IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received September 24, 2020, accepted September 30, 2020, date of publication October 6, 2020, date of current version October 16, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3028770

Online Multiple Object Tracking Using Rule Distillated Siamese Random Forest

JIMI LEE, SANGWON KIM, AND BYOUNG CHUL KO, (Member, IEEE)
Department of Computer Engineering, Keimyung University, Daegu 42601, South Korea

Corresponding author: Byoung Chul Ko (niceko@kmu.ac.kr)

This work was supported by the National Research Foundation of Korea (NRF) funded by the Ministry of Education through the Basic Science Research Program under Grant 2019R1I1A3A01042506.

ABSTRACT In a multiple object tracking (MOT) system, an association check between the tracker and detected objects is an important factor in determining the tracking performance. Siamese convolution neural network (CNN) is the most popular data association method in MOT owing to its good matching performance and network sharing support. However, it is unsuitable for real-time online tracking in low-end systems because numerous parameters and operations are still required. In this article, instead of a CNN, we propose using a SiameseRF algorithm which combines Siamese structure and random forest (RF), enabling high-speed learning and classification. SiameseRF has a shared-rule based Siamese structure rather than shared weight, which improves the matching performance and solves existing slow CNN-based tracking issues. During the learning process, the shared RFs consisting of tree rules are learned in the directions of increasing similarity to the positive pair {anchor, positive} and increasing difference between the negative pair {anchor, negative}. However, because many rules that make up SiameseRF remain a burden for online processing, this study proposes an additional rule distillation algorithm to effectively remove redundant and unimportant rules causing an overfitting with SiameseRF. This reduction in the number of rules reduces the processing time and number of parameters in the rule distilled SiameseRF. In experiments conducted on MOT benchmark datasets, our proposed 30% rule distilled SiameseRF achieved up to a 1.12-times faster speed and a 1.13-times higher compression rate than basic SiameseRF while maintaining a similar or somewhat better tracking performance than other state-of-the-art CNN-based MOT algorithms.

INDEX TERMS Multiple object tracking, data association, Siamese CNN, SiameseRF, rule distillation.

I. INTRODUCTION

Although object tracking has been a long-studied subject in the field of computer vision, many problems in this area are yet to be resolved. Object tracking can be divided into single object tracking (SOT) and multiple object tracking (MOT) depending on the number of objects to be tracked. The purpose of SOT is to find and follow the position of a single object in a continuous video sequence. The tracking model used for SOT is relatively simple and fast compared to MOT, although MOT is essential for various tracking technologies such as video surveillance, autonomous driving, a human-computer-interface (HCI), and augmented reality (AR).

MOT can be divided into offline and online versions according to the tracker optimization method. Offline MOT methods use the past and future frames at the same time to create a tracking path, whereas online MOT methods utilize only the information available around the current frame. Many techniques using a deep neural network (DNN) have recently been developed as an alternative to conventional techniques for offline MOT, such as a conditional random field (CRF) [1], and have demonstrated a better performance in comparison with the previous learning methods [2]. Although a long-term appearance model using features from a DNN [3], DeepMatching [4], and a quadruplet convolutional neural network (CNN) [5] have been proposed, offline tracking is unsuitable for real-time object monitoring or other applications because all frames must be considered to verify the tracking path [6], [7]. In online MOT, a Kalman [8] or particle [9] filter based method has mainly been used, although DNN-based methods [6], [10]–[12] have also recently been applied.

Offline and online MOT are commonly based on the tracking-by-detection (TBD) paradigm. The TBD method consists of a detection step used for locating targets in each frame of a single video, as well as a data association step

The associate editor coordinating the review of this manuscript and approving it for publication was Mohammad Shorif Uddin.

for matching the detected objects to the targets and linking them to the corresponding trajectories. With TBD, Faster R-CNN [13], ResNet [14], YOLOv3 [15], and CornerNet [16] are mainly used as the object detection algorithms, and have an important effect on the performance level. However, regardless of how good the detection method is, if an object is missed or an inaccurate object is detected owing to an occlusion of the object or camera shaking, the tracking performance can significantly deteriorate. Therefore, various data association methods have been proposed to compensate for the inaccuracy of MOT detection. In particular, real-time tracking in MOT is closely related to the efficiency of the data association.

Although the greedy bipartite assignment [17] and optimal Hungarian [18] algorithms are traditional techniques for determining the data association, Siamese CNN [10], [5], [19]–[23] based associations have also received significant interest for real-time tracking. During the learning process, a Siamese CNN applies the same network to the detection and tracker and calculates the similarity based on the difference in the output feature. Therefore, a Siamese CNN does not need to maintain a separate network structure and has the advantage of a fast tracking.Although a Siamese structure shows a good matching performance between objects, the shared network for similarity matching still has a large number of hyper parameters and a slow tracking speed owing to the complex network structure when combined with a CNN [23]. Therefore, Siamese CNN-based MOT methods may have limited feasibility for real-time tracking in a real-world environment.

A. CONTRIBUTIONS OF THIS STUDY

In this study, in contrast to existing methods, we do not use a CNN-based Siamese structure to build an efficient joint learning framework in terms of the MOT performance and tracking speed. Instead, we propose a Siamese random forest (RF)[1] framework [24]; this framework combines an accurate RF with a Siamese structure that has high-speed learning and classification. However, because the RF structure is also a combination of decision trees composed of a number of rules, it is necessary to reduce the numbers of parameters and operations to improve the speed for real-time operations in low-end devices. In this study, we propose an additional rule distillation method that can effectively remove unimportant and redundant rules causing an overfitting with the proposed SiameseRF structure. Thus, rule-distilled SiameseRF can reduce the processing time and parameter storage requirement by reducing the number of rules.

The main contributions of this article are summarized as follows.

- A shared-rule based SiameseRF framework is recommended that combines an accurate RF with a Siamese structure.
- A rule distillation method is proposed that can remove duplicate or relatively inefficient rules from a tree by calculating the rule contribution to the basic SiameseRF.
- Condensed features are suggested from output feature maps of the first and second convolution layers as the local appearance feature of the object for similarity measurements.
- K-fold cross validation is adopted to determine the optimal numbers of rules and parameters while reducing the risk of an overfitting of the model.
- A frame-by-frame data association check is applied between the tracking and detecting through a SiameseRF -based similarity probability.
- It is proven that the MOT accuracy of the rule-distilled SiameseRF is similar or to some extent better than that of the basic SiameseRF and other state-of-the-art CNN-based MOT methods, and the tracking speed is the fastest despite using a CPU.

Figure 1 shows the overall architecture of the online MOT system using the rule-distilled SiameseRF proposed in this study. First, YOLOv3 [15] based on DarkNet53 is used to detect objects in the input image. Condensed appearance features (CAF) are then extracted from the feature maps of two layers of DarkNet53 corresponding to the object region at the same time as the object detection (Figure 1 (a)). In Figure 1 (b), the CAF is computed from the detection and tracker pairs, respectively, using the same structure as shown in Figure 1 (a). The CAF distance computed between two CAFs (D_t, T_t) is applied to rule-sharing SiameseRF (Figure 1 (c)), and a similarity score is used for an association check (Figure 1 (d)). For real-time MOT, by applying the proposed rule distillation technique, which removes unimportant rules constituting SiameseRF (Figure 1(e)), we can significantly reduce the computation speed of an online association while maintaining the matching accuracy. The state of the tracker is updated by considering the matched detection (Figure 1 (f)). The proposed SiameseRF structure is extremely efficient in reducing the processing time for MOT because it is impossible to simultaneously detect objects from a single-shot detector and the CAF.

Before discussing the related studies and proposed algorithm, we summarize the abbreviations for the main terminologies used in this article in Table 1.

The remainder of this article is structured as follows. In Section II, MOT-related studies focusing on Siamese-based tracking are reviewed. In Section III, we present the details of our proposed method in terms of its feature contribution and rule elimination. In Section IV, a comprehensive evaluation of the proposed method is provided based on the results of various experiments. Finally, some concluding remarks are given in Section V.

II. RELATED STUDIES

In studies on MOT tracking, long-term appearance models using features from a DNN [3], DeepMatching [4], and a quadruplet CNN [5] have demonstrated a better tracking

[1] The short version of SiameseRF was published at the MOTChallenge Workshop of CVPR 2020.

