ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/335855392

# MinneApple: A Benchmark Dataset for Apple Detection and Segmentation

**Preprint** · September 2019

CITATIONS
0

READS
438

**3 authors:**

Nicolai Häni
University of Minnesota Twin Cities
**13** PUBLICATIONS **111** CITATIONS
SEE PROFILE

Pravakar Roy
Microsoft AI and Research
**22** PUBLICATIONS **137** CITATIONS
SEE PROFILE

Volkan Isler
University of Minnesota Twin Cities
**204** PUBLICATIONS **3,930** CITATIONS
SEE PROFILE

**Some of the authors of this publication are also working on these related projects:**

Project SURVEYING AND SERVOING AS CANONICAL TASKS TO ENABLE FUTURE FARMS WITH COMMERCIAL OFF-THE-SHELF ROBOTS View project

Project SURVEYING AND SERVOING AS CANONICAL TASKS TO ENABLE FUTURE FARMS WITH COMMERCIAL OFF-THE-SHELF ROBOTS View project

All content following this page was uploaded by Nicolai Häni on 19 September 2019.

The user has requested enhancement of the downloaded file.

arXiv:1909.06441v1 [cs.CV] 13 Sep 2019

# MinneApple: A Benchmark Dataset for Apple Detection and Segmentation

Nicolai Häni[1], Pravakar Roy[1] and Volkan Isler[1]

***Abstract*— In this work, we present a new dataset to advance the state-of-the-art in fruit detection, segmentation, and counting in orchard environments. While there has been significant recent interest in solving these problems, the lack of a unified dataset has made it difficult to compare results. We hope to enable direct comparisons by providing a large variety of high-resolution images acquired in orchards, together with human annotations of the fruit on trees. The fruits are labeled using polygonal masks for each object instance to aid in precise object detection, localization, and segmentation. Additionally, we provide data for patch-based counting of clustered fruits. Our dataset contains over 41, 000 annotated object instances in 1000 images. We present a detailed overview of the dataset together with baseline performance analysis for bounding box detection, segmentation, and fruit counting as well as representative results for yield estimation. We make this dataset publicly available and host a CodaLab challenge to encourage comparison of results on a common dataset. To download the data and learn more about MinneApple please see the project website: `http://rsn.cs.umn.edu/index.php/MinneApple`. Up to date information is available online.**

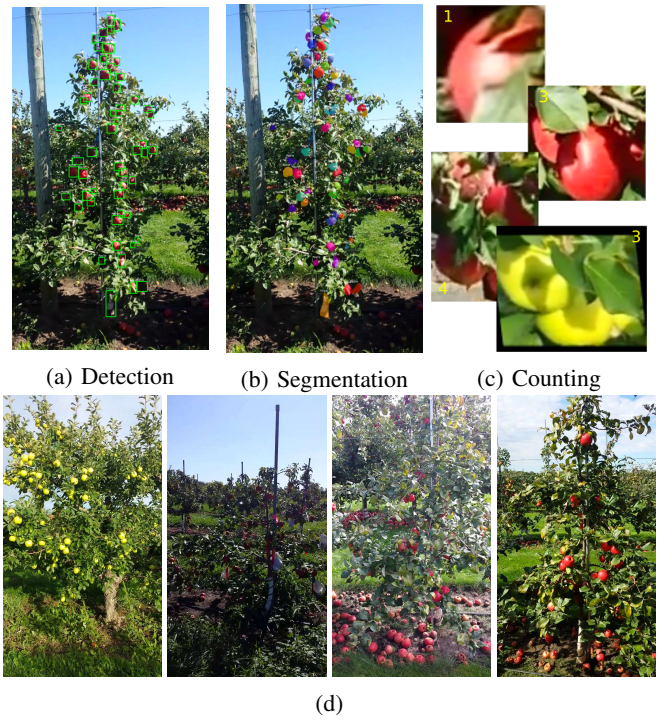
## I. Introduction

Detection, counting, and localization of fruits in orchards are important tasks in agricultural automation. They allow farmers to manage and optimize resources and make informed decisions during harvest. Fruit detection and localization are also precursors to automated fruit picking, which is one of the most labor-intensive processes.

Researchers have used a variety of sensor technologies and algorithms to tackle fruit detection, but cameras together with computer vision techniques are the most common. Unfortunately, using computer vision techniques in outdoor orchard settings comes with a unique set of challenges: 1) varying illumination conditions, 2) varying appearances of fruits on trees, 3) and fruit occlusion by foliage, branches or other fruits (see Figure 1d). While early detection methods primarily relied on hand-designed features [11] recent years have seen the incorporation of deep learning methods [28], [1], [2], [26], [12]. However, due to the lack of standardized benchmark datasets and testing metrics for precision agriculture, one can not compare these approaches directly with each other. In this paper, we introduce a new dataset and benchmark evaluation suit for apple detection, segmentation, and counting in orchard settings together with analysis of baseline algorithms.

