ITC 1/53
Information Technology and Control
Vol. 53 / No. 1 / 2024
pp. 53-70
DOI 10.5755/j01.itc.53.1.32625

Enhanced Two-Stream Bayesian Hyper Parameter Optimized 3D-CNN Inception-v3 Based Drop-ConvLSTM2D Deep Learning Model for Human Action Recognition

Received 2022/10/29 | Accepted after revision 2023/02/22

HOW TO CITE: Jeyanthi, A., Visumathi, J., Heltin Genitha, C. (2024). Enhanced Two-Stream Bayesian Hyper Parameter Optimized 3D-CNN Inception-v3 Based Drop-ConvLSTM2D Deep Learning model for Human Action Recognition. *Information Technology and Control*, 53(1), 53-70. https://doi.org/10.5755/j01.itc.53.1.32625

Enhanced Two-Stream Bayesian Hyper Parameter Optimized 3D-CNN Inception-v3 Based Drop-ConvLSTM2D Deep Learning Model for Human Action Recognition

A. Jeyanthi
Department of Computer Science and Engineering, Misrimal Navajee Munoth Jain Engineering College, Chennai, 600097, India

J. Visumathi
Department of Computer Science and Engineering, VelTech Rangarajan Dr. Sagunthala R&D Institute of Science and Technology, Chennai, India

C. Heltin Genitha
Department of Information Technology, St. Joseph's College of Engineering, Chennai, India

Corresponding author: jeyanthisuresh22@outlook.com

Human Action Recognition (HAR) has grown to be the toughest and attractive concern in the domains of computer vision, communication between a person and the surroundings, and video surveillance. In variation to the conventional methods that usually make use of the Long Short Term Memory model (LSTM) for training, this work designed dropout variant Drop-ConvLSTM2D, to provide more effectiveness in regularization for deep Convolution Neural Networks (CNNs). In addition, to speed up the runtime performance of the Deep Learning model, Bayesian

Hyper Parameter Optimization (BHPO) is also introduced to autonomously optimize, the hyper parameters of the trained architecture. In this study, a two-stream Bayesian Hyper Parameter optimized Drop-ConvLSTM2D model is designed for HAR to overcome the current research deficiencies. In one stream, an Inception-v3 model extracts the temporal characteristics from the optical frames which are generated through the dense flow process. In another stream, a 3D-CNN involves the mining of the spatial-temporal characteristics from the RGB frames. Finally, the features of Inception-v3 and 3D-CNN are fused using which the Drop-ConvLSTM2D model is trained to recognize human behavior. On perceptive public video datasets UCF-101, HMDB51, the quantitative assessments are conducted on the Drop-ConvLSTM2D BHPO model. For all hyper parameters, the built model explicitly obtains optimized values in this process, which can save time and improve performance. The experimental outcome shows that with a precision of at least 3%, the designed model beats the traditional two-stream model.

KEYWORDS: Human Action Recognition, Convolution Neural Networks, Drop-ConvLSTM2D, Bayesian Hyper Parameter Optimization.

1. Introduction

HAR by traditional hand-crafted feature extraction [21, 33, 34] was extremely difficult due to the existence of obstacles like varying object sizes in different frames, the existence of noise, and also the swiftness of activities. Thus, even though in recent times, several significant research studies have been implemented using Deep CNN [1, 23, 28, 39], the technique to precisely make out HAR from the RGB videos continues to be an exigent problem.

Nowadays deep learning has grown-up swiftly and has attracted many research efforts [7, 20, 24] in video analysis, natural language processing, and complicated aspects of data processing, to achieve unprecedented achievements. Deep learning may decrease the workload of feature design, unlike the conventional technique of HAR. In addition, through the end-to-end neural network, high-level and more complicated clues can be taught. Furthermore, for unsupervised incremental learning, the deep learning architecture will be suitable by stacking many layers of clues. Several researchers have sought deep learning in video-based behavior recognition, owing to its fare on extracting frame features [4, 14, 15, 19, 22].

In HAR, hybrid models with CNN and LSTM are taken for research by majority of researchers [1, 25]. LSTM concentrates on the pattern features and captures the dynamic time dependency of various movements. Furthermore, owing to the various parameters that necessitate an alteration in the course of the training period, the LSTM requires a longer training time. CNN is more capable of learning the key characteristics found in recursive patterns compared to LSTM [26]. However, in the convolution process, the mainstream of CNNs has a single parameter configuration, which radically restricts the model's flexibility. A larger convolution kernel can also be useful for gathering more data, although it raises the cost of CNN calculation. These challenges lead researchers to launch effective recognition strategies that can effectively solve these problems with expected recognition accuracy and low computational complexity.

We are provoked by the observation that, a dual-stream 3D-CNN [30] is always a suitable preference for constituting the Spatio-temporal clues of action videos. This study attempts to build and implement an innovative two-stream Bayesian Hyper Parameter optimized 3D-CNN Inception-based Drop-ConvLSTM2D model, considering the deficiencies identified in the present research. In one stream, an Inception-v3 model extracts the temporal characteristics over the optical frames which are generated using the dense flow process. In the remainder, a 3D-CNN mines spatial characteristics from the RGB frames. In the end, the feature maps of Inception- v3 and 3D-CNN are fused using which the Drop-ConvLSTM2D architecture is taught to understand human behavior.

2. Related Work

In this section, we discuss various previous works [35, 36] on HAR using dual-channel CNN. Karen et al. [26] crafted an architecture that uses separate Spatio-temporal recognition channels based on ConvNets capable of giving better performance even with less training data. The approach averages the pre-

dicted outcomes from RGB frames and a set of ten optical flow snapshots after running them via identical ConvNets tuned on ImageNet. This study is the basis for the majority of two-stream CNN designs. Here the flow stream has an adapted input Conv layer with twice as many input channels as flow frames because the flow has two channels, horizontal and vertical. Carreira et al. [11], introduced a very deep model codenamed "Two-Stream Inflated 3D ConvNets" (I3D) that proves the boost in outcome using transferability of features by pre-training on Kinetics dataset and fine-tuning to UCF-101[27]/HMDB-51 [13] dataset. They also redesigned many cutting-edge HAR architectures, to understand their transfer behavior and proved that the recognition rate exceeds all re-designed pre-trained and finely tuned models. In [40], authors discovered the notion of composing details over long-duration videos. The work introduced a feature pooling methodology to procedure each snapshot independent of other snapshots and used maxpool on local info for decoupling image-level features. They investigated the use of RNN using LSTM units those are merged to the final layers in the CNN.

In [36], the authors proved the procedure of batch normalization, dropout, and pre-training as acceptable methods. The work merged the temporal pooling network with LSTM in order for it to process snapshots of variable length for enhancing the two-stream model. They discovered a model to mine shorter-length snippets based on the video by using sparse samples rather than dense samples. The snippets are sent to spatial-temporal ConvNets. Finally, the outcomes of those ConvNets are merged for final prediction. In [41], authors applied a MotionNet that generates optical flow using successive snapshots. The outcome of the MotionNet is fused with a temporal channel CNN to correlate the optical flows to target classes. It also has a spatial channel CNN which is merged with the temporal channel CNN with late fusion.

In [38], the authors first created the optimized video by only taking into account the action regions with the highest levels of activity. The human skeleton and RGB clues were then mined using a two-stream CNN to reinforce the deep representation of humans for robust processing and expand the receptive field of feature extraction. In [10], the authors used the Kalman filter and the Gaussian mixture model to infer human motion from each video frame, which consists of a series of brief frames that only show a moving subject. To infer the pertinent features, these inputs are then sent into the hidden layer of the Gated Recurrent Neural Networks model. The action is then predicted using the training and testing phase by the prediction module.

The authors of [18] presented an adaptive sub-graph convolution module that can learn the relationships among sub-graphs and adaptively infer the high-level spatial characteristics of each sub-graph. They created a two-stream architecture to combine bone and joint characteristics, which improves the model's capacity for recognition. The channel attention mechanism and the spatial attention mechanism make up the Convolution Block Attention Module. The channel attention mechanism enables the model to focus on the channel characteristics with more information by teaching it how to change the weight parameters of each channel. By learning the weights of pixels in various spatial positions, the spatial attention mechanism improves the properties of significant positions.

The works mentioned above exploited dual-channel CNN streams to enhance the recognition rate of HAR. However, the majority of works apply two similar 3D CNNs for both the streams (temporal and spatial). Due to this, the efficient mixtures of features like RGB snapshot, optical flow, depth, and skeleton data, remain an unsolved issue in HAR.

