Adaptive Frame Extraction and Action Recognition Using Deep Learning

P.V.S.L Jagadamba[1], K.S.S. Harshitha[2], G. Jayasree[3], B.S. Sameera[4], K.A. Varshini[5]

[1]Professor, Department of (CSE),[2345]IV B.Tech,Department of Computer Science and Engineering ,Gayatri Vidya Parishad College of Engineering for Women, Visakhapatnam, India

Email: [1]drpvsljagadamba@gvpcew.ac.in, [2]harshithakammula11@gmail.com, [3]gubbalajaya777@gmail.com , [4]sivasameera123@gmail.com, [5]varshini26vinny@gmail.com

Abstract

Human Activity Recognition aims to recognize human action in a video, separated as a series of frames. Human activities have an inherent hierarchical structure that indicates the different levels of it, which can be considered as a three-level categorization. First, these are action primitives which constitute more complex human activities. After the action primitive level, the action/activity comes as the second level. Finally, the human activities that involve more than two persons and objects. This paper mainly discusses 51 different types of primitive actions. The videos have been analyzed to produce unique dynamic image using key frame extraction method. The obtained frames are dynamic images, utilized to build a Convolutional Neural Network. CNN is a type of Artificial Neural Network used in image recognition and processing which is specifically designed to process pixel data by using max pooling and classifiers. The pixels in the dynamic image focus on identity and motion of actors. As every sector needs automation, this network trains itself to recognize human action. The video provided to test has been split into dynamic images and classified against the model that has been built using training data.

Keywords: *Dynamic images, Deep learning, Convolutional Neural Networks, Key Frame extraction, Max pooling, Activation Function, Fully connected layers, OpenCV, Stride.*

1. Introduction

To recognize activity from a video, the video should be represented as a series of still images. In order to categorize an action in an efficient and accurate manner, features that provide meaningful information must be gathered from images and encoded for classification. Ideally, the representation model should be robust to variation in appearance of the actor(s), background, viewpoint while preserving sufficient information to accurately classify the action. A total of 51

actions were selected for the HMDB51 database, where the actions were broadly categorized into five groups:
1) General facial actions, 2) Facial actions with object manipulation, 3) General body movements, 4) Body movements with object interaction, and 5) Body movements for human interaction. There is a total of 6,766 clips in the HMDB51 dataset with each action containing at least 102 clips. To test the strengths and weaknesses in context of various nuisance factors, each video is annotated with a meta tag, which provides information like camera viewpoint, presence/absence of camera motion, video quality, number of actors involved in the action, and visible body part.

