Image Segmentation for Fruit Detection and Yield Estimation in Apple Orchards

Suchet Bargoti and James P. Underwood
Australian Centre for Field Robotics, The University of Sydney NSW, 2006, Australia
e-mail: s.bargoti@acfr.usyd.edu.au, j.underwood@acfr.usyd.edu.au

Received 6 March 2016; accepted 25 October 2016

Ground vehicles equipped with monocular vision systems are a valuable source of high-resolution image data for precision agriculture applications in orchards. This paper presents an image processing framework for fruit detection and counting using orchard image data. A general-purpose image segmentation approach is used, including two feature learning algorithms; multiscale multilayered perceptrons (MLP) and convolutional neural networks (CNN). These networks were extended by including contextual information about how the image data was captured (metadata), which correlates with some of the appearance variations and/or class distributions observed in the data. The pixel-wise fruit segmentation output is processed using the watershed segmentation (WS) and circular Hough transform (CHT) algorithms to detect and count individual fruits. Experiments were conducted in a commercial apple orchard near Melbourne, Australia. The results show an improvement in fruit segmentation performance with the inclusion of metadata on the previously benchmarked MLP network. We extend this work with CNNs, bringing agrovision closer to the state-of-the-art in computer vision, where although metadata had negligible influence, the best pixel-wise F1-score of 0.791 was achieved. The WS algorithm produced the best apple detection and counting results, with a detection F1-score of 0.861. As a final step, image fruit counts were accumulated over multiple rows at the orchard and compared against the post-harvest fruit counts that were obtained from a grading and counting machine. The count estimates using CNN and WS resulted in the best performance for this data set, with a squared correlation coefficient of $r^2 = 0.826$.

1. INTRODUCTION

Yield estimation and mapping in orchards is important for growers as it facilitates efficient utilization of resources and improves returns per unit area and time. With accurate knowledge of yield distribution and quantity, a grower can efficiently manage processes such as chemigation, fertigation, and thinning. Yield estimation also allows the grower to plan ahead of time their harvest logistics, crop storage, and sales (Aggelopoulou et al., 2011; Nuske, Wilshusen, Achar, Yoder, & Singh, 2014; Payne & Walsh, 2014). The standard approach to get yield information is currently manual sampling, which is labor intensive, expensive, and often destructive (Gemtos, Fountas, Tagarakis, & Liakos, 2013). Constrained by these costs, sampling is often done over a few individual crops, and the measures are extrapolated over the entire farm. Inherent human sampling bias and sparsity in the measurements can result in inaccurate yield estimation.

Recent advances in robotics and automation enable us to gather large-scale data with high spatial and temporal resolution. Unmanned ground vehicles (UGVs) or farmer-operated machinery can be equipped with standard color cameras to capture a detailed representation over large farms. Robust and accurate image processing techniques are required to extract high-level information from this data, such as crop location, health, maturity, crop load (yield), and spatial distribution. Image segmentation is the process of evaluating a semantic description of an image at a pixel or super-pixel level. For image data captured at an orchard, this means automatically labeling each pixel or groups of pixels as representing fruits, flowers, trunks, branches, and foliage. The parsed information can then be used in higher-level tasks, such as individual fruit detection, crop health analysis, and tree detection/branch modeling. This provides the grower with a rich farm inventory and enables further robotic operations, such as autonomous pruning, harvesting, and variable rate spraying.

A standard approach for image segmentation is to transform image regions into discriminative feature representations and parse them through a trained classifier, assigning each region a specific label. For identifying orchard fruits, this means extraction of a feature space that captures properties unique to the fruit, such as its color, texture, reflection, position, and shape. Farm image data

Direct correspondence to Suchet Bargoti, e-mail: s.bargoti@acfr.usyd.edu.au

are generally prone to a wide range of intraclass (within class) variations due to illumination conditions, occlusions, clustering, camera viewpoint, tree types, and seasonal maturity (translating to a different fruit size, shape, and color). Therefore, the feature space and the classifier need to be invariant to such characteristics.

Typically, prior work utilizes hand-engineered features to encode visual attributes that discriminate fruit from nonfruit regions (Nuske et al., 2014; Payne, Walsh, Subedi, & Jarvis, 2014; Wang, Nuske, Bergerman, & Singh, 2013). Although these approaches are well suited for the data set they are designed for, the feature encoding is generally unique to a specific fruit and the conditions under which the data were captured. As a result, the methods are often not transferable to other crops/data sets. In contrast, supervised feature learning approaches can be used to automatically learn transformations that capture the data distribution, enabling their use with different datasets (Hung, Nieto, Taylor, Underwood, & Sukkarieh, 2013). Such approaches have high model complexity and utilize extensive training exemplars to model the variations in the data (Krizhevsky, Sutskever, & Hinton, 2012).

Image classification algorithms have also been shown to benefit from prior knowledge about scene structure. For example, Tighe and Lazebnik (2013) and Brust, Sickert, Simon, Rodner, & Denzler (2015) specify a spatial prior over the labeled classes to aid image segmentation of public image data sets such as LabelMeFacade and SIFT Flow. A greater wealth of prior knowledge is often available in field robotics applications, as we often have access to contextual information about how the data were captured. For a typical image data set, such contextual information, which we refer to as metadata, can include camera trajectories, vehicle location, type of tree/fruit being scanned, distance from camera to trees, sun position, illumination incident angle, weather conditions, and so forth. While a direct physical model of how particular meta-parameters affect the data is not available, aspects of the relationship between metadata and object classes can be learned. Where there is correlation between metadata and appearance variations and/or class distributions, including metadata can improve classification performance. Available at no extra cost to typical data capturing process, our previous works (Bargoti & Underwood, 2015, 2016) have illustrated the use of metadata in allowing simpler classifiers to capture that space and provide a performance boost, leading to similar performance with reduced training exemplars.

This paper presents a study of different image segmentation frameworks for the task of image fruit segmentation. A conventional visible light camera mounted on a UGV is used to capture images of trees during preharvest season as the vehicle drives between different rows at an apple orchard block near Melbourne, Australia (Figure 1). We explore multiple supervised feature learning approaches of varying model complexities, while studying the effects of metadata toward image segmentation performance. The impact of accurate image segmentation for agricultural objectives is evaluated, including fruit detection and yield estimation compared to real-world fruit counts. We subsequently provide an orchard block yield map, which can enable the grower to optimize their farm operations. This paper extends from our previous work with evaluation using new segmentation architectures, including different configurations of the previously used MLP architecture (Bargoti & Underwood, 2016) and the more widely used convolutional neural networks (CNNs). Additional developments are also made for the fruit detection algorithms and performance metrics, and an in-depth discussion about the practical viability of the image processing approach is presented. The primary contributions from this paper are:

- Image fruit segmentation analysis using the previously benchmarked multiscale multilayered perceptron (MLP; Bargoti & Underwood, 2016; Hung et al., 2013) and more recent convolutional neural networks (CNN).
- A study of the utility of different metadata sources and their inclusion within the different classification architectures and training configurations.
- Analysis of the impact of accurate image segmentation toward fruit detection and yield estimation.

The remainder of this paper is organized as follows. Section 2 presents related work on image classification in outdoor scenes, both in the agricultural and the general computer vision context. The image processing components are presented over Sections 3 to 6, following the computation pipeline illustrated in Figure 2. Section 3 presents the different classification frameworks with the inclusion of additional metadata. In Section 4, we outline the experimental setup and the image data set, followed by Section 5 presenting the image segmentation results. Section 6 focuses on fruit detection and yield estimation performed over the image segmentation output. We present the practical lessons learned in Section 7, including new insights into image processing for orchard data. We conclude in Section 8, discussing the future directions of this work.

2. RELATED WORK

Computer vision in agriculture, agrovision (Kapach et al., 2012), has been explored in multiple literature studies for the purposes of fruit detection and yield estimation (Jimenez, Ceres, & Pons, 2000; Kapach et al., 2012; Payne & Walsh, 2014). Agrovision literature is typically data specific, designed for the task at hand, and can often be very heuristic when compared against the most recent work in the general computer vision community (Kapach et al., 2012). In this section, we discuss the key approaches used in agrovision while comparing them against some of the state-of-the-art techniques used in computer vision.

Figure 1. Left: The research ground vehicle Shrimp traversing between rows at an apple orchard, capturing tree image data. Location of apples manually illustrated in the field of view of the camera. Right: Satellite view of the 0.5-ha orchard test block.

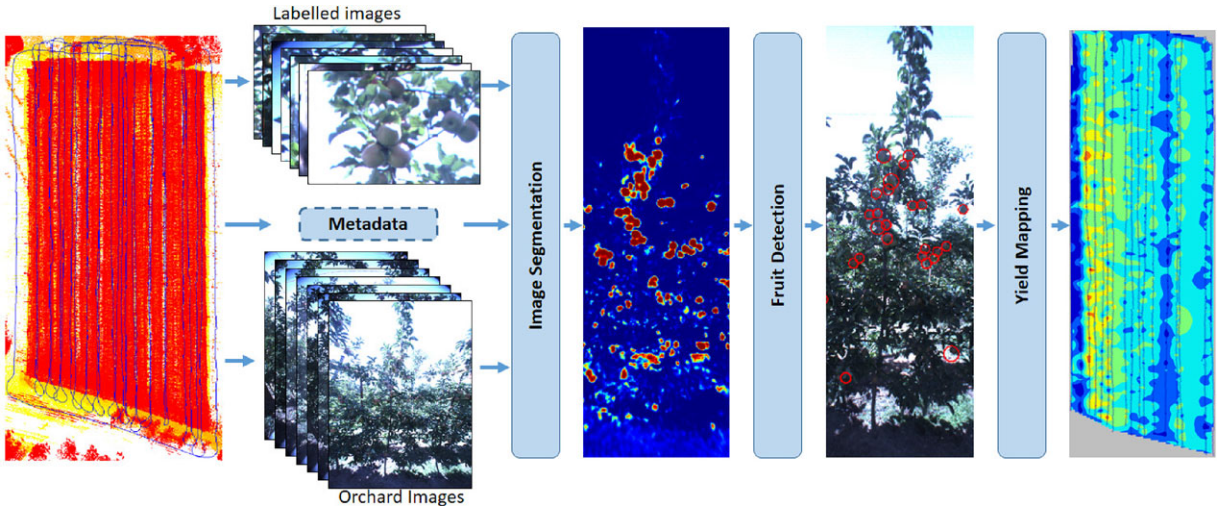


Figure 2. The image segmentation, fruit detection, and yield mapping pipeline. An image segmentation model is trained using labeled images extracted sparsely from the farm. This is used to segment the dense image data captured by a UGV. Detection algorithms are applied to the segmentation output to identify individual apples and accumulate the counts over the farm. The result is a dense farm yield map, which can be calibrated with ground-truth counts to provide a yield estimate.

Image processing at orchards spans a large variety of fruits, such as grapes (Font et al., 2015; Nuske et al., 2014), mangoes (Chhabra, Gupta, Mehrotra, & Reel, 2012; Payne et al., 2014), apples (Hung, Underwood, Nieto, & Sukkarieh, 2015; Ji et al., 2012; Kim, Choi, Choi, Yoo, & Han, 2015; Linker, Cohen, & Naor, 2012; Silwal, Gongal, & Karkee, 2014; Stajnko, Rakun, & Blanke, 2009; Wang et al., 2013), citrus (Annamalai, Lee, & Burks, 2004; Li, Lee, & Hsu, 2011; Qiang, Jianrong, Bin, Lie, & Yajing, 2014; Regunathan & Lee, 2005; Sengupta & Lee, 2014), kiwifruit (Wijethunga, Samarasinghe, Kulasiri, & Woodhead, 2009), and peaches (Kurtulmus, Lee, & Vardar, 2014). Fruit classification is

generally performed by transforming image regions into discriminative feature spaces and using a trained classifier to associate them to either fruit regions or background objects, such as foliage, branches, and ground. If conducted densely, image regions are contextual windows neighboring every pixel in the image, and the output is a densely segmented image. Postprocessing techniques can then be applied to differentiate individual objects of interest. A detection-specific approach, on the other hand, reduces the region search space by initially performing key-point detection. Here, interesting image regions (possible fruit candidates) are first extracted using manually tuned constraints designed for the particular data set. This is followed by feature extraction and classification as before.

