Comp 3710 Research Report

Nash Domenichelli

T00562771

November 29, 2019

Introduction

The main purpose of my research project was to build a machine learning model to determine the beginning and end of phases of a competitive Super Smash Bros. Ultimate tournament set. This is a video game that is played in tournaments competitively on the Nintendo Switch. This is normally done by someone watching the set, in person and using software to change a display usually fed to a livestream to display the score. What the end goal with this project is to build software that will automatically update score and determine who wins with computer vision and machine learning technologies. The main part of the research project I accomplished was teaching a model to determine the beginning, and end of sets, with the end being divided into two parts. The “start” label is attached to the first frame of the 3 in “3,2,1 GO” on the screen and is attached to the frame where “GO” starts to fade away. The tag “end1” label describes is the end of the actual match where the label starts on the first frame a character is hit, which will lead to a death. The “end1” label ends at the frame “GAME” starts to fade away, showing that the match is over. The “end2” label is started after the game ends, and the frame a short cutscene starts and says who the winner is. In the future, the end2 label should be the time the winner is determined by the characters on screen. Using these labels attached the frames, with every other frame given a blank label, as it is neither the start or the end, it is possible to create a neural network to take the image and label data for a given set, and apply a new set and see how it does against the new set.

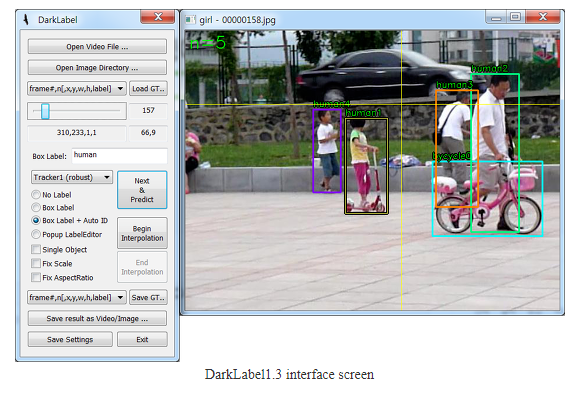
Background

I have looked up a little bit to see if anyone has done any similar research like this in the past, but as the field of eSports is fairly new, there is no one else doing this sort of research that I can find. There is a little bit of research in machine learning in the popular game *League of Legends* [1] and *Dota 2* [2], but those are to study characters and gameplay rather than just doings score display. These are also larger eSports with a bigger following and larger player base. Fighting games are a lot smaller in viewership as well as player base, but they have a very black and white way of telling who is winning, or who won the match. I picked fighting games to analyse because of this concrete way to tell what is going on at exact frames of a video of gameplay as well as my general interest and wanting to build a real tool our community could use.

I initially wanted to build this project and make a tool that will be able to update scores automatically, but due to time constraints, will only be implementing finding the beginning and end of videos. To get the videos in the first place, we livestream our weekly tournaments and then the person who was streaming, edits one large file of the day down to individual best of 3 or best of 5 tournament sets depending on the round of the tournament. This takes a lot of time for a simple task: watching the video of the footage for the day and cutting it where the tournament sets are. So, with a tool that could determine the beginning and end of sets, it would be possible to create a tool to do this automatically by saying start video at the first start frame, and crop until the last end2 frame, and export and look further and repeat.

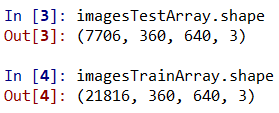
Project Details

The very first step in this project was creating the data set. To do this I went to our YouTube channel [3] and began downloading tournament sets that our community has recorded week after week. I downloaded the videos using YouTube to mp3 [4] using the video option and downloaded them using the video (.mp4) option. After downloading the videos, I had to manually label the frames using software I found online called Dark Label [5]. This software allowed me to load in and label my own video, as well as draw bounding boxes on a start frame and end frame and give all frames the same label in-between using the Begin Interpolation and End Interpolation buttons. I labeled the entire screen start during the start frames and the entire screen “end1” during the first end sequence and “end2” during the second end sequence. To save the labels, pressing the Save GT button created a text file in the format specified from the drop-down menu beside it.



I labeled about 10 videos producing over 100 thousand frames of data before I though I should start writing code for the training itself. The next part of this was splitting the videos into individual frames saved as jpgs instead of the mp4 video file. I found a python program online [6] to do this for me and started creating folders of frames and renaming the folder after the title of the video that the frames were from. This program was quick and dirty as it crashed when it was done making a jpg for every single frame in the video, but it worked.

