Synthetic Data driven Deep Learning for Plant Phenotyping

Zane K. J. Hartley

Supervised by Prof Andrew P. French & Dr Michael Pound

Thesis submitted to the University of Nottingham for the degree of Doctor of Philosophy

April 2024.

Abstract

The need for large quantities of high quality training data is one of the overarching problems facing the Computer Vision and Deep Learning research community. The need to seek versatile, scalable solutions to this problem is imperative as neural networks become involved with almost every aspect of the modern world. The topic of this thesis is training neural networks with Synthetic Data, one of the most promising solutions to the problem of data scarcity.

In this thesis I focus these attempts on plant phenotyping tasks, an important field of interest within Computer Vision concerned with the automatic measurement of the physical features of different plants. This thesis presents a number of Synthetic Datasets created with deep learning in mind, and then details a number of novel techniques for leveraging these datasets when working on phenotyping problems, focusing on domain adaptation, style transfer and network fine-tuning.

I present a heatmap guidance extension for style transfer, and a clustering approach to deep learning training to improve generalisation on diverse target datasets. Then my work on 3D reconstruction is presented, where domain adaptation is performed simultaneously with training a volumetric regression network, allowing for an unsupervised domain adaptation approach using an unlabeled train set. I present a series of experiments comparing Synthetic Data and fine-tuning approach between CNN and Transformer based architectures. Finally I look at Diffusion Models, a new form of generative neural network that promises to be the future of synthetic data generation.

Published Work

The following work from this thesis has been published:

Hartley, Zane K.J., Aaron S. Jackson, Michael Pound, and Andrew P. French. GANana: Unsupervised Domain Adaptation for Volumetric Regression of Fruit. Plant Phenomics 2021 (2021).

Hartley, Zane K.J., and Andrew P. French. Domain Adaptation of Synthetic Images for Wheat Head Detection. Plants 10, no. 12 (2021): 2633.

Acknowledgements

Firstly, I want to acknowledge the huge amount of support I have received from my supervisor Andy, since before my PhD even began. You have given me everything I have needed over the last 4 years from a supervisor, and I am incredibly grateful to have had such a good mentor for so long. Many thanks as well to Mike, your ideas and advice have been a great asset to my PhD, and I look forward to your continued mentorship in my academic career.

I want to give a big thank you to Aaron, for giving me a tremendous amount of support in the first half on my PhD, and for being a great friend to me since. Many thanks as well to my 후배 Janet, its been great having you as both a friend and neighbor in the office, and I appreciate having someone I can chat about both machine learning and K-pop with as needed.

I also want to give a massive thank you to both my parents, for giving me their unconditional love and support throughout my life, and especially so during my PhD. It is a tremendous privilege to be raised by parents who put me first in so many ways, and I appreciate you both so much for everything you've done to support me.

Finally I want to give a massive thank you to Georgia, for being the most supportive partner I could ever hope for during my PhD. I couldn't have done this without you, thank you so much.

Contents

Chapter 1

Introduction

1.1 General Introduction

This thesis presents a collection of research projects that focus on using Synthetic Data to solve the data shortage problem in machine learning. Focusing on the domain of plant phenotyping, *the acquisition of complex plant traits from images*, this body of work aims to investigate the impact of Synthetic Data on training deep learning models, and the importance of domain adaptation for tackling the domain gap problem this presents. Subsequently this work aims to investigate how emergent deep learning techniques are able to improve or leverage the use of synthetic data for training.

Tremendous progress has been made in the past decade in deep learning architectures, allowing for complex neural network models to be deployed in industry and real world fields across a variety of use cases. Today, industry application of deep learning to many fields relies on sufficient data being available to train state of the art models for the often niche applications for which AI can be effective. Plant phenotyping is one of many examples of this, as it has been shown to be a strong candidate for deep learning solutions while being especially difficult to collect data for (due to diverse crop types and importance of high image quality) and to annotate (due to small details carrying significance, and the expert knowledge needed for manual annotation). The work presented in this thesis looks at synthetic data as one solution to data scarcity in this area.

1.2 The Problem

Machine learning acquires its name from its underlying principle that a computer, given sufficient experience, can learn to solve a given problem. In practice, this means showing a computer a large volume of already-completed problems and comparing the machines prediction to the known solution, with the computer learning from the difference between the two. In Computer Vision, dataset sizes often start in the low thousands of images, going up to many million for the most challenging problems, and often containing extremely complex solutions.

Dataset size is extremely important to ensure good training of a neural network. Large models contain a large number of parameters that have random initial values that must be optimised during training. This large number of parameters allow the network to learn complex mappings between inputs and outputs, even when our input space is very large. However, for this very reason, large networks are often capable of *overfitting* to a small training dataset, a fail state where the network learns the solutions to its specific training inputs, rather than a generalized mapping from the input to output spaces.

As a result this creates a tremendous barrier for entry to anyone wanting to use machine learning, as almost any computer vision project must start with the collection on hundreds to thousands of images that must then be hand annotated. In complex cases such as panoptic segmentation (identifying the class and instance of every pixel within an image) this would mean drawing potentially hundreds of polygons around every single object in an entire image, an expensive process that is also prone to human error and bias.

Synthetic Data, created for the purpose of training deep neural networks is the focus of this PhD thesis, an approach that presents many challenges that this thesis seeks to address. Creating high quality synthetic data, with annotations is itself technically challenging, and a number of different possible solutions have been presented, using both conventional and machine learning technologies. As we explore in greater depth in this thesis, using synthetic data alone as training inputs for deep learning will yield poor results when tested on real images due to the domain shift problem. Synthetic images can have very different low level features from the real data it seeks to imitate, and as such a model fails to make predictions on real images that come from a different data distribution to that on which it was trained. Overcoming this hurdle is the main technical challenge addressed in this thesis.

1.3 Motivation

This research is motivated by a desire to help solve the data scarcity problem in Machine Learning, by helping to provide better ways of training neural networks via the use of different forms of synthetic data of Synthetic training data. By doing so we hope to make machine learning more affordable, efficient, and accessible, so that more people can use this powerful technology in the future. It is common in deep learning applications for a significant cost to be associated with the collection and annotation of data, not only financially but also in terms of time taken. The annotation of the ImageNet [26] dataset alone is estimated to have taken around 20 years collectively, assuming an image could be annotated every minute. The work carried out in this thesis was prompted by our desire to reduce these costs, as with the correct hardware we believe that image generation and annotation can be achieved in ten seconds or less, which would produce a five fold decrease in cost at the very least. In doing so make deep learning approaches to phenotyping more accessible by lowering this barrier for entry, allowing for a wider range of applications and uses.

The technical challenge in making synthetic data a viable solution to data scarcity comes from solving a set of problems that encompass the pipeline of using artificial data: data generation, domain adaptation, and training. The work in chapters 4 through 7 addresses these challenges by presenting multiple potential solutions to all three, and is motivated by the desire to make novel contributions and provide experimental data that can allow those in industry to make better synthetic data in future.

Our focus on Plant Phenotyping is based on our motivation to focus on an area with significant impact; as the use of deep learning in agriculture has significant implication for issues of global food security, as well as the impact of the climate crisis on agriculture in developing countries [63]. The main uses of automated plant phenotyping include crop monitoring (understanding the health of crops in a field to improve yields and detect stresses to plant growth such as disease), and analysis of new plants and pesticides, allowing for the development of more robust species for future use. In both cases, plant phenotyping is an important tool in agricultural settings, with a significant potential impact as well. The unique nature of plants provides many unique challenges not found in other domains, and research is needed to examine and solve domain specific challenges in a number of Computer Vision tasks. This thesis is motivated in part by the desire to explore and solve these domain specific aspects of synthetic data as it applies to phenotyping.

1.3.1 Research Questions

With these motivations in mind we set out a number of research questions we will seek to answer in the course of this thesis:

- What are the best methods for producing Synthetic Data? What are the pros and cons of these methods when compared against their alternatives? What specific considerations have to be made when producing synthetic data for plant phenotyping tasks?
- Do all types of Synthetic Data cause the domain shift problem? What are the best methods of overcoming or mitigating this problem to get better performance at downstream tasks?
- For what Computer Vision Problems can synthetic data be used? What are the advantages and limitations in each, considering factors such as cost, accessibility, and performance?

1.3.2 Objectives

Here we additionally outline our overall objectives for this thesis that will guide our work to addressing our stated research questions.

- The primary objective of this thesis is to develop pipelines for synthetic data generation and use using a range of state-of-the-art, and novel techniques.
- Secondly this thesis will contribute to the research discourse surrounding the use of synthetic data for deep learning training, focusing on core problems of interest such as domain adaptation and generalization.
- Our third overall objective is to expand the role of generative AI, for the purpose of generating synthetic training data. This includes better use of cutting edge technology, as well as a focus on more complex computer vision challenges.

1.4 Contributions

1. We develop a novel clustering approach to domain adaption of synthetic datasets when using a diverse target dataset. By using our method we demonstrate that object detection scores can be improved compared to conventional *source/target* style transfer.

2. We present a low-cost domain adaptation guidance mechanism using heat map regression to improve domain adaptation when focusing on object detection. Use of this technique will allow other researchers working with domain adaptation to make the most out of limited training data, reducing costs while improving performance.

3. This thesis presents an integrated domain adaptation, volumetric regression network that allows for unsupervised 3D reconstruction. This novel approach to 3D reconstruction is versatile enough to allow a deep learning model to be trained for any 3D target object(s), without the need to laboriously capture a large training dataset, as would otherwise be needed.

4. We design and run a series of experiments evaluating the effectiveness of

synthetic data and domain adaptation on Transformer architectures, comparing the impact of these methods against CNN based methods.

5. We present an investigation into the use of diffusion models for generating synthetic training data for deep learning neural networks, giving useful data to other researchers about how effectively diffusion generated images can generalize to real images when used as training data.

6. We develop a novel method of generating synthetic data for instance segmentation, including both images and annotation masks using conventional diffusion architecture. This approach stands to allow for other researchers to generate large amounts of instance segmentation data much more cheaply than with conventional hand annotation.

7. We present a collection of synthetic datasets for plant phenotyping problems that were used to train the networks for the experiments presented, in addition we include the pipelines used to create some of our handcrafted synthetic datasets as well as links to Github repositories where our code can be found.

1.5 Overview of thesis

This thesis is organised into the following chapters:

Chapter 2: A background of the key themes and areas of Computer Science this thesis focuses on.

A description of Synthetic Data in Machine Learning is presented giving details on the aspects of the field relevant to this thesis. Then Plant Phenotyping is fully introduced as an application area for computer vision, along with a taxonomy of Plant Phenotyping problems and brief overview of current techniques and research.

Chapter 3: A literature review of key ideas this thesis aims to build upon.

First this chapter looks at state-of-the-art and current developments in Computer

Vision, looking in detail at research surrounding the different problems we look at in this thesis. A broad overview CNN and Transformer based computer vision is then discussed, covering the current state of the computer vision field and the impact of new Transformer architectures. We then include a breakdown of Domain Adaptation especially where it concerns Synthetic to Real Style Transfer and other forms of Unsupervised Domain Adaptation. Finally we discuss other research into synthetic data, covering current idea on best practice in the creation of new datasets and their uses.

Chapter 4: Domain Adaptation for wheat head detection
In chapter 4, two novel methods for improving wheat head detection performance are evaluated. As well as presenting our created datasets, our clustering approach is shown to improve a networks performance when generalizing onto a diverse test dataset. This work makes up part of our 2021 paper titled *Domain Adaptation of Synthetic Images for Wheat Head Detection* and includes our synthetic datasets of wheat images.

Chapter 5: Unsupervised Domain Adaptation for Volumetric Regression of Fruit:
A novel method of 3D reconstruction is presented for the unsupervised domain adaptation problem. Focusing on 3D reconstruction of fruit as a phenotyping problem, we present a method of predicting high precision 3D models from a single 2D image. We present our dataset and the results of our experiments used for our 2021 paper *Unsupervised Domain Adaptation for Volumetric Regression of Fruit.*

Chapter 6: Use of Synthetic Data in CNN and Transformer based architectures.
Here a series of experiments is presented using Synthetic Data and domain adaptation for plant phenotyping tasks, where we present our finding using the current state of the art Transformer models and examine the relevance of Synthetic Data in the context of the ongoing competition between CNN and Transformer based model.

Chapter 7: Use of diffusion models for synthetic data generation.

Here we investigate diffusion models, and the ability to apply their impressive generative capability to the create of new data for training other neural networks. We present and evaluate a number of different methods for generating image-label pairs using current diffusion models.

Chapter 8: Finally a discussion of the work presented in this thesis with a summary of our findings and conclusions to be drawn from our research. We end with a number of suggestions for future work building on the ideas presented in the previous chapters.

1.5.1 Publications and Engagement

During the completion of this PhD, I have made a number of publications some of which overlap with the projects presented in later chapters. Additional engagement with the research community has been achieved through presenting a number of talks and attending conferences and workshops. A complete list of each of these is included below.

Publications

- Domain Adaptation of Synthetic Images for Wheat Head Detection.A paper containing the work shown in chapter 4, published in 2021.
- GANana: Unsupervised Domain Adaptation for Volumetric Regression of Fruit. A paper containing the work shown in chapter 5, published in 2021.
- Unlocking Comparative Plant Scoring with Siamese Neural Networks and Pairwise Pseudo Labelling. A paper completed working with industry sponsor *Syngenta* and published in the CVPPA workshop at ICCV.

Talks and Presentations

- Creating and Using Synthetic Data for Plant Phenotyping Problems. A presentation given to the *Syngenta* Computer Vision and Deep Learning Tech Meeting during 2023.
- Diffusion-based Synthesis of Training Data for Neural Networks. A talk given at the UK Plant Phenomics 2023 Town Hall and Conference in 2023

based on an accepted extended abstract of the same name.

- Unlocking Comparative Plant Scoring with Siamese Neural Networks and Pairwise Pseudo Labelling. Paper talk and poster session at CVPPA 2023 workshop at ICCV 2023 in Paris.

Chapter 2

Background

2.1 Introduction

This thesis draws upon a number of different core research areas, plant phenotyping, Synthetic Data, Domain adaptation, and more broadly, Machine Learning and Computer Vision. In this chapter we provide some further context on each of these areas.

This chapter contains 4 sections: Section 2.2 gives context on deep learning approaches, as well as the current state of the art, focusing on the core problems we focus on in this thesis. A comparison and discussion of Convolutional Neural Networks and Transformer architectures is also included. In section 2.3 a discussion of Synthetic data in deep learning is included, looking at its use in a variety of applications. Then in section 2.4 the domain gap problem is considered, and the use of domain adaptation in literature is reviewed. Finally, section 2.5 gives an overview of plant phenotyping. The wide variety of different problems computer vision is often applied to are described here, and the importance of deep learning research and Computer Vision for plant science and agriculture.

2.2 Deep Learning

In the past decade deep learning has emerged to become the dominant machine learning system at the heart of a wide range of different computer science fields across industry and research. This has been especially true in Computer Vision where Convolutional Neural Networks have completely eclipsed traditional methods, creating a new paradigm of solutions to image processing problems.

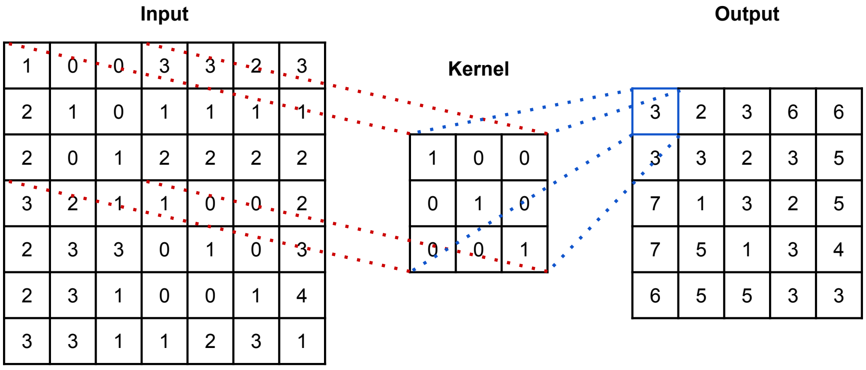
Deep learning evolved from multi-layer perceptron models, an early form of neural network [39]. MLPs are large networks of neurons connected by weights in a series of layers, each triggered by an activation function. By feeding data into the first layer as an input, and designing the final layer to produce a meaningful output, MLPs are able to perform a number of tasks, from regressing a value to classifying the input into one of a number of categories.

Modern deep learning models are generally large neural networks containing millions of learnable parameters across many hidden internal layers. When we train a model, we aim to optimise it for a task using examples with known solutions, a training set of labeled data. For any sample of our training data we can use backpropagation [96] and gradient descent to adjust the parameters of the neural network to produce an output as close as possible to the known solution. To do this we perform a forward pass, passing our input through the network. The output of this forward pass is then compared against the true results, known as the ground truth, using a loss function which calculates the degree to which the output is correct or incorrect.

By iterating this process a large number of times we can train neural networks that are able to accurately make predictions about previously unseen examples. This fundamental challenge of machine learning is called generalization, the ability of a model to perform well on unseen data. It is common for a network trained for too long, or on train dataset that is too small to become extremely good at making predictions for the training data, while being extremely poor at generalizing. We refer to this problem as overfitting, and a network that fails in this way can be seen as not learning to solve the desired problem, but instead simply learning the solutions to one small subset of inputs.

2.2.1 Convolutional Neural Networks

In Computer Vision we face additional problems when applying a machine learning approach to images. Unlike other forms of data, which may have tens or hundreds of inputs, a small 256x256 image already has over fifty thousand individual pixels, each often containing 3 or more channels. Even in 2023, with the most powerful hardware, it is impossible for a neural network to compute the relationship between so many inputs and make sense of them. Indeed, even if this were possible it would be incredibly wasteful (many of the relationships between individual pixels would have no significant meaning), and make scaling to high resolution images incredibly challenging.



1	0	0	3	3	2	3
2	1	0	1	1	1	1
2	0	1	2	2	2	2
3	2	1	1	0	0	2
2	3	3	0	1	0	3
2	3	1	0	0	1	4
3	3	1	1	2	3	1

1	0	0
0	1	0
0	0	1

3	2	3	6	6
3	3	2	3	5
7	1	3	2	5
7	5	1	3	4
6	5	5	3	3

Figure 2.1: Convolutions act as a sliding window passing across a grid of pixels. For each position of the kernel or filter, we are able to calculate an output for a cell in our output. By constructing or learning specific values for the kernel, we can identify different features of an image that is then captured in the output.

To make images compatible with neural networks, a tool from conventional image processing was borrowed [61]. This was the *convolution matrix* or *kernel*, a small matrix that passes over the image as a sliding window shown in figure 2.1. Depending on its values a kernel can perform a number of operation on the image, identifying useful features and outputting a transformed version as output. We can intuit that pixels that are close together spatially are likely to be related to one another, allowing convolutions with small receptive fields to extract information from small regions of images while shrinking the spatial resolution. By making the handful of parameters of a kernel (size, stride, padding, etc) learnable parameters of a neural network, we allow the model to identify features of interest during training, as well as reducing the computational cost of running the model.

CNNs stack learned convolutional layers together, with pooling layers, reducing the spatial resolution further and activation layers, which discard unneeded information, in order to reduce the spacial resolution to a usable size. Afterwards networks are designed with their desired output in mind, often with a number of fully connected layers. This may conclude with a number of outputs corresponding to confidence values of each class, or a more complex nature of bounding boxes and class labels.

2.2.2 The Importance of Data

Outside of network design, two of the fundamental factors in the effectiveness of deep learning have been computational power, and availability of good data. Recent increases in computational power have made large neural networks viable, however the need for large training sets has increased with the growing scope of applications for CNNs. Effective datasets require not just sufficient size, but also quality and diversity, among a range of other factors. In the late 2000s, a number of large, high quality datasets appropriate for training neural networks became available [27], enabling much greater research into complex computer vision neural network design.

Many of these datasets however, are either general purpose datasets like Imagenet, or datasets for a narrow set of applications, such as face recognition or street view images for autonomous vehicles. While this allows CNNs to demonstrate remarkable performance in research papers, applying them across diverse and highly specialised fields becomes difficult as for each problem new training data will need to be collected and manually labelled.

2.3 Synthetic Data

Artificial means of increasing dataset size are common in Computer Vision. Data augmentation refers to modifying a training set to inflate its size by transforming images in a number of ways to increase variety. Data Synthesis expands this idea by generating entirely new images that have the same or similar distribution as our intended target domain.

Generating data this way has a number of advantages:

- **Dataset size.** Datasets collected manually have a finite size, and are constrained by cost, time and other limitations. Once a dataset is collected it is often very difficult to collect additional data from the same distribution, as elements such as lighting, camera and other environmental factors may have changed outside the control of the user. Synthetic datasets generated automatically can be scaled extremely easily, and while there is a theoretical limit to the amount of variety that can be generated, this is often orders of magnitude higher than what we can expect from real datasets. A dataset generated this way can have a theoretically limitless size. By artificially increasing the size of our training datasets we can help to reduce the overfitting problem that is common in deep learning problems.
- **Automatic Labelling.** For computer vision problems manually labelling images can be extremely expensive and time consuming. This especially true where labelling requires expert knowledge, for example in biological fields, such as medical imaging or plant phenotyping. Additionally more complex computer vision tasks such as panoptic segmentation can take substantially more time to manually annotate than classification or detection problems, but are usually about as easy to annotate automatically, and have almost no additional cost.
- **Error Reduction.** By removing the humans from the image labelling process, we avoid the issue of human errors. Even for simple problems we can expect humans tasked with labelling large number of images to make mistakes. For problems such as segmentation, detection and counting, it is also possible for there to be ambiguity in an image that can make human labelling unreliable.
- **Reuse of assets.** Digital assets have a lot of reusability, and after incurring a one time cost to develop a dataset, generation of new images can be done at no extra cost. This often makes generating synthetic data cheaper than collecting real data from scratch. For 3D rendering, video game assets can often be used to generate training data due to its inherent realism and high fidelity as seen in the GTA5 dataset [90] being used for self driving

cars. Similarly for synthetic data generated either by GANs or compositing, images from a number of sources can be repurposed to create new data for training.

2.3.1 Challenges

Using synthetic data in training has a number of challenges that must be overcome for them to be effective. A highly realistic dataset of synthetic images can not simply be applied to a neural network if we expect the network to generalize well onto unseen real images. This is known as domain shift, a problem caused by two datasets being from different data distributions and preventing generalization between the two. In more general cases domain shift could be a significant difference in image content, such as different modalities of medical scans of the same organ (such as MRI and CT scans) [55], or even more overtly different subject matter such as images of cats vs dogs. In many cases the domain gap between two sets of images could be of no relevance, or even imperceivable to a human, such as two sets of images collected with different cameras, or with different lighting conditions, yet this is enough to completely fool a neural network.

With this in mind it is understandable that synthetic data can at best be seen as imperfect data, however synthetic data has other challenges that must be overcome to make it effective training data for neural networks. For any given problem, a process for image synthesis must be developed, we demonstrate a few such methods in chapters 4, 5 and 7, however in all cases that process must be designed to not only maximise image quality but also variety. Just as it is common for real datasets to contain data bias, it is also likely that an image generation pipeline will have biases built into it, and minimising these must be a key consideration during generation. In addition, it is likely that real datasets will contain anomalous results that cannot be predicted or modelled synthetically. Outliers such as these will need to be identified or mitigated at some point in the pipeline else will have a detrimental effect on downstream performance.

It is also important to consider the computational and human cost of generating Synthetic Data. In many cases this is considered to be of minimal concern, but since doing so often requires to collaboration of domain experts, 3D artists or experts in generative models, there is often a much larger up front cost to develop a new dataset that is only recouped by the ability to generate extremely large numbers of new samples. Generating thousands of images is almost always faster and cheaper than manual collection and annotation in a research or large scale industry setting, but the reliance on expensive technology and expert domain knowledge presents challenges in making synthetic data widely available.

Finally, while synthetic datasets have proven their effectiveness in academic literature and in industry, there still exists very little research into benchmarking different synthetic datasets. Until effective ways of testing the effectiveness of synthetic datasets are developed it remains challenging to fully evaluate any individual dataset or approach to image synthesis.

2.4 Domain Adaptation

Domain adaptation is a field of machine learning that has developed from transfer learning. Transfer learning is the commonly used approach of retraining networks on new training data in order to improve performance either on different test data or when constrained by limited data or computational power. The goal of transfer learning is to leverage the knowledge the network gained on the original problem for a new problem. Domain adaptation extends this idea, aiming to train a network on a source domain that is capable of performing well on another target domain from a different, but related distribution.

2.4.1 Definition

Formally for any machine learning problem we have a distribution $\mathcal{D}$ of label-value pairs from input and output spaces $\mathcal{X}$ and $\mathcal{Y}$ respectively. In supervised machine learning our objective it to learn a mapping from a point in $\mathcal{X}$ to a point in $\mathcal{Y}$ using a sample of known pairs in our training set taken from $\mathcal{D}$ that can perform well on unseen samples that are also from $\mathcal{D}$.

In domain adaptation we consider two different but related distributions $\mathcal{D}_\mathcal{S}$ and $\mathcal{D}_\mathcal{T}$. Here our goal is to learn a mapping from points in $\mathcal{X}$ to points in $\mathcal{Y}$ using samples of $\mathcal{D}_{\mathcal{S}}$ so as to minimize error when predicting outputs for samples from $\mathcal{D}_{\mathcal{T}}$.

Generally speaking the target domain is assumed to be related to the source domain, while often being significantly different in one or more ways. In research different datasets of the same subject matter are used to show the effectiveness of domain adaptation. A common example of this is written character datasets such as MNIST, USPS, and SVHN, which can be used to verify a models ability to generalise across domains. Due to domain shift, a neural network trained on just one of these datasets would perform unfavourably when tested on the others compared to unseen examples from the train sets domain. Effective use of domain adaptation suggests that there are transferable features between the two domains that the network can learn however, which we hope to leverage when training models in this way.

Domain adaptation methods generally assume that the shift needed to align the two domains will primarily happen in the latent space of a deep neural network. Pixel space features including basic shapes such as edges and corners that are captured by the first few layers of a network will generally be consistent across domains. When performing domain adaptation it is common to freeze these early layers or reduce the learning rate of the network so the latent shift can be learned by the network without losing the knowledge learned from the source domain.

2.4.2 Taxonomy

Domain adaptation problems are varied, and make up a broad taxonomy of related problems. We often build a taxonomy based on presence of labels for our target dataset.

- **Supervised Domain Adaptation.** Cases where we have labels for both our source and target data. This case is the most similar to Transfer Learning.
- **Semi-Supervised Domain Adaptation.** Cases where we have labels for

all images in our source dataset, and a partially labelled target dataset. In some cases this could be as few as a single labelled image for our target, however the total number of images in our target dataset could be very large.

- **Unsupervised Domain Adaptation.** Cases where we only have labels for our source dataset and our target dataset is completely unlabelled [38]. This is often considered the most useful form of domain adaptation as, if successful, it will best enable us to leverage the huge quantity of unlabelled images that already exist, and makes it cheaper and easier to create new dataset for machine learning.

In addition to this we can also consider cases of homogeneous vs heterogeneous domain adaptation, considering cases where the two domains do or do not share their feature space, i.e. $\mathcal{X}_{\mathcal{S}} \neq \mathcal{X}_{\mathcal{T}}$ or $\mathcal{Y}_{\mathcal{S}} \neq \mathcal{Y}_{\mathcal{T}}$. Additionally we also consider the cases of domain adaptation where we use intermediary domains between our target and source, called Multi-step Domain Adaptation, versus the more traditional one-step approach we see in most literature.

2.5 Plant Phenotyping in Computer Vision

In Computer Vision, plant phenotyping is described as the area concerned with the acquisition of complex plant traits via observations, usually performed by hand by a domain expert. In a computer vision context, *digital plant phenotyping* specifically refers to the automatic acquisition of traits from images and is the primary understanding of the term *plant phenotyping* used in this thesis. Automating the process of measuring phenotypic traits in plants has many important applications and is important in closing the *phenotype-genotype gap* whereby the ability to record measurable characteristics in plants has lagged behind the mapping and profiling of a plant's genome. Many of the use-cases for phenotyping technology comes from the development of new plant breeds for agriculture, as well as the automation of crop monitoring. While domain experts are often still better able to analyse phenotypic traits than computer vision techniques, automating this process allows phenotyping to be scaled up to many millions of images enabling the processing of many more plants.

2.5.1 Motivations

Advances in digital plant phenotyping technology will help improve sustainability in global agriculture in coming years. Tools for plant phenotyping will play an important role in coming decades at meeting the food needs of the earths growing population; expected to peak at nearly 11 billion people by the end of the century. This increased pressure comes at the same time as rising global temperatures, and the climate crisis leading to pressures on farmers as yields are often put at risk by the changing environmental conditions.

Precision agriculture and plant breeding, enabled by phenotyping will be important at mitigating the effects of climate change on global agriculture, and the impact it will play on crop yields. Even small changes in temperature can have severe effects on farmers by reducing their ability to grow crops while extreme weather conditions like draughts become more common. These kinds of measurements can aid in crop monitoring by detecting *stresses* to plants, measuring growth and even predicting future problems before they present themselves fully. Beyond this, digital phenotyping also can be used in the process of plant breeding through the discovery of robust genotypes as well as the development of argochemical products and related technology.

2.5.2 Review of Plant Phenotyping Problems

There have been a number of taxonomies of plant phenotyping problems presented, most notably those from Ubbens et al [109] and Choudhury et al [23], as well as comprehensive reviews of image analysis of plants [73]. Choudhury's distinction between *Structural*, *Physiological* and, *Temporal* phenotyping problems are the most relevant to current research, where a majority of research activity focuses on structural phenotyping, including the work done in this thesis.

Structural Phenotyping

Measuring many of the most fundamental features of a plant, including holistic features such as height and width, individual plant components such as leaves [29] and organs, as well as non-geometric features such as weight. Many of the problems in this area of phenotyping are directly analogous to common Computer Vision problems such as detection and segmentation. Additional examples of structural phenotyping include, calculating plant biomass, leaf volume and curvature prediction, and segmentation of individual plant components.

A lot of research in the area of structural phenotyping [23], including our own presented in chapter 5, are concerned with 3D phenotyping. Due to the difficulty of dealing with the complex nature of plants, attempts to capture 3D information about their structure from one or multiple angles can often make predicting different structural traits easier. Similarly plant phenotyping often is done using RGB-LiDAR (RGBL) cameras, where LiDAR is used to capture depth information in addition to color channels which can make 3D reconstruction easier.

Physiological Phenotyping

Looking for features generally related to the health of the plant. Plants can be affected by any number of stresses (negative external circumstances) which present in different ways that normally take expert knowledge to detect and identify. In many ways this kind of phenotyping could be the most important for ensuring food security, as being able to protect crops from stresses caused by the climate crisis is essential. Problems in physiological phenotyping can often take the form of detection [37], classification and regression though they are less generalised problems than is found in structural phenotyping.

Being able to use Computer Vision to monitor physiological phenotype of different plants helps agricultural scientists better understand different plants resistance to droughts, heatwaves and other abiotic stresses that threaten crop yields in a changing world. In addition to this is other forms of plant stresses caused by plant diseases, insects and other biological hazards are also important to be able to identify and detect using computer vision. Quick response to plant disease can often prevent the loss of huge yields, and automated detection is an essential technology to ensure food security.

Temporal Phenotyping

Temporal phenotyping generally looks at time phenotypic plant traits over time, an extremely important part of agricultural research where temporal effects are always of interest to biologists. Despite this importance, temporal phenotyping is perhaps the least common form of phenotyping in the context of computer vision research output. The most common areas of study relate to plant growth rates [3], and the ability to make predictions based on currently observable phenotyping information. Often this work is combined with other forms of phenotyping, taking a number of measures of the same plant over time and then combining the structural or physiological phenotyping discussed above with the time scale information to gain insight into the temporal variation of specific phenotypes [22].