Fruit detection through key-point extraction and classification is often applied over vineyards and orchards. For example, Nuske et al. (2014) exploits radial symmetries in the specular reflection of the individual berries to extract key-points, which are then classified as berries or not-berries. Using key-points allows distinct grape-berries to be identified, which is important to extract measurements that are invariant to the stage of the berry development. The detected regions are then used for yield estimation and prediction, in which grape bunch models are designed to convert image detections to true farm grape counts. To detect citrus fruit, Sengupta and Lee (2014) first use circular Hough transforms (CHT) to extract key-points. Alternatively, Liu, Whitty, and Cossell (2015) and Song et al. (2014) use simple color classifiers for key-point extraction for grape bunches and peppers, respectively. For fruit detection, image patches are extracted around each key-point and a combination of color and texture filters are computed. The patches can then be classified as fruit or not-fruit using a trained classifier, such as a support vector machine (SVM) or a randomized KD-forest.

Image segmentation, on the other hand, returns a rich likelihood map of the fruits, onto which a threshold can be applied to obtain a binary fruit mask detailing regions of the image that contains fruit (Linker et al., 2012; Payne et al., 2014; Sa, McCool, Lehnert, & Perez, 2015; Yamamoto, Guo, Yoshioka, & Ninomiya, 2014). Payne et al. (2014) designs a set of heuristic measures based on local colors and textures to classify individual pixels as mangoes or non-mangoes. Blob extraction was done on the resultant binary mask to identify individual mangoes. Linker et al. (2012) incorporates further postprocessing for apple detection where in individual blobs are expanded, segmented, and combined to manage occluded fruit and fruit clusters. Stajnko et al. (2009) instead uses shape analysis and template matching to extract circular apples from the segmented image. Yamamoto et al. (2014) implements a second classification component on tomato blobs extracted via image segmentation to remove any background detections.

Typically, orchard image data are subject to highly variable illumination conditions, shadowing effects, fruits/crops of different shapes and sizes, captured over different seasons, and so forth (Payne & Walsh, 2014), which makes classification a challenging task. To simplify and minimize the variations in the data, one can enforce constraints on the environment or the data gathering operation. For example, pepper detection in Song et al. (2014) is conducted in a greenhouse with controlled illumination conditions. Equivalently, in Nuske et al. (2014), Payne et al. (2014), and Font et al. (2015) the data are captured at night using strobes, which significantly restricts the illumination variance. However, in orchards, it is generally more practical to conduct large-scale experiments under natural daylight conditions, and for commercial applications simple hardware such as cameras can be easily incorporated onto tractors, which operate more frequently during the day.[1] Therefore, image classification under natural illumination conditions is an open and important problem.

As stated in the previous section, hand-engineered feature encoding often restricts methods to particular fruits/data sets as they are designed to capture data-specific interclass variations (i.e., differences between trees, leaves, and fruits), while being invariant to the intraclass variations. Although the methods stated above have produced promising performance over the respective fruits/data sets, they are distinct and ad hoc, seldom replicated, and often disconnected from progress in the general computer vision literature (Kapach et al., 2012). For widespread application, it would be more efficient to have a unified image processing approach compatible with different fruits or capturing configurations.

A general-purpose adaptive feature learning algorithm is therefore desirable, as proposed in Hung et al. (2013). Here, a pixel-wise image segmentation framework was presented, that utilizes a multiscale feature learning architecture to learn relevant feature transformations for pixel-level classification. The proposed approach was shown to outperform classification using hand-engineered features. In addition, the same architecture has been used for different data sets such as almonds (Hung et al., 2013), apples (Bargoti and Underwood, 2015, 2016; Hung et al., 2015), and tree trunks (Bargoti, Underwood, Nieto, & Sukkarieh, 2015) without any changes to the image segmentation pipeline. However, even though such a feature learning approach avoids the need for manually designed features, there are opportunities for significant improvements, particularly to address the reliability in adverse illumination conditions, such as underexposed and overexposed images, where in performance was observed to deteriorate.

[1]Night time tractor operations are also common for some fruits at some times of the year, but day/night operation will allow most flexibility for adoption.

There have been major developments in the state-of-the-art methods for image segmentation and object detection outside the agrovision literature. With the advancements of parallel computing using GPUs, deeper neural network architectures, which host a significantly larger number of model parameters, are showing potential in capturing large variability in data (Krizhevsky et al., 2012). For example, Ning et al. (2005); Ciresan, Giusti, Gambardella, and Schmidhuber (2012); Pinheiro and Collobert (2013); and Ganin and Lempitsky (2014) use multi-layered CNNs for image segmentation, in which individual patches representing contextual regions around pixels are densely classified in an image. More recently, CNNs have been shown to yield improved segmentation performance when a spatial prior on the classes is available. Brust et al. (2015) performed road image segmentation while incorporating the pixel position to help the classifier learn that road pixels are predominantly found near the bottom half of images. Finally, CNNs have also been applied for direct object detection (Girshick, Donahue, Darrell, & Malik, 2016; Ren, Girshick, & Sun, 2015), and have achieved state-of-the-art object detection accuracy for multiple computer vision data sets.

In this paper, we evaluate the performance of a previously benchmarked multiscale MLP architecture for image segmentation in agrovision (Hung et al., 2013), including an extension with metadata, which we have shown improves performance significantly in Bargoti and Underwood (2015, 2016). We further extend this study with comparison against CNNs, with and without the addition of metadata, showing improved pixel classification performance. We evaluate the utility of improved image segmentation toward fruit detection and yield estimation. With this, we also shorten the gap between image processing techniques used in agrovision to the current work in computer vision literature, which is a limitation addressed in the literature survey conducted in Kapach et al. (2012).

3. IMAGE SEGMENTATION

Image segmentation is the task of transforming individual pixels in an image into class labels. In this paper, we present multiple image segmentation architectures for the binary classification of orchard image data into fruit/non-fruit classes. These include a multiscale MLP and a deep CNN architecture. We then extend these neural network architectures with the inclusion of orchard metadata. All network training is done at the pixel level; however, inference is performed over the whole image, resulting in a dense probabilistic output of the fruit and non-fruit classes. This can be used to obtain a binary fruit mask and subsequently to perform fruit detection or yield estimation as detailed in Section 6.

3.1. Multiscale Multilayered Perceptron

Given the success of the image segmentation framework in Hung et al. (2013) for different fruit types, the reference segmentation architecture in this paper is based on a multi-scale multilayered perceptron (which we denote as ms-MLP). The classifier takes as input a contextual window around individual pixels from the raw RGB image, with the windows sampled at different image scales. The data are propagated through multiple fully connected layers and the output of the classifier is a probability of a given pixel belonging to the class fruit/non-fruit. The multiscale patch representation of each pixel provides scale invariance for classification and allows us to capture local variations at different scales, such as edges between fruits and leaves and between the trees and the skyline, while keeping the input dimensions low.

A three layer ms-MLP architecture is illustrated in Figure 3. Given an image I_k, we are interested in finding the label of each pixel at location (i, j) in the image. The input to the network are contextual patches $I^s_{i,j,k}$ surrounding the pixel at location (i, j) in the k^{th} image over scales $s \in \{1, \ldots, S\}$. The image patches are initially forward propagated into the first hidden layer using nonlinear sigmoid transformations. The activation outputs are then concatenated over the different scales:

$$H_1 = \bigcup_{s=1}^{S} \sigma(W_1^s I^s_{i,j,k} + b_1^s), \tag{1}$$

where, $\sigma(z) = 1/(1 + e^{-z})$ is the sigmoid activation function. The weights for the individual scales W^s are a set of linear filters/dictionaries transforming the input data into a more discriminative space. As done in Farabet et al. (2013), input patches from each scale are treated independently. The concatenated first layer output H_1 is then propagated through subsequent densely connected layers:

$$H_m = \sigma(W_m H_{m-1} + b_m) \tag{2}$$

for $m = \{2, \ldots, M\}$. The final layer is propagated through a softmax regression layer to obtain a class probability for the pixel (i, j) belonging to the fruit or non-fruit class.

$$P(y_{i,j} = c | H_{M-1}) = \frac{e^{W_M^c H_{M-1}}}{\sum_{l=1}^{2} e^{W_M^l H_{M-1}}}, \tag{3}$$

where W_M^l are the final layers weights corresponding to class l. All parameters $(\mathbf{W}, \mathbf{b})$ of the network are learned in an end-to-end supervised way, by minimizing a cross-entropy loss function. An L_2 regularization penalty term is used to minimize overfitting. Optimization is done with stochastic gradient descent (SGD) using momentum and a linearly decaying learning rate.

The literature (and our own experimentations) has shown that unsupervised pretraining boosts classification performance for fully connected networks as they learn

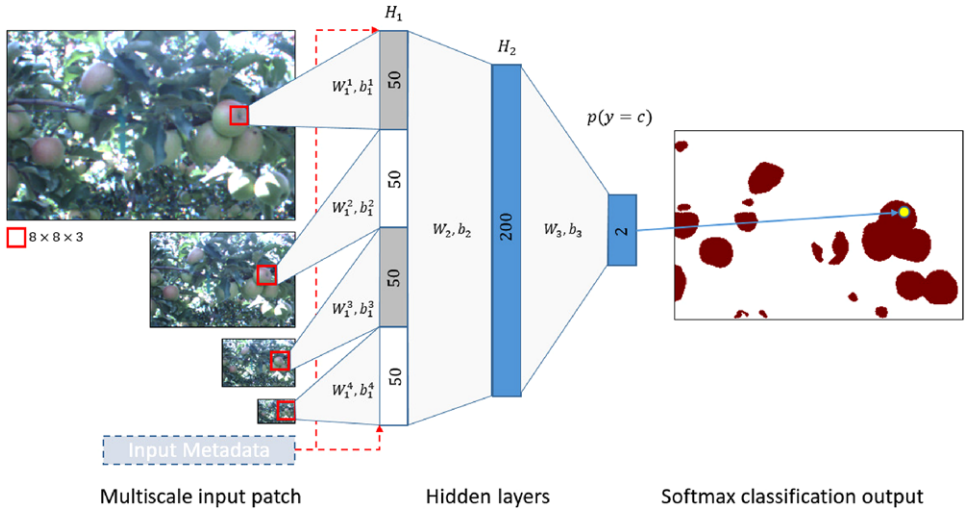


Figure 3. The multiscale Multilayered perceptron architecture with three layers. Centered at individual pixels, a contextual window is extracted over multiple image scales, and the raw data are forward-propagated through multiple hidden layers. The output softmax layer returns the probability of that pixel belonging to a given class. The optimal layer sizes are annotated on the individual layers above. Metadata are appended to the input layer, and their associated weights to the hidden layer are learned during the training phase (red dashed line).

generalized features, which are useful for initializing the supervised training (Erhan et al., 2010). Each set of the first layer filters $W_1^s \ \forall s \in \{1, \ldots, S\}$ are therefore prelearned using a held-out data set using a De-noising auto encoder (DAE) with a sparsity penalty (see Hung et al., 2013, for details). The learned weights consist of a combination of edge and color filters. The deeper layers are initialized by using the sparse initialization scheme proposed in Martens (2010).

3.2. Convolution Neural Networks

CNNs are feed-forward neural networks, which concatenate several types of forward-propagating layers, with convolutional layers playing a key role. Like the MLP, the network computes the probability of a pixel being a fruit or non-fruit, using as input the image intensities of a square window centered on the pixel itself. However, instead of using small multiscale patches, CNNs can take as input larger-/high-resolution patches covering the same contextual region. This is due to the ability of the CNNs to share smaller-scale filters in each layer, minimizing the number of model parameters.

A typical CNN consists of a succession of convolutional, max-pooling, and fully connected layers as shown in Figure 4. Each convolutional layer performs a two-dimensional convolution of its input maps with a square filter. This is followed by a nonlinear activation function using the Rectified Linear Unit (ReLU) and a max-pooling subsampling layer as used in Krizhevsky et al. (2012). The output of each block is then:

$$H_m = pool(ReLU(W_m H_{m-1} + b_m)), \tag{4}$$

where $H_0 = I_{i,j,k}$. The output from each layer is a feature map with successive layers covering increasingly large receptive fields, while encoding more complex variations in the input data. The convolution and pooling layers are followed by fully connected layers as per Eq. (2) but using the ReLU activation function instead of the sigmoid. Using a softmax activation function for the last layer (Eq. (3)) we obtain the class probability of the pixel/image patch. The network is trained in a similar fashion to the ms-MLP; however, unsupervised pretraining is not required with CNNs. Instead, dropout, an additional regularization penalty on the fully connected layers, is introduced during training.

