FotonNet: A Hardware-efficient Object Detection System using 3D-depth Segmentation and 2D-deep Neural Network Classifier

Gurjeet Singh[1], Sunmiao[2], Shi Shi[2] and Patrick Chiang[1,2,3]
[1]*Dept. of EECS, Oregon State University, Corvallis, U.S.A.*
[2]*State Key Laboratory of ASIC & System, Fudan University, Shanghai, China*
[3]*PhotonIC Technologies, Shanghai, China*

Keywords: Object Detection, 3D Data, Hardware, Depth Sensors.

Abstract: Object detection and classification is one of the most crucial computer vision problems. Ever since the introduction of deep learning, we have witnessed a dramatic increase in the accuracy of this object detection problem. However, most of these improvements have occurred using conventional 2D image processing. Recently, low-cost 3D-image sensors, such as the Microsoft Kinect (Time-of-Flight) or the Apple FaceID (Structured-Light), can provide 3D-depth or point cloud data that can be added to a convolutional neural network, acting as an extra set of dimensions. We are proposing a hardware-based approach for Object Detection by moving region of interest identification closer to sensor node in the hardware. Due to this approach, we do not need a large dataset with depth images to retrain the network. Our 2D + 3D system takes the 3D-data to determine the object region followed by any conventional 2D-DNN, such as AlexNet. In this method, our approach can readily dissociate the information collected from the Point Cloud and 2D-Image data and combine both operations later. Hence, our system can use any existing trained 2D network on a large image dataset and does not require a large 3D-depth dataset for new training. Experimental object detection results across 30 images show an accuracy of 0.67, whereas 0.54 and 0.51 for FasterRCNN and YOLO, respectively.

1 INTRODUCTION

The ability for robots and computers to see and understand the environment is becoming a burgeoning field, needed in autonomous vehicles, augmented reality, drones, facial recognition, and robotic helpers. Such systems require the detection and classification of objects to perform various tasks. In 2012, Krizhevsky et al. (Krizhevsky et al., 2012) (Deng et al.,) introduced the CNN (Convolutional Neural Network) based Deep Neural Network technique for image classification which outperformed existing methods. After the successful implementation of image classification, Ren et al. (Ren et al., 2015) implemented Region-Based CNN for Object Detection. The Fully Convolutional Network provided a solution to semantic segmentation, followed by Mask RCNN (Long et al., 2015), which further improved the capabilities of the Deep Neural Network.

All the networks mentioned above were assuming a standard 2D-image dataset, due to the wide prevalence of low-cost 2D-image sensors (i.e., cellphones with open-source sharing of their RGB data images). Recently, low-cost 3D-image sensors that can be embedded into cellphones (i.e., FaceID, Google Tango) have begun arising, enabling the potential for ubiquitous RGB+D sensor data, that can be used to improve object detection. Previous work from Gupta et al. (Gupta et al., 2014) trained a FasterRCNN on RGBD dataset NYUD (Silberman et al., 2012), using HHA encoding of 3D-depth images with 2D-images. This approach considers depth data as an additional NN input and not as independent values coming from different sensors, and thereby needs a large amount of depth data to train the neural network. Unfortunately, since 3D-depth sensors are not widely available as of yet, the ability to create a clean, accurate, and well-annotated RGB + Point Cloud dataset (~5000 images) as large as ImageNet (~1.2 million images) is still not possible. Unavailability of depth sensors also limits researchers from testing a large number of corner cases (i.e., difficult scenarios such as different SNR situations for 2Dimages or 3D-depth data.) Hence, the accuracy of a CNN trained with a small amount of RGB+D data will be inferior to a CNN trained with a large amount of RGB data.

