IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received October 22, 2019, accepted November 11, 2019, date of publication November 14, 2019,
date of current version November 26, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2953455

Spatio-Temporal Representation Matching-Based Open-Set Action Recognition by Joint Learning of Motion and Appearance

YONGSANG YOON[1], **(Student Member, IEEE), JONGMIN YU**[1,2], **(Student Member, IEEE), AND MOONGU JEON**[1], **(Senior Member, IEEE)**

[1]School of Electrical Engineering and Computer Science, Gwangju Institute of Science and Technology, Gwangju 61005, South Korea
[2]Department of Electrical Engineering, Curtin University, Perth, WA 6102, Australia

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

This work was partly supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea Government (MSIT) (No. 2014-3-00077, Development of Global Multitarget Tracking and Event Prediction Techniques Based on Realtime Large-Scale Video Analysis) and the National Research Foundation of Korea (NRF) grant funded by the Korea Government (MSIT) (No. 2019R1A2C2087489).

ABSTRACT In this paper, we propose the spatio-temporal representation matching (STRM) for video-based action recognition under the open-set condition. Open-set action recognition is a more challenging problem than closed-set action recognition since samples of the untrained action class need to be recognized and most of the conventional frameworks are likely to give a false prediction. To handle the untrained action classes, we propose STRM, which involves jointly learning both motion and appearance. STRM extracts spatio-temporal representations from video clips through a joint learning pipeline with both motion and appearance information. Then, STRM computes the similarities between the ST-representations to find the one with highest similarity. We set the experimental protocol for open-set action recognition and carried out experiments on UCF101 and HMDB51 to evaluate STRM. We first investigated the effects of different hyper-parameter settings on STRM, and then compared its performance with existing state-of-the-art methods. The experimental results showed that the proposed method not only outperformed existing methods under the open-set condition, but also provided comparable performance to the state-of-the-art methods under the closed-set condition.

INDEX TERMS Action recognition, open-set recognition, spatio-temporal representation, joint learning of motion and appearance.

I. INTRODUCTION

Action recognition is one of the most challenging aspects of computer vision research, because the complexity and variety of human behaviors makes recognition difficult. Action recognition studies have attracted increasing attention in recent years, with extensive applications in fields such as real-world surveillance systems [1]–[3] and human biometrics [4]–[6]. During the past few decades, numerous studies have been conducted in efforts to develop and improve methods that can achieve precise action recognition. Among them, several methods have been proposed which recognize human actions based on hand-crafted features using stochastic or deterministic modeling methods, such as dense or sparse extraction of features [7]–[10] and video-level encoding [11]–[14]. Dollár *et al.* [10] proposed a descriptor to characterize the cuboids of spatiotemporally windowed data surrounding a feature point. They assumed that employing direct 3D counterparts for commonly used 2D interest point detectors were inadequate. Willems *et al.* [15] proposed extracting spatio-temporal points of interest, which were scale-invariant (both spatially and temporally) and which densely covered the whole video content. Wang *et al.* [16] proposed building trajectories with optical flow instead of using a Kanade-Lucas-Tomasi feature tracker [17]. However, optical flow is very vulnerable to camera movement, which produces extra background flows. To compensate, Wang and Schmid [18] proposed removing background trajectories by correcting the camera motions.

In recent years, deep neural networks have shown outstanding performance for extracting features in various vision areas, such as instance segmentation [19], [20],

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

object detection [21], [22] image classification [23], [24], and pose estimation [25]–[28]. The CNN feature, which is a more sophisticated and deeper representation of visual information, has also led to great improvements in action recognition [29].

However, understanding the temporal context still poses a huge difficulty. In order to address this issue, two different approaches have been proposed. Karpathy *et al.* [30] introduced a single stream network which first extracts spatial information from 2D CNN features and then fuses them later in four different ways to acquire temporal information. Simonyan and Zisserman [31] proposed a two-stream network which captures both appearance and motion information simultaneously. Instead of a single network for spatial context, [31] explicitly captures motion features from the stacked optical flow vectors.

Most methods try to learn spatio-temporal features by utilizing 2D convolution operation, while other works such as [32] have tried to utilize 3D convolution. 3D convolution shows great improvements in capturing spatio-temporal information. However, it is more difficult to train than the usual 2D-CNN since 3D-CNN has many more parameters in the convolution network. Most works based on deep neural networks [31]–[34] have outperformed conventional methods based on hand-crafted features [8]–[10].

Despite the great strides made by the above studies, *open-set* action recognition is still complicated because it requires managing some actions which are not trained. Fig. 1 illustrates the differences in the recognition task under open-set and closed-set conditions. While action recognition is a difficult problem in itself because of the complexity and variability of human actions, the open-set condition makes action recognition even harder because it contains the unconfined action category.

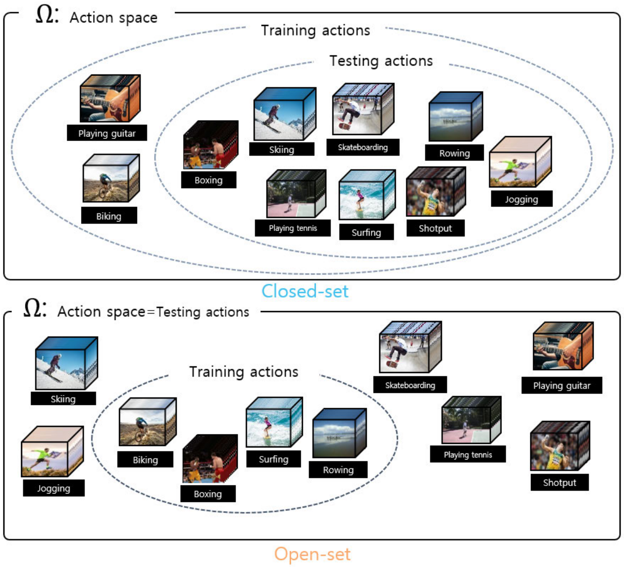


FIGURE 1. A comparison of closed-set and open-set conditions are illustrated. The upper and lower diagrams describe the closed-set condition and open-set condition states, respectively. In the closed-set condition, the action class set is a subset of the action class training set, which means that only trainable action classes appear in the action for testing. In the open-set condition, on the other hand, the set of training actions is a subset of the entire set of actions. Every action including the training action can be given as a testing action, even when it is not learned.

To resolve this issue, we propose a spatio-temporal representation (ST-representation) matching (STRM) method based on joint learning of motion and appearance. In the training stage, STRM jointly learns spatio-temporal features from appearance and motion of the training samples in kinetics-600 [35] only. After STRM is fully trained, the action gallery is constructed with joint ST-representations extracted from the training samples in other datasets (*e.g.* UCF101, HMDB51). The open-set action recognition process using STRM is as follows: Initially, STRM extracts joint spatio-temporal representa- tions (joint ST-representations) from a given video. Then, STRM computes the similarity between the extracted repre- sentation and the representations stored in the action gallery. Next, STRM gives the action label of the video that is most similar to the given video. This approach allows the untrained action classes to be recognized without re-training the model.

The key contributions of this work can be summarized as follows:

- First, we propose a novel method for open-set action recognition which is able to recognize unseen action classes in the training step. The proposed method extracts spatiotemporal representations by jointly learning appearance and motion information, and can improve the discriminative power when extracting the spatiotemporal representation.
- Second, we provide a method for learning the joint spatiotemporal representation of motion and appearance data. Our learning method provides a more discriminating feature extraction function than existing methods.

For demonstration and verification, we present extensive experimental results of action recognition under closed-set and open-set conditions. The experimental results include a performance evaluation involving STRM hyper-parameters, and comparisons of STRM performance with existing state-of-the-art methods.

This paper is organized as follows. In Section II, we discuss related works and existing action recognition research, including visual recognition studies for the open-set condition. Then, we explain the proposed method for open-set action recognition in Section III. In Section IV, we present the experimental setting, datasets, and quantitative comparisons with hyper-parameters and other existing methods, followed by a summary and conclusions in Section V.

II. RELATED WORKS

In this section, several existing studies of human action recognition under closed and open set conditions will be discussed. The general approach of video based action recognition will be discussed in section II-A. Action recognition utilizing both hand-created features and deeply-learned feature under the closed-set condition will be introduced in section II-B. Also, a few action recognition studies under the open-set condition will be introduced in section II-C.

A. GENERAL ACTION RECOGNITION

Human action can be defined as a series of consecutive small unit actions which involve simple movements of the limbs (such as bending the knees or raising an arm) performed by actors. For action recognition to succeed, it is crucial to consider both spatial and temporal information. Human action recognition has been studied extensively using various approaches to extract distinctive features across RGB frames, and then combine them into video-level representation, to understand the entire context.

Although spatial information is usually obtained from RGB frames, many recent works [36], [37] have utilized a skeleton model, which can be estimated using pose estimation (*e.g.* [25]–[28]). Analyzing the skeleton model has a big advantage in that action can be clearly captured by focusing on the movements of joints. Nonetheless, in this paper, we will focus on video based action recognition since most visual sensors in the real world are RGB cameras, and the method is still important.

B. CLOSED-SET ACTION RECOGNITION ON VIDEOS

An action class is defined specifically as a certain typical behavior, such as playing soccer, playing a guitar, or putting on makeup. The goal of video based action recognition is to classify the actions performed in a given video clip. These actions are trained in advance. However, this means that the model can only classify trained actions. This condition is called a closed-set problem. Most works have focused on this condition.