Some of the temporal elements that are of interest to researchers relate to discrete events such as time to germinate, time between new leaves and time to flower. Other elements of growth to be measures are more continuous and relate to growth of individual components, such as leaf length or plant height as seen in structural tasks, while others might relate to the progression of physiological traits, such as stress propagation over time.

Unfortunately, of the three types of phenotyping problem there is by far the least available computer vision datasets for temporal problems, making this a much harder area of study to work in than the other areas. Furthermore as problems in this area are less analogous to generic computer vision problems, temporal phenotyping is considered a much more niche area of research leading to a smaller amount of research interest in temporal problems.

Chapter 3

A Review of the Application of Synthetic Data and Deep Learning to Plant Phenotyping Problems

In this section we provide a comprehensive literature review of all state of the art research that relates to our own work presented in subsequent chapters. Following on from section 2.2 we cover the most recent advances in Computer Vision problems and new deep learning technologies. We then continue this with an overview of other literature looking at solutions using Synthetic data, with some focus given to other research looking at Plant Phenotyping. Each experimental chapter of this thesis also includes an additional analysis of related work, as seen in sections 4.3, 5.4, 6.2, and 7.5

3.1 Deep Learning and Computer Vision

Computer Vision is the broad field of AI related to learning to analyse visual information such as pictures and videos. Computer Vision lends itself to a wide range of domains and applications, and as we discuss below is an area of substantial interest to the research community.

3.1.1 Object Detection in Computer Vision

Object detection is one of the most common computer vision problems. Detection is commonly defined as combining localisation, where bounding boxes must be drawn around instances of objects within an image, with classification where classes are assigned to one or more objects within an image. Neural networks designed for object detection are therefore tasked with predicting and classifying bounding boxes for all objects in a scene from a predefined set of classes. This problem presents a number of challenges for computer scientists, such as dealing with very large number of objects or separating different instances of objects that occlude each other. For machine learning models an additional problem is the variable length outputs, as the number of bounding boxes will vary by input. Different approaches to solving these problems have been suggested often assuming an upper bound for the number of objects in an image and detecting objects up to that limit.

Region proposal algorithms are found in the R-CNN family of neural networks. Originally regions were extracted using selective search algorithm [42], using feature-based approaches to predict boxes that may be of interest. Any regions detected this way can then be used as inputs to a classification CNN, The network then only has to classify those containing the desired object and make minor adjustments where necessary. This can be improved upon by using a feature map extractor prior to region proposal to predict regions in the image that are thought to potentially contain an object [41]. This method allows for efficiency gains by proposing regions in a much smaller feature space, rather than passing full resolution regions into a CNN. The current state of the art models are completely CNN based, using a *region proposal network* to predict bounding boxes more quickly[88]. Using an extension of this approach instance segmentation can also be performed, as we only need to find the pixels related to the object inside each box to perform instance segmentation across the entire image [44].

The other most commonly used CNN approach to object detection is the YOLO

('You only look once') approach, originally crafted as a means to achieve real time object detection, and currently on its 7th version [113] (although the continuity between later versions is indirect). As the name suggests, YOLO makes only one pass to detect all objects in the entire image, making it extremely efficient in addition to it demonstrating state-of-the-art accuracy. YOLO treats the image as a grid of cells, and makes a number of predictions about all boxes simultaneously, with each cell considering objects whose center is contained within it. In two concurrent branches, YOLO predicts a series of likely objects from each grid cell with associated confidences, while also predicting a most likely class for each tile. YOLO based models use a system of *anchors* to predict bounding boxes. In object detection anchors are a set of predefined bounding box dimensions based on the known information about the target class (for example that humans are generally much taller than they are wide), and this allows the network to only have to predict offsets for these predetermined anchors. In our experiments we use the popular YOLOv5 Pytorch implementation [33] which was the most popular model in 2022 at the time our experiments were completed.

3.1.2 Transformers for Computer Vision

First proposed in 2017 by Vaswani et al [112], the Transformer is a neural network architecture related to Sequence-to-Sequence models that has come to be one of the most popular paradigms in machine learning research. The underlying design principle of a transformer model is the attention mechanism, whereby the model is able to give greater emphasis to the most relevant parts of the input when computing the output. While a full discussion of Transformer architecture is beyond the scope of this thesis, we will briefly discuss the general model architecture as well as its intuition.

Sequence-to-sequence models have been a popular basis for Language models for some time, with both Recurrent Neural Networks (RNNs) [95] and later, Long short-term memory (LSTM) [47] models having existed for many years, treating an input string as a chain of inputs, and sharing information between successive elements. Recurrent Neural Networks are an encoder-decoder model used for sequence-to-sequence translation, first encoding each element of the input in turn and then decoding a prediction of each element of output. This approach to sequence processing is effective, but has two major downsides. Firstly the vanishing or exploding gradient problem means that long sequences become impossible to process, as gradients may tend to zero or infinity through sufficiently long inputs, though this problem is somewhat mitigated by LSTM blocks. Secondly, because of the heavily serialised design of the network, they are inefficient to train on modern hardware that relies of parallelisation for efficiency.

The Transformer model presented by Vaswani et al [112] in 2017, resolves these issues and produces substantial improvements in performance and efficiency. Transformer models work by treating each word as a vectorised token, in some predetermined embedding space, and then combining these vectors with a further positional embedding based on its position in the sequence. These tokens are then passed through a self-attention layer, which calculates the relationship between every pair of element elements, intuitively this calculates how relevant is each individual token to every other token. This is followed with a fully connected layer, which combined with the self attention layer makes up each attention block. Multiple attention blocks after often stacked to make up the encoder, while the decoder is similarly a stack of blocks containing attention and fully connected layers. Decoder blocks then also contain a masked attention block, that hides part of the label during training, enabling the network to solve the sequential problem in parallel, which results in improved computational efficiency.

The idea of global attention-style mechanisms for Computer Vision were first introduced in 2018 by Wang et al [115], followed in 2020 when Dosovitskiy et al first presented their visual transformer (ViT) [31] adapting the transformer architecture for image processing and replacing word tokens with image patches, demonstrating state of the art performance in image classification. ViT is presented as an alternative to CNN models and discards many of their inductive biases, most importantly replacing the CNNs focus on locality with the transformers *global attention*. It is worth noting however that ViT achieved its impressive results using a substantially more expensive training regime than its CNN based competition, having been trained on a private dataset of 300 million images, rather than the more conventional ImageNet dataset that contains only one million images.

Speculation that Transformer architectures could replace CNNs to become the dominant design pattern in Computer Vision has been a common part of Computer Vision literature in recent years. The distinction between the two has however become increasingly blurred as some models implement convolutions within Vision Transformers, while other Transformer models such as the Swin Transformer [67] take inspiration from CNNs by including hierarchical feature maps and shifting patch windows between layers. Most commonly convolutions are used as part of the embedding process, and then convolutions have also been incorporated in the Attention blocks themselves [119]. Since 2020 we have seen a large number of different Transformer models presented in academic literature, expanding to more common computer vision tasks including detection and segmentation, as well as to other areas of machine learning research.

DETR [17] is one such transformer model of particular interest as it was the first significant attempt to apply transformers to object detection problems. Unlike Yolo and RCNN-based models, DETR approaches detection as a direct *set prediction* problem, having a fixed size set of predictions, meaning it does not need to use non-maximal suppression to remove numerous overlapping predictions. Instead, unwanted predictions found in the image are given null labels, with real predictions given a class label instead. In additon, DETR is an example of model that uses a ResNet CNN as a layer preceding tokenisation, encoding the image into a feature space rather than giving the transformer raw pixel data.

3.1.3 Generative Computer Vision Models

Generative Adversarial Networks

Using deep learning models to generate new information has become an area of great interest in recent years, with its first major peak in 2014 with the introduction of Generative Adversarial Networks, or GANs. The first GAN was produced in 2014 by Goodfellow et al [43], and has subsequently been iterated on many times.

The GAN framework for image generation conceptualises the task as training a model to learn a distribution, from which it can then generate new samples within that distribution from an input random noise. This generative model is then paired with a discriminator, a type of classifier, which learns to identify between real images from that distribution, and images generated by the rival generator. As the discriminator learns to distinguish between real and fake images, it forces the generator to produce increasingly realistic images in order to fool the discriminator. When trained together the generator is eventually able to produce images that are able to fool not only the discriminator but also humans, creating images that are photo realistic, or otherwise match the distribution of the training set.

Generative models have improved a great deal over the years. Goodfellow's model was quickly iterated upon with cGAN by Mehdi Mirza et al. the same year [74]; cGAN or conditional GAN introduces the idea of an input token that gives control over the input, a common feature of many modern GANs. A further significant contributions was DCGAN by Radford et al [86] that significantly improved the ability of GANs to produce high resolution, photo-realistic images, and became the basis for many future GAN papers. Then in 2017 Martin Arjovsky et al introduced their Wasserstein GAN [6], which uses a Wasserstein distance function to calculate the distance between the generated and target distribution, allowing for better training and more high quality samples.

Adversarial models have also become a core feature in style-transfer models, where an image from one domain is transformed to match another domain, often called the *source* and *target* domain respectively. Target domains are normally represented by another image set or singular image that contains the desired characteristics. These kinds of models are of particular interest to this thesis and form the basis of some of our work in domain adaptation.

Image-to-Image GANs

Important to our work is the idea of *Image-to-image* GANs, models concerned with modifying or changing an Image rather than generating new samples from noise. Many such models are designed to be used for *Style Transfer*, a computer vision problem concerned with modifying the appearance of images. Originally emerging from the earlier *Texture Transfer* problem, seen in earlier papers such as Efros at al [32], Style Transfer considers each image to be made up of *content*, and *style* and the goal of a style transfer model is to learn a transformation that changes the style of an image while preserving the content. A popular example of Style Transfer is taking photographs and transforming them to match the style of popular artists, a more complex transformation than texture and only made realistically possible with modern deep learning approaches.

Given a dataset of paired images of the same content in different styles this problem is conceptually straight-forward and could be easily framed as a supervised Computer vision problem, however datasets of paired images are unlikely to exist for many styles increasing the challenge. The earliest uses of CNNs for style transfer includes work by Gatys et al [40], who attempt to separate content and style reconstructions, which can then be recombined in different combinations to generate new images. Also in 2016 Johnson et al [56] improve this by using separate perceptual loss functions for style and content to enforce high quality style transfer.

In 2017 many GAN based approaches to style transfer emerged, many of which focused on solving problems that were not possible with standard CNN approaches. Introduced by Zhu et al [123], CycleGAN is perhaps one of the most popular *Image-to-image* generative frameworks that still sees use today. Intuitively, CycleGAN is based on the idea that by forcing changes made by the generator to be reversible, we ensure that content is preserved even when style is changed. Using two pairs of Generators and Discriminators, CycleGAN learns both a forward and backwards transformation between the two styles, meaning that an image can be restored to its original style. The same year Pix2Pix by Isola et al [50] presented a wide range of image-to-image problems, including style transfer but also looking at a range of other image-to-image tasks such as inpainting, colourisation and harmonization (making elements of a composite image consistent with the rest of the image). Both CycleGAN and Pix2Pix helped popularise image-to-image GANs in the years after their development, and CycleGAN in particular can used to improve the realism of synthetic data, and is used in chapters 4 and 5.

Diffusion Models

First introduced in 2015, diffusion models are another form of generative model that allow us to create synthetic images using a machine learning model. The idea behind this model is to create a forward process that iteratively destroys information through diffusion and then train a model to undo the steps in that process and restore information. First demonstrated in Sohl-Dickstein et al [101] in 2015, generating images based on the CIFAR-10 dataset, diffusion models potential was not fully recognised at the time, and was overshadowed by the much larger interest in GANs. This approach to image generation was revisited in 2020 in [46], where the diffusion model was more fully formalised and high quality images were first produced giving diffusion much more attention.

Diffusion models can be summarised as follows. Starting from an image from our training set we define a forward diffusion process as a Markov chain, where in each step we add Gaussian noise to our sample. Each time step in the chain is an increasingly noisy image, with the final step being an image that has been completely destroyed leaving only Gaussian noise remaining. A deep learning model is then used to learn a backwards process, where it removes noise from the image in steps. For each reverse step of the chain, given an image at time step *t*, the model predicts the Gaussian noise added during the forward process at time step *t-1.* By subtracting the predicted noise from the image at *t*, we can get an estimate for the image at *t-1.* In practice we can train a model by giving it a training image with all the noise added up to step *t* and predicting either the total noise or the original image. We can then sample new images from diffusion models by taking pure Gaussian noise and inputting it into our model and predicting the image for *t=0* where all the noise in removed and a complete image remains.

During 2021, a number of papers released demonstrating different setups for diffusion models with [28] improving the fidelity of generated images at higher resolution and GLIDE [78] introducing text to image generation to the diffusion process, which has now become the hallmark of diffusion-based image generation. GLIDE uses two methods to guide the model using text captions, first *Contrastive Language-Image Pretraining* guidance, introduced by Radford et al [85], uses a pair of encoders for images and captions, using a contrastive cross-entropy loss to encourage high dot products for matching pairs, and low products for mismatched pairs. In order to function, the CLIP model was trained on images with added noise to better function in the reverse process. Secondly, GLIDE implements *Classifier-free Guidance* in which the model receives an image paired with both a caption and with an empty or null input sequence. Then the goal is to maximise the difference between the two during training, which causes the strength of the caption to be magnified, leveraging the models own knowledge of the caption.

More recently in 2022, as well as achieving mainstream attention, diffusion models have seen a large number of major new contributions [87] [98]. Stable Diffusion [91] tackled the problem of the high computational cost of training diffusion models, by shifting the diffusion process described above to take place in a lower dimensional latent space. Their *Latent Diffusion Model* or LDM separates the learning process from the pixel space, which allows the model to run with a smaller number of parameters, improving efficiency while still boasting state-of-the-art performance. 2022 also saw the publication of Cold Diffusion by Bansal et al [9], which also made a number of significant contributions. The paper showed that similar performance could be made with other mechanisms of degradation being used in place of Gaussian noise, including a number of deterministic transforms, using different transforms can allow for the model to learn a wider range of patterns and better model the underlying distribution of the training data.

Overall despite its recent emergence to the forefront of AI research, it is likely that diffusion models will remain a significant feature of generative computer vision work for some years, and will be highly relevant to the focus of this thesis. Our own work using diffusion to generate training data can be found in chapter

7.

3.2 Domain Adaptation

As discussed in section 2.4, domain adaptation is an area of research concerned with allowing neural networks to perform well on different domains to the one they were originally, or primarily, trained on. As Domain Adaptation is a large area of research, we limit the scope of this literature review to the use of domain adaptation for images and Computer Vision problems.

Since Domain Adaptation seeks to align the two domains, for Image based problems we can take two general approaches. Firstly, if we assume a model has some kind of encoder-decoder architecture, we can train an encoder to align its representation of domain invariant features, allowing the decoder to perform downstream tasks on this invariant representation. Alternatively we can align images in the *Pixel Space*, modifying the source image to match the appearance of the target domain through style transfer. Both Liu et al [66] and de Melo et al [25] present good evaluations of the wide range of different approaches to domain adaptation popular in current literature.

3.2.1 Latent Space Alignment

Domain Divergence Measures. Different distance functions can be used to measure the discrepancy between different latent spaces. By using a shared feature extractor on both domains, we can then use a distance function such as maximum mean discrepancy to learn the network domain invariant features such as the location of key points within the image. One such example of this approach is Long et Al [68], who apply MMD to their Joint Adaptation Network, to enforce similarity between the latent distribution of their source and target domains. Simlarly Sun and Saenko introduce their CORAL loss [104] in their 2016 paper Deep CORAL, which extends previous work [103] on domain adaptation to a deep learning space.

Adversarial Feature Alignment. Adversarial training can also be used to achieve latent space alignment. In 2015, Ganin et al [38] presented their model DANN highlighting that domain adaptation relies on a model learning domain invariant features, and enforcing this using an additional domain classifier that is trained adversarially, enforcing a similar feature distribution across domains. In 2019 Hsu et al [48] also present an adversarial approach, this time using an intermediate domain between source and target (in this case using a synthetic dataset to adapt between the KITTI roads and Cityscapes datasets), allowing for a progressive adaptation.

3.2.2 Style Transfer

Domain style transfer is a method of domain adaptation that often applies when two distributions of images are spatially similar; common examples include different medical imaging modalities different datasets of the same content. Using this method we take our two input spaces $\mathcal{X}_\mathcal{S}$ and X_T and attempt to use a style transfer technique to transform $\mathcal{X}_\mathcal{S}$ into a new distribution $\mathcal{X'}_\mathcal{S}$ which is considered to be closer to the target $\mathcal{X}_\mathcal{T}$. We are then able to train our task network on this transformed version of the source dataset.

CycleGAN is perhaps the most well known of the style transfer GANs presented in recent years. First presented by Zhu et al [123], their model enabling high quality unpaired image-to-image style transfer has been the starting point for an extremely large body of research, including some of the work presented in this thesis. Though CycleGAN was not originally presented as a model for domain adaptation, we can see examples of it being adapted to be used for this purpose such as by Apple in their work on eye tracking [100], using CycleGAN to convert Synthetic Images to the Real domain. Similarly Mueller et al also uses an extended CycleGAN for their model GANHands [75], which adds an additional segmentation loss, to the CycleGAN model to enforce pixel perfect style transfer, to create realistic training data from synthetic hands. In plant phenotyping a similar approach was presented by Barth et al [10], who also used CycleGAN for image segmentation using a Synthetic training dataset, this time for the segmentation of fruit and stems of pepper plants. In 2020, Park et al present their model

CUTNet [81], which uses a Contrastive, patch-based loss to create a generalized accuracy enhancing effect, similar effect to the segmentation seen in GANHands. Each of these examples is extremely relevant to our own work in chapters 4, 5 and 6.

3.3 Synthetic Data

In the literature there has been interest in Synthetic Data across a variety of fields since before machine learning had even been established. Early attempts at Synthetic data stemmed from Data Augmentation, the process of inflating a training datasets overall size by applying transformations such as flips, rotations and colour distortions to create new instances based on genuine images from the original set. As time has gone on Synthetic Data has become a large part of Deep Learning Vision for the many advantages discussed in section 2.3.

3.3.1 Datasets and Uses

Prior to 2014, virtually all Synthetic datasets of objects were either composites of real data or 3D renders, using video games or other graphical rendering applications (though both approach with still remains popular today). Examples of 3D renders like ShapeNet [18] or more recently the Falling Things dataset [107] are examples of large datasets of 3D models of objects with accompanying labels that allow the data to be used for deep learning. A common approach to using synthetic data involves combining synthetic and real images, often superimposing artificial elements onto real backgrounds [45], an effect which can be enhanced through image harmonization.

More recently there have also been interest in using video games and video game engines to generate synthetic data. GTA V [114] [90], in particular has been used to create a number of annotated datasets for different purposes, chosen for its high level of realism and the ability to exploit its rendering engine to produce pixel level segmentation masks of incredibly detailed scenes. Indeed outdoor scenes are one of the most popular applications for synthetic data [90] [16] [93], along with indoor scenes with the primary applications being use for self-driving vehicles and SLAM (simultaneous localization and mapping).

3.3.2 Realism

It is difficult to apply a specific metric to the quality individual synthetic datasets, making it hard to evaluate the impact of different approaches to creation of datasets. The domain shift problem is an established fact of working with synthetic data, however the degree to which *photo-realism* plays a part in this is much debated. Works such as Mayer et al [69] speculate that photo-realism is less important than low level features of realism, such as camera artifacts, which can be more easily simulated on synthetically generated imagery. Abu et al presents some similar findings [2], using a number of different camera effects on images from the virtual KITTI dataset, and showing this increases the ability of models trained on it to generalize back onto real images.

3.3.3 Best Practice

With the development of increasingly detailed and photo-realistic synthetic data, there has been much research into its effectiveness as training data in recent research. Indeed there still remains debate around the best practices around its use, with many approaches using a combination of real and synthetic data, either in combination as seen in [11] or at different stages of training. Nowruzi et al [79] uses the popular approach of first training on synthetic data before finetuning on a smaller dataset of real images. This fine tuning approach has been shown to be an effective method of training, often emphasising that a large synthetic dataset of moderate realism is able to learn the network a wider variety of content, while low level realism can then be learned from the real images, gaining the advantages of both. Where this multi-stage training regime is used, common practice is to freeze weights of early layers of a network, as seen in [45], allowing network training to focus on learning deep, domain invariant features.

3.4 Deep Learning in Plant Phenotyping

Research into improving digital phenotyping is carried out with a goal of enabling the automation of agricultural processes that are currently manually performed or unfeasible. In the experimental chapters of this thesis we make use of the different deep learning technologies described in this chapter for a range of phenotyping tasks. Synthetic data takes this even further improving the efficiency and accessibility of the AI technologies in themselves and aiding the implementation of computer vision technology.

Chapter 4

Improving Plant Organ Counting via Domain Adaptation

Counting crops from aerial photographs has become an area of serious interest for the Plant Phenotyping community in recent years. Neural networks have reached a point where the ability to detect, segment and measure very small plant components at volume is now possible, while technology such as drone operated cameras have made it much easier to capture huge amounts of image data quickly and easily. While this kind of performance is now attainable, it is a sterling example of a problem that is incredibly expensive to produce training data for, with each image containing potentially hundreds of components (leaves, stem, flowers, fruit are common examples) that needs individually segmenting.

In this work we present an annotated Synthetic Dataset designed for the object detection of wheat heads, we base our design on the popular Global Wheat Head Dataset or *GWHD [24]*, a multi-institutional project consisting of annotated datasets of top-down wheat images. The 2020 dataset we use contains 4700 images from different countries annotated with wheat head bounding boxes, different versions of the dataset are intended for use in research as a means to improve performance on challenging phenotyping tasks such as detection of individual wheat spikes. The outline for our dataset creation pipeline and use is shown in figure 4.1. We describe a pair of novel approaches to improving performance on object detection via domain adaptation which we demonstrate on our own synthetic dataset. First we introduce a guidance system for pixel-level domain adaptation, seeking to improve how content is maintained during style transfer. Secondly we use a clustering approach to control data synthesis, allowing for a more accurate domain adaptation when targeting a diverse distribution as seen in the GWHD.

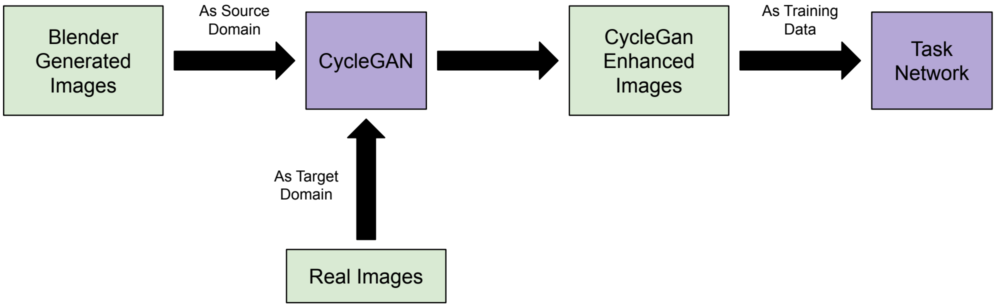


Figure 4.1: Here we show the intended pipeline for Synthetic data generation and improvement used in this chapter. This general overall outline is extended in this chapter as well as chapters 5 and 6.

4.1 Motivation

For this work we have a number of specific motivations.

Firstly, we are motivated by significance of solving the wheat head counting problem, and by improving the large scale automation of crop monitoring more generally. We hope that in the future it will be possible to use these techniques to gain accurate information on the health and yield of crops using only image data and machine learning models, giving useful data to both farmers and plant scientists. Advancements in agricultural AI of this type will likely in future lead to more automated farms, which will enable better monitoring of crops at cheaper costs.

Secondly, the specific dataset we are targeting is the GWHD, a dataset of nearly 5000 images containing individual annotations of nearly 200,000 individual wheat heads. By the standards of plant phenotyping datasets, this is already fairly generous, as many datasets consist of much fewer images. Despite the advantage of its size, the dataset contains a number of different subsets gathered from different institutions across multiple countries. Different subsets will see differences in wheat appearance caused by different climate and growing conditions, as well as differences in image appearance caused by different camera and image capture setup and hardware choices. This large diversity of images makes this training set extremely good for producing models that will generalize well, but also making the test set a challenging target for models aiming to perform well on diverse targets. By being able to demonstrate good performance on the GWHD test set using only or primarily synthetic data, we would be able to greatly reduce the future cost of building similar datasets for similar problems, either by requiring only unlabeled real images and relying on domain adaptation, or by reducing or eliminating their need entirely.

More generally, as there are a wide variety of different crop species even considering just variations of wheat. We are motivated to develop a pipeline of building synthetic plant datasets that allow for data of different species to be built flexibly and with reusability as a core consideration. If successful we hope that our approach to data synthesis would allow for new datasets to be created merely by switching our key assets, which could be produced cheaply.

4.2 Challenges

This project faces a number of challenges related first to the overall problem of wheat head counting, a non-trivial challenge in Plant Phenotyping, and second to the specific task of using domain adaptation on synthetic data. Here we break down these challenges and discuss both key obstacles and some considerations in overcoming them.

4.2.1 Wheat Head Counting

Here we list challenges specific to the task of individual detection of large number of wheat heads in an image.

- **Detection of many small objects.** Wheat head detection is more challenging than other object detection tasks such as detecting leaves or fruit as the individual wheat heads are extremely numerous and small in size. Many object detection networks have inductive biases that tend towards less numerous and larger objects, and this can make problems such as this harder to solve.
- **Occlusion.** Due to the density of wheat heads in any given images, with most of the dataset containing more than fifty wheat heads, it is extremely likely that each image will contain numerous cases of occlusion between wheat heads.
- **Diverse datasets.** The Global Wheat Head challenge attempts to reflect the wide range of wheat species, growth stages, and environments that exist across the world, as such the dataset contains images from 12 different countries and institutions. Because of this, we can expect it to be challenging to achieve good performance across all subsets of our test dataset.

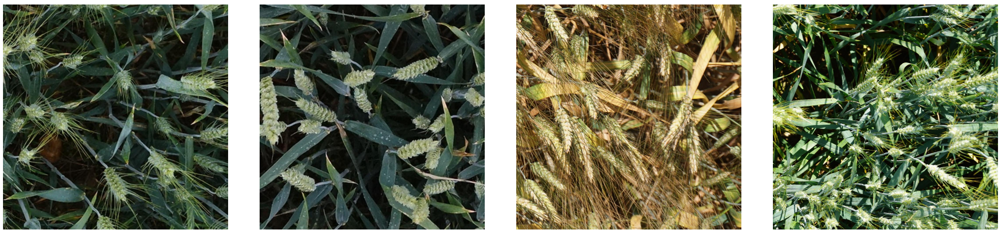


Figure 4.2: Examples of images from the Global Wheat Head Counting Dataset used in this chapter, highlighting the high level if diversity and challenge presented by wheat head counting.

4.2.2 Domain Adaptation

Here we list challenges specific to domain adaptation, specifically in the case of synthetic to real style transfer in the case of the GWHD.

- **Content Preservation.** During style transfer it is common for the content of the image to become distorted, causing the image to become misaligned from the annotations. Here we present a guidance system designed to reduce distortion and preserve this alignment.
- **Image Diversity.** As described above, the GWHD was created with diversity in mind; however style transfer networks often perform best with largely homogeneous datasets. In cases such as this, it is common for the network to simply devolve to producing images that are either an average of images in the target dataset, or mostly resembling the most common subset.
- **Image Resolution.** As style transfer models are often extremely large neural networks, often with multiple sub-networks, performing style transfer on high resolution images often can be hamstrung by limitations in available hardware. This is especially challenging for tasks such as detection of segmentation of small objects where high resolution is crucial to ensure accuracy.

In this chapter we present our novel approaches to domain adaption with a focus on overcoming the challenges of *content preservation* and *image diversity*, in order to work within computational constraints with some flexibility we downscale images for our experiments as described below.

4.3 Related Work

Synthetic data has been identified as a useful tool for plant phenotyping tasks in a number of other works. Two papers by Ward et al [118] [117], demonstrate a 3D modelled synthetic dataset of arabidopsis plants that does not use augmentations. This idea is then extended by Giuffrida et al, who present *Arigan* [110], a conditional-GAN approach to generating Arabidopsis plant images. Zhu et al. go further by constructing new segmentation masks synthetically using a compositing technique, and then generating matching images used a conditional-GAN using the masks as input [124].

Plant modelling tools have been leveraged for the creation of synthetic data in a number of works. Lindenmayer systems [65], developed first in 1968 are one such method, that can be used to create organic structures through iterative rules and become popular in the 1990 book *Algorithmic Beauty of Plants* [84]. We see use of L-systems in a range of works in both biology and science [20] [62] [108], and select this approach for our own 3D modelling. Another popular method is *Functional Structural Plant Model* (FSPM), which also allows synthetic plants to be created digitally. FSPM has been used in research to allow the accurate modelling of wheat [14] [34], Tomato plants [111] and other crops in a range of different research contexts.

Domain adaptation has in recent years been applied to different plant related domains. In 2020 Ayalew et al used a Domain Adversarial neural network to perform domain adaptation between different datasets of the same species [7] for object counting. Looking at using a single source dataset for both wheat and arabidopsis respectively they then targeted two different target datasets for each plant, demonstrating the methods viability.

More recently, our work bears some similarity to Najafian et al's work also on the GWHD [76]. Rather than a generative deep learning approach, instead a number of conventional Computer vision techniques were applied to synthetic images created using compositing to create realistic synthetic data for pre-training a network, after which the network would be fine tuned on a dataset more specific to the downstream task. Since the publication of our work we have also seen Mei et al demonstrate a similar domain adaptation strategy on rosette plants, using a density map for leaf counting [71].

4.4 Materials and Methods

In this section we describe our synthetic dataset and our pipeline for generating such images, our domain adaptation network, and the experiments we conducted.

4.4.1 Network Selection

For all our experiments we selected Detectron 2 as our task network. Detectron 2 is an implementation of Faster RCNN described in chapter 3, and was selected for its state of the art performance in object detection as well as its ability to perform well on large numbers of small objects. Additionally good support for custom datasets allows us to easily train the model with our synthetic images.

4.4.2 Synthetic Data Creation

To perform domain adaptation we created a pipeline to create synthetic images that would allow us to supplement the existing GWHC dataset of real images used for our experiments. By using 3D rendering we aim to create images that are as close as possible to the images in our target dataset to use as training samples. To achieve this, our pipeline uses Blender [15], a 3D graphics tool used for creating and rendering 3D scenes for a wide range of applications. Blender is especially useful for the creation of Synthetic data for deep learning as it has a number of features that we can use when creating our images:

- **Python scripting.** Blender supports full scripting using the Python programming language, allowing for automated scripting of both the randomised scene generation that creates each new arrangement of wheat as well as our rendering pipelines which generates each image and bounding box annotations.
- **Render and lighting settings.** Support for different rendering engines, as well as control over lighting and shader settings allows for a high level of control over realism when designing scenes through the tuning of a wide variety of parameters. For example using the *Cycles* rendering engine gives us a high degree of control over the shadows or reflections in our rendered scenes.
- **Plugin Support.** Support for additional plugins allow for additional features to be easily added, which can make automated scene design easier.
- **GPU Acceleration.** As this project requires the rendering of many tens of thousands of images, GPU accelerated rendering allows the creation of large

> datasets in days rather than weeks, even when rendering at high resolutions. Since the same compute resources are also used for training our neural network, this allows for an efficient reuse of expensive hardware.

Our goal is to use Blender to randomly generate new 3D models of scenes containing a large number of wheat heads similar to those images from our target dataset, creating a good artificial base that will allow for as accurate style-transfer as possible. As shown in other work on style transfer, more realistic performance is generally achieved when the source images have a strong resemblance to their target [123]; and we make a number of design decisions in our 3D model based on this insight. To this end we create a pipeline for generating new scenes automatically using a Python script which we can then render and, at the same time, capture ground truth data for from the randomised scene.

