High throughput measurement of *Arabidopsis thaliana* fitness traits using transfer learning

Peipei Wang[1,2‡], Fanrui Meng[1,2‡], Paityn Donaldson[1], Sarah Horan[1], Nicholas L. Panchy[3], Elyse Vischulis[4], Eamon Winship[1], Jeffrey K. Conner[1,5,6], Melissa D. Lehti-Shiu[1*], Shin-Han Shiu[1,2,4,6,7*]

[1]Department of Plant Biology, Michigan State University, East Lansing, MI, 48824, USA

[2]DOE Great Lake Bioenergy Research Center, Michigan State University, East Lansing, MI, 48824, USA,

[3]National Institute for Mathematical and Biological Synthesis, University of Tennessee, 1122 Volunteer Blvd., Suite 106, Knoxville, TN, 37996-3410, USA

[4]Genetics and Genome Sciences Graduate Program, Michigan State University, East Lansing, MI, 48824, USA

[5]W.K. Kellogg Biological Station, Michigan State University, 3700 E. Gull Lake Drive, Hickory Corners, MI, 49060 USA.

[6]Ecology, Evolution, and Behavior Graduate Program, Michigan State University, East Lansing, MI, 48824, USA

[7]Department of Computational Mathematics, Science, and Engineering, Michigan State University, East Lansing, MI 48824, USA

[‡]: Joint first authors

*: Joint corresponding authors: lehtishi@msu.edu, shius@msu.edu

ABSTRACT

Revealing the genetic effects of phenotypic variation is frequently challenging because of genetic redundancy, condition-specific requirements for genes, and/or changes in physiology or development that are too subtle to detect. Such effects can potentially be detected by measuring plant fitness, which reflects the cumulative effects of genetic changes over the lifetime of a plant. However, fitness is challenging to measure accurately, particularly in species with high fecundity and relatively small propagule sizes such as *Arabidopsis thaliana*. Here, we evaluated the performance of an image segmentation-based (ImageJ) and a transfer learning-based approach where a pre-trained model for detecting unrelated objects was repurposed for measuring two Arabidopsis fitness traits: seed and fruit counts. Although ImageJ was straightforward to use, the Pearson's Correlation Coefficient (r) between true and predicted seed counts was only 0.92 because seeds touching each other were undercounted. In contrast, the transfer learning-based approach yielded near perfect seed counts (r=0.9998). Using the latter approach, we were also able to detect and count fruits (r=0.99), although undercounting remains an issue for images with large numbers of fruits. Through data augmentation, we generated images of different resolutions, contrasts, brightness, and blurriness and used them to establish final models robust to differences in image properties. Beyond providing models to facilitate the investigation of Arabidopsis fitness traits, this study exemplifies how pre-trained models for different purposes can be reused for plant biology applications through transfer learning and machine vision.

Keywords: fitness traits; deep learning; transfer learning; machine vision

INTRODUCTION

A major goal of biology is to understand the molecular basis for development of organisms and their adaptation to different environments (Mcdonald, 1983). One approach to identify loci contributing to organism development/adaptation is to evaluate the effects of genetic variants on phenotypes. However, it is often challenging to investigate such effects because gene functions may be masked by genetic redundancy (Bouché and Bouchez, 2001) and/or be condition-specific (Hirsch et al., 1998; Meissner et al., 1999); the physiological or developmental changes caused by loss of gene function may also be too subtle to detect. This challenge can be alleviated by measuring the effects of genetic variations on fitness (i.e., the ability of an individual to survive and reproduce) because it reflects the cumulative effects of genetic changes over the lifetime of a plant. Accurate estimates of fitness are therefore valuable for several fields of study, including plant genetics, evolution, and plant breeding.

The increasing availability of genomics resources for model plants such as *Arabidopsis thaliana* has made it possible to systematically identify genomic regions that contribute to fitness (Külheim et al., 2002; Ganeteg et al., 2004; Fournier-Level et al., 2011; Ågren et al., 2013; Kerwin et al., 2015; Taylor et al., 2019). Among fitness measures, the most direct measure is the number of progenies produced (Thomson and Hadfield, 2017). In Arabidopsis, a predominantly selfing plant, the total number of seeds produced per plant is a particularly good estimate of fitness because it incorporates both male and female contributions. However, because Arabidopsis seeds are small (area ranging from ~0.1–0.2 mm^2; Jahnke et al., 2016) and produced in large numbers (up to thousands of seeds per plant Boyes et al., 2001; Morales et al., 2020), it is difficult to obtain accurate seed counts. As a consequence, fruit (silique) number (Busoms et al., 2015) and total fruit length (Roux et al., 2004; Kerwin et al., 2015) are often used to measure fitness. Both measures have been shown to be correlated with seed production but with highly variable correlations across studies, ranging from r^2=0.960 for fruit number (Mauricio and Rausher, 1997) and

r^2=0.988 (Roux et al., 2004) to r^2=0.256 (Gnan et al., 2014) for fruit length. In addition, fruit numbers (up to 450 per plant, Hamidinekoo et al., 2020) are typically counted manually, and can be error-prone. Thus, to better measure fitness of plants, both fruit and seed numbers should be evaluated using methods that are not hindered by propagule size or number.

Several programs have been designed to increase the efficiency and accuracy of seed analyses. Some are aimed at measuring the properties of individual seeds rather than obtaining high throughput seed counts, e.g., seed size and shape (Herridge et al., 2011; Tanabata et al., 2012; Moore et al., 2013). These approaches typically require that seeds be separated before capturing seed images, which increases the time needed for processing. Other systems have been designed to separate seeds mechanically. For example, the *pheno*Seeder device separates seed using a “pick-and-place” robot (Jahnke et al., 2016); Morales et al. (2020) used a large-particle flow cytometer to separate seeds, and counted an average of 12,000 seeds per hour with high accuracy (relative error < 2%). Approximately 10,000 seeds can be processed per hour using the BELT imaging system combined with the phenoSEED algorithm, which acquires images of individual seeds as they pass through an imaging chamber (Halcro et al., 2020). Nonetheless, a drawback of these methods is that they require specialized equipment, hindering their widespread adoption.

Another approach that has been increasingly used in plant biology is machine vision, the application of machine learning algorithms to image analysis (Mochida et al., 2019). Deep learning approaches, in particular Convolutional Neural Network (CNN)-based frameworks, such as Single Shot Detector (SSD) (Liu et al., 2016), Faster Region Based CNN (Faster R-CNN) (Ren et al., 2017), and You Only Look Once (Yolo) (Redmon et al., 2016), have been developed to detect vastly different objects (e.g., cats, cars) in images. These frameworks can be applied to other tasks such as fruit and seed counting using transfer learning methods (Yang et al., 2020), whereby knowledge (i.e., prediction models) from one task is transferred to a new task. For example, starting from the pre-trained

residual network ResNet101 (He et al., 2016), the seeds of rice, lettuce, oat, and wheat, were detected with 96% recall and 95% precision using Mask R-CNN (Toda et al., 2020). However, the detection of much smaller objects using CNN-based approaches remains challenging (Cao et al., 2019). This is likely because CNNs create low-level abstractions of the images, and if the objects are too small, the resulting abstractions are too simple to be used to distinguish whether the object is present or not. Although the CNN-based models developed by Toda et al. (2020) detected seeds with high accuracy, the smallest seeds tested were lettuce seeds, which have areas ranging from 1.5–3.6 mm^2 (Penaloza et al., 2005) and are ~10 times larger than Arabidopsis seeds. Another consideration is that the most convenient way to count all the seeds from an Arabidopsis plant, which can produce thousands of seeds (Boyes et al., 2001; Morales et al., 2020), would be to put all the seeds in a single image, thus resulting in a relatively small ratio of seed size to image size. However, due to the small image size (1024*1024 px^2 or 2000*2000 px^2) used in Toda et al. (2020), the ratio of seed size to image size was relatively large (>5000 px^2 per barley seed), and the number of seeds that could be included in an image was also limited. Therefore, it is important to assess how well the CNN-based approaches perform in detecting objects as small as Arabidopsis seeds in an image containing thousands of them.

CNN-based approaches have also been used in fruit counting. For example, using R-CNN, wheat spikes can be detected, counted, and analyzed to estimate yield based on images captured in the field (correlation between true and predicted counts: R^2=0.93 with slope of 1.01, Hasan et al., 2018). Starting from two pre-trained models (ResNet and ResNext), Afonso et al. (2020) applied the Mask R-CNN approach to detect and count tomato fruits from images captured in the greenhouse, obtaining an F1 of 0.94 when fruits were only partially overlapping with each other. Using the LeNet and DenseNet-Basic models as a starting point, DeepPod was developed to effectively count Arabidopsis fruits but had a high number of false negatives at higher numbers of fruits (R^2=0.90 with a slope of ~0.70, Hamidinekoo et al., 2020) In addition, when using DeepPod the

inflorescences need to be harvested before the seeds are mature when the fruits are still green, preventing the harvesting of seeds for future propagation or analysis. Thus, it is important to develop tools or models to detect and count mature fruits when seeds need to be saved for future experiments. Because Arabidopsis fruits shatter easily when dry, such tools should ideally be able to count fruits at different stages, including intact fruits and those that have already dehisced and released seed, when measuring fruit and seed production of plants grown to maturity (Conner and Rush, 1997).

In this study, with the goal of comprehensively measuring Arabidopsis fitness, we evaluated two existing approaches—segmentation with ImageJ (Schneider et al., 2012) and a deep learning approach, Faster R-CNN (Ren et al., 2017)—for counting seeds. Faster R-CNN has been widely used for detecting multiple types of objects, including those related to plant-based applications such as tomato disease types (Wang et al., 2019) and seedlings growing in the field (Jiang et al., 2019). Using the transfer learning framework (Yang et al., 2020), a pre-trained model, faster_rcnn_inception_v2_coco for general image classification, was adopted as a starting point for training seed and fruit detection and counting models. All the seeds an Arabidopsis plant produced were counted at once. We also applied Faster R-CNN to count fruits in whole plant images that were captured after seeds were mature. To be able to measure seed and fruit numbers robustly in diverse images, we established Faster R-CNN models using input images with varying resolution, contrast, brightness, and blurriness. The final seed and fruit models are provided along with extensive documentation so that they can be readily used by the research community.