The process employed by the majority of traditional studies on video-based action recognition can be described in three steps. 1) The high dimension visual features which describe the local area of the video are extracted either sparsely [8]–[10] or densely [7], [16] in high dimension. 2) The extracted visual features are merged into a fixed length description such as video-level or clip-level. One of the most popular approaches has been bag-of-visual words, which is formed using clustering methods (such as DBSCAN or *K*-means clustering) to represent the video. 3) A classifier, such as Random Forest (RF) [38] or Support Vector Machine (SVM) [39] is trained on the fixed length feature (i.e., BoVW) for the final prediction.

Hand-crafted feature based approaches usually extract a local spatial-temporal feature from around the trajectories or region of interest, such as 3D-Hessian [15], Cuboids [10], 3D-Harris [8]. It is also common to extract feature trajectories with matching SIFT descriptors or KLT trackers across the frames. However, this approach is often insufficient to represent motions. To overcome the lack of trajectories, Wang *et al.* [16] proposed a novel descriptor named the Dense trajectory based on motion boundary histograms. In it, the feature points are densely sampled from each frame first and then tracked based on the optical flow field to capture both appearance and motion information. Subsequently, the improved Dense Trajectory (iDT) [18] method was proposed to improve the estimation of trajectories by correcting the camera motion, thus removing the background trajectories.

This led to a significant improvement in iDT performance, which is still comparable with recent works. A histogram descriptor is computed to capture both appearance and motion information from the points of interest, such as a Histogram of Gradient and Histogram of Flow (HOG/HOF) [9]. While conventional methods have focused on combining independent points of interest from multiple frames to form a local descriptor, advanced aggregation approaches proposed recently have utilized dense trajectories such as Fisher Vector (FV) [11], improved Dense Trajectory (iDT) [18].

Deeply-learned feature-based approaches (*e.g.* convolutional neural networks (CNNs)) have produced remarkable improvements in action recognition in the last few years. Many works utilized a CNN feature map to obtain abundant visual information from a frame. Reference [32] used 3D convolutions on the video volume to extract both spatial and temporal volumes simultaneously, rather than using 2D convolutions across the frames. However, most of these works have only considered visual information, excluding motion information, since it is not easy to train a CNN model to learn motions. Later, Simonyan and Zisserman [31] proposed a two-stream network approach to learn motion features explicitly from stacked optical flow vectors. Instead of using a single network just for visual context, this architecture uses two separate networks - one for visual context (RGB video) and the other for motion context (Optical flow). The two streams are trained separately and combined later, using SVM for the final prediction. Motivated by [31], many subsequent works have followed this two-stream approach. Zhu *et al.* [33] proposed a hidden two-stream network which has an additional network named the motion-net. The motion net captures motion information between adjacent frames directly, instead of using a conventional optical flow algorithm. With this motion net, the framework is trainable in an end-to-end fashion, making training much faster than using the original optical-flow. Several recent methods have focused on modeling a long-range temporal structure using combination of 2D convolution and Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) such as [34], [40]. Generally, these methods extract visual information from continuous video frame sequences and directly feed it to RNN and LSTM to model the temporal structure.

C. OPEN-SET ACTION RECOGNITION

Most existing works were developed under the closed-set condition, in which only trained actions can be classified. Thus, these existing works will often fail to give a correct prediction in the real-world situation since some actions have not been trained. Furthermore, it is difficult to define every

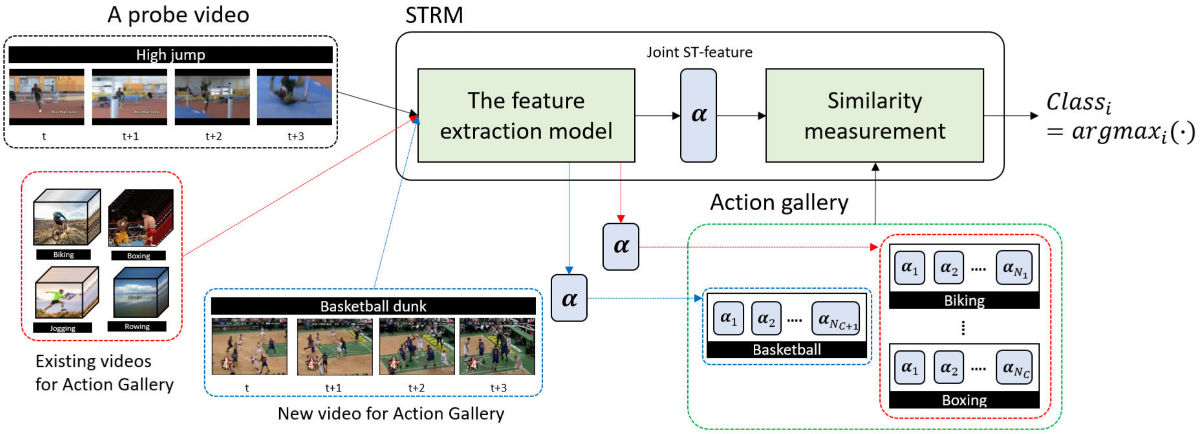

A probe video
High jump
t
t+1
t+2
t+3
STRM
Joint ST-feature
The feature extraction model
α
Similarity measurement
$Class_i = argmax_i(\cdot)$
Action gallery
α
α
Biking
Boxing
Jogging
Rowing
Existing videos for Action Gallery
Basketball dunk
t
t+1
t+2
t+3
New video for Action Gallery
α_1 α_2 .... $\alpha_{N_{C+1}}$
Basketball
α_1 α_2 .... α_{N_1}
Biking
⋮
α_1 α_2 .... α_{N_C}
Boxing

FIGURE 2. The workflow of STRM for open-set action recognition. Solid black arrows represent the process for action recognition. The red dotted box is a set of videos for the action gallery and the red arrows are representing the flow of constructing the action gallery. The blue dotted box is a video samples for new action class and blue lines are representing the flow of appending new action class to existing action gallery. N_j denotes the number of samples of j^{th} action class and C is the number of total action classes in current action gallery.

kind of action and to collect valid video samples in the real world. This condition is called an open-set problem, and has been drawing increasing attention in recent years.

Among the various approaches recently proposed for the open-set condition [41]–[48], the meta-learning method by [42] discriminates an untrained class from a trained class and learns it as a new class (unknown class), making the model learn a new class (unknown class) without re-training the model. Bendale and Boult [41] proposed SVM-based recognition by extending Nearest Class Mean type algorithms [9], [15] to a Nearest Non-Outlier(NNO) algorithm to learn new classes continuously. The Open Deep Network (ODN) proposed by Shu *et al.* [43], on the other hand, is able to detect the class of a given sample, whether it is learned or not, then the model dynamically re-builds itself by adding a new category in the classification layer. Reference [43] employed four steps to recognize actions in the open-set condition: 1) ODN first determines whether a class of given sample is known or unknown by applying multi-class triplet thresholding. The intra-class association is combined with triplet thresholding because a single threshold value is not enough to handle differences in different actions. 2) If the class of the sample is unknown, it is labeled manually. 3) ODN is updated using transfer learning with very few annotations without retraining the entire systems. 4) ODN is fine-tuned with known and new class samples obtained in the second step. Shu *et al.* [44] introduced prototype learning to ODN [43] to improve its robustness in detecting unknowns and updating deep neural networks. The prototype learning provides *prototypes* which are concise representations for each known action class, and *prototype radius* which is a certain range for regularization for prototypes. Mishra *et al.* [47] proposed a generative based zero-shot action recognition. It utilizes the relationship between attributes of the action class which are represented as a probability distribution in the visual space with Word2Vec embedding [49] and synthesizes unseen class data from the learned action classes.

III. SPATIO-TEMPORAL REPRESENTATION MATCHING (STRM)

In this section, we describe the learning and action recognition processes in the STRM method. Section III-A explains the extraction method for joint ST-representation for motion and appearance. We describe the open-set action recognition process using STRM in Section III-B and explain the training process and computational complexity of STRM in Section III-C.

A. JOINT SPATIO-TEMPORAL REPRESENTATION EXTRACTION

In contrast to most existing action recognition methods based on the classification approach [50]–[53], STRM employs a verification approach which computes similarities between the given multiple inputs and selects the pair with the highest similarity. Fig. 2 shows the methodological detail of STRM. As shown in Fig. 2, STRM can be regarded as a framework which consists of a joint representation extraction of extraction model $\mathcal{F}_e$ and the similarity computation part, where each model is composed of the corresponding parameters such as weights and biases. To recognize an action class of given video clip v, STRM first extracts the joint ST representation using the given motion v_m and appearance v_a data as follows:

$$\alpha = \mathcal{F}_e(v; \theta_e) = \mathcal{F}_e(v_m, v_a; \theta_e), \tag{1}$$

where α is an extracted joint ST-representation from v_m and v_a, and θ_e is a set of parameters corresponding to the model for extracting the joint representation. We also utilized the original dense optical flow and raw RGB frame for v_m and v_a, respectively, as in many previous works.

The joint representation extraction model is composed of CNN [54] and LSTM [55] for extracting the spatial model f_s and temporal feature model f_t, respectively. In the joint representation extraction model $\mathcal{F}_e$, spatial features are extracted from every frame in a given video clip using the CNN model. This is represented as follows:

$$o_s = f_s(v_m, v_a; \theta_s), \tag{2}$$

where o_s is the output of the CNN, and θ_s is a set of parameters corresponding to the CNN. In this work, we employed various CNN models to extract the spatial features. The CNN models we used in the experiment were: VGG-19, ResNet-34, and DenseNet-40. We evaluated these three CNN models and compared their performances to each other, and the experimental results are presented in Section IV.

To learn discriminative features which cover both the complexity and the diversity of human actions, we attempted a joint learning method for motion and appearance based on the convolutional neural network. Fusing of multiple data is a commonly used approach in visual recognition studies for scene segmentation [56], [57], object detection [58], and event detection [59]–[61]. In recent action recognition studies [50]–[53], several have reported that using multiple information can provide better action recognition performance than using a single data type only.