3.3. Adding Metadata

Metadata corresponding to individual pixels can be incorporated to image segmentation architectures by appending the information to one of the fully connected layers. We can define each set of metadata for a given input instance $I_{i,j,k}$, by $d^n_{i,j,k} \ \forall n \in N$, where N is the set of different meta-parameters, for example, sun position, tree type, and so forth. The different metadata can then be concatenated together.

$$D_{i,j,k} = \bigcup_{n=1}^{N} d^n_{i,j,k}. \tag{5}$$

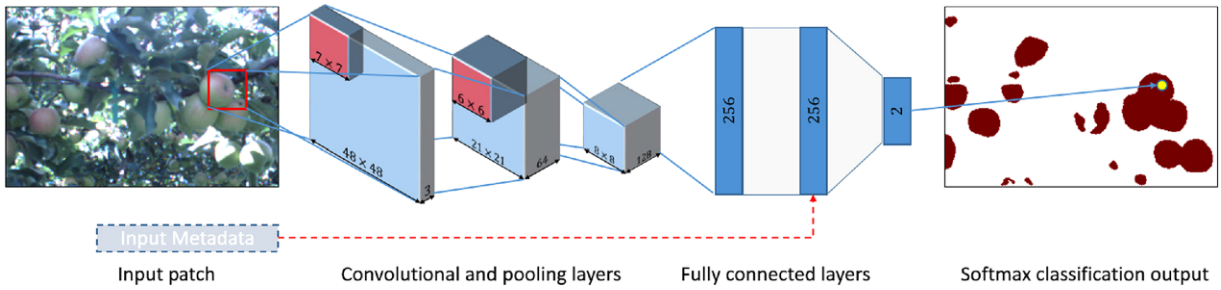


Figure 4. The patch based convolutional neural network architecture with two convolution+pooling layers and two fully connected layers. Each pixel is defined by a large contextual region around it (48 × 48 color pixels) and is propagated through the CNN. The output softmax layer returns the probability of that pixel belonging to a given class. The figure shows the optimal configuration for this data set. Metadata are appended to the fully connected layer (red dashed line).

For the ms-MLP, the propagation to the first hidden layer is then given by:

$$H_1 = \bigcup_{s=1}^{S} \sigma(W_1^s I_{i,j,k}^s + U^s D_{i,j,k} + b_1^s), \tag{6}$$

where U^s are the set of weights learned for each scale for the scale independent metadata input D. The metadata have an additive effect on the biases b_1^s, where each metadata component shifts the responses from individual filters learned over the same layer. The metadata can equally be appended to deeper layers of the network or first propagated through a disconnected hidden layer; however, for the ms-MLP, our experiments found best performance when raw metadata were merged with the input layer.

On the other hand, for the CNN architecture the metadata are added as inputs to a single fully connected layer (see Figure 4) as:

$$H_{m_{fc}} = ReLU(W_{m_{fc}} H_{m_{fc}-1} + U D + b_{m_{fc}}). \tag{7}$$

Both networks can be trained using the same back-propagation algorithm as before. The computational expense is linear to the number of nodes in a fully connected layer. Because the dimension of the metadata are significantly smaller than the size of the layers, the additional computational expense of this configuration is negligible.

3.4. Scene Inference

During prediction, given a new image patch $I_{i,j,k}$, we can use feed-forward propagation to predict the probability $p(y_{i,j} = fruit)$ for the pixel (i, j) to belong to the fruit class. In practice, for both the ms-MLP and the CNN architecture, this can be performed densely via sliding windows to obtain a segmented image, but this approach is highly inefficient. Instead, a more computationally efficient approach is to redesign the learned models as fully convolutional operations enabling fast-dense prediction over arbitrary-sized inputs.

For the ms-MLP, we transform the first layer weights W_q^s into patch-wise kernels, which are convolved over the test images sampled at different scales $s \in \{1, \ldots, S\}$. The filter responses from the smaller scales are up-sampled using linear interpolation to the same scale as the original image. The different scales are then concatenated and forward-propagated through the subsequent fully connected layers. Zero-padding is applied to the input image to manage border effects from the convolution operations.

For the CNN architecture, the convolution and pooling operations are easily scalable to larger images than the training input patches. The fully connected layers can also be seen as 1×1 convolutions in a spatial setting, making the entire CNN simply a sequence of convolutions, activations, and pooling operations.

However, the output dimensions of CNNs are typically reduced due to the pooling layers, which subsample the data in order to keep the filters small and the computational requirements reasonable. As a result, the segmentation output from a network reduces the size of the input by a factor equal to the pixel stride of the pooling operations. For example, if the network includes two 2×2 pooling layers with a stride of 2 pixels each, only 1 in every 4 pixels of the input image will be labeled. Some work with CNNs (Farabet, Couprie, Najman, & LeCun 2013; Ning et al., 2005) upscale the label output by the subsampling factor to get to the input image size.

To obtain a full resolution segmentation output without interpolation, input shifting and output interlacing is used. If the network down-samples by a factor of f, we shift the input image (after zero padding) in the x and y directions by $\Delta_x, \Delta_y \in \{0, \ldots, f-1\}$. This results in f^2 inputs, each of which are passed through the CNN architecture producing down-sampled segmentation outputs. These are then interlaced such that the fine-resolution predictions correspond to the pixels at the centers of their receptive fields (Long, Shelhamer, & Darrell, 2015; Pinheiro & Collobert, 2013; Sermanet et al., 2013).

(a) Kanzi (red apples) (b) Pink Lady (pink-green apples) (c) Pink Lady (green apples)

Figure 5. Apple varieties found within the scanned block at the apple orchard. Their appearance ranges from bright red (Kanzi) to pink-green and leafy green (variants of Pink Lady). Sample images taken using a hand held DSLR camera. Colours best viewed online.

4. EXPERIMENTAL SETUP

The orchard image segmentation architectures were tested at an apple orchard in Victoria, Australia (Figure 1). Data were gathered over a 0.5-ha block, which hosts a modern Güttingen V-trellis structure, where in the crops are planted over pairs of trellis faces arranged to form a V shape. Built with support poles and wires along the rows, these structures provide better support of the tree limbs. They also allow for more sunlight for the fruit and easier harvesting. The trees hosted different apple varieties, including Kanzi and Pink Lady. The fruit color ranged from bright red to a mixture of pink-green and completely green as shown in Figure 5. The experiments were conducted 1 week before harvest (March 2013) in which the apple diameter ranged from 70 to 76 mm.

4.1. Data Capturing

The testing platform *Shrimp* is a perception research ground vehicle, built at the Australian Centre for Field Robotics at The University of Sydney (Figure 1). Among an array of sensor types, it is equipped with a Point Grey Ladybug3 spherical digital video camera containing six 2MP cameras oriented to capture a complete 360-deg panoramic view.

To collect the data set, the vehicle was teleoperated between 17 orchard rows at 1.5 to 2 ms^{-1}. Two of the rows hosted the Kanzi apple variety, whereas the rest were Pink Lady. The operator walked behind the vehicle, manually guiding it along the centerline of the rows, which were planted 4 m apart. Images of size 1232×1616 were captured at 5 Hz through the camera facing the trellis structure. The field of view of the camera was critical in its selection as it was able to capture the 4-m-tall trees from a distance of 2 m as illustrated in Figure 1. The data were collected under natural illumination conditions, and the particular times of day, sun angles, and weather conditions were not specifically selected to optimize illumination. Finally, an onboard Novatel SPAN global positioning inertial navigation system (GPS/INS) provided estimates of the vehicle position and pose, which was used to localize each image. The vehicle trajectory is illustrated on the left in Figure 2.

We have also experimented with autonomous centerline following using LiDAR for subsequent datasets, however, manual control yields data more similar to what would be expected from a manually driven tractor, which is one likely adoption strategy for the technology. Fully autonomous row following is likely to yield more consistent imagery, which improve image classification performance.

4.2. Image Data Set

In total the image data set consists of over 8,000 images of 1232×1616 pixels each. As it would be impractical to manually label this much data, a subset was collected by random subsampling. Each image was divided into 32 subimages with 308×202 pixels and a fixed number of subimages were randomly sampled from each row to maximize the diversity of the data. In total, 1,100 subimages were collected and manually annotated with binary pixel-level labels for the fruit and non-fruit class. Subsectioning the image into smaller sections makes the manual labeling process easier and results in greater spatial variance within the data set. Examples of the reduced data set and their associated pixel-wise labels are shown in Figure 6. Apples in the images varied in size from 25 to 50 pixels in diameter. This variation is attributed to the distance to the imaging platform and viewing angle.

4.3. Segmentation Architecture

For training the ms-MLP, a similar architecture to the one presented in Hung et al. (2015) was used. Individual instances/patches of size $[8 \times 8 \times 3]$ were randomly sampled over image scales: $[1, 1/2, 1/4, 1/8]$. A separate natural scene data set (Brown & Susstrunk, 2011) was used to initialize the first layer filters with a DAE. To remove pixel-level correlations on the image patches and force unit variance, ZCA whitening was used for preprocessing (as done in Hung et al., 2015). For the CNN, lacking equivalent implementation in agrovision, we build a network around previous

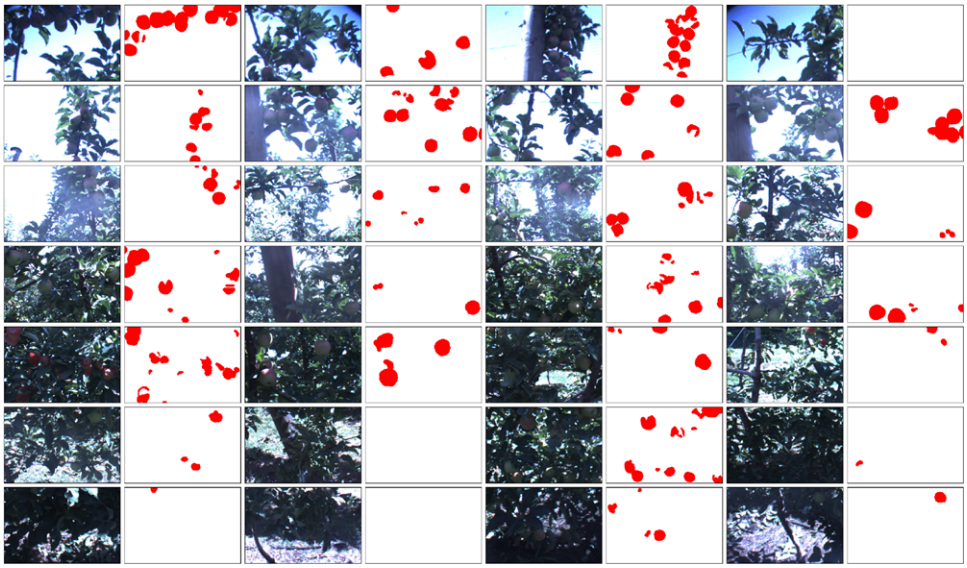


Figure 6. Sample sub-images and associated ground truth pixel labels, randomly extracted from the orchard dataset for training. Images are vertically stacked in the figure according to their true height in the original data.

Figure 7. Image patches (48×48) sampled from the labelled data to be used as input to the CNN architecture. Patches in the first and second rows are labelled 'not-apple' and 'apple' respectively.

work done for patch-wise classification over electron microscopy images (Ciresan et al., 2012) and urban scenes Brust et al. (2015). Pixel-centered, single-scale patches are extracted from the labeled data set, while ensuring that they are large enough to identify/contain the fruit. Examples of 48×48 patches are shown in Figure 7 with the corresponding labels denoting the class of the centre pixel.

