MONITORING MAIZE PHENOTYPE WITH UAV IMAGE VIA COMPUTER VISION TECHNIQUES

ALPHAVETICAL ORDER. AUTHORS CONTRIBUTE EQUALLY

Ming-Chiang Chang
Department of Material Science and Engineering
Cornell University
Ithaca, NY 14853
mc2663@cornell.edu

Ruyu Yan
Department of Computer Science
Cornell University
Ithaca, NY 14853
ry233@cornell.edu

ABSTRACT

Crop phenotyping is an essential practice for farming and plant breeding, while traditional methods require a heavy labor force input. In this paper, we present a vision-based system for automated phenotyping for maize. We trained a crop row detector based on the Faster R-CNN framework, which allows us to further analyze the aerial image data at the plot level. We performed precise normalized difference vegetation index (NDVI) calculation on the predicted rows, applying plant segmentation with thresholding. We further present a latent vector learning method based on the variational autoencoder architecture to track the varying phenotypical representation over the lifespan of maize. Such feature vectors will be used for differentiating between genotypes.

1 Introduction

Maize is one of the most important cereal crops in the world. Monitoring its growth is essential for not only farming practices but also selective breeding. In agricultural studies, a combination of field survey and unmanned aerial vehicle (UAV) imaging is one of the most common data collection methods for monitoring. Nevertheless, it requires a large amount of humanitarian effort to do precise phenotyping of the crops. The research on selective breeding requires phenotyping data including crop's size and density, flowering days, normalized difference vegetation index (NDVI), and end of season yield to compare between genotypes. Some of the information, which is currently gathered by field survey, is largely preserved in the UAV images but not extracted in practice. In addition, obtaining abundant UAV images is takes little time and human effort because it can be easily automatized, thus making it a perfect playground for data-hungry machine learning techniques. We aim to make the data processing procedure autonomous by training neural networks, which can significantly reduce the labor input and requirement for expertise in the fieldwork.

2 Problem Definition

In plant science, one important experiment is to modify the genotypes of crops and find one gene combination that gives intended improvements in plants phenotype. The improvement can be in growth rate, flowering timing, yield, etc. However, as there are many potential combinations of gene modification, usually many experiments are conducted at the same time on a large field and it becomes difficult for human to keep track of how each genotype is performing. We intend to use machine learning to help agriculture experimentalist with this difficulty.

2.1 Data Description

The data are from the Robbins lab at the College of Agriculture and Life Sciences (CALS) at Cornell University. The raw image data is captured by a MicaSense camera with five spectra bands (blue, green, red, red edge, and near infrared), each of which is taken at a slightly different position due to the stability of the UAV. Each image includes a

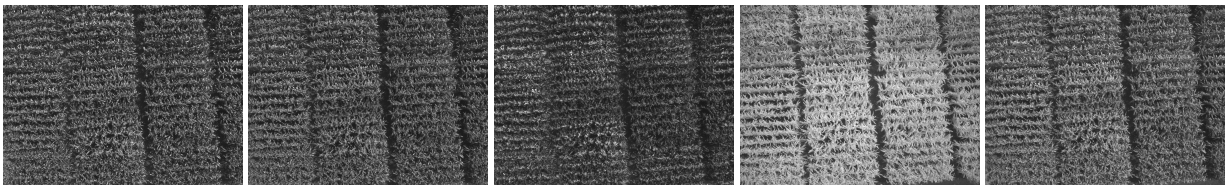

Figure 1: Example raw data (a) Blue (b) Green (c) Red (d) Red edge (e) Near infrared channel

portion of the field that includes about 100 plots, which is what people in agricultural field call a row of crop. Each batch was taken on a different date of the same field. We are given low resolution data of separated plots for 10 different dates (each are separated by 10 days on average) and raw images from the UAV for three different dates (each separated by about a month). A set of example raw data is shown in Figure 1.

