooling layer as

$$\delta_i^l = \frac{\partial E_1}{\partial \mathbf{z}_{p,i}^l} = \frac{\partial E_1}{\partial \mathbf{z}_{p,i}^{l+1}} \frac{\partial \mathbf{z}_{p,i}^{l+1}}{\partial \mathbf{x}_{p,j}^l} \frac{\partial \mathbf{x}_{p,j}^l}{\partial \mathbf{z}_{p,j}^l}, \gamma_i^l = \frac{\partial E_1}{\partial \mathbf{z}_{q,i}^l} = \frac{\partial E_1}{\partial \mathbf{z}_{q,i}^{l+1}} \frac{\partial \mathbf{z}_{q,i}^{l+1}}{\partial \mathbf{x}_{q,j}^l} \frac{\partial \mathbf{x}_{q,j}^l}{\partial \mathbf{z}_{q,j}^l},$$
$$\frac{\partial E_1}{\partial \mathbf{z}_{p,j}^{l+1}} = \delta_j^{l+1}, \frac{\partial \mathbf{z}_{p,i}^{l+1}}{\partial \mathbf{x}_{p,j}^l} = \sum_{j=1}^{k^{l+1}} \mathbf{w}_{ij}^{l+1}, \frac{\partial \mathbf{x}_{p,j}^l}{\partial \mathbf{z}_{p,j}^l} = 1,$$
$$\frac{\partial E_1}{\partial \mathbf{z}_{q,j}^{l+1}} = \gamma_j^{l+1}, \frac{\partial \mathbf{z}_{q,i}^{l+1}}{\partial \mathbf{x}_{q,j}^l} = \sum_{j=1}^{k^{l+1}} \mathbf{w}_{ij}^{l+1}, \frac{\partial \mathbf{x}_{q,j}^l}{\partial \mathbf{z}_{q,j}^l} = 1, \tag{17}$$

where $\mathbf{z}_{p,i}^{l+1} = \sum_{j=1}^{k^{l+1}} \mathbf{x}_{p,j}^l * \mathbf{w}_{ij}^{l+1} + b_j^{l+1}$ and $\mathbf{z}_{q,i}^{l+1} = \sum_{j=1}^{k^{l+1}} \mathbf{x}_{q,j}^l * \mathbf{w}_{ij}^{l+1} + b_j^{l+1}$ are total input maps at the next convolution layer. Note that $\mathbf{x}_{p,j}^l = \mathbf{z}_{p,j}^l$ and $\mathbf{x}_{q,j}^l = \mathbf{z}_{q,j}^l$ at the pooling layer.

We can derive the sensitivities of the feature map i by summing the convolved sensitivities of the next convolution layer as

$$\delta_i^l = \sum_{j=1}^{k^{l+1}} \left(\delta_j^{l+1}\right) * \tilde{\mathbf{w}}_{ij}^{l+1}, \gamma_i^l = \sum_{j=1}^{k^{l+1}} \left(\gamma_j^{l+1}\right) * \tilde{\mathbf{w}}_{ij}^{l+1}. \tag{18}$$

Convolution Layer. For CP(1-3) layers, the gradient of the function with respect to $\{\mathbf{w}_{ij}^l, b_j^l\}$ can then be evaluated with backpropagated sensitivities of the pooling layers. Note that the number of kernels of the convolution layer is exactly the same with that of the pooling layer. The sensitivities of the convolution layers are then derived as

$$\delta_j^l = \frac{\partial E_1}{\partial \mathbf{z}_{p,j}^l} = \frac{\partial E_1}{\partial \mathbf{z}_{p,j}^{l+1}} \frac{\partial \mathbf{z}_{p,j}^{l+1}}{\partial \mathbf{x}_{p,j}^l} \frac{\partial \mathbf{x}_{p,j}^l}{\partial \mathbf{z}_{p,j}^l}, \gamma_j^l = \frac{\partial E_1}{\partial \mathbf{z}_{q,j}^l} = \frac{\partial E_1}{\partial \mathbf{z}_{q,j}^{l+1}} \frac{\partial \mathbf{z}_{q,j}^{l+1}}{\partial \mathbf{x}_{q,j}^l} \frac{\partial \mathbf{x}_{q,j}^l}{\partial \mathbf{z}_{q,j}^l},$$
$$\frac{\partial E_1}{\partial \mathbf{z}_{p,j}^{l+1}} = \delta_j^{l+1}, \frac{\partial \mathbf{z}_{p,i}^{l+1}}{\partial \mathbf{x}_{p,j}^l} \Rightarrow \frac{\partial (up(\mathbf{z}_{p,i}^{l+1}))}{\partial \mathbf{x}_{p,j}^l} = 1, \frac{\partial \mathbf{x}_{p,j}^l}{\partial \mathbf{z}_{p,j}^l} = f'\left(\mathbf{z}_{p,j}^l\right),$$
$$\frac{\partial E_1}{\partial \mathbf{z}_{q,j}^{l+1}} = \gamma_j^{l+1}, \frac{\partial \mathbf{z}_{q,i}^{l+1}}{\partial \mathbf{x}_{q,j}^l} \Rightarrow \frac{\partial (up(\mathbf{z}_{q,i}^{l+1}))}{\partial \mathbf{x}_{q,j}^l} = 1, \frac{\partial \mathbf{x}_{q,j}^l}{\partial \mathbf{z}_{q,j}^l} = f'\left(\mathbf{z}_{q,j}^l\right), \tag{19}$$

where $\mathbf{z}_{p,j}^l = \sum_{i=1}^{k^l} \mathbf{x}_{p,i}^{l-1} * \mathbf{w}_{ij}^l + b_j^l$ and $\mathbf{z}_{q,j}^l = \sum_{i=1}^{k^l} \mathbf{x}_{q,i}^{l-1} * \mathbf{w}_{ij}^l + b_j^l$. We perform upsampling $up(\cdot)$ with the factor r for making $\mathbf{z}_{p,j}^{l+1}(\mathbf{z}_{q,j}^{l+1})$ of the pooling layer be the same size as $\mathbf{x}_{p,j}^l(\mathbf{x}_{q,j}^l)$ of the convolution layer. The up-sampled $up(\mathbf{z}_{p,i}^{l+1})$ and $up(\mathbf{z}_{q,i}^{l+1})$ are exactly the same with $\mathbf{x}_{p,i}^l$ and $\mathbf{x}_{q,i}^l$.

In convolution layers, the sensitivities δ_j^l and γ_j^l become

$$\delta_j^l = up\left(\delta_j^{l+1}\right) \bullet f'\left(\mathbf{z}_{p,j}^l\right), \gamma_j^l = up\left(\gamma_j^{l+1}\right) \bullet f'\left(\mathbf{z}_{q,j}^l\right). \tag{20}$$

From the above sensitivities, the gradients of the energy function with respect to $\left\{\mathbf{w}_{ij}^l, \mathbf{b}_j^l\right\}$ are

$$\begin{aligned} \frac{\partial E}{\partial \mathbf{w}_{ij}^l} &= \left(\frac{\partial E_1}{\partial \mathbf{z}_{p,j}^l} \frac{\mathbf{z}_{p,j}^l}{\mathbf{w}_{ij}^l} - \frac{\partial E_1}{\partial \mathbf{z}_{q,j}^l} \frac{\mathbf{z}_{q,j}^l}{\mathbf{w}_{ij}^l}\right) + \frac{\partial E_2}{\partial \mathbf{w}_{ij}^l}, \\ &= \sum_{p,q} \left(\delta_j^l * (\tilde{\mathbf{x}}_{p,i}^{l-1})^{\mathrm{T}} - \gamma_j^l * (\tilde{\mathbf{x}}_{q,i}^{l-1})^{\mathrm{T}}\right) + \lambda \mathbf{w}_{ij}^l, \\ \frac{\partial E}{\partial b_j^l} &= \left(\frac{\partial E_1}{\partial \mathbf{z}_{p,j}^l} \frac{\mathbf{z}_{p,j}^l}{b_j^l} - \frac{\partial E_1}{\partial \mathbf{z}_{q,j}^l} \frac{\mathbf{z}_{q,j}^l}{b_j^l}\right) + \frac{\partial E_2}{\partial b_j^l}, \\ &= \sum_{u,v} \left(\delta_j^l - \gamma_j^l\right) + \lambda b_j^l. \end{aligned} \tag{21}$$

Here, the first term of the bias gradient evaluation represents the sum of differences over all elements in δ_j^l and γ_j^l.

