Human Task Recognition using CNN

Avula Rohitha, Koneni Madhu Swapnika and Aarthi G

Abstract - In this fast pacing world, computers are also getting better in terms of their performance and speed. It is capable of solving very complex problems like understanding an image, understanding videos and live capturing and processing of data. Due to advancement in technologies like computer vision, machine learning techniques, deep learning methods, artificial intelligence, etc., various models are being made so that prediction of outputs is made simpler and of high accuracy and precision. Our project model is built using a convolutional neural network (CNN). Our dataset consists of 599 videos in which 100 videos was assigned to each category (6 categories: walking, jogging, running, boxing, hand waving and hand clapping). In this project, we have used a set of labelled videos which was used to train our three models. The CNN is used as a base network in all the three models to process all the videos from the dataset, that is, to read all the frames and convert into heat maps. Logistic regression, K-means clustering and SVM were the algorithms which were implemented in our models. Out of our three models, the best model is used to give prediction for the actions performed in the video. The results show that with better algorithm techniques, the performance of the model is also improved.

***Index Terms* – artificial intelligence, convolutional neural network, deep learning techniques, machine learning techniques.**

I. Introduction

Research in Human Activity Recognition is in massive demand due to its applications in Health care domain, Computer Vision, Household safety and Robot Learning. Human Activity Recognition (HAR) aims to identify the actions carried out by a person given a set of observations of him/her and the surrounding environment.

Human activity recognition (HAR) is a widely studied computer vision problem. Applications of HAR include video surveillance, health care, and human-computer interaction. As the imaging technique advances and the camera device upgrades, novel approaches for HAR constantly emerge. Our project aims to provide a comprehensive introduction to the video-based human activity recognition, giving an overview of various approaches as well as their evolutions by covering both the representative classical literatures and the state-of-the-art approaches.

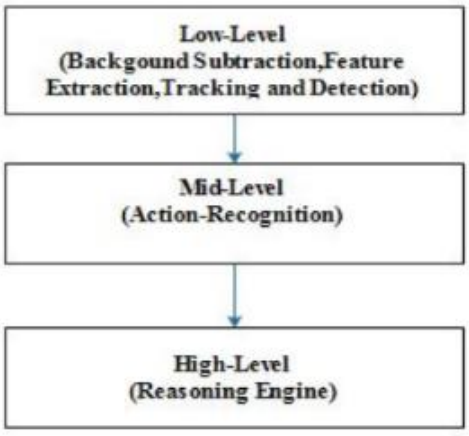

Fig. 1. General framework of Human Activity Recognition.

A. Convolution Neural Networks

Convolutional Neural Networks (CNNs) use weighted sharing, down sampling, and local connection techniques that greatly reduce the number of required parameters and the complexity of the neural network. CNNs have been compared to traditional methods of image feature extraction such as the Histogram of Oriented Gradient (HOG) and Scale-Invariant Feature Transform (SIFT) methods; however, CNNs can typically extract more abstract and comprehensive features. In addition, CNNs avoid the need for complex image preprocessing because they can use the original images directly as input.

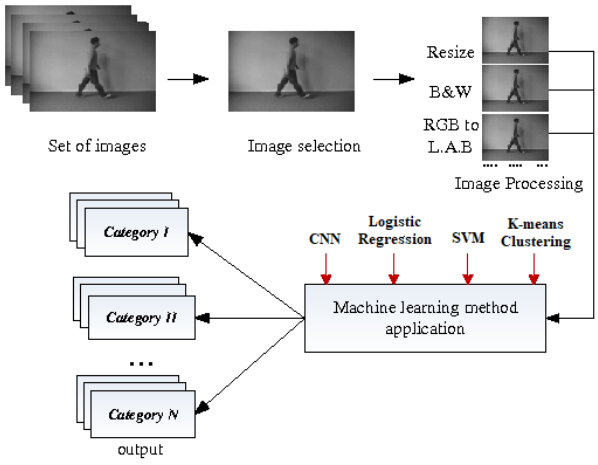

Fig. 2. Architecture of our proposed model.

CNNs are mainly composed of a convolutional layer, a pooled layer, and a fully connected layer. The convolutional layer is a key part of the CNN. The function of this layer is to extract features from input images or feature maps. Each

convolutional layer can have multiple convolution kernels, which are used to obtain multiple feature maps. The convolution layer is calculated as follows:

$$x_j^l = f\left(\sum x_i^{l-1} * k_{ij}^l + b_j^l \right)$$

where x_i^{l-1} is the characteristic map of the output of the previous layer, x_j^l is the output of the ith channel of the jth convolution layer, and f(.) is called the activation function. Here, M_j is a subset of the input feature maps used to calculate u_j^l, k_{ij}^l is a convolution kernel, and b_j^l is the corresponding offset.

