EE244 Final Project: Crop & Weed Classification

Abstract

Agricultural robots could help growers hard-pressed by increasing costs and decreasing labor supplies by automating growing and harvesting tasks. Weeding is an important and critical component of growing health crops; improper weeding can have devastating effects on a grower's yield. Yet, before a robot can mechanically remove the weeds, the automated system needs to localize and classify the weeds to avoid accidentally removing the desired crop. This paper covers a preliminary attempt to use RGB images to classify sugar beets and 9 commonly identified weeds.

Git Repository: `https://github.com/mepix/CropWeedClassification`

1 Introduction

Growers have long struggled with pest and weed abatement when growing healthy crops. With a shrinking agricultural labor pool, growers have been turning to automated techniques to remove weeds and maintain crop health. While there are several companies (FarmWise, Carbon, etc), accurately distinguishing crop from harmful weeds remains an active area of research. Weeds can dramatically impact crop yield. Between 2007 and 2013, weeds resulted in $27 Billion crop loss for corn farms and $16 Billion for soy farmers (around 50% loss for each crop). Accurately detecting weeds using computer vision methods could save growers billions of dollars annually. This project utilizes an RGB image data set to train a ResNet classifier to distinguish between sugar beets and weeds.

2 Related Works

2.1 Machine Learning

Researchers have been studying both machine learning and robotics for decades. While classical approach such as K-Nearest Neighbors (KNN) and K-Means Algorithms are used for labelling and grouping objects within some environments, recent work has focused on neural networks and deep learning. All machine learning approaches require data. One data set that researchers utilize to benchmark classification approaches is the CIFAR-10 data set [4]. This data set contains 60,000 images that belong to 10 different classes. One neural network that performs well on this dataset is ResNet [3]. ResNet is a residual network that overcomes some of the problems that can occur with overfitting when training a deep neural network. Yolo is another commonly utilized network architecture for classification [7].

2.2 Agricultural Robotics

In the agricultural domain, researchers have largely been focused on harvest automation. While significant work has been done on end effector design, path planning, and control methodologies. Recently, researchers have focused on improving the growing process with automated weeding. Work on this task requires both robust system design and massive amounts of data for machine learning to distinguish between weeds and crops. One notable research group working on the data problem is the Stachniss Lab at the University of Bonn in Germany [2]. They have developed a robotic platform

called PhenoRob[1] to collect high fidelity image and senor data of growing crops. Other researchers have focused on training classifiers using the Yolo3 neural network and design optimized systems for eliminating the weeds [6, 9]. Figure 1 shows two of these systems. Partel et al. was able to train a binary classifier based upon Yolo3 to achieve an average precision of 85% and a recall of 95% when classifying pepper plants and weeds using 1000 images.

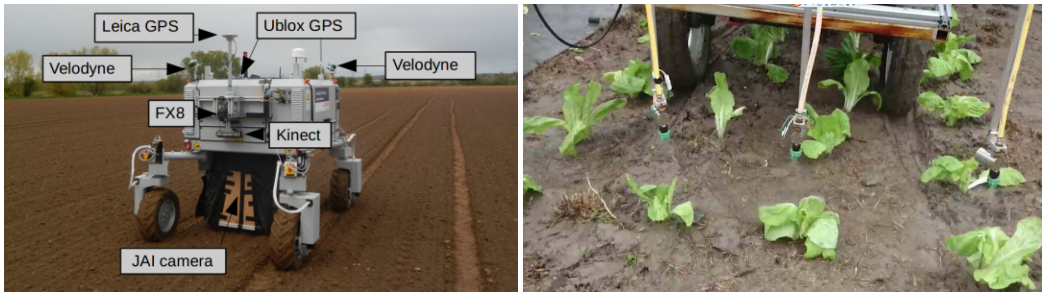

Figure 1: The data collection research platform PhenoRob developed by the Stachniss Lab (left)[2] and an automated camera and spray system (right)[9].