In extracting spatial features with the CNN model, the proposed kernel-level fusion is utilized to extract joint representation. Since STRM learns two heteromorphic data simultaneously, it is essential to regularize these data to calculate the gradients and stabilize network learning. STRM initially regularizes the data by using the expectation and variation of data, and then normalizes it using the min and max values of the regularized data. The above normalization process $\mathcal{N}$ is represented as follows:

$$\mathcal{N}(v) = \frac{\frac{v-\mu}{\sqrt{\sigma^2}} - \min(\frac{v-\mu}{\sqrt{\sigma^2}})}{\max(\frac{v-\mu}{\sqrt{\sigma^2}}) - \min(\frac{v-\mu}{\sqrt{\sigma^2}})} = \bar{v}, \tag{3}$$

where v is the input of the normalization process and can be regarded as a batch of training dataset (v_m, v_a) or an output o of an arbitrary layer in CNN; and μ and σ are the expectation and standard variation of v respectively. After normalizing both motion and appearance with the above scheme, they are combined into a joint representation using a fusion task.

The kernel-level based on CNN is defined as follows:

$$o = \gamma(\mathcal{N}(v_m) \odot W_m + \mathcal{N}(v_a) \odot W_a + b), \tag{4}$$

where o is an output of an arbitrary layer applying the fusion task with convolutional operation $\odot$, and γ is the activation function, such as a rectified linear (ReLu) unit [62], softmax function, and sigmoid function, which is a typical operation for CNNs. W_m, W_a are the weights for motion and appearance data, respectively, and b denotes a bias of this fusion layer. With this kernel-level fusion operation in extracting spatial features, STRM can consider both motion and appearance information simultaneously, without any additional functions. According to Feichtenhofer *et al.* [53], depending on the position of the fusion layer, the fusion method can be categorized into two types: early fusion and late fusion. We conducted experiments on these fusion types and discuss their performances in Section IV-C. A set of extracted spatial features $\boldsymbol{o_s} \in \{o_s^1, o_s^2, o_s^3, \ldots, o_s^n\}$, where o_s^i is the spatial feature of the i^{th} frame, is applied to the temporal feature extraction model based on the recurrent network using LSTM. The goal of the recurrent network is to discover temporal characteristics from the sequences of spatial features. The temporal feature extraction model is represented by

$$\boldsymbol{\alpha} = f_t(\boldsymbol{o_s}; \theta_t), \tag{5}$$

where θ_t is a set of parameters corresponding to the temporal feature extraction model based on the recurrent network. Our recurrent network is composed of LSTM cells. Generally, a LSTM cell has the following three gates: 1) input gate i_t, 2) forget gate g_t, 3) output gate c_t.

The input gate i_t computes the weight to determine the influence of the new input on the current internal state s_t at time t. The activation function of the input gate has the following recurrent form:

$$i_t = \sigma(W_{is}s_{t-1} + W_{ih}\alpha_{t-1} + W_{ix}o_s^t + b_i), \tag{6}$$

where $\sigma(\cdot)$ is the sigmoid function which regulates the input to a value between 0 and 1. This means that when the value is close to 1, the input feature o_s^t becomes more important. W_{is} and W_{ih} are weight matrices corresponding to the state s_{t-1} and hidden state α_{t-1}. W_{ix} is the weight matrix for the spatial feature o_s, and b_i is the bias.

The forget gate g_t modulates the previous state s_{t-1} to control its contribution to the current state. It is defined as

$$g_t = \sigma(W_{gs}s_{t-1} + W_{gh}\alpha_{t-1} + W_{gx}o_s^t + b_g), \tag{7}$$

where W_{g*} denotes the weight matrices for $s_{t-1}, \alpha_{t-1}, o_s^t$ and b_g denotes the bias. Using these inputs s_{t-1}, α_{t-1}, and o_s^t, and forget gate units g_t, the internal state s_t of each LSTM cell is updated as follows:

$$s_t = g_t \otimes s_{t-1} + i_t \otimes \tanh(W_{sh}\alpha_{t-1} + W_{sx}o_s^t + b_s), \tag{8}$$

where $\otimes$ indicates the element-wise product and W_{s*} denotes the weight matrices related to the hidden state α_{t-1} and spatial feature o_s^t.

The output gate c_t determines the influence of the current state on the future state. It is defined as

$$c_t = \sigma(W_{cs}s_t + W_{ch}\alpha_{t-1} + W_{cx}o_s^t + b_c), \tag{9}$$

where W_{c*} denotes the weight parameters corresponding to s_t, h_{t-1}, and o_s^t and b_c denotes a bias of this gate. The hidden state of a memory cell is estimated as

$$\alpha_t = c_t \otimes \gamma(s_t), \tag{10}$$

where γ is the ReLu activation function.

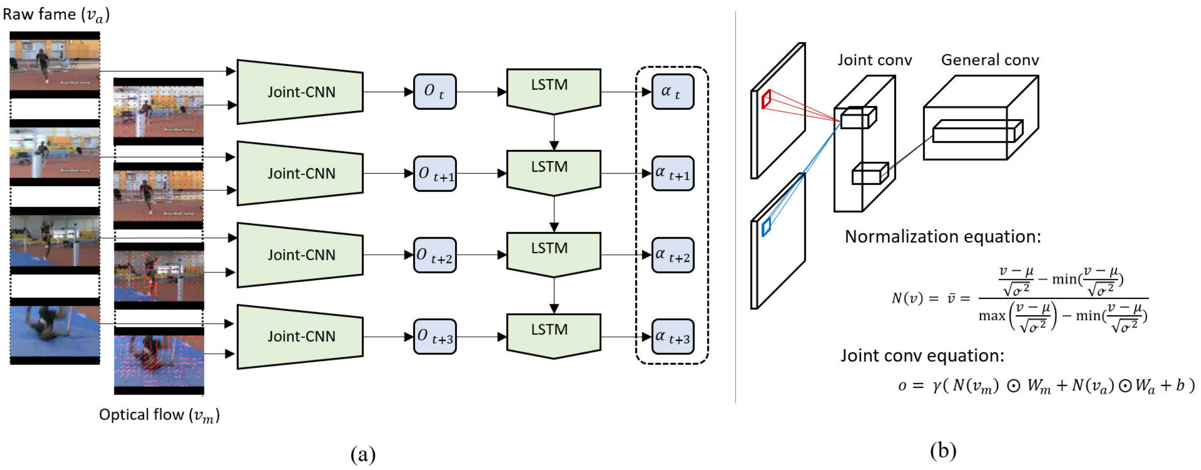


FIGURE 3. (a) illustrates the structural detail of the representation extraction model for the joint ST-representations. (b) shows the conceptual image for the joint convolutional layer. In (a), green colored objects denote the functional components of the model, and the blue colored objects represent the extracted result from each component.

The recurrent network of STRM performs the above operations to extract the joint ST-feature $\boldsymbol{\alpha} \in \{\alpha_1, \alpha_2, \alpha_3, \ldots, \alpha_N\}$, where N is the number of LSTM cells, and α_i is the output of the i^{th} LSTM cell. Note that the number of elements α_i in the joint ST-feature will vary depending on the number of frames in each video clip. For consistency in the joint ST-feature, we regulated the number of representations by selecting key representations which can describe the context of the video well, using K-means clustering [63]. This process can be represented as follows:

$$\text{argmin}_{\mu^c} \sum_{i=1}^{K} \sum_{\alpha_t \in \boldsymbol{\alpha}} ||\alpha_t - \mu_i^c||,$$

where μ^c is the centroid of the clustering method, and K is the number of centroids applied in the clustering method. As a result, the key ST-representation $\boldsymbol{\mu^c} \in \{\mu_1^c, \mu_2^c, \mu_3^c, \ldots, \mu_K^c\}$ is obtained by STRM. The entire process of the ST-representation extraction using STRM is given below.

$$\mathcal{F}_e(\boldsymbol{v}) = \mathcal{F}_e(\boldsymbol{v_m}, \boldsymbol{v_a}) = f_k(f_t(f_s(\boldsymbol{v_m}, \boldsymbol{v_a}; \theta_s); \theta_t)) = \boldsymbol{\alpha}, \tag{11}$$

where f_k is K-means clustering. The key joint ST-representation α is applied to the similarity measurement model for open-set action recognition. Fig. 3 represents the structural detail of the representation extraction model of STRM. The process for open-set action recognition with STRM is described in the following section.

B. OPEN ACTION RECOGNITION USING STRM

In contrast to the general action recognition methods based on a classification approach which recognizes action via the probabilities of classifiers, STRM employs a verification approach based on the similarity measurement. If the similarity between two representations is high enough, they will likely belong to the same action class. Based on this assumption, the STRM verification process is as follows. 1) STRM extracts key joint ST-representations from anchor video clips v_p using the feature extraction model $\mathcal{F}_e$ to construct an action gallery. The anchor video clips consist of N action classes, and each action class has M action samples. Thus, the action gallery will have $N{*}M$ representations $\bar{\boldsymbol{\alpha}}$. 2) STRM extracts key joint ST-representations from a given video clip (probe) v_p using the feature extraction model $\mathcal{F}_e$. 3) Similarities between the probe representation $\boldsymbol{\alpha}^p$ and every representation $\bar{\boldsymbol{\alpha}}_{\boldsymbol{i},\boldsymbol{j}}$ are stored in the action gallery, where i and j indicate the index of representation. 4) In each action class, the representation $\bar{\boldsymbol{\alpha}}_{\_,j}$ which has the highest similarity is selected. As a result, N representations $\bar{\boldsymbol{\alpha}}_{i,\_}$ are the candidate action class for a given video. 5) The representation which has the highest similarity among candidates is selected for the final prediction. 6) STRM predicts the action of the probe video clip as the action class of the selected representation. This process can be represented as follows:

$$class_i, i = \text{argmax}_i(\text{argmax}_j \mathcal{D}(\boldsymbol{\alpha_p}, \alpha_{i,j})),$$

where $\mathcal{D}$ is a similarity function (*i.e.*, l_2-norm), i is the index of the action class, and j is the index of the representation in each action class. $\boldsymbol{\alpha_p}$ denotes the extracted ST-representation from a given input video clip. $\alpha_{i,j}$ is the j^{th} ST-representation of the i^{th} action classes stored in the action gallery. In this work, we employed l_2-norm to compute the similarity.

