Human Action Recognition

Aman Pandey (2017csb1127)
IIT Ropar
Ropar, Punjab

Amit Srivastava (2017csb1189)

Abstract

Video analysis tasks have seen great variations and it has been moving from inferring the present state to predicting the future state. This has been possible with the developments in the field of Computer Vision and Machine Learning. In vision-based action recognition tasks, various human actions are inferred based upon the complete movements of that action. It also helps in prediction of future state of the human by inferring the current action being performed by that human. It has become a hot topic in recent years since it applies to real world problems directly, be it visual surveillance, autonomous driving vehicle, entertainment etc. Many research works have been done in this field to develop a good human action recognizer. Also, it is expected that more and more work is going to be done in this regards since Human Action Recognition finds its use in numerous applications.

In this report, we will be seeing the details of the work done by us and the methodology adopted for the same. We will go through the techniques used to implement a system that is suitable for human activity recognition. We will also compare the different models used for action recognition and note which one is better than other. Finally, we will conclude by noting down the results we get from the final model we have made.

Introduction

The aim of Human Action Recognition (HAR) is to determine what is happening in a video by some computation. Human Activities can be classified in various categories including Human-Human interaction and Human-Object interaction. This classification is based on human actions where actions of humans are specified by their gestures, poses etc. Human action recognition is challenging due to variations in motion, illumination, partial occlusion of humans, viewpoint, anthropometry of people involved in the various interactions. Some of the difficulties in recognition are more related to the problem of detecting and tracking people in videos while others are more related to the process of recognizing the action. Additionally, action classes present large intra-class variability and low inter-class variability. On the one hand, action classes are subject to large intra-class variability because of the viewpoint of the camera. It is not the same gesture when a person raises an arm with the camera in a frontal viewpoint as when the person lifts the same arm and he/she is viewed from the side. Besides, a raised left arm, right arm, or both arms are different gestures that correspond to the same raised hand action. Then there is the issue of the individual style of a person in executing a gesture, not only in terms of the time duration but also the way in which the gesture is performed. Another aspect to consider is the anthropometric differences across humans, differences due to age or size or body part ratios.

1 Statement Of The Problem

Given a video, is it possible to recognize the action being performed in that video by an automatic system? An action recognition system is built on basic steps: first, the input video or sequence of frames; second, the extraction of low-level features from the frames; and finally, mid-level pose/gesture or action descriptions from low-level features.

2 Significance Of The Problem

Human action recognition (HAR) is useful in various applications like video surveillance, identity recognition, to create intelligent systems for human-machine interaction and much more. It is a challenging problem in the field of machine learning and computer vision. A good feature representation is a key to human action recognition. Detecting features and working just on spatial region is not enough for HAR but we also need to examine features from the perspective of changes with respect to time. In recent years, many kinds of action representation methods have been proposed, including local and global features based on temporal and spatial changes, trajectory features based on key point tracking, motion changes based on depth information, and action features based on human pose changes. With the

successful application of deep learning to image classification and object detection, many researchers have also applied deep learning to human action recognition.

3 Purpose

The main purpose of this project is to build an automatic system, which when given a video, will be able to classify the action being performed in the video.
To achieve this purpose, we have built a model that will be able to classify those videos. To build that model, we required a big dataset that we have used to train our model and then use that model to present a solution to above classification problem. We have talked more about the dataset in next section.

4 Dataset

The dataset used in building our model is UCF101. UCF101 is an action recognition data set of realistic action videos, collected from YouTube, having 101 action categories. This data set is an extension of UCF50 data set which has 50 action categories.
With 13320 videos from 101 action categories, UCF101 gives the largest diversity in terms of actions and with the presence of large variations in camera motion, object appearance and pose, object scale, viewpoint, cluttered background, illumination conditions, etc, it is the most challenging data set to date. As most of the available action recognition data sets are not realistic and are staged by actors, UCF101 aims to encourage further research into action recognition by learning and exploring new realistic action categories.
The videos in 101 action categories are grouped into 25 groups, where each group can consist of 4-7 videos of an action. The videos from the same group may share some common features, such as similar background, similar viewpoint, etc.
The action categories can be divided into five types: 1)Human-Object Interaction 2) Body-Motion Only 3) Human-Human Interaction 4) Playing Musical Instruments 5) Sports.

