IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received January 15, 2020, accepted February 15, 2020, date of publication February 24, 2020, date of current version March 3, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2975912

OneShotDA: Online Multi-Object Tracker With One-Shot-Learning-Based Data Association

KWANGJIN YOON[1], JEONGHWAN GWAK[2], YOUNG-MIN SONG[3], YOUNG-CHUL YOON[4], AND MOONGU JEON[3]

[1]SI-Analytics Company, Ltd., Daejeon 34051, South Korea
[2]Department of Software, Korea National University of Transportation, Chungju 27469, South Korea
[3]School of Electrical Engineering and Computer Science, Gwangju Institute of Science Technology, Gwangju 61005, South Korea
[4]Robot Center, LG Electronics, Seoul 07336, South Korea

Corresponding author: Moongu Jeon (mgjeon@gist.ac.kr)

This work was supported in part by the Institute of Information and Communications Technology Planning and Evaluation (IITP) grant funded by the Korean Government (MSIT) under Grant 2014-3-00077, in part by the AI National Strategy Project, and in part by the National Research Foundation of Korea (NRF) grant funded by the Korean Government (MSIT) under Grant 2019R1A2C2087489 and Grant 2017R1D1A1B03036423.

ABSTRACT Tracking multiple objects in a video sequence can be accomplished by identifying the objects appearing in the sequence and distinguishing between them. Therefore, many recent multi-object tracking (MOT) methods have utilized re-identification and distance metric learning to distinguish between objects by computing the similarity/dissimilarity scores. However, it is difficult to generalize such approaches for arbitrary video sequences, because some important information, such as the number of objects (classes) in a video, is not known in advance. Therefore, in this study, we applied a one-shot learning framework to the MOT problem. Our algorithm tracks objects by classifying newly observed objects into existing tracks, irrespective of the number of objects appearing in a video frame. The proposed method, called *OneShotDA*, exploits the one-shot learning framework based on an attention mechanism. Our neural network learns to classify unseen data samples using labels from a support set. Once the network has been trained, it predicts correct labels for newly received detection results based on the set of existing tracks. To analyze the effectiveness of our method, it was tested on the MOTchallenge benchmark datasets (MOT16 and MOT17 datasets). The results reveal that the performance of the proposed method was comparable with those of current state-of-the-art methods. In particular, it is noteworthy that the proposed method ranked first among the online trackers on the MOT17 benchmark.

INDEX TERMS Data association, deep learning, multi-object tracking, object recognition, one-shot learning.

I. INTRODUCTION

Multi-object tracking (MOT) is considered one of the most challenging problems in computer vision research. Recently, *tracking-by-detection* methods have attracted significant interest, because they can isolate the problem of object detection from object tracking, which helps them focus on the tracking tasks, such as track management, initiation, and termination, as well as data association.

There are several methods available for track initiation/termination. Several studies [1]–[4] have adopted a straightforward rule wherein tracking starts if there is a detection result, and ends if there is no detection result. A subtle difference between such approaches is the number of repeated detections (misdetections) used for track initiation (termination). In [1]–[4], a new track hypothesis was generated at every frame for each detection result that was not associated with an existing track. A track was terminated if the number of consecutive misdetections exceeded a predefined threshold. To eliminate false trajectories, tracks shorter than a predefined threshold were deleted from the track set after the tracking process was completed.

Another strategy involves optimizing an objective function over the space of trajectories [11]–[13]; it is necessary to perform both track management and data

The associate editor coordinating the review of this manuscript and approving it for publication was Victor Sanchez.

association simultaneously, based on the optimization results. Zhang *et al.* [11] proposed a network-flow-based global optimization method for MOT. They constructed a network using a set of detection results from a video, and computed the global best trajectories by identifying the min-cost flow of the network. Initialization and termination of trajectories were handled intrinsically after the solution had been computed. Pirsiavash *et al.* [12] used an approach similar to that of Zhang *et al.* , except that they adopted a greedy algorithm (shortest path) for a flow network. In [13], the authors used the multiple-hypothesis tracking (MHT) for track management. The MHT saves track proposals in a tree structure that grows with new detections for each frame, that is, the tree describes all the possible data association results originating from a single detection result. Track initiation and termination and data association in the MHT are treated as solving an optimization problem, that is, the maximum weighted independent set (MWIS), within a certain time window.

In the tracking-by-detection paradigm, data association entails connecting detection outputs across video frames and screening misdetections. This problem can be considered a form of statistical estimation such as the likelihood estimation of $p(\mathcal{Z}|\mathcal{T})$, where $\mathcal{Z}$ is the set of detections and $\mathcal{T}$ is the set of trajectories. The distribution of likelihood determines the probability of associated detections belonging to the same object when the track proposal $\mathcal{T}$ has been satisfied. Online tracking methods recursively estimate the likelihood based on the detection set up to the current frame [1]–[4], [13]. In contrast, offline/batch methods [11], [12] use the detection results for an entire video sequence.

Recently, MOT has been accomplished by identifying and distinguishing between objects appearing in the sequence. Therefore, many recent MOT methods have focused on re-identification and distance metric learning to distinguish between objects by computing the similarity/dissimilarity scores between them. However, it is difficult to generalize such approaches for arbitrary video sequences, because some important information, *e.g.*, the number of objects (classes) in a video, is not known in advance. In this study, we propose a novel data association strategy called *OneShotDA* that exploits one-shot learning frameworks such as those in [14]–[16]. In such frameworks, the class of a query sample is determined by the samples in a gallery set. For example, in [16], predictions for the samples in a query set are obtained based on a relation module that computes the distance between a query feature and the features in a gallery set. By following the protocol of the one-shot framework, our method classifies a newly received detection result (query sample) into an existing track (gallery set), or vice versa.[1] Specifically, our model can predict the label of a query sample based on the labels of the gallery set by using an attention mechanism that indicates a corresponding sample in the gallery set.

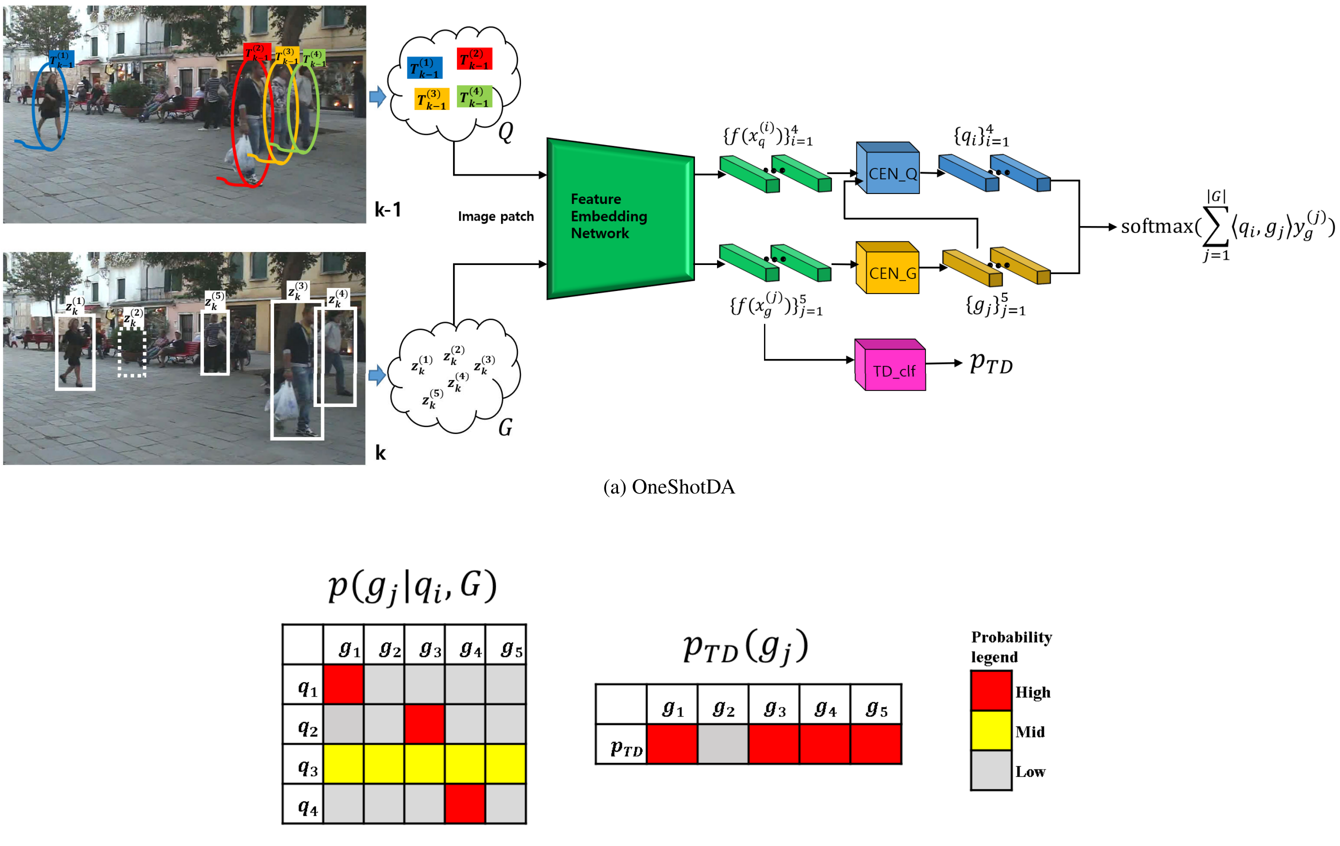
Let Q be the query set, and $x_q^{(i)}$ be the i-th query sample. G is the gallery set. $G = \{(x_g^{(j)}, y_g^{(j)})\}_{j=1}^{|G|}$, where $x_g^{(j)}$ and $y_g^{(j)}$ are the j-th sample and label in the gallery set, respectively. In Figure 1(a), $\mathcal{T}_{k-1}$, which is the track set at frame $k-1$, represents the query set, and $\mathcal{Z}_k$, which is the detection set at frame k, represents the gallery set. The feature embedding network (FEN) in OneShotDA takes a sample from both Q and G as an input, and generates a feature vector $f(\cdot)$. Note that the FEN processes samples in Q and G using the same weights. Next, conditional embedding networks (CENs) are used to embed feature vectors to generate more robust features. CEN_Q and CEN_G are the CENs for the query set and gallery set, respectively. Additionally, we include a network called TD_clf that estimates the probability of accurate detection for a given input (detection response). We shall detail each component of the OneShotDA in the following sections.