Industry has also recently been making more inroads into agricultural automation. Initial attempts at industry were originally focused on broad acre row crops (corn, wheat, soy, etc.), but as the price of sensors and computers have reduced in price, a new focus as emerged to focus on specialty crops (strawberries, tomatoes, and leafy greens). One of the early movers in this domain was Blue River Technologies which invented a "see and spray" system [8]. This system uses a camera system and a neural network to localize, classify, and spray weeds. More recent enterprise has focused on pesticide free means to identify and remove weeds. FarmWise Robotics mechanical removes weeds with a tool while Carbon Robotics uses a laser to kill the weeds (Figure 2). All of these systems require accurate classifiers to distinguish desirable crop from weeds to avoid reducing yield and ruining the grower's profit.

Figure 2: FarmWise Robotics (left) uses a mechanical tool to remove weeds while Carbon Robotics (right) uses a laser. Both systems uses computer vision to classify weeds and crops.

3 Technical Approach

3.1 Data Set Selection

Within the agricultural domain, there are several data sets available for the distinguishing crop from weeds [5]. This project utilizes the Sugar Beets 2016 Data Set from the Stachniss Lab at the University of Bonn in Germany [2]. This data set contains 12,340 RGB and NIR images with pixel level annotations for three classes: crop, weed, and background. This data set contains a subset which contains 283 images with 9 classes of weeds annotated. Since the entire data set is 5TB, a subset of images will be selected for this project. Figure 3 shows an annotated mask and RGB image pair.

[1] `https://www.phenorob.de`

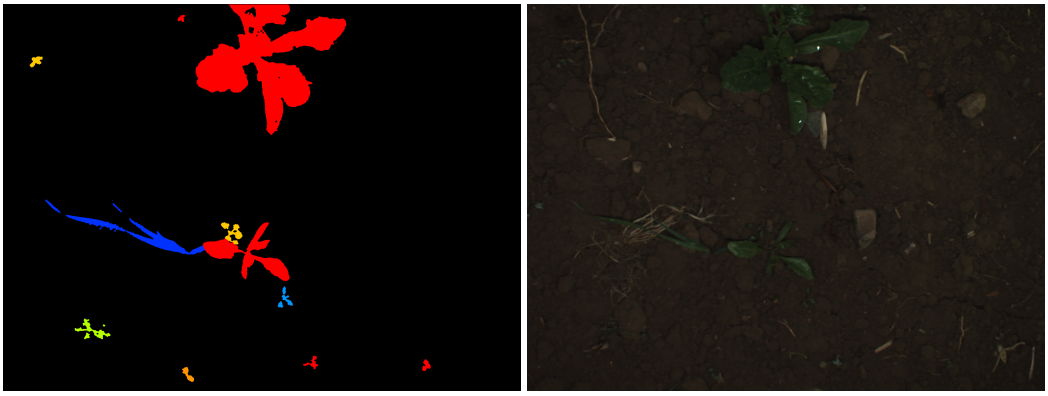

Figure 3: The Sugar Beets Data set contains pairs of annotated masks and RGB images. The annotated masks use the color red to indicate which pixels represent the sugar beets crop while the other shades represent the other weeds.

3.2 Data Set Preparation

The data set required significant processing before training the classifier. Two primary steps were involved with the data set preparation: extraction and cleanup. In the extraction stage, the paired annotated masks and RGB images were identified and copy from the larger data set into a separate folder for training. In the cleanup stage, the masks used to split the RGB images into smaller images to create the data set.

The extraction stage was necessary to parse through the original data set provided by the Stachniss Lab. Since the data set was extremely large (5TB), the data was broken down into several compressed files.[2] The downloaded folders contained subfolders for the annotated masks and the original RGB images. A Python script (`preprocessdata set.py`) was created to search the folders and match the annotated images with the RGB images and copy them into a new folder.[3] The matching was performed using the timestamp on the frame numbers. This process yield 251 image pairs, slightly fewer than the 283 image pairs claimed to be present in the data set.

