Methods S1

Plant growth conditions

For plants grown for fitness analysis (Table S3) and for obtaining seed scan images for training seed counting models (Table S1), seeds were resuspended in 0.1% agarose, and five seeds from a single genotype were pipetted into a single cell of a 200-cell flat filled with Arabidopsis mix (1:1:1 SureMix, vermiculite, and perlite). Four flats were placed together to form a single block, and genotypes were randomly assigned to cells within the block. Randomization was performed separately for the border cells (the outer two rows surrounding the four flats), to allow them to be excluded from analysis if edge effects were observed. The genotypes used for fitness analysis were grown in two separate blocks: the *pap*, *hon*, and *eif4b* mutants and their wild-type (WT) controls were grown together in one block, and the *aprl*, *pfa-dsp*, and *kin7* mutants and their WT controls were grown in another. After stratification for 5–7 days in the dark at 4°C, the flats were transferred to a growth chamber and grown under a 16-h light/8-h dark cycle with a light intensity of 110–130 μmoles/m²/s at 21°C. Seedlings were thinned to one per cell after 1 week. Plants were watered two to three times per week until most plants had undergone global arrest. When plants were completely dry, the number of intact and completely or partially shattered fruits from each plant were counted by hand to obtain the total number of fruits per plant. All intact fruits were transferred to glassine envelopes and allowed to dry for at least one month before counting seeds. In some cases, fruits remained green after the plant had dried; these fruits contained seeds but did not complete development before water was withheld. These fruits were included in the number of fruits but were not collected for seed counts.

For plants grown for fruit imaging (Table S2), seeds were sown in 200-cell flats filled with MetroMix 360 (five seeds per cell), stratified for 7 days in the dark at 4°C, then grown under a
light intensity of ~150 μmoles/m²/s at 21°C. Genotypes were randomly assigned to cells within a block as described for the fitness analysis, except that the block consisted of 12 flats arranged in a 3×4 grid. Seedlings were thinned to one per cell after 1 week. 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, processing, and counting with the segmentation method using ImageJ

Prior to seed imaging, we separated the seeds from the chaff by passing seeds through a sieve and then picked out any remaining siliques with forceps. The time required for this step was typically 30–60 seconds per plant. Seeds were scattered on the lid of a 60-mm petri plate, which was then tapped with fingers to ensure that seeds did not cover each other. For some plate lids counted with the segmentation method using ImageJ, seeds were separated from each other and moved away from the lid edges using forceps. The lids were placed in a template made from white acrylic (295 mm × 210 mm × 10 mm, Fig. 1a) on a standard desktop scanner (Epson V600). This template was designed to be the same height as the plate lid to minimize artifacts resulting from light loss. Twelve 60-mm diameter holes (four rows × three columns) were cut into the template using a laser cutter. There was 6 mm between holes in each row and 10 mm in each column. 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 (described in the following paragraph). Seeds were scanned at 1200 dots per inch and in 24-bit color, and scans were saved as jpeg files.

The ImageJ (version 1.52a, https://imagej.nih.gov, Schneider et al., 2012) workflow is shown in Fig. 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). An ImageJ macro was written to automate seed counting. First, 12 circular search regions per scanned image were defined to confine the search to the 12 plate lids containing the seeds. 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 using the “Analyze Particles” tool (function of ImageJ). Only above-threshold regions
larger than 0.06 mm² and with a circularity value (calculated as $C = 4\pi \times \frac{\text{area}}{\text{perimeter}^2}$) of 0.25 to 1 were counted.

**Seed image processing and counting with an object detection method using 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”. We first experimented with multiple object detection algorithms with default settings, namely Faster R-CNN (Ren et al., 2017), Mask R-CNN (He et al., 2017), You Only Look Once (Redmon et al., 2016), and Single Shot MultiBox Detector (Liu et al., 2016). We also tested multiple pre-trained Faster R-CNN models (that were retrained for seed detection), namely Inception v.2 (Szegedy et al., 2016), ResNet50 (He et al., 2016), ResNet101 (He et al., 2016), and NASNet (Qin & Wang, 2020). In our preliminary run, we found that Faster R-CNN combined with Inception v.2 as the initial model was the most straightforward approach because, compared with the other algorithms, it was relatively easy to implement, best met the computational requirements in our institution, and yielded better performance before extensive hyperparameter tuning. Thus, in all subsequent analyses, we focused on Faster R-CNN and Inception v.2.

