Human Task Recognition using CNN

Avula Rohitha, B.K. Hem Charan

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 of basic human actions like Running, Boxing, walking etc. 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. 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.

Keywords: Computer Vision, Heat maps, Accuracy, Precision, Artificial Intelligence, Machine learning, Deep learning, Performance, Convolutional Neural network.

I. INTRODUCTION

Research in Human Task Recognition is in massive demand due to its applications in Health care domain, Computer Vision, Household safety and Robot Learning.
The Main Aim of Human Task Recognition (HTR) is to identify the actions performed out by a person in the surrounding environment.

Human Task recognition (HTR) is a widely Researched area in computer vision problem. Some of the applications of HTR are Robotic movements, Video monitoring, health monitoring, and E-health systems. This paper aims to provide a overview of video based human task recognition with 3 different models

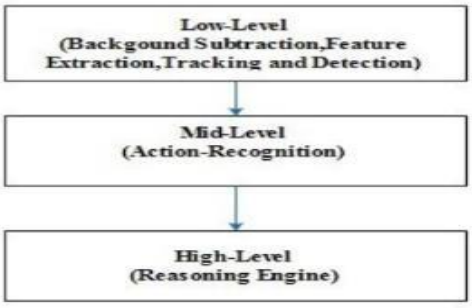

Fig. 1. General framework of Human Activity Recognition.

Revised Manuscript Received on September, 2020.

Avula Rohitha, School of Electronics Engineering (SENSE), Vellore Institute of Technology, Chennai (Tamil Nadu), India Email: rohithaavula15@gmail.com

B.K. Hem Charan, School of Electronics Engineering (SENSE), Vellore Institute of Technology, Vellore (Tamil Nadu), India Email: hemcharanbagul809@gmail.com

A. Convolution Neural Networks

To reduce the complexity and the number of parameters required, down sampling and weighted sharing are used by Convolutional Neural Networks (CNNs).

Traditional Methods of image feature extraction have been compared to CNN; however, a greater number of features are extracted by CNN. Additionally, CNNs can use the original images directly as input this avoids the need for complex image processing.

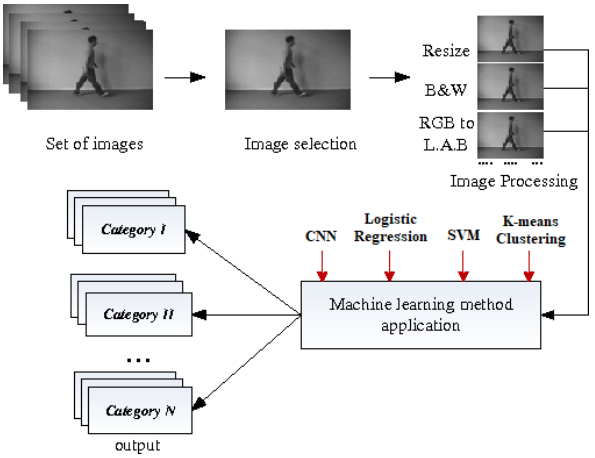

Fig. 2. Architecture of our proposed model.

The 3 main layers of CNN are convolutional layer, fully connected layer and a pooled layer. The crucial part of CNN is convolutional layer, it extracts feature from given image or feature maps. Multiple feature maps can be obtained by several convolution kernels.
The convolution layer is calculated by below equation:

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

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, $_{ij}k^l$ is a convolution kernel, and b_j^l is the corresponding offset.

Both horizontal and vertical axes are convolved over by a filter. To extract different patterns/features from the image multiple filters are used. Feature map is generated when output filter is convolved over entire previous layers of a image. Every filter account for 1 feature map. The 3D array is stacked by feature maps, which helps to provide inputs for the further layers. This function is carried out by Convolutional layer. It is accompanied by pooling layers, which reduces the spatial dimension. Max pooling and Average pooling layer takes the max value and average value in that filter respectively. The content of image is encoded into 1D array and this the significance of this architecture.