RESULTS

Seed Counting Using ImageJ

Because the “Analyze Particles” function of ImageJ is widely used for analysis of seed morphology (Cervantes et al., 2016), we first attempted to develop a pipeline for seed counting that incorporated ImageJ analysis based on

segmentation of seed areas. With our seed scanning setup (see **METHODS**), when fewer than 200 seeds were placed on the plate lid and separated using forceps, seeds were detected and counted with high accuracy (Pearson's Correlation Coefficient [PCC] between true and predicted seed counts was 0.998, slope=0.9998, 60 images, **Figure 1, Table S1**). Our ImageJ pipeline allowed the detection of about 52 template images (each containing 12 plate lids, thus a total of 624 plate lids) per hour with a typical laptop (Intel(R) Core i7-7500U CPU, 16GB RAM).

However, when seeds were placed on plate lids without separation, big clumps of seeds were not counted by ImageJ, and small clumps where a small number of seeds were touching each other were recognized as single seeds (**Figure 2A**). The prediction accuracy of ImageJ drops off as the number of seeds increases (**Figure 2C, Table S2**); this is because the more seeds there are on the plate lid, the more likely it is that seeds touch each other, leading to an increase in the false negative rate of prediction. Moreover, the detection of seeds could be disrupted by scratches or letters on the plate lids, and seeds outside the predefined circular search regions were not detected (purple arrowheads in **Figure S1**). Thus, to obtain accurate counts based on segmentation, it is necessary to separate seeds and confine them to the center of the plate lid, which is time consuming and not amenable to high-throughput analysis.

Improved Seed Counting by Faster R-CNN

Next, we evaluated the performance of the deep learning approach Faster R-CNN in seed counting. Since it is very time-consuming to annotate a large number of small objects for model training, we first split the 256 whole-plate images into 1024 quarter-plate images, and then manually labeled a subset (180) of these quarter-plate images to speed up the training process. A total of 160 labeled quarter-plate images (*Training image set 1* in **Figure 3A**) were used to build the models. The other 20 labeled quarter-plate images were set aside as the validation set (*validation image set* in **Figure 3A**). Three hyperparameters

were tested to optimize the model performance: proposals—the number of detected regions (i.e., seeds) in an image; scales—relative sizes of detected regions; and aspect ratios—shapes of the detected regions (**Figure S2**). The established models were then applied to the validation set to evaluate the performance of models based on different hyperparameter combinations. Model performance was measured using the F-measure (F1)—harmonic mean of precision (proportion of the detected areas that are true seeds) and recall (proportion of true seeds that are detected).

We first examined 63 hyperparameter combinations: three proposal numbers (100, 500, and 1000), three scales (A to C), and seven aspect ratios (A to G, **Figure S3**; for scale and aspect ratio values see **Table S3**), resulting in 63 models ($Model_{seed}$ 1–63). We found that models with 100 proposals had the least accuracy with F1 < 0.750 (**Figure S3A**), while models with 500 and 1000 proposals had much improved performance, with F1 values around 0.970 (**Figure S3B,C**). This is because the seed counts in most of the quarter-plate images in the validation set were larger than 100 (**Table S4**). For models with 500 or 1000 proposals, scale-B and scale-C had higher F1 values than scale-A, but there were no differences between the F1 values of the seven different aspect ratios (**Figure S3B,C**). Because the computational efficiencies were higher for scale-B and aspect ratio-A than for other hyperparameter combinations (**Figure S4**), the combination of scale-B and aspect ratio-A was used for downstream model building. Three additional models were also established based on scale-B and aspect ratio-A with 3000, 5000, and 10,000 proposals ($Model_{seed}$ 64-66) with no improvement in F1 values compared with models with 500 and 1000 proposals (**Figure S3D**). Even though building Faster R-CNN models with a higher number of proposals requires more computational resources (Ren et al., 2017), we opted to use 10,000 proposals for further model building to allow detection of a large number of seeds in an image.

$Model_{seed}$ 66—built with 10,000 proposals, scale-B, and aspect ratio-A—was used to detect seeds in the remaining 844 quarter-plate images to produce "*in*

silico" seed annotations for the second-round modeling (**Figure 3A**). The predicted coordinates of seeds for a set of four quarter-plate images were combined and converted to the corresponding coordinates in the original whole-plate image. The seed coordinates in the whole-plate images were manually corrected (i.e., false negatives were manually labeled, and false positives were removed) to produce a new set of seed annotations, resulting in 211 labeled whole-plate images. Then a new model, $Model_{seed}$ 67, with the same parameters as $Model_{seed}$ 66, was built using 161 (*Training image set 2* in **Figure 3B**) out of these 211 images (**Figure 3**). The remaining 50 labeled whole-plate images (*Test image set* in **Figure 3B**) were used to evaluate the performance of $Model_{seed}$ 67, which had an average F1 of 0.992 (**Table S2**). When further examining the seeds detected in detail, we found that in contrast to the ImageJ approach, $Model_{seed}$ 67 correctly predicted seeds regardless of whether they were in contact with each other or not (**Figure 2B**). In addition, in contrast to the ImageJ approach (**Figure 2C**), the prediction accuracy was not influenced by the total number of seeds in an image (PCC=0.9998, p=1.7e-83, **Figure 2D**), and the differences between true and predicted seed counts were close to zero, much smaller than those in ImageJ analysis (**Figure 2E**). Furthermore, $Model_{seed}$ 67 allowed the detection and counting of seeds in about 240 whole-plate images per hour using 1 GPU (Nvidia Tesla K80) with 4 GB of GPU memory in a UNIX cluster, or about 33 images per hour using a laptop with 16 GB of memory. These results suggest that our Faster R-CNN-based models provide highly accurate Arabidopsis seed counts and can be used for large-scale fitness studies.

Impact of Seed Density on the Faster R-CNN Model

The number of seeds in an image has a detrimental effect on the performance of ImageJ, but not on that of Faster R-CNN, as the correlation between the true and predicted numbers of seeds is nearly perfect (**Figure 2D**). This suggests that the Faster R-CNN model performance was not affected by the seed density. To verify this, a measure of seed density is needed. This is because the same

number of seeds can be evenly distributed on a plate lid, or crowded in a specific area so that seed density is very high in some parts of the lid but low in others. Thus, we generated a seed density index (SDI) to measure seed density in an image by drawing a circle with a radius of 30 pixels (corresponding to 0.62 mm) from the center of a seed, then calculating how many seeds had their central points located within the circle. We choose 30 pixels because it is approximately the total length of two seeds. Finally, the average number of seeds per circle in a whole-plate image was defined as the SDI (**Figure 4A**).

Using this strategy, we calculated the SDIs of the test set images and determined the PCC value between SDI and the performance of $Model_{seed}$ 67 (**Figure 4B**). Example whole-plate images with different SDIs are shown in **Figure S5**. The results showed that the higher the seed density, the lower the model performance (PCC between SDI and F1 was -0.581, *p*=9.8e-06, **Figure 4B**; for the correlation between SDI and other performance measures see **Figure S6**). Nevertheless, the effect of seed density on the performance of $Model_{seed}$ 67 was small, as the F1 only dropped from 1.000 for an SDI of 1.157 to 0.971 for an SDI of 3.100 (**Figure 4B, Table S2**). An F1 of 0.971 with a recall of 0.968 indicates that for an image with 1000 seeds, there would only be 32 false negatives (seeds not detected) and 25 false positives (seeds detected in an image area with no seeds or a seed area counted more than once). Consistent with this, there was no significant correlation between the SDI and the difference between true and predicted seed counts (PCC=-0.206, *p*=0.15), in contrast to the significant negative correlation observed for ImageJ (PCC=-0.886, *p*=1.2e-17, **Figure 2F**). In addition, we calculated the SDIs for the predicted seed coordinates and found that the PCC value between true SDIs and prediction-based SDIs was 0.997 (*p*=1.5e-54; **Figure 4C**), demonstrating that our Faster R-CNN model also predicts the location of seeds very well.

Model Improvement through Data Augmentation

Our goal is to provide a seed counting model that can be widely used by different researchers, who may have seed images with different properties. Thus, we investigated the utility of $Model_{seed}$ 67 for seed counting using images with four varying attributes: resolution, contrast, brightness, and blurriness (**Figure 5A**). These modified seed images were created by modifying the properties of the test set images (**Figure 3B,** for the image property settings see **Table S5**). Note that the test set images have not been used for training or validating $Model_{seed}$ 67. Thus, they are ideal for testing the model independently. In the modified test set, there were 1750 images in total, including the original test set images (50) and modified images with 34 different attributes (34x50, light green box, **Figure 3B**). A slight but significant decrease in F1 was observed when the brightness of the images was ≤ 0.60 (p=0.01, one-sided Wilcoxon signed-rank test) relative to the original images, while the F1 dropped dramatically when the relative brightness was ≥ 1.20 (p=6.4e-08, **Figure 5B**). A significant decrease in F1 was also observed when the relative contrast of images (relative to the original image) was ≤ 0.50 (p=1.0e-07) or ≥ 1.75 (p=5.0e-4), the relative blurriness was ≥ 1.50 (p=6.7e-10), or the relative resolution was ≤ 0.50 (p=9.1e-10, **Figure 5B**). These results suggest that although $Model_{seed}$ 67 is suitable for a range of image qualities, the seed detection accuracy will decrease dramatically when the image properties deviate from the training images beyond a certain point.

To improve the robustness of $Model_{seed}$ 67, we applied data augmentation, in which the size and properties of training datasets are increased so better prediction models can be built (Shorten and Shorten, 2019). To accomplish this, we used 20 of the 161 training set 2 images to produce additional images with 21 different property settings (21 x 20, darker green box, **Figure 3B,** for the image property settings see **Table S5**). These 420 additional images, together with the original 161 images, were used to build the new $Model_{seed}$ 68 (**Figure 3B**), with the same hyperparameter settings as $Model_{seed}$ 67. $Model_{seed}$ 68 was then used to detect seeds in the modified test set images (again, these were not used in any step of the training process). Although there was a slight decrease in F1 when the relative blurriness was ≥ 3.00 (p=0.04, median F1 decrease=0.002) or

when the relative resolution was ≤ 0.30 (p=0.02, median F1 decrease=0.003, **Figure 5B**), $Model_{seed}$ 68 (blue, **Figure 5B**) performed better than the non-augmented $Model_{seed}$ 67 in all situations (red, **Figure 5B**) and thus, the augmented model is robust to different image properties.

