Noname manuscript No.
(will be inserted by the editor)

Skeleton based Human Activity Recognition using ConvLSTM and Guided Feature Learning

Santosh Kumar Yadav · Kamlesh Tiwari · Hari Mohan Pandey · Shaik Ali Akbar

Received: date / Accepted: date

Abstract Human activity recognition aims to determine actions performed by a human in an image or video. Examples of human activity include standing, running, sitting, sleeping, *etc.* These activities may involve intricate motion patterns and undesired events such as falling. This paper proposes a novel deep convolutional long short-term memory (ConvLSTM) network for skeletal-based activity recognition and fall detection. The proposed ConvLSTM network is a sequential fusion of convolutional neural networks (CNNs), long short-term memory (LSTM) networks, and fully connected layers. The acquisition system applies human detection and pose estimation to pre-calculate skeleton coordinates from the image/video sequence. The ConvLSTM model uses the raw skeleton coordinates along with their characteristic geometrical and kinematic features to construct the novel guided features. The geometrical and kinematic features are built upon raw skeleton coordinates using relative joint position values, differences between joints, spherical joint angles between selected joints, and their angular velocities. The novel spatiotemporal guided features are obtained using a trained multi-player CNN-LSTM combination. Classification head including fully connected layers is subsequently applied. The proposed model has been evaluated on the KinectHAR dataset having 130,000 samples with 81 attribute val-

Santosh Kumar Yadav
Academy of Scientific and Innovative Research (AcSIR), Ghaziabad, UP-201002, India,
CSIR-Central Electronics Engineering Research Institute (CEERI), Pilani, Rajasthan-333031, India, and
DeepBlink LLC, 30 N Gould St Ste R, Sheridan, WY 82801, United States
E-mail: santosh.yadav@pilani.bits-pilani.ac.in

Shaik Ali Akbar
Academy of Scientific and Innovative Research (AcSIR), Ghaziabad, UP-201002, India, and
CSIR-Central Electronics Engineering Research Institute (CEERI), Pilani, Rajasthan-333031, India
E-mail: saakbar@ceeri.res.in

Kamlesh Tiwari
Department of CSIS, Birla Institute of Technology and Science Pilani, Pilani Campus, Rajasthan-333031, India
E-mail: kamlesh.tiwari@pilani.bits-pilani.ac.in

Hari Mohan Pandey
Department of Computer Science, Edge Hill University, Lancashire, United Kingdom
E-mail: Pandeyh@edgehill.ac.uk

ues, collected with the help of a Kinect (v2) sensor. Experimental results are compared against the performance of isolated CNNs and LSTM networks. Proposed ConvLSTM have achieved an accuracy of 98.89% that is better than CNNs and LSTMs having an accuracy of 93.89% and 92.75%, respectively. The proposed system has been tested in realtime and is found to be independent of the pose, facing of the camera, individuals, clothing, *etc.* The code and dataset will be made publicly available.

Keywords Activity Recognition · CNNs · LSTMs · ConvLTM · Skeleton Tracking

1 Introduction

The basic aim of human activity recognition systems is to automatically recognize the activities of an individual with the raw data obtained from sensors. The application of activity detection can be found in many areas like human-computer interaction, video surveillance, sports analysis, video understanding, *etc.* [24,34,19]. Monitoring fall detection and early reporting is also an important application of human activity recognition. The world population is expected to have a 25% increase in the elder population by 2050, it is necessary to assist elderly adults over the age of 65 [6]. Fall is a major cause of an accident and even death, especially in the case of the elderly. An estimated $31 billion is spent on direct medical costs for fall injuries in the US [27], making fall prevention and early reporting necessary.

The classical studies typically studied action recognition using monocular RGB videos [42] which makes it hard to comprehensively represent actions in 3D space. In recent years, for human activity recognition, low-cost and high mobility sensors, like Microsoft Kinect are being widely adopted. Kinect's ability to track skeleton joints has attracted significant attention from computer vision researchers, and different algorithms have been proposed using the skeleton joint information for recognizing human activities. Skeleton joints extracted from the Kinect can be used to calculate features invariant to the human body size, appearance, and change in camera viewpoints [39].

There have been some works on activity recognition in the past, where a combination of convolutional neural networks (CNNs) and long short-term memory (LSTM) networks are utilized for activity recognition [11,33,12]. Many works involved running different models in parallel like an ensemble classifier and fused the scores of each model to predict the final class labels. For example, Li *et al.* [17] used the score fusion of CNN and LSTM models. Although this method improves upon using either CNN or recurrent neural networks (RNNs) individually, it does not utilize the advantages of both models. As done by [17], a lot of previous works on activity recognition involves feeding the skeleton data directly into the model and it is up to the neural network to extract features from those coordinates.

Only skeletal features are not enough for recognizing all the activities. There are a few activities like falling, differentiating between running and walking, *etc.*, which require the rate of change of coordinates of the center of mass, velocity, acceleration, and other derived features

like head to floor distance and the joint angles between the joint points for recognition. Such features cannot be modeled by CNN and LSTM techniques directly, and they require hand engineering. Also, using only the hand-engineered features results in a model which is shallow and dataset-dependent [32].

In this paper, a set of derived features along with the raw skeleton joint coordinates are fed to the deep learning networks as shown in Fig. 2. Initially, we applied skeletal tracking algorithms using a Kinect (v2) sensor and collected 3D joints locations for each frame, and made a skeletal of bones. A set of features have been extracted from the raw data obtained from the Kinect (v2) sensor for improving the model efficiency. The standard features like velocity, acceleration, position of the center of gravity, angle between different body joints, *etc.*, have been derived from the raw body coordinates. After extracting the features, the dataset is preprocessed and inputted to the deep learning models, which consist of fourteen derived features along with seventeen skeleton coordinates. The proposed ConvLSTM model is a sequential combination of CNNs, LSTM, and dense layers, where CNN is used for feature filtering, LSTM is used for classification, and dense layers are used for feature mapping (as shown in Fig. 2). LSTM cells can represent the contextual dependencies in the temporal domain effectively, while the CNNs perform better to process the feature set with more spatial information. By combining both, we retrieved the best set of features containing spatial and temporal information. Finally, a fully connected network is applied to these features to get the classification scores. We have experimentally found that using the combination of CNNs and LSTMs in a serial manner results in better efficiency as compared to using either of them individually, or using it in a parallel mode.

Major contributions of this manuscript are highlighted as follows:

1. The paper presents a privacy-preserving activity recognition and fall detection system using the data obtained from Kinect (v2) sensor.
2. We propose a ConvLSTM model, which is a sequential combination of CNNs, LSTMs, and dense layers. LSTM cells are used to represent the contextual dependencies in the temporal domain, while CNNs are used for extracting the spatial features. The combination of these gives the best set of spatiotemporal features.
3. To preserve the privacy of the user, instead of passing the raw videos directly to the network, a set of derived features along with skeletal joint coordinates are fed to the deep learning network. A set of fourteen derived features along with seventeen skeleton coordinates are inputted to the networks for recognizing the activities and detecting falls.
4. A new dataset is presented for activity recognition and fall detection. This dataset has been collected with all possible variations and has all the features and enough complexity generally required for training a system. During experimentation, this dataset is used for testing the performance of the proposed model to recognize the activities and detecting falls (described in Section 4.1).
5. Experimental results show that the ConvLSTM model achieves better accuracy (98.89%) as compared to the LSTM (92.75%) and CNNs (93.89%) individually.

The outline of this paper is organized as follows. Section 2 presents the literature survey. Section 3 describes the proposed methodology along with feature extraction, CNNs, LSTM, and ConvLSTM models. Section 4 describes the experimental results and the data collection procedure, and also, it compares the performances of CNNs, LSTM, and ConvLSTM models. Section 5 presents the concluding remarks and proposes future directions.

