Capstone Proposal

Machine Learning Engineer Nanodegree

Logan Spears
June 18, 2017

Domain Background

Video classification is gaining attention as image classification becomes more and more accurate. Datasets such as Youtube 8M and UCF101 are enabling the development of better models for classifying videos in the same way that ImageNet and others did for images. I was reviewing the UCF101 dataset while considering a topic to choose for this project and I noticed "tennis swing" was one of the recognized activities. Being a tennis player and fan myself, I began to imagine an AI that could function as a line judge, score keeper, or game statistician. In this project I have chosen to implement a critical step in understanding abstract concepts of a tennis match: shot type classification.

Problem Statement

Tennis is a dynamic game with multiple shot types: forehand, backhand, volley, serve, etc. Even an inexperienced tennis player or fan can distinguish between these shots. The goal of this project is to create an AI that replicates that functionality by completing the following tasks:

1. Aquire and preprocesses videos of tennis practice and matches
2. Label clips within the video by shot type
3. Train a model on the video input features and labels
4. Run predictions with the model until it performs with sufficient accuracy

Datasets and Inputs

Dataset Selection

Videos used in this project were search for on and downloaded from Youtube because it has a large catalog of videos that can be filtered by the presence of a Creative Commons license which permits private or even commercial use with proper attribution. Long videos (over twenty minutes) that contained scenes of people playing tennis in a professional or causal setting were selected. Each video was downloaded in 480p (480 x 640 resolution, acc 44100 Hz audio) in the MP4 container format.