Fruit Counting Using Faster R-CNN Models

Compared with seed number, total fruit count is an even more frequently used proxy for fitness. When scoring total Arabidopsis fruit counts, an important consideration is that, because dry fruits shatter easily, it is not always possible to harvest all fruits produced by a single plant after seeds have matured, especially for plants growing in the field. In this case, the best method would be to count all fruits (including dehisced ones) and count seeds per fruit for a subset that haven't dehisced, and then calculate total seed number by multiplying the number of seeds per fruit with the total fruit number. Thus, to obtain more accurate estimates of seed production per plant, it is necessary to record the numbers of both intact and shattered fruits. With these considerations in mind, we developed Faster R-CNN models to count all fruits without harvesting the fruits first. When capturing the images for fruit counting, a pink background was used to maximize the contrast between the background and the dark, dry fruits and the pale replum of shattered fruits that remained after the valves fell from the fruit (**Figure 6A,D**). Because there were much fewer fruits in each image than the number of seeds when performing seed counting, and the fruits were much larger, we manually labeled the fruits in 120 images without using the two-step strategy used for seed counting.

Eighty, 20, and 20 images were randomly selected and used as training, validation, and test sets, respectively (**Figure 6A**). Different combinations of hyperparameter values (**Table S6**) were evaluated ($Model_{fruit}$ 1-75, **Figure 6A**). Surprisingly, all models built with different hyperparameter combinations performed similarly on images in the validation set, with an average F1 of 0.925 (**Figure S7**). Thus, to minimize the computational cost (lower scales or aspect

ratios) while maximizing the number of fruits detected per plant (more proposals), the model built with $scale_{fruit}$-A, aspect $ratio_{fruit}$-A, and 500 proposals ($Model_{fruit}$ 21) was used. $Model_{fruit}$ 21 was applied to the test set images, resulting in an average F1 of 0.914 (**Table S7**). This F1 value translates into 1 false positive and 15 false negatives for an image with 100 fruits. Although the PCC between true and predicted fruit counts was 0.990 (p=6.7e-17), the detection error increased with an increasing number of fruits in an image and, because the fitted line had a slope of 0.785, the error was mostly due to undercounting or false negatives (**Figure 6B,C**). Upon further examination, we found that the majority of the false negatives were unopened fruits, especially those overlapped with the stem or with each other. One potential reason for errors in detection of these fruits is that they are similar to the stem in color and shape. Another reason may be the relatively small number of labeled intact fruits (543) compared with the number of pale replums (2082) in our training images. Thus, our model can potentially be improved by increasing the number of the intact fruits in the training set.

To assess the robustness of our model on images with different qualities, we applied $Model_{fruit}$ 21 on test set images with different image properties (**Figure 6D,** modified test set, 700 images in total, for the image property settings see **Table S5**). Significant decreases in F1 were observed when the relative image brightness was ≤ 0.70 (p=0.04) or ≥ 1.40 (p=0.02), the relative contrast was ≤ 0.50 (p=0.02) or ≥ 1.50 (p=0.03), the relative blurriness was ≥ 2.0 (p=0.002), or the relative resolution was ≤ 0.6 (p=0.05) (**Figure 6E**). By including images with different properties (**Table S5**) in the training set (1840 images in total), a new model, $Model_{fruit}$ 76, was established and applied to the modified test set. A significant but slight decrease in the resulting F1s was only observed when the relative resolution was ≤ 0.3 (p=0.02, median F1 decrease=0.01) (**Figure 6E**), indicating the robustness of $Model_{fruit}$ 76. Using this model 180 images could be processed per hour using a UNIX node with 1 GPU and 4 GB graphics memory or 90 images per hour could be processed using a laptop (1 CPU, 16 GB memory).

DISCUSSION

Fitness upon loss of function of a gene is one of the best measures of gene functionality because it reflects the ability of a plant to survive and reproduce given all the phenotypic effects of the mutation over the lifetime of the individual. For self-pollinating species such as Arabidopsis, fitness is better assessed by counting the numbers of seeds than fruits, as they more directly reflect the number of offspring and reproductive success. Due to the lack of an effective tool enabling high throughput counting of small seeds en masse, seed counts are often estimated indirectly, for example by dividing the total seed weight per plant by the estimated individual seed weight in the same batch (Cvetkovic et al., 2017), or by multiplying the fruit count by the average fruit length (Kerwin et al., 2015; Taylor et al., 2019). However, these approaches may not yield accurate estimates of seed production due to potential measurement errors and the imperfect correlation between seed number and fruit length (Roux et al., 2004). Here, we developed a tool employing a deep learning approach, Faster R-CNN, to count seeds and fruits with a near perfect accuracy. In particular, the Faster R-CNN-based predictions outperform those of ImageJ, a well-known platform with macros/modules for segmentation and morphology extraction (Schneider et al., 2012; Cervantes et al., 2016; Vasseur et al., 2018). Most importantly, the advantages of the Faster R-CNN model over ImageJ are that it can detect individual seeds when the seed density is high without the need to separate them or when seeds partially overlap with artifacts such as scratches on the substrate surface. Also, it is not necessary to predefine regions indicating the location of the seeds. Our final model almost perfectly detected and counted seeds with an F1 close to 1, and allowed robust seed detection for images with multiple different properties or qualities.

With the significance of the deep learning approach in seed counting noted, there are three general challenges that we faced during model development. The first challenge is that deep learning approach-based object detection tasks require a large amount of labeled data (in our case, labeled seeds). In this study, we

adopted a two-step modeling strategy to reduce the labor needed for seed annotations. In step 1, we split the images and used a subset of the split images to build a preliminary model (F1<0.975) and applied it to the remaining images. While the predictions were not perfect, this step drastically reduced the manual annotations needed because we only needed to correct for mis-predictions to boost our seed labels by ~5 fold (29,360 labels in the first-round, 138,929 labels in the second-round). Using this much larger set of seed labels, in step 2, new models were built that had improved model performance (F1=0.992), indicating the effectiveness of our strategy. An alternative strategy to increase the number of labels is based on the concept of domain randomization. Using this approach, Toda et al. (2020) generated a synthesized dataset for model training by randomly pasting individual images of seeds from a seed image pool to virtual canvases. The number, position, and direction of seeds on the canvases were all randomized, and seeds could be crowded together or covered by each other (Toda et al., 2020). However, the F1 of seed counting in that study was ~0.95, which may be because the morphology of seeds in the seed image pool (20 seed images for each of the 20 cultivars) was not representative of seed shapes in larger samples. In this study, we labeled 105,045 seeds for the training set of $Model_{seed}$ 67, making it more likely that all seed types investigated here were representative.

The second major challenge is the difficulty in adopting the deep learning framework for our specific applications. The pre-trained model that was used as the starting point of our Faster R-CNN models was originally used for detecting vastly different objects, from cats and dogs to TVs and cars. Although Faster R-CNN is widely used for image-based applications, the computer science literature frequently does not have sufficient or proper documentation facilitating applications in other domain areas. In addition, the rapid development cycle of the modeling software, Tensorflow, made the activation energy for getting the approach to work very high; this was because we encountered frequent anonymous errors that took significant amounts of time to troubleshoot. With the hope of passing on our experience and facilitating similar applications of our

models, we have documented our approaches extensively on our GitHub page, from model training to model application.

The third challenge is the high degree of computational complexity. In the **RESULTS** section, we emphasized that we can *apply the model* with modest computational resources with reasonable speed for detecting Arabidopsis seeds and fruits. However, to *build the model*, there is a substantial requirement for computational resources. For example, for building one of the first-stage seed counting models with a particular set of hyperparameters, the average run time was >4 h while requiring the use of three GPUs with >300 Gb RAM requested. Over the course of the project, over thousands of computing jobs requiring similar amounts of resources were run for model building, particularly during the early phase of development. Thus, development of the seed and fruit counting models was far from trivial computationally, and it can be challenging to build a model from scratch.

The Faster R-CNN approach also shows promise in fruit detection and counting (R^2=0.98, slope=0.79). In contrast to the finding that the Faster R-CNN approach outperformed the ImageJ approach for seed counting by a large margin, our Faster R-CNN fruit counting model did not perform as well as an ImageJ-based segmentation and skeletonization approach that was previously developed to enable high throughput measurement of fruit number (R^2=0.91, slope= ~1, Vasseur et al., 2018). One possible reason underlying the performance difference between our seed and fruit counting models could be the huge difference in label numbers (there were about 52 times more labeled seeds in the training set than labeled fruits). The performance of the fruit counting model is thus expected to be improved when more fruit labels are included to train the model. In addition, one notable caveat of our approach is the undercounting at higher fruit numbers; this was mainly due to overlap between intact fruits and between intact fruits and stems. To remedy this, one approach is to rearrange the inflorescences before capturing the images to avoid the overlapping among fruits and stems. Another potential approach is to analyze multiple images (or

frames of a movie) taken at different angles or to examine the 3D reconstruction of the inflorescence.

Nevertheless, the performance of our fruit counting model was better than that of another recently published CNN-based approach, DeepPod (R^2=0.90, slope= ~0.70, Hamidinekoo et al., 2020). In Hamidinekoo et al. (2020), the task (i.e., fruit detection) was first divided into four classification tasks: the detection of the tip, body, and base of the fruits and the detection of the stem. The separately detected parts were then joined together as a whole fruit. As the authors noted, the post-processing step (i.e., combining the four regions into a whole fruit) affected the final fruit detection performance. In our study, the fruits were labeled and detected as whole objects, thus avoiding the need for post-processing. In addition, different from Hamidinekoo et al. (2020) and Vasseur et al. (2018), where most fruits and the stems were fresh and green, fruits in our study were dry and light brown to gray, or contained only the pale replum remaining after seeds were dispersed. Thus, our fruit counting approach is expected to be applicable to a wider range of Arabidopsis fruit developmental stages. This is especially important when plants must be grown to maturity, and seed counts are estimated by calculating the number of seeds per intact fruit and multiplying by the total number of fruits (intact and dehisced).

