Ollscoil Chathair
Bhaile Átha Cliath
Dublin City University

Deep Learning for Semantic Segmentation of Agricultural Imagery

Robert McConnell

Bachelor of Engineering
in
Electronic and Computer Engineering

Supervised by Prof. Noel O'Connor
2018/19

Acknowledgements

I would like to thank my supervisor, Prof. Noel O'Connor for his continued support and guidance throughout the project and in the field of Deep Learning.

During our weekly project meetings, Dr. Kevin McGuinness and Dr. Haolin Wei gave a huge amount of their time and expertise to help with any issues that came up over the course of the project. I would like to thank them for all of the help that they provided.

I would also like to thank the Insight Centre for Data Analytics for providing the GPU resources which resulted in invaluable performance gain and saved time while working on my project and to Dr. Haolin Wei for setting up access to the GPU and providing assistance with queries related to its usage.

DCU School of Electronic Engineering
Assignment Submission

Student Name:	Robert McConnell
Student Number:	14458218
Programme:	Electronic and Computer Engineering
Project Title:	Deep Learning for Semantic Segmentation of Agricultural Imagery
Lecturer:	Prof. Noel O'Connor
Project Due Date:	08/04/2019

Declaration

I declare that this material, which I now submit for assessment, is entirely my own work and has not been taken from the work of others, save and to the extent that such work has been cited and acknowledged within the text of my work. I understand that plagiarism, collusion, and copying is a grave and serious offence in the university and accept the penalties that would be imposed should I engage in plagiarism, collusion, or copying. I have read and understood the Assignment Regulations set out in the module documentation. I have identified and included the source of all facts, ideas, opinions, viewpoints of others in the assignment references. Direct quotations from books, journal articles, internet sources, module text, or any other source whatsoever are acknowledged and the source cited are identified in the assignment references.

I have not copied or paraphrased an extract of any length from any source without identifying the source and using quotation marks as appropriate. Any images, audio recordings, video or other materials have likewise been originated and produced by me or are fully acknowledged and identified.

This assignment, or any part of it, has not been previously submitted by me or any other person for assessment on this or any other course of study. I have read and understood the referencing guidelines found at http://www.library.dcu.ie/citingrefguide08.pdf and/or recommended in the assignment guidelines.

I understand that I may be required to discuss with the module lecturer/s the contents of this submission.

I/me/my incorporates we/us/our in the case of group work, which is signed by all of us.

Signed:

Robert McConnell

Abstract

With increasing demands on food production, Artificial Intelligence is being used in the Agricultural domain to automate visual inspection tasks and to perform robotic harvesting. Semantic Segmentation is a specific Deep Learning problem which involves labelling regions by assigning every pixel in an image to a meaningful class. The difficulty in performing Semantic Segmentation in an agricultural setting is that it can be difficult for a Convolutional Neural Network to cater for all environmental conditions such as change in seasons and weather along with the large amount of variance between crops. This project aims to address this issue by using Neural Style Transfer techniques to augment the dataset and improve generalisation.

Contents

List of Figures

Chapter 1

Introduction

A 2009 report by the Food and Agriculture Organisation of the United Nations[6] shows the need for a large increase in the production of food as the world's population is estimated to increase by 34 percent by 2050 all while the land available for food production decreases. This is leading to a new agricultural environment focused around efficiency of food production using 'smart farming' and 'ag-tech' where food is produced in greenhouses that require accurate monitoring, measuring and analysis of the plants being grown. This has led to the area of image analysis becoming critical to the progress of the future agricultural domain where image identification and classification is necessary for areas like robotic harvesting and crop management. The aim is to add a degree of artificial intelligence to enable increased selectivity, precision and robustness.

1.1 Problem and Boundaries

The original proposal of this project was to identify an appropriate baseline approach to semantic segmentation following the capsicum annuum (bell pepper) paper[1] and to investigate its applicability in agricultural settings. As the aim was to investigate current research, the problem and its boundaries needed to be defined before progressing with the implementation.

1.1.1 Deep Learning in Agriculture

A survey on deep learning in agriculture published in the 'Computers and Electronics in Agriculture' journal[7] in 2018 outlined the current status of the field. The survey showed that deep learning has begun to be applied in the area of Agriculture with most of the research in this area appearing since 2015 and with more papers being written each year. The survey studied comparisons between 40 research efforts in an attempt to determine if deep learning performed better than other existing techniques in respect to classification performance. The findings indicated that deep learning provided high accuracy and outperformed existing commonly used image processing techniques. While fields such as facial recognition, language translation and autonomous vehicles have been the target of many research papers, there has been difficulty in translating the models into the area of Agriculture. The main bottleneck has been computer vision performance. The challenges of computer vision in the Agricultural domain stem from the high amount of variation within object classes and changing environmental conditions throughout the day and seasons. Papers such as Counting Apples and Oranges with Deep Learning [8] attempt to use difficult datasets with high levels of occlusions, depth variation and uncontrolled illumination to create methods of combatting the inherent difficulty with segmenting Agricultural produce. The life cycle of a fruit or vegetable can change rapidly day by day and it can be difficult to define the stages of growth. This can require experts to annotate training images, who may be difficult to involve.
A considerable drawback and barrier in the use of Deep Learning is the need of large datasets to serve dur-

ing the training. The problem with many available datasets is the low variation among the different classes and issues with lighting, changing environmental conditions, occlusions, plants over-lapping and clustering. One approach is to train the models with synthetically generated images. This approach was tested in the canonical paper of this project by Barth et al.[1], which involved the creation of a synthetic dataset of bell pepper images.

1.1.2 Semantic Segmentation

This project addresses a specific deep learning problem known as semantic segmentation. Semantic Segmentation (SS) involves labelling regions by assigning every single pixel in an image to a meaningful class. Figure 1.1. shows examples of SS. The images on the left are the input images, the images on the right show the ground-truth and the middle images are the segmented results. The ground truth is a hand-annotated image which assigns a class label to each pixel in an image and is the ideal predicted output. The aim of the deep learning model is to find the class labels that match the ground truth with a high degree of accuracy. In traditional SS, each class in an image is of equal importance and should be segmented as accurately as possible. However, as argued in Importance-Aware Semantic Segmentation for Autonomous Vehicles [9], this may not be the best approach where the speed and accuracy of segmentation is critical, such as the case with Autonomous Vehicles. The paper introduces the concept of Importance-Aware Loss where objects can be emphasised as critical. This is possibly something that could be explored in the future relating to Agriculture where for example a fruit harvesting machine may need to quickly and accurately determine the location and size of a fruit without focusing on leaves in the background that are not critical for operation. In this project each class is of equal importance.