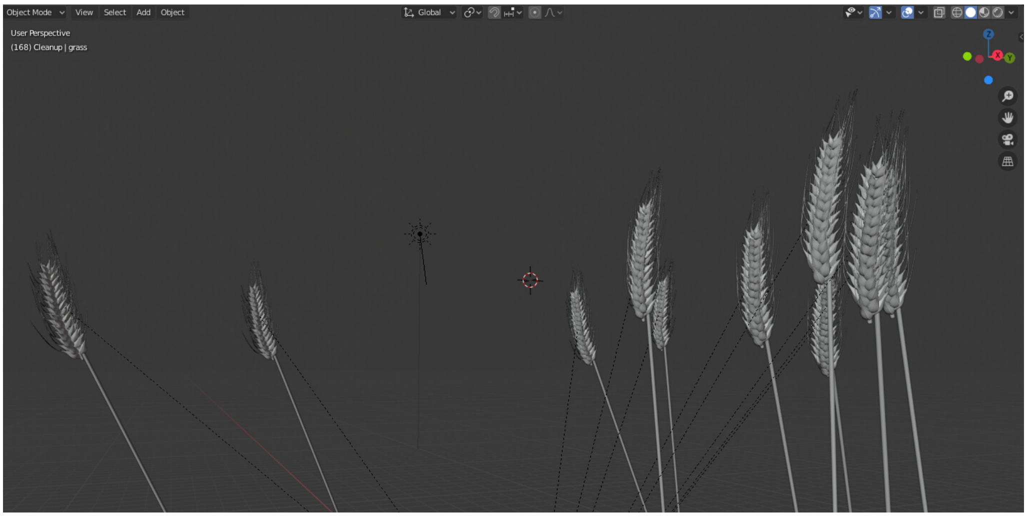


Figure 4.3: Wheat head models generated in Blender, before the effects of lighting have been applied by the rendering engine.

To create an instance of our scene, we generate a set number of wheat crops, consisting of an automatically generated stem cylinder and a hand crafted wheat ear asset that is a connected on top to create the finished look of each plant. We choose not to also add leaves to each wheat crop as these are not needed for wheat head detection, and will be added later during style transfer.

Each wheat stem was created using a custom Python extension that implements Lindenmayer Systems (L-systems), where natural looking structures can be generated via expansion rules defined as a string [83].

L-Systems

L-systems are a model for generating structures using a defined set of rules, that can be iterated over to create an increasingly complex structure. Below we include a simple representation of Lindenmayer's original L-system for modelling the growth of algae, which contains a starting axiom and two variables, each with an expansion rule. These variables can then be represented as plant components such as roots, stems, leaves and wheat heads to generate complex models of plants in 3D environments such as Blender. In order to generate a variety of different plants for our scenes, we can expand this concept by introducing non-deterministic rules into our L-systems, allowing different instances to grow in different arrangements.

```
variables : A B
constants : none
axiom   : A
rules   : (A -> AB), (B -> A)
which produces:

n = 0 : A
n = 1 : AB
n = 2 : ABA
n = 3 : ABAAB
n = 4 : ABAABABA
```

Though the structures for wheat crops are extremely simple, generating them this way gives us a great deal of control over their structure. By using L-systems in this way, it also makes our pipeline extremely versatile and leaves open the possibility to use out pipeline for more complex plants.

For each scene we generate a randomly determined wind *speed* and *direction.* The rules used to generate the L-system for each stem in a given image will be modified based on these values causing the wheat stem for each plant to tilt in the direction of the wind, in proportion to the simulated wind speed. As such while every crop will be slightly randomised the overall scene is able to consistently model the strong bending due to wind force common in images of wheat fields, matching the effect of the wind seen in real images from the GWHD.

Scene Generation and Capture

For each image we generate a target number of wheat plants, between 10 and 60, as this is the approximate range of values represented in our target dataset. Unlike the real GWHD we include the same number of images for all wheat head counts, by doing this we hope will help the network perform better on outlier test images with very few wheat heads in them which are poorly represented in the real train data. The generated wheat plants are positioned in a uniform random distribution above a background image of soil, then our python script adds addition hand-crafted 3D models of foliage, which are also randomly scaled, rotated, and positioned in the frame to match the composition of the real target images. Our scene also contains three different lighting sources positioned around the wheat, which we move and adjust the brightness of as part of our scene generation pipeline. When the scene is rendered this will allow for realistic lighting and shadows to be applied, aiming to simulate the challenging extreme ranges of contrast, brightness and exposure seen in real images from the GWHD. Finally we render the scene as viewed from a Blender Camera object, positioned above the scene. All images were captured in 1024x1024 resolution; the high resolution was chosen as it is the same resolution as images from the global wheat head dataset (GWHD) and will make style transfer easier by matching the resolution with our synthetic renders. In addition it is unlikely an object detection network will be able to detect individual small objects like wheat heads in a lower resolution image. We show an example of the scene in blender in figure 4.4, showing the camera positioned above a typical scene.

Image rendering was done using an NVIDIA Titan GPU using the Cycles render engine, which is generally considered to be the most realistic of the rendering engines packaged within Blender, albeit at a high computational cost. Each image and label can be captured and the ground truth recorded in under 5 seconds on average, with the greatest overhead actually being the creation and deletion of each scene rather than the image Rendering. Without GPU support however the render time would be much greater and result in nearly 30 seconds per image. We do note that some variation does occur during rendering depending on the number of wheat heads in an image, with render time increases as we increase the number of wheat heads in each generated scene during data generation. Overall we can produce in excess of fifteen thousand image-label pairs per day after our scene model has been created. Even with the time taken to both learn Blender to a sufficient level, and create the scene taken into account, it is likely that the time and cost associated with creating this synthetic data is a fraction of that taken to collect the same quantity of real data. Additionally, we hope that our 3D scene and script created would be reusable, reducing the cost of creating future datasets.

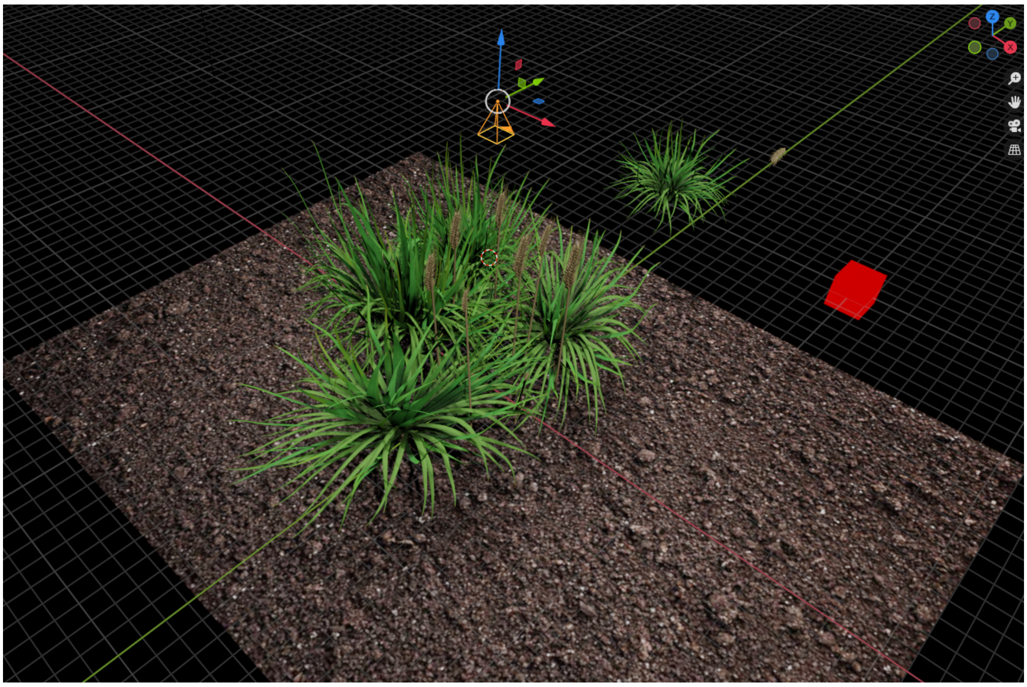


Figure 4.4: An image of our blender setup, showing synthetic wheat heads dispersed among foliage and the camera used to capture each image positioned above the scene. Scripts for scene generation and l-systems can be found at github.com/zanehartley/zhlsystems.

4.4.3 Generating Ground Truths

For each image we were required to create accurate ground truth labels showing bounding boxes for each ear of wheat. In order to do this we used Blender's camera data combined with the world coordinates for each wheat head to calculate the coordinates of a bounding box encompassing the entire wheat head using maths based on a pinhole camera model. Details of the camera model needed are available within the blender camera settings. We extract the values of the bounding box within the camera frame and exported these to a CSV file for each image which can then be used as annotations for our image. Our complete synthetic dataset $\mathbb{S}_I$ contains over 5000 new images with over 100,000 new wheat head annotations stored as csv data to be used in training for our experiments.

Finally to create additional labels for our heatmap support we used OpenCV [51] to create Gaussian heatmaps for each image. Gaussian heatmaps were chosen as they are a common representation for similar problems, and could be simultaneously predicted during domain adaptation. To create these heatmaps we used the extracted bounding box values by first taking the center points of each bounding box and then adding a small circle of noise for each wheat head. This created a set of labels $\mathbb{S}_H$; this process was repeated to create heatmaps for the real world training images $\mathbb{R}_H$ used as our target domain which are supplied with bounding box labels. We explain the use of these heatmaps in section 4.4.7.

4.4.4 Use of CycleGAN

Our goal is to create new images that can be used for training our network by transforming our synthetic images to the *real* domain and thus mitigate the domain shift problem. We aim to perform style transfer on our images using an extended CycleGAN model, an adversarial neural network used for changing the domain of images that we discuss in section 3.1.3 which we treat as one unified *real domain* dataset. Popular examples of domain transfers performed using CycleGAN include horses-to-zebras, apples-to-oranges and photo-to-art, often using specific artists styles.

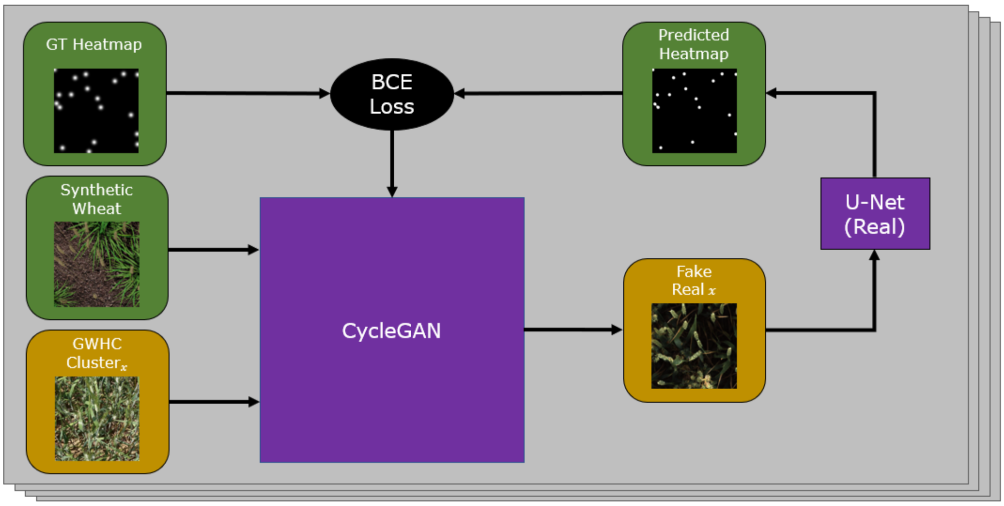


Figure 4.5: Our CycleGAN model with Gaussian heatmap support; this process is repeated for each cluster extracted from our target dataset (represented by the four layers in the figure), producing multiple outputs for each input synthetic image.

4.4.5 CycleGAN Structure

As described in chapter 3, CycleGAN uses 4 separate networks working together consisting of two of the conventional generator-discriminator pairs we are familiar with in usual GANs. Each generator takes an input image from one of the two domains, and outputs the transformed image in the other domain, with one network going from domain A to B and vice versa. CycleGAN's name is derived from this structure; an image can be passed from one generator and into the other, making a cycle going from domain A to B and back to A. For example this might be Horse to Zebra to Horse. This is important as it verifies that the original image can always be restored from the transformed version, which ensures that the network does not destroy *content* information during style transfer.

In order to enforce high quality style transfer the CycleGAN model contains a number of loss functions.

- **Discriminator Loss.** A discriminator evaluates an image either from the target dataset or the output of its corresponding generator, attempting to determine the authenticity of the image. For each real or synthetic image the network will predict a probability of the image being real. This is a sigmoid cross entropy loss that we seek to minimize, meaning we aim to make the discriminator better at correctly assigning high probabilities of it being real to real images and low probabilities to synthetic images. This loss is fundamentally the same as that we see in most conventional GAN models.

- **Cycle Consistency Loss.** An image from the source domain is passed through the source-to-target generator, followed by a backwards pass through the target-to-source generator. This output is then compared against the original source image using an L1 loss, which is then back propagated through both generators. This loss is perhaps the most important contribution from CycleGAN, ensuring that content is preserved during style change, by making the effect of each generator reversible. The same process takes place simultaneously in the other direction, going from target to source domain and back.
- **Identity Loss.** An image from the source domain is passed through the target to source generator, the output is then compared against the original using an L1 loss. As the image is already in the source domain we want to encourage the generator to leave the image unchanged. This means that elements of the image that already match the target domain (for example the background) will remain unchanged whereas known objects will be transformed, though this is less relevant in our case as all of our image represents our source (synthetic) domain.

4.4.6 Problems with CycleGAN

CycleGAN is generally well regarded as one of the first high quality style transfer GANs and performs well on many of these domain transfer tasks. But for *synthetic-to-real* domain transfer for use in creating training data, CycleGAN exhibits a number of problems that we seek to address in this chapter.

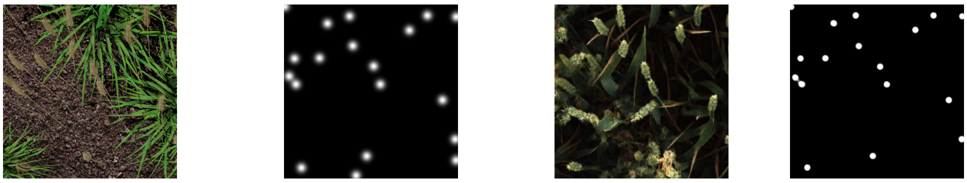
Firstly, while CycleGAN excels at re-texturing objects to their new domains, often we can observe some distortion around the edges of objects, especially where the difference between domains is greater, or with smaller details. This is especially a problem where we are attempting to style transfer training data, as this distortion can cause a misalignment between image and labels. As the different ears of wheat captured in the GWHD are varied in appearance, it is common for this distortion to occur around wheat heads often causing this misalignment.

Secondly we observe that when target datasets are especially varied, the *source-to-target* generator fails to generate a wide range of images that reflect that diversity in the images it creates. The generator instead learns to generate either an average of all the target domains, or to instead simply favour the most common subset of the target domain.

4.4.7 Extending CycleGAN with heatmap support

To solve the first problem of distortion and content shift we adapt a CycleGAN model as shown in figure 4.5 to predict Gaussian heatmaps of wheat head locations from the output of both the *real to synthetic* and *synthetic to real* generators. By doing this we aim to preserve the locations and geometry of wheat ears in images transformed by the generators, as wheat heads added or removed will create inaccuracy in the predicted heatmaps. To do this we extend the architecture with two lightweight UNets that predict heatmaps for the outputs of each generator. Our CycleGAN is set up with default parameters for training, and our synthetic dataset and the training split of the GWHC dataset are used as source and target domains respectively. All images are resized to 400x400 due to the high VRAM constraints of combining CycleGAN with additional models as support, making style transfer in the images full 1024x1024 impossible with 12GB VRAM GPUs such as NVIDIA Titans. Additionally, for all experiments we perform, all training images and testing images are resized to 400x400 to be consistent with GAN output images.

For each iteration, our model receives a source image $\mathbb{S}_{I_i}$ and a target image $\mathbb{R}_{I_i}$ and transforms each to the other's domain, *synthetic* and *real* respectively. The predicted heatmaps are compared against $\mathbb{S}_{H_i}$ and $\mathbb{R}_{H_i}$ and we apply a binary cross entropy loss to simultaneously train each UNet and enforce accurate translation of the wheat heads by the generator. An example of both the input and output images along with ground truth and the predicted heatmap can be seen in figure 4.6 After training our model we produce dataset $\mathbb{H}$ where all images from $\mathbb{S}$ have been converted by the *synthetic* and *real* generator. We also produce data $\mathbb{C}$ where we use an unmodified CycleGAN as a baseline to compare our extended model against.



Synthetic Input *Ground Truth Heatmap* *Output Image* *Predicted Heatmap*

Figure 4.6: Examples of our Heatmap Supported pipeline, showing input image and groundtruth paired with a corresponding output and predicted gaussian heatmap. High similarity between ground truth and predicted labels indicates wheat head location has been preserved during augmentation.

4.4.8 Feature-Based Clustering to Improve Diversity of Generated Images

Using CycleGAN to convert images to the real domain using our *real* wheat head image dataset presents problems due to the heterogeneity of the images present in our target domain. Rather than producing images that represent the diverse range of images in the GWHC dataset, the generator instead learns a translation to an *average* wheat image representation. This is a problem we want to avoid as the average image is not a meaningful representation of real life.

To counter this, we apply a preprocessing step of splitting our dataset into distinct visual appearance clusters to be trained as separate targets for our generator. To do with we use a pretrained InceptionV3 feature extractor to obtain a feature vector from each image in $\mathbb{R}$. We then apply K-means clustering to group the images into clusters of similar images. We select a value of k=4 to achieve the best compromise between maximising the number of clusters, while ensuring enough images are present in each set to make CycleGAN viable. Increasing the value of k may lead to more diversity in the data produced, however while we can easily change this value a higher number of clusters means we must run additional models which is more computationally expensive and may lead to poorer results due to the smaller size of individual clusters. Visual inspection confirms that each cluster appears visually similar to other images in the same group. In our case k=4 makes sense due to the four main appearance modes in the dataset; examples of each modes can been seen in figure 4.7.

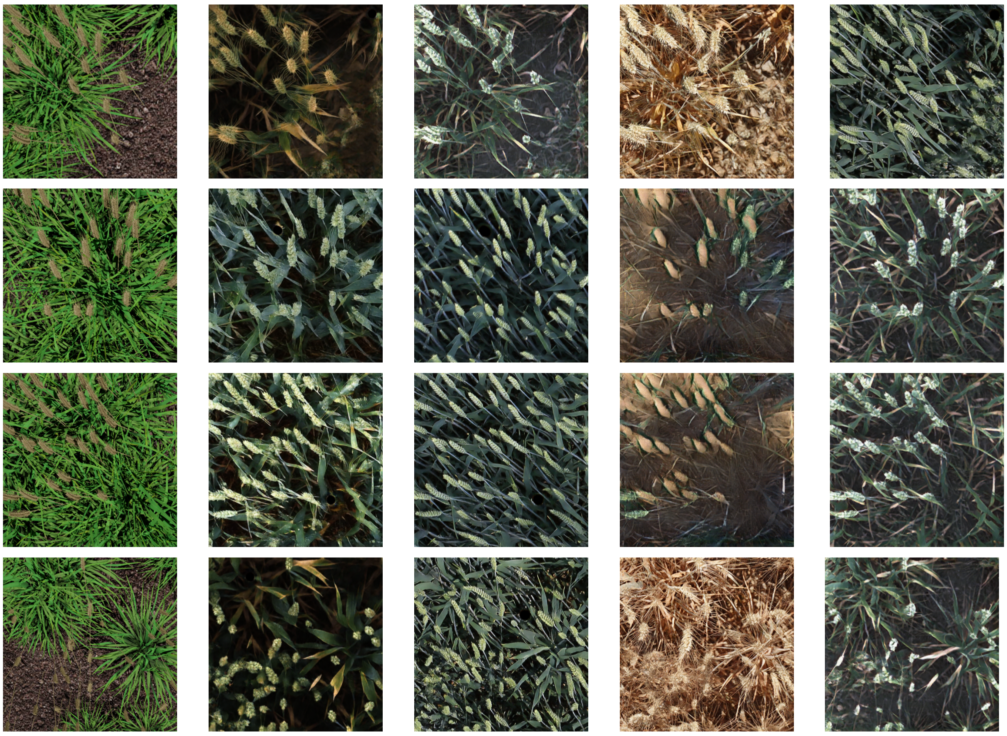
For our experiments we trained four of our extended CycleGAN models using each cluster as a target respectively, testing this methodology both with and without heatmap support. Each model was then used to generate a *real* representation for each image in our synthetic dataset, in doing so creating four style-transferred images for each synthetic image and quadrupling the amount of data available for training Detectron. By combining these four sets of augmented images we create our final dataset $\mathbb{K}$ which we use to obtain our final scores.

4.4.9 Experiments

We conduct a series of experiments to evaluate the ability of our synthetic data to improve the object detection model by supplementing our original *real* dataset. We then conduct additional experiments to evaluate the impacts of both our heatmap and clustering approaches described previously by comparing models trained with and without each of these techniques.

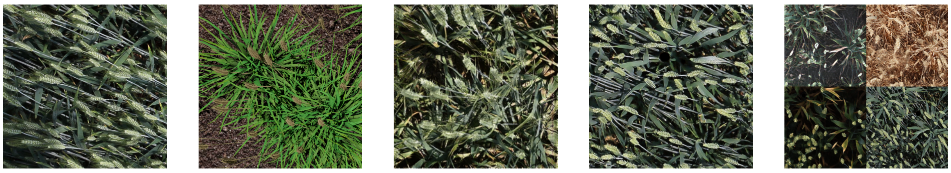
(1) Real Only. Here we establish a baseline performance achieved by using only the real data from our training split $\mathbb{R}$ of the GWHC dataset, containing over 3000 images. For all our tests all images have been resized to 400x400 for consistency. We expect this set to perform well as it is already diverse and highly representative of the test split.

(2) Synthetic and Real. We evaluate the performance gain combining our synthetic dataset $\mathbb{S}$ with the real training data $\mathbb{R}$ - but without GAN modification. Synthetic data has been leveraged to improve performance in a number of other domains however we hypothesise that it is unlikely to have a major impact



Original *Cluster 1* *Cluster 2* *Cluster 3* *Cluster 4*

Figure 4.7: Examples of 4 images produced from a single synthetic image for our Heatmap Supported 4 clusters experiment. Each cluster represents a different broad category of images from ℝ, these subsets may be grouped by wheat species and growth stage, time of day or other factors in how the image was captured. Here we can see that cluster 2 in particular produces more hallucinated wheat heads than the other datasets, and this is most pronounced when using synthetic data with a lower number of plants.



Dataset ℝ Dataset 𝕊 Dataset ℂ Dataset ℍ Dataset 𝕂

Figure 4.8: Images from all dataset listed in section 4.4.9. ℝ is used in experiments 1, 2, 5, and 7, 𝕊 is used in experiment 2, ℂ is used in experiment 3, ℍ is used in experiment 4 and 5, and 𝕂 is used in experiments 6 and 7.

on performance due to the substantial domain shift between synthetic and real images.

(3) CycleGAN. Here we evaluate the performance achieved using images from ℂ which have been augmented by an unmodifed CycleGAN, as our training set. We perform this experiment to verify that our support network improves scores against a suitable comparison. Due to cases where the CycleGAN either removes true wheat heads or incorrectly adds new ones (hallucinating heads where there should be none) to the image, we hypothesis that this network will perform poorly as the network will receive penalty for predicting what appears as true wheat heads but there is no label in the annotation data.

(4) Heatmap Supported, No Real. We evaluate the performance when Detectron is trained using only images in $\mathbb{H}$, generated by our CycleGAN with Gaussian regression support. We hypothesise that this network will outperform experiment three as a result of our heatmap support, but it is unlikely to perform as experiments that include real images.

(5) Heatmap Supported and Real. Here we combine $\mathbb{H}$ with the *real* images from $\mathbb{R}$. We expect this to boost performance on the network a great deal compared with experiment 4 because of the inclusion of images from the target domain.

(6) Heatmap Supported 4 Clusters, No Real. As described in section 4.4.8, we create 4 subsets of the original training set to create converted datasets for each of the 4 targets using our extended model. We combine these four datasets into a combined set $\mathbb{K}$. We expect this network to perform well even when real images are not used to train detection due to images in $\mathbb{K}$ being a close likeness to images in our test set.

(7) Heatmap Supported 4 Clusters and Real. Finally we combine $\mathbb{K}$ with $\mathbb{R}$. We expect this model to perform the best of all the experiments listed as it combines all the advances of our approach plus real image data.

4.4.10 Training

Our CycleGAN with heatmap support models were run using standard CycleGAN settings as found at https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix. Both generators are extended with an additional model to predict Gaussian heatmaps of wheat head locations, both of which share their Adam optimisers and learning rates of $2e - 4$ with their respective generator. These models were each trained for 100 epochs after which GAN training performance began to degrade, which was verified empirically.

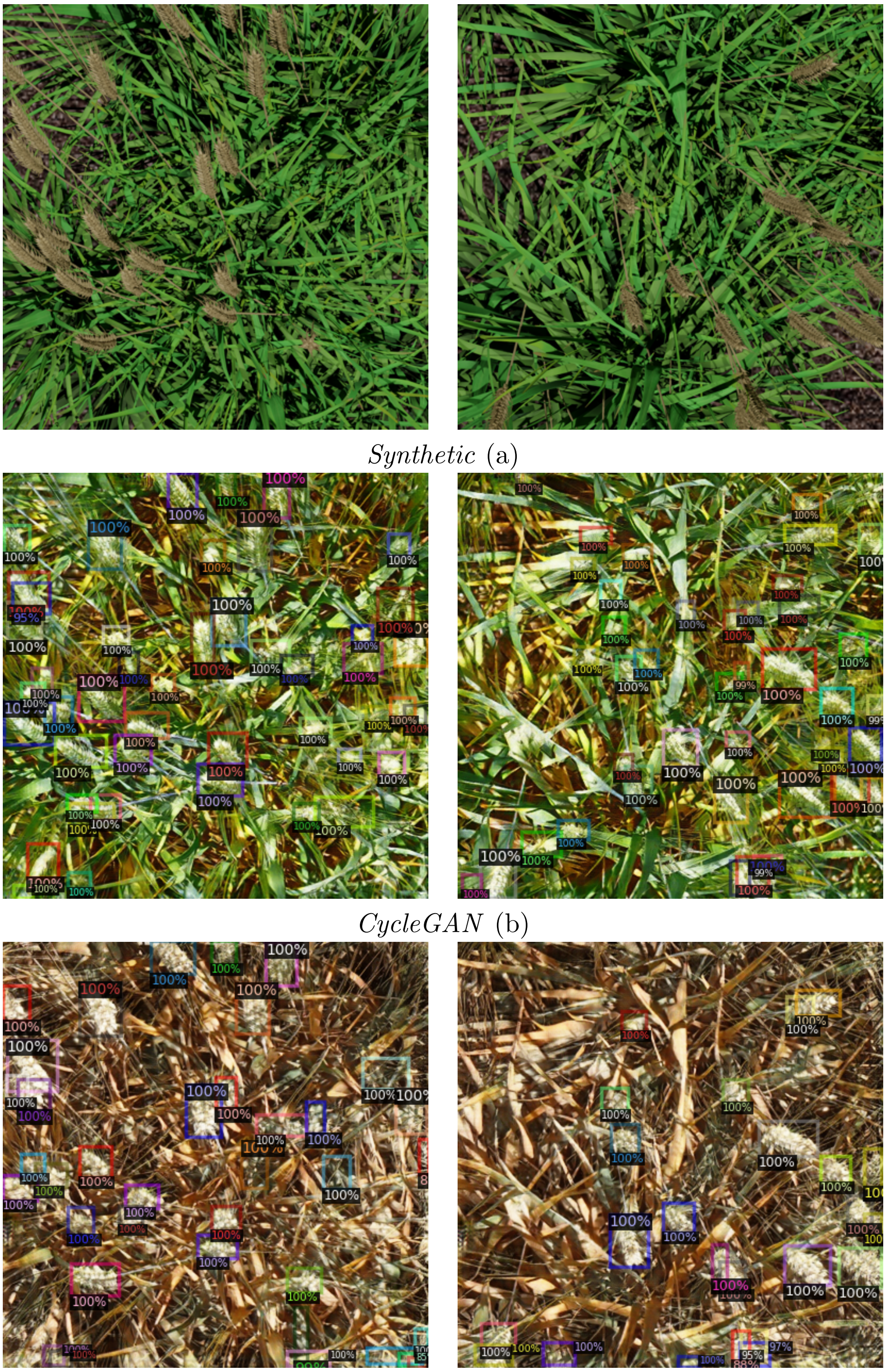
Our experiments were performed with an unmodified Detectron 2 using standard settings, trained on NVIDIA GTX Titan X (Pascal) GPUs for 60 epochs.

4.5 Results

In this section we show the images produced by our domain adaptation models as well as presenting the results of the experiments described above. We compare our results against a baseline achieved using only real image in our training dataset. For our evaluation we use *Mean Intersect Over Union (IoU)* which scores the average overlap between predicted and ground truth bounding boxes, as well as *Mean Euclidian Distance* which measures the average distance between bounding box centers.

4.5.1 Domain Adaption Results

In figure 5.2 we compare augmentation between a conventional CycleGAN and our extended model. We see that visually the images produced by both models look very different. The conventional CycleGAN image being more saturated than images produced by the supported network, though it is likely more imperceivable changes are being made by the generator that are difficult to identify by eye. We also observe that Detectron makes more predictions for the image produced by the unchanged CycleGAN model and as shown in table 4.1 this leads to worse overall performance because of a large number of false positives. It is difficult to infer exactly why this is due to the black-box nature of deep learning models, and a thorough analysis into AI explainability is beyond the scope of this thesis. Overall we see a small positive performance increase thanks to our support network component (seen in the differences between experiments 3 and 4), but the impact was a less important contribution to our overall result than we originally expected. This may be due to the unmodified CycleGAN proving more effective at maintaining geometry than we had initially hypothesized.



Synthetic (a)

CycleGAN (b)

Heatmap Supported CycleGAN (c)

Figure 4.9: Comparison of synthetic images (a) augmented with unmodified CycleGAN model (b) and CycleGAN with heatmap support (c). Images in column C demonstrate a realistic colour and higher contrast between wheat heads and background. We observe that for the image from column B (dataset ℂ) more predictions are made, leading to a lower accuracy.

Experiment	**Mean IoU ± SD**	**Mean Euclidean Distance**
(1) Real Only	0.8262 ± 0.07	13.9079
(2) Synthetic and Real	0.8568 ± 0.06	10.7702
(3) CycleGAN, No Real	0.5625 ± 0.35	16.9433
(4) HM Support, No Real	0.5987 ± 0.33	12.7729
(5) HM Support and Real	0.8497 ± 0.06	11.1276
(6) HM Support 4 Clusters, No Real	0.6622 ± 0.30	14.7164
(7) HM Support 4 Clusters and Real	**0.8642** ± 0.06	**10.5617**

Table 4.1: Results of the experiments described in section 4.4.9, showing results for Mean IoU (higher is better) and Mean Euclidian Distance (lower is better) reported for test split of 100 GWHC images. The best scores in both metric were achieved by experiment 7 which has been highlighted.

Figure 4.7 shows an output example of our CycleGANs used to create dataset $\mathbb{K}$ for experiments 6 & 7 along with an input image. Here we can see examples of how a single synthetic image from our source domain can be transformed to a number of different domains each matching a subset of the overall *real* target domain. In all cases wheat heads appearing in the synthetic image, and by extension their annotations, have had locations maintained well, and appear realistic while maintaining their geometry. Some additional wheat heads are observed to be hallucinated especially in cluster 4, due to that dataset containing a larger number of wheat heads overall but this is less so than we would expect to see using a conventional CycleGAN.