A filter is convolved over both x-axis and y-axis. Multiple filters are used in order to extract different patterns from the image. The output of one filter when convolved throughout the entire image generates a 2-d layer of neurons called a feature map. Each filter is responsible for one feature map. These feature maps can be stacked into a 3-d array, which can then be used as the input to the layers further. This is performed by the layer known as Convolutional layer in a CNN. These layers are followed by the Pooling layers, that reduce the spatial dimensions of the output (obtained from the convolutional layers).

In short, a window is slid in both the axes and the max value in that filter/window is taken (Max Pooling layer). Sometimes Average pooling layer is also used where the only difference is to take the average value within the window instead of the maximum value. Therefore, the convolutional layers increase the depth of the input image, whereas the pooling layers decreases the spatial dimensions (height and width). The importance of such an architecture is that it encodes the content of an image that can be flattened into a 1-dimensional array.

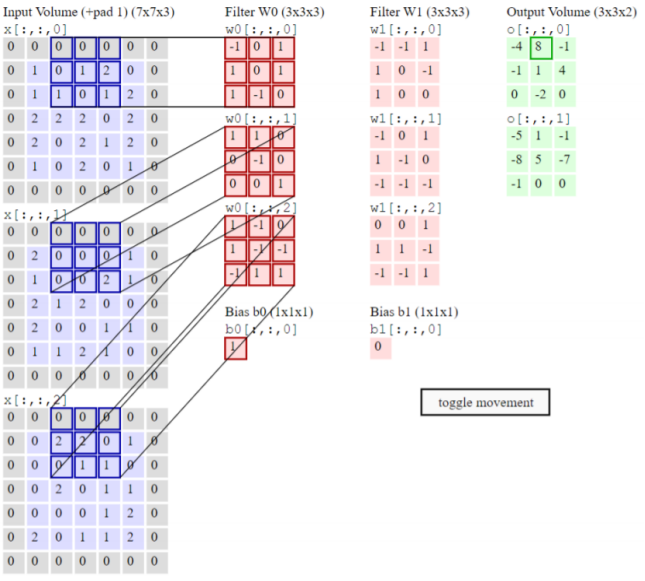

Fig. 3. Convolution of kernels are done with the help of window/filter to produce heat maps. (neurons are created on various hidden layers using this process)

We discussed how CNNs can be used in case of images. What we use is specifically known as 2-d convolutional layers and pooling layers. It's 2-dimensional because the filter is convolved along the x-axis and y-axis of the image. But in case of a video, we have an additional temporal axis – z-axis. So, a 3-d convolutional layer is used – where the filter (also 3-dimensional) is convolved across all the three axes. Multiple convolutional and pooling layers are stacked together. These are followed by some fully-connected layers, where the last layer is the output layer. The output layer contains 6 neurons (one for each category). The network gives a probability of an input to belong to each category/class as shown in Fig. 1.

B. Logistic Regression

Logistic regression produces a logistic curve, which is limited to values between 0 and Logistic regression is similar to a linear regression, but the curve is constructed using the natural logarithm of the "odds" of the target variable, rather than the probability. Moreover, the predictors do not have to be normally distributed or have equal variance in each group.

In the logistic regression the constant (b0) moves the curve left and right and the slope (b1) defines the steepness of the curve. By simple transformation, the logistic regression equation can be written in terms of an odds ratio.

$$p / (1 - p) = \exp (b0 + b1x)$$

Finally, taking the natural log of both sides, we can write the equation in terms of log-odds (logit) which is a linear function of the predictors. The coefficient (b1) is the amount the logit (log-odds) changes with a one-unit change in x.

$$\ln(p / (1-p)) = b0 + b1x$$

As mentioned before, logistic regression can handle any number of numerical and/or categorical variables.

$$p = 1 / (1 + e^{-(b0 + b1x1 + b2x2 + \ldots + bpxp)})$$

There are several analogies between linear regression and logistic regression. Just as ordinary least square regression is the method used to estimate coefficients for the best fit line in linear regression, logistic regression uses maximum likelihood estimation (MLE) to obtain the model coefficients that relate predictors to the target. After this initial function is estimated, the process is repeated until LL (Log Likelihood) does not change significantly.

$$\beta^1 = \beta^0 + [X^T W X]^{-1} . X^T (y - \mu)$$

β is a vector of the logistic regression coefficients.
W is a square matrix of order N with elements $ni\pi i (1 - \pi i)$ on the diagonal and zeros everywhere else.
μ is a vector of length N with elements $\mu i = ni\pi i$.

C. K- Means Clustering

K-means clustering is one of the simplest unsupervised learning algorithms that solve the well-known clustering problem. The procedure follows a simple and easy way to

classify a given data set through a certain number of clusters (assume k clusters) fixed a priori. The main idea is to define k centroids, one for each cluster. These centroids should be placed in a cunning way because of different location causes different result. So, the better choice is to place them as much as possible far away from each other.