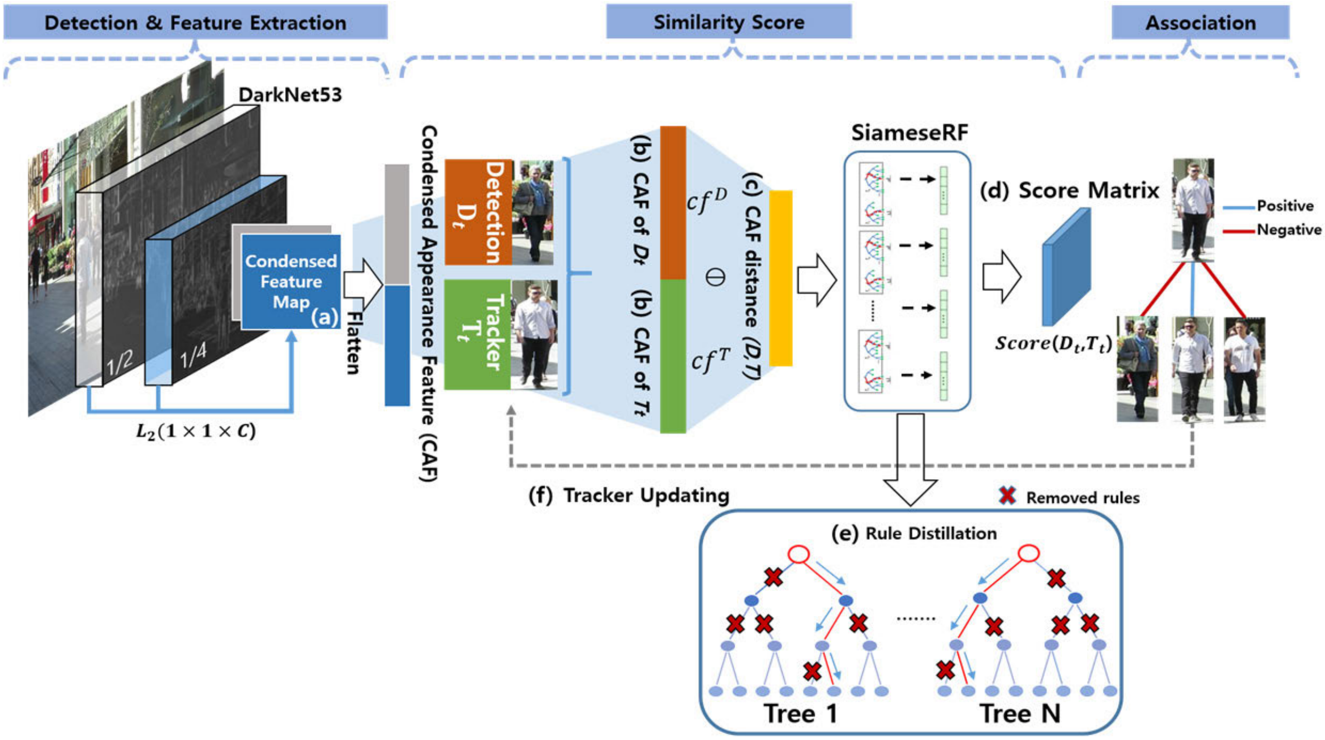


FIGURE 1. Procedure of rule distille SiameseRF learning. (a) Condensed features are extracted from the feature maps of two layers of DarkNet 53 from the detection and target pair, (b) the condensed appearance feature (CAF) is computed from detection and tracker, and (c) the CAF distance vector is estimated between two corresponding CAFs. Then, the CAF vector is applied to the rule sharing SiameseRF and (d) a similarity score is used for an association check. (e) The proposed rule distillation technique removes unimportant rules from the trained SiameseRF, and (f) the tracker's state is updated by considering a matched detection.

performance. However, such methods are unsuitable for online tracking because the network structure is complicated and the object tracking path of multiple frames must be analyzed. Moreover, the purpose of this study is to track multiple objects in a single video sequence in real time, and we mainly focus on online MOT methods based on a DNN. Sanchez-Matilla *et al.* [25] proposed an online multi-target tracker that utilizes both high- and low-reliability target detection in a probability hypothesis density particle filter framework. This tracker also conducts a data association immediately after the prediction phase, eliminating the need for computationally expensive labeling procedures such as clustering. Fang *et al.* [26] proposed a recurrent autoregressive network, which is a temporal generative modeling framework used to characterize the appearance and motion dynamics of multiple objects over time. The external memory explicitly stores previous inputs of each trajectory in a time window, whereas the internal memory learns to summarize the long-term tracking history and associate detections by processing the external memory. In addition, Kim *et al.* [27] proposed an online MOT tracking using a pre-trained CNN model and a teacher–student training mechanism. With this method, multiple trackers are trained every frame using a student-random fern to consider the variations in an object's appearance, instead of through the end-to-end online learning of a CNN. These methods have the advantage of an online MOT performance, but have the disadvantage of requiring a long tracking time during the process of checking the long-term tracking history [26] or re-learning [27] for each tracker.

Wang *et al.* [28] proposed the joint learning of an object detection and appearance embedding model using a CNN-based single-shot detector for a data association. Although this method provides a simple and fast association method in conjunction with a joint model, it has certain disadvantages in that many ID switches occur because it is too dependent on the detection dataset and the embedding feature is not optimized for a re-identification.

Zhou *et al.* [29] proposed a deep alignment network-based MOT with occlusion and motion reasoning. Because inaccurate detections are first corrected through a deep alignment network, the deep features from an alignment network have better representation power, thus leading to more consistent tracks. A discriminative association cost matrix is constructed using a coarse-to-fine schema with spatial, motion, and appearance information. They also proposed a fine-grained spatial alignment model [30] to effectively handle challenging scenarios such as complex poses, inaccurate detection, and occlusions arising from person re-identification or MOT. In particular, with this method, a pose resolution network is first designed using a channel parsing block to extract pose information at the pixel level. Given the extracted pose

TABLE 1. Main terminologies and corresponding abbreviations used in this article.

Terminologies	Abbreviations
Single object tracking	SOT
Multiple object tracking	MOT
Deep neural network	DNN
Tracking-by-detection	TBD
Random Forest	RF
Siamese Random forest	SiameseRF
Condensed appearance features	CAF
Condensed feature map	CFM
Identity	ID
Global average pooling	GAP
Multiple object tracking accuracy	MOTA
Multiple object tracking precision	MOTP
Average false alarms per frame	FAF
Ratio of mostly tracked targets	MT
Ratio of mostly lost targets	ML
Number of false positives	FP
Number of false negatives	FN
Number of IDentity switches	IDsw
Number of fragmentations	Frag
Frames per second	Hz

information, a locally reinforced alignment mode is then further proposed to address the misalignment problem between different local parts.

Tian *et al.* [31] proposed an association solution for use under motion noise or long-term occlusions. With this method, detections are assembled into small tracklets based on meta-measurements of the object affinity, and the association task for tracklets-to-tracks is solved using structural information based on a motion pattern between them. Xu *et al.* [32] introduced the choice of appropriate loss functions for end-to-end training of MOT methods by proposing a differentiable proxy of the MOT accuracy and precision. In addition, deep Hungarian-net was proposed to provide a soft approximation of the optimal prediction-to-ground-truth assignment. As a similar approach, Pang *et al.* [33] proposed an end-to-end model TubeTK, which only needs one step training, by introducing a bounding-tube to indicate temporal–spatial locations of objects in a short video clip. Although these methods are fast, they require a post-processing operation to connect the tracklets, and thus are closer to semi-on-line tracking than to a full on-line tracking method.

Tracking using a Siamese CNN for person re-identification in MOT has recently been studied [10], [5], [19]–[22]. A Siamese CNN applies the same network to the detection and tracker and calculates the similarity in the difference between output function values. Therefore, a Siamese CNN does not need to maintain a separate network structure and has the advantage of a fast tracking.

Leal-Taixe *et al.* [10] proposed a Siamese CNN to learn the descriptor encoding of local spatio-temporal structures between two input image patches by aggregating the pixel values and optical flow information. A Siamese CNN estimates the likelihood that two pedestrian detections belong to the same tracked entity.

In [19], Wang *et al.* first pre-trained a Siamese CNN on the auxiliary data and jointly learned the temporally constrained metrics online to construct the appearance-based tracklet affinity models. For a reliable association between tracklets, a loss function incorporating a temporally constrained multi-task learning mechanism is proposed. Unlike the dual architecture of the above two methods, Son *et al.* [5] proposed a quadruplet architecture by modifying a Siamese CNN and triplet networks to learn object associations for MOT. This method combines the shape of the detection with sequence-specific motion-aware locations for metric learning, and the entire network is trained end-to-end. However, because online end-to-end learning of an entire network requires certain amounts of training time and system resources, it is unsuitable for real-time environments. Zhu *et al.* [6] also introduced dual matching attention networks with both spatial and temporal attention mechanisms based on a Siamese structure association. Bergmann *et al.* [20] proposed 'Tracktor' based on a Siamese network that tackles MOT by exploiting the regression head of a detector to conduct a temporal realignment of the object bounding boxes. Lee and Kim [21] proposed a feature pyramid Siamese network to address the simplicity of a basic Siamese structure. This method extends the Siamese network by applying a feature pyramid network to the plain Siamese architecture and by developing a new multi-level discriminative feature. Chu and Ling [22] proposed an end-to-end Siamese based MOT model including feature extraction, affinity estimation, and a multi-dimensional assignment. To further improve the tracking robustness during tracking, it includes single object tracking and the prediction of a dedicated target management.

Although the CNN-based Siamese tracking approaches are a good choice for applications in an on-line MOT than general CNN-based approaches, these studies still have certain limitations: a shared network for similarity matching maintaining a large number of hyper parameters, a slow tracking speed owing to the complex CNN structure for real-time tracking, and a performance degradation on heavily moving cameras.

Table 2 summarizes the representative DNN based MOT approaches in terms of their main features, association method, and online tracking.

III. MULTIPLE OBJECT TRACKING

Unlike CNN-based MOT methods, we propose a SiameseRF framework that combines an accurate RF with a Siamese structure having high-speed learning and classification. Because this study follows MOT-based TBD, the object detection must precede each input sequence. In this study, YOLOv3 [15] is employed as an object detector based on DarkNet53, which is a real-time object detection system.