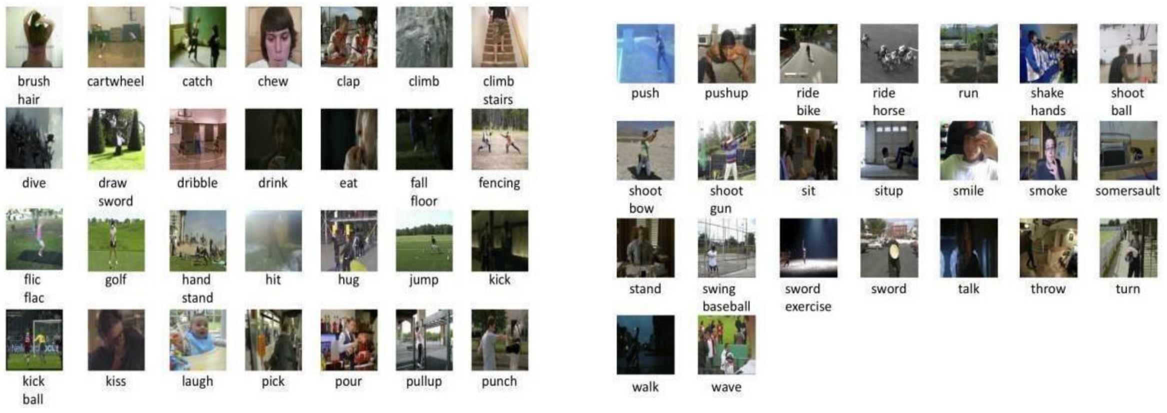


Figure 1. 51 actions of HMDB-51

1.1 Convolutional Neural Networks:

Convolutional neural networks configurations have been succeeded in large-scale image and video recognition. The input to our ConvNets is a fixed-size 224 × 224 RGB image. The preprocessing that we do is subtracting the mean RGB value, computed on the training set, from each pixel. The image is passed through a stack of convolutional layers, where we use filters with a very small receptive field: 3 × 3. In one of the configurations we also utilize 1 × 1 convolution filters, which can be seen as a linear transformation of the input channels (followed by nonlinearity). The convolution stride is fixed to 1 pixel; the spatial padding of conv. layer input is such that the spatial resolution is preserved after convolution, i.e. the padding is 1 pixel for 3 × 3 convolutional layers. Spatial pooling is carried out by five max-pooling layers, which follow some of the convolutional layers. Max-pooling is performed over a 2 × 2-pixel window, with stride 2. A stack of convolutional layers (which has a different depth in different architectures) is followed by three Fully-Connected (FC) layers.

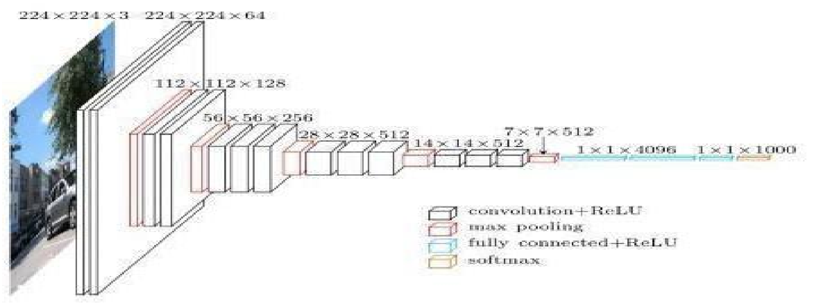


Figure 2. VGG16 Neural network Architecture

1.2 DYNAMIC IMAGES

A Dynamic image is a standard RGB image that summarizes the appearance and dynamics of a whole video sequence. The dynamic image captures the video dynamics directly at the level of the image pixels, by applying a pooling operator before any of the CNN layers are evaluated. The dynamic image is equivalent to a still, that captures the gist of the dynamics and appearance of a whole video sequence or subsequence. The dynamic image is obtained as the parameter of ranking machine learned to sort the frames of the video temporally the other kind of work is that the ranking machine is computed directly at the level of the image pixels as well as any intermediate level of a CNN feature extractor.

1.3 Key Frame Extraction:

Video can be defined as a huge volume data object and contains a high redundancies and intensive information. The process of splitting the video into its major components, include scene segmentation and K.F extraction. One of the problems we faced in K.F extraction process is determining the size of K.F set. This can be solved in three ways.

1) The number of K. Fs is determined as a priori a constant value before starting the extraction process. It is known as rate constant K.F extraction. 2) The K.F number remains unknown until the process of extraction finishes. Level of visual change determines the size of K.F set. If the shot contains a lot of actions and movements, then the required K.F to represent that shot is more than static one. 3) An appropriate size of K.F is determined before the whole extraction process is executed.

2. Proposed work

A 2GB HMDB-51 dataset has to be pre-processed for building a model. The model has to be trained against spitted data. Each split contains 70 training and 30 testing videos with the excess videos excluded from the split. All the videos in the dataset have been normalized for a consistent height of 240 pixels and the widths have been scaled accordingly, ranging between 176 and 592 pixels, to maintain the original aspect ratio. The total work involves:

Pre-process data:

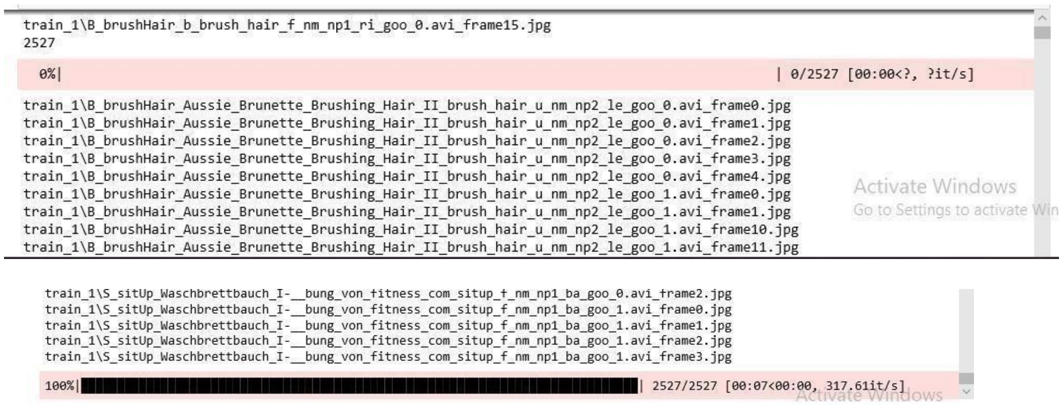
The raw data which always collected may not be in the required format so it is necessary to refine the data. Here the required data should be in the form of "*_class name- name of video file". It can be of either .avi or .mp4 but not a combination of both. First rename the video files in the data as per required format and later write them to a text file.This text file is helpful to locate video with its corresponding action.

Data frame:

Read the text files and generate a data frame for them. Now get the tag name from the video files and append them to data frame.

Frame Generation and CSV file:

From videos, generate frames and store them in a folder in .jpg format. To generate frames, we can use methods of OpenCV. Now store this image file and their class names in a data frame. This data frame is converted into a CSV file.



```
train_1\B_brushHair_b_brush_hair_f_nm_np1_ri_goo_0.avi_frame15.jpg
2527

  0%|                                                        | 0/2527 [00:00<?, ?it/s]

train_1\B_brushHair_Aussie_Brunette_Brushing_Hair_II_brush_hair_u_nm_np2_le_goo_0.avi_frame0.jpg
train_1\B_brushHair_Aussie_Brunette_Brushing_Hair_II_brush_hair_u_nm_np2_le_goo_0.avi_frame1.jpg
train_1\B_brushHair_Aussie_Brunette_Brushing_Hair_II_brush_hair_u_nm_np2_le_goo_0.avi_frame2.jpg
train_1\B_brushHair_Aussie_Brunette_Brushing_Hair_II_brush_hair_u_nm_np2_le_goo_0.avi_frame3.jpg
train_1\B_brushHair_Aussie_Brunette_Brushing_Hair_II_brush_hair_u_nm_np2_le_goo_0.avi_frame4.jpg
train_1\B_brushHair_Aussie_Brunette_Brushing_Hair_II_brush_hair_u_nm_np2_le_goo_1.avi_frame0.jpg
train_1\B_brushHair_Aussie_Brunette_Brushing_Hair_II_brush_hair_u_nm_np2_le_goo_1.avi_frame1.jpg
train_1\B_brushHair_Aussie_Brunette_Brushing_Hair_II_brush_hair_u_nm_np2_le_goo_1.avi_frame10.jpg
train_1\B_brushHair_Aussie_Brunette_Brushing_Hair_II_brush_hair_u_nm_np2_le_goo_1.avi_frame11.jpg
```