This recognition process based on the similarity measure allows us to recognize action classes which are not included in the training dataset more effectively than the existing approaches (*e.g.* [50]–[53], [64], [65]). These existing methods are only able to classify trained action classes, since their classifier is fixed in the training step (*e.g.* a fixed-length fully-connected layer in the CNN model), which requires model retraining or parameter modification if they are asked to classify a new action class. In the verification task in STRM, on the other hand, just adding some ST-representations of new

Algorithm 1 Action Recognition Using STRM

```
Input: Video clips $[v_1, v_2, \dots v_P]$
Output: Class labels $[label_1, label_2, \dots, label_P]$
for $p = 1$ to $P$: given video clips do
    · Spatiotemporal feature extraction
      $\boldsymbol{\alpha_p} = \mathcal{F}_e(v_m^p, v_a^p)$
    for $i = 1$ to $I$: action classes do
        for $j = 1$ to $J$: video clips do
            · Spatiotemporal feature extraction
              $\boldsymbol{\alpha_{i,j}} = \mathcal{F}_e(v_m^{i,j}, v_a^{i,j})$
            $\boldsymbol{s_{i,j}} = \mathcal{D}(\boldsymbol{\alpha_p}, \boldsymbol{\alpha_{i,j}})$
        end for
        Select the highest similarity in $i$-$th$ class videos.
        $s_i = \max([s_{i,1}, s_{i,2}, \dots, s_{i,J}])$
    end for
    Select action class where the highest similarity $s_i$
    belongs to
    $i = \text{argmax}_i([s_1, s_2, \dots, s_I])$
    $label_p = action_i$
end for
return Class labels $[label_1, label_2, \dots, label_P]$
```

Algorithm 2 Joint Learning of Motion and Appearance by STRM

```
Input: $(v^a, v^p, v^n, a^a, a^p, a^n)$, where $v^a$, $v^p$, and $v^n$ are anchor
    video, positive video, and negative video, while $a^a$, $a^p$, and
    $a^n$ are the corresponding labels for these videos respectively.
Output: The optimized network parameters $\boldsymbol{\theta} = \{\boldsymbol{W}, \boldsymbol{b}\}$,
    where $\boldsymbol{W}$ and $\boldsymbol{b}$ are the sets of weight and bias parameters
    of the model.
    for The number video set in the training dataset do
        · Optical flow extraction
        Extract optical flow $v_m$ from video $v_a$
        · Key joint ST-representation extraction
        $\mathcal{F}_e(v_m, v_a) = f_t(f_s(\boldsymbol{v_m}, \boldsymbol{v_a}; \theta_s); \theta_t) = \boldsymbol{\alpha}$,
        $\boldsymbol{\alpha} = \{\alpha^a, \alpha^p, \alpha^n\}$
        · Compute the classification result
        $\mathcal{F}_{cls}(\boldsymbol{\alpha}) = \boldsymbol{\bar{a}}, \boldsymbol{\bar{a}} = \{\bar{a}^a, \bar{a}^p, \bar{a}^n\}$
        · Loss computing
        $\mathcal{L} = \mathcal{L}_{tri}(\alpha_a, \alpha_p, \alpha_n) + \frac{\lambda}{2} \sum_{v_*} \mathcal{L}_{cls}(\bar{a}^*, a^*)$
        · Update parameters
        $\theta = \theta + \gamma \frac{\delta \mathcal{L}}{\delta \theta}$, where $\gamma$ is a learning rate.
    end for
    return $\boldsymbol{\theta} = \{\boldsymbol{W}, \boldsymbol{b}\}$
```

action classes is enough, making it very straightforward and simple. Methodologically, this approach is inspired by the Siamese network [66] which measures the similarity between two input images for matching.

The computational complexity for action recognition using STRM is $O(N^2)$, where N is the number of both the action classes and videos in each class. Note that the computation cost for extracting the ST representation $\boldsymbol{\alpha}$ from $F_e(v_m, v_a)$ is not considered. Instead, only the similarity function ($e.g.$ l_2-norm) between the given video clips and every video in the action gallery is considered. The process is described in Algorithm. 1. Consequently, the final time complexity of our proposed model for N given videos is $O(N^3)$.

C. LEARNING STRM

To recognize action under the open-set condition, STRM employs a verification approach, unlike the conventional action recognition approaches [31], [33], [53], [67] which handle the action recognition problem as a classification problem. By measuring the similarity of two given input data types, the verification approach provides a class- invariant recognition method.

In this work, a triple loss function is utilized to train STRM as follows:

$$\mathcal{L}_{tri}(v^a, v^p, v^n) = \sum_{i=1}^{N} [\|\mathcal{F}_e(v^a) - \mathcal{F}_e(v^p)\|_2^2 - \|\mathcal{F}_e(v^a) - \mathcal{F}_e(v^n)\|_2^2 + \delta], \quad (12)$$

where v^a, v^p, and v^n are anchors, positive and negative video samples, respectively. There is a margin between the positive and negative pairs. The goal of triplet loss is straightforward, namely, to minimize the distance from anchor v^a to positive input v^p and to maximize the distance from anchor v^a to negative input v^n. To train STRM with triplet loss, the anchor video v^a of an arbitrary action class is selected first. Next, the positive sample v^p and the negative sample v^n are selected randomly from video samples in the same action class and in a different action class as the anchor, respectively. Although triplet loss enables the model to learn discriminative features between classes, it is still hard to distinguish between representations of some actions even when they belong to different action classes. To minimize such misleading representations, we added a classification loss to the existing loss function to learn the distinctive features of each action class. The classification loss is estimated with a fully-connected layer attached to the end of the LSTM cell as a classifier. The classification loss using cross-entropy is defined by

$$\mathcal{L}_{cls}(\mathcal{F}_{cls}(\alpha), \bar{a}) = -\sum_{i=1}^{C} a_i \log(\mathcal{F}_{cls}(\alpha)_i), \quad (13)$$

where $\mathcal{F}_{cls}$ is a simple classification model which consists of fully connected layers with a softmax function, and $\mathcal{F}_{cls}(\dot{)}_i$ denotes the i^{th} unit of the output of the classification model. α, C and $\bar{a}$ are the extracted joint ST-representation, the number of action classes in the training dataset and the given label corresponding to an input video, respectively. This combining loss function is a commonly used approach in various visual recognition studies [61], [68], [69]. According to [68], the center loss, which simultaneously learns the center for the deep features of each class and penalizes the distances between the deep features and their corresponding class centers, showed better performance than conventional loss

TABLE 1. Dimentionality of LSTM on STRM. 'Dim' represents the dimensionality of the input, state and the output units.

	Input units	State units	Output units
Dim	256	512	256

functions such as softmax loss. The final loss is defined as

$$\mathcal{L} = \mathcal{L}_{tri}(v^a, v^p, v^n) + \frac{\lambda}{2} \sum_{v^a, v^p, v^n \in \boldsymbol{v}} \mathcal{L}_{cls}(\mathcal{F}_{cls}(\mathcal{F}_e(v^*)), \bar{\alpha}^*), \quad (14)$$

where λ is a hyper-parameter to regulate the weight of the classification loss. We set the λ value at 0.01 which showed best performance. We empirically determined the value of λ by repeating the experiments. v_* and $\bar{\alpha}^*$ denote the input video clips v^a, v^p, and v^n and the corresponding annotation for each videos. Algorithm 2 shows the entire STRM training process.

IV. EXPERIMENTS

In this section, we present the experiments used to demonstrate the effectiveness of the proposed action recognition method under the open-set condition. In Section IV-A, we explain the experimental settings, datasets, and the experimental protocol for open-set action recognition. In section IV-B, we describe the performance analysis dependent on the hyperparameter setting, and its comparison with existing methods. In Section IV-C, the performance of STRM is compared with those of other existing methods for closed-set and open-set conditions.

A. EXPERIMENTAL SETTING AND DATASET

Experiments were carried out to analyze the action recognition performance when it was dependent on the setting of STRM hyperparameters, and compared with existing action recognition methods for closed-set and open-set conditions. In the representation model, we applied several existing network models: VGG-19 [70], ResNet-34 [23], and DenseNet-40 [24].

All of the CNNs based models were trained with stochastic gradient descent (SGD). We employed a learning rate decay of 0.0001 and momentum of 0.9. The learning rate was initialized to 0.1, and divided by 10 in 20, 40, and 60 epochs. The batch size was set at 16 when training the models, and the setting details for the LSTM are described in Table 1. All experiments were carried out using an Nvidia Titan Xp GPU and 3.20 *Ghz* CPU. The source codes for these experiments were implemented based on the Tensorflow library.

