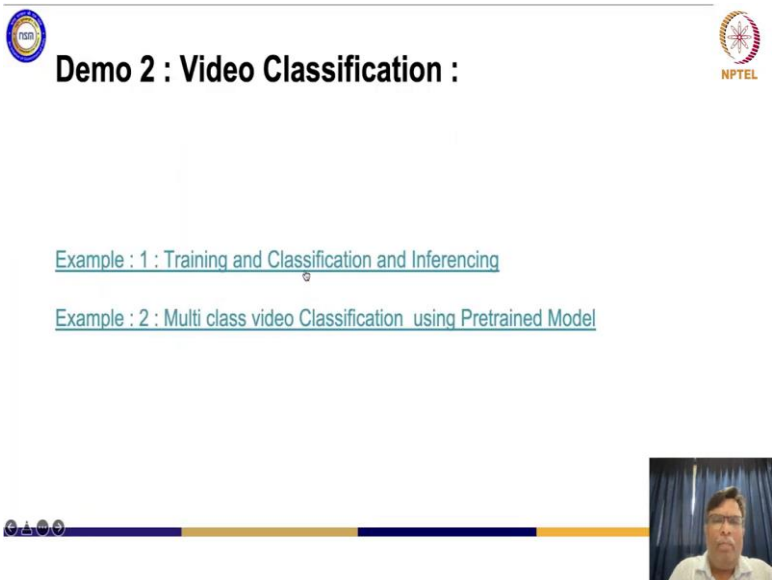
Applied Accelerated Artificial Intelligence
Prof. Satyadhyan Chickerur
Department of Computer Science and Engineering
Indian Institute of Technology, Palakkad

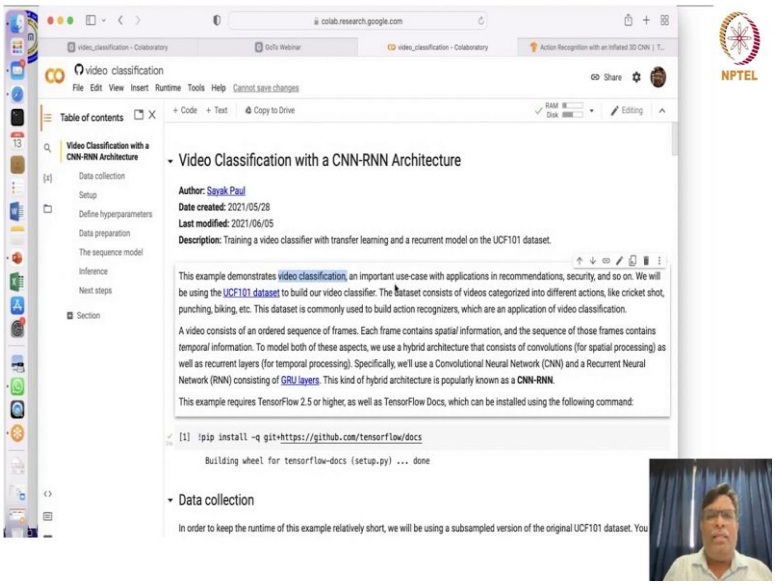
Lecture - 55
Applied AI: Smart City (Intelligent Video Analytics)
Session 1 - Part 2

(Refer Slide Time: 00:19)



Video classification example, 1 minute, yeah. So, this we have already there, so yeah.

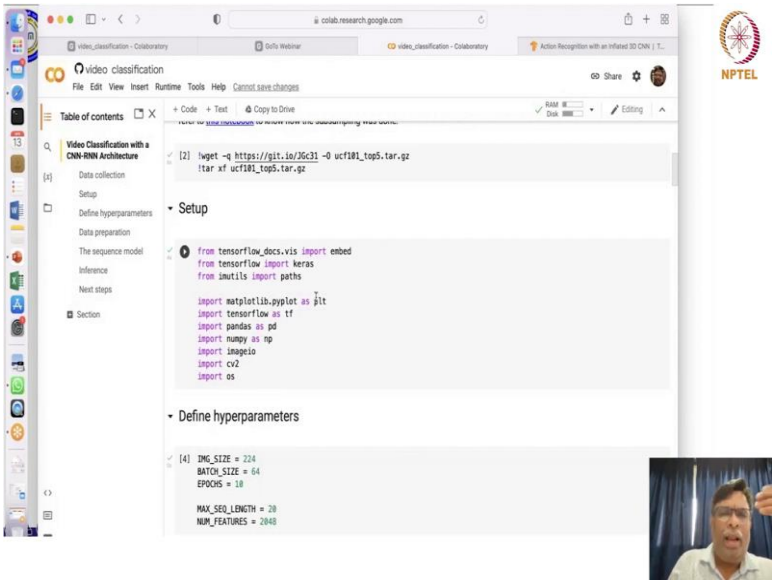
(Refer Slide Time: 00:34)



So, this example basically talks off a CNN-RNN architecture and this basically is trying to train a video classifier with transfer learning and it uses a recurrent neural network model and it uses UCF101 dataset. So, this demonstration actually gives us the video classification it is an important use case and this particular UCF101 dataset consists of different actions right it has got cricket shot, punching, biking ok.

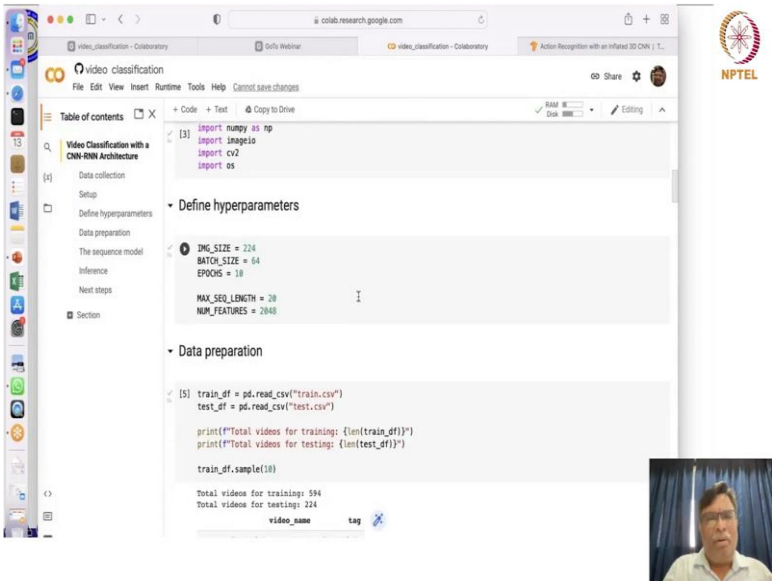
And this particular data set is generally used to build action recognizers right which are an application of video classification. So, this is the first step wherein you are trying to set up your environment right.

(Refer Slide Time: 01:35)



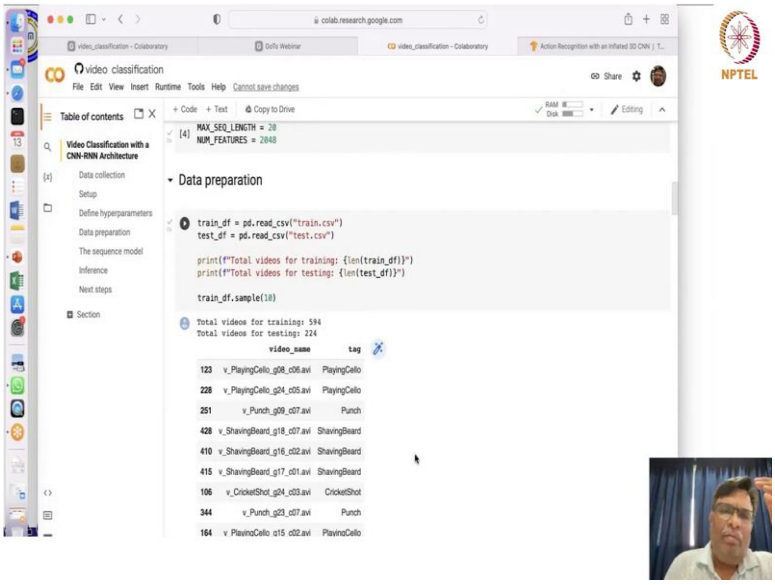
And then this is where you collect in the data right or the data set UCF101 then you basically set up your environment wherein you are using CV2 pandas, matplotlib, tensor flow ok keras and all of that.