Singh, G., Sunmiao, ., Shi, S. and Chiang, P.
FotonNet: A Hardware-efficient Object Detection System using 3D-depth Segmentation and 2D-deep Neural Network Classifier.
DOI: 10.5220/0008958604610468
In *Proceedings of the 9th International Conference on Pattern Recognition Applications and Methods (ICPRAM 2020)*, pages 461-468
ISBN: 978-989-758-397-1; ISSN: 2184-4313

We are purposing a hardware architecture with 3D and 2D sensors in which the 3D sensor has integrated Region-of-Interest detection, and the 2D image is used for classification. This architecture separates the use of depth data from image data. Due to such architecture, there is no need for a large depth dataset for training the neural network. Also, we can use a general-purpose classification network trained on large image datasets such as Imagenet which reduces the need to retrain the network again and again as well as on local node.

2 PREVIOUS WORK

In this section, we overview and discuss some prior object detection techniques and network architectures used for images and depth datasets.

2.1 Faster R-CNN

Ren et. al. (Ren et al., 2015) proposed a Region Proposal Network (RPN) to obtain bounding region boxes. These boxes are then sent to a classification layer to decide whether it contains an object or not. Since there are overlapping regions of bounding boxes, non-max suppression (NMS) is used to combine different adjacent bounding boxes. For their experiment, both RPN and classification networks share the VGG-16 model (13 CNN and 3 FCN layers) for feature extraction and the NMS limit to 0.7, resulting in bounding boxes of 2000 for each image. Gupta et al. (Gupta et al., 2014) used Faster RCNN to learn rich features from RGBD images of the NYUD dataset. They transformed depth information into 3-channels into HHA transformation. Figure 1 shows the basic operation of Faster R-CNN, showing the anchor boxes using the Region Proposal Network(RPN), and then the final output.

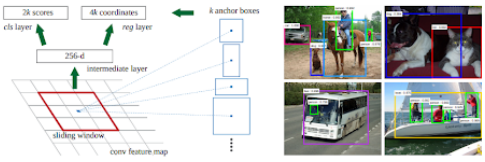

Figure 1: Region Proposal Network and the Output from Pascal VOC (Ren et al., 2015).

2.2 Mask R-CNN

He et. al.(He et al., 2017) demonstrates Mask R-CNN which is an extension of Faster R-CNN. Mask R-CNN extends Faster R-CNN by adding an object mask detector in parallel for instance segmentation.

Faster RCNN does not provide a pixel representation of the object in a scene. Instead, it just provides a bounding-box around an object. To address this problem, He et al. uses an ROI align layer that gives pixel to pixel representation from the bounding boxes. Furthermore, there is an ROI pool layer that combines local bounding boxes. Figure 2 demonstrates the ROI pool layer and instance segmentation layers used for instance segmentation that operates on top of Object Detection.

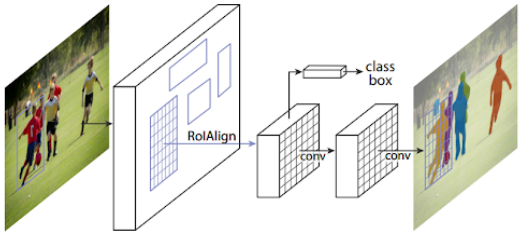

Figure 2: Mask R-CNN architecture for instance segmentation (He et al., 2017).

2.3 You Only Look Once (YOLO)

Redmon et al. (Redmon et al., 2016) propose a much simpler neural network for Object Detection. This network does not need Region Proposal Network (RPN) layer for training. Instead, the picture is divided into several predesigned anchor boxes. Next, Redmon et al. run classifiers such that overlapping bounding boxes are removed using Non-max suppression (NMS). The advantage of this network lies in its end-to-end training. In this way, Redmon et al. can train networks faster and easier by sacrificing some accuracy. Figure 3 shows a detailed YOLO architecture with 24 convolutional layers and three fully connected layers.

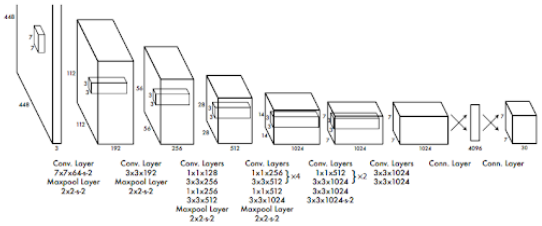

Figure 3: YOLO architecture (Redmon et al., 2016).