Literature Review

Human Action Recognition (HAR) has been worked upon and studied well in recent years. In starting years, the methods used to find out interest points of an action were mostly done manually by looking at various feature points for various actions. But in recent years, efforts have been made to detect those feature points rather automatically by a machine. Although it is difficult to extract perfect interest points from a computer, yet progress has been made towards it to reduce the manual work being done.

One of the early methods proposed to provide solution for HAR was to process frames from a video separately and do some kind of averaging to get the results. Although this method is good when one wants to detect an action from one image, but for videos this method is not a good idea since we are ignoring the time-based changes in the human actions. This method was based on CNN (Convolutional Neural Networks). This method basically classifies various images based on the features of the image. Firstly, we train our model on a large number of frames and then the trained model is used to predict the action in the video. But in this method, we are not looking at the video as a transition from one state to another, rather we are looking a video as just a group of multiple images. Thus, the temporal nature is being ignored in this case.

Another method that tries to attack this problem of HAR uses spatio-temporal features of a video. The action being performed by a human is dependent on the sequence of activities performed by him at various time intervals. This method is based on RNN (Recurrent Neural Networks). We basically try to train a model based on features at various time stamps. This ensures that we take into account the sequence of the poses/gestures performed. Essentially, we mean that an action is comprised of a sequence of poses/gestures and we try to learn that sequence also instead of just learning the poses separately.

Space-time approaches recognize activities based on space-time features or on trajectory matching. Basically, there is 3D space-time where the activity occurs which consists of sequence of 2D spaces in time. An activity is associated with a set of space-time feature trajectories.

The final model that we have built is based on CNN+RNN (more specifically CNN + LSTM). Here we extract features from a CNN and then the output features from CNN are passed onto the RNN for learning the action sequence. Then we try to classify the videos according the model trained.

Recently there has been a huge growth in computer vision research with regard to HAR. There has been emphasis on activities where the entity to be recognized may be seen as a stochastically predictable sequence of state. Researchers have conceived and used many stochastic techniques like hidden Markov Model etc, to infer useful results for HAR. There have been introduced rule-based approaches as well where events are determined by modelling an activity using rules of attributes that describe that event. Each activity can be considered as a set of primitive rules which enables such a model to be built for HAR.

Shape-based methods have also been introduced to tackle this problem where human body can be represented by 2D patches or 3D shapes of some kind.
Concluding, we can say that there is huge research being done for HAR in the field of Computer Vision and in future we may develop a system where we can have much higher accuracy in our predictions.

Methodology

We used “Google Colab” to run the programs. It provides us approx. 12 GB of RAM and variable GPU depending on the traffic.
Steps include:
a) Working with the dataset.
b) Extracting features from the frames using CNN.
c) Passing those features into a recurrent neural network to train the model and classifying the videos based on that.
d) Saving the best model and then using it to classify the videos.

5 Working with the dataset

We downloaded the dataset “UCF101” and extracted in google drive. It contains 13320 videos belonging to 101 different classes.
Then we separated the video in two folders namely “test” and “train”. This splitting was based on the test train spilt file provided in their official site.
Then, we extracted the frames from each video in their corresponding folders as .jpg files.

6 Extracting Features using CNN

To extract features from the images, we used a pre-trained model provided by Keras named Inception-V3.
Inception-V3 is a model provided by Keras whose weights pre-trained on ImageNet dataset, which is a huge dataset consisting of large number of images of different classes. It saves us from building our own architecture from scratch which could take far more time and efforts and not necessarily provide better results than when using Inception-V3.
Thus, in our project we used this Inception-V3 model to extract features. In our initial work, we just settled with this work by fine tuning some outermost layers and predicting the class solely based on this. But, we improved our design by using RNN which is shown in next step.