In this paper, we investigate how the dual-stream CNN could enhance the recognition rate with different CNNs for each stream. The need for modeling our architecture is, that when two streams with identical CNNs are tuned and fused, it may yield a huge number of redundant features due to horizontal and vertical components of optical flow frames mined over the RGB frame.

In our experimentation, we apply the 3D-CNN model only to mine spatial characteristics from RGB frames and we apply the Inception-v3 model to mine motion characteristics from optical flow as we found Inception models produce better feature extraction and performance than other network models with our experiments [18].

The Inception models are the flavors of the CNN crafted by Google, particularly for image classification. An Inception model differs from conventional CNNs; as the inception models convolute the same input tensor with many filters and merge the outcomes, whereas conventional CNNs do a single convolution

operation on each tensor. The parameters are thus decreased and the complexity in terms of computation is reduced as compared to 3D-CNN.

In the proposed model, mined Spatio-temporal features are decoupled using Conv-fusion, and as a technical contribution of work, we apply the above clues to train an innovative Drop-ConvLSTM2D design to build more stable and effective regularized deep CNNs.

Finally, using BHPO, the trained model is optimized; which is another noteworthy hallmark that highlights our work from previous models where hyper-parameters are tuned using painstaking trial and error. In short, the overarching summary in this paper:

In short, the overarching summary in this paper:

1 In contrast to conventional two-stream models, this model uses two different CNN over two streams; a state-of-the-art 3D-CNN for spatial feature mining and an Inception-v3 model for temporal feature mining.
2 To reduce variations with respect to mean and variance, the decoupled features are taught using Drop-ConvLSTM2D.
3 A ConvNet architecture search is conducted across multiple dimensions using BHPO by training on the UCF-101, HMDB51 to get heterogeneous Deep Learning architecture.

3. Methodology of the Proposed Framework

This section describes the 3D-CNN architecture, the optical flow method and also describes the Drop-ConvLSTM2D, then a Bayesian optimization method.

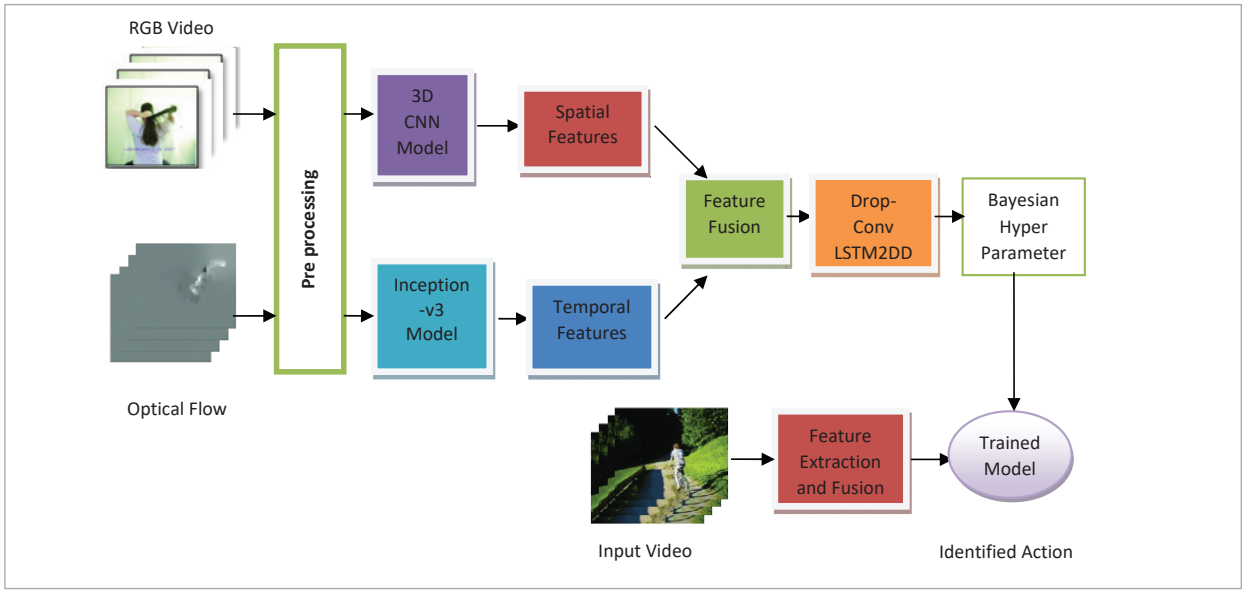
The discussed model is composed of five modules as shown in Figure 1. They are:

1 Video Data Preprocessing
2 Feature representation and mining
   Spatial feature mining
   Temporal feature mining
3 Spatial-Temporal Feature fusion
4 Training the Drop-ConvLSTM2D model with Bayesian Hyper Parameter optimization
5 Action Classification using a trained model

3.1. Data Preprocessing

Global Contrast Normalization (GCN) through zero component analysis (ZCA) whitening is applied to preprocess the input video which can preclude the frames from viewing different contrast levels. The average frame value is subtracted and the image is resized to maintain the standard deviation as a constant value across frames [39]. The whitening method

Figure 1
Proposed 3D-CNN Inception-v3 based Drop-ConvLSTM2D model



of Zero Component Analysis emphasizes facilitating the average covariance, which is maximal between the whitened pixel and the first frame. For example, eliminating neighboring correlations in neighboring pixels makes the data with a reduced amount of redundant information.

3.2. Feature Mining

3.2.1. Spatial Feature Extraction

The 3D-CNN architecture makes use of the pre-processed RGB frames as input for the extraction of spatial features.

3D CNN Architecture

Every given input map is transformed by a shift window with *(N,N)* kernel to produce an unique pixel in an individual output feature map. In addition, the 3D convolution layers are exploited to acquire movement information out from aggregate stacked frames. Equations (1)-(2) define the Nth 3D feature map pertaining to the i-th convolution layer.

$$V_i^N = \delta(W_i^N * x + b_i^N), \quad (1)$$

$$V_j^N = \delta(W_j^N v_i + b_j^N), \quad (2)$$

where, W stands for the weights of filter, x stands for the input frame, δ is the Convolution operation and b is the bias.

The pooling layer is taken into account in reducing the size of feature map and parameters. There are two widely accepted methods of pooling: maximum and average. The max-pooling operation is deployed in this study, since by offering an abstracted form of the clues, max-pooling aids in reducing over-fitting. Additionally, it lowers the computational cost by lowering the number of parameters to learn. It is determined by the utmost value achieved between nearby inputs as represented in Equation (3).

$$P_i^j = \max_{r \in R}(c_{i \times T+r}^j), \quad (3)$$

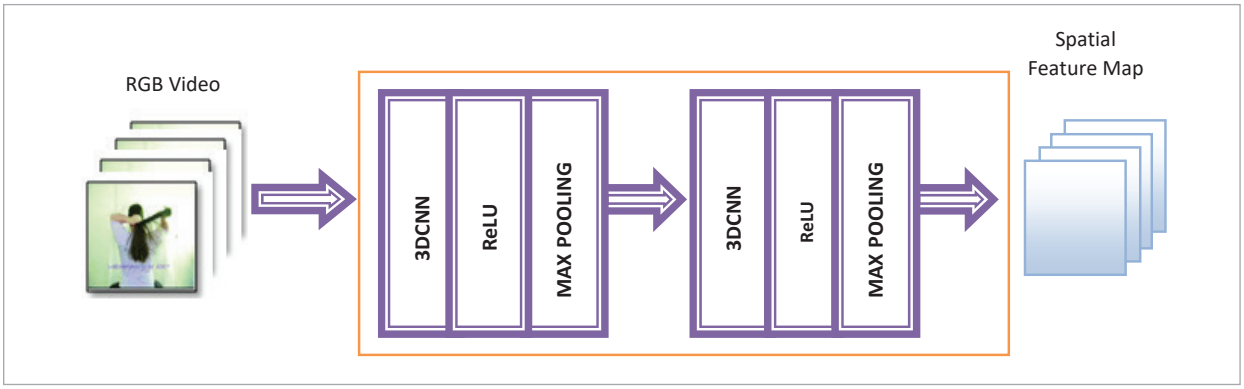
where, R describes the size of pooling, T denotes the pooling stride, and C is output of convolution layer.

This sparse connection network can remarkably diminish the parameter size by stacking convolution and the pooling layers in mining the significant clues of the input. The Restricted Linear Unit (ReLU) [9] is used to improve the transformation in terms of non-linearity in the network.

With Conv3D and MaxPooling 3D layers, the Keras Sequential API is employed. In particular, two 3DCNN layers of (3,3,3) kernel each with 8,16 filters are utilized. ReLU activation functions are added to the uniform Keras initializer. A three-dimensional max-pooling layer of (2,2,2) pool sizes is deployed to down-sample the feature maps, which can save valuable computational resources. The stride and padding are sizes of (1,1,1).