2.4 VoxelNet

VoxelNet (Zhou and Tuzel, 2017) is a unique way of running object detection just on 3D point-cloud data (with no other sensor data input). Point Clouds are

sparse, such that points clouds are converted to voxels and therefore just samples with certain chosen thresholds. Next, random samples are chosen and converted to point-wise inputs for feature learning. Figure 4 shows various layers and final output for VoxelNet architecture.

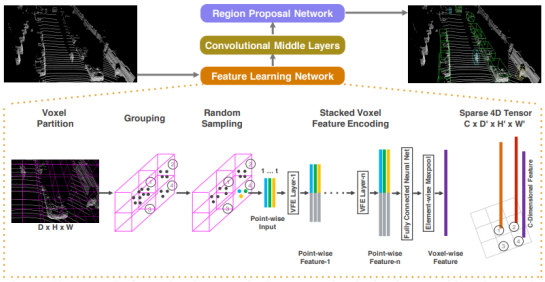

Figure 4: VoxelNet architecture (Zhou and Tuzel, 2017).

2.5 3D Object Detection Networks

Ever since the introduction of depth data from sensors such as Microsoft Kinect, researchers are trying to incorporate depth data into computer vision problems. Lin et al. proposed a method for 3D object detection(Lin et al., 2013) using 2D segmentation and 3D geometry. Song et al. (Song et al., 2015) propose deep sliding models for Amodal 3D object detection. Schwarz et. al.(Schwarz et al., 2018) used multiview RGBD cameras to perform heuristics on depth data for their object picking robot. Gupta et al. (Gupta et al., 2014) provided HHA encoding to train the network on Faster RCNN. Gupta and Hoffman provided several transfer learning techniques to transfer weights from 2D network to depth data (Hoffman et al., 2016) (Gupta et al., 2016).

2.6 Shortcomings of Prior Art

Most of the networks previously described methods use only conventional 2D image datasets. NYUD possesses RGB+D data, but unfortunately, the dataset exhibits a very high density of cluttered objects, with many of these objects exhibiting no depth features (curtains, windows, etc.) Unfortunately, with such a cluttered RGB+D dataset, any training using RGB data and depth data, depth data will not contribute much to learning. For example, we previously tried to use the depth data of NYUD for training, but because there are not many features to learn from depth. It is challenging to stop overfitting because of the small number of images.

In regards to VoxelNet, the network is trained with a number of different objects (Car, Bike, etc.), such that training for another network with a different set of objects is impractical. For both NYUD and VoxelNet, a large amount of 3D depth data for regular training of neural-network is required. Unfortunately, there currently exists no 3D-depth dataset large enough to rival conventional 2D-image datasets (ImageNet, etc.)

Ever since the publication of RGBD datasets like NYUD and SUN RGBD (Song et al., 2015), researchers are trying to incorporate depth dataset into learning and detection processes. Methods range from using heuristics to use encoding techniques such as HHA. Researchers also tried to use transfer learning techniques to train depth weights using ImageNet datasets. All these techniques are computationally intensive and hard to move closer to hardware such as a SOC. These techniques also make these systems more complex.

3 PROPOSED WORK

In this paper, we propose a DNN Object Detection system that dissociates the depth-data from the RGB data. In this way, our proposed system does not need to require a large training dataset for the depth data but still simultaneously extracts meaningful information from the depth sensor. Furthermore, our system combines this depth data with conventional 2D-trained image data to generate a practical, low-complexity object detection and classification system.

This paper is constructed as follows. First, we will give a system overview of our system, followed by test results, a comparison with current state-of-the-art systems, and then concluding with our future research direction.