2 Related Works

Various literature has been proposed for activity recognition and fall detection using single or multiple cameras. Multi-view cameras were found to improve accuracy [2], but they lead to higher complexity and duplicate cost [25]. Low-cost depth sensors like Kinect are recently investigated to deal with the above limitations. These approaches used low-resolution depth information from Kinect for joint localization to detect falls. Many literatures have proposed fall detection systems using Kinect depth sensors such as Gasparrini *et al.* [9] placed Kinect sensor on the ceiling and used depth images for fall detection while Uden *et al.* [28] placed Kinect sensor under the bed for detecting fall along with other activities like leaving the room, feet in front of the bed, and activity in the room. However, these approaches suffer in a real-time environment to give accurate results.

Researchers have used different techniques to construct classification models for human activity recognition, for example, Wang *et al.* [31] used Deep fully-connected networks (DFNs) to facilitate a better representation of data as compared to artificial neural networks (ANNs). Vepakomma *et al.* [29] took hand-engineered features obtained from the sensors for human activity recognition. Hammerla *et al.* [11] used five hidden layer DFNs for feature extraction. Generally, DFNs with more number of hidden layers serve as the dense layer for other deep learning algorithms. A few researchers have also used autoencoders, a variety of ANNs used for unsupervised learning, for activity recognition. The aim of an autoencoder is to memorize the dataset representation, typically for dimensionality reduction. Almaslukh *et al.* [1] and Wang *et al.* [30] utilized greedy approaches in which each layer was pre-trained and then fine-tuned. In comparison to this, Li *et al.* [20] have utilized the sparse autoencoders by adding the Kullback Leibler (KL) divergence and introducing noises to the cost function, which ultimately improved the performance for activity recognition. Stacked Autoencoders (SAEs) are used for learning the features in an unsupervised manner, which may be used to enhance the feature extraction for HAR. However, SAEs depend upon the number of layers and their activation functions that makes them hard to search for the optimal solution.

Deep learning based action recognition can be categorized into two broad categories, *i.e.* CNN based approaches [7,17,14,35] and RNN based approaches [8,43,21,16,41]. CNNs have obtained promising results in the image/video classification, signal processing, *etc.* It performs better for processing the feature set with more spatial information [36]. CNNs comprise one or many convolutional layers. Once the convolution operation completes, pooling and the fully connected

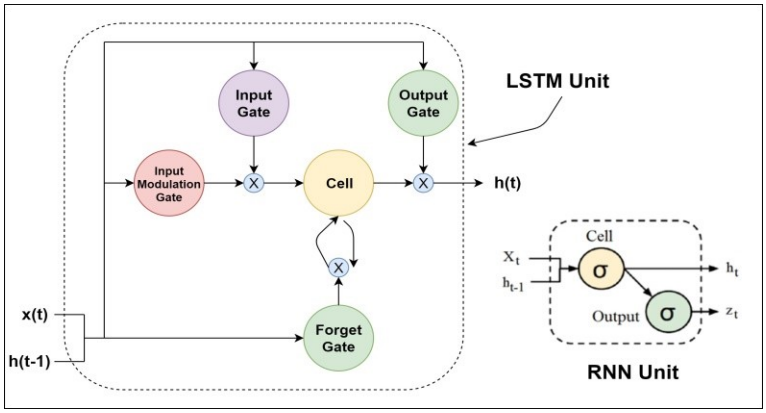


Fig. 1: Sequence learning using, (1) LSTM, (2) RNN [40].

layers are combined to perform the classification at the final output [15]. Ding *et al.* [7] had investigated different skeletal features using CNNs for the 3D action recognition. They encoded the spatial skeleton features into images by incorporating various encoding techniques. Also, they studied the performance implications of the different skeleton joints in the features extraction. Li *et al.* [17] proposed two-stream CNNs in which one stream was applied on raw joint coordinates whereas the other one was used for the motion data from the subtraction of joint coordinates from subsequent frames. Ke *et al.* [14] converted the skeletal features into images and passed it to deep CNNs. Weng *et al.* [35] utilized Naive-Bayes mutual information maximization (NBMIM) [38] to CNNs for the action recognition. For the sequential information, RNNs are best suited. They are widely applied in various fields like speech recognition and natural language processing [40,18, 10]. Activity recognition can also be considered a sequential problem. Du *et al.* [8] have presented a skeleton-based activity recognition system using an end-to-end deep learning model consisting of hierarchical RNNs. In their methodology, they have divided the human skeleton obtained from the Kinect sensor into five different parts. These parts are then fed to five different bidirectional RNNs. Among various RNN architectures, LSTMs are most popular due to their memory capacity and remembering useful data for an extended period (Fig. 1). For activity recognition, LSTMs are very robust with real-world recognition [13]. Zhu *et al.* [43] proposed an approach to automatically learn the human skeletal representations. They used RNNs and LSTMs to learn the long-term temporal dependency in the dataset. To model the joint co-occurrences with the LSTMs and RNNs, joint position values were used as the input for each time slot. Liu *et al.* [21] proposed a new gated scheme in LSTM for sequential action recognition. Lee *et al.* [16] proposed a temporal sliding LSTM, which includes short, medium, and long-term units. Zhang *et al.* [41] presented an element-wise attention gate using RNN's for action recognition.

The combination of CNNs and LSTMs is among the most emerging hybrid models for activity recognition. These are especially being applied to vision tasks involving sequential inputs and outputs. CNNs basically consists of two modules. First, feature map construction or extraction,

Table 1: Summary of the literature review.

Author	Sensing-Modality	Learning Method	Dataset	Classes	Subjects
Gasparrini *et al.* [9], 2014	RGB-D Camera (Kinect)	Ad-Hoc Segmentation	Self-collected for Fall detection	2	NA
Jiang Wang *et al.* [32], 2013	RGB-D Cameras (Kinect Sensors)	Actionlet Ensembles	MSR-Action3D dataset, DailyActivity3D dataset, Multiview 3D Event, Cornell Activity Dataset, and CMU MoCap Dataset	20, 16, 8, 4, 5	10, NA, 8, 16, 5, NA
Vepakomma *et al.* [29], 2015	Wearable Inertial Sensors	Multilayer Feed Forward Neural Network	Self-collected	22	2
Hammerla *et al.* [11], 2016	Wearable Inertial Sensors	Bidirectional LSTMs	Opportunity, PAMAP2, DaphNet Gate Detection	-,12,-	-,9,10
Ding *et al.* [7], 2017	RGB-D Camera (Kinect Sensor)	CNNs	NTU RGB+D	60	40
Li *et al.* [17], 2017	RGB-D Cameras (Kinect Sensors)	CNN + LSTM	NTU RGB+D	60	40
Ke *et al.* [14], 2017	RGB-D Camera (Kinect)	CNNs	SBU Kinect Interaction Datset, CMU Dataset, and NTU RGB+D	8, 45, 60	-, -, 40
Du *et al.* [8], 2015	RGB-D Cameras (Kinect Sensors) and other wearable sensors	Hierarchical RNNs	MSR Action3D dataset, Berkeley MHAD, Motion Capture Dataset HDM05	20, 11, 130	10, 12, 5
Liu *et al.* [21], 2017	RGB-D Cameras (Kinect Sensors) and Other Sensors	Spatiotemporal LSTMs	NTU RGB+D, UT-Kiect Dataset, SBU Interaction Dataset, SYSU-3D dataset, ChaLearn Gesture Dataset Names, MSR Activity 3D dataset, and Berekeley MHAD	60, 10, 8, 12, 20, 20, 11	40, -, -, 40, 27, 10, 12
Lee *et al.* [16], 2017	Many RGB-D Cameras (Kinect Sensors)	Temporal Sliding LSTMs	MSR-Action 3D, UT-Kinect Action, NTU RGB+D, Northwestern UCLA, UWA3DII-datasets	20, 10, 60, 10, 30	10, 10, 40, 10, 10
Zhang *et al.* [41], 2018	RGB-D Cameras (Kinect Sensors)	Element-wise Attention using LSTMs	NTU RGB+D, Northwestern UCLA, SYSU Human Object Interaction, JHMDB datasets	60, 10, 12, 21	40, 10, 40, -
Ordonez *et al.* [22], 2016	Wearable Inertial Sensors	CNN + LSTM	Opportunity and Skoda Datasets	17, 10	20, -
Singh *et al.* [26], 2017	Wearable Inertial Sensors	CNN + LSTM	Self-collected	1	13

and second, a basic classifier. The hybrid model of CNN and LSTM uses CNN for feature extraction, and LSTM for feature classification. Ordonez *et al.* [22], Yao *et al.* [37] and Singh *et al.* [26], used a combination of CNNs and LSTMs, and demonstrated a tremendous improvement in result. They run both CNN and LSTM models in parallel and used score fusion for the final prediction. Table 1 presents a summary of the literature survey. Work in this research demonstrates a sequential ConvLSTM approach using the best features of both CNN and LSTM. We have focused on combining the CNNs, LSTMs, and dense layers in a sequential manner and take advantage of all three methods.