The experimental protocol for the open-set action recognition was designed in three steps: 1) Model training, 2) Action gallery construction, and 3) Performance evaluation. We referred to the experimental protocols for unconstrained face recognition [80] and person re-identification [81] for the protocol design. In the model training step, STRM was trained using just the training set in the Kinetics-600 dataset [82] which contains a total of around 500k videos collected from the Youtube website. After STRM was fully trained with the Kinetics-600 dataset, action gallery construction and performance evaluation were carried out with the UCF101 dataset and HMDB51 dataset. These two datasets provide three train/test splits in the experimental setup for videos, to distinguish whether a video is assigned for training or testing. In this experimental protocol, the splits for training were used to construct an action gallery, and the others were assigned for the model evaluation step. For instance, when the evaluation was performed with the UCF101 dataset, the action gallery was constructed with the training part of the given split in UCF101, and performance was then evaluated using the test part of the split. The details of these datasets are described below.

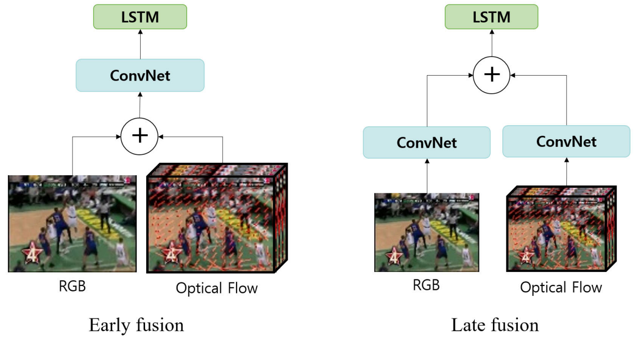


FIGURE 4. The details of the fusion approaches are illustrated. ***Left***: In the early-fusion type, two different input data types are combined before the convolutional operation. ***Right***: In the late-fusion type, two different input data types are combined after the convolutional operation. These interpretations for fusion approaches can be referenced in Feichtenhofer *et al.* [53].

Kinetics dataset [35] is a large-scale and human-focused action dataset collected from Youtube videos which includes a wide range of actions in high quality video. Kinetics-600, which has 600 human action categories, contains training and validation videos of around 392 and 30K, respectively. The length of every video clip is at least 10 seconds. The action categories in this dataset cover a wide range of actions including interactions between humans, such as playing instruments in addition to interactions between humans and objects such as applying facial cream, baking cookies, and base jumping.

HMDB dataset [9], [83], [84] contains 6.8K RGB videos collected from a wide range of sources such as Youtube and movies. Each video is labeled into 51 distinctive action classes (e.g. running, walking, climbing, jumping, a person kicking a ball etc) and each class contains at least 101 video clips. Videos in this dataset were trimmed to have a playtime of less than 10 seconds. To ensure the consistency of video clips, more than two human observers validated each video clip.

UCF101 dataset [85], which was published in 2012, is a challenging dataset because it contains large variations in camera motion and object appearance as well as a cluttered background. It contains 13K RGB videos collected from Youtube, from which 9.5K videos were used for training and 3.5K videos for testing. These videos have a wide range of

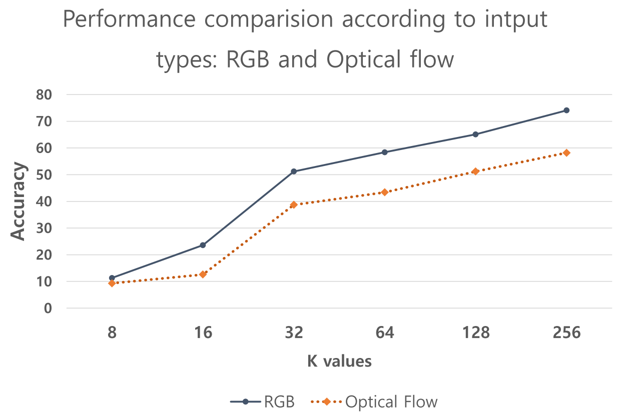


FIGURE 5. Action recognition performance on the UCF101 dataset depending on the input types and the K values. The performances were evaluated using a single input type instead of using joint learning. The solid blue line indicates the recognition performance of STRM using only RGB, while the dotted red line indicates the recognition performance of STRM using only optical flow.

playing times and have been labeled into 101 action classes, including various instrument performances and sports activities. This dataset was also divided into 25 groups with each group sharing a common feature such as a similar viewpoint or background.

B. PERFORMANCE ANALYSIS DEPENDING ON HYPER-PARAMETER SETTINGS

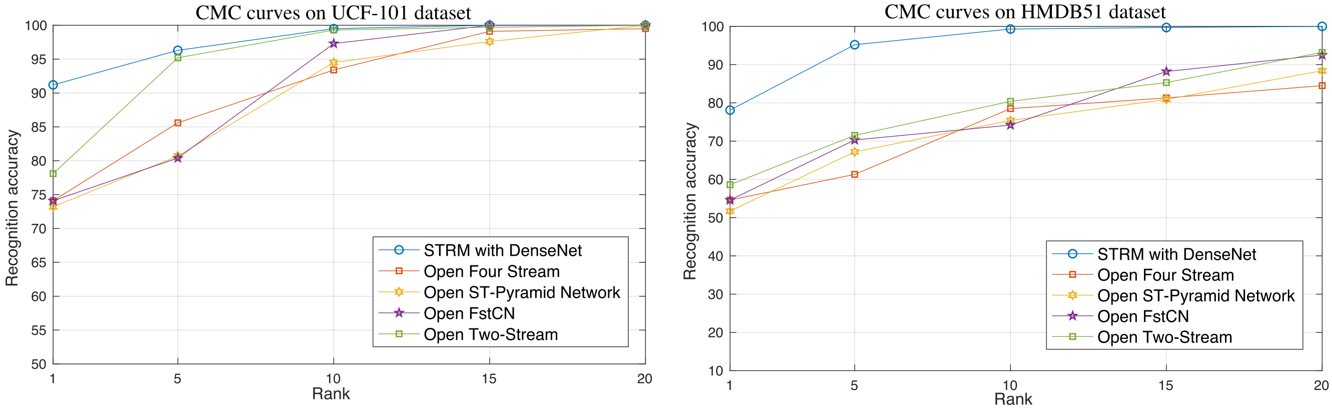
TABLE 2. Action recognition performance on the UCF101 dataset based on the fusion approaches and the values of K.

K Fusion	8	16	32	64	128	256
Early fusion	17.3	41.6	53.2	83.6	89.2	89.4
Late fusion	21.6	45.4	61.0	85.4	91.2	91.6

Since STRM extracts ST-representation by jointly learning motion and appearance information, we carried out an experiment to investigate how each type of information affected recognition performance. We modified the STRMs into a single pipeline structure and trained the STRMs with RGB and optical flow separately. Fig. 5 shows the action recognition performances depending on the input types and the K values. The STRM with RGB achieved better performance than optical flow, regardless of K values. The experimental results indicate STRM can capture the proper temporal context when only appearance information is given, but optical flow is not enough to capture temporal context for action recognition in the open-set condition.

The experiments were then conducted on multiple K values and two different fusion types using the UCF101 dataset, to investigate the influence of K values and fusion approaches on the accuracy of action recognition. As shown in Table 2, the minimum and maximum values of K were 8 and 256, respectively, and the K value doubled from 8 to 256. The performance increased dramatically and growth rate was higher than 10% until the K value reached 64. It then started to decrease slowly, exhibiting a performance of 83.6% at $K = 64$, and only changed slightly when the K value doubled from 128 to 256.

Experiments were also conducted using early fusion and late fusion. The structural difference between early and late fusion on a neural network is illustrated in Fig. 4. The early fusion approach combines multiple pieces of information before extracting a feature using a network model. Late fusion combines two pieces of information after extracting



Method	Rank 1	Rank 5	Rank10	Rank15	Rank20
Open Four Stream	74.1	85.6	93.4	99.1	99.5
ST-Pyramid Network	73.2	80.7	94.5	97.6	100
Open FstCN	74.1	80.4	97.3	100	100
Open Two-Stream	78.1	95.2	99.3	99.7	100
STRM with DenseNet-40	91.2	96.3	99.5	100	100

(a)

Method	Rank 1	Rank 5	Rank10	Rank15	Rank20
Open Four Stream	54.6	61.3	78.5	81.3	84.5
ST-Pyramid Network	51.7	67.2	75.4	80.9	88.4
Open FstCN	54.6	70.3	74.2	88.2	92.5
Open Two-Stream	58.6	71.5	80.4	85.3	93.2
STRM with DenseNet-40	72.3	85.4	96.4	98.3	99.1

(b)

FIGURE 6. Graphs showing the cumulative match characteristic (CMC) curves for the UCF-101 dataset and HMDB51 dataset. (a) and (b) are graphs of the CMC curves, and the corresponding accuracy table for UCF101 and HMDB51, respectively.

the abstracted features using networks. For every K value, the late-fusion approach showed slightly better accuracy than the early-fusion case. This shows that it is better to fuse two input types after convoluting them separately than to directly concatenate the two heterogeneous raw inputs.

Comprehensively, the entire experimental results can be interpreted as follows. The action recognition performance was usually proportional to the value of K. However, the performance improvement dependent on K did not increase linearly. In addition, since the computational cost also increased rapidly using our method, it should be carefully considered when deciding on the K value. The overall action recognition accuracies showed that late fusion can provide more precise action recognition performance than the early fusion approach.

C. COMPARISON WITH STATE-OF-THE-ART METHODS

We demonstrated the effectiveness of STRM for action recognition under the open-set condition. STRM was compared with both closed-set methods and open-set methods simultaneously. The UCF101 dataset and HMDB51 dataset provide three splits which can be used to train and evaluate an action recognition method. The test parts of these three splits were used for performance evaluation. Table 3 shows the average performance over three splits for the proposed method and other existing methods under closed-set and open-set conditions on the two benchmark datasets.