2.2 Intended Results

The task of monitoring maize phenotype can be divided into two stages. The goal of the first stage is to process the orthoimages (multi-channel images) of the maize field by identifying the locations of crop rows. Since the experimental unit is usually one or two row of maize, isolating the plots in an UAV image is a critical and fundamental task for monitoring the phenotype of the plants.
The second stage is to extract useful information from the cropped plot images. With the time series data, it is desired to have a model that is capable of giving the relative size/age of a certain plot by comparing it to a reference plot.
In addition to phenotype, Normalized Difference Vegetation Index (NDVI) is a good indication of how much the crop is growing. By using the result from the first stage and obtaining homography of different channels, the NDVI can be readily calculated and give useful comparison between different genotype.

3 Related Work

Crop row detection is not a particularly new task to the computer vision field. Previously, there are some work done using Hough transform [Rovira-Más et al., 2005, Winterhalter et al., 2018] but the this method degrades significantly when there are image distortions or noises. Recently, neural network methods are applied to these problems. [Pang et al., 2020] trained a faster R-CNN and uses some algorithm to accurately find corn rows. However, this study only work with sparsely planted sprouts, which is a much easier problem than ours. [Osco et al., 2021] propose a way to train a network that can effectively obtain both row bounding boxes and plant counts. However, it does not have the concept of plots so it cannot be used directly for our purpose.
Crop density measurement is crucial for precision agriculture. Knowing the density of crops is the first step to help farmers make plans strategically or to help related departments do crop surveys quickly. Previous works on corn focus on either small corn sprouts[Kitano et al., 2019] or fully grown corn[Ma et al., 2021]. From the best of our knowledge, there is currently no model that can identify corn at all stages. Generalizability at different lighting conditions is also not yet clearly demonstrated. Getting labeled data for densely packed crop field is also very expensive and sometimes impossible. In our case, isolated plots carry enough information since it is the basic unit of the experiment and thus is what we choose to focus on. Crowd counting [Dai et al., 2021] is a similar topic but more complicated in many aspects because human have much more diversified appearance and it is still a research frontier.
[Li et al., 2020] reviewed many previous works on monitoring the growth of crops using 3D point cloud data. However, collecting and analyzing 3D data is much more laborious than 2D data (e.g. image data collected by a UAV) and UAV images already includes most of the information needed for gene engineer monitoring.

4 Method

4.1 Image preprocessing

Figure 2a shows an example of the raw data. The maizes are planted in rows and are separated into plots. The images from UAVs are often slightly skewed and the rows of maize are not horizontal. This is problematic for object detection because bounding boxes are ill-defined if the row of maize is not horizontal, especially when the rows are really tight. This can also make non-maximum suppression unreliable. To adjust the rotation, we assume that a straight up image

