IMPLEMENTATION OF MACHINE LEARNING TECHNIQUES FOR DEFECT DETECTION IN REAL-TIME

Prudhvi Challapalli[1], Challa Sai Bhanu Teja[2], Pinnaka Naveen Chandra[3], Vanaparthi Revanth[4], Dr Boddu Sekhar Babu[5]

[1,2,3,4]Students, [5]Professor, Department of Computer Science and Engineering, Koneru Lakshmaiah Education Foundation, Guntur (A.P), India, 522502.

ABSTRACT: *Every fruit or vegetable comes from the plants are not perfect and edible. So, the vegetable or fruit warehouse owners segregate them by recruiting labors at high cost. If we can automate this process of segregation the owners of those store can save a lot of money. For detecting which fruit or vegetable is defective, we can use machine learning techniques like Mask R-CNN, YOLO and Faster R-CNN. These machine learning models can also be used to make the segregation process simple and quick.*

If we can use these machine learning models in a mobile app (which will detect rotten vegetables or fruits) then we can reduce the labor and we can assign them to other crucial jobs.

If we can build an AI powered warehouse robot (using these machine learning models) we can totally cutdown the labor expenses.

Key Words: Mask R-CNN, AI powered warehouse robot, Mobile App

1. INTRODUCTION

1.1 Problems faced during segregation process

In many countries large number of workers work together to segregate the defective or rotten vegetables (or) fruits from a big stack. Now a days, due to the shortage of labor, the labor cost is increasing day by day. By using these machine learning techniques, the owners of the vegetable (or) fruit warehouses can save lots of money.

If the warehouse owner recruit labors. Due to recklessness or high work load, sometimes labor cannot find all the rotten vegetables or fruits. So, the vegetable or fruit warehouse owners will not be able to satisfy their customers. It is very hard for a human to work at a stretch from morning to evening finding the defective ones. These machine learning techniques will make the job of labors easy and simple.

1.2 History of CNNs

1.2.1 R-CNN (2014)

R-CNN was born on 2014. It is an early application of CNN to Object Detection. Object Detection which is the task of finding the different objects in an image and classifying them. CNN can lead to dramatically higher object detection performance on the best co-co-co set.

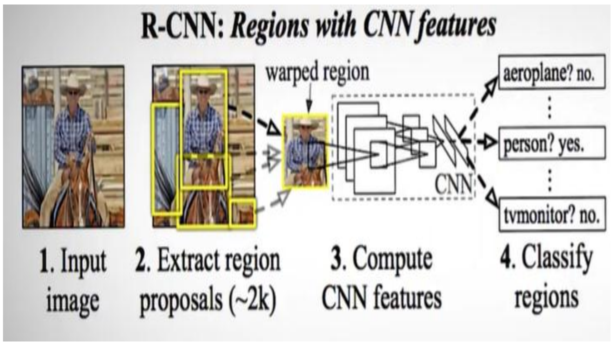


Fig.1: R-CNN

R-CNN creates these bounding boxes or region proposals using a process called selective search at a high-level selected search looks at the image through windows of different sizes and for each image tries to cluster together adjacent pixels by texture colour or intensity to identify objects once the proposals are created R-CNN makes the region to a standard square size and passes it through to a modified version of ALEXNET which at the time was the winning submission to image net 2012 that inspired R-CNN on the final layer step four of the CNN R-CNN as a support vector machine or SVM that simply classifies whether this is an object and if so what object.

1.2.2 FAST R-CNN (2015)

Fast R-CNN is speeding up and simplifying R-CNN. R-CNN works well but it's quite slow for a few simple reasons it requires a four pass of ALEXNET for every single region proposal and for every single image which could reach up to Two thousand forward passes per image which is slow. It must train three different models separately the CNN to generate the image features. The classifier that predicts the class and regression model to tighten the bounding boxes this makes the process extremely hard to train.

In 2015, Ross Girshick, the first author of R-CNN solved both these problems, leading to the second algorithm called FAST R-CNN. Main features of FAST R-CNN are ROIPOOLING and combining all models into single network.