For all architectures, balanced sampling was done over the two classes due to the large class imbalance in the data set. Hyperparameters such as the learning rate, learning rate decay factor, initial momentum, L_2 penalty, number of epochs, architecture width and depth were optimized over a held out validation set. The architectures were developed in Python using the open-source deep learning library, Pylearn2 (Goodfellow & Warde-Farley, 2013).

The metadata used for this data set included a combination of pixel positions, orchard row numbers, and the sun's position relative to the vehicle body frame. The sun's azimuth and elevation were calculated using the time of day and vehicle's pose and geographical position. The sun's elevation remained fairly constant over this data set; therefore, only the azimuth was considered. These properties were chosen as they were qualitatively observed to correlate with some of the intraclass variation in the data. For example, Figure 6 illustrates the variation in appearance as a function of the vertical position (pixel height) in the captured image, with the image regions higher up on the tree appearing brighter and hosting apples viewed from a different pose. Row number data were introduced because farm operations were typically conducted on a row by row basis, for example, certain rows were planted with certain apple varieties.

The combined metadata information (Eq. (5)) was incorporated within the ms-MLP and the CNN as shown by the dashed lines in Figures 3 and 4, respectively. Each metaparameter can be formatted as either a single continuous unit or multiple discrete units. For example, the height of a pixel on the image (in the range $[0 - 1616]$) can be normalized and represented by a single number in the range $[0 - 1]$ or represented as a one-hot encoded vector. In the latter, image height is discretized into a fixed number of bins with a single active unit denoting the pixel height (as done in Rao

et al., 2014). However, a single-unit representation restricts the number of parameters available to represent the metadata and prevents us from learning a different set of biases of individual filters responses over the variations within each meta-parameter. A one-hot encoding was therefore used, where a few channel sizes were tested and ultimately eight channels were used for each continuous input data: pixel i, j positions (p_i, p_j) and the sun azimuth angle (s_ψ). The row numbers (r_n) could be encoded directly as one-hot vectors. To test how the learning algorithm would cope with irrelevant metadata information, training of the ms-MLP architecture was also conducted with the inclusion of uniformly distributed random noise as information. No further hyperparameter optimization was performed following the inclusion of the metadata.

5. SEGMENTATION RESULTS

To evaluate the image segmentation performance, the labeled data set (1,100 images) was randomly divided into an $80-10-10$ split of training, validation, and testing images. Single-scale and multiscale patches were sampled (balanced between the classes) from the set of training images to train the CNN and ms-MLP architectures.[2] In this section, we evaluate the performance of the different architectures using the F1-score evaluated over instances randomly sampled from the test image set. The data are naturally imbalanced, with substantially more background than apple pixel instances. Hence, we report on the apple F1-score specifically, to avoid reporting an artificially high classification performance that is dominated by the background. The class threshold for the optimal F1-score was evaluated over the instances sampled from the validation set. In all tests, mean and one-standard-deviation classification results were obtained over 10 iterations, while shuffling the $80-10-10$ split. The last part of this section utilized the patch based models for whole-image inference, presenting qualitative segmentation results over images from the test set.

5.1. Multiscale Multilayered Perceptron

For baseline comparison, the original architecture from Hung et al. (2015), a two-layer MLP with 200 hidden units was trained with 200,000 training instances, which we denote as ms-MLP-2. Different sources of metadata were then added to this configuration, and the MLP retrained each time. Training each MLP network until convergence over the validation set ($\sim$ 30 epochs) took roughly 5 min on a GPU-enabled desktop. The classification results are shown in Table I.

There is a clear improvement in classification results with the inclusion of all of the metadata increasing the F1-score the most by 6.2% ($F1 : 0.683 \rightarrow 0.725$). On it's own, p_i (the height within the original image) was found to be the most important individual metadata parameter. This agreed with qualitative analysis of the metadata, in which the greatest appearance variations were observed with changes in pixel height/height on the tree due to illumination changes, as shown in Figure 6. In addition, if some metadata were hypothesized to have no direct relationship with the extrinsic variations in data (such as random noise or p_j), the classification results were no worse. The lack of information from the sun position, s_ψ, is possibly because in this data set, the sun azimuth angle was either $+90$ deg or -90 deg, and, therefore, did not cause much variation on its own to captured images. Interestingly, the learned weights U from Eq. (6) corresponding to these metadata had a much weaker response, meaning that during the learning phase, the algorithm discovers which inputs are not relevant to the classification task. In addition, this analysis guides us toward the most important factors of variation, which in turn might be physically controllable, to improve subsequent data collection.

Table I. Pixel-wise fruit classification results with the original ms-MLP architecture as presented in Hung et al. (2015). Combinations of metadata are added to the network, and the corresponding classification results listed as the absolute F1-score and as difference from the default method.

NN Architecture	Metadata	Pixel F1-Score
ms-MLP-2	None	0.683 ± 0.015
	Noise	$0.683\ (+0.000 \pm 0.002)$
	p_i	$0.715\ (+0.032 \pm 0.005)$
	p_j	$0.683\ (+0.000 \pm 0.002)$
	r_n	$0.694\ (+0.011 \pm 0.004)$
	s_ψ	$0.684\ (+0.001 \pm 0.003)$
	p_i, r_n	$0.721\ (+0.038 \pm 0.005)$
	p_i, p_j, r_n, s_ψ	$0.725\ (+0.042 \pm 0.005)$

Pixel position (p_i, p_j), row number (r_n), sun azimuth (s_ψ).

5.1.1. Optimal ms-MLP architecture

We extend from Hung et al. (2015) by searching over different combinations of depth and width for the MLP architecture. A grid search was performed over networks with a depth of two to five layers and with varying widths of 200 to 1000 units per layer. The optimal fruit classification performance was obtained with a three-layer MLP with each hidden layer containing 200 hidden units. We denote this network as ms-MLP-3*, and it is illustrated in Figure 3. As before, metadata were then added within this network at the input layer. During the optimization phase, we experimented with adding the metadata to deeper layers instead of the input layer and propagating it through independent hidden layers before merging with the image data

[2]Because of spatial correlation in appearance, dense sampling of training instances is not necessary.

Table II. Pixel-wise fruit classification using different ms-MLP architectures. The networks used are ms-MLP-2 from Hung et al. (2015) and ms-MLP-3*, optimized over width and depth. Metadata results are listed for both networks as the absolute F1-score and as change from the configuration not using metadata.

NN Architecture	Metadata	Pixel F1-score
ms-MLP-2	None	0.683 ± 0.015
	p_i, p_j, r_n, s_ψ	$0.725\ (+0.042 \pm 0.005)$
ms-MLP-3*	None	0.728 ± 0.016
	p_i, p_j, r_n, s_ψ	$0.751\ (+0.023 \pm 0.008)$

Pixel position (p_i, p_j), row number (r_n), sun azimuth (s_ψ).

as done in Rao et al. (2014). However, the best results were obtained when the raw metadata were added alongside the input multiscale image data. The segmentation results are reported in Table II. Going from the ms-MLP-2 to the optimized ms-MLP-3* network, the F1-score increases from 0.683 to 0.728. With the inclusion of metadata (using all meta-parameters), there is a statistically significant improvement in the F1-score by +0.023 to 0.751. The magnitude of the improvement is smaller, possibly due to the higher model complexity automatically capturing some of the appearance variations and class distributions in the data.

5.1.2. Varying training size

As mentioned in Section 1, a classifier is more likely to be invariant to intraclass appearance variations given a high model complexity and enough training data. In the previous section, we showed this to be true when the model complexity was increased. The training size for the patch-based segmentation architectures is the number of pixels (neighborhood patches) extracted from the set of labeled images for training. A larger number of training instances would typically result in better classification performance but results in a linear increase in algorithm run-time during training. Using the ms-MLP-3* configuration, we evaluate the effects of training size on the classification results, with and without the inclusion of metadata. The classification F1-score (shown in Figure 8) increases as we sample more instances from the training set, reaching convergence around 500,000 training instances. Theoretically, a total of $\sim 56 \times 10^6$ patches can be sampled from the 900 training images; however, there can be significant overlap in information due to the contextual area covered by single training instances. The addition of the metadata to the architecture results in improved classification performance in all cases, having a greater advantage when restricted with the number of training instances. In addition, for a fixed F1-score, the inclusion of metadata allows the same result with only a small fraction of the training instances (e.g., the same

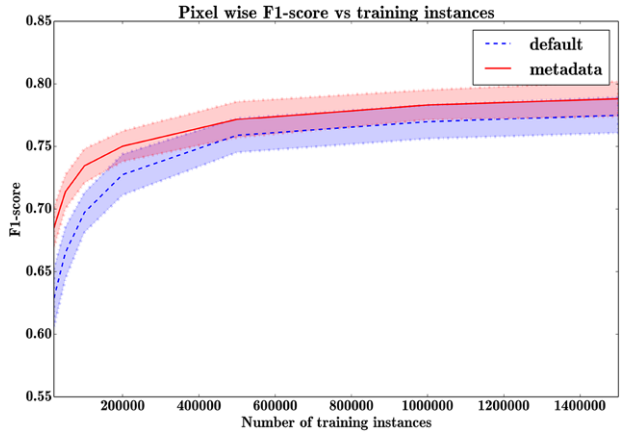


Figure 8. Classification results against number of training instances with and without metadata over the two segmentation architectures. The shaded regions illustrate one standard deviation in the results.

F1-score can be achieved with the metadata model while using just 200,000 training instances, compared to 500,000 instances when no metadata are used). As the classification performance converges, the performance improvement due to metadata converges to approximately $+0.013 \pm 0.007$.

5.2. Convolutional Neural Network

For the optimal CNN configuration, we searched over different depths, widths, and input sizes while monitoring the F1-score over the held-out validation set. We tested a range of input sizes, covering a $\{32 \times 32\}$, $\{48 \times 48\}$, $\{64 \times 64\}$ region around a pixel. For the convolutional layers, we tested a depth of two to three layers, each containing a convolution, activation, and pooling operation. These were followed by two fully connected layers as typically done in computer vision literature. The optimal architecture was found by doing a coarse grid search and is denoted in this paper as CNN* (illustrated in Figure 4). The first convolutional layer filters the $48 \times 48 \times 3$ input image patch with 64 kernels of size $7 \times 7 \times 3$ and is followed by a ReLU activation and a max-pooling operation over a 2×2 patch with a stride of 2 pixels. The second convolutional layer filters the subsampled $21 \times 21 \times 64$ input with 128 kernels of size $6 \times 6 \times 64$ followed by a similar activation and pooling layer. The resulting $8 \times 8 \times 128$ output is propagated through two fully connected layer with 256 nodes each and finally a softmax layer.

Training the CNN architecture until convergence over the validation set ($\sim$ 50 epochs) took roughly 3 hr while using 200,000 training instances. The CNN patch classification results are shown in Table III. Without metadata, the F1-score has increased from 0.728 (ms-MLP-3*) to 0.791 with the CNN* architecture.

Table III. Pixel-wise fruit classification results with the CNN* architecture using the default method and with the addition of Metadata. Metadata results listed as the difference in F1-score from the default method.

NN Architecture	Metadata	Pixel F1-score
CNN*	None	0.791 ± 0.011
	p_i, p_j, r_n, s_ψ	$0.797\ (+0.006 \pm 0.008)$

Pixel position (p_i, p_j), row number (r_n), sun azimuth (s_ψ).

As before, we add the metadata to the classification architecture, which was optimally placed alongside the first fully connected layer. In contrast to what was observed with the ms-MLP architectures, the increase in F1-score with the inclusion of the metadata to CNN* is minimal at +0.006 (Table III). In addition, in practice we observed that in some configurations (e.g., if the metadata were added to the second fully connected layer), the performance was slightly worse.

5.3. Whole-Image Segmentation

To acquire the input for consequent fruit detection and yield estimation and to qualitatively analyze the segmentation results, the trained ms-MLP and CNN architectures were utilized to segment whole images using the techniques discussed in Section 3.4. The image inference was conducted using the ms-MLP-3* architecture with and without metadata and with the CNN* architecture without metadata. Inclusion of metadata with the CNN architecture was not tested here due to the minimal classification gain it provided.