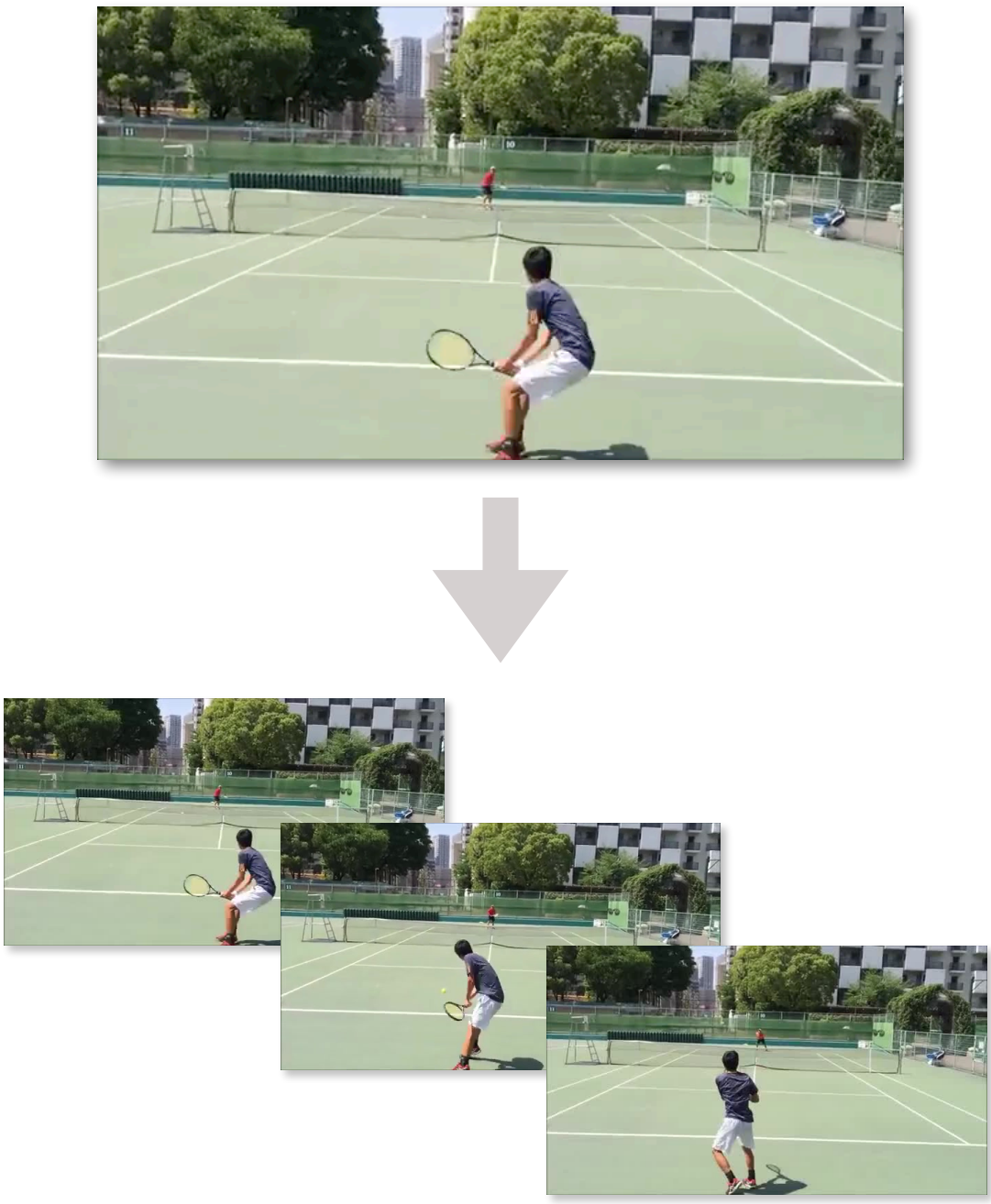
Dataset Diversity

A diverse set of inputs is needed for any classifier to generalize well to unseen data. Shot classification is no exception and needs a diverse collection of players, locations, and conditions to be truly generalizable. The dataset should include players that are: young, old, tall, short, righties, lefties, base-liners, serve-and-volleyers, etc. Various court types such as hard, red clay, grass, Har-Tru, etc. should be included. While additional factors such as weather, time of day, recording viewpoint, lighting, and others are important, it is impossible to predict and plan for all possible combinations. The best strategy is to acquire a large heterogeneous dataset and keep an eye out for missing conditions.

Preprocessing

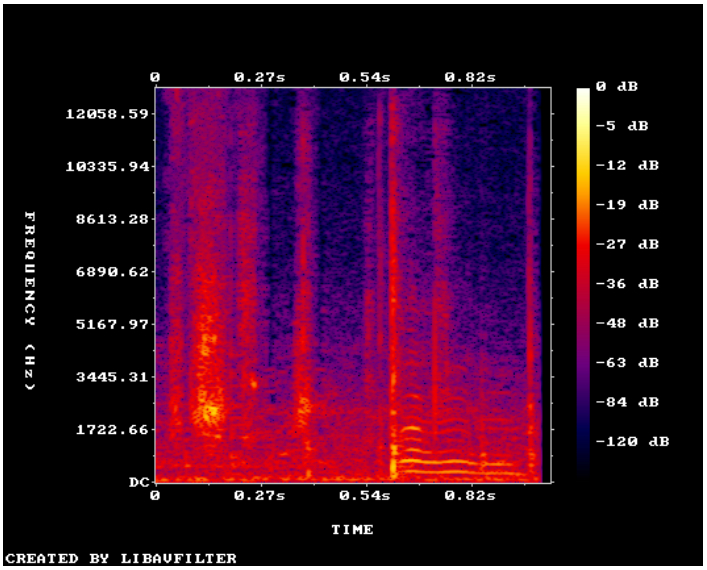
MP4 files are not used directly as input, but are preprocessed into image and audio derivatives. Using ffmpeg, a video conversion utility, JPEG frames are produced from a source MP4 file. Example usage of the ffmpeg is shown below with the source video being converted into individual frames:

```
$ ffmpeg -i video.mp4 -r 30.0 video-%4d.jpg
```



The JPEG files' pixels are used directly as input to train and run inference on the shot classifier. The other source of input is the videos' audio data. Audio files such as mp3 are compressed and not used directly but instead converted to a more machine learning friendly format (Geitgey). For this project I chose the spectrogram (visual representation of frequencies over time) format which is generated using ffmpeg's "showspectrumpic" command. Below is an example of the command and its output:

```
$ ffmpeg -i video.mp4 -lavfi showspectrumpic=s=300x300
```



Solution Statement

In this project shot classification is modeled as a multi-class classification problem using video data as input. Using an architecture consisting of Google's Inception v4 pre-trained Convolutional Neural Network (CNN), a Long Short Term Memory (LSTM) based Recurrent Neural Network (RNN), and a custom CNN this project creates a Deep Neural Network (DNN) capable of understanding and classifying what type of shot is in a video clip.