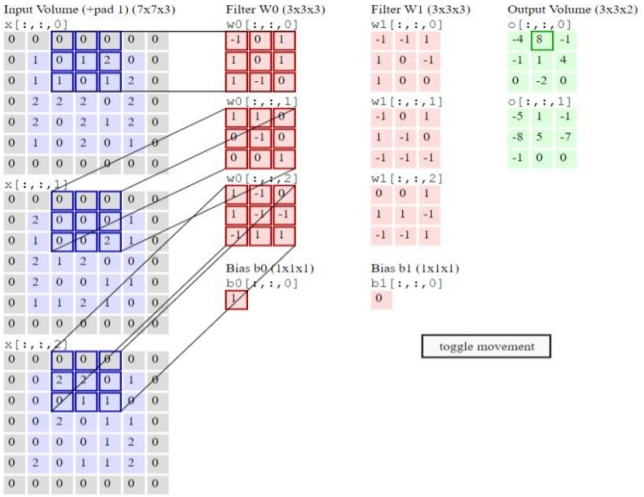

Fig. 3. Convolution of kernels are done with the help of window/filter to produce heat maps. (neurons are created on various hidden layers using this process)

Here we discussed about how CNNs are used in images.

We commonly use 2D convolution Neural network, filter is convolved along both horizontal and vertical axes of the image, but in video a temporal axis- z-axis is added so we use 3D convolution neural network. In 3D convolutional Neural network, filter is convolved across all the 3 axes. Pooling layers and other multiple convolutional layers are stacked together along with a last fully connected layer called output layer; it consists of 6 neurons. Each neuron is assigned for one category.

II. PROPOSED MODEL

This paper proposes a framework for Human Task Recognition, as shown in Fig 4. Videos are running sequences of images. Each image is called as Frame. In day-to-day life, user generated videos are increasing drastically due to increase in Video sharing communities. These videos consist of Multiple persons and their movements. So, my main goal is to develop a model with an approach that are able to achieve good overall recognition rate and fundamentally different from existing models. The primary model used here is Convolutional Neural Network and the input to the model will be a 3D stack of frames. These labelled inputs are trained by the model and some preprocessing techniques must be done before we send the data for training or testing.

These are explained in below sections.

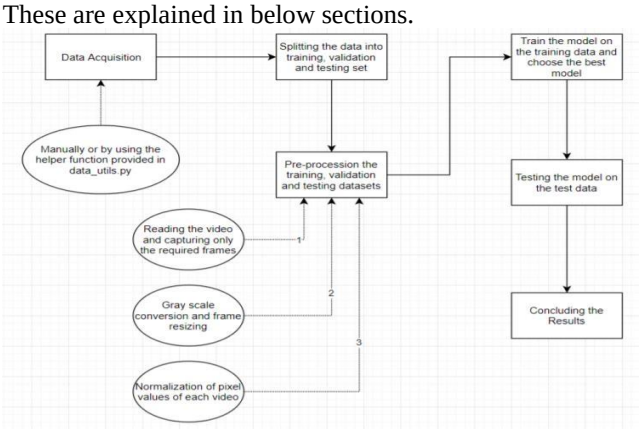

Fig. 4. The proposed human task recognition framework of our model.

III. SCOPE

Technically, the model needs to learn to differentiate between various human actions, given some samples of these actions. There are lot of applications of video recognition such as:

- Real-time tracking of location of an object – This is very helpful for tracking location of a person or an object from the CCTV video.
- Learning the patterns involved in the movement of humans – By creating a model that can learn different human actions, we can use this model in proper functioning of autonomous robots' mechanism.
- This mechanism can be used for dumb people to easily communicate through actions.

IV. DATASET

Our dataset consists of six different types of human actions performed in four different scenarios (s1, s2, s3, s4).

The database contains about 2391 sequences with a frame rate of 25fps. To have a average of 4 seconds and spatial resolution of 160x120 pixels all sequences are down sampled.