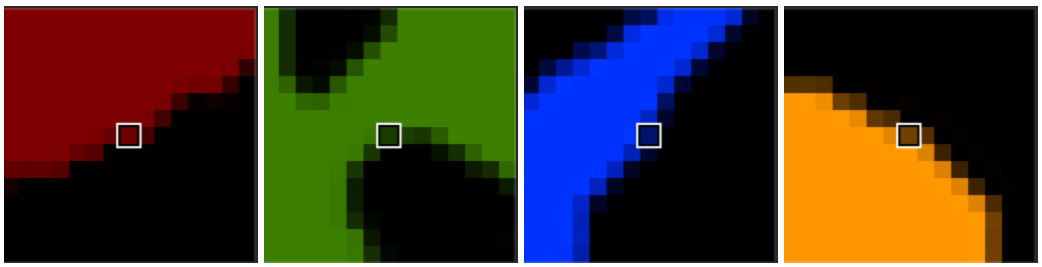

Figure 4: The annotated masks appeared to show signs of compression and/or scaling since the RGB values were not uniform across the masked regions. While this was most noticeable at the edges of the masks, there were also inconsistent pixel values within the boundaries of the masked regions.

After the images were extracted into a new folder, the masks need to be cleaned and then used to create a subset with more, smaller images. The masks appeared to show signs of compression and/or scaling since the RGB values were not uniform across the masked regions. While this was most noticeable at the edges of the masks, there were inconsistent pixel values within some of the masks as well (Figure 4). A second Python script was written (`cleanupmask.py`) to cleanup the masks.[4] The cleanup process used OpenCV[1] to create contours for each of the masks and then select a region of interest around the contour (Figure 5). A minimum contour area threshold was applied to eliminate

[2] `https://www.ipb.uni-bonn.de/datasets_IJRR2017/`

[3] `https://github.com/mepix/CropWeedClassification/blob/main/scripts/preprocessdataset.py`

[4] `https://github.com/mepix/CropWeedClassification/blob/main/scripts/cleanupmask.py`

regions that were too small to be a desired region of interest. Figure 6 shows a selection of the 3,998 created images.

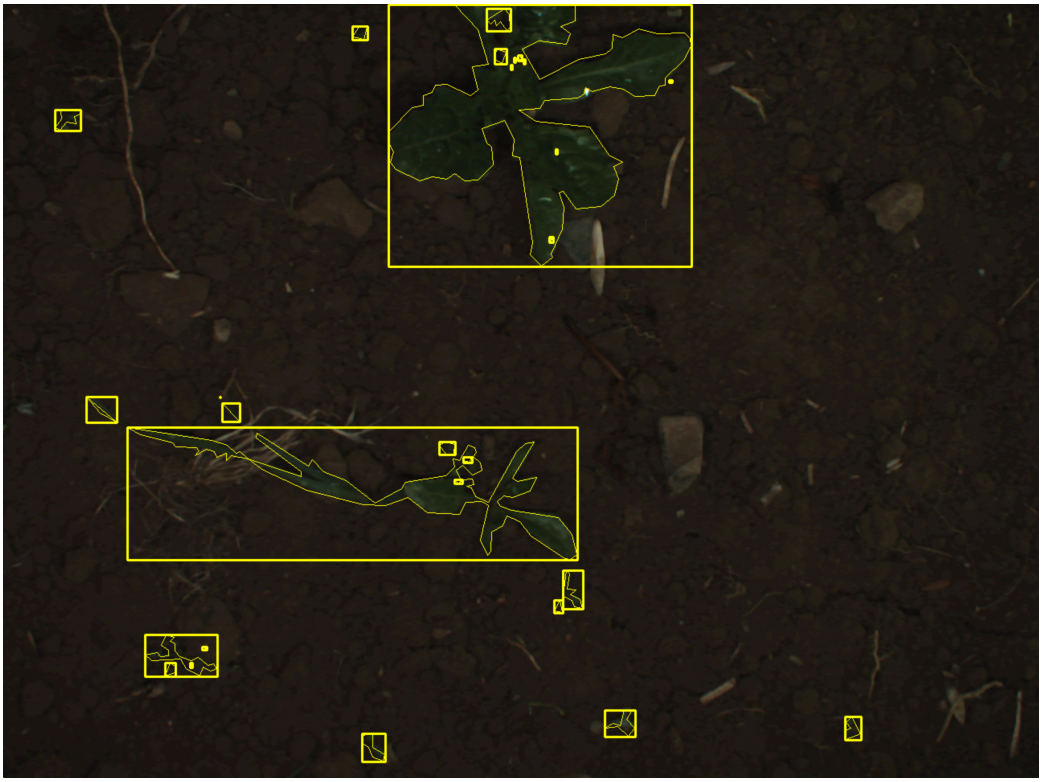

Figure 5: Contours were created from the masks and then used to define regions of interest in the RGB images. The smaller regions were filtered out and the remaining regions were used to create the training and test data.

