HUMAN ACTION RECOGNITION

DEEPAK PANT
2017CSB1072
Indian Institue of Technology Ropar

Abstract

One of the most important task of Video analysis is to determine the action that is being done in it and predicting what would happen next. Recognition of actions performed in the video are a prerequisite for determining the future state. Vision-based action recognition and prediction from videos are such tasks, where action recognition is to infer human actions (present state) based upon complete action executions. This task is an important part of designing intelligent visual surveillance, self-driving cars and retrieval of videos given a text of information about it.

Many attempts have been devoted in the last a few decades in order to build a robust and effective framework for action recognition and prediction.

In this project, I tried three approaches, 2 of them with 2DCNN and 1 of them with 3DCNN. The second approach proved to be the best with 40 percent test accuracy and 77 percent validation accuracy on UCF 101, a dataset consisting of over 13000 videos and having 101 output classes.

1 Introduction

The term human action studied in computer vision research ranges from the simple limb movement to joint complex movement of multiple limbs and the human body. This process is dynamic, and thus is usually conveyed in a video lasting a few seconds.

Action recognition is a fundamental task in the computer vision community that recognizes human actions based on the complete action execution in a video. Action recognition and prediction algorithms empower many real world applications.

A visual surveillance system powered by action recognition algorithms may increase chances of preventing and recognising criminal activities and reduce the damage caused by such actions.

Video retrieval is an important part of systems which provide today's mode of entertainment and education. However currently it is done by attached metadata, text, tags and keywords. These factors can be incorrect, irrelevant and obscure making error rate high for video retrieval. Human action recognition helps analyze actions performed in the video and can result in a much accurate video retrieval system.

Despite significant progress has been made in human action recognition and prediction, state-of-the-art algorithms still mis-classify actions due to several major challenges in these tasks.

Same tasks are performed differently by different humans. Actions in one category can be wide ranging throughout the dataset. In addition, perspective of capturing video makes it quite difficult. Different people may show different poses in executing the same action. All these factors confuse a lot of existing action recognition algorithms. Furthermore, similarities exist in different action categories. For instance, running and walking involve similar human motion patterns. These similarities would makes it challenging to differentiate and consequently contribute to mis-classifications.

Current action recognition algorithms show impressive performance on available datasets. However they are unable to generalize it for real world due to these datasets being quite smaller than what is required.

Also not all frames are equally discriminative. Videos may contain a small set of key frames thus making most of the frames redundant and finding the discriminative frames is quite hard.

Also these algorithms require annotated datasets which are obviously of much smaller size. Action datasets such as HMDB51 and UCF-101 contain thousands of videos, but still far from enough for training deep networks with millions of parameters.

2 Literature Review

This task's variability makes it quite difficult for traditional machine learning approaches to achieve high accuracy. Most of the state of the art algorithms require deep neural architectures to function.

One of the promising approach is twostream fusion. In this two deep ANNs are used - one for spatial discrimination and other for temporal discrimination. Combining temporal net output across time frames so that long term dependency is also modeled. This results in 94.2 percent accuracy on UCF-101 dataset.

The best performing approach currently is I3D. Instead of a single 3D network, authors use two different 3D networks for both the streams in the two stream architecture. Also, to take advantage of pre-trained 2D models the authors repeat the 2D pre-trained weights in the 3rd dimension. The spatial stream input now consists of frames stacked in time dimension instead of single frames as in basic two stream architectures.

3 Method and Experimental Results

First Approach (2DCNN): I extracted all the frames of a video and added them to a list of images. For reducing memory usage I extracted only the first 6 frames of every video with each frame being downsized to 3 channels of 112*112 pixel image.

Then I split them to create a validation set to determine whether the model was overtraining. They were passed through VGG19 to extract relevant features. VGG-19 is a convolutional neural network that is trained on more than a million images from the ImageNet database [1]. The network is 19 layers deep and has learned rich feature representations for a wide range of images. The network has an image input size of 224-by-224. This network is characterized by its simplicity, using only 3ÃŮ3 convolutional layers stacked on top of each other in increasing depth. Reducing volume size is handled by max pooling. Two fully-connected layers, each with 4,096 nodes are then followed by a softmax classifier.

These features were then passed through a custom neural network to learn the labels. This model consisted of various dense layers with ReLU activation and dropout layers for regularization. At the end was a 101 size output layer with softmax activation giving probabilty distribution of the given image to corresponding classes. The classes were one-hot encoded. The model used Adam optimizer and categorical cross-entropy as its loss function. This model ran for 200 epochs and scored an validation accuracy of 23 percent and loss of 3.7.

The resulting weights were then used for prediction of test videos. These videos were extracted to frames using the same method as above. The mode of resulting classification was used to classify the video. This resulted in 15 percent test accuracy.
One of the reason of low accuracy was of not shaping the image to the one used by the input layer of VGG19 which resulted in a loss of its accuracy.

Second Approach(2DCNN): This approach is quite similar to the first one. I extracted the frames now at the rate of 0.5fps. The reason for this was because earlier only a part of video was being analyzed which might have been redundant in significant part of the dataset.

Also I was disregarding later information. I also resized the video frames to 3 channels of 224*224 pixels. Similar to last time I passed the images through VGG19 to extract its features. These were again passed through a model of same dimensions and optimizer and loss function. This model ran for 200 epochs and scored a validation accuracy of 83 percent and train accuracy of 96 percent.

The test videos were extracted similarly and the mode was taken to determine video's class. This resulted in 45 percent accuracy.

The discrepancy of quite dissimilar validation and test accuracy is because frames of all videos were indiscriminately put in train and validation sets making validation much easier than testing.

Third Approach(3DCNN): This approach is similar to 2nd approach till creating feature vectors of every frame of a video. After that it stacks them all to create a 3D representation of the video. Only 3 frames per video were used. This is feeded to a custom neural network of 3DConvNets with dropout layers and average pooling. This network had Adam optimizer and loss function as categorical crossentropy. This approach is not tested yet.

4 Conclusion

In this project, I developed a video classifier for common human actions. The Dataset I used was UCF-101. Different approaches lead to following conclusions: The pretrained networks work best when used according to their specification. Not using them accordingly made the accuracy substantially worse. Also using frames sparsely in videos is better than sequential frames.
The average performance of the model is due to disregarding temporal information. RNN on frames of the video may enhance the model's performance substantially. Using ConvNets on both spatial frames and time aka fusing result of two streams may also increase performance.