7 Passing The Features to RNN and training the model

Instead of classifying the images just with CNN, we now use CNN+RNN model. Here the feature output from Inception-V3 are first taken by removing the top classifying layer. It gives a vector of features.
We used LSTM, which is a special kind of RNN. It is also provided by keras in python. The features we obtained from Inception-V3, are passed to this LSTM architecture. After that, we defined some of our own layers. We tried with dense layers with 'relu' activation with some dropouts between multiple layers. After trying with different layers, we settled down to a 2048-wide LSTM layer followed by a 512 dense layer and used 0.5 dropout. Note that we used the features from a video as input to the LSTM. Instead of passing features from every frame, we reduced the number of frames to be passed. This reduced overfitting and also made the program faster due to less features being processed now. Features from approx. 30-50 frames per video are sent as sequence to the LSTM.
At the final layer, we used 'softmax' activation, which is basically used to provide a class as output in terms of probability, i.e. it turns the input numbers to probabilities that sum to 1. Thus, it outputs a probability distribution.
We used "categorical cross entropy" as a loss measure and adam optimizer was used with learning rate of .Also, the metrics for training and testing was 'accuracy' and 'top 5 categorical accuracy'. Then we compiled our model using these parameters and are ready for training.
Thus training is based on above mentioned architecture. The training and testing samples are divided according to the file provided in official website. This ensures uniform division of videos.
The model is trained to learn in such a way to minimize the validation loss. When we stop learning, i.e. the validation loss keeps on increasing, then we stop the training. The accuracy metrics is used.
When doing the training, we save the model weights whenever validation loss is decreased. When training finishes, we get the recently saved weights as the best weights and we will use those weights further in classifying our own videos.

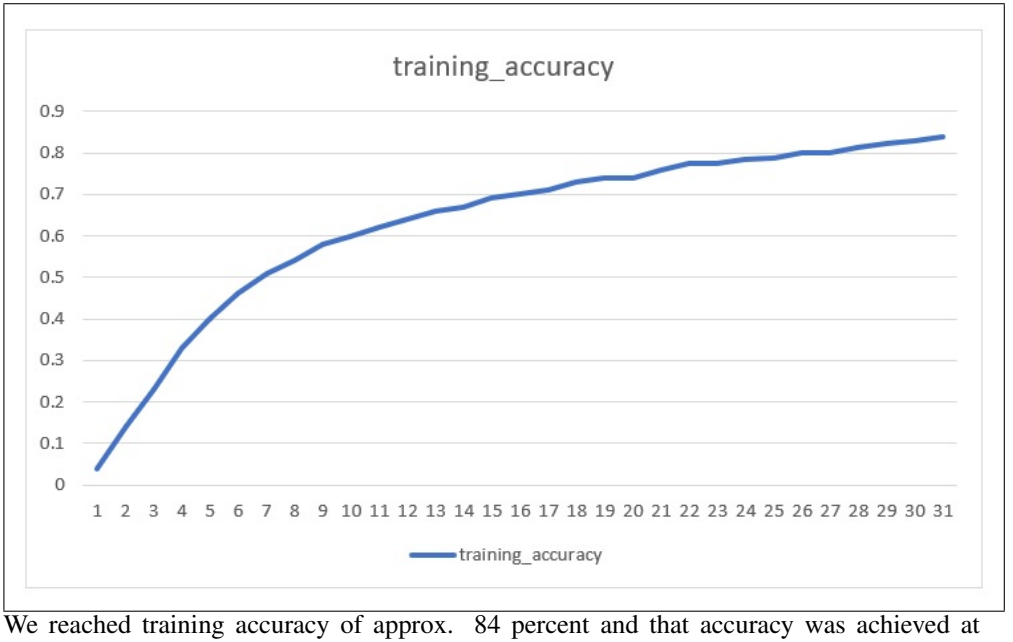
8 Saving the best model and classifying videos