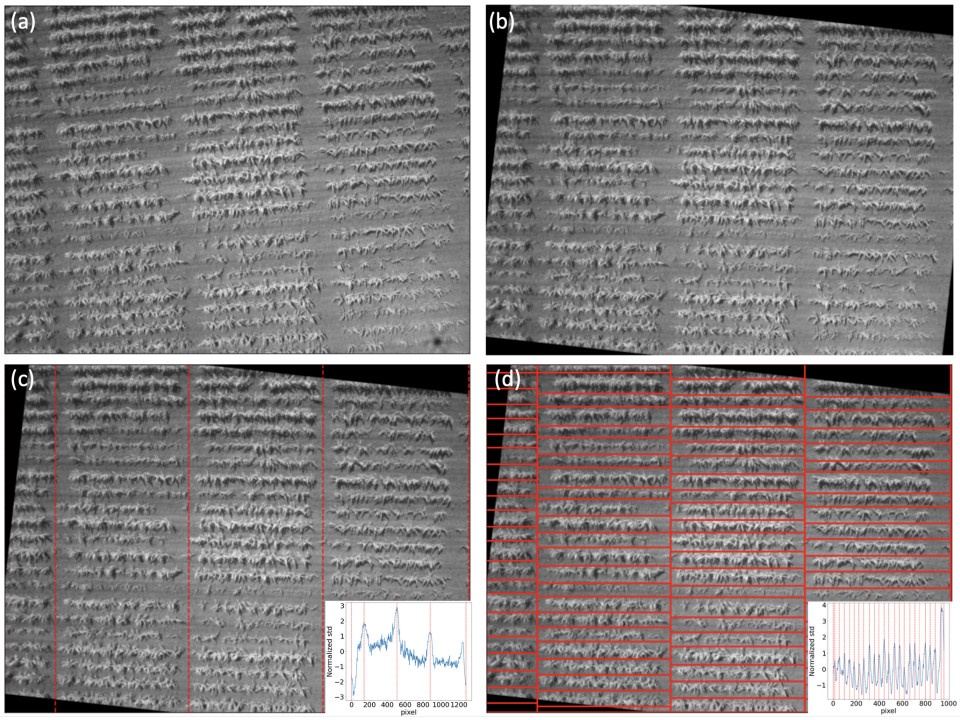

Figure 2: (a) An original UAV image. (b) An properly-rotated image. (c) Rotated image with row separated. Inset is the result of doing peak finding on normalized standard deviation. (d) Rotated image with plots separated. Inset is the result of doing peak finding on intensity of rows.

should have the largest variation in the vertical direction (perpendicular to the rows).The rotation correction algorithm rotates the raw image by an angle within 20 degrees, and search for the proper rotation that maximizes the variation of the column sum. The rotated image is shown in Figure 2b. Afterward, to segment the image into blocks, by utilizing the fact that the ground has a significant difference in terms of image intensity with the maize, plots can be separated by applying peak-finding algorithms on the intensity. This allows us to draw rough bounding box for each of the plot, as shown in Figure 2c-d. This data can then be used to pretrain the network described in the Section 4.2.1. Finally, to fully utilize the multi-channel data, we have to align the slightly shifted images of five different channels by applying transformations with homography. We use the SIFT descriptor since it is invariant to illumination changes, which is equivalent to the difference due to the spectra band. With the feature correspondence, we find the homography with RANSAC. Finally, images in five channels are all straightened up and aligned.

4.2 Maize row detection

4.2.1 Network Structure

Faster R-CNN [Ren et al., 2015] implemented in Detectron2 [Wu et al., 2019] is used to train a neural network that is capable of detecting the bounding boxes of maize rows. The network is separated into 3 parts: first is the backbone network which generates the feature maps that act as the input of the later two networks; the next part is the regional proposal network, whose purpose is to propose the bounding boxes; finally, the region of interest (ROI) head is another network that takes the bounding box locations and the feature maps as input and fine-tune the bounding boxes and classification results. We will go through the exact structure of each of the network in the following paragraph.
Backbone network consisit of a Resnet-50 network [He et al., 2016] that works parallel with a Feature Pyramid Network (FPN) [Lin et al., 2017]. Resnet-50 start with a basic 2d convolutional layer with 7x7 kernel followed by a maxpooling layer. The output is passed into chains of bottleneck blocks, which consist of three convolution layers in serial, with the middle one having a smaller number of channels, and a shortcut with or without convolution. The FPN takes output feature maps at 5 different dimensions, in this case with 1/4, 1/8, 1/16, 1/32 size of the original map, and pass them through a 1x1 convolutional network. The result of this network is then separated into two streams. The first passes through another 3x3 convolutional network as an output. The other is upsampled by a convolutional network to have the same dimension and number of feature maps as the previous feature map output from the Resnet-50 and is added to that