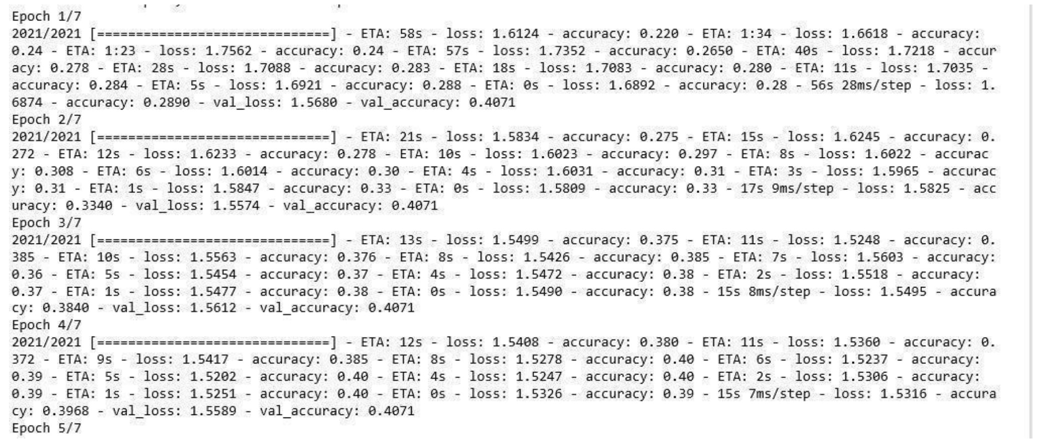
```
train_1\S_sitUp_Waschbrettbauch_I-__bung_von_fitness_com_situp_f_nm_np1_ba_goo_0.avi_frame2.jpg
train_1\S_sitUp_Waschbrettbauch_I-__bung_von_fitness_com_situp_f_nm_np1_ba_goo_1.avi_frame0.jpg
train_1\S_sitUp_Waschbrettbauch_I-__bung_von_fitness_com_situp_f_nm_np1_ba_goo_1.avi_frame1.jpg
train_1\S_sitUp_Waschbrettbauch_I-__bung_von_fitness_com_situp_f_nm_np1_ba_goo_1.avi_frame2.jpg
train_1\S_sitUp_Waschbrettbauch_I-__bung_von_fitness_com_situp_f_nm_np1_ba_goo_1.avi_frame3.jpg

100%|████████████████████████████████████████████████████████| 2527/2527 [00:07<00:00, 317.61it/s]
```

Figure 3. Frame generation

Building Model:

VGG16 is a Convolutional neural network model proposed by K.Simonyan and A. Zisserman. Image net is a dataset of 15 million labelled high-resolution images belonging to roughly 22,000 categories. If you have less amount of data then instead of training your model from scratch you can try Transfer Learning. VGG16 trained against this ImageNet dataset is helpful to produce the interested features. Create the base model and train the images with actions. In which we adjust the number of layers and weights to train the model for accurate results. Now after training the model validate the model using validation set. Reshape them so that we may not have any fitting issues, and get the shape of train and validation set. Now apply the weights for our layers in model, where we have used RELU for activation function and SoftMax for function complete. Now compile our model to measure the model performance by calculating accuracy. Fit the model between the test and validation set to acquire the accuracy. Save model as we can't always train the model every time we test the model or recognize the actions we need to save the model.



```
Epoch 1/7
2021/2021 [==============================] - ETA: 58s - loss: 1.6124 - accuracy: 0.220 - ETA: 1:34 - loss: 1.6618 - accuracy:
0.24 - ETA: 1:23 - loss: 1.7562 - accuracy: 0.24 - ETA: 57s - loss: 1.7352 - accuracy: 0.2650 - ETA: 40s - loss: 1.7218 - accur