(a) CUHK (b) ETHMS and PETS

Fig. 6. (a) Training samples. (b) Collected samples during tracking.

4.2.4 Online Transfer Learning (OTL)

Once the discriminative deep appearance model is trained, we can use it for MOT directly. However, as shown in Fig. 6, the data statistics are often different due to the differences of object sizes and orientations, occlusion patterns, and illuminations. In addition, the object appearances frequently vary during tracking shown in Fig. 6b-bottom. To handle this problem, we perform online transfer learning, which updates the parameters of the pre-learned deep model with incoming tracking results.

For OTL, inspired by [41], we use the learned mid-level feature representations from CP1 and CP2 layers of the pre-trained deep network in Fig. 4. However, different from [41] that considers transfer learning only, we consider the transfer and online learning together for onilne MOT. Usually, much more training samples are required to change the structure of a pre-trained network than to re-train parameters of the network only. For this reason, we re-train the network parameters only while maintaining its structure for the efficiency of online learning. Particularly, in our case, we only re-train the high-level representations of CP3 and FC1 layers since it provides the best results. Here, we reset the parameters $\{\mathbf{W}^l, \mathbf{w}_{ij}^l, \mathbf{b}^l, b_j^l\}$ of both layers before OTL. From extensive evaluations, we found that the parameter initialization before transfer learning is very essential for achieving better performance. The performance evaluation for different strategies of transfer learning is provided in Fig. 10.

In the proposed framework, the online transfer learning is automatically performed during tracking based on the Algorithm 1 when the average dissimilarity score for tracklets and detections falls below to 0.5. This score is the average of the elements of the appearance affinity matrix excluding appearance affinities of association pairs and evaluated by Eq. (4). At each frame, training samples are collected from the image patches with the positions and sizes of detections. Since the detections are sometimes inaccurate and coming from scene clutter, we use only detections associated with tracklets for training. In particular, we choose image patches from the detections associated with the tracklets with high confidence since image patches from the tracklets with low confidence are more likely to be polluted by occlusion as shown in Fig. 7. Given N image patches with labels, we can collect $N!/2!(N-2)!$ positive and negative sample pairs. For reducing the tracking complexity, however, we randomly select 200 positive and 400 negative sample pairs for each update.

The main benefit of our online transfer learning is that we can greatly improve the MOT performance because the deep appearance model can be adapted to be more suitable for tracklet association of the specific tracking sequence as proved in Table 1. We can further resolve complexity and overfitting problems of online deep learning since we only

Fig. 7. Training samples from the trackets with high confidence (red) and low confidence (blue).

re-train the parameters of the higher layers while keeping the parameters of other layers.

Algorithm 1. Deep Appearance Model Learning

Input: A training set containing sample pairs and their labels $\mathbb{B} = \{\mathbf{x}_p, \mathbf{x}_q, y_{pq}\}$
Output: Updated parameters: $\left\{\mathbf{W}^l, \mathbf{b}^l, \mathbf{w}_{ij}^l, b_j^l\right\}$.
// Initialize weights and biases
Initialize parameters $\left\{\mathbf{W}^l, \mathbf{b}^l, \mathbf{w}_{ij}^l, b_j^l\right\}$ in Section (4.2.5)
For $i = 1$ to E(number of Epoch) **do**
  Randomly select sample pairs $\{\mathbf{x}_p, \mathbf{x}_q, y_{pq}\}$ in $\mathbb{B}$.
  // Forward propagation
  For $l = 1$ to L **do**
    Extract pairwise feature maps $\left\{\mathbf{x}_{p,j}^l, \mathbf{x}_{q,j}^l, \mathbf{x}_p^l, \mathbf{x}_q^l\right\}$ using Eq. (8) or Eq. (9)
  end
  // Back propagation
  For $l = L$ to 1 **do**
    Compute gradients $\{\frac{\partial E}{\partial \mathbf{W}^l}, \frac{\partial E}{\partial \mathbf{b}^l}, \frac{\partial E}{\partial \mathbf{w}_{ij}^l}, \frac{\partial E}{\partial b_j^l}\}$ for $\left\{\mathbf{x}_{p,j}^l, \mathbf{x}_{q,j}^l, \mathbf{x}_p^l, \mathbf{x}_q^l\right\}$ by Eq. (16) and (21)
  end
  // Update parameters
  For $l = 1$ to L **do**
    Update parameters $\{\mathbf{W}^l, \mathbf{b}^l, \mathbf{w}_{ij}^l, b_j^l\}$ with $\{\frac{\partial E}{\partial \mathbf{W}^l}, \frac{\partial E}{\partial \mathbf{b}^l}, \frac{\partial E}{\partial \mathbf{w}_{ij}^l}, \frac{\partial E}{\partial b_j^l}\}$ using Eq. (12)
  end
end

4.2.5 Details of Discriminative Deep Model Learning

When learning the model, we resize and normalize the images as discussed in Section 4.2.1. For CP(1-3) layers, we use 12, 16, and 24 kernels of the sizes, 9×9, 5×5, and 5×5, respectively. We use the same scaling factor $r = 2$ for all the pooling layers. From the CP3 layer, we extract the 1152-dimensional feature by concatenating all elements of the 24 maps of the size 12×4. We then fully connect it with 150 neurons in the FC1 layer.

For parameter initialization, we use the simple way described in [42]. The biases b_j^l ($\mathbf{b}^l$) are set to 0 ($\mathbf{0}$) and the weights $\{\mathbf{w}_{ij}^l, \mathbf{W}^l\}$ at each layer are determined as $\{\mathbf{w}_{ij}^l, \mathbf{W}^l\} \sim U\left[-\frac{\sqrt{6}}{\sqrt{\eta^{l-1}+\eta^l}}, \frac{\sqrt{6}}{\sqrt{\eta^{l-1}+\eta^l}}\right]$, where $U[-a, a]$ represents the uniform distribution in the range $[-a, a]$. η^l is the multiplication of the kernel size and the kernel number as $m^l \times m^l \times k^l$ for the CP layers. However, η^l is the number of neurons for the FC1 layer.

To solve Eq. (11), we fix β to 3 since optimizing it trivially reduces the energy without improving other parameters as in [40]. From the evaluations, we set the size of the mini-batch to 50, threshold $\tau = 1$, the regularization parameter $\lambda = 5 \times 10^{-4}$, and learning rate $\mu = 10^{-3}$ with the general rules of thumb in deep learning (all the parameters are fixed except for online learning evaluation in Section 6.1.)

5 Discussion

In this section, we discuss important issues, related to this paper.

Online and Real-Time Tracking.The online tracking systems should satisfy the following conditions:

- (c1) The output (i.e., tracking result) of the system depends only on previous and current inputs, but not future inputs, and is immediately available with each incoming frame.
- (c2) The output of the system is computed and fixed at each passing frame and it is not allowed to manipulate the result of any past frame under any circumstances.