As shown in Figure 1(b), the OneShotDA maps the class of a query sample to labels in the gallery set, which is defined by $p(y_g|x_q, G)$. The left table in Figure 1(b) contains the probability distribution of data associations for the scenario depicted in Figure 1(a). For example, q_3, an embedded vector of $T_{k-1}^{(3)}$, has a low probability of data association, because the object is completely occluded in frame k. The right table in Figure 1(b) contains the probabilities of accurate detection for the gallery set, which are estimated by TD_clf. We also demonstrate that the proposed data association mechanism can be easily integrated with any online MOT system. In the experiments section, we track objects using a combination of the MHT framework [7] and the proposed OneShotDA. Tables 1 and 2 summarize the notations and abbreviations used in this article.

TABLE 1. Notation summary: We define the mathematical notations used for formalizing problems in this article.

Notation	meaning
$\mathcal{Z}$	set of detections
$\mathcal{T}$	set of trajectories
G	gallery set
Q	query set
$x_g^{(i)}$	i-th sample in the gallery set
$y_g^{(i)}$	label of the i-th sample in the gallery set
$x_q^{(i)}$	i-th sample in the query set
$T_k^{(i)}$	i-th track proposal at frame k
$z_k^{[i]}$	detection result chosen by the i-th track proposal at frame k
$LR(\zeta)$	score function (likelihood ratio) of the track proposal ζ
$\mathcal{Z}_k$	set of detection results at frame k
$z_k^{(i)}$	i-th detection result at frame k
H_1	true-track hypothesis
H_0	false-alarm hypothesis
$\texttt{app}(\zeta)$	a function that returns the appearance feature of the input ζ
$\texttt{kin}(\zeta)$	a function that returns the kinematic component of the input ζ
$\texttt{cls}(\zeta, \psi)$	label of a sample ζ, which is one of ψ
$y_q^{(i)}$	true label of the i-th query sample
$\hat{y}_q^{(i)}$	estimated label of the i-th query sample
p_{TD}	probability of true detection

[1] The use of detection results as a query set or the use of existing tracks as a query set is possible.


k-1
k
Q
G
Image patch
Feature Embedding Network
$\{f(x_q^{(i)})\}_{i=1}^4$
$\{q_i\}_{i=1}^4$
CEN_Q
CEN_G
$\{f(x_g^{(j)})\}_{j=1}^5$
$\{g_j\}_{j=1}^5$
$\mathrm{softmax}(\sum_{j=1}^{|G|}\langle q_i, g_j\rangle y_g^{(j)})$
TD_clf
p_{TD}
$p(g_j|q_i, G)$
$p_{TD}(g_j)$
Probability legend
High
Mid
Low

(a) OneShotDA

(b) Output of OneShotDA

FIGURE 1. **(a): Architecture of the proposed OneShotDA method. Our network consists of FEN, CEN_Q, CEN_G, and TD_clf. (b): Outputs of OneShotDA.**

TABLE 2. **Abbreviation summary: These abbreviations are used in our model.**

Abbr.	meaning
CEN	Conditional Embedding Network
CEN_G	CEN for the set G
CEN_Q	CEN for the set Q
CNN	Convolutional Neural Network
DPM	Deformable Part Model detector [5]
FEN	Feature Embedding Network, generates a feature vector for a given input (x_q and x_g)
LSTM	Long Short-Term Memory [6]
MHT	Multiple Hypothesis Tracking [7]
MOTA	Multi-Object Tracking Accuracy [8]
OneShotDA	proposed data association mechanism using one-shot learning with an attention technique
R-Cnn	Region-based Convolutional Neural Network [9]
SDP	Scale Dependent Pooling [10]
TD_clf	Neural network for identifying true detection results, which follows the FEN

The contributions of this study can be summarized as follows:

- We propose a novel data association mechanism called *OneShotDA* that can classify newly generated detection outputs into existing tracks using one-shot classification.
- We adopt a training strategy that is customized for one-shot learning and suitable for data association tasks. We also demonstrate the way training samples are generated using MOT datasets such as the *MOTChallenge* datasets [8].
- We demonstrate that OneShotDA can be easily integrated with any online MOT system (MHT in this study).[2]
- We demonstrate that the effectiveness of the proposed MOT system can match those of the state-of-the-art methods. It is noteworthy that the proposed method ranks first among online trackers when evaluated on the MOT17 benchmark.

II. RELATED WORKS

Modern MOT methods based on the tracking-by-detection paradigm can be categorized into offline and online methods. Offline methods utilize the detection results from all the video frames to construct robust trajectories, whereas, online approaches process videos sequentially in a frame-by-frame manner by recursively updating existing tracks with new detection results.

The offline methods are commonly set up by representing the problem as a graph wherein each detection result represents a node, and the edges represent possible links.

[2] The MHT tracker used in this research is now available on the web: https://github.com/yoon28/pymht

Ma *et al.* [17] formulated the problem as a hierarchical correlation clustering (HCC), a modified form of the correlation-clustering-based tracking method [18]. Tang *et al.* [19] used a combination of the lifted multicut problem (LMP) formulation and body pose information. Henschel *et al.* [20] also utilized body pose information; however, they formulated the problem as a min-cost graph labeling problem [21].

It is known that online methods are often impeded by long-term occlusion issues, because only the current frame and previous frames are available [22]. The MHT [7], a robust online tracking method, attempts to resolve this issue by constructing a track tree that describes all the possible data association results within a particular time window. Even if this time window causes the MHT to produce delayed tracking results, it is still considered an online tracker because only the current detection set is used to update the scores of previous tracks [7]. In other words, the MHT recursively estimates the likelihood of the previous tracks reoccurring based on current observations. Recently, various MHT algorithms [13], [22], [23] have been proposed for MOT tasks. In [13], long-term appearance modeling was incorporated into the MHT, where the tracker estimated the online appearance features for each track. In [23], an LSTM network was adopted to score track proposals in the MHT. The authors also proposed a bilinear LSTM model, a modified version of the original LSTM model [6] for a gating network. In [22], the authors proposed an iterative MWIS algorithm for the MHT, making it possible to solve the MWIS problem based on the solutions of previous frames.

There have also been many recent studies based on MOT. Sun *et al.* [24] tracked objects using a novel deep network that can infer object affinities across different frames by analyzing exhaustive permutations of the extracted features. Their network also accounts for the appearance and disappearance of objects between video frames. Yang *et al.* [25] proposed an online MOT algorithm that uses two-step data association combined with an improved sparse-based appearance affinity model and rank-based motion affinity model. They tracked objects by fusing trajectory dynamics information, and proposed a novel two-step data association framework. He *et al.* [26] proposed a tracking-by-animation framework to achieve both label-free and end-to-end learning for MOT, unlike tracking-by-detection frameworks, that isolate the detection task from the tracking task. Their differentiable neural network first tracks objects in input frames, and then animates the tracked objects in reconstructed frames. Learning is driven by reconstruction error based on backpropagation. Zhang *et al.* [27] tracked objects in multi-modal scenarios by adopting a deep architecture that can be trained in an end-to-end manner, thereby enabling the joint optimization of the base feature extractors of each modality and an adjacency estimator for cross-modality. Wen *et al.* [28] proposed an MOT algorithm based on a non-uniform hypergraph that can model different degrees of dependency among tracklets for a unified objective. Their method can model higher-order dependencies among objects and tracklets. Voigtlaender *et al.* [29] extended the MOT to MOT and segmentation. They also presented a tracking method that jointly addresses detection, tracking, and segmentation using a single convolutional network. Long *et al.* [30] tackled unreliable detection by selecting candidates from the outputs of both detection and tracking. They also demonstrated that the identification ability of their tracker could be improved by using appearance representations trained on a person re-identification dataset. In [31], the authors tracked the objects using recurrent neural networks (RNNs), and demonstrated that RNNs can effectively address the problems of trajectory estimation and data association.

