Action Recognition Using Recurrent Neural Network

Sai Bharat Kumar Konakalla
50336876
University at Buffalo
skonakal@buffalo.edu

Pranay Reddy Dava
50340873
University at Buffalo
pranayre@buffalo.edu

Abstract

Research in human action recognition has accelerated significantly since the introduction of powerful machine learning tools such as Convolutional Neural Networks (CNNs). However, effective and efficient methods for incorporation of temporal information into CNNs are still being actively explored in the recent literature. Recurrent neural network (RNN) and long short-term memory (LSTM) have achieved great success in processing sequential multimedia data and yielded the state-of-the-art results in speech recognition, digital signal processing, video processing, and text data analysis. In this report, we propose a novel action recognition method by processing the video data using convolutional neural network (CNN) and LSTM network.

1. Introduction

Action recognition in videos are imperative elements of computer vision research. Over the last few years, deep learning techniques dramatically revolutionized research areas such as image classification, object segmentation and object detection. Likewise, Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) have been popular in the action recognition task. However, various network architectures have been proposed with different strategies on the incorporation of video temporal[5] information. However, despite all these variations, their performance improvements over the fine tuned image classification network are still relatively small.

The action recognition is not same as the image classification, the most distinctive property of the video data is its distinctive length. Images can be readily sized to the same spatial resolution, while for videos it is difficult to subsample videos temporally. Therefore, we can say it is difficult for 3D CNN to achieve action recognition performance.

Effective and efficient methods for incorporation of temporal information into CNNs are still being actively explored in the recent literature. Convolutional neural network (CNN) models have been extensively used in recent years to solve the problem of image understanding giving state-of-the-art results in tasks like classification, recognition, retrieval, segmentation and object detection. Motivated by this success there have been several attempts to extend convolutional neural networks for video understanding and classification. An important distinction between images and videos is the temporal information that is encoded by the sequence of frames. Most CNN models fail to capture this temporal information. Recurrent neural networks have shown promising results in modelling sequences. In this work we present a neural network model which combines convolutional neural networks and recurrent neural networks.

Project focuses on recognizing the action performed in a video using Recurrent neural Network and CNN with high end architectures. We are using UCF101 dataset. UCF101 is an action recognition data set of realistic action videos, collected from YouTube, having 101 action categories. Human activity recognition, or HAR, is a challenging time series classification task. It involves predicting the action of a person based on video clips and traditionally involves deep domain expertise and methods to correctly engineer features from the raw data in order to fit a machine learning model. The problem is to predict the activity, which is a multivariate time series classification task.

2. Dataset

We have used UCF-101[4] dataset which contains 101 actions and 13,320 videos in total. With 13320 videos from 101 action categories, UCF101 gives one of 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. 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 recogni-

Actions	101
Clips	13320
Groups per Action	25
Clips per Group	4-7
Mean Clip Length	7.21 sec
Total Duration	1600 mins
Min Clip Length	1.06 sec
Max Clip Length	71.04 sec
Frame Rate	25 fps
Resolution	320×240
Audio	Yes (51 actions)

Figure 1. Summary of Characteristics of UCF-101

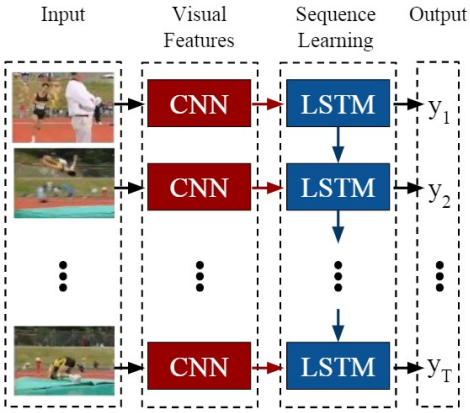

Figure 2. Data Flow in this project[2]

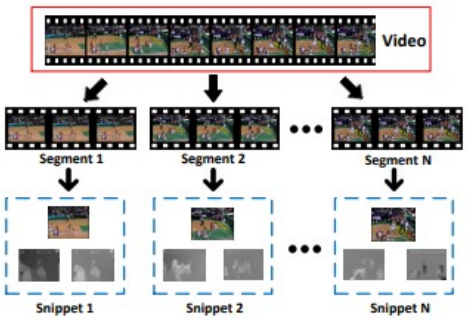

Figure 3. Relevent Frame Extraction from Videos[3]

tion by learning and exploring new realistic action categories. 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.**

