IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received June 25, 2021, accepted July 25, 2021, date of publication July 28, 2021, date of current version August 5, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3101175

Predicting Actions in Videos and Action-Based Segmentation Using Deep Learning

FAYAZ A. MEMON[1], UMAIR A. KHAN[1], ASADULLAH SHAIKH[2], ABDULLAH ALGHAMDI[2], PARDEEP KUMAR[1], AND MESFER ALRIZQ[2]

[1]Department of Computer Systems Engineering, Quaid-e-Awam University of Engineering, Science and Technology, Nawabshah 67450, Pakistan
[2]College of Computer Science and Information Systems, Najran University, Najran 61441, Saudi Arabia

Corresponding author: Umair A. Khan (umair.khan@quest.edu.pk)

This work was supported by the Deputy for Research and Innovation, Ministry of Education, Saudi Arabia, through the Institutional Funding Committee at Najran University, Saudi Arabia, under Grant NU/IFC/INT/01/008.

ABSTRACT In this paper, we propose a technique to recognize multiple actions in a video using deep learning. The proposed approach is concerned with interpreting the overall context of a video and transforming it into one or more appropriate actions. In order to cope with multiple actions in a video, our proposed technique first determines the individual segments/shots in a video using intersections of color histograms. The segmented parts are then fed to the action recognition system comprising a combination of a Convolution Neural Network (CNN) and a Long-Short-Term Memory (LSTM) network trained on our action vocabulary. The segments are then labeled according to their predicted actions and a compact set of distinct actions is produced. Using the corpus generated by the shot detection phase, which includes the location of keyframes in shots, and start/end timestamps of a shot, we can also perform video segmentation based on an action query. Hence, the proposed technique can be used for a number of tasks such as content censoring, on-demand scene retrieval, video summarizing, and query-based scene/video retrieval, to name a few. The proposed technique also stands apart from the existing approaches which either do not take into account the motion information for action prediction or do not perform action-based video segmentation. The experimental results presented in this paper show that the proposed technique not only finds the complete set of actions present in the video but can also find all the relevant parts in a video based on an action query.

INDEX TERMS Action recognition, video segmentation, deep learning, transfer learning.

I. INTRODUCTION

Action recognition refers to the act of classifying an action present in a given image or video. It is a multi-classification problem in which the input is either an image or video, and the output is a class label. Recognizing human actions in video streams is one of the most challenging tasks and has received substantial attention over the last decade by a number of researchers. Analyzing human actions from video streams not only involves coping with human motion and appearance, but it also needs to address a description of a human's intention, emotion, and thoughts [1]. Human action recognition is prone to errors and is affected by stress and/or high workload. At the same time, the sheer volume of videos and images generated these days necessitate the need for an automatic action recognition system. Action recognition is useful in many applications including real-time surveillance [2], [3], crowd behavior monitoring [4], [5], biomechanical analysis of actions for sports and medicine [6], [7], anomaly detection [8], [9], automatic video organization/tagging [10], [11], health care [12], [13], elder care [14], [15], child/animal monitoring [16] and video censoring [17].

Traditional machine learning approaches for action recognition based on hand crafted features suffer from varying environmental conditions and/or are dependent on the mathematical models of the image features. At the same time, they are limited to extract discriminative features from raw video frames. These approaches rely on object detection, pose detection, dense trajectories, or structural information. On the other hand, deep learning based approaches are more effective to learn complex data and a high-level representation directly from raw video data. They can discover the intricate patterns in the complex data on the fly and are capable to build distributed representations from complex data.

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

Convolutional Natural Network (CCN) [18] and Recurrent Neural Network (RNN) [19] are two widely used deep learning frameworks for action recognition tasks. CNN networks have been extensively used for object recognition, object tracking, pose estimation, text detection and recognition, visual saliency detection, action recognition, and scene labeling. CNN computes features from each video frame and obtains a video-level prediction by grouping features from all frames. An obvious limitation of this technique is that it falls short of capturing adequate motion information. RNNs including long short-term memory (LSTM) [20] result in a better performance in sequence-based tasks due to their ability of long-term temporal modeling. Hence, a better technique is to utilize LSTM to model the dynamics of frame-level features.

Despite using LSTM networks to capture the motion information, the redundancy of information in successive video frames limits the action recognition performance. Hence, it is necessary to identify only the key frames in a video that suffice to recognize the action. A better approach is to identify individual shots in a video and then finding the key frame of each shot. The key frame detection itself should be robust to various variants of shot/scene changes.

Since the efficacy of deep learning for identifying patterns in complex data, such as videos, is well-established; hence, it comes as the most suitable candidate for action recognition in videos. Nevertheless, an inherent issue in deep learning is to train a deep neural network from scratch which requires ample training time and huge hardware resources. In order to cope with this issue, transfer learning [21] can be used to train a pre-trained network on an entirely different task by transferring its knowledge from a previous learning task. Experiments on transfer learning have shown that the low-level features of a task learned by a deep neural network are transferable to an entirely different task with much less training time [22]. Hence, by freezing the weights of low-level layers of deep neural networks which represent the rudimentary training information, the higher-level layers of the network can be re-trained on a new task. The action recognition technique proposed in this paper first finds the key frames in a video using the intersection of HSV color histograms. The key frames are then supplied to a combination of CNN and LSTM, trained on an action vocabulary, to predict the action(s) in the video.

The salient points of the contribution in this paper are summarized as follows.

1) An action vocabulary is developed and the relevant dataset is constructed from real scenarios.
2) A combination of CNN and LSTM is trained on the action vocabulary.
3) A key frame detection algorithm is proposed to reduce the redundancy in video information and increase the robustness and execution time of action recognition.
4) The proposed algorithm segments a video with respect to the predicted actions. The proposed system is evaluated on a number of test scenarios to ascertain its efficacy.
5) The proposed system is able to predict multiple actions in a video.

The rest of the paper is structured as follows. Section II provides an in-depth review of the existing techniques for action recognition. Section III describes the methodology of our proposed technique. Section IV presents the evaluation results of the proposed technique. Finally, Section V offers conclusions and future directions.

II. RELATED WORK

The existing techniques on action recognition can be classified into two categories: (i) based on hand-crafted image features, and (ii) based on deep learning. The techniques based on hand-crafted image features require a priori model of the image features to be learned. Whereas, deep learning does not require any mathematical model of the features; it rather discovers the underlying features during training. Traditional techniques of action recognition are based on different types of feature descriptors, detectors and feature trajectories. In past years, various types of feature descriptors and detectors for action recognition have been proposed. These include cuboid descriptors [23], Scale Invariant Feature Transform (SIFT) [24] and its extension 3D-SIFT [25], Histogram of Oriented Gradients (HOG) [26], HOG3D [25], Extended Speeded Up Robust Features (SURF) [27], Motion Boundary Histogram (MBH) [28], Local Binary Pattern (LBP) [29], Local Ternary Pattern (LTP) [30], and shape and motion based feature descriptors [31]–[33]. In addition to this, Efros *et al.* [34] introduced a novel motion descriptor to recognize human actions at distance. In this work, action recognition was done only on tracking a bounding box without using any information about the movements. Laptev and Perez [35] used local space time features with an SVM classifier to classify human actions. They evaluated their results with a video database containing 2391 sequences of six human actions performed by 25 people in four different scenarios.

The action recognition using Spatio-temporal words by finding space-time interest points [36], HOG/HOF features [37], bag of words (SIFT, HOG and HOF) features [38] and combination of multiple features (HOG and HOF) [39] achieved significant results on the datasets of KTH, Skating, HOHA, and HOHA1. However, these approaches have several limitations. They fail to distinguish similar actions, are unable to capture temporal information, and are less efficient in recognizing the overall context. In addition, these techniques express actions literally and lack exact timing alignment and correspondence with real actions. Apart from this, these techniques generally suffer from varying environmental conditions, are dependent on the mathematical models of the image features, and are limited to extract discriminative features from raw video frames.

Action recognition using deep learning is gaining increasing attention. These techniques generally use CNNs, single

stream networks, two-stream networks, two streams fusion networks, 3D-CNN networks to process videos, and RNN networks, particularly LSTM networks. Some more recent architectures [40], [41] have focused on using attention mechanisms for detecting salient parts of a video. The main idea behind these approaches is to capture long-term temporal information between video frames to correctly recognize the actions.

The major breakthrough in deep learning was introduced by Krizhevsky *et al.* [42] for the image classification problem. The objective was to employ an efficient method to train a model with 1.2 million high-resolution images to classify new images in 1000 different classes. This approach was based on a deep convolutional neural network. Later on, Karpathy *et al.* [43] explored the idea of fusing temporal information from consecutive video frames using a 2D pre-trained convolution network. The learned Spatio-temporal features did not capture motion features due to the limited diversity of the dataset. Learning detailed features turned out to be tough and the results were significantly inferior as compared to the algorithms based on the hand-crafted features.