4.5.2 Wheat Detection

For each of our experiments we evaluate our results using both Mean IoU of bounding boxes, and mean Euclidean Distance between center points of the boxes. We report results of all our experiments in table 4.1. In experiment 1, the baseline achieved by training Detectron with only $\mathbb{R}$ performed well, as expected. Experiment 2 exceeded our expectations by increasing the baseline score, suggesting the synthetic data we have created did a good job at imitating the target domain and this had a less substantial domain shift problem than would be expected from most synthetic datasets.

In experiments 3, 4 and 6 we compare the performances achieved by $\mathbb{C}$, $\mathbb{H}$, and $\mathbb{K}$ before introducing images from $\mathbb{R}$. We observe that the heatmap supported

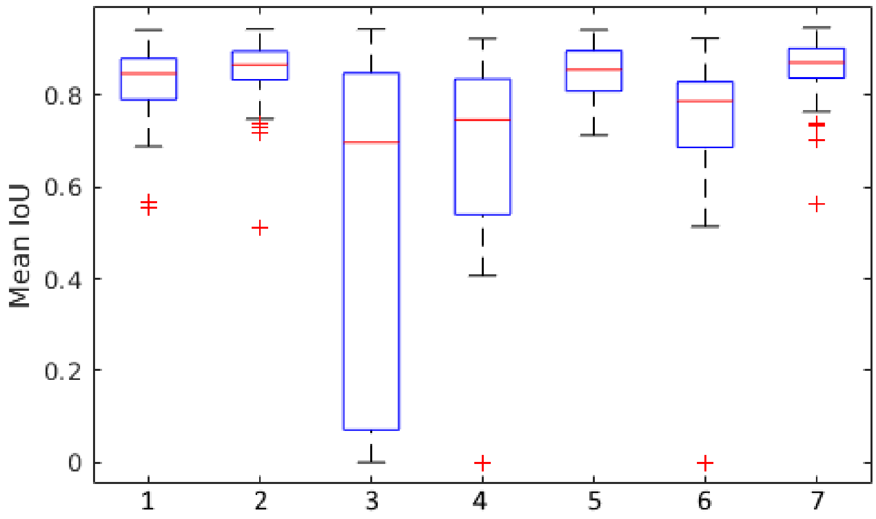


Figure 4.10: A boxplot visualisation of data for each of our Mean IoU scores.

network offers a small improvement over an unmodified CycleGAN, and that our clustering approach leads to a much larger improvement.

Finally in experiments 5 & 7 we compare scores achieved when we add the real images from ℝ to ℍ and 𝕂 respectively. Surprisingly experiment 5 shows only a small improvement, not even beating our results from experiment 2 using unaltered synthetic data. Experiment 7 got our highest scores in most metrics, beating our baseline by nearly 5%.

In figure 4.10 we show further data on the results of our experiments. Here we see that each experiment without real images during training has much higher numbers of outliers, where individual data points are outside the trend of the results as a whole. Some of the images included in our test set have very small numbers of wheat heads, and poor performance on these images may have increased the spread of the data in these cases. We also observe that experiments 2, 5 and 7 (which include real data during training) all have very low standard deviations shown in table 4.1 in addition to their very high scores overall, indicating that most data points are close together and suggesting our model has a consistent performance across the entire dataset,. This suggests consistently high performance of the proposed approach across all test instances.

4.6 Discussion

4.6.1 Analysis of Results

In our results we see that the difference between experiments 2 and 5 is within one standard deviation, with experiment 2 (using unmodified synthetic images) achieving good scores in both Mean IoU and Euclidean Distance. We hypothesize that with heatmap support alone there is still some loss of accuracy, which we expect might be caused by domain adaptation failures. It is, however, likely that the strong performance of $\mathbb{S}$ in experiment 2 indicates that we have succeeded in producing a high quality synthetic wheat dataset well suited to use as a foundation for further enhancements.

We observe that in experiments 6 and 7 the addition of our clustering approach produces the best scores in both metrics. Experiment 6 is noteworthy for having the best score achieved for any method that did not use real images at all, suggesting that an unsupervised adaptation of our methodology might be viable (we present an unsupervised domain adaptation approach in chapter 5). Similarly experiment 7 achieving improvements over all other methods shows the efficacy of our overall approach.

A further take home message is just how much benefit adding real images has, versus synthetic alone (experiment 4 vs. 5; experiment 6 vs. 7). Performance improvements of as much as 20% can be found by combining real images into the set. This highlights the value of any labelled real images that can be added to the train set.

As discussed in section 4.4.7, for all the experiments presented in table 4.1 we used a confidence threshold of 0.7 for all predictions (meaning only predicted boxes that score 0.7 or higher with a ground truth bounding box will be counted as a true positive). During additional experimentation we observed that for experiments 4 and 6, where no real images were used higher scores could be increased by lowering the confidence threshold to 0.4, although this would produce much lower scores for all experiments where real images from $\mathbb{R}$ are included in the train set. As images from these networks were never exposed to any real images during training, it is likely that it has lower confidence of any predictions made on our test set, and as such the high 0.7 threshold causes some correct predictions to be discarded. However, it is clear that optimising this per experiment, although feasible for a particular real world problem, would not be considered fair, so was kept constant for the results in Table 1.

4.6.2 Future Work

The method presented in this chapter is well suited to being applied to other detection tasks especially where data availability is a limitation. Especially in fields like plant phenotyping where data for different crop varieties is often unavailable, we believe our method could be applied to detection of disease foci, or necrosis as well as detection and counting of a range of plant parts and organs, such as stems, sticks, internodes, leaves or fruit.

Additionally the pipeline we present in this chapter for generating 3D scenes in Blender could be easily extended for further use in other plant phenotyping tasks, and potentially packaged as a Blender plugin, allow for the development of new datasets of Synthetic plants. One such example is weed detection in fields, such as in the the phenobench benchmark, where synthetic data could both suppliment real data and reduce cost of annotating such a large dataset. Additionally our pipeline also includes code generating segmentation masks in addition of bounding boxes for object detection, a much more expensive task to produce handmade annotations for.

4.7 Conclusion

In this chapter we have presented a new approach to improving scores on Wheat Head Detection using a supported CycleGAN and a novel clustering method that allows us to increase the quantity of our training data while also increasing diversity in our data produced by domain adaptation. Our results show our methodology improves scores when tested on unseen images from the GWHC dataset compared to a baseline score set by real images from the same dataset. Our method is also highly generalizable and could be easily adapted to work on other plant phenotyping problems especially where smaller quantities of training data are available for example additions to the synthetic pipeline could allow for the detection of weeds or other dangers to crops, or our pipeline could generate additional annotations like segmentation labels. By supplementing and replacing real data with synthetic image as seen in this chapter we are able to reduce to human cost in collecting and annotating real world data; an approach that is increasingly effective at larger scale, especially due to the reusability to synthetic assets and pipelines. To facilitate other researchers to make use of and extend our work we provide code for synthetic scene generation at github.com/zanehartley/zhlsystems and code for our extended CycleGAN at github.com/zanehartley/CycleGaussian.

Chapter 5

Leveraging Domain Adaptation for Unsupervised 3D Reconstruction of Fruits

Chapter 2 demonstrated that synthetic data created in a 3D rendering program can be used as valuable additional training data for Computer Vision phenotyping problems. We show that applying style transfer to synthetic images using real analogous images as a target dataset can allow us to improve scores, and supplement real data where dataset size is limited. This chapter extends these previous ideas focusing on the more challenging task of 3D reconstruction, and moving from semi-supervised domain adaptation to a fully unsupervised approach, substantially increasing the challenge and significance of the task.

5.1 Introduction

3D reconstruction is the core Computer Vision task of capturing the shape of objects in 3 dimensional space; and in recent years it has been used in agriculture and plant sciences for a range of applications. Extraction of 3D phenotype, structural and morphological traits can be useful to biologists, but 3D modelling of plant organs also has many applications as part of a pipeline for other downstream phenotyping tasks. Plant organs, such as leaves, roots and fruit are some of the most prominent targets for 3D phenotyping. As described in section 2.5, automatic inspection of crops can be useful in predicting yields. For fruits in particular 3D structure can also be used to assess market class, where exterior traits such as shape, size and outer quality are extremely important. These kinds of tasks have historically been performed manually at time of harvest, which has drawbacks of cost as well as introducing inconsistency. For other targets, such as leaves or entire plants, 3D reconstruction can be used to assess stages of growth more accurately than the 2D alternatives. For downstream tasks, we can use 3D information about plant structure to aid in other common tasks such as plant organ counting, e.g. counting number of fruits on a single crop.

In this chapter, we will attempt to answer the following research questions.

1. Can synthetic data be used to train a CNN for more complex tasks, such as 3D reconstruction? And what level of performance can be achieved using this method?

2. Are we able to extend an existing style transfer network so as to perform unsupervised domain-adaptation using our synthetic dataset at the same time as 3D reconstruction?

3. To what extent is building a 3D dataset for Unsupervised Domain Adaptation cheaper and easier than it would be to collect real 3D training samples? And consequently, how realistic is synthetic data as a replacement for real data in this kind of problem?

To this end we design and implement a Convectional Neural network that performs simultaneous domain adaptation and 3D reconstruction from a single image. We demonstrate the efficacy of our approach on bananas, chosen because they present a challenging variety of both 3D shape as well as colour and texture - for example, they are asymmetric, and exhibit bruising and other unique textural features. While 3D reconstruction of entire plants is also an area of interest, this is much less feasible to perform from a single image due to the greater level of occlusion presented. There is good availability of representative 3D models and photographs of bananas that may be used to produce synthetic and real datasets, aiding our domain adaptation approach that exploits both real and simulated data.

5.2 Motivation

Accurate 3D reconstruction of fruit is a task with a wide range of downstream applications that allow us to extract a number of important traits from individual specimen. Automated harvesting using robotics is an area of significant interest [5] and requires precise understanding of the 3D structure so as to allow robots to interact with each individual fruit without causing damage. 3D reconstruction can also be used as part of a pipeline for estimating biomass and yield, as volume is closely linked with the mass of a fruit, vegetable or crop. This is extremely important as non-invasive biomass estimation is a core challenge of automated agricultural monitoring.

Analysis of fruit structure and form is also an important component in *Grading*, the process by which a fruit can be classified based on quality. A numeric value is often given to each specimen, and this value is extremely important for sorting as grade often affects price and marketability. Factors that contribute to a fruits grade include shape and size as well as color and presence of defects, and there is a history of Computer Vision being used as part of this process. In most cases where computer vision has already replaced human graders, a variety of 2D approaches have been presented allowing for a limited level of analysis, often to be used as only one part of a multi-stage grading pipeline that will still contain humans as well and thus remain expensive and time consuming.

In the past 5 years we have seen an increasing number of 3D reconstruction approaches presented [94] [8] [54], however in most cases we see either multi-view approaches, or the implementation of additional sensors, usually to capture depth information. As a result of this most of these approaches are expensive and time consuming, usually because of their complex setups which also makes them less easily implementable outside of an experimental setting.

Having identified both the value of 3D reconstruction of fruit and also the limitations of current approaches, we were motivated to improve on this approach by utilizing recent advances in deep learning for 3D reconstruction. While 3D reconstruction is not a primary focus of this thesis, we were also motivated to apply a synthetic data approach to a problem that is this complex, as 3D computer vision problems are generally considered much more challenging and face additional problems that make them more difficult to solve.

We select bananas as

5.3 Challenges

3D reconstruction is perhaps one of the most challenging reconstruction problems, especially in a monocular setting where we are limited to a single input image. In this section we consider some of the challenges this work seeks to tackle.

- **Range of angles.** 3D reconstruction can often be performed most easily when the camera and target are fixed, often with the target against a known background. In our work we aim to overcome this limitation, allowing for images to be taken from a range of angles and distances. We aim to represent a wide range of angles and distances from the camera in our synthetic data, which we hypothesise will teach the network to perform well on real images with a similary range of camera positions.
- **Computational cost.** Neural networks working with 3D data face significant constraints caused by the large spacial dimensions of the volumes being used. Network design for this kind of problem is heavily constrained, especially without leading to especially long training times and bottlenecks, however this often comes at the cost of resolution and accuracy.
- **Generating diverse data.** Unlike in chapter 4, where we were solving an object detection problem, for a 3D reconstruction problem we need to be able to produce a range of training samples that contain different 3D representations. This is more challenging than object detection as in that

problem simply positioning wheat ears in different positions in the frame was enough to produce added variety, which would be insufficient in the case of 3D reconstruction.

5.4 Previous Work

3D reconstruction is a fundamental Computer Vision problem with a wide range of applications and approaches. 3D reconstruction is in many settings done using some combination of lasers, as seen in Dornbusch [30], time of flight cameras as seen in Kazmi et al [57] or other sensors, with specialised technology being the most reliable way to perform extremely accurate reconstruction even today. The challenge for computer vision researchers is to look for approaches that use only conventional RGB cameras. It has been common for 3D reconstruction to be achieved using a variety of techniques broadly classified as photogrammetry, and even more recently by deep learning. In this section, we look at some of the approaches used in academic literature, specifically for plants and fruit.

5.4.1 Photogrammetry

Photogrammetry is, in the context of 3D Reconstruction, the approach of combining a large number of images from multiple views of an object. Often this is done by finding correspondences between the images; these correspondences will generally be determined by various kinds of image features, and using these known correspondences to combine the data from the images into a 3D representation. Zhang et al [122] in 2008 use this approach to reconstruct corn plants, using a pipeline identifying leaf boundaries, followed by a reconstruction step that assembles the edges detected in each image using a stereo vision intersection algorithm. More recently Rose et al [94] reconstruct tomato plants using structure from motion data using a multi-view stereo camera setup. Ayob et al [8], also use a multi-view photogrammetry system, with the aim of performing accurate volume estimation of chilli plants. What all of these methods have in common is the triple cost of setup cost, time to capture each instance, and complexity, making the systems impractical to use in most agriculture settings.

Working with smaller objects such as fruit or individual plant organs present additional challenges. In Jadhav et al [54] they use 3D reconstruction as part of a pipeline for estimating volume and maturity of different fruits, placing individual fruit on a calibration checkerboard and capturing a number of images to collate into a volumetric representation. Even as recently as 2022, Feldmann et al [36] reconstruct a range of fruits using a setup comprising a piece of fruit on a rotating platform against a well calibrated background. While these photogrammetry approaches have improved over time, they still struggle to overcome the challenges that are intrinsic to their multi-view design.

5.4.2 Monocular

As discussed before, many approaches to fruit reconstruction involves multiple viewing angles, with much of the challenge being involved in the composition of the data from these images into a volume. Looking more broadly at 3D reconstruction research, most of the 2D approaches that are more common are preferred in part because they can be performed with a single camera overlooking a single known background which can be a point of consistency or even be used for calibration.

The 3D Reconstruction method used in this project was originally presented by Jackson et al in two papers, one on human face [52] and another on human body 3D reconstruction [53]. These papers present *Volumetric Regression Networks* as a method for performing accurate reconstruction in an end-to-end fashion, and the approach was more flexible than other popular approaches. Similar to depth-map regression, they model the problem as a mapping of a 2D plane to a 3D volume, including an estimation of the occluded reverse side of the object. These works use a number of different architecture designs based on encoder-decoder CNNs for semantic segmentation, most notably the skip connections used for reconstructing spacial resolution from UNet [92] and the stacked hourglass structure presented in Newell et al [77].

5.5 Materials and Methods

In this section we describe our approach to the problem of generating 3D volumes via unsupervised domain adaptation; in particular, how we crafted our datasets and designed the architecture of our model. In addition we describe the experiments we conducted in order to test the efficacy of our proposed architecture.

5.5.1 Training Dataset

To train our model we utilised two different datasets. The first is a collection of 25,000 images of synthetic bananas created in Blender [15], and the second is an image dataset of real bananas of a similar size. Together these datasets represent our labelled source dataset and our unlabelled target dataset for unsupervised domain adaptation.

For our synthetic dataset we capture image data by rendering images of over 5 *master* 3D banana models from freely-available online sources [1]. Each model was chosen for its perceived realism, with more importance given to 3D geometry than to texture (as texture will be modified by style transfer during domain adaptation), with some consideration given to adding as much variety as possible into the dataset. Examples of these can be seen in figure 5.1.

In Blender these 5 models were then modified by scaling randomly along each axis to between 60 and 100 percent of their original size, followed by random in-plane rotation to create 5000 variations of each. We used the original provided textures for all captures of each master Banana, however we adjusted the brightness of the light source between 0.5 and 1.5 times our default value, as well as adjusting some values of specular reflection to increase image variety. Renderings were captured of the augmented models, along with the random transformation parameters used. The corresponding meshes were then used to create 3D volumes under the same transformations, and were saved into an HDF5 file [106] for input into PyTorch. For each rendering, a randomly selected image from the Common Objects in Context (COCO) dataset [64] was used as a background

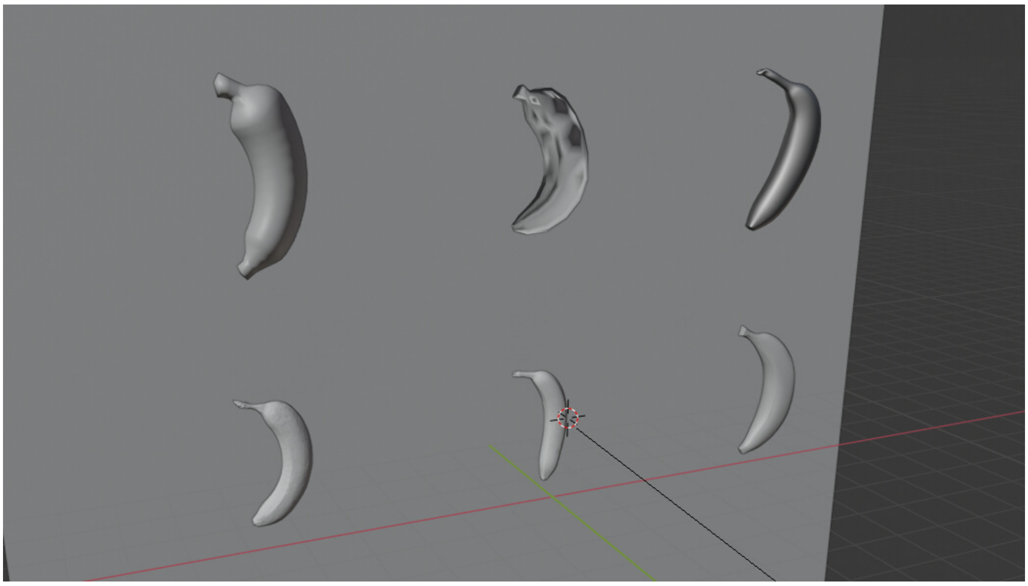


Figure 5.1: Image shown being captured in Blender. Here we see examples of six 3D banana models used to create our synthetic dataset (the top left model was later discarded due to lack of realism causing poor domain adaptation). As described above, these models were chosen to represent a diverse selection of different possible shapes of Banana and thus provide a good base for domain adaptation.

image, increasing variety in the training set and encouraging the generator to ignore the background. Augmentation and rendering was performed automatically in Blender, with volumetric ground truth produced in python.

Our second dataset, consisting of real images, is a collection drawn from three sources. First, images were taken from a dataset [70] originally used for ripeness classification networks. Second, the *Top Indian Fruits* dataset, contains many images of bananas in various states of ripeness and health [82]; from this we selected only the examples of healthy bananas and discarded the associated per-image ripeness and quality labels. Finally, we collected additional images ourselves, allowing us to add images with more variations in lighting and angle. To further increase the variety in our dataset, these images were also augmented with scaling, flips and rotations to generate $25,000$ different examples.

While overall our synthetic dataset was quite time consuming to create, (especially including the voxelisation process described below) we note that the cre-

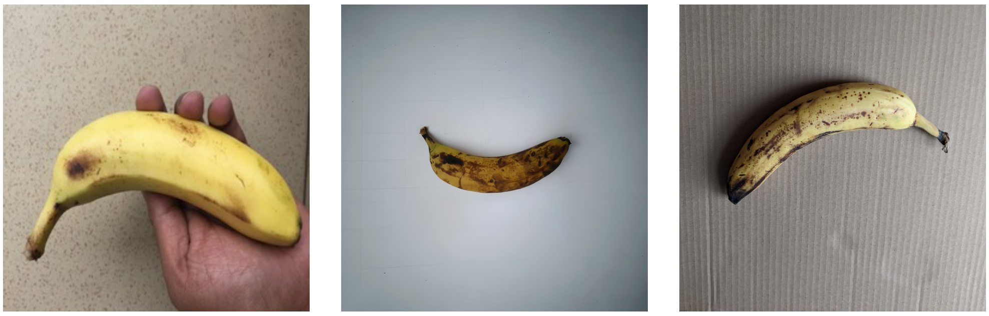


Figure 5.2: Examples of real images taken from Top Indian Fruit [82], Fayoum's ripeness dataset [70], and our own dataset of real bananas.

ation of real 3D datasets of image-label pairs are incredibly expensive and almost exclusively reserved for high interest subjects such as people and vehicles. While we are able to produce thousands of training instances with a single 3D model by deforming the model, creating a real dataset would have taken thousands of real bananas, making it practically infeasible. Additionally, to reuse our pipeline for other objects we would only need to collect or commission another handful of 3D models of the intended subject, whereas for any new subject matter an entire new set of real objects would need to be procured. As a result it is likely that potentially hundreds of hours could be saved from dataset creation alone by using our method, especially in the case of larger objects that would take significantly longer to capture with 3D scanning hardware.

5.5.2 Voxelisation Procedure

The rendering process in Blender saves the applied rotation and projection matrix with each banana rendering. In order to bring the 3D model into alignment with the rendered image such that it may be accurately voxelised, we first apply the rotation transformation, followed by projection transformation. Doing this will ensure that the voxel representation for each banana will align with the image.

The projection matrix destroys depth information in the Z axis with respect to the image plane, meaning we need to recalculate the distance of the banana from the camera so we can position the voxel representation correctly in 3D space. We

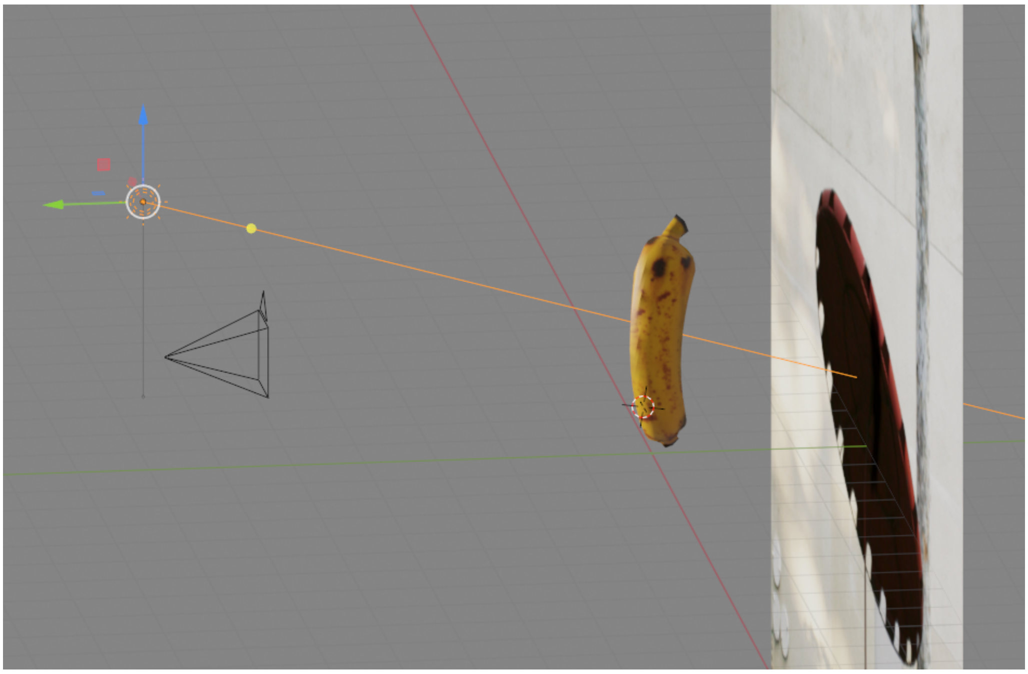


Figure 5.3: Image shown being captured in Blender. Here we see a 3D model of a banana being captured against a background of a random coco image having been transformed by translations, scale and rotations. A light source positioned behind the camera is directed at the banana adding realistic lighting to the rendered image.

recover this by using the standard deviation of the 2D axes, before and after the projection step, as a scaling factor for the Z axis. The standard deviation of x and y is used because it is invariant to any translation which may have been applied during projection. More concretely, where M and M_{proj} are the unprojected and projected meshes respectively, of x, y, z coordinates,

$$M_{proj,z} = \frac{M_z}{2} \left(\frac{\mathrm{std}(M_{proj,x})}{\mathrm{std}(M_x)} + \frac{\mathrm{std}(M_{proj,y})}{\mathrm{std}(M_y)} \right). \tag{5.1}$$

Voxelisation is performed by tracing rays through each plane, x, y and z to produce three intermediate volumes. These are combined into a single 3D volume by finding all voxels that intersect at least two of the intermediate volumes. This approach reduces artefacts but is slightly slower than performing voxelisation from a single plane[1]. Our final volumes have a resolution of 256 × 256 × 128. Higher depth resolution is unnecessary in this problem domain due to bananas generally being relatively flat in a one dimension, assuming they are laid on a flat surface.

[1] We use Adam Aitkenhead's implementation, available here http://uk.mathworks.com/matlabcentral/fileexchange/27390-mesh-voxelisation

5.5.3 Volumetric Regression Network

For our 3D reconstruction we use a Volumetric Regression Network to map our 2D images to 3D volumes. We evaluated a number of models for this task, including U-Net [92] and stacked hourglass models shown in [53] by using three small sections of our synthetic dataset as a training, validation and test sets respectively, and testing the ability of each network to perform volumetric regression on the simpler supervised learning task without the domain shift problem. Testing showed that a modified U-Net implementation achieved the best performance in 3D reconstruction, while using fewer parameters than the stacked hourglass. We use standard spatial convolutions throughout the network, and reconfigure the U-Net to use three down-sampling layers followed by three up-sampling layers, with skip connections between layers of the same spacial resolution. Comparing this loss against the true volume of the synthetic image gives us our Volumetric Loss, for which we selected a Binary Cross Entropy (BCE) implementation as for each voxel in 3D space the a value of either zero or one must be predicted by the network.

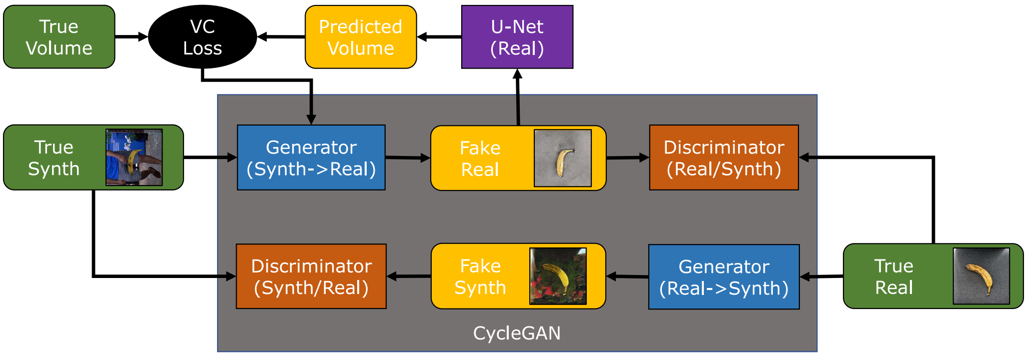


Figure 5.4: The proposed Volumetrically Consistent CycleGAN (VCC) using our *real* Banana dataset as a target.

5.5.4 Volumetrically Consistent CycleGAN

It has been shown that CNNs trained on purely synthetic data do not generalise well onto real images [118], however large performance increases can be achieved by including a small fraction of real training data [118]. Unsupervised domain adaptation approaches take this one step further, requiring only labels for our synthetic set; indeed our approach implements this idea by requiring only labeled synthetic data supplemented with different datasets of *unlabeled* real photographs as input. Our goal is to train our end-to-end network to produce a 3D reconstruction of objects from the images in the *real* domain. We extend a CycleGAN implementation [123] with our VRN in a similar fashion as that shown in chapter 4. Our model, shown in Figure 5.4 performs unpaired image-to-image translation between *real* and *synthetic* images.

Our novel addition here is our combination of our VRN that performs 3D reconstruction on the output of the *synthetic-to-real* generator with CycleGAN as a domain adaptation network. While the structure of CycleGAN remains the same, we extend the model with our Volumetric Regression Network such that the output is the *synthetic to real* generator is input to the VRN. As a result of this architecture the Volumetric loss is applied first to the generator, which ensures that the 3D structure of the object is preserved when changing the domain of the image and additionally the U-Net. This *Volumetric Consistency Loss* (VC loss) loss is given a weighting of 1.0, relative to all CycleGAN weights which were determined by the original authors of CycleGAN. This value was determined empirically, though further fine tuning may improve time taken for convergence.

Due to the number of parameters of a high resolution CycleGAN model running concurrently with our 3D VRN on the same GPU, our model faces significant challenges in regards to VRAM. Because of the large number of parameters and size of the data, we took a number of steps to streamline our model. Firstly, as mentioned above this was one of the considerations that caused us to choose a U-Net model rather than a stacked hourglass model that would have required a much larger number of parameters. Secondly, we reduced the number of layers in our U-Net from 4 down to 3, reducing the number of parameters and memory usage. Finally we reduced the resolution in the z dimension (depth) from 256 to 128, removing any training examples that would now intersect the edge of the bounded space. We make the assumption that for all reasonable test cases, the bananas will be lying flat and thus likely to be satisfied by this constraint due to their geometry.

5.5.5 Experiments

(1) VRN Trained with Synthetic Data Only. Here we establish a baseline in terms of performance i.e. what level of performance we can achieve on real images when trained only on synthetic renders. Synthetic images have been successfully leveraged in many domains, but the domain gap between synthetic and real images often leads to poor generalisation.

(2) VRN Trained on CycleGAN Images. We evaluate the performance of the VRN on real images, when synthetic training images have first been *refined* to look more realistic. CycleGAN is trained to translate the synthetic images into the target domain of real images which are then used to train our VRN as carried out in experiment 1.

(3) GANana VRN. GANana combines the VRN and CycleGAN in a single model, shown in figure 5.4, that can be trained end to end. Images are refined by the CycleGAN at the same time as our VRN is trained to extract a 3D volume. The approach taken by GANana ensures that refined images preserve the high level structural features necessary for volumetric reconstruction while simultaneously closing the domain gap between the two sets of images.

(4) GANana VRN using PASCAL VOC. In this experiment we use the same architecture from experiment 3, but replace our *real* banana dataset described in Section 5.5.1 with unlabelled images from the PASCAL VOC dataset. We hypothesised that a larger and more diverse range of images from the *real* domain may compensate for using images that do not match the particular subject of our source domain; and if so, reduce the need to build a domain-specific dataset.

(5) Ganana VRN using Gaussian Noise. For this negative test we replace our target domain dataset with random noise. We hypothesise that this will force our generator to transform our image almost entirely into noise, maintaining only the high level features needed to regress the banana. By excluding images from the target domain we prevent the model from performing domain adaptation, and any improvement on our baseline score can be attributed to augmentation. Unlike our previous experiments, in this example, losses from the VRN and CycleGAN will, we hypothesise, be sufficiently opposed to each other such that it will be impossible to produce good results.

(6) Ganana VRN using Synthetic Target. In our final experiments we train on pairs of *identical* images from our synthetic dataset as both the source and target domain. By keeping the source and target domain the same, CycleGAN is no longer encouraged to transform input images, as any transformation made by the generator can only make each image differ from the target. Instead we hypothesise that it will apply subtle augmentations to each image, improving robustness of our VRN while being prevented from significantly altering the high level features of each image. Increased variability of the input data means the VRN in our model must be more resilient to augmentations produced by the generator, which may enable it to perform well on images in our target domain. In this sense we can consider the goals of our the CycleGAN and VRN to be better aligned, which we believe will improve performance.