Figure 6: The new images produced from the original data set. These images were created by using the annotated mask to determine a region of interest around a masked item. The region of interested was then scaled to a 512×512 region and then saved as a new image.

Next, the images were split into a set of training and test data with a 75%/25% split (2,966 / 1032 images). This ratio was selected since a 25% test subset yielded 10 or fewer images for some classes. A smaller percentage seemed insufficient for testing. Since the resulting data set was extremely skewed, a second data set was created with three classes: crop, big weeds, and small weeds. The small weeds category combined classes 2-9 (0-indexed) from the original 10-class data set. Figure 7 shows the data class distributions for both the 10-class and 3-class data set.

However, the approach used to generate the new data sets was not perfect. Figure 8 highlights some of the challenges associated with this approach that introduce bad data. Some images contained relatively small weeds, so the region of interest contained mostly dirt while other regions contained

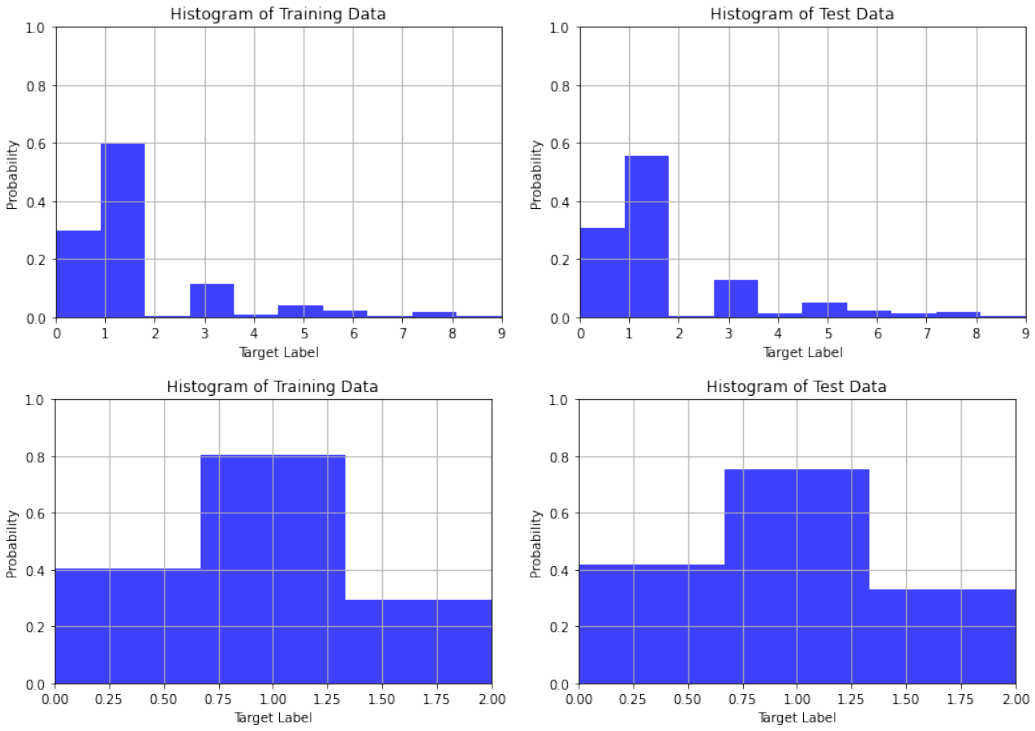

Figure 7: In the 10-class data set, class 0 represents the sugar beat crop while the remaining 9 classes are weeds (top). In the 3-class data set, class 0 still represents the crop, but classes 2 through 9 have been combined into a category for smaller weeds (bottom).

overlapping crop and weeds. The resulting data sets were skimmed to remove some of the worst images.

