

**Vision and Perception project**

**Topic: Mask\_RCNN and LSTM Video Classification**

**presented: Gbenimachor Alex Ugochukwu, Sajja Neeha Chowdary, Nithin Reji Johnson and Kanan Rahimli Husayn**

**ABSTRACT**

Computer Vision have gotten to the level, where a lot people are researching on various ways to improve the dash cam on an autonomous vehicle or the vision of the robot and also for improving the imaging application for the radiography and MRI.

With the applications, like Mask\_RCNN and other deep learning application designed to improve vision the application we might have chance of improving the way the machines will learn how to view and interact with world around them.

The main of the paper, is the get understanding of how we can improve the vision of the machines and also improve understanding about the activity performed by their human counterparts, like said improving their interaction with world.

**INTRODUCTION**

Before we go into, the main event which is the Video classification with LSTM, we will talk a little bit about the mask which was the build block to th next next part of our project.

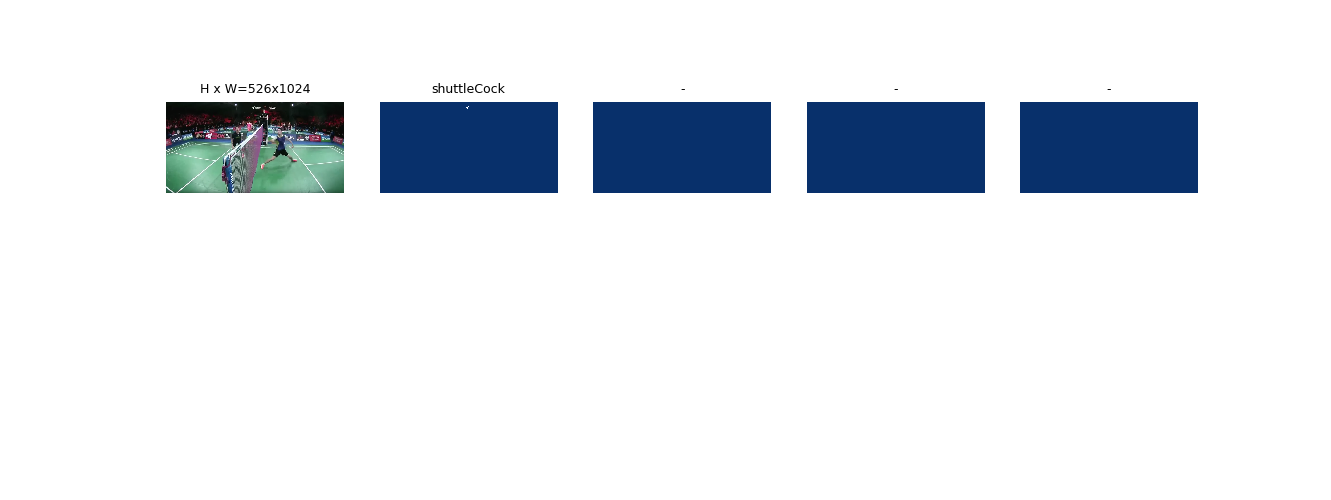
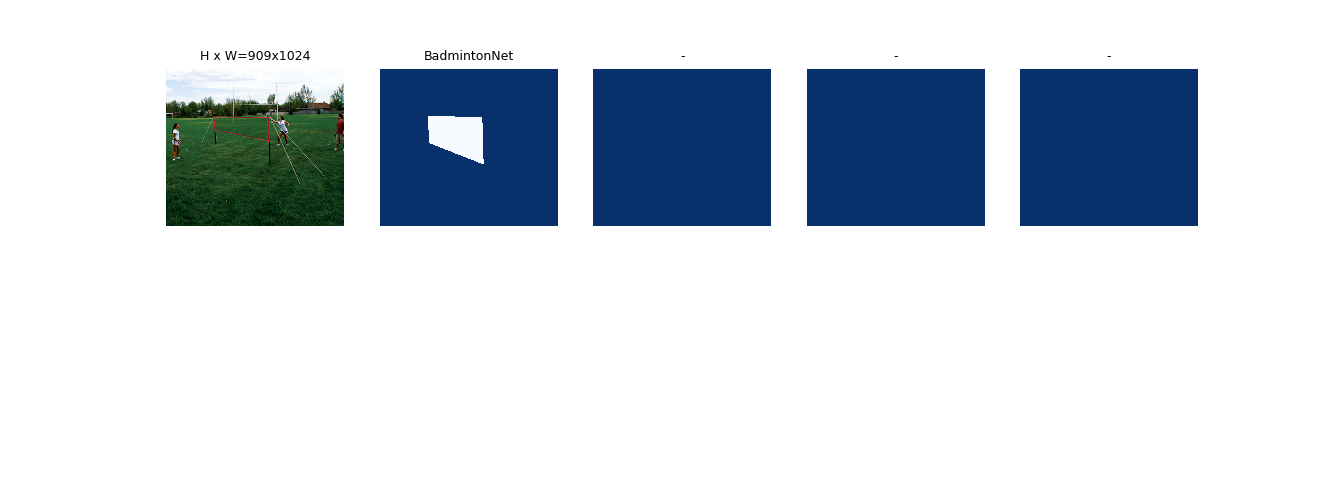
Mask RCNN, is an extension of the Fast RCNN which is developed by Ross Girshick et al use in learning region proposal in real time. The Mask RCNN extends the Fast RCNN with Instance Segementation, which is the task of identifying outlines of objects at Pixel level, Waleed Abdulla. Aim of the Mask\_RCNN is identifying images using pixels.