One(few)-shot learning aims to train a classifier to recognize unseen classes during training using only a single (few) labeled example(s). Because many deep learning systems require hundreds or thousands of samples, one(few)-shot learning has attracted significant interest [14]–[16], [32]. In such frameworks, the class of a query sample is determined by the samples in a gallery set. In [32], the authors proposed a novel strategy for one-shot classification using Siamese neural networks for verification. In [14], the MatchingNet model was proposed to map a small labelled gallery set and an unlabeled query sample to the correct label. MatchingNet compares the cosine distances between a query feature and each gallery feature. Snell *et al.* [16] proposed ProtoNet, which also predicts the class of a query sample based on distance; however, ProtoNet uses the Euclidean distance between a query and the gallery. Sung *et al.* [15] presented RelationNet, which, apart from replacing distance with a learnable relation module, is based on a similar concept.

In OneShotDA, we adopt the one-shot learning framework to classify new samples based on known examples, which solves the data association problem in MOT. There are multiple available options for this task, including existing frameworks such as MatchingNet, RelationNet, or ProtoNet. Therefore, we consider the one-shot framework as a data association solver for MOT.

III. MULTIPLE HYPOTHESIS TRACKING

In this section, we briefly review the MHT model presented in [7] and discuss how we combine MHT with the proposed OneShotDA.

MHT maintains a track proposal by constructing a track tree that describes all possible data association results originating from a single detection result (i.e., root node) and its branches. Each node in the track tree can either correspond to a real detection result obtained from an object detector or be a dummy detection result representing a misdetection. Let $T_k^{(i)}$ be the i-th track proposal at frame k maintained by the MHT and let t be its length. Then, $T_k^{(i)} = \{z_{k-t+1}^{[i]}, z_{k-t+2}^{[i]}, \ldots, z_{k-1}^{[i]}, z_k^{[i]}\}$, where $z_l^{[i]}$ is a detection result chosen by the i-th track proposal at frame l. As mentioned previously, $z_l^{[i]}$ can be a real detection result from $\mathcal{Z}$ or a dummy detection result. Note that $z_l^{(j)}$ represents the j-th detection at frame l, meaning $z_l^{(j)} \in \mathcal{Z}_l$ in which $\mathcal{Z}_l$ is a set of detections at frame l (i.e., $\mathcal{Z}_l \subset \mathcal{Z}$).

Following the original formulation in [7], the score for each track proposal is defined as the likelihood ratio (LR) between the true-track (H_1) and false alarm (H_0) hypotheses (Eq. 1).

$$LR(T_k^{(i)}) = \frac{p(T_k^{(i)}|H_1)}{p(T_k^{(i)}|H_0)} \frac{P_0(H_1)}{P_0(H_0)} \tag{1}$$

In Eq. 1, $P_0(H_1)$ and $P_0(H_0)$ are the prior probabilities of the true target and false alarm hypotheses, respectively. Based on the chain rule, the likelihood is factorized as

$$LR(T_k^{(i)}) = \frac{\prod_{l=1}^{t} p(z_{k-l+1}^{[i]}|T_{k-l}^{(i)}, H_1)}{\prod_{l=1}^{t} p(z_{k-l+1}^{[i]}|H_0)} \frac{P_0(H_1)}{P_0(H_0)}, \tag{2}$$

where we assume that detection results are conditionally independent given the false alarm hypothesis and t is the track length of $T_k^{(i)}$. This equation can be further factorized based on the independence assumption between kinematic and appearance terms as follows:

$$LR(T_k^{(i)}) = \frac{\prod_{l=1}^{t} p_K(\mathtt{kin}(z_{k-l+1}^{[i]})|\mathtt{kin}(T_{k-l}^{(i)}), H_1)}{\prod_{l=1}^{t} p_K(\mathtt{kin}(z_{k-l+1}^{[i]})|H_0)} \times \frac{\prod_{l=1}^{t} p_A(\mathtt{app}(z_{k-l+1}^{[i]})|\mathtt{app}(T_{k-l}^{(i)}), H_1)}{\prod_{l=1}^{t} p_A(\mathtt{app}(z_{k-l+1}^{[i]})|H_0)} \frac{P_0(H_1)}{P_0(H_0)}, \tag{3}$$

where, $\mathtt{app}(\cdot)$ is a function that returns the appearance feature of a given input and $\mathtt{kin}(\cdot)$ returns the kinematic component of an input (*e.g.*, the coordinate of the bounding box). The kinematic term of $LR(T_k^{(i)})$ at frame $(k - l + 1)$ under the true-track hypothesis is assumed to be a Gaussian and is estimated via Kalman filtering. Constants are set for the false alarm hypotheses of both the appearance and kinematic terms ($p_A(\mathtt{app}(z_{k-l+1}^{[i]})|H_0)$ and $p_K(\mathtt{kin}(z_{k-l+1}^{[i]})|H_0)$ are set to $C_{\mathtt{app}}$ and $C_{\mathtt{kin}}$, respectively).

We use the log likelihood ratio for track scoring by taking the logarithm of Eq 3. Additionally, the track initiation score is defined as $\ln[\frac{P_0(H_1)}{P_0(H_0)}]$ and a constant C_β is set. Track termination is performed after a track is updated with dummy detection results for τ_{miss} consecutive frames. To maintain the size of the feasible track proposals, the track tree is pruned such that it does not exceed a tree depth of τ_D. Tree pruning can be performed after finding the best set of proposals. Once the best set is identified, we select an ancestor of the best proposal with a distance τ_D and prune the subtrees diverging from that node.

Finally, the likelihood for the appearance term at frame l under the true-track hypothesis is estimated by OneShotDA. The true-track likelihood at frame l is written as

$$p_A(\mathtt{app}(z_l^{[i]})|\mathtt{app}(T_{l-1}^{(i)}), H_1) = p(\mathtt{cls}(z_l^{[i]}, \mathcal{Z}_l)|T_{l-1}^{(i)}, \mathcal{Z}_l), \tag{4}$$

where $\mathtt{cls}(z, \mathcal{Z})$ is a function that retrieves the label of a detection result z determined by $\mathcal{Z}$. Therefore, the right side of Eq. 4 estimates the probability that a track $T_{l-1}^{(i)}$ is classified as $\mathtt{cls}(z_l^{[i]}, \mathcal{Z}_l)$.

IV. DATA ASSOCIATION WITH ONE-SHOT LEARNING

For the data association task of MOT, we decided to adopt the one-shot architecture of MatchingNet [14] because its contextual embedding provides robust input features, particularly when two difficult examples are very close to each other in the feature space [14]. However, simply applying MatchingNet directly to our domain is not possible because the data association problem does not match a detection to a track and the proposed system must identify false alarms.

Let Q and G be a set of query samples and a set of gallery samples, respectively. Each sample consists of image-label pairs (i.e., $Q = \{(x_q^{(i)}, y_q^{(i)})\}_{i=1}^{|Q|}$, and $G = \{(x_g^{(j)}, y_g^{(j)})\}_{j=1}^{|G|}$). $|Q|$ and $|G|$ represent the sizes of sets Q and G, respectively. The labels $y_g^{(j)}$ in the gallery set are $|G|$-sized vectors, each of which is one-shot encoded such that the j-th component is set to 1.

OneShotDA estimates the probability distribution of the label of the i-th query sample for the labels in the gallery set (Eq. 5).

$$p(\hat{y}_q^{(i)}|x_q^{(i)}, G) = \mathtt{softmax}(\sum_{j=1}^{|G|} \langle q_i, g_j \rangle y_g^{(j)}), \tag{5}$$

where $\langle \cdot, \cdot \rangle$ is the inner product between two vectors. q_i and g_j are the corresponding embedding vectors of $x_q^{(i)}$ and $x_g^{(j)}$, respectively. In Eq. 5, the probability distribution of the query sample's label $\hat{y}_q^{(i)}$ is computed by applying the $\mathtt{softmax}$ function over the gallery set's labels. Therefore, the class of the query sample is computed from the gallery set with the maximum probability (i.e., $\arg\max_{\hat{y}_q^{(i)} \in G} p(\hat{y}_q^{(i)}|x_q^{(i)}, G)$). This process can be viewed as an attention mechanism pointing to a corresponding sample in the gallery set. It is important to note that the estimated label of the query sample is not the same as the label in the query set (i.e., $\hat{y}_q^{(i)} \neq y_q^{(i)}$). This is because the label of query sample only represents its class [14]. In addition, as mentioned previously, the labels in the query set are used for network training, meaning we can train the network twice per query-gallery pair by swapping the two sets.