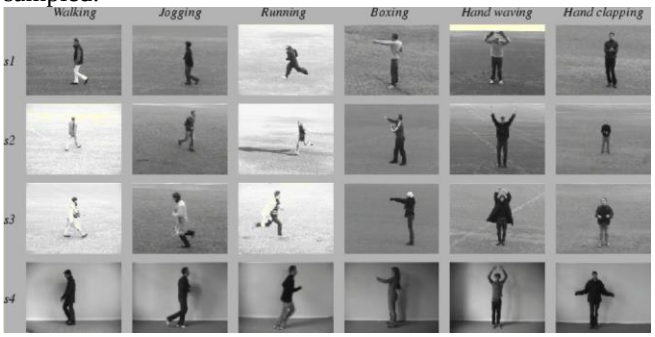

Fig. 5. Image frames of six categories present in the dataset (walking, jogging, running, boxing, hand waving and hand clapping).

The dataset contains 100 videos for each category (5 categories) and 99 videos for Handclapping with a total of 599 videos. The training set, a validation set and a test set were formed using these sequences. The training set is trained by classifiers, while to optimize the parameters for each method validation set is used. To load the data, we converted these text labels into integers according to the below table 1.

TABLE 1: CONVERTING TEXXT LABELS INTO INTEGERS

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

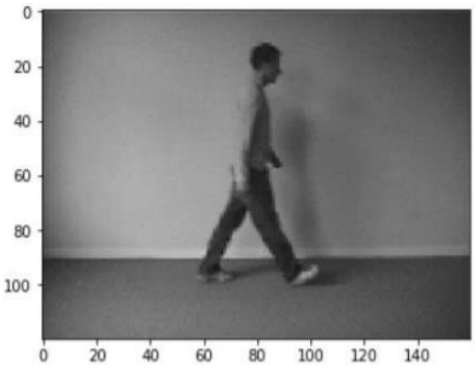

Fig. 6. Image frame from a sample video

Fig. 2. is a single frame from a sample video of walking. From the above figure we can observe that spatial dimensions are 160 x 120 pixels. To load a single video NumPy array is used and shape of an array is – (1, 515, 120, 160, 3) This indicates that:

- There is 1 video
- The video has 515 frames
- Each frame contains 3 channels – Red(R), Green(G) and Blue(B).

For reading entire dataset similar methodology is followed. For reading videos into numpy ndarrays there is a helper class called Videos. Its 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

This is used to load training, validation and test data.

V. EXPERIMENTAL SETUP

A. Data pre-processing

Reading in the video frame-by-frame. All sequences are captured at 25fps, which means there are 25 frames for each second of the video. Most of the frames will be redundant because in seconds of time, human cannot perform very significant movement. Hence, only subset of frames will be extracted from video. This also helps in reducing the size of input which results to train the model faster and also prevents overfitting.

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 methods are used for frame extraction like: Extracting fixed set of frames from the total frames present in the video. Extracting fixed number of frames per second from the video. This method is better strategy for extracting the frames uniformly from the video.

Spatial dimension should be same for each frame. As a result, to acquire required size each frame will have to be resized. Frames are converted into grayscale to simplify the computations.

B. Normalization

The pixel values ranging from 0 to 255 would have to be normalized in order to help our model to get a better performance. Different techniques of normalization can be applied such as: Min-max Normalization – Get the values of pixels ranging from 0 to 1, Z-score Normalization – determines the number of standard deviations from the mean. One hot encoding technique is used to encode the categorical labels, it also converts into a format that works better for classification and regression models as shown in Fig 3.

C. Loading data

To load the dataset and performing the necessary pre-processing steps is one of the important parts of the project. A class named Videos and a function called read_videos() is developed to read and process the videos and we used NumPy for storage and processing.