*This work was not supported by any organization

[1]The authors are with the Department of Computer Science and Engineering, University of Minnesota, Minneapolis, MN, 55455, USA haeni001@umn.edu, royxx268@umn.edu, isler@umn.edu




(a) Detection (b) Segmentation (c) Counting

(d)

Fig. 1: MinneApple contains precise semantic object instance annotations, from which one can extract bounding boxes for detection 1a and semantic labels 1b. We also provide an additional dataset to evaluate patch based counting of overlapping fruits 1c. MinneApple contains data from 17 different tree rows sporting large variety 1d.

Benchmark datasets have been popular and at the forefront of progress in computer vision. Many of the popular datasets in computer vision [4], [9], [19] contain a large number of images and categories. The COCO dataset, for example, contains a category for apples. However, an apple detector trained on this dataset will perform poorly in orchards, since the dataset was created to detect apples in general settings. These popular datasets additionally contain a small number of instances per image and contain iconic images, where the objects to be detected take up a large portion of the image, all of which hurts detection performance.

In contrast, MinneApple contains $1000$ images with over $41,000$ labeled instances of apples. The object instances are small compared to the image size, and a single image may contain between 1 and 120 objects. We collected data from multiple fruit varieties over two years, to create the largest and most diverse dataset of its kind. We hope that this dataset

will provide an important stepping stone in advancing the field of precision agriculture.

The rest of the paper is organized as follows: In section II, we introduce current datasets and testing methods, as well as some of the algorithms used as baselines. Then we introduce the dataset and annotation procedure in section III. Section V contains the dataset statistics and we evaluate benchmark algorithms in section VI.

## II. Related Work

Many computer vision techniques rely on large datasets for training, testing, and comparing different approaches to a given problem. They not only provide the means to train and evaluate new algorithms but encourage direct comparison of results. Ultimately, they provide the means for researchers to tackle new and more challenging research problems. The ImageNet [4], Pascal VOC [9] and the COCO [19] datasets have made millions of *labeled* images available to the public and enabled breakthroughs in image classification and object segmentation. Similarly, researchers released specialized datasets for autonomous driving [3], [30], [10] or pedestrian detection [8], [5]. While precision automation and automated yield mapping have seen much research effort [1], [29], [25], [2], [13], [12], each of these papers used their own datasets of varying completeness and level of detail.

### A. Fruit detection

The first step in a yield estimation or fruit picking pipeline is the detection of the fruit. Early methods mostly relied on static color thresholds for detection. The limitations of these methods were often compensated by adding additional sensors, such as thermal- or Near Infrared (NIR) cameras. Gongal et al. [11] offer a comprehensive overview of these early detection methods. More recent papers used object detection networks to detect fruits [28], [1], [12]. Sa et al. [28] used a combination of NRI and RGB images of fruits in indoor environments. Their dataset contains only 122 images from which training and test data are extracted. Bargoti and Underwood [1] used a similar network for apple detection. They released their dataset of roughly 1000 image crops that they used for training and testing. The images are of size $308 \times 202$ pixels with circular annotation of the fruits. Stein et al. [29] used a Faster RCNN network to detect mango fruits. [2] used a Fully Convolutional Network (FCN) to compute feature maps. Integrating these feature maps gives them a yield estimate. They split the dataset of 71 images $50/50$ between training and testing. In our previous work [25], [26], [27], [12] we presented results on HD sized images, showing parts of an orchard row. We presented multiple methods including semi-supervised Gaussian Mixture Model (GMM) a Faster R-CNN object detector and a semantic segmentation network. The training dataset contained 100 images, and the test set contained 207 images. In contrast, we have increased the size of the dataset by a factor of $\times 3.5$ for this work.

### B. Fruit counting

After detection of the fruits, they need to be counted. Rahnemoonfar and Sheppard [21] used synthetic data to train a network to classify images according to fruit counts. They test their approach on 100 annotated images. Chen et al. [2] used a fully convolutional network together with a regression head for counting. They used a total of 71 orange- and 21 apple images from which they extracted image patches for training. Roy and Isler [26] proposed an unsupervised counting method based on Gaussian Mixture Models. They used a manually annotated dataset of 440 images for testing. In our previous work [13], we used a neural network to count clustered fruits. We trained a network on 13000 patches and tested our approach on 4 different datasets with a total of 2800 images.

