Machine Learning Engineer Nanodegree

Capstone Proposal

Pete Martin
September 1, 2018

Proposal

Domain Background

College and professional sports around the world generate large amount of video. In addition to professionally produced content (such as TV broadcast) there is also a huge amount of user-generated video content (fans capturing sport events on their phones for example).

It is therefore increasing challenging for humans to provide adequate metadata about sports video content manually and there is increased need for machines to help humans in tasks of classification, detection, feature extraction, clustering, etc.

Many large companies have recognized that machines can help humans in creating and processing metadata on other types of content. Google famously created the Google Photos project which classifies images and videos uploaded by users.

I'm currently very interested (for personal and professional reasons) in American football as the sport. I found there are very large number of football games captured on video but there is relatively little metadata attached to various parts of the captured game. I aspire to create a machine learning system that helps humans in processing this kind of video content.

Problem Statement

There is infinite amount of metadata that can be attached to a video of a football game. One of the challenges is knowing exactly when action on the field starts & stops. Humans would ideally like to be able to find individual segments of play / action and easily skip to the next and previous ones. The problem is that in an average football game there could be as many as 140 different plays. A human could easily find start positions (time markers) of each of the plays but it would require the human to watch the whole video. This could take a long time to complete and it would only cover that one video.

Hence the problem we are trying to solve here is: use ML to automate identification of football action sub-clips within the long-form video of the game.

One possible solution would be to extract training set of images (video frames) that represent two classes:

- **Positive**: images containing football team lineup (start of an action on the field). Example of such images is here
- **Negative**: images not containing team lineup (middle of the action on the field). Example of such image is here

This would therefor be binary classification problem / solution. The performance of the algorithm could be measured in number of ways. Based on non-scientific observation, the ratio of Positive instances to Negative is about 1:5. Since the data set is not very well balanced we will not rely purely on the accuracy metric. The Benchmark Model below explains our approach to how we are going to measure the performance of the solution.

Well trained and tested classifier could then as part of video processing (in production environment) extract every single frame from a video and classify it as either containing team lineup or not.

Optimizations could be done to perhaps process only every N-th frame or video frames spaced 1 second appart.

Datasets and Inputs

The final data set that we will use for this project will compose of 1,000 images which were extracted from videos for following 5 college football games:

- 2009 | Boston College Eagles (BC) @ Clemson Tigers
- 2009 | Boston College Eagles (BC) @ Virginia Tech Hokies (VT)
- 2009 | Boston College Eagles (BC) @ Maryland Terrapins (UMD)
- 2007 | Miami Hurricanes @ Boston College Eagles (BC)
- 2007 | Boston College Eagles (BC) @ Clemson Tigers

Each game will have 200 images extracted from it. Below are some examples of both positive and negative instances:

Positive:

- positive 1
- positive 2
- positive 3
- positive 4
- positive 5

Negative:

- negative 1
- negative 2
- negative 3
- negative 4
- negative 5

The data set will be manually labeled (classified as being either positive or negative). We expect the ratio of positive to negative instances to be about 1:5.

In this section, the dataset(s) and/or input(s) being considered for the project should be thoroughly described, such as how they relate to the problem and why they should be used. Information such as how the dataset or input is (was) obtained, and the characteristics of the dataset or input, should be included with relevant references and citations as necessary It should be clear how the dataset(s) or input(s) will be used in the project and whether their use is appropriate given the context of the problem.

Solution Statement

The solution to the above stated problem is a binary classifier that predicts which of the video frames (images) are in the Positive class and which ones in the Negative class. A well performing model will have good Recall since we want to focus on the Positive class of images.

The core model is really the main goal of this project however we will also create a wrapper that can takes as input a video file containing a football game and it returns timestamps at which positive frames begin to appear (indicating start of football play / action on the field).

Benchmark Model

I am not aware of anybody attempting to classify football video frame images into above mentioned classes. If we used a random classifier as our benchmark and we assume the model is evaluated based on the Recall metric (see justification below) then assuming ratio of 1:5 of positive to negative instances, this random benchmark model would produce as example:

- Assume test data set contains 100 positive instances (images containing football teams lining up just before the action starts) and 500 negative instances (images containing other parts of the game)
- Random benchmark performance assuming it predicted positive / negative outcomes at 50/50 rate over the 600 instances:

```
True Positives = 50
All Positives = 100

Recall = True Positives / All Positives

Recall = 50 / 100 = 0.50
```

Evaluation Metrics

Based on non-scientific observation the ratio of negative class instances to positive instances is about 5:1. Since the data set is not very well balanced we will use Recall as our main metric.

```
Recall = True Positives / all positives
```

So as example:

- Assume test data set contains 100 positive instances (images containing football teams lining up just before the action starts) and 500 negative instances (images containing other parts of the game)
- If the model predicted 110 of the images to be positive but only 90 of them were in fact positive, then the Recall is calculated as:

```
True Positives = 90
All Positives = 100

Recall = True Positives / All Positives

Recall = 90 / 100 = 0.90
```

Ideally we would like to accomplish Precision of `0.95` or more.

In addition we will look at the Confusion Matrix as well as the ROC and AUC visualizations to provide more color to the performance of the model.

In our tests we will use 20/80 split of testing vs training data and will also employ the Cross-Validation technique.

Project Design

Assuming the project proposal is approved, we will firstly expand the training and test data sets of images. As described in the **Datasets and Inputs** section above, we will extract these from a variety of football games. We want to get as much realistic variance as possible (e.g. different teams, different camera angles, etc). Once the image data is collected we will go through manual labeling process.

To train the model we will have to employ Convolutional Neural Network along with Transfer Learning techniques (the base model is of course TBD as we will try a number of different ones including VGG16/19, Resnet50, InceptionV3, etc). We will leverage Keras library for this purpose. Different pre-trained CNN models require different input image size so there will be fair deal of image pre-processing involved to allow us to leverage these models.

The CNN model will be optimized through Grid Search methodology on a number of its hyper-parameters including things like:

- convolution window size
- stride
- padding

In addition we will attempt few different network architectures (number and density of layers) as well as different activation functions.

To train the model we will use Data Augmentation naturally since the data set will already contain video frames where the camera is at different angles and proximity to the players on the field.

The project will run as a Jupyter Notebook on a local laptop environment and will hopefully be able to computationally process in reasonable amount of time. However, if the computations are taking too long, we will transfer the Notebook to an AWS account and a GPU-optimized server (we have instructions for how to set this up as part of the Udacity course).

Assuming we arrive at a model with satisfactory performance, we will additionally create a command-line processor that takes in as input a full-length video of a football game and returns the time-stamps within the video where it believes actions / play begin. This will likely be accomplished by predicting frames sequentially within the video (perhaps spaced out every 0.5 - 1.0 seconds).