Ji *et al.* developed a 3D CNN model that can capture the motion information in adjacent frames [44]. This model extracts both spatial and temporal features by using 3D convolutions. Tran *et al.* [45] used video level 3D convolutions. A 3D convolution-based network was trained on the Sports1M1 dataset for feature extraction. The long-range temporal modeling was still a problem and training such huge networks was computationally expensive. Yang *et al.* [46] proposed asymmetric, one-directional 3D convolutions to approximate the traditional 3D convolution. The asymmetric 3D convolution decreases parameters, significantly reduces the computational cost and outperforms traditional 3D-CNN models with respect to both effectiveness and efficiency.

The performance of a single-stream network was improved by using a two-stream network in order to capture local temporal information. In two-stream networks, separate spatial and temporal recognition streams based on ConvNets were used. In these approaches, Wang *et al.* [47] used Temporal Segment Networks (TSN), Carreira and Zisserman [48] used inflated 3D networks (I3D), Diba *et al.* [49] used Temporal 3D networks (T3D), and Zhu *et al.* [50] proposed Deep networks with Temporal Pyramid Pooling (DTPP) which was an end-to-end video-level representation learning approach. These approaches achieved significant accuracy on various datasets; however, the long-term temporal information was still a major issue in these approaches.

Donahue *et al.* [51] designed a deep recurrent network by cascading a CNN with an RNN. CNN computed frame features were then fed to the LSTM for feature sequence modeling so that the video-level representation could be learned in both temporal and spatial dimensions. Ng *et al.* [52] integrated the architecture of temporal feature pooling with an LSTM to enable the model to work with arbitrary length frames.

Majid *et al.* [53] used C2LSTM to capture motion as well as spatial features and time dependencies. The network was evaluated on two well-known benchmarks: UCF101 and HMDB51. The results confirmed the efficacy of C2LSTM to capture motion as well as spatial features and time dependencies. Li *et al.* [40] developed a spatio-temporal attention (STA) model, Dai *et al.* [41] developed a two-stream attention-based LSTM architecture for action recognition in videos, and Zhang *et al.* [54] proposed a Knowledge Integration Network (KINet) for action recognition. These approaches achieved significant accuracy on the datasets of UCF-1011, HMDB512, Kinetics-4003.

Khan *et al.* [55] proposed a deep learning based algorithm for movie tags prediction and segmentation. In this work, the authors constructed a tag vocabulary and created a dataset for the vocabulary. They further proposed an efficient shot detection algorithm to find keyframes in the movie. However, their technique does not take into account the motion information in the video which is more efficient to consider for action recognition. Sargano *et al.* [56] proposed a method for recognizing human actions using a pre-trained deep CNN architecture and SVM-KNN hybrid classifier. They achieved a higher accuracy on UCF sports and KTH datasets as compared to the hand-crafted feature-based methods. Rafiq *et al.* [57] proposed a technique for sports video scene classification using a pre-trained AlexNet convolutional neural network with emphasis on video summarization. They evaluated their results on cricket videos and compared their technique with various deep-learning models. Elharrouss *et al.* [58] proposed an approach for multiple human action detection, recognition and summarization in videos using Histogram of Oriented Gradients (HOGs) of the frames in each shot. In this approach, the actions are recognized by performing a comparison between the generated HOG and the existing HOGs in the training phase.

From the literature review, it is evident that a plethora of research has been directed towards action recognition; however, some issues still pertain such as background clutter and/or fast irregular motion, occlusion, viewpoint changes, high computational complexity, and sensitivity to illumination changes. CNNs were initially used for action recognition on frame-by-frame basis and found to show superior performance than that of hand-crafted features based techniques. Later, 3D CNNs showed even better performance by processing multiple frames simultaneously. In some recent architectures, CNNs have been combined with RNNs to incorporate motion information. Nevertheless, the existing techniques do not deliver a promising performance. A short summary of approaches for action recognition presented in this literature review is shown in Table 1.

The action recognition technique proposed in this paper considers the limitations of the existing techniques by combining a CNN with an LSTM, and processing only the key frames of a video to eliminate redundancy and decrease computational complexity. Another striking feature of this work is to segment the video based on the predicted actions. This is

TABLE 1. Summary of literature review.

Technique	Pros	Cons
A motion descriptor to recognize human actions [34]	Can be used to classify actions, transfer 2D/3D skeletons, and synthesize novel action sequences.	Action recognition done only by tracking a bounding box without using any motion information
A SVM classifier for human action recognition using local space-time features [35]	High recognition performance as compared to other related approaches	Some categories can be confused with others. Local features performs better than others
Multiple types of feature descriptors, detectors and feature trajectories to recognize actions [36]–[39]	Not class-specific	Fails to distinguish similar actions, unable to capture temporal information, less efficient in recognizing the overall context, lacks exact timing alignment, may suffer from environmental variations.
A 3D convolutional networks as a feature extractor [45]	An organized study to find the best temporal kernel length for 3D ConvNets.	The long-range temporal modeling, computationally expensive
A 3D CNN model [44]	Captures the motion information encoded in adjacent frames	Computation-intensive, higher storage cost, difficult to learn, labeling of data required.
Asymmetric, 3D convolutions to approximate the traditional 3D convolution [46]	Decreases parameters, significantly reduces the computational cost, outperforms traditional 3D-CNN models	Computation-intensive, higher storage cost, difficult to learn, labeling of data required.
A two-stream network to capture local temporal information [47]–[50]	Higher accuracy on various benchmark datasets	Long-term temporal information is the main issue
LSTM networks to capture local temporal information [51], [52]	Higher accuracy on various benchmark datasets	Long-term temporal information is the main issue
C2LSTM to capture motion as well as spatial features and time dependencies [53]	Efficient to capture motion as well as spatial features and time dependencies	Long-term temporal information is the main issue
Attention mechanisms for detecting salient parts of a video [40], [41]	Captures long term temporal information among video frames to correctly recognize the actions.	Generally more complex architectures for action recognition
A Knowledge Integration Network (KINet) [54]	Higher accuracy on the datasets of UCF-1011, HMDB512, Kinetics-4003.	Generally more complex architectures for action recognition

particularly important for a number of tasks such as content censoring, on-demand scene retrieval, video summarizing, and query based scene/video retrieval.

III. METHODOLOGY

Our proposed action recognition system involves two main steps: (i) detection of shot boundaries and their respective key frames, and (ii) applying the action recognition network on the individual key frames. The shot boundaries are detected using the intersection of the color histogram of successive video frames. Afterwards, the middle frame of each shot is selected as its key frame. The key frames are then fed to the action recognition system. The predicted actions belonging to the same class are merged together to create a composite action. Finally, the shots are split based on the predicted actions. Once these steps are applied on the video, as shown in Figure 1, the system also retrieves a particular action based on an action query. In addition, the system also generates a summary of the video with the start/end frames times corresponding to each action. The following sections discuss these steps in detail.

A. DATASET

One of the challenges in deep learning is the availability and completeness of the datasets. Deep learning models can be trained and tested either on standard benchmarks such as UCF-101 [59], UCF11 [60], HMDB-51 [61], YouTube8M [62] and KTH [35] or they can be trained and tested on a newly developed dataset. UCF11 action dataset consists of 11 action classes which contain 1600 videos captured at a frame rate of 29.97 fps. The videos in each class are organized into 25 groups, each containing at least four videos. The videos in each group share some common features such as the same actors, similar backgrounds, and similar viewpoints. This dataset adequately addresses a variety of factors such as illumination conditions, object appearance and pose, large variations in camera motion, object scale, viewpoint, and cluttered background.

KTH is one of the standard datasets which is widely used for comparing the performance of action recognition systems. This dataset consists of six action classes (i.e., jogging, hand waving, boxing, running, walking and hand clapping) in which each class contains videos involving a single actor (people) performing single action recorded in a constrained environment with a static background. Each action in the dataset is performed by 25 different actors in four different scenarios: outdoor (s1), outdoor with scale variation (s2), outdoor with different clothes (s3), and indoor (s4). This dataset contains $25 \times 4 \times 6 = 600$ videos which are captured at a frame rate of 25fps with a resolution of 160×120. Each of these 600 video clips is further dived into four segments.

A newly developed dataset usually contains classes for a particular scenario. In this work, the proposed technique is evaluated on a real scenario in which various actions are performed by students, teachers and others in an educational institute. For this purpose, first, an action vocabulary comprising nine actions is constructed. These action classes with their semantic meanings are shown in Table 2.