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). The ground truth annotations, i.e., the coordinates of the bottom-left ($x_{\text{min}}$, $y_{\text{min}}$) and upper-right ($x_{\text{max}}$, $y_{\text{max}}$) corners of each annotated seed area, were saved in extensible markup language (xml) format, and then converted to comma-separated values (csv) format. For images used for model training, the csv files were further converted to the TFrecord format. 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. These whole-plate seed coordinates 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.

To speed up the training process, a pre-trained model (faster_rcnn_inception_v2_coco) was used as a starting point. To optimize Arabidopsis seed detection, three hyperparameters were tested: 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 (Fig. 3a and Fig. S1, Ren
et al., 2017). For the hyperparameter space tested see Table S6. 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 not used in the model training—were fed into the frozen models in jpeg format, and model predictions were output in csv files containing the coordinates of the bottom-left and upper-right corners of each predicted seed area. 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) over (o) the union (U) of a ground truth area and a prediction area. A seed was regarded as correctly detected if its IoU was \(\geq 0.5\) and mispredicted if its IoU was \(< 0.5\). The F-measure (F1) score was calculated as a measure of performance for a model as follows: 

\[
F1 = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}},
\]

where precision = \(\frac{TP}{TP + FP}\) and 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 for any predicted area (i.e., the ground truth seed area was not detected).

For the hyperparameter tuning, we found that models with 100 proposals (Modelseed 1–21) had the least accuracy, with \(F1 < 0.750\) (Fig. S2a), because there were more than 100 seeds in most of the validation images (Table S7). In contrast, models with 500 (Modelseed 22–42) and 1000 proposals (Modelseed 43–63) had F1 values around 0.970 (Fig. S2b,c). 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 (Fig. S2b,c). The combination of scale-B and aspect ratio-A was used for downstream model building because the computational efficiency of this combination was the highest among all combinations (Fig. S3). Three additional models were established based on scale-B and aspect ratio-A with 3000, 5000, and 10,000 proposals (Modelseed 64–66), but F1 values did not improve over those of Modelseed 22–63 (Fig. S2d). Even though building Faster R-CNN models with a higher number of proposals requires more computational resources, we opted to use 10,000 proposals (Modelseed
66) for further model building to allow detection of a large number of seeds in an image.

References

He K, Gkioxari G, Dollar P, Girshick R. 2017. Mask R-CNN. In: 2017 IEEE International Conference on Computer Vision (ICCV). Venice: IEEE, 2980–2988.

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

Liu W, Anguelov D, Erhan D, Szegedy C, Reed S, Fu C-Y, Berg AC. 2016. SSD: Single Shot MultiBox Detector. In: Leibe B, Matas J, Sebe N, Welling M, eds. Lecture Notes in Computer Science. Computer Vision – ECCV 2016. Cham: Springer International Publishing, 21–37.

Qin X, Wang Z. 2020. NASNet: A Neuron Attention Stage-by-Stage Net for Single Image Deraining. \textit{arXiv:1912.03151 [cs, eess]}. 

Redmon J, Divvala S, Girshick R, Farhadi A. 2016. You Only Look Once: Unified, Real-Time Object Detection. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Las Vegas, NV, USA: IEEE, 779–788.

Ren S, He K, Girshick R, Sun J. 2017. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. \textit{IEEE Transactions on Pattern Analysis and Machine Intelligence} 39: 1137–1149.

Schneider CA, Rasband WS, Eliceiri KW. 2012. NIH Image to ImageJ: 25 years of image analysis. \textit{Nature Methods} 9: 671–675.

Szegedy C, Vanhoucke V, Ioffe S, Shlens J, Wojna Z. 2016. Rethinking the Inception Architecture for Computer Vision. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Las Vegas, NV, USA: IEEE, 2818–2826.

Tzutalin. 2015. \textit{LabellImg}. Git code. \url{https://github.com/tzutalin/labellImg}. [accessed 12 July 2021].

The following Supporting Information is available for this article:

Fig. S1 The architecture of Faster R-CNN

Fig. S2 Hyperparameter tuning for seed counting models

Fig. S3 Computational efficiency of seed counting models

Fig. S4 Example false negatives from the segmentation method using ImageJ and Faster R-CNN
models