(Refer Slide Time: 01:54)



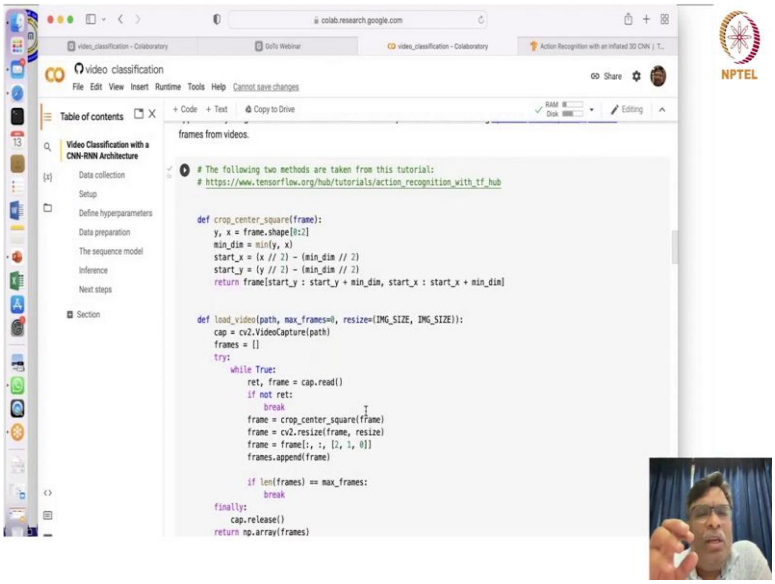
Then you define the hyper parameters what is the image size, what is the batch size, what are what is the epoch right and what is the maximum sequence length and number of features which you are trying to do.

(Refer Slide Time: 02:08)



Then you have this data preparation wherein how many videos are being used for training, how many videos are being used for testing right. So, you are trying to actually use this for training purpose and testing purpose and then you basically are trying to do ok.

(Refer Slide Time: 02:29)

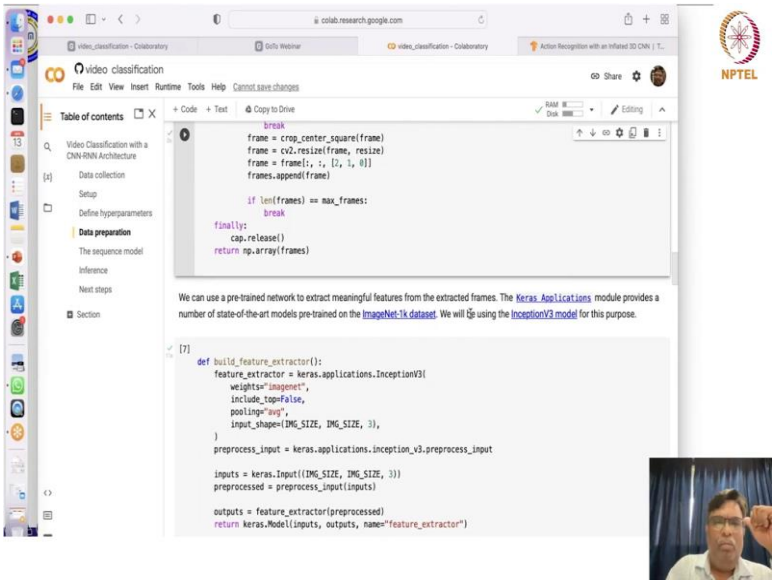


Action recognition in a sense you are trying to actually do action recognition with tf hub. So, this basic idea is that you are trying to generate a sequence model right. So, you are trying to load in the video ok you are trying to load in the video what is the frame we are trying to use ok.

And then we are going to actually crop the frame right for getting the center and then trying to understand the region of interest and then once you are able to extract right all these frames. So, you are trying to actually do to get the region of interest ok which basically gives you the action recognition.

So; obviously, in a frame in a static frame there will be some region of interest right and that region of interest you need to actually crop ok.

(Refer Slide Time: 03:48)

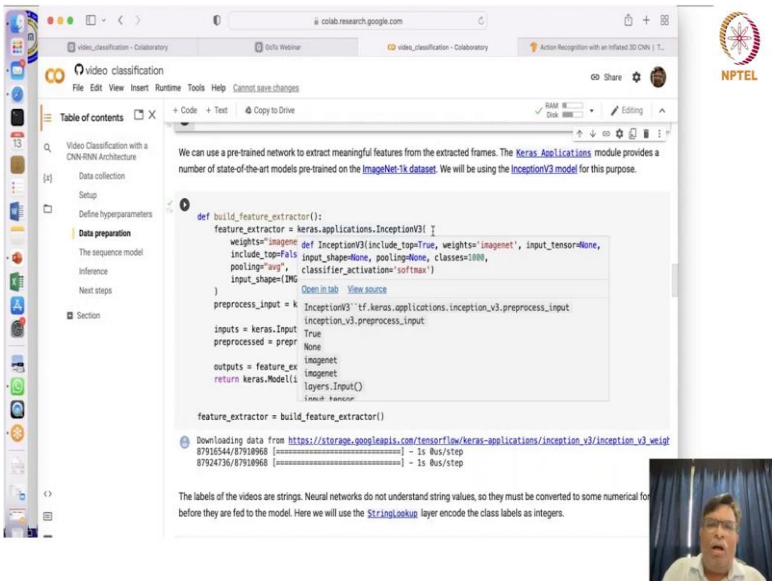


And then each of these frames ok could be or should be actually used ok buy a pretrained network ok to extract certain meaningful features right from these frames. So, now what effectively happens is, you have found out the region of interest in the frames and out of these region of interest or these portions of your frames you need to actually extract meaningful information right. So, effectively you are trying to develop a feature extractor ok.

Now, when you talk of a feature extractor, the idea is that you are trying to actually get those features which are more important right to actually generalize for various videos which you are going to get input for or you are going to get those videos as input right. Otherwise, a question basically is or might be asked that when you are trying to do a classification and you are trying to use CNN and RNN architecture.

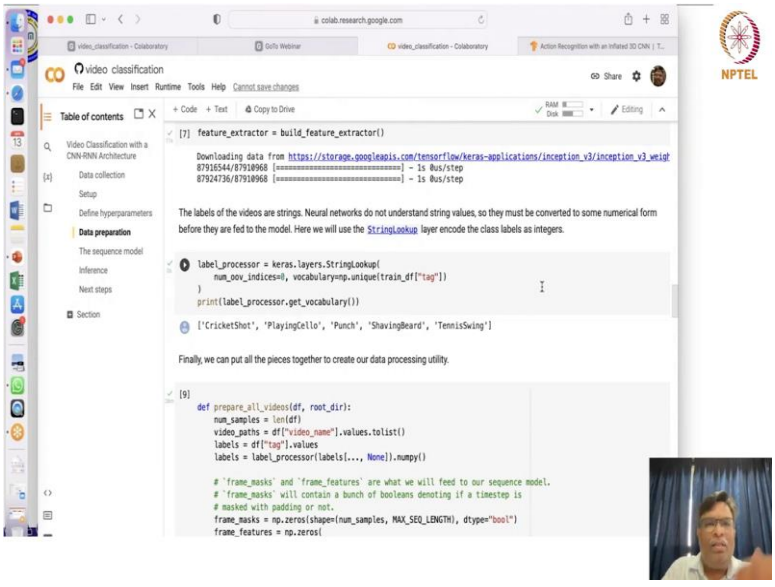
Why is that you are going to again extract features when CNN and RNN is actually capable of doing it on its own? Ok. So, the idea here is that the features are extracted or features might be extracted, but which are the specific features of our own interest ok is something which we need to actually boil down to from lot of features which this type of a network would have given us right.