The proposed framework is a frame-by-frame MOT framework and the output of each part shown in Fig. 3 relies only on the past and current results of other parts. Furthermore, the tracking results (i.e., box locations, sizes, and IDs) for each frame are immediately stored with each incoming frame (without any frame delay) and fix them for evaluation. Therefore, our framework meets all the conditions ((c1) and (c2)) for online tracking.

TABLE 1
Performance Comparison with Different Association and Appearance Learning Algorithms

Dataset	Method	MOTP ↑	MOTA ↑	GT	MT ↑	PT	ML ↓	IDS ↓	FG ↓	REC ↑	PRE ↑	FAF ↓
	A. Data association comparison (with the proposed appearance learning)											
	a1 - w/o LC-assoc.	72.55 %	62.53 %	417	53.24 %	38.61 %	8.15 %	1026	1209	76.44 %	84.70 %	1.88
	a2 - w/o HC-assoc.	72.83 %	63.01 %	417	58.27 %	34.77 %	6.95 %	612	1037	78.57 %	83.76 %	2.52
PETS	a3 - Greedy [10]	72.22 %	61.92 %	417	55.16 %	36.93 %	7.91 %	419	1041	77.29 %	83.72 %	2.27
ETHMS	a4 - with DA-TEP [14]	72.58 %	64.90 %	417	59.95 %	33.33 %	6.71 %	452	973	79.25 %	83.04 %	2.03
Town Centre	**B. Appearance learning comparison (with the proposed data association)**											
(Average)	b1 - w/o online learning	72.73 %	57.29 %	417	51.56 %	41.25 %	7.19 %	478	863	75.25 %	79.98 %	2.80
	b2 - with ILDA [12]	73.38 %	65.74 %	417	64.27 %	29.50 %	**6.24 %**	503	722	80.26 %	85.03 %	2.16
	b3 - w/o OTL	73.43 %	67.36 %	417	63.79 %	29.98 %	**6.24 %**	393	681	80.44 %	86.63 %	1.82
	p1 - with all	**74.31 %**	**69.85 %**	417	**66.43 %**	26.86 %	6.71 %	**314**	**446**	**82.17 %**	**86.98 %**	**1.78**

*For Each Metric, **the Best Results are Marked in Red Color and Bold.** (Refer to the supplementary material, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPAMI.2017.2691769 for more results).*

(a) Bahnhof ♯462 (b) Bahnhof ♯484 (c) Bahnhof ♯491

Fig. 8. The ID correction by the LC-association: ID 48 is changed to ID 43 by occlusion. However, it is successfully recovered to its original ID after the LC-association.

Indeed, the past tracking results could be corrected with the LC-association and it would lead the improvement of tracking performance. For instance, the ID of a tracklet can be completely replaced with the ID of the associated tracklet, and some missing results can be filled by interpolating the states of the associated tracklets. However, in our implementation, we do not change the past results at all for online tracking. When a LC-association event occurs, we replace the ID of the newer tracklet with the ID of the older tracklet only for the current frame. This allows us to correct the corrupted ID by any reasons only at the current frame not past frames. For example, suppose that we have two trackets, an older tracklet T^i in the range of $[t-l, t-m]$ with the ID i from $(t-l)$ to $(t-m)$, and a newer tracklet T^j in the range of $[t-n, t]$ with the ID j from $(t-n)$ to t, where $l < m < n$. If a LC-association event occurs between T^i and T^j (T^i and T^j are linked) at the current frame t, then we change the ID of a newer tracklet T^j from j to i only for the current frame and, in this case, T^j has the ID j from $(t-n)$ to $(t-1)$ and the ID i at time t.

In fact, this ID correction by the LC-association increases the number of ID switches. However, in our experiments, using the LC-association improves the overall performance as proved in Table 1 since more accurate appearance and motion models can be generated by combining the models of associated tracklets as described in Section 3.5. Fig. 8 shows the ID correction by LC-association. The IDs shown in the figure are actually from our results to illustrate that we do not change the ID of the associated tracklet for the past frame. The object ID remains 43 at frame ♯484 in our results even though it is changed to 48 at frame ♯491.

Our system was implemented using MATLAB on a PC with a 3.07 GHZ CPU (single core) without parallel processing. The run-time relies on the total amount of detections and training samples used for online learning. For the less crowded ETH-Crossing (with 3,000 samples) and the crowded PETS-L2 (with 5000 samples) scenes, the run-times of our system are 0.56 and 1.81 (sec/frame), respectively. When excluding the online transfer learning, the run-times are 0.18 and 0.98 (sec/frame) for each sequence. We can also reduce the run-time by about 25 percent on average by performing the global association every 10 frames (the performance is degraded in return), which is performed every frame in our current implementation. The main bottleneck of our system occurs during the forward/backward propagation in the deep network. Even though the current implemented system does not work in real-time, we confirm that the speed can be greatly improved with GPU programming for the data propagation.

Transfer Learning and Fine-Tuning. Recently, it is common to use pre-trained networks (e.g., AlexNet [43], VGG16 [44], ResNet [45]) trained with very large datasets as a backbone for designing a deep network, and fine-tune them to be suitable for new tasks. Many works [1], [2], [41] show that the fine-tuning is often more effective rather than training an entire network from scratch.

Inspired by the effect of the transfer learning, we first train a discriminative appearance model with a large dataset, and fine-tune it for each tracking sequence. As shown in Fig. 10 and Table 1, the transfer learning improves the classification rate and overall MOT performance. The main reason of the performance improvement is that generic discriminative and data-specific discriminative features can be learned by pre-training and online transfer learning, and captured from lower and higher layers.

As described in Section 4.2.4, we only fine-tune the higher layers (CP3-FC1) during tracking. If a large number of data is available during tracking, fine-tuning a whole network would be the better idea. However, in many cases, the number of samples collected during tracking is relatively small, and the fine-tuning the whole parameters of the network from top to bottom with small samples increases the possibility of over-fitting, and also increases the complexity of the back-propagation. Therefore, we only fine-tune the parameters of the higher layers while keeping the representation of the lower layers.

6 Experiments

We first verify the effectiveness of the proposed deep learning by applying it for a person re-identification problem. Then, we analyze the performance improvement in MOT by the proposed data association framework and deep appearance learning in detail.

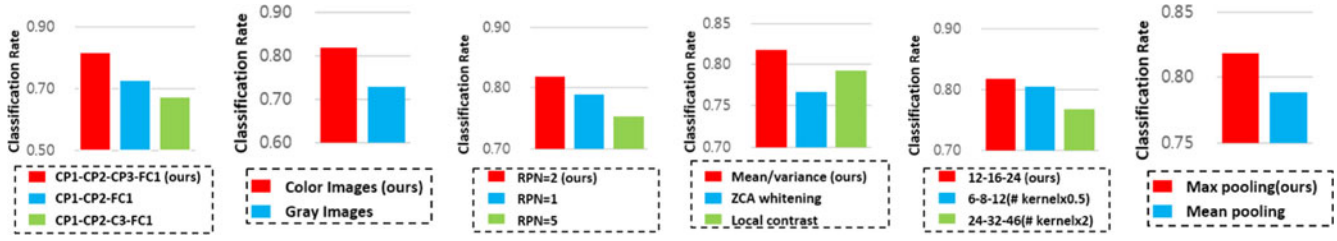
6.1 Evaluation of Deep Appearance Learning

Evaluation of Training Strategies. In order to determine the best architecture, we implement different versions of the network by changing its structure, parameters, and types of functions one-by-one, and compare their performance using the CUHK01/02 [36] person re-identification datasets. Based on the five-fold cross validation, we evaluate the classification rate by computing areas under the receiver operating characteristic curves for distances between object pairs and their labels.