3 Proposed Methodology

The proposed methodology of this paper is threefold. First, the human body frames are acquired from the Kinect-v2 sensor and tracked the 3D skeleton joint coordinates. Then a 3D joints normalization technique is applied for the preprocessing of the data. The 3D coordinates are used to make a 3D bounding box over the tracked human. Suitable features like velocity, acceleration, angle between skeleton joints, height, width *etc.* have been extracted for identifying different activities. For each activity, important features have been selected and then constructed feature vectors. Second, the dataset is stored in a CSV format consisting of the raw skeleton joint coordinates and the extracted features. Third, the dataset containing joint values along with extracted manual features are inputted to deep learning networks *i.e.* CNNs, LSTMs, and Con-

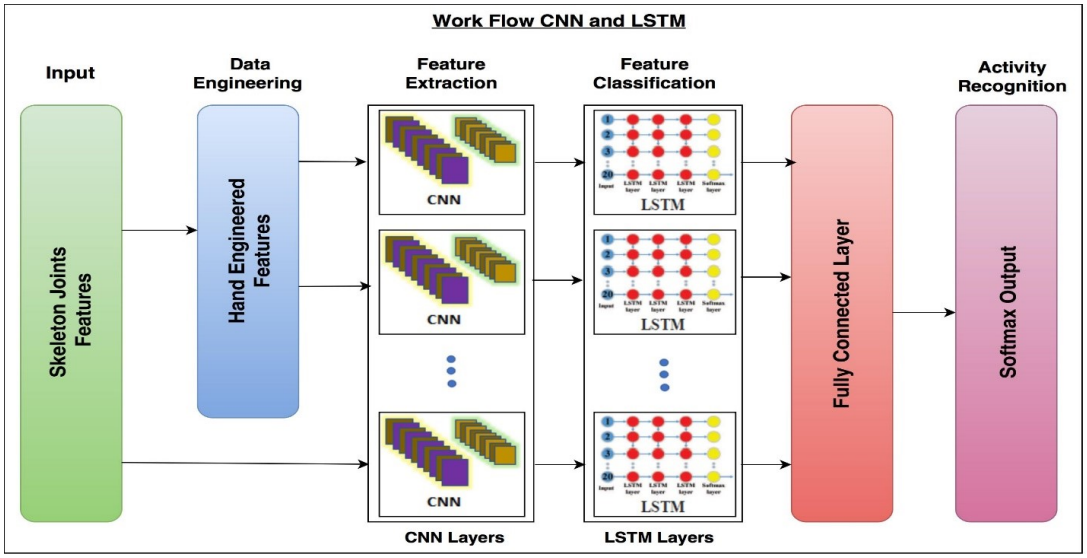


Fig. 2: The proposed model of ConvLSTM. From the raw input videos 3D skeleton coordinates are extracted which are passed to calculate the geometrical and kinematic features. The extracted features along with raw skeleton joint coordinates are passed to CNNs for extracting the automated spatial features. These spatial features are then passed to LSTMs for extracting the temporal features. Finally, fully connected layers are applied to classify the activities and calculated the Softmax scores.

vLSTM for the activity recognition and fall detection. Fig. 2 illustrates the proposed model of the ConvLSTM, which is a sequential fusion of CNNs and LSTMs.

3.1 Preprocessing

This subsection presents the preprocessing methods applied to the raw data obtained from the Kinect (v2) sensor. We applied skeleton joint estimation methods to acquire the body frames and applied 3D joints tracking methods. Fig. 3 presents the twenty-five skeleton joints tracked at each instance. These skeletal joints include knee right, hip right, knee left, hip left, foot left, ankle left, foot right, ankle right, head, spine mid, wrist left, shoulder right, shoulder left, wrist right, elbow right, and elbow left. All these joint values contain X, Y, Z coordinates in the space. As Chen *et al.* [4] stated, not all the skeletal joints are useful but only some of the skeletal joints are informative for a particular activity, in our case we excluded coordinates like hand tip, thumb, neck, *etc.*, which are not very important for recognizing the intended activities. Once the 3D skeletal coordinates are available, we applied 3D joints normalization techniques to make 3D bounding boxes across the tracked human skeleton. The bounding box varies as the person moves in the video. Fig. 4 presents the block diagram of the proposed automatic dataset labeling procedure. Table 2 describes the list of skeletal joints tracked, set of derived features, and activity class labels. The normalization technique used for stabilizing the convergence of the loss functions

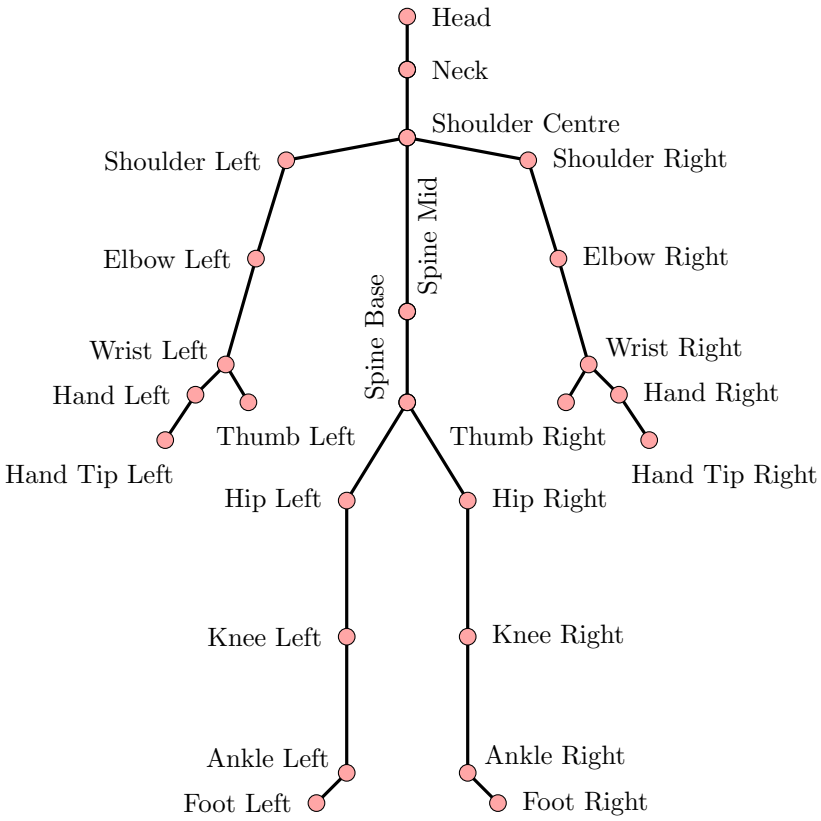


Fig. 3: The 25 skeleton joints tracked using Kinect (v2) sensor.

in the proposed work is min-max normalization. Suppose, if X represents the training dataset then:

$$X = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{1}$$

3.2 Geometric and Kinematic Features Calculation

The 3D human skeleton joint coordinates are used for evaluating different features and constructing the feature vectors. Feature vectors are used for deciding which joints of the body parts to be tracked for different activities. Only discriminative features are utilized for each activity.