(Refer Slide Time: 05:52)



So, we are trying to develop a feature extractor here right ok and here we are using inception v3 right for that purpose. So, now if you want to see the source and all, all of this is available in this.

(Refer Slide Time: 06:11)



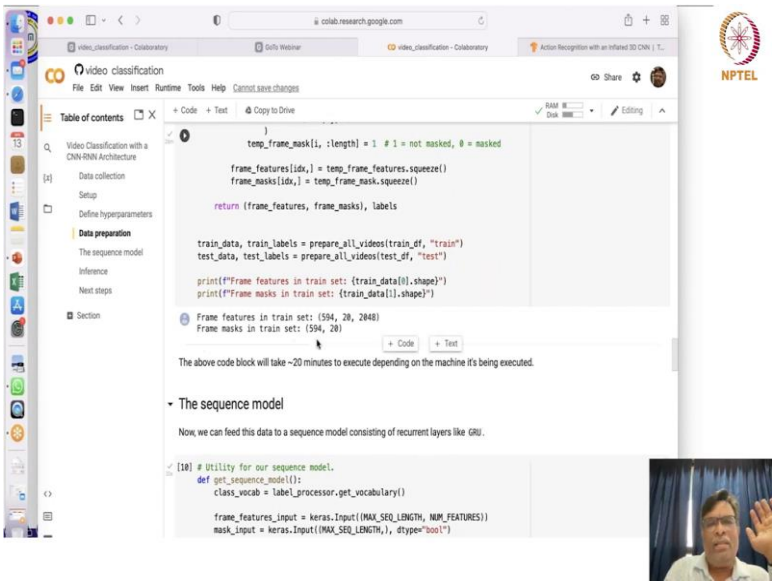
Now you have extracted the features or you have built the feature extractor now the labels for each of these videos are strings and when you say a neural network or for that matter a deep neural network it is not going to understand the string values. So, you are supposed to convert it into some form ok which is basically to be used by a neural network model.

So, you are going to convert it into some numerical form ok and then there is something which wherein we use string loop lookup ok which basically is a layer which encodes the class labels as integers. So, there has to be a encoding done for each of these classes as integers right. So, this is what is done here ok. So, if you see the label processor it is going to give us the labels like cricket shot, playing cello punch, shaving beard, tennis swing.

So, these are from that particular data set right. So, we are talking of video classification right. So, now, once you actually see all this. So, what have we done till now? We have actually done these steps wherein we have defined our hyper parameters, we have prepared in the data ok which needs to be used ok and then we have tried to find out ok which are the actions which we need to actually do things and then we have to basically build a feature extractor ok.

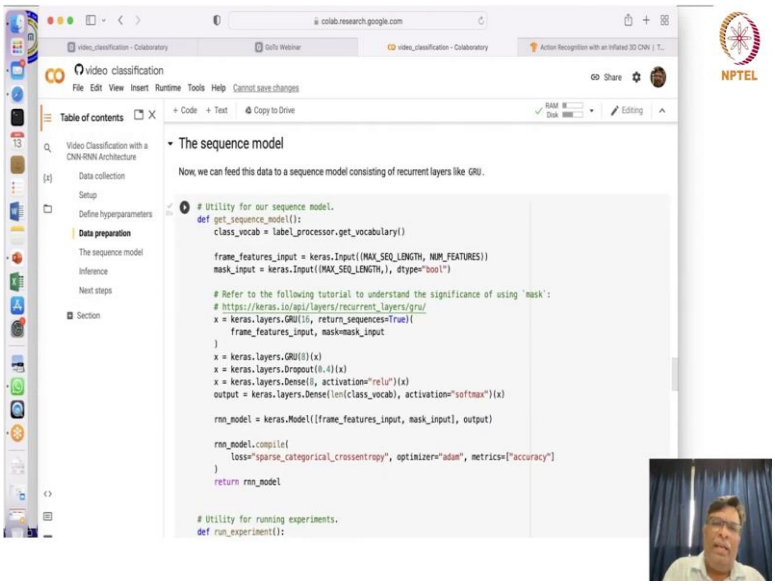
And then we can basically link each of these class of labels to each of these videos right and then you are going to prepare all the videos for each of these videos you are going to actually do all of this ok.

(Refer Slide Time: 08:33)



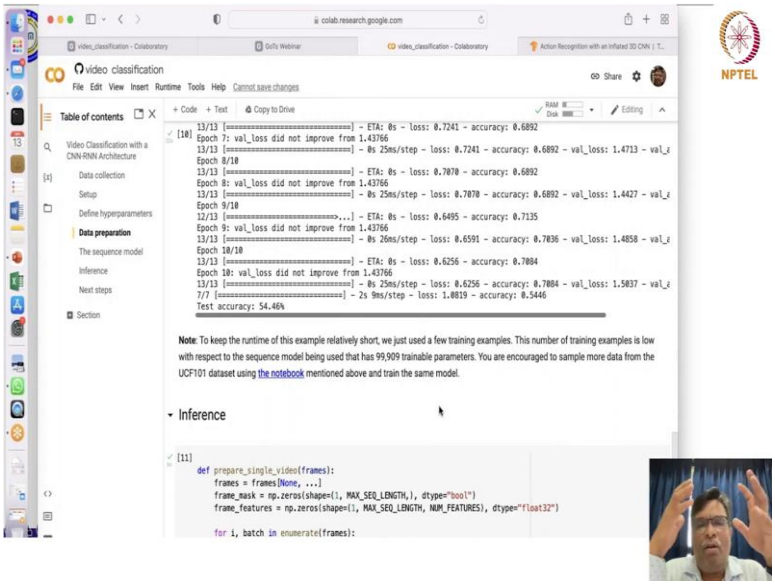
And then if you see the frame features in the training set is this and frame masks in the training set is this, now this we have already executed because it will take time right.

(Refer Slide Time: 08:48)



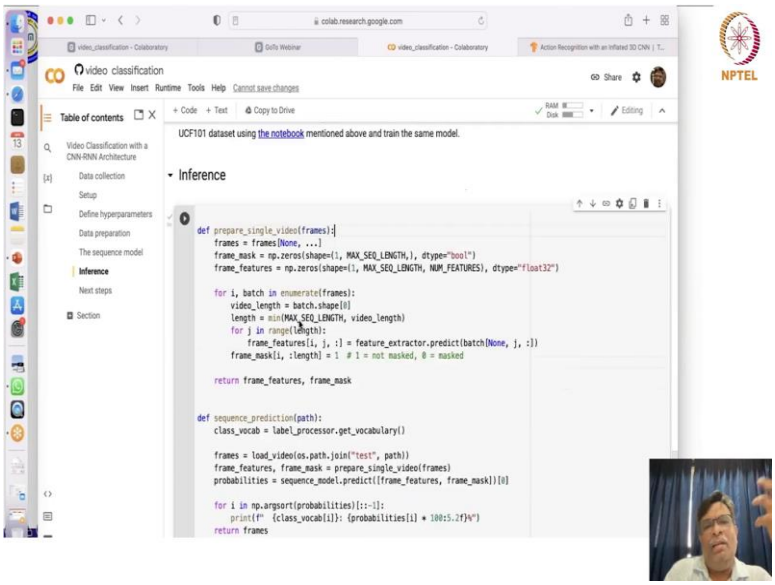
And then the sequence model is basically to feed this data to a sequence model ok of recurrent layers ok. So, what basically is being done here is, you are going to understand right why these mask models are required right. And then you basically try to run these experiments, train it ok for each of these epochs which we have already done.