In Fig. 9a, we see that removing the last convolution-pooling layer (CP3 or P3) considerably decreases the rate from 81.81 to 73.54 percent or 67.48 percent. In addition, using gray images instead of color images reduces the network discriminability as in Fig. 9b. Fig. 9c shows that learning with a skewed dataset (containing too many negative pairs) also degrades performance because the network is likely to be over-fitted to the negative class. Figs. 9d and 9e prove that employing other normalization methods and changing the kernel number within CP(1-3) layers do not help to increase the rate. In this case, using max pooling produces better results than mean pooling as can be seen in Fig. 9f.

We also evaluate the classification rate without using the learned distance metric. In this case, the similarity between extracted 150-dimensional output features is computed with the Bhattacharyya distance. However, the classification rate decreased by 9.03 percent, compared to when using the learned distance metric.

Evaluation of Transfer Learning. We also verify the benefits of transfer learning for MOT. To this end, we train the deep



(a) Network structure (b) Input maps (c) RPN (d) Normalization (e) Number of kernels (f) Pooling

Fig. 9. Performance comparison for various architectures: (a) Different network structures. (b) Color and gray image patches as input maps. (c) Different ratios of positive and negative sample pairs (RPN) for network training. (d) Different normalization methods, local contrast [37] and ZCA whitening [38]. (e) Different numbers of kernels of CP(1-3) layers. (f) Different pooling schemes for CP1-CP3 layers.

appearance model (shown in Fig. 4) with the CUHK01/02 datasets. We then re-train and evaluate it for the other person re-identification dataset, ETHZ [46], using transfer learning. The ETHZ dataset contains significant appearance variations due to occlusions, illumination variation, object motion, and camera viewpoint changes. Therefore, it is appropriate for preliminary feasibility study before applying our discriminative deep appearance model to MOT.

Fig. 10 compares the classification rates of the transfer-learned models for each sequence of the ETHZ dataset.[2] In all the cases, the transfer learning (Re-Learn (CP3-FC1)) enhanced the classification rates for our deep model, compared with the rates of Without Re-Learning and Re-Learn (All Layers). In addition, only re-training CP3-FC1 layers shows the higher rates than using other strategies. Also, we confirm that the parameter initialization is necessary for better transfer learning by comparing the rates of Re-Learn (CP3-FC1) and Re-Learn (CP3-FC1 w/o Init.).

Evaluation of Online Learning. Finally, we verify the effect of online learning in MOT. For this, we collect 3,000 training sample pairs from each sequence and divide them into 60 subsets. During 60 iterations, we gradually update the parameters of the CP3-FC1 layers of the pre-trained networks (using CUHK02 dataset) for each subset while keeping the parameters of other lower layers. For each iteration, we also evaluate their classification rates for 4,500 test sample pairs.

Fig. 11 shows the results. To show the effectiveness, we compare the performance of online learning with that of batch learning, which updates the network with all the samples at once. For batch learning, we decrease the learning rate to $\mu = 5 \times 10^{-4}$. Compared to classification rates of batch learning, the rates of online learning become almost similar after a few iterations. Fig. 12 demonstrates the extracted feature maps of CP1-CP3 layers for different datasets.

6.2 MOT System Implementation

We have implemented the proposed online tracking framework using MATLAB. Our source code is fully available at https://cvl.gist.ac.kr/project/cmot.html.

New Track Initialization. By applying the pre-trained detector at each frame, we can obtain object hypotheses with the detection responses. In order to find new object hypotheses, we search continuous and consistent detection responses having both overlapped areas and similar sizes within temporal sliding windows, which are not already associated with any tracklets in LC-and HC-association stages. In our implementation, we link detections when the ratio of an overlapped area over a union area of detections is more than 0.5. If more than two detections are overlapped in neighboring frames, we associate them with the maximum ratio based on the greedy algorithm [34]. For reducing false initiated tracks, we generate a new track with associated hypotheses only when the object hypotheses are associated in at least five subsequent frames.

Dataset: For performance evaluation, we use the following MOT datasets: CAVIAR [47], VS-PETS 2009 (PETS) [48], ETH Mobile scene (ETHMS) [46], and Town Centre [49]. Although the CAVIAR dataset contains 26 sequences, only

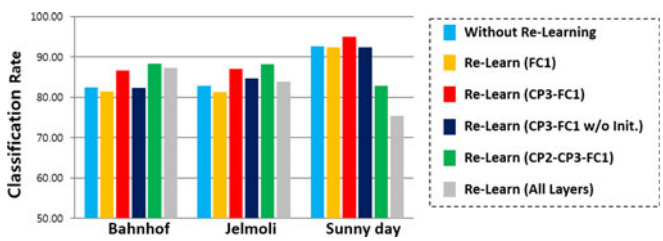


Fig. 10. Transfer learning evaluation on the ETHZ dataset.

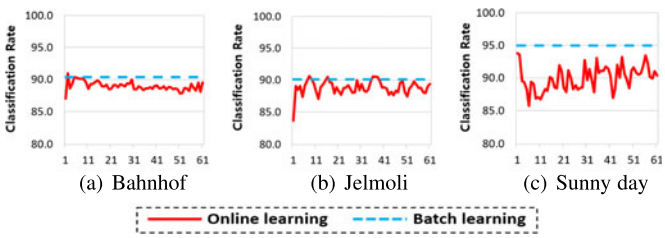
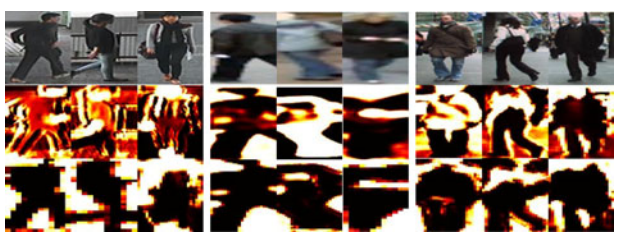


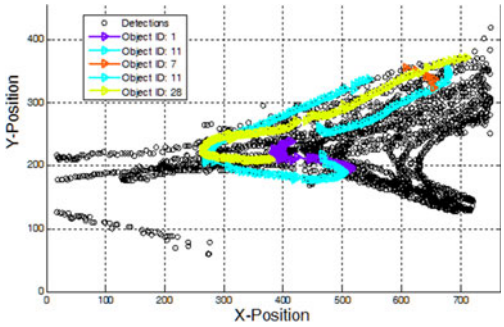
Fig. 11. Online learning evaluation on the ETHZ dataset.



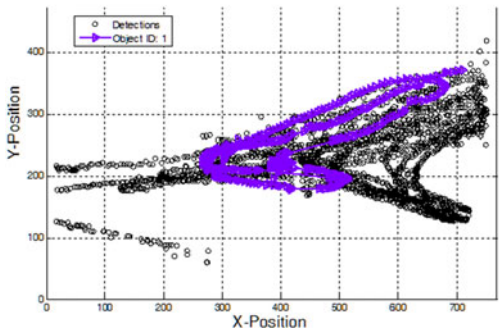
(a) CUHK (b) PETS (c) ETHZ

Fig. 12. From top to bottom, input images and mid-level features extracted from CP1 and CP2 layers of our appearance model are shown.

2. For example, Re-Learn (CP3-FC1) means that we only re-train the parameters of the CP3 and FC1 layers with ETHZ datasets while keeping the parameters of other layers learned from the CUHK datasets. In addition, Re-Learn (All Layers) means that we re-train the parameters of all the layers with the ETHZ datasets, and Without Re-Learning means that we did not re-train the parameters learned from the CUHK datasets.



