Precision Agriculture (2025) 26:24
https://doi.org/10.1007/s11119-024-10208-y

Joint plant-spraypoint detector with ConvNeXt modules and HistMatch normalization

Jonathan Ford[1] · **Edmund Sadgrove**[1] · **David Paul**[1]

Accepted: 17 November 2024 / Published online: 22 January 2025
© The Author(s) 2025

Abstract

Context Serrated tussock (*Nassella trichotoma*) is a weed of national significance in Australia which offers little to no nutritional value to livestock, and has the potential to reduce carrying capacity and agricultural return of infested pastures.

Aims The aim of this study was to adapt existing Convolutional Neural Networks (CNNs) for plant segmentation and spraypoint detection in the challenging environments of pastures.

Methods CNNs that were designed for joint plant and stem segmentation in crop fields were repurposed for dual-task applications in pastures. Given the poor performance of these models in complex pasture environments, a new model drawing inspiration from the recently proposed ConvNeXt was developed, tested for its effectiveness on unseen field data, and enhanced with a novel normalization technique, called HistMatch.

Key results Experimentation demonstrated that unlike pre-existing models, which were designed for the simpler environments encountered in early-stage crop fields, our model was able to generalize well to growing conditions not seen during training, achieving 0.807 mIoU and 0.796 F1-score for the plant and spraypoint tasks respectively. This is in comparison to pre-existing models, which achieved 0.270 - 0.454 mIoU and 0.073 - 0.496 F1-score for the same tasks. These results were further improved to 0.854 mIoU and 0.806 F1-score using HistMatch normalization. In spite of greater model complexity, our model had a inference time of 15.7 ms which was comparable to pre-existing models, and suitable for real-time applications.

Conclusion Models with greater complexity are required for the relatively complex environments encountered in pastures, but this greater complexity need not come at the expense of real time capability. HistMatch normalization can improve model accuracy, and is particularly effective in cases where models are struggling to generalize well to testing conditions that vary significantly from those seen during training.

Implications and impacts The successful adaptation and improvement of CNNs for weed management in pastures could significantly reduce the reliance on blanket herbicide application. HistMatch normalization could also be considered for other agricultural applications, including weed management and disease detection in crop fields and orchards.

Keywords Convolutional Neural Networks · ConvNeXt · Computer vision · Site-specific weed management · Precision agriculture.

Extended author information available on the last page of the article

Introduction

The use of computer vision in agriculture is a promising area of research, which offers, among other things, the potential for effective detection and treatment of weeds. Site-specific weed management (SSWM) is the process of detecting weeds in a field, and applying an appropriate treatment such as herbicide only where required. This reduces the use of chemicals, which can be expensive and have the potential to cause harm to both people and the broader natural environment (Liu & Bruch, 2020). Through automatic weed management, human effort is also reduced (Osorio et al., 2020). Serrated tussock (*Nassella trichotoma*, hereafter tussock) is a weed of national significance in Australia, which offers little to no nutritional value to livestock, and has the ability to significantly reduce the carrying capacity of infested land (Campbell, 1998). It was estimated in 2018 that the annual cost of weeds, including control costs and the cost of lost production in the Australian livestock industry was $AUD2,409 million ($USD1,583[1]) (McLeod, 2018). The cost of control for tussock has been estimated to be well in excess of $AUD100 ($USD66) per hectare of infested land (Johnson et al., 2022).

Computer vision is a field of artificial intelligence that enables computers to interpret, analyze, and understand visual information from the world, such as images or videos. Advanced techniques in computer vision, include neural networks, such as convolutional neural networks (CNNs), as well as other methods, including Gaussian process regression. Such techniques have been applied to the detection of weeds in pastures, including dockleaf (Binch & Fox, 2017; Smith et al., 2019; Zhang et al., 2018; Kounalakis et al., 2019; Schori et al., 2019), thistle species (Sadgrove et al., 2017, 2017, 2018), horehound (Sadgrove et al., 2021), and grass species (Farooq et al., 2019). Other agricultural applications include plant diseases detection (Saleem et al., 2019; Li et al., 2021), yield prediction (Van Klompenburg et al., 2020; Muruganantham et al., 2022), and price forecasting (Jin & Xu, 2024, 2024).

Studies on the detection of tussock in the literature are uncommon, and typically include hand crafted feature extraction, such as local binary patterns (Ojala et al., 2002) and histograms of oriented gradients (Dalal & Triggs, 2005). In one study (Farooq et al., 2019), hyperspectral images of tussock and other grass species were used to develop a support vector machine (Cortes & Vapnik, 1995) based classification framework, with an accuracy of 80.90%. In another study (Ford et al., 2023), utilising a shallow network trained using the extreme learning machine algorithm (Guang-Bin et al., 2004), images containing tussock and background pasture elements were segmented, with a mean Intersection over Union (mIoU) of 87.6%.

The use of multipurpose networks, to simultaneously carry out two (or more) tasks is one interesting area of active research in agricultural computer vision. An example is the use of joint plant segmentation and stem localization networks for use in crop fields (Lottes et al., 2018; Zhang et al., 2020). Such models are Convolutional Neural Networks (CNNs), built using the typical encoder/decoder structure of segmentation models, but with two decoder arms, one for each task. This "joint" encoder structure reduces the computational complexity of the network as a whole (as compared to a separate encoder for each task), and has been demonstrated to produce superior feature maps, allowing for more robust results, especially for the stem localization task (Lottes et al., 2018; Zhang et al., 2020).

[1] Based on the average exchange rate for the 12 months to June 2024

The advantage of having a joint plant and stem localization network is that it allows for the application of more varied treatments, including herbicide, physical removal of weeds, or fertilization of crop plants. Specifically, while herbicide and fertilizer may be applied to the whole plant, physical removal is often only applied to the stem, using methods such as a metal stamping tool, or jet of steam (Lottes et al., 2018; Kounalakis et al., 2018). In pastures, herbicide may either be applied to the whole plant, if uptake is through the leaves, or the centre of the plant (hereafter the spraypoint), if uptake is through the root system. The latter is true for Flupropanate which is commonly used to treat tussock, due to its ability to remain active in the ground for long periods of time[2].

To the best of our knowledge, no studies have been conducted on the development of a combined plant and spraypoint detector specifically for use in pastures. To address this gap, we undertook a controlled study in this area. In order to minimize complicating factors, all data were collected at a single location, with images captured over a period of nine months, encompassing conditions with varying levels of drought stress. This research paves the way for further exploration of multipurpose detectors in pastures, enabling more diverse treatment options and ultimately leading to improved outcomes for farmers. In addition to pre-existing models, a new model was developed, based on elements of ConvNeXt (Liu et al., 2022), which was considered due to its superior performance on complex image tasks such as ImageNet (Russakovsky et al., 2015).

Although previous work utilized the arrangement of crop plants in lines in order to produce models which generalized well to new field conditions (Lottes et al., 2020), this technique is not available in pastures, where no such uniform arrangement of plants is present. For this reason, and in order to assist with model generalization, a new normalization technique was developed, which we call HistMatch normalization. This technique generates a more uniform probability distribution for network input compared to pre-existing methods.

The major contributions of the paper are:

1. Development of a joint plant and spraypoint detector for use in pastures, based on pre-existing models (Lottes et al., 2018; Zhang et al., 2020), as well as a new design utilising elements of ConvNeXt (Liu et al., 2022). To the best of our knowledge, this is the first example of this kind of multipurpose detector designed specifically for use in pastures.
2. Testing of the generalization ability of models to new datasets with different growing conditions.
3. Development of a new normalization method, called HistMatch normalization, for use in cases where the pixel intensity distributions of test images may vary significantly from those seen during training.The remainder of the paper is structured as follows. Section 2 provides the approach taken in this study, including pre-existing models, our model, the dataset used, implementation details, normalization techniques used, and experiments conducted. Section 3 presents the results of this study, while Sect. 4 provides an analysis of these results. Finally Sect. 5 provides concluding remarks, and identifies some possible future directions.