(Refer Slide Time: 09:16)



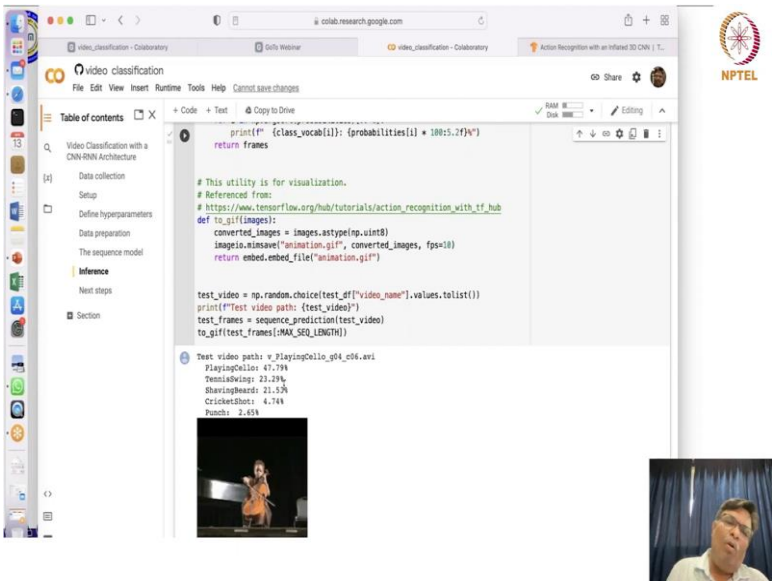
And then once this model is trained ok now this particular thing is basically is something like a sequence model with so, many trainable parameters right. And then using this UCF101 data set and then you basically can use this for inferencing ok.

(Refer Slide Time: 09:46)



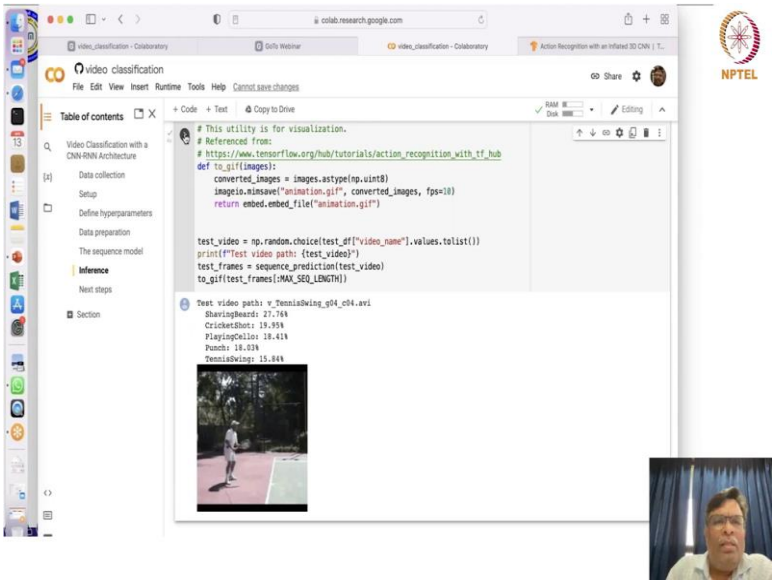
So, the idea is this that, if you run this particular thing ok it will randomly take certain images and then do the prediction.

(Refer Slide Time: 09:53)



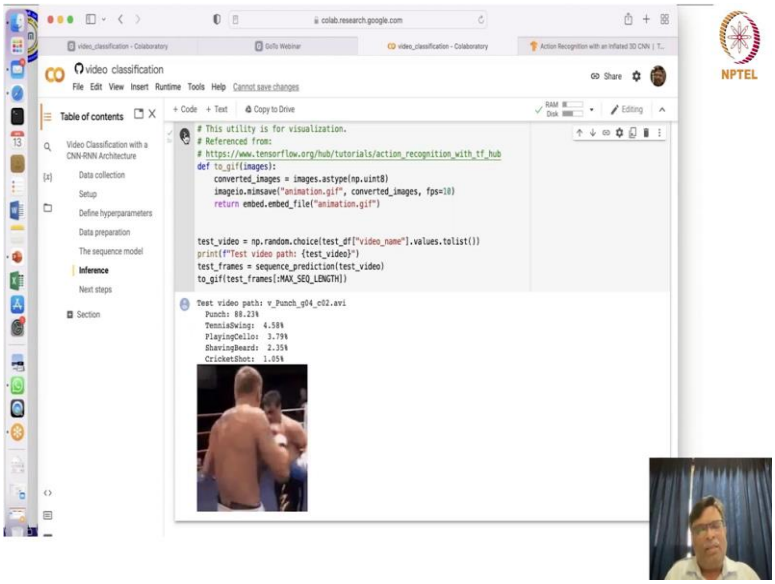
For example, this is the video path what is it doing? It is classifying it as playingaCello 47.79 percent it is not surely related to tennisswing, but it is basically trying to give it that particular percentage right. And then you have to get shot and punch with respective values the basic idea is, you need to train it more right we trained it with this data set with certain specific number of epochs just to show you like how fast we can do this ok. Otherwise, you can do it for various settings of hyper parameters, then try to do it we will try to see this again it will randomly choose certain videos and see here.

(Refer Slide Time: 10:40)



This is trying to detect that it is 27.76 percent as ShavingBeard and TennisSwing it gives me as 15.84 percent. So, this effectively is a wrong classification right. So, it is not necessary that every classification should be correct because of your training parameters, training values training epochs and all of that you understand that right. So, that is how it is.

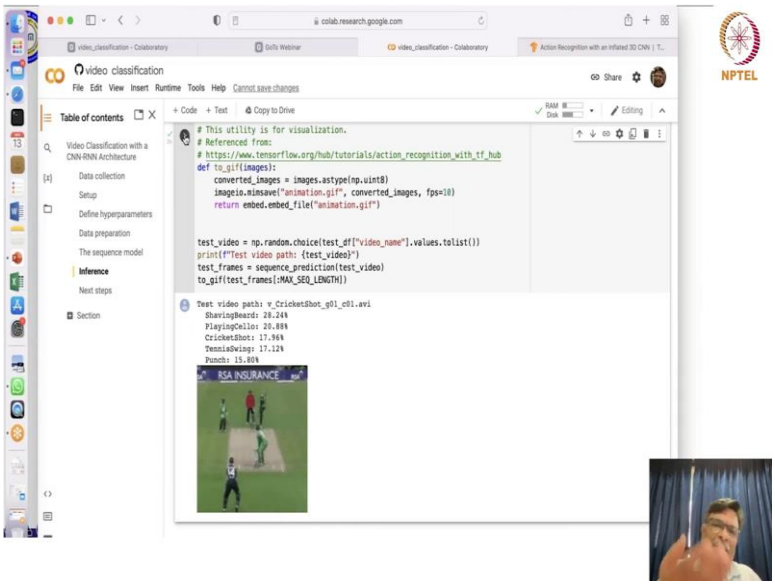
(Refer Slide Time: 11:20)



So, let us try to see one or two more videos and then we will see ok of what to be yes. So, this is randomly taking pictures ok which are not used for training and then this is

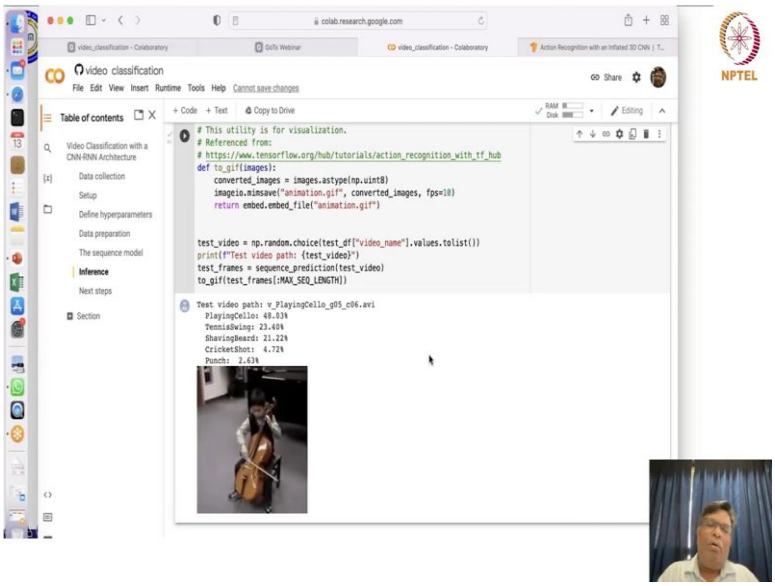
how it is going to do this right. So, what effectively is happening here is, these video classifications are not correct ok some of them let us try to see one or two more.