### C. Comparison of Datasets

Table I summarizes the problem with current fruit detection and counting datasets. Because labeling effort is time-consuming and costly, researchers have focused on small datasets with little if any in dataset variety. Acquired images are chopped into smaller chunks to increase the dataset size artificially. These crops show only a small portion of the original image, which shows in the small number of labeled

TABLE I: Comparison of datasets used in recent research papers on apple detection and counting.

| **Fruit Detection** | type | # train images | # test images | # annotations | # scenes | resolution | ground truth | public |
|---|---|---|---|---|---|---|---|---|
| Bargoti and Underwood [1] | outdoor | 729 | 112 | 5765 | 1 | $308 \times 202$ | circles | yes |
| Stein et al. [29] | outdoor | 1154 | 250 | 7065 | 1 | $500 \times 500$ | circles | yes |
| Sa et al. [28] | indoor | 100 | 22 | 359 | 1 | $1296 \times 964$ | boxes | yes |
| Liu et al. [20] | outdoor | 100 | - | - | 1 | $1920 \times 1200$ | boxes | no |
| **MinneApple (ours)** | outdoor | 670 | 331 | 41,325 | 17 | $1280 \times 720$ | polygons | yes |

| **Fruit Counting** | type | # train images | # test images | # scenes | resolution | public |
|---|---|---|---|---|---|---|
| Chen et al. [2] | outdoor | 47 | 45 | 1 | $1280 \times 960$ | no |
| Rahnemoonfar and Sheppard [21] | synthetic | 24,000 | 2400 | 1 | $128 \times 128$ | no |
| **MinneApple (ours)** | outdoor | 64597 | 5764 | 6 | varying | yes |

fruits. While this technique increases dataset size, it does not increase dataset variation, and the developed methods are prone to overfitting. Another issue is that the whole datasets are split into training/testing. Such splits lead to in-dataset testing, which makes it impossible to analyze an algorithm on its generalization capabilities.

The MinneApple dataset tries to correct these problems. The dataset contains only full resolution images. Data for the train/test splits are taken from different tree rows and different years. We included a variety of apple species and illumination conditions to avoid overfitting. The MinneApple dataset gives researchers a tool to test their algorithms in an unbiased way and compare to other approaches.

## III. Image Collection

The data for this paper were collected at the University of Minnesota's Horticultural Research Center (HRC) between June 2015 and September 2016. Since this is a university orchard, used for phenotyping research, it is home to a large variety of apple tree species. We collected video footage from different sections of the orchard using a standard Samsung Galaxy S4 cell phone. During data collection, we acquired video footage by facing the camera horizontally at a single side of a tree row and moving along the tree row with a speed of 1 m/s. We then extracted every fifth image from these video sequences. For the test datasets, we extracted every 30th image.

### A. Detection and Segmentation Datasets

For detection and localization of the fruits, we collected 17 different datasets over two years, ten for training and seven for evaluation. We included fruits of different colors and at different stages of the ripening cycle. The datasets were taken either from the sunny or shady side of the tree row, and we spread out data capture over multiple days to get more varied illumination conditions. See Figure 2 for samples of the annotated images in our dataset.

**Training Sets:** We sampled ten datasets from six different tree rows for training purposes. Dataset show either the front (sunny) or back (shady) side of a tree row. From these ten datasets, we randomly selected and annotated 670 images of resolution $1280 \times 720$ pixels. All of these datasets were acquired in 2015 at the HRC, and they contain different apple varieties, fruits across different growing stages and a variety of tree shapes.

**Test Sets:** To evaluate detection/segmentation and yield estimation performance; we arbitrarily chose four different sections of the orchard. We collected seven videos from these four segments in 2016. Acquiring datasets during different years guarantees the independence of the test set. Additionally, we collected yield estimation ground truth for three tree rows by hand collecting and counting per tree yield and by measuring fruit diameters after harvest. Yield estimation includes additional steps, such as fruit tracking and tree row merging. Since we do not include a baseline for tracking, we provide only anecdotal results for yield estimation in this paper.

### B. Counting Datasets

**Training Sets:** We provide two annotated datasets to train patch-based counting approaches. One of these datasets contains green, and one contains red apples, and both were acquired in 2015. Both datasets were obtained from the sunny side of the tree row. In total, we obtained 13000 image patches, which we annotated manually with a ground truth count. Additionally, we extracted 4500 patches at random that do not contain apples as negative examples. See Figure 2 for samples of the annotated images in our dataset.

**Test Sets:** The test dataset consists of a total of 2874 image patches taken from four image sequences. Two of the test datasets contain red apples, one contains greens, and one contains a mixture of colors. Additionally, we acquired the fourth dataset from a further distance to test the algorithms generalization capability for counting low-resolution fruits.

