Proceedings of 14th IOE Graduate Conference
Peer Reviewed
Year: 2023 Month: December Volume: 14
ISSN: 2350-8914 (Online), 2350-8906 (Print)

Human Action Recognition using Deep Learning Methods

Mikal Shrestha [a], Sanjeeb Prasad Pandey [b]

[a] *Department of Electronics and Computer Engineering, Pulchowk Campus, IOE, Tribhuvan University, Nepal*
[a] 078msdsa012.mikal@pcampus.edu.np, [b] sanjeeb@ioe.edu.np

Abstract
Human action recognition (HAR) has a numerous potential applications in surveillance, robotics, healthcare, virtual reality, human-computer interaction etc. The recognition of human actions is a complex problem that has emerged as a significant challenge in pattern recognition and video classification, as it involves varying motions with diverse target items and numerous objects appearing in various ways in various contexts. Due to their distinctive characteristics and superior capacity to analyze video sequences in order to comprehend human motion, human action identification techniques have attracted a lot of attention from next-generation technologies. As a result, methods for recognizing human action have been useful in numerous domains. Several methods for identifying human action relied heavily on deep learning techniques. Transfer learning is the method used to spread the new learning age. Conventional video processing techniques are insufficient to tackle this problem, so a modern hybrid deep learning framework is required to solve the problems with higher accuracy. Hybrid two stream (spatial-temporal) Convolutional Neural Network (CNN)- Long short-term memory (LSTM) model with transfer learning approach is used for action recognition, which involves pre-training a standard CNN on a generic dataset and then trained on a hybrid model for recognition on the target UCF-101 dataset. This research explores action recognition in UCF-101 dataset using multiple hybrid deep learning models and analyze how models performs in benchmark dataset. DenseNet201 with LSTM model outperforms in UCF-101 dataset achieving accuracy of 94.8%.

Keywords
Human Action Recognition, CNN, LSTM, UCF-101, Transfer Learning

1. Introduction

Video based human action recognition (HAR) is one of the most applicable fields in computer vision and pattern recognition. Many different fields including surveillance, robotics, healthcare, virtual reality, identity recognition, video searching, human-computer interaction, automatic labeling of a video according to the actions etc. are the potential use cases. Human action are indications that make the study of human behavior easier to understand. Human action recognition is a significant challenge in many applications that combine human computer interaction (HCI) and intellectual video surveillance for enhancing security in a variety of domains. It has emerged as one of the most difficult and attractive problems in the disciplines of pattern recognition and video classification. It can be difficult to identify human actions or activities in video sequences because of issues like backdrop clutter, partial occlusion, changes in scale, viewpoint, lighting, and look among others. Conventional approach always takes similar ideas of image recognition but human action involves varying motions with diverse target items, and numerous objects appear in various ways in various contexts. The issue itself is difficult to tackle using conventional video processing techniques. So modern hybrid deep learning framework is required to solve the existing problems with higher accuracy. A hybrid two stream(spatial-temporal) Convolutional Neural Network (CNN)- Long short-term memory (LSTM) model with transfer learning approach has been used for this research. Standard convolutional neural network were first train on a generic dataset with the help of transfer learning in pre-training phase and then applied to the target UCF-101 dataset to train using hybrid model for the recognition.

The main objectives of this research is: (a) to apply the concept of transfer learning and analyze its impact on accuracy; (b) to apply the hybrid deep learning model to recognize action; and (c) to analyze and compare the results of different performance metrics for benchmark dataset with existing results.

2. Literature Review

Human activity or action recognition techniques start from the handcrafted feature-based approach to advanced AI-based deep learning techniques. Human action recognition has seen impressive performance when using handcrafted features descriptors like extended SURF, HOG-3D, and other shape and motion-based features descriptors. For feature extraction and representation, handcrafted feature-based techniques need specially built feature detectors, descriptors, and vocabulary building procedures. This feature engineering procedure is labor-intensive and necessitates subject-matter expertise. More research is being done on approaches based on deep learning as a result of these restrictions. This method has been applied in a variety of fields, including object detection, speech recognition, and image classification, to mention a few. These models have also been investigated for the detection of human activities. Convolutional RBMs, learning spatiotemporal using 3D ConvNets, Deep ConvNets, and Two-stream ConvNets are some notable contributions that have produced impressive results. Online deep learning is also receiving increased attention, and some academics have suggested utilizing this approach to recognize actions.

A spatial stream and a temporal stream are the two forms of

networks suggested by Simonyan and Zisserman [1]. The spatial network was given access to uncompressed video frames, and optical flow fields served as the input for the temporal stream. The SoftMax score was then utilized to combine the two streams. The UCF-101 and HMDB-51 standards were used for training and assessing its architecture. The extension of two-stream networks included enhanced trajectory. Deep encoding of features learned from deep CNN architecture was accomplished via trajectory-constrained sampling and pooling. A hybrid paradigm was put up by Singh and Vishwakarma [2] for automating human activity recognition. Architecture Inception-v3 was selected. Additionally, they used the Bi-LSTM model to process the RGB frames. Instead of using optical flow, they employed the compact single dynamic motion image (DMI) method to deal with view variations and occlusions in images. They solely used RGB frames to learn the features in order to lessen the complexity of their model. A two-stream approach for activity recognition that combines residual- CNN and Transfer Learning was proposed [3]. In addition to previous fusion approaches, they used sum fusion, max fusion, weighted average fusion, and weighted product fusion. These researchers used 2D and 3D residual networks to create the two-stream model. They used the standard UCF101 HMDB-51 benchmark dataset to test the performance of their architectures. A spatial stream and a temporal stream are the two forms of networks suggested by Simonyan and Zisserman. The spatial network was given access to uncompressed video frames, and optical flow fields served as the input for the temporal stream. The SoftMax score was then utilized to combine the two streams. The UCF-101 and HMDB-51 standards were used for training and assessing its architecture.