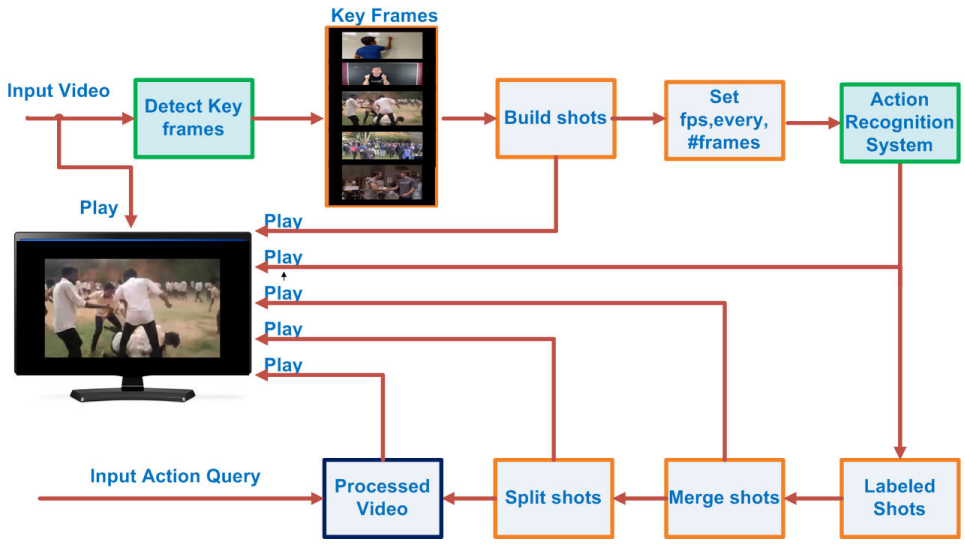


FIGURE 1. Proposed key frame based action recognition system.

TABLE 2. Action vocabulary.

Action	Semantic meaning
Crowd gathering/ Rallying	A large number of people in close proximity, throng, any group or set of people with something common
Fighting Scene	Action of scuffling, violence or conflict
Smoking	Students or the employees of a company smoking cigarettes in prohibited area
Fire Scene	Incident of fire at a school, university, etc.
Explaining	Teacher explaining in classroom
Writing on Board	Teacher writing on board
Gun Fire Scene	Action of gun fire using pistols, guns, rifles, etc.
Corona Temperature Check	People's temperature being checked with a temperature gun
Normal	Normal actions in the video representing none of the above-mentioned scenarios.

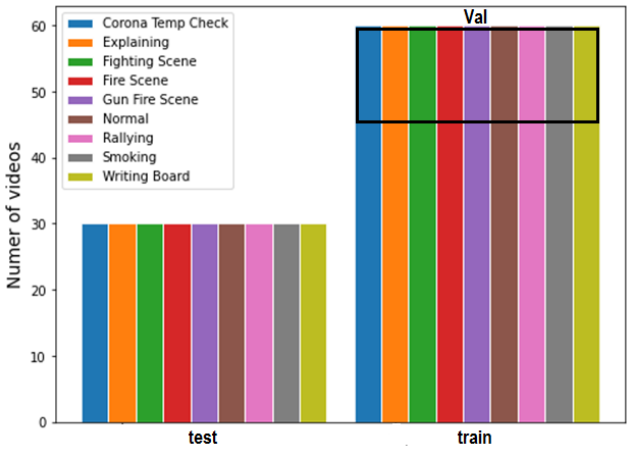


FIGURE 2. Partition of the dataset.

Subsequently, the dataset is collected for each action by obtaining the relevant features (video clips containing the actions of interest) from a number of sources (such as YouTube, Dailymotion, Vimeo, etc.) and our own created videos corresponding to each action. It is pertinent to mention that some of the actions have overlapping features, e.g., fighting scenes, rallying and gunfire scene share common features, which makes this training problem tougher than the one in which classes share little to no features. Apart from that, while collecting the dataset, various issues such as lighting, contrast, orientation, image quality and perspective are also kept in view. Ignoring these variations can drastically reduce prediction accuracy.

Our action vocabulary and the corresponding dataset are the first steps towards automated detection of the incidents/ events commonly occurring in an educational institute or other organizations. These events either entail real-time surveillance and response or require event-based archiving for later reference. The action vocabulary is scalable and subject to evolution as more relevant actions are identified and the appropriate dataset is collected. The dataset contains 810 action videos and each class in the dataset contains 90 videos of varying duration (2-15 sec). The dataset is split into train and test datasets, containing 60 videos (i.e., 67%) and 30 videos (i.e., 33%) pertaining to each class, respectively. Moreover, when training the model, 20% of train dataset (i.e., 12 videos) are used for validation, as shown in Figure 2. For extracting frames from individual videos for training, multiple options are offered based on the combination of three parameters, i.e., fps (frames per second), i and n, where the Boolean parameter fps, when set to true, extracts frames based on the video's frame rate, i represents the frequency of the frames to be extracted, and n represents the total number of frames to be extracted. A description of the combination of these parameters is given in Table 3.

TABLE 3. Criteria for selecting video frames for training and prediction.

Frame extraction parameters			Description
Fps	i	n	
True	0	0	Selecting frames based on fps, e.g., if $fps = true$, select each 25th frame
True	0	>0	Select specified number of frames based on fps, e.g., if $fps = 25$, $n = 10$, select only 10 frames after each 25th frame (for a 25 fps video)
False	>0	0	Select each i^{th} frame in the video
False	>0	>0	Select n frames after each i^{th} frame
False	0	>0	Select first n frames
False	0	0	Select all video frames

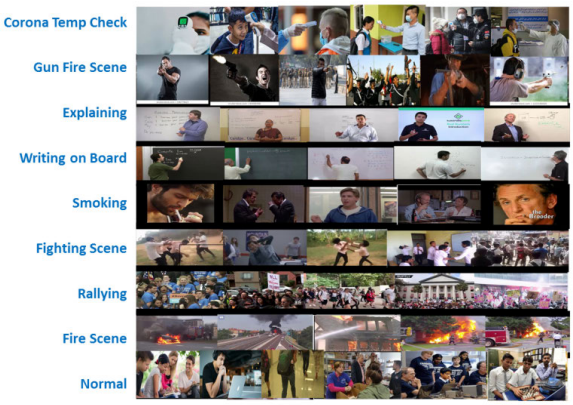


FIGURE 3. Some extracted video frames belonging to different classes of the train dataset.

This is pertinent to mention that the same combination of these parameters is used for action recognition when a given video is applied to the trained network. However, there is no significant influence of different parameter settings when classifying an action video, as shown in Figure 1 and Figure 5. These are frame extraction parameters that describe which frames and how many frames are to be considered for classifying a video. The rationale for offering these combinations for both training and action prediction is two-fold: i) to allow the user to analyze the impact of different combinations on training/testing accuracy for a given dataset, and ii) to analyze the impact of these parameters on the trade-off of training/processing time and the performance in terms of accuracy, precision and recall. Some of the labeled video frames from the training dataset are shown in Figure 3 for illustration.

B. SHOT BOUNDARY DETECTION

Our proposed algorithm for shot boundary detection is shown in Figure 4. This system first extracts key frames either based on a threshold (choice 1) or extracts the first n key frames based on an abrupt change in the visual contents (choice 2). If choice 1 is selected, the system computes the threshold on the basis of the mean and standard deviation of all normalized HSV histogram distances. This is in contrast to the technique in [63] where the threshold is computed on the basis of mean and standard deviation of the absolute differences in the grey level histogram between consecutive frames. The frame extraction rule used in choice 1 to calculate the threshold T is mentioned as follows,

$$T = f \times [\mu_d + \sigma_d], \quad where\, f = 1, 2, 3, \ldots \tag{1}$$

where f is a factor to obtain the desired number of shot boundaries and key frames, μ_d is the mean of all distances, and σ_d is the standard deviation of distances.

The value of factor f may vary from video to video depending on the type of the video. For a particular value of f, if a large number of key frames has been extracted, then the value of f is decreased, and vice versa. Moreover, for a video that contains sharp or sudden transitions, a lower value of f is used. Likewise, for a video that contains gradual or continuous transitions, a higher value of f is used.

To detect shot boundaries, a threshold is applied to each normalized HSV histogram distance. If the distance is greater than the threshold T, the frame is considered to be a key frame and is added into the key frame list.

If choice 2 is selected, the shot detection algorithm extracts the first n key frames based on the abrupt changes in the visual contents. The algorithm extracts key frames by finding those frames where consecutive frames have higher normalized HSV histogram distances. These n key frames can be determined by first sorting normalized HSV histogram distances in descending order. Subsequently, first n normalized HSV histogram distances are obtained. Finally, these distances are sorted in ascending order based on key frame IDs. Choice 2 for obtaining key frames is particularly useful when the shots are classified by sending a smaller number of key frames applied to the action recognition system. It removes redundancy and increases the robustness and execution time of the action recognition system. Furthermore, the number of key frames can be reduced on certain criteria such as: (i) if only one key frame per second is considered, and (ii) if more than one key frames per second are considered. In the latter case, only one key frame will be picked and others will be removed from the key frame list. Finally, middle key frame IDs are obtained. Using these key frame IDs, shots are built by calculating start and end times. The system also generates a summary of the video in terms of a number of shots present in the video with their start/end IDs and timestamps. The steps for extracting key frames and segmenting a video into shots are shown in Figure 4.

1) KEY FRAME EXTRACTION

Our proposed key frame extraction method is based on HSV color histogram. First, a normalized HSV histogram of each video frame is calculated and then the histogram distances of successive video frames are computed. The key frame extraction algorithm is described in Algorithm-I.