3.1 System Overview

Figure 5 shows an overview of the system architecture. We designed our system with the understanding that there currently exists no large dataset of depth data for training. First, our system generates bounding boxes from the depth sensor output by performing clustering on the 3D point cloud. After denoising and cleaning of the clustered depth objects, we use the clustered objects to split the 2D image into sub-images which are fed to a 2D-Image Deep Neural Classification Network.For Threshold Filtering, a threshold for the classification top-1 accuracy is set to judge the availability of the classification result. If the top-1 accuracy is greater than the threshold, then the output is accepted. Detailed operation for each subsystem is explained below.

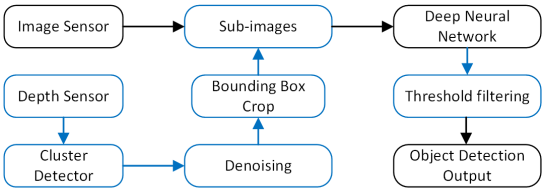

Figure 5: FotonNet System Architecture.

3.2 Bounding-box Detection

The bounding box detector generates bounding boxes proposals based upon the different clusters of point cloud data. In this system, the K-means Clustering Algorithm (MacQueen et al., 1967) is picked to produce a limited number of boxes. Compared with another classical algorithm Hierarchical clustering (Ward Jr, 1963), K-means clustering have a low complexity when the number of clusters is fixed. When the cluster number is 8, for 30000 points of point cloud the speed of these two algorithms is 28.08s and 1.54s (based on python 2.7, CPU only) separately. Considering the current depth cameras, they can offer around 3 ~40000 points. So K-means is more efficient in this situation. The algorithm flow is shown in Algorithm 1. We set the total number of objects we can detect as C depending on the scenario. In this dataset for the indoor scene, we set C to 8 which can obtain a satisfied result. At the same time, this method also limits the upper amount we can detect. Due to the complexity of K-means depending on the iteration times, the number of point cloud is decreased linearly to speed up. In this paper, the number of points in point cloud is around 30000. For K-means clustering, it costs much time to calculate distances between the current centroids and every point. As a useful solution observed, decreasing the number of point cloud linearly before we input the point cloud into K-means clustering can solve this problem efficiently. The more amount is discarded, then the speed is higher. In the same time, the accuracy will be effected for losing some critical information. Balancing the accuracy and speed, the ratio is set to 0.7, and the distance metric is Euclidean distance.

3.3 Denoising

After the clustering step, several simple denoising steps have implemented that act as a pre-processing and image cleanup that improves the subsequent object detection. First, the two clusters from the two top corners are eliminated since they are likely peripheral background images and not the main images for classification. Second, small-sized clusters which are a small percentage ratio of the entire point-cloud

Algorithm 1: Algorithm for K-means cluster detector:

Parameter setting: C(C > real number of objects) Set each point in the pointcloud P as a cluster
Initialize the center centroids $(\mu_1^0,,\mu_k^0),(\mu_j^0(j=1,....,k))$ randomly:
while $\sum_{j=0}^{k}||\mu_j^{i+1}-\mu_j^i||^2 > \delta$ **do**
  for *every i* **do**
    i) Compute the distance between each cluster to the current k centroids.;
    ii) Relabel the nearest centroid to the cluster.;
    iii) Compute the new average point in a cluster as the new centroid.;
    iv) Compute the difference between the new generating centroids and last centroids $\sum_{j=0}^{k}||\mu_j^{i+1}-\mu_j^i||^2$. ;
  end
end

are judged as noise (not real objects) and are also removed. Several other sub-images that are extraneous, such as point-clouds from the wall and the ground, have not been eliminated in this preliminary implementation.

3.4 Sub-images

After the detection of several different bounding boxes, we split the image into several sub-images so that we can perform object-detection on several objects detected by the depth sensor. The 3D-depth sensor is calibrated with the 2D-camera such that the bounding boxes can be projected on to the 2D-image plane, using the method from Park et al. (Park et al., 2014). In particular, we utilize Parks polygonal method and calculate the Projection Matrix for Singular Value Decomposition (SVD).