Taken together, our results illustrate that almost perfect performance can be obtained using Faster R-CNN-based models to detect Arabidopsis seeds. We also showed that a Faster R-CNN model is also capable of detecting Arabidopsis fruits and distinguishing them from the stem and background. It is intriguing that faster_rcnn_inception_v2_coco, the starting point of our model, was originally built to classify objects very different from Arabidopsis seeds and fruits. By using this pre-trained model as the initial model but changing the inputs (seed/fruit images) and the desired outputs (seed/fruit label and location), the model originally trained for a completely different purpose could be reused for our more specific application. This is consistent with the general idea of transfer learning (Yang et al., 2020) where information learned from one domain can be applied to

another domain area using machine learning. Beyond the specific application in this study, we expect that the general transfer learning framework can be applied widely to resolve other questions in biological sciences.

METHODS

Plant Materials and Growth Conditions

For seed counting, seeds in the Arabidopsis Col-0 background were sown in 200-cell flats filled with Arabidopsis mix (1:1:1 SureMix, vermiculite, and perlite), stratified for 5–7 days in the dark at 4°C, then transferred to a growth chamber and grown under a 16-h light/8-h dark cycle with a light intensity of 110–130 μmoles/m^2/s at 21°C. Seedlings were thinned to one per cell after 1 week. Plants were watered two to three times per week and grown to maturity, after which the fruits produced by individual plants were counted. Intact fruits were transferred to glassine envelopes and allowed to dry for at least one month before counting seeds. For fruit counts, seeds with the Arabidopsis Col-0 background were sown in 200-cell flats filled with MetroMix 360, stratified for seven days in the dark at 4°C, then grown under a light intensity of ~150 μmoles/m^2/s at 21°C. After five weeks (23 November 2018) flats were transferred to a plot at Kellogg Biological Station. Plants were removed from the field on 23 May 2019. Plants were cut at the base and imaged as described below.

Seed Image Scanning

Seeds from different numbers of dry fruits were scattered on the lid of a 60-mm petri plate. Forceps were then used to remove the fruit walls so that only the seeds remained. The petri plate lid containing seeds was tapped with fingers to ensure that seeds did not cover each other. For some plate lids counted with ImageJ, seeds were separated from each other and moved away from the edges of the plate lid using forceps. The uncovered petri dish lids were placed in a template made from white acrylic (295 mm X 210 mm X 10 mm, **Figure 1A**) on a standard desktop scanner. This template was designed to be the same height as

the plate lid to minimize artifacts resulting from light loss. For the template, 12 60-mm diameter holes were cut into the template using a laser cutter. These holes were arranged in four rows of three and were spaced 6 mm from the edges and from each other. Within a column, holes were spaced 10 mm from the edges and each other. The template was aligned to the top right corner of the scanner before every scan because even small changes (<1 cm) in the location of the template required changing the parameters for the circular search regions in the ImageJ algorithm (described in the following section). Seeds were scanned at 1200 dots per inch and in 24-bit color, and scans were saved as jpeg files.

Seed Image Processing and Counting with ImageJ

The ImageJ workflow is shown in **Figure 1**. Scanned images were first converted into 8-bit grayscale bitmap format with the im.convert('L') method in the Python Imaging Library (https://pythonware.com/products/pil). The converted images were opened in ImageJ version 1.52a (https://imagej.nih.gov), and seeds were counted using the "Analyze Particles" tool. A macro was written to automate seed counting. First, circular search regions were defined to confine the search to the plate lids containing the seeds; there were 12 regions (i.e., petri plate lids) per scanned image. Second, a threshold was applied by selecting pixels with intensities between 50 and 140, and pixels were converted to real world units using a predefined line covering a known distance (the distance across the plate lid, 60 mm). Third, seeds within the circular search regions were counted. Only above-threshold regions of interest greater than 0.06 mm^2 and with a circularity value of 0.25 to 1 were counted. Circularity was calculated by ImageJ as $C=4\pi*(area/perimeter^2)$. The image conversion program and the ImageJ macro were combined into a Windows batch script, in which a for-loop was used to run the ImageJ macro over all the images quickly in sequence. It took approximately 5 min to fully process 10 images. All software for imaging processing are available in our Github repository (see **COMPUTATIONAL RESOURCES**).

Seed Image Processing and Counting with Faster R-CNN

Each scanned image was first split into 12 sub images; each sub image contains a single plate lid and is referred to as a "whole-plate image". Then, in the initial Faster R-CNN modeling trial, each whole-plate image was split into four quarter-plate images. Seeds on the quarter-plate images were manually annotated using LabelImg v1.8.1 (Tzutalin, 2015), and the annotated coordinates of seeds were first saved in extensible markup language (xml) format, and then converted to comma-separated values (csv) format. The csv files contained the coordinates of the bottom-left (x_{min}, y_{min}) and upper-right (x_{max}, y_{max}) corners in each of the annotated seed areas, which were used as ground truth annotations. For images used for model training, the csv files were further converted to the TFrecord format.

Faster R-CNN (Ren et al., 2017), which was performed using Tensorflow object detection API (Huang et al., 2017) and implemented in Tensorflow v1.13.2 (Abadi et al., 2016) in python v3.6.4, was used for object detection. Faster R-CNN combines the generation of region proposals (i.e., circumscribing the areas of interest, a regression problem) and their classification (i.e., in our case, the object is a seed or not) into a single pipeline (Ren et al., 2017). The architecture of Faster R-CNN used in Ren et al. (2017) has two stages: 1) construction of a Region Proposal Network and 2) establishment of a box classifier, Fast R-CNN, which was adapted from (Girshick, 2015) (**Figure S2**). In the first stage (left panel in **Figure S2**), images were processed by a feature extractor (Inception V2, Szegedy et al., 2016), and the resulting feature maps were used to predict bounding boxes containing images of individual seeds (referred to as proposals). In the second stage (right panel in **Figure S2**), these proposals were used to crop features from the feature maps; these cropped features were subsequently used for classification and bounding box regression.

To speed up the training process, a pre-trained model (faster_rcnn_inception_v2_coco) was used as a starting point. To optimize the Arabidopsis seed detection, tuning was conducted for three hyperparameters: proposal, aspect ratio, and scale (**Figure 3A** and **Figure S2**). For the

hyperparameter space tested see **Table S3.** Each model was run on three graphics processing units (GPUs) for 40,000 steps using the Adam optimizer, with a batch size of five (five random images were used to train the model for each step) and a learning rate of 0.0002. A model was saved every 10 minutes during the run as a model checkpoint.

To evaluate the model performance for hyperparameter tuning, validation set images—seeds in these images were manually annotated but were not used in the model training—were fed into the frozen models in jpeg format, and the outputs of model prediction were files in csv format containing the coordinates of the bottom-left and upper-right corners in each of the predicted seed areas. Coordinates of ground truth annotations in validation set images were compared with those of predicted seed areas using the measure IoU, which is defined as the intersection (I) of a ground truth area and a prediction area over (o) the union (U) of the same ground truth and prediction areas. A seed was regarded as correctly detected if its IoU was ≥ 0.5. An IoU < 0.5 was considered to be a misprediction. Then the F-measure (F1) score was calculated as a measurement of performance for a model as follows: F1 = $\frac{2*precision*recall}{precision+recall}$, where precision = $\frac{TP}{TP+FP}$; recall = $\frac{TP}{TP+FN}$. TP (true positive) is the number of correctly detected seeds, and FP (false positive) is the sum of the number of predicted seed areas that contain no seeds and the number of predicted seed areas minus 1 (if the ground truth area was detected more than once). FN (false negative) is the number of seeds with a maximum IoU < 0.5 with any predicted areas (i.e., the ground truth seed area was not detected).

Fruit Image Capturing and Counting with Faster R-CNN

For capturing fruit images, each dry Arabidopsis plant was put on a pink paper background and was photographed with an iPhone 8 smartphone. The images were saved in jpeg format with dimensions of 3024 X 4032 pixels. Fruits in the images were manually annotated, and the annotated coordinates were then converted to the csv and TFrecord formats, as conducted for the seed images.

The same pre-trained Faster R-CNN model used for seed counting was used to build the fruit counting models, and the same three hyperparameters were tuned to optimize the model performance but with a different hyperparameter space (**Table S6**). For each hyperparameter combination, a model was saved after 6000 steps, when the performance had converged. A final model was established using hyperparameters selected based on performance on the validation set images.

FUNDING

This work was supported by the U.S. Department of Energy Great Lakes Bioenergy Research Center (BER DE-SC0018409) and National Science Foundation DGE-1828149 to S.-H.S.; IOS-2107215 to M.D.L. and S.-H.S.; and DEB-1655386 to J.C. and S.-H. S.

AUTHOR CONTRIBUTIONS

P.W., F.M., M.D.L., and S.-H.S conceived and designed the study. P.W., F.M., P.D., S.H., N.L.P., E.V., E.W., J.K.C., and M.D.L. performed the analysis. P.W., F.M., M.D.L., and S.-H.S wrote the manuscript. All authors read and approved the final manuscript.

ACKNOWLEDGEMENTS

We thank Christina B. Azodi, Bethany M. Moore, Siobhan Cusack, and Liang Xu for help in manual seed annotation, and thank Ally Schumacher for providing the photo of the template. We thank Dirk Colbry for helpful discussions.

COMPUTATIONAL RESOURCES

All the scripts used in this study and the final seed and fruit counting models are available on Github at:
https://github.com/ShiuLab/Manuscript_Code/tree/master/2021_Arabidopsis_seed_and_fruit_count

REFERENCES

Abadi, M., Barham, P., Chen, J.M., Chen, Z.F., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M.*, et al.* (2016). TensorFlow: A system for large-scale machine learning. In Proc. USENIX Symp. Oper. Syst. Des. Implement. (OSDI):265-283.

Afonso, M., Fonteijn, H., Fiorentin, F.S., Lensink, D., Mooij, M., Faber, N., Polder, G., and Wehrens, R. (2020). Tomato fruit detection and counting in greenhouses using deep learning. Front. Plant Sci. **11**:571299.

Ågren, J., Oakley, C.G., McKay, J.K., Lovell, J.T., and Schemske, D.W. (2013). Genetic mapping of adaptation reveals fitness tradeoffs in *Arabidopsis thaliana*. Proc. Natl. Acad. Sci. USA **110**:21077-21082.