In this layout, as seen in Figure 2, the 30 frames considered by the 3D-CNN model are of size '112x112' centered on the present frame. A group of hardwired kernels is initially utilized to build larger information channels from the input frames. 64-sample batch size is adopted, which can make every time 64 samples are

Figure 2
3D-CNN Architecture



fed forward through the model, creating predictions, optimization, and computation of loss.

By maintaining the spatial clues of the frames, the 3D-CNN method can extract the temporal characteristics which can be employed in action recognition. Since the preponderance of the actions are likely to have 32-50 frames per gesture, this 3D-CNN model may not be feasible for a thorough analysis. This necessitates for another network to acquire long-term temporal individuality that is essential. The mixture of the Optical flow Inception-based 3D-CNN algorithm with the Drop-ConvLSTM2D network was suggested to help to understand the long-time temporal features.

A cross-entropy loss function has been used by CNN model training which is quantified by Equation (4),

$$E_0(x_m, y_m) = \frac{1}{M}[\sum_{i=1}^{M}\sum_{k=1}^{N_c} y_{m,k} \log(q_{m,k})], \quad (4)$$

where x_m represents the training set, $q_{m,k}$ is the recognized label of the m^{th} sample k^{th} data, $y_{m,k}$ is a one-hot vector representing the category of the m^{th} sample k^{th} data, M is the sample size, and Nc is the count of output category labels.

3.2.2. Temporal Feature Extraction

Optical flow is a pertinent computer vision technique in estimating motion, tracking objects, and identifying actions. The distribution of the apparent swiftness of movement of the brightness model is called optical flow of an image. The key techniques of the optical flow approach are dense and sparse optical flow. A stream vector is utilized in dense optical flow, while each feature in the stream, e.g., the edges or corners of an object, is employed as sparse vectors in sparse. As a result, dense optical flow has a higher precision with higher discriminative power than sparse optical flow, but it has a higher computational cost. For each pixel in an optical flow, an optical frame vector is measured [6].

This work calculates the optical stream vector of human actions, $\mu = (\mu_L, \mu_H)$ at each frame using Equation (5) as:

$$I_L\mu_L + I_H\mu_H + I_t = 0, \quad (5)$$

where $I_L = \partial I/\partial x$, $I_H = \partial I/\partial y$, $I_t = \partial I/\partial t$, $\mu_L = dx/dt$, and $\mu_H = dy/dt$, where (x, y, t) is the frame in pixel (x,y) at time t, where $I(x,y,t)$ is the intensity in pixel (x,y) at time t, and μ_L, μ_H are the horizontal and vertical velocities in pixel (x, y).

Local optical flow vector O_T is calculated using equation (6) as:

$$\begin{aligned} O_T &= [O_L, O_H] \\ O_L &= [O_{L_{1,1}},...,O_{L_{1,18}},...,O_{L_{4,1}},...,O_{L_{4,18}}] \\ O_H &= [O_{H_{1,1}},...,O_{H_{1,18}},...,O_{H_{4,1}},...,O_{H_{4,18}}], \end{aligned} \quad (6)$$

1where, O_L,O_H is the optical flow sum in longitudinal and the optical flow info sum in transverse in 36 subareas respectively. An algorithm to compute dense optical flow is provided using Open-cv for all the points in the frame based on Gunner Farneback's two-frame motion estimate algorithm [6].

3.2.3. Inception-v3 Deep Learning Model

For temporal feature extraction, Inception-v3 [29] is employed on optical flow input. Inception-v3 is a large GoogleLeNet network. Inception-v3 is designed as an initial GoogLeNet model that combines several convolutionary filters of various sizes in a new filter.

The parameters are thus decreased and the complexity in terms of computation is reduced as compared to 3D-CNN. Figure 3 shows the fundamental architecture of Inception-v3. The various filter sizes help the model to generalize the objects of different sizes in that it offers a range of focus levels. It is noteworthy that the 1x1 convolutions are featured with '3x3' and '5x5' twigs to diminish the computation expensive also they reduce the tensor size by reducing the 3rd dimension in a 3D tensor.

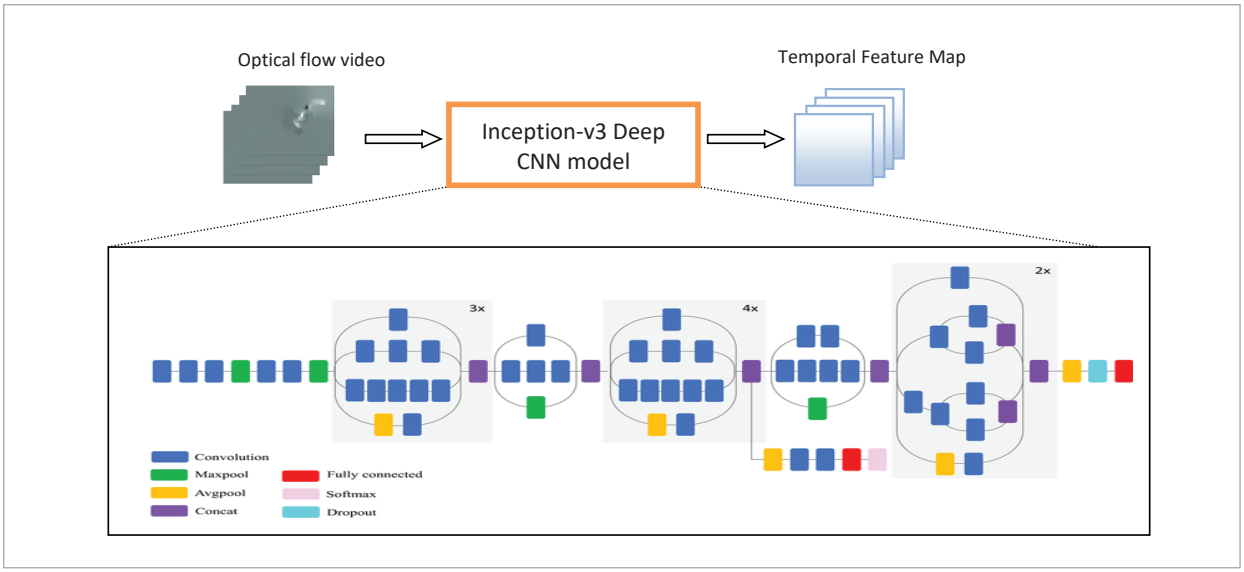
3.3. Spatial Temporal Feature Fusion

A fusion function [7] chords two feature maps and yields a fused map, where the fusion function is in Equation (7),

$$\begin{aligned} &f: X^a, X^b \to O \\ &X^a \in R^{H\times W\times D} \\ &X^b \in R^{H\times W\times D} \\ &O \in R^{H^H\times W^H\times D^H}, \end{aligned} \quad (7)$$

where f can be fed at various layers in the network when it is employed for feedforward ConvNet archi-

Figure 3
Temporal feature mining utilizing Inception-v3 (compressed model) model



tectures that consist of convolutional, pooling, fully connected, and nonlinear layers. Furthermore f is used for numerous types of fusion functions such as Sum-fusion, Bilinear-fusion, Concatenation-fusion, Max-fusion, and Conv-fusion. Amongst these different kinds of fusion functions, this model uses Conv-fusion because of its superior performance than others sufficing entirely for HAR.

Conv-fusion: Initially, X^a and X^b are stacked at the points i,j followed by that the stacked data are united with a filter bank $B \in R^{1\times1\times2D\times D}$ and biases $b \in R^D$ as depicted in Equation (8),