A CNN-based 3D architecture for multifunctional information channels produced from neighboring video frames has been introduced by Ji et al.. To extract spatial and temporal properties, they used 3D kernels [4]. According to experimental results, this architecture is more high-performance than its competitors' 2D frame-based rivals. Five 3D pooling layers were included, and a compact descriptor known as C3D was used to average the results of the first fully connected network layer. They did, however, create brief video clips and combine spatial and temporal data using a late score fusion. If a lengthy series of motions, such walking or swimming, took place over the course of several video frames and lasted a few seconds, this did not function. Not all of the acts in their full temporal form have been modeled. A deep, fractional spatial-temporal network was proposed by Sun et al (FstCN). The major goal was to factorize a 3D filter into a combination of 2D spatial and 1D temporal kernels on the bottom and higher network layers. The amount of network parameters that need to be investigated has been drastically cut down, which helps to reduce high kernel complexity and the inability to train video data. FstCN was examined by the HMDB-51 and UCF-101. The current CNN techniques were better. Additionally, it accomplished noteworthy success without the use of additional training videos.

According to research on visual cognition, humans never focus on a full scene at once, but rather successively on various sections of it to extract pertinent information. Therefore, using an attention mechanism [5] will aid in enhancing performance on tasks related to learning. For the purpose of recognizing actions, a recurrent soft attention model has been created. The likelihood of a location and class label at the following time step is predicted using LSTM. Then, to compute the expected value of the input at the following time-step, the soft attention method uses expectation over the feature slices at various regions. However, the method is computationally expensive because dynamic pooling requires the use of all the features. For the purpose of recognizing actions in videos, a hierarchical attention network [6] that takes into account static spatial information, short-term motion information, and long-term video temporal structures has been proposed. First, frame images and the associated optical flow images are used to extract appearance and motion features using two-stream ConvNets, respectively. Second, the video temporal structure is modelled using a hierarchical LSTM that has two layers. Then the appearance and motion attributes are used to compute the attention weights.

3. Methodology

The proposed methodology combines Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks to create a hybrid model for action recognition. Initially, the CNN extracts spatial features from video frames to capture spatial information effectively. The output from the CNN is then fed into the LSTM, which is designed to capture temporal dependencies and sequential patterns within the video sequence. Transfer learning is employed by initializing the CNN with pre-trained weights on a large dataset, enhancing the model's ability to generalize and learn intricate features. This integrated approach aims to achieve robust action recognition by effectively combining spatial and temporal information through the synergy of CNN and LSTM architectures with the benefits of transfer learning.

3.1 High Level System Workflows

Hybrid two stream (spatial and temporal stream) CNN-LSTM model was used for this research. Transfer Learning has been used to in pre-training phase for spatial stream to train on a generic dataset to the adjust weights which is then applied to the target dataset to recognize using custom LSTM model. Before receiving the RGB-frames, this architecture pre-trains

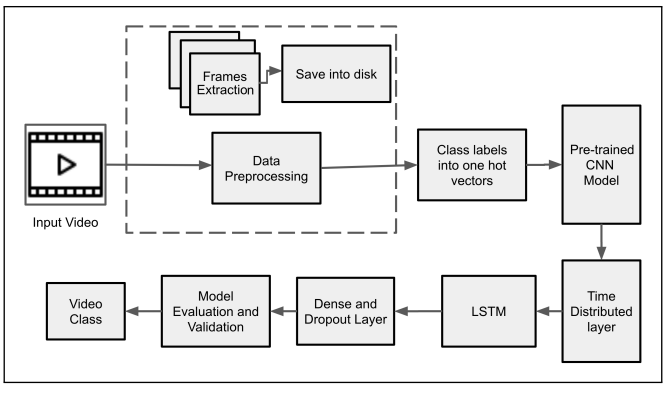

Figure 1: High level system workflows

the model using VGG16, DenseNet. After some pre-processing and data augmentation, these frames move across a layer that distributes time. The Temporal-CNN- LSTM architecture has been used in the temporal stream. The following figure shows the overall flow of the system.

3.2 Dataset Description

The UCF-101 dataset is a publicly available dataset and can be downloaded from the UCF (University of Central Florida) Center for Research in Computer Vision(crcv) website and also available in Kaggle [7]. The website provides detailed information about the data source, including the method used to collect the videos, the annotation process, and the evaluation protocol. The videos in UCF-101 are collected from YouTube and are diverse in terms of actions, subjects, and scenarios. The dataset is designed to reflect real-world human actions and to be representative of a wide range of actions performed by different people. It has 13,320 videos in 101 categories of human behavior. Total videos size is almost 6 GB and having frame rate of 25 FPS with the resolution of 320 × 240. The largest variety of actions is provided by UCF101, which has 13320 videos across 101 action categories. It is also the most difficult data set to date due to the wide range of camera motion, object appearance and pose, object scale, viewpoint, cluttered background, illumination conditions, and other factors. UCF101 intends to promote additional study into action recognition by learning and exploring new realistic action categories because the majority of the existing action recognition data sets are not realistic and are staged by actors.