[2] Information on the treatment of tussock can be found at https://weeds.dpi.nsw.gov.au/Weeds/serratedtussock

Methods

Pre-existing models

For the dual plant segmentation and stem detection task, two pre-existing studies were found. In the first, by Lottes et al. (Lottes et al., 2018), a model with a single encoder, and two decoders was developed and tested on images captured in early stage sugar beet fields, achieving a mean average precision (Everingham et al., 2010) of 0.753–0.792 for the stem task, and 0.838–0.873 for the plant task. In the second study by Zhang et al. (Zhang et al., 2020), an alternative network structure was developed and tested on images of cauliflower seedlings growing in fields with very high weed presence, achieving an F-measure score of 0.974 for the plant segmentation task, and detection rate of 0.978 for the stem task.

The primary building block of Lottes' model (Lottes et al., 2018) is the dense block, which is composed of L densely connected 3×3 convolutional layers, each of which outputs G feature maps, where G is called the growth rate. The dense block also makes use of bottleneck layers, which are implemented by a point-wise convolution, in order to reduce computational complexity. Downsampling operations are implemented using 5×5 convolutional layers, with a stride of 2, and in total, there are three downsampling operations in the encoder. Upsampling operations in the decoder arms are implemented using transpose convolutions (Zeiler et al., 2010) with a kernel size of 2×2 and a stride of 2. Skip connections are implemented using concatenation along the filter axis. An overview of the dense block, and the macro structure of Lottes' model is given in supplementary figure **S1**.

Zhang's model utilizes an encoder, which consists of multiple inverted residual modules (IRMs) as used in MobileNet V2 (Sandler et al., 2018). IRMs utilize depth-wise and point-wise convolutions to reduce computational complexity while maintaining model accuracy. Point-wise convolutions are convolutions with a kernel size of 1×1, whereas depth-wise convolutions are a form of convolution in which each input feature is convolved separately with its own corresponding filter. Downsampling is achieved by using a stride of 2 in the depth-wise convolution of the first IRM in each stage. At the macro scale, different decoder arms branch off the backbone at different resolutions for both the segmentation and the stem tasks. There is bidirectional feature map exchange between adjacent intra-task decoder arms in order to improve feature representation. There is also a cross-task feature fusion linking the coarsest segmentation decoder to each of the stem decoders. This forces the stem decoder to focus on those sections of the image in which the plant is located, while ignoring irrelevant background details. Each decoder arm produces its own detection mask, which is fused into a single mask for each of the two tasks (plant segmentation and stem localization). In addition, deep supervision is used to directly train each output of the decoder arms. An overview of the IRM, as well as the macro structure of Zhang's model is given in supplementary figure **S2**.

New model

Minimizing the number of parameters in a model is often seen as a desirable design choice (see for example (McCool et al., 2017; Khan et al., 2020; Abdalla et al., 2019)). In this study, however, focus was instead on the development of a model which had training and inference time that were similar to (or better than) pre-existing models found in the literature. These

are often more important considerations, because many iterations of training are required in order to optimize hyperparameter choices, and inference time is a critical factor for any model that is to be deployed in real time.

For our model, taking inspiration from Lottes' and Zhang's models, a joint encoder with two decoders for the spraypoint and plant was used. Starting with Zhang's model as a base, elements of ConvNeXt (Liu et al., 2022) were incorporated into our network structure, as listed below:

1. "Patchify" stem: which is an initial convolutional layer with a kernel size and stride of 4.
2. ConvNeXt modules: consisting of an inverted bottleneck design, a large kernel size for the depthwise convolutions, to improve spatial awareness, and fewer activation and normalization layers, which slightly reduces computational complexity. In addition, the depth-wise convolution is brought to the front, and the expansion rate is reduced to 4, which further reduces parameters and FLOPs. The ConvNeXt module is shown in Fig. 1a.
3. An adjustable learning rate scheduler with a linear warmup followed by cosine decay.
4. Adjusted stage ratio: with four ConvNeXt modules in the first and third stages, and twelve in the second stage.Other ConvNeXt elements, such as GELU activation (Hendrycks & Gimpel, 2016), Layer Normalization (Ba et al., 2016), and Layer Scale (Touvron et al., 2021) were not included in our design, because they were found to produce consistently worse results during preliminary testing. The stage ratio, which include a larger number of modules at the $16\times$ downsampling level, is a feature of the ConvNext model (Liu et al., 2022), and allows for a larger number of filters, while reducing the overall FLOPs of the network.

The use of the patchify stem also reduces the computational complexity of the model, by greatly reducing the resolution of the feature maps quickly, and allowing for a larger number of filters. For this reason, we also elected to maintain Zhang's downsampling technique, which is applied to the first module of each stage. Another result of the patchify stem was that the final output of our encoder was at a resolution downsampled by a factor of $32\times$ as opposed to the factor of $16\times$ for Zhang's model. The number of filters used in the patchify stem, and stages one to three were 64, 64, 128 and 128 respectively. The number of filters in the decoder for both the spray point and plant segmentation task were 64. Our network used only a single $32\times$ downsampled side output for the spraypoint task, and two side outputs at $32\times$ and $8\times$ downsampling for the plant segmentation task. The final result was a model which had far more parameters than either Lottes' or Zhang's model, but which had a comparable inference time, and could be trained roughly three times faster. The macrostructure of our model is show in Fig. 1b.

Dataset

The images used in this study were collected at Yelangello, NSW (lat. −34.863, long. 149.169) on three separate occasions spaced over 9 months (Oct-21, Apr-22, Jul-22). The growing conditions were diverse, both in terms of drought stress, as well as the length of the background pasture elements, as can be seen in the example images given in Fig. 3. Images

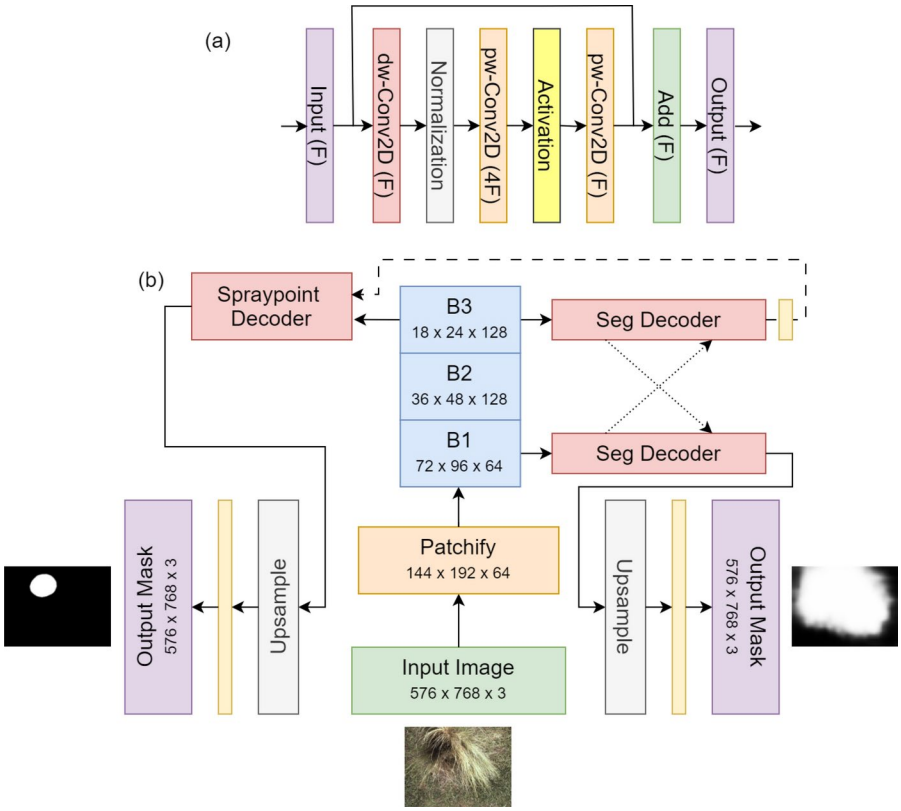
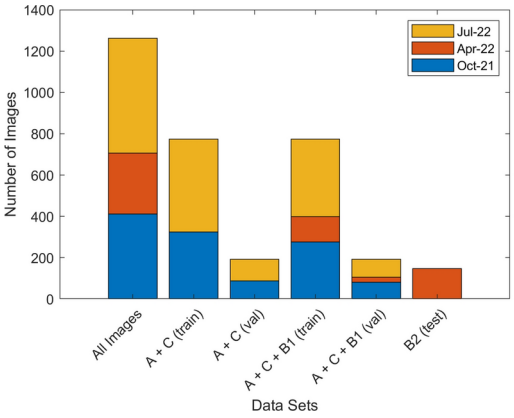


Fig. 1 Our model. (**a**) A ConvNeXt block, with a modified bottleneck design and fewer activation and normalization layers (which are explicitly shown here). (**b**) Macro structure showing the input image (green), patchify stem (orange), backbone (blue), decoders (red) and the output feature maps (purple). Loss layers are shown in yellow, and include losses for deep supervision (small rectangles), as well as for the final outputs of the model (large rectangles). Arrows display the flow of information through the model. In particular, the cross-task feature fusion is given with a dashed line, and the bidirectional feature map exchanges are given with a dotted line

were annotated using the VGG image annotator (Dutta & Zisserman, 2019), which allowed for a compromise between annotation accuracy and effort required per image. Three classes were used, being the negative, positive, and hard class, with the latter being for regions of the image where it was not clear whether or not tussock was present. The left section of Fig. 2 gives a breakdown of the number of images coming from the separate collection time for each of the various train, validation and test sets used in this study, while the right section shows the sets that were used in each of the experiments in this study.