The traditional performance metrics including Precision (P), Recall (R) and F1-Score (F) are used to evaluate the efficacy of the proposed algorithm as follows.

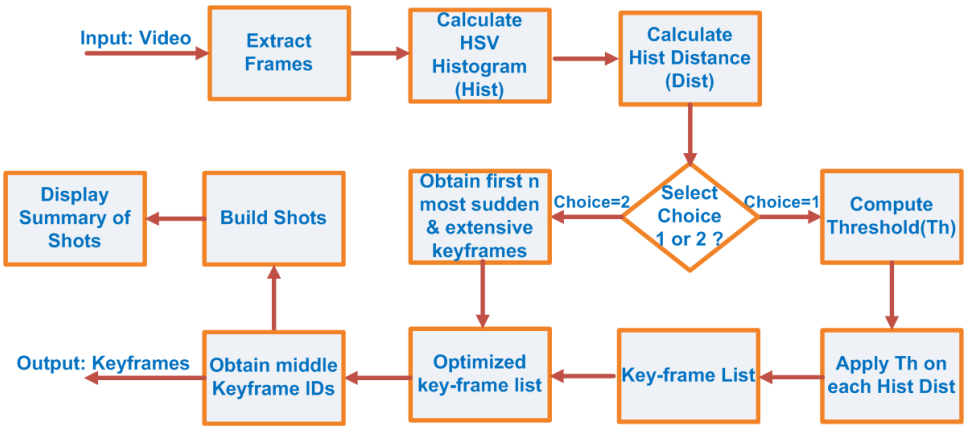


FIGURE 4. Shot detection system.

Algorithm 1 Key Frame Extraction

Calculate normalized HSV histogram of each frame *hist(i)*
for $i = 0$ to $N - 1$ **do**
  $dist(i) = abs(hist(i+1) - hist(i))$
  $total_{dist} = total_{dist} + dist(i)$
end for
if *choice* $= 1$ **then**
  $T = f \times [\mu(total_{dist}) + \sqrt{Var(total_{dist})}]$ %Threshold
  for $i = 0$ to $N - 1$ **do**
    %apply threshold on each distance
    if $dist(i) > th$ **then**
      $kFrame[j] = frame(i)$
    end if
  end for
else
  Sort *dist* w.r.t distances in ascending order
  Obtain first *n* distances from *dist* in *dist_n*
  Sort *dist_n* w.r.t frame IDs in ascending order
  Obtain key frame IDs from *dist_n*
end if
Optimize key frame list

$$P = \frac{\#correct\ key\ frames}{(\#correct\ key\ frames + \#false\ key\ frames)} \quad (2)$$

$$R = \frac{\#correct\ key\ frames}{(\#correct\ key\ frames + \#missed\ key\ frames)} \quad (3)$$

$$F = \frac{2PR}{P+R} \quad (4)$$

Compression Ratio (CR) is used to evaluate the compactness of the key frame sequence. It is the ratio of number of extracted key frames to the total number of frames i.e.,

$$CR = \frac{\#Extracted\ key\ frames}{Total\ number\ of\ frames} \quad (5)$$

Similarly, fidelity can be used to measure the correlation degrees between frames in a shot and the extracted key frames. Fidelity is a Semi-Hausdoff distance between the key frame set and the shot frame set where Semi-Hausdoff distance is the maximum of minimum distances between the set of key frames of a shot and its corresponding frames. If we have a set of shot $S = \{s_1, s_2, \ldots, s_n\}$ and a set of key frames $K = \{k_1, k_2, \ldots, k_n\}$, then Semi-Hausdoff distance d_{sh} is computed as follows,

$$d_{sh} = \max(d_i), \quad 1 \le i \le N \quad (6)$$

where d_i is the minimum distance between the set of key frames of a shot and its corresponding frames which is computed as follows,

$$d_i = \min(d(k_j, s_i)), \quad 1 \le i \le N, \ 1 \le j \le K \quad (7)$$

C. ACTION RECOGNITION

A CNN-RNN deep learning architecture is used to predict actions in the proposed technique. This architecture involves two separate CNN and RNN networks. In the first step, a Conv-2D model is trained on the dataset to extract spatial features. In the second step, an RNN model is trained on these spatial features with one or more hidden layers of LSTM cells to classify action sequences. This action recognition system is shown in Figure 5. For an input video, the system first generates frames on the basis of whether we need frames after every second or every frame, and the number of frames to be extracted. Once these frames are extracted, they are fed to the CNN network. The features obtained from CNN are added to the feature list. When these features are equal to the required number of frames, they are combined to create a sequence. This sequence of CNN features is fed to an RNN network for final prediction.

1) TRAINING DEEP NETWORKS

The following sections describe the steps of training our deep learning models for action recognition. In order to build and train a CNN, four different types of CNN networks are

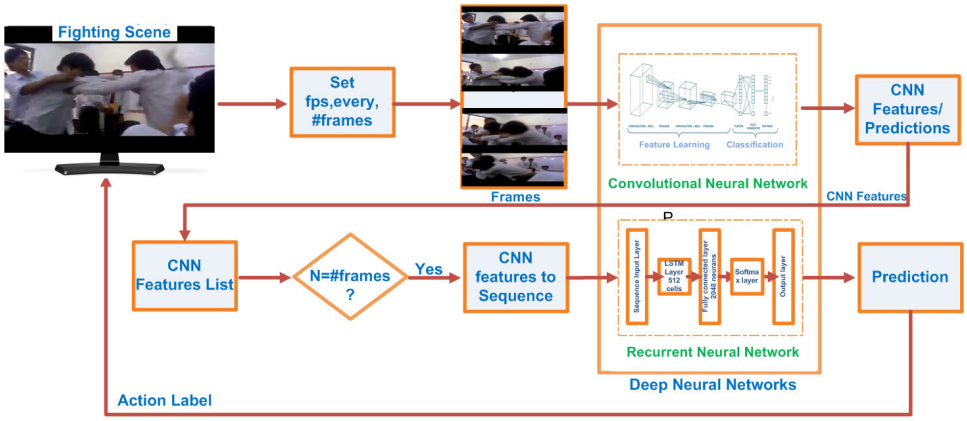


FIGURE 5. Action recognition system.

trained, i.e., Inception-V3 [64], Xception [65], Resnet50 [66] and VGG19 [67] on our action vocabulary. These pre-trained networks have been trained on 1.2 million high resolution images to classify new images in 1000 different classes. Training a new deep learning model from scratch requires a huge amount of data, high computational resources in terms of a number of GPUs and memory, and a long duration which, in some cases, may span several weeks. Therefore, instead of creating and training network from scratch, a network that has been already trained on a large dataset is utilized. This concept is called transfer learning. Transfer learning refers to transferring the knowledge gained through a learning tasks to an entirely different task with much less training time and computational power. This is in contrast to parameter loading which results in small performance improvement of a neural network by varying its hyper parameters such as learning rate and method of parameter initialization. Parameter loading is generally used when the training dataset is large and similar to the dataset on which the pre-trained model was trained.

The type of pre-trained network, number of custom layers used, type of optimization algorithm, and other parameters used during training are given in Table 4. The hyper-parameters, as mentioned in Table 4 have been selected experimentally by applying different settings and analyzing their effects on the efficacy of the trained model. The four CNN networks based on four different types of pre-trained networks have been used in this work. The layer-wise structure of the Inception-V3 based CNN network is shown in Figure 6.

The RNN model is trained on the spatial features obtained from the pooling layer of the CNN model. This model contains one or more hidden layers of LSTM cells to classify action sequences. In this work, four types of RNN models are trained: wide, wider, deep and deeper, where wide and wider LSTM networks are single-layered LSTM networks with the capacity of 128 and 512 neurons, respectively. On the other hand, deep and deeper LSTM networks are multi-layered networks with two and three LSTM cells, respectively. The layer wise structure of the wide LSTM RNN network is shown in Figure 7.

TABLE 4. Training parameters.

CNN Parameter	Description
Base CNN model	InceptionV3, pre-trained on 1000 classes of ImageNet
Modifications	Final layer modified and re-trained Addition of a dropout layer as penultimate layer Discarding the output of 20% neurons
No of layers	318
Types of custom layers used	Convolution, batch_normalization, dropout, global_average_pooling2d, fully connected, softmax
Input image size	299x299
Learning algorithm	Stochastic gradient descent (lr=0.01, decay=1e-6, momentum=0.9)
Activation function	Rectified Linear Units (ReLu)
Error estimate	Cross Entropy
Training epochs	50
Steps per epoch	263 (i.e. Batches)
Batch size	32

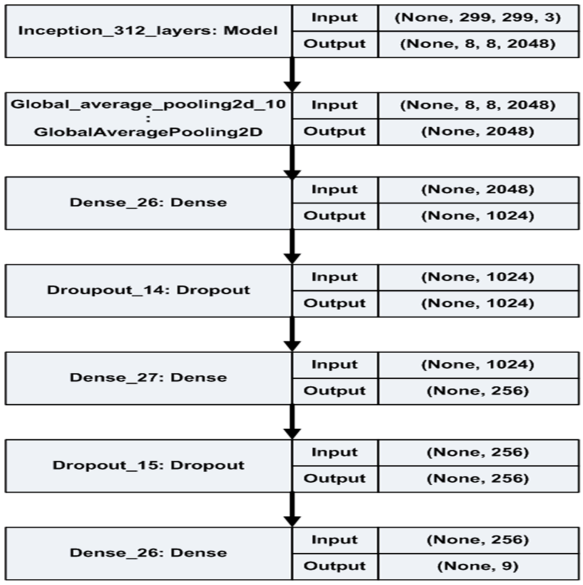


FIGURE 6. Layer-wise structure of Inception-V3 network.