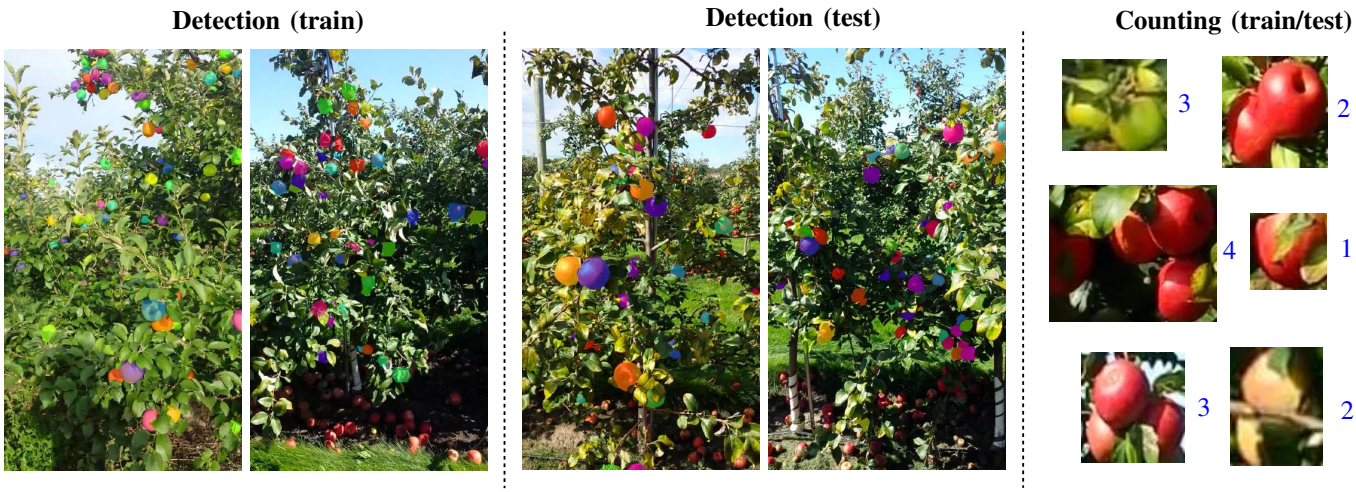



Fig. 2: Samples of annotated images of the detection, segmentation and counting datasets. The detection/segmentation datasets are annotated with object instance masks, while the counting dataset contains image patches and a corresponding ground truth count.

## IV. Image Annotation

We next describe how we labeled images for training and evaluation. We follow the annotation method in [12]. Following established protocol, annotations for train and validation data will be released, but not for test. We are currently finalizing the evaluation server for automatic evaluation on the test set.

**Detection and segmentation:** Fruits for the detection and segmentation datasets were annotated using the excellent VGG annotator tool [7]. We used polygons to label fruits on trees in the foreground, while the ones on the ground and trees in the background were not tagged. Additionally, we labeled the tree trunks where visible. Each of the objects in the scene was then categorized into fruit or tree trunk. We used an internally recruited workforce for the instance labeling task. Due to the large number of instances per image, the small object size and the many occlusions of fruits instances, labeling is an arduous task. Labeling a single image takes up to 30 minutes, which translates to roughly 18 workhours per 1000 instances. As such, we chose to assign each image to only a single worker for labeling. Each worker is instructed in proper labeling techniques before they can begin to annotate. After the worker annotated the first ten image frames, we conducted an in-person review to give feedback and issue the first round of corrections. While the instruction and initial feedback improved annotation quality considerably, we performed an additional verification step to correct each object instance if necessary.

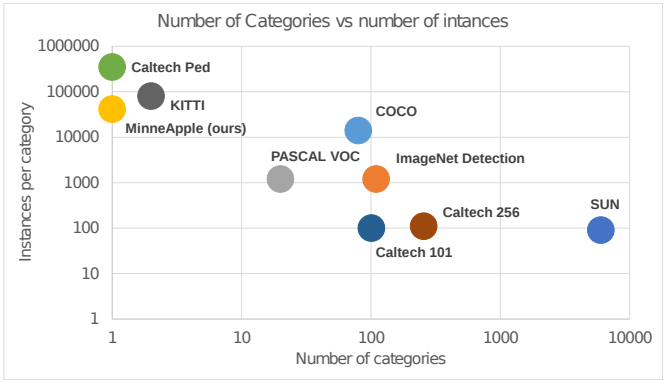
**Patch-based counting:** For the patch-based counting method we used a semi-supervised GMM detector [26] to detect image patches that are likely to contain fruits. The patches were cropped and annotated by hand with a single ground truth integer, representing the count. Due to the small resolution these patches and the large volume, the annotation task proved to be error-prone. We had two different workers annotate each image, and disparities were resolved by a third worker through a validation process.