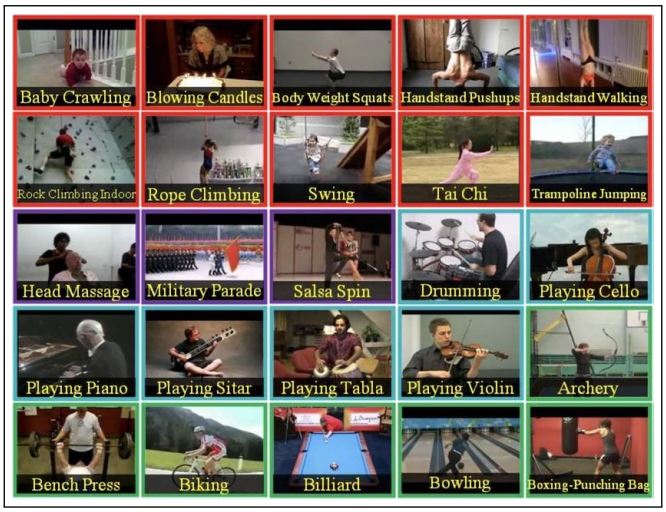

Figure 2: UCF-101 dataset sample

All classes of dataset are follows: Apply Eye Makeup, Apply Lipstick, Archery, Baby Crawling, Balance Beam, Band Marching, Baseball Pitch, Basketball Shooting, Basketball Dunk, Bench Press, Biking, Billiards Shot, Blow Dry Hair, Blowing Candles, Body Weight Squats, Bowling, Boxing Punching Bag, Boxing Speed Bag, Breaststroke, Brushing Teeth, Clean and Jerk, Cliff Diving, Cricket Bowling, Cricket Shot, Cutting In Kitchen, Diving, Drumming, Fencing, Field Hockey Penalty, Floor Gymnastics, Frisbee Catch, Front Crawl, Golf Swing, Haircut, Hammer Throw, Hammering, Handstand Pushups, Handstand Walking, Head Massage, High Jump, Horse Race, Horse Riding, Hula Hoop, Ice Dancing, Javelin Throw, Juggling Balls, Jump Rope, Jumping Jack, Kayaking, Knitting, Long Jump, Lunges, Military Parade, Mixing Batter, Mopping Floor, Nun chucks, Parallel Bars, Pizza Tossing, Playing Guitar, Playing Piano, Playing Tabla, Playing Violin, Playing Cello, Playing Daf, Playing Dhol, Playing Flute, Playing Sitar, Pole Vault, Pommel Horse, Pull Ups, Punch, Push Ups, Rafting, Rock Climbing Indoor, Rope Climbing, Rowing, Salsa Spins, Shaving Beard, Shotput, Skate Boarding, Skiing, Skijet, Sky Diving, Soccer Juggling, Soccer Penalty, Still Rings, Sumo Wrestling, Surfing, Swing, Table Tennis Shot, Tai Chi, Tennis Swing, Throw Discus, Trampoline Jumping, Typing, Uneven Bars, Volleyball Spiking, Walking with a dog, Wall Pushups, Writing On Board, Yo Yo.

3.3 Data Preprocessing and Feature Extraction

Long range temporal information of videos cannot incorporate by the existing deep learning architecture which operate on a single frame or stack of consecutive frame. For the long-range videos processing, the cost of ConvNet training increases. Segmentation of input video is done using sparse temporal sampling technique to cover the whole video. In this pre-processing phase, both RGB and Optical Flow (OF) frame will generate by the algorithm from the input video. And frames are normalized the min-max normalization and dropout approach to guarantee the same-size frames. This process is important for the fusion process; ignoring it may cause the overfitting problem. A video is a collection of sequences of frames. Thus, in order to process the videos respective frames contained in video should be extracted in order to get useful features of the particular action. Two major steps are being carried out which consists of the feature extraction by CNN model and the classification by RNN model. Due to resource limitation for training, 4040 videos has been used for training the models. Each classes having 40 videos. Spatial and temporal frames were extracted from the videos and resize that frame by 64*64.

The technique of extracting individual RGB frames in the RGB (Red, Green, and Blue) color space from a video stream is a key one in computer vision. Many computer vision tasks, such as object detection, tracking, and classification, require the use of this procedure. Among the 4040 videos, 11,131 spatial RGB frames were extracted. Convolutional Neural Network (CNN) is used as a feature extractor which means that this model helps to extract spatial features of the images. Images in the video are considered as frames. Thus, different architecture of CNN models is used for feature extraction. All the extracted frames are passed into the feature extractor which is the CNN model and all the features of those frames are extracted out. The data size is of huge size so all the extracted frames are resized to 64*64 as of now and given to the CNN model. Transfer learning generally refers to a process where a model trained on one problem is used in some way on a second related problem. Transfer learning relies on prior network training on a general dataset for feature extraction. The network weights are then modified for the classification task. With the help of transfer learning, the pre-trained features are extracted in the pre-training phase before passing into the classification model. Which will save the resources and improved efficiency. The common state of art architecture which are adopted as a feature extractor for transfer learning

are: DenseNet-201 and VGG-16. The models are pre-trained on the ImageNet dataset which is a very large-scale dataset of over 15 million labeled high-resolution images with roughly 22,000 categories.

Video frames are extracted at regular intervals to create a representative sequence. Frames are resized to a fixed height and width (64x64 pixels). Extracted frames are normalized by dividing pixel values by 255, ensuring they fall between 0 and 1. Normalized frames are saved as JPEG images for future reference, reducing the need for repetitive extraction. A sequence of 20 frames is chosen to represent the temporal aspect of the video. Dataset was created by iterating through class folders containing video files. For each video file, frames were extracted using the defined frames extraction process. If the number of frames matches the specified sequence length, add frames, labels and video paths to respective lists. And converted lists to NumPy arrays for compatibility with machine learning frameworks. For data augmentation, utilized the ImageDataGenerator from Keras to perform data augmentation. Augmentation includes rotation (20), width and height shifts (0.2), shear (0.2), zoom (0.2), and horizontal flipping. Augmented data is used during the model training process to expose the network to a diverse set of examples. The resulting dataset, consisting of preprocessed frames, corresponding labels, and video paths, is prepared for further use in training a neural network. The data augmentation process enhances the model's ability to generalize by introducing variability in the training set.