(Refer Slide Time: 11:50)

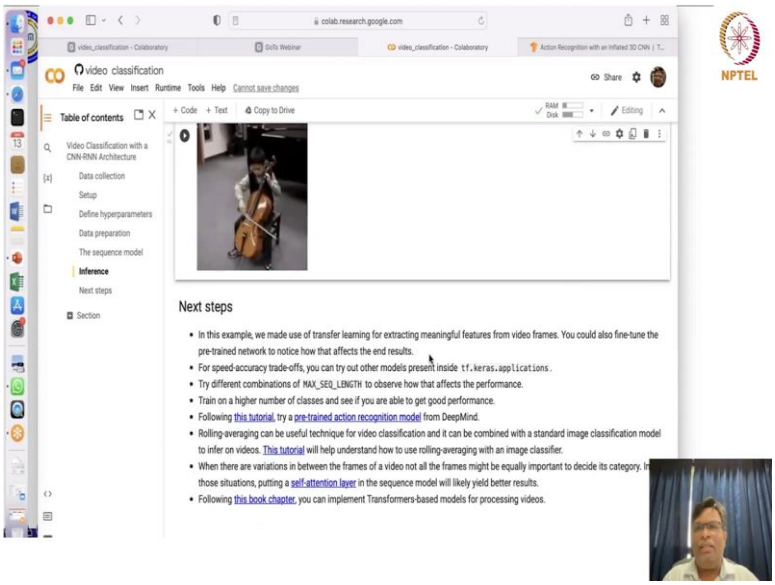


And let us try to understand like what is the accuracy see here. It is such a easy thing right, but still it is giving me shaving beard right.

(Refer Slide Time: 12:02)

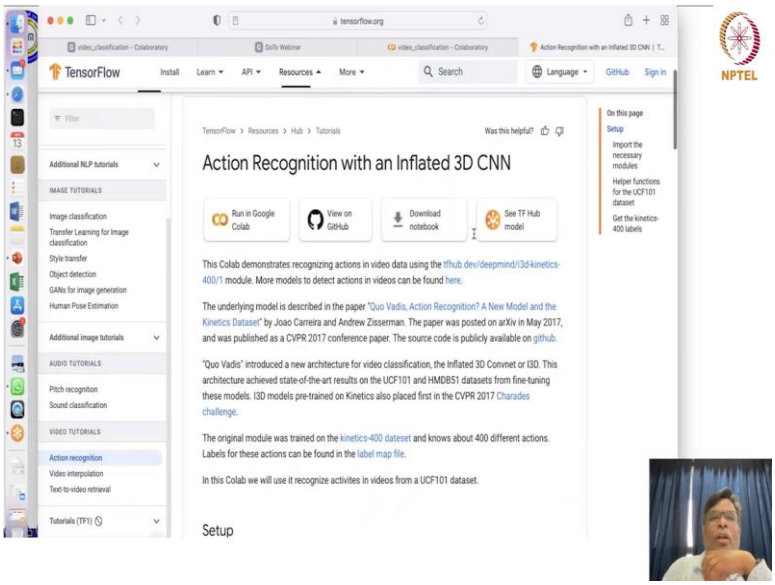


(Refer Slide Time: 12:11)

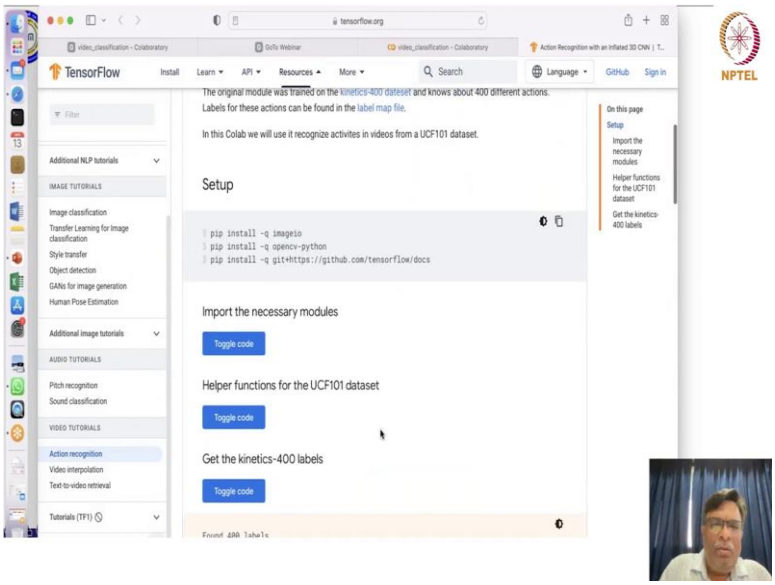


So, yes, so basically to remove all of this right, to remove all of this there is a concept are there is this particular approach right which is called as inflated 3D CNN usage right.

(Refer Slide Time: 12:36)



(Refer Slide Time: 12:41)

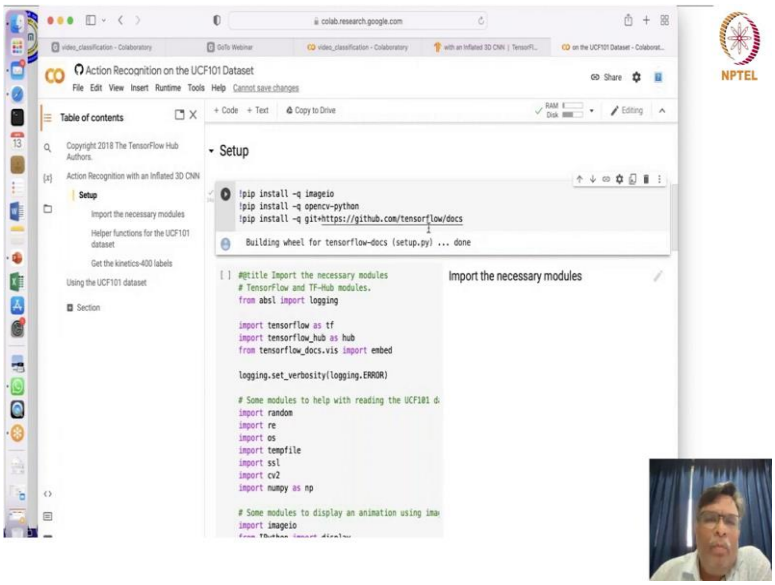


Now, what is basically inflated 3D CNN approach for action recognition is that it also does the same thing and you can use the kinetic data set right which will improve right your recognition capability. So, what is the difference? Here the input again is a video. So, the idea is it is a 3D input with a two dimensional frame with time as the third dimension ok.

So, technically speaking it is not a 3D model, but it is considered to be a three dimensional input it is considered to be a three dimensional input ok as a two dimensional image frame right and time as the third dimension. So, in this particular example we have tried to actually see right that these people have used a CNN with two strides or stride two after which you have a max pooling layer and multiple inception modules right. So, this is how it is.