5.5.6 Testing Dataset

In order to test our method, we built our own test dataset comprising 15 real banana models with associated 3D ground truth. Images were captured using the photogrammetry app Qlone, run on an Android phone [35]. For each model, a banana was placed on a calibration base and images were captured from numerous angles. The banana was then flipped onto a different side and the process was repeated to improve accuracy on the unseen surface. Figure 5.5 shows this process. The app combines the two meshes to generate a single 3D model of the banana for import into Blender, where any elements remaining errors such as reconstructed background could be removed manually. The process described in Section 5.5.2 was used to convert each model into a volume for use as ground

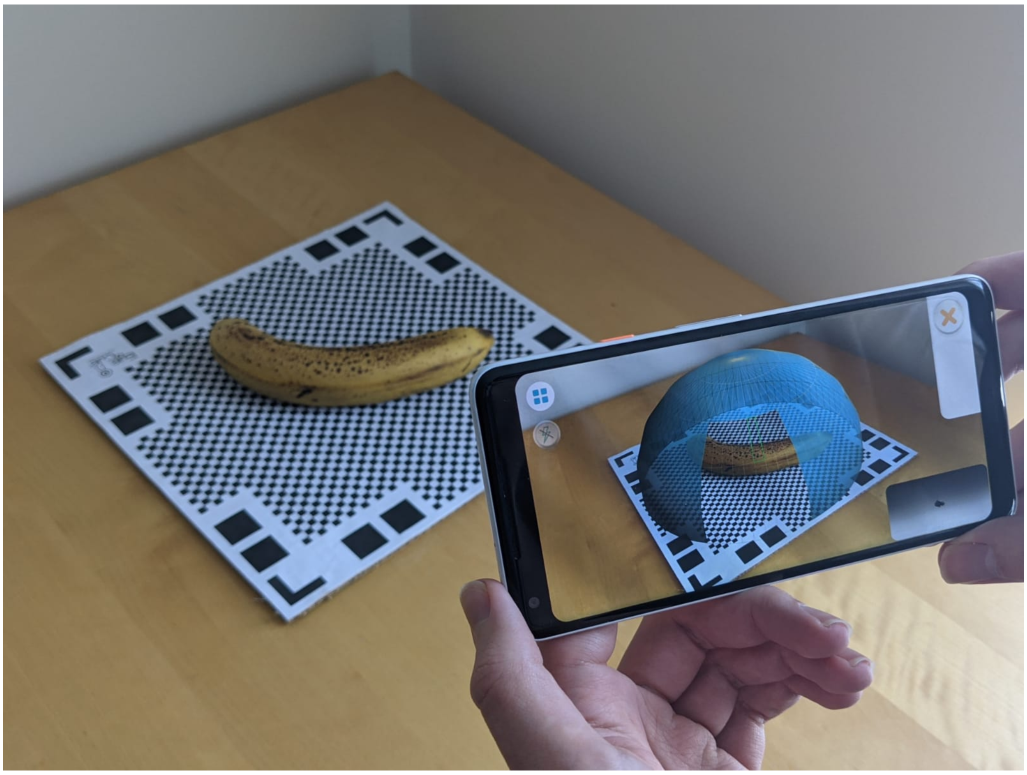


Figure 5.5: Demonstration of capturing instances for our test dataset through the Qlone photogrammetry mobile app. Bananas placed on the calibration mat would be captured from all angles, before being flipped and the reverse side then captured using the same method.

truth. Finally each model was paired with a single top-down image of the banana it was generated from, which would then make up each test image-volume pair. Each example took an average of 15 minutes to successfully capture, demonstrating the difficulty in feasibly collecting enough samples to create a suitable size dataset for training a VRN with real image-volume pairs, as has been demonstrated in previous works [53].

5.5.7 Training

Our network was trained in an end-to-end fashion using Adam optimizer [58], a learning rate of 2e$-$4 and default parameters for all CycleGAN models used in the architecture. We trained the model using a batch size of eight, and trained on eight NVIDIA Titan X (Pascal) graphics cards for 10 epochs until the model converged. Due to the large size of volumetric tensors, significant bottlenecks do occur in our training pipeline as a result of data loading, and more complex transforms. With this in mind we perform limited online augmentations to both images and volumes, including flips and 90 and 180 degree rotations, as these can be performed with the least computational cost, while also ensuring our network generalises well onto a wide range of test image examples. In order to decrease training time when loading our training data, we saved our dataset in HDF5 format, allowing it to be directly loaded as a PyTorch Tensor.

5.6 Results

Here we present the results of the experiments conducted to evaluate the effectiveness of the model described in Section 5.5.

5.6.1 Qualitative Results

We show the input with corresponding output, from the four experiments, in Figures 5.6 and 5.8. VRN trained on only synthetic images (experiment 1) fails almost completely when presented with a real image. GANana succeeds in experiments, (3), (4) and (6), with only the addition of unlabelled target images. The background images in Figure 5.6 (b) and (c) are from the original images, but in (d), (e), (f) and (g) the 2D image output from the *synth-to-real* generator component is used as a background, which gives an idea of how the generator transforms input images depending on the target dataset used in each experiment. These images demonstrate that the volumetric consistency prevents distortions to the original object's shape, and that the main difference from the transformation appears to be colour tone.

In Figure 5.7 we show the output of the generator both *with* (c) and (d), and *without* (b), the proposed volumetric consistency loss. CycleGAN is known to have a number of failure cases, especially where the two training domains are not sufficiently similar [123] and we see an example of this in experiment 2. Without the volumetric consistency loss, the model degenerates to creating very similar images that do not retain their structure, hardly resembling a banana at all; and as such we have not included it in our numerical results in Table 5.1. This fail state is consistent with what is observed in [75], where CycleGAN is unable to preserve geometry when transforming an image from the synthetic to real domain, and Muller et al are able to improve augmentation by using a 2D segmentation network to provide a support loss in order to generate images for hand tracking. We speculate that 3D renders and photographs of real bananas are not sufficiently similar for CycleGAN to produce good results; it is a strength of our model that it performs well even where these higher-level differences between our two datasets exist. As evidence of this we observe the GANana-enhanced images exhibit contrast and brightness changes that better match images from the target domain (observed in figure 5.7). As such CycleGANs learned transformations are more pronounced on images which differ more significantly from those in the target set, while appearing less extreme on more similar images as we observe in figure 5.7.

5.6.2 Quantitative Results

For each experiment, we compute both Volumetric Intersection over Union (VIoU), as well as Root Square Mean Error (RSME). To compute both metrics, we accounted for scale using the length, width and depth of each banana, before applying the Iterative Closest Point [12] (ICP) algorithm. This procedure was repeated three times for each sample, which we found to produce adequate alignment to obtain the best mapping between reconstruction and ground truth and avoid simple translation and rotation errors. ICP was needed as the scans produced by the Qlone app were scaled differently to the predicted 3D volume and not aligned with the individual photo. Our results are therefore presented after ICP alignment. This may bias the performance slightly, but the same procedure was used for all experiments for consistency.

Table 5.1: Volumetric IoU and RMSE reported on our collected dataset of real bananas.

Method	**2D IoU**	**VIoU**	**RMSE**
(1) VRN (Synth. Training)	41.74%	17.52%	7.59
(3) GANana (Bananas)	**92.36**%	76.37%	1.68
(4) GANana (VOC)	91.88%	**76.60**%	**1.65**
(5) GANana (Noise)	44.65%	33.04%	7.64
(6) GANana (Synth to Synth)	92.29%	73.62%	2.07

We present our numerical results in Table 5.1. The baseline VRN trained with synthetic data (1) performed very poorly on real images. This is likely due to the domain gap between real and synthetic images causing poor generalization between images which may on first impressions appear visually similar. Conversely, in experiments 3 and 4, using our volumetrically-consistent GAN, we are able to improve performance substantially, and both experiments achieve our highest VIoU scores. As predicted, experiment 5 shows a marked decrease in performance compared to our other experiments using our architecture especially in 2D IoU, but still outperforms our baseline score, despite images being almost completely indistinguishable from noise. Experiment 6, however, performs well and has scores that are comparable to experiments 3 and 4 and significantly above the baseline. This is an interesting and significant result, as these scores are achieved when testing on real images despite being trained with only synthetic images, and does not require a dataset of even general real images as a target.

5.6.3 Segmentation

Here we demonstrate that our method is capable of performing 2D segmentation. By taking the sum of the produced volumes through the Z axis, we predict a segmentation mask, to enable comparison with a silhouette from the source 2D image. We hand annotated foreground and background pixels for images in our testing dataset. In the second column of Table 5.1 we show Intersection over Union score, demonstrating that our method is also effective at training for 2D segmentation through domain adaptation, as well as measuring shape error as viewed from directly above.

5.6.4 Method Performance

Methods working with volumetric structures have a reputation for either being slow or inefficient. The volumes themselves are often large and can be difficult to work with. However, binary volumes which have large contiguous blocks of data (such as ours), are highly compressible. Our $256 \times 256 \times 128$ volumes are stored as one byte per voxel, thus requiring 8MB of memory per volume. However, on disk with LZ4 compression, they consume only 70KB to 90KB with minimal computational cost. Our architecture contains no 3D (Volumetric) convolutions, and instead uses only 2D (Spatial) convolutions, which are highly optimised to run on the GPU. Inference through our model takes 253ms on a single NVIDIA Titan X (Pascal). This is then followed by an additional 124ms to extract the surface from this volume (allocated a single core on an Intel Xeon E5-2698 v3 system, average for 1000 runs).

We believe that for many applications, the ability to create a 3D model in under 400ms from a single image is practical. Further improvements are likely possible to our pipeline, such as improved architecture though will require greater of computer resources to successfully implement.

5.7 Discussion

In this section we present analysis of our results and the effectiveness of our methodology as demonstrated by our experimentation. We also talk about some possible limitations of our method and possible improvements or additions that could be made to our pipeline.

5.7.1 Analysis of results

The difference in performance between experiments 3 and 4 are sufficiently small such that both results can be considered practically equivalent, and in both cases show good performance in both 2D and 3D scenarios. It is interesting to note that domain-specific target images are not shown to lead to substantial performance increases, and shows the broader applicability of our architecture into other domains, such as medical scans, or street scenes. Experiment 3 produces the highest score for 2D IoU, suggesting that using real bananas as our target domain encourages the generator to best maintain the outline of the banana during transformation. Experiment 4 shows that targeting PASCAL VOC with the GAN achieves comparable results in terms of VIoU and RMSE, compared to using our *real* banana dataset. This is significant as it demonstrates that our method is effective even if large datasets of the particular subject matter are unavailable. We believe this demonstrates our method's potential to work in other domains.

In experiment 5 the VRN is still able to extract a reasonable likeness to the true volume, suggesting that structural information must still exist in the noise images in order allow reconstruction of the volume via the VRN. In figure 5.8 however, we see that as the VRN is trained on images which have been passed through the CycleGAN with noise as a target, the loss of the VRN encourages preservation of high level features that enable the regression of the volumetric structure.

Experiment 6 performs well given it is trained exclusively on synthetic images, however we believe that the performance benefit obtained by using a real target dataset as seen in experiments 3 and 4 is worth the small additional cost of curating a selection of real images, particularly when they can be easily sampled from existing datasets. Aside from transforming images from one domain to another, it is conceivable that CycleGAN is simply performing image augmentation, thus treating the task as a domain generalisation problem and forcing our VRN to be robust to variation. The fact that experiment 6 performs comparably to experiments 3 and 4 without using images from the target domain would support this hypothesis.

5.7.2 Limitations and Failure States

In Figure 5.6, we see that in our results for experiment 5 there are a number of fail cases, where background pixels are interpreted as part of the volume by the VRN, and this leads to even poorer performance for 2D results shown in figure 5.8. These kinds of false positive results are not observed in other experiments or even the baseline; we hypothesise that this is caused by the *noise* target domain having no distinction between foreground and background pixels for the network to learn.

Although in our other experiments our Ganana VRN models performed well on our test dataset, it is likely our approach has limitations in its effectiveness that may lead to failure states. Because our training is based on automatically generated synthetic data it makes it more likely that failure states will emerge when images sufficiently different from the training set are tested. An example observed during testing was a failure state when the banana is not well centered in the frame, as they are in our synthetic models.

Similarly, although controlled-light phenotyping tasks are common, in other phenotyping tasks it is possible for extreme lighting situations to present a challenge. However, while our testing was carried out with controlled lighting, it is likely our networks will be more resilient to these kinds of changes as CycleGAN has the potential to improve the VRN's ability to generalise onto more varied images.

We also note that in the context of detecting defects in fruit, imperfections could appear on the reverse side of a piece of fruit. As such it is likely that in this context the model would have to be applied to each side of the fruit being scanned, though this would still be preferable to photogrammetry which requires many more images as well as calibration, or LIDAR which would be more expensive.

5.7.3 Future Work

In our work we focus on accurate 3D reconstruction of fruit using a methodology that has not yet been applied in a real-world phenotyping pipeline; as such we do not go as far as to calibrate our images to real-world units. In a future extension of this work, we believe that by capturing images using a well-calibrated capture environment, it would be possible to estimate both volume and mass of fruit using an extension of our proposed setup. The flexibility shown by our model would allow it to be used for a wide range of tasks in an agricultural setting, such as the capture of entire plants, as well as downstream tasks like monitoring growth.

5.8 Conclusion

We have presented our methodology for using a VRN trained on augmented synthetic data to address the problem of estimating accurate 3D models from a single view. These models, trained on a fruit dataset, provide detailed 3D reconstructions of the target object, ideally suited to downstream phenotyping tasks. Our results are obtained with a smaller data and annotation cost than conventional deep learning models by approaching the task as an unsupervised domain adaptation problem. As such our approach provides full reconstruction of the target object without the need for any manually annotated real-world images. We introduce a Volumetrically Consistent CycleGAN, in which a CycleGAN is used to transform an image from a labelled synthetic domain into an unlabelled real domain, while a volumetric regression network learns to reconstruct objects models in 3D. These networks are trained end-to-end, improving performance over a modular design. We have shown a significant improvement in volumetric segmentation scores and RMSE versus alternative approaches. Our approach performs well against ground truth generated using multi-image photogrammetry software, and demonstrates our model's ability to generate accurate reconstructions.

This accurate reconstruction of 3D models of plants is important in the push for automated size and quality control, as well as other phenotyping tasks such as informing biological modelling applications. Common hardware-based techniques such as LiDAR are costly and time consuming, unsuitable for very high-throughput pipelines. Our method is fast (<0.5 secs per image), accurate, requires no human interaction once trained, and works using a single RGB camera. We expect that the method concept will generalise to a wide range of other objects, including other fruit, vegetables and plant organs such as leaves. We provide code for our Volumetrically Consistent CycleGAN architecture at https://github.com/zanehartley/Ganana_Unsupervised_Domain_Adaptation_For_3D_Reconstruction that will enable future researchers to make use of the pipeline we have presented in this chapter. To apply this technique to new domains, the production of appropriate synthetic models is required, combined with sample images from the real domain. Our software pipeline, dataset and network will be made available online, to facilitate researchers training 3D reconstruction models in a variety of domains.

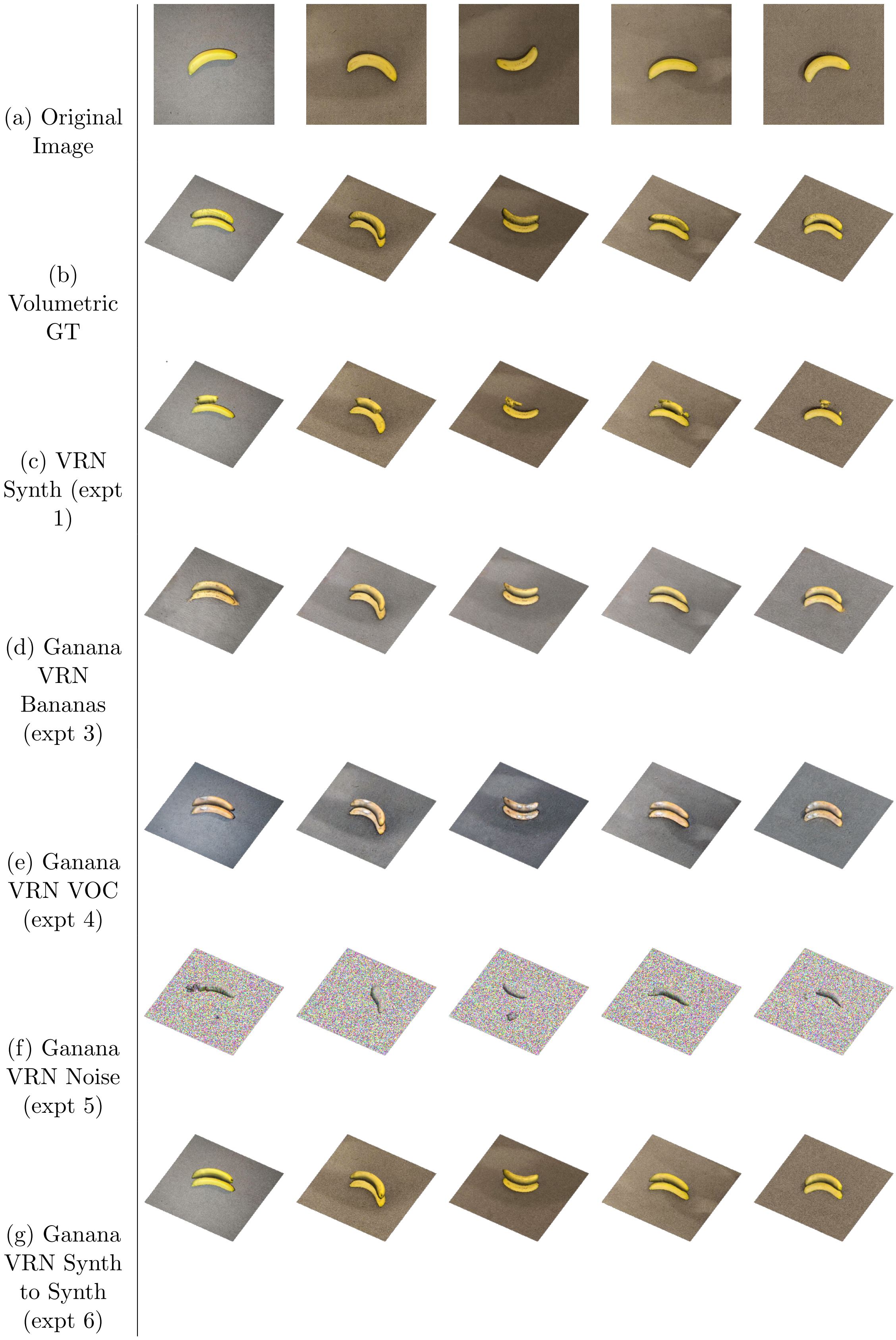

(a) Original Image
(b) Volumetric GT
(c) VRN Synth (expt 1)
(d) Ganana VRN Bananas (expt 3)
(e) Ganana VRN VOC (expt 4)
(f) Ganana VRN Noise (expt 5)
(g) Ganana VRN Synth to Synth (expt 6)

Figure 5.6: Example outputs of our experiments for Volumetric Regression showing ground truths (b), Synthetic training (c), and then our Ganana models with different target datasets (d), (e), (f), and (g). Each volume is shown as a 3D model hovering above the image it was extracted from.

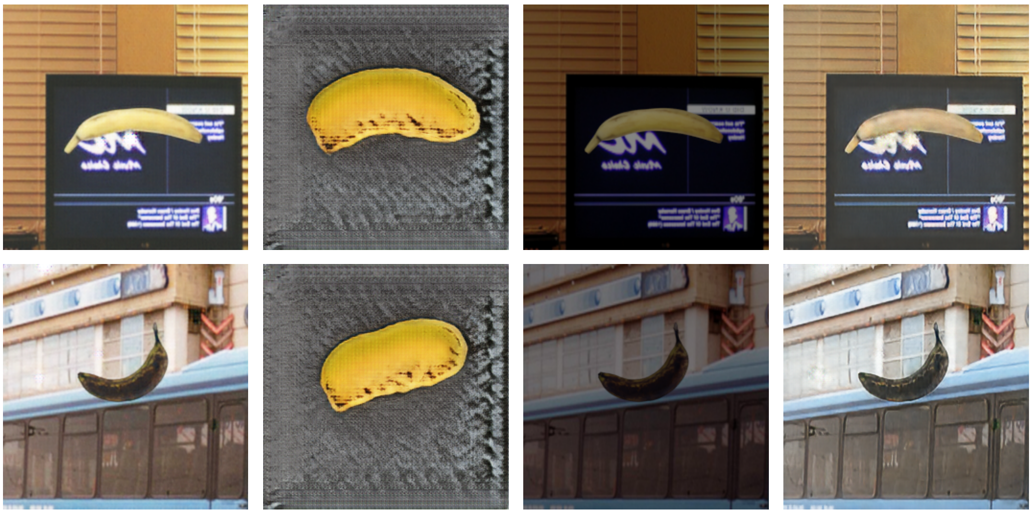

(a) Input (b) CycleGAN Enhanced (c) GANana Enhanced (d) Synth2Synth Enhanced

Figure 5.7: Output from *Synthetic to Real* generator from standard Cyclegan (b), Ganana with volumetric support (c) and CycleGan with Synthetic as Target (d).

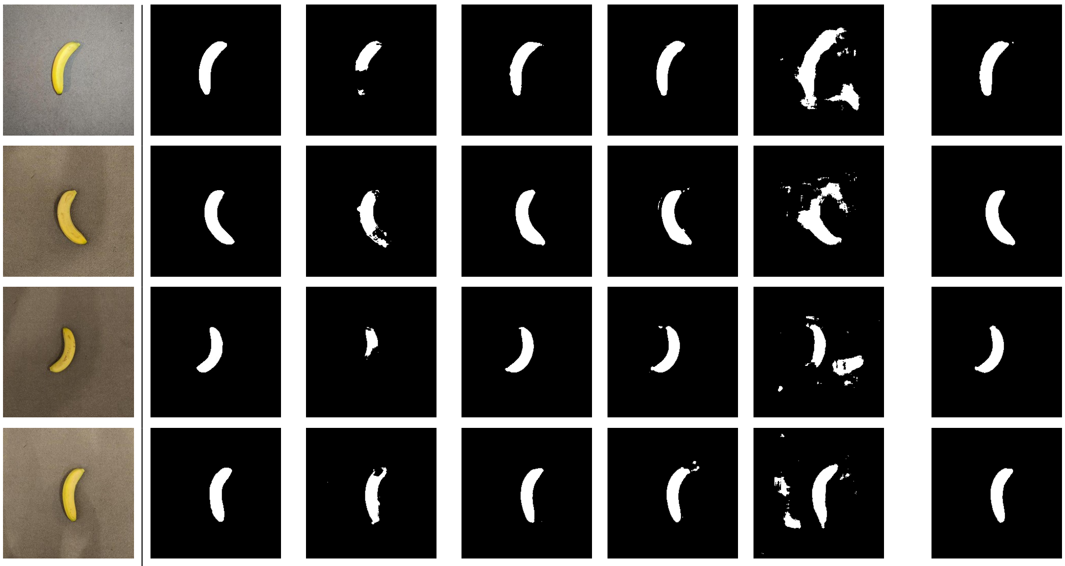

(a) Input (b) GT (c) VRN Synth (expt 1) (d) Bananas (expt 3) (e) VOC (expt 4) (f) Noise (expt 5) (g) Synth to Synth (expt 6)

Figure 5.8: Outputs of our experiments for 2D Segmentation showing ground truth segmentation masks (b), Synthetic training (c) and our Ganana models (d), (e), (f), and (g).

Chapter 6

Investigating the impact of synthetic data on Transformer and CNN models.

In the previous two chapters this thesis has focused on applying Synthetic data to plant phenotyping problems using state of the art Convolution Neural Networks. Our own work makes novel contributions to an emerging field championing Synthetic data for training CNNs, including new methods and applications for this approach. In this chapter we directly continue our work from chapter 4, this time looking at the impact of synthetic and style transfered data in the context of the emerging field of Transformer neural network architectures.

6.1 Introduction

Transformers are a new type of machine learning architecture first introduced for natural language processing. Very recently they have become a point of interest for the Computer Vision community with *attention* mechanisms being applied to images allowing networks to make decisions with global context, compared to the local approach of CNNs. Today transformers are a fast moving area of research, with an open debate about the future role of transformers and CNNs in Computer Vision research. In this chapter we aim to apply the domain adaptation of synthetic data approach to image transformers, comparing their impact with the current state of the art in CNN object detection models.

6.1.1 Motivation and Aims

As discussed in some detail in section 3.1.2 Transformer models have shown themselves to be capable of state of the art performance when applied to some of the most challenging computer vision tasks. In order to achieve many of their most impressive results Transformers have proven to be extremely data hungry, even by machine learning standards. Indeed in [31] ViT is trained using JFT-300M [105] a private dataset of over 300 million images and over a billion individual labels. Since datasets of this size are not widely available, reproducing these state of the art results is not easily possible, and engineers hoping to apply these new models to a similar level of success need access to extremely large new datasets. In light of this, our approach of using synthetic data augmented using a synthetic-to-real style change network is especially relevant to this new field of research.

Furthermore in this chapter we continue our work from chapter 4 on wheat head detection. Transformers have been proposed as a better way to design networks for challenging computer vision tasks, object detection being on of the most prominently targeted. We are motivated therefore to instigate the effectiveness of new Transformer models on a core plant phenotyping problem such as wheat head counting.

6.2 Background

We covered Transformers in some detail in section 3.1.2, but will give a short recap here as well as some more specific related work. Transformers were originally introduced by Vaswani et al [112] in 2017 as a type of language model used for Natural Language Programming, introducing the core concept of a global Attention mechanism. Attention blocks allow the network to learn the strength and importance of the relationships between different parts of their input, be that language or images. Multiple *attention heads* (parallel attention mechanisms

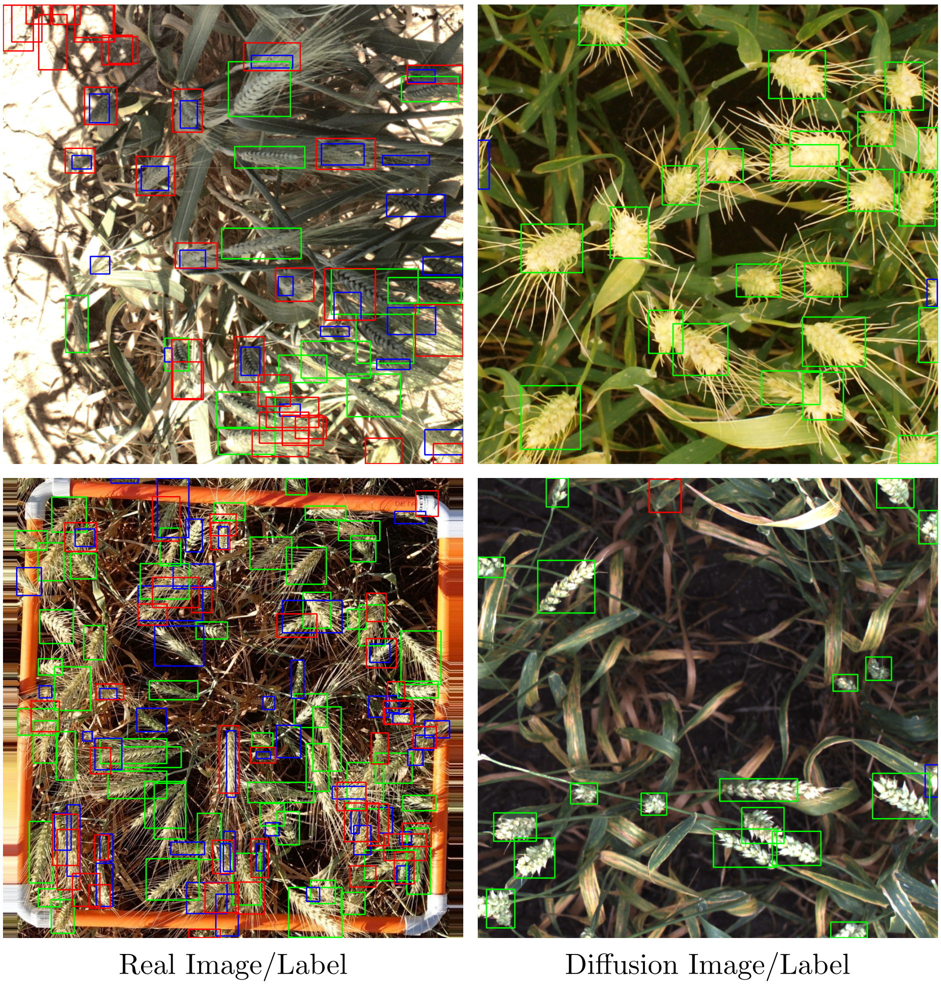


Figure 6.1: Examples detections from the GWHD, showing true positives in green, false positives in blue, and false negatives are shown in red.

that allow for multiple relations to be captured allow many different relations to be captured simultaneously, and by stacking many attention blocks it allows for complex relations to be considered. This approach also emphasises non-local features, as attention is captured on a global scale [31].

Despite some earlier non-local CNN models such as that presented by Wang et al[115], the Visual Tranformer was first introduced in 2020 by Dosovitskiy et al [31] which replaced the word embeddings from language models with patch embeddings that allowed the models to be redesigned for images.

DETR or the *detection transformer* is an object detection transformer introduced in 2020 by Carion et al [17], adapting earlier models such as ViT to predict bounding boxes for classes of interest within an image. DETR makes use of the *output* *sequence* from earlier transformers as predictions for a series of predicted objects, labeled with a class and position. Since the network cannot know beforehand how many instances will exist in a given image, the sequence length will be an upper bound of the number of objects we might expect to predict. For each object detected a token from the output sequence will be given its class and location, and for unneeded tokens a null class is assigned to them to show they do not refer to an object.

6.3 Material and Methods

In this section we discuss our choices of networks used for our experiments, including both our choice of Transformer architecture, and state of the art CNN models to compare it against. We then also describe improvements made to our synthetic dataset over that demonstrated in chapter 4, along with our decision to simplify our domain adaptation approach for this chapter.

6.3.1 Models Selected

In order to best evaluate Transformers in the context of both synthetic data, and plant phenotyping, we select the task of object detection to base our experiments on. Object detection is one of the most popular tasks in the digital plant phenotyping space, with a range of different applications, from the detection of individual plant components, up to segmenting crops from aerial photography. For our analyses we select one popular detection Transformer along with two different state-of-the-art CNN models with different advantages and disadvantages. We justify our selection of each model below.

YOLOv5 is the most widely used version of the YOLO object detection family [33]. Using a system of anchors to predict all bounding boxes for an image in a single pass, YOLO models are considered benchmarks for both accuracy as well as speed at test time. Many systems that require real time detection use YOLO networks for this reason. YOLO is generally considered to be the best architecture for dealing with detection of occluded objects, which is a common occurrence on the wheat head counting problem.