$$y^{conv} = y^{cat} * B + b. \tag{8}$$

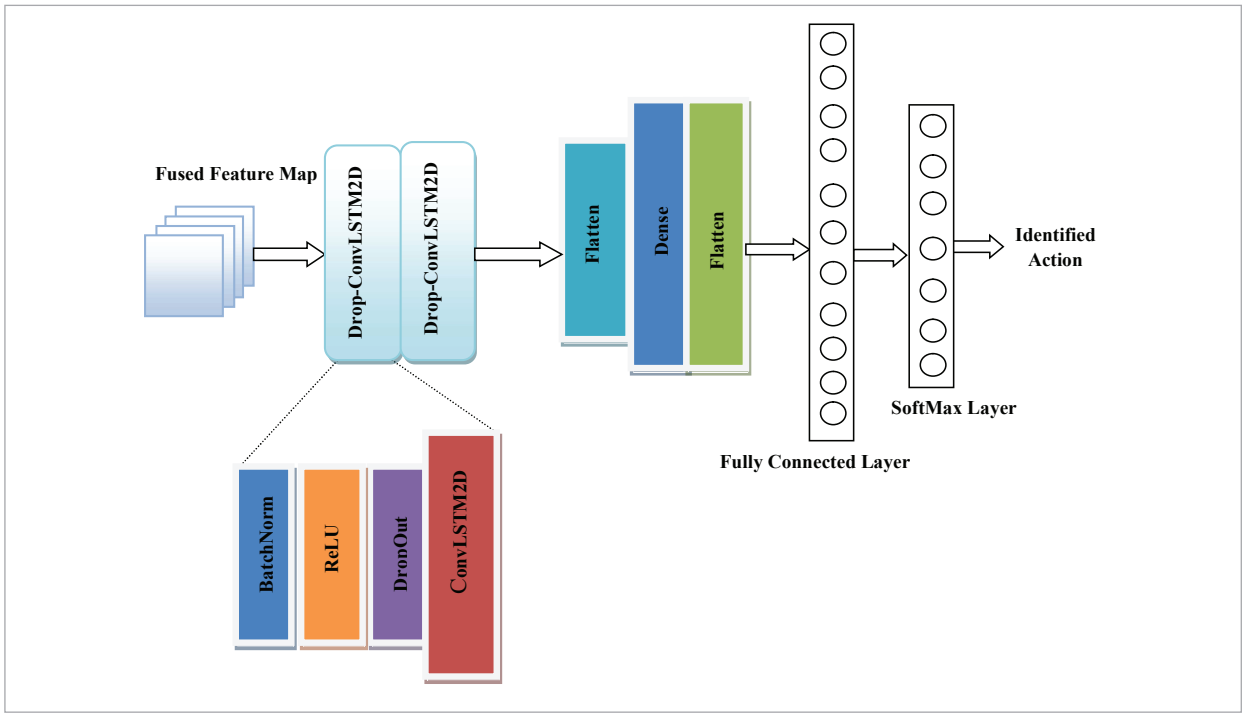
The filter B is used with a desire to drop off the dimension by the multiples of two and it can represent additive weighting of X^a, X^b especially at the same spatial position.

3.4. Drop-ConvLSTM2D Model

Aggregated features mined with the 3D-CNN and Inception-v3 architecture are then fed into the two stacks of Drop-ConvLSTM2D with 16 sizes of the unit, as depicted in Figure 4. A dropout variant Drop-Conv2D, developed by Cai et al. [2], could result in more stable and effective regularization for deep CNNs. Motivated by that, an innovative structurally more suited dropout variant Drop-ConvLSTM2D is implemented. Dropout operations are conventionally performed right between the convolution and BN layers which results in violent variations in terms of mean and variance obtained by the layer of BN. By locating the dropout operations before every Conv layer, as shown in the graphical overview Figure 4, the failure of conventional drop-neuron and drop-channel has been overcome. When you apply the convolution operation after a drop operation, you get a tiny variance and thus quick convergence. Drop-ConvLSTM2D enhancement leverages greater model capacity, better drop-out training, and regularization for enhanced training. More significantly, these upgrades incur negligible costs.

Generally, the fully-connected LSTM is fed with vectorized features to explore the temporal features. This yields the paucity of spatial similarity information during learning. Therefore, as an innovative approach Drop-ConvLSTM2D is deployed in the proposed DNN to understand the long-term spatiotemporal features. Formally, for the inputs X_1, ...,X_t, the cell states C_1, ...,C_t, the hidden states H_1, ...,H_t and the input gates i_t, forget gates f_t, output gates o_t, the key equations of ConvLSTM [25] are shown below, where * denotes the convolution operator and ◦ the Hadamard product:

Figure 4
Drop-ConvLSTM2d model for video action classification



$$i_t = \sigma(W_{xi} * X_t + W_{hi} * H_{t-1} + b_i), \quad (9)$$

$$f_t = \sigma(W_{xf} * X_t + W_{hf} * H_{t-1} + b_f), \quad (10)$$

$$o_t = \sigma(W_{xo} * X_t + W_{ho} * H_{t-1} + b_o), \quad (11)$$

$$C_t = f_t \circ C_{t-1} + i_t \circ tanh(W_{xc} * X_t + W_{hc} * H_{t-1} + b_c), \quad (12)$$

$$H_t = o_t \circ tanh(C_t). \quad (13)$$

Drop-ConvLSTM2D has been built using convolution kernel size (3,3) by associating through the stride (1,1) as given in Figure 4.

The training phase of a DNN model is complicated by a change in the density function of each layer's inputs with the parameter values of the preceding layer. To be able to prevent the variation in the distribution of output data, a lower learning rate is used, which maximizes training time. For resolving this issue, Batch Normalization (BN) is applied to normalize the LSTM layers' values to ensure that the mean and variance of the total do not vary as the underlying parameters are distributed, and also to isolate each layer's parameters from the other layers effectively.

Given the hidden layer inputs of the network as $x_1, \ldots, x_k$, the mean value μ_x and variance ς is computed using (14),(15).

$$\mu_x \leftarrow \frac{1}{k}\sum_{i=1}^{k} x_i, \quad (14)$$

$$\varsigma \leftarrow (1/k)\sum_{i=1}^{k}(x_i - \mu_x)^2. \quad (15)$$

Next, each dimension is normalized to x,

$$\hat{x}_i \leftarrow (x_i - \mu_x / \sqrt{\varsigma + \partial}). \quad (16)$$

Parameters γ and β are added at each activation to reconstruct the value by scaling and sliding the normalized value as follows:

$$O_i \leftarrow \beta\hat{x}_i + \eta = BN_{\beta,\eta}(x_i). \quad (17)$$

Parameters β and η are learned throughout training to improve back propagation.

Dropout layer is adapted to every section of Drop-ConvLSTM2D with a value of 0.2 and then determined the output probability using the Restricted Linear Unit (ReLU) function to strengthen the DNN for non-linear transformation.

Two stacks of BN, ReLU, Dropout, and ConvLSTM2D layers are constituted to understand the fused spatial-temporal features of video input. A 'Flatten' layer is added between the convolution layer and the FC layer to convert a two-dimensional matrix of features into a vector that is supplied into a classifier.

The last layer of the architecture requires a softmax standardized exponential function for forecasting the commensurate probability distribution of performing various activities using the classifier. It converts neuron output values between 0 and 1, which is known to predict behavior likelihood, and the greatest one is the outcome of categorization.

A number of neurons in softmax layer are equal to the class count. The approach is described as,

$$q_k = \frac{\exp(p)}{\sum_{k=1}^{N_c} \exp(p)}, \tag{18}$$

where, c is the activity class and N_c is the count of activity classes.

4. Bayesian Parameter Optimization

In the DNN, the parameters like learning rate, number of neurons for the various layers, the dropout, scale of the mini-batch, etc. unleash a recognition rate. Those parameters are called hyper parameters. Choosing the best hyper parameters [32] in the training phase is an optimization problem. The hyper parameter optimization is formulated using

$$X^* = {}_{x \epsilon X} min\, f(x), \tag{19}$$

where, x is the set of hyper parameters, X is the range of values of x, $f(x)$ is the objective function to diminish the error value estimated with the testing data.

In this proposed work Bayesian optimization [32,37] is exploited for tuning hyper parameters like learning rate, the number of filters, the drop-out layer, and the batch size in the network in Drop-ConvLSTM2D.

The theory of Bayesian optimization is Bayes' theorem; according to which, given the observations $D_{1:k}$, the posterior distribution of a model A, $P(A|D_{1:k})$ is proportional to the likelihood of $D_{1:k}$ given A, multiplied by the prior probability of A

$$P(A|D_{1:k}) = P(D_{1:k}|A)P(A), \tag{20}$$

Bayesian optimization picks out the smallest amount of a function, f(y), on a bounded set, Y. Bayesian optimization is exploited to investigate the perfect measure of sample values where the function appears to be small, as shown in Figure 5. With more trials for function f(y), the DNN model is more likely to make better decisions.

This research achieved the optimum network architecture by reducing the validation error that was identified with Bayesian optimization [2] as exposed in Figure 6.