In Eq. 5, q_i and g_j are embedding vectors mapped from the image space into a latent space. One potential method for performing mapping for each sample is to train an embedding network and apply the network to each sample independently (*e.g.*, $q_i = f(x_q^{(i)}), g_j = f(x_g^{(j)})$ for $i = 1, \ldots, |Q|, j = 1, \ldots, |G|$, where f is a CNN). We made f an FEN and searched the ResNet family [33] to find the optimal FEN structure. However, embedding each sample independently means we cannot encode information regarding the entire set, so the classification function in Eq. 5 is simply nearest neighbor classification based on an inner product. To resolve this issue, we train a CEN that embeds the feature vector further by incorporating all other samples. This can improve the accuracy of classification, particularly in cases where some samples are very close to each other (i.e., hard samples).

Specifically, CEN_Q is the CEN for set Q, which reads samples in G through the softmax function over the cosine similarity measures.[3] Therefore, the conditional embedding vectors q_i are defined as

$$q_i = f_q([f(x_q^{(i)}), r_i]), \tag{6}$$

$$r_i = \sum_{j=1}^{|G|} a_{ij} g_j, \tag{7}$$

$$a_i = \texttt{softmax}(\sum_{j=1}^{|G|} \langle f(x_q^{(i)}), g_j \rangle y_g^{(j)}). \tag{8}$$

f is the FEN (ResNet [33]). The last layer in f is activated by the `tanh` function. In Eq. 6, f_q is a fully connected network whose output has the same size as $f(x_q^{(i)})$. $[\cdot, \cdot]$ is a concatenation operator between two vectors. The output of f_q is also activated by the `tanh` function. In Eq. 7, a_{ij} represents the j-th component of a_i. Therefore, the conditional embedding vector q_i incorporates all elements in the set G based on the weighted average g_j.

Next, we present the CEN for the set G, which is denoted as CEN_G. g_j is generated by embedding an additional sample with the samples in set G using bidirectional LSTM (Bi-LSTM) [6].

$$g_j = \overrightarrow{h}_j + \overleftarrow{h}_j + f(x_g^{(j)}), \tag{9}$$

$$\overrightarrow{h}_j, \overrightarrow{c}_j = \texttt{LSTM}(f(x_g^{(j)}), \overrightarrow{h}_{j-1}, \overrightarrow{c}_{j-1}), \tag{10}$$

$$\overleftarrow{h}_j, \overleftarrow{c}_j = \texttt{LSTM}(f(x_g^{(j)}), \overleftarrow{h}_{j+1}, \overleftarrow{c}_{j+1}), \tag{11}$$

where $\overrightarrow{h}$ and $\overleftarrow{h}$ are the outputs of forward and backward LSTM, respectively. $\overrightarrow{c}$ and $\overleftarrow{c}$ are the cells of the corresponding LSTM networks. CEN_G is similar to the network in [14] as we also add a skip connection between the input and output.

TABLE 3. Division of training and validation sets in which D, F, and S in the MOT17 dataset represent the DPM, Faster R-CNN, and SDP, respectively.

Dataset	Training set	Validation set
MOT16	02, 04, 10, 11, 13	05, 09
MOT17	02-{D, F, S}, 04-{D, F, S}, 10-{D, F, S}, 11-{D, F, S}, 13-{D, F, S}	05-{D, F, S}, 09-{D, F, S}

A. TRAINING OneShotDA

To train the network, we generated training samples from the training sets in the MOTchallenge datasets [8]. Specifically, we used subsets of the MOT16 and MOT17 datasets (Table 3). We used public detection methods and classified the samples as true detection results and false alarms. True detection results are detection results whose intersection over union (IoU) is greater than the threshold τ_{IoU}, where each ground truth bounding box has at most one true detection result. This is an assignment problem based on the maximum total IoU score. We solved this problem using the Hungarian algorithm [34]. The remaining detection results that were not chosen by the Hungarian method were classified as false alarms. Note that because objects in the dataset are frequently occluded by each other, we filter out small ground truth bounding boxes using non-maximum suppression with an IoU threshold τ_{IoU}^{GT} prior to identifying true detection results.

Next, a training sample is constructed using two consecutive video frames from the video sequence. Let l be a particular frame, then Q and G are the detection sets from l and $l+1$, respectively. Additionally, Q and G can be detection results from $l+1$ and l, respectively. In this manner, we compute loss twice using one query-gallery pair by swapping the two sets. Let L_{CE} be the cross entropy loss, where Q consists of the detection set at l, and $L_{CE'}$ be the loss, where Q consists of the detection set at $l+1$. These losses measure the classification error for the query set. If a query sample is either a false alarm or missing in the gallery set, that sample is not used to compute loss. Note that the size of the query set $|Q|$ will be the batch size and the size of the gallery set $|G|$ will be the number of classes. Therefore, the class size and batch size are not fixed. This can be achieved based on the attention mechanism we adopted.

Additionally, identifying false alarms is crucial because many false alarms are present in detector outputs. To that end, we attach a fully connected layer (TD_clf) with a size of one following the FEN. Therefore, the network takes a feature vector $f(\cdot)$ from the FEN and outputs a prediction p_{TD} that indicates whether or not the input is a true detection result. We then define L_{TD} as the binary cross entropy loss, which measures the classification error between the label of a true detection result and the predicted probability of the true detection result (i.e., p_{TD}).

Finally, the final loss L is defined as

$$L = L_{CE} + L_{CE'} + \lambda_{TD} L_{TD}, \tag{12}$$

where λ_{TD} is the weight for L_{TD}. Therefore, training OneShotDA, including the FEN, CEN, and TD_clf, is accomplished using a single training sample by minimizing L in one step.

B. CORE COMPONENTS OF OneShotDA

As discussed above, we use a one-shot learning framework for MOT because it has the ability to unravel the association problem between an unseen object and existing tracks. In this work, our one-shot framework was derived from MatchingNet [14] with many modifications. In this section, we summarize the role of each component in OneShotDA, namely the FEN, CEN (CEN_Q and CEN_G), and TD_clf.

- FEN: This network generates a feature vector for a given input (i.e., $x_q^{(i)}$ and $x_g^{(j)}$), where $x_q^{(i)} \in Q$ and $x_g^{(j)} \in G$.

[3] We use the similarity measure of an inner product space. Because similarity can be influenced by not only the direction of the inner product, but also the norm of the feature vector, we activate all feature vectors with a `tanh` function.

We use the ResNet family for this network and investigate the performance of each residual network in terms of tracking accuracy. (section V-C)

- CEN: This network further embeds a feature vector that is outputted by the FEN to generate more robust features. The features outputted by this network are sufficient for distinguishing samples from each other, even when they are close to each other. (section V-C)
  - CEN_Q: CEN for the set Q. This network helps a sample in Q read entire elements of G and outputs a conditional embedding vector for a given input.
  - CEN_G: CEN for the set G. A conditional embedding vector for each element in G is generated using bidirectional LSTM with the set G itself.
- TD_clf: This network takes a feature vector $f(\cdot)$ and outputs a prediction p_{TD}, indicating whether or not the input is a true detection result.

V. EXPERIMENTS AND ANALYSIS

In this section, we analyze the tracking performance of the OneShotDA tracker on the MOTChallenge datasets (MOT16 and MOT17). Additionally, ablation analysis is performed to identify the best hyperparameter settings.

A. IMPLEMENTATION DETAILS

In all the experiments, the values of the parameters τ_D, τ_{IOU}, and τ_{IOU}^{GT} were $30, 0.334$, and 0.5, respectively. τ_{miss} was set to $2.3\texttt{fps}$, where $\texttt{fps}$ is the frames per second for each sequence. Additionally, we have exhaustively searched $C_{\texttt{app}}$, $C_{\texttt{kin}}$, and C_{β} to find good parameters, and decided to set to $0.1, 0.1$, and 2.0, respectively. Input images were resized to 288×96 pixels and normalized to the range of 0 to 1. We also augmented the training set with uniform random rotation in the angle range of $[-8.5, 8.5]$, random horizontal flipping, and random brightness changes. As mentioned previously, the ResNet family was used for the FEN. In the ablation analysis section, we investigate the performances of the ResNet family, ResNet34, ResNet50, and ResNet101, as well as their output sizes (feature vectors). Note that the size of a feature vector determines the size of the Bi-LSTM in CEN_Q because the output cell of Bi-LSTM and the corresponding feature vector are added inside CEN_Q. The ResNets were initialized with weights pre-trained on the ImageNet datasets, except for the final fully connected layers. The final layer was replaced with our feature embedding layer for various output sizes (see the ablation analysis section). The stochastic gradient descent optimizer with a momentum 0.9 was used for training all the networks, and the initial learning rate was set to 10^{-4}. The learning rate decreased after every 3000 iterations based on exponential decay with a decay rate of 0.95 until the minimum learning rate of 10^{-7} was achieved. λ_{TD} was set to 1.