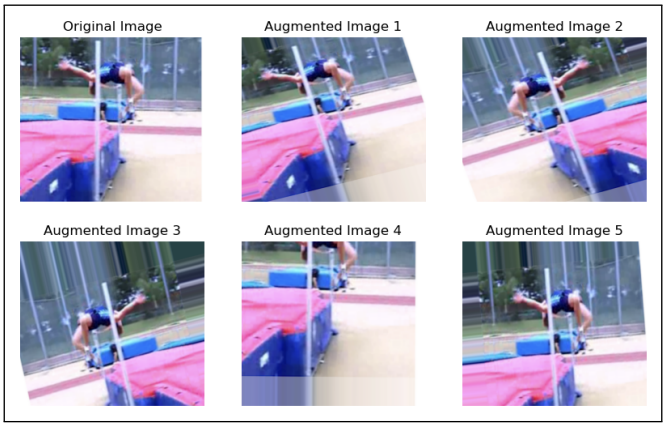

Figure 3: Data augmentation representation

3.4 Model Architectures

The following model architectures were used to train and validate the model to analyses and compare the performance of different models.

3.4.1 LRCN Model

The Long Short-Term Memory (LSTM) layer, the Dense layer, and the Convolutional layer make up the model's architecture [8]. A sequence of frames with the form (SEQUENCE LENGTH, IMAGE HEIGHT, IMAGE WIDTH, 3), where SEQUENCE LENGTH is the sequence's length, IMAGE HEIGHT and IMAGE WIDTH are the frames' height and width, and 3 denotes the RGB channels, is the model's input. With the Time Distributed wrapper, the Convolutional layers are individually applied to each frame of the sequence. To minimize the spatial dimensions of the feature maps, the model employs four Convolutional layers with 16, 32, 64, and 64 filters, respectively, each followed by a MaxPooling2D layer. A Dropout layer is placed after each Convolutional layer to avoid overfitting. After the Convolutional layers, the output from each frame is flattened and fed to a LSTM layer with 32 hidden units to capture the temporal dependencies between frames. Finally, a Dense layer with a softmax activation function is added to produce the output classification probabilities for each video class.

3.4.2 DenseNet201 and LSTM

This model combines a Long Short-Term Memory (LSTM) network with a DenseNet201 convolutional neural network (CNN). A sequence of frames with the form (SEQUENCE LENGTH, IMAGE HEIGHT, IMAGE WIDTH, 3), where SEQUENCE LENGTH is the sequence's length, IMAGE HEIGHT and IMAGE WIDTH are the frames' height and width, and 3 denotes the RGB channels, is the model's input. A pre-trained CNN model with weights from the ImageNet dataset is called DenseNet201. To avoid overfitting and maintain the learned features, the layers of the DenseNet201 model are frozen. To handle the input frame sequence, the TimeDistributed layer is applied to the DenseNet201 model as a whole. The TimeDistributed GlobalAveragePooling2D layer uses the output from the DenseNet201 model to average the spatial dimensions of the feature maps over all frames in the sequence, producing a fixed-length vector for each frame. An LSTM layer with 256 hidden units is then given the output from the TimeDistributed GlobalAveragePooling2D layer, which learns the temporal dependencies between the frames in the sequence. In order to avoid overfitting, the output from the LSTM layer is subsequently passed through two fully linked Dense layers with 128 and 64 neurons each. In order to create the output classification probabilities for each video class, a Dense layer with a softmax activation function is added last.

3.4.3 VGG-16 and LSTM

A pre-trained VGG16 convolutional neural network (CNN) and a long short-term memory (LSTM) network make up this model. The LSTM network is used to describe temporal dependencies between frames, while the VGG16 CNN is used to extract spatial characteristics from each frame of a video. Only the LSTM layers are trainable during training since the VGG16 layers have been frozen. The VGG16 model is first applied to each frame of the input video sequence via a TimeDistributed layer of the model. The output of the VGG16 model is then passed through another TimeDistributed layer that uses a GlobalAveragePooling2D layer to condense the feature maps' spatial dimensions into a single vector every frame. Following the addition of the dense layer with 128 units and ReLU activation function, the LSTM layer with 256 units is added. To regularize the model, a dropout layer with a rate of DROPOUT RATE is added. Next, a dense layer with 64 units and ReLU activation is applied. In order to generate the output probabilities for the various classes, a dense layer with a softmax activation function is added last.

3.5 LSTM Model for Recognition

Sequential frames that change from time to time make up video. Hence, features that are strongly related to time or change over time are considered temporal features. The phrase "temporal characteristics" refers to the order of frames in a movie that are linked together over time steps. The information contained in the sequence itself is used by recurrent neural networks (RNN) to carry out recognition tasks. RNNs have loops, therefore their output depends on both the input and the output from earlier iterations. RNNs have the drawback of being unable to actually learn long-term dependencies. Long Short-Term Memory (LSTM) is therefore employed to resolve this problem. After the feature is extracted from the individual frames by CNN model, we will get corresponding spatial features of those extracted frames which is then flattened then series of feature vectors will be passed as a sequential input to the LSTM model. The model is trained with those feature vectors of the spatial features and further classification task of different action is carried out. A fully connected layer followed by softmax activation function is present after the LSTM layer where softmax activation function gives probabilistic score of the corresponding action. Adam optimizer is used for optimization algorithms and categorical cross entropy is used for calculating the loss. Adam creates an optimization technique that can handle sparse gradients on noisy situations by combining the best features of the AdaGrad and RMSProp algorithms. Cross entropy loss is computed between the labels and predictions using categorical cross entropy.

CNN is most useful for extracting spatial elements, it is crucial to leverage the temporal characteristics from these RGB video frames. Since the input of the entire model is a series of photos that are chronologically related, an LSTM is produced in the present model. The LSTM network in this model is set up to have only one direction. It has one LSTM layer and two completely connected layers.