Images were resized to 768×576 pixels for all models, which corresponded to a ground sampling distance of $\sim$ 1mm/pixel. Data augmentation, consisting of random rotations, zooms, vertical and horizontal flips, and contrast and brightness shifts were used for all experiments. However, because Tensorflow utilizes the CPU for all dataset manipulations (including augmentations, in our case), it was found that this became a significant computational bottleneck for training. For this reason, a bank of augmented images were produced



Experiment	Train	Validation	Test
1 (1a)	A + C (train)	A + C (val)	A + C (val)
2 (2a)	A + C (train)	A + C (val)	B2
3	A + C + B1 (train)	A + C + B1 (val)	B2

Fig. 2 The datasets used in this study. On the left is a breakdown of the number of images coming from each of the three collection dates. On the right is a table showing which datasets were used for training, validation and testing in each of the experiments

prior to training to be used by all models, which increased the size of the training set from 774 to 92,880 images.

Implementation details

All coding was carried out in Python with Keras/Tensorflow version 2.13.1. For Lottes' model, no source code could be found online, so it was recreated from scratch, based on the original paper (Lottes et al., 2018). The code for Zhang's model, downloaded using the link provided in their paper (Zhang et al., 2020), was in Tensorflow. For the sake of consistency, this code was converted into Keras, and modified so it could work with a multiclass problem.

All models were trained for a total of 350 epochs. The batch size was set to 16 for Lottes' and Zhang's model, as this was the maximum size possible without running into memory errors. For our model, batch sizes between 16 and 128 were tested, and a size of 64 was selected, as this performed slightly better during validation.

All models were trained using AdamW[3], with an optimal learning rate and weight decay calculated greedily via line search. The learning rate was decayed by a factor of 10 at epoch 150 for Zhang's model, and at epochs 50 and 250 for Lottes' model. For our model, a linear warmup of 100 epochs, followed by cosine decay was used. Training was carried out on an HPC cluster with 16 1-core 2.79 GHz Intel Xeon (Icelake) processors, 256 GB RAM, and a 40 GB NVIDIA A100-PCIE GPU. Inference time was calculated using an XPS 8950 Desktop running Linux, with a 10 GB NVIDIA GeForce RTX 3080 GPU.

Weights were initialized using the default Glorot uniform (Glorot & Bengio, 2010) initializer for Zhang's and our models. For Lottes' model, according to the instructions provided in the original paper (Lottes et al., 2018) He initialization (He et al., 2015) was used. Keras provides both He normal and He uniform weight initializers, the latter of which was selected, because it performed slightly better during validation.

[3] Adam and AdamW are identical in Keras, save for the default value for weight decay.

Stem-localization is a task that results in highly imbalanced classes, due to the fact that only a tiny proportion of the image pixels will correspond to a plant stem. For this reason, both Lottes' and Zhang's models employ an IOU loss for this task, which is designed to be more stable in problems with high class imbalance. For the task of plant segmentation, a more conventional cross-entropy loss is used. Although the original IOU loss paper by Rahman and Wang (Rahman & Wang, 2016) only considers the binary classification task, the problem generalizes to the multiclass problem by calculating the average loss across all classes. The formula for the IOU loss, $\mathcal{L}_{IOU}$, is given in Eq. 1, where V is the set of pixels in the training batch, $y^{(c)}$ is the ground truth label for class c, $\hat{y}^{(c)}$ is the predicted label for class c, and k is the number of classes.

$$\begin{aligned} I^{(c)} &= \sum_{v \in V} y_v^{(c)} \hat{y}_v^{(c)} \\ U^{(c)} &= \sum_{v \in V} y_v^{(c)} + \hat{y}_v^{(c)} - I^{(c)} \\ \mathcal{L}_{IOU} &= 1 - \frac{1}{k} \sum_{i=1}^{k} \frac{I^{(c_i)}}{U^{(c_i)}} \end{aligned} \tag{1}$$

Cross entropy loss was used for the plant segmentation task, while IOU loss averaged over the positive and negative classes was used for the stem task. Each task was assigned equal weight. Loss was calculated at the end of each epoch, and the weights that produced the best validation loss were restored at the end of training.
In order to remove noise, and small spurious detections from the spraypoint segmenter, postprocessing, including Gaussian smoothing and removal of small connected components was applied, with the optimal parameters of Gaussian sigma, and minimum component size determined during validation using grid search.

Training time was measured as the average of five runs. The inference time, which is the time taken to pass a single image through the network, was measured as the average for 1000 separate passes through the model. This is reported as the step time when calling the Keras Model.evaluate method with a batch size of 1.

Normalization techniques

Several different data preprocessing (or normalization) techniques were used. In every case each channel was normalized separately, across either a single image (per-image), or an entire batch of images (per-batch). In the first method, the data was rescaled to the interval $[-0.5, 0.5]$. We called this method "default" normalization, as it was the technique used in Lottes' and Zhang's papers. The formula for default normalization is presented in Eq. 2, where I_{max} and I_{min} are the maximum and miminum pixel values, and λ is a small constant to prevent division by zero:

$$\hat{I} = \frac{I - I_{min}}{\max(I_{max} - I_{min}, \lambda)} - 0.5 \tag{2}$$

In the second method, which we called "standard normal" or "naïve" normalization, the data was centered around 0, by subtracting the mean, and dividing by the standard deviation. This method will transform the data into a standard normal distribution ($\mathcal{N}$), but only if the data were normally distributed to start with. This method is presented in Eq. 3, where $\mu(I)$ and $\sigma(I)$ are the mean and standard deviations of the image pixel values:

$$\hat{I} = \frac{I - \mu(I)}{\max(\sigma(I), \lambda)} \tag{3}$$

For the third method, which we called HistMatch normalization, we need first to introduce the technique of image histogram matching. This is the process whereby the image histogram[4] of one image is transformed to the histogram of another image. An example of histogram matching is provided on the right of Fig. 4. Using this concept, we "histogram match" the image data to $\mathcal{N}$.

This histogram matching is shown pictorially for a single channel in Fig. 4. The preprocessing function, $g : [0, 1] \to [-\infty, \infty]$[5] can be found by comparing the cumulative distribution functions for the image histogram (denoted F_I), and $\mathcal{N}$ (denoted Φ). Starting on the x-axis for a given $x \in [0, 1]$, we draw a vertical line to F_I, then a horizontal line to Φ, and finally a vertical line back to the x-axis to find $g(x)$. The normalization step is then performed by broadcasting g across all pixels in the image. As can be seen in Fig. 4, the naïve approach (dotted cyan) does not match $\mathcal{N}$ very closely, while the HistMatch approach (dotted green) does, except where (in this case) overexposure causes a loss of intensity information.

The algorithm for HistMatch normalization is presented in Algorithm 1. Φ refers to the cumulative distribution function of a standard normal distribution, and count is a function that returns the count of pixels for which a condition is true. P is the set of all possible pixel values, $\hat{n}$ and $\hat{y}$ are hyperparameters which determine those values, $Y \subset [-\hat{y}, \hat{y}]$, of cardinality $\hat{n} + 1$, for which we calculate Φ, and I is the input image of size $h \times w$. Where applicable, functions are broadcast over a vector or matrix. On line 1, we calculate the cumulative histogram, h for the input image I, and on line 2, we calculate the midpoints between adjacent entries of h, in order to account for the discrete distribution of pixel values. In lines 3 and 4, we define the function g, which maps every pixel value $p \in P$ to the point $y \in Y$, which minimizes the absolute difference between $h(p)$ and $\Phi(y)$. Finally, in line 5, this function, g is broadcast over the input image. A Python implementation of HistMatch normalization is presented at the end of the supplementary material.