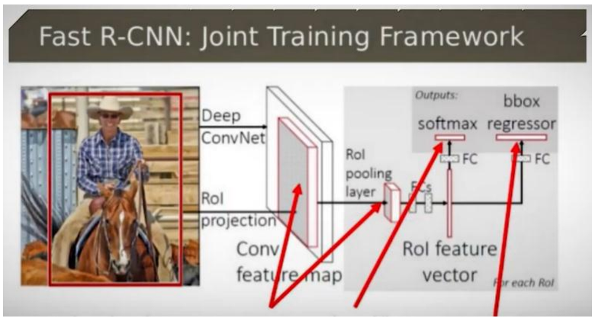


Fig.2: Fast R-CNN

1.2.3 FASTER R-CNN (2016)

Faster R-CNN speeding up the region proposal even with these advancements there was still one remaining bottleneck in the Fast R-CNN process the region proposal. Faster R-CNN has a fully convolutional network on top of the features of the CNN grating was know as region proposal network.

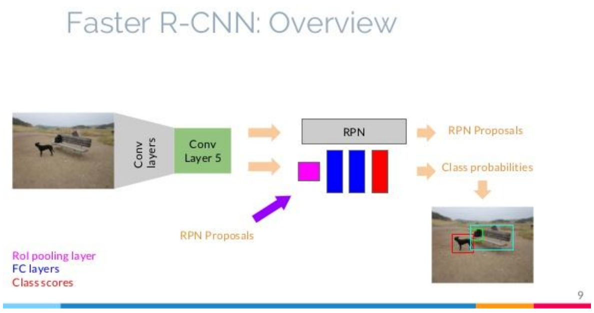


Fig.3: Faster R-CNN

1.2.4 MASK R-CNN (2017)

Mask R-CNN is the extending of Faster R-CNN 4 pixel level segmentation. CNN is to effectively locate different objects in an image with bounding boxes the extention of these techniques to go one step further and locate pixels of the image instead of just bounding boxes this problem know as image segmentation is what gaming a at all explored at Facebook AI using an architecture know as Mask R-CNN. In Mask R-CNN in a fully convolutional network or a CNN is added to the top of the CNN features of a Faster R-CNN to generate a Mask segmentation output.

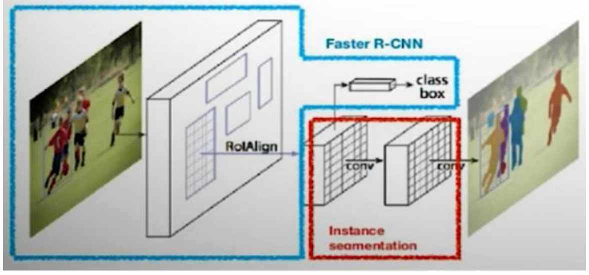


Fig.4: MASK R-CNN

2. METHODOLOGY

We are using machine learning techniques like Faster R-CNN and Mask R-CNN to detect rotten fruits or vegetables.

2.1 FASTER R-CNN

Faster R-CNN speeding up the region proposal even with these advancements there was still one remaining bottleneck in the Fast R-CNN process the region proposal.

In the very first step to detecting these locations of the objects is generating a bunch of potential bounding boxes or regions of interest to test in Fast R-CNN the proposals were created using the selected search a fairly slow process it was found to be the bottleneck of the overall process. The inside of Faster R-CNN was the region proposals depended on the features of the image that were already calculated but a fourth pass of the CNN.