## V. Dataset Statistics

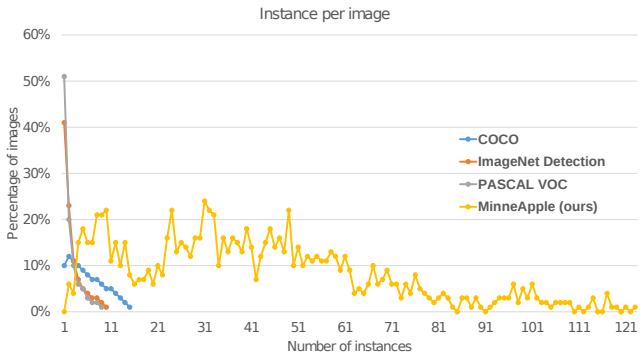
Here, we give an overview of the properties of MinneApple in comparison to other object detection datasets. These include COCO [19], ImageNet Detection [4] and PASCAL VOC [9]. Each of these datasets varies considerably in the number of annotated images, image types, number of categories, number of instances per image, and the size of the annotated objects. The MS COCO dataset was created to show common objects in their natural context. The goal of the ImageNet Detection dataset was to detect a large number of object categories. PASCAL VOC contains fewer categories but focuses on objects in natural images. Our MinneApple dataset, on the other hand, focuses on detecting many small objects in highly cluttered environments.

A summary of the datasets showing the number of instances per category is shown in Figure 3a. MinneApple contains fewer categories but far more instances per category than most datasets. In this it is comparable to other specialized datasets such as the Caltech Pedestrian Detection [6] dataset or the KITTI [10] dataset. Figure 3b shows the number of instances per image in comparison to other datasets. MinneApple contains 1.5 categories and 41.2 instances on average per image. In contrast, the COCO dataset has 3.5 categories and 7.7 instances, and the ImageNet and PASCAL VOC datasets both have less than two categories and three instances per image on average. The spread of the number of instances per image compared to COCO, ImageNet, and PASCAL VOC, is also more extensive. The MinneApple dataset can contain between 1 and 120 objects, while the other datasets have maximally 15 object instances.

Finally, we analyze the average size of objects in the dataset. In general, smaller objects are harder to detect and require specialized network structures [16]. For COCO, PASCAL VOC and ImageNet Detection, roughly 50% of all objects occupy no more than 10% of the image itself. The other 50% contains objects that occupy between 10 and 100% of the image (evenly distributed). Our MinneApple dataset contains almost exclusively small instances. The

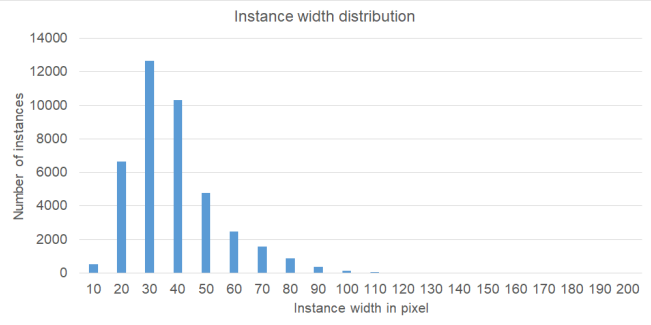



(a)

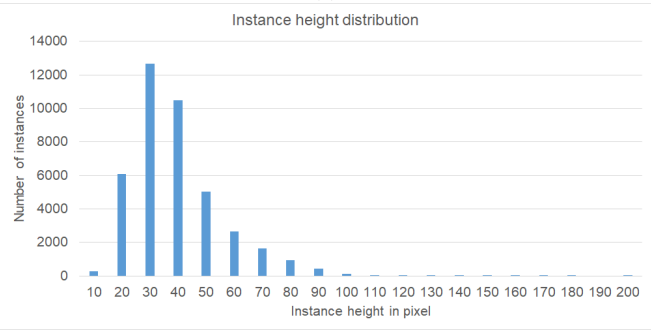



(b)

Fig. 3: a) Number of annotated instances per category for some common datasets in comparison. b) Distribution of number of annotated object instances per image for COCO, ImageNet Detection, PASCAL VOC compared with MinneApple. While the other datasets contain mainly 1-5 objects, ours contains up to 120 instances per image.




(a)




(b)

Fig. 4: Width and height distribution of the objects in our dataset. The dataset contains mainly small object instances with area $< 50^2$ pixels

average object size is only $40 \times 40$ pixels in an image of $1280 \times 720$ pixels, making up only 0.17% of the original image size.

## VI. Algorithmic Analysis

We run a set of state-of-the-art algorithms on each the tasks of object detection, segmentation, and counting to establish a common baseline for future work.

### A. Detection and Segmentation Baselines

For the following experiments, we take a subset of 600 images from our dataset for training. The leftover 30 images are used for validation during training. We test each algorithm individually on each of the 331 test images and report average performance over the test dataset.

**Detection evaluation metrics:** For bounding box detection, we follow established evaluation protocols used by other object detection datasets [9], [19]. We report Average Precision (AP) as our main evaluation metric. Namely, we use AP starting at Intersection over Union (IoU) threshold 0.5 and increase it in intervals of 0.05 up to 0.95 (shorthand notation is AP@0.5:0.05:0.95). Additionally, we provide AP@0.5 and AP@0.75. Since our dataset contains many small objects, we report AP scores for small (object area $< 32^2$ pixels), middle ($32^2 \geq$ object area $\geq 96^2$) and large objects (area $> 96^2$). We evaluate three different models.