The next step is to take each point belonging to a given data set and associate it to the nearest centroid. When no point is pending, the first step is completed and an early groupage is done. At this point we need to re-calculate k new centroids as barycenters of the clusters resulting from the previous step. After we have these k new centroids, a new binding has to be done between the same data set points and the nearest new centroid. A loop has been generated. As a result of this loop we may notice that the k centroids change their location step by step until no more changes are done.

In other words, centroids do not move any more. Finally, this algorithm aims at minimizing an objective function, in this case a squared error function. The objective function

$$J = \sum\sum || X_i^{(j)} - C_j ||^2 ,$$

where $|| X_i^{(j)} - C_j ||^2$ is a chosen distance measure between a data point $X_i^{(j)}$ and the cluster centre C_j, is an indicator of the distance of the n data points from their respective cluster centres.

The algorithm is composed of the following steps:

1) Place K points into the space represented by the objects that are being clustered. These points represent initial group centroids.

2) Assign each object to the group that has the closest centroid.

3) When all objects have been assigned, recalculate the positions of the K centroids.

4) Repeat Steps 2 and 3 until the centroids no longer move. This produces a separation of the objects into groups from which the metric to be minimized can be calculated.

Although it can be proved that the procedure will always terminate, the k-means algorithm does not necessarily find the most optimal configuration, corresponding to the global objective function minimum. The algorithm is also significantly sensitive to the initial randomly selected cluster centers. The k-means algorithm can be run multiple times to reduce this effect.

D. Support Vector Machine

SVM is one of the most popular, versatile supervised machine learning algorithms. It is used for both classification and regression task [1]. Hyperplanes are decision boundaries that help classify the data points. Data points falling on either side of the hyperplane can be attributed to different classes. Also, the dimension of the hyperplane depends upon the number of features. If the number of input features is 2, then the hyperplane is just a line. If the number of input features is 3, then the hyperplane becomes a two-dimensional plane. It becomes difficult to imagine when the number of features exceeds 3. Hyperplane is line divides data point into two classes, written as:

$$y = a * x + b$$
$$a * x + b – y = 0$$

Let vector X = (x,y) and W = (a,-1) then in vector form, hyperplane is

$$W . X + b = 0$$

Support vectors are data points that are closer to the hyperplane and influence the position and orientation of the hyperplane. Using these support vectors, we maximize the margin of the classifier. Deleting the support vectors will change the position of the hyperplane. These are the points that help us build our SVM. Length of vectors are also called as norms. It tells how far vectors are from the origin. Length of vector x (x_1 , x_2 , x_3) is calculated as:

$$|| x || = \sqrt{ (x_1^2 + x_2^2 + x_3^2)}$$

II. LITERATURE SURVEY

In this section literature review is conducted for different types of activity recognition in different environments. In this survey we have used almost the same way as used by Singh et al. [2]. All the review papers are elaborated below in the following section: In [3] Sunyoung Cho et al. proposed method of human recognition by the use of annotation tags in videos and was able to improve recognition of humans from the video. The method consists of framework consisting of visual content and tags for activity recognition. And this framework found very difficult to differentiate between high fives and handshakes, or hugs and kisses, as they are having similar actions. Finally, activity class is made by evaluating classification scores from visual information and tags.

In [4] Ru Ke et al. analyzed the review on ongoing progress created towards the video primarily based act recognition which incorporates single, multiple, crowd behavior and non-normal activities by mistreatment technologies like feature extraction, object segmentation. And the most common domain space for his or her review was surveillances, tending and diversion environment. In [5] Muhammad Shahzad et al. proposed not only on a single recognition of actions but also about the style of an actor. Hierarchical approach supported typical action recognition and uneven linear modeling is employed. IXMAS knowledge set consists of multi-actor multi-action data set is employed for computations. And were able to succeed terribly high accuracy on a typical action recognition knowledge set by mistreatment options invariant to location, scale and rotation.

In [6] Bo Zhang et al. proposed a framework for analyzing human interactions in unconstrained videos such as TV shows. User generated videos are exponentially

growing because of video sharing communities such as Vimeo and YouTube. These videos consist of persistent changes of camera angle, multiple persons, fast body movements. Due to which a newer approach is used which consists of MIP. Motion Interchange Pattern is employed to sight camera viewpoint changes in an exceedingly video, and to extract the salient motion points within the bounding box and to urge region of interest (ROI) in every frame. eventually Self Similarity Matrix (SSM) is created by computing the big displacement optical flow for the salient pixels within the bounding box. we tend to exploit MIP to sight camera viewpoint changes in TV human interaction videos and construct a motion mask to extract the salient motion points in every frame of the video.

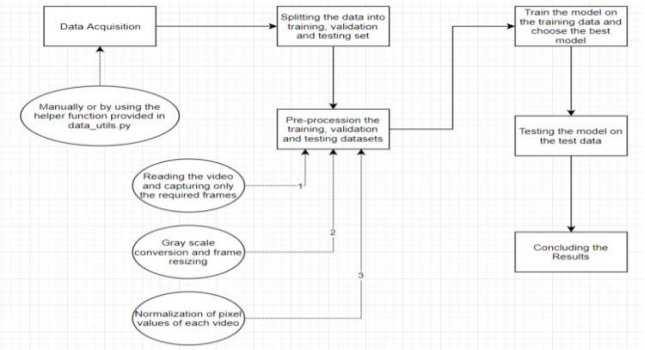

Fig. 4. The proposed human task recognition framework of our model.