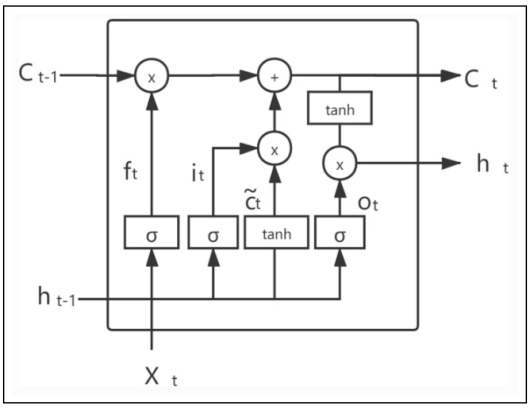

Figure 4: Kernel structure of LSTM

$$
\begin{aligned}
i_t &= \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i) \\
f_t &= \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f) \\
c_t &= f_t c_{t-1} + i_t \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \\
o_t &= \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o) \\
h_t &= o_t \tanh(c_t)
\end{aligned} \tag{1}
$$

Figure shows the kernel of the LSTM layer, where x represents the input signal, tanh stands for the activation functions, and c and h stand for the cell state and hidden state, respectively. Three gates are used by the LSTM to decide which information is useful. The three gates in the LSTM are described by the equations above. The parameter matrix W is represented in these equations. The information that will not be used in the current cell is selected by the forget gate. The following cell's input is determined by the input gate. The hidden state outputted from the current cell is determined by the output gate. In particular, each cell in the LSTM layer of this model contains 512 hidden units. The number of classes in the dataset is reflected in the output shape of this block. Between full-connected layers, a dropout layer is introduced to decrease the overfitting.

Optimizers: Different optimizers like RMSProp, AdaGrad, SGD, Adam is used for optimizing the model by minimizing the loss. Among all of them, Adam was best fit optimizer for this thesis research. Adaptive Moment Estimation (Adam), is an optimization algorithm widely used in training neural networks. Combining elements of both the momentum and RMSprop algorithms, Adam dynamically adjusts the learning rates for each parameter individually. Its adaptive nature allows it to handle sparse gradients and noisy data effectively. By maintaining per-parameter learning rates and incorporating moment estimates, Adam enhances convergence speed and overall training efficiency.

Algorithm:

Require: α: Stepsize
Require: $\beta_1, \beta_2 \in [0, 1)$: Exponential decay rates for the moment estimates
Require: $f(\theta)$: Stochastic objective function with parameters θ
Require: θ_0: Initial parameter vector
$m_0 \leftarrow 0$ (Initialize 1st moment vector)
$v_0 \leftarrow 0$ (Initialize 2nd moment vector)
$t \leftarrow 0$ (Initialize timestep)
while θ_t not converged **do**
$t \leftarrow t + 1$
$g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$ (Get gradients w.r.t. stochastic objective at timestep t)
$m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$ (Update biased first moment estimate)
$v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$ (Update biased second raw moment estimate)
$\widehat{m}_t \leftarrow m_t / (1 - \beta_1^t)$ (Compute bias-corrected first moment estimate)
$\widehat{v}_t \leftarrow v_t / (1 - \beta_2^t)$ (Compute bias-corrected second raw moment estimate)
$\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \widehat{m}_t / (\sqrt{\widehat{v}_t} + \epsilon)$ (Update parameters)
end while
return θ_t (Resulting parameters)

Activation Functions: An activation function in a neural network is a mathematical operation applied to each neuron in a layer of the network. It introduces non-linearity to the model, allowing it to learn complex patterns and relationships in the data. Activation functions determine the output of a node, given its input or set of inputs. They play a crucial role in shaping the output of a neural network and enabling it to approximate complex, non-linear mappings between inputs and outputs. relu(R(z)), tanh, softmax activation functions were used for this research.

$$
\begin{aligned}
R(z) &= \max(0, z) \\
\delta(z) &= \frac{e^z - e^{-z}}{e^z + e^{-z}} \\
\text{softmax}(z)_i &= \frac{e^{z_i}}{\sum_{j=1}^{N} e^{z_j}}
\end{aligned} \tag{2}
$$

3.6 Cross Entropy

To determine the difference between two probability distributions, cross-entropy is widely used. A one hot distribution can usually be used to define the "true" distribution. The amount of "wrong" or "far" the model's forecast from the actual data determines cross entropy loss. The categorical cross-entropy loss is a commonly used loss function in deep learning, particularly when working with multi-class classification problems. In a multi-class classification problem, the goal is to predict one of several mutually exclusive classes for a given input. The categorical cross-entropy loss is used to quantify the difference between the predicted class probabilities and the true class labels. The predicted class probabilities are usually produced by a softmax activation function applied to the final layer of the neural network. The categorical cross entropy loss punishes the model more for incorrect predictions with higher confidence. This is because the logarithmic nature of the loss means that high confidence incorrect predictions contribute more to the loss than low-confidence incorrect predictions.

$$L_{CE} = -\sum_{i=1}^{C} y_i \cdot \log(p_i) \tag{3}$$

where,

L_{CE} : Categorical Cross-Entropy (Logarithmic Loss)

C : Number of classes

y_i : True probability for class i

p_i : Predicted probability for class i

4. Results and Discussion

Several experiments were conducted to confirm the usefulness and efficiency of the proposed framework. Important metrics from earlier studies are utilized to describe how well these experiments performed. The experiments are performed on a Google Collab and Jupyter Notebook which is executed and trained on a MacBook Pro M1 having 16 GB Ram and 256 GB Disk. Python 3 was the used programming language. The used packages are TensorFlow, Keras, OpenCV, Numpy, Matplotlib, Moviepy etc.

4.1 Dataset Representation

There are altogether 4040 videos of different human actions which consists 101 categories or classes. Each class consists of almost 40 videos each. And 80,800 frames were extracted for further processing and analysis.

Dataset was split into training, testing and validation sets by 70%, 15% and 15% to facilitate robust model evaluation. In training set, 3434 samples for frames were used, each consisting of 20 frames with dimensions 64x64 pixels in RGB format and (3434,101) labels sample were used.

For evaluation purposes, testing and validation set was created, consists of 606 samples of frames, labels. These split datasets enable the model to be trained on a diverse set of examples and evaluated on unseen data, providing insights into its effectiveness and generalization capabilities.