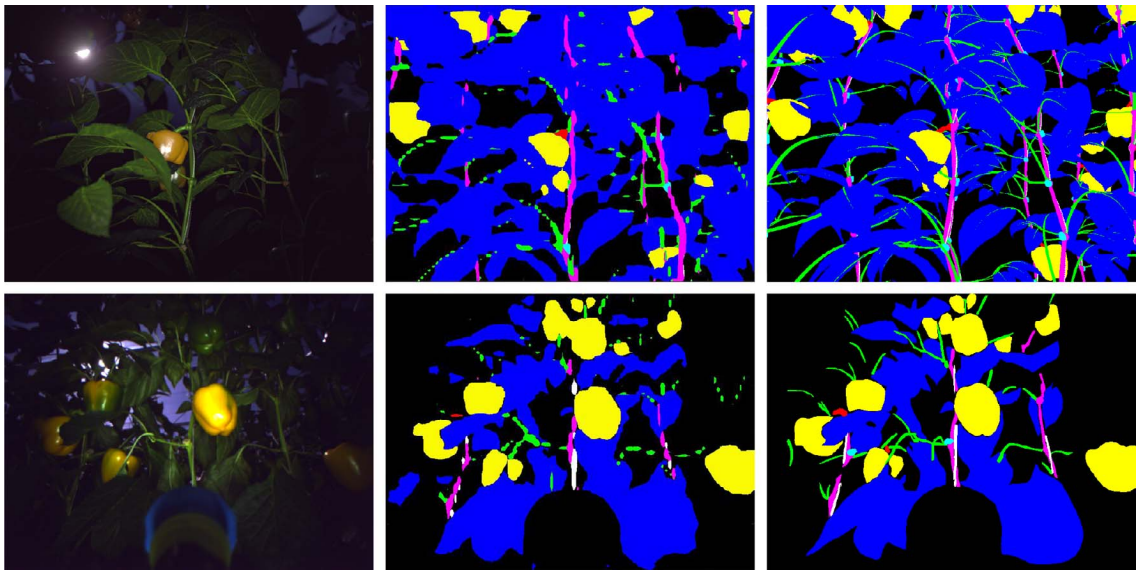


Figure 1.1: Semantic Segmentation Results from Barth et al. Paper [1].

1.1.3 Proposed Hypothesis

The baseline for this project is to match the results achieved in the capsicum annuum paper[1]. Once this baseline model is developed, experiments can be run to determine the models ability to detect bell peppers in images from an external dataset, ImageNet[10], which were taken in varying environmental conditions. The hypothesis of this paper is that by applying style transfer as a data augmentation technique to the original dataset, the key issue of deep learning in agriculture can begin to be addressed by resolving the issue

of changing environmental conditions. To test the hypothesis, the original dataset will be modified using neural style transfer and then tested using the ImageNet dataset to see if the retrained model will perform better than the baseline approach.

1.2 Programming Languages and Frameworks

1.2.1 Python

Python[11] is an interpreted, high-level, general-purpose programming language that has a design philosophy focused on code readability. Python is one of the fastest growing programming languages and has been a goto for fast prototyping. This makes it the ideal choice for a rapidly growing area of research where code is shared with the hope of being improved and built on, i.e. machine learning. The syntax for Python is very readable which can reduce the need for as much documentation. Python is also advantageous when developing new solutions as it is easy to leverage the open-source work of others. Python also has a large array of machine learning libraries. An example of one used in this project is NumPy. NumPy adds support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. There are two widely used versions of Python, version 2.7 and 3.x. Version 3.x is currently in development but the language was cleaned up since version 2.7 with less focus on backwards compatibility. While the versions are similar, there are some aspects of each that can break the other. Python 2.7 was used in this project due to the models that would be used.

1.2.2 VirtualEnv

The main purpose of a Python virtual environment (VirtualEnv[12]) is to create an isolated environment for Python projects. It is a way of keeping dependencies specific to a project separate from other projects. It is especially useful when working on a shared server where other users can affect versions and libraries. During the early part of this project, there were recurring issues with conflicting versions of different Python libraries and dependencies. This consumed large amounts of times and the issues tended to be difficult to trace and resolve. Creating a virtual environment for the project resolved the recurring issues and allowed focus to be diverted to the project implementation.

1.2.3 TensorFlow

The TensorFlow[13] framework is a free and open-source software library developed by the Google Brain team for internal use in Google but was later released under the Apache 2.0 open-source licence. It is used for building and deploying machine learning models and there are a large number of models that have been written and developed using it. The motivation for using TensorFlow over other frameworks such as PyTorch or Keras was that the DeepLab model that was used for the baseline experimentation had a very active repository on GitHub. The framework was difficult to work with at times but most problems that arose had already been brought up on the issues section of the repository and in Stack Overflow.

1.2.4 Jupyter Notebooks and Google Colaboratory

A Jupyter Notebook is a document which contains a combination of Python code and rich text elements, including figures. Notebooks are human-readable documents as well as being executable. They are web-based interactive environments which are an excellent tool for prototyping and testing small portions of code. A large portion of the initial learning and research was performed using Jupyter notebooks due to the simplicity and visual feedback.
Google Colaboratory started as a part of Project Jupyter, but the development was eventually taken over by Google. Colab is a Jupyter notebook stored in Google Drive which connects the notebook to a cloud-based

runtime allowing for Python code to be executed without any hardware setup. Colab has GPU acceleration and while it is not as powerful as a server with a dedicated GPU, it is more powerful than most personal computers. For these reasons Colab was used during the initial development of the baseline. Something to note is that Colab runtime sessions are limited to 12 hours after which the session will be reset and all data will be lost. If using Colab for a task that takes longer than 12 hours it is necessary to create a way of saving the model as an exportable checkpoint so that it can be restored from without data and progress loss. A task that Colab notebooks are very useful for is model demos using frozen inference models. This project has an associated demo that is briefly discussed at the end of the report.

1.2.5 GPU Server

The computation involved in a Neural Network consists of very large amounts of matrix convolutions and multiplications. This type of computing performs far better on Graphics Processing Units compared to CPUs [14]. The GPU used for training the model was a Nvidia GeForce GTX 1080 Ti hosted on a server in the Insight Centre for Data Analytics in DCU. To access the server off-site, a gateway was needed. This worked well but made it difficult to use some features such as Jupyter notebooks and TensorBoard as it would have required a relatively complex ssh tunnelling setup.

1.3 Report Structure

This report is written with the aim of being understood without any previous background in machine learning or deep learning. To help with this, the next chapter focuses on building a technical background for understanding later parts of the report. Due to the nature of the project, there was a large amount of research performed, most of which is outlined in Chapter 2 along with a review of some key literature which formed the basis of this project. As a result it is one of the largest components of the report. Chapter 3 will discuss the design and implementation of the proposed solution along with an explanation of the reasoning behind the design decisions. Following this there is a chapter on the experiments performed to test the hypothesis and the results from these experiments. The discussion and impact of the results will be investigated in Chapter 5 which also addresses some of the shortcomings of the project and cases where the implementation failed. The next chapter will address the ethical issues of the project with a focus on issues that directly relate to the project but also addressing some broader ethical issues in artificial intelligence and machine learning. Finally Chapter 7 will conclude the report and discuss how the project could be taken further with additional research. All cited work is referenced in the reference section using IEEE standards and the code developed over the course of the project is outlined in the Appendix.

1.4 Summary

In this chapter the project was introduced along with the proposed hypothesis and some context and motivation. The development environment was also discussed with some terminology that may not make sense until after the next chapter. In the next chapter, some background information will be provided to better understand the project. It also contains some of the key literature that the project is based on and the datasets that were used.

Chapter 2

Technical Background and Literature Review

Machine learning is a branch of Artificial Intelligence that studies algorithms and statistical models which perform a task by relying on patterns and inference instead of the traditional approach of using explicit instructions. Deep Learning is a subset of machine learning based on learning features in a way similar to humans using large amounts of data and Deep Neural Networks modelled on the human brain to learn features.
This chapter looks at some of the technical background for this area and literature reviews of several papers that are relevant to addressing the proposed problem.