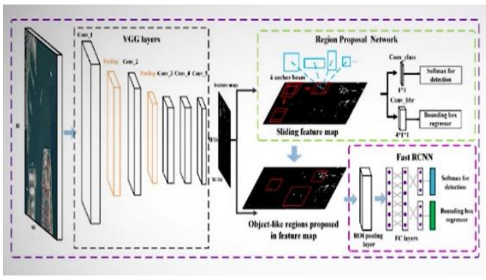


Fig.5: FASTER R-CNN

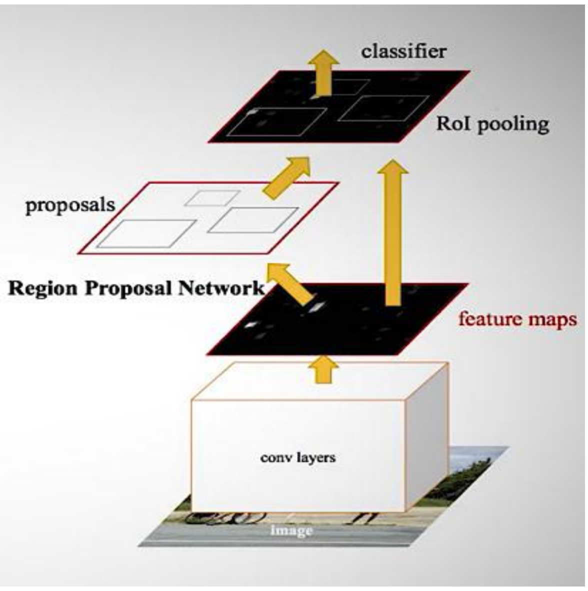


Fig.6: FASTER R-CNN

First step of classification the solution was to reuse those same CNN results for region proposals instead of running a separate selective search algorithm.

The single CNN is used to both carry out the region proposals and classification this way only one CNN needs to be trained and we get the region proposals almost for free all termed cost-free region proposals.

2.1.1 How Regions are generated ?

Faster R-CNN has a fully convolutional network on top of the features of the CNN grating was know as region proposal network.

2.1.2 Region proposal network

The region proposal network proposed by passing a sliding window over the CNN feature map and as each window outputting K potential bounding boxes and scores for how good each of these boxes is expected to be for each such anchor box we offered one bounding box and score per position in the image we then pass each such bounding box that is likely to be an object in to Fast R-CNN to generate a classification and tighten the bounding boxes.

2.1.3 Flowchart of FASTER R-CNN

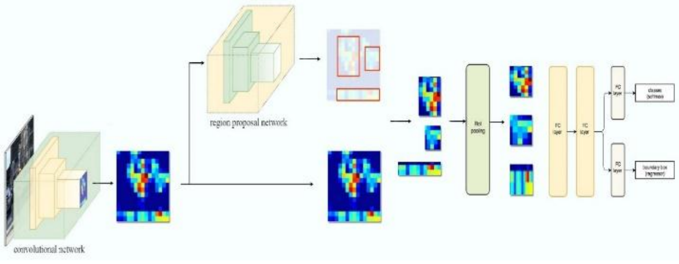


Fig.7: Flowchart of Faster R-CNN

2.2 MASK R-CNN

This is Parallel to classification and bounding box regressor network of the Faster R-CNN model Mask R-CNN does is by adding a branch to Faster R-CNN in that outputs a binary mask this is whether or not a given pixel is part of an object this branch is just a fully convolutional network on top of a CNN based feature map with the following outputs but the inputs which are seen in feature map and the outputs is a matrix of ones on all the locations where the pixel belongs to the objects and 0 elsewhere this is also known as a binary mask however the Mask R-CNN authors had to make one small adjustment to make the pipeline work as expected.

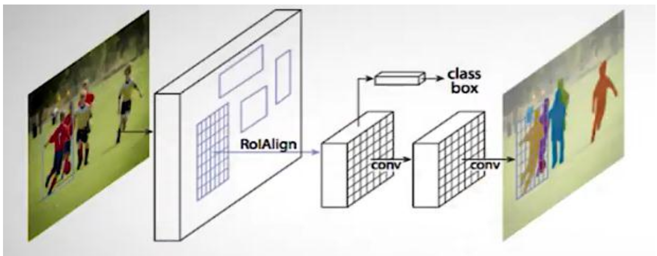


Fig.8: MASK R-CNN

2.2.1 ROI ALIGN

ROI ALIGN is Realigning Ropu to be more accurate when run without modifications on the original Faster R-CNN architecture the Mask R-CNN authors realized that the regions of the feature map selected by ROI pool was slightly misaligned from the regions of the original image since the image segmentation required pixel level specificity unlike bounding boxes this naturally let to inaccuracies the authors were able to solve this by cleverly adjusting the ROI pool to be more precisely aligned using a method known as ROI aligned once these masks are generated the Mask R-CNN combines them with such classifications and bounding boxes from Faster R-CNN to generate precise segmentations.