3.5 Deep Neural Network

Our proposed Object Detection system has the advantage in that it can use any existing classification network that has been trained on large datasets of 2D images (as opposed to VoxelNet or NYUD). The advantage of this system is that we can use an already pre-trained network and therefore do not need to train a new network for depth data. For this initial experiment, we utilize AlexNet due to its widespread availability.

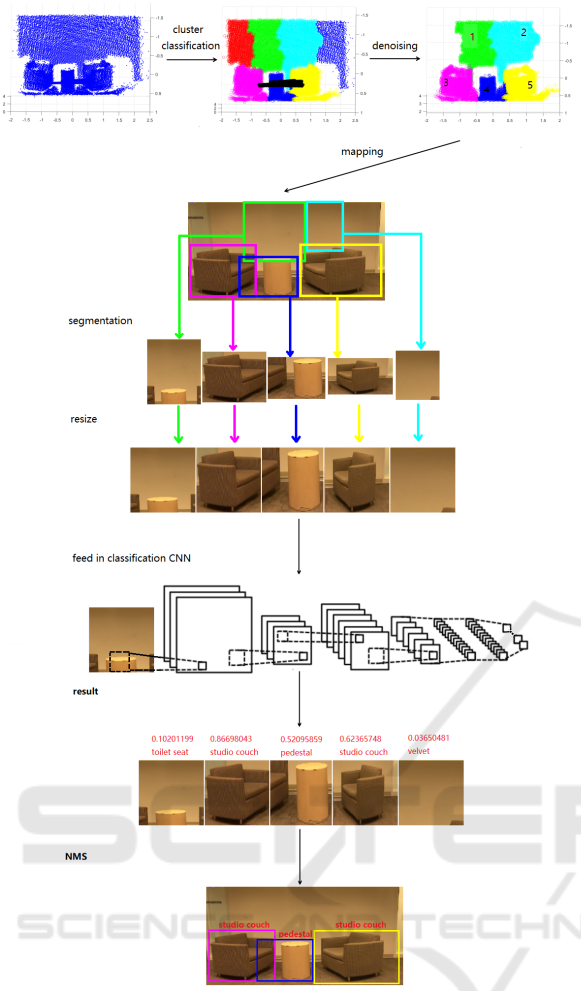

Figure 6: Architecture for FotonNet.

One advantage for our system is that it implements AlexNet inference using available systems like NVDLA open-source deep learning inference engine (NVD, b), thereby enabling a future low-cost ASIC implementation. For the performance estimation of the system, we used a ZYNQ Ultrascale+ SoC to run the corresponding parts of the entire system, including the image clustering, preprocessing, and AlexNet inference. The preprocessing part is written in C++, compiled with the gcc-linario arm-linux cross compiler. Sub-images are generated accordingly and fed into the NVDLA User Mode Driver. The NVDLA small configuration hardware is programmed into the programmable logic (PL) part of the Zynq SoC. An NVDLA full configuration can run AlexNet at 1100 Frames per second with 2048 Multiply-accumulate units (MACs) when its batch size is 16 (NVD, a). The NVDLA small configuration we are using exhibits 64 MACs, which uses 32 times less hardware than the full configuration, as it is not the bottleneck of speed for our implementation. The implementation workflow of the entire system is shown in Figure 7.

3.6 Object Detection Output

After Classification, we obtain the output with corresponding bounding boxes. We apply a threshold λ on the final sub-images in order to determine if there are any extra bounding boxes. Depending on the ability of CNN, if the object in the proposed box is not valid, then the top-1 score is further lower than a true one. In our experiments, the classification probability below a certain threshold (i.e., 0.2) will be discarded..