[4] The image histogram is a histogram of the proportion of image pixels corresponding to each intensity level

[5] For the purpose of the figure, we have pre-normalized the image pixels to the range [0, 1], however in practice this step would be unnecessary, and g would be defined over $\{0, 1, \ldots, 255\}$

Algorithm 1 HistMatch normalization

Require: $\mathbf{P} = \{0, 1, \ldots, 255\}$
Require: $\hat{n} \in \mathbb{N}$
Require: $\hat{y} \in \mathbb{R}^+$
Require: $\mathbf{Y} = \{-\hat{y} + \frac{2n\hat{y}}{\hat{n}} | n \in \{0, 1, \ldots, \hat{n}\}\}$
Require: $\mathbf{I} \in \mathbf{P}^{h \times w}$

$\mathbf{h} \leftarrow [\frac{1}{hw}\mathtt{count}(I \leq p)]_{p \in \{-1\} \cup \mathbf{P}}$
$\mathbf{h} \leftarrow [\frac{1}{2}(\mathbf{h}_p + \mathbf{h}_{p-1})]_{p \in \mathbf{P}}$
$A : \mathbf{P} \times \mathbf{Y} \rightarrow \mathbb{R}_0^+, A(p, y) = |\mathbf{h}_p - \Phi(y)|$
$g : \mathbf{P} \rightarrow \mathbf{Y}, g(p) = \arg\min_y A(p, y)$
return $g(\mathbf{I})$

Experiments

Initially, three experiments were conducted. In the first experiment, datasets A and C were combined (hereafter dataset A + C) and used in a 80%/20% train/validation split, in order to calculate suitable hyperparameters for the models, and provide baseline metrics against which to measure the other experiments. In the second experiment, dataset A + C was again used for training and validation, and half of dataset B (hereafter dataset B2) was used for calculating test statistics. This experiment was conducted in order to see how well the models generalized to images taken with growing conditions that they had not seen during training. In the third experiment, images in dataset A + C were randomly selected and replaced with the other half of dataset B (hereafter called B1), yielding dataset A + C + B1. This dataset was used for training and validation, while B2 was again used for testing. This experiment was conducted to produce a baseline result for the B2 dataset in which the models had been partially trained with images taken at the same time, and for comparison with experiment two. A summary of the datasets used for each experiment is provided in the right section of Fig. 2. For experiments 1, 2 and 3, "default" per-batch normalization was used.

Experiments 1a and 2a were also conducted as repetitions of experiments 1 and 2 respectively, in which all six normalization techniques (i.e. default, naïve or HistMatch applied either per-image or per-batch) were compared.

Finally, the models were evaluated using the publicly available Cauliflower dataset from the study by Zhang et al. (Zhang et al., 2020). Two different versions of our model were used for this dataset, with the first being identical to the version used in the previous experiments, and the second, which we call ours-4s having an additional lateral decoder arm for each task. The structure of the ours-4s model is given in the supplementary material (figure **S3**).

The version of the dataset we downloaded did not include stem annotations. To address this, we created our own stem annotations by first marking the stem location as a point using the VGG image annotator (Dutta & Zisserman, 2019), and then applying a Gaussian filter at that location with $\sigma = 7$ for Lottes' and Zhang's models, and $\sigma = 15$ for our model. In both cases, the brightness of the generated annotations was adjusted so that the center pixel, corresponding to the stem location, had a value of 255. We opted for a higher σ value for our

model to provide a larger target, accommodating the fact that our model produces an output downsampled by a factor of 32 (or 8 for ours-4s model) for the stem task.

For Lottes' and Zhang's models, we used the same learning rate and augmentation strategy as in the original study (Zhang et al., 2020). Specifically, the training dataset was increased by a factor of four using horizontal and/or vertical flips. For our model, the optimal learning rate of 0.001 was found using a line search, and the augmentation strategy was expanded by incorporating zooms and rotations in addition to flips. Default, per image normalization was used. For the stem task, the IOU-loss (positive class only) was used for Lottes' and Zhang's model, and the IOU-loss (positive and negative classes) used for our model. For the plant segmentation task, crossentropy loss was used for all models.

For experiments using our dataset, the plant segmentation task was evaluated using the per-pixel mean intersection over union (IoU) across positive and negative classes. For the spraypoint task, per-object precision, recall, and F1 score were employed. A spraypoint detection was considered positive if its centroid was within a radius of 50 pixels (equivalent to 50 mm) from the nearest unassigned ground truth centroid. All metrics presented include the mean and standard deviation across five separate runs.

For experiments using Zhang's dataset, we followed the same metrics as those used in the original study by Zhang et al. (Zhang et al., 2020). Specifically, the F1 score was used for the plant segmentation task. For the stem task, since each image in this dataset contained exactly one stem, the geometric mean of the average activation was used to determine the predicted stem location. The successful detection rate (SDR), defined as the ratio of successful detections to the total number of test images, and the average Euclidean distance (ED) between the predicted and ground truth stem locations (in pixels) for successful detections were reported. A detection was considered successful if it was within 17.5 pixels of the ground truth location.

Results

Table 1 shows the optimal hyperparameter as well as the number of parameters, FLOPs, training time and inference time of each of the three models, as determined in experiment 1, according to the methodology presented in Sect. 2.

Tables 2 and 3 provide the results for experiments 1, 2 and 3, for the plant and spraypoint tasks respectively, showing the mean and standard deviation of the metrics used for five

Table 1 List of hyperparameters for training and post-processing, and specifications for Lottes', Zhang's and our models for experiments on our dataset

Hyperparameter/ Specification	Lottes	Zhang	Our
Batch Size	16	16	64
Weight Decay	0.01	0.01	0.1
EMA	Yes	No	No
Learning Rate	0.003	0.003	0.001
Minimum Size (CC)	2000	2000	4000
Gaussian Sigma	10	10	0
Parameters	59,045	153,263	2,643,779
GFLOPs	10.0	11.3	4.9
Training Time (s)	14,475	15,809	5,064
Inference Time (ms)	14.8	21.5	15.7

Table 2 Results of experiments 1, 2 and 3 for the plant task

Experiment	Model	IoU(-)	IoU(+)	mIoU
One	Lottes	0.871 ± 0.013	0.606 ± 0.043	0.739 ± 0.027
	Zhang	0.953 ± 0.000	0.865 ± 0.001	0.909 ± 0.001
	Our	$\mathbf{0.957 \pm 0.001}$	$\mathbf{0.874 \pm 0.003}$	$\mathbf{0.915 \pm 0.002}$
Two	Lottes	0.089 ± 0.096	0.451 ± 0.025	0.270 ± 0.055
	Zhang	0.332 ± 0.225	0.576 ± 0.083	0.454 ± 0.154
	Our	$\mathbf{0.829 \pm 0.016}$	$\mathbf{0.785 \pm 0.029}$	$\mathbf{0.807 \pm 0.022}$
Three	Lottes	0.646 ± 0.039	0.559 ± 0.028	0.602 ± 0.026
	Zhang	0.879 ± 0.011	0.870 ± 0.010	0.875 ± 0.011
	Our	$\mathbf{0.885 \pm 0.004}$	$\mathbf{0.873 \pm 0.003}$	$\mathbf{0.879 \pm 0.003}$

Results are given as mean ± standard deviation of five runs, and the best result for each experiment is given in bold

Table 3 Results of experiments 1, 2 and 3 for the stem task

Experiment	Model	Precision	Recall	F1-Score
One	Lottes	0.467 ± 0.069	0.358 ± 0.011	0.404 ± 0.031
	Zhang	0.858 ± 0.023	$\mathbf{0.900 \pm 0.030}$	0.878 ± 0.005
	Our	$\mathbf{0.875 \pm 0.014}$	$\mathbf{0.900 \pm 0.022}$	$\mathbf{0.887 \pm 0.015}$
Two	Lottes	0.049 ± 0.017	0.143 ± 0.062	0.073 ± 0.026
	Zhang	0.482 ± 0.267	0.620 ± 0.092	0.496 ± 0.141
	Our	$\mathbf{0.941 \pm 0.020}$	$\mathbf{0.693 \pm 0.061}$	$\mathbf{0.796 \pm 0.041}$
Three	Lottes	0.518 ± 0.149	0.195 ± 0.101	0.266 ± 0.097
	Zhang	0.834 ± 0.042	0.676 ± 0.065	0.745 ± 0.041
	Our	$\mathbf{0.931 \pm 0.016}$	$\mathbf{0.698 \pm 0.012}$	$\mathbf{0.797 \pm 0.011}$