2.1. Clip Groups

The clips of one action class are divided into 25 groups which contain 4-7 clips each. The clips in one group share some common features, such as the background or actors. The videos are downloaded from YouTube and the irrelevant ones are manually removed. All clips have fixed frame rate and resolution of 25 FPS and 320 × 240 respectively. The videos are saved in .avi files compressed using DivX codec available. The audio is preserved for the clips of the new 51 actions. Figure 1 summarizes the characteristics of the dataset.

3. Implementation

3.1. Data Flow

We used the Figure 2 to get an abstract understanding of what we are doing in this project. We will explain each of those modules in detail.

3.2. Preprocessing:

Videos are nothing but a pile of frames, so we have captured 40 frames for every video in our dataset. After capturing the frames from the videos, we used transferred learning from Inception V3 model which used ImageNet dataset for their training to extract features from the frames. We extracted 2048 features from every frame using CNN that is Inception pretrained weights. The frames extracted from the videos are then resized to (299,299,3) dimension as the input shape to Inception V3 model is fixed. While extracting frames from the videos we took the frames in such a way that the sequence of frames describes the whole video. For every video, the extracted feature output from CNN will be of (1,40,2048) shape as it contains 40 frames and 2048 features for every frame. In the dataset for every category of videos there are different sub-categories, like a particular sub-category has some 'n' number of videos, so to train the model well we have divided the frames extracted into training and testing as for every sequence of 5 videos 4 videos goes to training and 1 video goes to testing data so that the model will be exposed to every kind of video in the dataset.As mentioned above, each video is divided into different segements and then we chose a snippet from each segement to capture the whole information as shown in Figure 3.

3.3. Feature Extraction

We have used Transfer learning for feature extraction. Our problem in this case looked similar to Image-net problem (extracting features). Transfer learning is a machine learning method which utilizes a pre-trained neural network. Here, we used image recognition model, Inception-v3. This can be divided into two parts as Feature extraction part and classification part. Feature extraction is through

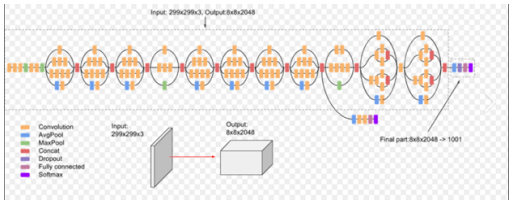

Figure 4. Schematic diagram of Inception V3.

convolution neural network and classification is through fully connected and softmax layers.As we are doing the feature extraction using transfer learning, we use only the feature extraction part but not the classification part. So, we eliminated the softmax layer and took the output from the layer before the softmax which is "AvgPool" layer. Avg means that global average pooling will be applied to the output of the last convolutional block, and thus the output of the model will be a 2D tensor. Thus we have (1,2048) feature vector for every frame.Feature extraction Neural network is done as shown in figure 4

3.4. Model Training

Our idea was to use recurrent neural network to capture the temporal dynamic behavior of the video. Although we can use Convolutional Neural Networks to derive the relationship between frames, that was not enough to capture the temporal relationship. We experimented this by using Convolutional neural networks alone to predict the accuracy .The result is not as good as RNN.

We used keras tensorflow package to create RNN's and also experimented with the following networks:

- SimpleRNN layer with tanh activation function
- SimpleRNN layer with relu activation function
- LSTM

Networks other than LSTM are facing vanishing gradient problems when trained on this data. LSTM worked better than the rest for our task. We have used LSTM layer with 2048 nodes and a dropout of 0.5. Also, we added fully connected dense layers to this LSTM to increase the depth and used this model to predict the category of a video.

LSTM networks are well-suited to classifying, processing and making predictions based on time series data, since there can be lags of unknown duration between important events in a time series. LSTMs were developed to deal with the vanishing gradient problem that can be encountered when training traditional RNNs. Relative insensitivity to gap length is an advantage of LSTM over RNNs, hidden Markov models and other sequence learning methods in numerous applications.