Bouché, N., and Bouchez, D. (2001). Arabidopsis gene knockout: phenotypes wanted. Curr. Opin. Plant Biol. **4**:111-117.

Boyes, D.C., Zayed, A.M., Ascenzi, R., McCaskill, A.J., Hoffman, N.E., Davis, K.R., and Gorlach, J. (2001). Growth stage-based phenotypic analysis of *Arabidopsis*: A model for high throughput functional genomics in plants. Plant Cell **13**:1499-1510.

Busoms, S., Teres, J., Huang, X.Y., Bomblies, K., Danku, J., Douglas, A., Weigel, D., Poschenrieder, C., and Salt, D.E. (2015). Salinity Is an agent of divergent selection driving local adaptation of *Arabidopsis* to coastal habitats. Plant Physiol. **168**:915-929.

Cao, C.Q., Wang, B., Zhang, W.R., Zeng, X.D., Yan, X., Feng, Z.J., Liu, Y.T., and Wu, Z.Y. (2019). An improved Faster R-CNN for small object detection. In IEEE Access:106838-106846. https://doi.org/10.1109/Access.2019.2932731.

Cervantes, E., Martin, J.J., and Saadaoui, E. (2016). Updated methods for seed shape analysis. Scientifica (Cairo) **2016**:5691825.

Conner, J.K., and Rush, S. (1997). Measurements of selection on floral traits in black mustard, *Brassica nigra*. J. Evol. Biol. **10**:327-335.

Cvetkovic, J., Muller, K., and Baier, M. (2017). The effect of cold priming on the fitness of *Arabidopsis thaliana* accessions under natural and controlled conditions. Sci. Rep. **7**:44055.

Fournier-Level, A., Korte, A., Cooper, M.D., Nordborg, M., Schmitt, J., and Wilczek, A.M. (2011). A map of local adaptation in *Arabidopsis thaliana*. Science **334**:86-89.

Ganeteg, U., Külheim, C., Andersson, J., and Jansson, S. (2004). Is each light-harvesting complex protein important for plant fitness? Plant Physiol. **134**:502-509.

Girshick, R. (2015). Fast R-CNN. In Proc. IEEE Int. Conf. Comput. Vis.:1440-1448. https://doi.org/10.1109/iccv.2015.169.

Gnan, S., Priest, A., and Kover, P.X. (2014). The genetic basis of natural variation in seed size and seed number and their trade-off using *Arabidopsis thaliana* MAGIC lines. Genetics **198**:1751-1758.

Halcro, K., McNabb, K., Lockinger, A., Socquet-Juglard, D., Bett, K.E., and Noble, S.D. (2020). The BELT and phenoSEED platforms: shape and colour phenotyping of seed samples. Plant Methods **16**:49.

Hamidinekoo, A., Garzon-Martinez, G.A., Ghahremani, M., Corke, F.M.K., Zwiggelaar, R., Doonan, J.H., and Lu, C. (2020). DeepPod: a convolutional neural network based quantification of fruit number in *Arabidopsis*. Gigascience **9**:giaa012.

Hasan, M.M., Chopin, J.P., Laga, H., and Miklavcic, S.J. (2018). Detection and analysis of wheat spikes using Convolutional Neural Networks. Plant Methods **14**:100.

He, K.M., Zhang, X.Y., Ren, S.Q., and Sun, J. (2016). Deep residual learning for image recognition. In Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.:770-778. https://doi.org/10.1109/Cvpr.2016.90.

Herridge, R.P., Day, R.C., Baldwin, S., and Macknight, R.C. (2011). Rapid analysis of seed size in *Arabidopsis* for mutant and QTL discovery. Plant Methods **7**:3.

Hirsch, R.E., Lewis, B.D., Spalding, E.P., and Sussman, M.R. (1998). A role for the AKT1 potassium channel in plant nutrition. Science **280**:918-921.

Huang, J., Rathod, V., Sun, C., Zhu, M.L., Korattikara, A., Fathi, A., Fischer, I., Wojna, Z., Song, Y., Guadarrama, S.*, et al.* (2017). Speed/accuracy trade-

offs for modern convolutional object detectors. In Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.:3296-3297. https://doi.org/10.1109/cvpr.2017.351.

Jahnke, S., Roussel, J., Hombach, T., Kochs, J., Fischbach, A., Huber, G., and Scharr, H. (2016). phenoSeeder - A robot system for automated handling and phenotyping of individual seeds. Plant Physiol. **172**:1358-1370.

Jiang, Y., Li, C.Y., Paterson, A.H., and Robertson, J.S. (2019). DeepSeedling: deep convolutional network and Kalman filter for plant seedling detection and counting in the field. Plant Methods **15**:141.

Kerwin, R., Feusier, J., Corwin, J., Rubin, M., Lin, C., Muok, A., Larson, B., Li, B., Joseph, B., Francisco, M.*, et al.* (2015). Natural genetic variation in *Arabidopsis thaliana* defense metabolism genes modulates field fitness. eLife **4**:e05604.

Külheim, C., Agren, J., and Jansson, S. (2002). Rapid regulation of light harvesting and plant fitness in the field. Science **297**:91-93.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., and Berg, A.C. (2016). SSD: Single Shot MultiBox Detector. In Computer Vision (Eccv):21-37. https://doi.org/10.1007/978-3-319-46448-0_2.

Mauricio, R., and Rausher, M.D. (1997). Experimental manipulation of putative selective agents provides evidence for the role of natural enemies in the evolution of plant defense. Evolution **51**:1435-1444.

Mcdonald, J.F. (1983). The Molecular-Basis of Adaptation - a Critical-Review of Relevant Ideas and Observations. Annu. Rev. Ecol. Syst. **14**:77-102.

Meissner, R.C., Jin, H.L., Cominelli, E., Denekamp, M., Fuertes, A., Greco, R., Kranz, H.D., Penfield, S., Petroni, K., Urzainqui, A.*, et al.* (1999). Function search in a large transcription factor gene family in Arabidopsis: Assessing the potential of reverse genetics to identify insertional mutations in R2R3 *MYB* genes. Plant Cell **11**:1827-1840.

Mochida, K., Koda, S., Inoue, K., Hirayama, T., Tanaka, S., Nishii, R., and Melgani, F. (2019). Computer vision-based phenotyping for improvement of plant productivity: a machine learning perspective. Gigascience **8**:giy153.

Moore, C.R., Johnson, L.S., Kwak, I.Y., Livny, M., Broman, K.W., and Spalding, E.P. (2013). High-throughput computer vision introduces the time axis to a quantitative trait map of a plant growth response. Genetics **195**:1077-1086.

Morales, A., Teapal, J., Ammerlaan, J.M.H., Yin, X., Evers, J.B., Anten, N.P.R., Sasidharan, R., and van Zanten, M. (2020). A high throughput method for quantifying number and size distribution of *Arabidopsis* seeds using large particle flow cytometry. Plant Methods **16**:27.

Penaloza, P., Ramirez-Rosales, G., McDonald, M.B., and Bennett, M.A. (2005). Lettuce (*Lactuca sativa* L.) seed quality evaluation using seed physical attributes, saturated salt accelerated aging and the seed vigour imaging system. Electron. J. Biotechnol. **8**:299-307.

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). You Only Look Once: unified, real-time object detection. In Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.:779-788. https://doi.org/10.1109/Cvpr.2016.91.

Ren, S.Q., He, K.M., Girshick, R., and Sun, J. (2017). Faster R-CNN: towards real-time object detection with Region Proposal Networks. In IEEE Trans. Pattern Anal. Mach. Intell.:1137-1149. https://doi.org/10.1109/TPAMI.2016.2577031.

Roux, F., Gasquez, J., and Reboud, X. (2004). The dominance of the herbicide resistance cost in several *Arabidopsis thaliana* mutant lines. Genetics **166**:449-460.

Schneider, C.A., Rasband, W.S., and Eliceiri, K.W. (2012). NIH Image to ImageJ: 25 years of image analysis. Nat. Methods **9**:671-675.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.:2818-2826. https://doi.org/10.1109/Cvpr.2016.308.

Tanabata, T., Shibaya, T., Hori, K., Ebana, K., and Yano, M. (2012). SmartGrain: high-throughput phenotyping software for measuring seed shape through image analysis. Plant Physiol. **160**:1871-1880.

Taylor, M.A., Wilczek, A.M., Roe, J.L., Welch, S.M., Runcie, D.E., Cooper, M.D., and Schmitt, J. (2019). Large-effect flowering time mutations reveal

conditionally adaptive paths through fitness landscapes in *Arabidopsis thaliana*. Proc. Natl. Acad. Sci. USA **116**:17890-17899.

Thomson, C.E., and Hadfield, J.D. (2017). Measuring selection when parents and offspring interact. Methods Ecol. Evol. **8**:678-687.

Toda, Y., Okura, F., Ito, J., Okada, S., Kinoshita, T., Tsuji, H., and Saisho, D. (2020). Training instance segmentation neural network with synthetic datasets for crop seed phenotyping. Commun. Biol. **3**:173.

Tzutalin (2015). LabelImg. Git code. https://github.com/tzutalin/labelImg.

Vasseur, F., Bresson, J., Wang, G., Schwab, R., and Weigel, D. (2018). Image-based methods for phenotyping growth dynamics and fitness components in *Arabidopsis thaliana*. Plant Methods **14**:63.

Wang, Q.M., Qi, F., Sun, M.H., Qu, J.H., and Xue, J. (2019). Identification of tomato disease types and detection of infected areas based on deep convolutional neural networks and object detection techniques. Comput. Intell. Neurosci. **2019**:9142753.

Yang, Q., Zhang, Y., Dai, W., and Pan, S. (2020). Transfer Learning (Cambridge: Cambridge University Press).

FIGURE LEGENDS

Figure 1. Workflow and Performance for Seed Counting Using ImageJ When Seeds Were Deliberately Separated.

(**A**) Workflow. Seeds from 12 different plants were scattered and manually separated from each other on the lids of 12 petri plate lids, which were placed in a template and scanned. Twelve search areas, each with a diameter of 60 mm (yellow circles), were predefined. A threshold was applied by selecting pixels with intensities between 50 and 140 to separate the seed areas (red) from the background. Then pixels were converted to real-world distance units in mm. The “Analyze Particles” tool was used to detect and count the seeds.