2.1 Deep Learning

An Artificial Neural Network (ANN) is a system that consists of a collection of connected nodes called Artificial Neurons, where each connection (Synapse) can transmit a signal from one Artificial Neuron to another. If a neuron receives a signal, it can be processed and signal additional neurons that are connected to it. In Deep Learning, the models contain many layers of these interconnected neurons and with each layer information is extracted. The power of the network comes from its ability to learn representations in the data that it is provided with and how to best relate this to a predicted output. The ability to predict comes from the multi-layered structure of the networks which is what gives deep learning its name. The hierarchical structure can learn to represent features at different scales and combine them to get higher-order features. Neural Networks are a field of study that is rapidly developing with future areas of improvement involving considering the costs of activating each neuron, and to actively perceive patterns by directing attention to relevant parts of data in the same way that humans do [15].

2.1.1 Neural Networks

A multilayer perceptron is a simple example of a neural network. It is sometimes referred to as a 'vanilla' neural network as it contains the minimum requirements for a neural network, uses backpropagation for training and does not contain features such as feedback connections as seen in recurrent neural networks. It consists of at least three layers of nodes, an input layer, a hidden layer and an output layer. Figure 2.1. shows a multilayer perceptron.
All of the nodes in the network with the exception of the input layer are known as artificial neurons. A neuron is a function that takes in an input and returns a value known as its activation. The neuron is 'lit up' when its activation is above a certain value. For example the first layer in a neural network could be an image where each pixel in the image is a neuron and each greyscale value is its activation.

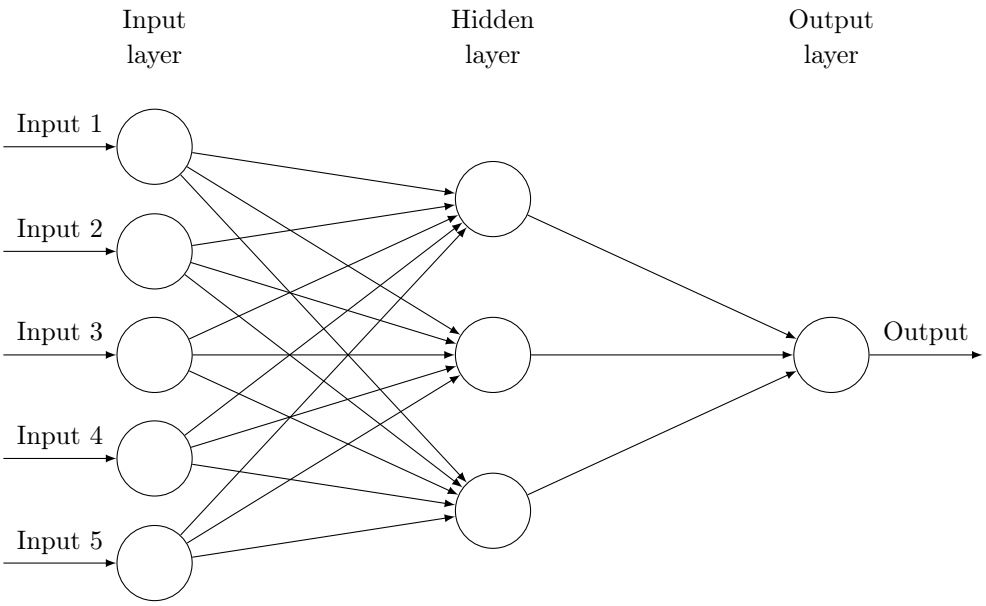


Figure 2.1: Neural Network Graph showing Input, Output and Hidden Layers

Neural networks contain hidden layers where activations in one layer determine the activations in the next layer and the core of the network comes down to how this occurs. This is loosely analogous to how in the brain some neurons firing cause other neurons to fire. The hidden layer values are determined using an activation function. Each connection between neurons has a weight applied to its input. The weighted input is then passed through an activation function. The activation function is used to determine the threshold at which the neuron is activated and the strength of the output signal. An example of an activation function is the Sigmoid function, which is a non-linear function with an output between 0 and 1. This is especially used for models where the output is a predicted probability as probabilities exist in the range of 0 to 1. The deep learning models that have been implemented in this project do not use the sigmoid function but instead use a function that has been shown to provide better results and is the currently the most popular activation function, the Rectified Linear Unit (ReLU). This activation function will be discussed later in the report.

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

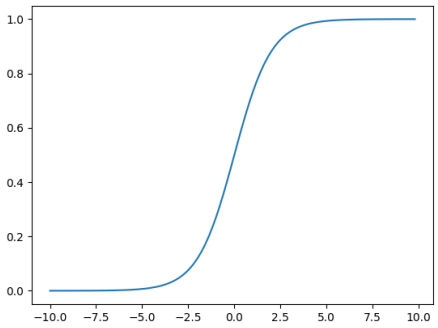


Figure 2.2: Sigmoid Function created in matplotlib

2.1.2 Training Neural Networks

Learning can simply be defined as finding the right weights and biases to solve the problem. To train a model, it needs to be provided with data and information defining what the data is. For example, when training a model to perform semantic segmentation, images are fed into the network along with a 'ground-truth' that defines the class that each pixel in the image belongs to. The hope is that once the model has been trained, it can generalise to images that are not in the training data. To test this, the model is provided with data that it has not seen before and tested on how accurately it segments the images.

2.1.3 Convolutional Neural Networks

A Convolutional Neural Network (CNN) is a class of deep, feed-forward ANNs. It is inspired by the visual cortex of an animal in the way that the connectivity pattern between neurons in organised. Individual neurons respond to stimuli only in a restricted region, known as the receptive field. They consist of an input and output layer along with multiple hidden layers, typically consisting of convolutional layers, pooling layers and fully connected layers. There is a large body of literature showing that CNNs perform better than previous methods in the task of image segmentation. One particular paper by Long et al. [4] introduced an approach of using fully-connected CNNs, trained end-to-end, pixels-to-pixels. This was shown to exceed previous best results and produces an output of size corresponding to the inputs spacial dimensions. "Semantic segmentation faces an inherent tension between semantics and location: global information resolves what while local information resolves where... Combining fine layers and coarse layers lets the model make local predictions that respect global structure." Long et al. [4].
The objective of the Convolution Operation is to extract the high-level features such as edges, from the input image. 'Off-the-shelf' Convolutional Neural Networks can achieve highly precise results [16], however, a smaller effective stride is necessary to classify the finer details of an image [17]. They are also very susceptible to producing poor results when small datasets with low amounts of variation are used [18].

Convolution Operation

To better explain some of the ways that different research papers have modified the approach to convolutional layers it is necessary to talk about the parameters that define them. The first parameter is the *kernel size* which defines the field of view of the convolution. The kernel or filter is the first part of a convolutional layer which is involved in carrying out the convolution operation. The kernel moves throughout the image from left to right until it parses the complete width before moving to the leftmost size of the image on the next line and repeats the process until the image has been traversed. This way of moving through an image is known as a raster scan. If an image has multiple colour channels, the kernel will have the same depth of the input image.
The next parameter is known as the *stride*. The stride defines the step size that the kernel moves as it traverses the image. This is usually set to 1 where each pixel will be addressed but can have other values such as 2 for downsampling an image.
Padding defines the way in which the border of the image is addressed. This can prevent an issue know as edge effects where the kernel is passing over a pixel on the border of the image and does not have enough information to calculate a result. To avoid this a padded border can be added which extends the size of the image and hence keeps the spacial output dimensions equal to the output.