Shape Structure: (sample features, sequence length, height, width, rgb image)

Table 1: UCF-101 dataset's features description and initial shape structure

Features	Description	Shape
Frames	A sequence of frames extracted from video data. Represents the visual content over time, providing information about the actions and events occurring in a video.	(4040, 20, 64, 64, 3)
Labels	Categorical labels assigned to each sequence of frames. Identifies the action or category associated with the corresponding video sequence.	(4040)

Table 2: UCF-101 dataset's shapes after train-test split

Features	Train Shape	Test/Validation Shape
Frames	(3434, 20, 64, 64, 3)	(606, 20, 64, 64, 3)
Labels	(3434, 101)	(606, 101)

4.2 Experiments on Different Architectures

4.2.1 Experiment-1: LRCN Model

In this experiment, LRCN (Long-term Recurrent Convolutional Networks) model has been used. Almost 25 minutes time were required to train the model for 64 epochs. This is a first experiment and simple architecture with batch size of 8, dropout rate of 0.5. Relu activation function has been used in convolutional layer and softmax activation function has been used in Dense layer and Adam optimizer were used to optimize the model.

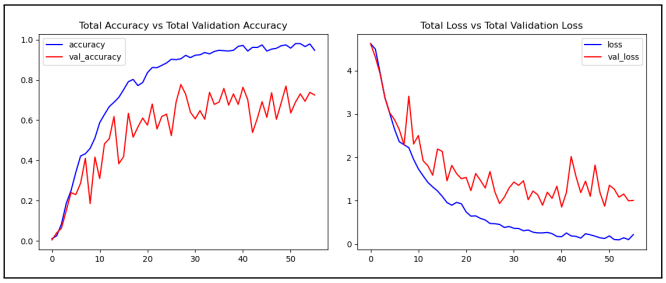

Figure 5: Accuracy and Loss plot for LRCN Model

4.2.2 Experiment-2: DenseNet201 with LSTM Model

In this experiment, DenseNet201 pretrained model has been used in pretraining phase and then it passed through the custom model for fine tuning. Almost 4-hour time were required to train the model for 32 epochs. This experiment was done with dropout rate of 0.5 and batch size of 16. Adam Optimizer has been used to optimize the model and adjust the learning rate of the model.

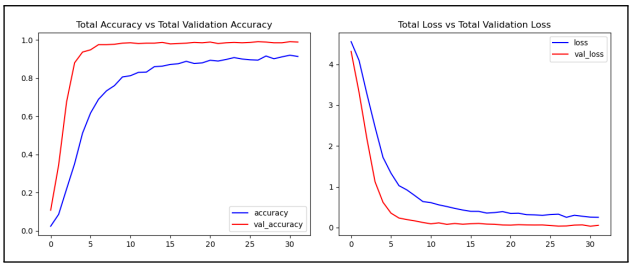

Figure 6: Accuracy and Loss plot for DenseNet201 with LSTM Model when dropout 0.5

When dropout rate is o.4, almost 1.5-hour time were required to train the model for 32 epochs.

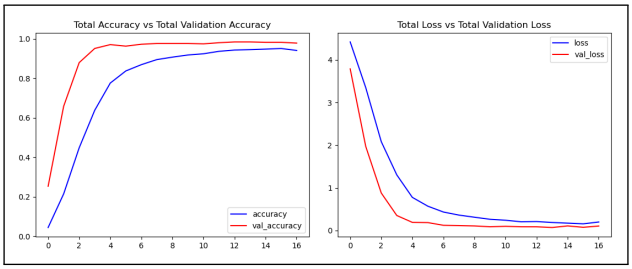

Figure 7: Accuracy and Loss plot for DenseNet201 with LSTM Model when dropout 0.4

4.2.3 Experiment-3: VGG-16 with LSTM Model

In this experiment, VGG16 pretrained model has been used in pretraining phase and then it passed through the custom model for fine tuning. Almost 4-hour time were required to train the model for 32 epochs. This experiment was done with dropout rate of 0.5 and batch size of 16. Adam Optimizer were used to optimize the model and adjust the learning rate of the model.

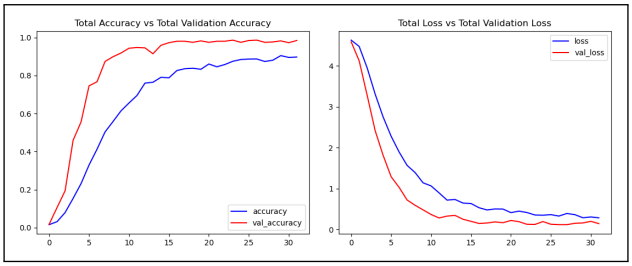

Figure 8: Accuracy and Loss plot for VGG-16 with LSTM Model ehrn dropout 0.5

When dropout rate is 0.4, almost 3-hour time were required to train the model for 32 epochs.

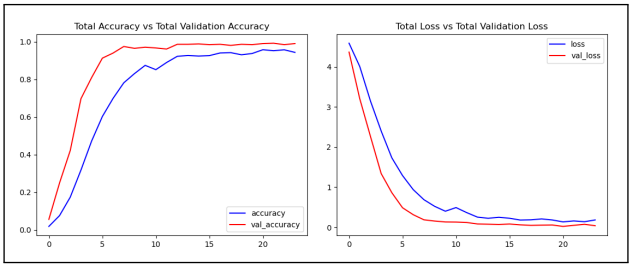

Figure 9: Accuracy and Loss plot for VGG-16 with LSTM Model when dropout 0.4

From the multiple experiments using hyperparameters tuning, LRCN model converse when epochs= 64 and batch size of 8 but VGG-16 and DenseNet201 with LSTM model converse when batch size of 16 and 32 epochs only.

4.3 Experimental Results

In deep learning, performance analysis and validation play a critical role to compare the performance of models using different metrics and to validate the model performance. Performance analysis assesses the model's effectiveness and accuracy in solving the problem through various metrics, such as accuracy, precision, recall, and F1 score. Meanwhile, validation ensures that the model is working correctly by testing it on independent data and identifying any issues that could affect its reliability. These two components combine to provide a comprehensive evaluation of the model's ability to solve the problem and guarantee its success and dependability. 70% of dataset has been used for training, 15% for testing and 15% for validation. Testing has been done on 15 classes.