(a) Fragmented trajectories using system (a1)



(b) A single long trajectory using system (p1)

Fig. 13. Detections (black circles) and estimated trajectories of a single object (color lines) for the PETS-L1 sequence over 520 frames.

20 sequences were used as in [4] to ensure fair comparison with other methods. In the PETS dataset, tracking sequences S2.L1 and S2.L2 were used. In the ETHMS dataset, the SUNNY DAY and BAHNHOF sequences of street scenes taken by a moving camera were selected.

Detection and Ground Truth. For a fair comparison, we use the same detections and the ground truth when evaluating performance of different methods. For VS-PETS 2009, ETHMS, CAVIAR and Town Centre datasets, we obtain detections and ground truth from [50], [4], [7], and [49], respectively.

(a) Tracking results without online learned appearance models (b1)

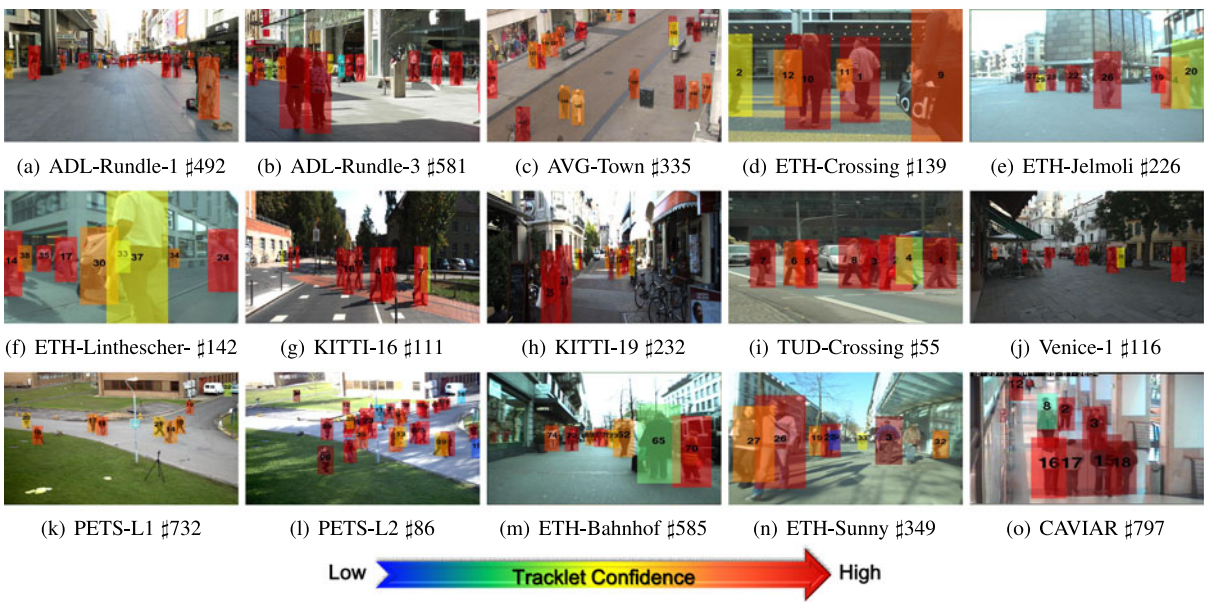
(b) Tracking results with the online learned deep models (p1)

Fig. 14. Tracking results: IDs (7, 21) are changed by occlusions in the top rows, but IDs (9, 18) correctly kept in the bottom rows.

System Parameters. All parameters have been determined experimentally, and remained unchanged for all datasets. From an extensive evaluation, we find that most parameters do not affect the overall system performance significantly. In the affinity model in Eq. (4), all parameters (i.e., positions, sizes and velocities) are automatically determined by tracking results except for O^F and O^B, which are set to $\mathrm{diag}[16^2 32^2]$. The normalized distance with parameter $c = 25$ is used for the appearance affinity. The same threshold $\theta = 0.4$ is used for the HC- and LC- association.

6.3 Performance Evaluation

Evaluation Metrics. We use the common CLEAR MOT [51] consisting of multiple metrics. The multiple object tracking precision (MOTP ↑) evaluates the intersection area over the union area of bounding boxes. The multiple object tracking accuracy (MOTA ↑) calculates the accuracy composed of false negatives, false positives, and identity switching (IDS ↓). In addition, the metrics used in [4], [7] are computed: the number of trajectories in the ground truth (GT), the ratio of mostly tracked trajectories (MT ↑), the ratio of mostly lost trajectories



(a) ADL-Rundle-1 ♯492 (b) ADL-Rundle-3 ♯581 (c) AVG-Town ♯335 (d) ETH-Crossing ♯139 (e) ETH-Jelmoli ♯226

(f) ETH-Linthescher- ♯142 (g) KITTI-16 ♯111 (h) KITTI-19 ♯232 (i) TUD-Crossing ♯55 (j) Venice-1 ♯116

(k) PETS-L1 ♯732 (l) PETS-L2 ♯86 (m) ETH-Bahnhof ♯585 (n) ETH-Sunny ♯349 (o) CAVIAR ♯797

Fig. 15. Tracking results for the PETS, ETHMS, CAVIAR and 2015 MOTChallenge datasets. At each frame, tracklets with different confidence are illustrated with different color boxes. The identities of tracked objects are marked in black. (Refer to the supplementary material, available online for more results.)

TABLE 2
Performance Comparison with Other MOT Systems

Dataset	Method	Setting	MOTP ↑	MOTA ↑	GT	MT ↑	PT	ML ↓	IDS ↓	FG ↓	REC ↑	PRE ↑	FAF ↓
PETS S2L1	⋆ PIRHMPT [4]	Batch	-	-	19	78.90 %	21.10 %	**0.00 %**	1	23	89.50 %	**99.60 %**	**0.02**
	⋆ OLMOAP [5]	Batch	-	-	19	89.50 %	10.50 %	**0.00 %**	**0**	9	91.80 %	**99.00 %**	0.05
	Energy Min. [6]	Batch	80.20 %	90.60 %	23	91.30 %	4.35 %	4.35 %	11	**6**	92.40 %	98.40 %	0.07
	TINF [52]	Batch	63.12 %	90.04 %	-	95.00 %	5.00 %	0.00 %	3	-	-	-	-
	Conf. Map [9]	Online	56.30 %	79.70 %	-	-	-	-	-	-	-	-	-
	⋆ proposed (p1 - with all)	Online	**80.54 %**	**90.96 %**	19	**100.00 %**	0.00 %	**0.00 %**	4	8	**95.30 %**	94.66 %	0.31
PETS S2L2	NOMT [25]	Batch	**70.50 %**	53.40 %	42	14.30 %	76.20 %	9.50 %	142	208	-	-	-
	MHT-DAM [26]	Batch	61.40 %	**59.20 %**	42	23.81 %	71.43 %	4.76 %	120	**162**	-	-	-
	LP-SSVM [53]	Batch	**70.50 %**	41.50 %	42	7.10 %	76.20 %	16.70 %	212	249	-	-	-
	MDP [24]	Online	69.80 %	47.80 %	42	14.30 %	78.60 %	7.10 %	206	362	-	-	-
	⋆ proposed (p1 - with all)	Online	70.27 %	50.77 %	43	**42.86 %**	57.14 %	**0.00 %**	**113**	173	**64.42 %**	**83.69 %**	**2.96**
ETHMS (Bahnhof Sunny Day)	⋆ DP [27]	Batch	-	-	125	50.20 %	39.90 %	9.90 %	**4**	143	67.40 %	**91.40 %**	-
	⋆ PIRHMPT [4]	Batch	-	-	125	58.40 %	33.60 %	8.00 %	11	23	76.80 %	86.60 %	-
	⋆ Online CRF [7]	Batch	-	-	125	68.00 %	24.80 %	7.20 %	11	**19**	79.00 %	90.40 %	-
	⋆ DC-CRF [50]	Batch	-	-	125	66.40 %	25.40 %	8.20 %	57	69	77.30 %	87.20 %	-
	TBD [13]	Online	-	-	125	62.40 %	29.60 %	8.00 %	45	69	78.70 %	85.50 %	-
	⋆ proposed (p1 - with all)	Online	**69.93 %**	**73.26 %**	125	**76.00 %**	19.20 %	**4.80 %**	33	39	**85.06 %**	88.27 %	**0.77**
Town Centre	⋆ Stable tracking [49]	Online	**77.20 %**	61.20 %	230	60.87 %	27.39 %	11.74 %	292	230	78.90 %	**82.00 %**	**2.75**
	⋆ proposed (p1 - with all)	Online	76.50 %	**64.40 %**	230	**63.04 %**	27.39 %	**9.57 %**	**164**	**226**	**83.90 %**	81.30 %	3.05
CAVIAR (20Seq.)	Two-steps [22]	Online	—	—	140	84.29 %	12.14 %	3.57 %	14	0	81.80 %	—	0.136
	⋆PRIMPT [4]	Batch	—	—	143	86.00 %	13.30 %	0.70 %	**4**	17	88.10 %	96.60 %	**0.082**
	⋆OLMOAP [5]	Batch	—	—	143	89.10 %	10.20 %	0.70 %	5	11	90.20 %	96.10 %	0.095
	⋆OLDAM[29]	Batch	—	—	143	84.60 %	14.70 %	0.70 %	11	11	89.40 %	**96.90 %**	0.085
	⋆ proposed (p1 - with all)	Online	**89.07 %**	**88.25 %**	143	**92.15 %**	7.85 %	**0.00 %**	8	**7**	**92.26 %**	96.27 %	0.084