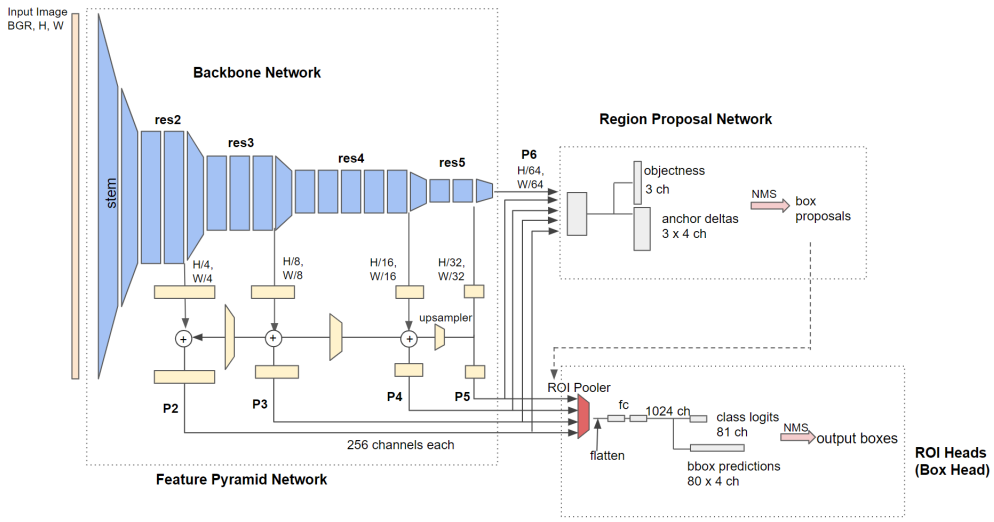

Figure 3: Detailed architecture of Base-RCNN-FPN.[Honda]

feature map output. The rest of the network does this recursively so the FPN also has 4 streams of output feature maps to the RPN and ROI heads. This structure allows feature information to back propagate and performs better at training. Each of the 256 channel feature maps at different level/scale is pass into the RPN separately. The RPN consist of a 3x3 2d convolutional layer and the output of which is passed into one 1x1 convolutional layer that condense the output to 1 channel, which is the objectness logit score, and another 1x1 convolutional layer that condense the output to 4 channels, which is describe the bounding box position. After the bounding box proposal, boxes that are in the top 2000 in terms of objectiveness are selected. None-maximum suppression is applied and finally output 1000 bounding box proposals. During training, before feeding the proposals to ROI head block, the proposals go through proposal sampling, which separates proposals into foreground and background by setting the threshold IOU of each of the bounding boxes; if a proposal has a IOU with a ground truth box that is larger than the threshold, then it is declared as foreground. Otherwise the proposal counts as background. The sampling then randomly chooses the proposal so that the sampled boxes has a predefined ratio of foreground and background. To crop the region that corresponds to the proposal from the feature map, ROIAlign described in [Ren et al., 2015] is used and each ROI is pooled to have size 7×7. The ROI is then passed through 2 fully connected layer and turned into a $\mathbb{R}^{1024}$. Then this vector is passed into two different fully connected layer that produces the class score (one number) and the box deltas (4 numbers).
The training is done by stochastic gradient descend using binary cross entropy loss as the lost function for classification and l1 loss as the loss function for the location mismatch in both RPN and ROI head.
For more detail about the network configuration and how to train it, please refer to [Ren et al., 2015] and the Detectron2 package (the rpn_R_50_FPN_1x configuration is used).

4.2.2 Pretraining

The network is initialized with the weights trained on ImageNet [Deng et al., 2009], which is included in the Detectron2 package. However, the aspect ratio of a single row of corns is very specific and is not typically the ratio of everyday objects in popular models, which are trained on COCO or ImageNet. Here, the scale parameter of the RPN is set to include only 0.5 and 1.0, so that RPN does not generate shapes that are unrealistic. In addition, because labeled data is expensive and limited, the heuristic described in preprocessing are applied to quickly generate large amount of psuedo-training data and to pretrain the network so it is properly initialized. The images of the edges of the field is excluded from the pretraining data because our heuristic does not produce accurate data for those images. As shown in later section, pretrained network already has some concept of the rows of maizes and therefore is a very good starting point for actual training.

4.2.3 Training

131 images are hand labeled and each has about 100 plots in it, which make the number of total training labels to be around 13000. We particularly include more hand-labeled images for the edges of the field to improve the accuracy of the model on these kind of images. The labeled data are augmented by mirroring them sideways and up-side-down, and

by adding Gaussian noise (524 images after data augmentation). Training is done using the simple SGD with same hyperparameters in pretraining, but the learning rate is set to 0.001 for the purpose of fine-tuning.

4.3 Autonomous Maize Phenotyping

4.3.1 NDVI Calculation

The normalized difference vegetation index (NDVI) is a graphical indicator that measures photosynthetic activity of plants. It is computed with the near-infrared and red reflectance by

$$NDVI = \frac{NIR - Red}{NIR + Red}$$

Low NDVI values indicate moisture-stressed vegetation and higher values indicate a higher density of green vegetation. For maize, the NDVI value increases in the growing season, and gradually decrease when the plant enters the tasseling stage. Crops of the same genotype are expected to have a similar NDVI growth curve, which makes it an important phenotype to track for selective breeding.
The common approach for computing plot-level NDVI is taking the average values from the near-infrared and red sub-graphs. However, it introduces noise from the pixels of the ground. With well-aligned images of the blue, green, red, red-edge, and near-infrared channels, we can accurately compute the average NDVI for each plot in the field. We observed that the plants are most distinguishable with a high pixel value in the red-edge channel. We then apply a threshold to the red-edge channel to segment out a mask for plants in the image. Finally, in each predicted bounding box, we compute NDVI only for the plant pixels and take the average.