(**B**) An example of an image with detected seeds (left) and an enlarged image showing the seeds (right). Red region with number: individual detected seed area. (**C**) Correlation between true and predicted seed counts using ImageJ when seeds were deliberately separated.

Figure 2. Comparison Between the Performances of ImageJ and Faster R-CNN-based Seed Counting for the Test Set Images of Seeds that Were Not Deliberately Separated.

(A, B) The same seed scan image analyzed by ImageJ (**A**) and Faster R-CNN (**B**). Three different regions of the plate lid with different densities are outlined. Region 1 has low seed density, region 2 has moderate density, and region 3 has a high density. In (**A**) the red colored regions represent the segmented areas identified by ImageJ; seeds outlined in yellow and assigned numeric IDs were counted. In (**B**) the blue rectangles represent seeds detected by Faster R-CNN.

(**C,D**) Correlation between true and predicted seed numbers from ImageJ (**C**) and Faster R-CNN (**D**) analysis of the test set.

(**E**) Distribution of differences between true and predicted seed numbers. Red lines: ImageJ; blue lines: Faster R-CNN.

(**F**) Correlation between seed density index (SDI) and difference between true and predicted seed counts.

Each dot in (**C,D,F**) corresponds to one of the 50 test set images. The red line in (**C**) is the regression line obtained using the loess method. The blue lines in (**D,F**) are fitted regression lines for Faster R-CNN predictions. The red line in (**F**) is the fitted linear regression line for ImageJ-based predictions. PCC: Pearson correlation coefficient.

Figure 3. Workflow for Building Faster R-CNN-based Seed Counting Models.

(**A**) First-round modeling for enriching annotated seed labels. Each of the 256 whole-plate images was split into four quarter-plate images. Among the 1024 quarter-plate images, 180 were used in first-round modeling, and the remainder (844) were used in second-round modeling described in (**B**). Seeds in the 180 quarter-plate images were manually annotated, and then these annotated

images were further split into training set 1 (160) and a validation set (20) to train and evaluate models, respectively. Sixty-three combinations of three hyperparameters (i.e., 3 proposal numbers × 3 scales [A, B, and C] × 7 aspect ratios [AR-A through G]; for scale and aspect ratio values see **Table S3**) were used to build 63 models. The optimal scale (B) and aspect ratio (AR-A) were selected based on the model performance on validation set images (**Figure S3**). An additional three models ($Model_{seed}$ 64-66) were built using three different proposal values, and the final best model, $Model_{seed}$ 66, with 10,000 proposals, was applied to the 844 quarter-plate images reserved for second-round modeling to generate *in silico* seed annotations.

(**B**) Second-round modeling. The 844 quarter-plate images with seed predictions from $Model_{seed}$ 66 were rejoined together to reconstruct 211 whole-plate images with *in silico* seed annotations, which were then manually curated and used as ground truth seed annotations. $Model_{seed}$ 67 was built using 161 (training set 2) out of the 211 annotated images with the same hyperparameters used in $Model_{seed}$ 66, and was evaluated using the test set (50 independent images) and the modified test set (i.e., 50 test set images not used for modeling plus 1,700 images modified from the test set images that had different image properties [blurriness, brightness, contrast, and resolution values]). For data augmentation, 20 images from training set 2 with different image properties were combined with training set 2, resulting in 581 images (modified training set 2), which were used to build $Model_{seed}$ 68. The modified test set was used to evaluate the performance of $Model_{seed}$ 68.

Figure 4. Effect of Seed Density on the Performance of the Faster R-CNN Models.

(**A**) Examples with different seed density index (SDI) values. The radius of each circle is 30 pixels (0.62 mm).

(**B,C**) Relationship between SDI and model performance (**B**) and between the true SDI and SDI based on prediction (**C**) for test images. Each dot corresponds

to one of 50 test set images. Blue lines are the fitted linear regression lines. F1: F1 value at 0.5 IoU (Intersection over Union).

Figure 5. Improvement of Model Robustness Using Training Images with Different Properties.

(**A**) Examples of seed images with different relative brightness, contrast, blurriness, and resolution values that were derived from the same original image.

(**B**) Model performance for $Model_{seed}$ 67 and $Model_{seed}$ 68 on the modified test set (**Figure 2B**). Red boxplot: $Model_{seed}$ 67; blue boxplot: $Model_{seed}$ 68. F1: F1 value at 0.5 IoU.

Figure 6. Fruit Counting Using Faster R-CNN Models.

(A) Fruit counting workflow.

(**B**) Relationship between true and predicted fruit numbers.

(**C**) Relationship between fruit number in an image and the model performance.

(**D**) Examples of the same fruit image with different relative brightness, contrast, blurriness, and resolution values.

(**E**) Model performance for $Model_{fruit}$ 21 and $Model_{fruit}$ 76 on test images with different properties. Red boxplot: $Model_{fruit}$ 21; blue boxplot: $Model_{fruit}$ 76. F1: F1 at 0.5 IoU.

PCC: Pearson correlation coefficient.

SUPPLEMENTAL INFORMATION

Figure S1. Example False Negatives from ImageJ Analysis and Faster R-CNN Models.

(**A-B**) One example seed scan image analyzed by ImageJ (**A**) and the Faster R-CNN Model 67 (**B**). Purple arrowheads: example false negatives from ImageJ and Faster R-CNN; green arrowheads: seeds correctly detected by Faster R-CNN.

Figure S2. The Architecture of Faster R-CNN.

The seed images are first processed using a feature extractor (Inception V2), which extracts features from the input images by assigning importance (weights or biases) to the objects in the images. The output of the feature extractor is a feature map, indicating the locations and strength (indicated by color gradient) of the detected features in an image. Then a large number of anchors (rectangles with different aspect ratios [width/height] and scales [relative size]) are generated and placed uniformly throughout the feature map. By applying one of the key modules of Faster R-CNN, Regional Proposal Network, each anchor is assigned an objectness score, which is an indication of how likely it is that the anchor contains an object. A predefined number of anchors (object proposals) are selected based on the rank of objectness scores. Next, to determine whether an anchor contains an object and to adjust the anchors to better fit the location of the seed, the feature maps and the proposals are processed using another module, the Fast R-CNN Detector. Two scores are obtained: a classification score (the likelihood that the proposal region contains a seed) and regression score (the location of the detected seed)**.**

Figure S3. Hyperparameter Tuning for Seed Counting Models.

(**A-C**) Performance of models trained on training set 1 with 100 (**A**), 500 (**B**), and 1,000 (**C**) proposals at different scales (columns) and with different aspect ratios (colored lines). Performance was evaluated using the validation set. For scale and aspect ratio values see **Table S3.**

(**D**) Model performance for different proposals based on the scale-B and aspect ratio-A combination.

X axis: the number of training steps; y axis: F1 at 0.5 IoU.

Figure S4. Computational Efficiency of Seed Counting Models.

Computational efficiency of seed counting models trained with training set 1, with 100 (**A**), 500 (**B**), and 1,000 proposals (**C**) at different scales (A, B, and C columns) and with different aspect ratios (colored lines).

X axis: the number of training steps; y axis: global steps per second.

Figure S5. Example Images with Different SDI Values.

Six seed images with SDI values ranging from low (1.157) to high (3.100).

Figure S6. Effect of Seed Density on the Performance of the Faster R-CNN Models Using Different Measures of Performance.

(**A-C**) Relationship between SDI and precision (**A**), recall (**B**), and accuracy (**C**). Each blue dot represents one of the 50 test set images, and lines are the fitted linear regression lines.

Figure S7. Hyperparameter Tuning for Fruit Counting Models.

Performance of fruit counting models trained on images in the training set with different proposal numbers (rows) at different scales (columns) and with different aspect ratios (colored lines). Performance was evaluated using images in the validation set. For scale and aspect ratio values see **Table S6.** x axis: the number of training steps; y axis: F1 at 0.5 IoU.

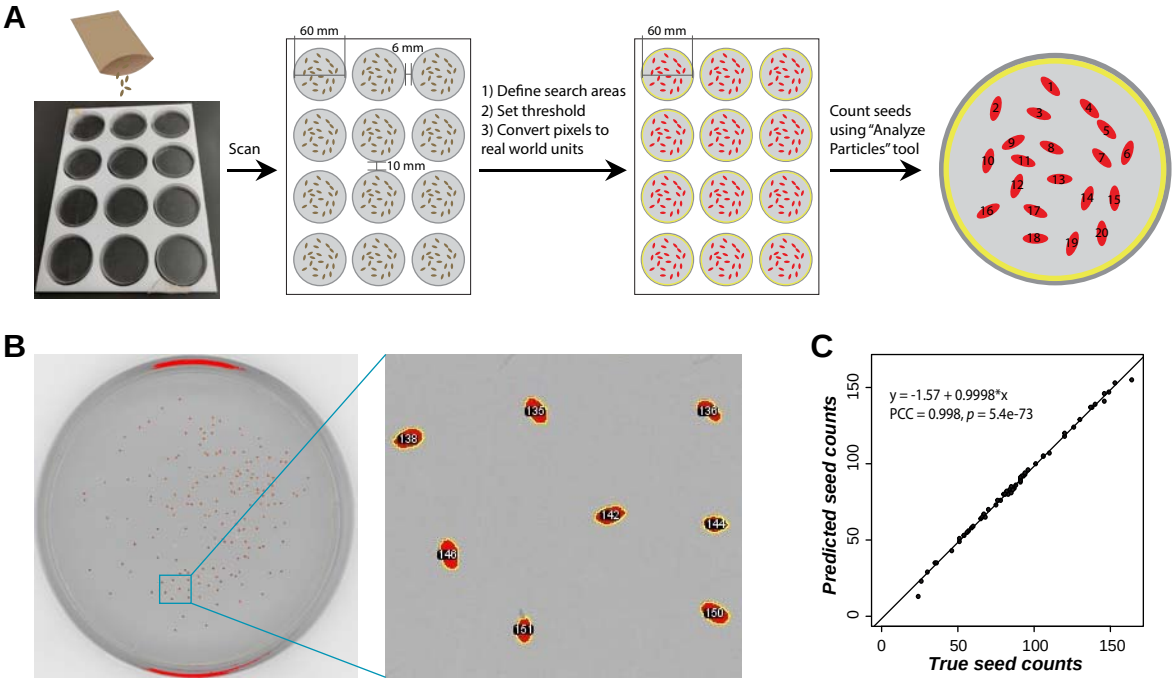

Figure 1. Workflow and Performance for Seed Counting Using ImageJ When Seeds Were Deliberately Separated.