TABLE 2. **Summary of the representative deep learning based MOT approaches and their main features.**

Reference	Main algorithm	Association	Online
Sanchez-Matilla et al. [25]	Probability hypothesis density particle filter	Particle filter	O
Fang et al. [26]	Temporal generative modeling framework	Recurrent autoregressive network	O
Kim et al. [27]	Teacher–student training mechanism	Random forest	O
Wang et al. [28]	Joint learning of an object detection and appearance embedding model	CNN	O
Zhou et al. [29]	Deep alignment network	Hungarian algorithm	O
Tian et al. [31]	Meta-measurements of the object affinity	Structural information	O
Xu et al. [32]	Differentiable proxy of the MOT	Deep Hungarian-net	O
Pang et al. [33]	End-to-end model TubeTK	IoU-based greedy algorithm	Δ
Leal-Taixé et al. [10]	Siamese CNN based fast tracking.	Siamese CNN	O
Wang et al [19]	Siamese CNN + temporally constrained metrics	Siamese CNN	O
Zhu et al. [6]	Dual matching attention networks	Siamese CNN	O
Lee and Kim [21]	Feature pyramid Siamese network	Siamese CNN	O
Chu and Ling [22]	End-to-end Siamese network	Siamese CNN	O

YOLOv3 uses a single neural network to predict the bounding boxes and class probabilities directly from full images in a single evaluation.

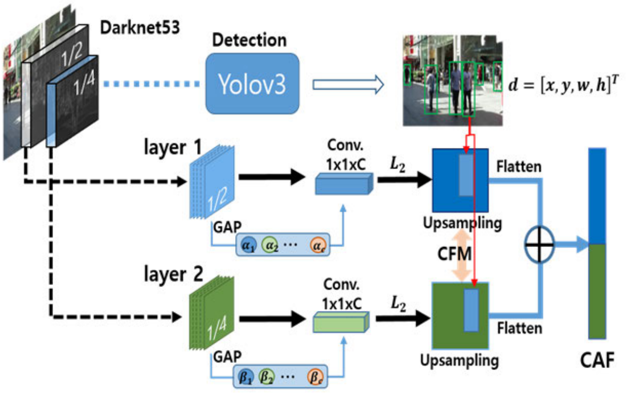


FIGURE 2. **Procedure of CAF extraction. Two CFMs are created from the output feature maps of the first and second convolution layers. Condensed features extracted from the corresponding location of the two CFMs are concatenated into a 1250-dimensional CAF feature.**

A. FEATURE EXTRACTION OF AN OBJECT

To match the existing trackers in the previous frame and the newly detected objects in the current frame, a similarity score matrix must be generated using SiameseRF, and applying this matrix, an object with a high score must be connected to the corresponding tracker. At this time, the most basic and important step for measuring the similarity is the feature extraction. An RF is known to achieve an excellent performance for tabular data but has a poor performance for unconditioned data such as image and video. Therefore, an optimal feature extraction that can effectively distinguish objects should be applied as a preprocessing step of Siamese RF.

Figure 2 shows the procedure of CAF feature extraction from an object. The local appearance feature of the object for a similarity measurement uses the output feature maps of the first and second convolution layers of the DarkNet53 network inspired by [34]. According to this study, the feature information hidden in different two layers has the potential for a feature discrimination capacity; however, deeper layers can easily damage the scene feature structure [34]. We, therefore, combine two feature layers at the front part of the network. We, therefore, combine two feature layers at the front part of the network. The experiment for selecting the optimal two convolution layers for feature extraction is described in section V-A. Let us suppose we can obtain C feature maps, $A^c \in \mathbb{R}^{u \times v}$, with each element indexed by i and j of layer l. Therefore, $A^c_{i,j}$ indicates the activation at location (i, j) of feature map A^c. Inspired by [35], channel-wise global average pooling (GAP) is applied to A^c to capture the feature importance weight α^c_l for a detected object. A condensed feature map (CFM) at location (i, j) of layer l is then estimated by a combination of importance weights $\alpha^c_l (1 \times 1 \times C)$ and feature maps of the same location, followed by L2 normalization operation.

$$CFM^l\,(i,j) = L_2\left\{\sum\nolimits_c \alpha^c_l A^c_l\,(i,j)\right\}. \tag{1}$$

Two CFM^l values are then upsampled to the input image resolution using a bilinear interpolation. We extract the object's partial condensed features from the same location of a detected object. The partial condensed features are normalized to a size of 25 × 25 and flattened again (1 × 625D). Two condensed features are concatenated to become one final CAF (1 × 1250D). The CAF branch operates independently from the YOLOv3 training because there are no learnable parameters and it does not require space for a parameter reduction. In addition, by reusing the DarkNet53 network for the feature extraction of the YOLOv3 detector, we can reduce the amount of unnecessary computations and resource demand.

B. SIAMESE RANDOM FOREST

To associate CAF^i_t of a detected object i and CAF^k_t of a tracker k at the current time t, it is necessary to determine whether the detected object and tracker are the same object. One way to measure the similarity of two images in real time is to use the distance function D.

Chopra *et al.* [36] proposed the use of a Siamese network, which is a way to learn output function f through a deep learning method. The learning of the Siamese network is realized by training a network consisting of two identical CNNs that share the same set of weights. The Siamese CNN converts image a and image b into vector representations of $f(a)$ and $f(b)$ using shared networks.

To learn f, weights of a Siamese CNN are trained normally using a Triplet loss function. A Triplet loss is a method for creating a loss function from three images, and the idea is to minimize the distance between the same identities *Anchor (a)* and *Positive (p)* and maximize the distance between different *Anchor (a)* and *Negative (n)* identities. The triplet loss for configuring a Siamese network is as follows:

$$L(a,p,n)=max\left(\|f(a)-f(p)\|_2-\|f(a)-f(n)\|_2+\alpha,0\right) \tag{2}$$

where α is a margin used to create a sufficient difference between values.

In this study, we propose a SiameseRF framework that combines a Siamese structure and an RF instead of a CNN; this enables high-speed learning and classification.

In the SiameseRF training process, an initial RF consisting of L ensemble trees is created. Two RFs receive the CAF distance vector of an {*anchor, positive*} pair and an {*anchor, negative*} pair as inputs, as shown in Figure 3. We call an element-wise distance vector between the two CAFs of the {*anchor, positive*} pair the AP distance. Meanwhile, an element-wise distance vector between the two CAFs of the {*anchor, negative*} pair is called the AN distance. The two RFs share the same structure.

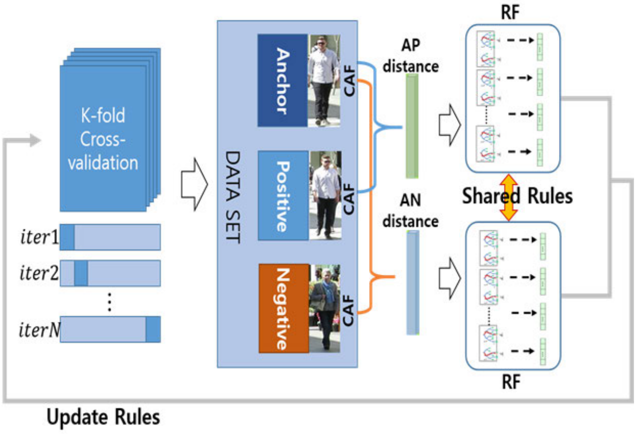


FIGURE 3. K-fold cross validation for the training process used to train the shared rules of SiameseRF.

As the input of each RF, the CAF difference vectors, AP and AN, are input as a feature. A vector AP, AN $\in R^{1\times m}$ is a distance vector if and only if the following holds:

$$\mathrm{AP}_i=d(a_i,p_i),\ \mathrm{AN}_i=d(a_i,n_i),\quad for\ 1\le i\le m, \tag{3}$$

where d is an L_2 distance function, $a_i \in Anchor$, $p_i \in Positive$, $n_i \in Negative$, and m is the number of samples in each label, *Anchor*, *Positive*, and *Negative*.

To repeat the training phase of a shared RF, in this study, K-fold cross validation is adopted to improve the accuracy of the model. The K-fold cross-validation method automatically determines the optimal rule numbers and parameters while reducing the risk of an over-fitting. The Siamese RF learning based on a K-fold validation is as follows:

- Step 1: K-1 folds are selected from the entire training data set S, and the excluded fold is used as the validation set.
- Step 2: The AP-distance vector for a sample pair {*anchor, positive*} and the AN-distance vector for a sample pair {*anchor, negative*} is estimated using CAFs.
- Step 3: The distance vectors of the sample pairs are input into each RF sharing the rules. The decision to update the rules consisting of shared RF depends on whether the K-fold cross-validation converges.
- Step 4: RF composed of L trees is trained using the AP and AN distance vectors.
- Step 5: After the training of K-1 folds, AP and AN distance vectors are obtained from pairs {*anchor, positive*} and {*anchor, negative*} of samples in the validation set. A triplet loss of SiameseRF is computed using Eq. (4), which is modified from Eq. (3). This process is performed for all n pairs in the validation set.

$$L(a,p,n)_{\mathrm{k}} = max\left(\|1-\mathrm{RF}_k(AP)\|^2-\|\mathrm{RF}_k(AN)\|^2+\alpha,0\right) \tag{4}$$

  where RF_k represents the RF structure used in k-th fold.
- Step 6: Store the trained RF structure and total loss J. Steps 1–6 are repeated until each fold has been used as the testing fold.

$$J_k=\sum\nolimits_{i=1}^{n} L(a_i,p_i,n_i)_k^i \tag{5}$$

- Step 7: When the learning is completed for all K-folds, the RF with the smallest total loss J is determined as the final SiameseRF.

$$k=arg\min_{k\in K} J_k \tag{6}$$

In the learning process, the shared SiameseRF is trained in the direction in which the similarity between the positive pair increases and the difference between the negative pair increases. Unlike a Siamese CNN, SiameseRF does not share weights, but instead rules consisting of a tree.