Figure 5
Framework of Bayesian optimization for DNN model

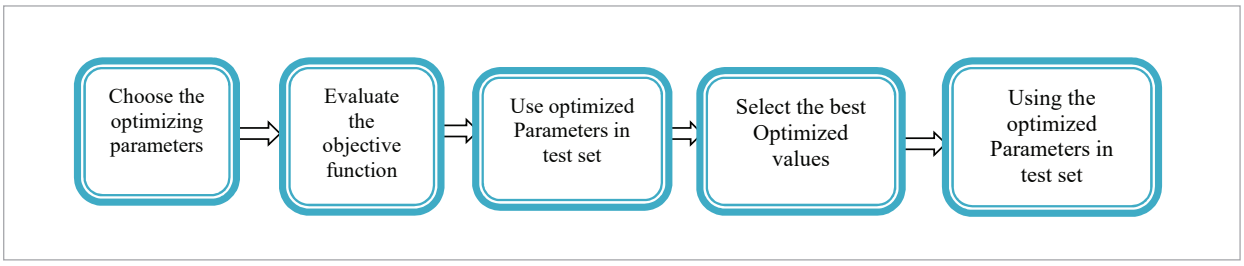


Figure 6
Recognition accuracy Vs Number of Filters

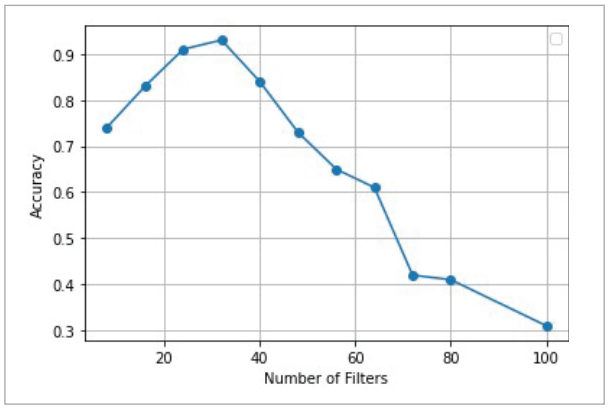


Figure 7
Recognition accuracy Vs Learning Rate

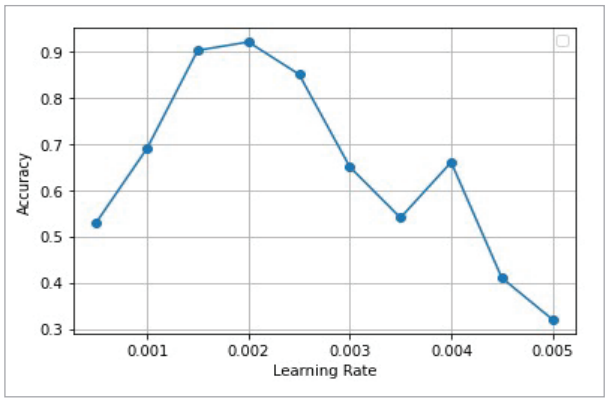


Figure 8
Recognition accuracy Vs Batch Size

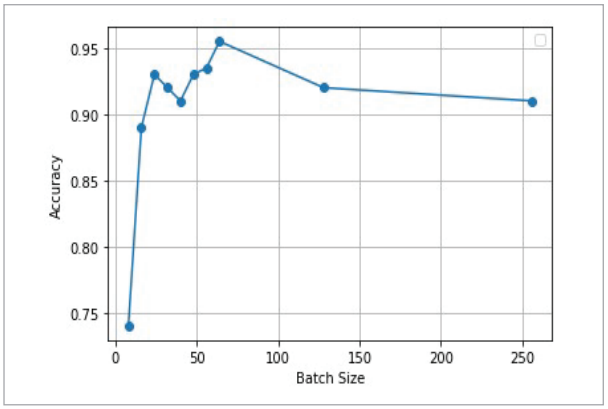
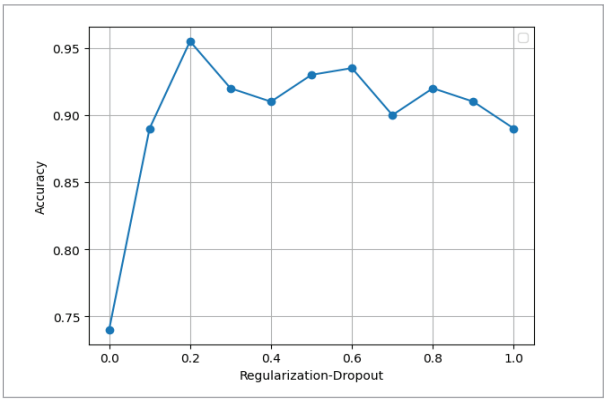


Figure 9
Recognition accuracy Vs Dropout



4.1. Parameters Setting

4.1.1. Number of Filters in Drop-ConvLSTM2D Layer

The analyses have been carried out, to validate the control of the filter count in the Drop-ConvLSTM2D model on the recognition accuracy as shown in Figure 6. It exhibits that the accuracy will be at the minimum when each Drop-ConvLSTM2D contains only eight filters. The network lacks the capacity for learning and data processing due to the fewer filters, which yields a low-performance rate. The recognition rate improves with the filter count. As the filter count increases, the layered architecture will become more complicated, and the model's training fastness would have been hampered. When the filter count is 32, the recognition accuracy reaches 95.8%; hence the filter count 32 is chosen in this architecture.

4.1.2. Learning Rates

The learning rate is the most vital hyper parameter in designing the deep architecture. It decides the learning capacity of the deep learning network to understand the problem. It determines the amount of weight by which the network should be updated in the course of training. If the learning rate is high the result oscillates in training iterations; else if it is too low, the network may be trapped in a suboptimal goal state. The design is probed at different learning rates as showed in Figure 7. As the learning rate of 0.002 is leveraging the model's recognition rate, a learning rate of 0.002 is adapted in this model.

4.1.3. Batch Size

The batch size is the number of samples passed to the model for updating the network parameters which

determines how accurately the error gradient [16] can be estimated in deep learning architecture. Within a suitable range, increasing the size of the batch can determine the gradient descent direction more accurately and cause less training fluctuation. Recognition effects are seen in Figure 8. The average identification rate is 95.8% for batches of 64 observations. Consequently, 64 is recommended as the appropriate batch in this review.

4.1.4. Regularization

Drop out is the methodology for regularization that enables a DNN to study the most discriminative features which are needed with relevance to various subsets of the other neural units. Drop out must have an optimal value, as the loss function will not be affected if it is zero, and if it is high, it will remove the potency of the neural units in a layer that may lead to poor training. The Drop out of 0.02 is adopted in this design since the recognition rate is high at this value as depicted in Figure 9. Table 1 shows the hyper parameters of the design, Table 2 shows Hyper parameter values throughout the training phase and Table 3 shows the best observed optimal points, respectively.

Table 1
The choosen hyper parameters

Hyper parameters	Initial Value	Final Value
Learning Rate	0	1
Number of filterss in Drop-ConvLSTM2D	10	50
Batch Size	16	512
Regularization-Dropout	0	1

Table 2
Hyper parameter values during training

I	Learning Rate	Number of filters in Drop-ConvLSTM2D	Batch Size	Regularization Dropout
1	0	32	512	0.4
2	0.05	128	128	0.3
3	0.01	64	64	0.5
4	0.1	256	256	0.8
5	0.002	32	36	0.2

Table 3
Best observed feasible points

I	Learning Rate	Number of filters in Drop-ConvLSTM2D	Batch Size	Regularization -Dropout
1	0.002	32	64	0.2

The precision of the model before tuning of hyper parameters is 87.2% and after 100 iterations the accuracy of the model hits 95.8%. The experimental parameters of the designed model are depicted in Table 4.

Table 4
Experimental parameters of Drop-ConvLSTM2D on UCF-101

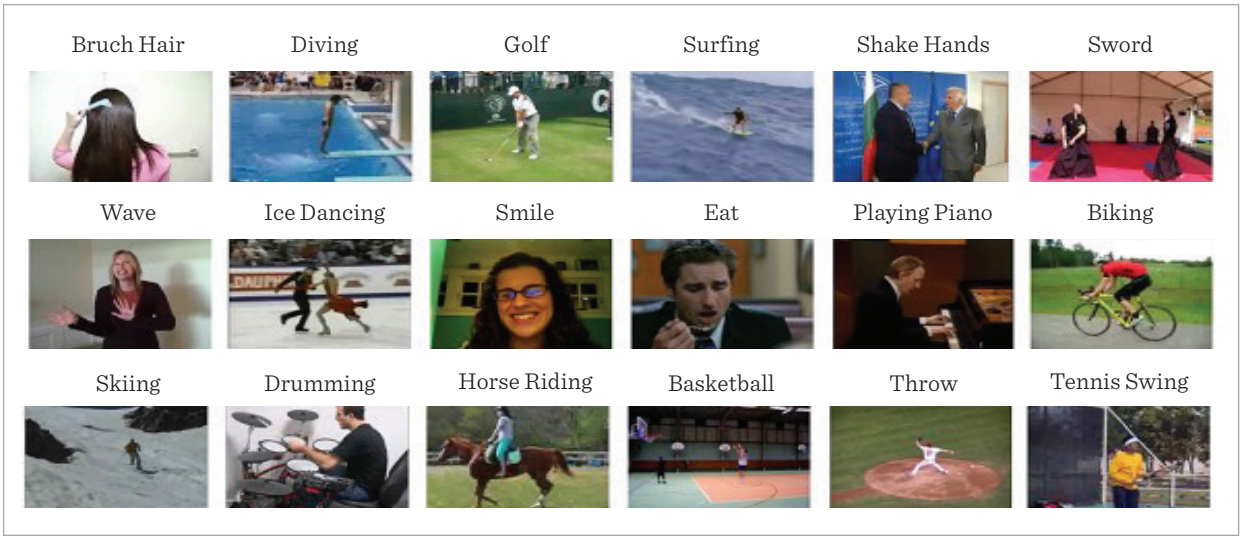
Parameters	Value
Input Vector Size	112×112×30
Convolution Kernal size	Conv3D -3×3×3 ConvLSTM2D-1×3
Number of Filter	Conv3D-8,16 ConvLSTM2D-32,32
Pool size	3×3
Activation Function	ReLu
Drop-ConvLSTM Layer	2
Dropout	0.2
Learning Rate	0.002
Batch size	64
Epoch	100