2.1.4 VGG-16 Architecture

VGG-16 is a convolutional neural network architecture developed by the Visual Geometry Group from Oxford. The model contains 16 weight layers and Figure 2.3. illustrates the architecture. The model comes from a paper entitled 'Very Deep Convolutional Networks for Large-Scale Image Recognition' and has achieved very high accuracy in ImageNet.

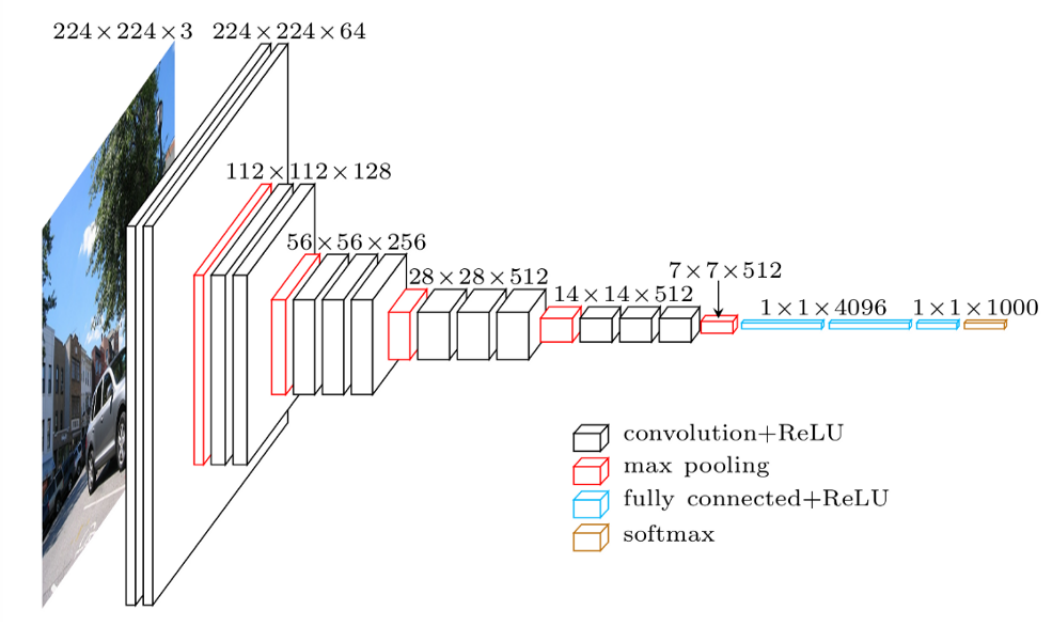


Figure 2.3: VGG-16 Macroarchitecture [2]

2.2 Datasets

2.2.1 Data synthesis methods for semantic segmentation in agriculture: A Capsicum annuum dataset

This paper written by R. Barth, J. IJsselmuiden, J. Hemming and E.J. Henten formed both the baseline for the proposed project and the dataset. The paper provides synthesis methods for generating a large-scale synthetic dataset of agricultural scenes for semantic image segmentation. The ultimate goal in mind is to add AI to harvesting and crop management systems to increase selectivity, precision and robustness. Historically the bottleneck for this goal has been computer vision performance. Large annotated agricultural datasets are infeasible to obtain manually. The aim of the paper is the generation of large-scale semantic segmentation datasets on a plant-part level. This allows 'pre-training' of computer vision models. The scope of the paper is primarily synthetic image generation methodology. The challenges of computer vision in the agricultural domain come from the large variation within object classes and changing environment conditions. This is the issue that this project attempts to address and an approach can be seen in the following chapters. Some computer vision tasks require per pixel labelling of the image (semantic segmentation). Specifically for localisation in robotics for harvesting, disease detection and phenotyping. i.e weakly labelling data with bounding boxes will not suffice. Convolutional neural networks for image classification and segmentation has further increased the training dataset size requirement. Models require a large number of distinct data samples for the optimisation to converge properly without overfitting occurring.
A plant can be modelled functionally, structurally or both. Functional plant models represent the interaction of internal and external plant processes. On the other hand, structural plant models focus solely on physical appearance. The plant models and object meshes used in this paper were made using the commercial software PlantFactory 2015 Studio. While intended for game and video production, it produces realistic plant models with the ability to randomly generate plant instances which can be exported as mesh models. To simulate fruit maturity levels, a colour gradient was projected on a noise texture. This ranged from unripe green to ripe yellow. To virtually collect data, a simulated camera was added with similar optical properties as the hardware that would be used. The colours were replaced by greyscale values in a single channel to

reduce the label size by 98%. In the paper, 5 experiments were run with semantic segmentation deep learning networks using the DeepLab framework. For each experiment overfitting was prevented by selecting the optimal model by periodically checking the model's performance on the separate validation set. The Jaccard Index Similarity Coefficient was used as an evaluation measure to calculate performance. This index is also known as the intersection-over-union (IOU) and is widely used for semantic segmentation evaluation.
After generating a synthetic image dataset of 10,500 images, 5 experiments were presented to determine if the hypothesis that such datasets for agriculture are a valid and valuable tool for computer vision learning methods. Some computer vision and learning methods are sensitive to object colour, affecting the generalisation of the method to new images with different colour distributions.
Although modelling can be time consuming, it facilitates the generation of large-scale, detailed datasets under a broader set of conditions. This paper presented an example for a single point in time for a single variety of Capsicum annum. The approach taken in this paper is generic and applicable to any crop, during multiple stages of growth. It must be noted that human perceptual evaluation of images often employs sensory completion to make up for differences or absences. It was suggested to include intra plant part pose variations and deformations in forthcoming research. Within a dataset, label frequencies are often highly unbalanced. To counter that effect, object occurrence statistics can be used for normalisation. The model in this paper did not take intra-plant properties into account. From the results, it was concluded that the increase in performance using synthetic data bootstrapping compared to the other approaches might be caused by the increased training sample number with high similarity that was made available to the CNN. The results present a starting point to determine how synthetic data can be used to improve segmentation performance. Segmentation results show a promising next step for semantic part localisation in agricultural.

2.2.2 PASCAL VOC 2012

The Pascal Visual Object Classes (Pascal VOC)[19] challenge consists of two components, a publicly available dataset of images with corresponding ground truth annotations and an annual competition and workshop. There is also a pre-trained checkpoint available of the model trained on the DeepLab architecture which was used in one of the project experiments.

2.2.3 ImageNet Large Scale Visual Recognition Challenge

The ImageNet[10] is a very large database of classified images. While the images are not labelled for use in semantic segmentation training, the dataset contains collections of images from different categories. The bell pepper category was used in the later parts of the paper as an way of measuring the models ability to generalise to unseen data from outside the training dataset.

2.3 Literature

2.3.1 Deep Learning by Ian Goodfellow, Yoshua Bengio and Aaron Courville

Deep Learning by Goodfellow et al., published as part of MIT's Adaptive Computation and Machine Learning series is written by several of the fields leading specialists. It is one of the first books to cover the field of deep learning in depth while still remaining accessible to newcomers. While the book is challenging in parts and focuses on the mathematical and theoretical side of deep learning as opposed to a more practical method using code examples, it provided information about many of the aspects covered in this project. The book is split into three sections, the first section is 'Applied Math and Machine Learning Basics'. This section helped to build a solid foundation of background material by introducing linear algebra, probability and optimisation in the context of deep learning along with the basic concepts of machine learning. The next section, 'Deep Networks: Modern Practises', helped with the learning about the deep learning necessary to complete this project successfully. It covers the most popular and important methods currently in use such as feedforward networks and convolutional networks in a theoretical manner. The final section, 'Deep Learning Research', is about the future of deep learning and goes through some of the promising techniques currently being developed such as generative adversarial networks which could be harnessed as part of future research in this project.