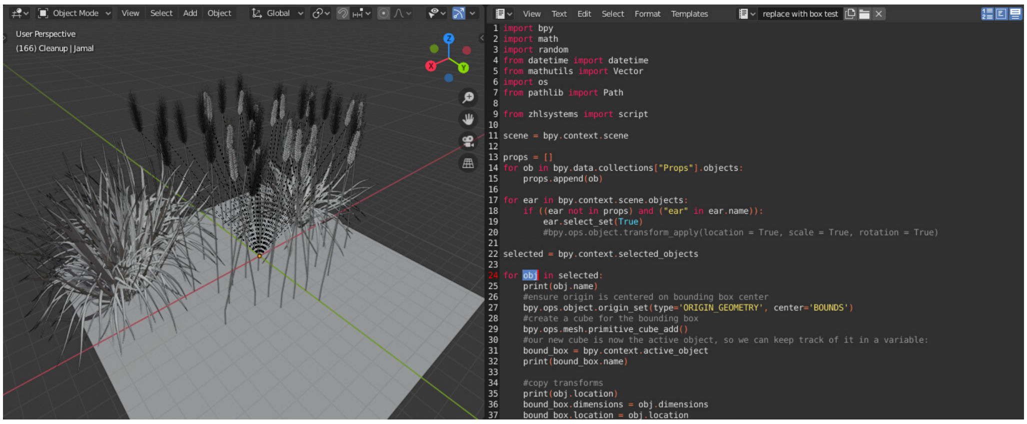
Faster-RCNN is a detection model from the RCNN family of architectures [89]. Relying on a region proposal network to predict boxes, and is one of the most widely used detection models due to its implementation in Facebook's Detectron 2 deep learning package. RCNN based models are generally considered the most accurate models for object detection, especially for detecting smaller objects, as is the case for detecting wheat heads; however this is generally considered at the cost of efficiency, as the model is slower to run that YOLO models.

DETR is a visual transformer for object detection introduced in 2020 [17]. The models design attemps a more streamlined and intuitive approach to object detection. Rather than using anchors or region proposal networks, DETR directly predicts the locations of objects and assigns category labels to them simultaneously. During training a bipartite matching algorithm [59] is used to pair each prediction with its most likely label to enable training. DETR has demonstrated state-of-the-art performance on a number of object detection benchmarks, and its design makes it good for problems with a highly variable number of objects to detect, as is the case in our wheat head counting challenge.

6.3.2 Dataset Creation

We create a synthetic dataset for our experiments similar to those described in chapter 4. For this set of experiments we make a number of improvements to improve the quality of the data created, as well as using an off the shelf style transfer model called CUTnet (explained fulling in section 6.3.3) to create our domain shifted synthetic dataset. Below we list the changes made to our synthetic data generation pipeline from chapter 4, including justification for the changes made.

- **Background Images.** In order to improve the quality of our synthetic data, when rendering our image in Blender we introduce empty background



```
import bpy
import math
import random
from datetime import datetime
from mathutils import Vector
import os
from pathlib import Path

from zhlsystems import script

scene = bpy.context.scene

props = []
for ob in bpy.data.collections["Props"].objects:
    props.append(ob)

for ear in bpy.context.scene.objects:
    if ((ear not in props) and ("ear" in ear.name)):
        ear.select_set(True)
        #bpy.ops.object.transform_apply(location = True, scale = True, rotation = True)

selected = bpy.context.selected_objects

for obj in selected:
    print(obj.name)
    #ensure origin is centered on bounding box center
    bpy.ops.object.origin_set(type='ORIGIN_GEOMETRY', center='BOUNDS')
    #create a cube for the bounding box
    bpy.ops.mesh.primitive_cube_add()
    #our new cube is now the active object, so we can keep track of it in a variable:
    bound_box = bpy.context.active_object
    print(bound_box.name)

    #copy transforms
    print(obj.location)
    bound_box.dimensions = obj.dimensions
    bound_box.location = obj.location
```

Figure 6.2: Our upgraded wheat scene being captured in Blender. Here we see our addition of a second wheat head model as well as improved foliage. On the right hand side we show our use of Blenders advanced scripting engine, allowing us to generate scenes such as the one shown automatically.

images from the GWHD training set [24] to act as background plates, replacing the generic *ground* images used as background plates in our previous version of our model. Images from the training set were filtered for images that contain no wheat heads of their own, and the set was further filtered manually to remove images containing unwanted content or that were otherwise inappropriate for this use. This left us with a set of images containing no wheat heads but that represented a cross section of the backgrounds present in wheat images in the dataset. Since CycleGAN models are trained to leave the target domain unchanged during style-transfer, this should encourage the model to ignore the background and focus style transfer on the foreground content.

- **Wheat head diversity.** In order to add additional diversity we included an additional wheat head model in Blender, which was used at random alongside the original 3D model. The two models used were selected to represent ears that have both prominent and shorter awns (the spiky protuberances that extend from the ear). As the GWHD is sourced from a number of institutions, different subsets of the dataset have different lengths of awn, so we make this inclusion so as to be more Representative of the target dataset.

- **Improved bounding boxes.** During early testing, we observed the model's

> tendency to predict oversized boxes for wheat heads, which may have been caused by our Blender data. When capturing the bounding box data, the recorded dimensions include the smallest box that completely encapsulates the widest dimensions of our wheat ear models, often leading to oversized bounding boxes. To fix this inaccuracy, we adjusted the capturing mechanism for bounding boxes to only encapsulate the wheat head itself, and not the surrounding awns (sometimes alternatively called the beard), as this is how bounding boxes are applied in the GHWD and should lead to improved accuracy.

We continued to use the same L-system approach to generating wheat stems giving them lean and direction to emulate a wind direction as seen in chapter 4. The rest of our pipeline is the same as described in section 4.4.2, including rendering settings, which continue to capture all images in 1024x1024 resolution.

Using our adjusted design, we again generated over 10,000 images with bounding box information to create our training data. For the images generated, we created a uniform spread of images with a range of ten through sixty wheat heads, excluding any wheat heads that were generated out of frame.

6.3.3 CUTnet style transfer

Similar to previous models, we use a style transfer network to improve the appearance of our synthetic images. For this series of experiments we replace our own modified CycleGAN networks from chapter 4 with another popular model with a similar design idea. CUTnet[81] (contrastive unpaired translation net) is a model that expands on the CycleGAN approach, attempting to improve the consistency of information during style transfer. To this end the model introduces a patch wise contrastive learning mechanism shown in figure 6.3. This technique extends a regular CycleGAN by dividing the generator outputs into patches, where a random output patch from the transformed image is compared against a series of patches from the original image, including the patch from the corresponding position. Since the matching patch has the same content, we would expect it to be relatively the most similar to the output patch, compared to others from the input. The addition of a contrastive loss therefore aims to enforce a high similarity with the matching patch, while simultaneously aiming for a low similarity with all other patches. The result of this is that we can expect to see a more highly enforced preservation of content during style transfer, achieving a similar effect to that we achieved in chapter 4, without the need for an additional network, though figure 6.3 does still show evidence of the hallucination of addition wheat heads as we previously experienced, showing that this method is still imperfect. Since all experiments were conducted using the CUTnet enhanced datasets, references to Synthetic data in our experiments refers exclusively to these enhanced images, and not to the RAW synthetic images generated in Blender.

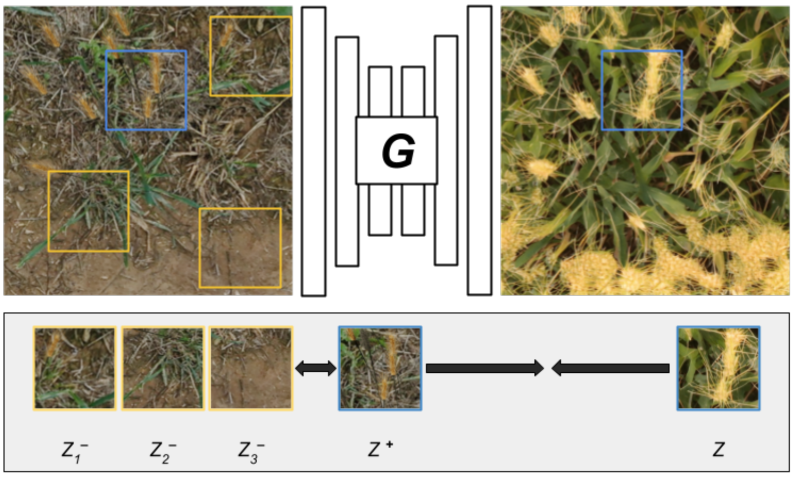


Figure 6.3: CUTnet improves upon CycleGAN by trying to minimise the distance between corresponding patches, while maximising the distance between the target patch Z and the negative patches.

6.4 Experiments

6.4.1 Comparing Transformers to State-of-the-art CNNs

For our experiments we initialise our three selected models, Faster RCNN, YOLOv5 and DETR, with their respective default hyper-parameter setups for our specific experiments. It is likely that each model could achieve better performance with hyper-parameter tuning, but we choose to leave each model in their default state wherever possible to minimise any bias caused by advantageous tuning. We then run a series of experiments to compare the impact of using a training regime using synthetic data on each of our models. We use a conventional training regime for incorporating synthetic data, using our large synthetic dataset for a pretraining phase, followed by a finetuning phase on our real data from the Global Wheat Head Dataset.

We compare scores between our synthetic enhanced training regime against a baseline using only real data using the recently released private test set, not available during the the work completed in chapter 4, which allows a more thorough evaluation. We calculate Average Precision 50 and 75, meaning that a 50 and 75 percent IoU is required by each prediction to be registered as a true positive, refered to as IoU_{50} and IoU_{75} respectively. We also calculate Mean Average Precision (MaP) where each subset of the test set has a score calculated separately and the mean of all subsets is taken accounting for the differences in size of each dataset.

6.4.2 Testing Transformers with small training sets

Subsequently we analyse the impact of synthetic data more deeply by running additional experiments where we limit the quantity of real data used in the fine tuning process. In many real world scenarios, very limited real data will be available for training neural networks. We model this here by running tests using the full 3290 real images, 1000 real images selected at random, and 100 real images also selected randomly. By doing this we hope to be able to better analyse the impact of synthetic data in cases where it plays a more significant role in the training of our networks.

For experiments using limited sets of real data, random subsets were created, with the smaller subsest being included within the larger subsets for consistency. For the second set of experiments, the same subsets were used to produce a *style-*

transferred train set with CUTnet, followed by finetuning with the same real images.

6.4.3 Evaluation Metrics

For our experiments we select three different metrics for evaluation, MaP, $\mathbf{AP}_{50}$ and $\mathbf{AP}_{75}$. MaP (which could also be considered MaP_{50}), is a metric that first finds the mean accuracy of each subset of the target dataset using IoU_{50}, and then calculates the average of these. By calculating the accuracy this way, equal weighting is given to each subset, meaning that larger subsets are weighted equally with smaller ones. This is especially important as we can observe that many of the most challenging subsets of the GHWD are substantially smaller than the larger, easier ones. A high MaP score would indicate that a network performs relatively well on these more challenging datasets. Our second metrics $\mathbf{AP}_{50}$ and $\mathbf{AP}_{75}$ instead calculate the average accuracy over all images in the dataset, regardless of subset. These two metrics differ by their IoU threshold, with $\mathbf{AP}_{50}$ requiring a 50 percent intersection to confirm a match between a prediction and ground truth and $\mathbf{AP}_{75}$ requiring 75 percent IoU to consider a match. As such we can expect the networks to perform much more poorly on the $\mathbf{AP}_{75}$, but their relative scores will give an indication of each network's ability to detect the boundaries of each wheat head with a high degree of accuracy.

6.5 Results and Discussion

Network	**Experiment**	**MaP**	$\mathbf{AP}_{50}$	$\mathbf{AP}_{75}$
Faster RCNN	(1) Real Only	0.504	0.624	0.204
	(2) Finetuning w/ CUTnet Data	0.459	0.620	0.312
YOLOv5	(3) Real Only	0.561	0.669	0.322
	(4) Finetuning w/ CUTnet Data	**0.570**	**0.675**	**0.323**
DETR	(5) Real Only	0.512	0.537	0.178
	(6) Finetuning w/ CUTnet Data	0.539	0.579	0.209

Table 6.1: Results of the experiments described in Section 6.4. Here we compare results of our method on three state of the art object detection models. Faster-RCNN and YOLOv5 are both CNN based architectures whereas DETR is a transformer based model.

In table 6.1 we show that under standard conditions, DETR is outperformed by both CNN based architectures when applied to the Global Wheat Challenge.

Network	Experiment	MaP	$\mathbf{AP}_{50}$	$\mathbf{AP}_{75}$
Real Images Only	(1) 100 Real Images	0.327	0.374	0.104
	(2) 1000 Real Images	0.428	0.486	0.167
	(3) 3290 Real Images	0.512	0.537	0.178
CUTnet with Real Target & Finetuning	(4) 100 Real Targets	0.375	0.412	0.116
	(5) 1000 Real Targets	0.487	0.516	1.170
	(6) 3290 Real Targets	**0.539**	**0.579**	**0.209**

Table 6.2: Results of the experiments described in Section 6.4 using limited quantities of real images on DETR. We compare the impact of our synthetic data in the case of training with limited quantities of real data to both perform our style transfer and to finetune our network.

However we do see that pre-training on Synthetic data has the most significant impact on the performance of DETR compared the other other networks, with Faster RCNN having no improvement at all from pretraining. We do note that our experiments with DETR have the best MaP score relative to their $\mathbf{AP}_{50}$ and $\mathbf{AP}_{75}$ scores, being only 0.025 points between MaP and $\mathbf{AP}_{50}$. As MaP scores have a weighted average of all subsets, the smaller and more challenging subsets of GWHD gain a larger relative weighting. We can infer from DETRs scores that it is likely performing best on the most difficult subsets of the dataset, and not having its MaP scored diminished as a result.

In table 6.2 we show a set of experiments using limited amounts of the GHWD for both style transfer and for finetuning our tast model. In this case, we perform all our analyses on DETR, using the 3 scenarios: the full dataset of 3290 images, along with subsets of 1000, and 100 images. As we hypothesised, using the maximum number of real images in combination with our synthetic data gives the best possible results with a MaP score of 5.39, a small improvement over the same test using real images only.

6.5.1 Analysis of Results

In our experiments we show that DETR does not demonstrate a significant departure from other state of the art models for this particular challenge. We see that while DETR does yield better results than Faster RCNN, this is not by a substantial margin, and moreover it is outperformed in all metrics by YOLOv5, despite YOLO being expected to be the least appropriate model for this particular challenge based on its known weakness on tasks with small objects.

We also see no strong indication that transformer models are significantly suited to wheat head counting, or use of synthetic data in general. Contrary to our hypothesis, DETR performs better than expected even with a limited training set of 100 real images when compared to our experiments with much greater numbers of images.

It is also worth acknowledging that contrary to our expectations, YOLO outperformed Faster-RCNN by a margin of more than 5 percentage point for both MaP experiments. We hypothesise that while YOLO based models are conventionally poorer at problems involving a large number of small objects than a region proposal model, some of the more recent advancements in YOLOv5 such as its Feature Pyramid Network allows it to perform better on objects of more extreme scales.

A more significant observation is that the application of our synthetic data provides only modest improvements over real data only. From the results shown in table 6.2 we see about a five percentage point improvement gained from adding synthetic data in the case of one hundred real, down to 2.5 percent gain with the full real dataset.

We also see more meaningful gains in performance with the increase of real images than we do from the injection of synthetic images. This might have a number of causes, but we hypothesise that the style transfer with low number of real images is much poorer, leading to little improvement. Improvements seen from having higher quality style-transferred images are likely thus significant because they coincide with a larger number of real images being used during training.

6.6 Conclusion

In this chapter we have presented our experimental finding using transformers on an upgraded synthetic dataset for the Global Wheat Challenge we saw in chapter 4. We find that a state of the art transformer model does not outperform a comparable state of the art CNN model. We also performed additional tests by constraining our real training set and find that in cases where training data is especially limited we are able to supplement our training regime with our synthetic data and achieve a greater performance, though this was less than our original hypothesis. Overall we find that at present Transformers are not an obvious replacement for CNN based architectures, although considering their relative infancy they hold a great deal of promise for the future. With further development of transformer models, it is possible that much greater performance could be achieved in a number of Computer vision tasks due to the advantages of their global attention mechanisms over currently popular architectures.

Chapter 7

Diffusion for Synthesis of Training Data

In chapters 4, 5 and, 6 we have so far demonstrated how synthetic data can be created for training CNNs using a combination of 3D rendering and GANs to create artificial images. In this chapter we consider diffusion models as a new method for generating synthetic images with their corresponding annotations, exploring this as a possible new method of generating photo realistic images that can be used to train neural networks with a significantly reduced domain gap. A number of different approaches to generating high quality synthetic images and corresponding semantic and instance segmentation masks is presented.

7.1 Introduction

Generative Diffusion Neural Networks are the major new paradigm in image synthesis. Reaching mainstream attention in 2021 and 2022 with the widespread uses[21] of a number of popular models such as Stable Diffusion[91] and Midjourney, much attention has been given to the useful work these diffusion models could potentially be put to. While much of the discussion revolves around using diffusion to generate artistic works and the related ethics of the technology, for researchers in Machine Learning these models present a perhaps even greater opportunity, allowing us to generate training images to suit whatever problems we are trying to solve. As described in section 3.1.3, diffusion approaches to image generation are a fairly recent area of research, and is likely to become a major area of interest in future years. Figure 7.1 shows an outline of our intended approach, leveraging a small number of real training images to generate a large number of highly realistic samples.

In this chapter we adapt current diffusion architectures to generate not just new images, but also matching ground truth labels for semantic and instance segmentation. We first present some preliminary experiments using diffusion to generate data for semantic segmentation, along with experiments to test its efficacy. We describe a number of potential approaches to generating image-label pairs for instance segmentation, along with their advantages and problems, and generate our own dataset using our preferred method containing over 20,000 images. We then experiment on our custom dataset comparing its performance at instance segmentation in a number of settings, including its ability to generalize onto different but similar datasets. We focus our work on the instance segmentation of leaves on rosette plants, such as arabidopsis, tobacco and spinach.

7.2 Motivation

For this work we were motivated by the incredible opportunities presented by generative diffusion networks. Compared to the need to hand craft Synthetic Data as shown in our previous work, diffusion models have demonstrated their ability to generate new examples having trained on relatively small datasets compared to GAN models (either for image generation or style transfer), and without the manual creation of digital scenes used for 3D rendering. Instead many diffusion models use text prompts to guide an images content and style, giving more flexible controls to the user, and allowing images to be fine tuned to specific requirements such as may be the case in designing training data.

Images generated by diffusion models are generally compared favourably with even the highest quality of GANs, and we hypothesise in this chapter that diffusion generated images will have a much smaller domain shift than other synthetic images making it appropriate for creating training data as a result. If this hypothesis is found to be even partially correct, we anticipate that a much smaller domain gap between diffusion based images and the real images they target will lead to a reduced need to deal with the complexities and limitations of domain adaptation.

Our choice of rosette plants as a target is based on a number of factors. Leaf counting and segmentation is often used to help calculate yield, growth stage and biomass of plants, meaning that it is an important part of many phenotyping pipelines for core downstream tasks. Due to a high volume of academic interest, there are already a large number of popular datasets already available for leaf segmentation, including CVPPP and Komatsuna, giving us a number of different datasets to work with for both our training and evaluation. Additional use of these datasets can be made when evaluating the ability of such a model to generalize onto similar unseen plants for example applying a model trained on arabidopsis to tobacco images.

While the problem of instance segmentation of leaves is not trivial, it is considered to be solved to a high degree in a supervised deep learning context [13]. We can assume that given sufficient high quality data, a state of the art machine learning model would perform extremely well on this task. By choosing a task with these characteristics we hope that we are best able to isolate the impact of our diffusion model on the problem, which we believe makes our results as meaningful as possible.

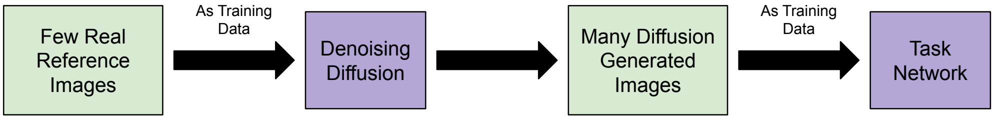


Figure 7.1: Here we show the intended pipeline using diffusion to create synthetic training data. Crucially a small number of real images can be used to train a diffusion model that can then generate theoretically infinite new sample images.

7.3 Aims

Our aim for this work is to use a diffusion model to generate image label pairs that we can use to train a CNN. As with our previous approaches to generating data with Blender, we must first be able to simultaneously generate our labels with our images, for this we explore a few approaches to this, focusing on the tasks of semantic and instance segmentation.

In this work we aim to answer the following research questions:

- Can current diffusion models be used to generate image-label pairs? What are the limitations and strengths of data generated this way?
- Which Computer Vision problems are possible to solve using current diffusion model architectures? Specifically is this an effective method for generating instance segmentation masks?
- To what extent does the domain shift problem occur between images generated by a diffusion model and those used to train it?

7.4 Background

Our experiments focus on different types of segmentation of the leaves of rosette plants. Semantic and Instance segmentation are two of the most popular computer vision tasks, and have been the focus of research for a number of years. In semantic segmentation we aim to assign a semantic class to every pixel in the image a class prediction. Often these classes will be one for each object of interest and then a background class for everything else. For cases with only a single class (as in our own experiments) this can also be thought of as foreground-background segmentation. Instance segmentation is a related form of the image segmentation problem in which we seek to separately label different instances of the same class, correctly identifying boundaries where one or more instance of the same object occlude each other.

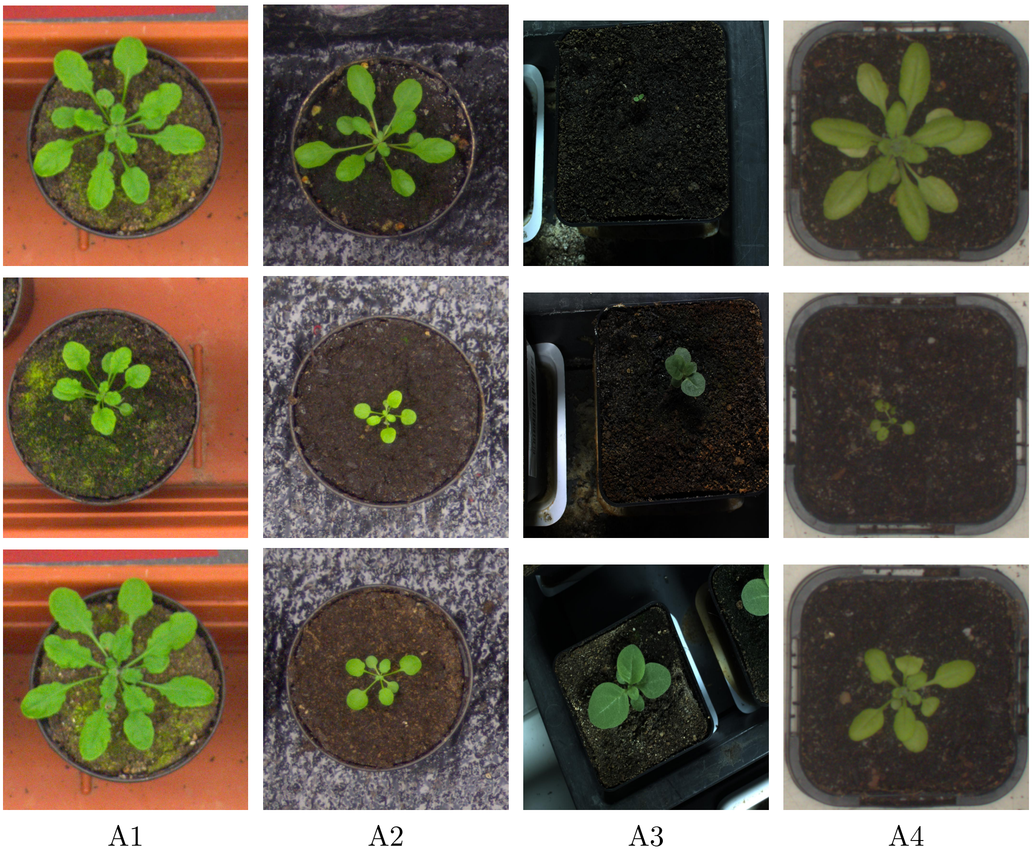


Figure 7.2: Example images from the CVPPA Leaf Segmentation Dataset, comprised of subsets A1, A2, A3 and A4. Subsets A1, 2 and 4 contain examples of Arabidopsis, while subset A3 contains tobacco plants. All four sets show a collection of plants at a range of different growth stages.

We use the *Computer Vision Problems in Plant Phenotyping* (CVPPP) leaf segmentation dataset for our training images as well as the associated challenge for some of our evaluations. The CVPPP leaf segmentation challenge shown in figure 7.2 is an online competition for segmenting individual leaves from arabidopsis and tobacco plants [72] from top down images of plants in different growth stages. The challenge originally took place in 2017, extending the earlier leaf counting challenge which used the same image set. The dataset contains a set of training images divided into 4 different subsets, with the smallest subset containing 27 images, and the largest containing 624 images, with subsets A1, A2 and A4 containing arabidopsis plants, and subset A3 containing high resolution photos of tobacco plants. Each image is paired with a set of annotations including a binary mask showing pixels containing leaves in white and the background in black, and an RGB mask, showing each individual leaf in a unique RGB color on a black background. We also use images from the Komatsuna datasets as part of our evaluation which contain addition images and labels of Arabidopsis and Komatsuna (spinach) plants respectively.

7.5 Related Work

At present there are only a handful of research papers [4] using diffusion models as a source of synthetic training data, most likely reflecting how recent the technology is. While there is much interest in using diffusion for generating training data for other neural networks, at present most research interest is currently focused on improving the quality of generated images, or improving the ways that image generation can be controlled.

Constrained diffusion models are of course very common, with the most common constraints being in the form of text prompts. One possible approach to generating synthetic data would be to using the annotation as a constraint from which to generate an image.

Some examples of research into generating training images images this way with text prompts include work by Andreas Stöckl [102] looking at generating images in Stable Diffusion for classes from the ILSVRC [97]. While this example shows encouraging results for the future us of diffusion generated models, non-geometric prompts such as text prompts are only appropriate for classification tasks and are not easily adjusted to more complex tasks.

Extending this *annotation first approach* to detection, can be seen in Xie et al [121] and is reminiscent of GAN based approaches such as Pix2Pix [50]. Xie's diffusion model is designed to embedd specific pieces of semantic information from their prompt to bounding boxes within the overall image allowing them to create images that will align with object detection masks that are used as inputs. *Composer* by Huang et al is a more general constrained model allowing control over image synthesis using 8 different components including shape, semantics, style and palette [49].

These approaches unfortunately face the limitation that for every new image we hope to generate for a training set, we must first somehow construct a new annotation to use as input. While for problems such as object detection this could be done automatically, by simply generating random bounding boxes, it becomes more complex for problems such as segmentation where pixel perfect masks must be provided and domain knowledge may be needed to maintain realism. To this end we focus our research on the approach of simultaneous image and label generation.

Some specific research into generating images with semantic masks in the past year include Wu et al [120] who use the cross attention maps between prompts and the generated image to extract semantic masks from generated images. Finally Park et al [80] uses a similar approach to our method shown below, focusing on generating semantic segmentation masks of faces concurrently with image generation.

This chapter focuses on leaf segmentation of rosette plants, a challenge attempted by a number of other methods for generating synthetic data have already been highlighted in previous chapters [118] [117] [110]. Since the CVPPP dataset is rather small (especially the A3 Tobacco subset), this challenge is especially susceptible to the overfitting problem that we have discussed previously, however there is a broad range of different approaches to this challenge that we will now consider.

The CVPPP leaf counting and segmentation challenge has been thoroughly explored by the three conventional approaches to synthetic data generation, compositing, 3D modelling, and GANs. Both [99] and [60] use compositing methods (sometimes also called collage), using real leaves from the provided train dataset, and recombining them to produce new images. This approach is effective in generating new images with accurate segmentation labels, but it is limited, and both papers attempt a number of additional augmentations to create greater training variety from their images. [118], [117] and, [108] all use 3D rendering approaches, for both the counting and segmentation problem. Since these works do not makes use of any kind of domain adaptation ([117] instead opting for a domain generalization strategy), these synthetic datasets have very limited generalization onto real test images. Finally [110] presents *Arigan*, a conditional GAN for generating synthetic images for leaf counting. While this model shows some success, the images overall are quite low quality and less than a hundred generated images were selected as good enough to use for training purposes.

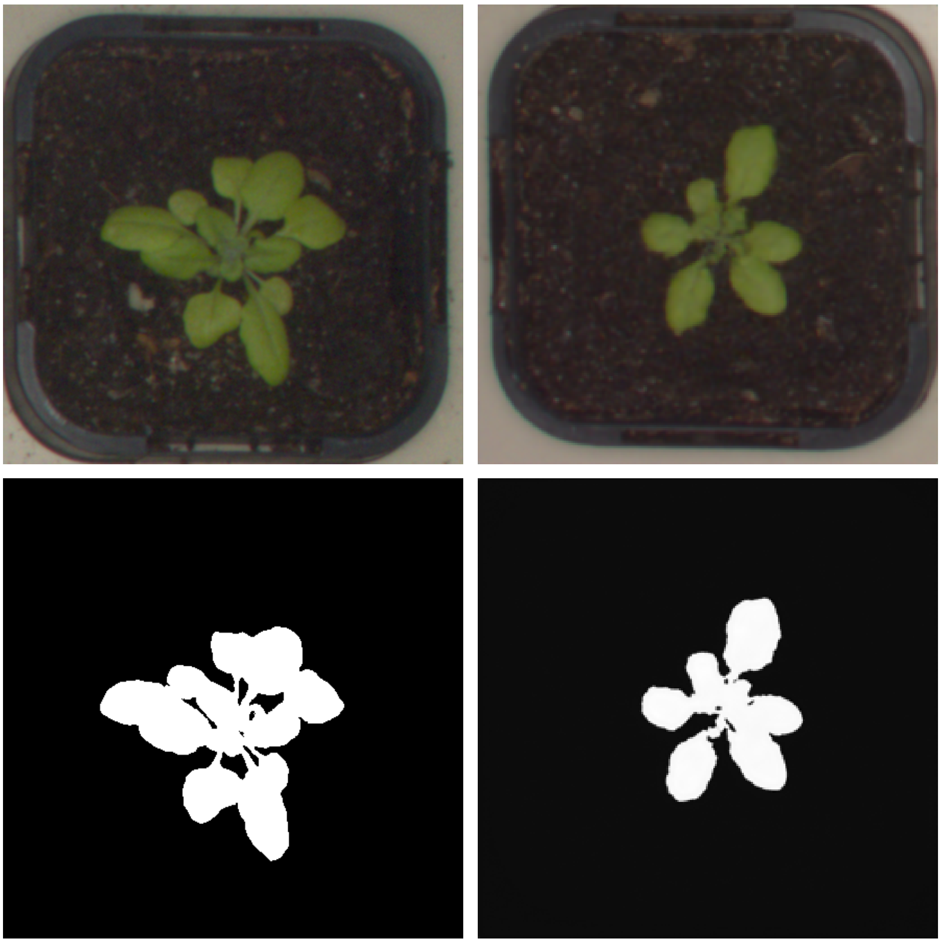
7.6 Semantic Segmentation

In this section we discuss our experiments, focusing on semantic segmentation of arabidopsis leaves. For many purposes, semantic segmentation can still be a useful plant phenotyping tool, especially for estimating biomass, as number of *plant* pixels can be an extremely good proxy for the total amount of plant matter. Though our eventual intention is to extend our approach to instance segmentation we begin with semantic segmentation, as it is a much simpler challenge conceptually, and is easier to generate labels for under most usual circumstances.

7.6.1 Image Generation

We investigate the ability of diffusion models to generate image-mask pairs by extending the model to generate four channel images and concatenating our training images with their matching segmentation masks to create a four layer image RGB-M (mask). We then extend our Diffusion model (an implementation of [46]) to generate new samples based on a 4 channel input that the model would then be trained on. Functionally this extra channel is no different to the three colour channels of the image, and could similarly be done with a depth or transparency channel as would be more common for 4 channel images. Examples of images used for training and our generated output can be seen in figure 7.3.

We train our diffusion model on the CVPPP A4 dataset for 100 epochs by which time the model is able to produce highly realistic output images. We then generate a dataset of 10,000 synthetic images, extracting the 4th channel as a separate binary mask which we then use as our label, examples of our image and labels can be seen in figure 7.3.