4 EXPERIMENT

Since the depth data is used here to perform clustering, it is difficult to compare our system with previous benchmarks. In order to perform a side by side comparison of our system versus other algorithms, we captured both the 2D-Image and 3D-Depth data of 30 images with various indoor objects inside. We tested our proposed FotonNet using these 30 images and point cloud data, while for the conventional networks (Girshick et al., 2018) and YOLO darknet (Redmon and Farhadi, 2018), we only used the RGB images from the scenes and ran through detection. The output of FotonNet is shown in Figure 8. We also tried to train Faster RCNN, YOLO and AlexNet on NYUDv2 dataset with 40 classes with both image and depth data. All the networks were trained for about 150 epochs and trained on 795 training images and tested on 654 images. Even after converging error the networks were failed to achieve significant object detection ability. We assume it is because of the small number of training images and high and dense objects in the scenes.

We tested this system with our 30 test images and calculated the mean Average IOU. The Average IOU for our system is 0.72413, with a measured. The results of our architecture and the parameters sets are as follows:

- Clustering Time (ARM A53): 0.5 sec
- Classification Time: 0.0545 sec

In order to compare our system versus others, we ran our test images through each of the networks, thereby providing the IOU, latency, and size of each network. We also provide latency and accuracy as reported by existing neural networks. For the AlexNet network using Xilinx 's ChaiDNN library on ZCU104 FPGA. We used onboard ARM A53 for clustering tasks. The speed of our system then will depend on

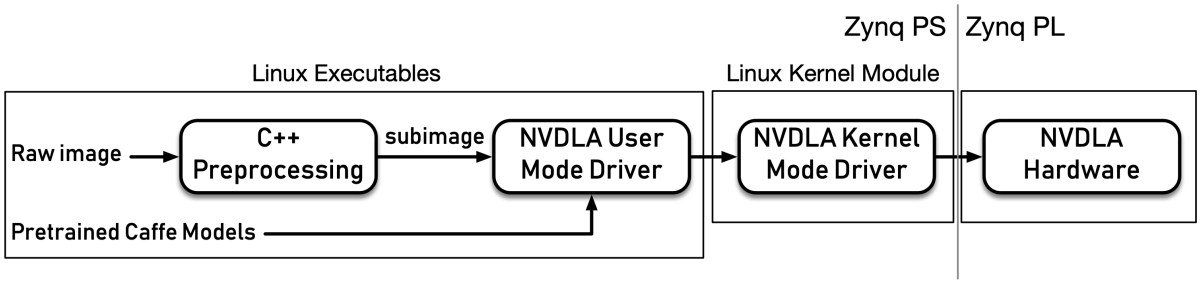

Figure 7: Implementation Architecture.

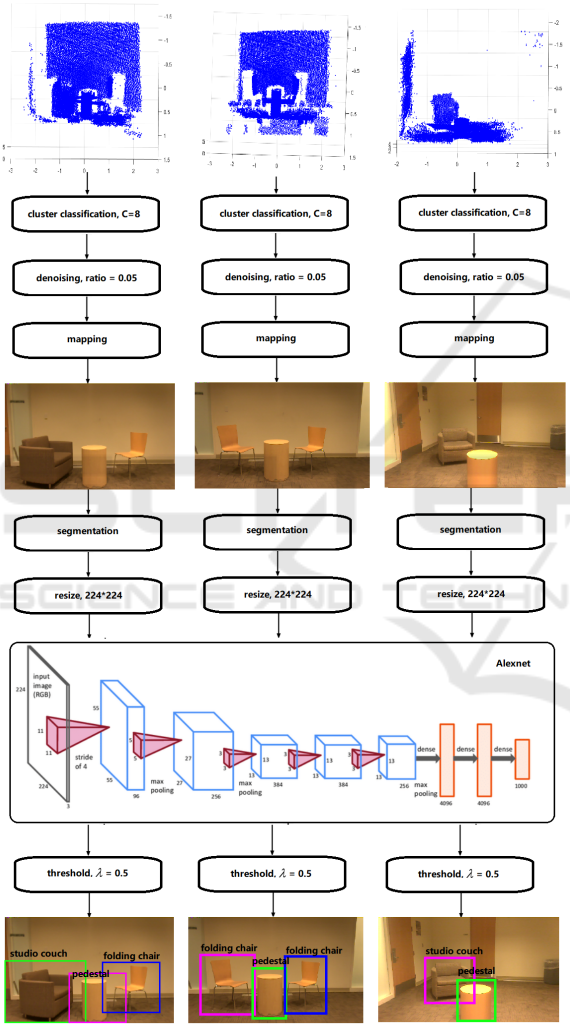

Figure 8: Final output of the system.