Results are given as mean ± standard deviation of five runs, and the best result for each experiment is given in bold

Table 4 Results with HistMatch (image) normalization for experiment 2a, in which models were tested on growing conditions not seen during training, given as mean ± standard deviation of five runs

Model	Plant	Spraypoint		
	mIoU	**Precision**	**Recall**	**F1**
Lottes	0.703 ± 0.052	0.634 ± 0.052	0.572 ± 0.040	0.599 ± 0.019
Zhang	0.821 ± 0.054	0.735 ± 0.043	0.732 ± 0.039	0.733 ± 0.037
Our	$\mathbf{0.854 \pm 0.006}$	$\mathbf{0.843 \pm 0.023}$	$\mathbf{0.772 \pm 0.034}$	$\mathbf{0.806 \pm 0.028}$

Best results for each metric in bold

iterations of each model. In addition, supplementary figure **S4** shows the confusion matrices for all models for experiments 1, 2 and 3, for the plant segmentation task.

Figure 5 shows the results for experiments 1a and 2a, in which different normalization methods were tested. The results for HistMatch (image) normalization for each model for experiment 2a is provided in Table 4. A table of all results for experiments 1a and 2a is provided in the supplementary material (tables **S1** and **S2**).

Table 5 presents the results from experiments conducted using Zhang's dataset. These results include the findings reported by Zhang et al. (Zhang et al., 2020) for both Lottes' and

Table 5 Results for experimentation on Zhang's dataset, both as reported by Zhang et al. (Zhang et al., 2020), and as carried out in this study.

Study	Model	Plant		Stem			
		F1		SDR		ED (pixels)	
(Zhang et al., 2020)	Lottes	**0.975**		0.912		2.62	
	Zhang	0.974		0.978		2.56	
This study	Lottes	0.956	±0.002	0.931	±0.010	3.85	±0.09
	Zhang	0.974	±0.001	0.995	±0.004	**2.21**	±0.07
	Ours	0.968	±0.0002	**0.997**	±0.000	2.45	±0.04
	Ours-4s	**0.975**	±0.0002	0.996	±0.001	2.38	±0.02

For our study, the results are given as the mean ± standard deviation of five runs. Best results for each metric is in bold

Zhang's models, as well as the outcomes for the four models considered in our study, which include Lottes' and Zhang's models, in addition to our two models.

Discussion

This paper presented a study on a joint plant and spraypoint detector which was tested on images of tussock taken in south-east Australia. Lottes' and Zhang's model were used as is, and an alternative model was proposed for testing, which utilised several elements of ConvNeXt, including a patchify stem, ConvNeXt modules, and a learning rate schedule with linear warmup followed by Cosine decay. Our model performed best in all of the metrics tested, as show in in Tables 2 and 3, although it was only slightly better than Zhang's model in the first experiment, in which dataset A + C was used for both training and validation. When tested on growing condition which were not included in the training set (experiment 2/2a), our model was able to achieve metrics of $0.8 - 0.85$ in both tasks, as show in Table 4.

For the plant segmentation task, it was difficult to separate each tussock plant from its neighbours, due to poorly defined boundaries. Moreover, the tussock plants were often large, and not centered, with only part of the plant appearing in an image. This made it impractical to use an object-based approach in the interpretation of results. The mIoU metric gives a balanced, single score, which takes into account how well both the positive and negative classes were detected. False negative regions may lead to inadequate, or missing treatment of a target plant, whereas false positive regions may lead to herbicide overuse, and damage to the background pasture.

By comparison, for the spraypoint task, each spraypoint formed a clearly defined unit, and for this reason per-object metrics were used. The recall is the proportion of target plants that were detected by the model, whereas the precision is the proportion of the model's detections which corresponded to actual (ground truth) target plants. The F1 score combines both these metrics into a single score, allowing for easier comparison of models.

For the stem detection task, the IOU loss averaged over the positive and negative classes proved to be a better choice than the alternative approach, in which only the positive class is considered when calculating the loss. This latter approach is faithful to the original work of Rahman and Wang (Rahman & Wang, 2016). However, in our case, we found that this

occasionally led to instability in training, during which the model would learn to predict almost all of the pixels as positive[6].

Lottes' model performed significantly worse than the other two models as show in Tables 2 and 3, even in experiment one, in which the training and validation sets were drawn from the same dataset. One possible reason for this may be due to the fact that it had fewer downsampling steps, and hence a lower effective receptive field size[7], coupled with the fact that many tussock plants were large, with some covering most of an image. Another possible reason is that the number of parameters was insufficient to capture the more complex nature of a pasture, as compared to the comparatively simpler environment in an early stage crop field, with small (mostly) non-overlapping green plants against a soil background, as was the case in the original study by Lottes et al. (Lottes et al., 2018).

Zhang's model performed well on the first experiment, but did not generalize well to field conditions not seen in training. This was largely due to these models predicting most pixels as positive, as seen in supplementary figure **S4**. Again, this may be because the model has relatively few parameters, which cannot learn sufficiently robust features to be useful in the general case. This result highlights the need to test models on a dataset which is wholly separate from the training dataset, as without this, our conclusion would likely have been that the performance of Zhang's model was almost as good as our new model.

In the original paper by Zhang et al. (2020), the authors tested their model on images containing a single plant, and used the spacial average of the activation map as the stem location, suggesting that connected component analysis be used if multiple plants were present in a frame. However, in our study, multiple fragmented detections were produced by Zhang's model which required post-processing, in order to achieve acceptable results. The postprocessing for each model included applying a Gaussian blur to the image in order to join close, disjoint regions, followed by the removal of small detections. In this case, while Lottes' and Zhang's model required both of these steps, our model only required the removal of small detections, as shown in Table 1.

Experiment 3 incorporated half of dataset B (B1) into the training set, which allowed us to gain an understanding of whether dataset B2 was inherently easier or harder than dataset A + C. The results of our and Zhang's models demonstrated that for the plant segmentation task, dataset B2 was not significantly harder, but for the spraypoint detection task it was, especially with respect to the number of false negatives (and hence a significantly lower recall). Closer examination of dataset B2 revealed that a lot of the missed spraypoints were on the edge of the image (for example, see rows (c) and (d) in Fig. 3), and in fact, in this location, it was often difficult for the annotator to decide whether to include it as a ground-truth annotation, or place it in the "hard" class. Interestingly, there was no significant difference between experiments two and three for our model in the spraypoint task, indicating that the model was already robust in detecting spraypoints, even when tested on images with growing conditions it had not been trained on.

When comparing the results of experiments two and three, it is evident that our model was the only one able to generalize effectively. The reduction in performance metrics for the

[6] Due to the way that Rahman and Wang's loss is implemented, a model which classifies all pixels as positive has a loss of $1 - p$, where p is the proportion of pixels that are in the positive class. By contrast, a model which classifies all pixels as negative has a loss of 1. This means that when the model is struggling to distinguish the two classes, it is likely to err on the side of classifying most or all of the pixels as positive.