4.3.2 Latent Vector Learning

Due to the high degree of non-linearity of a plot image, linear feature extraction method like nonnegative matrix factorization does not work well on the maize data. Thus, a variational autoencoder is used to learn a latent manifold of maize plots. The implementation is based on a git repository [Pidhorskyi, 2019]. For the encoder, there are 5 consecutive convolution layer + batch normalization [Ioffe and Szegedy, 2015] pairs. All convolution layers uses 4×4 kernel and has stride of 2 in both dimensions, thus the output feature map has 1/2 size in both dimensions. On the other hand, the number of channels for the first layer is 64 and is doubled for every layer. The input image is re-scaled to 128×128 and the output of encoder is a $4 \times 4 \times 2048$ tensor. This output is then flattened and be fed into two fully connected layer with output size of 512, which is the learned mean and standard deviation of the image in the latent space. The decoder has the inverse structure of the encoder. It takes the latent vector $z \in \mathbb{R}^{512}$ and outputs a $\mathbb{R}^{128 \times 128}$ matrix. The training process is done by SGD using an ADAM optimizer [Kingma and Ba, 2014] with learning rate of 0.5, $\beta_1 = 0.5$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$ and weight decay of 10^{-5}. The loss function is the reconstruction error plus KL divergence the predicted mean and standard deviation value in the latent space with normal distribution.

4.3.3 Phenotype Tracking

With the trained VAE, each of the plot can be encoded to a vector in the latent manifold $a_i \in \mathbb{R}^{512}$ where i is the time order. The ith plot matrix $\mathbf{A}_i \in \mathbb{R}^{512 \times 10}$ can be constructed by stacking the vector in the time order. To learn a growing curve in the latent space, we assume that there is a linear transformation that map the latent space vector to the vector that describe the number of days after planting $\mathbf{d} \in \mathbb{R}^{10}$. With the linear assumption, the transformation x can be found by solving the following least square optimization problem

$$\min_{\mathbf{x}} ||\mathbf{A}_i^T \mathbf{x} - \mathbf{d}||^2 \tag{1}$$

Since we do not have a ground truth for this, we can only compare all of the plots to a standard reference plot. The standard reference plot is chosen by the plot that has the smallest square error for the fit, which is to find solution for the following equation

$$\operatorname*{argmin}_i (\min_{\mathbf{x}} ||\mathbf{A}_i^T \mathbf{x} - \mathbf{d}||^2) \tag{2}$$

The learned $\mathbf{x}$ vector can then be used to map each encoded latent vector to a reference day-after-plant number and can be used as a way to track the phenotype of maizes. This works if there exists a linear mapping. This can also be generalized to non-linear mapping by replacing L2-norm with other loss function. In a most extreme case, one can train a neural network to learn such mapping.