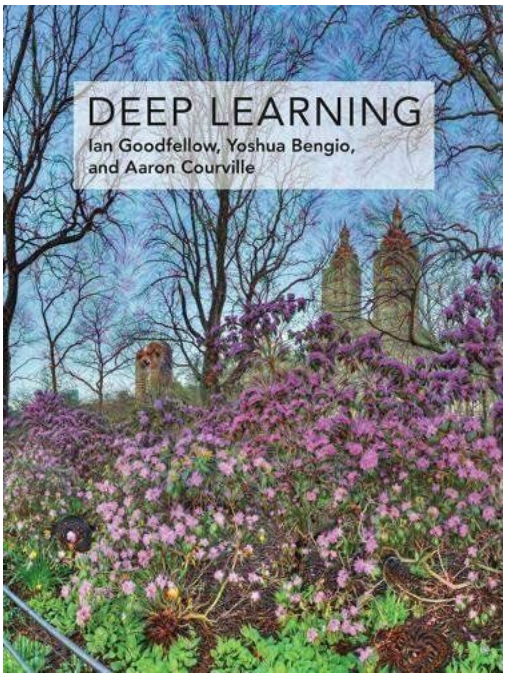


Figure 2.4: Deep Learning by Goodfellow et al. MIT Press 2016

2.3.2 Fully Convolutional Networks for Semantic Segmentation

A breakthrough paper published in 2015 written by Shelhamer et al. proposed a new approach for tackling semantic segmentation using Convolutional Neural Networks that led to a large number of advances in the area. Newer models such as the Deeplab model used in this project are based on the principles founded in this paper. For the task of semantic segmentation this is a key paper and has been cited thousands of times. The paper shows that using convolutional networks trained end-to-end, pixels-to-pixels improved on the previous best result in semantic segmentation. To produce accurate and detailed segmentations, a skip architecture was defined which combines semantic information from a deep, course layer with appearance information from a shallow, fine layer.

Convolutional networks yield hierarchies of features and are powerful visual models. Covnets have shown positive results in problems such as classification and bounding box object detection but prior to this paper, covnets applied to semantic segmentation problems had a large number of shortcomings. Compared to classification and detection, segmentation is a harder task. In classification the aim is to identify the object class. In detection, the location and size of the object also needs to be determined. In semantic segmentation every pixel needs to be labelled. In semantic segmentation, global information resolves what is in an image and local information determines where it is in the image. Skip architecture fuses feature hierarchy to combine deep, course, semantic information and shallow, fine, appearance information.

Fully Convolutional Networks

Each layer in the convolutional network is a three-dimensional array of size $h \ x \ w \ x \ d$ where h and w are spacial dimensions and d is the feature dimension. The first layer in the network is the input image with dimensions h x w and with d colour channels. In image classification problems, the input image is downsized and is passed through multiple convolutional and fully connected layers and the output is a single predicted label. If the image is not downsized, there is not a single label but instead there is an output of multiple labels. Due to max pooling, the image will be smaller than the input but the output can be upsampled giving the label map.
During convolution the size of the output image decreases, in the case of semantic segmentation, the image has to be upsampled to increase the output size. There are various upsampling methods, for example a nearest neighbour approach could be taken where each pixel is duplicated. The approach that this paper takes is known as FCN-8 which uses a proceedure known as backward convolution or deconvolution which reverses the forward and backward passes of the convolution.

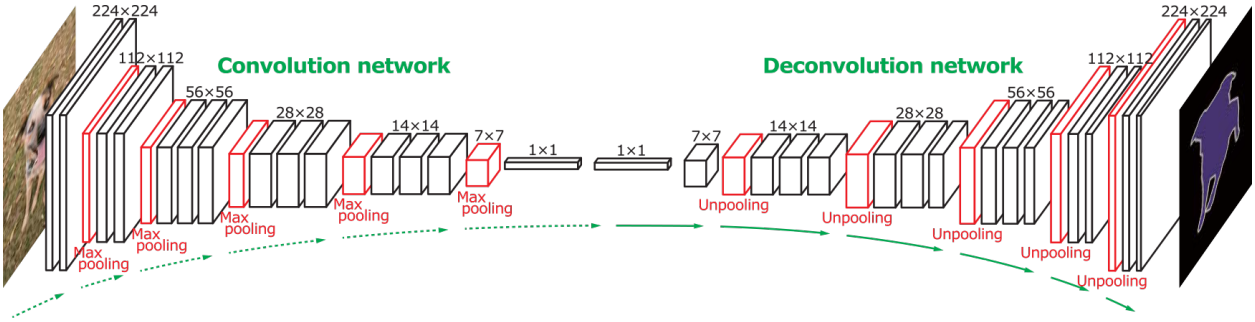


Figure 2.5: Convolution and Deconvolution Network Architecture from Noh et al. [3]

Figure 2.5. shows a Convolutional network and also a Deconvolutional network as described in the Noh et al. paper. The Deconvolutional network approach based on instance-wise prediction, performs well with object scale variations by eliminating the limitation of fixed-size receptive field in the fully convolutional network. The figure also shows many pooling layers which reduce the amount of data that is processed at later stages in the system and can help to reduce over-fitting [20]. Over-fitting refers to a model matching the data too-well and effectively memorising it. This causes the model to not perform well on data outside the training dataset as it has an inability to generalise.

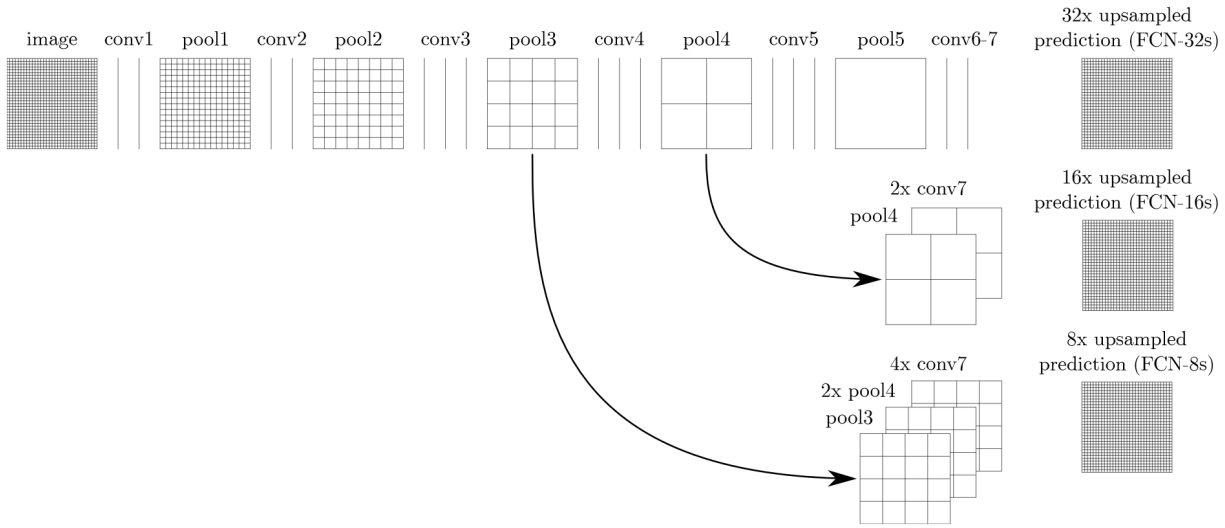


Figure 2.6: Model Architecture showing Skip Layers from Fully Convolutional Networks for Semantic Segmentation [4]

The new convolutional net for segmentation that was defined in the paper can be seen in Figure 2.6. The network combines layer of the feature hierarchy and refines the spacial precision of the output. After passing through the 7th convolution, the output size is small so a 32x upsampling is performed to match the input image. This results in a course label map as spacial information is lost when going deeper into the network but deep features are also obtained. To enhance the result, the output containing deep features can be combined with the output from shallower layers which contain more location information. This is performed by fusing the output via element-wise addition.