As written above, we have saved the model weights for that iteration where we get the least validation loss. This is generally a good practice because if we save model with best accuracy only, no matter what the loss is, then we can't say that it is the best model. So, least loss is what we used when specifying the best model. Although it may vary depending on the tasks given, but generally for HAR, it is good enough.
Now, to classify videos, we just define the same architecture we defined for training. Then we load the model weights and now we are ready to classify the videos. Just take a video, extract frames and then use Inception-V3 to extract features from that video. Pass those features into the architecture and we get the class to which the video belongs.

Experiments And Conclusion

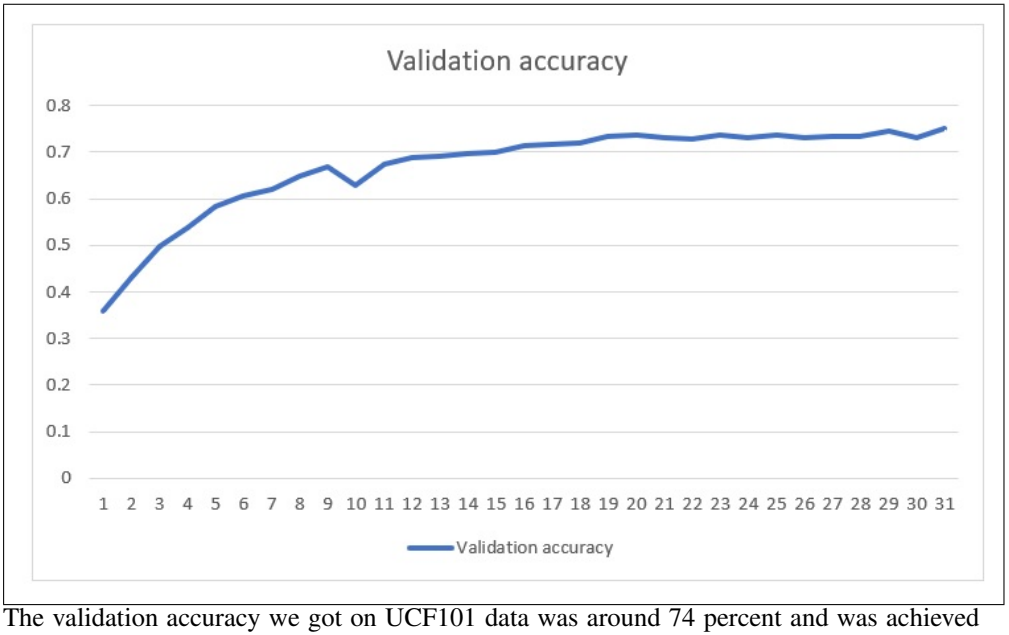
9 Experiments

We trained our model on UCF101 dataset using the architecture described in the previous section and the results were obtained.
We experimented with different hidden layers added to our architecture so as to get the best model.
Earlier the method we used included CNN only and the validation accuracy was very low in that case when compared to the new model we defined.
The following graphs shows various accuracies and loss we found out during the experiment:

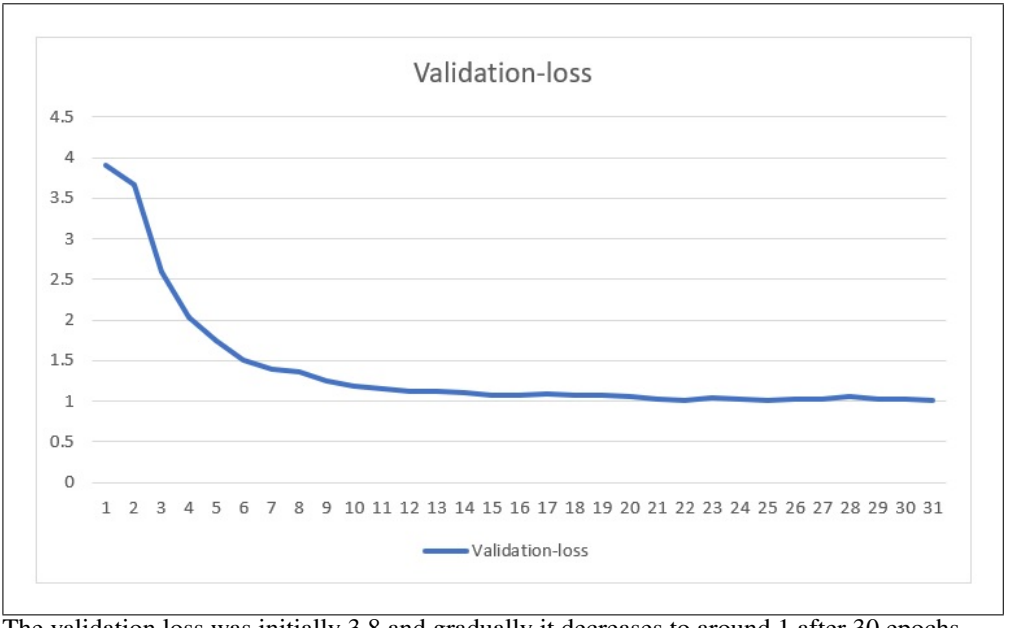


We reached training accuracy of approx. 84 percent and that accuracy was achieved at around 30 epochs.

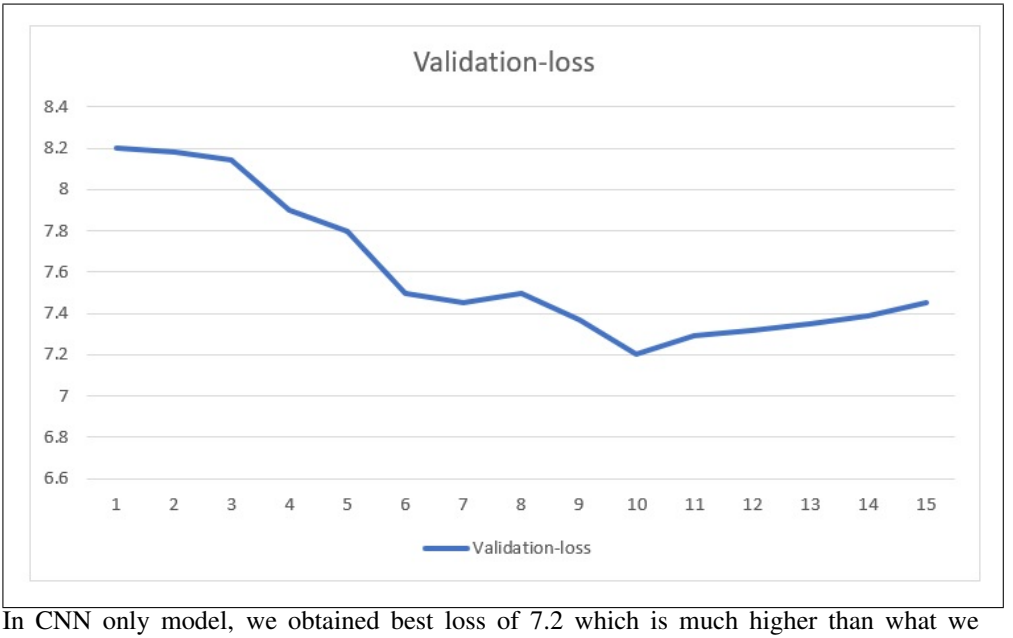
The training loss decreased with number of epochs. It dropped down from 4.2 to 0.6 with 30 epochs.