The detailed procedures for a training of SiameseRF are described in Algorithm 1.

After training the SiameseRF using K-fold verification, during the actual tracking, the CAFs extracted from a detection object and a tracker. The L_2 distance vector is computed from two CAFs and is input into the trained SiameseRF. The similarity probability of the two objects becomes the appearance score for the association, as shown in Figure 4.

C. TRACKER ONLINE ASSOCIATION

For real-time MOT, this study applies a frame-by-frame data association check between the tracker and detection through a SiameseRF-based similarity probability. The process of an association check is as follows: objects are detected using

Algorithm 1 Training of Siamese RF

```
Input:
S: a dataset consisting of N labeled samples $(x_i, y_i) \ldots (x_n, y_n)$, where $y_i \in \{0, +1, -1\}$ is the label for the anchor, and positive and negative classes, respectively
O: dataset of K-1 folds, V: dataset for validation
K: folding factor,
RF: a set of candidate RFs, J: a set of total loss of candidate RFs
for $k = 1$ to $K$ do
    $O_k \leftarrow$ select $K-1$ folds from $\mathbf{S}$ $(k \notin O_k)$
    $\mathbf{V} \leftarrow$ get remaining k fold for validation
    for $i = 1$ to $m$ do
        Select pair$\{(a_i, p_i), (a_i, n_i)\}$ from $O_k$
        Distance vectors AP and AN estimation using Eq. (3)
    end
    // RF growing with L trees using AP and AN vector
    for $l = 1$ to $L$ do
        $RF_k \leftarrow Tree(d, r)_l$   //d: depth, r: rules
    end
    Compute Triplet loss $L(a, p, n)_k$ and total loss $J_k$
    $\mathbf{RF} \leftarrow RF_k$
    $\mathbf{J} \leftarrow J_k$
end
Find RF having the smallest total loss using Eq. (6)
Output: SiameseRF $\leftarrow$ RF(k)
```

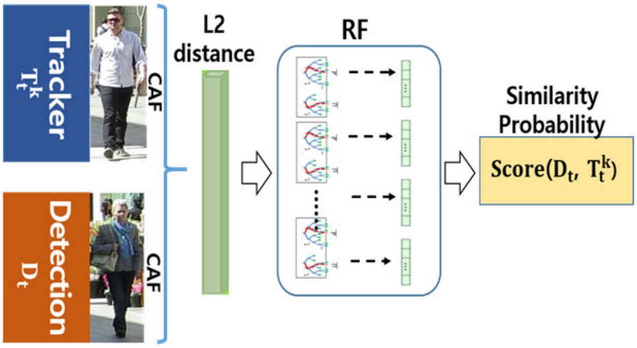


FIGURE 4. Estimation SiameseRF probability between a tracker and detection pair used for testing.

YOLOv3; each object and tracker measures the similarity by combining the probability value of the SiameseRF, the aspect ratio, and the distance; using the Hungarian algorithm, the detection with the highest similarity is linked to the tracker's trajectory; and the tracker status information is then updated using formula (7).

Tracker set $\mathbf{Tr}_t = \{\mathrm{tr}_t^i, \mathrm{tr}_t^{i+1}, \ldots, \mathrm{tr}_t^N | 1 \leq t\}$ of the t-th frame can be defined as a list of ordered bounding boxes $\mathrm{tr}_t^i = [x_t^i, y_t^i, w_t^i, h_t^i]$ and the target detection set $\mathbf{D}_t = \left\{\mathrm{d}_t^j, \mathrm{d}_t^{j+1}, \ldots\right\}$, and the initial vector of the j-th detection is automatically set as $\mathrm{d_i}$= $[\mathrm{cx_i cy_i w_i h_i}]^{\mathrm{T}} d_t^j = \left[x_t^j, y_t^j, w_t^j, h_t^j\right]$. Here, (x, y) is the center position, whereas w and h are the width and height, respectively, of an object's bounding box. When t = 0, elements of tracker set $\mathbf{Tr}_0$ are initialized to elements of $\mathbf{D}_0$. Actually, the process of matching the tracker and the detection starts from frame t > 1.

In every frame, objects detected by YOLOv3 are assigned to the tracker using the Hungarian method of Algorithm 2. In general, the Kalman filter, which is frequently used in online tracking, is used to predict the location of the previous tracker in the current frame. However, because the test images include those in which the camera was shaking or in which objects are moving unpredictably to the front, left, or right, or multiple objects are moving at various speeds, the tracker checks the association with the detections within a limited boundary instead of through a prediction using a Kalman filter.

If the detected object and tracker are matched, the state of the tracker tr_t^i is updated by combining the states of the current tracker tr_{t-1}^i and detection d^j using the following:

$$\mathrm{tr}_t^i = \delta \cdot \mathrm{tr}_{t-1}^i + (1-\delta) \cdot \mathrm{d}_t^j. \tag{7}$$

By contrast, if the tracker does not match during τ frames, it is considered to have disappeared and is deleted. Similarly, if the detected object does not match any tracker, it is assigned as a potential tracker, and if the number of matches between the tracker and detection occurs over τ frames, it is assigned as a new tracker; otherwise, it is recognized as a false detection and removed.

Three measures are used to calculate a similarity score for Hungarian matching between the tracker and detection. This study applies stepwise affinity measures for generating matrix $\mathbf{M}$inspired by the approach in [27]. As the first step, we construct a pairwise blank affinity matrix $\mathbf{M}$ between the predictions and detections. Then, only detections (j) within a certain radius of the tracker (i) are defined as valid matching pairs of affinity $\mathbf{M}_{i,j}$. In detail, based on the center (x, y) of the i-th tracker, a search area having a tracker height ($h \times \alpha$, $1 \leq \alpha < 2$) as a radius is created, and detections included in the circle are determined as candidates for comparison of tracker i. Parameter α for the radius depends on the movement of the camera. That is, it has a value of '1' in a static camera and a value greater than '1' in the case of an ego-motion camera.

The first measure is the inverse probability value of SiameseRF, $\hat{P}_{Siam}\left(\mathrm{tr}^i|\mathrm{d}^j\right) = 1 - P_{Siam}\left(\mathrm{tr}^i|\mathrm{d}^j\right)$. This value is the most important factor in measuring the similarity between the tracker and detection. The second measure is the aspect ratio variation $A_{ratio}\left(\mathrm{tr}^i, \mathrm{d}^j\right)$between the tracker i and detection j. Because the object deformation and rotation can be characterized by the change in aspect ratio, we measured the degree of aspect ratio variation by modifying of [37] in frame t as follows:

$$A_{ratio}(\mathrm{tr}^i, \mathrm{d}^j) = \max\left(\frac{r(d^j)}{r(tr^i)}, \frac{r(tr^i)}{r(d^j)}\right), \tag{8}$$

where $r = h/w$.

The third measure is the relative L1-distance $Dis\left(\mathrm{tr}^i, \mathrm{d}^j\right)$ between the center of track i and detection j in the appearance space. Finally, for the cost function of association matching, we combine three distance measures using a weighted sum.

$$c\left(tr^{i}, d^{j}\right) = \alpha \cdot \hat{P}_{Siam}\left(\mathrm{tr}^{i}|\mathrm{d}^{j}\right) + \beta \cdot A_{ratio}(\mathrm{tr}^{i}, \mathrm{d}^{j}) + \gamma \cdot Dis\left(\mathrm{tr}^{i}, \mathrm{d}^{j}\right) \quad (9)$$

where α, β, and γ denote the weights, which are 0.4, 0.2, and 0.4, respectively. To obtain an appropriate weight, we measured the MOTA using the MOT 16 dataset. When 3 weights, α, β, and γ were equally assigned by 0.33 each, the MOTA was measured to be 53.8%. On the other hand, when α and γ were given as 0.4 and β was given as 0.2, MOTA was improved by 4.1% to 57.9%. This is because the ratio of the size of the object represented by β is different when a pedestrian is standing and when walking, especially when occlusion occurs. Therefore, when the weight of β is 0.2 and α and γ for the SiameseRF score and distance are equal to 0.4, it showed the best MOT performance. These weights are also adjustable according to the characteristics of the dataset.