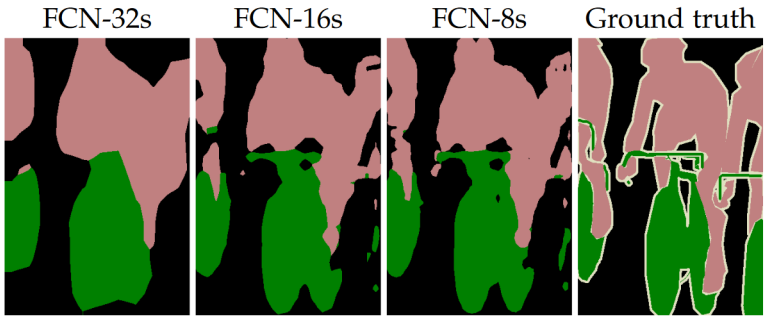


Figure 2.7: Results from refining fully convolutional networks but fusing information from layers with different strides to improve spacial detail taken from the paper [4]

2.3.3 DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution and Fully Connected CRFs

The model used for semantic image segmentation in this project is an implementation of DeepLab written by Chen et al. and open-sourced by Google in 2016. While the Shelhamer et al.[4] paper was an important breakthrough paper in the field of semantic segmentation that let to a lots of research in the area, this is a state of the art model that is more complicated but performs much better.
The model has gone through several iterations, each with improved results. The version of the model used in this project is the latest implementation, Deeplabv3+. The model is broadly composed of two steps,

an encoding and a decoding phase. In the encoding phase, essential information is extracted from the image using a convolutional neural network and in the decoding phase the extracted information is used to reconstruct the output in appropriate dimensions.
DeepLab uses atrous convolution to address the same issue that occurred in the FCN paper above, where a series of convolutions and pooling makes the output feature map very small. Atrous convolution allows the field of view of filters to be enlarged to incorporate a larger context. It is an efficient mechanist to control the field-of-view and find the best trade off between a large and small field-of-view. Section 2.1.3 introduced the various parameters that a convolution operation has, an additional parameter is added to atrous convolutions known as the dilation rate. It defines a space between values in a kernel. For example, a 3x3 kernel with a dilation rate of 2 will have the same field of view as a 5x5 kernel. This means that the kernel has the same field of view but at a lower computational cost.
DeepLab architecture uses a neural network architecture known as Spacial pyramid pooling (SPP). SPP uses multiple scaled versions of the input for training which captures multi-scale information. This makes the model robust to changes in the size of objects. DeepLab performs an atrous version of SPP where parallel atrous convolutions with different dilation rates are applied in the input feature map and are fused together. This is done using parallel versions of the same underlying network to train using inputs of different scale and combining the features later in the network.
The final architecture component that is employed by DeepLab is a Fully Connected Conditional Random Field (CRF).

2.3.4 Image Style Transfer Using Convolutional Neural Networks

Style transfer is the process of modifying the style of an image while preserving its content. This paper by Gatys et al. gives an approach where when given an input image and a style image, an output image can be created with the content of the input image but in the style of the style image. This project does not use this approach but instead uses a modified implementation by Luan et al.[5] which is discussed in the following chapter.
The approach taken by Gatys et al introduces a neural algorithm that can separate and recombine the content and style of an image. The problem of style transfer can be considered as a texture transfer problem where the goal is to synthesis a texture from a source image while constraining the texture synthesis in order to preserve the semantic content of the target image. The problem with previous approaches is that they all use low-level image features such as intensity of the target image to inform the transfer. Ideally, a style transfer should be able to extract the semantic content from the target image and then render the semantic content in the style of the source image. Recent advances in Deep Convolutional Neural Networks has given the ability to extract high-level semantic information from images. Therefore the goal of the work was to introduce a neural algorithm which uses the generic feature representations learned by CNNs. The method reduces to an optimisation problem within a single neural network. The style transfer algorithm combines a parametric texture model based on CNNs with a method of inverting the image representations. The images are generated by preforming a pre-image search to match feature representations.
The paper outlines an approach where a pre-trained Convolutional Neural Network, VGG-16, trained to perform object recognition and localisation is used to extract the features. Figure 2.3. shows the architecture. As only the feature extractors, i.e convolutional layers and max pooling layers are being used, the classifier parts, i.e the blue and yellow fully connected and softmax layers can be ignored. In the paper, the generated results were more appealing when average pooling was used instead of max pooling. As layers that are responsible for style and layers that are responsible for content can be distinguished between, they layers can be separated to independently work on content and style.
The task is then an optimisation problem where content loss, style loss and total variation loss will be minimised. Content loss is the distance between input and output images and is used to preserve the content. Style loss is the distance between style and output images where a new style is desired. Total

variation loss is a regularisation where spatial smoothness is used to denoise the output image.
Figure 2.8. shows the style transfer algorithm. The content and style are extracted and stored. Then the style image $\vec{a}$ is passed through the network and its style representation A^t.