**1.MASK RCNN**

Mask\_RCNN(Regional Convolutional Neural Networks) is a two stage framework, which involve the scanning of Groundtruth images and generates proposal of images with the objects and the next task is classifying these images.

During the project we faced the most challenging part of getting images and performing the instance segmentation of the images, extracting the position of these images and passing the coordinates to the Mask\_RCNN with images as input to generate bounding boxes and mask.

Below is the image of what goes on the background mask:



**The End Results of our Training is given below:**



**2.Next Part : Video Classification with LSTM**

**2.1.INTRODUCTION**

there are two significant research areas on the comprehension of videos: video classification and video captioning. Video classification, focuses on automatically labeling videos based on video contents and frames, while video captioning is to generate short descriptions for videos and capture dynamic information such as human actions etc. Specifically, a big challenge in video classification and captioning is to fuse different video frames over time. In our case, we use video classification with Tensorflow & Keras. Our approach is VGG16+Single High Dimensional LSTM of 256: this approach uses VGG16 to extract features from individual frame of the video, the sequence of frame features are then taken into LSTM recurrent networks for classifier.

**2.2.RELATED WORK**

Video classification has sequential frames input. The basic idea is to classify every frames per video and then output the video category based on the results of majority vote [1]. In this case, video classification is treated as image classification, which, however, cannot keep the sequential information of videos. In terms of using sequential frames . clearly describe two approaches: 3D temporal max pooling and Long Short Term Memory (LSTM) [3]. It turns out LSTM [3] can maintain the dynamic contents of videos to make more accurate predictions.

**2.3.OUR APPROACH**

First, we performed a transfer learning of the video frames and labels as input to our Single High Dimensional LSTM to perform the classification of the Frames.

The following steps are done:

• All frames are extracted and converted with labels as the activity name.

• Extraction of a sequence of images from videos with a constant size and equally spaced

• Saving the extracted features into a npy file so it can be reused further.

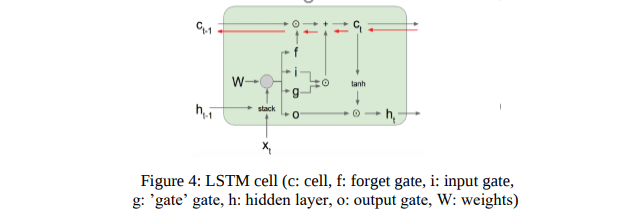
• Feeding these npy files through LSTM

• Training LSTM RNN for classifying videos based on the image frames.