D. Refinement and Testing

MODEL 1

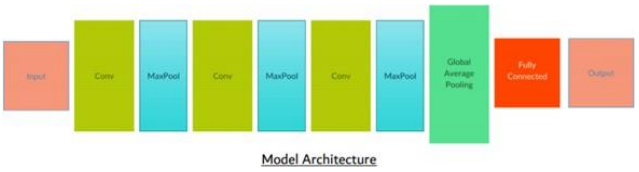

Fig. 8. Model 1 architecture

This model was trained for 40 epochs on training data. The weights which gave best performance on validation were loaded, it is then tested on test data.

MODEL 2

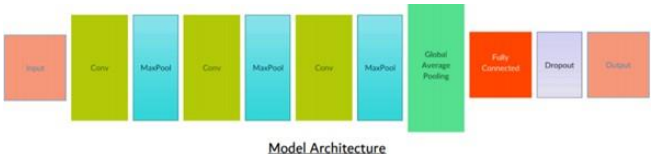

Fig. 9. Model 2 architecture

Dropout method was performed in order to prevent overfitting. This method deactivates a fraction of the neurons, it is performed only while training and not during testing. This method helps in updating the weights of remaining neurons. This is applied to the fully connected layers.

This model was trained for 40 epochs on training data. The weights which gave best performance on validation were loaded, it is then tested on test data.

MODEL 3

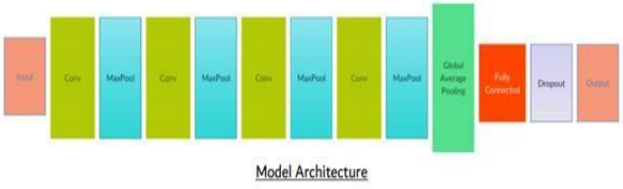

Fig. 10. Model 3 architecture

Till now, only 200 frames are given as the input to the models. But the approach is not appropriate to extract these frames because from each video, 200 contiguous frames were extracted. We know that the human body does not make much of a movement because human body performs activities with a certain speed. Therefore, there is need to extract every frame for each second of video. A different approach is used for Model 3 to collect only certain number of frames from each second, say 5 frames per second were extracted the for 10 seconds, we get 10 x 5 = 50 frames.

The normalization of pixels was also changed from [0, 1] to [-1, 1]. This is to make mean of the pixel to 0, which helps model to converge faster. This model was trained for 40 epochs on training data. The weights which gave best performance on validation were loaded, it is then tested on test data.

VI. RESULTS

Training model 1: In this model training data is being trained. Using validation each epoch is validated. This model was trained for 40 epochs on training data. Also, the weights which gave best performance were saved in a file.
Evaluating model 1: The model is then evaluated on test data. To test the performance accuracy is used.
Training Loss vs Validation Loss:

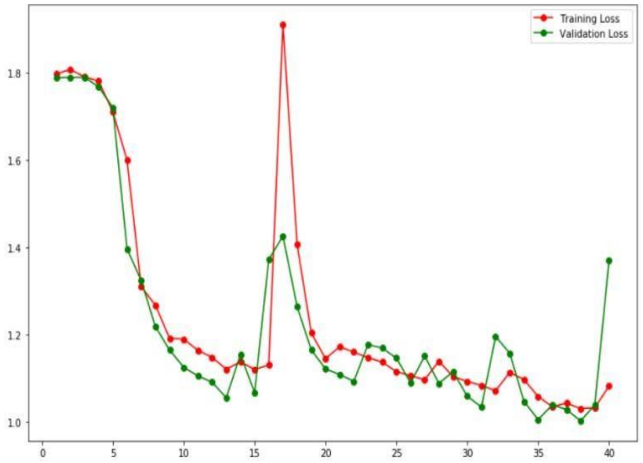

Fig. 11. Learning curve of model 1.

