Models and Datasets in Action Recognition

Truman Thames
Fayetteville State University
1200 Murchison Rd, Fayetteville, NC 28301
tthames@broncos.uncfsu.edu

Dr. Boqing Gong
University of Central Florida
4000 Central Florida Blvd, Orlando, FL 32816

Abstract

From the beginning of time, Action Recognition with video has been plagued by the problem of not having enough labeled video data to saatisfactorily train large models. Recently, with the release of the kinetics dataset and its large amount of data, it has become possible to train action recognition networks so that they are showing similar progress as image recognition networks. With this increase in data it should be possible to compare different action recognition networks and how they perform on datasets with sizes of different orders of magnitude. Using the kinetics dataset and differently sized subsets, such an experiment and analysis can be carried out.

1. Introduction

Action Recognition is an area of computer vision that has undergone much development with deep netowrks in the past few years. As computing power increases and datasets are developed more complex neural networks are able to be used to perform action recognition.

1.1. Research Statement

The purpose of our research project is to study the effects of differently sized datasets on the performance of deep learning models. There are two primary ways in which different datasets can affect performance of a model. If we increase the size of a dataset and keep the same classes then the acuracy of the model should go up. If the classes change, then the effect is not very clear. This is why it is important to have the kinetics dataset. From this dataset we can simulate the other UCF101 and HMDB51 datasets. We can trim the Kinetics down to the size of the HMDB51 and have a subset of Kinetics instead of having to hope that the style of videos and the classes are similarly difficult testing for a deep learnin algorithm. Keeping the cinsistency in the dataset is important if we want meaningful results on the effect of dataset size on performance. For further research using this data on the performance of the simulated HMDB51 versus HMDB51 and simulated UCF101 versus UCF101 we can see how using youtube videos works as a dataset versus clips from movies or real world video clips.

I have not yet completed my first stage of testing the full Kinetics dataset with the Temporal-Segment-Networks two stream network. With the trouble that I have been having compiling and running the code from the TSN github page, I am deciding to use another model as my first round of testing. I have prepared the kinetics dataset and extracted the rgb frames as jpegs for each of the 300,000 videos in the dataset. As soon as I have constructed an acceptable model, I will be able to test only restricted by the time needed to train and test the models.

1.2. Kinetics Dataset

The Kinetics dataset was released on May 19, 2017 by Deepmind along with a paper detailing the performance of two stream, 3d convnet and LSTM networks with Kinetics, UCF101 and HMDB51 as training sets. The Kinetics datasets is composed of about 300,000 ten-second clips of youtube videos (there is also a small amount of shorter clips). Each clip comes from a distinct youtube video and is placed into one of 400 distinct (but sometimes similar e.g. ”playing basketball” and ”shooting basketball”) classes. The kinetics page on Deepmind’s site provides a list of the youtube links and time stamps the beginning and end of the clips. From there, we wrote a python script to leverage the youtube-dl library, threadpoolexecutor, and os.call to download and clip the videos in parallel using 8 threads.

This dataset is the main focus of our research. It being so large allows for it to be contracted to different sizes and to remove possibly ambiguous classes or to keep such classes and fine

1.3. UCF101 Dataset

The UCF101 dataset is composed of 13320 videos each one a member of one of 101 distinct classes.

1.4. HMDB51 Dataset

The HMDB51 dataset is composed of 7000 videos each a member of one of 51 distinct classes

1.5. Two-Stream Convolution Networks

A Two-Stream Convnet is a network with two parallel streams of convolution networks. One of the streams takes as input an rgb frame. The other stream takes a number (in the case of TSN, 10) of optical flow frames that follow the rgb frame. The optical flow frames capture the temporal information as optical flow is a representation of how pixels in an image move.

1.6. 3-D Convolution Networks

A 3D Convnet takes as input the sequence of rgb images all at once as a three dimensional tensor. It then uses three dimensional convolutions to process the entire tensor at once to copmute the class.

1.7. LSTM Convolution Networks

An LSTM Convnet takes the rgb frames sequentially and uses the memory of the LSTM module to keep information that it has processed about previous frames to use for its classification of later frames. The classification of the last frame is used ans the classification of the entire image stream.

2. Methodology

I acquired the kinetics dataset using the youtube-dl python library. From there I used the Temporal Segment Networks github code and made modifications as necessary for it to run on a CentOS cluster. These modifications have included attempting to compile the code on the system and realizing that the cpp portions of the code necessary to run the evaluation and logging would not compile due to the version of gcc on the compiler. As I do not have superuser access to the cluster and the prerequisites of the gcc compiler were too numerous and difficult to track down, I could not compile it natively on the cluster. The cluster administrator also informed me that he could not install a recent gcc version to the cluster.

I opted to instead install the code on a container running ubuntu16 that had already been installed to the system. From there I was able to proceed until I found that the dense-flow package was not properly extracting the optical flow from the videos using the gpus in the cluster. I was able to extract get it to work, but I realized that it would take far too long to extract the optical flow from any reasonable amount of videos in the time I had at UCF. It only extracted a few hundred each hour.

I decided to then just run a smaller test model on just the rgb frames of the kinetics dataset. As the extractor in dense-flow was not operating as it should, I decided to write a script to use ffmpeg to extract the rgb frames from the videos. After this was done, I started to train the TSN model on the frames from the Kinetics dataset. After a small number of iterations I tested the model wights and found that I was getting a NaN accuracy rating. After searching for the problem with the model, retraining with different data, and making sure that my data was sane and the classes correct, I have decided to do my first round of tests with another model.

2.1. Datasets

The Kinetics dataset is the dataset chosen for this project

2.2. Models

2.3.

3. Results

Results pending further testing.