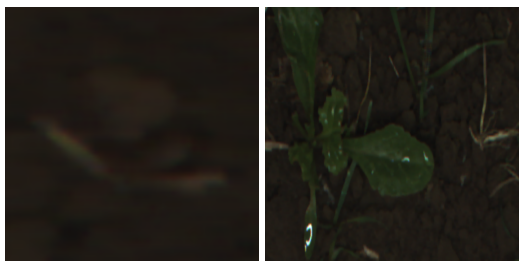

Figure 8: Not all of the images in new subset are ideal for training. Some images contained relatively small weeds, so the region of interest contained mostly dirt (left). Other instances contained overlapping crop and weeds, which made it hard to determine the ideal region of interest (right).

3.3 Classification

The classification software was developed using PyTorch and executed in Google Colab. PyTorch is a Python machine learning framework that includes software objects and functions for key machine learning tasks. The software developed for this project largely follows the official PyTorch tutorials and includes a data loader for the custom data set and training, and evaluation functions.[5] The primary classifier used for this analysis was ResNet18. Two other network architectures were evaluated (a simple 3 layer neural network and AlexNet), but were abandoned due to poor performance and insufficient resources to fully train, tune, and debug.

[5] `https://github.com/mepix/CropWeedClassification/blob/main/CropCNN.ipynb`

The software is structured so that key network parameters can be easily tuned empirically. For the data set, it was determined that a batch size of 32 images and 100 epochs worked well to guarantee convergence of the training set. The optimizer could be selected as either Adam or Stochastic Gradient Descent (SGD). For the Adam optimizer, the learning rate was set to 0.01, a step size of 25, and a gamma of 0.01. When the SGD optimizer was selected, the ideal learning rate appeared to be 0.001. The Adam optimizer seemed to provide better performance on the data set and was used to generate the final results. The goal of the network is to minimize a loss function to improve the accuracy of the classifier. Since this is a multi-class classifier, cross entropy loss was utilized (Equation 1).

$$l_n = -\sum_{c=1}^{C} \log \frac{\exp x_{n,c}}{\exp(\sum_{i=1}^{C} x_{n,i})} y_{n,c} \tag{1}$$

3.4 Metrics

The primary metrics used for analyzing the performance of the network are Precision, Recall, and the F_β-Criterion. Precision is the percentage of correctly labelled examples among those labeled as such by the classifier (Equation 2). In a search engine, precision is the fraction of returned results that is relevant to the original query. Recall is the percentage of correctly labelled examples identified as such by the classifier among all truly positive examples (Equation 3). Likewise, in a search engine, recall is the fraction of returned results that are successfully retrieved. The F_β-Criterion compares the relative importance of precision versus recall in the classifier (Equation 4). Since both precision and recall are of equal importance in the application of automated wedding, this analysis will use $\beta = 1$. This F_1-Score is also known as the dice coefficient (Equation 5).

$$Pr = \frac{N_{TP}}{N_{TP} + N_{FP}} \tag{2}$$

$$Re = \frac{N_{TP}}{N_{TP} + N_{FN}} \tag{3}$$

$$F_\beta = \frac{(\beta^2 + 1) \times Pr \times Re}{\beta^2 \times Pr + Re} \tag{4}$$

$$F_1 = \frac{2 \times Pr \times Re}{Pr + Re} \tag{5}$$

The network training time was also observed during the classification process.

4 Experimental Results

4.1 10-Class Data Set

4.1.1 Without Random Weighted Sampling

Initially, the classifier was trained using a random split of training and test data. However, this configuration produced a low accuracy for both the training and test data sets. The training loss plateaued at 1.1 and a peak accuracy of 0.60. For the test data, the global accuracy was 0.56. However, this number is deceiving since the classifier only ever assigns two labels: "Crop" or "Weed1" (Figure 9). This is likely due to the extreme skew in the distribution of the class instances within the data set. Table 1 shows the precision, recall, and F1 score for this classifier. This classifier achiever a relatively high precision and recall score for Weed1, but terrible metrics for Weeds 2-9.