**Faster RCNN:** The latest implementation of Faster RCNN [22] with a ResNet-50 backbone. The network uses pretrained COCO weights for initialization. Faster RCNN consists of a region proposal head and two branches for bounding box regression and classification. We used parameters in the paper for optimization.

**Tiled Faster RCNN:** A reimplementation of Bargoti and Underwoods [1] proposed model. Due to memory constraints, they split the training images into $500 \times 500$ pixel chunks, with an overlap of 50 pixels. The detections of the individual chunks are aggregated and filtered using non-maximum suppression. We added a ResNet-50 backbone, a Feature Pyramid (FPN) [17] head for region proposal to the network and trained it with focal loss [18]. The network was initialized with weights pretrained on COCO [19]. We follow [1], [12] in our choice of parameters for optimization.

**Mask RCNN:** Implementation of a Mask RCNN [14] with a ResNet-50 backbone, pretrained on the COCO dataset. In addition to using bounding box inputs as Faster RCNN, Mask RCNN has an additional branch predicting the object instance mask.

If we compare the bounding box detection results in Table II, we find that processing the image in a tiled fashion performs worse than the detectors operating on the whole image. We hypothesize that this is due to the additional filtering step at the end, where non-maximum suppression is used to filter out overlapping bounding boxes. If we compare the two state of the art object detectors, we find that Faster RCNN slightly outperforms Mask RCNN. This is somewhat unexpected since Mask RCNN has access to additional information (the instance masks). Further, we find that all the detectors struggle on smaller object instances. Future research should focus on improving the object detection performance on small and medium-sized objects to achieve significant gains in overall performance. Our findings confirm Hoiem et al. [16], which found that object size is

TABLE II: Fruit detection benchmark results for object detection approaches. Higher numbers are better and the bold marked numbers indicate the highest performing approach.

| | | Metric | | | | | |
|---|---|---|---|---|---|---|---|
| | | AP [@ 0.5:0.05:0.95] | AP [@ 0.5] | AP [@ 0.75] | AP [small] | AP [middle] | AP [large] |
| Method | Tiled FRCNN [1] | 0.341 | 0.639 | 0.339 | 0.197 | 0.519 | 0.208 |
| | Faster RCNN [22] | **0.438** | **0.775** | **0.455** | **0.297** | **0.578** | **0.871** |
| | Mask RCNN [14] | 0.433 | 0.763 | 0.449 | 0.295 | 0.571 | 0.809 |

one of the main error factors in object detection.

**Semantic segmentation evaluation metrics:** While bounding box prediction is the method predominantly used for object detection; we recognize that there exist other methods which only achieve bounding box prediction after an additional post-processing step. These methods include mainly detection through semantic segmentation. To avoid explicit bias towards bounding box prediction methods, we introduce separate benchmark algorithms for semantic segmentation. For evaluation, we follow the established metrics used by the COCO dataset [19]. We report Intersection over Union (IoU) as the primary challenge metric. Additionally, we report class IoU for apples, pixel accuracy, and class accuracy for apple pixels. We evaluate four different models.

**Semi-supervised GMM:** A semi-supervised clustering method based on Gaussian Mixture Models (GMM), developed by Roy and Isler [25]. The model is pretrained on an unlabeled dataset, different from the ones contained in the train and test sets.

**User-supervised GMM:** The same model as in the semi-supervised case. The method uses human supervision to create a single model per tree row in the test set.

**UNet (not pretrained):** A semantic segmentation network, based on a fully convolutional network architecture [23], [12]. The images in the train and test sets are split into $224 \times 224$ sized chunks, and the weights of the network are initialized randomly.

**UNet (pretrained):** the same model as the one before, but the weights are initialized from a pretrained ImageNet network.

TABLE III: Fruit detection benchmark results for semantic segmentation approaches. Higher numbers are better and the bold marked numbers indicate the highest performing approach.

| | | Metric | | | |
|---|---|---|---|---|---|
| | | IoU | Class IoU | Pixel Acc. | Class Acc. |
| Method | Semi-supervised GMM [27] | 0.635 | 0.341 | 0.968 | 0.455 |
| | User-supervised GMM [27] | 0.649 | **0.455** | 0.959 | 0.634 |
| | UNet [12] (no pretraining) | 0.678 | 0.397 | 0.960 | 0.818 |
| | UNet [12] (pre-trained) | **0.685** | 0.410 | **0.962** | **0.848** |

If we compare the average performance of all methods in Table III, we see that UNet with pretrained weights outperforms all others. Only in the class IoU case, the user-supervised GMM method outperforms the UNet. These results are in direct contrast to our previous work [12], where the user-supervised GMM outperformed UNet. Keep in mind though, that for this work the amount of training data increased by almost one order of magnitude. These results indicate that the deep learning network previously underperformed due to a lack of data. We further find that using pretrained weights improves performance slightly. We hypothesize that this is due to the large variety of images found in the ImageNet dataset, which allows the network to learn more descriptive features during pretraining.