B. MOTCHALLENGE DATASETS AND METRICS

In this study, we used the MOTChallenge datasets [8] to train our OneShotDA network and test the tracking performance of the OneShotDA tracker. The training and validation dataset separation is detailed in Table 3. The test set of MOT17 includes a total of seven sequences, each of which comes with three sets of public detection results. These three public sets come from different detectors, namely the deformable part model (DPM) [5], faster-RCNN [9], and scale-dependent pooling (SDP) [10]. The MOT16 test set consists of the same sequences as those in MOT17, but it only contains the DPM detection set. It must be noted that the ground truth labels are not shared for the same sequences across MOT16 and MOT17.

The metrics used for measuring tracker performance are the same as those used in [8]. MOT accuracy (MOTA) measures performance by aggregating three error sources, namely false positives, missed targets, and identity switches. IDF1 [35] computes the ratio of correctly identified detection results over the average number of ground truth and computed detection results. MOTA and IDF1 are considered the main criteria for tracker performance. We also report mostly tracked (MT) objects, mostly lost (ML) objects, the total number of false positives (FP), false negatives (FN), and identity switches (IDsw), and the total number of times a trajectory is fragmented (Frag).

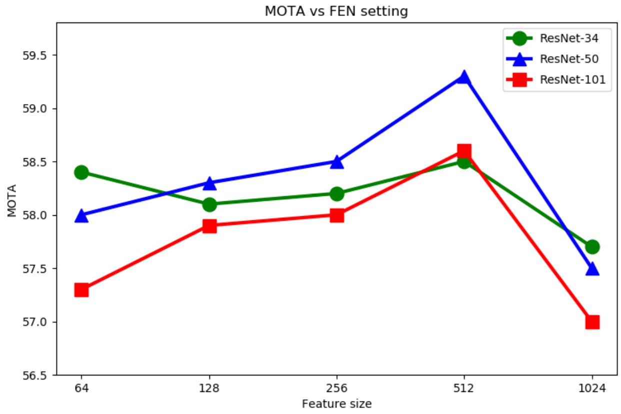


FIGURE 2. Ablation study on different FEN architectures. The x-axis is the feature size and the y-axis is the MOTA score.

C. ABLATION ANALYSIS

We conducted ablation studies with different hyperparameter settings to achieve optimal performance on the validation set and its subsets. In Figure 2, we consider the ResNet family and corresponding output size for the subsets of our validation set (i.e., MOT17-05-{F,S} and MOT17-09-{F,S}). It is important to note that not only does the architecture of the FEN affect the tracking performance, but the size of the feature vectors is also a crucial aspect for performance. In this study, all networks were trained for 5 epochs. The CENs were consistently initialized for each setting and retrained from the beginning. The sizes of the feature vectors were sampled from a logarithmic scale ranging from 64 to 1024 (i.e., {64, 128, 256, 512, 1024}). The analysis results for various FEN settings are presented in Figure 2. According to the

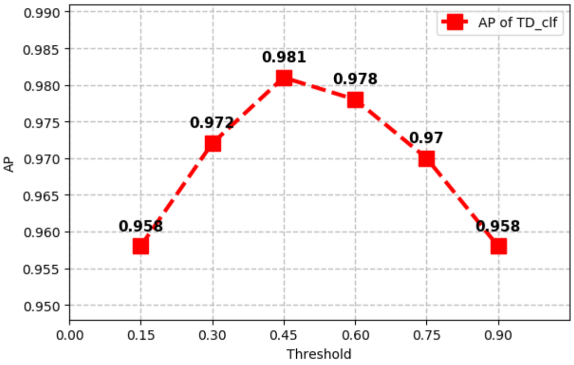
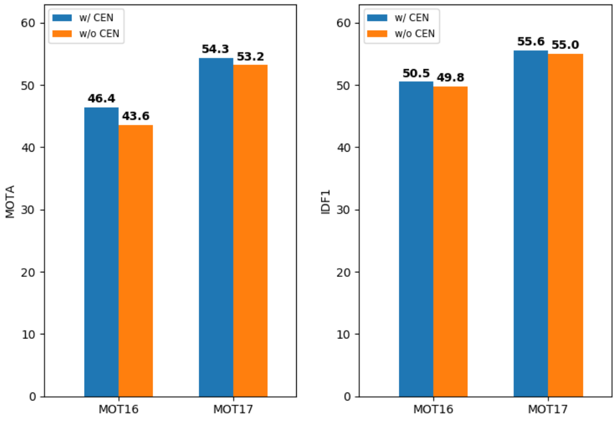


FIGURE 3. **Analysis of p_{TD} for the validation set.**

results in Figure 2, we selected the ResNet50 with 512 outputs for our FEN. Our model achieves the maximum MOTA (59.3) using ResNet50-512. Furthermore, it is important to note that the model seems to suffer from overfitting when the parameter size is greater than that in ResNet50 (25.6 M) or the feature size is greater than 512. To determine if this assumption was correct, we trained a ResNet101-1024 network with additional epochs because the low performance of such a large model could potentially result from a low convergence rate based on its large parameter size. However, we found that the MOTA of the large model continued to decrease or remain constant while its training loss consistently decreased during continued training.



(a) MOTA

(b) IDF1

FIGURE 4. **w/ CEN vs w/o CEN, evaluated on the validation set. The tested model is the ResNet50 with 512 output neurons.**

Next, we investigated the contribution of the CEN by measuring its performance in terms of MOTA and IDF1 on the validation set. Figure 4(a) presents comparative results in terms of MOTA, and Figure4(b) presents comparative results in terms of IDF1. It can be inferred from Figure 4(a) that our CEN efficiently improves the performance on the MOT16 dataset. OneShotDA with the CEN improves the tracking performance by 2.8% (MOTA) and 1.1% (IDF1). This is because OneShotDA, without the CEN, struggles to associate objects identified by the DPM detector whose outputs are much noisier in comparison with those of Faster R-CNN and SDP. Additionally, performance is consistently improved by the CEN for the MOT17 dataset (Figure 4(b)).

The performance in terms of predicting true detection results (p_{TD}) was also investigated. This analysis helped us in selecting a good threshold value for identifying true detection results in the detection outputs. Figure 3 presents the average-precision (AP) score for each threshold value. The values are evenly distributed at intervals of 0.15. We achieve an AP of 0.981 at a threshold value 0.45. Therefore, we chose 0.45 as the threshold value for p_{TD} when testing our OneShotDA tracker on the test set.

D. MOT PERFORMANCE ANALYSIS

In this experiment, we used a ResNet50-512 network as the FEN and trained the network with additional epochs. Our network was trained for a total of 8 epochs.

We first present the performance analysis of our network as a binary classifier. Each prediction is considered the output of a classification representing how likely it is for two objects to be assigned the same identity. As shown in Figure 5, the average precision of the precision-recall curve is 0.8957. The classification results for the validation set indicate that OneShotDA is trained properly and makes precise association predictions.

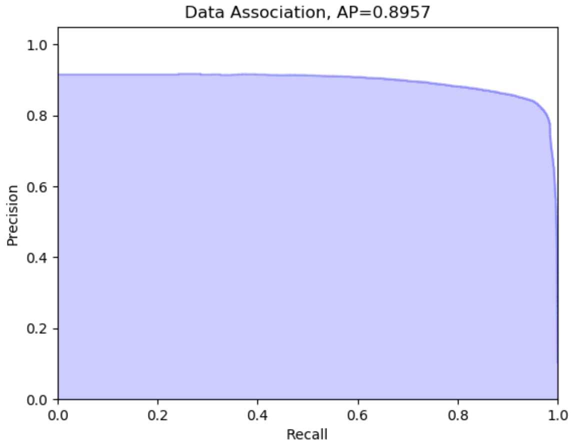


FIGURE 5. **Precision-recall curve.**

Finally, we present performance comparisons between the OneShotDA tracker and existing state-of-the-art methods such as HCC [17], LMP [19], GCRA [36], KCF16 [37], MOTDT [30], JBNOT [20], eHAF17 [39], TLMHT [22], EAGS16 [38], MHT_DAM [13], MHT_bLSTM [23], and EDMT17 [40]. These methods were evaluated on the MOTChallenge server.[4] To provide a reasonable comparison, only officially published and peer-reviewed entries in the MOT16 and MOT17 benchmarks were considered. Additionally, we collected MHT-based trackers, categorized as online