At a dropout rate of 0.4, the VGG-16 model with LSTM exhibits signs of overfitting, while the LRCN model demonstrates notably low accuracy. Conversely, the DenseNet201 model with LSTM achieves superior accuracy compared to the other two configurations. It's worth noting that both the DenseNet201 and VGG-16 models with LSTM were trained for 32 epochs, whereas the LRCN model was trained for 64 epochs.

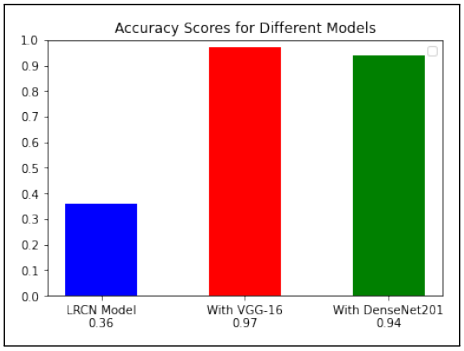

Figure 10: Accuracy comparison when dropout is 0.4

When dropout is 0.5 all, the three models were trained well as compare to above and got good accuracy. In overall comparison, DenseNet201 with LSTM model gives the better accuracy compare to other models.

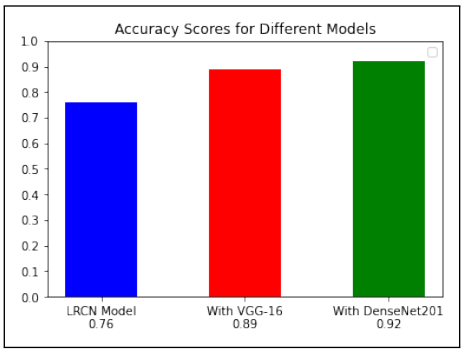

Figure 11: Accuracy comparison when dropout is 0.5

Testing is important to validate the model performance and can determine how model can recognize the action with their confidence level. For testing, 15 class out of 101 to test videos of UCF-101 dataset are used. Test class are: “Diving”, “Drumming”,"EyeMakeup”, “Fencing”, “horserace”, “LongJump”, “Rafting”, “Skiing”, “Swing”, “Tabletennis”, “Taichi”, “Typing”, “Volleybalspiking”, "Walkingwithdog”. The following table shows which class are correctly predicted by the model with their confidence value.

Table 3: Test table for UCF-101 dataset

Test Videos	Predict Correctly?	Confidence
v_Diving_g08_c05.avi	Yes	0.97
v_Diving_g04_c03.avi	Yes	0.99
v_Diving_g08_c02.avi	No	0.71
v_Diving_g01_c04.avi	Yes	0.98
v_Drumming_g05_c01.avi	No	0.68
v_Drumming_g07_c07.avi	Yes	0.98
v_EyeMakeup_g01_c01.avi	Yes	0.98
v_EyeMakeup_g08_c01.avi	No	0.47
v_Fencing_g01_c02.avi	Yes	0.97
v_HorseRace_g01_c01.avi	Yes	0.98
v_LongJump_g07_c05.avi	Yes	0.99
v_LongJump_g06_c03.avi	Yes	0.99
v_Rafting_g12_c01.avi	Yes	0.99
v_Rafting_g09_c02.avi	Yes	0.96
v_Skiing_g01_c03.avi	No	0.59
v_Skiing_g10_c03.avi	Yes	0.97
v_Swing_g01_c01.avi	No	0.18
v_Swing_g05_c06.avi	Yes	0.91
v_TTShot_g08_c01.avi	Yes	0.88
v_TTShot_g13_c03.avi	No	0.26
v_TaiChi_g13_c03.avi	No	0.32
v_TaiChi_g15_c03.avi	Yes	0.98
v_Typing_g10_c01.avi	Yes	0.94
v_VSpiking_g08_c01.avi	No	0.72
v_VSpiking_g04_c06.avi	No	0.48
v_WalkDog_g15_c02.avi	Yes	0.93

Table 4: Performance comparison with the state-of-the-art models for UCF-101 dataset

Authors	Year	Accuracy %
Simonyan and Zisserman [1]	2014	88.00
Zhu et al. [9]	2016	93.10
Ullah et al. [10]	2017	91.21
Chen et al. [11]	2017	93.70
Luo et al. [12]	2019	90.60
Dai et al. [13]	2019	92.20
Cao and Xu [14]	2019	93.60
de Almeida Maia et al. [15]	2020	93.91
Jaouedi et al. [16]	2020	89.30
Bo et al. [17]	2020	89.80
Huang et al. [18]	2020	92.70
Wang et al. [19]	2020	90.30
Y Abdulazeem, HM Balaha, WM Bahgat [3]	2021	94.87
DensNet201 and LSTM (own)	2023	94.8

5. Conclusion and Future Enhancement

Deep learning-based human action recognition has recently emerged as a fascinating field of study and development. The precision of human action recognition systems has significantly increased with the quick developments in computer vision and deep learning. Deep learning algorithms like CNN and RNN have demonstrated success in identifying human actions in video and frame data. Potential uses for this include robotics, surveillance, healthcare, and sports analysis. Systems of this kind can give us insights into how people behave and act in various settings, which can help us better understand human behavior. More precise, effective, and dependable systems that may be used in a variety of sectors are anticipated as a result of ongoing research and development. For the combination of CNN and LSTM model with transfer learning approach gives a good results to recognize the human actions in UCF-101 dataset. In the future, further extension of the presented approach could be training temporal and spatial features separately and apply fusion process lately to generate the output which take larger resources but could get better accuracy.

References

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

[2] T. Singh and D. K. Vishwakarma. *A deeply coupled ConvNet for human activity recognition using dynamic and RGB images.* Neural Comput. Appl., vol. 33, no. 1, pp. 469-485, 2021.