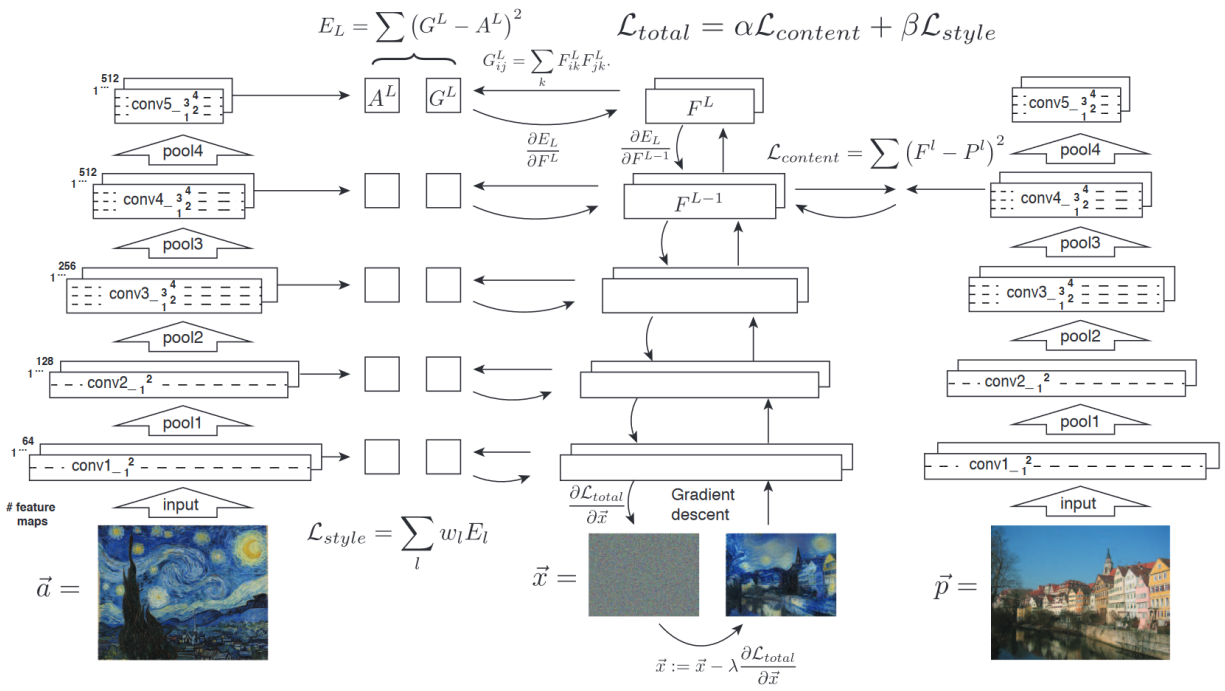


Figure 2.8: Style transfer algorithm from Gatys et al. [5]

2.4 Summary

The aim of this chapter was to provide enough background information to properly understand the later parts of the report. Some of the core literature was also discussed and in particular the capsicum annuum dataset paper which formed the baseline for this project. The following set of chapters will discuss the design and implementation of the proposed solution along with results from experiments carried out and a discussion on the performance of the models.

Chapter 3

Design and Implementation of Solution

This project can largely be split into two main stages, the initial stage of investigating the current status of deep learning and semantic segmentation in agriculture, along with implementing a baseline approach from current research and a stage of attempting to improve on the current research. A lot of the investigation and research has been discussed in the previous chapter on technical background and literature review. This chapter will mainly focus on the stages after this initial stage of research, from implementing the baseline to attempting to improve on the results by applying a neural style transfer to the original dataset.

3.1 Baseline Approach

This project builds on the work of the capsicum annuum dataset paper[1] previously discussed. The first step was to match the results achieved in this paper. The paper made the dataset that it created publicly available and outlined its approach but the code for the paper was not provided. The main focus of the paper was the creation of a synthesised dataset and whether it improved semantic segmentation results. As the quality of the synthetic dataset needed to be quantised, five experiments were performed using the DeepLab model [21]. The initial target was to replicate these experiments and results. While Caffe was used in the paper, the TensorFlow framework [13] was opted for in this project due to the access to resources and the fact that it has the DeepLab models available for use [22].
The baseline experiments are outlined below as seen in the dataset paper.

A *Train: synthetic. Test: synthetic.*
This experiment was run to obtain a performance reference point of the model when having access to a large and detailed annotated dataset for this domain.

B *Train: synthetic. Test: empirical.*
To determine to what extent a synthetically trained model can generalise to a similar set in the same domain without fine-tuning.

C *Train: empirical. Test: empirical.*
As a reference to see if the model can learn using a small dataset, using empirical data.

D *Train: PASCAL VOC. Fine-tune: empirical. Test: empirical.*
To compare the effect of bootstrapping with a non-related dataset.

E *Train: synthetic. Fine-tune: empirical. Test: empirical.*
To assess the effect of bootstrapping with a related dataset.

Some of these experiments were designed purely to test the quality of the synthetically generated data and as a result were not applicable to the baseline of this project. The key experiment, and the one that would be used as a comparison for future work in this project is experiment A. The only other experiment that would be run is experiment D due to its relative simplicity, it was useful as an intermediate step when implementing the baseline. In this report, experiment A will be known as the baseline and experiment D will be known as the intermediate experiment.

3.1.1 Intermediate Experiment

The intermediate experiment involved using a PASCAL VOC pre-trained checkpoint and fine-tuning the model with empirical data (1-30) from the dataset. The model was also evaluated using empirical data (41-50).

Converting Dataset to TFRecords

TensorFlow has its own binary storage format known as a TFRecord. When working with larger datasets, using a binary storage format can significantly impact the performance of the import pipeline and as a result decreases the training time of the model. The data also takes up less space on the disc and can be copied and read more efficiently. There are also advantages to using TFRecords other than to improve performance. The filetype is optimised for use with TensorFlow making it easy to combine multiple datasets. This is very beneficial for datasets that are too large to be fully stored in memory and are thus processed in batches, as is the case with the dataset used in this project. Batches are loaded from the disk and are then processed. The downside of using TFRecords is that the data needs to be converted to this format and with limited documentation available it can be a difficult process.
The TFRecord contains all of the data required during the training and testing of the model. As the file stores all of the data as a sequence of binary strings, the structure of the data needs to be specified before writing it to the file. The structure that is used is not a standard Python class but is instead a 'protocol buffer' which is a method of serializing structured data in an efficient way developed by Google. Before this structure could be created, several preprocessing steps needed to be completed on the dataset.
The first step was that the colour map needed to be removed from the ground truth images. Figure 3.1. shows an example of the ground truth which clearly shows what each class is labelled as. This colourmap needed to be removed so that the labels were greyscale with each class defined as a different greyscale value ranging from 0 to 8, as there are 8 different classes in the dataset.

Figure 3.1: Synthetic Ground Truth Label Mask from Dataset

Along with the dataset images and segmentation maps, a list needed to be defined that split the training and test data. This involved generating three text files containing the filenames of the training data, validation data and a combination of the two. In the case of the intermediate experiment, this meant that the empirical images 1-30 were used to fine-tune, i.e. to train, and images 41-50 were used as test data. This was simple to do using a Shell Script that parsed the dataset directories.

Transfer Learning using PASCAL VOC Pre-Trained Checkpoint

The purpose of this test in the paper was to see the effect of bootstrapping with a non-related dataset using the empirical data. Convolutional Neural Networks require training time and a large dataset to converge to an effective feature distribution. The idea was that bootstrapping would provide a stable starting point from which fine-tuning could quickly converge to a new optimum of the new dataset. This problem area is known as transfer learning and it has proven to be successful in many classification problems.
Transfer learning is a method allowing accurate models to be built in a timesaving way. Instead of starting the training process from scratch, previous learnings and patterns from similar but different problems are leveraged. Transfer learning is usually preformed using pre-trained models. A pre-trained model is a model that was trained on a large dataset to solve a similar problem. TensorFlow provides several DeepLab pre-trained models as part of its Model Zoo including PASCAL VOC 2012[19], Cityscapes and ADE20K.

DeepLab Model Architecture

The DeepLab model was discussed in detail in the technical background chapter which contains a review of the literature surrounding the models development. DeepLab models can be trained with weakly semi-supervised learning such as bounding boxes or with strong supervision such as semantic segmentation with per-pixel labelling as is the case with this project. The training of a strong supervision model was only possible due to the annotated dataset which contained ground truth class labels on a per-pixel basis. The underlying architecture of the DeepLab convolutional neural network model is VGG-16. This architecture was originally intended for global object detection but was modified for semantic segmentation by using fully convolutional layers and the use of the á trous algorithm. The Xception network backbone was used as it is a powerful network structure for server-side deployment. It is currently the most powerful backbone provided n the DeepLab implementation.
The hyperparameters for the baseline experiment were chosen to match the ones defined in the paper. In the paper the hyperparameters were manually optimised using validation dataset and a range of models and dataset configurations defined in the Deep Learning book by Goodfellow et al. and in a paper written by the books co-author, Yoshua Bengio [23]. The aim was to find a learning rate that reduced the validation set loss towards zero, learning rates of 0.1, 0.01, 0.001 and 0.0001 were explored and while IOU performance differed, none of the models or learning rates caused overfitting.
The investigation resulted in the choice of Adaptive Moment Estimation (ADAM) being selected with $\beta_1 = 0.9, \beta_2 = 0.999$,$\varepsilon$=$10^{-8}$ and a base learning rate of 0.001 for 30,000 iterations with a batch size of 4.