Angle Between Skeleton Joints: The 3D coordinates of different body joints are connected using a line and drawn a skeleton. Here, only 10 joints, namely shoulder left, shoulder center, shoulder right, spine base, knee left, hip right, ankle left, hip left, ankle right, and knee right, are utilized for calculating the angle values. These are most relevant among the twenty-five skeleton joints to recognize the activities. A set of angle values are calculated using these joint coordinates.

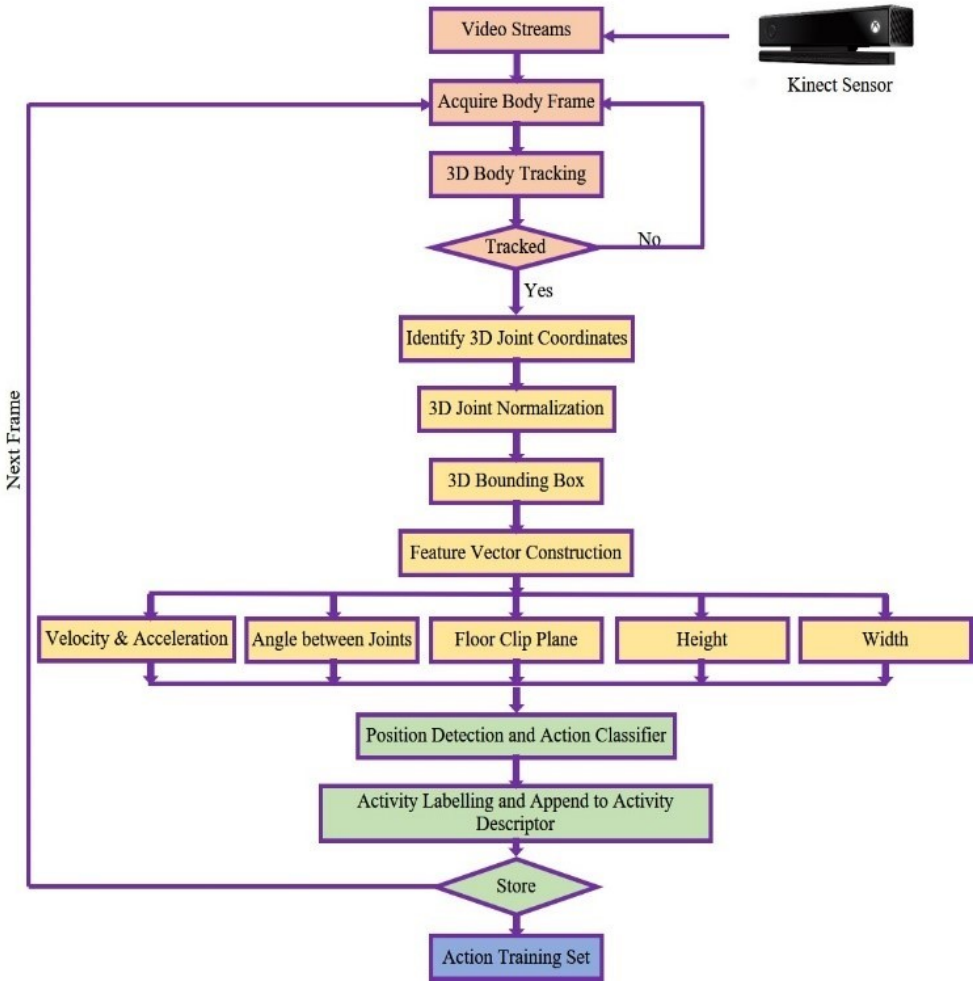


Fig. 4: The block diagram of the proposed data collection procedure. The input video streams from the Kinect-v2 sensor are used to acquire the 3D skeleton joint coordinates. Then a 3D joint normalization technique is used for the normalization. A bounding box is made across the practitioner using the upper and lower body joint coordinates. Suitable features such as the angle between joints, velocity, acceleration, height, width, *etc.* are calculated and stored in the dataset along with their activity labels. This dataset is later passed to deep learning algorithms for training the network.

Fig. 5 presents an example of calculating the joint angles between the left side of the ankle, knee, and hip. The average of the difference of hip to knee values and ankle to knee values are used for calculating the angle values. If $P,\ Q,\ R$ represents the distance before coordinate values as $P_1 = x_1 - y_1$, $Q_1 = x_2 - y_2$, and $R_1 = x_3 - y_3$, then the angle between skeleton joints are:

$$\theta = \frac{PQR}{(PQ * QR)} \tag{2}$$

Table 2: Feature Set Specifications.

Sr.	Label	Description
1	Skeleton joints from Kinect (v2)	Hip Right, Foot Left, Knee Right, Knee Left, Hip Left, Foot Right, Ankle Right, Ankle Left, Head, Spine Mid, Spine Base, Wrist Left, Shoulder Right, Shoulder Left, Elbow Left, Wrist Right, Elbow Right
2	Derived Features	Velocity in X, Y, Z directions, Distance from the floor, Angle Left Standing, Acceleration in X, Y, Z directions, Height, Angle Standing, Angle Sitting Left, Angle Sitting Right, Angle Standing Right, Width
3	Class Labels	Bending, Fall, Lying Down, Sitting, Standing, Walking Fast, Walking Slow

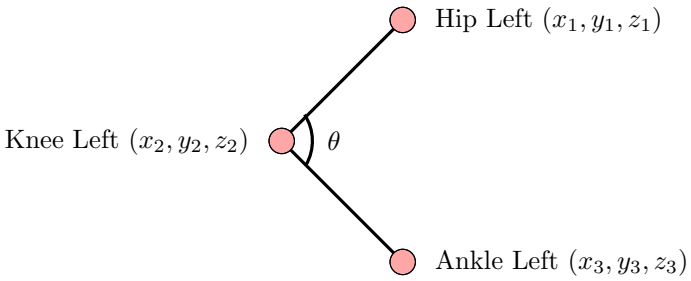


Fig. 5: An example of the angle calculation from the right side between the ankle, knee, and hip.

$$where,\ \ PQR = P_1{*}P_2{+}Q_1{*}Q_2{+}R_1{*}R_2;\ \ PQ = \sqrt{P_1^2 + Q_1^2 + R_1^2}\ ;\ and\ \ QR = \sqrt{P_2^2 + Q_2^2 + R_2^2}$$

$$Angle = \frac{cos^{-1}(\theta) * 180}{pi} \tag{3}$$

Velocity Estimation: Velocities in the X, Y, Z directions are estimated using the differences between the positions of the human skeleton at the time instance of t and $t+1$. The displacement between the two consecutive frames is calculated using the spine mid joint coordinate of the person. Next, the displacement per unit time is used to calculate the velocity of the person.

$$Velocity = \frac{Displacement\ of\ the\ tracked\ person\ between\ frames}{Time} \tag{4}$$

Acceleration Estimation: Acceleration in the X, Y, Z directions are estimated using the changes in the velocity between consecutive frames as follows:

$$Acceleration = \frac{Velocity\ of\ the\ tracked\ person\ between\ frames}{Time} \tag{5}$$

Distance from Floor: This is used to estimate the distance between the floor and joint coordinate of the head of the tracked person.
Depth Estimation: Depth is the distance from the camera to the nearest object. It was estimated with the help of the head joint's Z coordinate of the tracked person.
Width Estimation: The difference between maximum right joint and maximum left joint coordinates are used to calculate the depth. The extreme right joint values are calculated using the elbow right, hip right, knee right, shoulder right, hand-tip right, ankle left, foot right, and head joint values. A similar procedure is used to find the left extreme joints using all the left side joint coordinates.