Whole-image segmentation using the ms-MLP architecture on the 308×202 images from the test image set took 0.60 s/image. The largest computational overhead is during bilinear up-sampling of the multiscale first layer outputs. The resultant binary mask for a few images is shown in Columns 3 and 4 in Figure 9, with and without metadata. The metadata configuration is picking a greater region of apples and even some new apples that the no-metadata configuration does not detect (e.g., center right apple in the third row and center left apple in the fourth row). In addition, over certain regions it has a much lower false positive rate as seen in the top row example. However, there are some instances where adding metadata lowers precision as seen by the false detection of the trunk in the third row.

The CNN-based image segmentation is performed using the shift and stitch method, which took 0.24 s/image on the 308×202 images. The resultant binary mask is shown in the last column in Figure 9. As expected from the performance metrics, we see the CNN* network outperforming the others in both segmentation precision and recall. For example, even under the challenging illumination example, the third row trunk is not misclassified. The improved performance could be attributed to the higher-resolution input and the greater number of available parameters to capture the data distribution. Finally, we observe a smoother results with the CNN, which is due to the pooling layers enforcing some translation invariance in the data.

5.4. Network Comparison

To compare the differences between the concept of an apple between the two segmentation frameworks, CNN and ms-MLP, the percentage of fruit detected by each system (i.e., the recall rate) was evaluated. We limit our comparison to CNN* and ms-MLP-3*, both without metadata, and apply morphological erosion and dilation to smooth any noisy segmentation output. The set of fruit ground-truth pixels were partitioned into ones that were detected by both networks, just one of the networks, and neither of the networks. For each of these partitions, the mean per-pixel classification probability by each network was evaluated. The corresponding results, averaged over 10 folds, are shown in Figure 10. Of all the fruit pixels, 66.3% were classified correctly by both the classifiers. The CNN uniquely identified a further 12% of the pixels with a high probability of 0.95, and the ms-MLP registered these as false negatives with a probability of 0.69 (below the learned optimal threshold). On the other hand, ms-MLP uniquely identified 4.4% of the fruit pixels. The higher total recall rate of 78.3% for the CNN network, compared to 70.7% for the ms-MLP network, contributes to the higher F1-score reported in previous sections. For the fruit pixels that were misclassified by both networks, CNN delivers a higher mean confidence for the fruit class.

6. FRUIT DETECTION AND YIELD ESTIMATION

The output from the image segmentation methods above are class probability maps or binary masks indicating the location of individual fruit pixels in the images. In this section, we present methods to translate this to agronomically meaningful information, such as detection of individual fruits, fruit counts, and maps of fruit yield. The segmentation architectures we focus on are the ms-MLP-3* (no metadata), the ms-MLP-3* with metadata, and the CNN* (no metadata). The metadata configuration of the CNN network is not tested any further as it yielded minimal improvements in the segmentation results. Furthermore, we evaluate the fruit detection and yield estimation performance using different ground-truth data, comparing the impact of different segmentation architectures toward higher-level agricultural tasks.

6.1. Fruit Detection

The standard approach for object detection in the computer vision literature is to first perform dense prediction using sliding windows over the image, score each window

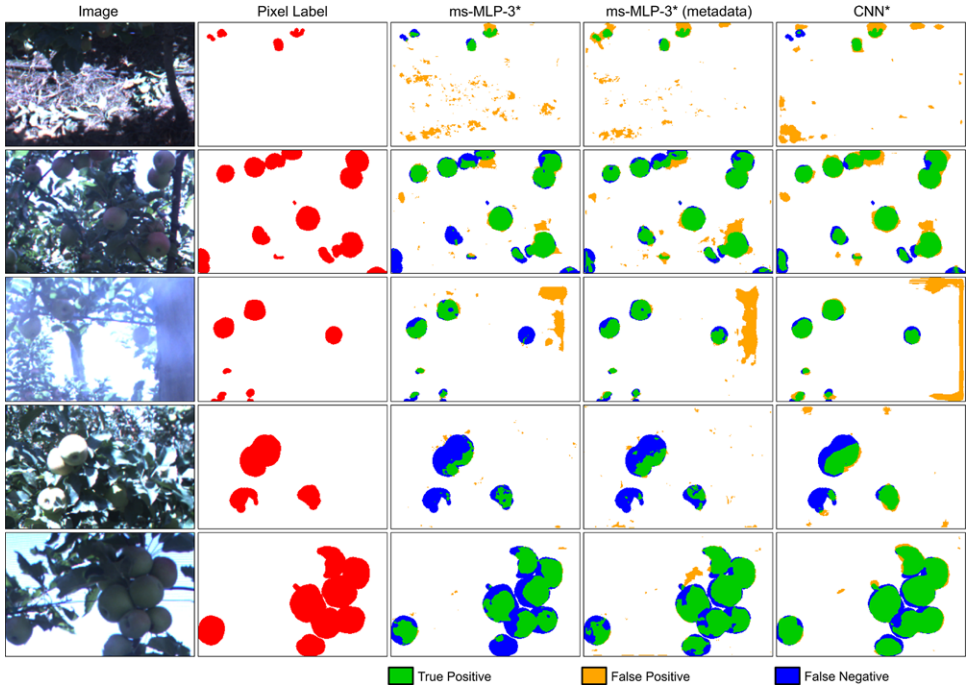
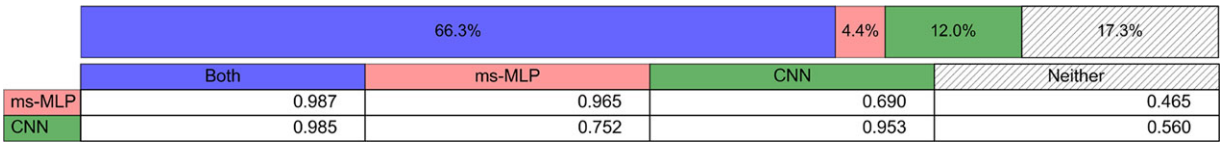


Figure 9. Image segmentation results. Five sample image sections from the test image set (first column) and their ground truth labels (second column). The third and fourth columns illustrate the segmentation output from the ms-MLP-3* architecture without and with metadata. The fifth column shows the segmentation output from the CNN* architecture. Segmentation output near the image borders are masked out during performance evaluation.



	Both	ms-MLP	CNN	Neither
ms-MLP	0.987	0.965	0.690	0.465
CNN	0.985	0.752	0.953	0.560

Figure 10. Network comparison between ms-MLP and CNN for image segmentation performance. The partitioned bar shows the percentage of true fruit pixels detection by both networks, by ms-MLP alone, by CNN alone and by neither network respectively. For the set of pixels in each of these partitions, the table below is the mean classification probability from each network.

via a trained classifier (as done in the previous section), and then remove overlapping windows via nonmaximum suppression. Training is done using image patches centered around objects of interest; therefore, during prediction the probability maps peak near the center of individual objects. However, for image segmentation, where patches are trained using the center pixel class, the probability map plateaus around fruit region (i.e., no local maxima near the center of the fruit). Therefore, for fruit detection, we aim to split the binary fruit mask, obtained from the probability map, into separate regions denoting individual fruits.

If individual fruits were spread apart over the segmented image and appeared without any occlusions, fruit detection would simply involve selecting the disjoint regions in the binary output. However, in orchards, fruits are often clustered together and appear occluded in the image data because of other fruits and foliage. A suitable fruit detection algorithm needs to be able to separate individual fruits in clusters and connect disjoint regions of the same fruit, formed due to occlusions. For this, we implement two different detection techniques, the watershed segmentation (WS) algorithm (Roerdink & Meijster, 2000) and the circular Hough transform (CHT) algorithm (Atherton & Kerbyson, 1999).

The WS algorithm is used for separating connected objects in an image in which each object is characterized by a local maximum and a contour leading to its boundary. For a segmented fruit image, this contour is evaluated by computing distances of individual fruit pixels to the nearest background pixel, resulting in a local maximum typically situated around the center of each fruit. The detection output using the WS approach is shown in the third column

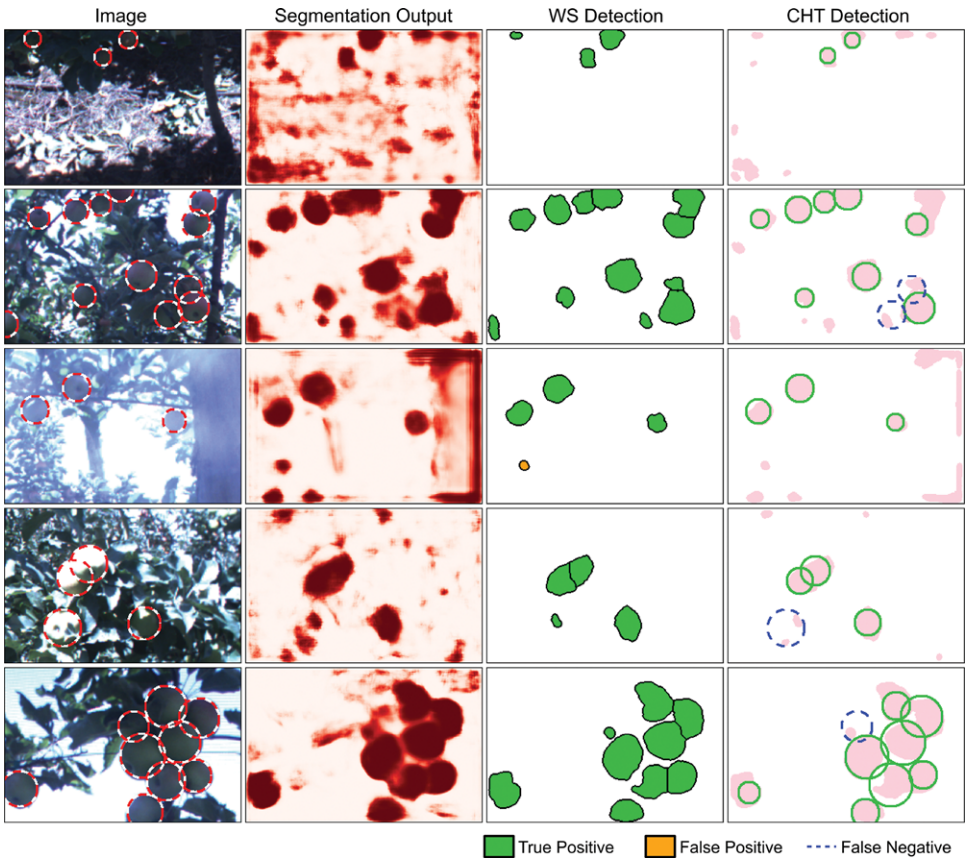


Figure 11. Apple detection results. The first column contains the orchard images with manually annotated fruit. The second column is the segmentation output from the CNN* algorithm. Individual fruit detection is done using the Watershed Segmentation and Circular Hough Transform algorithms, with the output shown in columns 3 and 4 respectively. False detections near the image boundaries are ignored during evaluation.

on Figure 11. Individual fruits are discernible even if they appear in clusters. However, the algorithm cannot merge fragments of a single fruit that become separated due to foreground occlusion or misclassification. To overcome this, we also tested CHT on the binary segmentation output. The algorithm is reliant on the circular nature of the fruits and has the added advantage of detecting partially circular regions, enabling the potential to merge disjoint fruit regions into a single detection. Example of detections using the CHT are shown in the fourth column in Figure 11. For both detection operations, the segmentation output is preprocessed with morphological erosion and dilation to enforce local consistency.

To evaluate fruit detection performance, we rely on ground-truth available at the individual fruit level. This information can be extracted from the pixel-labeled data by doing fruit detection using the approaches mentioned above (as done in Bargoti & Underwood, 2016). However, this may induce a bias in the ground-truth where in clustered or occluded fruits are not accurately represented due to errors in the detection algorithms. Therefore, in this paper, we annotate the same set of images with individual fruit labels, labeling them with a circular marker with variable radius, and storing the center and radius of each fruit.[3] Some of the annotations are shown in the left column in Figure 11. A greedy 1-nearest neighbor, one-to-one matching was performed between the ground-truth and estimated detections, with a detection denoted as true positive if it was within the annotated fruit region. The remaining detections were classified as false positives. True fruit locations not associated with any detections were denoted as false negatives. The detection F1-scores using the WS and CHT algorithms on the segmentation outputs from three different architectures are shown in Table IV. For fruit counting, we evaluate the total number of fruit detections per image (true positives

[3]The python-based annotation toolbox is available at https://github.com/acfr/pychetlabeller.git

Table IV. Individual fruit detection results (F1-score) using the watershed segmentation (WS) and circular Hough transform (CHT) detection algorithms on the image segmentation output. The segmentation architectures are the ms-MLP-3* (without and with metadata) and the CNN* (without metadata).