4. Methods/Techniques used

4.1. Preprocessing Methods

The video dataset consists of 101 categories. We retrieved 40 frames from each video which are later used to extract features. We used OpenCV package to capture frames in a video. Here are some more important packages which helped us doing the project. We used the idea shown in Figure 3.

4.1.1 OS

This package helps in traversing through the file system and retrieve files from the folder. The retrieved videos are sent to extract frames from it.

4.1.2 OpenCV

- VideoCapture method in OpenCV allows us to read images from video.
- Resize method is used to resize the image dimensions.

4.2. Feature Extraction Methods

To extract features from the images we used existing pre-trained model. We have used different custom and existing architectures to do this. Inception-V3 model with pre-trained weights in keras gave us the best results. One of the customized models with four Maxpool2D layers also gave a result almost as good as the pre-trained model. Also, we used different pre-processing ways to make the CNN understand the movement of the frames rather than extracting edges or other classic features.

This step became the bottle neck for our approach, extracting features for every frame (approximately 120-150 frames for a video) and for around 3000 videos took approximately 12 hours.

4.2.1 Keras

We used this package to create customized and use existing Neural Network models. This package is very flexible, it allowed us to customize everything needed for a model by using its in-built methods. We experimented a lot of neural networks but these are the one we used in our project.

- LSTM
- Inception-V3 Image-net model

4.3. Methods used in Training

4.3.1 Recurrent Neural Network

A recurrent neural network (RNN) is a class of artificial neural networks where connections between nodes form a

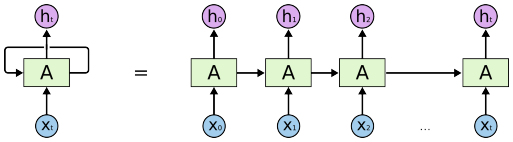

Figure 5. Recurrent Neural Network

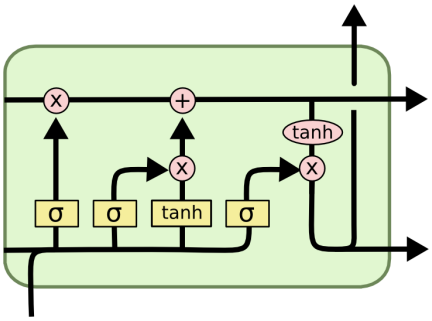

Figure 6. LSTM Network

directed graph along a temporal sequence. This allows it to exhibit temporal dynamic behavior. Derived from feedforward neural networks, RNNs can use their internal state (memory) to process variable length sequences of inputs. We used a variant of RNN(LSTM) for our project.Classic RNN and LSTM architectures are shown using Figures 5 and 6

4.3.2 Compilation Setup

We have used categorical cross entropy as loss function, adam as optimizer and TopKCategoricalAccuracy, accuracy as metrics. We tried Adam and sgd optimizers, sgd gave 'nan' loss values so we proceeded with Adam. As the model is very sensitive we used a low learning rate(0.000001) with decay.

TopKCategoricalAccuracy[1]: TopK Categorical Accuracy calculates the percentage of records for which the targets (yTrue) are in the top K predictions (yPred). We rank the yPred predictions in the descending order of probability values. If the rank of the yPred present in the index of the yTrue is less than or equal to K, it is considered accurate. We then calculate TopK Categorical Accuracy by dividing the number of accurately predicted records by the total number of records. We used this metric to see if our model is confusing between some video categories. Results have shown that some categories that looked similar confused the model.