In [7] Robert Bodor et al. drafted the concept for automated "smart video" system which aims to track pedestrians and for detection of people under panic situations or there is suspicious motion or activities at or near critical transportation assets. The protection of assets such as flyovers, bridges, tunnels and dams from attacks is hot area of research. The tracking of pedestrians done by the evolution of velocity and position path using a Kalman filter The software tracks by using vision algorithms to classify the motion and activities of each pedestrian or individual pedestrians whenever they come to the field of the camera. In [8] Oscar D. Lara et al. proposed the mobile platform to produce period human action recognition. It consists of a MECLA library which is used for classification algorithms analysis and a mobile application for period human action recognition running at intervals over Body Area Network. It will be enforced in real eventualities meeting accuracy, latency, and energy consumption necessities. The accuracy of the C4.5 classifier is additionally satisfactory, verifying the use of this rule during this and for different applications.

In [9] Chiro Kobayashi et al. proposed the method of retrieving the particular human behaviors by using words in the movie. The problem of finding the particular scene in the video consisting of human actions is very challenging task. To overcome this problem verbalizing of human behaviors in a movie with image recognition technology is done. Several image processing techniques, such as background subtraction, dilation, and to find objects in the space are used to recognize in the videos. In [10] Abdalrahman Eweiwi et al. presented the study of Human Action Recognition using Temporal Key Poses authors proposed simple and also effective approach to recognition activities of humans from videos. And the author uses the approach of human activity recognition in static pictures mistreatment action key poses and motion-based approaches mistreatment the variants of Motion History pictures (MHI) and Motion Energy Images (MEI). a bunch of temporal templates were used for various actions by playacting k-means cluster on cause templates and extract key cause templates. MuHAVi information set for action recognition is taken into account for task.

In [11] W. Lu et al. proposed a method to track the person of interest and to recognize action of that person. Interest region is tracked by using the previous time template to estimate the current state of player. The recognition of actions can be estimated by calculating Principle Component Analysis Histogram of Oriented Gradient (PCA-HOG). It is the descriptor for the tracking area in the frames and Maximum Likelihood Estimation of previous observations. The experiments are analysed for two games soccer and hockey. 10 possible templates for each action are used in soccer and hockey game. In [12] M. S. Ryoo et al. presented the novel approach for detection of human activities and prediction. The main goal is to develop early recognition of unfinished activities as against the after-the-fact classification of completed activities by representing the activity as associate degree integral bar chart of spatio temporal options. Dynamic bag-of-words methodology is used to detect human activities at an early stage. The results of implementation guarantee that the proposed approach is able to recognize on human activities and their interactions at much earlier time than the previous methods.

In [13] Nacim et al. proposed work on real-time crowd motion analysis. With increasing demand of autonomous video surveillance, a system is proposed to detect abnormal activities of crowd in public areas. Motion heat map is computed of the image. The motion heat map computes hot and cold regions for images. The hot regions in the frame represented the region with high motion and cold region represented region with low motion intensities. Escalator of airport is used as dataset for analysis of videos. In [14] Rohit Nair et al. proposed the Intelligent Detection Techniques for Activities in HD Video Surveillance Systems. It is used for suspicious activity detection and human fall detection, for each indoor and outside environment. The enforced model captures and analyzes live high-definition (HD) video that's streamed from a far-off camera. The suspicious activity detection methodology like automobile awaiting long term.

III. SCOPE

Technically, the model would have to learn to differentiate between various human actions, given some examples of these actions. There are potentially a lot of applications of video recognition such as:

- Real-time tracking of an object - This could be very helpful for tracking the location of an object (like a vehicle) or a person from the video recorded by a CCTV.
- Learning the patterns involved in the movement of humans - If we are able to create a model that can learn how we (humans) perform various activities (like walking, running, exercising etc.), we can use this model for proper functioning of the movement mechanisms in autonomous robots.
- It can be useful for creating applications dumb people so that they can easily communicate through actions.

IV. DATASET

Our dataset contains six types of human actions (walking, jogging, running, boxing, hand waving and hand clapping) performed several times by 25 subjects in four different scenarios: outdoors denoted as s1, outdoors with scale variation denoted as s2, outdoors with different clothes denoted as s3 and indoors denoted as s4. Currently the database contains 2391 sequences and all sequences were taken over homogeneous backgrounds with a static camera with 25fps frame rate. The sequences were down sampled to the spatial resolution of 160x120 pixels and have a length of four seconds in average.