The detailed procedures for an online association are described in Algorithm 2.

```
Algorithm 2 Online Association
Input:
Tracker set $\mathbf{Tr}_t = \{\mathrm{tr}_t^i, \mathrm{tr}_t^{i+1}, \ldots, \mathrm{tr}_t^I | 1 \le t\}$
Target detection set $\mathbf{D}_t = \left\{\mathrm{d}_t^j, \mathrm{d}_t^{j+1}, \ldots, \mathrm{d}_t^J\right\}$
Affinity matrix $\mathbf{M}_{i,j}, i \le I, j \le J$
Initialize affinity matrix $\mathbf{M}$
Generate affinity matrix $\mathbf{M}$ for the trackers and the detections
For $i = 1$ to $I$
    For $j = 1$ to $J$ do
        Compute search area of $\mathrm{tr}_t^i$
        Find a valid matching pair $\left(\mathrm{tr}_t^i, d_t^j\right)$ from $\mathbf{D}_t$ and
        compute a cost function using Eq. (9)
        $\mathbf{M}_{i,j} \leftarrow c\left(\mathrm{tr}_t^i, d_t^j\right)$
    end
end

While (no further tracker is available)
    Find a pair $(\mathrm{tr}_t^i, d_t^j)$ from $\mathbf{M}_{i,j}$that has a minimum cost
    function.
End While
The state of the tracker $\mathrm{tr}_t^i$ is updated using Eq. (7)
```

IV. RULE DISTILLATION OF SIAMESE-RF

SiameseRF itself shows a good performance in measuring the similarity between the detection object and the tracker; however, because SiameseRF is based on an RF composed of multiple trees, it is necessary to reduce the number of trees or reduce the rules of the tree for real-time online MOT. However, because the number of trees is closely related to the similarity matching performance, it cannot be removed without care, and the method of removing the rules while maintaining the number of trees is an effective alternative. The proposed lightweight method of an RF can explain how much influence each feature has through an analysis of the contribution of node rules constituting each tree.

In SiameseRF learning, the nodes of each tree constituting the RF not only have a class distribution but also learned features and threshold values for an optimal tree generation. This characteristic of the tree node enables an analysis of how the rules of the tree generated in the learning process affect the prediction result. Based on this fact, we can remove duplicate or relatively inefficient rules from the tree by calculating rule contribution $Contri(r_i)$.

The i-th rule in the tree Te is composed of several nodes. We compute the feature contribution $feat.Contri\,(i, j) = \left(Pr_{j-1} - Pr_j\right)$ for the j-th node using the class probability distribution $\mathrm{Pr} = \{\mathrm{pr}_1, \ldots pr_{No.of.Class}\}$ between the parent node (j-1) and the child node (j). Finally, SiameseRF with some rules removed can maximize the efficiency of the RF by reducing the number of essential parameters without significantly affecting the accuracy.

This method is defined as a rule distillation herein, and SiameseRF is made lighter by removing the rules of the tree using Algorithm 3.

```
Algorithm 3 Rule Distillation of SiameseRF
Input:
$Te$: trained tree of SiameseRF
$dR$: distillation rate, P: depth of a tree
$\mathbf{R}$: ordered rule set
Initialize rule set $\mathbf{R} = \emptyset$
For each tree $Te$ in SiameseRF do
    Induce an i-th rule $r_i^{Te}$ from a trained tree $Te$
    The rule contribution $\mathrm{Contri}(r_i^{Te})$ is computed for the
    i-th rule

    $$\mathrm{Contri}\left(r_i^{Te}\right) = \frac{\sum_j^P feat.Contri(i,j)}{\sum_{t=1}^{Te} No.of.Class(Te)} \quad (10)$$

    Append rule $r_i^{Te}$ and $\mathrm{Contri}\left(r_i^{Te}\right)$ to $\mathbf{R}[l]$
End for
Sort rules in $\mathbf{R}[l]$ according to rule contribution
Eliminate rules with low contribution by $dR$ %.
Output: dR% rules in $\mathbf{R}$
```

SiameseRF, with a distilled rule, reduces the memory space for parameter storage and enables a faster association by removing unnecessary rules while maintaining the tracking accuracy. In Section V-A, the association matching according to the rule distillation rate is presented based on the experimental results.

V. EXPERIMENTAL RESULTS

To measure the tracking performance of the proposed SiameseRF, in this study, several experiments were conducted on the benchmark databases that are mainly used in the MOT challenges:

- MOT16 [38]: This dataset consists of a set of 14 sequences with more complex scenarios, different perspectives, camera movements, and weather conditions. All sequences were annotated by experts under strict standards. MOT16 is annotated not only for pedestrians, but also for vehicles, sitting people, occluding objects, and other important object classes.
- MOT17 [39]: This dataset is an extended version of the MOT16 dataset and includes three sets of detections from a deformable part model [38], Faster-RCNN [13], and scale dependent pooling [40] for a more comprehensive evaluation of the tracking algorithms. MOT17 is mainly used to compare and test the tracking performance of objects detected by three public detectors.

For validation and evaluation of the proposed method, we used CLEAR MOT metrics [7] such as the multiple object tracking accuracy (MOTA), the multiple object tracking precision (MOTP), the average false alarms per frame (FAF), the ratio of mostly tracked targets (MT), the ratio of mostly lost targets (ML), the number of false positives (FP), the number of false negatives (FN), the number of IDentity switches (IDsw), the number of fragmentations (Frag),[2] and Hz (frames per second). The two intuitive metrics MOTP and MOTA can be defined in [7].

$$\text{MOTP} = \frac{\sum_{i,t} d_t^i}{\sum_t c_t} \tag{11}$$

where c_t is the number of matches found for time t and d_t^i is the distance between the object o_i and its corresponding hypothesis.

$$\text{MOTA} = 1 - \frac{\sum_t (m_t + fp_t + mme_t)}{\sum_t g_t} \tag{12}$$

where g_t is the number of objects present at time t, and m_t, fp_t, mme_t are the number of misses, false positive, and mismatches, respectively.

For SiameseRF learning, the area containing the pedestrian was cropped to the size of a pedestrian from the CFM by applying the ground-truth in the sequence of the MOT16 training set. The positive pair $\{anchor, positive\}$ and negative pair $\{anchor, negative\}$ are composed of 31,170 images, respectively. During the learning process, the optimal rules and the tree parameters are automatically determined while reducing the risk of an overfitting using a five-fold cross validation.

Both YOLOv3 and SiameseRF are implemented in C++, and YOLOv3 for object detection runs in a single RTX2080ti GPU environment, whereas SiameseRF runs using an Intel Core i7-9700K processor under the Windows 10 environment. Tracking, including feature extraction, works in parallel with the object detection part.

A. CONVOLUTION LAYER SELECTION FOR FEATURE EXTRACTION

In Section III-A, we mentioned that, when extracting the CFM for a similarity measurement, we combined the output feature maps of the two consecutive convolution layers of DarkNet53. Therefore, this experiment attempted to determine the appropriate layer-pair of Darknet53 for improving the object matching accuracy without increasing the processing time and parameter numbers. Experiments were conducted using the MOT16 sequences 2 and 9 because they were captured using a static camera.

Figure 5 shows the matching accuracy according to the change in layer pairs. When a layer pair '1 + 2' was used, the average matching accuracy was 83.05%, which is better than that when other pairs were used. As we determined from the experimental results, the middle layer pair ('2 + 3') of Darknet53 achieves a good matching accuracy compared to the latter layer pairs, but a 7.35% lower accuracy than the first layer pair. In general, although the latter layer of the CNN expresses important abstraction information of an object well, the matching accuracy is considered to be low because distinctive information may be lost for a small object.

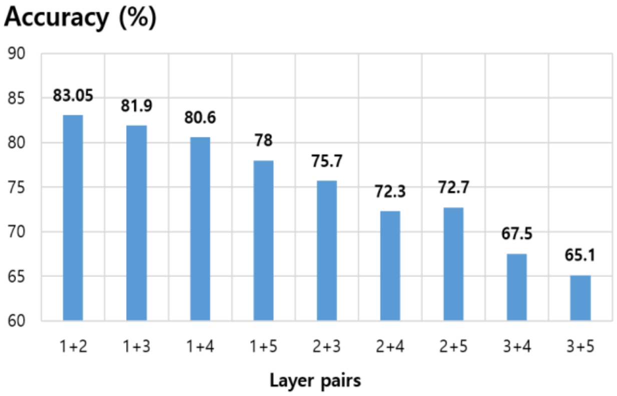


FIGURE 5. Nine possible pairs of experiment results for determining the layer-pair of DarkNet53. Combining a '1 + 2' pair shows the best matching performance.

These results prove that the matching accuracy is strongly dependent on the quality of the object features, and if the size of the feature map is too small, it may influence the degradation of the matching performance.

B. DETERMINATION OF RULE DISTILLATION RATIO

In the proposed SiameseRF structure, we compared and tested the effect of the tracking performance of MOTA, the computational quantity, and the number of parameters when the ratios of the rule distillation constituting an RF were different. Moreover, although an RF improves the performance by averaging multiple decision trees trained using a bagging method to reduce the overfitting and variance, the performance cannot be improved further owing to unnecessary rules that are still overfit or have a high variance [27]. These problems can be solved to a certain extent by

[2] Track fragmentation occurs when a tracker tracks an object's trajectory or track into two or more separate track instances [42].