Figure 7. Sample frames for 6 action classes of UCF101

5. Results

As per our thoughts we initially extracted features from Apply Eye Makeup, Apply Lipstick, Archery, Baby Crawling, Balance Beam, Band Marching, Baseball Pitch, Basketball Shooting, Basketball Dunk, Bench Press, Biking, Billiards Shot, Blow Dry Hair, Blowing Candles, Body Weight Squats, Bowling, Boxing Punching Bag which are of 20 categories and divided the first 90 videos as training and the remaining as testing in every category and sent to LSTM for classification. When this is performed we observed some unexpected results in our training process. Confusion matrix is plotted uing heatmap as shown in figure 8

Experiment model:

As we can see from the above confusion matrix the model got confused between some of the categories like 7 and 8 in which videos are equally distributed that means the model was unable to differentiate between category 7 and category 8. Here category 7 and 8 corresponds to Basketball and BasketballDunk respectively. Also between 23 and 24. Here category 23 and 24 corresponds to Cricket ball and Cricket shot. This may be because of the most similar features between them and also we could see the probability has been almost equally distributed between those classes. Because of this we got very low validation accuracy which is of 0.28 but the top '2' categorical accuracy was 0.60 which is significantly good number. From this we can conclude that the model was unable to learn the differences between the similar categories. Sample frames of some categories are shown in figure 7.

- Validation accuracy: 28
- Top '2' categorical accuracy: 63

Final Model:

As we got to know the cause for low validation accuracy, in our next experiment we have selected the video

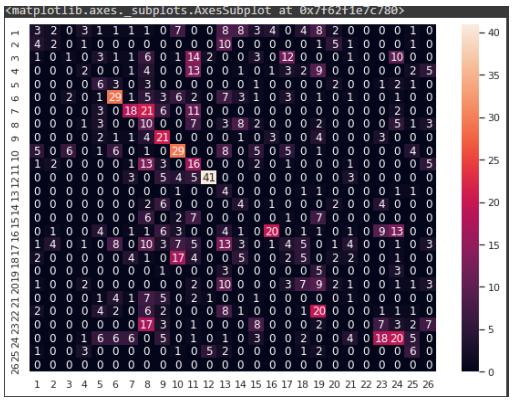

Figure 8. Experimental results

Super Category	Selected video categories.
Human-Object Interaction	'ApplyEyeMakeup','BrushingTeeth', 'HammerThrow', 'HulaHoop', 'Typing'.
Body - Motion	'PullUps', 'RopeClimbind', 'TaiChi', 'Swing', 'TrampolineJumping'.
Human – Human Interaction	'BandMarching','HeadMassage'.
Playing Musical Instruments	'PlayingCello', 'Drumming'.
Sports	'Archery','BasketballDunk', 'BoxingPunchingBag','BreastStroke', 'FieldHockeyPenalty','HorseRace','PoleVault', 'SumoWrestling','IceDancing', 'Kayaking'.

Figure 9. Video Categories

categories in such a way that they are not much similar to each other. As our dataset has 5 super categories 1)Human-Object Interaction 2) Body-Motion Only 3) Human-Human Interaction 4) Playing Musical Instruments 5) Sports. So taking leverage on these super categories we picked 4 to 5 sub categories from every super category. Shown in Figure 8

By doing this, we achieved a validation accuracy of 75 with "top 2 categorical accuracy" as 87 which is a decent result. Also, we can see there is no overfit in the data and the videos are pretty well classified as shown in the confusion matrix(Figure 10). Training, Testing and Experimaental results also looked great as shown in Fig11

- Validation accuracy : 75
- Top 2 categorical accuracy : 87

6. Resources

We have used google drive to store our dataset which is of 6.5 GB size and we have mounted it to Google colab to use the data. We have used Google colab for our entire project and its specifications are as follows:

- n1-highmem-2 instance
- 2vCPU @ 2.2GHz

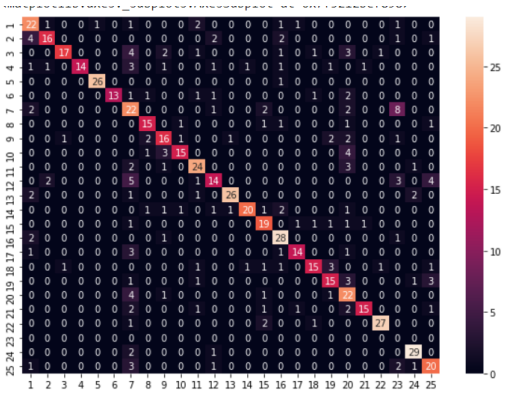

Figure 10. Final Model results

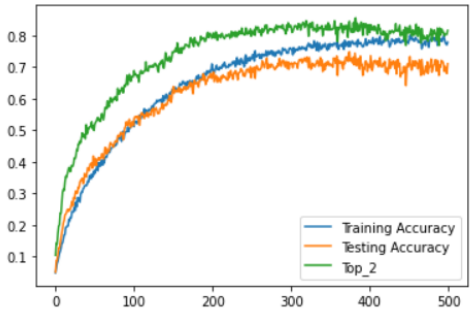

Figure 11. Epochs vs Accuracies

- 13GB RAM
- 100GB Free Space
- idle cut-off 90 minutes
- maximum 12 hours
- GPU: single 12GB NVIDIA Tesla K80 GPU

For extracting features from 25 categories, we ran the code on GPU for 12 hours, as we got some mounting time issues we extracted features in the batch wise in two different machines and stacked each other's extracted data.

7. Conclusion:

Action detection and understand the activity is of utmost importance as activity recognition finds its application in various fields such as personal healthcare like fitness tracking, fall detection of elderly people, monitoring functional and behavioral health using wearables. We can say that deep learning architectures have been used largely due to its advantage of hierarchically self-derived features, which help represent the data better compared to the handcrafted

features. Therefore, it is highly important to design and develop robust data mining techniques to extract the knowledge and deep learning techniques to infer and validate that knowledge from data which will allow the Action recognition system to make intelligent decisions.

8. Future Scope:

There is a lot of scope in Action recognition and this field is worth exploring. There are have been various attempts to successfully model sequences. Most recent promising results state of art achieved 94 percent. Here are a few ideas which could be explored in future work.

- Actions are predominantly determined by movement of joints of the person involved in the action. A dataset which contains both video and 3D skeletal data was released in CVPR'16. Using skeleton data along with the video would definitely yield better results in action recognition.
- Attention models have worked well in various tasks. Use of attention to focus on the correct frames is another possible area to explore on this dataset.
- Actions are also determined by the objects involved in the action. Explicit object detection can lead to better understanding the context of the video.
- At present the research in NLP field is high which can be used to generate text by recognizing the actions performed in the video which yields to greater applications.

9. Applications:

Action recognition and detection has found applications in critical domains such as unmanned driving, medical robotics, sports analysis, and safety monitoring. There is still significant room for improvement, for example by applying weakly- and self-supervised learning techniques to reduce annotation costs, adversarial learning to improve model robustness, or incremental leaning for online action detection. By automatically monitoring human activities, home-based rehabilitation can be provided for people suffering from traumatic brain injuries. One can find applications ranging from security-related applications and logistics support to location-based services. Activity recognition systems have been developed for wildlife observation and energy conservation in buildings.

10. Contributions

Both of us worked on every module in this project. So that we could learn and implement different techniques. Video preprocessing and feature extraction are the time complex parts of this project, so we shared these tasks. Below are individual contributions to this project.

Sai Bharat Kumar Konakalla:

- Preprocessing task
- Experimented with different custom CNN's to use it as a feature extractor
- Data and feature extraction loads
- Model Training

Pranay Reddy Dava:

- Transfer Learning research and finding different extraction models
- Data formatting and segmentation
- Data and feature extraction loads
- Model Training

References

[1] Goutham Dommaraju. Keras' accuracy metrics, 2014. Supplied as additional material `tr.pdf`. 4

[2] Rishikesh Sanjay Ghewari. Action recognition from videos using deep neural networks, 2014. Face and Gesture submission ID 324. Supplied as additional material `fg324.pdf`. 2

[3] Ziyi Liu 1 Qilin Zhang 2 Zhenxing Niu 3 Gang Hua 4 Jinliang Zang 1, Le Wang 1 and Nanning Zheng 1. Attention-based temporal weighted convolutional neural network for action recognition. *Journal of Foo*, 14(1):234–778, 2004. 2

[4] Amir Roshan Zamir Khurram Soomro and Mubarak Shah. Ucf101: A dataset of 101 human action classes. *Journal of Foo*, 13(1):234–778, 2003. 1

[5] Shivam Sharma. Deep learning architectures for action recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 12(1):234–778, 2002. 1