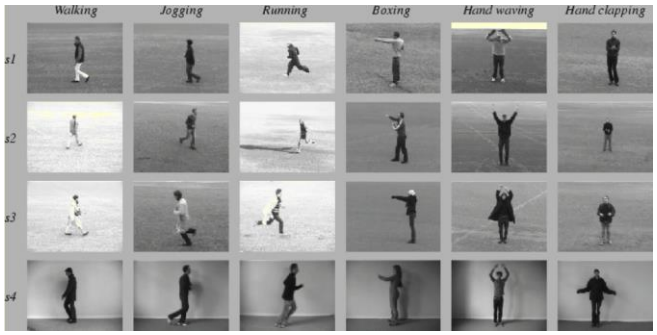

Fig. 5. Image frames of six categories present in the dataset (walking, jogging, running, boxing, hand waving and hand clapping).

The dataset contains 599 videos – 100 videos for each of the 6 categories (with the exception of Handclapping having 99 videos). In our experiments all sequences were divided with respect to the subjects into a training set (8 persons), a validation set (8 persons) and a test set (9 persons). The classifiers were trained on a training set while the validation set was used to optimize the parameters of each method. While loading the data, we convert these text labels into integers according to the table 1.

TABLE 1: CONVERTING TEXXT LABELS INTO INTEGERS

Boxing	0
Hand clapping	1
Hand waving	2
Jogging	3
Running	4
Walking	5

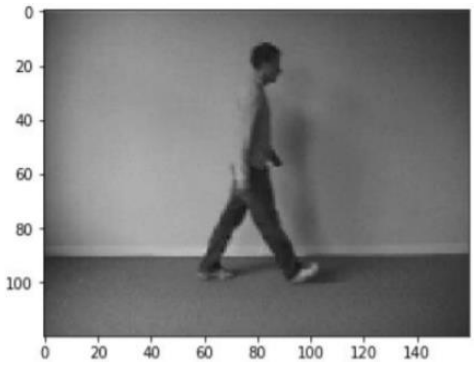

Fig. 6. Image frame from a sample video

Fig. 2. is a single frame from a sample video of walking. It can be observed that the spatial dimensions of the video (width x height) are 160 x 120 pixels. Also, on loading a single video into a NumPy array in python, the shape of the array obtained was – (1, 515, 120, 160, 3) This indicates that:

- There is 1 video
- The video has 515 frames
- The spatial dimension of the video is 160 x 120 (width x height) pixels
- Each frame has 3 channels – Red(R), Green(G)

  and Blue(B) A similar methodology would be used for reading in the entire dataset.

There is a helper class called Videos in utils.py for reading the videos into numpy ndarrays. The class provides some additional functionalities like:

- Setting the target size for each frame of a video
- Conversion each frame to gray scale
- Various options to extract a subset of frames from each video
- Normalizing the pixel values of each video

We will use this to load out training, validation and test data.

V. EXPERIMENT SETUP

A. Data pre-processing

Reading in the video frame-by-frame. The videos were captured at a frame rate of 25fps. This means that for each second of the video, there will be 25 frames. We know that within a second, a human body does not perform very significant movement. This implies that most of the frames (per second) in our video will be redundant. Therefore, only a subset of all the frames in a video needs to be extracted. This will also reduce the size of the input data which will in

turn help the model train faster and can also prevent over-fitting.

Class Label	Mapped Integer
Boxing	0
Handclapping	1
Handwaving	2
Jogging	3
Running	4
Walking	5

Before One-hot Encoding

Class Label	0	1	2	3	4	5
Boxing	1	0	0	0	0	0
Handclapping	0	1	0	0	0	0
Handwaving	0	0	1	0	0	0
Jogging	0	0	0	1	0	0
Running	0	0	0	0	1	0
Walking	0	0	0	0	0	1

After One-hot Encoding

Fig. 7. Left: Image of text labels mapped to integer before one-hot encoding. Right: The adjacency matrix is created after one-hot encoding.

Different strategies would be used for frame extraction like: Extracting a fixed number of frames from the total frames in the video – say only the first 200 frames (i.e., first 8 seconds of the video). Extracting a fixed number of frames each second from the video – say we need only 5 frames per second from a video whose duration is of 10 seconds. This would return a total of 50 frames from the video. This approach is better in the sense that we are extracting the frames sparsely and uniformly from the entire video.

Each frame needs to have the same spatial dimensions (height and width). Hence each frame in a video will have to be resized to the required size. In order to simplify the computations, the frames are converted to grayscale.

B. Normalization

The pixel values range from 0 to 255. These values would have to be normalized in order to help our model converge faster and get a better performance. Different normalization techniques can be applied such as: o Min-max Normalization – Get the values of the pixels in a given range (say 0 to 1) o Z-score Normalization – This basically determines the number of standard deviations from the mean a data point is. We would finally get a 5-dimensional tensor of shape – (, , , ,) - 'channels' can have the value 1 (grayscale) or 3 (RGB) - 'number of frames' - the extracted frames (will have to be the same for each video) Also, the categorical labels should be encoded using a technique called One-hot Encoding. One-hot Encoding converts the categorical labels into a format that works better with both classification and regression models as shown in Fig. 3.