Real Image/Label Diffusion Image/Label

Figure 7.3: Examples samples from our diffusion model showing comparison against real image label pair for the CVPPA dataset.

7.6.2 Experiments

In order to test the effectiveness of this approach, we conduct a set of experiments evaluating our network's ability to generate useful training images with accurate masks. To test our data we train a DeeplabV3 model [19] four times, on the real A4 dataset, on our Synthetic dataset, on another Synthetic dataset from Ward et al [116] (referred to here as CSIRO) and finally on a mix of the two trained first on the diffusion generated data and then fine tuned on the real set. We then test these datasets on all four CVPPP subsets as well as the Komatsuna datasets.

7.6.3 Results

Our results shown in table 7.1 show mixed results. Our real dataset performs well in datasets A2, A3, and A4 but very poorly in A1. In contrast to this we see that

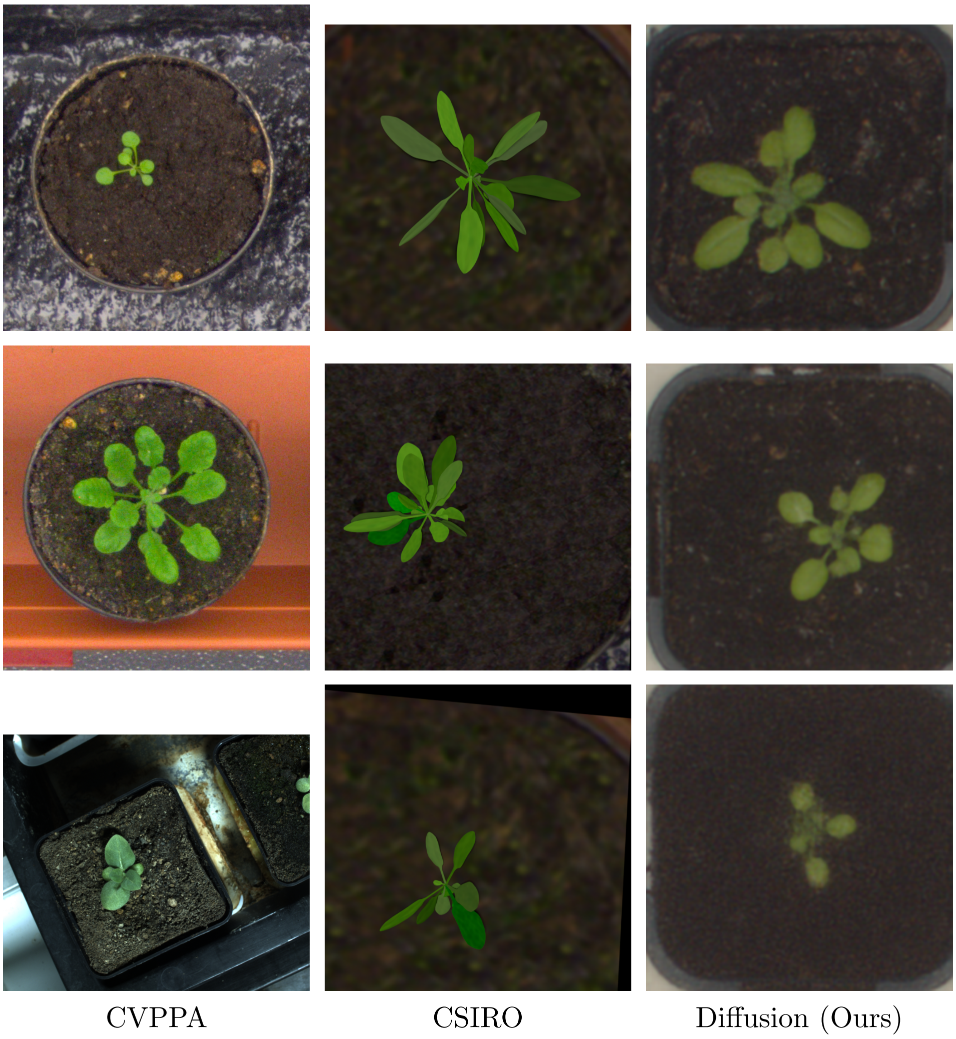


Figure 7.4: Comparison of real, CSIRO and diffusion images used in our experiments.

both synthetic datasets (CSIRO and our diffusion dataset) perform well in A1, with CSIRO outperforming the other datasets. Directly comparing diffusion data to CSIRO's 3D rendered synthetic data, we see that both approaches perform similarly, diffusion outperforming CSIRO by small margins on datasets A2, A3 and A4. Overall we see that diffusion images perform roughy as well as comparable synthetic data for this task, and demonstrate a capability to generalize to different levels onto datasets other than that which it was generated from.

In the next section we hope to extend this work by looking at whether we can extend this task to that of instance segmentation, presenting the first published work to apply diffusion models in this way. We choose to extend our work to instance segmentation due to generating instance masks being more novel, and the

Training Dataset	Test Set (IoU)			
	A1	A2	A3	A4*
(1) Real Only	0.188	**0.457**	**0.512**	**0.708**
(2) CSIRO	**0.718**	0.364	0.421	0.637
(3) Diffusion Only	0.605	0.381	0.459	0.691
(4) Real & Diffusion	0.195	0.390	0.397	**0.708**

Table 7.1: Results of experiments on semantic segmentation using diffusion generated images with highest scores in bold. For these experiments we generate a synthetic dataset using images from CVPPP A4 as our training data, and then test it on other subsets of the dataset.

greater value implicit in using synthetic data for instance segmentation training, caused by the increased costs of annotating instance masks and greater challenge relative to semantic segmentation.

7.7 Instance Segmentation

In this section we continue our development of generating image-label pairs on the more challenging task of instance segmentation. Instance segmentation has many advantages over semantic segmentation, allowing for analysis of individual components (be that fruit, leaves, etc), and a direct means of counting, which can be useful as part of a phenotyping pipeline. Instance segmentation is both a more challenging computer vision problem to solve, but also creates a greater challenge for generating masks, as each instance needs a separate label and cannot be encoded in a binary mask in the same way as semantic masks. Here we describe a collection of different approaches to label generation as well as some experiments we conduct to compare our synthetic data to real data and other synthetic datasets.

7.7.1 Training Dataset

To train our model, we aim to build a new dataset of synthetic images along with instance segmentation masks using a diffusion model. We test a number of different approaches to image generation looking at their impact on the quality of images created and the effectiveness of each approach to generating masks.

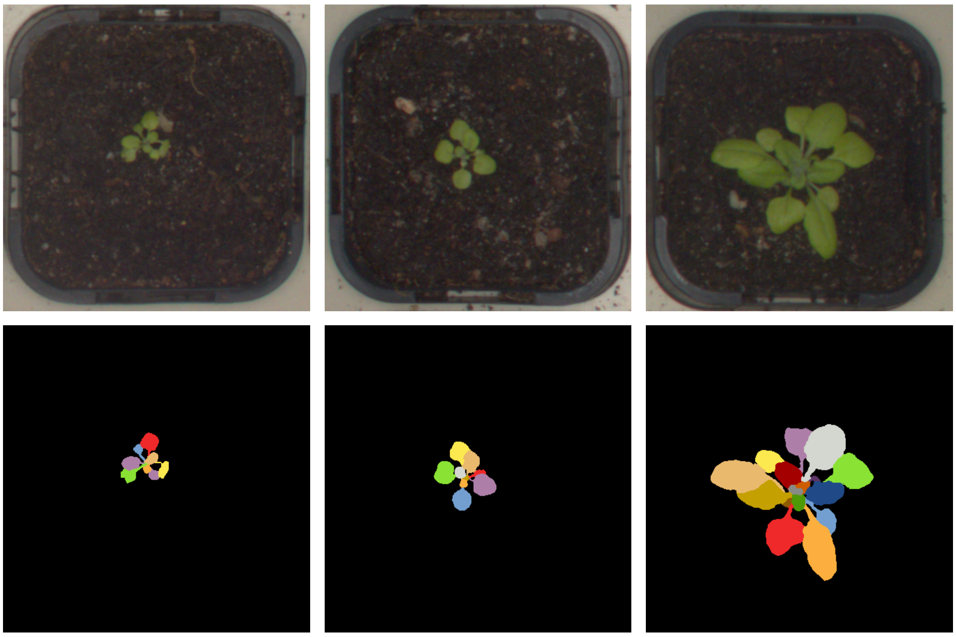


Figure 7.5: Examples images from the CVPPA leaf segmentation dataset along with their RGB masks.

As shown in section 7.6, the task of semantic segmentation is fairly straight forward as there is only one class for the leaves and a second class for the background. This allows us to use a binary mask for the segmentation mask and combine this with the original image to create a four channel image for the diffusion model to learn. For instance segmentation we encounter a more challenging problem, as we need to encode instance segmentation information in such a way as to be able to generate samples using our diffusion model. Many deep learning approaches to instance segmentation today store information in JSON or similar formats which could not be easily output using our diffusion method without significant modifications. In particular, the unknown number of total labels make it more challenging as the mask must be a much more flexible format.

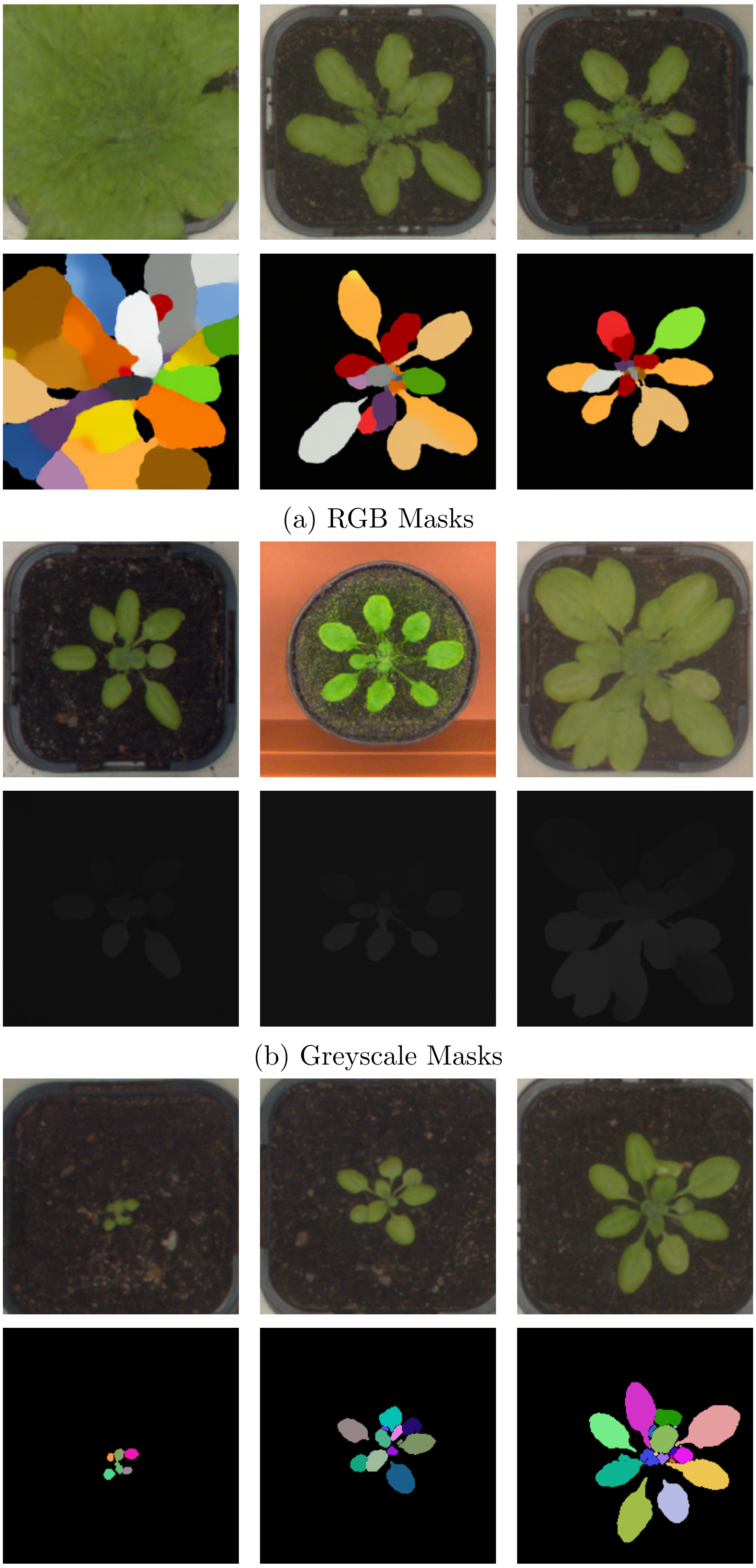
To solve this problem we use the images from the CVPPP training set along with their instance segmentation labels, which are encoded as RGB masks, with different colour values assigned to each leaf instance, and black assigned to the background, we show examples of this in figure 7.5. To generate our images, we continue our approach from our preliminary set of semantic segmentation experiments, using a modified version of the original denoising diffusion model presented in [46]. Our approach to image generation is again to generate image-mask pairs simultaneously, treating the combined image and mask as a large multi-channel image that can be separated to isolate both the image and the mask to use for network training afterwards.

We attempt a number of different methods to generate instance segmentation masks using this framework which we describe below, outlining their advantages and disadvantages.

Generating RGB Masks

Our initial approach is a direct extension of our approach in 7.6, and involves generating RGB segmentation masks for image, with the background coloured in black and each leaf instance given its own RGB color. During training of the diffusion network the 3 channel image was concatenated with the 3 channel mask and the network was configured for 6 channel diffusion.

After training we generate 10,000 samples from our trained model, we show examples of the output in figure 7.6. As can be observed by inspection the images generated are very good, although some failure cases do occur. In the case of the generated masks we observe that even the masks corresponding to higher quality images exhibit problems. Masks can be seen to have edges blurred with the background and with each other. In many cases each individual leaf was made up of a number of regions of marginally different colors. Because of the blurring and noise present in the generated masks, we found it was not possible to extract the appropriate mask without applying significant image post-processing, a task that was made more difficult by the high level of variation across all our samples. Furthermore, we observed than the quality of the image samples produced by the model was lower, and trained slower than the 4 channel model used for semantic segmentation. We hypothesise that the diffusion model performs increasingly poorly as you increase the number of channels.



(a) RGB Masks

(b) Greyscale Masks

(c) Outline Masks

Figure 7.6: Examples samples from our diffusion models for the methods of generating segmentation masks. Method (a) shows our attempt to directly generate RGB masks in the format originally shown in our training data. Method (b) shows labels generated as greyscales, with values ordered from top to bottom by intensity. Method (c) shows our final approach, where labels are generated as outlines and then flood filled.

Generating Greyscale Masks

Having identified these limitations to using diffusion to generate masks in this way we made a number of additional attempts to generate instance segmentation masks using a diffusion model. For our next attempt we reduced the mask to a single grayscale channel with a different value for each leaf, counting up sequentially from 0 with leaves labelled from the top to the bottom of the image. We hypothesised that by reducing the number of channels, and labelling the leaves in a more meaningful way rather than the random approach taken with the original RGB masks that this would help the network to generate higher quality images as well as keeping mask accuracy good enough to be usable as training data.

As before, we generate 10,000 samples (also shown in figure 7.6) and find that while image quality does improve, we still observe too low quality mask generation to make good training data. Here we observe that rather than generating unique greyscale values for each leaf, a gradient is generated across the entire mask without clear borders between instances. We hypothesise that diffusion models as they are currently designed do not lend themselves to this kind of mask generation, due to losses being low for pixels of different instances sharing the same intensity. While we consider potential solutions to this problem in section 7.11.2, we instead focus the rest of this work on our chosen mask representation presented below.

Generating Outline Masks

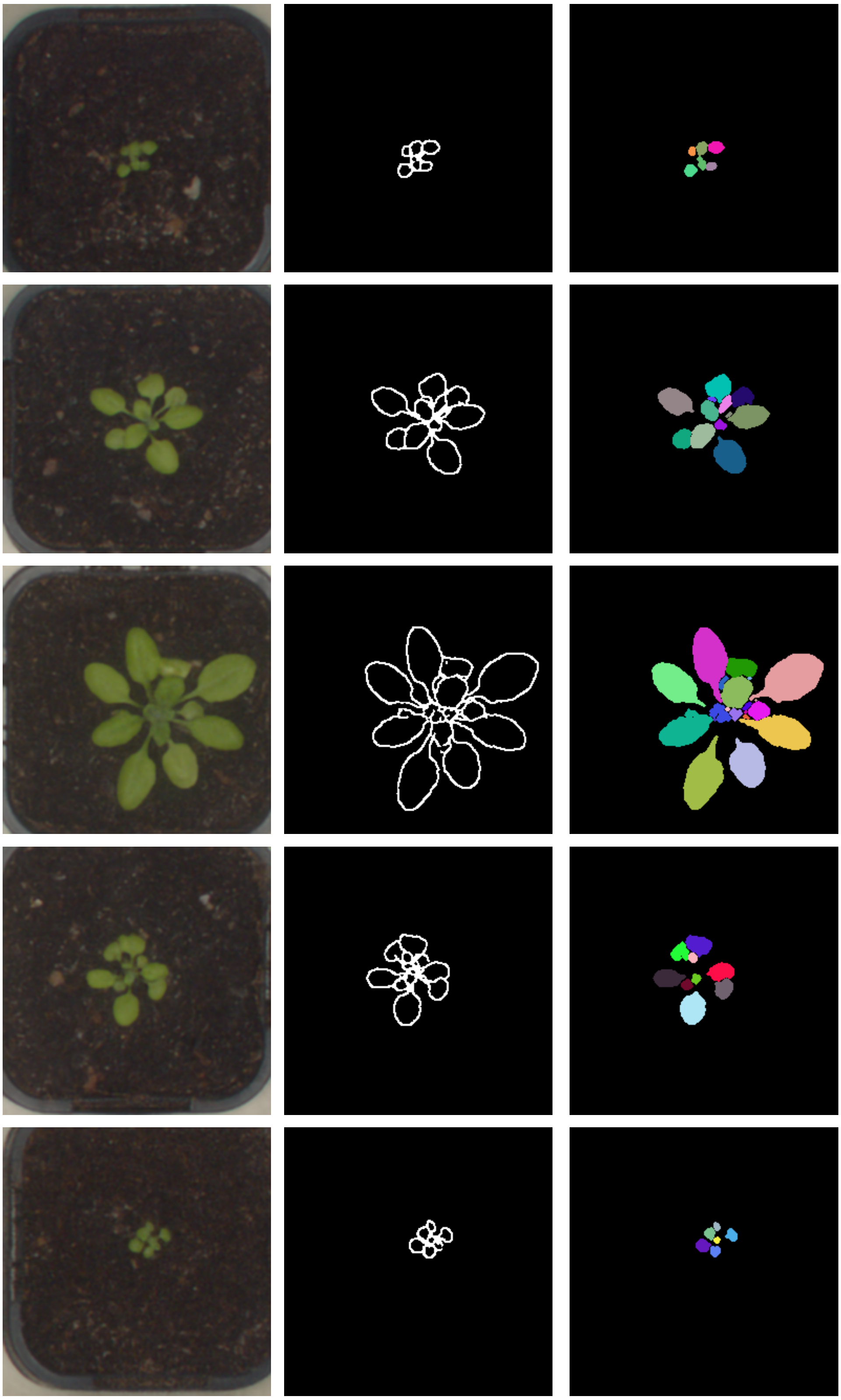
Finally, we attempt a more complex multi-step process combining the diffusion process with a post processing step. Having observed that we get much better performance from our semantic model generating binary masks we attempt convert our RGB masks to a binary representation using an outline based approach to mask generation. We hypothesise that by creating a binary representation of the instance segmentation mask that we can then restore to the full RGB format, that we can create high quality image-mask pairs using diffusion without compromising on mask correspondence.

With this in mind we create a *rgb to binary* conversion for masks from our training set using a Sobel edge detector on the provided RGB instance segmentation masks followed by a binary filter to create a black and white outline. We continue this approach by using a dilation of the detected edges, to discourage breaks in the outlines generated by our denoising diffusion model. We generate a training set of images using this approach and for each one extract the outlined based mask. To restore the instance segmentation masks from the binary, we reverse this transformation using first a flood fill, detecting regions from the outlines and filling them with unique RGB colors. We then remove extremely large and small regions, based on the intuition that very large regions will be filled in segments of the background, and extremely small regions will be gaps between leaf edges and remove any other small artefacts using noise removal techniques restoring an accurate representation of the RBG mask for each image. Examples of this are again shown in figure 7.6.

To complete this process we attempt to remove samples from the generated set that are easily detectable fail cases, for example those with an extremely large or small number of leaves. Any images that are removed can then simply be replaced by generating additional samples until a complete dataset of 20,000 images has been created. Overall roughly half of all images generated are identified as fail cases and removed, from which 20,000 images were then selected at random to make our final train set.

7.8 Experiments

We conduct a series of experiments testing our diffusion datasets for both segmentation accuracy and generalisation across different datasets. For all experiments each diffusion model is trained for 200 epochs on an NVIDIA A6000 GPU, at which point we empirically determined that no further increase in image quality. Since the goal of our work is to produce a synthetic dataset that is a suitable replacement for collecting large real dataset, the goal of these experiments is to



(a) Synthetic Image (b) Synthetic Outline (c) Synthetic RGB Mask

Figure 7.7: Examples from our diffusion model showing high quality samples produced after identified failure cases have been filtered out of the dataset.

verify first whether our synthetic dataset performs comparably with real dataset, and secondly the compare their performance with synthetic datasets generated with more conventional means.

7.8.1 Bounding Box Detection Experiment

In this experiment, we test our results against the unused CVPPA splits for the task of bounding box detection. For our test we compare the results of training Faster-RCNN with three datasets:

- (1) The 100 real images from the CVPPA A4 dataset used to create our synthetic data.
- (2) The CSIRO Synthetic Data created with 3D modelling of 10,000 3D rendered plants [116].
- (3) Our diffusion generated dataset of 20,000 images.

The test sets used are each of other splits from the CVPPA dataset, as well as the unused images from the A4 split that were not used during training.

7.8.2 Instance Segmentation Experiment

In this test, we test our results against the unused CVPPA splits for the task of instance segmentation. We use the same datasets as for bounding box detection using their instance segmentation masks instead of bounding boxes. This time the model used is Mask-RCNN, set up with default hyper parameters.

7.9 Qualitative Results

We present a series of images generated from each of our diffusion networks in figure 7.7, along with their corresponding annotation outline. In this section we provide some general analysis of the quality of the images produced.

7.9.1 Image Quality

From our samples presented we can see that the diffusion models are able to produce a wide variety of plausible new images most of which are largely indistinguishable from real images to the human eye. Importantly we see that the network is able to produce images of both small and larger plants, reflecting some of the diversity in the training data.

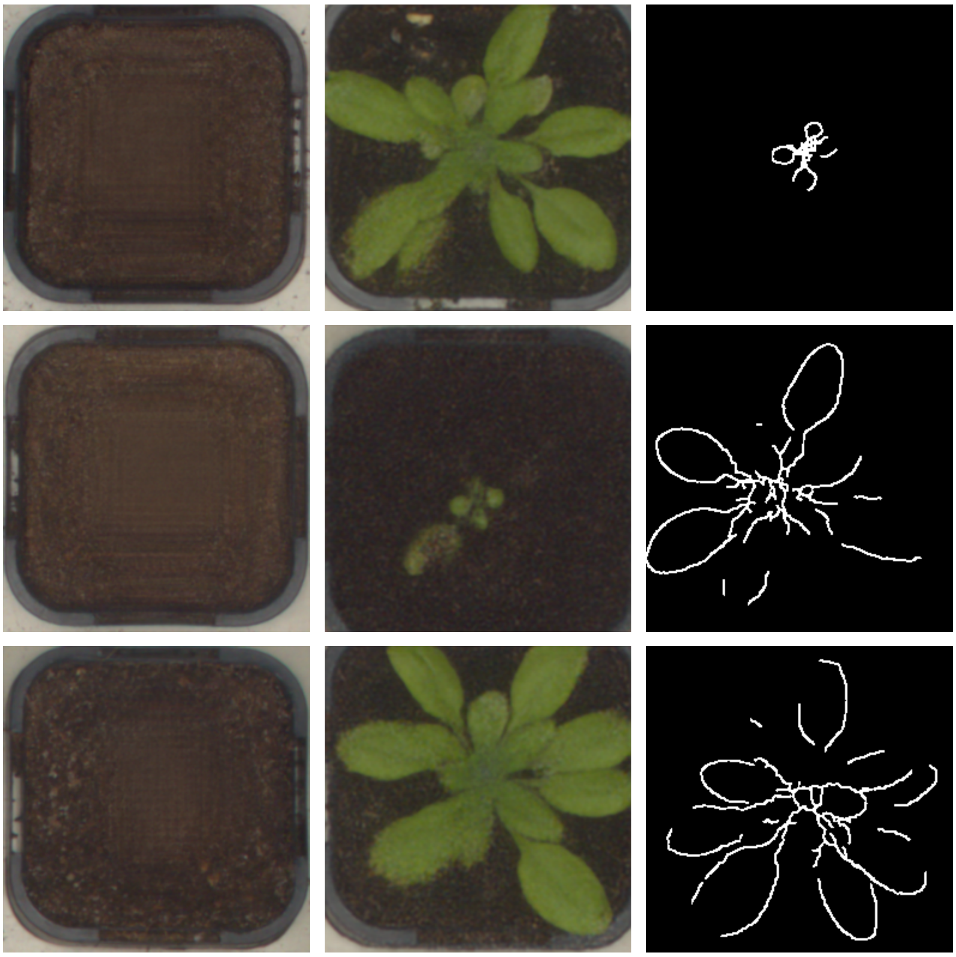
7.9.2 Annotation Quality

Looking at our annotation outlines presented in figure 7.7, we observe that a majority of the annotations are high quality and have a good correspondence with the leaves in the matching image. Conversely we can see that a number of annotations do suffer from not forming a complete boundary for each leaf such as those seen in column (c) in figure 7.8. Due to our flood filling method of creating RGB segmentation masks these instances will lead to poor quality labels for the image. The most common problem caused by this is missing labels, where one or more of the leaves present in the image will not have a mask in the annotation.

7.9.3 Failure Case Detection

Here we show a selection of images generated by our diffusion models that represent failure cases. Because these failure cases happen at the image generation phase and are intermittent, we have the opportunity to automatically remove offending images from the dataset prior to training our segmentation models. By generating a much larger number of images than we intend to use for our downstream task, we can ensure that our final dataset remains our desired size while maximizing for training image quality.

In column (a) of figure 7.8, we can see that there are examples of images generated where the model generates an empty pot without any plant in it at all. We estimate that depending on the experiment, this occurs on average in approximately 10 percent of all images generated. Fortunately of all the failure cases we present, this case is extremely easy to remove from the pool of images as in each of these cases the accompanying mask that is generated is also blank, making them the easiest to filter out automatically.



(a) Missing Plant (b) Distorted Leaves (c) Outline Gaps

Figure 7.8: Examples samples from our diffusion model showing different failure cases we observe.

We also observe cases where the model generates images where the leaves of the arabidopsis plant shown have become merged together such as those in column (b) of figure 7.8. This is most common when training on the A4 subset of the CVPPP due to the large number of leaves in the older plants, due to the acclusion between different leaves.

We also show failure cases where our outline annotation strategy fails to generate complete outlines for the leaves in the image in column (c) of figure 7.8. We find that failure cases such as these vary greatly depending on the image set we train our model on, specifically performing worst on CVPPP subset A4, which contains most of the largest plant and subsequently the images most dense with leaves and featuring a lot of occlusion. In some cases, these images can be identified after performing our flood filling phase. By detecting the largest regions in each image and removing images for which that region is larger than a predefined threshold, we are able to identify images where the flood fill has spilled outside the bounds of an individual leaf and remove them from the dataset.

7.10 Quantitative Results

We show results for our experiments in tables 7.2 and 7.3. Here we see that our diffusion generated dataset does not perform as well at detection and instance segmentation as we saw with semantic segmentation in section 7.6. In the case of bounding box detection, we see that neither synthetic dataset (experiments 2 and 3) are able to perform as well as the real dataset, with the CSIRO dataset outperforming our dataset, especially for A2.

Our instance segmentation experiments similarly show that synthetic data is unable to match the real data in terms of performance. We do see that unlike for the previous experiment, in the case of instance segmentation, diffusion generated images perform much better than the 3D model plants in the CSIRO dataset. Our best results for instance segmentation are seen in datasets A2 and A4 where we score 30 percent IoU for our diffusion dataset, suggesting that while our method shows some promise further work is needed to achieve top level performance.

Training Dataset	Test Set (IoU)			
	A1	A2	A3	A4*
(1) Real Only	**0.48**	**0.46**	**0.41**	**0.64**
(2) CSIRO	0.13	0.40	0.23	0.20
(3) Diffusion Only	0.06	0.12	0.21	0.22

Table 7.2: Results of experiments on bounding box detection using diffusion generated images. For these experiments we generate a synthetic dataset using images from CVPPP A4 as our training data, and then test it on other subsets of the dataset.

7.11 Discussion

Overall the results presented demonstrate significant progress in the development of a synthetic data pipeline using diffusion generative models, but a failure for

Training Dataset	Test Set (IoU)			
	A1	A2	A3	A4*
(1) Real Only	**0.56**	**0.48**	**0.40**	**0.63**
(2) CSIRO	0.01	0.07	0.07	0.07
(3) Diffusion Only	0.13	0.33	0.21	0.33

Table 7.3: Results of experiments on instance segmentation using diffusion generated images. For these experiments we generate a synthetic dataset using images from CVPPP A4 as our training data, and then test it on other subsets of the dataset.

these results to translate into good quantitative performance in its current form. In this section we highlight a selection of important takeaways from our results presented above.

Diffusion Generalizes to real data without domain adaptation. From our high results (most significantly in our semantic segmentation experiments in section 7.6), we can conclude that the domain gap between diffusion generated and real images is extremely small, and does not require domain adaptation or harmonization to make effective training data.

Diffusion generated data is as effective at generating training data training data as 3D modelling. Our diffusion data generally performs comparatively or better than the 3D modelled plants in the CSIRO dataset we test against. While 3D modelling has many advantages over our diffusion approach, such as a greater level of control, this is promising as it suggests that the quality of the images generated by diffusion are competitive with those from the 3D models, and indicates they may be a suitable replacement for industry use with further development. We hypothesis that the artificial appearance of the CSIRO plants is the main reason for their poor performance, preventing the CNN from generalizing onto real images.

Creation and alignment of labels is the main challenge for creating synthetic data with diffusion. We hypothesise that the significant drop in performance between our semantic and instance segmentation performance is mostly caused by image artifacts and missing leaves in our labels. If annotations with perfect correspondence could be created it is likely they would achieve higher scores in our detection and instance segmentation experiments.

Diffusion represents a low human cost for dataset production. While generation of our diffusion dataset still requires the collection and annotation of a small number of real images, we believe this still represents a lower human cost than creation of a domain specific 3D scene, and this gap is likely to widen with more complex plants. Similarly, despite having the same cost as the real images, our diffusion based approach still allows for the generation of very large datasets as theoretically infinite samples can be generated from the trained model.

7.11.1 Comparison to other data generation methods

As a major focus of this chapter has been on the current effectiveness of diffusion models for generating synthetic data, it is important to compare this approach to data generation against other methods of data generation, including those we show in this thesis.

Synthetic data generally has a high upfront cost, associated with the creation of tools of neural network models needed for its generation, and these costs then remain low during data generation. This infers advantages when the upfront cost is low and the quantity of data required is high. Diffusion models differ from 3D modelling approaches as they still require some initial real data in order to train the initial model. By keeping our initial input data limited to 100 images, we hope to show that this initial cost to create this dataset would be low for other researchers seeking to reuse this pipeline, and hopefully thus make diffusion preferable to 3D modelling as a method of 3D data generation.