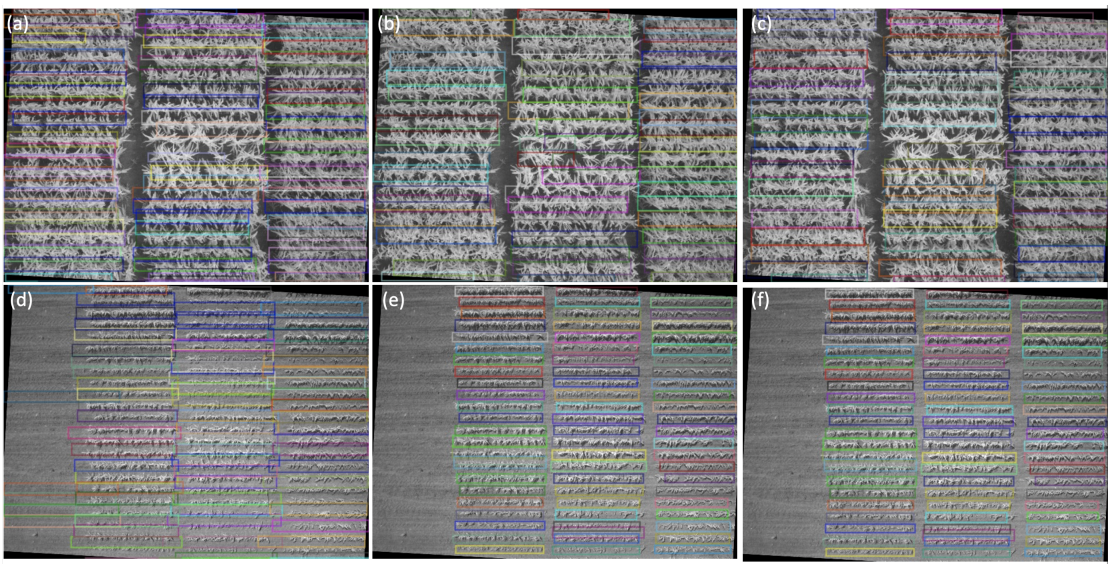

Figure 4: Detection results. (a)(d) Detection results from pretrained network (b)(e) Detection results from directly-trained network (c)(f) Detection results from pretrained+training network

Table 1: Average precision test results. (Done using the package developed in Padilla et al. [2021])

Training methods	AP
Pretrain	72.19%
Direct training	91.17%
Pre+training	**94.26%**

5 Experiment Results

5.1 Crop Row Detection

To demonstrate the pretraining process is necessary, we test the average precision (AP) of each of the following cases: 1. trained on pretraining data. 2. directly train on labeled data. 3. trained on pretraining data then do fine-tuning with hand-labeled data (we call it pre+training). The output of each case is listed in Figure 4. Because these models work pretty well on easy cases, e.g. early in the growth season when there is no overlapping between plants and when the plots filled the whole image, only harder cases are showcased. Figure 4a-c are the results from each of the training method on the image in which maize stands start to overlap each other. For the pretraining model, it is understandable since the heuristic does not always find the best bounding box. Comparing the results from directly-trained model and pre+training model, while both do a decent job on the regions other than the center part, judging just by looking, the pre+training model misses less boxes and predicts bounding boxes that are more accurate. This statement is supported by the average precision test results listed in Table 1, where the pre+training model reaches a very good AP of 94.26%. The test is performed on 15 hand-labeled images that are not included in the training set. The reason that the pre+training model works better is becuase the pretraining act as a prior which gives a prior believe on where the bounding boxes are. Therefore, even when the model is confused, it can start with a pretty good guess of where the bounding box can be. In Figure 4d-f, models are tested on images on the edge of the field. In this case, both directly trained model and pre+training model perform well.

5.2 Maize Phenotype monitoring

NDVI of maizes at a certain time can be calculated using the detection result and the method described in Section 4.3.1. An example of segmentation and NDVI calculation is shown in Figure 5a-b. The mask does no perform perfectly but since NDVI is a ratio between channels so imperfect mask should still give good estimation of NDVI. In the result computed with the sampled data, we observed that crops planted in the same area have similar NDVI at the same time of measure, as shown in 5c. There is trend of NDVI increase between June and July and decrease afterwards, which matches the growing and tasseling seasons. Although we do not have ground truth data, we can make a preliminary