the number of clusters we generate. Assuming on average we generate 8 clusters, the speed of our system is approximately 55fps. Please note, we are assuming that the clustering algorithm is available and integrated into the on-board 3D-depth sensor. However, the speed can be easily increased if we use the NVDLA classification engine, which reports a speed of 1100 fps. One significant advantage of FotonNet is its ease-of-deployment. By dissociating depth from the RGB data, it will be easier to expand object detection to even more classes and neural networks than just limited Object Detection classes. Thus, this approach makes it easier to implement and expand than currently existing systems.

5 CONCLUSION

We demonstrate a new HW-system architecture for object detection which leverages low-cost 3D depth sensors. The size of this network is significantly reduced compared with the status quo because of the reduction in the number of ROIs and the elimination of any extra training steps needed for object classification. We used this approach to keep in mind that we are going to move Boundary box detection on-chip in Depth Sensor. We found that our approach outperforms YOLO and Faster-RCNN in object detection. In this paper, we experimented with the hierarchical clustering algorithm. We will try to come up with a faster clustering algorithm and try to migrate that functionality on the chip.

REFERENCES

Nvdla performance. https://github.com/nvdla/hw/blob/nvdlav1/perf/NVDLA_OpenSource_Performance.xlsx. Accessed: 2018-11-14.

Nvdla primer. http://nvdla.org/primer.html. Accessed: 2018-11-14.

Deng, J., Dong, W., Socher, R., Li, L., Li, K., and Fei-Fei, L. I. A large-scale hierarchical image database. computer vision and pattern recognition, 2009. cvpr 2009. In *IEEE Conference on*, pages 248–255.

Girshick, R., Radosavovic, I., Gkioxari, G., Dollár, P., and He, K. (2018). Detectron. https://github.com/facebookresearch/detectron.

Gupta, S., Girshick, R., Arbeláez, P., and Malik, J. (2014). Learning rich features from rgb-d images for object detection and segmentation. In *European Conference on Computer Vision*, pages 345–360. Springer.