[4] https://motchallenge.net/

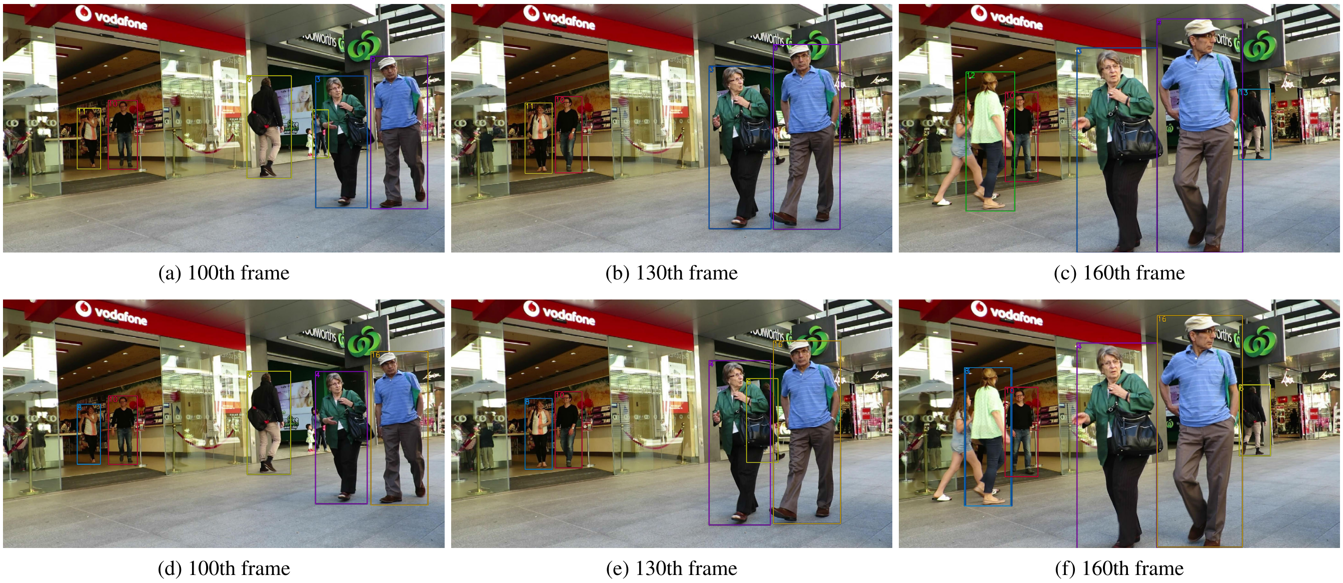

(a) 100th frame
(b) 130th frame
(c) 160th frame
(d) 100th frame
(e) 130th frame
(f) 160th frame

FIGURE 6. Tracking results on the MOT17-09-FRCNN dataset. In the top row, (a)–(c), present the results for JBNOT [20] and in the bottom row, (d)–(f) present our results for the same sequence. The man wearing a black jacket in the 100th frame is consistently tracked by our tracker. However, JBNOT fails to track the man because he is occluded by other objects.

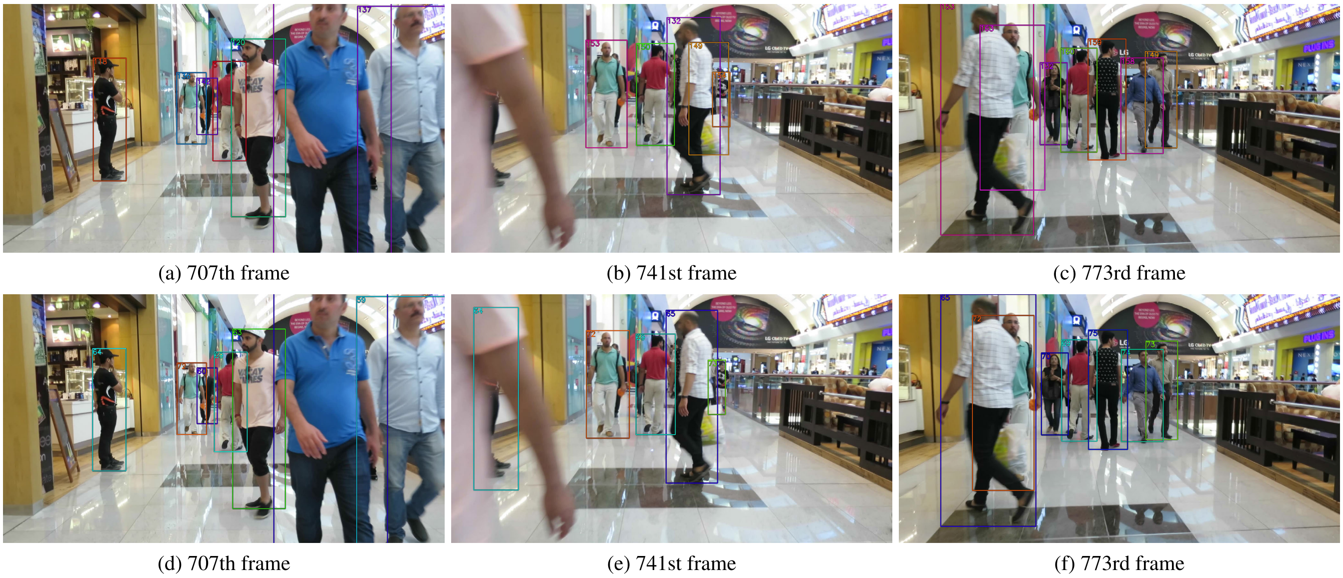
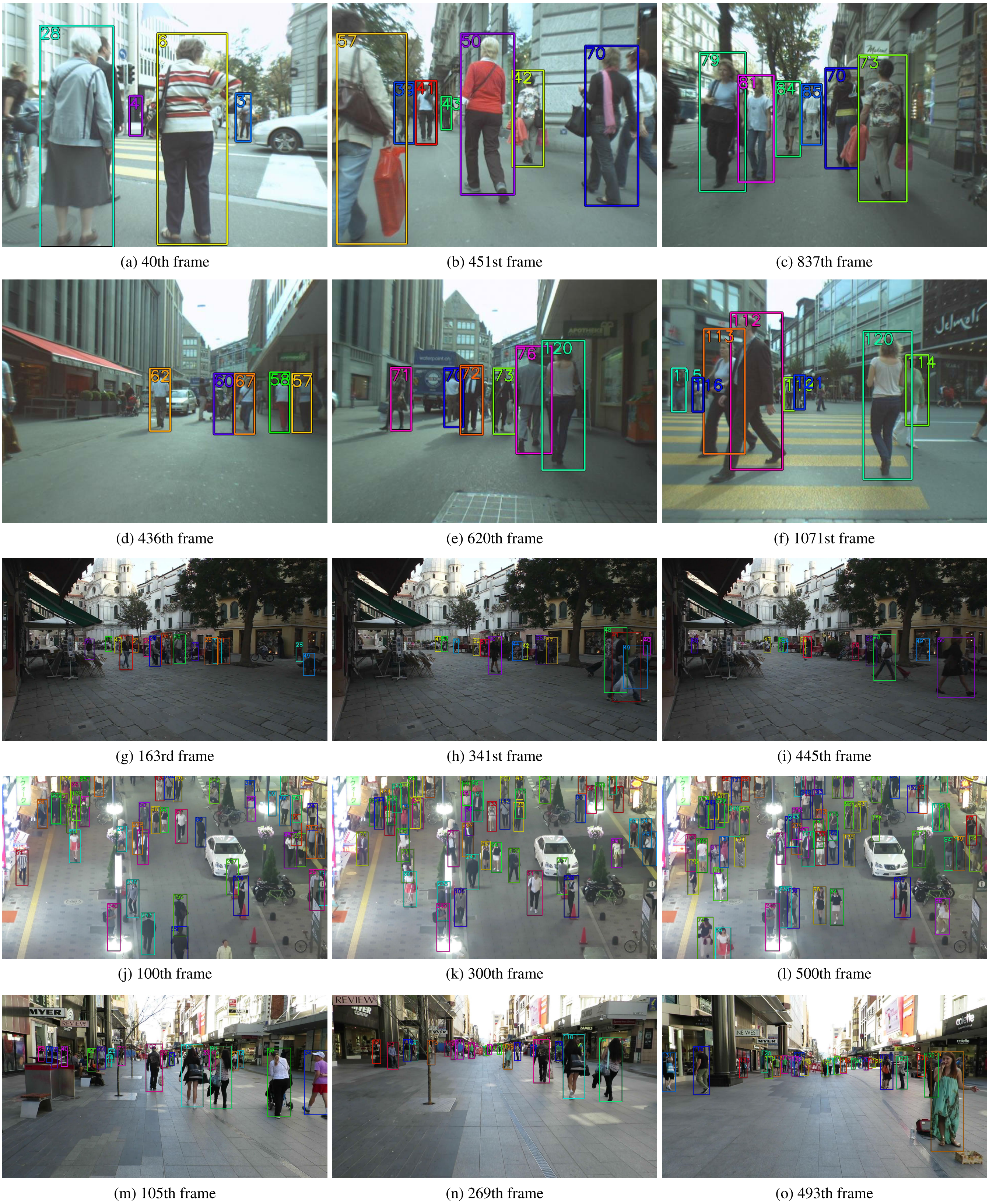