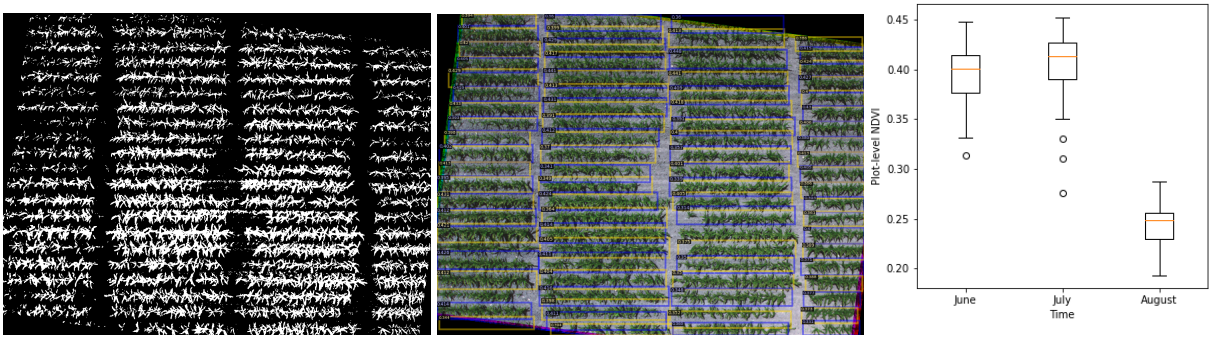

Figure 5: (a) Plant segmentation mask. (b) Reconstructed RGB image with NDVI computed in each predicted plot. (c) Plot-level NDVI on sampled data.

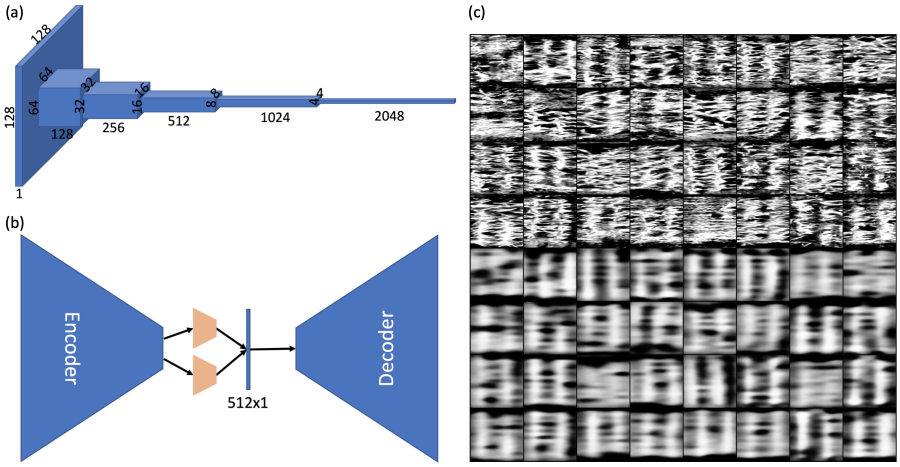

Figure 6: (a) Encoder network structure. There is a batch normalization layer between each convolution layer but is ignored for simplicity in the figure (b) The full VAE structure. Decoder is an inversed encoder. The orange networks are 32768 by 512 fully connected layers. (c) The original plot images (top half) and the reconstructed images (bottom half).

judgement that our calculation matches the expectation.
We attempted to train a VAE which has the structure described in section 4.3.2 and shown in Figure 6a-b. The reconstruction result of the VAE is shown in Figure 6c. Comparing the top half original images to the bottom half reconstruction images, we see that the VAE did learned some large scale structure from the original image set. However, no detail of the maize is learned, which is the critical part for doing phenotype tracking. Several parameters, including learning rate, latent space dimension, network depth and network channels, is adjusted to try to improve the learning result. Still, the images are still underfitted and we concluded that it is most possible that the data size is not enough to properly train this VAE. Since the latent space vector does not properly represent the feature of an image, doing the method we proposed in Section 4.3.3 will not give useful result. We describe how to move forward from this point in Section 6.

6 Future work

Individual maize plant detection is hard because of insufficient labeled data and the difficulty of labeling it. At the later stage of the growth season, the maize overlaps with each other to the extent that it is very difficult to draw the bounding box of a single maize stand for a human. With the data we have, which gives time stamps of the cornfield at different time periods, it is possible to compare the field at different dates and obtain reasonably good training data by referencing images and get the bounding boxes for single maize stand at earlier dates.
Performing our data analysis on the raw UAV images, we have not been able to register the plot-level phenotyping results to the experiment controls of corresponding rows in the field. The plant science researchers have been using

software for stitching the raw UAV images into a large image of the entire field, and hand label the bounding boxes of rows. We can potentially simplify the process by setting some markers around the field for calibration, and match the predicted bounding boxes to the experiment ids based on the sequential order. With this implemented, we can directly use results from our detection, which has much more data, to train the VAE. We believe that this framework will work much better than what we have now.
The other potential improvement is to have a neural network that takes a latent vector and map it to a number that represents days-after-planting. If we can have ground truth for each row, this network can be trained together with the VAE and add constraint on what the latent space manifold can be. With this, the assumption that there exist an linear mapping between latent space and days-after-planting can be removed.