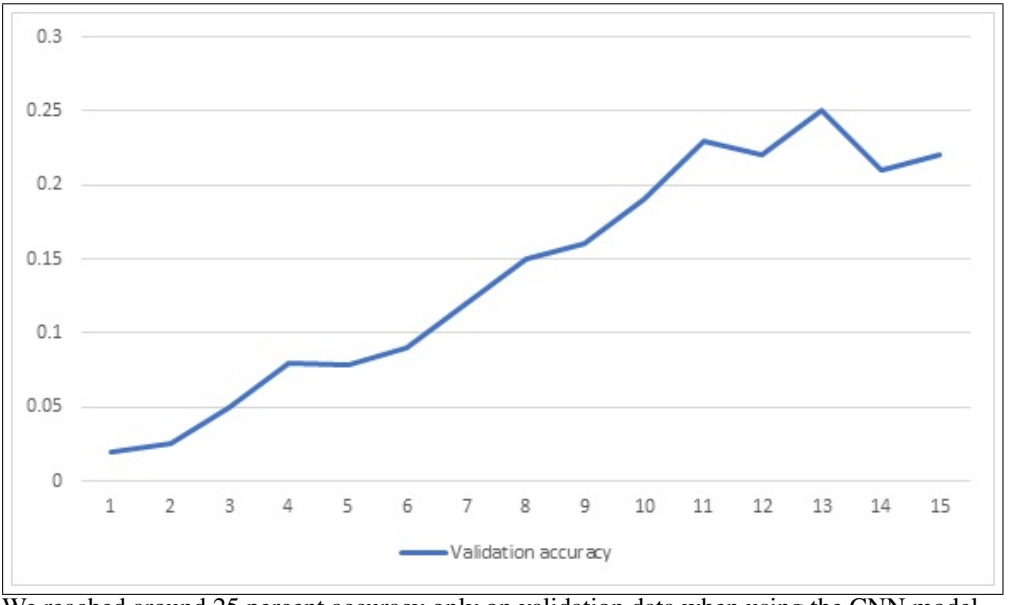
The validation accuracy we got on UCF101 data was around 74 percent and was achieved with 30 epochs.



The validation loss was initially 3.8 and gradually it decreases to around 1 after 30 epochs.



In CNN only model, we obtained best loss of 7.2 which is much higher than what we obtained in our new model which was nearly 1.



We reached around 25 percent accuracy only on validation data when using the CNN model only. This is very much less compared to 74 percent accuracy we obtained on CNN+RNN model.

10 Conclusion And Learnings

10.1 Conclusion

We can use deep learning models to solve the human action recognition problem. When we do not take into account the temporal nature of the video into account, then we lack in classifying videos with good accuracy.

Spatio-temporal understanding provides a better way to human action recognition.

The new model we used for implementation gave about 74-75% validation accuracy on such a big dataset UCF101. And the loss was approximately 1. Whereas, the previous model just gave 25% accuracy with a higher loss.

Although this 74% accuracy achieved is not that good when compared to 94% state-of-the-art accuracy, yet it is not that bad even because we were limited by the resources and lack of knowledge of some other better techniques.

We hope to learn new techniques to solve HAR problem which will enhance our understanding.

10.2 Learnings

During this whole project, we learned many new things.

We learned about neural network, their different types: CNN and RNN. Since we had to study them in detail for the implementation, it enhanced our knowledge about neural networks.

With these learnings, we hope to work more towards this field as in the future most of the things will be computational and we would like to keep pace with it. Also, it is very interesting to learn that how a machine is made to act and think like humans.

We learned about Human Action Recognition methods and also how it can very useful in several applications.

We learned many new libraries provided in Python. It makes it easy to implement our ideas. Main focus of learning was CNN and RNN. We learnt them from scratch and also saw various applications where these can be used.