3D Object Detection in the Wild

Matt Clifford
Supervised by Dr R. Santos-Rodriguez

November 28, 2018

1 Introduction

With recent advancements in mixed reality technology [*googleglassesandhololens*], there is an interest in the use of mixed reality to help create 'smart spaces', where mixed reality users are informed and guided through environments unknown to them.

Fracture Reality [$fracturereality.io$], has specific interest in 3D object detection for future projects. They specialise in creating bespoke mixed reality software for both private and government sectors. They mostly work with mixed reality visualisations of maps of an environment, for example to aid the control centres in airports [$https://www.youtube.com/watch?time_continue = 2v = t5L - jBLD04I$]. Although they have many projects that would benefit from 3D object detection, an ongoing project investigating how the affect of mixed reality is in tackling circulation issues [REF]. The use of 3D object detection would identify and locate objects such as stairs, doors, elevators and escalators. This aids the user in identifying if these objects are correct pathway for them, addressing bottle neck problems. A variant use of this would be to help the visually or navigationally impaired, with mixed reality help from detected objects of importance.

The objects of interest for the object detection are on a case to case basis, due to specific needs of each project. Fracture Reality are able to create some object specific use case data, but in the region of hundreds of examples, and due to the specificity of each task, finding existing object datasets might not be possible. This leaves retraining a detector for every new object is infeasible. The need for a system that can detect objects given as little training examples as possible is therefore needed. This can be achieved by utilising general knowledge learnt from similar or relevant tasks and quickly applying it to the specific task of interest[$https://arxiv.org/pdf/1310.1531.pdf$].

2 Literature review

In 2012, Alex Krizhevsky et al. revolutionised computer vision with a convolution neural network (CNN), inspired by [$http://yann.lecun.com/exdb/publis/pdf/lecun-01a.pdf$], which performed image classification on the ImageNet dataset [REF]. The CNN, named AlexNet, consists of 5 convolution layers followed by 3 fully-connected layers. It won ImageNet's ILSVRC-2010 and ILSVRC-2012 image classification contests [$alex_net$]. In [$https://www.nvidia.cn/content/tesla/pdf/machine-learning/imagenet-classification-with-deep-convolutional-nn.pdf$], they claim 'All of our experiments suggest that our results can be improved simply by waiting for faster GPUs and bigger datasets to become available', and since AlexNet's success, there have been consistent advancements in CNN state of the art. In 2014, K. Simonyan and A. Zisserman introduced VGG16[$https://arxiv.org/abs/1409.1556$], consisting of 16 convolution layers followed by 3 fully-connected layers. VGG16 achieves a top-5 error rate of 7.4% on ILSVRC-2014 compared to AlexNet's 17.0%. [$VGG16$] achieves this performance boost by using smaller convolution filters, 3x3 compared to AlexNet's 11x11 which decreases the number of weights to train at each convolution layer, alongside a deeper convolution architecture, which can extract deeper relevant semantic meaning from the images. Further performance improvements were made by the use of 'inception models' by C. Szegedy et al [$https://arxiv.org/pdf/1409.4842.pdf$][$https://arxiv.org/abs/1512.00567$], which stack the outputs of several convolutions of the same input, followed by filter concatenation. As well as 'residual learning'[$https://arxiv.org/abs/1512.03385$], which connects the outputs of multiple convolution layer. A combination of 'inception' and 'residual' models was formulated by C. Szegedy et al [$https://arxiv.org/abs/1602.07261$]. Further improvements to 2D image classification include [$https://arxiv.org/abs/1712.00559$] [$https://arxiv.org/abs/1707.07012$].

To identify the location of objects in images, [$https://arxiv.org/abs/1311.2524$] uses a regional CNN (RCNN) model. Regional proposals of the image, which are run though a modified AlexNet. The positive classification results from the regional proposals are then adjusted using a linear regression model to obtain better object bounding boxes. Computational speed ups are proposed in Fast-RNN [$https://arxiv.org/abs/1504.08083$], which pools the regional proposals. Further computational speed up are proposed in Faster-RNN [$https://arxiv.org/abs/1506.01497$], which combines the selective search regional proposals into the CNN. [$https://arxiv.org/abs/1506.02640$] proposes a computational speed up by using a search grid rather than regional proposals, an approach known as 'you only look once' (YOLO).

[https://arxiv.org/pdf/1803.06199.pdf] proposes a method of using the YOLO object detection approach in 3D space named Complex-YOLO, using only point cloud data from LIDAR depth sensors. Complex-YOLO uses a Euler-Region Proposal Network which estimates the orientation of objects by adding an imaginary and real part for each proposal box box. This results in 5 times speed up in object detection from the previous state of the art, with on par or better accuracy evaluated on only LIDAR data from KITTI benchmark suit [$https://ieeexplore.ieee.org/document/6248074$]. The KITTI benchmark suit is an autonomous driving dataset with 200,000 3D object annotations captured in cluttered scenarios, with up to 15