We compared STRM using closed-set based methods and open-set based methods. The closed-set methods were: iDT [18], Two-stream [31], FstCN [71], MoFAP [72], MIFS [8], LTC [34], R-STAN [73], ST-Pyramid Network [74], ATW [75], DOVF [76], Four-Stream [77], TLE [78], and DTPP [79]. The open-set methods were: ODN [43], P-ODN [44], SDMM [48], and Mishra *et al.* [47].

Since there is a shortage of studies on open-set action recognition, several existing methods were modified to evaluate their action recognition performance under an open-set condition. Two-stream [31], ST-Pyramid Network [74], Four-Stream [77], and FstCN [71] were selected as competitive methods for STRM. For fair experiments, the selected methods were trained with the same hyperparameter setting used for training STRM, as mentioned in section IV-A. After the classification layer in these methods was removed, the action gallery was constructed, and a performance evaluation was carried out using the features extracted from the previous layer of the removed classification layers. The results were sorted in ascending order of accuracy for UCF101. We set the K value at 128 since the accuracy difference between $K = 128$ and $K = 256$ was negligible, and less computation was more cost effective. All accuracies of the open-set action recognition methods were defined as rank 1 accuracy. Table 3 shows the performances of several existing methods and their input types.

In the experimental results on the UCF101 dataset, the best accuracy among the closed-set action recognition methods was achieved by DTPP, which realized 96.2% action recognition accuracy. On the other hand, the highest accuracy of 91.2% was achieved by STRM using the DenseNet-40 among the open-set action recognition methods. STRM with VGG-19 and ResNet-32 showed 87.4% and 89.2% recognition accuracies, respectively. Additionally, among the open-set methods, SDMM [48] achieved a recognition accuracy of 86.63% making it less comparable to other methods. There was a 5.0% gap between the best performances of the closed-set and open-set action recognition methods. However, considering the challenging issue of having to recognize unseen actions in the training step under the open-set condition, this difference in action recognition accuracy is relatively reasonable. Nevertheless, the experimental results show that STRM can provide comparable action recognition performance, even though it did not achieve the best recognition accuracy in our experiments.

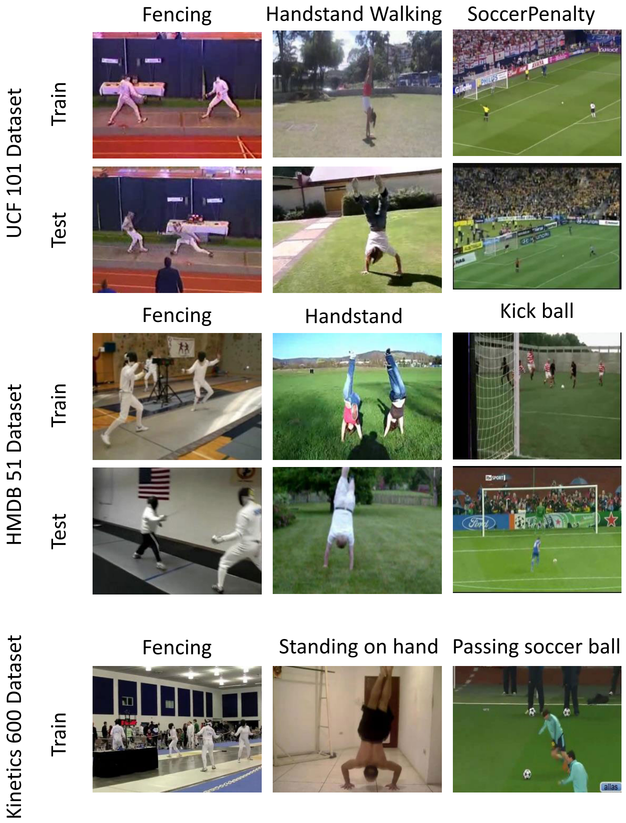


FIGURE 7. Example snapshots for (a) UCF-101 dataset, (b) HMDB51 dataset, and (c) Kinetics dataset. STRM is trained with just training-set in Kinetics and tested on UCF and HMDB. As shown in the examples, video samples share common feature such as background color or viewpoint if they belong to the same dataset. The difference of common features between datasets can affect the recognition performance.

The experimental results on the HMDB51 dataset were similar to the experimental results on the UCF101 dataset with DTPP achieving the best recognition accuracy of 76.3% among the closed-set methods. The second highest accuracy of 72.5% was achieved by Four-Stream. Among the open-set action recognition methods, the best performance was achieved by STRM with the DenseNet-40 model, which

TABLE 3. Action recognition accuracies (%) on the HMDB51 and UCF101 datasets. 'Features' denotes the type of input data for each model and includes RGB (image), optical flow (OF), and improved Dense trajectories (iDT) [18]. 'Protocol' represents the condition, either closed-set condition or open-set condition. * indicates we have implemented the model. The boldface figures represent the best performance in each section.

Method	Protocol	Features	UCF101	HMDB51
iDT [18]	Closed	RGB	85.9	57.2
Two-stream [31]	Closed	RBG+OF	88.0	59.4
FstCN [71]	Closed	RGB	88.1	59.1
MoFAP [72]	Closed	iDT	88.3	61.7
MIFS [8]	Closed	iDT	89.1	65.1
LTC [34]	Closed	RGB+OF	91.7	64.8
R-STAN [73]	Closed	RGB	92.7	64.4
ST-Pyramid Network [74]	Closed	RGB+OF	94.6	68.9
ATW [75]	Closed	RGB+OF	94.6	70.5
DOVF [76]	Closed	RGB+OF	94.9	71.7
Four-Stream [77]	Closed	RGB+OF	95.5	72.5
TLE [78]	Closed	RGB+OF	95.6	71.1
DTPP [79]	Closed	RGB+iDT	**96.2**	**76.3**
ODN [43]	Open	RGB	76.07	46.01
P-ODN [44]	Open	RGB	76.2	**67.36**
SDMM [48]	Open	RGB	**86.63**	58.30
Mishra *et al.* [47]	Open	RGB	76.68	52.58
Open Four-Stream*	Open	RGB+OF	74.1	54.6
Open ST-Pyramid Network*	Open	RGB+OF	73.2	51.7
Open FstCN*	Open	RGB	74.1	54.6
Open Two-Stream*	Open	RGB+OF	78.1	58.6
STRM$_{VGG-19}$ $(K = 128)$	Open	RGB+OF	87.4	63.7
STRM$_{ResNet-34}$ $(K = 128)$	Open	RGB+OF	89.2	65.8
STRM$_{DenseNet-40}$ $(K = 128)$	Open	RGB+OF	**91.2**	**72.3**

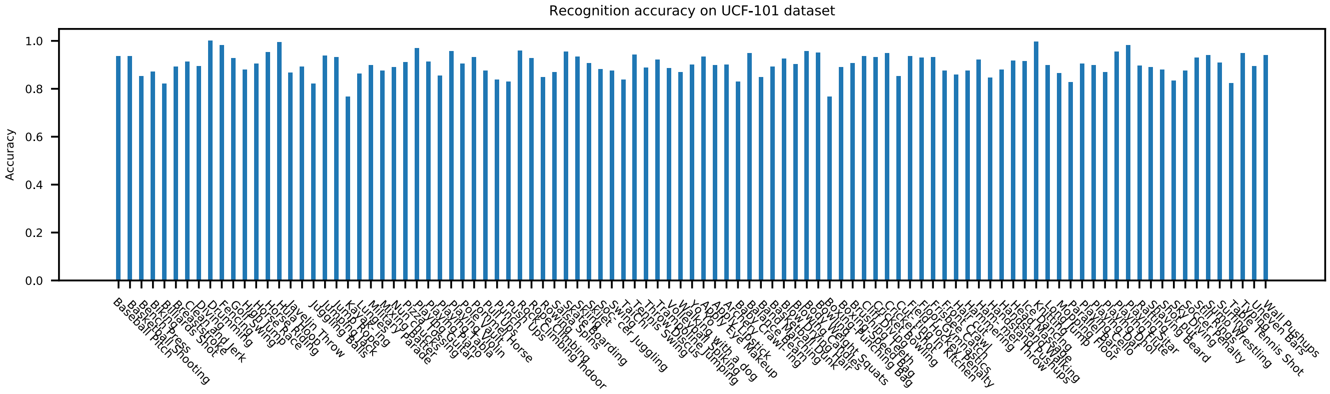


FIGURE 8. Action recognition accuracies for the action classes in the UCF101 dataset.

had 72.3% accuracy on the HMDB51 dataset. STRM with VGG-19 and ResNet-34 achieved accuracies of 63.7% and 65.8%, respectively. Besides STRM, the highest performance among the other open-set methods was 67.36%, achieved by P-ODN [44].