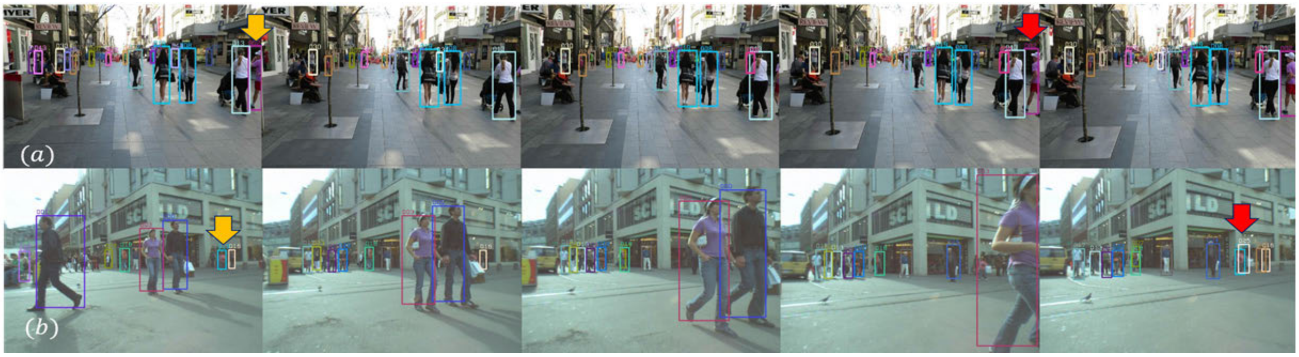


FIGURE 6. Tracking results for a moving camera using the proposed SiameseRF method: (a) tracking results for partial and short-term occlusions and (b) cases of switching ID owing to a long-term occlusion.

removing less important or redundant rules. All MOTA 16 test sequences were used in the comparative experiments.

As shown in Table 3, when the number of rules is decreased by 10%, there is no significant difference in the number of operations or performance of the model, but it can be seen that the number of parameters is greatly reduced. From the results, we observe that the performance of the model is maintained until 30% of the rules are removed; important rules having a high contribution are retained and non-important rules are effectively removed. However, if more than 40% of the rules are removed, we observe that the decline in MOTA also increases by approximately 22% as important rules with a high contribution are also removed.

TABLE 3. Comparison results of MOTA, number of rules, parameters, and operations according to the change in ratio of the rule distillation on MOT 16 sequences.

Rule ratio	MOTA (%)	# of Rules (M)	# of Param. (M)	# of Op. (M)
1.0	57.9	0.2061	2.4641	0.0126
0.9	57.4	0.1936	2.4098	0.0125
0.8	57.2	0.1863	2.3498	0.0124
0.7	57.2	0.1656	2.1842	0.0122
0.6	34.5	0.1450	2.0055	0.0121
0.5	34.4	0.1243	1.8095	0.0120

Thus, it was found that the proposed tracking system was most suitable in terms of performance and weight reduction when 30% of the rules were removed.

C. EVALUATION ON THE MOT16 CHALLENGE DATASET

In the first experiment for the tracking performance, the proposed method is compared with the latest online multi-object tracking method using the MOT 16 dataset, as shown in Table 4. As the measures of the MOT performance, ten CLEAR MOT metrics described in the previous section were used. We compared the performance of eight state-of-the-art methods to verify the effectiveness of the proposed tracking method: (1) DMAN [6] using dual matching attention networks, (2) RAR 16 [26] applying a recurrent autoregressive network, (3) JCSTD [31] using meta-measurements, (4) Tracktor [20] using Tracktor based on a Siamese network, (5) TrctrD16 [32] applying deep Hungarian net, (6) MLT [43] using a multiplex labeling graph, (7) the proposed SiameseRF without a rule distillation, and (8) the proposed SiameseRF with a 30% rule distillation. Table 3 only measures the tracking performance of the proposed method based on the results of a private detection. Among the comparison methods, DMAN [6] used a Siamese CNN similar to the proposed method.

As shown in Table 4, the proposed algorithm shows a similar tracking performance and a relatively fast-tracking speed compared to the state-of-the-art MOT algorithms based on a CNN. Although with JCSTD [31], which shows the fastest processing speed on a CPU among the six comparison methods, the MOTA and MT are 10.5 and almost 2-times lower than those of the proposed method, respectively. In terms of the MOTA performance, RAR16 [26] showed the best performance at 63%, but it can be seen that the tracking speed is possibly 7.75-times slower than that of the proposed rule distillated method (8). The proposed SiameseRF in (7) shows a performance similar to MOTA when applying the latest three methods, i.e., (4), (5), and (6), and specifically, MOTP showed the highest value of 79.4%. However, frequent IDsw and fragment events are problems requiring improvement.

D. EVALUATION ON MOT17 CHALLENGE DATASET

In the second experiment on the tracking performance, we compared the state-of-the-art methods with the proposed method using the MOT 17 dataset, as shown in Table 5. To verify the effectiveness of the proposed tracking method, the same performance was measured using 10 state-of-the-art methods: (1) DMAN [6], (2) DEEP_TAMA [41] using deep temporal appearance matching, (3) STRN [45] applying a spatial-temporal relation network, (4) FAMNet [22] using a multi-object assignment, (5) Tracktor [20], (6) TrctrD17 [32], (7) YoonKJ17 [46] applying one-shot-learning, (8) the proposed SiameseRF without rule distillation, and (9) the

TABLE 4. Comparison results of 10 CLEAR MOT metrics using eight state-of-the-art MOT methods on MOT 16 sequences.

	Method	MOTA(%)↑	MOTP(%)↑	FAF↓	MT(%)↑	ML(%)↓	FP↓	FN↓	IDsw↓	Frag↓
Online	(1) DMAN [6]	46.1	73.8	1.3	17.4	42.7	7,909	89,874	532	1,616
	(2) RAR16 [26]	**63.0**	78.8	2.3	**39.9**	22.1	13,663	**53,248**	482	1,251
	(3) JCSTD [31]	47.4	74.4	1.4	14.4	36.4	8,076	86,638	1,266	2,697
	(4) Tracktor [20]	56.2	79.2	**0.4**	20.7	35.8	**2,394**	76,844	617	1,411
	(5) TrctrD16 [32]	54.8	77.5	0.5	19.1	37	2,955	78,765	645	1,515
	(6) MLT [43]	52.8	76.1	0.9	21.1	42.4	5,362	80,444	**299**	**702**
	(7) SiameseRF	57.9	79.3	1.4	28.5	**22.1**	8,196	66,538	2,051	2,549
	(8) SiameseRF+ rule distillation (0.7)	57.2	**79.4**	1.2	28.2	23.5	7,265	68,860	2,000	2,520

TABLE 5. Comparison results of 10 CLEAR MOT metrics using 10 state-of-the-art MOT methods on MOT 17 sequences.

	Method	MOTA(%)↑	MOTP(%)↑	FAF↓	MT(%)↑	ML(%)↓	FP↓	FN↓	IDsw↓	Frag↓
Online	(1) DMAN[6]	48.2	75.7	1.5	19.3	38.3	26,218	263,608	2,194	5,378
	(2) Deep_TAMA [44]	50.3	76.7	1.4	19.2	37.5	25,479	252,996	2,192	3,978
	(3) STRN [45]	50.9	75.6	1.4	18.9	**33.8**	25,295	249,365	2,397	9,363
	(4) FAMNet [22]	56.2	**79.2**	**0.4**	20.7	35.8	**2,394**	**76,844**	**617**	**1,411**
	(5) Tracktor [20]	**56.3**	78.8	0.5	21.1	35.3	8,866	235,449	1,987	3,763
	(6) TrctrD17[32]	53.7	77.2	0.7	19.4	36.6	11,731	247,447	1,947	4,792
	(7) YoonKJ17 [46]	51.4	77	1.6	21.2	37.3	29,051	243,202	2,118	**3,072**
	(8) SiameseRF	55.8	77.9	0.7	**21.6**	34.4	12,822	233,573	3,174	3,689
	(9) SiameseRF+ rule distillation (0.7)	55.4	77.8	0.8	**21.6**	34.3	14,079	**233,092**	4,684	3,859

proposed SiameseRF with 30% rule distillation. Unlike in Table 4, in Table 5, public detection results are listed instead of private detection to evaluate a more accurate tracking performance. Similar to Table 4, the proposed method shows a good performance in MOTA, which is the best tracking performance indicator. In particular, SiameseRF with 30% rule distillation showed a 0.4% lower MOTA than the full version (8) but demonstrated the best performance in MT.

E. EVALUATION ON TRACKING SPEED

The tracking speed of the proposed SiameseRF was compared with that of five other methods, as shown in Table 6. For objective speed comparison, performance was compared only for the results of experiments in the same Intel CPU i7 environment among the algorithms used in Table 4 and Table 5.[3]

TABLE 6. Comparison results of tracking speed according to the change in ratio of the rule distillation on MOT 16 and 17 sequences.

Method	Tracking Speed (Hz)	
	MOT 16	MOT 17
DMAN [6]	0.3	0.3
FAMNet [22]	-	1.6
JCSTD [31]	8.8	-
MLT [43]	5.9	-
Deep_TAMA [44]	-	1.5
SiameseRF	11.1	12.4
SiameseRF+ rule distillation (0.7)	**12.4**	**13.6**

In Table 6, Hz represents frames per second for only the tracking; it excludes object detection. Similar to the proposed method, the JCSTD [31] method showed the fastest processing time among the CNN-based methods at 8.8 Hz in MOT 16 dataset. However, this method not only showed a slower speed of 2.3 Hz than SiameseRF but was also 3.6 Hz slower than rule-distilled SiameseRF. In the case of rule-distilled SiameseRF, a processing speed of approximately 1.3 Hz faster than the basic SiameseRF method was shown while maintaining the overall performance. For the MOT17 dataset, FAMNet [22] showed the fastest processing speed at 1.6 Hz among other three comparison methods. However, this method has a large speed difference of 10.8 Hz from the basic SiameseRF. This means that if only the CPU is used for MOT, real-time processing cannot be expected in terms of tracking speed when MOT algorithms are based on only CNN. Overall, the proposed method has a tracking performance similar to that of a CNN-based method and shows a faster tracking speed than state-of-the-art comparison methods indicated in Table 6.