[7] For a study on effective receptive field, the reader is referred to (Luo et al., 2016)

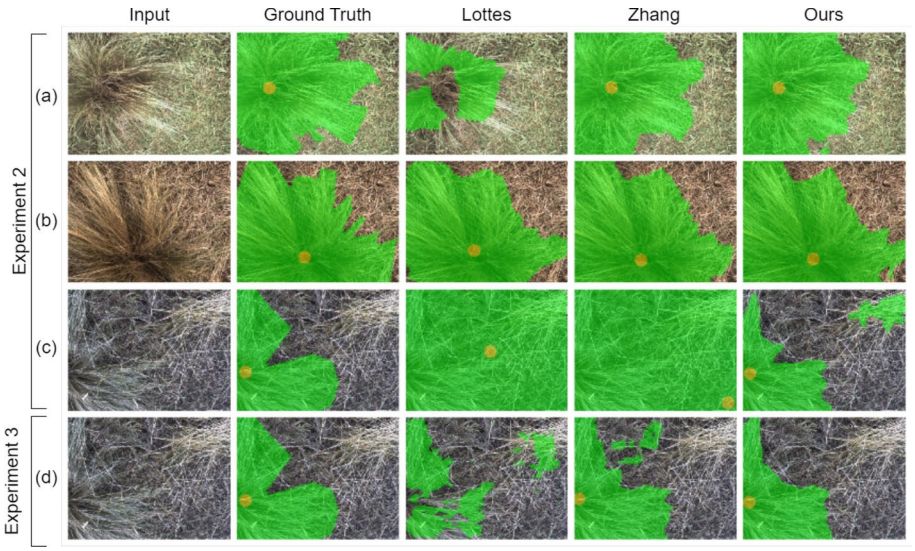


Fig. 3 Example images and segmentation results for Lottes', Zhang's and our models, for experiments 1 to 3. Images are from the Oct-21 (**a**), Jul-22 (**b**) and Apr-22 (**c**, **d**) sets. Images in row (**c**) show an example where models were tested on field conditions not seen during training. In this situation Lottes' and Zhang's models often classified most or all pixels as positive. Green positive (plant), yellow positive (spraypoint centroid). Row (**d**) shows the same image as in row (**c**), but where images from the Apr-22 dataset were included during training. In this case, there was typically little difference in performance between Zhang's and our model, whereas Lottes' model often continued to struggle

plant and spraypoint tasks was just 0.072 and 0.001, respectively, when the test set included growing conditions not seen during training. In contrast, Zhang's model showed a decline of 0.421 and 0.249 for the same tasks, while Lottes' model exhibited reductions of 0.332 and 0.193.

Our model contained a patchify stem, which resulted in a rapid reduction in the spatial resolution of the feature map in the early stages of our model. Taken in conjunction with a reduction in the expansion rate and fewer activation and normalization layers in our ConvNeXt modules, this allowed us to use more filter banks, while having fewer FLOPs than both Lottes' and Zhang's model. The result was a model with 17 times as many parameters as Zhang's model, but which was faster at both inference and training ($1.4\times$ and $3\times$ respectively). Also of interest is the fact that Lottes' model was marginally faster than our model at inference ($1.1\times$), in spite of the fact that it required $2\times$ as many FLOPs. This demonstrated that neither parameters nor FLOPs are necessarily useful indicators for predicting model speed.

During the interpretation of results for experiment two as shown in Tables 2 and 3, in which the models' ability to generalize was tested, it was found that Lottes' and Zhang's models sometimes performed poorly, predicting most pixels as positive for the plant segmentation task. To investigate why this was the case, we experimented with histogram matching techniques, in which the histogram of one image is transformed to match the histogram of a second image. In this way, we could transform the histogram of an image in the test set to match the histogram of an image in the training set, in order to make the test image more closely resemble the kinds of image seen during training. With this technique, we got

ambiguous results, with models sometimes performing better and sometimes performing worse than without histogram matching.

One possible way to overcome differences in image histogram distributions between separate datasets could be to adopt an additional step during data augmentation, in which images are transformed to a randomly generated image histogram. However, to adopt this, one would need to find a probability distribution for the image histograms, which closely matched the kinds of images likely to be encountered by the model, and find suitable bounds for the parameters of this distribution. Instead, we realised that it would be easier to include a preprocessing step (applied to an image immediately prior to input into the model), which transformed the image histogram of the input image to a pre-chosen probability distribution, the most obvious choice being $\mathcal{N}$, and this was the technique which we used in this study. As discussed in Sect. 2.5, the naïve approach to this problem does not produce a distribution that closely matched $\mathcal{N}$, which lead us to develop our own HistMatch normalization.

Experiments 1a and 2a, were conducted to determine whether the choice of normalization method was important, with the results given in Fig. 5. Paying particular attention to experiment 2a, in which models were tested on growing conditions not seen during training, it was found that HistMatch normalization applied to each image separately was the best choice by a considerable margin for Lottes' model for all metrics tested. For Zhang's and our models, the differences in normalization methods were not so pronounced, with HistMatch normalization providing the best results in three out of four metrics for Zhang's model (compared to one for naïve), and two out of four metrics for our model (compared to one each for default and naïve). HistMatch normalization simplifies the input data distribution by transforming it into a standard normal distribution, as illustrated in Fig. 4. This approach may offer the greatest benefit to simpler models with fewer parameters, which may struggle to manage the underlying complexities in the original data distribution (Fig. 5).

Finally, experiments were conducted on Zhang's dataset to evaluate how well our model generalized to the tasks of plant segmentation and stem detection in crop fields. In addition to the original model used in previous experiments, an adjusted model, referred to as Ours-4s, was also tested. This modified version included an additional decoder for each task, resulting in smaller upsampling steps of $4\times$ for segmentation and $8\times$ for stem detection, compared to $8\times$ and $32\times$, respectively, in the original model. These adjustments

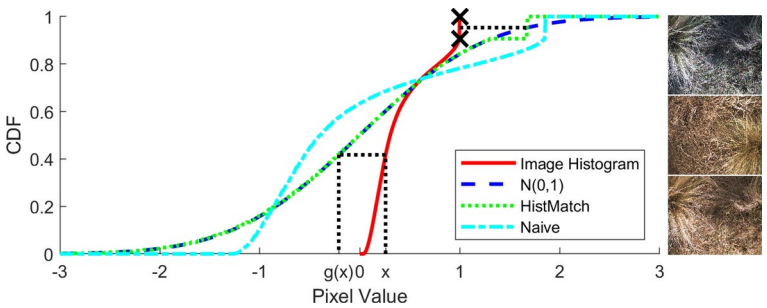


Fig. 4 Demonstration of HistMatch normalization, in which the image pixels are transformed to $\mathcal{N}$. To account for the discrete distribution of pixel intensities, the midpoint between adjacent points on F_I (dark crosses) is used when calculating g. On right of figure, is an example of histogram matching to a reference image. From top to bottom, original image, reference image, original image histogram matched to reference image

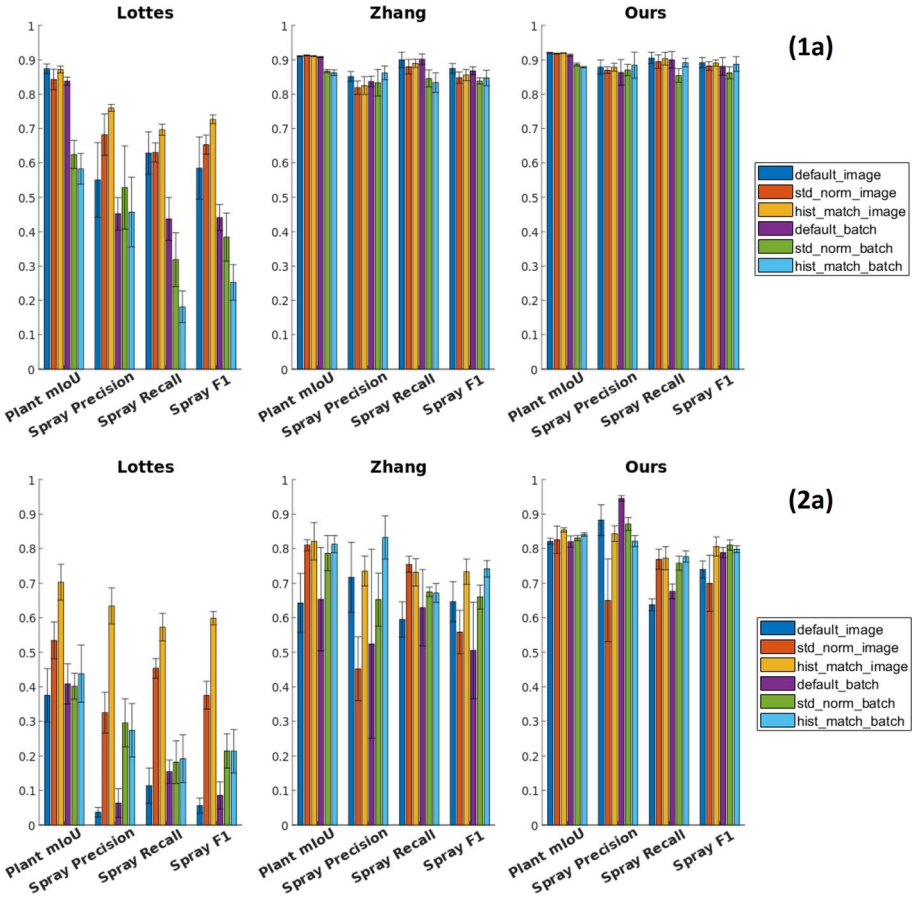


Fig. 5 Experiment 1a (top) and 2a (bottom): plant mIoU, and spraypoint precision, recall and F1-score for different normalization methods. In experiment 1a, the same dataset was divided into a training and a validation set. In experiment 2a, the test images were drawn from a different dataset to the training images (different growing conditions)

were expected to enhance the segmentation of fine plant edges and improve the accuracy of detecting small features such as the plant stem. The Ours-4s model was designed as an intermediate between the large upsampling stages of our original model and Zhang's model, which does not require final upsampling, as its classification layer operates at the full image resolution.