	WS Detection		CHT Detection	
Segmentation Architecture	F1-score	r-squared	F1-score	r-squared
ms-MLP-3* (no metadata)	0.833 ± 0.014	0.693 ± 0.041	0.799 ± 0.022	0.660 ± 0.056
ms-MLP-3* (with metadata)	0.839 ± 0.007	0.735 ± 0.051	0.810 ± 0.016	0.644 ± 0.049
CNN*	0.861 ± 0.009	0.787 ± 0.043	0.854 ± 0.013	0.749 ± 0.035

+ false positives). The counting evaluation metric used is the coefficient of determination (r^2) between the detection counts and the annotated counts over the images in the test sets. The corresponding regression results for the different configurations are also shown in Table IV. All detection hyperparameters (e.g., minimum distances between fruits, fruit size) were evaluated over the validation image set.[4]

The combination of WS fruit detection and CNN* image segmentation approach was best performing, with a detection F1-score of 0.861. With both the CNN and ms-MLP architectures, the detection performance with the CHT approach was marginally worse than WS detection. We also observed minor improvements with the ms-MLP-3* network with the added metadata; however, they are within 1-standard deviation of the mean. The regression scores follow a similar trend, with the strongest r-squared fit of 0.787 observed with CNN* and WS detection.

The CNN and ms-MLP segmentation algorithms were also compared at the fruit level as previously done at the pixel level in Section 5.4. As before, we limited our focus to the ms-MLP-3* and CNN* architectures, and use the WS detection algorithm as it was the better performing of the two detection approaches. Using the ground-truth whole-fruit annotations on the test image set, apples were identified as uniquely and/or jointly detected by the CNN and ms-MLP frameworks. For each fruit detection, a pseudo-detection probability was defined as the mean pixel probability evaluated over the ground-truth pixels contained within the annotated fruit region. The portion of true positive and false negative rates from both frameworks and their associated mean probabilities averaged over 10 folds are reported in Figure 12. In total, 82.5% of the fruits were detected by both frameworks, with the CNN having higher confidence with a detection probability of 0.92 compared to 0.83 for ms-MLP. Another 3.5% of the fruits were detected with the ms-MLP alone and 3.6% by the CNN alone. A total of 10.5% of fruits were not detected by either framework. The fruits that were detected using the CNN framework alone had a mean detection probability of 0.85. These fruits were missed by the ms-MLP framework with a much lower mean detection probability of 0.74. However, the differences in the detection probability for the fruits that were uniquely identified by the ms-MLP framework is small (0.84 for CNN and 0.86 for ms-MLP). After a closer qualitative examination of such instances, we observed numerous cases where the pixel segmentation output was accurate with the CNN framework but the WS algorithm failed to correctly separate the fruits. The net portion of detected fruits (recall rate) is similar for both algorithms; however, the CNN approach delivers higher precision with fewer false positives, resulting in the improved F1-score reported in Table IV.

6.2. Yield Estimation

Accurate image segmentation and fruit detection can enable accurate yield estimation. However, even with perfect segmentation, followed by an appropriate fruit detection algorithm, we cannot directly observe the true number of apples due to occlusions. Instead, we make the assumption that on average there is a constant ratio of visible fruits to occluded fruits, allowing us to either map variations in the yield at the farm or perform yield estimation given some calibration data. At this apple orchard, the grower provided the calibration data by counting and weighing the post-harvest produce, separately per row.[5] The data covered 15 rows, ranging from 3,000 to 12,000 apples per row.

Yield estimation was performed using the original 1232×1616 images captured densely over the farm. The images were first down-sampled to every 0.5 m along the row to minimize image overlap (and hence double counting) in subsequent frames. Any remaining overlap was avoided by manually choosing a fixed region of interest along the image centre, within which the detected fruits were counted. Fruit counts were accumulated over the images per row to give an uncalibrated row count estimate. Similar to Hung et al. (2015), yield estimation accuracy was evaluated using the r-squared correlation coefficient between the estimated row

[4]Some of the detection hyperparameters are related to the size of the fruit in the images, which will be extracted automatically in the future by evaluating the distances from the sensor to the fruit.

[5]Harvest fruit weights and counts are normally measured per orchard block. Per-row counts are typically too labor intensive for a commercial orchard to routinely perform.

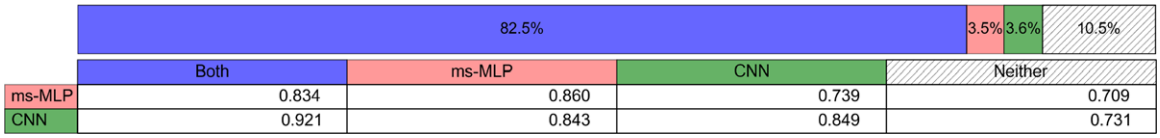


Figure 12. Network comparison between ms-MLP and CNN for fruit detection using the image segmentation output and WS detection algorithm. The partitioned bar shows the percentage of manually annotated fruits detected by both networks, by ms-MLP alone, by CNN alone and by neither network respectively. For the set of annotated fruits in each of these partitions, the table below is the mean detection probability from each network.

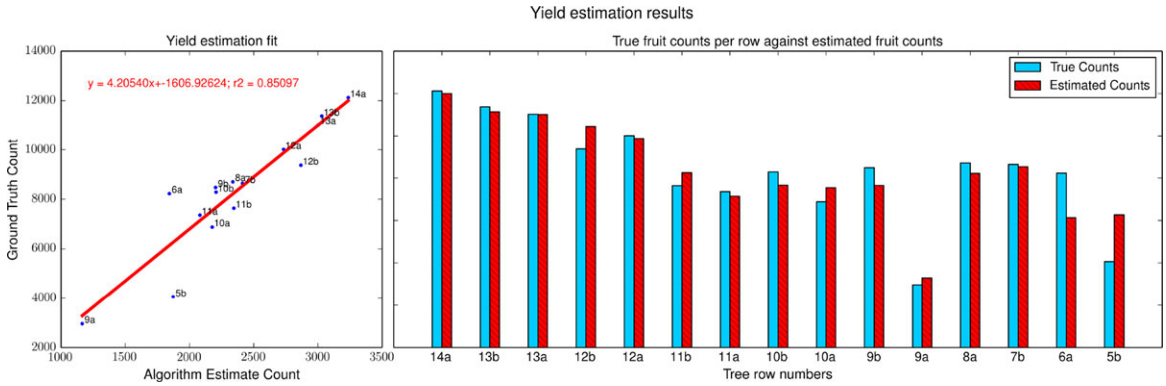


Figure 13. Yield estimation results. On the left is the predicted yield over 15 rows fitted against the post-harvest ground truth per row. The estimated yield counts against the true counts are shown on the right.

Table V. Yield estimation results using different image segmentation architectures and fruit detection algorithms. The results are the *r*-squared fit between the true per-row counts and the estimation counts, and the average absolute error in the yield estimation over the different rows.

	WS Detection		CHT Detection	
Segmentation Architecture	r-squared	Error (%)	r-squared	Error (%)
ms-MLP-3* (no metadata)	0.753 ± 0.047	13.3 ± 1.25	0.635 ± 0.051	17.53 ± 1.14
ms-MLP-3* (with metadata)	0.771 ± 0.036	12.2 ± 1.13	0.705 ± 0.072	14.4 ± 2.14
CNN*	0.826 ± 0.021	10.84 ± 0.62	0.763 ± 0.015	14.15 ± 0.32

yield and the true counts as shown on the left in Figure 13. The evaluation was done over multiple image segmentation training iterations, using the three different segmentation architectures. The results are shown in Table V.

Across all segmentation architectures, the use of WS detection algorithm produced more accurate yield estimation results compared to the CHT algorithm. This is on par with what was observed with the detection and counting results from Table IV. Using the ms-MLP-3* configuration (without metadata) and WS detection, we achieved an *r*-squared value of 0.753. This is greater than the baseline score of 0.635 achieved using CHT detection, comparable to 0.656 reported in Hung et al. (2015),[6] evaluated over a single iteration using the ms-MLP-2 network and the CHT detection framework. As reported in Bargoti & Underwood (2016), using the ms-MLP-3* architecture with metadata, the linear fit with WS detection increased to 0.771. Finally, with the CNN* architecture, the *r*-squared increased to 0.826, the best yet on this dataset. A linear fit between the algorithm row counts and the true row counts is shown on the left in Figure 13.

The calibrated linear models relating the algorithm counts to true fruit counts can be used to estimate the yield in each row. The count estimates using the optimal configuration is shown on the right in Figure 13. Yield estimation error can be evaluated by accumulating the absolute estimation errors per row normalized against the total fruit count. The associated results are shown in Table V. The use of metadata on the ms-MLP-3* architecture, improved this

[6]The previous publication mistakenly reported an *r*-squared value of 0.81, which was in fact the *r*-value.

error from 13.3% to 12.2%, whereas with the CNN* architecture, this improved to 10.84%.

Finally, a yield map can be produced by geo-referencing the counts per image given the vehicle position. The predicted counts can then be interpolated over a fixed grid to produce a yield map of the orchard block as shown in Figure 14. The yield map shows spatial variability in yield at the orchard block as illustrated by the figure insets, showing examples of areas with low- and high-yield count. The observed spatial variations correlate to certain features of the block, with Row 9*a* featuring trees planted more recently, and rows 14*b* − 15*b* at the very left in Figure 14 hosting a different variety of apple with much lower yield (harvest counts not available for these varieties).

7. DISCUSSION

Image segmentation accuracy for orchard image data increased when utilizing more complex classification architectures. Operating under natural illumination conditions, the previously proposed ms-MLP network by Hung et al. (2015) resulted in a fruit segmentation F1-score of 0.683. Optimizing this architecture with a deeper network increased the score to 0.728, which was further increased to 0.751 with the inclusion of metadata, as previously shown in Bargoti & Underwood (2016). With the CNN approach, we obtain the highest F1-score of 0.791, which is a modest improvement, yet importantly brings the state-of-the-art in agro-vision for apple detection closer to state-of-the-art in computer vision generally. The CNN architecture did not require any data pre-processing or layer-wise training prior to the supervised learning phase; however, training time was considerably slower compared to ms-MLP (5 min VS. 3 hr). For prediction, the CNN model could be implemented as a fully convolutional operation, leading to two to three times faster inference. With both architectures, it was important to conduct a thorough hyperparameter search over a held out validation data set in order to obtain the optimal segmentation results.

Orchard metadata were incorporated into the different architectures to help model some of the appearance variations and/or class distributions observed in the data. The maximum improvement in pixel-classification F1-score with metadata was observed when operating with the sub-optimal ms-MLP-2 network (Table I) and/or with minimal training instances (Figure 8). However, performance gains started to converge as the classification model complexity increases or additional training instances were provided. For all ms-MLP configurations, inclusion of metadata never degraded performance, however, the same was not true with CNNs where careful network configuration was required to avoid performance degradation with the addition of metadata.

In practice, such metadata information is available at no extra costs, in terms of both data acquisition and computation. However, careful consideration is required regarding how the meta-parameters may change between training and testing. When training is performed with limited variability within the meta-parameters, the system will not be able to extrapolate to unseen observations, e.g. cannot extrapolate to new rows. With limited training data (e.g. a small number of labelled images), inclusion of metadata can even degrade performance (Bargoti & Underwood, 2016). Therefore, the utility of the metadata needs to be first tested over a held out validation set. Additionally, if constrained with computational resources or time, and hence restricted to simpler classifiers, it can be advantageous to include such information to help obtain improved segmentation performance. Whereas, if a CNN approach is employed, the potentially negligible performance gain from metadata might not warrant its inclusion.