Similar to the experimental results on the UCF101 dataset, there was a 4.0% gap between DTPP and STRM with the DenseNet-40. However, the overall experimental results showed that STRM can provide reasonable and comparable recognition performance compared with the other approaches, in spite of having to perform under the disadvantage of open-set condition. The main reasons open-set achieve relatively lower performances than closed-set methods have to do with the characteristics of the dataset and the training approach. A benchmark dataset is usually divided into two separate subsets, the training-set and testing-set. These two subsets share common features (such as similar background color, similar viewpoints) since they belong to the same dataset. On the other hand, in our experiments, we trained STRM with just the Kineitcs dataset then tested

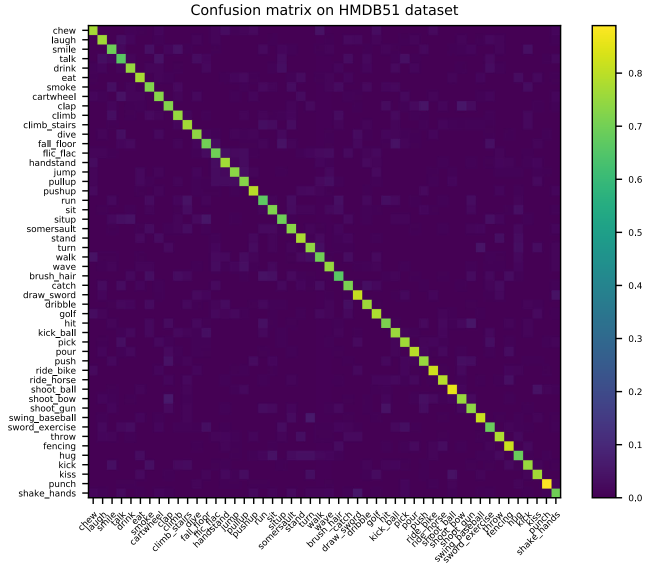


FIGURE 9. **The confusion matrix on HMDB51 dataset.**

it on UCF101 and HMDB51. Moreover, closed-set methods are taught to recognize only learned action classes in the training step. These factors reduce the performances of open-set methods. Fig. 7 shows the snapshot examples of the UCF101 dataset, HMDB51 dataset, and Kinetics-700 dataset. The differences in the datasets used for model training and testing can affect the recognition performance.

Additionally, it is interesting to note that a combination of two input data types usually produced higher accuracy than a single input data type. DOVF and TLE, whose inputs are the combination of raw image (RGB) and optical flow (OF) achieved 94.6% and 95.6% accuracy, respectively, while R-STAN, MIFS and FstCN, which have a single input data type, achieved 92.7%, 89.1% and 88.1% accuracy, respectively. Such results raise the possibility that performance could be improved by using STRM with iDT as the input data.

In addition to the quantitative evaluation, we also compared the discriminative power of the learned representations of the open-set action recognition methods, using a cumulative match characteristic (CMC) curve. Fig. 6a and Fig. 6b show the CMC curve of each open-set action recognition method for the UCF-101 dataset and HMDB51 dataset, respectively. Table 6a and Table 6b display recognition accuracies based on rank. The experimental results shown in Fig. 6 demonstrate that STRM with DenseNet-40 provided more discriminative power than the other methods, achieving an accuracy of 91.2% on the UCF-101 dataset and 72.3% on the HMDB51 dataset. These figures are the best performances among the open-set action recognition methods.

The performances for each action class in UCF101 are plotted in Fig. 8. The results show that the actions 'Band marching' and 'Handstand push-ups' are easy to recognize, while 'Parallel bars' and 'High jump' are hard to recognize. The performances for all action classes in HMDB51 are plotted in Fig. 9. The results show that actions 'push' and 'climb-stairs' are easy to recognize, while 'laugh' and 'kiss' are hard to recognize in HMDB51.

V. CONCLUSION

In this paper, we proposed the STRM method for open-set action recognition. STRM extracts joint ST-representations from motion and appearance data and computes the similarity between the joint ST-representation and the representations listed in the action gallery. The action class which has maximum similarity is assigned as the action class for the given data. This approach enables STRM to process actions under the open-set condition, by recognizing non-trained action classes without having to retrain itself. The experimental results demonstrated the effectiveness of the proposed method for open-set action recognition.

Despite the outstanding performance shown in the experimental results, STRM has several drawbacks in learning and recognizing tasks. The main drawbacks of the proposed method can be summarized as follows. First, to extract good joint ST-representations a large-scale and well-classified dataset is essential for training the model. This problem is an inherent issue for most existing visual recognition methods based on deep neural networks. Second, the computational complexity involved in recognizing action classes is higher than in methods based on the classification approach, and could increase exponentially as the scale of the action gallery increases.

Future works need to consider the above drawbacks and concentrate on developing methods for data generation or data augmentation based on a generative model, to improve action recognition performance, even when a large-scale and well- classified dataset cannot be used. Additionally, a knowledge distillation approach which can reduce the computational complexity of the proposed model by compressing the network scale needs to be explored.

REFERENCES

[1] L. Scime and J. Beuth, "Anomaly detection and classification in a laser powder bed additive manufacturing process using a trained computer vision algorithm," *Additive Manuf.*, vol. 19, pp. 114–126, Jan. 2018.

[2] W. Sultani, C. Chen, and M. Shah, "Real-world anomaly detection in surveillance videos," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 6479–6488.

[3] Y. Bao, Z. Tang, H. Li, and Y. Zhang, "Computer vision and deep learning–based data anomaly detection method for structural health monitoring," *Struct. Health Monit.*, vol. 18, no. 2, pp. 401–421, 2019.

[4] Y. Zhang, Y. Huang, L. Wang, and S. Yu, "A comprehensive study on gait biometrics using a joint CNN-based method," *Pattern Recognit.*, vol. 93, pp. 228–236, Sep. 2019.

[5] Q. Zhang, D. Zhou, and X. Zeng, "HeartID: A multiresolution convolutional neural network for ecg-based biometric human identification in smart health applications," *IEEE Access*, vol. 5, pp. 11805–11816, 2017.

[6] J. Zhu, S. Liao, D. Yi, Z. Lei, and S. Z. Li, "Multi-label CNN based pedestrian attribute learning for soft biometrics," in *Proc. Int. Conf. Biometrics (ICB)*, May 2015, pp. 535–540.

[7] H. Wang, M. M. Ullah, A. Klaser, I. Laptev, and C. Schmid, "Evaluation of local spatio-temporal features for action recognition," in *Proc. Brit. Mach. Vis. Conf. (BMVC)*, 2009, pp. 1–124.

[8] I. Laptev, "On space-time interest points," *Int. J. Comput. Vis.*, vol. 64, nos. 2–3, pp. 107–123, 2005.

[9] I. Laptev, M. Marszałek, C. Schmid, and B. Rozenfeld, "Learning realistic human actions from movies," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2008, pp. 1–8.

[10] P. Dollár, V. Rabaud, G. Cottrell, and S. Belongie, "Behavior recognition via sparse spatio-temporal features," in *Proc. IEEE Int. Workshop Visual Surveill. Perform. Eval. Tracking Surveill.*, Beijing, China, Oct. 2005, doi: 10.1109/VSPETS.2005.1570899.

[11] X. Peng, C. Zou, Y. Qiao, and Q. Peng, "Action recognition with stacked Fisher vectors," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2014, pp. 581–595.

[12] J. Sánchez, F. Perronnin, T. Mensink, and J. Verbeek, "Image classification with the Fisher vector: Theory and practice," *Int. J. Comput. Vis.*, vol. 105, no. 3, pp. 222–245, 2013.

[13] H. Jégou, F. Perronnin, M. Douze, J. Sánchez, P. Pérez, and C. Schmid, "Aggregating local image descriptors into compact codes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 9, pp. 1704–1716, Sep. 2012.

[14] Z. Cai, L. Wang, X. Peng, and Y. Qiao, "Multi-view super vector for action recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 596–603.

[15] G. Willems, T. Tuytelaars, and L. Van Gool, "An efficient dense and scale-invariant spatio-temporal interest point detector," in *Proc. Eur. Conf. Comput. Vis.* Berlin, Germany: Springer, 2008, pp. 650–663.

[16] H. Wang, A. Kläser, C. Schmid, and C.-L. Lin, "Action recognition by dense trajectories," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2011, pp. 3169–3176.

[17] C. Tomasi, "Tracking of point features," Carnegie Mellon Univ., Pittsburgh, PA, USA, Tech. Rep. CMU-CS-91-132, 1991.

[18] H. Wang and C. Schmid, "Action recognition with improved trajectories," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 3551–3558.

[19] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 2961–2969.

[20] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, "Path aggregation network for instance segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8759–8768.

[21] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 779–788.

[22] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 2117–2125.

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

[24] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 4700–4708.

[25] G. Rogez, P. Weinzaepfel, and C. Schmid, "LCR-Net: Localization-classification-regression for human pose," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 3433–3441.

[26] G. Rogez, P. Weinzaepfel, and C. Schmid, "LCR-Net++: Multi-person 2D and 3D pose detection in natural images," *IEEE Trans. Pattern Anal. Mach. Intell.*, to be published.

[27] R. Gu, G. Wang, and J.-N. Hwang, "Efficient multi-person hierarchical 3D pose estimation for autonomous driving," in *Proc. IEEE Conf. Multimedia Inf. Process. Retr. (MIPR)*, Mar. 2019, pp. 163–168.

[28] X. Zhou, Q. Huang, X. Sun, X. Xue, and Y. Wei, "Towards 3D human pose estimation in the wild: A weakly-supervised approach," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 398–407.

[29] Y. Zhou, X. Sun, Z.-J. Zha, and W. Zeng, "MiCT: Mixed 3D/2D convolutional tube for human action recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2018, pp. 449–458.

[30] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 1725–1732.

[31] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 568–576.

[32] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3D convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2015, pp. 4489–4497.

[33] Y. Zhu, Z. Lan, S. Newsam, and A. G. Hauptmann, "Hidden two-stream convolutional networks for action recognition," 2017, *arXiv:1704.00389*. [Online]. Available: https://arxiv.org/abs/1704.00389

[34] G. Varol, I. Laptev, and C. Schmid, "Long-term temporal convolutions for action recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 6, pp. 1510–1517, Jun. 2018.

[35] J. Carreira, E. Noland, A. Banki-Horvath, C. Hillier, and A. Zisserman, "A short note about kinetics-600," 2018, *arXiv:1808.01340*. [Online]. Available: https://arxiv.org/abs/1808.01340

[36] I. Lee, D. Kim, S. Kang, and S. Lee, "Ensemble deep learning for Skeleton-based action recognition using temporal sliding LSTM networks," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 1012–1020.