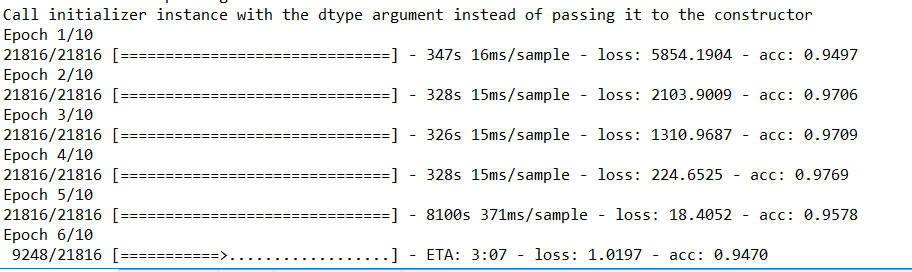
This begins the part where I had to write my own code. Using the TensorFlow and keras libraries I had to fit a model with a numpy array of images. So, I created code that goes through the folder of frames, and coverts the current image to a numpy array, and appends it to a list to return in that function. In the same function, I read from my file of labels and matched them to the images, skipping the images that have no label and giving a label of 4, meaning blank or no label. I also had to run through the ground truth files and ignore everything in the middle because the files have the bounding box coordinates, so I got rid of them for simplicity sake, as I labeled the full image unless the gameplay was in a specific spot, then I just labels the gameplay. After converting everything to the correct data types, I could run my model. I only tested with one video but that video had just under 30 thousand frames to work with and that model took an hour to create.



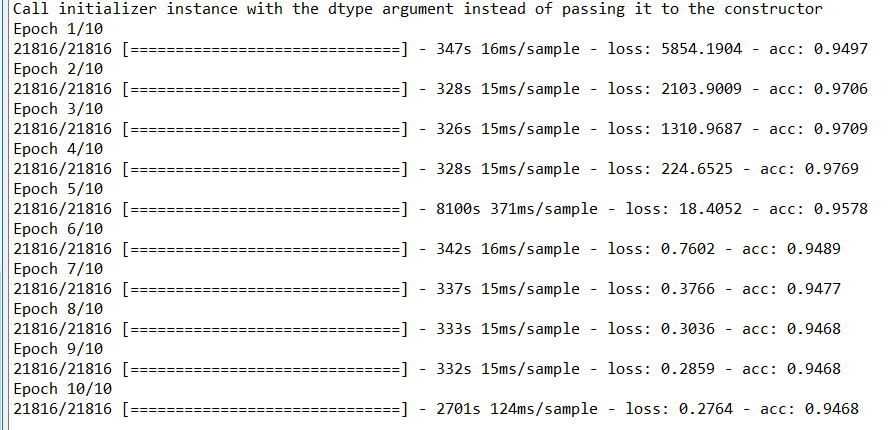
The model trained off 21816 images and tested against 7706 images. The other numbers in the 4-tuple are the dimensions, and the last number is the color channels (rgb) = 3.

Results

Progress during Epoch 6 (screenshot taken in case of memory failure or crash)



Progress after all 10 epochs



Final accuracy of the model



I was shocked after just one training session with all the default settings for creating the model that I used in the research assignment was this accurate.

Discussion and Conclusion

The results were very successful in proving that determining the start and end of videos are detectable and quite accurately as well. I was thinking of setting labels on certain aspects of the video that has a clear tell on what part of the games it is such as the “3,2,1,GO” at the beginning of a match and “GAME” at the end of a match, but decided to just label the entire area and see if the model would pick it up itself. To determine the winner of a match I was thinking to label the character portraits that are visible the entire match as “player1” and “player2” and then having those same labels during the end cutscene “end2” on the winner and loser, but there are over 70 characters in the game and it would be difficult to create a diverse enough dataset that would cover the entire cast of the game. Another sort of background idea with this project was to label and train different video games or even sports and see how accurately it could detect videos that have a grayer area too them.

Future Plan

I want to research this problem more and eventually build the tools to make an automatic video editor and even label who wins in games to see if it is possible to build a model with those images and labels. If that is possible, I would also like to build an automatic scoring tool instead of someone having to sit behind a screen all day and manually update scores. I would be open to further discussion on this project if anything is unclear. Also, I am interested in doing further research and getting academic credit for it if that is possible down the line.

References

1. <https://openai.com/five/>
2. <https://esportsone.com/>
3. <https://www.youtube.com/channel/UCFNJST4tJrqa29L9kn19x1g>
4. <https://ytmp3.cc/en/>
5. <https://darkpgmr.tistory.com/16>
6. <https://gist.github.com/keithweaver/70df4922fec74ea87405b83840b45d57>