### *B. Patch-based Fruit Counting Baselines*

Next to the benchmark dataset for object detection and segmentation, we provide a dataset for patch-based fruit counting. We report baseline results for approaches which were previously published in [13]. We evaluate two approaches. **GMM:** An unsupervised method based on Gaussian Mixture Models. This method fits a mixture of Gaussians probability distribution to a previously segmented image. **CNN** This method uses a network to classify the fruits into k distinct classes. The network is based on a ResNet50 [15] backbone, and we choose to classify six classes. Table IV shows the counting accuracy. The ResNet50 network outperforms the GMM model on all of the test sets.

TABLE IV: Fruit cluster counting benchmark results.

| Method | Dataset 1 | Dataset 2 | Dataset 3 | Dataset 4 |
|---|---|---|---|---|
| GMM [26] | 88.0 % | 81.8 % | 77.2 % | 76.1 % |
| CNN [13] | **88.8** % | **92.68 %** | **95.1 %** | **88.5 %** |

### *C. Yield estimation*

Fruit detection and counting are integral to solving the problem of yield estimation. However, yield estimation contains additional steps to map detections and counts to tree row yield. For one, we need to track fruits across the image sequence to avoid double counting. We address tracking fruits across images in [25], [26], [27]. Due to the planar structure of modern apple orchards, fruits can be seen from both sides of the tree row. We propose a solution to this problem in [24]. Table V shows yield estimation results using these tracking components together with the unsupervised GMM detection method and the CNN counting. These results have been previously published in [12], and we mention them here for completion. Using this combination of components, we achieve between 95.5 and 97.8% accuracy with respect to the harvested ground truth.

TABLE V: Yield estimation results in terms of fruit counts.

| | Harvested fruit counts | Merged fruit counts from both sides | | Summed fruit counts single sides | |
|---|---|---|---|---|---|
| | | GMM [27] | CNN [12] | GMM [27] | CNN [12] |
| Dataset-1 | 270 | 256 (94.81%) | **258** (95.56%) | 348 (128.89%) | 347 (128.52%) |
| Dataset-2 | 274 | 252 (91.98%) | **268 (97.81%)** | 411 (150%) | 405 (147.81%) |
| Dataset 3 | 414 | 392 (94.68%) | 405 (97.83%) | **422 (101.93%)** | 430 (103.86%) |

## VII. Conclusion

We introduced a new dataset for detecting and segmenting apples in orchards and a second dataset for counting clustered fruits. With this collection of annotated object instances, we hope to help the advancement of object detection, segmentation, and counting of small objects in cluttered environments. In creating this dataset, we wanted to emphasize the need for a diverse and unbiased dataset, containing a large number of object instances and apple varieties between the individual tree rows. Dataset statistics and results from the baseline algorithms indicate that the images contain challenging scenarios for current state-of-the-art object detection algorithms.

There are several promising directions for future work to improve the performance of detection and counting algorithms using this dataset. Our analysis of state-of-the-art object detectors indicates that networks can gain in accuracy by putting a broader focus on small object instances (area $< 32^2$ pixels). Similarly, semantic segmentation networks may employ weighting schemes to address the class imbalance between foreground (object instances) and background pixels. We hope that our dataset will help computer vision researchers working on fruit detection.

To download and learn more about MinneApple please see the project website: `http://rsn.cs.umn.edu/index.php/MinneApple`.

## VIII. Acknowledgements

This work was supported by the USDA NIFA MIN-98-G02, and UMN MnDrive. The authors acknowledge the Minnesota Supercomputing Institute (MSI) at the University of Minnesota for providing resources that contributed to the research results reported within this paper `http://www.msi.umn.edu`.

## References

[1] S. Bargoti and J. Underwood. Deep fruit detection in orchards. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 3626–3633. IEEE, 2017.

[2] S. W. Chen, S. S. Shivakumar, S. Dcunha, J. Das, E. Okon, C. Qu, C. J. Taylor, and V. Kumar. Counting Apples and Oranges With Deep Learning: A Data-Driven Approach. *IEEE Robotics and Automation Letters*, 2(2):781–788, 2017.

[3] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The Cityscapes Dataset for Semantic Urban Scene Understanding. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3213–3223, Las Vegas, NV, USA, June 2016. IEEE.

[4] J. Deng, W. Dong, R. Socher, L.-J. Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, Miami, FL, June 2009. IEEE.