[3] The operating environment of each comparison method was based on the results of the MOT Challenge.

TABLE 7. A detailed summary of the tracking results of our Basic SiameseRF tracker on MOT 16 and 17 Challenge benchmarks.

Sequence	MOTA↑	MOTP↑	MT	ML↓	FP↓	FN↓	Recall↓	Precision↓	FAF↓	ID Sw. ↓	Hz↑
MOT16[38]											
SiameseRF											
MOT16-01	57	78.3	9	3	300	2365	63	93.1	0.7	82	18.3
MOT16-03	65.9	79.5	58	16	4068	30689	70.6	94.8	2.7	870	3.0
MOT16-06	58.9	79.2	88	42	946	3588	68.9	89.4	0.8	212	9.2
MOT16-07	53.6	79.2	10	5	555	6819	58.2	94.5	1.1	205	4.2
MOT16-08	34	81.6	10	23	453	10430	37.7	93.3	0.7	171	14.4
MOT16-12	46.1	79.5	18	35	547	3850	53.6	89	0.6	77	25.5
MOT16-14	42.9	76.4	23	44	1327	8797	52.4	88	1.8	434	3.4
ALL	57.9	79.3	28.5	22.1	8196	66538	63.5	93.4	1.4	2051	11.1
SiameseRF+rule distillation											
MOT16-01	55.7	78.6	10	4	234	2474	61.3	94.4	0.5	127	19.1
MOT16-03	65.2	79.5	57	18	3654	32019	69.4	95.2	2.4	672	4.2
MOT16-06	58	79.3	89	45	938	3612	68.7	89.4	0.8	297	9.7
MOT16-07	52.4	79.4	9	6	449	7073	56.7	95.4	0.9	247	5.4
MOT16-08	33.4	82.1	9	24	357	10649	36.4	94.5	0.6	137	15.6
MOT16-12	46.2	79.8	19	34	475	3915	52.8	90.2	0.5	71	26.8
MOT16-14	42	76.7	21	47	1158	9118	50.7	89	1.5	449	5.7
ALL	57.2	79.4	28.2	23.5	7265	68860	62.2	94	1.2	2000	12.4
MOT17[39]											
SiameseRF											
MOT17-01-DPM	41.8	77.3	5	11	56	3679	43	98	0.1	16	23.15
MOT17-03-DPM	67.3	78.1	61	19	2001	31771	69.6	97.3	1.3	422	1.8
MOT17-06-DPM	52.7	77.5	49	88	367	5113	56.6	94.8	0.3	99	12
MOT17-07-DPM	42.6	78.4	5	21	171	9416	44.3	97.8	0.3	115	7.1
MOT17-08-DPM	26.9	82.8	8	40	126	15217	28	97.9	0.2	90	10.5
MOT17-12-DPM	46.4	82.2	18	40	75	4532	47.7	98.2	0.1	36	26.1
MOT17-14-DPM	32.4	76.2	11	77	350	11988	35.1	94.9	0.5	158	11.2
MOT17-01-FRCNN	43.1	76.6	6	11	191	3461	46.3	94	0.4	21	20.62
MOT17-03-FRCNN	67.9	77.6	56	17	2184	30982	70.4	97.1	1.5	421	2.1
MOT17-06-FRCNN	55.6	76.9	56	62	566	4546	61.4	92.7	0.5	121	11.5
MOT17-07-FRCNN	42.1	78.1	6	20	342	9290	45	95.7	0.7	148	6.4
MOT17-08-FRCNN	26.5	82.6	8	37	141	15299	27.6	97.6	0.2	80	13.3
MOT17-12-FRCNN	45	82.2	16	42	69	4674	46.1	98.3	0.1	23	22.6
MOT17-14-FRCNN	33.4	75.3	15	72	714	11362	38.5	90.9	1	231	7.6
MOT17-01-SDP	43.2	76.5	6	10	181	3445	46.6	94.3	0.4	39	19.52
MOT17-03-SDP	72.4	77.1	72	15	3194	25199	75.9	96.1	2.1	535	1.9
MOT17-06-SDP	55.7	76.8	62	65	631	4453	62.2	92.1	0.5	135	13.1
MOT17-07-SDP	44.8	77.8	8	18	349	8833	47.7	95.8	0.7	142	5.2
MOT17-08-SDP	27.8	81.7	9	35	201	14960	29.2	96.8	0.3	88	10.6
MOT17-12-SDP	46	81.9	18	43	178	4477	48.3	95.9	0.2	26	25.4
MOT17-14-SDP	35.9	75.4	13	67	735	10876	41.2	91.2	1	228	8.3
ALL	55.8	77.9	21.6	34.4	12822	233573	58.6	96.3	0.7	3174	12.4
SiameseRF+rule distillation											
MOT17-01-DPM	41.5	77.2	5	11	62	3678	43	97.8	0.1	33	24.3
MOT17-03-DPM	66.9	78	60	18	2234	31728	69.7	97	1.5	661	3.3
MOT17-06-DPM	52.2	77.5	49	87	408	5054	57.1	94.3	0.3	170	12.6
MOT17-07-DPM	42.1	78.4	5	21	207	9409	44.3	97.3	0.4	162	7.8
MOT17-08-DPM	26.8	82.5	7	39	138	15204	28	97.7	0.2	113	12.0
MOT17-12-DPM	46.1	82.1	18	40	88	4531	47.7	97.9	0.1	52	27.2
MOT17-14-DPM	32.1	76.2	12	78	375	11975	35.2	94.6	0.5	204	12.9
MOT17-01-FRCNN	42.3	76.6	6	10	216	3458	46.4	93.3	0.5	50	22.3
MOT17-03-FRCNN	67.7	77.6	54	17	2345	30908	70.5	96.9	1.6	576	3.3
MOT17-06-FRCNN	54.5	76.8	58	61	641	4441	62.3	92	0.5	279	12.4
MOT17-07-FRCNN	41.4	78.1	6	19	402	9284	45	95	0.8	220	7.1
MOT17-08-FRCNN	26.3	82.8	9	39	166	15291	27.6	97.2	0.3	106	13.8
MOT17-12-FRCNN	44.7	82.2	16	42	82	4672	46.1	98	0.1	35	23.7
MOT17-14-FRCNN	33.1	75.3	15	72	741	11335	38.7	90.6	1	288	9.3
MOT17-01-SDP	42.4	76.5	6	10	205	3447	46.6	93.6	0.5	62	21.5
MOT17-03-SDP	71.9	76.9	69	15	3463	25187	75.9	95.8	2.3	727	3.0
MOT17-06-SDP	54.6	76.8	66	64	708	4380	62.8	91.3	0.6	262	14.0
MOT17-07-SDP	44.2	77.9	8	18	398	8823	47.8	95.3	0.8	210	5.9
MOT17-08-SDP	27.6	81.8	9	36	221	14951	29.2	96.5	0.4	124	13.0
MOT17-12-SDP	45.5	81.9	17	44	195	4478	48.3	95.6	0.2	52	27.1
MOT17-14-SDP	35.4	75.4	14	66	784	10858	41.3	90.7	1	298	10.1
ALL	55.4	77.8	21.6	34.3	14079	233092	58.7	95.9	0.8	4,684	13.6

Figure 4 shows a qualitative example of the proposed approach in the case of occlusions. When the camera movement is low, the proposed SiameseRF accurately tracks partially hidden or temporarily hidden pedestrians, as shown in Figure 4 (a). However, as shown in Figure 4(b), if the camera movement is large or a full occlusion occurs over the time window, the tracker will go missing (yellow arrow) or the tracker ID will switch to another tracker (red arrow). Therefore, a method to solve the problem of ID switching from long-term occlusions without losing the advantage of fast online tracking should be studied in the future.

A detailed summary of the MOT Challenge benchmark results for the proposed basic and 30% rule distilled SiameseRF tracker is shown in Table 7. For the corresponding results for each sequence of the other trackers mentioned in this study, please refer to the official MOTChallenge webpage available at the MOT Challenge website.[4]

VI. CONCLUSION

In this study, we proposed a new SiameseRF for MOT. SiameseRF is not CNN-based and does not share weights, but instead shares rules that make up the RF. Therefore, unlike a CNN, it showed an excellent object matching performance without requiring many parameters or a long computation time. In addition, the rule distillation algorithm, which can effectively remove the rules that make up the RF, makes SiameseRF lighter and allows a faster matching. We verified the good performance of the proposed SiameseRF through various benchmark datasets and proved that both real-time and online tracking are possible because of the low computational time.

VII. DISCUSSION

Although the SiameseRF-based MOT proposed in this article is run on a CPU, the matching performance of the model is improved and the issue of the existing slow CNN-based tracking is resolved. However, the proposed method is still unsuitable for real-time operation on low-end systems such as limited embedded devices because numerous parameters and operations are still required. Moreover, because the proposed method relies only on appearance features, for a higher tracking performance, the object and camera movements should be additionally considered. In addition, there is a need for an ID switching solution owing to the long-term occlusion of the trackers.

As a future study, we plan to develop a more robust association checking algorithm that can consider both long- and short-term occlusions of the object. In addition, we will focus on reducing the weight and optimizing the algorithm so that it can be installed and used in low-end embedded systems required by autonomous vehicles or robots.

