Bi-Weekly Report Number 4

Computer vision for object detection in medicine

Team Number 6

Author	*Email*
Benedict Chan	benedict.chan.17@ucl.ac.uk
Shirin Harandi	shirin.harandi.17@ucl.ac.uk

COMP0016 System Engineering

November 29, 2018

Department of Computer Science

University College London

Week Overview

For our project we needed a way to detect objects from a video feed. To do this we needed to use and object detection API. When researching into this we ran into 2 popular options used for object detection and tracking: OpenCV object tracking (https://www.learnopencv.com/object-tracking-using-opencv-cpp-python/) and Google's TensorFlow object detection API. The main difference between the 2 are the way in which they track objects in a video stream.

OpenCV uses tracking algorithms alongside object detection to predict the path of objects. This means that it is faster than detection as you are tracking an object from the previous frame. Therefore, in the next frame you can predict the location of the object using information such as speed and trajectory from the previous frame. It also means that if objects get obscured, the tracker will still be able to detect the object where an object detection system may fail to identify the object.

Google's TensorFlow object detection API on the other hand uses a series of detections on images to locate each object and therefore 'track' them frame by frame. The API has a wide range of training models, with different levels of speed and accuracy. This therefore allows us to experiment with the different models to find one that can meet all our needs. As we are using object recognition on video stream, we need our software to be able to detect objects as quickly and accurately as possible as the system is used during operations, we also need it to be accurate to minimise the risk of error as much as possible.

In the end we decided to use the Google's TensorFlow API. This is because although it may be slower, most of the objects will be remaining stationary so the tracking algorithms won't help much. Furthermore, a lot of the objects will look quite similar, so we need a solution that can identify between the different instruments accurately.

When trying to set up a training model, we ran into a few problems. To use the API for our intended use, we needed to install TensorFlow-GPU which requires an NVIDIA graphics card. As we did not have access to these on a portable level, we have to host our application on a cloud service. However, as we will be training our own models, this will cost a lot of money, so we have decided to train our models on UCL's own ML servers (Blaze) but run our detection application on the cloud. For our hosting services we decided to use Microsoft Azure over Google Cloud Platform due to the speed of deployment and our experience using the hosting service.

List of tasks Completed

- Research TensorFlow
- Research into alternative object detection APIs
- Research into Microsoft Azure and Google Cloud Platform as hosting services
- Attempt to setup TensorFlow GPU

Plan for the next two weeks

- Setup TensorFlow on lab machines
- Setup blaze
- Take preliminary photos for generating models

Individual tasks completed

Benedict

Researched into the range of tools that are available for our project. By comparing OpenCV and TensorFlow we were able to see what is more suitable for out project and in turn can give the best results.

Shirin

Research into tools that will be used in the course of the project. By looking into different cloud platforms, we were able to decide which would better for the duration of our project and came to the conclusion that Azure would be a suitable choice.