As shown in Table 5, Ours-4s model performed best for the plant segmentation task, achieving an F1 score of 0.975, which was equivalent to the results reported for Lottes' model by Zhang et al. (Zhang et al., 2020). For the stem detection task, our model achieved the highest SDR of 0.997, followed closely by Ours-4s at 0.996 and Zhang's model at 0.995. In terms of ED, which is the distance between the predicted stem and the ground truth stem, Zhang's model performed best with an ED of 2.21 pixels, compared to 2.38 for Ours-4s and 2.45 for our model.

The stem detection metrics for Zhang's and Lottes' models in our study were significantly better than those reported by Zhang et al. (2020), except for the ED for Lottes' model. This is likely due to the fact that we produced our own annotations for the stem task. In contrast, for the segmentation task, Lottes' model performed worse in our study compared to Zhang et al.'s reported results, suggesting there may have been implementation differences between the two studies. Unfortunately, we were unable to find an online version of Lottes' model for direct comparison with our implementation. Although we made every effort to follow the original paper (Lottes et al., 2018) as closely as possible, it is likely that some discrepancies between the two implementations contributed to the performance differences.

Our models demonstrated performance comparable to or slightly better than Zhang's model on the plant F1 and stem SDR metrics, though they performed somewhat worse on the ED metric. Notably, this was achieved despite our models employing final bilinear upsampling steps of $8\times$ for the plant task and $32\times$ for the stem task (or $4\times$ and $8\times$ for Ours-4s model). In contrast, Zhang's model generates output masks at full resolution, eliminating the need for upsampling. Despite these large upsampling steps, our models were able to locate the stem with an average error of 2.45 pixels (or 2.38 pixels for Ours-4s model). While Zhang's model slightly outperformed ours in terms of the ED metric, it came at a significantly higher computational cost. The two full-resolution decoder arms in Zhang's model, one for each task, account for 6.1 GFLOPs, which is more than half of the total computational cost of 11.3 GFLOPs. This highlights the importance of carefully considering model design, particularly as to whether the task at hand requires the additional output resolution and whether the corresponding computational overhead is justified. Minimizing computation at full resolution can help optimize efficiency with minimal or no performance loss.

Conclusion and future directions

This carefully controlled study has demonstrated the feasibility of a joint plant and spraypoint detector for use in pastures, paving the way for further research into multi-task models in pastoral environments. It was observed that dual-task models designed for early-stage crop fields struggled to perform well in pastures, especially when tested on growing conditions not represented in the training set. In contrast, our model significantly outperformed other models in this setting on both the segmentation and spraypoint tasks, achieving at least 30% better metrics. The challenges faced by other models were mitigated through the use of HistMatch normalization, suggesting that this technique may be particularly effective in enhancing model generalization in environments where robust performance is difficult to achieve.

To advance the development of multi-task networks in pasture environments, further data collection is required. Specifically, creating a dataset that includes multiple long-blade grass species, in addition to tussock, such as African lovegrass (*Eragrostis curvula*) and Chilean needle grass (*Nassella neesiana*), both of which are significant weeds in Australia, would be instrumental in demonstrating the effectiveness of such networks in distinguishing between different target species. Unfortunately, to our knowledge, no such multi-task dataset currently exists.

Furthermore, the integration of additional tasks, such as biomass or nutrient content estimation, which are typically handled by separate networks from the segmentation network (Skovsen et al., 2019; Mortensen et al., 2019) could further expand the potential of multi-task networks in pastures. However, this would necessitate additional network design, as these tasks involve regression rather than classification.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s11119-024-10208-y.

Acknowledgements The first author was supported by a Commonwealth of Australia RTP scholarship, with a top-up scholarship from the "Novel Autonomous Robotic Weed Control to Maximise Agricultural Productivity" CRC (CRCPIX000099). The authors would like to thank the NSW Department of Primary Industries and Agent Oriented Software for providing the dataset.

Author contributions All authors contributed equally to this paper.

Funding Open Access funding enabled and organized by CAUL and its Member Institutions

Data availability The data used in this study will be made available on request by email to the first author.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

Abdalla, A., Cen, H., Wan, L., Rashid, R., Weng, H., Zhou, W., & He, Y. (2019). Fine-tuning convolutional neural network with transfer learning for semantic segmentation of ground-level oilseed rape images in a field with high weed pressure. *Computers and Electronics in Agriculture, 167*, 105091. https://doi.org/10.1016/j.compag.2019.105091

Ba, J.L., Kiros, J.R., Hinton, G.E. (2016). Layer normalization. arXiv preprint arXiv:1607.06450

Binch, A., & Fox, C. W. (2017). Controlled comparison of machine vision algorithms for Rumex and Urtica detection in grassland. *Computers and Electronics in Agriculture, 140*, 123–138. https://doi.org/10.1016/j.compag.2017.05.018

Campbell, M. (1998). Biological and ecological impact of serrated tussock (nassella trichotoma (nees) arech.) on pastures in australia. *Plant Protection Quarterly, 13*, 80–85.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning, 20*(3), 273–297. https://doi.org/10.1007/BF00994018

Dalal, N., Triggs, B. (2005). Histograms of Oriented gradients for human detection. In *IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*. IEEE, San Diego, California . https://doi.org/10.1109/CVPR.2005.177

Dutta, A., Zisserman, A. (2019). The VIA annotation software for images, audio and video. In *Proceedings of the 27th ACM international conference on multimedia*. MM '19. ACM, New York, NY, USA. https://doi.org/10.1145/3343031.3350535

Everingham, M., Van Gool, L., Williams, C. K., Winn, J., & Zisserman, A. (2010). The pascal visual object classes (voc) challenge. *International Journal of Computer Vision, 88*, 303–338. https://doi.org/10.1007/s11263-009-0275-4

Farooq, A., Jia, X., Zhou, J. (2019). Texture and shape features for grass weed classification using hyperspectral remote sensing images. In *IGARSS 2019 - 2019 IEEE international geoscience and remote sensing symposium* (pp. 7208– 7211). IEEE, Yokohama, Japan . https://doi.org/10.1109/IGARSS.2019.8900132

Ford, J., Sadgrove, E., & Paul, D. (2023). Developing an extreme learning machine based approach to weed segmentation in pastures. *Smart Agricultural Technology, 5*, 100288. https://doi.org/10.1016/j.atech.2023.100288

Glorot, X., Bengio, Y.(2010). Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, pp. 249– 256 . JMLR Workshop and Conference Proceedings

He, K., Zhang, X., Ren, S., Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision* (pp. 1026– 1034). https://doi.org/10.1109/ICCV.2015.123

Hendrycks, D., Gimpel, K. (2016). Gaussian error linear units (gelus). arXiv preprint arXiv:1606.08415

Huang, G.-B., Zhu, Q.-Y., Siew, C.-K. (2004). Extreme learning machine: a new learning scheme of feedforward neural networks. In *2004 IEEE international joint conference on neural networks (IEEE Cat. No.04CH37541)*. IEEE, Budapest, Hungary . https://doi.org/10.1109/IJCNN.2004.1380068

Jin, B., Xu, X. (2024). Wholesale price forecasts of green grams using the neural network. *Asian Journal of Economics and Banking*

Jin, B., & Xu, X. (2024). Forecasting wholesale prices of yellow corn through the gaussian process regression. *Neural Computing and Applications, 36*(15), 8693–8710.