5. Experimental Information on the UCF-101 and HMDB51 Dataset

5.1. Datasets

The UCF-101 and HMDB51 are two datasets used for evaluating the experiment, which is very common in the field of HAR research. There are 13320 videos in UCF101 from 101 YouTube video collections. There are 25 classes of between 4-7 videos per category. The categories of behavior may consist of five types: Human-Object Interactions, Body-Motion Only, Human-Human Interaction, Playing Musical Instruments, and Sports. HMDB51 includes 6766 videos, categorized by 51, including five types of behavior:

Figure 10
Sample action categories of UCF-101 and HMDB51 Data sets



gestures of the face, facial actions with object interactions, movements of the body, and movements of the body by touching the objects. 9.5K training videos are included in each UCF-101 split; 3.7K training videos are included in the HMDB51 split. With '320x240' spatial resolution and 30fps frame rate, HMDB51 has fewer groups and images than UCF-101. In this study "Google Colab" is utilized to execute the programs. It supports nearly 25 GB of RAM and variable GPU based on the network traffic. Sample action categories of UCF-101 and HMDB51 data sets are depicted in Figure10.

5.2. Performance Metrics Obtained for Proposed Model

We benchmarked our model using accuracy, precision, recall, F1-score.

$$Accuracy = TP+TN/(TP+FP+TN+FN), \quad (20)$$

$$Precision = TP/(TP+FP), \quad (21)$$

$$Recall = TP/(TP+FN), \quad (22)$$

$$F1\ Score=2*(Recall*Precision)/(Recall+Precision) \quad (23)$$

The Precision, Recall, and F1-score of the proposed network for the UCF-101, HMDB51 data set on 12 classes are included in Tables 5-6.

Table 5
Performance metrics for proposed model on UCF-101 data set

Action	Precision	Recall	F1-score
Apply_EyeMakeup	0.99	0.97	0.98
Apply_Lipstick	0.87	0.87	0.87
Archery	0.95	0.96	0.95
Baby_Crawling	0.86	0.95	0.90
Balance_Beam	0.95	0.96	0.95
Band_Marching	0.92	0.95	0.93
Base_BallPitch	0.89	0.90	0.89
BasketBall	0.98	0.99	0.99
BasketBall_Dunk	0.97	0.95	0.96
Bench_Press	0.96	0.91	0.94
Biking	0.97	0.93	0.95
Billiards	0.99	0.92	0.95

Table 6
Performance metrics for proposed model on HMDB51 data set

Action	Precision	Recall	F1-score
Brush_hair	0.85	0.79	0.82
Cartwheel	0.80	0.87	0.83
Chew	0.81	0.86	0.84
Clim_stairs	0.84	0.84	0.84
Draw_sword	0.92	0.85	0.88
Catch	0.80	0.84	0.82
Clap	0.82	0.83	0.83
Climb	0.82	0.83	0.83
Dive	0.94	0.86	0.90
Dribble	0.84	0.78	0.81
Drink	0.74	0.85	0.79
Eat	0.73	0.77	0.75

5.3. Confusion Matrix of Proposed Model: UCF-101 and HMDB51 Data Set

The recognition rate of the DL architecture is monitored with the help of the confusion matrix which exhibits how to update parameters of the deep learning model designed by identifying errors in training process. Figures 11, Figure 12 project the confusion matrices for 12 classes in the UCF-101 dataset and HMDB51 dataset.

Figure 11
Confusion matrix on UCF- 101 for 12 classes

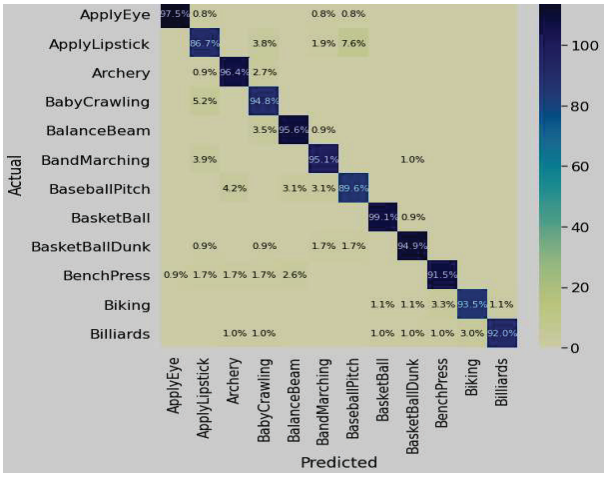
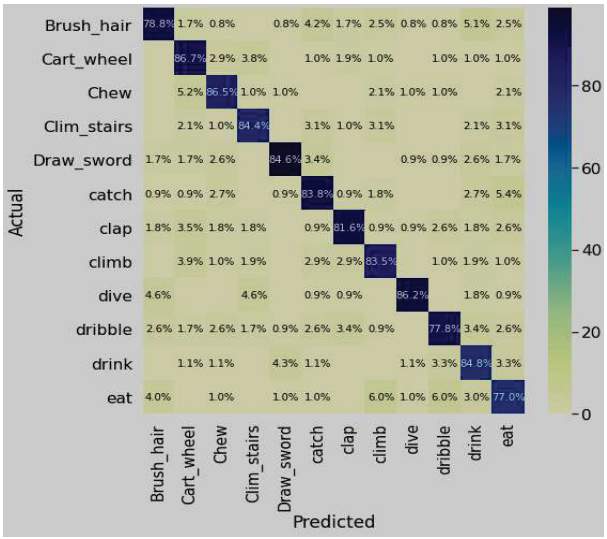


Figure 12
Confusion matrix on HMDB51 for 12 classes



5.4. Recognition Accuracy (%) of Proposed Model in Assessment with State-of-the-Art Models on UCF- 101 Data Set and HMDB51

The proposed model's classification accuracy is compared to the front running models with respect to average accuracy as exposed in Table 7.

Here, the top 1 accuracy is accounted in, as most of the models in this comparison do not specify the top 5 accuracy. As shown in this table, the two-stream Drop-convLSTM2D gets the highest top 1 recognition accuracy among all methods, which is 95.8% on UCI-101 and 70.5% on HMDB-51. Compared with the two-stream VGG model, the designed model exceeds by 3.3% on UCI-101 and 5% on HMDB-51. The handcrafted model (IDT) achieves 85.9% accuracy, which is 10% less than our proposed model on UCI-101 and 13% on HMDB-51. This work has achieved comparable results with other front running models [3, 17, 31] of HAR. Benefitting from the advanced temporal stream, the proposed model can also have higher recognition accuracy than most of these methods. In summary, the proposed model accomplishes higher recognition accuracies in both the spatial stream and the temporal stream than the traditional two-stream CNN model and the other state-of-the art approaches.

Table 7
Comparing Proposed Model Test Results on UCF-101 and HMDB51 data sets

	Method	UCF-101	HMDB51
Handcrafted	[34] Improved Dense Trajectories	85.9%	57.2%
2D CNN	[12] Slow-Fusion	65.4%	-
3D CNN	[30] Res3D (fine tuned)	85.8%	54.9%
Multi-stream 2D CNN	[5] LRCN (RGB) [5] LRCN (Flow) [5] LRCN (fusion)	68.19% 77.46% 82.66%	- - -
	[26] Spatial Stream [26] Temporal Stream [26] Two-Stream (avg) [26] Two-stream (SVM) [7] Two-Stream VGG	73% 83.7% 86.9% 88.0% 92.5%	40.5% 54.6% 58.0% 59.4% 65.4%
Temporal segment network	[31] Spatio-Temporal VLAD	95.6%	71.4%
Two stream	[3] Spatio-Temporal Heterogeneous two streaam network	93.3%	65.9%
TSCN	[17] Temporal segment connection network	94.2%	70.3%
Proposed method	TwoStreamDrop-ConvLSTM2D	95.8%	70.5%

The overarching summary of the observations is

1 Only spatial features can be extracted by 3D-CNN.
2 Introducing Inception to mine temporal features from optical flow improves the recognition rate.
3 A heterogeneous Drop-ConvLSTM2D can reduce gradient variance and thus faster convergence.