For each dataset ***the Best Results are Marked in Red Color and Bold.*** *In addition, the systems evaluated with the same detections and ground truth used in our system are marked with an asterisk* ⋆.

(ML ↓), the ratio of partially tracked trajectories (PT), i.e., , $1 - PT - ML$, the number of track fragments (FG ↓), recall (REC ↑), precision (PRE ↑), and false alarms per frame (FAF ↓). Here, ↑ represents that higher scores indicate better results, and ↓ denotes that lower scores indicate better results.

Comparison of Data Association. A comparison between different versions of the proposed system is given in Table 1. Based on the framework shown in Fig. 3, we have implemented the proposed system (p1) with all proposed algorithms and compare it with the systems (a1)-(a4) using different association algorithms in Table 1-A. The details of each system are described as follows:

(a1) MOT system *without LC-association*;
(a2) MOT system *without HC-association*;
(a3) MOT system *with greed association [10]*;
(a4) MOT system *with DA-TEP [14]*;
(p1) MOT system *with all proposed algorithms.*

In the greedy bipartite association [10], we construct an association score matrix between tracklets and detections using the same affinity model presented in Section 3.2. Then, the pairs having the minimum score in the association matrix are selected in ascending order until no further valid pair is available. In the data association with track existence probability (DA-TEP) [14], we compute the association matrix by evaluating the posterior association probability for each tracklet-to-detection pair, and determine optimal matching pairs minimizing a total cost of the matrix using the Hungarian algorithm.

From the results of (p1) and (a1)-(a4), we can see the effect of our data association algorithm. As expected, the proposed system (p1) truly shows the better performance for the most metrics. In particular, our full system (p1) noticeably reduces IDS and FG rates and increases MT rate against the system (a1). Fig. 13 also supports this analysis: with the LC-association, longer trajectories are built by linking fragmented trajectories. Furthermore, the comparison between (p1) and (a2) verifies that the association accuracy of our system is improved when using HC-& LC-association together, because the HC-association reduces the matching ambiguity in the global association. The systems (p1) and (a4) show the better results than (a3). This result implies that exploiting the reliability of tracklets as well as the likelihood (*or* affinity) can improve the data association accuracy.

Comparison of Appearance Learning. Table 1-B show the results of systems with difference appearance learning:

(b1) MOT system *without online-learned appearance models*;[3]
(b2) MOT system *with incremental LDA [12]*;
(b3) MOT system *with our deep model without OTL*;

By comparing our systems (p1) and (a1), we can see that the proposed appearance learning greatly enhances the performance for all the metrics. The comparison between the system (p1) and systems (b2) and (b3) proves that we can further reduce the IDS and FG rates when using our deep and OTL. Fig. 14 shows IDs of objects are correctly maintained using our appearance learning. These results indicate

3. As an appearance model, we use the RGB color histogram with 192 bins, and compute appearance affinity using the Bhattacharyya distance.

TABLE 3
Performance Comparison with Other MOT Systems on the 2015 & 2016 MOT Challenge Benchmark

Benchmark	Method	Setting	Detector	MOTA ↑	MOTP ↑	FAF ↓	MT ↑	PT	ML ↓	FP ↓	FN ↓	IDS ↓	FG ↓
	proposed (p1 - with all)	Online	Private	51.30 %	74.20 %	1.2	36.30 %	41.50 %	22.20 %	7110	22271	544	1335
	proposed (p1 - with all)	Online	Public	32.80 %	70.70 %	0.9	9.70 %	48.70 %	42.20 %	4983	35690	614	1583
2015	MDP-SubCNN [24]	Online	Private	47.50 %	74.20 %	1.5	30.00 %	51.40 %	18.60 %	8631	22969	628	1370
MOT	MDP [24]	Online	Public	30.30 %	71.30 %	1.7	13.00 %	48.60 %	38.40 %	9717	32422	680	1500
Challenge	NOMTwSDP [25]	Batch	Private	55.50 %	76.60 %	1.0	39.00 %	35.20 %	25.80 %	5594	21322	427	701
	MHT-DAM [26]	Batch	Public	32.40 %	71.80 %	1.6	16.00 %	40.20 %	43.80 %	9064	32060	435	826
	SiameseCNN [54]	Batch	Public	29.00 %	71.20 %	0.9	8.50 %	43.10 %	48.40 %	5160	37798	639	1316
	proposed (p1 - with all)	Online	Public	43.90 %	74.70 %	1.1	10.70 %	44.90 %	44.40 %	6450	95175	676	1795
	EAMTT [55]	Online	Public	38.80 %	75.10 %	1.4	7.90 %	43.00 %	49.10 %	8114	102452	965	1657
2016	OVBT [56]	Online	Public	38.40 %	75.40 %	1.9	7.50 %	45.20 %	47.30 %	11517	99463	1321	2140
MOT	NOMT [25]	Batch	Public	46.40 %	76.60 %	1.6	18.30 %	40.30 %	41.40 %	9753	87565	359	504
Challenge	MHT-DAM [26]	Batch	Public	42.90 %	76.60 %	1.0	13.60 %	39.50 %	46.90 %	5668	97919	499	659
	LINF1 [57]	Batch	Public	41.40 %	74.80 %	1.3	11.60 %	37.10 %	51.30 %	7896	99224	430	963

The results are sorted according to the setting and MOTA score. Due to the page limit, we present the results for recent publications. (More results can be found in the Mot Challenge website.)