The hardware/software setup used for our experiments is described in Table 5. Using this setup, CNN takes 3.75 hours for training 50 epochs; whereas, RNN takes 20 minutes for

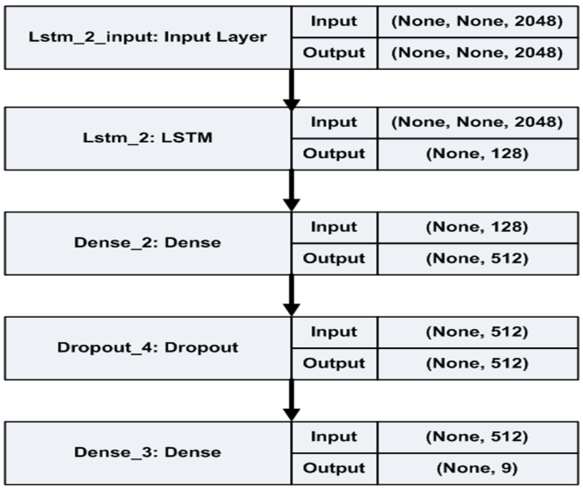


FIGURE 7. Layer-wise structure of wide LSTM RNN network.

TABLE 5. Experimental setup.

Hardware/Software	Specifications
Microprocessor	Intel Core i7
Random Access Memory	8GB
Graphical Processing Unit (GPU)	NVIDIA GTX 750
Deep Learning Framwork	Tensorflow 2.3.0, Keras 2.2.4
Operating System	UBUNTU 16.04
Programming Languages/Libraries	Python 3.5 with OpenCV 3.0, Matplotlib, Pandas

150 epochs. An image size of 299 × 299 and a batch size of 32 are used in the experiments.

IV. EXPERIMENTAL RESULTS

CNN and RNN models are trained according to the data spilt shown in Figure 2. The training-validation accuracy and training-validation loss of CNN and RNN models are shown in Figure 8 and Figure 9. It is evident that these models successfully learn the features of the training data and generalize well. The inception-V3 based CNN model learns faster than other models and achieves a training accuracy of around 100% after 30 epochs, and a maximum validation accuracy equal to 86% after 43 epochs. It can also be seen that this model shows no over-fitting. The RNN models are trained on 150 epochs with a batch size of 32 and a dropout of 0.2. The wide RNN model achieves a maximum validation accuracy of 95%. However, it can be seen that the model starts over-fitting after 140 epochs.

The performance of an action recognition architecture (i.e., CNN-RNN) on the test dataset is described by a confusion matrix as shown in Figure 10. This matrix visualizes the accuracy of a classifier by comparing the actual and predicted classes. For example in our case, for gun fire class, out of 30 samples, 29 samples are correctly classified as gun fire scene, while 1 sample is mis-classified as fighting scene. Also, 1 sample of Smoking class, 1 sample of Corona Temp Check and 2 samples of fighting scene class are classified as gun fire scene. The prediction accuracy of the gun fire class is 97%. The average prediction accuracy of our proposed system is 94%. The precision, recall and F1-Score of each class are shown in Table 6. The average precision, recall and F1-score are around 95% which shows the efficacy of our proposed action recognition system. The precision, recall and F1-Score of each class are shown in Table 6. The average precision, recall and F1-score is around 95% which shows the efficacy of our proposed action recognition system.

TABLE 6. Classification report.

Action	Precision	Recall	F1-Score
Corona Temp Check	1.00	0.85	0.92
Fighting Scene	0.82	1.00	0.90
Writing Board	0.88	0.96	0.92
Gun Fire Scene	0.90	0.90	0.90
Rallying	0.95	1.00	0.98
Fire Scene	1.00	1.00	1.00
Explaining	0.97	0.90	0.93
Smoking	1.00	0.93	0.96
Normal	1.00	1.00	1.00
Average/Mean	0.96	0.95	0.95

The performance of the proposed technique is also evaluated in terms of Receiver Operating Characteristic (ROC) curves, as shown in Figure 11. A ROC curve shows the true positive rates against the false positive rate at various threshold settings. Figure 11 shows that the model has higher value of Area Under Curve (AUC) which represents the degree of separability. All the curves on the ROC graph are clustered at the top-left corner which indicates a high prediction accuracy. In order to ascertain how well the model performs on unseen data and to assess the effectiveness of the model, a 5-fold cross validation technique is used. In this case, the dataset is divided into 5 equal partitions or folds. The model is trained 5 times. Each time, it uses 1 different fold for the testing set (i.e., 20% of a dataset comprising 18 videos in each class) and union of four other folds as the training set (i.e., 80% of a dataset comprising 72 videos in each class), as shown in Figure 12. The accuracy of each fold and the average accuracy is also shown in Figure 12.

The performance of the proposed CNN + RNN architecture is also evaluated using the same experimental setup as given in Table 5 on two benchmark datasets: UCF11 and KTH. No optical flow images (i.e., a single stream of images) are used and very little parameter tuning is performed. UCF11 dataset is split into a train (80%) and test (20%) datasets based on their annotations [60], each of which contains action videos of different groups. The performance on this dataset is evaluated by calculating the average accuracy over all classes. Similarly, the KTH dataset is split into train, validation and test datasets according to people ID [35] which contains action videos performed by 8, 8 and 9 different people, respectively. The performance on this dataset is measured in terms of average accuracy over all action classes.

The accuracy of each class of the UCF11 dataset is shown in Figure 13a, and the accuracy of each class of the KTH dataset is shown in Figure 13b. From Figure 13b, it can

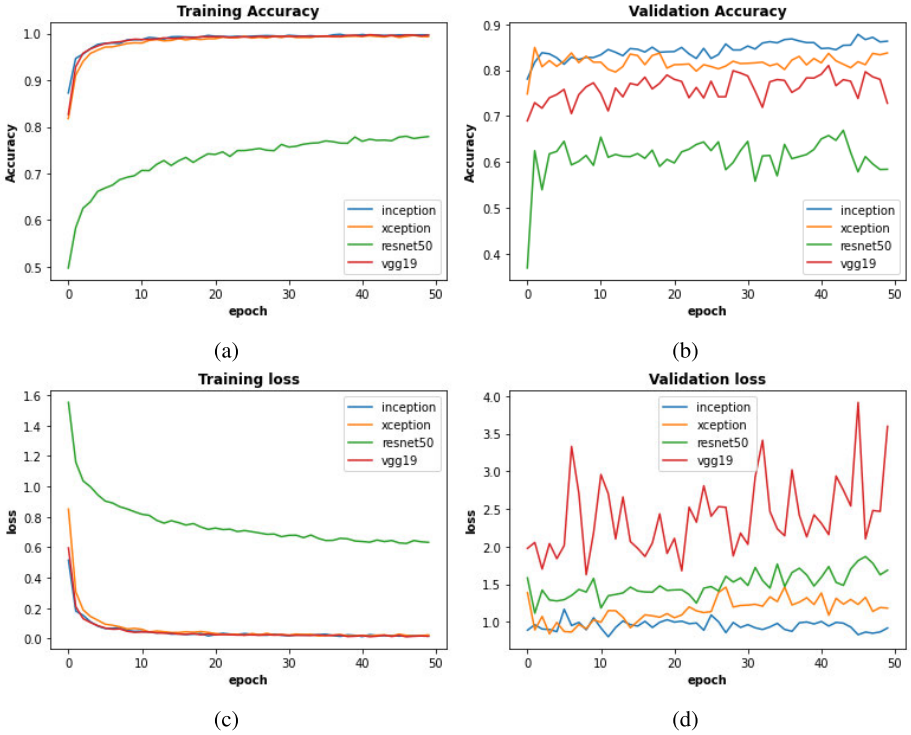


FIGURE 8. Training-validation accuracy and loss of CNN network.