acy: 0.278 - ETA: 28s - loss: 1.7088 - accuracy: 0.283 - ETA: 18s - loss: 1.7083 - accuracy: 0.280 - ETA: 11s - loss: 1.7035 -
accuracy: 0.284 - ETA: 5s - loss: 1.6921 - accuracy: 0.288 - ETA: 0s - loss: 1.6892 - accuracy: 0.28 - 56s 28ms/step - loss: 1.
6874 - accuracy: 0.2890 - val_loss: 1.5680 - val_accuracy: 0.4071
Epoch 2/7
2021/2021 [==============================] - ETA: 21s - loss: 1.5834 - accuracy: 0.275 - ETA: 15s - loss: 1.6245 - accuracy: 0.
272 - ETA: 12s - loss: 1.6233 - accuracy: 0.278 - ETA: 10s - loss: 1.6023 - accuracy: 0.297 - ETA: 8s - loss: 1.6022 - accurac
y: 0.308 - ETA: 6s - loss: 1.6014 - accuracy: 0.30 - ETA: 4s - loss: 1.6031 - accuracy: 0.31 - ETA: 3s - loss: 1.5965 - accurac
y: 0.31 - ETA: 1s - loss: 1.5847 - accuracy: 0.33 - ETA: 0s - loss: 1.5809 - accuracy: 0.33 - 17s 9ms/step - loss: 1.5825 - acc
uracy: 0.3340 - val_loss: 1.5574 - val_accuracy: 0.4071
Epoch 3/7
2021/2021 [==============================] - ETA: 13s - loss: 1.5499 - accuracy: 0.375 - ETA: 11s - loss: 1.5248 - accuracy: 0.
385 - ETA: 10s - loss: 1.5563 - accuracy: 0.376 - ETA: 8s - loss: 1.5426 - accuracy: 0.385 - ETA: 7s - loss: 1.5603 - accuracy:
0.36 - ETA: 5s - loss: 1.5454 - accuracy: 0.37 - ETA: 4s - loss: 1.5472 - accuracy: 0.38 - ETA: 2s - loss: 1.5518 - accuracy:
0.37 - ETA: 1s - loss: 1.5477 - accuracy: 0.38 - ETA: 0s - loss: 1.5490 - accuracy: 0.38 - 15s 8ms/step - loss: 1.5495 - accura
cy: 0.3840 - val_loss: 1.5612 - val_accuracy: 0.4071
Epoch 4/7
2021/2021 [==============================] - ETA: 12s - loss: 1.5408 - accuracy: 0.380 - ETA: 11s - loss: 1.5360 - accuracy: 0.
372 - ETA: 9s - loss: 1.5417 - accuracy: 0.385 - ETA: 8s - loss: 1.5278 - accuracy: 0.40 - ETA: 6s - loss: 1.5237 - accuracy:
0.39 - ETA: 5s - loss: 1.5202 - accuracy: 0.40 - ETA: 4s - loss: 1.5247 - accuracy: 0.40 - ETA: 2s - loss: 1.5306 - accuracy:
0.39 - ETA: 1s - loss: 1.5251 - accuracy: 0.40 - ETA: 0s - loss: 1.5326 - accuracy: 0.39 - 15s 7ms/step - loss: 1.5316 - accura
cy: 0.3968 - val_loss: 1.5589 - val_accuracy: 0.4071
Epoch 5/7
```