C. Loading data

One of the most important part of the project was to load the video dataset and perform the necessary pre-processing steps. So, we developed a class (Videos) that had a function called (read_videos()) that can be used to for reading and processing videos. Creating this was very challenging as we concentrated on generalizing this function for any kind of videos (not specific to this project. We have used NumPy (wherever) for storage and processing of the videos (much faster than in-built python lists with a ton of extra functionalities).

D. Refinement and Testing

MODEL 1

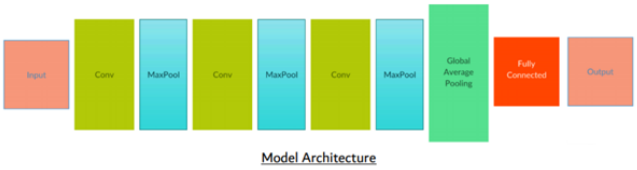

Fig. 8. Model 1 architecture

The model was trained on the training data for 40 epochs. The weights of the model which gave the best performance on the validation data were loaded. The model was then tested on the test data.

MODEL 2

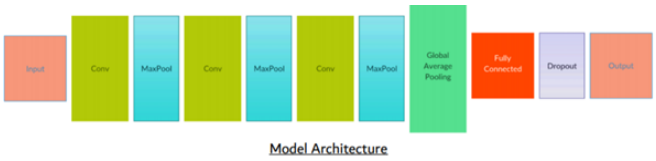

Fig. 9. Model 2 architecture

In order to prevent overfitting, there is a method called Dropout. What this does is that at each epoch, a fraction of the neurons (of the layer to which dropout is applied) are deactivated. This forces the network to use and update the weights of the remaining neurons. The dropout is applied to the fully-connected layers.

Note: Dropout is used only while model training, and not during the testing. In this model, I've added a dropout for the hidden layer (with 32 neurons) in the fully-connected layers. Rest of the model is same as Model-1.

The model was trained on the training data for 40 epochs. The weights of the model which gave the best performance on the validation data were loaded. The model was then tested on the test data.

MODEL 3

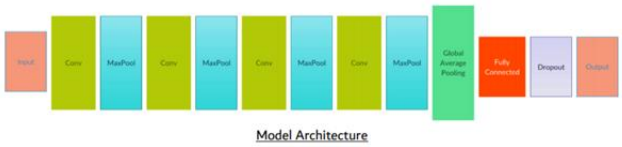

Fig. 10. Model 3 architecture

Till now, 200 frames for each video were extracted and given as the input to the models. But the approach to extract these frames is not very appropriate. What was being done is that from each video, 200 contiguous frames (8 seconds) were being extracted. We know that the human body performs these activities (running, boxing etc.) with a

certain speed. Within one second, the human body does not make much of a movement. Therefore, we do not need to collect every frame for each second of video that we are capturing. A different approach could be used, where only a certain number of frames are extracted for each second.

Now, we would be extracting only 5 frames per second (first 5 frames for each second). So, suppose we have a video of 10 seconds, we will get 10 x 5 = 50 frames. There is also a maximum limit on the number of frames that should be extracted from each video. I have set this value to 40. So, these 40 frames will be selected from the front of the extracted frames.

The range of normalized pixels has also been changed from [0, 1] to [-1, 1]. This is because the mean of the pixels would then be 0, which would help the model converge faster. The model was trained on the training data for 40 epochs. The weights of the model which gave the best performance on the validation data were loaded. The model was then tested on the test data.

VI. RESULTS

Training model 1: Here, the model (Model-1) is being trained on the training data. For each epoch (iteration), the model is being validated using the validation data. The model is trained for 40 epochs. Also, the model (model's weights) that performed the best on the validation set is being saved in a file.

Evaluating model 1: The best model weights are loaded and then the model is evaluated on the test data. We have used accuracy as the metric to test the model's performance on the test data.

Training Loss vs Validation Loss:

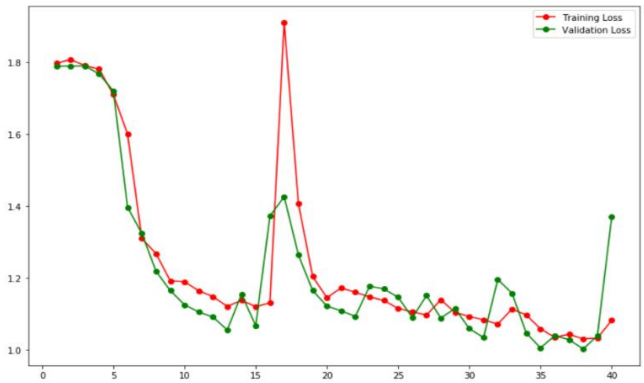

Fig. 11. Learning curve of model 1.