$$Width = abs(Max.\ Right\ Joint - Max.\ Left\ Joint) \tag{6}$$

Height Estimation: The difference between extreme top joints and extreme bottom joints is used to calculate the height. The extreme bottom is calculated using the ankle left, knee right, ankle right, knee left, foot right, ankle left, foot left, and ankle right joint coordinates. Extreme top calculated using the head, hand tip left, hand tip right, ankle right, elbow right, ankle left, elbow left, knee right, and knee left joints values.

$$Height = abs(Top\ Joint - Bottom\ Joint) \tag{7}$$

3.3 Classification Models

In this section, the final dataset along with derived features are inputted to the deep learning network for classification. Three different classification methods have been used for the classification namely CNN, LSTM, and the newly designed ConvLSTMs.
Convolutional Neural Network Architecture: Firstly, the activity recognition has been implemented using the CNNs [23]. Let $X_t^0 = [X_1, X_2, \ldots, X_n]$ be the readings from the sensor data as an input vector. Here, n is the number of input samples. The convolutional layer's output can be given as:

$$C_i^{l,j} = \sigma(B_j + \sum_{m=1}^{M} W_m^j * X_{i+m-1}^{0,j}) \tag{8}$$

where, l correspond to index of layer, and σ is a sigmoid activation function. B_j is a bias corresponding to the j^{th} feature, and M is the filter size. W_m^j represents the weight for the j^{th} feature map and m^{th} filter index.

Three input channels are used for the RGB dataset as the input layers. In the convolution layer, six filters are passed along with setting the kernel sizes, padding, and ReLU activation

functions. Max-pooling is used as a pooling layer, which down-samples the image data and reduces the dimensionality for reducing the processing time. The output of which has been passed to the fully connected layers. Ultimately, the Softmax scores give the probabilities of the classes. As far as the negative log-likelihood cost function is considered, it is minimized using a stochastic gradient descent optimizer.

Long Short-Term Memory Architecture: Secondly, we have implemented activity recognition using LSTM, an improved version of RNN, which avoids the vanishing gradient problem and consists of memory cells. LSTMs mainly consist of gates such as forget, input, and output gates to control and protect the cell states [3]. Forget gate is a binary gate used to decide how much information to let through. The input gate layer is used to decide the new information that needs to be stored in the cell state. The output gate consists of a sigmoid gate, which decides what parts of the cell to give as output. Passing the cell state through the *tanh* layer, and multiplying it with the output obtained from the *sigmoid* gate provides the final output (Fig. 1). The standard equations describing the actions of each gate can be given as follows:

$$i_t = \sigma\ (W_{(Xi)}\ X_t + W_{(Hi)}\ H_{(t-1)} + W_{(Ci)}\ C_{(t-1)} + B_i) \tag{9}$$

$$f_t = \sigma\ (W_{(Xf)}\ X_t + W_{(Hf)}\ H_{(t-1)} + W_{(Cf)}\ C_{(t-1)} + B_f) \tag{10}$$

$$o_t = \sigma\ (W_{(Xo)}\ X_t + W_{(Ho)}\ H_{(t-1)} + W_{(Co)}\ C_t + B_o) \tag{11}$$

$$C_t = f_t\ C_{(t-1)} + i_t\ tanh\ (W_{(Xc)}\ X_t + W_{(Hc)}\ H_{(t-1)} + B_c) \tag{12}$$

$$H_t = o_t\ tanh\ (C_t) \tag{13}$$

where, W_i, W_f, W_o are the weight matrices, and X_t is an input to the LSTM cells at the time instance t. σ is the *Sigmoid* activation function, whereas *tanh* is a *hyperbolic tangent* activation function. f, i, and o are the forget, input and output gates, respectively. C represent the state of a memory cell. $B_i, B_c, B_f,\ and\ B_o$ are the bias vectors.

A different combination of batch sizes, hidden layers, and learning rates has been investigated. The best results were obtained using 32 hidden layers for 7 classes with a learning rate of 0.0025, lambda loss amount of 0.0015, and batch size of 1500. Two LSTM cells were stacked which adds deepness to the network. For loss computation, the Softmax loss function was used and optimized using Adam optimizer.

ConvLSTM Architecture: In this, a ConvLSTM network has been proposed using the fusion of CNNs, LSTM, and dense layers. Here, CNNs are used for spatial feature extraction, LSTMs are used for sequence prediction, and dense layers are used for mapping the features to get more separable space (Fig. 2). Fig. 6 demonstrates a traditional activity recognition model, where the

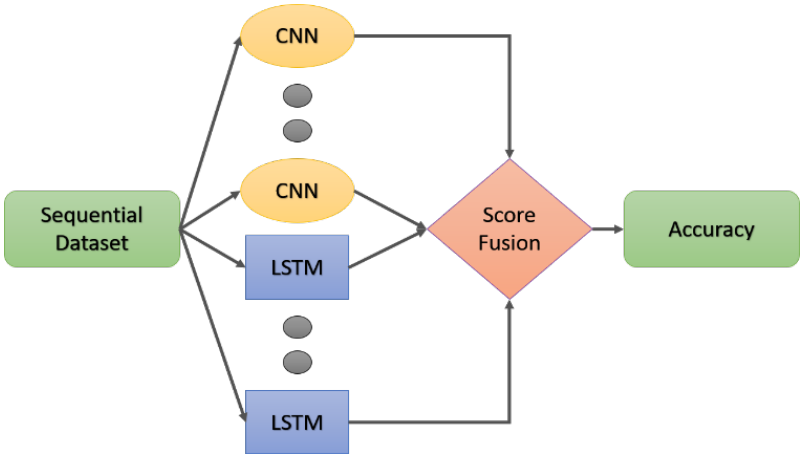


Fig. 6: Demonstrates a parallel ConvLSTM model with score fusion.

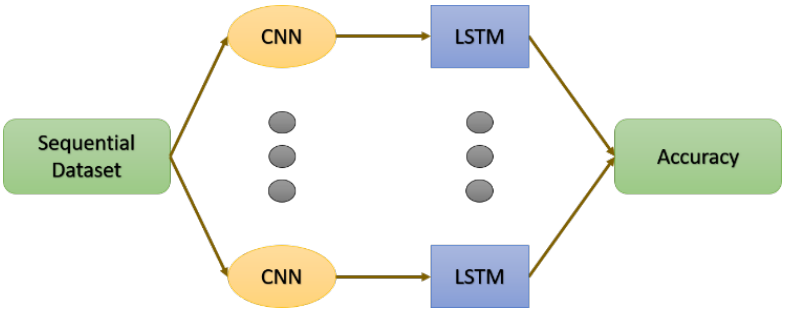


Fig. 7: Demonstrates a sequential ConvLSTM mode.

parallel fusion of CNN and LSTM has been performed for the ConvLSTM model. This has been used in many works, previously [11,33,12]. Although this approach is much better than using either only CNN or only LSTM, yet it does not use the efficiency of both the models to their fullest.

In this, a sequential fusion of CNN, LSTM, and dense layers is used as shown in Fig. 7. Here, the outputs of the last hidden layer of CNNs are inputted to the LSTM layers followed by the fully connected layers for the classification. The equations of ConvLSTM can be given as follows:

$$F_t = \sigma\ (W_{(XF)}\ *\ X_t + W_{(HF)}\ *\ H_{(t-1)} + B_F) \tag{14}$$

$$I_t = \sigma\ (W_{(XI)}\ *\ X_t + W_{(HI)}\ *\ H_{(t-1)} + B_I) \tag{15}$$

$$\check{C}_t = tanh\ (W_{(X\check{C})}\ *\ X_t + W_{(H\check{C})}\ *\ H_{(t-1)} + B_{\check{C}}) \tag{16}$$

$$O_t = \sigma\ (W_{(XO)}\ *\ X_t + W_{(HO)}\ *\ H_{(t-1)} + B_O) \tag{17}$$

$$C_t = F_t\ \odot\ C_{(t-1)} + I_t\ \odot\ \check{C}_t \tag{18}$$

$$H_t = O_t\ \odot\ tanh\ (C_t) \tag{19}$$

where, σ ($sigmoid$) and $tanh$ ($hyperbolic - tangent$) are non-linear activation functions. $\odot$ represents the $Hadamard$ product, and $*$ represents the convolution operations. The inputs (X_t), cells (C_t), hidden states (H_t), forget gates (F_t), input gates (I_t), input-modulation gates ($\check{C}_t$), and output gates (O_t) are all $M \times N \times F$ (rows, columns, feature maps) dimensional $3D$ tensors. The memory cell C_t is the most crucial module, acting as an aggregator of the state information controlled by the gates.