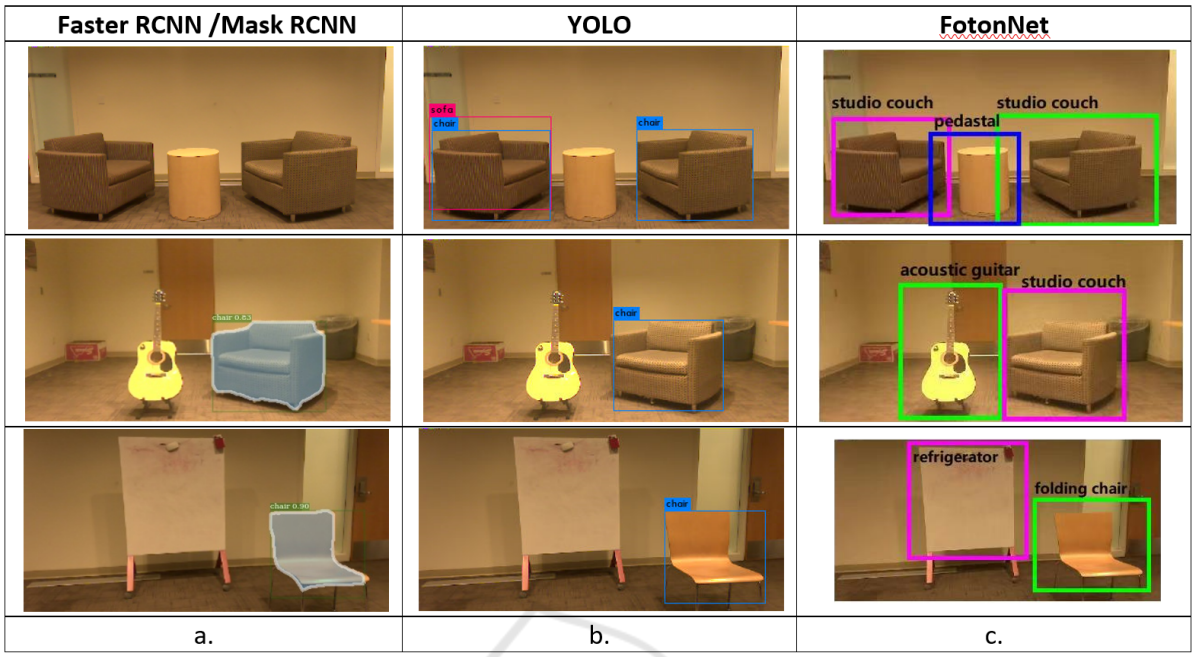

Figure 9: a. Sample output from Faster RCNN /Mask RCNN b. Sample output from YOLOv3 c. Sample output from FotonNet.

Table 1: Comparison of System with Existing State of the art methods.

Network & Usability & Rough Estimate of HW-Complexity & Accuracy & Latency (FPS) & No. of training Parameters
Faster RCNN Mask RCNN & No training data for depth & Large Network & 0.759 (COCO test (mAP)) 0.53968 (30 images(IOU)) & 7 & VGG16 + RPN + Regression Layer
YOLO & No training data for depth & Large Network & 0.634 (COCO test(mAP) 0.506849 (30 images(IOU))& 21 & 24 Conv Layer + 2 FC Layer
FotonNet & No need for training when using large classification network & Low Complexity, Cluster and Classification can be moved on chip & —- (COCO Test) 0.67 (30 images(IOU)) & 55(Assuming Clustering is performed by sensor locally) & Using Pre- Trained AlexNet

Gupta, S., Hoffman, J., and Malik, J. (2016). Cross modal distillation for supervision transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2827–2836.

He, K., Gkioxari, G., Dollár, P., and Girshick, R. (2017). Mask r-cnn. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pages 2980–2988. IEEE.

Hoffman, J., Gupta, S., Leong, J., Guadarrama, S., and Darrell, T. (2016). Cross-modal adaptation for rgb-d detection. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5032–5039. IEEE.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.

Lin, D., Fidler, S., and Urtasun, R. (2013). Holistic scene understanding for 3d object detection with rgbd cameras. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1417–1424.

Long, J., Shelhamer, E., and Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440.

MacQueen, J. et al. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA.

Park, Y., Yun, S., Won, C. S., Cho, K., Um, K., and Sim, S. (2014). Calibration between color camera and 3d lidar instruments with a polygonal planar board. *Sensors*, 14(3):5333–5353.

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on*

computer vision and pattern recognition, pages 779–788.

Redmon, J. and Farhadi, A. (2018). Yolov3: An incremental improvement. *arXiv*.

Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99.

Schwarz, M., Milan, A., Periyasamy, A. S., and Behnke, S. (2018). Rgb-d object detection and semantic segmentation for autonomous manipulation in clutter. *The International Journal of Robotics Research*, 37(4-5):437–451.

Silberman, N., Hoiem, D., Kohli, P., and Fergus, R. (2012). Indoor segmentation and support inference from rgbd images. In *European Conference on Computer Vision*, pages 746–760. Springer.

Song, S., Lichtenberg, S. P., and Xiao, J. (2015). Sun rgb-d: A rgb-d scene understanding benchmark suite. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 567–576.

Ward Jr, J. H. (1963). Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, 58(301):236–244.

Zhou, Y. and Tuzel, O. (2017). Voxelnet: End-to-end learning for point cloud based 3d object detection. *arXiv preprint arXiv:1711.06396*.