**3.** **[CNN+LSTM]**

**[CNN]**

The use of LSTM for video classification, based on research began recently as 2017 and over the the years, people have come with lots of implementation of these video classifications. We will discuss a little bit about the LSTM. A common LSTM unit is composed of a cell, an input gate, an output gate and a forget gate. The cell remembers values over arbitrary time intervals and the three gates regulate the flow of information into and out of the cell. LSTM also outputs a hidden vector for each input activation frame. As shown in Figure 1, each cell (c) in LSTM layers accepts stacked ht-1 and xt as inputs. The inputs enter four gates after dot producting with weights (W). Each gate has different functions. Compared to the vanilla recurrent neural network, LSTM has uninterrupted gradient flow, which is easier to back-propagate. LSTM is also more stable without gradient exploding or vanishing. Referring to [3], we build up our LSTM architecture for video classification, as shown in Figure 4 and Table 4. The first LSTM layer accept the inputs from the pretrained CNN model, and then the second LSTM receives sequential outputs from the previous LSTM layer. In the end, we apply the output into fully-connected layer and calculate the final softmax score.



**LSTM** is a recurrent neural network (RNN) architecture that **REMEMBERS** values over arbitrary intervals. LSTM is well-suited to classify, process and predict time series given time lags of unknown duration. Relative insensitivity to gap length gives an advantage to LSTM over alternative RNNs, hidden Markov models and other sequence learning methods

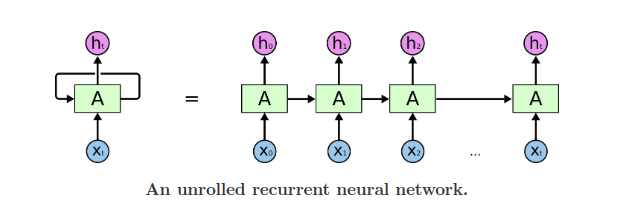
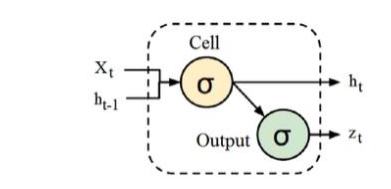


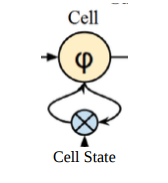
Image by Christopher Colah

The structure of RNN is very similar to hidden Markov model. However, the main difference is with how parameters are calculated and constructed. One of the advantage with LSTM is insensitivity to gap length. RNN and HMM rely on the hidden state before emission / sequence. If we want to predict the sequence after 1,000 intervals instead of 10, the model will forget the starting point by then. LSTM REMEMBERS. What is the architecture which allows LSTM to REMEMBER?

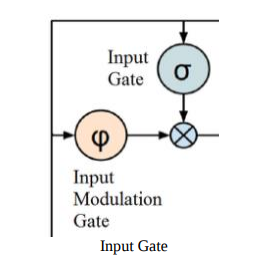


RNN cell takes in two inputs, output from the last hidden state and observation at time = t. Besides the hidden state, there is no information about the past to **REMEMBER**

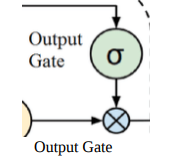
The **long-term memory** is usually called the **cell state**. The looping arrows indicate recursive nature of the cell. This allows information from previous intervals to be stored with in the LSTM cell. Cell state is modified by the forget gate placed below the cell state and also adjust by the input modulation gate. From equation, the previous cell state forgets by multiply with the forget gate and adds new information through the output of the input gates.