7 Conclusion

We proposed a way to train an end-to-end system for maize phenotyping using UAV image data. The Faster R-CNN based detector segments rows of crops in the field and enables precise plot-level NDVI computation, which saves labor force for hand labeling. We further proposed a latent vector learning method using VAE that can be used for inferring the growing stage of the crop. The goal for autonomous phenotyping is to help with differentiating between genotypes in selective breeding.
The proposed crop row detector and plot-level NDVI calculation are robust enough with our test data set, but we still need to generalize the result by comparing against the hand-labeled data from the Robbins Lab. Future works will be oriented towards multi-modal data registration and data set construction for training the VAE model.

8 Source code

The source code is on the following github repository: https://github.com/MingChiangChang/maize_phenotype/. We are not allowed to distribute the image data yet.

References

F Rovira-Más, Q Zhang, JF Reid, and JD Will. Hough-transform-based vision algorithm for crop row detection of an automated agricultural vehicle. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, 219(8):999–1010, 2005.

Wera Winterhalter, Freya Veronika Fleckenstein, Christian Dornhege, and Wolfram Burgard. Crop row detection on tiny plants with the pattern hough transform. *IEEE Robotics and Automation Letters*, 3(4):3394–3401, 2018.

Yan Pang, Yeyin Shi, Shancheng Gao, Feng Jiang, Arun-Narenthiran Veeranampalayam-Sivakumar, Laura Thompson, Joe Luck, and Chao Liu. Improved crop row detection with deep neural network for early-season maize stand count in uav imagery. *Computers and Electronics in Agriculture*, 178:105766, 2020.

Lucas Prado Osco, Mauro dos Santos de Arruda, Diogo Nunes Gonçalves, Alexandre Dias, Juliana Batistoti, Mauricio de Souza, Felipe David Georges Gomes, Ana Paula Marques Ramos, Lúcio André de Castro Jorge, Veraldo Liesenberg, et al. A cnn approach to simultaneously count plants and detect plantation-rows from uav imagery. *ISPRS Journal of Photogrammetry and Remote Sensing*, 174:1–17, 2021.

Bruno T Kitano, Caio CT Mendes, André R Geus, Henrique C Oliveira, and Jefferson R Souza. Corn plant counting using deep learning and uav images. *IEEE Geoscience and Remote Sensing Letters*, 2019.

Yong-Yang Ma, Zhan-Li Sun, Zhigang Zeng, and Kin-Man Lam. Corn-plant counting using scare-aware feature and channel interdependence. *IEEE Geoscience and Remote Sensing Letters*, 2021.

Feng Dai, Hao Liu, Yike Ma, Xi Zhang, and Qiang Zhao. Dense scale network for crowd counting. In *Proceedings of the 2021 International Conference on Multimedia Retrieval*, pages 64–72, 2021.

Zhenbo Li, Ruohao Guo, Meng Li, Yaru Chen, and Guangyao Li. A review of computer vision technologies for plant phenotyping. *Computers and Electronics in Agriculture*, 176:105672, 2020.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28:91–99, 2015.

Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. `https://github.com/facebookresearch/detectron2`, 2019.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017.

Hiroto Honda. Digging into detectron 2 — part 1. `https://medium.com/@hirotoschwert/digging-into-detectron-2-47b2e794fabd`. Accessed: 2021-12-15.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

Stanislav Pidhorskyi. Vae. `https://github.com/podgorskiy/VAE`, 2019.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Rafael Padilla, Wesley L. Passos, Thadeu L. B. Dias, Sergio L. Netto, and Eduardo A. B. da Silva. A comparative analysis of object detection metrics with a companion open-source toolkit. *Electronics*, 10(3), 2021. ISSN 2079-9292. doi:10.3390/electronics10030279. URL `https://www.mdpi.com/2079-9292/10/3/279`.