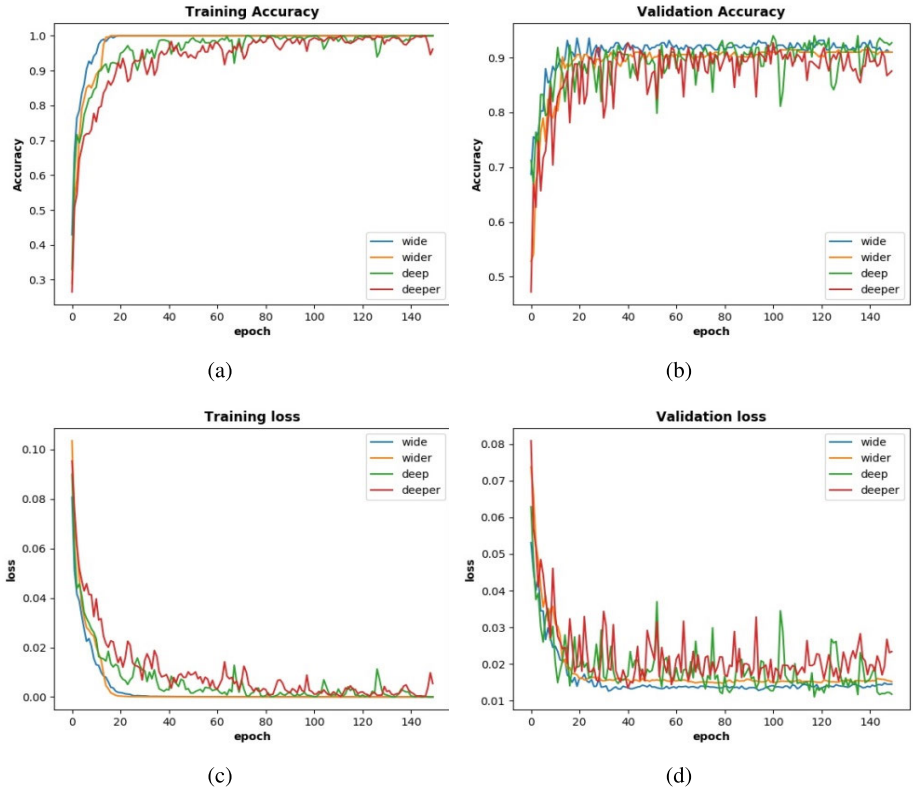


FIGURE 9. Training-validation accuracy and loss of RNN network.

be seen that only two classes, i.e., jogging and running have less than 90% accuracy because these contain similar actions. Therefore, the trained model fails to distinguish the actions of these classes. 17% of actions of jogging class are mis-classified as running class, and 28% actions of running class are mis-classified as jogging class. Average accuracy

TABLE 7. Evaluation results of the key frame extraction.

Test video	Factor	#Frames	# Key frames Extracted	Precision (%)	Recall (%)	F1-Score (%)	Compression Ratio (%)
acrobacia[1]	1	175	11	90.90	83.33	86.96	6.28
kylie[2]	2	204	8	87.5	100	93.33	3.92
Boris Sends a Spike [3]	3	1980	22	95.45	80.77	87.50	1.11
test_scene[4]	2	1062	12	100.00	100.00	100.00	1.13

TABLE 8. Corpus generated by the key frame extraction and action recognition algorithms.

Predicted Action	Start ID	End ID	Start Time	End Time	Shot Duration (sec)	Action Duration (sec)	%age	#Shots
Explaining	000	096	000	3.27	3.27	5.14	14.50%	2
	724	779	24.19	26.06	1.87			
Fighting Scene	97	199	3.27	6.71	3.44	6.77	19.10%	2
	780	879	26.06	29.4	3.34			
Fire Scene	200	273	6.71	9.18	2.47	2.47	06.97%	1
Rallying	274	364	9.18	12.25	3.07	3.07	08.66%	1
Smoking	366	474	12.25	15.88	3.64	3.64	10.27%	1
Writing Board	475	574	15.88	19.22	3.34	3.34	09.42%	1
Corona Temp Check	575	619	19.22	20.72	1.50	1.50	04.23%	1
Gun Fire Scene	620	723	20.72	24.19	3.47	9.51	26.83%	2
	880	1061	29.4	35.44	6.04			
Video length: 35.44 sec								

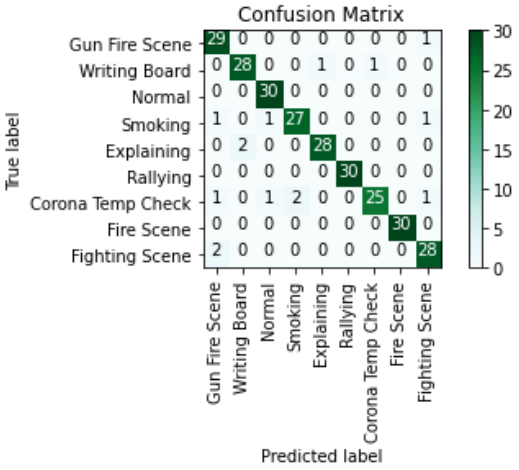


FIGURE 10. Confusion matrix.

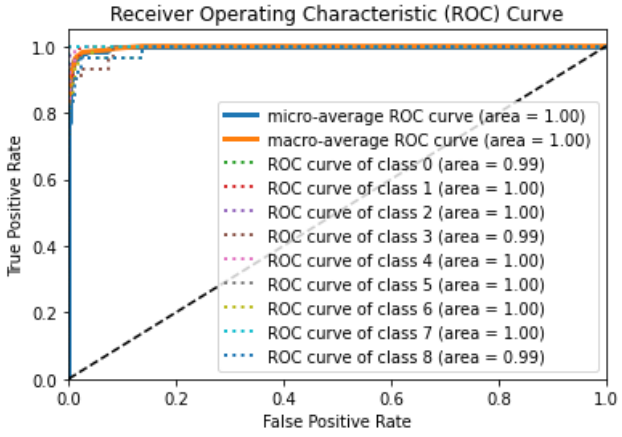


FIGURE 11. Receiver operating characteristics (ROC) curves.

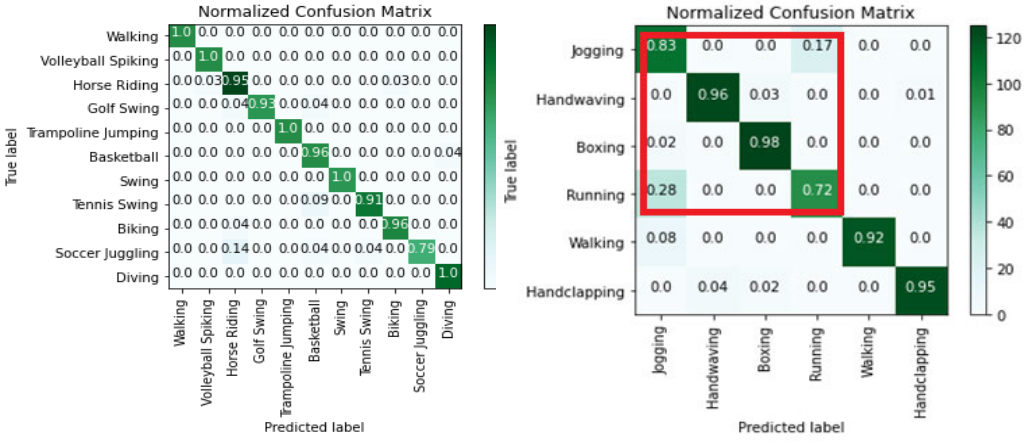
Folds	Dataset			Accuracy (%)
	Train (80%)		Test (20%)	
Fold#1	Test	Train		95.3
Fold#2	Train	Test	Train	93.8
Fold#3	Train	Test	Train	92.2
Fold#4	Train	Test	Train	94.1
Fold#5	Train		Test	95.0
Average Accuracy				94.1

FIGURE 12. 5-Fold cross validation results.

of 95% on the UCF11 dataset and 89% on the KTH dataset is achieved.

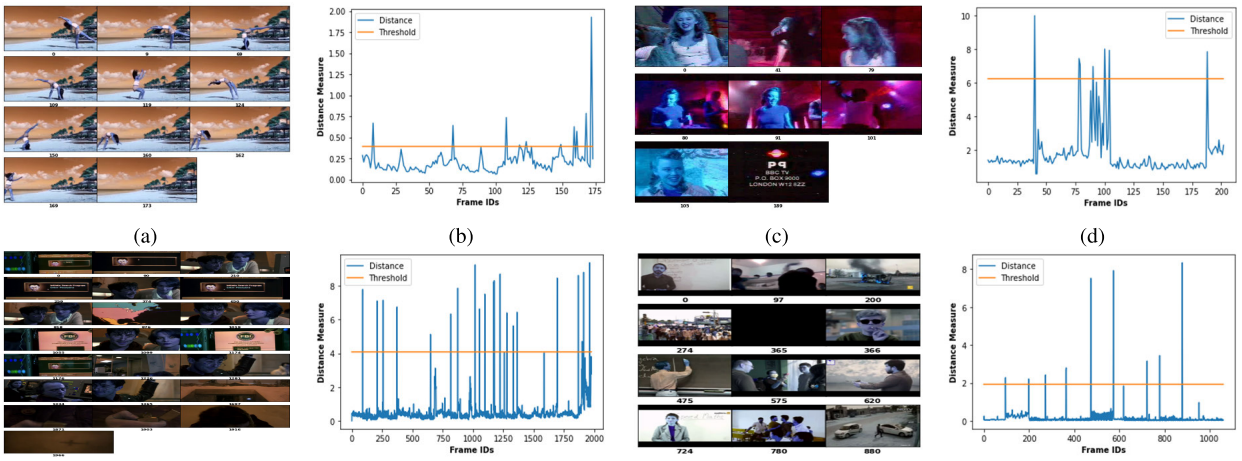
The number of key frames extracted from various test videos along with their histogram distances and threshold values are shown in Figure 14. For example, only 12 key frames were extracted from test video as shown in Figure 14d, which is composed of 1062 frames portioned into 12 shots. The extracted key frames do not have any overlapping or similarity even in a very short-duration video. This shows that our algorithm extracts distinct (non-redundant) information from videos which can be sent to the recognition system.