Benchmark Model

UCF's Center for Research in Computer Vision published results in 2012 for UCF101 Activity Recognition Data Set showing a prediction accuracy of 43.9% (Soomro, Zamir and Shah). The dataset includes 13,320 videos and 101 actions with about 130 videos per action. With only five actions (forehand, backhand, volley, serve, or none), a similar number of videos per action, and four years of advancement in machine learning technology it should be possible to significantly out perform these results and achieve greater than a 75% accuracy on the tennis shot dataset.

Evaluation Metrics

Multi-class classification problem typically use accuracy as an evaluation metric. Accuracy is defined by the following equation:

$$accuracy = \frac{tp + tn}{n}$$

True positive, *tp*, represents the number of correct predictions when the result was positive out of *n* inputs. True negative, *tn*, represents the number of correct predictions when the result was negative. The result being positive or negative is essentially a binary multi-class classification problem. A generalized version of accuracy is defined by the following equation:

$$accuracy = \frac{tp_1 + tp_2 + tp_3 \ldots tp_i}{n}$$

In this modified equation the sum of true positives from all classes is divided by the number of inputs. This is the evaluation metric that will be used in the project.

Project Design

Labeling

Videos are divided into one second clips so that shots can be labeled separately. While this approach is problematic for quick successive shots, it works the vast majority of the time in this project's dataset. The clips is labeled with one of the following shot types: forehand, backhand, volley, serve, or none. There are actually more shot types in tennis such as overheads, half-volleys, approach shots, tweeters (always a crowd favorite), and volleys are usually distinguished between forehand and backhand. These were purposefully excluded because they were not present in the test dataset in sufficient quantities.