[5] P. Dollar, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: A benchmark. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 304–311, Miami, FL, June 2009. IEEE.

[6] P. Dollar, C. Wojek, B. Schiele, and P. Perona. Pedestrian Detection: An Evaluation of the State of the Art. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(4):743–761, Apr. 2012.

[7] A. Dutta, A. Gupta, and A. Zissermann. *VGG Image Annotator (VIA)*. 2016.

[8] A. Ess, B. Leibe, K. Schindler, and L. van Gool. Robust Multiperson Tracking from a Mobile Platform. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(10):1831–1846, Oct. 2009.

[9] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The Pascal Visual Object Classes (VOC) Challenge. *International Journal of Computer Vision*, 88(2):303–338, June 2010.

[10] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? The KITTI vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361, Providence, RI, June 2012. IEEE.

[11] A. Gongal, S. Amatya, M. Karkee, Q. Zhang, and K. Lewis. Sensors and systems for fruit detection and localization: A review. *Computers and Electronics in Agriculture*, 116:8–19, Aug. 2015.

[12] N. Häni, P. Roy, and V. Isler. A comparative study of fruit detection and counting methods for yield mapping in apple orchards. *Journal of Field Robotics*, Aug. 2019.

[13] N. Häni, P. Roy, and I. Volkan. Apple Counting using Convolutional Neural Networks. In *Intelligent Robots and Systems (IROS), 2018 IEEE International Conference on*. IEEE, 2018.

[14] K. He, G. Gkioxari, P. Dollr, and R. Girshick. Mask R-CNN. *arXiv:1703.06870 [cs]*, Mar. 2017. arXiv: 1703.06870.

[15] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, Las Vegas, NV, USA, June 2016. IEEE.

[16] D. Hoiem, Y. Chodpathumwan, and Q. Dai. Diagnosing Error in Object Detectors. In D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, A. Fitzgibbon, S. Lazebnik, P. Perona, Y. Sato, and C. Schmid, editors, *Computer Vision ECCV 2012*, volume 7574, pages 340–353. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[17] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature Pyramid Networks for Object Detection. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 936–944, Honolulu, HI, July 2017. IEEE.

[18] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar. Focal Loss for Dense Object Detection. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2999–3007, Venice, Oct. 2017. IEEE.

[19] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollr. Microsoft COCO: Common Objects in Context. *arXiv:1405.0312 [cs]*, May 2014. arXiv: 1405.0312.

[20] X. Liu, S. W. Chen, S. Aditya, N. Sivakumar, S. Dcunha, C. Qu, C. J. Taylor, J. Das, and V. Kumar. Robust Fruit Counting: Combining Deep Learning, Tracking, and Structure from Motion. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1045–1052, Madrid, Oct. 2018. IEEE.

[21] Maryam Rahnemoonfar and Clay Sheppard. Deep Count: Fruit Counting Based on Deep Simulated Learning. *Sensors*, 17(12):905, Apr. 2017.

[22] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 91–99. Curran Associates, Inc., 2015.

[23] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[24] P. Roy, W. Dong, and V. Isler. Registering Reconstructions of the Two Sides of Fruit Tree Rows. In *Intelligent Robots and Systems (IROS), 2018 IEEE International Conference on*. IEEE, 2018.

[25] P. Roy and V. Isler. Surveying apple orchards with a monocular vision system. In *International Conference on Automation Science and Engineering (CASE)*, pages 916–921. IEEE, Aug. 2016.

[26] P. Roy and V. Isler. Vision-Based Apple Counting and Yield Estimation. In D. Kuli, Y. Nakamura, O. Khatib, and G. Venture, editors, *2016 International Symposium on Experimental Robotics*, volume 1, pages 478–487. Springer International Publishing, Cham, 2017.

[27] P. Roy, A. Kislay, P. A. Plonski, J. Luby, and V. Isler. Vision-based preharvest yield mapping for apple orchards. *Computers and Electronics in Agriculture*, 164:104897, Sept. 2019.

[28] I. Sa, Z. Ge, F. Dayoub, B. Upcroft, T. Perez, and C. McCool. DeepFruits: A Fruit Detection System Using Deep Neural Networks. *Sensors*, 16(8):1222, Aug. 2016.

[29] M. Stein, S. Bargoti, and J. Underwood. Image Based Mango Fruit Detection, Localisation and Yield Estimation Using Multiple View Geometry. *Sensors*, 16(11):1915, Nov. 2016.

[30] F. Yu, W. Xian, Y. Chen, F. Liu, M. Liao, V. Madhavan, and T. Darrell. BDD100k: A Diverse Driving Video Database with Scalable Annotation Tooling. *arXiv:1805.04687 [cs]*, May 2018. arXiv: 1805.04687.

View publication stats