Initially, the helper functions are defined to increase the reusability and readability of the code. The hyper-parameters like the size, number of layers, steps, batch sizes, and learning rate (0.0001) have been set to an optimum value. Next, we have constructed the LSTM cells and reshaped the dataset for LSTM into sequence length, batches, and channels. Then we applied ReLU activation and set dropout regularization, which operates simultaneously on gates, cells, and output responses of LSTM neurons. Finally, the logit functions are used for cost function measurement. Further, we used Adam optimizer for cost function optimization and utilized gradient clipping. For the training of the network, we set the checkpoint path, saver function, initialized session, set iterations, computed loss and accuracy on the validation dataset, and saved the checkpoint for further testing the model. The performance is tremendously improved by using a sequential model (Fig. 7) as compared to score fusion (Fig. 6). The experimental results and comparisons of three different models *i.e.* CNNs, LSTMs, and ConvLSTM for activity recognition are presented in the next sections.

4 Experimental Result

This section presents the dataset building procedure, and experimental results using the deep learning algorithms *i.e.* CNNs, LSTMs, and ConvLSTM. The proposed model has been tested on a newly collected KinectHAR dataset recorded by the Kinect-v2 sensor. Only skeleton joints coordinates along with suitable features are stored for inputting to the deep learning model.

4.1 Dataset Building

With the development of cost-effective RGB-D sensing technologies, now it is more convenient to get 3D and depth data. We utilized Microsoft Kinect (v2) sensor for the data collection, which is a depth sensor-based motion-sensing input device that offers a convenient way to record and capture human skeleton joints. The name of Kinect is a combination of kinetic and connects [4]. It produces three-dimensional RGB-D data. Kinect runs at 30 fps and has a resolution of 640×480p for both *i.e.* video and depth [5]. Kinect (v2) has a sensing range of 4 meters [5].

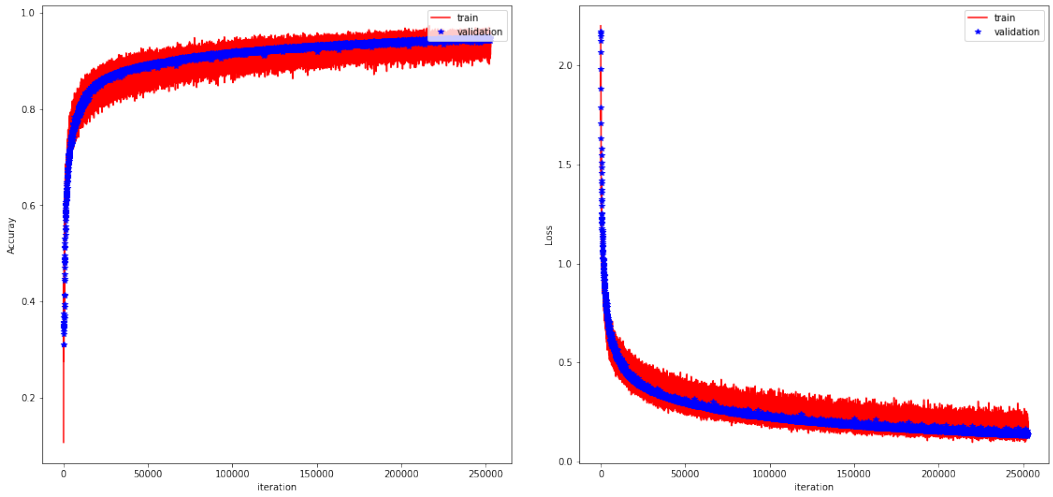
For the dataset collection, 20 different people (12 males and 8 females) have participated and performed seven different activities which include sitting, standing, bending, walking fast, walking slow, lying, and fall activities. Every person performed each activity for more than two minutes with all possible variations, so that it can identify activities accurately in a real-time environment. As we do not record the videos, only the skeleton coordinates are used, the privacy is preserved. Throughout the experiment, Kinect (v2) sensor was placed at a two-meter height above the ground. All the experiments have been performed at a range of 0.5 meters to 4.0 meters in front of the camera. The final dataset contains a total of 130,000 samples with 81 attribute values. The source codes and dataset will be made publicly available to the research community.

4.2 Model Evaluation

This section describes the experimental results obtained using three different deep learning algorithms, namely CNN, LSTM, and ConvLSTM. In the ConvLSTM, CNN is used for feature filtering, LSTM is used for the sequential classification, and fully connected layers are used for feature mapping, as illustrated in Fig. 2. LSTM cells represent the contextual dependencies in the temporal domain effectively, while the CNNs perform better to process spatial features. The combination of these gets the best set of spatial features from CNNs and long-term temporal dependency from LSTMs.

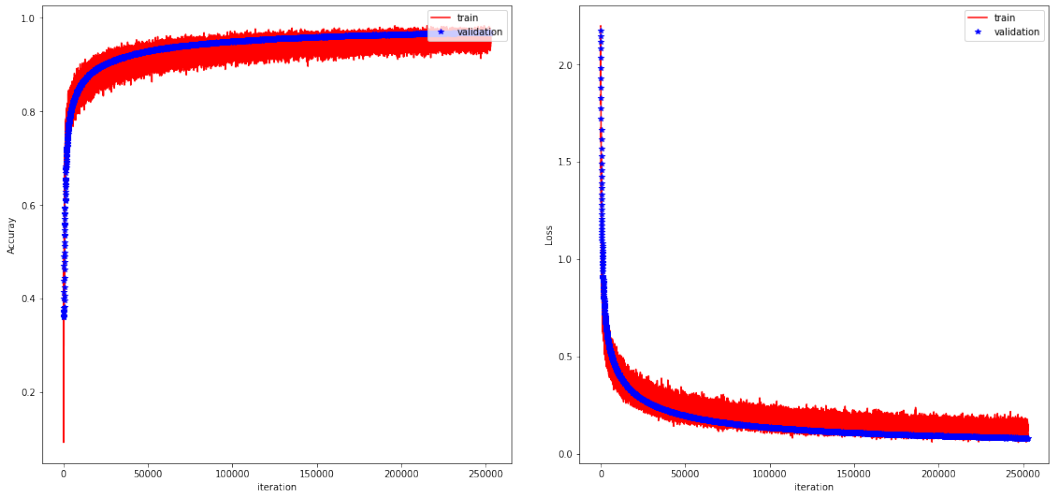
The dataset has been split into train and validation sets with a ratio of 60:20, while the remaining 20% is left for the testing. The training data has been used for training the classifiers, while the validation dataset has been used for measuring the performance and accuracy of the trained model. Model loss is obtained using categorical cross-entropy, due to its suitability for measuring the performance of the final layer with Softmax activation. All three models were trained for 200 epochs on a machine that has NVidia TITAN-X GPU. The performance of the system has been measured based on the different measures, which include precision, recall, $F1$-score, and accuracy. If we represent T_N as true negatives, T_P as true positives, F_N as false negatives, and F_P as false positives, then the performance metrics can represent as follows:

$$Recall = \frac{T_P}{T_P + F_N} \tag{20}$$



(a) Model accuracy curve (b) Model loss curve

Fig. 8: Model accuracy and loss curves using LSTMs.