4.1.2 With Random Weighted Sampling

In an attempt to compensate for the extreme bias in the data set, weighted random sampling was implemented in the training process. This process applied a weight to the samples so that the all samples were equally represented in the batch for each epoch. The results included a better

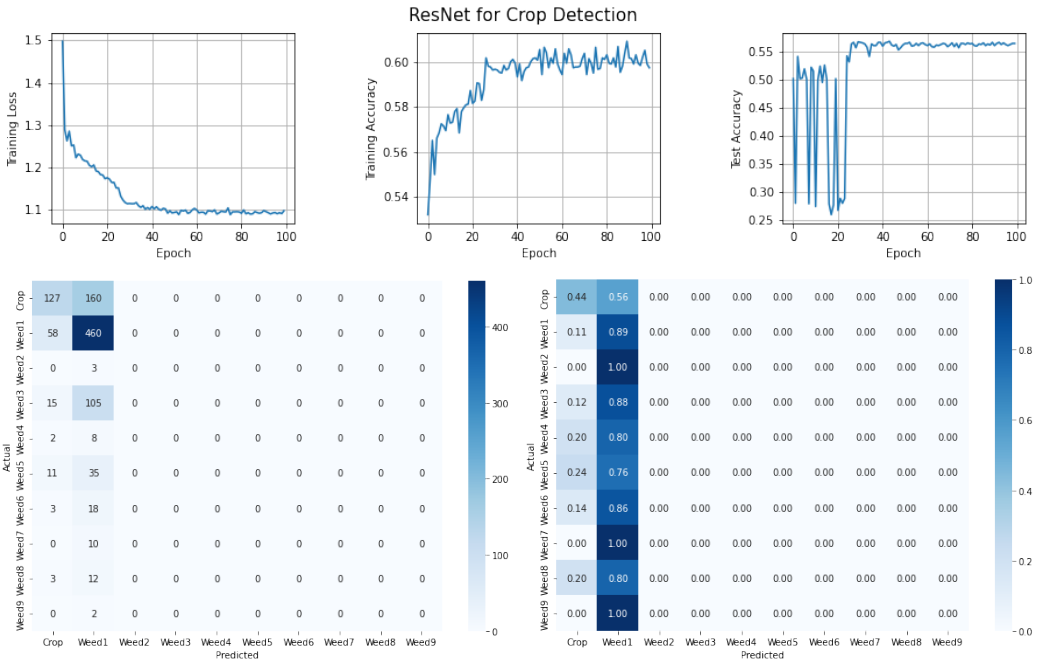

Figure 9: The classifier trained without weighted random sampling of the classes within the training set achieved poor performance. The classes Weed 2-9 are not well represented and are most frequently classified as Weed 1. This particular instance took 52 minutes and 19 seconds to train.

Table 1: Metrics Without Weighted Random Sampling

Labels	Precision	Recall	F1 Score
Crop	0.58	0.44	0.50
Weed1	0.57	0.89	0.69
Weed2	0.00	0.00	0.00
Weed3	0.00	0.00	0.00
Weed4	0.00	0.00	0.00
Weed5	0.00	0.00	0.00
Weed6	0.00	0.00	0.00
Weed7	0.00	0.00	0.00
Weed8	0.00	0.00	0.00
Weed9	0.00	0.00	0.00

representation of the classes, but still showed significant bias. The training loss plateaued at 0.08 and achieved a peak training accuracy of 0.97. However, the global test accuracy was only 0.48! The training plots and the confusion matrix for the results with weighted random sampling are shwon in Figure 10. This time around, other classes which previously had a recall of 0 now had a recall of 0.10. Table 2 contains the precision, recall, and F1 scores for this classifier. While somewhat improved, these results still demonstrate a highly biased classifier.

4.2 3-Class Data Set

After determining that the 10-class classifier achieved poor performance, a classifier was trained on a 3-class data set. This classifier achieved a training loss of 0.55 and a peak test accuracy of 0.77. The peak global test accuracy was 0.55 (Figure 11). While the precision and recall seemed to once again favor Class 1, now Weed (Big), the rest of the metrics improved for this classifier (Table 3).