5.5. Loss and Accuracy During Training and Testing

The loss aids in the optimization of the deep learning algorithm, while the accuracy aids in the assessment of the algorithm's efficiency.

The graph Figure 13 exhibits the training and validation accuracy on the UCI 101 data set for every 20 epochs up to 100 epochs. Both the graphs increase over time, specifically the training accuracy increases gently whereas validation accuracy rises with ripples over time.

The graph Figure 14 exhibits the training and validation loss on the UCI 101 data set for every 20 epochs up to 100 epochs. The training loss consistently declines with the iterations, but the validation loss increases and falls throughout the procedure.

Figure 13
Training accuracy Vs Validation accuracy on UCI-101

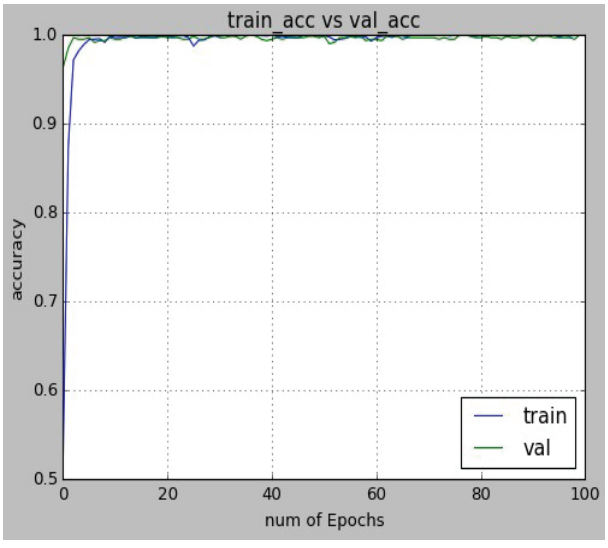
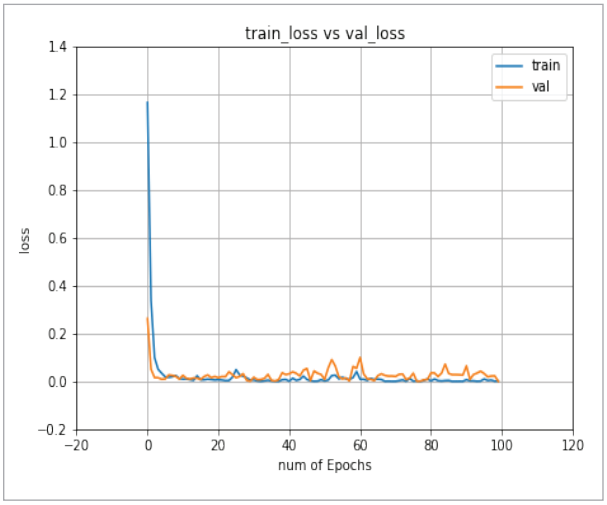


Figure 14
Training loss Vs Validation loss on UCI-101



5.6. Runtime Analysis on UCI-101

In order to show the persuasiveness of the proposed work, the run time of the work is compared with the run time of various CNN configurations. Table 8 represents the evaluation of the processing time of the proposed work and various CNN configurations for training with 200 epochs and classifying the video. This procedure was repeated ten times and the average processing time for each best trained model is presented in Table 8. All the experiments were carried out on Google Colab which can provide 25GB of internal RAM with GPU-Tensor through a Cloud environment.

The architecture of 3D-CNN used for experimentation is as follows: Two Conv3D and MaxPooling 3D layers, of (3,3,3) kernel each with 8, 16 filters are utilized. ReLU activation functions are added to the uniform Keras initializer. A three-dimensional max-pooling layer of (2,2,2) pool sizes is deployed to down-sample the feature maps, which can save valuable computational resources. The stride and padding are sizes of (1,1,1).

For the second configuration, in addition to the above 3D-CNN architecture, the LSTM with 100 units followed by Drop out and Dense layer is used for experimentation. As spotted in Table 8, the proposed work is faster than other methods in terms of training and classification.

Table 8
Runtime Analyze on UCI-101

Model	Training Time (minutes)	Classifying Time (seconds)
3DCNN	17	15
3DCNN+LSTM	29	30
Two-Stream network [26]	12	23
Proposed model	11	10

5.7. Ablation Study of Dropout Layer

We conducted experiments using a proposed model with and without the Dropout layer on a UCI-101 data set. As in Table 9, the presence of Dropout improves the recognition rate due to its job of leveraging greater model capacity and regularization.

Table 9
Recognition rate with and without dropout layer on UCI-101

Model	Recognition rate
Proposed model with ConvLSTM2D	86.1%
Proposed model with Drop-ConvLSTM2D	95.8%

6. Conclusion and Future Improvements

The aim of this work is to construct a hybrid two-stream deep learning architecture for HAR. In the designed architecture, a novel Drop-ConvLSTM2D model is developed in two stream 3D-CNN Inception-v3 based network. In the first stream, 3D-CNN mines spatial-temporal features through RGB frames. In the other stream, Inception-v3 extracts temporal features from Optical flow images that are obtained using dense optical flow [33]. The obtained spatiotemporal features are fused using Conv-fusion to train the Drop-ConvLSTM2D model. Finally, the Drop-ConvLSTM2D model is optimized using Bayesian Hyper Parameter Optimization. An examination of UCF101 and HMDB51 datasets reveals that the

original two steam networks were outperformed by the integrated model. The outcomes obtained illustrate the benefit of integrating the flow stream with the RGB stream using two different CNNs. Furthermore, to the to the fullest of our discernment, the novel Drop-ConvLstm2D model is the heterogeneous combination of 3D-CNN over RGB frames and the Inception-v3 over optical flow frames to acknowledge human activity. Finally, the result of the experiments emphasizes that, in identifying and classifying human behaviors from videos, the suggested model achieved substantial benefits. Although with this approach, we exhibit encouraging outcomes on both UCF-101 and HMDB51 datasets, the performance is quite far from precise. The proposed model has more time needs; and there is still scope for change. Future research will focus on more efficient methods to reduce the time complexity, so that the model would be applied for real-time human action prediction.

Data Availability

The video datasets UCF-101, HMDB51 used to support the findings of this study are included in the article in reference section [27], [13] in page no 18.

UCF-101 data set is available at https://www.crcv.ucf.edu/data/UCF101.php

HMDB51 data set is available at https://serre-lab.clps.brown.edu/resource/hmdb-a-large-human-motion-database/#Downloads

Disclosure

The research neither received any funding nor is performed as part of the employment.

Conflicts of Interest

We declare that there are no conflicts of interest.

References

1. Arif, S., Wang, J., U. I., Hassan, T., Fei, Z. 3D CNN Based Used Feature Maps with LSTM Applied to Action Recognition. Future Internet, 2019, 11(2), 42. https://doi.org/10.3390/fi11020042
2. Cai, S., Shu, Y., Chen, G., Ooi, B. C., Wang, W., Zhang, M. Effective and Efficient Dropout or Deep CNN, 2019. arXiv preprint arXiv:1904.03392
3. Chen, E., Bai, X., Gao, L., Tinega, H. C., Ding, Y. A Spatiotemporal Heterogeneous Two-Stream Network for Action Recognition. IEEE Access, 2019, 7, 57267-75. https://doi.org/10.1109/ACCESS.2019.2910604
4. Dollar, P., Rabaud, V., Cottrell, G., Belongie, S. Behavior Recognition via Sparse Spatio-Temporal Features. IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance, 2019, 65-72. https://doi.org/10.1109/VSPETS.2005.1570899
5. Donahue, J., Anne Hendricks, L., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., Darrell, T. Long-term Recurrent Convolutional Networks for Visual Recognition and Description. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, 2625-2634. https://doi.org/10.1109/CVPR.2015.7298878
6. Farnebäck, G. Two-Frame Motion Estimation Based on Polynomial Expansion. Scandinavian Conference on Image Analysis, 2003, 363-370. https://doi.org/10.1007/3-540-45103-X_50
7. Feichtenhofer, C., Pinz, A., Zisserman, A. Convolutional Two-Stream Network Fusion for Video Action Recognition. Proceedings of the IEEE Conference on Computer Vision Pattern Recognition, 2016, 1933-1941. https://doi.org/10.1109/CVPR.2016.213
8. Hammerla, N. Y., Halloran, S., Plötz, T. Deep Convolutional and Recurrent Models for Human Activity Recognition Using Wearable, 2016. arXiv preprint arXiv:1604.08880
9. Hinton, G. E., Nair, V. Rectified Linear Units Improve Restricted Boltzmann Machines. Proceedings of the 27th International Conference on Machine Learning (ICML-10) 2010, 807-814.
10. Jaouedi, N., Boujnah, N., Bouhlel, M. S. A New Hybrid Deep Learning Model for Human Action Recognition. Journal of King Saud University-Computer and Information Sciences, 2020, 32(4), 447-453. https://doi.org/10.1007/s12530-020-09345-2
11. Joao, C., Zisserman, A. Quo Vadis Action Recognition? A New Model and the Kinetics Dataset. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, 6299-6308. https://doi.org/10.1109/CVPR.2017.502

12. Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., Fei-Fei, L. Large-Scale Video Classification with Convolutional Neural Networks. Proceedings IEEE Conference on CVPR, 2014, 1725-1732. https://doi.org/10.1109/CVPR.2014.223

13. Kuehne, H., Jhuang, H., Garrote, E., Poggio, T., Serre, T. HMDB: A Large Video Database for Human Motion Recognition. IEEE International Conference on Computer Vision, 2011, 2556-2563. https://doi.org/10.1109/ICCV.2011.6126543

14. Laptev, I. On Space-Time Interest Points. International Journal of Computer Vision, 2005, 64, 107-123. https://doi.org/10.1007/s11263-005-1838-7

15. Laptev, I., Lindeberg, T. Space-Time Interest Points. Proceedings of 9th IEEE International Conference Computer Vision, 2003, 1, 432-439. https://doi.org/10.1109/ICCV.2003.1238378

16. Lecun, Y., Bottou, L., Bengio, Y., Haner, P. Gradient-Based Learning Applied to Document Recognition. Proceedings IEEE, 1998, 86, 2278-2324. https://doi.org/10.1109/5.726791

17. Li, Q., Yang, W., Chen, X., Yuan, T., Wang, Y. Temporal Segment Connection Network for Action Recognition. IEEE Access, 2020, 8, 179118-179127. https://doi.org/10.1109/ACCESS.2020.3027386

18. Li, X., Meng, F., Zhao, F., Guo, D., Lou, F., Jing, R. Two-Stream Adaptive-Attentional Subgraph Convolution Networks for Skeleton-Based Action Recognition. Multimedia Tools and Applications, 2022, 81(4), 4821-4838. https://link.springer.com/article/10.1007/S11042-021-11026-4 https://doi.org/10.1007/s11042-021-11026-4

19. Liu, J., Luo, J., Shah, M. Recognizing Realistic Actions from Videos in the Wild Recognizing Realistic Actions from Videos in the Wild. Proceedings of IEEE CVPR, 2009, 1996-2003. https://doi.org/10.1109/CVPR.2009.5206744

20. Niebles, J.C., Chen, C.W., Fei Fei, L. Modeling Temporal Structure of Decomposable Motion Segments for Activity Classification. Proceedings of European Conference Computer Vision, 2010, 392-405. https://doi.org/10.1007/978-3-642-15552-9_29

21. Peng, X., Zou, C., Qiao, Y., Peng, Q. Action Recognition with Stacked Fisher Vectors. European Conference on Computer Vision, 2014, 581-595. https://doi.org/10.1007/978-3-319-10602-1_38

22. Qian, H., Mao, Y., Xiang, W., Wang, Z. Recognition of Human Activities Using SVM Multi-Class Classifier. Pattern Recognition, 2010, 31(2), 100-111. https://doi.org/10.1016/j.patrec.2009.09.019

23. Radu, V., Tong, C., Bhattacharya, S., Lane, N.D., Mascolo, C., Marina, M.K., Kawsar, F. Multimodal Deep Learning of Activity and Context Recognition. Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies, 2018, 1(4), 1-27. https://doi.org/10.1145/3161174

24. Ronao, C.A., Cho, S.B. Human Activity Recognition with Smartphone Sensors Using Deep Learning Neural Networks. Expert Systems with Applications, 2016, 59, 235-244. https://doi.org/10.1016/j.eswa.2016.04.032

25. Shi, X., Chen, Z., Wang, H., Yeung, D.Y., Wong, W.K., Woo, W.C. Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting. Advances in Neural Information Processing Systems, 2015, 28

26. Simonyan, K., Zisserman, A. Two-Stream Convolutional Networks for Action Recognition in Videos. Advances in Neural Information Processing Systems, 2014, 568-576. https://doi.org/10.3837/tiis.2021.10.011

27. Soomro, K., Zamir, A.R., Shah, M., UC101: A Dataset of 101 Human Actions Classes from Videos in the Wild. CRCV-TR-12-01, 2012, arXiv preprint arXiv:1212.0402

28. Sun, J., Wang, J., Yeh, T. C. Video Understanding: From Video Classification to Captioning. Computer Vision and Pattern Recognition, 2017, 1-9

29. Szegedy, C., Vanhoucke, V., Ioe, S., Shlens, J., Wojna, Z. Rethinking the Inception Architecture on Computer Vision. IEEE Conference on CVPR, 2016, 2818-2826. https://doi.org/10.1109/CVPR.2016.308

30. Tran, D., Ray, J., Shou, Z., Chang, S.F., Paluri, M. ConvNet Architecture Search for Spatiotemporal Feature Learning, 2017. arXiv preprint arXiv:1708.05038

31. Tu, Z., Li, H., Zhang, D., Dauwels, J., Li, B., Yuan, J. Action-Stage Emphasized Spatiotemporal VLAD for Video Action Recognition, IEEE Transactions on Image Processing, 2019, 28(6), 2799-2812. https://doi.org/10.1109/TIP.2018.2890749

32. Victoria, A.H., Maragatham, G. Automatic Tuning of Hyperparameters Using Bayesian Optimization. Evolving Systems, 2021, 217-223. https://doi.org/10.1007/s12530-020-09345-2

33. Wang, H., Klaser, A., Schmid, C., Liu, C.L. Action Recognition by Dense Trajectories. Proceedings of IEEE CVPR, 2011, 3169-3176. https://doi.org/10.1109/CVPR.2011.5995407

34. Wang, H., Schmid, C. Action Recognition with Improved Trajectories. Proceedings of the IEEE Internatio-

nal Conference on Computer Vision, 2013, 3551-3558. https://doi.org/10.1109/ICCV.2013.441

35. Wang, H., Ullah, M.M., Klaser, A., Laptevd, I., Schmid, C. Evaluation of Local Spatio-Temporal Features for Action Recognition, British Machine Vision Conference, 2009, 124.1-124.11. https://doi.org/10.5244/C.23.124
36. Wang, L., Xiong, Y., Wang, Z., Qiao, Y., Lin, D., Tang, X., Van Gool, L. Temporal Segment Networks: Towards Good Practices for Deep Action Recognition. Proceedings of European Conference Computer Vision, Springer, 2016, 20-36. https://doi.org/10.1007/978-3-319-46484-8_2
37. Wu, J., Chen, X. Y., Zhang, H., Xiong, L. D., Lei, H., Deng, S. H. Hyperparameter Optimization for Machine Learning Models Based on Bayesian Optimization. Journal of Electronic Science and Technology, 2019, 26-40. https://doi.org/10.11989/JEST.1674-862X.80904120
38. Xiong, X., Min, W., Han, Q., Wang, Q., Zha, C. Action Recognition Using Action Sequences Optimization and Two-Stream 3D Dilated Neural Network. Computational Intelligence and Neuroscience, 2022. https://doi.org/10.1155/2022/6608448
39. Xu, C., Yang, J., Gao, J. Coupled Learning CNN for Object Recognition. Multimedia Tools and Applications, 2019, 78, 573-589. https://doi.org/10.1007/s11042-017-5262-0
40. Yue-Hei, N. G. J., Hausknecht, M., Vijayanarasimhan, S., Vinyals, O., Monga, R. Toderici, G. Beyond Short Snippets: Deep Networks for Video Classification. CVPR, 2015, 4694-4702. https://doi.org/10.1109/CVPR.2015.7299101
41. Zhu, Y., Lan, Z., Newsam, S., Hauptmann, A. Hidden Two-Stream Convolutional Networks for Action Recognition. Springer International Publishing, 2018, 363-378. https://doi.org/10.1007/978-3-030-20893-6_23