(A) Workflow. Seeds from 12 different plants were scattered and manually separated from each other on the lids of 12 petri plate lids, which were placed in a template and scanned. Twelve search areas, each with a diameter of 60 mm (yellow circles), were predefined. A threshold was applied by selecting pixels with intensities between 50 and 140 to separate the seed areas (red) from the background. Then pixels were converted to real-world distance units in mm. The "Analyze Particles" tool was used to detect and count the seeds.

(B) An example of an image with detected seeds (left) and an enlarged image showing the seeds (right). Red region with number: individual detected seed area.

(C) Correlation between true and predicted seed counts using ImageJ when seeds were deliberately separated.

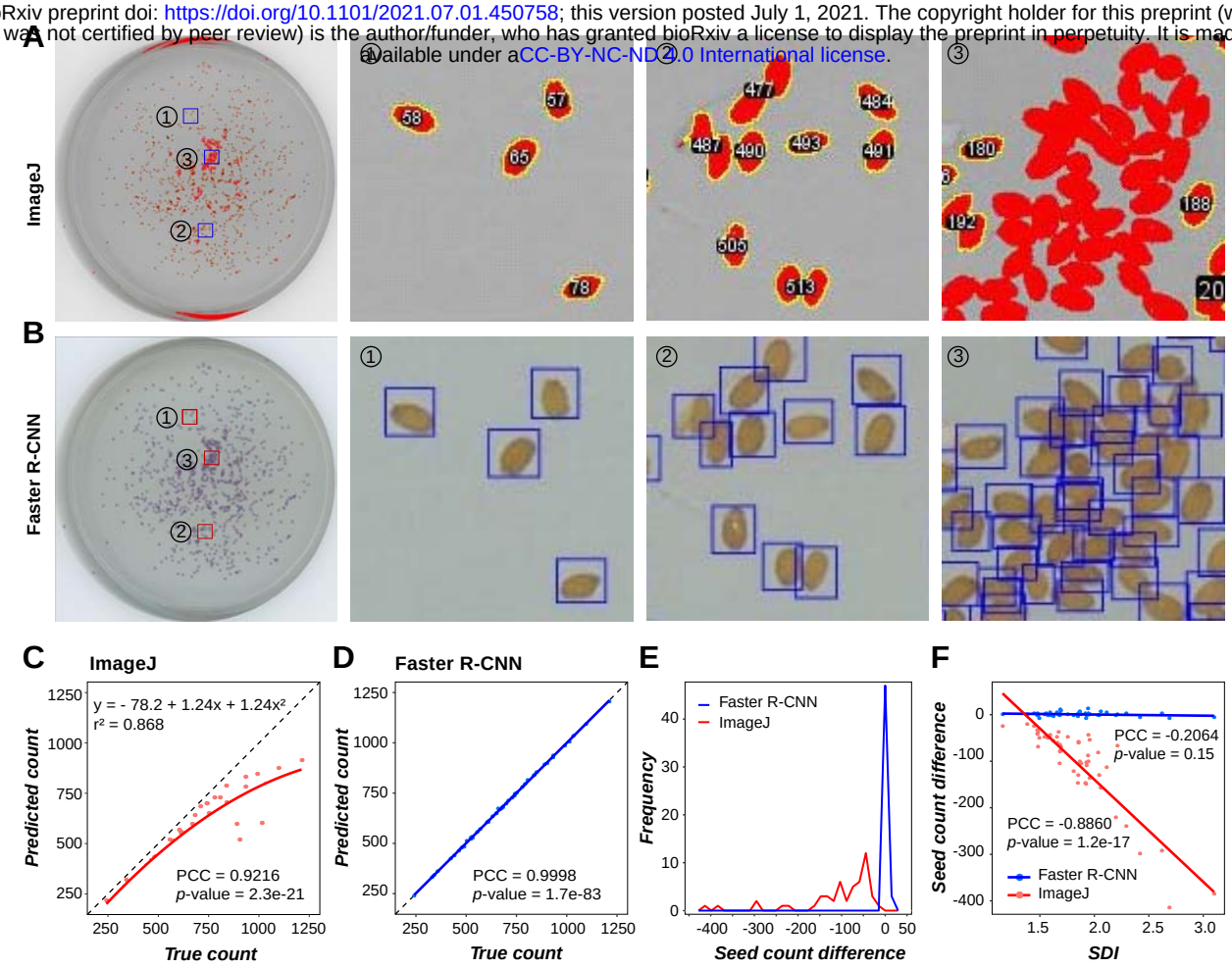

Figure 2. Comparison Between the Performances of ImageJ and Faster R-CNN-based Seed Counting for the Test Set Images of Seeds that Were Not Deliberately Separated.
(A, B) The same seed scan image analyzed by ImageJ **(A)** and Faster R-CNN **(B)**. Three different regions of the plate lid with different densities are outlined. Region 1 has low seed density, region 2 has moderate density, and region 3 has a high density. In (A) the red colored regions represent the segmented areas identified by ImageJ; seeds outlined in yellow and assigned numeric IDs were counted. In **(B)** the blue rectangles represent seeds detected by Faster R-CNN.
(C,D) Correlation between true and predicted seed numbers from ImageJ **(C)** and Faster R-CNN **(D)** analysis of the test set.
(E) Distribution of differences between true and predicted seed numbers. Red lines: ImageJ; blue lines: Faster R-CNN.
(F) Correlation between seed density index (SDI) and difference between true and predicted seed counts.
Each dot in **(C,D,F)** corresponds to one of the 50 test set images. The red line in **(C)** is the regression line obtained using the loess method. The blue lines in **(D,F)** are fitted regression lines for Faster R-CNN predictions. The red line in **(F)** is the fitted linear regression line for ImageJ-based predictions. PCC: Pearson correlation coefficient.

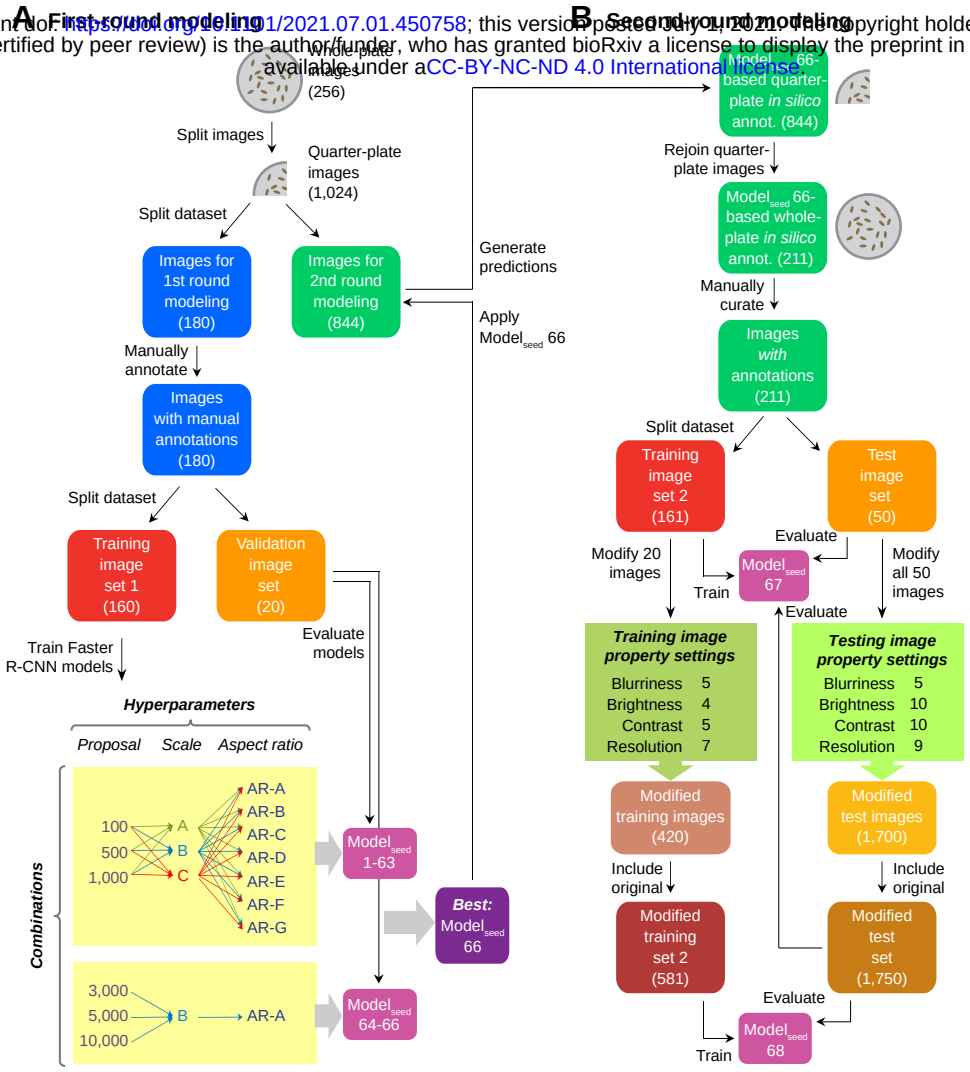

Figure 3. Workflow for Building Faster R-CNN-based Seed Counting Models.
(A) First-round modeling for enriching annotated seed labels. Each of the 256 whole-plate images was split into four quarter-plate images. Among the 1024 quarter-plate images, 180 were used in first-round modeling, and the remainder (844) were used in second-round modeling described in **(B).** Seeds in the 180 quarter-plate images were manually annotated, and then these annotated images were further split into training set 1 (160) and a validation set (20) to train and evaluate models, respectively. Sixty-three combinations of three hyperparameters (i.e., 3 proposal numbers x 3 scales [A, B, and C] x 7 aspect ratios [AR-A through G]; for scale and aspect ratio values see Table S1) were used to build 63 models. The optimal scale (B) and aspect ratio (AR-A) were selected based on the model performance on validation set images (**Figure S3**). An additional three models ($Model_{seed}$ 64-66) were built using three different proposal values, and the final best model, $Model_{seed}$ 66, with 10,000 proposals, was applied to the 844 quarter-plate images reserved for second-round modeling to generate in silico seed annotations. **(B)** Second-round modeling. The 844 quarter-plate images with seed predictions from $Modelseed_{66}$ were rejoined together to reconstruct 211 whole-plate images with in silico seed annotations, which were then manually curated and used as ground truth seed annotations. $Model_{seed}$ 67 was built using 161 (training set 2) out of the 211 annotated images with the same hyperparameters used in $Model_{seed}$ 66, and was evaluated using the test set (50 independent images) and the modified test set (i.e., 50 test set images not used for modeling plus 1,700 images modified from the test set images that had different image properties [blurriness, brightness, contrast, and resolution values]). For data augmentation, 20 images from training set 2 with different image properties were combined with training set 2, resulting in 581 images (modified training set 2), which were used to build $Model_{seed}$ 68. The modified test set was used to evaluate the performance of $Model_{seed}$ 68.