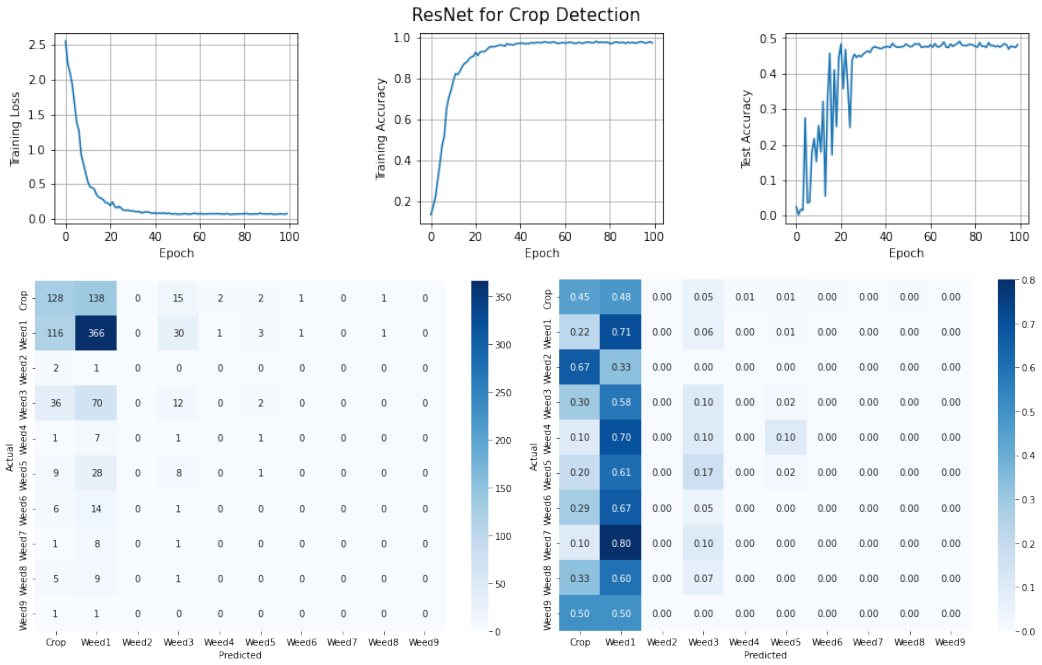

Figure 10: The classifier trained with weighted random sampling exhibited improved performance, but some classes were neither consistently nor accurately classified. This particular instance took 55 minutes and 39 seconds to train.

Table 2: Metrics With Weighted Random Sampling

Labels	Precision	Recall	F1 Score
Crop	0.42	0.45	0.43
Weed1	0.57	0.71	0.63
Weed2	0.00	0.00	0.00
Weed3	0.17	0.10	0.13
Weed4	0.00	0.00	0.00
Weed5	0.11	0.02	0.04
Weed6	0.00	0.00	0.00
Weed7	0.00	0.00	0.00
Weed8	0.00	0.00	0.00
Weed9	0.00	0.00	0.00

5 Conclusions

Overall, the trained classifiers performed poorly on the Sugar Beets data set. While weighted random sampling improved the performance of the 10-class classifier, the results still showed a heavy bias. The 3-class classifier showed the best results, but still failed to achieve a per class accuracy comparable to other weed classification systems. The benchmark system developed by Partel et al. achieved an

Table 3: Metrics for 3-Class Classifier

Labels	Precision	Recall	F1 Score
Crop	0.52	0.52	0.52
Weed (Big)	0.56	0.59	0.57
Weed (Small)	0.50	0.48	0.49