As we can observe from the learning curve in Fig. 8, during the first 15 epochs, the training and validation loss both decrease steeply. But after that, the model starts to overfit a little. Here, overfitting means that although the model performed better on the training data, but its performance on the validation data got degraded. This usually happens when our model is too complex for the data and it starts to memorize the training data. We can see that during the last 5 epochs, there is a huge difference between the training and validation loss. The model gave an accuracy of 37% on the test data.

Training model 2: Here, the model (Model-2) is being trained on the training data. For each epoch (iteration), the model is being validated using the validation data. The model is trained for 40 epochs. Also, the model (model's weights) that performed the best on the validation set is being saved in a file.

Evaluating model 2: The best model weights are loaded and then the model is evaluated on the test data. We have used accuracy as the metric to test the model's performance on the test data.

Training Loss vs Validation Loss:

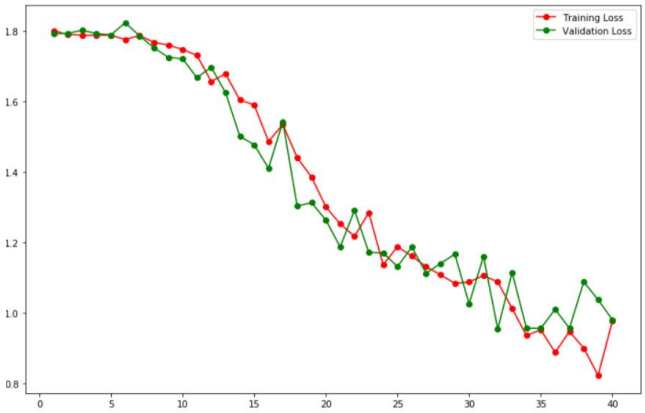

Fig. 12. Learning curve of model 2.

There is a gradual decrease in both the training and validation loss. Also, the difference between the training and validation loss is not very large, suggesting that our model is no longer overfitting the training data. The model gave an accuracy of 58.5% on the test data. So, just by adding a dropout layer, the model's accuracy increased by almost 22%. This shows that the dropout prevented our model from overfitting. We can see this in the learning curve of this model in Fig. 9.

Training model 3: Here, the model (Model-3) is being trained on the training data. For each epoch (iteration), the model is being validated using the validation data. The model is trained for 40 epochs. Also, the model (model's weights) that performed the best on the validation set is being saved in a file.

Evaluating model 3: The best model weights are loaded and then the model is evaluated on the test data. We have used accuracy as the metric to test the model's performance on the test data.

Training Loss vs Validation Loss:

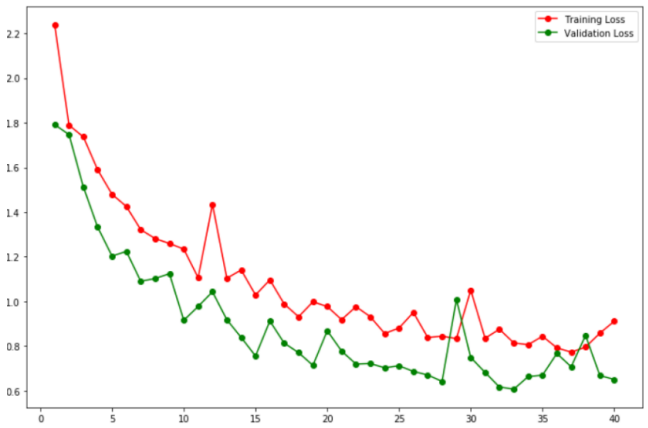

Fig. 13. Learning curve of model 3.

The model gave an accuracy of 64.5% on the test data. This model gave a higher accuracy than the previous models, despite using 5 times lesser data for training. In this model, another pair of convolutional and max pooling layer was added. This made the output of the final convolutional layer to have a depth of 1024 (earlier it was 256) and the learning curve is not steep as shown in the Fig. 10.

Also, this model used NADAM as the optimizer (instead of ADAM). In Keras, the default values of learning rate for ADAM optimizer is set to 0.001. For NADAM, the default value of learning rate is 0.002 and there is a scheduled decay of learning rate. Using NADAM as the optimizer gave better results than ADAM. Also, at the end of 40 epochs, the model did not overfit when the optimizer used was NADAM, but in case of ADAM, the model showed some signs of overfitting.

VII. EXPERIMENT ANALYSIS

Some of the important specifications of the final model:

- The depth of the vector obtained by the last convolutional layer is 1024.
- A Global Average Pooling layer (GAP) then takes the average value from each of these 1024 dimensions and gives a 1-dimensional vector representing the entire video.
- The GAP is followed by a fully-connected layer containing 32 neurons. This fully-connected layer also has a dropout of 0.5, meaning that for each epoch, 50% of the neurons of this layer will be deactivated. This is what helps the model prevent overfitting.
- Finally, there is the output layer with 6 neurons (one for each category). The network gives a probability for the input video to belong to each of the 6 categories.
- All the convolutional layers have 'ReLU' as the activation function. It gives the best performance out of a CNN.