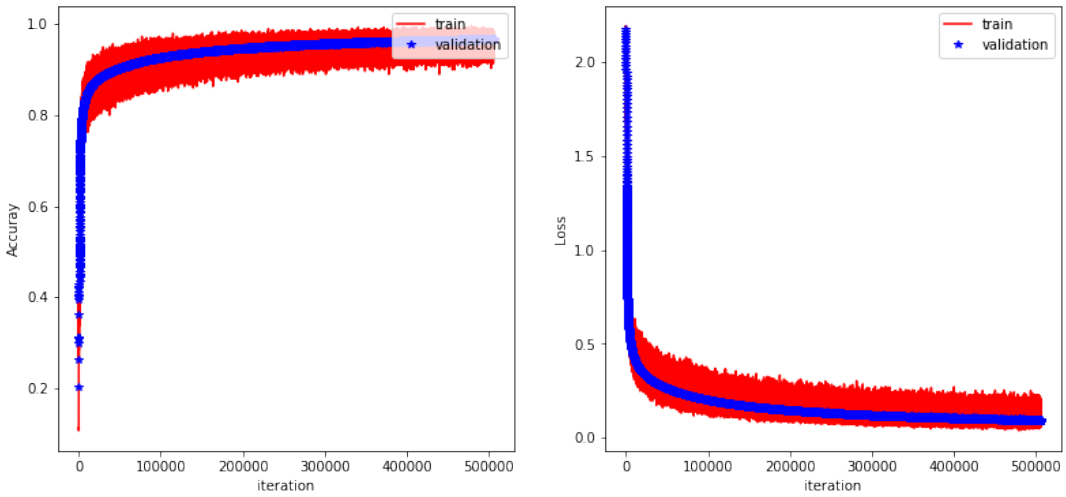


(a) Model accuracy curve (b) Model loss curve

Fig. 9: Model accuracy and loss curves using CNNs.

$$Precision = \frac{T_P}{T_P + F_P} \tag{21}$$

$$F1 - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{22}$$



(a) Model accuracy curve (b) Model loss curve

Fig. 10: Model accuracy and loss curves using ConvLSTM.

Table 3: Precision, recall, and F1-score using LSTM, CNNs, and ConvLSTM.

Sr. No.	Algorithm	Performance Metrics	Score Percentage
1	LSTM	Precision	91.09%
		Recall	90.83%
		$F1$-Score	90.55%
2	CNN	Precision	92.71%
		Recall	92.49%
		$F1$-Score	92.01%
3	ConvLSTM	Precision	97.89%
		Recall	97.83%
		$F1$-Score	97.75%

$$Accuracy = \frac{T_P + T_N}{T_P + F_P + F_N + T_N} \tag{23}$$

Table 3 presents the precision, recall, and $F1$-score values for different algorithms performed with ± 1 percentage change. We have applied different machine learning and deep learning algorithms such as SVMs, decision trees (DT), random forest (RF), artificial neural networks (ANNs), CNNs, LSTMs, and ConvLSTMs. The comparison of their accuracies is shown in Table 4. All the accuracies and plots are calculated for 200 epochs. The accuracy and loss curves using LSTM, CNN, and ConvLSTM are shown in Fig. 8, Fig. 9, and Fig. 10 respectively. Pro-

Table 4: Comparison of accuracies using different algorithms.

Sr. No.	Algorithm	Model Accuracy
1	SVM	70.21%
2	DT	74.80%
3	RF	76.48%
4	ANN	78.59%
5	LSTM	92.75%
6	CNN	93.89%
7	**ConvLSTM**	98.85%

Table 5: Activity-wise precision, recall, and $F1$-score using the ConvLSTM.

Sr. No.	Activity	Precision	Recall	$F1$-Score
1	Standing	96%	100%	98%
2	Walking Slow	92%	96%	94%
3	Walking Fast	99%	95%	97%
4	Sitting	100%	100%	100%
5	Bending	96%	100%	98%
6	Fall	83%	100%	91%
7	Lying Down	100%	98%	99%

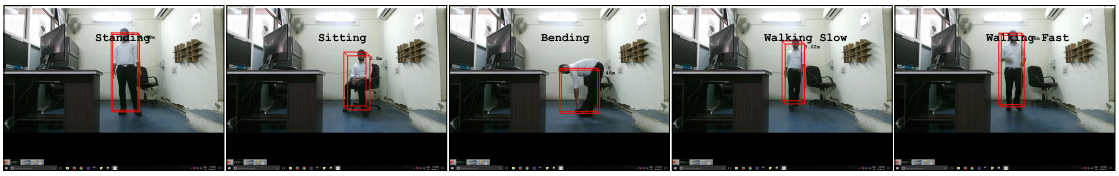


Fig. 11: Realtime testing of the standing, sitting, bending, walking slow, and walking fast activities(working model is presented to demonstrate realtime testing as https://www.youtube.com/watch?v=2PqkyXMVBLg.

posed ConvLSTM results in a better accuracy compared to LSTM and CNNs, as illustrated in Table 4. The accuracy of ConvLSTM is approximately 5-6% better compared to either LSTM or CNN individually. Table 5 presents the precision, recall, and $F1$-score of each class obtained using the ConvLSTM model. As we can see from the Table 3 and Table 4, ConvLSTM gives the best results in comparison to other algorithms, so we have stored the trained model using ConvLSTMs and tested the model in real-time. Fig. 11 presents the activity recognition results obtained in realtime. The performance is sufficiently high for the general adoption of the system.

5 Conclusion

This paper presented a privacy-preserving activity recognition and fall detection system using a single Kinect (v2) sensor and ConvLSTM. The proposed system derives geometrical and kinematic features and passes them along with the raw skeleton coordinates into deep learning networks. As the system uses only derived features along with raw skeleton joint coordinates and does not use the actual images of the user, the privacy of the user is protected. We proposed a simple and effective method based on the sequential fusion of CNNs and LSTM, named as ConvLSTM model. The performance of the deep learning-based classification algorithms, namely CNN, LSTM, and ConvLSTM, has been compared on the novel dataset consisting of 130,000 samples along with 81 attribute values. The proposed system recognizes standing, walking slow, walking fast, sitting, bending, fall, and lying down activities. The proposed system is unobtrusive to the users and independent of the camera orientation, clothing, *etc.* The system gives sufficiently high performance for activity recognition and fall detection for the general adoption of the system. The source code and presented dataset will be made publicly available.

Acknowledgements

The authors would like to thank anonymous reviewers and our parent organizations for extending their support for the betterment of the manuscript. Special thanks to Dr. J. L. Raheja (Chief Scientist, CSIR-CEERI, Pilani) and everyone who had participated in the data collection. We appreciate the assistance provided by CSIR, India.

Conflict of Interest

The authors declare that there is no conflict of interest.

Data availability statement

The authors have provided information of data availability in the manuscript.

Author contributions

Santosh Kumar Yadav: Conceptualization, Methodology, Writing-Original Draft, Editing. Kamlesh Tiwari: Conceptualization, Methodology, Writing-Original Draft, Editing. Hari Mohan Pandey: Methodology, Writing-Original Draft, Editing. Shaik Ali Akbar: Methodology, Writing-Original Draft, Editing.