[37] L. Shi, Y. Zhang, J. Cheng, and H. Lu, "Two-stream adaptive graph convolutional networks for skeleton-based action recognition," in *Proc. CVPR*, Jun. 2019, pp. 12026–12035.

[38] A. Liaw and M. Wiener, "Classification and regression by randomforest," *R News*, vol. 2, no. 3, pp. 18–22, 2002.

[39] J. A. K. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural Process. Lett.*, vol. 9, no. 3, pp. 293–300, Jun. 1999.

[40] J. Y.-H. Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici, "Beyond short snippets: Deep networks for video classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 4694–4702.

[41] A. Bendale and T. Boult, "Towards open world recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 1893–1902.

[42] H. Xu, B. Liu, L. Shu, and P. Yu, "Open-world learning and application to product classification," 2018, *arXiv:1809.06004*. [Online]. Available: https://arxiv.org/abs/1809.06004

[43] Y. Shu, Y. Shi, Y. Wang, Y. Zou, Q. Yuan, and Y. Tian, "ODN: Opening the deep network for open-set action recognition," in *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, Jul. 2018, pp. 1–6.

[44] Y. Shu, Y. Shi, Y. Wang, T. Huang, and Y. Tian, "P-ODN: Prototype based open deep network for open set recognition," 2019, *arXiv:1905.01851*. [Online]. Available: https://arxiv.org/abs/1905.01851

[45] E. M. Rudd, L. P. Jain, W. J. Scheirer, and T. E. Boult, "The extreme value machine," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 3, pp. 762–768, Mar. 2018.

[46] W. Yang, Y. Wang, and G. Mori, "Human action recognition from a single clip per action," in *Proc. IEEE 12th Int. Conf. Comput. Vis. Workshops (ICCV)*, Sep./Oct. 2009, pp. 482–489.

[47] A. Mishra, V. K. Verma, M. S. K. Reddy, S. Arulkumar, P. Rai, and A. Mittal, "A generative approach to zero-shot and few-shot action recognition," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Mar. 2018, pp. 372–380.

[48] Z. Xu, R. Hu, J. Chen, C. Chen, J. Jiang, J. Li, and H. Li, "Semisupervised discriminant multimanifold analysis for action recognition," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 10, pp. 2951–2962, Oct. 2019.

[49] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 3111–3119.

[50] K. Mikolajczyk and H. Uemura, "Action recognition with motion-appearance vocabulary forest," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2008, pp. 1–8.

[51] L. Wang, Y. Qiao, and X. Tang, "Action recognition and detection by combining motion and appearance features," *THUMOS Action Recognit. Challenge*, vol. 1, no. 2, p. 2, 2014.

[52] M. Bregonzio, T. Xiang, and S. Gong, "Fusing appearance and distribution information of interest points for action recognition," *Pattern Recognit.*, vol. 45, no. 3, pp. 1220–1234, 2012.

[53] C. Feichtenhofer, A. Pinz, and A. Zisserman, "Convolutional two-stream network fusion for video action recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 1933–1941.

[54] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

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

[56] S. Hong, J. Oh, H. Lee, and B. Han, "Learning transferrable knowledge for semantic segmentation with deep convolutional neural network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 3204–3212.

[57] Y. Wei, J. Feng, X. Liang, M.-M. Cheng, Y. Zhao, and S. Yan, "Object region mining with adversarial erasing: A simple classification to semantic segmentation approach," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 1568–1576.

[58] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 580–587.

[59] J. Yu, K. C. Yow, and M. Jeon, "Joint representation learning of appearance and motion for abnormal event detection," *Mach. Vis. Appl.*, vol. 29, no. 7, pp. 1157–1170, 2018.

[60] J. Yu, S. Park, S. Lee, and M. Jeon, "Representation learning, scene understanding, and feature fusion for drowsiness detection," in *Proc. Asian Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 165–177.

[61] J. Yu, S. Park, S. Lee, and M. Jeon, "Driver drowsiness detection using condition-adaptive representation learning framework," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 11, pp. 4206–4218, Nov. 2019.

[62] G. E. Dahl, T. N. Sainath, and G. E. Hinton, "Improving deep neural networks for LVCSR using rectified linear units and dropout," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, May 2013, pp. 8609–8613.

[63] J. A. Hartigan and M. A. Wong, "Algorithm AS 136: A k-means clustering algorithm," *Appl. Statist.*, vol. 28, no. 1, pp. 100–108, 1979.

[64] J. Liu, A. Shahroudy, D. Xu, and G. Wang, "Spatio-temporal LSTM with trust gates for 3D human action recognition," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 816–833.

[65] J. Liu, G. Wang, P. Hu, L.-Y. Duan, and A. C. Kot, "Global context-aware attention LSTM networks for 3D action recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 3671–3680.

[66] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah, "Signature verification using a 'siamese' time delay neural network," in *Proc. Adv. Neural Inf. Process. Syst.*, 1994, pp. 737–744.

[67] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu, "Dense trajectories and motion boundary descriptors for action recognition," *Int. J. Comput. Vis.*, vol. 103, no. 1, pp. 60–79, 2013.

[68] Y. Wen, K. Zhang, Z. Li, and Y. Qiao, "A discriminative feature learning approach for deep face recognition," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 499–515.

[69] J. Yu, D. Ko, H. Moon, and M. Jeon, "Deep discriminative representation learning for face verification and person re-identification on unconstrained condition," in *Proc. 25th IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2018, pp. 1658–1662.

[70] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*. [Online]. Available: https://arxiv.org/abs/1409.1556

[71] L. Sun, K. Jia, D.-Y. Yeung, and B. E. Shi, "Human action recognition using factorized spatio-temporal convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2015, pp. 4597–4605.

[72] L. Wang, Y. Qiao, and X. Tang, "MoFAP: A multi-level representation for action recognition," *Int. J. Comput. Vis.*, vol. 119, no. 3, pp. 254–271, 2016.

[73] Q. Liu, X. Che, and M. Bie, "R-STAN: Residual spatial-temporal attention network for action recognition," *IEEE Access*, vol. 7, pp. 82246–82255, 2019.

[74] Y. Wang, M. Long, J. Wang, and P. S. Yu, "Spatiotemporal pyramid network for video action recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 1529–1538.

[75] J. Zang, L. Wang, Z. Liu, Q. Zhang, G. Hua, and N. Zheng, "Attention-based temporal weighted convolutional neural network for action recognition," in *Proc. Int. Conf. Artif. Intell. Appl. Innov.* Cham, Switzerland: Springer, 2018, pp. 97–108.

[76] Z. Lan, Y. Zhu, A. G. Hauptmann, and S. Newsam, "Deep local video feature for action recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, Jul. 2017, pp. 1–7.

[77] H. Bilen, B. Fernando, E. Gavves, and A. Vedaldi, "Action recognition with dynamic image networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 12, pp. 2799–2813, Dec. 2018.

[78] A. Diba, V. Sharma, and L. Van Gool, "Deep temporal linear encoding networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 2329–2338.

[79] J. Zhu, Z. Zhu, and W. Zou, "End-to-end video-level representation learning for action recognition," in *Proc. 24th Int. Conf. Pattern Recognit. (ICPR)*, Aug. 2018, pp. 645–650.

[80] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller, "Labeled faces in the wild: A database for studying face recognition in unconstrained environments," Univ. Massachusetts, Amherst, MA, USA, Tech. Rep. 07-49, 2007.

[81] W. Li, R. Zhao, T. Xiao, and X. Wang, "DeepReID: Deep filter pairing neural network for person re-identification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 152–159.

[82] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev, M. Suleyman, and A. Zisserman, "The kinetics human action video dataset," 2017, *arXiv:1705.06950*. [Online]. Available: https://arxiv.org/abs/1705.06950

[83] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre, "HMDB: A large video database for human motion recognition," in *Proc. Int. Conf. Comput. Vis.*, Nov. 2011, pp. 2556–2563.

[84] H. Jhuang, T. Serre, L. Wolf, and T. Poggio, "A biologically inspired system for action recognition," in *Proc. IEEE 11th Int. Conf. Comput. Vis.*, 2007, pp. 1–8.

[85] K. Soomro, A. R. Zamir, and M. Shah, "UCF101: A dataset of 101 human actions classes from videos in the wild," 2012, *arXiv:1212.0402*. [Online]. Available: https://arxiv.org/abs/1212.0402

YONGSANG YOON received the B.S. degree in computer science from Chonnam National University, Gwangju, South Korea, in 2015. He is currently pursuing the Ph.D. degree with the School of Electrical Engineering and Computer Science, Gwangju Institute of Science and Technology, Gwangju. His research interests include artificial intelligence, machine learning, and pattern recognition.

JONGMIN YU received the B.S. degree in computer science from Chungnam National University, Daejeon, South Korea, in 2013. He is currently pursuing the Ph.D. degree with the School of Electrical Engineering and Computer Science, Gwangju Institute of Science and Technology (GIST), Gwangju, South Korea, and the Ph.D. degree with the School of Electrical Engineering, Computing, and Mathematical Sciences, Curtin University, Perth, WA, Australia. His current research interests include artificial intelligence, machine Learning, pattern recognition, and mathematical understanding of his research areas.

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 the master's degree researcher, he was involved in optimal control problems with the University of California at Santa Barbara, Santa Barbara, CA, USA, from 2001 to 2003, and then moved to the National Research Council of Canada, where he was involved in the sparse representation of high-dimensional data and the image processing, until July 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.