TABLE 4
Performance Comparison with Different Deep Learning Algorithms Using Same Detections and Ground Truth on PETS, ETHMS and Town Centre Datasets

Dataset	Method	MOTP ↑	MOTA ↑	GT	MT ↑	PT	ML ↓	IDS ↓	FG ↓	REC ↑	PRE ↑	FAF ↓	Speed (spf)
Average	Siamese deep network [17]	71.80 %	62.80 %	417	54.70 %	38.10 %	7.20 %	643	1268	77.80 %	84.70 %	2.16	1.23
Results	Deep learning tracker [30]	72.20 %	64.40 %	417	53.50 %	38.60 %	7.90 %	501	1188	77.30 %	84.10 %	1.87	10.85
	p1 - with all	**74.31 %**	**69.85 %**	417	**66.43 %**	26.86 %	**6.71 %**	**314**	**446**	**82.17 %**	**86.98 %**	**1.78**	**1.20**

*For each metric, **the Best Results are Marked in Red Color and Bold** (Refer to the supplementary material, available online for more results).*

that the proposed appearance learning is beneficial for discriminating appearances of objects.

The more tracking results are shown in Fig. 15. Although the objects are frequently occluded and their appearances and motions are changed with time, our system robustly tracks the objects while keeping their original IDs.

Comparisons with Other Systems. Table 2 shows a quantitative comparison between the proposed system (p1) and other MOT systems. Overall, our system achieves better performance in terms of MOTP and MOTA. Although we could not find the MOTP and MOTA for the ETHMS and CAVIAR datasets in other literature, other metrics show the robustness of our system. When compared to other online tracking systems [8], [9], [13], [22], [24], [49], our system (p1) is far superior to their performance for all metrics.

Compared to the performance of the recently proposed batch tracking systems [5], [6], [7], [25], [26], [50], [53], our system (p1) is still competitive. Notably, this improvement is achieved without using future frame information and without employing multiple (color, shape, and/or texture) features (*c.f.* [4], [5], [7], [9]).

Our system produces slightly more IDS and FG than some batch systems for the ETHMS dataset. The reason of the more IDS and FG is that occlusions are more frequently occurred in the ETHMS dataset when persons are passing by each other. Furthermore, in many cases, tracked objects are fully occluded by other objects since the dataset is captured from the frontal view cameras on the ground plane.[4]

4. Even though our tracking system successfully recovers its original ID in the next frames, the IDS number can be increased since we do not correct the ID of the previous frames as discussed in Section 5.

However, our system is still comparable to other batch systems, and achieves the best performance in terms of MT and ML. It implies that our system can robustly construct tracks under challenging conditions.

MOTChallenge Benchmark Evaluation. We further evaluate the proposed system (p1) on 2015 & 2016 MOTChallenge benchmark [51], [58] for more comparison with other systems. Table 3 shows the performance of different tracking systems on the challenge test datasets. In this experiment, we have tuned our system parameters with the provided training datasets only, and used the same parameters as described in Section 6.2 except O^F and O^B — we set O^F and O^B to $\mathrm{diag}[32^2 64^2]$ for the AVG-TownCentre sequence due to its low frame-rate. In 2015 MOTChallenge, we evaluated our system with the public benchmark detections and private detections provided by [25], whereas we only evaluated it with the public benchmark detections in 2016 MOTChallenge.

As shown in Table 3, our system achieves the competitive performance with other state-of-the-art systems, and the best scores for the most metrics among all online systems. This shows that our system can produce high performance for many different scenarios, and proves the robustness of our system.

Comparisons with Other Deep Learning Methods. To show effectiveness of our deep appearance learning, we compare our method with other deep learning methods [17], [30]. We implemented several systems by incorporating other deep learning methods into our MOT framework and exploiting them for appearance models. Table 4 shows their performance for several datasets. For a fair comparison, we use the codes open to the public for [17], [30], and use the same training dataset and network structure.

Our system (p1) shows the best performance for the most metrics. In particular, (p1) yields greatly improved the rates of MT, IDS, FG and REC. For the complexity of appearance learning, our method is more efficient than [17], [30]. Our system (p1) is much (about 9 times) faster than [30] because N deep networks are required in the deep learning tracker [30] for given N objects, while only one network is trained and updated in our method.

7 Conclusion

In this paper, we have proposed a robust online multi-object tracking method based on tracklet confidence and the discriminative deep appearance learning. We evaluate the tracklet confidence using the detectability and continuity of a tracklet, and then build optimal tracklets by sequentially linking tracklets and detections using the proposed high and low confidence association according to their confidence. Here, for more accurate and robust association between tracklets and detections, we learn the discriminative deep appearance model from a large dataset with the new energy function, and adapt the learned deep model by using online transfer learning for improving the discriminability between objects in specific sequences during tracking.

We have proved that the proposed methods lead to the noticeable performance improvement through extensive experiments. Since we have used a simple linear motion model, learning nonlinear motions [5] and relative motions [7] would be beneficial in order to further improve the description ability of our system.

Acknowledgments

This work was supported by the ICT R& D program of MSIP/IITP [No. B0101-15-0266, Development of High Performance Visual BigData Discovery Platform] and the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP) [No. NRF-2015R1A2A1A01005455]. Kuk-Jin Yoon is the corresponding author.

References

[1] S. Ren, K. He, R. B. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. Advances Neural Inf. Process. Syst.*, 2015, pp. 91–99.
[2] W. Liu, et al., "SSD: Single shot multibox detector," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 21–37.
[3] W. Brendel, M. Amer, and S. Todorovic, "Multiobject tracking as maximum weight independent set," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2011, pp. 1273–1280.
[4] C.-H. Kuo and R. Nevatia, "How does person identity recognition help multi-person tracking?" in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2011, pp. 1217–1224.
[5] B. Yang and R. Nevatia, "Multi-target tracking by online learning of non-linear motion patterns and robust appearance models," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2012, pp. 1918–1925.
[6] A. Milan, S. Roth, and K. Schindler, "Continuous energy minimization for multitarget tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 1, pp. 58–72, Jan. 2014.
[7] B. Yang and R. Nevatia, "An online learned CRF model for multi-target tracking," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2012, pp. 2034–2041.
[8] B. Wu and R. Nevatia, "Detection and tracking of multiple, partially occluded humans by Bayesian combination of edgelet based part detectors," *Int. J. Comput. Vis.*, vol. 75, no. 2, pp. 247–266, 2007.
[9] M. D. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. J. Van Gool, "Online multiperson tracking-by-detection from a single, uncalibrated camera," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 9, pp. 1820–1833, Sep. 2011.
[10] G. Shu, A. Dehghan, O. Oreifej, E. Hand, and M. Shah, "Part-based multiple-person tracking with partial occlusion handling," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2012, pp. 1815–1821.
[11] X. Song, J. Cui, H. Zha, and H. Zhao, "Vision-based multiple interacting targets tracking via on-line supervised learning." in *Proc. Eur. Conf. Comput. Vis.*, 2008, pp. 642–655.
[12] S. H. Bae and K. Yoon, "Robust online multi-object tracking based on tracklet confidence and online discriminative appearance learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2014, pp. 1218–1225.
[13] F. Poiesi, R. Mazzon, and A. Cavallaro, "Multi-target tracking on confidence maps: An application to people tracking," *Comput. Vis. Image Understanding*, vol. 117, no. 10, pp. 1257–1272, 2013.
[14] S. H. Bae and K. Yoon, "Robust online multiobject tracking with data association and track management," *IEEE Trans. Image Process.*, vol. 23, no. 7, pp. 2820–2833, 2014.
[15] S. Avidan, "Ensemble tracking," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recog.*, 2005, pp. 494–501.
[16] C. Huang, Y. Li, and R. Nevatia, "Multiple target tracking by learning-based hierarchical association of detection responses," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 4, pp. 898–910, Apr. 2013.
[17] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recog.*, 2005, pp. 539–546.
[18] M. Isard and A. Blake, "CONDENSATION—conditional density propagation for visual tracking," *Int. J. Comput. Vis.*, vol. 29, no. 1, pp. 5–28, 1998.
[19] K. Okuma, A. Taleghani, N. de Freitas, J. J. Little, and D. G. Lowe, "A boosted particle filter: Multitarget detection and tracking," in *Proc. Eur. Conf. Comput. Vis.*, 2004, pp. 28–39.
[20] Y. Cai, N. de Freitas, and J. J. Little, "Robust visual tracking for multiple targets," in *Proc. Eur. Conf. Comput. Vis.*, 2006, pp. 107–118.
[21] C. Huang, B. Wu, and R. Nevatia, "Robust object tracking by hierarchical association of detection responses," in *Proc. Eur. Conf. Comput. Vis.*, 2008, pp. 788–801.
[22] J. Xing, H. Ai, and S. Lao, "Multi-object tracking through occlusions by local tracklets filtering and global tracklets association with detection responses," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2009, pp. 1200–1207.
[23] L. Zhang, Y. Li, and R. Nevatia, "Global data association for multi-object tracking using network flows," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2008, pp. 1–8.
[24] Y. Xiang, A. Alahi, and S. Savarese, "Learning to track: Online multi-object tracking by decision making," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 4705–4713.
[25] W. Choi, "Near-online multi-target tracking with aggregated local flow descriptor," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 3029–3037.
[26] C. Kim, F. Li, A. Ciptadi, and J. M. Rehg, "Multiple hypothesis tracking revisited," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 4696–4704.
[27] H. Pirsiavash, D. Ramanan, and C. C. Fowlkes, "Globally-optimal greedy algorithms for tracking a variable number of objects," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2011, pp. 1201–1208.
[28] H. Grabner and H. Bischof, "On-line boosting and vision," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recog.*, 2006, pp. 260–267.
[29] C.-H. Kuo, C. Huang, and R. Nevatia, "Multi-target tracking by on-line learned discriminative appearance models," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recog.*, 2010, pp. 685–692.
[30] N. Wang and D.-Y. Yeung, "Learning a deep compact image representation for visual tracking," in *Proc. 26th Int. Conf. Neural Inf. Process. Syst.*, 2013, pp. 809–817.
[31] H. Li, Y. Li, and F. Porikli, "DeepTrack: Learning discriminative feature representations by convolutional neural networks for visual tracking," in *Proc. British Mach. Vis. Conf.*, 2014, pp. 1–12.
[32] W. Li, R. Zhao, T. Xiao, and X. Wang, "DeepReID: Deep filter pairing neural network for person re-identification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2014, pp. 152–159.
[33] J. Hu, J. Lu, and Y.-P. Tan, "Discriminative deep metric learning for face verification in the wild," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2014, pp. 1875–1882.
[34] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows—Theory, Algorithms and Applications*. Englewood Cliffs, NJ, USA: Prentice Hall, 1993.