Fruit detection and yield estimation performance (as distinct from pixel-wise performance) was best when combined with the segmentation results from the state-of-the-art CNN architecture. The inclusion of metadata on the ms-MLP architecture resulted in minor detection, fruit counting, and yield estimation improvements. For fruit detection (from pixels to whole fruit), the WS algorithm performed better than CHT over all evaluation measures. This could be because the segmented fruits do not appear as circular objects due to noise in the segmentation output or due to heavy occlusions from non-fruit objects. Practically, the WS algorithm was also easier to tune, relying on a single hyperparameter, whereas the CHT algorithm was governed by six hyperparameters.

The proposed yield estimation and yield mapping technique provides the grower with information about spatial variability in their crop at a much finer scale than otherwise available. With uncalibrated estimates, growers can plan labor schedule and warehouse space, do harvest scheduling, and selective harvesting. With calibrated estimates, future farming strategies can be modified and tested at an annual basis. At this apple orchard, harvest counting is typically conducted at a block level, and the row-wise information presented in this study provided new insights into the orchard block. The grower noticed the smooth trend in decreasing yield from left to right in Figure 13, from which he inferred he needed additional pollinator trees in the lower yield sections of the orchard block.

7.1. Image Processing Errors

Most common image segmentation errors were in regions with poor image quality, due to adverse illumination conditions (e.g., sun flares and underexposure in shadows) or in ambiguous image regions (e.g., occluded apples between green leaves). In addition, there were errors and inconsistencies within the human-labeled data, where fruits were either missed or background fruits (in an adjacent row) were inconsistently labeled, limiting the performance

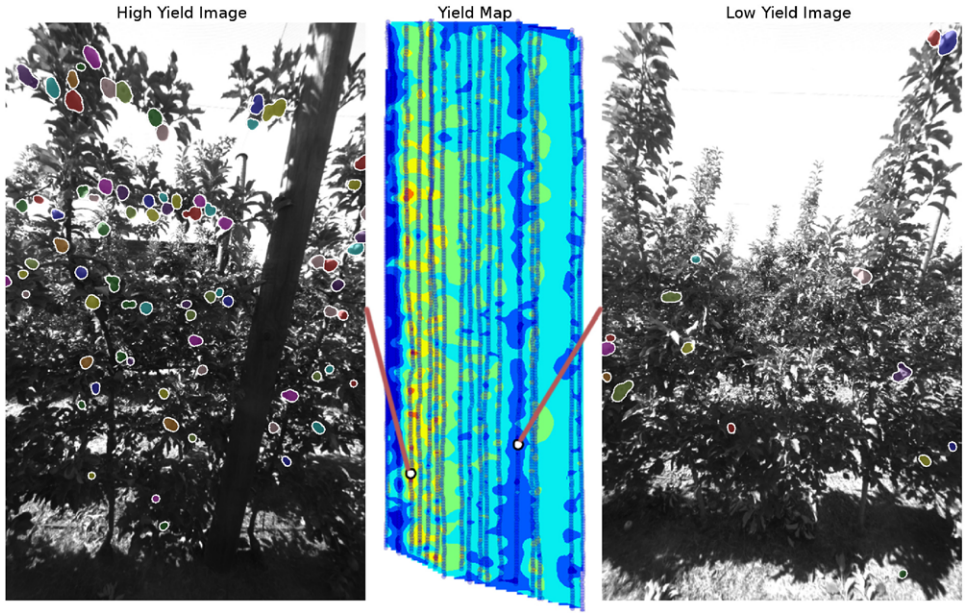


Figure 14. Apple yield map for the orchard block. Individual geo-referenced images are segmented using the CNN architecture and fruit detection is performed using CHT to obtain a fruit count per image. The yield map is computed through bilinear spatial interpolation of these counts. The insets show image samples with fruit detections in areas of high and low yield.

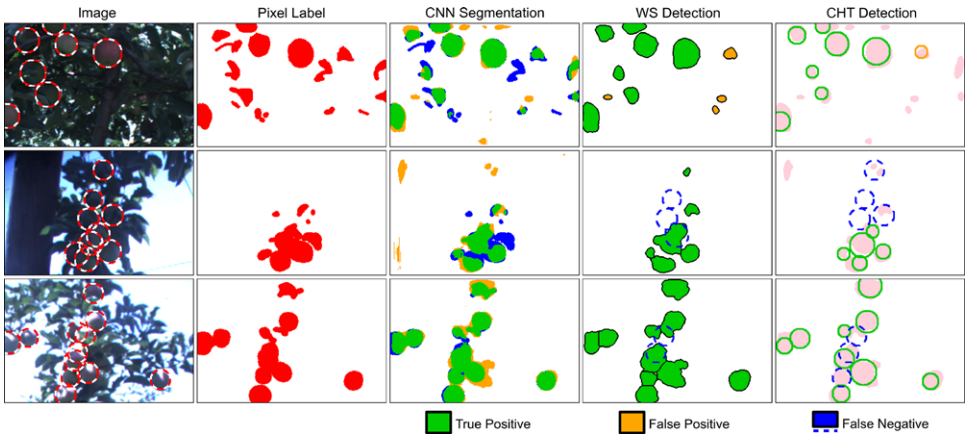


Figure 15. Instances of erroneous fruit segmentation, detection and manual labelling. The first column contains the input image data with annotated fruit detections. The second column contains the separately annotated pixel level labels, which were used to train the segmentation networks. The output from the trained CNN* network is shown in column three. The last two columns illustrate the output from the WS and CHT detection algorithms. The examples in the first two rows illustrate discrepancies in the ground truth, between the annotated detection and the pixel-level labels. The second and third examples illustrate under counting during detection due to clustered fruit regions.

and evaluation measures of the segmentation and detection algorithms. The first two rows in Figure 15 are examples of discrepancies in the ground-truth, seen through the differences in the pixel-wise and fruit-wise labels.

The primary causes of detection errors were poor image segmentation, undercounting of fruits appearing in clusters, and double-counting disjoint fruit regions. The WS and CHT detection algorithms are efficient at splitting fruit regions occurring in clusters of two or three apples but are unable to discern individual fruits in larger clusters. Examples on Rows 2 and 3 in Figure 15 illustrate cases where fruit clusters are under counted by the detection algorithms. However, such large cluster occurrences were rare due to standard thinning operations employed in this

orchard block, which are commonly used on orchards to optimize the quality of fruit (Wang et al., 2013). Errors with the CHT detection algorithm were also observed for instances in which occluded fruit were not circular in the image. Although outside the scope of this paper, the detection results could be further improved with more specific shape-based detection approaches such as the ones used in (Linker et al., 2012), in which fruits are defined as arc segments rather than whole circles.

Errors in yield estimation can be attributed to erroneous fruit detections (leading to overcounting or undercounting), detection of fruits from background rows, and double-counting or missed detections due to lack of fruit registration. In rows with sparse foliage, more of the background trees can be detected, therefore skewing the yield counts. Background removal via depth estimation (e.g., through stereo vision) would be required to minimize this error. To accurately accumulate fruit counts between frames, multiple viewpoint fruit registration would be required as done in Wang et al. (2013) and Moonrinta, Chaivivatrakul, Dailey, and Ekpanyapong (2010), which use stereo camera configurations to localize each fruit in $3D$ space.

7.2. Lessons Learned

The image segmentation, fruit detection, and yield estimation approaches presented in this paper can be extended to other orchards, with data sets captured under variable illumination conditions over different seasons. The learning approaches for image segmentation make them flexible and adaptable to a wide range of classification problems, while the same detection framework could be used for a variety of circular fruits, such as peaches, citrus, and mangoes. For practical deployment of such a system in commercial orchards, we reflect on a number of lessons learned from this work.

Data collected under natural illumination conditions is subject to extensive intraclass variations. Therefore, we become reliant on extensive training examples and more complex classification architectures to learn the highly nonlinear distribution. However, where possible, it would be advantageous to minimize such variations by capturing image data on an overcast day or when the sun is directly overhead or in the background (e.g., in the morning or the afternoon with the camera facing the other way). A more controlled operation could involve nighttime imagery with strobes as done in Payne et al. (2014); however, as mentioned in Section 2, daytime operations are more desirable for integration with the current farming practices.

Supervised image segmentation requires training examples, provided as pixel-wise labels in this paper, denoting fruit and non-fruit regions (the same as the ones used in Hung et al., 2015 and Bargoti and Underwood, 2016). The labeled data set was small compared to the collection of images acquired at the orchard block; however, pixel-wise annotation was still a time-consuming operation, taking approximately 1.5 hr per 100 images (308×202). In practice, it was also challenging to manually label images consistently, particularly where adverse illumination conditions caused partial under or overexposure (e.g., some of the images in Figure 6). One strategy to reduce the labeling costs/inconsistencies is to adopt fruit-level labeling, which we currently use only for the evaluation of detection performance. The whole-fruit labeling time was significantly shorter at under 30 min per 100 images. Future work is required to determine the optimal training regime for image segmentation or detection when using whole-fruit labels.

When extending the image segmentation framework to a new data set with different fruit or illumination characteristics, a new optimal model would need to be learned. This would require a data scientist to manually label a subset of the data and the computational constraints would then drive the choice of the segmentation architecture. These costs could be minimized by using/transferring segmentation models learned over previous data sets; however, further work is required to understand the limitations.

Differentiating from other works presented in Section 2, our aim had been to minimize hand-engineering features for image processing. In practice, while the neural network architectures enable this for image segmentation, the process of tuning hyperparameters is not completely trivial and is required to achieve good performance from these algorithms. Practically, it was easier to tune hyperparameters for the ms-MLP architecture compared to the CNN architecture because of its considerably shorter training time. The proposed fruit detection and yield mapping pipeline can be considered as a hybrid approach, as fruit detection relies on hand-engineered detection algorithms but does not require any additional training data or feature learning. A suitable next step would involve end-to-end learning for detection as done in Ren et al. (2015) to identify and count individual fruits. Such an approach can be trained with fruit-level annotations and is computationally cheaper than image segmentation as dense prediction is not required. However, orchard scenes represent a significantly different visual environment from typical urban scenes the detection framework is evaluated over. The cited approach performs classification on sampled object proposals in the scene and the cluttered nature of orchard data may cause difficulties in identifying fruit and, therefore, could result in degraded performance. Further considerations also need to be made for detecting fruits in clusters.

Because of occlusions and clustering, it is not possible to have all the apples clearly visible in the image data. Therefore, for yield estimation or prediction, the algorithm output for apple counting needs to be calibrated against a real ground-truth source. This can be done at the tree, multitree or row-segment, row (as done in this paper), or at an orchard-block level. The smaller-scale data collection could overlap with sparse manual sampling operations

conducted by agronomists, whereas large-scale calibration could be done post-harvest from the grading and counting machines. Both operations are already performed, so the data are available at no extra cost. For harvested counts, additional sources of error include fruit that fell to the ground after imaging but before harvest, fruit that is not successfully harvested, and fruit that was allocated to the wrong bin during collection. However, in situations in which these occur uniformly across the farm, it can be accounted for at the calibration stage. Further work is required to understand the amount of calibration data required, the cost of collecting it, and sensitivity to collection errors.

Finally, the system currently operates offline, providing the grower or an agronomist with a dense spatial yield map within a day or two of obtaining the data. However, with a prediction time of ~ 7 s for a 1232×1616 image, a pretrained detection pipeline could potentially be deployed on a robotic harvester or sprayer for real-time operation.

8. CONCLUSION

This paper presented an image processing framework for detecting fruit and estimating yield in an orchard. General-purpose feature learning algorithms were utilized for image segmentation, followed by detection and counting of individual fruits. The image segmentation was performed using ms-MLP and a CNN. Within these architectures, we incorporated metadata (contextual information about how the image data were captured) to explicitly capture relationships between meta-parameters and the object classes to be learned. The pixel-wise image segmentation output was postprocessed using the WS and CHT algorithms to detect and count individual fruits.

The different stages of the pipeline were evaluated on image data captured over a 0.5-ha apple orchard block using a monocular camera mounted on an unmanned ground vehicle. Captured under natural lighting conditions, the data set covered different apple varieties with appearances ranging from red, pink-green, to green. Metadata incorporated into the architectures included pixel position, row numbering and sun position relative to the camera. Following our previous work in Bargoti & Underwood (2016), the metadata yielded a boost in performance with the ms-MLP network. However, the best segmentation F1-score of 0.791 was obtained using a CNN network, with which the inclusion of metadata had negligible impact. The improved segmentation performance with the CNN also translated to more accurate fruit detection with both detection algorithms, with the best detection F1-score of 0.861 achieved using WS detection. The fruit counts were accumulated over individual rows at the orchard and compared against the post-harvest counts, where the best yield estimation fit was obtained with the same configuration, resulting in a squared correlation coefficient of $r^2 = 0.826$.