Figure 4. Model building

Pooling Layers

Pooling layers are generally used to reduce the size of the inputs and hence speed up the computation. If we Consider a 4 X 4 matrix, applying max pooling on this matrix will result in a 2 X 2 output. For every consecutive 2 X 2 block, we take the max number. Here, we have applied a filter of size 2 and a stride of 2. These are the hyper parameters for the pooling layer. Apart from max pooling, we can also apply average pooling where, instead of taking the max of the numbers, we take their average. In summary, the hyper parameters for a pooling layer are: Filter size, Stride, Max or average pooling

If the input of the pooling layer is Nh X Nw X Nc, then the output will be

[{(Nh – f) / s + 1} X {(Nw – f) / s + 1} X Nc].

3. Result and Analysis

In this section, train multi-way soft-max classifiers for the classes of UCF101/HMDB51 (using their training data), and evaluate on their test sets. The video files of HMDB51 dataset has been remaned using built in functions and the names are copied to text file. Deep learning mainly deals with the image data, so the text file stores the video paths which directly points to video in the dataset.
The following are the frames obtained for HMDB-51 training dataset.

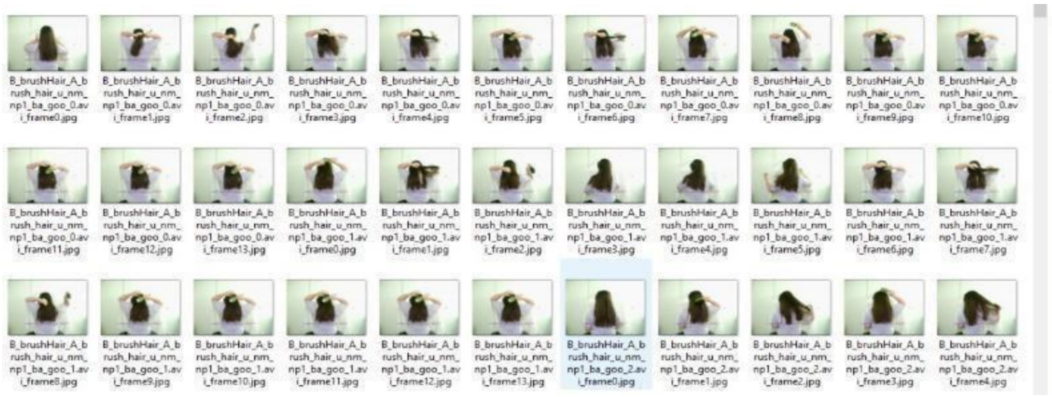

B_brushHair_A_brush_hair_u_nm_np1_ba_goo_0.avi_frame0.jpg
B_brushHair_A_brush_hair_u_nm_np1_ba_goo_0.avi_frame1.jpg
B_brushHair_A_brush_hair_u_nm_np1_ba_goo_0.avi_frame2.jpg
B_brushHair_A_brush_hair_u_nm_np1_ba_goo_0.avi_frame3.jpg
B_brushHair_A_brush_hair_u_nm_np1_ba_goo_0.avi_frame4.jpg
B_brushHair_A_brush_hair_u_nm_np1_ba_goo_0.avi_frame5.jpg
B_brushHair_A_brush_hair_u_nm_np1_ba_goo_0.avi_frame6.jpg
B_brushHair_A_brush_hair_u_nm_np1_ba_goo_0.avi_frame7.jpg
B_brushHair_A_brush_hair_u_nm_np1_ba_goo_0.avi_frame8.jpg
B_brushHair_A_brush_hair_u_nm_np1_ba_goo_0.avi_frame9.jpg
B_brushHair_A_brush_hair_u_nm_np1_ba_goo_0.avi_frame10.jpg
B_brushHair_A_brush_hair_u_nm_np1_ba_goo_0.avi_frame11.jpg
B_brushHair_A_brush_hair_u_nm_np1_ba_goo_0.avi_frame12.jpg
B_brushHair_A_brush_hair_u_nm_np1_ba_goo_0.avi_frame13.jpg
B_brushHair_A_brush_hair_u_nm_np1_ba_goo_1.avi_frame0.jpg
B_brushHair_A_brush_hair_u_nm_np1_ba_goo_1.avi_frame1.jpg
B_brushHair_A_brush_hair_u_nm_np1_ba_goo_1.avi_frame2.jpg
B_brushHair_A_brush_hair_u_nm_np1_ba_goo_1.avi_frame3.jpg
B_brushHair_A_brush_hair_u_nm_np1_ba_goo_1.avi_frame4.jpg
B_brushHair_A_brush_hair_u_nm_np1_ba_goo_1.avi_frame5.jpg
B_brushHair_A_brush_hair_u_nm_np1_ba_goo_1.avi_frame6.jpg
B_brushHair_A_brush_hair_u_nm_np1_ba_goo_1.avi_frame7.jpg
B_brushHair_A_brush_hair_u_nm_np1_ba_goo_1.avi_frame8.jpg
B_brushHair_A_brush_hair_u_nm_np1_ba_goo_1.avi_frame9.jpg
B_brushHair_A_brush_hair_u_nm_np1_ba_goo_1.avi_frame10.jpg
B_brushHair_A_brush_hair_u_nm_np1_ba_goo_1.avi_frame11.jpg
B_brushHair_A_brush_hair_u_nm_np1_ba_goo_1.avi_frame12.jpg
B_brushHair_A_brush_hair_u_nm_np1_ba_goo_1.avi_frame13.jpg
B_brushHair_A_brush_hair_u_nm_np1_ba_goo_2.avi_frame0.jpg
B_brushHair_A_brush_hair_u_nm_np1_ba_goo_2.avi_frame1.jpg
B_brushHair_A_brush_hair_u_nm_np1_ba_goo_2.avi_frame2.jpg
B_brushHair_A_brush_hair_u_nm_np1_ba_goo_2.avi_frame3.jpg
B_brushHair_A_brush_hair_u_nm_np1_ba_goo_2.avi_frame4.jpg

Figure 5. Training frames

The frames generated are used for building model and the video that to be tested is provided as testing data for the built model. The model has been trained with 98% accuracy. Now the testing video has been split into series of frames and sent for classification. The below are the frames generated for the testing video of Brush Hair class.