[35] B. Babenko, M.-H. Yang, and S. Belongie, "Robust object tracking with online multiple instance learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 8, pp. 1619–1632, Aug. 2011.
[36] W. Li and X. Wang, "Locally aligned feature transforms across views," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2013, pp. 3594–3601.
[37] K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun, "What is the best multi-stage architecture for object recognition?" in *Proc. IEEE 12th Int. Conf. Comput. Vis.*, 2009, pp. 2146–2153.
[38] I. J. Goodfellow, D. Warde-Farley, M. Mirza, A. C. Courville, and Y. Bengio, "Maxout networks," in *Proc. 30th Int. Conf. Mach. Learn.*, 2013, pp. 1319–1327.
[39] P. Y. Simard, Y. LeCun, J. S. Denker, and B. Victorri, "Transformation invariance in pattern recognition-tangent distance and tangent propagation," in *Neural Networks: Tricks of the Trade*, 2nd ed. Berlin, Germany: Springer, 2012.
[40] A. Mignon and F. Jurie, "PCCA: A new approach for distance learning from sparse pairwise constraints," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2012, pp. 2666–2672.
[41] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Learning and transferring mid-level image representations using convolutional neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2014, pp. 1717–1724.
[42] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. 13th Int. Conf. Artif. Intell. Statist.*, 2010, pp. 249–256.
[43] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Advances Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.
[44] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *Computing Research Repository*, abs/1409.1556, 2014.
[45] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 630–645.
[46] A. Ess, B. Leibe, K. Schindler, and L. van Gool, "A mobile vision system for robust multi-person tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, Jun. 2008, pp. 1–8.
[47] C. dataset, 2009. [Online]. Available: http://homepages.inf.ed.ac.uk/rbf/caviardata1/
[48] P. dataset, 2009. [Online]. Available: http://www.cvg.rdg.ac.uk/pets2009/index.html
[49] B. Benfold and I. Reid, "Stable multi-target tracking in real-time surveillance video," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2011, pp. 3457–3464.
[50] A. Milan, K. Schindler, and S. Roth, "Detection- and trajectory-level exclusion in multiple object tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2013, pp. 3682–3689.
[51] L. Leal-Taixé, A. Milan, I. Reid, S. Roth, and K. Schindler, "MOTChallenge 2015: Towards a benchmark for multi-target tracking," *Computing Research Repository*, abs/1504.01942, 2015.
[52] A. Dehghan, Y. Tian, P. H. S. Torr, and M. Shah, "Target identity-aware network flow for online multiple target tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2015, pp. 1146–1154.
[53] S. Wang and C. C. Fowlkes, "Learning optimal parameters for multi-target tracking," in *Proc. British Mach. Vis. Conf.*, 2015, pp. 4.1–4.13.
[54] L. Leal-Taixe, C. Canton-Ferrer, and K. Schindler, "Learning by tracking: Siamese cnn for robust target association," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog. Workshops*, Jun. 2016, pp. 418–425.
[55] R. Sanchez-Matilla, F. Poiesi, and A. Cavallaro, "Online multi-target tracking with strong and weak detections," in *Proc. Eur. Conf. Comput. Vis. Workshops*, 2016, pp. 84–99.
[56] Y. Ban, S. O. Ba, X. Alameda-Pineda, and R. Horaud, "Tracking multiple persons based on a variational bayesian model," in *Proc. Eur. Conf. Comput. Vis. Workshops*, 2016, pp. 52–67.
[57] L. Fagot-Bouquet, R. Audigier, Y. Dhome, and F. Lerasle, "Improving multi-frame data association with sparse representations for robust near-online multi-object tracking," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 774–790.
[58] A. Milan, L. Leal-Taixé, I. Reid, S. Roth, and K. Schindler, "MOT16: A benchmark for multi-object tracking," *Computing Research Repository*, abs/1603.0083, 2016.

Seung-Hwan Bae received the BS degree in information and communication engineering from Chungbuk National University, in 2009 and the MS and PhD degrees in information and communications from the Gwangju Institute of Science and Technology (GIST), in 2010 and 2015, respectively. He was a senior researcher at Electronics and Telecommunications Research Institute (ETRI) in Korea from 2015 to 2017. He is currently an assistant professor in the Department of Computer Science and Engineering at Incheon National University, Korea. His research interests include multi-object tracking, object detection, deep learning, dimensionality reduction, medical image analysis, etc.

Kuk-Jin Yoon received the BS, MS, and PhD degrees in electrical engineering and computer science from the Korea Advanced Institute of Science and Technology (KAIST), in 1998, 2000, and 2006, respectively. He was a post-doctoral fellow in the PERCEPTION team in INRIA-Grenoble, France, for two years from 2006 to 2008, and joined the School of Information and Communications, Gwangju Institute of Science and Technology (GIST), Korea, as an assistant professor in 2008. He is currently an associate professor and a director of the Computer Vision Laboratory, GIST. His research interests include stereo, visual object tracking, SLAM, structure-from-motion, etc.