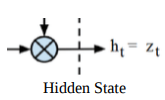
The **remember vector i**s usually called the **forget gate**. The output of the forget gate tells the cell state which information to forget by multiplying 0 to a position in the matrix. If the output of the forget gate is 1, the information is kept in the cell state. From equation, sigmoid function is applied to the weighted input/observation and previous hidden state.

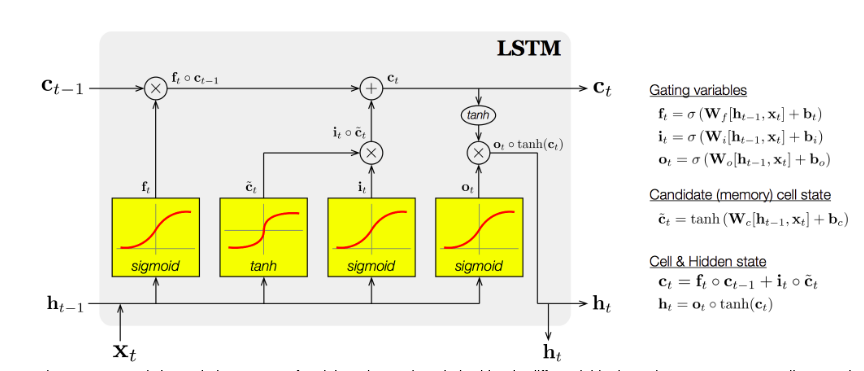


The **save vector** is usually called the **input gate**. These gates determine which information should enter the cell state / long-term memory. The important parts are the activation functions for each gates. The input gate is a sigmoid function and have a range of [0,1]. Because the equation of the cell state is a summation between the previous cell state, sigmoid function alone will only add memory and not be able to remove/forget memory. If you can only add a float number between [0,1], that number will never be zero / turned-off / forget. This is why the input modulation gate has an tanh activation function. Tanh has a range of [- 1, 1] and allows the cell state to forget memory.



The **focus vector** is usually called the **output gate**. Out of all the possible values from the matrix, which should be moving forward to the next hidden state?



The working memory is usually called the **hidden state**. What information should I take to the next sequence? This is analogous to the hidden state in RNN and HMM.

Another View of LSTM Cell The first sigmoid activation function is the **forget gate**. Which information should be forgotten from the previous cell state (Ct-1). The second sigmoid and first tanh activation function is our **input gate**. Which information should be saved to the cell state or should be forgotten? The last sigmoid is the **output gate** and highlights which information should be going to the next hidden state.

Below the is the Hyper-parameter of our LSTM

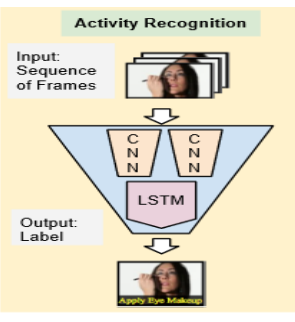
Hidden Size = 256

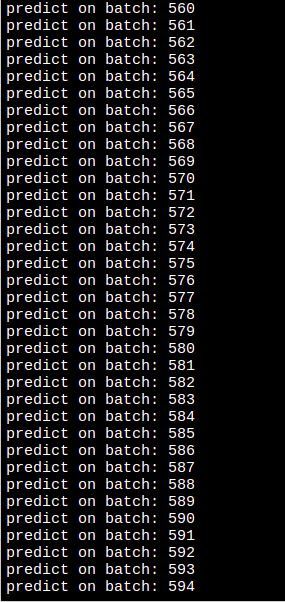
Batch\_size = 128

Number of Epoch = 500

learning rate = 0.00005

Once we have all the vector files, then we run the LSTM with this configuration:



**4. VIDEO CLASSIFICATION TRAINING PROCESS**

**5. RESULT AND CONCLUSION**

We achieved an 81% accuracy on frames using CNN-LSTM architecture, 77% accuracy on videos using CNN-LSTM.

In Future, we hope that these will be implemented into autonomous machines and robot, like we mentioned Earlier to improve the way Visualize the world.