cars and 30 pedestrians in each image. The data is obtained from a stereo camera and LIDAR sensor mounted on top of a car that is driven in the real world. Although the KITTI benchmark suit is a rich 3D object dataset, it is not as directly applicable to mixed reality application due to the sensing quality differences between LIDAR and portable depth camera. As well as KITTI only focusing on 8 autonomous driving classes such as pedestrians, cars and bicycles. [$https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=arnumber=7785079tag=1$] is a large-scale 3D object dataset with 32040 object poses and 45 different object. The point cloud data is triangulated from 11 different views, making highly detailed scenes. The scenes are controlled and do not represent what would be captured in the wild. SUNRGBD benchmark suite [$http://rgbd.cs.princeton.edu/paper.pdf$] is a 3D object dataset consisting of 10,335 images with 64,595 3D object bounding boxes. The data is collected on various portable RGBD cameras such as the Kinect device, with indoor scenes focusing on objects such as doors, tables and chairs. A similar quality popular dataset is the Pascal Visual Object Classes (VOC)2012 [$http://host.robots.ox.ac.uk/pascal/VOC/voc2012/index.html$], which consits of 11,530 annotated object images, indoors and outdoors with 20 classes such as chairs, cars, dogs. However, PASCAL VOC 2012 only consists of 2D data. [$https://www-cs.stanford.edu/~roozbeh/papers/wacv14.pdf$] extends the PASCAL VOC 2012 dataset with proposed 3D CAD style projections of the 2D objects. Although rich, this 3D data of the object is dissimilar to that of a depth sensor, leaving SUNRGBD the most suitable starting dataset for this project.

[$http://openaccess.thecvf.com/content_ICCV_2017/papers/Dwibedi_Cut_Paste_and_ICCV_2017_paper.pdf$] proposes a 'cut and paste' style approach to synthesising 2D object detection datasets. First a object mask is predicted for the object, which is then applied to the image to 'cut and paste' the object into background scenes. Occlusions, truncations and blending are then applied to the object to help it more naturally fit into the scene. [$https://www.cv-foundation.org/openaccess/content_cvpr_2016/pa$] shows that training scene detectors on synthetic data produces comparable results on real life tests to detectors trained on the SUNRGBD state of the art dataset.

Current state of the art 3D object mostly detectors use LIDAR derived point clouds, trained on the KITTI benchmark suite [$http://openaccess.thecvf.com/content_cvpr_2018/papers/Xu_PointFusion_DeepSe$ $//openaccess.thecvf.com/content_cvpr_2018/papers/Xu_Multi-Level_Fusion_Based_CVPR_2018_paper.pdf$][$http:$ $//openaccess.thecvf.com/content_cvpr_2018/papers/Luo_Fast_and_Furious_CVPR_2018_paper.pdf$][$http:$ $//openaccess.thecvf.com/content_cvpr_2018/papers/Zhou_VoxelNet_End-to-End_Learning_CVPR_2018_paper.pd$] or controlled, detailed objects [$http://openaccess.thecvf.com/content_ICCV_2017/papers/Buch_Rotational_Subg$ $//openaccess.thecvf.com/content_ICCV_2017/papers/Balntas_Pose_Guided_RGBD_ICCV_2017_paper.pdf$][$http:$ $//openaccess.thecvf.com/content_cvpr_2018/papers/Tekin_Real-Time_Seamless_Single_CVPR_2018_paper.pdf$]. [$http://openaccess.thecvf.com/content_cvpr_2018/papers/Qi_Frustum_PointNets_for_CVPR_2018_paper.pdf$] and [$http://openaccess.thecvf.com/content_ICCV_2017/papers/Lahoud_2D-Driven_3D_Object_ICCV_2017_paper.$] use 2D object detectors to aid the regional proposal of the 3D object, by searching only the 3D space in the point cloud occupied from the projected frustum obtained from the 2D object detector. This vastly reduces the search space and determines the object class for the 3D detector, resulting in improved speed and detection compared to using point cloud data

alone, especially if the object suffers from occlusions or has a sparse representation. [$http$: $//openaccess.thecvf.com/content_cvpr_2018/papers/Ren_3D_Object_Detection_CVPR_2018_paper.pdf$] uses latent support surfaces for 3D object detection on the SUNRGBD dataset. Another notable 3D object detector that use the SUNRGBD dataset is [$http$: $//openaccess.thecvf.com/content_ICCV_2017_works$ $R_Consistent_3D_ICCV_2017_paper.pdf$].

Common representation – do a little research,, Auto encoders to possibly solve this

Unsupervised learning for creating data cheaply

3 Project plan

- collect dataset

- CNN as baseline for classification

- faster-RCNN as baseline for 2D object detection

- Transfer 2D model to 3D, elaborate on this

- Auto-encoder to find common representation

- Generating objects in scenes using cut and paste in 3D

4 Progress

Extracted relevant annotations and labels form SUNRGBD dataset Using 2D annotations, only take a subset of objects and crop full image to just the object area. Train CNN on these cropped images to have a baseline 2D classifier.

- show some examples of the dataset with annotations

- show examples of cropped inputs

- show network architecture

- discuss how this will help with RCNN

- how RCNN will be used for the frustum method