(a) 707th frame
(b) 741st frame
(c) 773rd frame
(d) 707th frame
(e) 741st frame
(f) 773rd frame

FIGURE 7. Tracking results for the MOT17-11-SDP dataset. In the top row, (a)–(c) present the results for JBNOT [20]. In the bottom row, (d)–(f) present our results for the same sequence. After the occlusion occurs, many ID-switches take place in the JBMOT tracker, but our tracker consistently tracks objects, even during occlusions (*e.g.,* ID-64, ID-65, ID-69, and ID-72).

trackers in this study, for the purpose of simple comparisons. Trackers were grouped according to their tracking mode (offline and online). In Tables 4 and 5, our method exhibits performance comparable to those of existing state-of-the-art methods. It is noteworthy that OneShotDA ranks first among online trackers on the MOT17 benchmark. Our tracker outperforms all other online trackers in the MOT17 group by 0.5% if we compare it with MOTDT. However, our methods did not outperform the JBNOT, the state-of-the-art offline method in MOT17.

Our tracker seems to prefer DNN-based detectors to traditional detectors based on the fact that it ranks first among online trackers on the MOT17 dataset but ranks lower on the MOT16 dataset. We determined that our simple implementation of the function $\mathtt{app}(\cdot)$ for tracks could degrade model performance on the MOT16 dataset. The function $\mathtt{app}(T_l^{(i)})$ simply retrieves an image patch of $T_l^{(i)}$, which is an image of the latest update with a detection result. Because the detector outputs in MOT16 are comparatively noisy, we believe this function is insufficient for returning an image feature



(a) 40th frame (b) 451st frame (c) 837th frame

(d) 436th frame (e) 620th frame (f) 1071st frame

(g) 163rd frame (h) 341st frame (i) 445th frame

(j) 100th frame (k) 300th frame (l) 500th frame

(m) 105th frame (n) 269th frame (o) 493th frame

FIGURE 8. Qualitative results. (a)-(c): MOT16-05, (d)-(f): MOT16-06, (g)-(i): MOT17-01-FRCNN, (j)-(l): MOT17-03-SDP, (m)-(o): MOT17-07-SDP.

TABLE 4. Results on the MOT16 dataset. We grouped methods according to their tracking mode (offline and online). The red numbers for each metric represent the best performance (offline/online) and the blue numbers represent the second best performance (online). The methods marked with * are MHT-based trackers. (Accessed on August 1, 2019.)

Tracker	Mode	MOTA↑	IDF1↑	MT↑	ML↓	FP↓	FN↓	IDsw↓	Frag↓	FPS↑
HCC [17]	offline	49.3	50.7	17.8	39.9	5,333	86,795	391	535	0.8
LMP [19]	offline	48.8	51.3	18.2	40.1	6,654	86,245	481	595	0.5
GCRA [36]	offline	48.2	48.6	12.9	41.1	5,104	88,586	821	1,117	2.8
KFC16 [37]	online	48.8	47.2	15.8	38.1	5,875	86,567	906	1,116	0.1
MOTDT [30]	online	47.6	50.9	15.2	38.3	9,253	85,431	792	1,858	20.6
TLMHT* [22]	online	48.7	55.3	15.7	44.5	6,632	86,504	413	642	4.8
EAGS16* [38]	online	47.4	50.1	17.3	42.7	8,369	86,931	575	913	197.3
MHT_DAM* [13]	online	45.8	46.1	16.2	43.2	6,412	91,758	590	781	0.8
MHT_bLSTM* [23]	online	42.1	47.8	14.9	44.4	11,637	93,172	753	1,156	1.8
Ours	online	47.0	50.1	16.5	41.8	7,901	88,179	627	945	3.5

TABLE 5. Results on the MOT17 dataset. We grouped methods according to their tracking mode (offline and online). The red numbers for each metric represent the best performance (offline/online) and the blue numbers represent the second best performance (online). The methods marked with * are MHT-based trackers. (Accessed on August 2, 2019.)

Tracker	Mode	MOTA↑	IDF1↑	MT↑	ML↓	FP↓	FN↓	IDsw↓	Frag↓	FPS↑
JBNOT [20]	offline	52.6	50.8	19.7	35.8	31,572	232,659	3,050	3,792	5.4
eHAF17 [39]	offline	51.8	54.7	23.4	37.9	33,212	236,772	1,834	2,739	0.7
MOTDT [30]	online	50.9	52.7	17.5	35.7	24,069	250,768	2,474	5,317	18.3
TLMHT* [22]	online	50.6	56.5	17.6	43.4	22,213	255,030	1,407	2,079	2.6
MHT_DAM* [13]	online	50.7	47.2	20.8	36.9	22,875	252,889	2,314	2,865	0.9
EDMT17* [40]	online	50.0	51.3	21.6	36.3	32,279	247,297	2,264	3,260	0.6
MHT_bLSTM* [23]	online	47.5	51.9	18.2	41.7	25,981	268,042	2,069	3,124	1.9
Ours	online	51.4	54.0	21.2	37.3	29,051	243,202	2,118	3,072	3.4

typifying a track. Because the function $\mathtt{app}(\cdot)$ can be of any type, incremental updates of appearance features can resolve this issue.

We further examined the performance of our method by incorporating qualitative analysis. In Figures 6 and 7, the robustness of our tracker against ID-switches is analyzed via frame-by-frame investigation. We compare the results to JBNOT [20], which achieved the top rank on the MOT17 dataset in terms of MOTA but with inferior ID-switch performance compared to our tracker (Table 5). In Figure 6, the top row presents partial results for JBNOT on the MOT17-09-FRCNN dataset, and the bottom row presents the results for our method on the same sequence. This figure suggests that our tracker consistently tracks the man with the black jacket (ID-6 in our results). However, this object is lost by JBNOT, which initiates a new track with ID-13 after the object is occluded by other objects. Figure 7 presents the results on the MOT17-11-SDP dataset, where the top row represents JBNOT and the bottom row represents our model. These results demonstrate the robustness of our tracker against ID-switches. Our tracker consistently tracks objects, even during heavy occlusions (*e.g.*, ID-64, ID-65, ID-69, and ID-72 in our results), while many switches occur for the JBNOT tracker. Finally, we present the qualitative results in Figure 8.

VI. CONCLUSION AND FUTURE WORK

In this study, a novel data association mechanism called OneShotDA was presented and integrated with MHT to perform online MOT. The proposed network classifies existing tracks by pointing to corresponding detection results using an attention mechanism. OneShotDA can solve the data association problem of MOT and identify false positives in detector outputs. To train the proposed network, we employed a novel training strategy tailored for one-shot learning that is suitable for data association tasks. We also demonstrated how training samples can be generated from MOTChallenge datasets. In a series of experiments, our OneShotDA tracker delivered performance comparable to the performances of existing state-of-the-art methods. Additionally, our tracker ranked first among online trackers on the MOT17 dataset. For future work, we plan to devise an incremental learning method to learn the appearance function of a track (i.e., $\mathtt{app}(T_l^{(i)})$ in this work). In online tracking mode, detection results come in a sequential order, meaning incremental learning can help our tracker in updating learned appearance features based on newly associated detection results.

REFERENCES

[1] L. Chen, H. Ai, C. Shang, Z. Zhuang, and B. Bai, "Online multi-object tracking with convolutional neural networks," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2017, pp. 645–649.

[2] A. Sadeghian, A. Alahi, and S. Savarese, "Tracking the untrackable: Learning to track multiple cues with long-term dependencies," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 300–311.

[3] K. Yoon, D. Kim, Y.-C. Yoon, and M. Jeon, "Data association for multi-object tracking via deep neural networks," *Sensors*, vol. 19, no. 3, p. 559, Jan. 2019.

[4] N. Wojke, A. Bewley, and D. Paulus, "Simple online and realtime tracking with a deep association metric," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2017, pp. 3645–3649.

[5] D. Forsyth, "Object detection with discriminatively trained part-based models," *Computer*, vol. 47, no. 2, pp. 6–7, Feb. 2014.

[6] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[7] S. Blackman and R. Popoli, *Design and Analysis of Modern Tracking Systems*. Norwood, MA, USA: Artech House, 1999.

[8] A. Milan, L. Leal-Taixe, I. Reid, S. Roth, and K. Schindler, "MOT16: A benchmark for multi-object tracking," 2016, *arXiv:1603.00831*. [Online]. Available: http://arxiv.org/abs/1603.00831

[9] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017.