**Fig. S5** Example images with different SDI values

**Fig. S6** Effect of seed density on the performance of the Faster R-CNN models using different measures of performance

**Fig. S7** Hyperparameter tuning for fruit counting models

**Fig. S8** Fitness measurements for T-DNA insertion mutants of 12 genes

**Fig. S9** The proportion of fruits produced by twelve mutants that are shattered or green

**Table S1.** Lines used for training seed counting models (see separate file)

**Table S2.** Lines used for training fruit counting models (see separate file)

**Table S3.** Lines used for analysis of fitness (see separate file)

**Table S4.** Fitness data for *pap2, pap9, hon4, hon5, eif4b1* and *eif4b2* (see separate file)

**Table S5.** Fitness data for *aprl5, aprl7, pfa-dsp3, pfa-dsp5, kin7.2* and *kin7.4* (see separate file)

**Table S6.** Hyperparameter space for seed counting (see separate file)

**Table S7.** Seed counts in 20 quarter-plate images in the validation set (see separate file)

**Table S8.** Hyperparameter space for fruit (silique) counting (see separate file)

**Table S9.** Seed counting with the segmentation method using ImageJ (see separate file)

**Table S10.** Seed counting for 50 test set seed images using Model\textsubscript{seed} 67 (see separate file)

**Table S11.** Image property settings for Model\textsubscript{seed} 68 and Model\textsubscript{fruit} 76 (see separate file)

**Table S12.** Fruit counting for 20 test fruit images using Model\textsubscript{fruit} 21 (see separate file)
**Fig. S1** The architecture of Faster R-CNN. The seed images are first processed using a feature extractor (Inception v.2), 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).
Fig. S2 Hyperparameter tuning for seed counting models. (a-c) Performance of 63 models (Model\textsubscript{seed} 1–63) trained on training set 1 with 100 (a), 500 (b), and 1,000 (c) proposals at three scales (columns) and with seven aspect ratios (colored lines, for scale and aspect ratio values see Table S6). Performance was evaluated using the validation set. (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 (Intersection over Union).
**Fig. S3** 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, shown in the left, middle, and right columns, respectively) and with different aspect ratios (colored lines). x axis: the number of training steps; y axis: global steps per second.
**Fig. S4** Example false negatives from the segmentation method using ImageJ 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. Yellow line in (a): boundary of circular search region; blue box in (b): detected seed area.
**Fig. S5** Example images with different SDI values. Six seed images with SDI values ranging from low (1.157) to high (3.100).
**Fig. 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.
Fig. 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 S8. x axis: the number of training steps; y axis: F1 at 0.5 IoU (Intersection over Union).
Fig. S8 Fitness measurements for T-DNA insertion mutants of 12 genes. (a-c) Seed counts for plants after removing plants with shattered fruits from analysis for *pap2* (a), *hon5* (b), and *kin7.4* (c). (d-z, α-δ) Fruit counts (d, g, j, m, p, s, v, y, β) and seed counts for plants before (e, h, k, n, q, t, w, z, γ) and after (f, i, l, o, r, u, x, α, δ) removing plants with shattered fruits from analysis for *pap9* (d-f), *hon4* (g-i), *kin7.2* (j-l), *eif4b1* (m-o), *eif4b2* (p-r), *aprl5* (s-u), *aprl7* (v-x), *pfa-dsp3* (y, z, α), and *pfa-dsp5* (β-δ). Values are shown for the mutants and the corresponding wild-type (WT) control. Sample sizes are shown in parentheses on the x axis. *p*-values are from Wilcoxon signed-rank tests. Horizontal line in the box: median value; box range: interquartile range (IQR), i.e., 25th (Q1) to 75th percentile (Q3); whisker below box: Q1 – 1.5*IQR to Q1; whisker above box: Q3 to Q3 + 1.5*IQR; violin plot: distribution of datapoint values; dot: datapoint from an individual plant; yellow: loss-of-function mutant; cyan: WT.
**Fig. S9.** The proportion of fruits produced by twelve mutants that are shattered or green. (a-l)
The number of shattered plus green fruits reported as the percentage of total fruit number for
*pap2* (a), *pap9* (b), *hon5* (c) *hon4* (d), *eif4b1* (e), *eif4b2* (f), *aprl5* (g), *aprl7* (h), *pfa-dsp3* (i), *pfa-dsp5* (j), *kin7.2* (k), and *kin7.4* (l). Values are shown for the mutants and the corresponding wild-type (WT) control. Sample sizes are shown in parentheses on the x axis. *p*-values are from Wilcoxon signed-rank tests. Horizontal line in the box: median value; box range: interquartile range (IQR), i.e., 25th (Q1) to 75th percentile (Q3); whisker below box: Q1 – 1.5*IQR to Q1; whisker above box: Q3 to Q3 + 1.5*IQR; violin plot: distribution of datapoint values; dot: datapoint from an individual plant; yellow: loss-of-function mutant; cyan: WT.