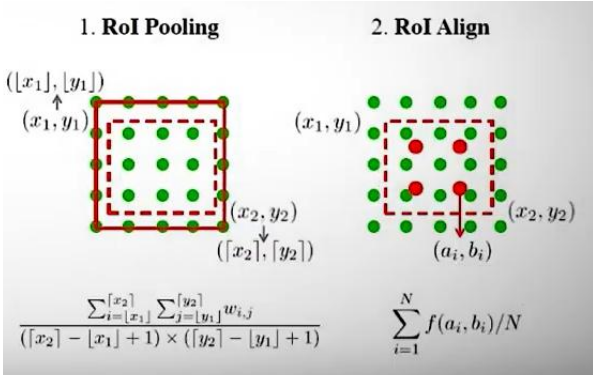

1. RoI Pooling
2. RoI Align
$(\lfloor x_1 \rfloor, \lfloor y_1 \rfloor)$
(x_1, y_1)
(x_2, y_2)
$(\lceil x_2 \rceil, \lceil y_2 \rceil)$
(x_1, y_1)
(x_2, y_2)
(a_i, b_i)
$\frac{\sum_{i=\lfloor x_1 \rfloor}^{\lceil x_2 \rceil} \sum_{j=\lfloor y_1 \rfloor}^{\lceil y_2 \rceil} w_{i,j}}{(\lceil x_2 \rceil - \lfloor x_1 \rfloor + 1) \times (\lceil y_2 \rceil - \lfloor y_1 \rfloor + 1)}$
$\sum_{i=1}^{N} f(a_i, b_i)/N$

Fig.9: ROI Align in MASK R-CNN

2.2.2 Flowchart of MASK R-CNN

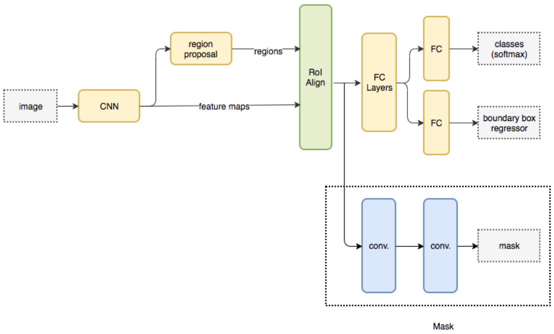
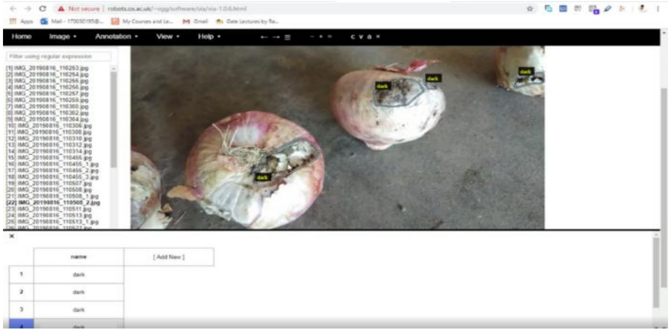


Fig.10: Flowchart of MASK R-CNN

3. IMPLEMENTATION OF MASK R-CNN

3.1 Creation of Custom Dataset

We have chosen onions as dataset to implement these machine learning algorithms. Due to lack of open datasets we had to create a custom dataset. For creating custom dataset, we went to onion warehouse and took many photos of rotten onions and fresh onions. After that we annotated those images using VGG Image Annotator (VIA).The version of VGG Image Annotator is 1.0.1.



	name	[Add New]
1	dark	
2	dark	
3	dark	

Fig.11: Annotating images using VIA

3.2 Step by Step Detection

3.2.1 Anchor sorting and filtering

Anchor sorting and filtering is a function which produces multiple bounding boxes(with different sizes and aspect ratios) while centering on each pixel.The bounding boxes which are produced are called anchor boxes. In the training dataset we assume everty single anchor box as a training example, in order to train the model,we need to assign two types of labels for each anchor box. The first label is category of the target present in the anchor box (category) and the second label is offset of the ground-truth bounding box which is relative to the anchor box (offset). In object detection, the primary task is to generate various anchor boxes and to predict the categories and offsets for every anchor box. And other tasks are to adjust the anchor box position according to the predicted offset to obtain the bounding boxes to be used for prediction and ultimately filter out the prediction bounding boxes that needs to be the output. It conceptualizes every step of the first stage (Region Proposal Network), it also exposes positive and negative anchors(along with anchor box refinement).