References

1. Almaslukh, B., AlMuhtadi, J., Artoli, A.: An effective deep autoencoder approach for online smartphone-based human activity recognition. Int. J. Comput. Sci. Netw. Secur **17**(4), 160–165 (2017)
2. Auvinet, E., Multon, F., Saint-Arnaud, A., Rousseau, J., Meunier, J.: Fall detection with multiple cameras: An occlusion-resistant method based on 3-d silhouette vertical distribution. IEEE transactions on information technology in biomedicine **15**(2), 290–300 (2010)
3. Bengio, Y., Simard, P., Frasconi, P.: Learning long-term dependencies with gradient descent is difficult. IEEE transactions on neural networks **5**(2), 157–166 (1994)
4. Chen, H., Wang, G., Xue, J.H., He, L.: A novel hierarchical framework for human action recognition. Pattern Recognition **55**, 148–159 (2016)
5. Cippitelli, E., Gasparrini, S., Gambi, E., Spinsante, S.: A human activity recognition system using skeleton data from rgbd sensors. Computational intelligence and neuroscience **2016** (2016)
6. DeSA, U., et al.: World population prospects: the 2012 revision. Population division of the department of economic and social affairs of the United Nations Secretariat, New York **18** (2013)
7. Ding, Z., Wang, P., Ogunbona, P.O., Li, W.: Investigation of different skeleton features for cnn-based 3d action recognition. In: 2017 IEEE International Conference on Multimedia & Expo Workshops (ICMEW), pp. 617–622. IEEE (2017)
8. Du, Y., Wang, W., Wang, L.: Hierarchical recurrent neural network for skeleton based action recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1110–1118 (2015)
9. Gasparrini, S., Cippitelli, E., Spinsante, S., Gambi, E.: A depth-based fall detection system using a kinect® sensor. Sensors **14**(2), 2756–2775 (2014)
10. Grushin, A., Monner, D.D., Reggia, J.A., Mishra, A.: Robust human action recognition via long short-term memory. In: The 2013 International Joint Conference on Neural Networks (IJCNN), pp. 1–8. IEEE (2013)
11. Hammerla, N.Y., Halloran, S., Plötz, T.: Deep, convolutional, and recurrent models for human activity recognition using wearables. arXiv preprint arXiv:1604.08880 (2016)
12. Hou, L., Wan, W., Han, K., Muhammad, R., Yang, M.: Human detection and tracking over camera networks: A review. In: 2016 International Conference on Audio, Language and Image Processing (ICALIP), pp. 574–580. IEEE (2016)
13. Inoue, M., Inoue, S., Nishida, T.: Deep recurrent neural network for mobile human activity recognition with high throughput. Artificial Life and Robotics **23**(2), 173–185 (2018)
14. Ke, Q., An, S., Bennamoun, M., Sohel, F., Boussaid, F.: Skeletonnet: Mining deep part features for 3-d action recognition. IEEE signal processing letters **24**(6), 731–735 (2017)
15. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. nature **521**(7553), 436–444 (2015)
16. Lee, I., Kim, D., Kang, S., Lee, S.: Ensemble deep learning for skeleton-based action recognition using temporal sliding lstm networks. In: Proceedings of the IEEE international conference on computer vision, pp. 1012–1020 (2017)
17. Li, C., Wang, P., Wang, S., Hou, Y., Li, W.: Skeleton-based action recognition using lstm and cnn. In: 2017 IEEE International Conference on Multimedia & Expo Workshops (ICMEW), pp. 585–590. IEEE (2017)
18. Li, J., Luong, M.T., Jurafsky, D.: A hierarchical neural autoencoder for paragraphs and documents. arXiv preprint arXiv:1506.01057 (2015)
19. Li, M.W., Geng, J., Hong, W.C., Zhang, L.D.: Periodogram estimation based on lssvr-ccpso compensation for forecasting ship motion. Nonlinear Dynamics **97**(4), 2579–2594 (2019)
20. Li, Y., Shi, D., Ding, B., Liu, D.: Unsupervised feature learning for human activity recognition using smartphone sensors. In: Mining intelligence and knowledge exploration, pp. 99–107. Springer (2014)
21. Liu, J., Shahroudy, A., Xu, D., Kot, A.C., Wang, G.: Skeleton-based action recognition using spatio-temporal lstm network with trust gates. IEEE transactions on pattern analysis and machine intelligence **40**(12), 3007–3021 (2017)
22. Ordóñez, F.J., Roggen, D.: Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition. Sensors **16**(1), 115 (2016)

23. O'Shea, K., Nash, R.: An introduction to convolutional neural networks. arXiv preprint arXiv:1511.08458 (2015)
24. Poppe, R.: A survey on vision-based human action recognition. Image and vision computing **28**(6), 976–990 (2010)
25. Rougier, C., Meunier, J., St-Arnaud, A., Rousseau, J.: Robust video surveillance for fall detection based on human shape deformation. IEEE Transactions on circuits and systems for video Technology **21**(5), 611–622 (2011)
26. Singh, M.S., Pondenkandath, V., Zhou, B., Lukowicz, P., Liwickit, M.: Transforming sensor data to the image domain for deep learning—an application to footstep detection. In: 2017 International Joint Conference on Neural Networks (IJCNN), pp. 2665–2672. IEEE (2017)
27. Stevens, J.A., Corso, P.S., Finkelstein, E.A., Miller, T.R.: The costs of fatal and non-fatal falls among older adults. Injury prevention **12**(5), 290–295 (2006)
28. Uden, L., Pérez, J., Herrera, F., Rodrıguez, J.: Advances in intelligent systems and computing: Preface. Advances in Intelligent Systems and Computing **172** (2018)
29. Vepakomma, P., De, D., Das, S.K., Bhansali, S.: A-wristocracy: Deep learning on wrist-worn sensing for recognition of user complex activities. In: 2015 IEEE 12th International conference on wearable and implantable body sensor networks (BSN), pp. 1–6. IEEE (2015)
30. Wang, A., Chen, G., Shang, C., Zhang, M., Liu, L.: Human activity recognition in a smart home environment with stacked denoising autoencoders. In: International conference on web-age information management, pp. 29–40. Springer (2016)
31. Wang, J., Chen, Y., Hao, S., Peng, X., Hu, L.: Deep learning for sensor-based activity recognition: A survey. Pattern Recognition Letters **119**, 3–11 (2019)
32. Wang, J., Liu, Z., Wu, Y., Yuan, J.: Learning actionlet ensemble for 3d human action recognition. IEEE transactions on pattern analysis and machine intelligence **36**(5), 914–927 (2013)
33. Wang, P., Li, W., Gao, Z., Zhang, J., Tang, C., Ogunbona, P.O.: Action recognition from depth maps using deep convolutional neural networks. IEEE Transactions on Human-Machine Systems **46**(4), 498–509 (2015)
34. Weinland, D., Ronfard, R., Boyer, E.: A survey of vision-based methods for action representation, segmentation and recognition. Computer vision and image understanding **115**(2), 224–241 (2011)
35. Weng, J., Weng, C., Yuan, J., Liu, Z.: Discriminative spatio-temporal pattern discovery for 3d action recognition. IEEE Transactions on Circuits and Systems for Video Technology **29**(4), 1077–1089 (2018)
36. Yadav, S.K., Tiwari, K., Pandey, H.M., Akbar, S.A.: A review of multimodal human activity recognition with special emphasis on classification, applications, challenges and future directions. Knowledge-Based Systems p. 106970 (2021)
37. Yao, S., Hu, S., Zhao, Y., Zhang, A., Abdelzaher, T.: Deepsense: A unified deep learning framework for time-series mobile sensing data processing. In: Proceedings of the 26th International Conference on World Wide Web, pp. 351–360 (2017)
38. Yuan, J., Liu, Z., Wu, Y.: Discriminative subvolume search for efficient action detection. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition, pp. 2442–2449. IEEE (2009)
39. Zhang, J., Li, W., Ogunbona, P.O., Wang, P., Tang, C.: Rgb-d-based action recognition datasets: A survey. Pattern Recognition **60**, 86–105 (2016)
40. Zhang, P., Lan, C., Xing, J., Zeng, W., Xue, J., Zheng, N.: View adaptive recurrent neural networks for high performance human action recognition from skeleton data. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2117–2126 (2017)
41. Zhang, P., Xue, J., Lan, C., Zeng, W., Gao, Z., Zheng, N.: Adding attentiveness to the neurons in recurrent neural networks. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 135–151 (2018)
42. Zhang, X., Xu, C., Tian, X., Tao, D.: Graph edge convolutional neural networks for skeleton-based action recognition. IEEE transactions on neural networks and learning systems (2019)
43. Zhu, W., Lan, C., Xing, J., Zeng, W., Li, Y., Shen, L., Xie, X.: Co-occurrence feature learning for skeleton based action recognition using regularized deep lstm networks. arXiv preprint arXiv:1603.07772 (2016)