The performance of the proposed key frame extraction algorithm is evaluated using different evaluation metrics, as shown in Table 7. In the first step, shot boundaries in each video are found by watching all the test videos and noting down the start and end time stamps of each shot. This ground truth is used to compare the shot boundaries with the ones found by our key frame extraction algorithm. Table 7 shows the evaluation results of our key frame extraction algorithm



(a) UCF-11 dataset (b) KTH dataset

FIGURE 13. Confusion matrix.



(a) (b) (c) (d)

(e) (f) (g) (h)

FIGURE 14. Sample key frames (a, c, e, g) and the histogram distances (b, d,f, h). The distances above the threshold represent key frames in b, d, f, h.

(a) Explaining (b) Fighting (c) Fire (d) Rallying

(e) Smoking (f) Writing (g) Temperature check (h) Fighting

FIGURE 15. Action prediction for each extracted shot.

for various test videos of varying lengths. It is evident that our key frame extraction algorithm has a reasonable compression ratio and F1-score.

The action prediction corresponding to each shot is shown in Figure 15. The adjacent shots which belong to the same class are merged into a single composite shot. The system

also generates a summary of the actions present in the video, as shown in Table 8. This table shows the actions performed in a video, their start/end times, length of each shot, and percentage of each action with respect to the video length.

V. CONCLUSION AND FUTURE WORK

In this paper, an action recognition technique to predict actions in videos and segment the videos accordingly has been proposed. A CNN + RNN network is trained on a carefully constructed action vocabulary. Furthermore, a key frame extraction algorithm is also proposed which finds the shot boundaries in a video and suggests the most important frames in a video to be fed to the CNN + RNN network. This not only eliminates redundancy in information processing and improves prediction accuracy, but also alleviates computational complexity. The corpus generated by the key frame detection and recognition system is further used for segmenting the video with respect to the predicted actions. In this way, the proposed system can be used for query based contents retrieval from a videos, contents censorship, efficient archiving, classification, automatic tags generation, and video summarizing.

In the future, we aim to enhance the action vocabulary and the action dataset to incorporate more actions. In the same way, we also aim to use more complex networks to better capture the intricate features of each action. In addition, we are also investigating the potential of audio information on the efficacy of action recognition.

REFERENCES

[1] D. Wu, N. Sharma, and M. Blumenstein, "Recent advances in video-based human action recognition using deep learning: A review," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, May 2017, pp. 2865–2872.

[2] S. Danafar and N. Gheissari, "Action recognition for surveillance applications using optic flow and SVM," in *Proc. Asian Conf. Comput. Vis.* Berlin, Germany: Springer, 2007, pp. 457–466.

[3] Y. Han, P. Zhang, T. Zhuo, W. Huang, and Y. Zhang, "Going deeper with two-stream ConvNets for action recognition in video surveillance," *Pattern Recognit. Lett.*, vol. 107, pp. 83–90, May 2018.

[4] T. Hassner, Y. Itcher, and O. Kliper-Gross, "Violent flows: Real-time detection of violent crowd behavior," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. Workshops*, Jun. 2012, pp. 1–6.

[5] A. B. Mabrouk and E. Zagrouba, "Abnormal behavior recognition for intelligent video surveillance systems: A review," *Expert Syst. Appl.*, vol. 91, pp. 480–491, Jan. 2018.

[6] K. Soomro and A. R. Zamir, "Action recognition in realistic sports videos," in *Computer Vision in Sports*. Cham, Switzerland: Springer, 2014, pp. 181–208.

[7] C. Panagiotakis, I. Grinias, and G. Tziritas, "Automatic human motion analysis and action recognition in athletics videos," in *Proc. 14th Eur. Signal Process. Conf.*, 2006, pp. 1–5.

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

[9] K. Singh, S. Rajora, D. K. Vishwakarma, G. Tripathi, S. Kumar, and G. S. Walia, "Crowd anomaly detection using aggregation of ensembles of fine-tuned ConvNets," *Neurocomputing*, vol. 371, pp. 188–198, Jan. 2020.

[10] S. Ilyas and H. U. Rehman, "A deep learning based approach for precise video tagging," in *Proc. 15th Int. Conf. Emerg. Technol. (ICET)*, Dec. 2019, pp. 1–6.

[11] Z. Wu, T. Yao, Y. Fu, and Y.-G. Jiang, "Deep learning for video classification and captioning," in *Frontiers of Multimedia Research*. San Rafael, CA, USA: Morgan & Claypool, 2017, pp. 3–29.

[12] Y. Gao, X. Xiang, N. Xiong, B. Huang, H. J. Lee, R. Alrifai, X. Jiang, and Z. Fang, "Human action monitoring for healthcare based on deep learning," *IEEE Access*, vol. 6, pp. 52277–52285, 2018.

[13] F. Ali, S. El-Sappagh, S. M. R. Islam, D. Kwak, A. Ali, M. Imran, and K.-S. Kwak, "A smart healthcare monitoring system for heart disease prediction based on ensemble deep learning and feature fusion," *Inf. Fusion*, vol. 63, pp. 208–222, Nov. 2020.

[14] Z. Zhou, E. E. Stone, M. Skubic, J. Keller, and Z. He, "Nighttime in-home action monitoring for eldercare," in *Proc. Annu. Int. Conf. IEEE Eng. Med. Biol. Soc.*, Aug. 2011, pp. 5299–5302.

[15] M. Zerkouk and B. Chikhaoui, "Spatio-temporal abnormal behavior prediction in elderly persons using deep learning models," *Sensors*, vol. 20, no. 8, p. 2359, Apr. 2020.

[16] V. Sturm, D. Efrosinin, N. Efrosinina, L. Roland, M. Iwersen, M. Drillich, and W. Auer, "A chaos theoretic approach to animal activity recognition," *J. Math. Sci.*, vol. 237, no. 5, pp. 730–743, Mar. 2019.

[17] S. Dhanwal, V. Bhaskar, and T. Agarwal, "Automated censoring of cigarettes in videos using deep learning techniques," in *Decision Analytics Applications in Industry*. Singapore: Springer, 2020, pp. 339–348.

[18] 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.

[19] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

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

[21] H.-W. Ng, V. D. Nguyen, V. Vonikakis, and S. Winkler, "Deep learning for emotion recognition on small datasets using transfer learning," in *Proc. ACM Int. Conf. Multimodal Interact.*, Nov. 2015, pp. 443–449.

[22] S. Khan, N. Islam, Z. Jan, I. U. Din, and J. J. C. Rodrigues, "A novel deep learning based framework for the detection and classification of breast cancer using transfer learning," *Pattern Recognit. Lett.*, vol. 125, pp. 1–6, Jul. 2019.

[23] P. Dollár, V. Rabaud, G. Cottrell, and S. Belongie, "Behavior recognition via sparse spatio-temporal features," in *Proc. IEEE Int. Workshop Vis. Surveill. Perform. Eval. Tracking Surveill.*, Oct. 2005, pp. 65–72.

[24] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, 2004.

[25] P. Scovanner, S. Ali, and M. Shah, "A 3-dimensional sift descriptor and its application to action recognition," in *Proc. 15th Int. Conf. Multimedia*, 2007, pp. 357–360.

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

[27] 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.

[28] N. Dalal, B. Triggs, and C. Schmid, "Human detection using oriented histograms of flow and appearance," in *Proc. Eur. Conf. Comput. Vis.* Berlin, Germany: Springer, 2006, pp. 428–441.

[29] T. Ojala, M. Pietikäinen, and T. Mäenpää, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 7, pp. 971–987, Jul. 2002.

[30] X. Tan and B. Triggs, "Enhanced local texture feature sets for face recognition under difficult lighting conditions," *IEEE Trans. Image Process.*, vol. 19, no. 6, pp. 1635–1650, Jun. 2010.

[31] A. Fathi, Y. Li, and J. M. Rehg, "Learning to recognize daily actions using gaze," in *Proc. Eur. Conf. Comput. Vis.* Berlin, Germany: Springer, 2012, pp. 314–327.

[32] M. Jain, H. Jégou, and P. Bouthemy, "Better exploiting motion for better action recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 2555–2562.