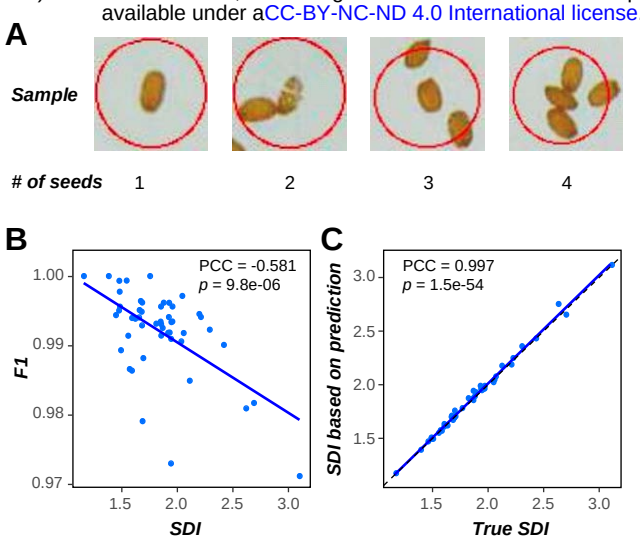

Figure 4. Effect of Seed Density on the Performance of the Faster R-CNN Models.
(A) Examples with different seed density index (SDI) values. The radius of each circle is 30 pixels (0.62 mm).
(B,C) Relationship between SDI and model performance **(B)** and between the true SDI and SDI based on prediction **(C)** for test images. Each dot corresponds to one of 50 test set images. Blue lines are the fitted linear regression lines. F1: F1 value at 0.5 IoU (Intersection over Union).

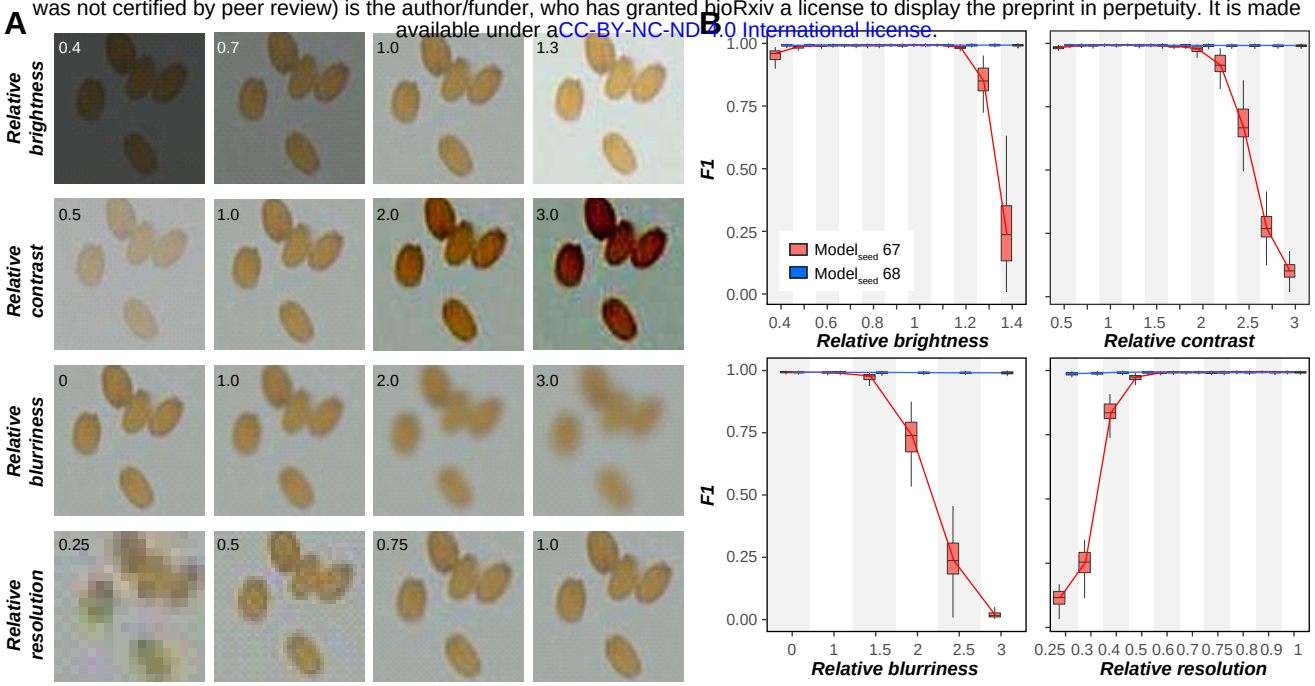

Figure 5. Improvement of Model Robustness Using Training Images with Different Properties.
(A) Examples of seed images with different relative brightness, contrast, blurriness, and resolution values that were derived from the same original image.
(B) Model performance for $Model_{seed}$ 67 and $Model_{seed}$ 68 on the modified test set **(Figure 2B)**. Red boxplot: $Model_{seed}$ 67; blue boxplot: $Model_{seed}$ 68. F1: F1 value at 0.5 IoU.

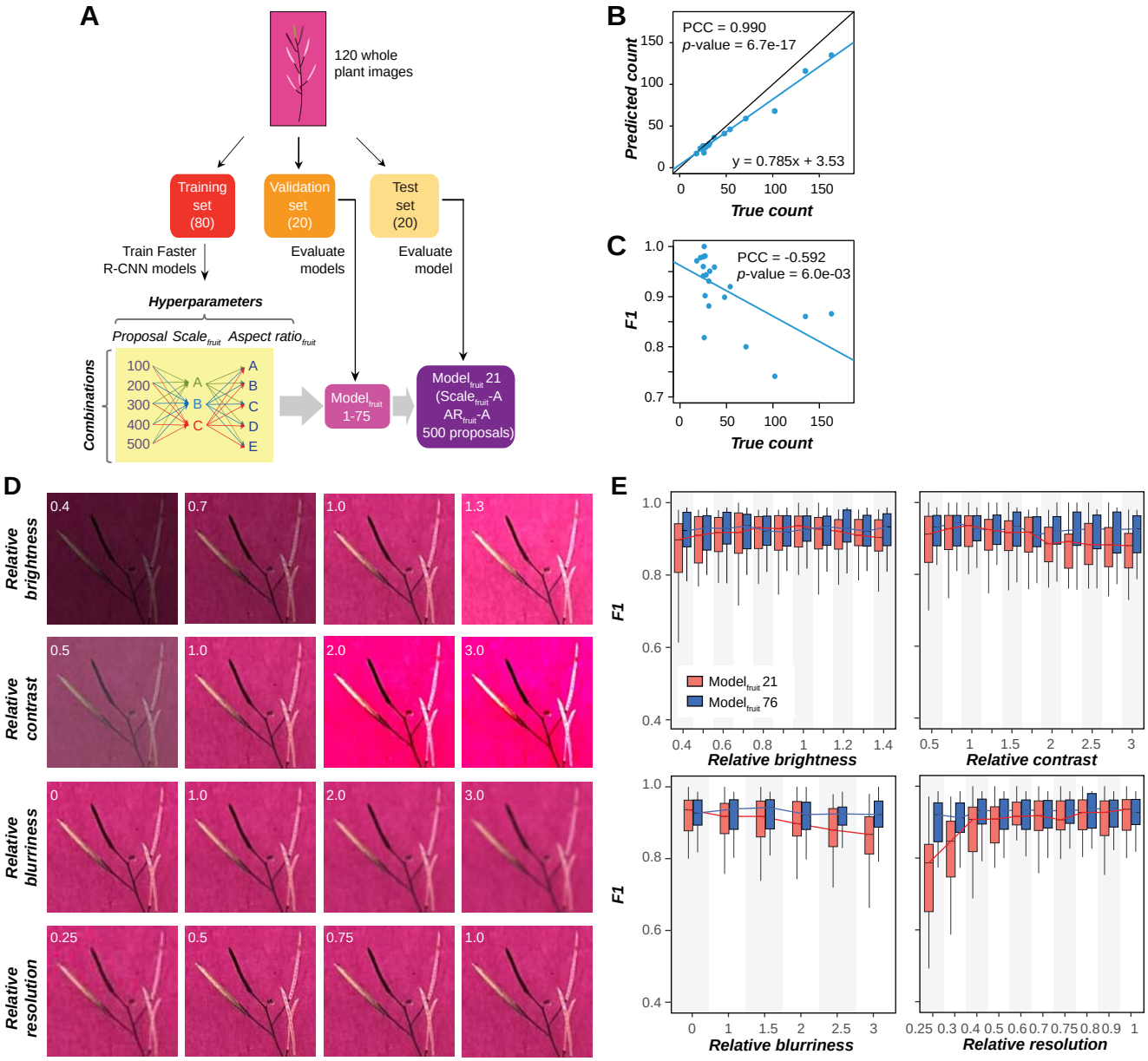

Figure 6. Fruit Counting Using Faster R-CNN Models.

(A) Fruit counting workflow.

(B) Relationship between true and predicted fruit numbers.

(C) Relationship between fruit number in an image and the model performance.

(D) Examples of the same fruit image with different relative brightness, contrast, blurriness, and resolution values.

(E) Model performance for $Model_{fruit}$ 21 and $Model_{fruit}$ 76 on test images with different properties. Red boxplot: $Model_{fruit}$ 21; blue boxplot: $Model_{fruit}$ 76. F1: F1 at 0.5 IoU.

PCC: Pearson correlation coefficient.