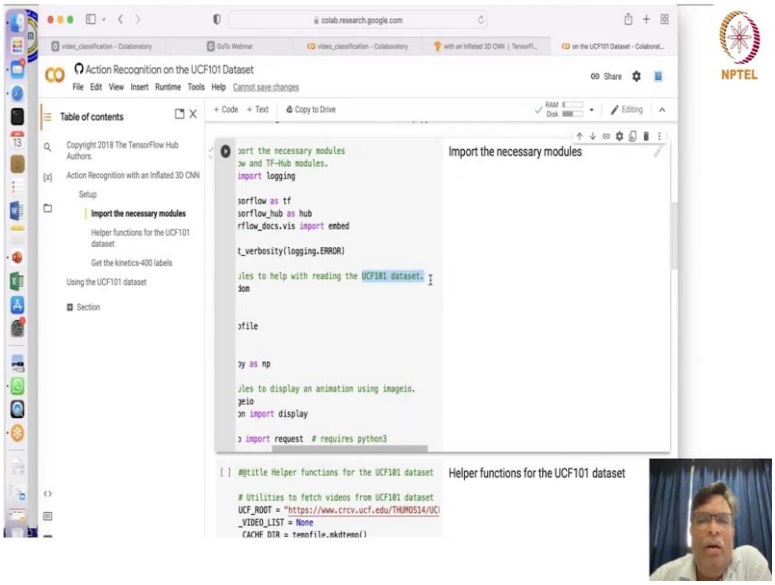
So, in the previous thing what I have done is, I had actually run it before, but now what I am going to do is I am going to run it so, that. So, let me just run in the colab thing now give me some time yes.

(Refer Slide Time: 14:59)

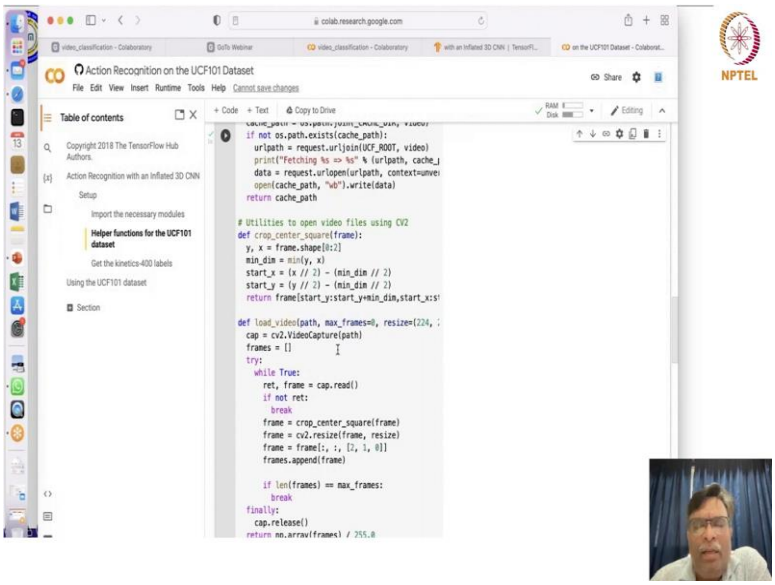


So, ok let me just close that session previous one ok. So, in this case this is basically a solution which was at just the first rank in CVPR 2017 charades challenge right. So, ok once we are able to install the various modules in the our environment we will just try to understand and import the various necessary modules and again here we are using UCF101.

(Refer Slide Time: 15:48)

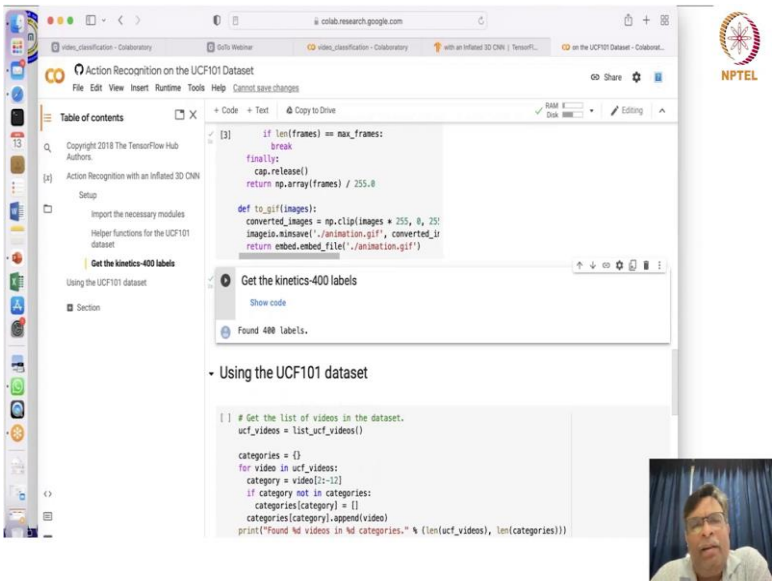


(Refer Slide Time: 16:04)



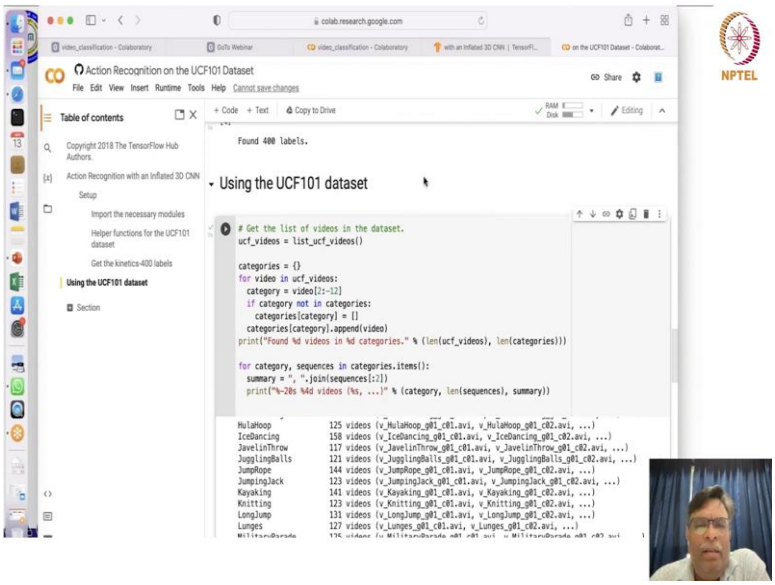
And then yeah again the same thing we are trying to open the video file using CV2 and then crop the center square frame and then trying to load everything right.

(Refer Slide Time: 16:18)



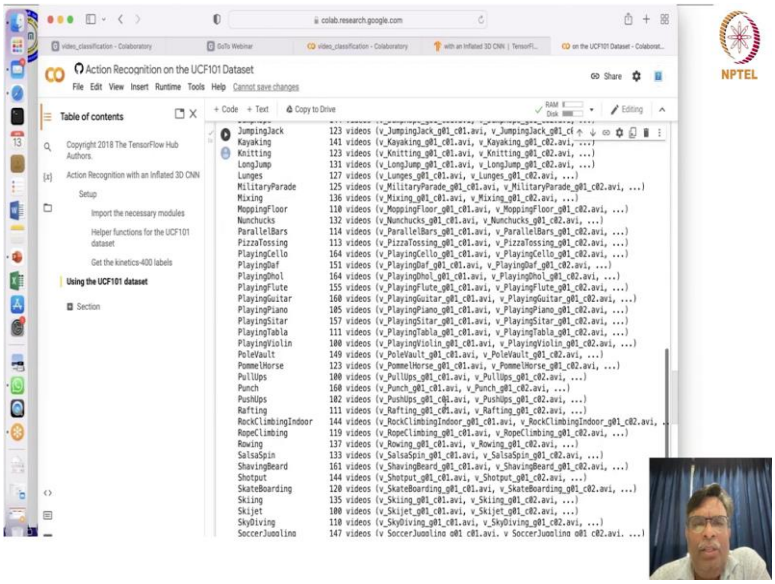
And then for each of these videos we are trying to get the kinetic right cameras labels.