[3] Y Abdulazeem, HM Balaha, and WM Bahgat. *Human Action Recognition Based on Transfer Learning Approach.* IEEEAccess, vol.3, 2021.

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

[5] L Li, L Zhuang, and S Gao S Wang. *HaViT: Hybrid-attention based Vision Transformer for Video Classificatio.* CVF openaccess, 2022.

[6] M Baccouche, F Mamalet, C Wolf, and C Garcia. *Sequential Deep Learning for Human Action Recognition.* Springer, 2011.

[7] *“UCF-101 Dataset official website”. [Online]. Available “https://www.crcv.ucf.edu/data/UCF101.php” [Accessed: Dec-15, 2022].*

[8] Donahue et al. *Long-term recurrent convolutional networks for visual recognition and description.* IEEE Trans. Pattern Anal. Mach. Intell, 2017.

[9] W. Zhu, J. Hu, G. Sun, X. Cao, and Y. Qiao. A key volume mining deep framework for action recognition. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pages 1991–1999, 2016.

[10] A. Ullah, J. Ahmad, K. Muhammad, M. Sajjad, and S. W. Baik. Action recognition in video sequences using deep bi-directional lstm with cnn features. *IEEE Access*, 6:1155–1166, 2018.

[11] H. Chen, J. Chen, R. Hu, C. Chen, and Z. Wang. Action recognition with temporal scale-invariant deep learning framework. *China Commun*, 14(2):163–172, Feb 2017.

[12] X. Luo, O. Ye, and B. Zhou. An modified video stream classification method which fuses three-dimensional convolutional neural network. In *Proc. Int. Conf. Mach. Learn., Big Data Bus. Intell. (MLBDBI)*, pages 105–108, Nov 2019.

[13] W. Dai, Y. Chen, C. Huang, M.-K. Gao, and X. Zhang. Two-stream convolution neural network with video-stream for action recognition. In *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, pages 1–8, Jul 2019.

[14] D. Cao, L. Xu, and D. Zhang. Cross-enhancement transform two-stream 3d convnets for action recognition. 2019. arXiv:1908.08916.

[15] H. de Almeida Maia, D. T. Concha, H. Pedrini, H. Tacon, A. de Souza Brito, H. de Lima Chaves, M. B. Vieira, and S. M. Villela. Action recognition in videos using multi-stream convolutional neural networks. In *Deep Learning Applications*, pages 95–111. Springer, 2020.

[16] N. Jaouedi, N. Boujnah, and M. S. Bouhlel. A new hybrid deep learning model for human action recognition. *J. King Saud Univ. Comput. Inf. Sci*, 32(4):447–453, May 2020.

[17] Y. Bo, Y. Lu, and W. He. Few-shot learning of video action recognition only based on video contents. In *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, pages 595–604, Mar 2020.

[18] Y. Huang, Y. Guo, and C. Gao. Efficient parallel inflated 3d convolution architecture for action recognition. *IEEE Access*, 8:45753–45765, 2020.

[19] H. Wang and J. Li. Human action recognition algorithm based on multi-feature map fusion. *IEEE Access*, 8:150945–150954, 2020.

[20] L Wang, Y Xu, J Cheng, H Xia, J Yin, and J Wu. *Human Action Recognition by Learning Spatio-Temporal Features With Deep Neural Network.* IEEEAccess, 2018.

[21] Y Zhao, KL Man, J Smith, and K Siddique. *Improved two-stream model for human action recognition.* EURASIP Journal, 2020.

[22] AB Sargano, X Wang, and P Angelov. *Human action recognition using transfer learning with deep representations.* Neural Networks, 2017.

[23] AJ Suresh and J Visumathi. *Inception ResNet deep transfer learning model for human action recognition using LSTM.* Elsevier, 2020.

[24] Ming Fu, Qun Zhong, and Jixue Dong. *Sports Action Recognition Based on Deep Learning and Clustering Extraction Algorithm.* Computational Intelligence and Neuroscience, 2022.

[25] A Diba, AM Pazandeh, and L Van Gool. *Efficient Two-Stream Motion and Appearance 3D CNNs for Video Classification.* arxiv.org, 2016.

[26] Z Xu, J Hu, W Deng, and L Van Gool. *Recurrent convolutional neural network for video classification.* IEEE International Conference, 2016.

[27] T Dagiuklas V Tsakanikas. *Video surveillance systems-current status and future trends.* Elsevier, 2018.

[28] B. Fernando, P. Anderson, M. Hutter, and S. Gould. *Discriminative hierarchical rank pooling for activity recognition.* IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2016.

[29] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool. *Temporal segment networks for action recognition in videos.* IEEE Trans. Pattern Anal. Mach. Intell, 2019.

[30] B. Zhang, L. Wang, Z. Wang, Y. Qiao, and H. Wang. *Real-time action recognition with enhanced motion vector CNNss.* IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2016.

[31] C. Feichtenhofer, A. Pinz, and A. Zisserman. *Convolutional two-stream network fusion for video action recognition.* Proc. IEEE Conf. Com-put. Vis. Pattern Recognit. (CVPR), 2016.

[32] H. Wang and J. Li. *Human action recognition algorithm based on multi- feature map fusion.* IEEE Access, 2021.

[33] S. Gollapudi and A. Zisserman. *Learn Computer Vision Using OpenCV.* Springer, 2019.

[34] K. Soomro, A. R. Zamir, and M. Shah. *UCF101: A dataset of 101 human actions classes from videos in the wild.* arXiv, 2012.

[35] J. Kukačka V. Golkov and D. Cremers. *Regularization for deep learning: A taxonomy.* arXiv, 2017.

[36] S. Hochreiter and J. Schmidhuber. *Long short-term memory.* Neural Comput, 1997.

[37] S. Sharma, R. Kiros, and R. Salakhutdinov. *Action recognition using visual attention.* arXiv, 2015.