The feature-learning-based state-of-the-art image segmentation architectures enable us to process challenging orchard image data without needing to hand-engineer the feature extraction phase. To better understand their generalization capabilities, future work will involve image segmentation and fruit detection for different fruits/orchards. For commercial realization, future work will also investigate different labeling strategies for training the segmentation/detection algorithms and for transferring learning between different fruits/datasets.

ACKNOWLEDGMENTS

This work is supported by the Australian Centre for Field Robotics at The University of Sydney and Horticulture Australia Limited through Project AH11009 Autonomous Perception System for Horticulture Tree Crops. Thanks to Calvin Hung for his guidance with the MLP architecture and to Kevin Sander for his support during the field trials at his apple orchard. Further information and videos available at http://sydney.edu.au/acfr/agriculture

REFERENCES

Aggelopoulou, A. D., Bochtis, D., Fountas, S., Swain, K. C., Gemtos, T. A., & Nanos, G. D. (2011). Yield prediction in apple orchards based on image processing. Precision Agriculture, 12(3), 448–456.

Annamalai, P., Lee, W. S., & Burks, T. F. (2004). Color vision system for estimating citrus yield in real-time. In ASAE Annual International Meeting, St. Joseph, Michigan.

Atherton, T. J., & Kerbyson, D. J. (1999). Size invariant circle detection. Image and Vision Computing, 17(11), 795–803.

Bargoti, S., & Underwood, J. (2015). Utilising metadata to aid image classification in orchards. In IEEE International Conference on Intelligent Robots and Systems (IROS), Workshop on Alternative Sensing for Robot Perception (WASRoP), IROS 2015: Hamburg Germany.

Bargoti, S., & Underwood, J. (2016). Image classification with orchard metadata, In IEEE International Conference on Robotics and Automation (ICRA), Stockholm, 2016, pp. 5164–5170. Curran Associates.

Bargoti, S., Underwood, J. P., Nieto, J. I., & Sukkarieh, S. (2015). A pipeline for trunk detection in trellis structured apple orchards. Journal of Field Robotics, 32(8), 1075–1094.

Brown, M., & Susstrunk, S. (2011). Multi-spectral SIFT for scene category recognition. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 177–184). Providence, USA: Curran Associates.

Brust, C. A., Sickert, S., Simon, M., Rodner, E., & Denzler, J. (2015). Convolutional patch networks with spatial prior for road detection and urban scene understanding. In International Conference on Computer Vision Theory and Applications (VISAPP) (pp. 510–517). Berlin, Germany: SCITEPRESS.

Chhabra, M., Gupta, A., Mehrotra, P., & Reel, S. (2012). Automated detection of fully and partially riped mango

by machine vision. In Proceedings of the International Conference on Soft Computing for Problem Solving (SocProS): Vol. 131. Advances in intelligent and soft computing (pp. 153–164). Roorkee, India: Springer.

Ciresan, D., Giusti, A., Gambardella, L. M., & Schmidhuber, J. (2012). Deep neural networks segment neuronal membranes in electron microscopy images. In Advances in neural information processing systems 25 (pp. 2843–2851). Lake Tahoe, USA: Harrahs and Harveys.

Erhan, D., Bengio, Y., Courville, A., Manzagol, P.-A., Vincent, P., & Bengio, S. (2010). Why does unsupervised pre-training help deep learning? Journal of Machine Learning Research, 11, 625–660.

Farabet, C., Couprie, C., Najman, L., & LeCun, Y. (2013). Learning hierarchical features for scene labeling. IEEE Transactions on Pattern Analysis and Machine Intelligence, 35(8), 1915–1929.

Font, D., Tresanchez, M., Martínez, D., Moreno, J., Clotet, E., & Palacín, J. (2015). Vineyard yield estimation based on the analysis of high resolution images obtained with artificial illumination at night. Sensors, 15(4), 8284–8301.

Ganin, Y., & Lempitsky, V. S. (2014). N^4-fields: Neural network nearest neighbor fields for image transforms. CoRR, abs/1406.6.

Gemtos, T., Fountas, S., Tagarakis, A., & Liakos, V. (2013). Precision agriculture application in fruit crops: Experience in handpicked fruits. Procedia Technology, 8, 324–332.

Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2016). Region-based convolutional networks for accurate object detection and segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence, 38(1), 142–158.

Goodfellow, I., & Warde-Farley, D. (2013). Pylearn2: A machine learning research library. arXiv preprint arXiv:1308.4214 (pages 1–9).

Hung, C., Nieto, J., Taylor, Z., Underwood, J., & Sukkarieh, S. (2013). Orchard fruit segmentation using multi-spectral feature learning. In IEEE International Conference on Intelligent Robots and Systems (IROS) (pp. 5314–5320), Toko, Japan: Curran Associates.

Hung, C., Underwood, J., Nieto, J., & Sukkarieh, S. (2015). A feature learning based approach for automated fruit yield estimation. In Field and Service Robotics (FSR) (pp. 485–498). Brisbane, Australia: Springer.

Ji, W., Zhao, D., Cheng, F., Xu, B., Zhang, Y., & Wang, J. (2012). Automatic recognition vision system guided for apple harvesting robot. Computers & Electrical Engineering, 38(5), 1186–1195.

Jimenez, A. R., Ceres, R., & Pons, J. L. (2000). A survey of computer vision methods for locating fruit on trees. Transactions of the ASAE-American Society of Agricultural Engineers, 43(6), 1911–1920.

Kapach, K., Barnea, E., Mairon, R., Edan, Y., & Ben-Shahar, O. (2012). Computer vision for fruit harvesting robots–state of the art and challenges ahead. International Journal of Computational Vision and Robotics, 3(1-2), 4–34.

Kim, D., Choi, H., Choi, J., Yoo, S. J., & Han, D. (2015). A novel red apple detection algorithm based on AdaBoost learning. IEIE Transactions on Smart Processing & Computing, 4(4), 265–271.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems (pp. 1097–1105). Lake Tahoe, USA: Curran Associates.

Kurtulmus, F., Lee, W., & Vardar, A. (2014). Immature peach detection in colour images acquired in natural illumination conditions using statistical classifiers and neural network. Precision Agriculture, 15(1), 57–79.

Li, P., Lee, S.-H., & Hsu, H.-Y. (2011). Study on citrus fruit image data separability by segmentation methods. Procedia Engineering, 23, 408–416.

Linker, R., Cohen, O., & Naor, A. (2012). Determination of the number of green apples in RGB images recorded in orchards. Computers and Electronics in Agriculture, 81, 45–57.

Liu, S., Whitty, M., & Cossell, S. (2015). Automatic grape bunch detection in vineyards for precise yield estimation. 14th IAPR International Conference on Machine Vision Applications (MVA), Tokyo, 2015, pp. 238-241, Curran Associates.

Long, J., Shelhamer, E., & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 3431–3440). Boston, USA: Curran Associates.

Martens, J. (2010). Deep learning via Hessian-free optimization. In Proceedings of the 27th International Conference on Machine Learning (ICML) (pp. 735–742). Haifa, Israel: Omnipress.

Moonrinta, J., Chaivivatrakul, S., Dailey, M. N., & Ekpanyapong, M. (2010). Fruit detection, tracking, and 3D reconstruction for crop mapping and yield estimation. In International Conference on Control Automation Robotics & Vision (ICARCV) (pp. 1181–1186). Singapore: Curran Associates.

Ning, F., Delhomme, D., LeCun, Y., Piano, F., Bottou, L., & Barbano, P. E. (2005). Toward automatic phenotyping of developing embryos from videos. IEEE Transactions on Image Processing, 14(9), 1360–1371.

Nuske, S., Wilshusen, K., Achar, S., Yoder, L., & Singh, S. (2014). Automated visual yield estimation in vineyards. Journal of Field Robotics, 31(5), 837–860.

Payne, A., & Walsh, K. (2014). Machine vision in estimation of fruit crop yield. In Plant image analysis: fundamentals and applications (pp. 329–374). Boca Raton, USA: CRC Press.

Payne, A., Walsh, K., Subedi, P., & Jarvis, D. (2014). Estimating mango crop yield using image analysis using fruit at "stone hardening" stage and night time imaging. Computers and Electronics in Agriculture, 100, 160–167.

Pinheiro, P. H. O., & Collobert, R. (2013). Recurrent convolutional neural networks for scene parsing. CoRR, abs/1306.2.

Qiang, L., Jianrong, C., Bin, L., Lie, D., & Yajing, Z. (2014). Identification of fruit and branch in natural scenes for citrus harvesting robot using machine vision and support vector

machine. International Journal of Agricultural and Biological Engineering, 7(2), 115–121.

Rao, D., De Deuge, M., Nourani-Vatani, N., Douillard, B., Williams, S. B., & Pizarro, O. (2014). Multimodal learning for autonomous underwater vehicles from visual and bathymetric data. In IEEE International Conference on Robotics and Automation (ICRA) (pp. 3819–3825). Hong Kong: Curran Associates.

Regunathan, M. & Lee, W. S. (2005). Citrus fruit identification and size determination using machine vision and ultrasonic sensors. In 2005 ASAE Annual Meeting (p. 1) at St. Joseph, Michigan. American Society of Agricultural and Biological Engineers.

Ren, S., He, K., Girshick, R. B., & Sun, J. (2015). Faster {R-CNN:} towards real-time object detection with region proposal networks. CoRR, abs/1506.0.

Roerdink, J. B. T. M., & Meijster, A. (2000). The watershed transform: Definitions, algorithms and parallelization strategies. Fundamenta Informaticae, 41(1-2), 187–228.

Sa, I., McCool, C., Lehnert, C., & Perez, T. (2015). On visual detection of highly-occluded objects for harvesting automation in horticulture. In IEEE International Conference on Robotics and Automation (ICRA). ICRA: Seattle, USA; Curran Associates .

Regunathan, M. & Lee, W. S. (2005). Citrus fruit identification and size determination using machine vision and ultrasonic sensors. In 2005 ASAE Annual Meeting (p. 1) at St. Joseph, Michigan. American Society of Agricultural and Biological Engineers.

Sengupta, S., & Lee, W. S. (2014). Identification and determination of the number of immature green citrus fruit in a canopy under different ambient light conditions. Biosystems Engineering, 11, 51–61.

Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., & LeCun, Y. (2013). Overfeat: Integrated recognition, localization and detection using convolutional networks. arXiv preprint arXiv:1312.6229.

Silwal, A., Gongal, A., & Karkee, M. (2014). Identification of red apples in field environment with over the row machine vision system. Agricultural Engineering International: CIGR Journal, 16(4), 66–75.

Song, Y., Glasbey, C. A., Horgan, G. W., Polder, G., Dieleman, J. A., & van der Heijden, G. W. A. M. (2014). Automatic fruit recognition and counting from multiple images. Biosystems Engineering, 118(1), 203–215.

Stajnko, D., Rakun, J., & Blanke, M. M. (2009). Modelling apple fruit yield using image analysis for fruit colour, shape and texture. European Journal of Horticultural Science, 74(6), 260–267.

Tighe, J., & Lazebnik, S. (2013). Superparsing. International Journal of Computer Vision, 101(2), 329–349.

Wang, Q., Nuske, S., Bergerman, M., & Singh, S. (2013). Automated Crop yield estimation for apple orchards. In Experimental Robotics: Vol. 88. Springer Tracts in Advanced Robotics (pp. 745–758). Quebec City, Canada: Springer International Publishing.

Wijethunga, P., Samarasinghe, S., Kulasiri, D., & Woodhead, I. (2009). Towards a generalized colour image segmentation for kiwifruit detection. In International Conference Image and Vision Computing New Zealand (IVCNZ) (pp. 62–66). Wellington, NZ: Curran Associates.

Yamamoto, K., Guo, W., Yoshioka, Y., & Ninomiya, S. (2014). On plant detection of intact tomato fruits using image analysis and machine learning methods. Sensors, 14(7), 12191–12206.