Johnson, S.B., Baker, S.J., Davidson, B., Harris, C.A., Sinclair, K., Wu, H., Bajwa, A.A. (2022). Making a difference to invasive grasses management on the ground. In *2022 Australasian weeds conference* (vol. 1, p. 316).

Khan, A., Ilyas, T., Umraiz, M., Mannan, Z. I., & Kim, H. (2020). CED-Net: Crops and weeds segmentation for smart farming using a small cascaded encoder-decoder architecture. *Electronics, 9*(10), 1602. https://doi.org/10.3390/electronics9101602

Kounalakis, T., Triantafyllidis, G. A., & Nalpantidis, L. (2019). Deep learning-based visual recognition of rumex for robotic precision farming. *Computers and Electronics in Agriculture, 165*, 104973. https://doi.org/10.1016/j.compag.2019.104973

Kounalakis, T., Triantafyllidis, G. A., & Nalpantidis, L. (2018). Image-based recognition framework for robotic weed control systems. *Multimedia Tools and Applications, 77*(8), 9567–9594. https://doi.org/10.1007/s11042-017-5337-y

Li, L., Zhang, S., & Wang, B. (2021). Plant disease detection and classification by deep learning-a review. *IEEE Access, 9*, 56683–56698.

Liu, B., & Bruch, R. (2020). Weed detection for selective spraying: A review. *Current Robotics Reports, 1*(1), 19–26. https://doi.org/10.1007/s43154-020-00001-w

Liu, Z., Mao, H., Wu, C.-Y., Feichtenhofer, C., Darrell, T., Xie, S. (2022). A convnet for the 2020s. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 11976–11986). https://doi.org/10.1109/CVPR52688.2022.01167

Lottes, P., Behley, J., Chebrolu, N., Milioto, A., & Stachniss, C. (2020). Robust joint stem detection and crop-weed classification using image sequences for plant-specific treatment in precision farming. *Journal of Field Robotics, 37*(1), 20–34.

Lottes, P., Behley, J., Chebrolu, N., Milioto, A., Stachniss, C. (2018) Joint stem detection and crop-weed classification for plant-specific treatment in precision farming. In *2018 IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 8233– 8238). https://doi.org/10.1109/IROS.2018.8593678

Luo, W., Li, Y., Urtasun, R., Zemel, R. (2016). Understanding the effective receptive field in deep convolutional neural networks. *Advances in Neural Information Processing Systems* **29**

McCool, C., Perez, T., & Upcroft, B. (2017). Mixtures of lightweight deep convolutional neural networks: Applied to agricultural robotics. *IEEE Robotics and Automation Letters, 2*(3), 1344–1351. https://doi.org/10.1109/LRA.2017.2667039

McLeod, R. (2018). Annual costs of weeds in australia. esys development pty limited. Centre for Invasive Species Solutions, Canberra

Mortensen, A.K., Skovsen, S., Karstoft, H., Gislum, R. (2019) The oil radish growth dataset for semantic segmentation and yield estimation. In *2019 IEEE/CVF conference on computer vision and pattern recognition workshops (CVPRW)* (pp. 2703– 2710). IEEE. https://doi.org/10.1109/CVPRW.2019.00328 . https://ieeexplore.ieee.org/document/9025602/

Muruganantham, P., Wibowo, S., Grandhi, S., Samrat, N. H., & Islam, N. (2022). A systematic literature review on crop yield prediction with deep learning and remote sensing. *Remote Sensing, 14*(9), 1990.

Ojala, T., Pietikainen, M., & Maenpaa, T. (2002). Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 24*(7), 971–987. https://doi.org/10.1109/TPAMI.2002.1017623

Osorio, K., Puerto, A., Pedraza, C., Jamaica, D., & Rodríguez, L. (2020). A deep learning approach for weed detection in lettuce crops using multispectral images. *AgriEngineering, 2*(3), 471–488. https://doi.org/10.3390/agriengineering2030032

Rahman, M.A., Wang, Y. (2016) Optimizing intersection-over-union in deep neural networks for image segmentation. In *International Symposium on Visual Computing* (pp. 234– 244). https://doi.org/10.1007/978-3-319-50835-1_22 . Springer

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., & Fei-Fei, L. (2015). ImageNet large scale visual recognition challenge. *International Journal of Computer Vision*. https://doi.org/10.1007/s11263-015-0816-y

Sadgrove, E., Falzon, G., Miron, D., & Lamb, D. (2017). Fast object detection in pastoral landscapes using a Colour Feature Extreme Learning Machine. *Computers and Electronics in Agriculture*. https://doi.org/10.1016/j.compag.2017.05.017

Sadgrove, E., Falzon, G., Miron, D., & Lamb, D. (2018). Real-time object detection in agricultural/remote environments using the multiple-expert colour feature extreme learning machine (MEC-ELM). *Computers in Industry*. https://doi.org/10.1016/j.compind.2018.03.014

Sadgrove, E., Falzon, G., Miron, D., & Lamb, D. (2021). The segmented colour feature extreme learning machine: Applications in agricultural robotics. *Agronomy, 11*(11), 2290. https://doi.org/10.3390/agronomy11112290

Sadgrove, E., Falzon, G., Miron, D., Lamb, D. (2017). Fast object detection in pastoral landscapes using a multiple expert colour feature extreme learning machine. Asian-Australasian Conference on Precision Pastures and Livestock Farming **1**

Saleem, M. H., Potgieter, J., & Arif, K. M. (2019). Plant disease detection and classification by deep learning. *Plants, 8*(11), 468.

Schori, D., Anken, T., Seatovic, D. (2019). Using Fully Convolutional Networks for Rumex Obtusifolius Segmentation, a Preliminary Report. In *2019 International Symposium ELMAR* (pp. 119– 122). IEEE, Zadar, Croatia . https://doi.org/10.1109/ELMAR.2019.8918914

Smith, L. N., Byrne, A., Hansen, M. F., Zhang, W., Smith, M. L. (2019). Weed classification in grasslands using convolutional neural networks. In M. E. Zelinski, T. M. Taha, J. Howe, A. A. Awwal, K.M. Iftekharuddin (Eds.), *Applications of machine learning* (p. 42. SPIE, San Diego, California) https://doi.org/10.1117/12.2530092

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.-C. (2018). Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4510– 4520). https://doi.org/10.1109/CVPR.2018.00474

Skovsen, S., Dyrmann, M., Mortensen, A.K., Laursen, M.S., Gislum, R., Eriksen, J., Farkhani, S., Karstoft, H., Jorgensen, R.N. (2019). The grassclover image dataset for semantic and hierarchical species understanding in agriculture. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops* (pp. 0–0).

Touvron, H., Cord, M., Sablayrolles, A., Synnaeve, G., Jégou, H. (2021). Going deeper with image transformers. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 32– 42).

Van Klompenburg, T., Kassahun, A., & Catal, C. (2020). Crop yield prediction using machine learning: A systematic literature review. *Computers and electronics in agriculture, 177*, 105709.

Zeiler, M.D., Krishnan, D., Taylor, G.W., Fergus, R. (2010). Deconvolutional networks. In *2010 IEEE computer society conference on computer vision and pattern recognition* (pp. 2528– 2535). IEEE. https://doi.org/10.1109/CVPR.2010.5539957.

Zhang, W., Hansen, M. F., Volonakis, T. N., Smith, M., Smith, L., Wilson, J., Ralston, G., Broadbent, L., Wright, G. (2018). Broad-leaf weed detection in pasture. In *2018 IEEE 3rd international conference on image, vision and computing (ICIVC)* (pp. 101– 105). IEEE, Chongqing, China. https://doi.org/10.1109/ICIVC.2018.8492831

Zhang, X., Li, N., Ge, L., Xia, X., Ding, N. (2020). A unified model for real-time crop recognition and stem localization exploiting cross-task feature fusion. In *2020 IEEE international conference on real-time computing and robotics (RCAR)* (pp. 327– 332). https://doi.org/10.1109/RCAR49640.2020.9303270

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Authors and Affiliations

Jonathan Ford[1] · Edmund Sadgrove[1] · David Paul[1]

✉ Jonathan Ford
jford3@myune.edu.au

Edmund Sadgrove
esadgro2@une.edu.au

David Paul
dpaul4@une.edu.au

[1] School of Science and Technology, University of New England, Armidale, NSW 2351, Australia