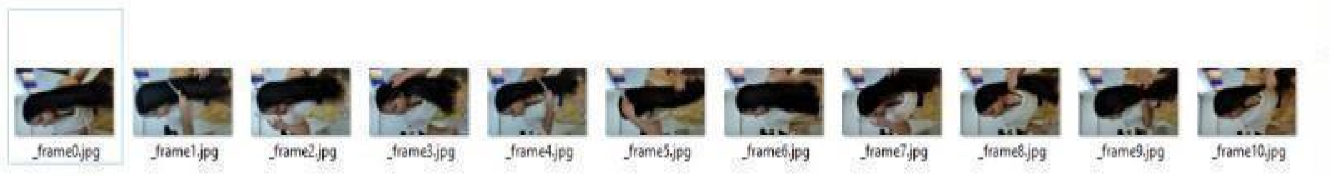


Figure 6. Testing video frames

To get accurate results it is more important to train model without any bias in the training set. If the model does not reach the desired accuracy, increment the epochs or add more layers for the network.But the model should be free from overfitting and underfitting problems, as this leads to wrong results,
Now, basing on the frames generated, the classifier is responsible to map frames.The given has been predicted as Brush Hair action by the network.

Conclusions

Adaptive frame extraction is achieved and the action of real time videos can also be recognized. Like the Brush Hair action predicted previously, 51 different actions can be recognized with built network. If we want to apply this for a greater number of classes then train the model of those actions such that even those actions can also be recognized. We have applied our training for two popular models called as VGG16, ResNet where VGG16 is good at using a smaller number of parameters and ResNet at equalizing the weights the accuracy provided by them have a slight variation.

Future Scope

This experiment is helpful to recognize the unauthorized or harmful actions being performed at sensible places. We can extend this work to intimate the illegitimate act directly by connecting a sensor in a continuous streaming video. The set of actions that are predicted as illegitimate should be learned separately by the network. This helps to identify culprit autonomously without human intervention.

References

[1] J. Carreira and A. Zisserman, "Quo vadis, action recognition? a new model and the kinetics dataset," arXiv preprint arXiv:1705.07750, 2017.

[2] H. Bilen, B. Fernando, E. Gavves, A. Vedaldi, and S. Gould. Dynamic image networks for action recognition. In IEEE International Conference on Computer Vision and Pattern Recognition CVPR, 2016.

[3] Ciresan, D. C., Meier, U., Masci, J., Gambardella, L. M., and Schmidhuber, J. Flexible, high performance convolutional neural networks for image classifification. In IJCAI, pp. 1237–1242, 2011.

[4] S. V. Porter, M. Mirmehdi, and B. T. Thomas, —A shortest path representation for video summarization‖, 12th International Conference on Image Analysis and Processing, 2003, pp. 460–465, Italy.

[5] Rachida Hannane, Abdessamad Elboushaki Karim Afdel1, P. Naghabhushan and Mohammed Javed, —An efficient method for video shot boundary detection and keyframe extraction using SIFT-point distribution histogram‖, International Journal of Multimedia Information Retrieval, Vol. 5, No. 2, 2016, pp. 89-104.

[6] Hakan Bilen,Basura Fernando,Efstratios Gavves and Andrea Vedaldi "Action Recognition with Dynamic Image Networks".

[7] Hu, N. Xie, L. Li, X. Zeng, and S. Maybank, ―A survey on visual content-based video indexing and retrieval, IEEE Transactions on Systems, Man, and Cybernetics—part c: Applications and Reviews, Vol. 41, No. 6, 2011, pp. 797 819.

[8] Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., and LeCun, Y. OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks. In Proc. ICLR, 2014.

[9] S. Ali and M. Shah "Human action recognition in videos using Kinematic features and multiple instance learning." TPAMI, vol.32, no. 2, pp.288-303, 2010.

[10] https://machinelearningmastery.com/deep-learning-models-for-human-activity-recognition/