From the learning curve Fig. 8, it can be observed that the training and validation loss both are decreasing steeply during first 15 epochs. After that the model performed better on training but its performance is degraded on validation data which means model starts to overfit. There is a large difference between training and validation loss during the last 10 epochs. This model gave an accuracy of 37%.
Training model 2: In this model (Model -2) training data is being trained. Using validation each epoch is validated. This model was trained for 40 epochs on training data. Also, the weights which gave best performance were saved in a file.
Evaluating model 2 The model is then evaluated on test data. To test the performance accuracy is used.
Training Loss vs Validation Loss:

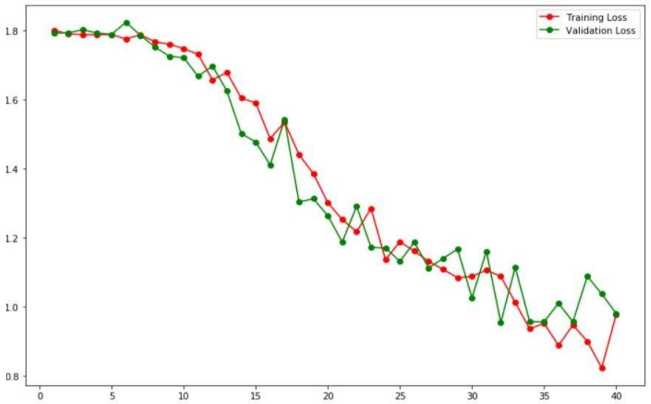

Fig. 12. Learning curve of model 2.

It can be observed from the learning curve that there is a gradual decrease in both training and validation loss. Also, there is no large difference between training and validation loss, implies that there is no overfitting. By adding a dropout layer model -2 gave an accuracy og 58.5% , it is almost increased by 22%.
Training model 3: In this model training data is being trained. Using validation each epoch is validated. This model was trained for 40 epochs on training data. Also, the weights which gave best performance were saved in a file.
Evaluating model 3 The model is then evaluated on test data. To test the performance accuracy is used.
Training Loss vs Validation Loss:

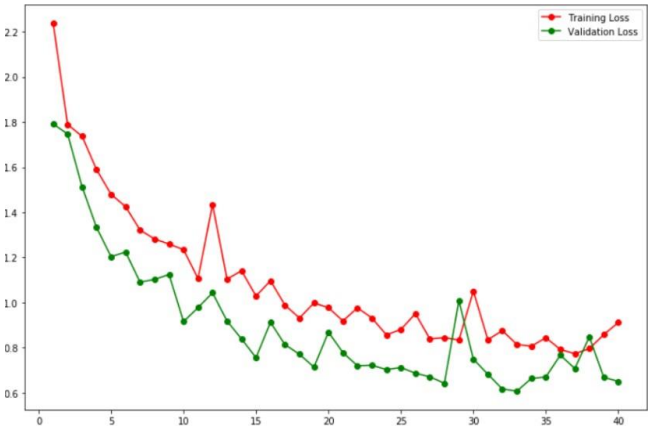

Fig. 13. Learning curve of model 3.

This model gave a higher accuracy of 64.5% than the previous models. Another pair of convolutional and pooling layer was added in this model. It can be observed that learning curve is not steep. This model made the output of last convolutional layer to have a depth of 1024. NADAM is used in this model instead of ADAM. The default values of learning rate of ADAM and NADAM is 0.001 and 0.002.

VII. EXPERIMENT ANALYSIS

Some of the important features of the final model:

- The output of final convolution has a depth of 1024
- From each of these 1024 dimensions, a 1D vector representing entire video is chosen by Average pooling layer which takes the average value.

- To prevent overfitting Dropout method is used. A dropout of 0.5 is produced on fully connected layer, that is 50% neurons gets deactivated.
- 'ReLU' is used as activation function for all the convolutional layers, because it gives the best performance.

Hence Model 3 is preferred for identification of human activities.

VIII. CONCLUSION

The proposed paper presents 3 different models for Human task recognition. Model-3 is preferred for identification of human activities due to its accuracy and some important features.

FUTURE WORK

In the future, these works can be implemented:

1) We can propose a sequence-to-sequence model which captures only important information from each video, the performance of such a model might better than our model.

2) Many web-applications can be developed, where the actions are being captured by webcam and model would give real time predictions of human actions.

3)This model can be applied for movement mechanism of autonomous robots.

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.