3.2.2 Region Proposal Networks

A Region Proposal Network (RPN) inputs an image (of every size) and outputs a set of object proposals (which are rectangular in shape), each with a specific score. We model this process with a fully convolutional network. We assume that both nets share a common set of convolutional layers. In order to generate region proposals, we have to slide a small network over the convolutional feature map output by the last shared convolutional layer. This small network takes as an input spatial window (with the dimensions n×n) as the input convolutional feature map. Every sliding window is mapped to a lower-dimensional feature map which is 256-d for ZF and 512d for VGG with ReLU activation function. This feature is fed into two sibling layer which are fully connected with a box-regression layer (reg) and a box-classification layer (cls).

The mini-network operates in a sliding-window fashion, the fully-connected layers are shared among all spatial locations. This architecture is normally implemented with a convolutional layer (with dimension of nxn) followed by two sibling convolutional layers (with dimension of 1x1) (for reg and cls, respectively).

3.2.3 Anchors

In location of every sliding-window, we have to simultaneously predict various region proposals where the number of maximum possible proposals for each location is denoted as K. So the regression layer has 4K outputs encoding the coordinates of K boxes and the CLS layer outputs 2K scores that estimate probability of every object. The K proposals are parameterized substitutional function to K reference boxes which we call anchors. An anchor is centered at the sliding window and it is associated with a scale and an aspect ratio. By default, we use three scales and three aspect ratios, yielding K = 9 anchors at each sliding position. For a convolutional feature map of a dimension W × H (typically ~2,400), there are WHK anchors in total.

3.2.4 Bounding Box Refinement

Bounding-box refinement is a popular technique to refine or predict localization boxes in recent object detection approaches. Generally, bounding-box regressors are trained to regress from either region proposals or fixed anchor boxes to nearby bounding boxes of a pre-defined target object classes. When it is trained on a relatively small set of annotated training dataset it successfully generalizes to unseen classes, and can be used to improve localization in many vision problems.

MASK R-CNN is first proposed to merge boxes during iterative localization. Relation network proposes to learn the relation between bounding boxes. Recently, IoU-Net proposes to learn the IoU between the predicted bounding box and the ground-truth bounding box. IoU-NMS is then applied to the detection boxes and guided by the learned IoU. Different from IoU-Net, we propose to learn the localization variance from a probabilistic perspective. It enables us to learn the variances for the four coordinates of a predicted bounding box separately 2 instead of only IoU. Our var voting determine the new location of a selected box based on the variances of neighbouring bounding boxes learned by KL Loss, which can work together with soft-NMS.

3.2.5 Mask Generation

Mask Generation is done by Mask R-CNN which is a deep neural network aimed to solve instance segmentation problem in machine learning or computer vision. In other words, it can separate different objects in an image or in a video. It gives you the object bounding boxes, classes and masks when an image is given as an input to it.

3.2.6 Weighted Histogram

It displays the weighted distribution of the data. If it displays proportions (rather than raw counts) then the heights of the bars are the sum of the standardized weights of the observations within each bin which depends on the meaning of the weight variables.

3.2.7 Logging to Tensor Board:

Machine learning always involves in understanding key metrics such as loss and how they change as training progresses. These metrics will help you to understand if you're overfitting. For example, if you're unnecessarily training for too long. You can compare these metrics across different training runs to help debug and improve your model. Tensor Board's Scalar dashboard allows you to anticipate these metrics using a simple API with less effort.