Hence Model 3 is preferred for identification of human activities.

Fig. 14. Sequence of frames of running action being detected by our model.

Fig. 15. Sequence of frames of jogging action being detected by our model.

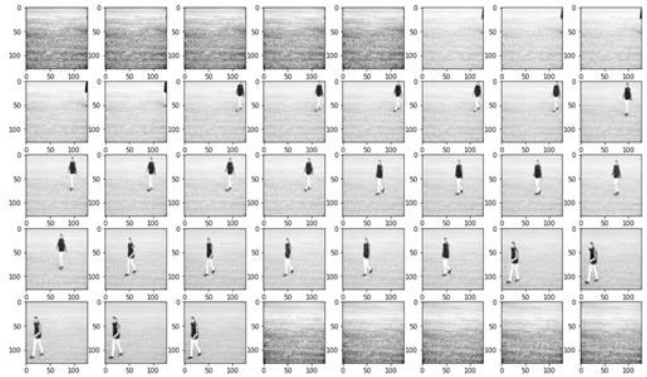

Fig. 16. Sequence of frames of walking action being detected by our model.

VIII. CONCLUSION

Human activity recognition remains to be an important problem in computer vision. HAR is the basis for many applications such as video surveillance, health care, and human-computer interaction. Methodologies and

technologies have made tremendous development in the past decades and have kept developing up to date. However, challenges still exist when facing realistic sceneries, in addition to the inherent intraclass variation and interclass similarity problem.

IX. FUTURE WORK

In the future, these works can be implemented:

1) A potential model that can give a much better performance than our current model can be used. The part where our model is lagging is that it is not able to extract features from the video and convert it into a 1-d vector, without losing much information. If we are able to use the concept of transfer learning in order to extract featured from the videos, it would give much better results – given that the model used is pre-trained on some similar dataset. But since no pre-trained models exist for video recognition, we can use the following approach: Use a pre-trained model (like InceptionV3 or ResNet) to encode each frame of the video into a 1-d vector. This will give us a sequence of 1-dimensional vectors (each representing a frame).

2) We can use a sequence-to-sequence model (like LSTM) to capture the temporal relationship between adjacent frames. Since this model will extract more information from each video, the performance of such a model might be a lot better than our proposed model.

3) A simple web-application could be developed, where the user can perform some action. This would be captured by a webcam and the model would give real-time predictions of the action being performed.

REFERENCES

[1] M. S. Ryoo, "Human activity prediction: Early recognition of ongoing activities from streaming videos," in ICCV, 2011.

[2] M. Ryoo and J. Aggarwal, "Spatio-temporal relationship match: Video structure comparison for recognition of complex human activities," in ICCV, 2009, pp. 1593–1600

[3] S. Singh, S. A. Velastin, and H. Ragheb, "Muhavi: A multicamera human action video dataset for the evaluation of action recognition methods," in Advanced Video and Signal Based Surveillance (AVSS), 2010 Seventh IEEE International Conference on. IEEE, 2010, pp. 48–55

[4] S. Ji, W. Xu, M. Yang, and K. Yu, "3d convolutional neural networks for human action recognition," IEEE Trans. Pattern Analysis and Machine Intelligence, 2013.

[5] J. Hou, X. Wu, J. Chen, J. Luo, and Y. Jia, "Unsupervised deep learning of mid-level video representation for action recognition," in AAAI, 2018.

[6] J. Sung, C. Ponce, B. Selman, and A. Saxena, "Human activity detection from rgbd images," in AAAI workshop on Pattern, Activity and Intent Recognition, 2011.

[7] J. L. Jingen Liu and M. Shah, "Recognizing realistic actions from videos "in the wild"," in CVPR, 2009

[8] I. C. Duta, B. Ionescu, K. Aizawa, and N. Sebe, "spatio-temporal vector of locally max pooled features for action recognition in videos," in CVPR, 2017.

[9] L. Wang, Y. Qiao, and X. Tang, "Action recognition with trajectorypooled deep-convolutional descriptors," in CVPR, 2015

[10] K. Jia and D.-Y. Yeung, "Human action recognition using local spatiotemporal discriminant embedding," in CVPR, 2008.

[11] C. Fanti, L. Zelnik-Manor, and P. Perona, "Hybrid models for human motion recognition," in CVPR, 2005.

[12] S. Rajko, G. Qian, T. Ingalls, and J. James, "Real-time gesture recognition with minimal training requirements and on-line learning," in CVPR, 2007.

[13] R. Blake and M. Shiffrar, "Perception of human motion," Annu. Rev. Psychol., vol. 58, pp. 47–73, 2007.

[14] D. Tran and A. Sorokin, "Human activity recognition with metric learning," in ECCV, 2008.

[15] H. S. Koppula and A. Saxena, "Learning spatio-temporal structure from rgb-d videos for human activity detection and anticipation," in ICML, 2013.