REFERENCES

[1] B. Yang, C. Huang, and R. Nevatia, "Learning affinities and dependencies for multi-target tracking using a CRF model," in *Proc. CVPR*, Jun. 2011, pp. 1233–1240.

[2] Z. Qin and C. R. Shelton, "Improving multi-target tracking via social grouping," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2012, pp. 1972–1978.

[3] 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.

[4] S. Tang, B. Andres, M. Andriluka, and B. Schiele, "Multi-person tracking by multicut and deep matching," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Oct. 2016, pp. 100–111.

[5] J. Son, M. Baek, M. Cho, and B. Han, "Multi-object tracking with quadruplet convolutional neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 3786–3795.

[6] J. Zhu, H. Yang, N. Liu, M. Kim, W. Zhang, and M. H. Yang, "Online multi-object tracking with dual matching attention networks," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 1–17.

[7] K. Bernardin and R. Stiefelhagen, "Evaluating multiple object tracking performance: The CLEAR MOT metrics," *EURASIP J. Image Video Process.*, vol. 2008, pp. 1–10, May 2008.

[8] M. Rodriguez, J. Sivic, I. Laptev, and J.-Y. Audibert, "Data-driven crowd analysis in videos," in *Proc. Int. Conf. Comput. Vis.*, Nov. 2011, pp. 1235–1242.

[9] M. Isard and A. Blake, "Condensation-conditional density propagation for visual tracking," *Int. J. Comput. Vis.*, vol. 29, no. 1, pp. 5–28, Nov. 1998.

[10] 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 Recognit. Workshops (CVPRW)*, Jun. 2016, pp. 418–425.

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

[12] 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.

[13] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. Adv. Neural Inf. Process. Syst.*, Dec. 2015, pp. 91–99.

[14] 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.

[15] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," 2018, *arXiv:1804.02767*. [Online]. Available: http://arxiv.org/abs/1804.02767

[16] H. Law and J. Deng, "CornerNet: Detecting objects as paired keypoints," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 1–17.

[17] 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 Recognit.*, Jun. 2012, pp. 1815–1821.

[18] H. W. Kuhn, "The hungarian method for the assignment problem," *Nav. Res. Logistics Quart.*, vol. 2, nos. 1–2, pp. 83–97, Mar. 1955.

[19] B. Wang, L. Wang, B. Shuai, Z. Zuo, T. Liu, K. L. Chan, and G. Wang, "Joint learning of convolutional neural networks and temporally constrained metrics for tracklet association," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2016, pp. 1–8.

[20] P. Bergmann, T. Meinhardt, and L. Leal-Taixe, "Tracking without bells and whistles," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 941–951.

[21] S. Lee and E. Kim, "Multiple object tracking via feature pyramid siamese networks," *IEEE Access*, vol. 7, pp. 8181–8194, 2019.

[22] P. Chu and H. Ling, "FAMNet: Joint learning of feature, affinity and multidimensional assignment for online multiple object tracking," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 1–10.

[23] M. Kim, S. Alletto, and L. Rigazio, "Similarity mapping with enhanced Siamese network for multi-object tracking," in *Proc. Neural Inf. Process. Syst. Workshop (NIPSW)*, Dec. 2016, pp. 1–8.

[24] J. Lee, S. Kim, and B. C. Ko, "Fast multiple object tracking using Siamese random forest without online tracker updating," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. BMTT Workshop (CVPRW)*, Jun. 2020, pp. 1–4.

[25] R. Sanchez-Matilla, F. Poiesi, and A. Cavallaro, "Online multi-target tracking with strong and weak detections," in *Proc. Eur. Conf. Comput. Vis. BMTT Workshop (ECCVW)*, Oct. 2016, pp. 1–16.

[26] K. Fang, Y. Xiang, X. Li, and S. Savarese, "Recurrent autoregressive networks for online multi-object tracking," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Mar. 2018, pp. 1–10.

[27] S. J. Kim, J.-Y. Nam, and B. C. Ko, "Online tracker optimization for multi-pedestrian tracking using a moving vehicle camera," *IEEE Access*, vol. 6, pp. 48675–48687, 2018.

[28] Z. Wang, L. Zheng, Y. Liu, and S. Wang, "Towards real-time multi-object tracking," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Aug. 2020, pp. 1–17.

[4] In MOT challenge website, SiameseRF is labeled as 'MOTRF' and SiameseRF+ rule distillation is labeled as 'LWTK'.

[29] Q. Zhou, B. Zhong, Y. Zhang, J. Li, and Y. Fu, "Deep alignment network based multi-person tracking with occlusion and motion reasoning," *IEEE Trans. Multimedia*, vol. 21, no. 5, pp. 1183–1194, May 2019.

[30] Q. Zhou, B. Zhong, X. Lan, G. Sun, Y. Zhang, B. Zhang, and R. Ji, "Fine-grained spatial alignment model for person re-identification with focal triplet loss," *IEEE Trans. Image Process.*, vol. 29, pp. 7578–7589, Jun. 2020.

[31] W. Tian, M. Lauer, and L. Chen, "Online multi-object tracking using joint domain information in traffic scenarios," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 1, pp. 374–384, Jan. 2020.

[32] Y. Xu, A. Sep, Y. Ban, R. Horaud, L. Leal-Taixe, and X. Alameda-Pineda, "How to train your deep multi-object tracker," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 6787–6796.

[33] B. Pang, Y. Li, Y. Zhang, M. Li, and C. Lu, "TubeTK: Adopting tubes to track multi-object in a one-step training model," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 1–11.

[34] C. Ma, X. Mu, and D. Sha, "Multi-layers feature fusion of convolutional neural network for scene classification of remote sensing," *IEEE Access*, vol. 7, pp. 121685–121694, 2019.

[35] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual explanations from deep networks via gradient-based localization," *Int. J. Comput. Vis.*, vol. 128, no. 2, pp. 336–359, Oct. 2019.

[36] 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 Recognit. (CVPR)*, Jun. 2005, pp. 539–546.

[37] L. Huang, X. Zhao, and K. Huang, "GOT-10k: A large high-diversity benchmark for generic object tracking in the wild," *IEEE Trans. Pattern Anal. Mach. Intell.*, early access, Dec. 4, 2019, doi: 10.1109/TPAMI.2019.2957464.

[38] 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

[39] *MOT Benchmark*. Accessed: Aug. 3, 2020. [Online]. Available: https://motchallenge.net data/MOT17/

[40] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 9, pp. 1627–1645, Sep. 2010.

[41] 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. 1–9.

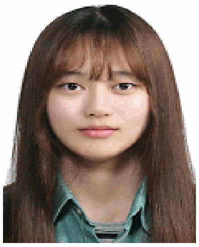
[42] A. W. Beigh, "Kinematic object track stitcher for post tracking fragmentation detection and correction," M.S. thesis, Master Sci. Elect. Eng., Univ. Dayton, Dayton, OH, USA, 2015.

[43] Y. Zhang, H. Sheng, Y. Wu, S. Wang, W. Ke, and Z. Xiong, "Multiplex labeling graph for near-online tracking in crowded scenes," *IEEE Internet Things J.*, vol. 7, no. 9, pp. 7892–7902, Sep. 2020.

[44] Y. Yoon, D. Kim, K. Yoon, Y. Song, and M. Jeon, "Online multiple pedestrian tracking using deep temporal appearance matching association," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshop (CVPRW)*, Jun. 2019, pp. 1–23.

[45] J. Xu, Y. Cao, Z. Zhang, and H. Hu, "Spatial-temporal relation networks for multi-object tracking," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 3988–3998.

[46] K. Yoon, J. Gwak, Y.-M. Song, Y.-C. Yoon, and M.-G. Jeon, "OneShotDA: Online multi-object tracker with one-shot-learning-based data association," *IEEE Access*, vol. 8, pp. 38060–38072, 2020.



JIMI LEE received the B.S. degree in computer engineering from Keimyung University, Daegu, South Korea, in 2018, where she is currently pursuing the master's degree with the Computer Vision and Pattern Recognition Laboratory. Her current research interest includes depth estimation from monocular image.

SANGWON KIM received the B.S. and M.S. degrees in computer engineering from Keimyung University, Daegu, South Korea, in 2018 and 2020, respectively, where he is currently pursuing the Ph.D. degree with the Computer Vision and Pattern Recognition Laboratory. His current research interests include depth estimation from monocular image, explainable AI, and deep model compression.

BYOUNG CHUL KO (Member, IEEE) received the B.S. degree from Kyonggi University, Suwon, South Korea, in 1998, and the M.S. and Ph.D. degrees in computer science from Yonsei University, Seoul, South Korea, in 2000 and 2004, respectively. From 2004 to 2005, he was a Senior Researcher at Samsung Electronics, Suwon, where he worked on the Ubiquitous Robotic Companion (URC) Project on the subject of robot event detection and face recognition using charge-coupled device (CCD) cameras. He is currently a Professor with the Department of Computer Engineering and the Vice Dean of the College of Engineering, Keimyung University, Daegu, South Korea. His current research interests include interpretable machine learning, deep model compression, advance driver assistance systems, and biomedical image processing. He received the excellent paper awards from several conferences. Furthermore, he was selected as the Best Researcher and Lecturer at Keimyung University, in 2013, 2014, 2015, 2018, and 2019. He has served in various international journals and conferences, including a Special Issue Editor for *Sensors* and a Committee Member of ACPR.