[10] F. Yang, W. Choi, and Y. Lin, "Exploit all the layers: Fast and accurate CNN object detector with scale dependent pooling and cascaded rejection classifiers," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2129–2137.

[11] L. Zhang, Y. Li, and R. Nevatia, "Global data association for multi-object tracking using network flows," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2008, pp. 1–8.

[12] H. Pirsiavash, D. Ramanan, and C. C. Fowlkes, "Globally-optimal greedy algorithms for tracking a variable number of objects," in *Proc. CVPR*, Jun. 2011, pp. 1201–1208.

[13] C. Kim, F. Li, A. Ciptadi, and J. M. Rehg, "Multiple hypothesis tracking revisited," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 4696–4704.

[14] O. Vinyals, C. Blundell, T. Lillicrap, and D. Wierstra, "Matching networks for one shot learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 3630–3638.

[15] F. Sung, Y. Yang, L. Zhang, T. Xiang, P. H. S. Torr, and T. M. Hospedales, "Learning to compare: Relation network for few-shot learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 1199–1208.

[16] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 4077–4087.

[17] L. Ma, S. Tang, M. J. Black, and L. Van Gool, "Customized multi-person tracker," in *Proc. Asian Conf. Comput. Vis.* Cham, Switzerland: Springer, 2018, pp. 612–628.

[18] S. Tang, B. Andres, M. Andriluka, and B. Schiele, "Subgraph decomposition for multi-target tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 5033–5041.

[19] S. Tang, M. Andriluka, B. Andres, and B. Schiele, "Multiple people tracking by lifted multicut and person re-identification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 3539–3548.

[20] R. Henschel, Y. Zou, and B. Rosenhahn, "Multiple people tracking using body and joint detections," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, Jun. 2019.

[21] R. Henschel, L. Leal-Taixe, D. Cremers, and B. Rosenhahn, "Fusion of head and full-body detectors for multi-object tracking," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2018, pp. 1428–1437.

[22] H. Sheng, J. Chen, Y. Zhang, W. Ke, Z. Xiong, and J. Yu, "Iterative multiple hypothesis tracking with tracklet-level association," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 29, no. 12, pp. 3660–3672, Dec. 2019.

[23] C. Kim, F. Li, and J. M. Rehg, "Multi-object tracking with neural gating using bilinear LSTM," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 200–215.

[24] S. Sun, N. Akhtar, H. Song, A. S. Mian, and M. Shah, "Deep affinity network for multiple object tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, to be published.

[25] H. Yang, J. Li, J. Liu, Y. Zhang, X. Wu, and Z. Pei, "Multi-pedestrian tracking based on improved two step data association," *IEEE Access*, vol. 7, pp. 100780–100794, 2019.

[26] Z. He, J. Li, D. Liu, H. He, and D. Barber, "Tracking by animation: Unsupervised learning of multi-object attentive trackers," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 1318–1327.

[27] W. Zhang, H. Zhou, S. Sun, Z. Wang, J. Shi, and C. C. Loy, "Robust multi-modality multi-object tracking," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2019, pp. 2365–2374.

[28] L. Wen, D. Du, S. Li, X. Bian, and S. Lyu, "Learning non-uniform hypergraph for multi-object tracking," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, Jul. 2019, pp. 8981–8988.

[29] P. Voigtlaender, M. Krause, A. Osep, J. Luiten, B. B. G. Sekar, A. Geiger, and B. Leibe, "MOTS: Multi-object tracking and segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 7942–7951.

[30] L. Chen, H. Ai, Z. Zhuang, and C. Shang, "Real-time multiple people tracking with deeply learned candidate selection and person re-identification," in *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, vol. 5, Jul. 2018, p. 8.

[31] A. Milan, S. H. Rezatofighi, A. Dick, I. Reid, and K. Schindler, "Online multi-target tracking using recurrent neural networks," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017, pp. 1–8.

[32] G. Koch, R. Zemel, and R. Salakhutdinov, "Siamese neural networks for one-shot image recognition," in *Proc. ICML Deep Learn. Workshop*, vol. 2, 2015.

[33] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[34] H. W. Kuhn, "The hungarian method for the assignment problem," *Nav. Res. Logistics*, vol. 52, no. 1, pp. 7–21, Feb. 2005.

[35] E. Ristani, F. Solera, R. Zou, R. Cucchiara, and C. Tomasi, "Performance measures and a data set for multi-target, multi-camera tracking," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 17–35.

[36] C. Ma, C. Yang, F. Yang, Y. Zhuang, Z. Zhang, H. Jia, and X. Xie, "Trajectory factory: Tracklet cleaving and re-connection by deep siamese bi-GRU for multiple object tracking," in *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, Jul. 2018, pp. 1–6.

[37] P. Chu, H. Fan, C. C. Tan, and H. Ling, "Online multi-object tracking with instance-aware tracker and dynamic model refreshment," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Jan. 2019, pp. 161–170.

[38] H. Sheng, X. Zhang, Y. Zhang, Y. Wu, J. Chen, and Z. Xiong, "Enhanced association with supervoxels in multiple hypothesis tracking," *IEEE Access*, vol. 7, pp. 2107–2117, 2019.

[39] H. Sheng, Y. Zhang, J. Chen, Z. Xiong, and J. Zhang, "Heterogeneous association graph fusion for target association in multiple object tracking," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 29, no. 11, pp. 3269–3280, Nov. 2019.

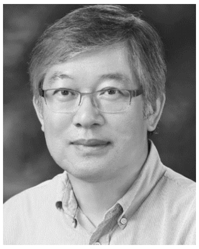
[40] J. Chen, H. Sheng, Y. Zhang, and Z. Xiong, "Enhancing detection model for multiple hypothesis tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jul. 2017, pp. 18–27.

KWANGJIN YOON received the B.S. degree in computer science from Sahmyook University, Seoul, South Korea, in 2009, the M.S. degree in image engineering from Chung-Ang University, Seoul, in 2011, and the Ph.D. degree in electrical engineering and computer science from the Gwangju Institute of Science and Technology, Gwangju, South Korea, in 2019. He is currently a Research Scientist with SI-Analytics. His research interests include multiobject tracking, computer vision, and deep learning.

JEONGHWAN GWAK received the Ph.D. degree in machine learning and artificial intelligence from the Gwangju Institute of Science and Technology (GIST), Gwangju, South Korea, in 2014. From 2002 to 2007, he worked for several companies and research institutes as a Researcher and a Chief Technician. From 2014 to 2016, he worked as a Postdoctoral Researcher with GIST. From 2016 to 2017, he worked as a Research Professor. From 2017 to 2019, he was a Research Professor with the Department of Radiology, Biomedical Research Institute, Seoul National University Hospital, Seoul, South Korea. In 2019, he joined the Korea National University of Transportation, where he is currently the Director of the Applied Machine Intelligence Laboratory. His current research interests include deep learning, computer vision, image and video processing, evolutionary computation, optimization, and relevant applications of medical and visual surveillance systems. He has served as a PC/TPC member or Chairman for many artificial intelligence, machine learning, and computer vision conferences. He is an Active Associate Editor of IEEE Access and *PLOS One*, a Guest Editor for *Applied Sciences* and IJCVR.

YOUNG-MIN SONG received the B.S. degree in computer science and engineering from Chungnam University, Daejeon, South Korea, in 2013, and the M.S. degree in information and communications from the Gwangju Institute of Science and Technology (GIST), Gwangju, South Korea, in 2015, where he is currently pursuing the Ph.D. degree in electrical engineering and computer science. His research interests include multiobject tracking and data fusion.

YOUNG-CHUL YOON received the B.S. degree in electronics and communications engineering from Kwangwoon University, Seoul, South Korea, and the M.S. degree in electrical engineering and computer science from the Gwangju Institute of Science and Technology, Gwangju, South Korea, in 2019. He is currently working as a Computer Vision Researcher with LG Electronics. His research interests include multiobject tracking and deep learning.



MOONGU JEON received the B.S. degree in architectural engineering from Korea University, Seoul, South Korea, in 1988, and the M.S. and Ph.D. degrees in computer science and scientific computation from the University of Minnesota, Minneapolis, MN, USA, in 1999 and 2001, respectively. As a Postgraduate Researcher, he worked on optimal control problems at the University of California at Santa Barbara, Santa Barbara, CA, USA, from 2001 to 2003. Then, he moved to the National Research Council of Canada, where he worked on the sparse representation of high-dimensional data and image processing until 2005. In 2005, he joined the Gwangju Institute of Science and Technology, Gwangju, South Korea, where he is currently a Full Professor with the School of Electrical Engineering and Computer Science. His current research interests include machine learning, computer vision, and artificial intelligence.