4. RESULTS:

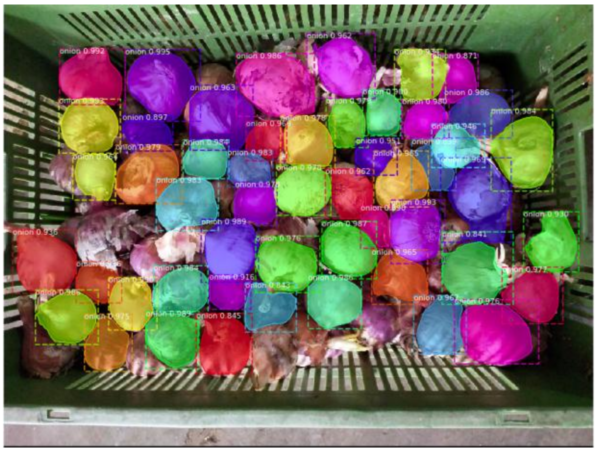


Fig.12: Detection of Onions using Trained MASK R-CNN model

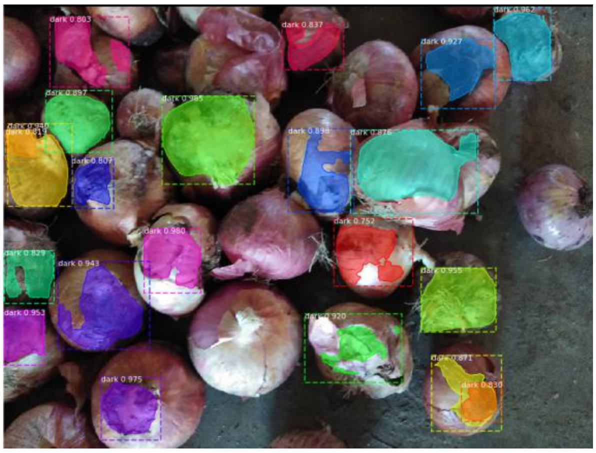


Fig.13: Detection of Defective Parts of Onions using Trained MASK R-CNN model

5. CONCLUSION AND FUTURESCOPE:

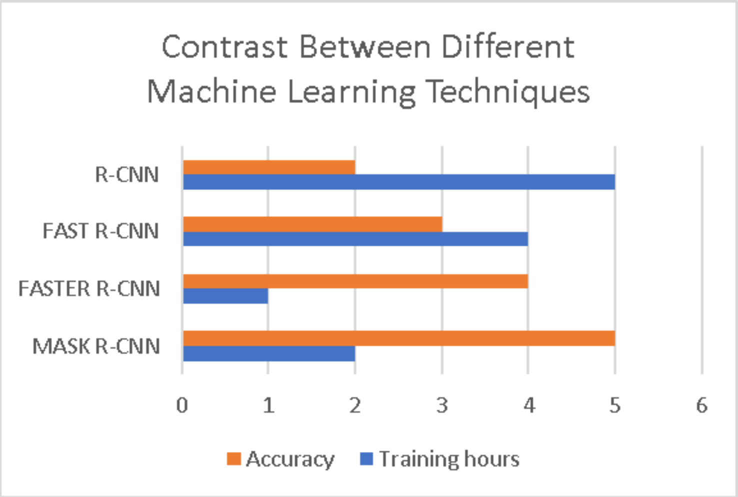


Fig.14: Comparison beween Accuracy and Training Hours of ML Models

We have chosen MASK R-CNN over all the machine learning techniques as it is more efficient and accurate. We got 0.9134 accuracy by using trained MASK R-CNN model. The limitation we found with MASK R-CNN model is that it requires costly hardware to handle the tasks effectively.

This MASK R-CNN model can also be used for the following:

- To identify suspects using public surveillance cameras.
- To create an auto aiming software for fighter planes.

6. REFERENCES:

[1] "Mask R-CNN" Facebook AI Research (FAIR) - Kaiming He, Georgia Gkioxari, Piotr Dollar and Ross Girshick March 2017

[2] Tang, P., Wang, C., Wang, X., Liu, W., Zeng, W., Wang, J.: Object detection in videos by high quality object linking. arXiv preprint arXiv:1801.09823 (2018)

[3] Arnab, A., Torr, P.H.: Pixelwise instance segmentation with a dynamically instantiated network. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,pp. 441–450 (2017)

[4] Girshick, R., Iandola, F., Darrell, T., Malik, J.: Deformable part models are convolutional neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,pp. 437–446 (2015)

[5] Fathi, A., et al.: Semantic instance segmentation via deep metric learning. arXiv preprint arXiv : 1703.10277 (2017)

[6] Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. Proceedings of the

IEEE Conference on Computer Vision and Pattern Recognition, pp.3431–3440 (2015)

[7] Li, Y., Qi, H., Dai, J., Ji, X., Wei, Y.: Fully convolutional instance-aware semantic segmentation. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp.2359–2367 (2017)

[8] Sun, C., Shrivastava, A., Singh, S., Gupta, A.: Revisiting unreasonable effectiveness of data in deep learning era. In: Proceedings of the IEEE International Conference on Computer Vision, pp.843–852 (2017)

[9] Hayder, Z., He, X., Salzmann, M.: Shape-aware instance segmentation (2016)