Architecture

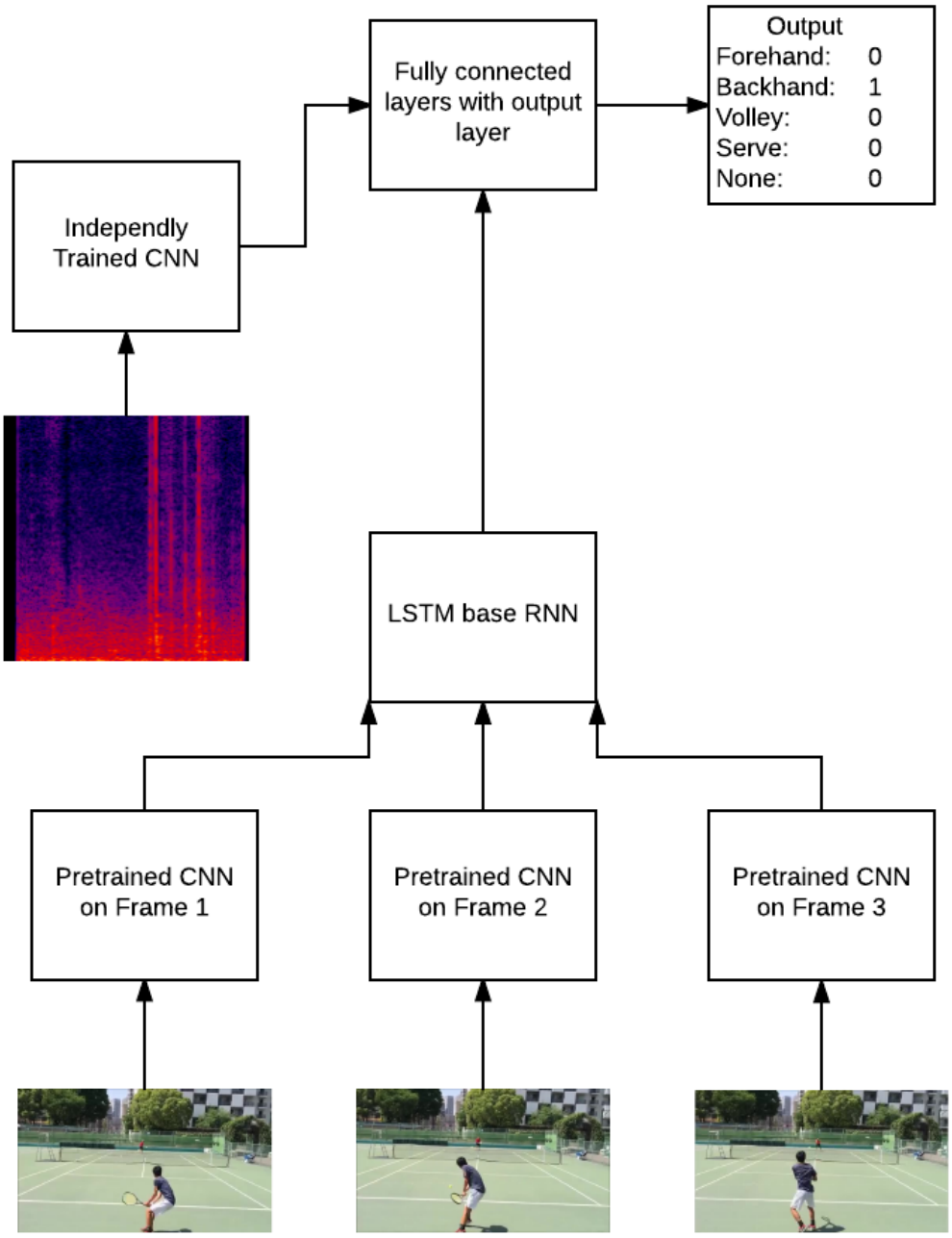
The industry leading architecture for image analysis is a Convolution Neural Network (CNN). ImageNet's winning submissions from the last few years have all used CNNs. Some of the winning models have be open sourced and expose both their architecture and pre-trained weights. Our model can reuse these pre-trained weights from the convolutional layers using a process called transfer learning. This enables our CNN to understand of shapes, human figures, textures, and other visual abstractions without the need for additional training.

CNNs can extract features from an image, but a "shot" is a pattern of body, ball, and racket positions over time. Natural language processing (NLP) faces a similar problem in which both a word and a sequence of words convey meaning. State of the art NLP uses a type of Recurrent Neural Network (RNN) called Long Short Term Memory (LSTM) to solve this challenge. RNN's accomplish this by having a memory of previous states and using that memory in combination with future input. Each layer in the RNN takes as input both the output of a frame's CNN and its internal state from the previous frame. The result is an RNN that can learn each frame's relationship to the entire sequence (Harvey).

During the preprocessing steps, the audio data from the video dataset was extracted and converted into spectrograms. Spectrograms show sampled and quantized frequency data in an image format that resembles a heat map. Since spectrograms are images, a CNN can again be employed to learn from the

audio data. While we could use transfer learning on the spectrograms, similarly to the video frame use case, the computer generated spectrograms are so different from the ImageNet dataset of natural and man-made images that it likely wouldn't be helpful. A few independently trained convolutional layers will be enough to learn from the relatively low entropy of a spectrogram.

Unifying the architecture and taking the audio and visual networks as input, a traditional neural network with a few hidden layers can feed an output layer with the five different shot classifications. The weights can be tuned by backpropagation optimizing a loss function over the entire network. Here is a diagram showing the entire network:



Works Cited

Geitgey, Adam. "Machine Learning Is Fun Part 6: How to Do Speech Recognition with Deep Learning." *Medium*. Medium, 23 Dec. 2016. Web. 18 June 2017.

Harvey, Matt. "Continuous Video Classification with TensorFlow, Inception and Recurrent Nets." *Hacker Noon*. Hacker Noon, 30 Dec. 2016. Web. 18 June 2017.

Khurram Soomro, Amir Roshan Zamir and Mubarak Shah. "UCF101: A Dataset of 101 Human Actions Classes From Videos in The Wild." *UCF Center for Research in Computer Vision* Nov. 2012. Web. 18 June 2017.