7.11.2 Future Work

The work presented in this chapter illustrates the current challenges and limitations of current diffusion models to generate synthetic data. Here we consider how we perceive this research could be continued in the next few years to enable diffusion models to be effective for data generation.

Improved Label Correspondence As highlighted by our work, one of the main challenges we face is achieving a high level of correspondence between the image and label. We believe that diffusion models that have been explicitly designed for simultaneous generation of images and labels will likely be needed to achieve the highest performance when generating annotated synthetic data. We also believe that an additional loss function designed to ensure coherence between image and label would help to ensure accuracy in label generation. Due to the fast moving nature of this field of research, and the increasing commercial interest in synthetic data, we believe that these kinds of advancements that fully leverage diffusion will emerge in the next 5 years, in which time we will also see better advances in photo realism and constraining image generation through a variety of modalities.

Mask to image generation As discussed in section 7.5, much of the current research into diffusion networks looks at different forms of constraints that allows the user to control the output of the image. We believe that a *mask first* approach to synthetic data generation has some advantages over our approach, and is another exciting area for further research. We hypothesise that a pipeline using two models, one generating a mask from noise, and then a second diffusion model generating the image from the mask could be an effective approach to ensuring high quality labels and images.

7.12 Conclusion

In this chapter we have investigated diffusion as a potential next approach to state of the art synthetic data generation. We have focused on a range of segmentation problems, and explored the challenges related to generating high quality image-label pairs for these problems.

Our results presented showed some success in generating synthetic data for both semantic and instance segmentation, illustrating that it promises to be a serious contender to 3D modelling as an approach to data generation in future. Both approaches to Synthetic Data have a high upfront cost, in developing the initial model, followed by an ability to produce large quantities of data cheaply afterwards. One advantage of Diffusion models is their potential to be extremely flexible, as we see with general purpose models such as DALL-E and Stable-Diffusion, which might make them more useful in the long term a means of generating new datasets.

In addition to our promising results, we also highlight the main challenges that need to be overcome in future work to fully enable diffusion to be best utilized to generate training data for use in industry and other research.

Overall we have presented a significant contribution to investigating a major application of diffusion technology. The approach using outlines we present in this work has been made available on github, at https://github.com/zanehartley/Cold_Diffusion_for_Synthetic_Data to enable other researchers to extend or reuse the method for other tasks. We believe that with future research discussed in section 7.11.2 and in chapter 5.8 which expands upon the ideas in this chapter, that diffusion will become the most efficient and high performing method of producing synthetic data.

Chapter 8

Conclusions and Future Work

8.1 Contributions

In this thesis we have contributed a set of important approaches to using synthetic data for deep learning, as well as a number of useful datasets and pipelines that can be replicated with the provided code. We also present experimental results, looking at areas of current and emerging interest, such as transformers and diffusion model.

Chapter 4 contributes our first synthetic dataset and its GAN-enhanced version, containing over 5000 synthetic images, and over 20,000 GAN-enhanced images respectively. This is a useful contribution as it significantly supplements the GWHD, one of the most popular plant phenotyping dataset used in research, with our results also highlighting the high impact of small numbers of real images when included in otherwise synthetic training datasets. We also make two novel contributions to synthetic training pipelines. Our novel clustering approach to domain adaptation is a versatile contribution, and is extremely practical for a wide variety of problems that use domain adaptation such as those that aim to reduce the domain gap for synthetic data. Our heatmap guidance addition to CycleGAN is another useful contribution; while there are a number of attempts to improve CycleGANs consistency for domain adaptation, our method demonstrates good performance while having limited computational cost.

In chapter 5 we introduce one of our major contributions, our Volumetrically Consistent CycleGAN , which enables the accurate prediction of a 3D volume of a fruit from a single 2D image. This network is, to our knowledge, the first unsupervised 3D reconstruction network. This is a very important contribution, as it massively reduces the cost of performing 3D reconstruction using deep learning. The ability to perform more complex tasks using synthetic data is also a crucial achievement as at the moment it is mostly limited to the more common problems such as detection or segmentation in other literature.

Chapter 6 makes a number of experimental contributions. While transformers have become popular in recent years, we presented some of the first results of using synthetic data comparing their impact on transformer vs state of the art CNNs for plant phenotyping. We also contribute our experimental results on use of limited real data for domain adaptation and its effect on performance, especially with data-hungry transformers. Overall our results showed no significant advantage to using synthetic data in Transformers compared to CNNs, instead showing that state of the art CNNs outperform a current state of the art Transformer model at the task of object detection when trained with our data.

Finally in chapter 7 we contribute our technique for generating synthetic data with annotations using diffusion models, the first we are aware of in academic literature. In addition to presenting our approach to data generation, we also show a range of experimental results looking at both the realism and accuracy of the images and annotations generated as well as their effectiveness at training state of the art deep learning models. The work generating instance segmentation results here is a very important contribution as diffusion models are likely to be one of the best ways to generate synthetic data in future, and work looking at the simultaneous generation of annotations for Computer Vision tasks will be extremely important. Overall our results are promising, showing extremely good qualitative performance of our diffusion network at producing both images and corresponding masks. Our quantitative results for semantic segmentation are also quite good, however for the more challenging task of instance segmentation we are unable to achieve performance comparable to real images, suggesting further work is needed to improve image and label quality.

Collectively these contributions fulfill the objectives stated in chapter 1, and represent an opportunity for savings in cost and time for biologists working in plant sciences and agriculture through the production of our data creation pipelines. Through our publications and outreach we have made numerous and varied contributions to the current discourse surrounding the use of synthetic data for digital plant phenotyping including sharing this work with both computer and plant scientists. Finally through our continued work with generative diffusion models we have made great strides in harnessing the newest state of the art AI technologies for this important field of research.

8.2 Future Work

As shown in this thesis, Synthetic Data is likely to play a large role in machine learning in the future, and as generative AI continues to be a significant area of research we can expect the generation of synthetic training data to be of a great deal in interest in future years. In this section we will discuss some potential future areas of research that show a great deal of promise in the coming decade.

8.2.1 Benchmarking and Analysis of Synthetic Data

Most research into using synthetic data current seeks to establish the effectiveness of their specific dataset or methodology. At present there is a lack of research into analysing and benchmarking synthetic data, and little understanding of best-practice regarding its creation and use. While many papers compare a new synthetic dataset to a real data equivalent, there are no papers to our knowledge that compare a number of different synthetic datasets to assess what makes them better or worse as training data.

8.2.2 Further applications in Agriculture

For each of the individual works in this thesis we have attempted to design our methodology to be adaptable to additional problems and not rely on domain specific elements of the tasks we solve. As a result a core piece of additional work would be to apply the synthetic data pipelines from chapters 4, 5 and 7 to create Synthetic Data of some of the thousands of other plant species used in research today. Future research into improving these pipelines also needs to be done with consideration for the wide range of different downstream applications, and so must continue this domain invariant approach. By widening the range of different datasets available, and making tools for creation easier to use, we hope to widen adoption by different parts of the phenotyping community will be made easier, allowing the these technologies to have the greatest possible impact.

8.2.3 Generalised Unsupervised 3D Reconstruction

In Chapter 5, we demonstrate a method of unsupervised domain adaptation for 3D reconstruction to great effect. The results presented are, however, constrained by a number of factors we describe in more depth in the chapter, most importantly that of computational resource, and because of these factors, the scope of this work was limited, most prominently in the reduction in spacial resolution of our targeted objects. We believe that the approach presented is extremely promising, and given even recent advancements in more powerful GPU hardware, the opportunity of using our architecture for 3D reconstruction of more complex objects is much more realistic. Future work could focus on using synthetic training data for a wider range of target objects, as well as refining our architecture to improve performance and training efficiency.

8.2.4 Diffusion for General Annotations

As shown in Chapter 7, diffusion models are an effective way of producing training data for computer vision problems. However, as we previously described there is currently no research demonstrating the specific use of diffusion models designed with the production of image label pairs in mind. While most extant research, including our own, currently focuses on making use of current diffusion models using creative approaches, in the future we expect models to be designed with duel image-label outputs to be designed specifically for this task. As prompt-based image generators like Dalle-2 have shown incredible promise in recent years, it is also possible that generating specifically designed synthetic data using prompts could replace our current approach of expanding small train sets by generating new samples as shown in our own work.

8.2.5 Further Unsupervised or Semi-supervised approaches

As shown in our work in chapter 5, approaches to synthetic data that allow us to bypass collecting real data are the gold standard in creating deep learning training data, as it most completely removes the associated cost. One of the biggest limitations of our other works, and indeed most of the synthetic data related works shown in the literature is that it still relies on real data for either fine-tuning, or to act as a target domain. Future work should focus on removing this limitation as much as possible, especially in regards to diffusion models that have already shown themselves to be effective at style transfer with regards to photo realism. It is possible in the future that approaches like ours in chapter 4, will not need a *real* dataset and can simply rely on the diffusion models understand of photorealism to achieve high quality domain adaptation.

8.2.6 Improving acceptance of synthetic data among end users

In addition to making further technical advances in the technology used for the production of synthetic data, further work must also be completed towards improving the rate of acceptance of these technologies especially among biologists. Development of general purpose tools for dataset creation in particular will better enable domain experts to utilise synthetic data without technical expertise. In addition to development of better tools, a greater emphasis on knowledge and data sharing and interdisciplinary collaboration is also needed to better enable the technology showcased in this thesis to make the greatest possible impact.

8.3 Summary

In this work we have presented and evaluated a number of different ways of producing and applying synthetic data to plant phenotyping problems with Computer vision. In doing so we have demonstrated the effectiveness of this approach to training machine learning models and shown that this area of research is a promising area for future research. In our work we have shown that 3D rendering synthetic data is an effective ways of producing data that works well on a wide range of problems, and that diffusion shows potential with further research to eventually surpass it. We showed that for problems that need expensive training data, such as 3D reconstruction and object detection with a large number of objects, that we can produce our own synthetic data much more cheaply and quickly, and in a way that allows our pipelines to be extremely reusable for future projects. We have then shown that synthetic data can then be effectively supplemented, either by using a small target set of real data, or (using Unsupervised Domain Adaptation) using only real images, without the need to create expensive annotations.

Continuing this research in the coming years we hope to see significant progress in the development of synthetic data, as well as wider use of the technology throughout industry. Specifically within the agricultural world the widespread adoption of synthetic data to improve plant analysis and efficiency will aid food security and sustainability throughout the climate crisis. More broadly the capability of deep learning will continue to improve, and with it the need for higher qualities and quantities of data.

Bibliography

[1] Sketchfab - the leading platform for 3d & ar on the web, 2021.

[2] Abu Alhaija, H., Mustikovela, S. K., Mescheder, L., Geiger, A., and Rother, C. Augmented reality meets computer vision: Efficient data generation for urban driving scenes. *International Journal of Computer Vision 126* (2018), 961–972.

[3] Agarwal, B. Detection of plant emergence based on spatio temporal image sequence analysis.

[4] Anagnostopoulou, D., Retsinas, G., Efthymiou, N., Filntisis, P., and Maragos, P. A realistic synthetic mushroom scenes dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2023), pp. 6282–6289.

[5] Arad, B., Balendonck, J., Barth, R., Ben-Shahar, O., Edan, Y., Hellström, T., Hemming, J., Kurtser, P., Ringdahl, O., Tielen, T., et al. Development of a sweet pepper harvesting robot. *Journal of Field Robotics 37*, 6 (2020), 1027–1039.

[6] Arjovsky, M., Chintala, S., and Bottou, L. Wasserstein generative adversarial networks. In *International conference on machine learning* (2017), PMLR, pp. 214–223.

[7] Ayalew, T. W., Ubbens, J. R., and Stavness, I. Unsupervised domain adaptation for plant organ counting. In *Computer Vision–ECCV 2020 Workshops: Glasgow, UK, August 23–28, 2020, Proceedings, Part VI 16*

(2020), Springer, pp. 330–346.

[8] Ayob, N. Z. S., Kamarauzaman, N., Sahriman, N., et al. Data acquisition for 3d surface modelling of chilli plant by using close range photogrammetry for volume estimation. In *2015 IEEE Conference on Systems, Process and Control (ICSPC)* (2015), IEEE, pp. 162–167.

[9] Bansal, A., Borgnia, E., Chu, H.-M., Li, J. S., Kazemi, H., Huang, F., Goldblum, M., Geiping, J., and Goldstein, T. Cold diffusion: Inverting arbitrary image transforms without noise. *arXiv preprint arXiv:2208.09392* (2022).

[10] Barth, R., Hemming, J., and van Henten, E. J. Improved part segmentation performance by optimising realism of synthetic images using cycle generative adversarial networks. *arXiv preprint arXiv:1803.06301* (2018).

[11] Bayraktar, E., Yigit, C. B., and Boyraz, P. A hybrid image dataset toward bridging the gap between real and simulation environments for robotics: Annotated desktop objects real and synthetic images dataset: Adoreset. *Machine Vision and Applications 30*, 1 (2019), 23–40.

[12] Besl, P. J., and McKay, N. D. Method for registration of 3-d shapes. In *Sensor fusion IV: control paradigms and data structures* (1992), vol. 1611, Spie, pp. 586–606.

[13] Bhagat, S., Kokare, M., Haswani, V., Hambarde, P., and Kamble, R. Eff-unet++: A novel architecture for plant leaf segmentation and counting. *Ecological Informatics 68* (2022), 101583.

[14] Blanc, E., Barbillon, P., Lecarpentier, C., Pradal, C., and Enjalbert, J. Functional–structural plant modeling highlights how diversity in leaf dimensions and tillering capability could promote the efficiency of wheat cultivar mixtures. *Frontiers in Plant Science 12* (2021), 734056.

[15] Blender Online Community. *Blender - a 3D modelling and rendering package.* Blender Foundation, Blender Institute, Amsterdam, 2021.

[16] Cabon, Y., Murray, N., and Humenberger, M. Virtual kitti 2. *arXiv preprint arXiv:2001.10773* (2020).

[17] Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., and Zagoruyko, S. End-to-end object detection with transformers. In *European conference on computer vision* (2020), Springer, pp. 213–229.

[18] Chang, A. X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012* (2015).

[19] Chen, L.-C., Papandreou, G., Schroff, F., and Adam, H. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587* (2017).

[20] Cieslak, M., Khan, N., Ferraro, P., Soolanayakanahally, R., Robinson, S. J., Parkin, I., McQuillan, I., and Prusinkiewicz, P. L-system models for image-based phenomics: case studies of maize and canola. *in silico Plants 4*, 1 (2022), diab039.

[21] Croitoru, F.-A., Hondru, V., Ionescu, R. T., and Shah, M. Diffusion models in vision: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2023).

[22] Das Choudhury, S., Bashyam, S., Qiu, Y., Samal, A., and Awada, T. Holistic and component plant phenotyping using temporal image sequence. *Plant methods 14*, 1 (2018), 1–21.

[23] Das Choudhury, S., Samal, A., and Awada, T. Leveraging image analysis for high-throughput plant phenotyping. *Frontiers in plant science 10* (2019), 508.

[24] David, E., Madec, S., Sadeghi-Tehran, P., Aasen, H., Zheng, B., Liu, S., Kirchgessner, N., Ishikawa, G., Nagasawa, K., Badhon, M. A., et al. Global wheat head detection (gwhd) dataset: a large and diverse dataset of high-resolution rgb-labelled images to develop and benchmark wheat head detection methods. *Plant Phenomics* (2020).

[25] de Melo, C. M., Torralba, A., Guibas, L., DiCarlo, J., Chellappa, R., and Hodgins, J. Next-generation deep learning based on simulators and synthetic data. *Trends in cognitive sciences* (2021).

[26] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition* (2009), Ieee, pp. 248–255.

[27] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09* (2009).

[28] Dhariwal, P., and Nichol, A. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems 34* (2021), 8780–8794.

[29] Dobrescu, A., Valerio Giuffrida, M., and Tsaftaris, S. A. Understanding deep neural networks for regression in leaf counting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops* (2019), pp. 0–0.

[30] Dornbusch, T., Lorrain, S., Kuznetsov, D., Fortier, A., Liechti, R., Xenarios, I., and Fankhauser, C. Measuring the diurnal pattern of leaf hyponasty and growth in arabidopsis–a novel phenotyping approach using laser scanning. *Functional Plant Biology 39*, 11 (2012), 860–869.

[31] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929* (2020).

[32] Efros, A. A., and Freeman, W. T. Image quilting for texture synthesis and transfer. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques* (2001), pp. 341–346.

[33] et. al., G. J. ultralytics/yolov5: v6.0 - YOLOv5n 'Nano' models, Roboflow integration, TensorFlow export, OpenCV DNN support, Oct. 2021.

[34] Evers, J., Vos, J., Yin, X., Romero, P., Van Der Putten, P., and Struik, P. Simulation of wheat growth and development based on organ-level photosynthesis and assimilate allocation. *Journal of Experimental Botany 61*, 8 (2010), 2203–2216.

[35] EyeCue Vision Technologies LTD. Qlone.

[36] Feldmann, M. J., and Tabb, A. Cost-effective, high-throughput phenotyping system for 3d reconstruction of fruit form. *The Plant Phenome Journal 5*, 1 (2022), e20029.

[37] Foucher, P., Revollon, P., Vigouroux, B., and Chasseriaux, G. Morphological image analysis for the detection of water stress in potted forsythia. *Biosystems Engineering 89*, 2 (2004), 131–138.

[38] Ganin, Y., and Lempitsky, V. Unsupervised domain adaptation by backpropagation. In *International conference on machine learning* (2015), pp. 1180–1189.

[39] Gardner, M. W., and Dorling, S. Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences. *Atmospheric environment 32*, 14-15 (1998), 2627–2636.

[40] Gatys, L. A., Ecker, A. S., and Bethge, M. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2016), pp. 2414–2423.

[41] Girshick, R. Fast r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (December 2015).

[42] Girshick, R., Donahue, J., Darrell, T., and Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2014), pp. 580–587.

[43] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In *Advances in neural information processing systems* (2014), pp. 2672–2680.

[44] He, K., Gkioxari, G., Dollar, P., and Girshick, R. Mask r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (Oct 2017).

[45] Hinterstoisser, S., Lepetit, V., Wohlhart, P., and Konolige, K. On pre-trained image features and synthetic images for deep learning. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops* (2018), pp. 0–0.

[46] Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems 33* (2020), 6840–6851.

[47] Hochreiter, S., and Schmidhuber, J. Long short-term memory. *Neural computation 9*, 8 (1997), 1735–1780.

[48] Hsu, H.-K., Yao, C.-H., Tsai, Y.-H., Hung, W.-C., Tseng, H.-Y., Singh, M., and Yang, M.-H. Progressive domain adaptation for object detection. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision* (2020), pp. 749–757.

[49] Huang, L., Chen, D., Liu, Y., Shen, Y., Zhao, D., and Zhou, J. Composer: Creative and controllable image synthesis with composable conditions. *arXiv preprint arXiv:2302.09778* (2023).

[50] Isola, P., Zhu, J.-Y., Zhou, T., and Efros, A. A. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2017), pp. 1125–1134.

[51] Itseez. Open source computer vision library. https://github.com/itseez/opencv, 2015.

[52] Jackson, A. S., Bulat, A., Argyriou, V., and Tzimiropoulos, G. Large pose 3d face reconstruction from a single image via direct volumetric cnn regression. In *Proceedings of the IEEE International Conference on Computer Vision* (2017), pp. 1031–1039.

[53] Jackson, A. S., Manafas, C., and Tzimiropoulos, G. 3d human body reconstruction from a single image via volumetric regression. In *Proceedings of the European Conference on Computer Vision (ECCV)* (2018), pp. 0–0.

[54] Jadhav, T., Singh, K., and Abhyankar, A. Volumetric estimation using 3d reconstruction method for grading of fruits. *Multimedia Tools and Applications 78* (2019), 1613–1634.

[55] Jafari, M., Francis, S., Garibaldi, J. M., and Chen, X. Lmisa: A lightweight multi-modality image segmentation network via domain adaptation using gradient magnitude and shape constraint. *Medical Image Analysis 81* (2022), 102536.

[56] Johnson, J., Alahi, A., and Fei-Fei, L. Perceptual losses for real-time style transfer and super-resolution. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14* (2016), Springer, pp. 694–711.

[57] Kazmi, W., Foix, S., and Alenya, G. Plant leaf imaging using time of flight camera under sunlight, shadow and room conditions. In *2012 IEEE International Symposium on Robotic and Sensors Environments Proceedings* (2012), IEEE, pp. 192–197.

[58] Kingma, D. P., and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[59] Kuhn, H. W. The hungarian method for the assignment problem. *Naval research logistics quarterly 2*, 1-2 (1955), 83–97.

[60] Kuznichov, D., Zvirin, A., Honen, Y., and Kimmel, R. Data augmentation for leaf segmentation and counting tasks in rosette plants. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops* (2019), pp. 0–0.

[61] LeCun, Y., Boser, B., Denker, J., Henderson, D., Howard, R., Hubbard, W., and Jackel, L. Handwritten digit recognition with a back-propagation network. *Advances in neural information processing systems 2* (1989).

[62] Leitner, D., Klepsch, S., Bodner, G., and Schnepf, A. A dynamic root system growth model based on l-systems: Tropisms and coupling to nutrient uptake from soil. *Plant and soil 332* (2010), 177–192.

[63] Li, L., Zhang, Q., and Huang, D. A review of imaging techniques for plant phenotyping. *Sensors 14*, 11 (2014), 20078–20111.

[64] Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. Microsoft coco: Common objects in context. In *European conference on computer vision* (2014), Springer, pp. 740–755.

[65] Lindenmayer, A. Mathematical models for cellular interactions in development ii. simple and branching filaments with two-sided inputs. *Journal of theoretical biology 18*, 3 (1968), 300–315.

[66] Liu, X., Yoo, C., Xing, F., Oh, H., El Fakhri, G., Kang, J.-W., Woo, J., et al. Deep unsupervised domain adaptation: A review of recent advances and perspectives. *APSIPA Transactions on Signal and Information Processing 11*, 1 (2022).

[67] Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., and Guo, B. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision* (2021), pp. 10012–10022.

[68] Long, M., Zhu, H., Wang, J., and Jordan, M. I. Deep transfer learning with joint adaptation networks. In *International conference on machine learning* (2017), PMLR, pp. 2208–2217.

[69] Mayer, N., Ilg, E., Fischer, P., Hazirbas, C., Cremers, D., Dosovitskiy, A., and Brox, T. What makes good synthetic training data for learning disparity and optical flow estimation? *International Journal of Computer Vision 126* (2018), 942–960.

[70] Mazen, F. M., and Nashat, A. A. Ripeness classification of bananas using an artificial neural network. *Arabian Journal for Science and Engineering 44*, 8 (2019), 6901–6910.

[71] Mei, J., Sun, K., and Luo, G. Cross-domain leaf counting with minimizing feature distances. In *2022 7th International Conference on Image, Vision and Computing (ICIVC)* (2022), IEEE, pp. 812–817.

[72] Minervini, M., Fischbach, A., Scharr, H., and Tsaftaris, S. A. Finely-grained annotated datasets for image-based plant phenotyping. *Pattern recognition letters 81* (2016), 80–89.

[73] Minervini, M., Scharr, H., and Tsaftaris, S. A. Image analysis: the new bottleneck in plant phenotyping [applications corner]. *IEEE signal processing magazine 32*, 4 (2015), 126–131.

[74] Mirza, M., and Osindero, S. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784* (2014).

[75] Mueller, F., Bernard, F., Sotnychenko, O., Mehta, D., Sridhar, S., Casas, D., and Theobalt, C. Ganerated hands for real-time 3d hand tracking from monocular rgb. In *Proceedings of the IEEE Conference*

on Computer Vision and Pattern Recognition (2018), pp. 49–59.

[76] Najafian, K., Ghanbari, A., Stavness, I., Jin, L., Shirdel, G. H., and Maleki, F. A semi-self-supervised learning approach for wheat head detection using extremely small number of labeled samples. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2021), pp. 1342–1351.

[77] Newell, A., Yang, K., and Deng, J. Stacked hourglass networks for human pose estimation. In *Computer Vision – ECCV 2016* (Cham, 2016), B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds., Springer International Publishing, pp. 483–499.

[78] Nichol, A., Dhariwal, P., Ramesh, A., Shyam, P., Mishkin, P., McGrew, B., Sutskever, I., and Chen, M. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741* (2021).

[79] Nowruzi, F. E., Kapoor, P., Kolhatkar, D., Hassanat, F. A., Laganiere, R., and Rebut, J. How much real data do we actually need: Analyzing object detection performance using synthetic and real data. *arXiv preprint arXiv:1907.07061* (2019).

[80] Park, M., Yun, J., Choi, S., and Choo, J. Learning to generate semantic layouts for higher text-image correspondence in text-to-image synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2023), pp. 7591–7600.

[81] Park, T., Efros, A. A., Zhang, R., and Zhu, J.-Y. Contrastive learning for unpaired image-to-image translation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IX 16* (2020), Springer, pp. 319–345.

[82] Patil, V. M. K. T. S. R. P. C. K. Fruitsgb: Top indian fruits with quality, 2020.

[83] Prusinkiewicz, P. Graphical applications of l-systems. In *Proceedings of graphics interface* (1986), vol. 86, pp. 247–253.

[84] Prusinkiewicz, P., and Lindenmayer, A. *The algorithmic beauty of plants.* Springer Science & Business Media, 2012.

[85] Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning* (2021), PMLR, pp. 8748–8763.

[86] Radford, A., Metz, L., and Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434* (2015).

[87] Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., and Chen, M. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125* (2022).

[88] Ren, S., He, K., Girshick, R., and Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems* (2015), C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, Eds., vol. 28, Curran Associates, Inc.

[89] Ren, S., He, K., Girshick, R., and Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems* (2015), pp. 91–99.

[90] Richter, S. R., Vineet, V., Roth, S., and Koltun, V. Playing for data: Ground truth from computer games. In *European conference on computer vision* (2016), Springer, pp. 102–118.

[91] Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. High-resolution image synthesis with latent diffusion models. In *Pro-*

ceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2022), pp. 10684–10695.

[92] Ronneberger, O., Fischer, P., and Brox, T. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention* (2015), Springer, pp. 234–241.

[93] Ros, G., Sellart, L., Materzynska, J., Vazquez, D., and Lopez, A. M. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2016), pp. 3234–3243.

[94] Rose, J. C., Paulus, S., and Kuhlmann, H. Accuracy analysis of a multi-view stereo approach for phenotyping of tomato plants at the organ level. *Sensors 15*, 5 (2015), 9651–9665.

[95] Rumelhart, D. E., Hinton, G. E., and Williams, R. J. Learning internal representations by error propagation. Tech. rep., California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

[96] Rumelhart, D. E., Hinton, G. E., and Williams, R. J. Learning representations by back-propagating errors. *nature 323*, 6088 (1986), 533–536.

[97] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al. Imagenet large scale visual recognition challenge. *International journal of computer vision 115* (2015), 211–252.

[98] Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E., Ghasemipour, S. K. S., Ayan, B. K., Mahdavi, S. S., Lopes, R. G., et al. Photorealistic text-to-image diffusion models with deep language understanding. *arXiv preprint arXiv:2205.11487* (2022).

[99] Sapoukhina, N., Samiei, S., Rasti, P., and Rousseau, D. Data augmentation from rgb to chlorophyll fluorescence imaging application to leaf segmentation of arabidopsis thaliana from top view images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops* (2019), pp. 0–0.

[100] Shrivastava, A., Pfister, T., Tuzel, O., Susskind, J., Wang, W., and Webb, R. Learning from simulated and unsupervised images through adversarial training. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2017), pp. 2107–2116.

[101] Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning* (2015), PMLR, pp. 2256–2265.

[102] Stöckl, A. Evaluating a synthetic image dataset generated with stable diffusion. *arXiv preprint arXiv:2211.01777* (2022).

[103] Sun, B., Feng, J., and Saenko, K. Return of frustratingly easy domain adaptation. In *Proceedings of the AAAI conference on artificial intelligence* (2016), vol. 30.

[104] Sun, B., and Saenko, K. Deep coral: Correlation alignment for deep domain adaptation. In *Computer Vision–ECCV 2016 Workshops: Amsterdam, The Netherlands, October 8-10 and 15-16, 2016, Proceedings, Part III 14* (2016), Springer, pp. 443–450.

[105] Sun, C., Shrivastava, A., Singh, S., and Gupta, A. Revisiting unreasonable effectiveness of data in deep learning era. In *Proceedings of the IEEE international conference on computer vision* (2017), pp. 843–852.

[106] The HDF Group. Hierarchical Data Format, version 5.

[107] Tremblay, J., To, T., and Birchfield, S. Falling things: A synthetic dataset for 3d object detection and pose estimation. In *Proceedings of the*

IEEE Conference on Computer Vision and Pattern Recognition Workshops (2018), pp. 2038–2041.

[108] Ubbens, J., Cieslak, M., Prusinkiewicz, P., and Stavness, I. The use of plant models in deep learning: an application to leaf counting in rosette plants. *Plant methods 14*, 1 (2018), 6.

[109] Ubbens, J. R., and Stavness, I. Deep plant phenomics: a deep learning platform for complex plant phenotyping tasks. *Frontiers in plant science 8* (2017), 1190.

[110] Valerio Giuffrida, M., Scharr, H., and Tsaftaris, S. A. Arigan: Synthetic arabidopsis plants using generative adversarial network. In *Proceedings of the IEEE International Conference on Computer Vision Workshops* (2017), pp. 2064–2071.

[111] Van Der Meer, M., De Visser, P. H., Heuvelink, E., and Marcelis, L. F. Row orientation affects the uniformity of light absorption, but hardly affects crop photosynthesis in hedgerow tomato crops. *in silico Plants 3*, 2 (2021), diab025.

[112] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems 30* (2017).

[113] Wang, C.-Y., Bochkovskiy, A., and Liao, H.-Y. M. Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. *arXiv preprint arXiv:2207.02696* (2022).

[114] Wang, Q., Gao, J., Lin, W., and Yuan, Y. Learning from synthetic data for crowd counting in the wild. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (2019), pp. 8198–8207.

[115] Wang, X., Girshick, R., Gupta, A., and He, K. Non-local neural networks. In *Proceedings of the IEEE conference on computer vision and*

pattern recognition (2018), pp. 7794–7803.

[116] Ward, D., and Moghadam, P. Synthetic arabidopsis dataset. In *CSIRO. Data Collection.* (2018).

[117] Ward, D., and Moghadam, P. Scalable learning for bridging the species gap in image-based plant phenotyping. *Computer Vision and Image Understanding 197* (2020), 103009.

[118] Ward, D., Moghadam, P., and Hudson, N. Deep leaf segmentation using synthetic data. *arXiv preprint arXiv:1807.10931* (2018).

[119] Wu, H., Xiao, B., Codella, N., Liu, M., Dai, X., Yuan, L., and Zhang, L. Cvt: Introducing convolutions to vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2021), pp. 22–31.

[120] Wu, W., Zhao, Y., Shou, M. Z., Zhou, H., and Shen, C. Diffumask: Synthesizing images with pixel-level annotations for semantic segmentation using diffusion models. *arXiv preprint arXiv:2303.11681* (2023).

[121] Xie, J., Li, Y., Huang, Y., Liu, H., Zhang, W., Zheng, Y., and Shou, M. Z. Boxdiff: Text-to-image synthesis with training-free box-constrained diffusion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2023), pp. 7452–7461.

[122] Zhang, W., Wang, H., Zhou, G., and Yan, G. Corn 3d reconstruction with photogrammetry. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences 330* (2008).

[123] Zhu, J.-Y., Park, T., Isola, P., and Efros, A. A. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision* (2017), pp. 2223–2232.

[124] Zhu, Y., Aoun, M., Krijn, M., Vanschoren, J., and Campus, H. T. Data augmentation using conditional generative adversarial networks for

leaf counting in arabidopsis plants. In *BMVC* (2018), p. 324.