(Refer Slide Time: 16:28)

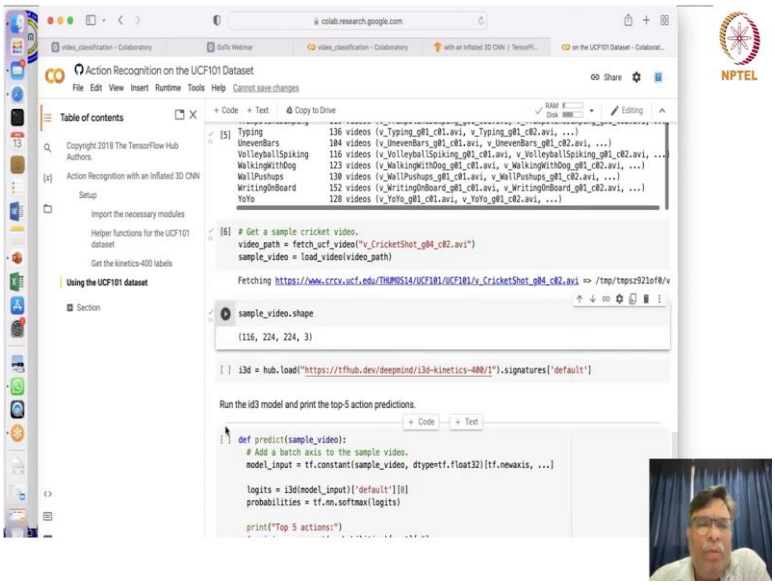


And then; so, what is the list of videos in the data set right?

(Refer Slide Time: 16:00)

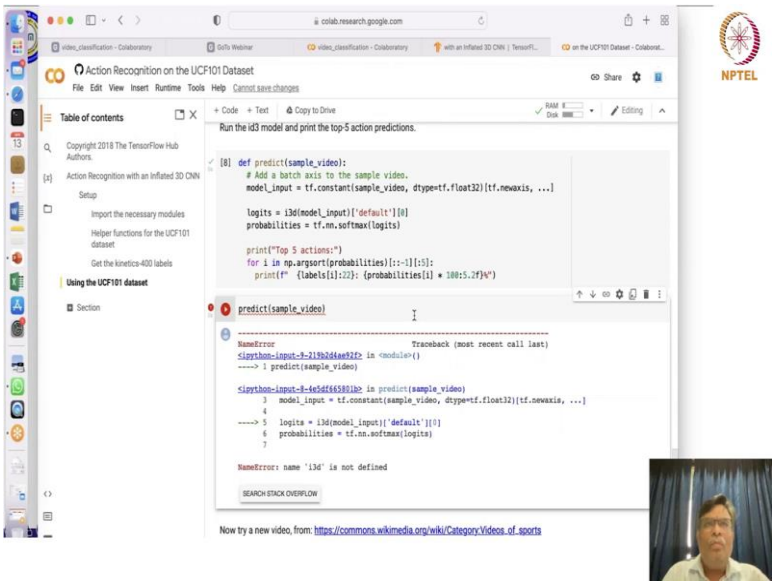


(Refer Slide Time: 16:39)



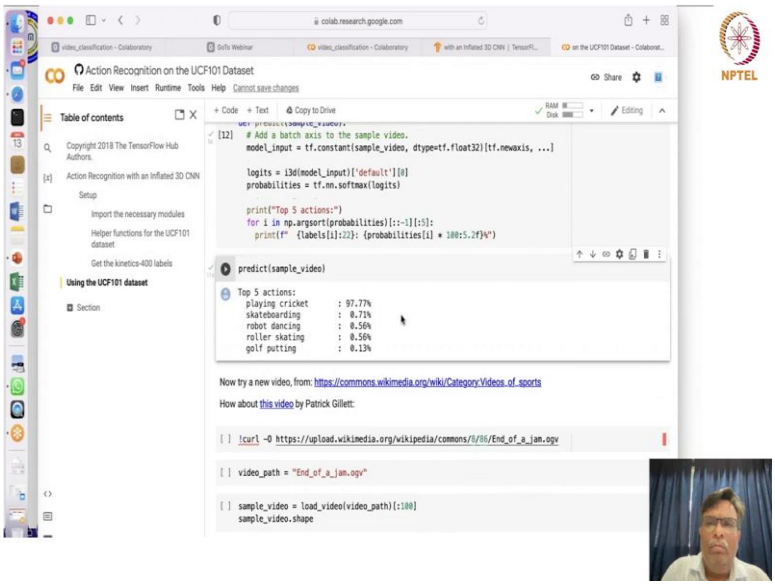
All of this videos in the initial this thing and then it is trying to show you certain things wherein.

(Refer Slide Time: 16:59)



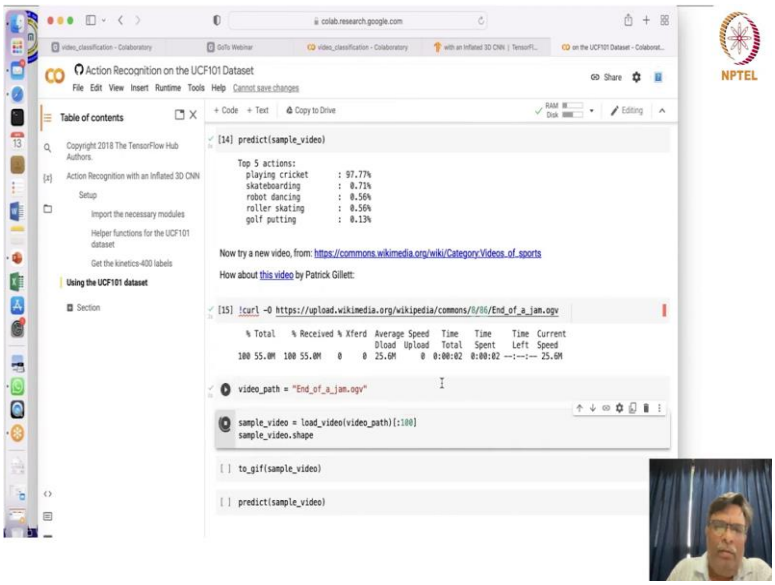
Let us try to; miss this ok.

(Refer Slide Time: 17:32)



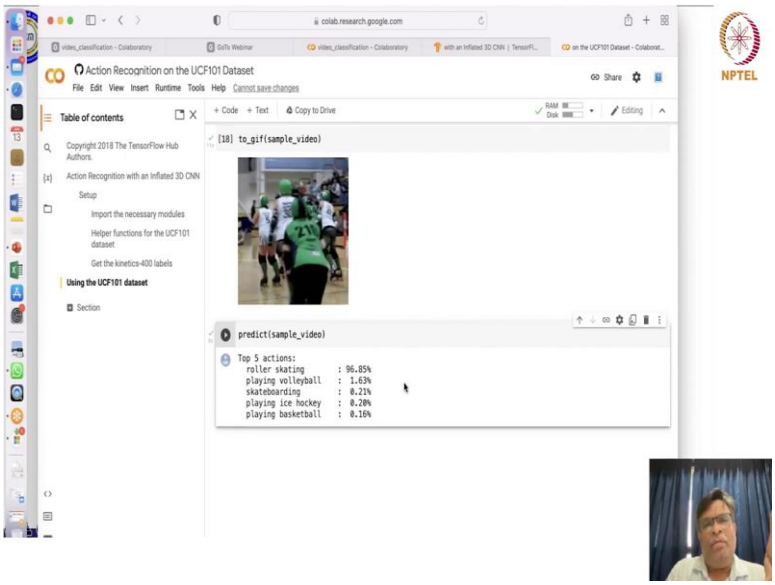
So, this sample video which you choose will give you the probabilities right it is not displaying what it will give you the probabilities.

(Refer Slide Time: 18:19)



And then you can basically try to download certain videos ok trying to understand then.