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

[34] A. A. Efros, A. C. Berg, G. Mori, and J. Malik, "Recognizing action at a distance," in *Proc. 9th IEEE Int. Conf. Comput. Vis.*, Oct. 2003, p. 726.

[35] C. Schuldt, I. Laptev, and B. Caputo, "Recognizing human actions: A local SVM approach," in *Proc. 17th Int. Conf. Pattern Recognit.*, vol. 3, Aug. 2004, pp. 32–36.

[36] J. C. Niebles, H. Wang, and L. Fei-Fei, "Unsupervised learning of human action categories using spatial-temporal words," *Int. J. Comput. Vis.*, vol. 79, no. 3, pp. 299–318, 2008.

[37] I. Laptev and P. Perez, "Retrieving actions in movies," in *Proc. IEEE 11th Int. Conf. Comput. Vis.*, Oct. 2007, pp. 1–8.

[38] M. Marszalek, I. Laptev, and C. Schmid, "Actions in context," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 2929–2936.

[39] D. Han, L. Bo, and C. Sminchisescu, "Selection and context for action recognition," in *Proc. IEEE 12th Int. Conf. Comput. Vis.*, Sep. 2009, pp. 1933–1940.

[40] J. Li, X. Liu, W. Zhang, M. Zhang, J. Song, and N. Sebe, "Spatio-temporal attention networks for action recognition and detection," *IEEE Trans. Multimedia*, vol. 22, no. 11, pp. 2990–3001, Nov. 2020.

[41] C. Dai, X. Liu, and J. Lai, "Human action recognition using two-stream attention based LSTM networks," *Appl. Soft Comput.*, vol. 86, Jan. 2020, Art. no. 105820.

[42] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 2, pp. 84–90, Jun. 2012.

[43] 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.

[44] S. Ji, W. Xu, M. Yang, and K. Yu, "3D convolutional neural networks for human action recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 1, pp. 221–231, Jan. 2013.

[45] 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. (ICCV)*, Dec. 2015, pp. 4489–4497.

[46] H. Yang, C. Yuan, B. Li, Y. Du, J. Xing, W. Hu, and S. J. Maybank, "Asymmetric 3D convolutional neural networks for action recognition," *Pattern Recognit.*, vol. 85, pp. 1–12, Jan. 2019.

[47] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool, "Temporal segment networks: Towards good practices for deep action recognition," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 20–36.

[48] J. Carreira and A. Zisserman, "Quo vadis, action recognition? A new model and the kinetics dataset," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6299–6308.

[49] A. Diba, M. Fayyaz, V. Sharma, A. H. Karami, M. M. Arzani, R. Yousefzadeh, and L. Van Gool, "Temporal 3D ConvNets: New architecture and transfer learning for video classification," 2017, *arXiv:1711.08200*. [Online]. Available: http://arxiv.org/abs/1711.08200

[50] 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.

[51] J. Donahue, L. A. Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, T. Darrell, and K. Saenko, "Long-term recurrent convolutional networks for visual recognition and description," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 2625–2634.

[52] 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. (CVPR)*, Jun. 2015, pp. 4694–4702.

[53] M. Majd and R. Safabakhsh, "Correlational convolutional LSTM for human action recognition," *Neurocomputing*, vol. 396, pp. 224–229, Jul. 2020.

[54] S. Zhang, S. Guo, L. Wang, W. Huang, and M. Scott, "Knowledge integration networks for action recognition," in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, 2020, pp. 12862–12869.

[55] U. A. Khan, M. A. Martinez-Del-Amor, S. M. Altowaijri, A. Ahmed, A. U. Rahman, N. U. Sama, K. Haseeb, and N. Islam, "Movie tags prediction and segmentation using deep learning," *IEEE Access*, vol. 8, pp. 6071–6086, 2020.

[56] A. B. Sargano, X. Wang, P. Angelov, and Z. Habib, "Human action recognition using transfer learning with deep representations," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, May 2017, pp. 463–469.

[57] M. Rafiq, G. Rafiq, R. Agyeman, G. S. Choi, and S.-I. Jin, "Scene classification for sports video summarization using transfer learning," *Sensors*, vol. 20, no. 6, p. 1702, Mar. 2020.

[58] O. Elharrouss, N. Almaadeed, S. Al-Maadeed, A. Bouridane, and A. Beghdadi, "A combined multiple action recognition and summarization for surveillance video sequences," *Int. J. Speech Technol.*, vol. 51, no. 2, pp. 690–712, Feb. 2021.

[59] 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: http://arxiv.org/abs/1212.0402

[60] J. Liu, J. Luo, and M. Shah, "Recognizing realistic actions from videos in the wild," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 1996–2003.

[61] 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.

[62] S. Abu-El-Haija, N. Kothari, J. Lee, P. Natsev, G. Toderici, B. Varadarajan, and S. Vijayanarasimhan, "YouTube-8M: A large-scale video classification benchmark," 2016, *arXiv:1609.08675*. [Online]. Available: http://arxiv.org/abs/1609.08675

[63] C. V. Sheena and N. K. Narayanan, "Key-frame extraction by analysis of histograms of video frames using statistical methods," *Procedia Comput. Sci.*, vol. 70 pp. 36–40, Jan. 2015.

[64] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2818–2826.

[65] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1251–1258.

[66] 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.

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

FAYAZ A. MEMON received the bachelor's and master's degrees in education from Mehran University of Technology, Jamshoro, Pakistan. He is currently serving as an Assistant Professor for the Department of Computer Systems Engineering, Quaid-e-Awam University of Engineering, Science and Technology, Nawabshah, Pakistan. His research interests include computer vision and machine learning.

UMAIR A. KHAN received the master's and Ph.D. degrees from Alpen-Adria University, Klagenfurt, Austria, in 2010 and 2013, respectively. Since then, he has been working as an Associate Professor and the Head of the Department of Computer Systems Engineering, Quaid-e-Awam University of Engineering, Science and Technology, Nawabshah, Pakistan. He was worked with the Fraunhofer Institute of Integrated Circuits, Erlangen, Germany, and the Machine Perception Laboratory, Hungarian Academy of Sciences, Budapest, Hungary, as a Research Scientist, in 2016 to 2017. His research interests include context-based information retrieval from images and videos using deep learning.

ASADULLAH SHAIKH received the B.Sc. degree in software development from the University of Huddersfield, U.K., the M.Sc. degree in software engineering and management from the University of Gothenburg, Sweden, and the Ph.D. degree in software engineering from the University of Southern Denmark. He is currently working as an Associate Professor, the Head of Research, and a Coordinator of seminars and training with the College of Computer Science and Information Systems, Najran University, Najran, Saudi Arabia. He has more than 95 publications in the area of software engineering in international journals and conferences. He has vast experience in teaching and research. His current research topics are UML model verification, UML class diagrams verification with OCL constraints for complex models, formal verification, and feedback technique for unsatisfiable UML/OCL class diagrams. He has worked as a Researcher at UOC Barcelona, Spain. He is also an Editor of the *International Journal of Advanced Computer Systems and Software Engineering* (IJACSSE) and an international advisory board of several conferences and journals. Further details can be obtained using www.asadshaikh.com

ABDULLAH ALGHAMDI received the B.Sc. degree in information systems from Al-Imam Muhammad Ibn Saud Islamic University, Saudi Arabia, the M.Sc. degree in networking and systems administration from Rochester Institute of Technology, Rochester, NY, USA, and the Ph.D. degree in computer and information systems engineering from Tennessee State University, Nashville, TN, USA. He is currently working as an Assistant Professor and the Head of Department at the Information Systems Department, College of Computer Science and Information Systems, Najran University, Najran, Saudi Arabia. He has many publications in international journals and conferences. He is experienced in teaching, administration, and research. His current research topics include security, privacy, the IoT, interdisciplinary applications, computer education, and academic leadership. He has been a reviewer and a guest editor of many journals, and he attended several conferences.

PARDEEP KUMAR received the master's and Ph.D. degrees from MUET, Jamshoro, Pakistan. He is currently working as a Professor and the Head of the Software Engineering Department, QUEST University, Nawabshah, Pakistan. He is also working as the Director of the Office of Research, Innovation and Commercialization (ORIC), QUEST. His research interests include the fields of the Internet of Things (IoT), machine learning, and big data analytics.

MESFER ALRIZQ received the B.Sc. degree in information systems from King Khalid University, Abha, Saudi Arabia, in 2009, the M.Sc. degree in information technology from Rochester Institute of Technology, Rochester, NY, USA, in 2013, and the Ph.D. degree in computer science from Western Michigan University, Kalamazoo, MI, USA, in 2020. He is currently working as an Assistant Professor and a Coordinator of the Department of Information Systems, College of Computer Science and Information Systems, Najran University, Najran, Saudi Arabia. He has good number of publications in international journals and conferences. He is an experienced teacher, an administrator, and a researcher. His current research interests include information technology, distributed artificial intelligence, human behavior modeling, multi-agent modeling, security, and sentiment analysis.