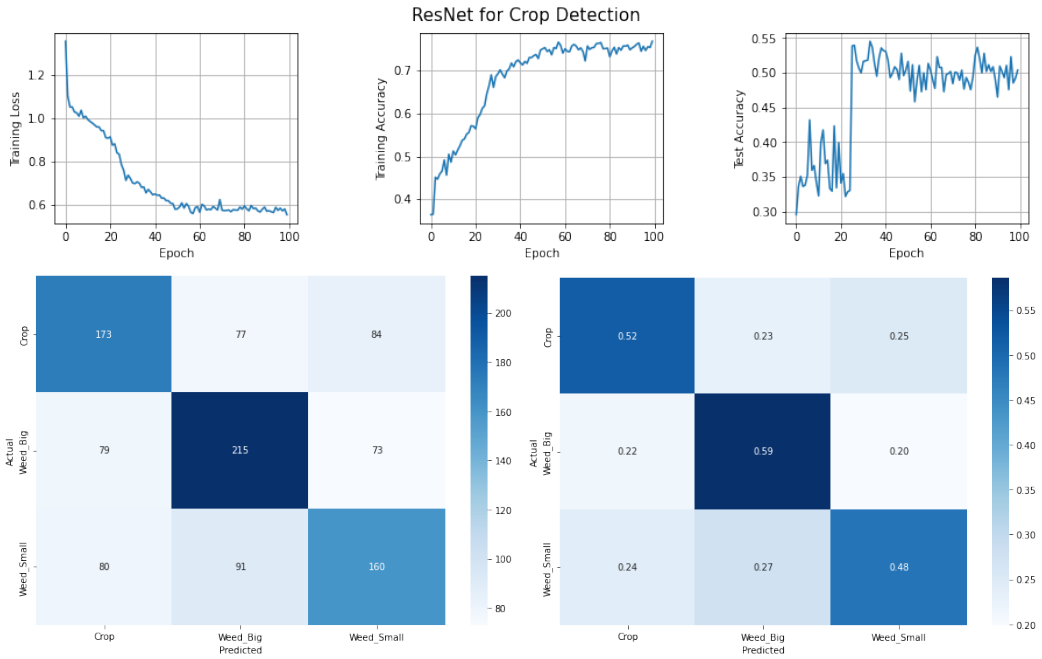

Figure 11: The 3-class classifier trained with weighted random sampling demonstrated the best performance. This particular instance took 48 minutes and 50 seconds to train.

average precision of 85% and a recall of 95% with Yolo3 [6]. For the 3-class case, the ResNet based classifier achieved an average precision of 53% and an average recall of 53%.

With more resources, several changes could be made to improve the performance of the classifier. First, the data processing pipeline could do a better job handling regions where the crops and weeds are growing next to each other, creating a two-class image. This occlusion could be confusing the classifier and could be improved with better masking when generating the new data set. Second, Data augmentation could be implemented to balance the data set. This could be achieved by either creating new synthetic data based upon the original samples or by duplicating existing samples with added noise and transforms (rotations, flips, skews, etc.). This could improve the performance of the classifier. Finally, only a small selection of network architectures and parameters were tested during this project. Additional tuning could potentially improve the classification results.

References

[1] Gary Bradski and Adrian Kaehler. Opencv. *Dr. Dobb's journal of software tools*, 3, 2000.

[2] Nived Chebrolu, Philipp Lottes, Alexander Schaefer, Wera Winterhalter, Wolfram Burgard, and Cyrill Stachniss. Agricultural robot dataset for plant classification, localization and mapping on sugar beet fields. *The International Journal of Robotics Research*, 2017.

[3] 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.

[4] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[5] Yuzhen Lu and Sierra Young. A survey of public datasets for computer vision tasks in precision agriculture. *Computers and Electronics in Agriculture*, 178:105760, 2020.

[6] Victor Partel, Sri Charan Kakarla, and Yiannis Ampatzidis. Development and evaluation of a low-cost and smart technology for precision weed management utilizing artificial intelligence. *Computers and electronics in agriculture*, 157:339–350, 2019.

[7] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[8] Blue River Technologies. Plant feature detection using captured images, Jan 2016.

[9] Mingchuan Zhou, Huanyu Jiang, Weinan Shi, and Alois Knoll. Design and optimization of the target spray platform. In *IEEE International Conference on Robotics and Automation-Workshop on Robotic Vision and Action in Agriculture*, 2018.