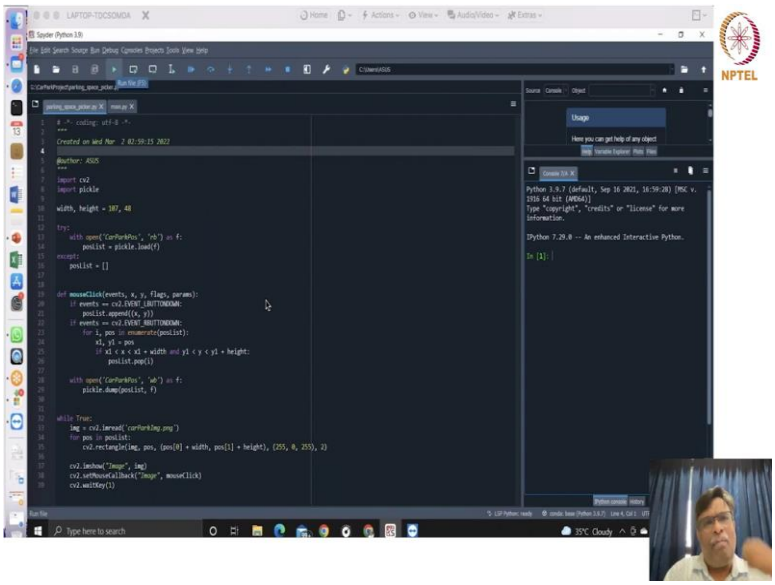
(Refer Slide Time: 18:59)



Yes. So, if you see this, this basically is trying to give you a prediction which is about 96.85 percent ok. Now, the idea here is that you have done almost the same thing, but then you have tried to actually link ok the kinetic label right with basically trying to use the kinetic data set ok along with the UCF101 and HMD basically data sets for fine tuning the models ok. And now these basic models which we are talking of a 3D convnet ok if you go to the 3D convnet model ok these are free trained models which are available to us.

So, 3D convnet model basically are available to us and then yeah. So, that is what basically is done in this particular thing I am not going into the technical details of 3D convnet and all you can basically refer the various links which are there here and then this thing which can happen right. So, now you can predict anything you can work on this in your leisure time and you can work basically on this, we have shared the links on our slides as well yeah. So, let me just show you some other demos one minute give me a sec.

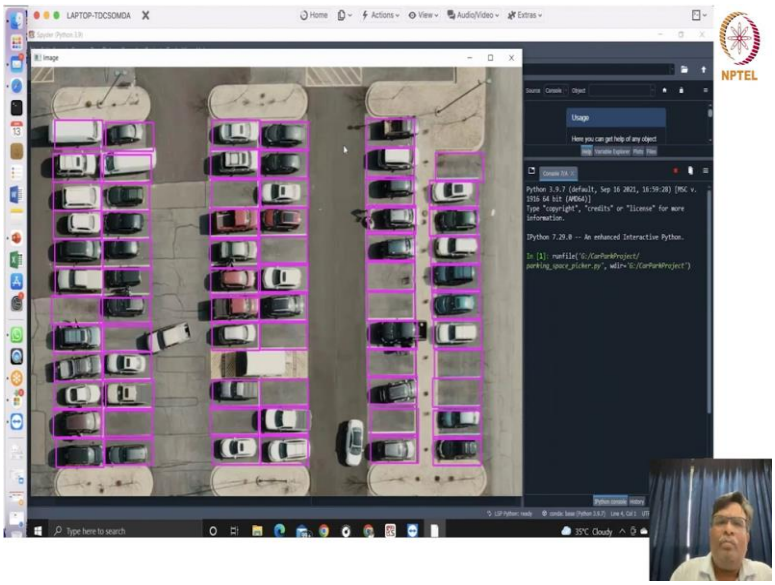
(Refer Slide Time: 20:54)



So, the idea is that this is a very simple program which we have been doing like everybody does it, but the idea is just to show you right how basically you can further automate it right which we will be doing it in the next class with deep stream right. So, the idea was how standard applications like this could be improved right using deep stream and GPU computing is what we are going to show you in the next class so, but we will show you how this is stand in the standard procedure done right.

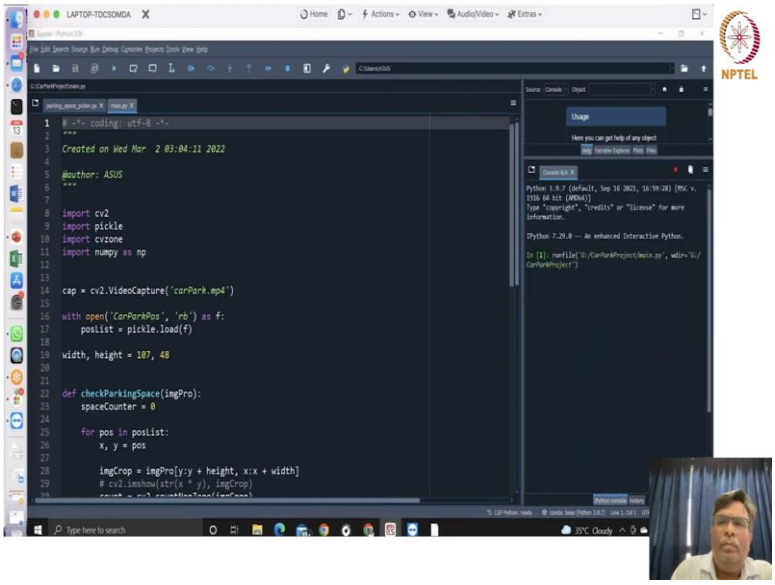
So, we have this width and height. So, trying to understand that and then trying to basically use mouse clicks to basically find out the areas which you are going to select. So, we will show you that and then we will see how this car parking thing works right. So, we will try to show you this yeah yes.

(Refer Slide Time: 21:55)

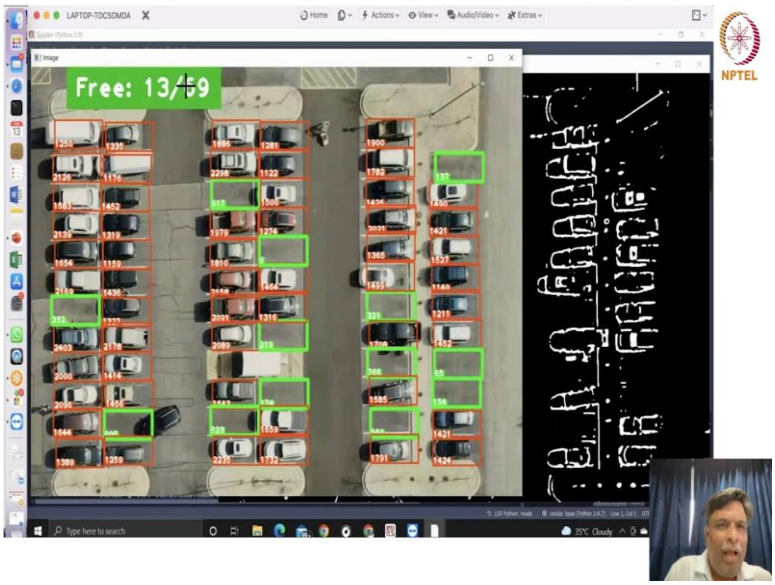


So, if you see this one minute. So, this is space for drawing yeah. So, this is how actually you are going to draw it ok yeah. So, this is how you are actually going to draw it and then now you can see like this we have put in and drawn ok for majority of this trying to understand this. And then trying to you can add in go on adding more areas and something like that that should not be a problem. So, this is the basic idea and now if you see and we start the next step of processing yes.

(Refer Slide Time: 22:52)



(Refer Slide Time: 22:57)



So, if you can see here it is actually going to show us the free slots the count of number of occupied slots and how basically the frames are getting processed ok and then thresholds are getting checked right. So, this is the standard computer vision application ok which might run on a GPU which might not run on a GPU. So, this is the standard application wherein you are trying to detect the free slots ok based on the features available using certain techniques of thresholding.

These are not very standard in the sense it is a standard technique, but it does not use any deep neural network type of approach right. So, how are we going to change this application ok? To a real time smart city application using something which is called as deep stream which is what we are going to discuss in the next session right.

So, I hope you understand the context with which this is told today so, that we can actually go into the details of certain specific applications which would be developed using things like deep stream ok which is a very very good set of libraries right which could be used for developing video analytics applications right. So, I suppose I am done.