Experiment Execution

This experiment was run on Google Colab as the model loaded a pre-trained checkpoint for the PASCAL VOC dataset and fine-tuning was then performed. The results from the experiment are not particularly applicable to the project as the experiment was a control to test the value of using synthetic data generated in the dataset paper. Focus was placed on setting up the dataset to work with the model.

3.1.2 Baseline Experiment

The baseline experiment did not use a pre-trained model but instead involved training the model with synthetic data (1-8750) from the dataset and testing with synthetic data (8851-8900).

Switching to GPU Server

The intermediate experiment was written in a Jupyter Notebook and executed on Google Colab. The notebook was written in such a way that it would be easy to execute other experiments such as the baseline. The notebook was modified for the new baseline using the same hyperparameters as the intermediate test as specified in the dataset paper but instead of using a checkpoint, the model would be trained from scratch. Once the notebook development was finished, an attempt was made to migrate the notebook from Colab to the GPU server. This proved to be a very difficult task and even when the ssh tunnelling was set up, the notebook would run but multiple issues arose. A call was made to rewrite the implementation in a series of scripts and a git repository was created to develop the solution. This proved to be a more appropriate setup for a large training experiment and the code was later adjusted for other parts of the project.

Stages of Experiment

The implementation consisted of several sub-stages. The first stage was handling of the dataset and the creation of TFRecords. This was discussed in the intermediate experiment and only a few modifications were necessary to incorporate the new training data. After the dataset was created, the model could be trained. This was performed for 30,000 iterations using the previously mentioned dataset split and the Xception model variant. Batch normalisation was utilised during the model training. After the model was trained, several checkpoints were created in the same format as the PASCAL VOC checkpoints. These checkpoints were used to evaluate the quality of the model using the mean IOU performance metric. The model results were then visualised by creating segmentation maps of the evaluated test images. Finally the model was exported as a frozen inference graph for use in the model demo.

Model Loss during Training

During the model training stage, a TensorBoard live visualisation was run using the same ssh tunnel that was created for the Jupyter notebook. Due to issues with the gateway, the connection was not persistent and TensorBoard was not further utilised.

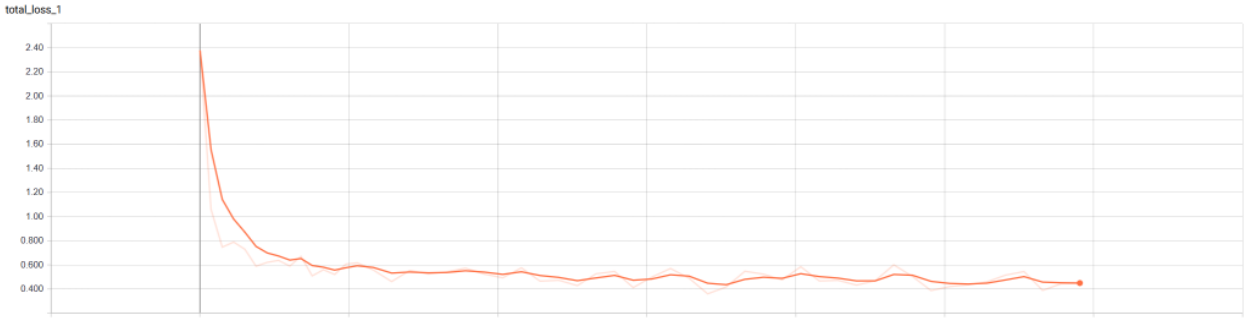


Figure 3.2: Plot of Loss from TensorBoard

Figure 3.2. shows the model loss for each iteration from step 1 to 30,000. The plot was generated using TensorBoard and each vertical line corresponds to 5000 iterations. The loss function used by the DeepLab model is the sum of cross-entropy terms for each spatial position in the CNN output map. The labels are equally weighted in the overall loss function except for ignored labels. The cross entropy rewards/penalises probabilities of correct classes only, it is independent of how the remaining probability is split between incorrect classes.

3.2 Neural Style Transfer

To address the key issue with applying semantic segmentation in an agricultural setting, Neural Style Transfer as a data augmentation technique with the aim of improving the models ability to generalise to bell peppers from outside of the original Capsicum Annuum dataset paper was investigated. The issue of models poorly generalising to data that is taken from a slightly different environment or in different conditions is a general problem in applying semantic segmentation to Agricultural data. A key consideration when applying a type of style transfer is that the shapes and locations of the classes must remain unchanged with only their 'styles' altered. The reason for this is that the aim is to use the same labels for the altered images as the original ones as having to relabel the images would make the task infeasible.
The approach of neural style transfer[5] outlined in the literature review section uses image representations derived from Convolutional Neural Networks to make the high level information explicit and produces images that combine the content of an image with the style of another. An issue with this for the bell pepper application is that it is mainly used for creating stylised photos that map the style of an artistic work and tends to create a blurred image that does not look photo-realistic. Another approach by Luan et al.[24] does not introduce 'unnecessary distortions' by adding a regularisation term so the image colours only undergo affine transformations from the input to the output.

3.2.1 Photo-realistic Style Transfer

The problem with Gatys[5] neural style algorithm in this project is that while it successfully transfers colours and style, it also introduces distortions that make the output look like a painting. This was the intent of the paper as the goal way to replicate artistic styles but in this application a photo realistic style is desired. Even when the input image and the style image are both photographs, the output exhibits distortions reminiscent of a painting. A paper by Luan et al. builds on the work of Gatys by constraining the transformation from the input to the output to be locally affine in colourspace and to express the constraint as a custom fully differentiable energy term with the intent of suppressing distortions.
To obtain photorealistic results there are two main challenges, structure preservation and semantic accuracy. With structure preservation the goal was to apply a transformation that would strongly affect colours while having no geometric effect. This would transfer the style without distorting edges. This is important in this project as not only is the shape of a bell pepper just as important as the texture, but also the ground truth labels from the unstylised images would be used as training labels so it was important to not modify the location or shape of the images when applying a style. Semantic accuracy refers to applying the style to the appropriate parts of the image, i.e pepper effects should be applied to peppers and leaf effects to leaves.
The algorithm takes two images, an input image which in this case is one of the synthetic images from the capsicum annuum dataset, and a reference style image which in this case is the effect that is to be applied such as a disease. The approach augments the neural style algorithm outlined in the literature review of the Gatys et al. paper by introducing two core ideas. The first is a photorealism regularisation term in the objective function during optimisation. This constrains the reconstructed image to be represented by a locally affine colour transform to prevent distortions. An affine transformation is a linear mapping method that preserves points and planes. Sets of parallel lines remain intact after an affine transform and it is usually used to correct geometric deformations. To preserver the structure of the input image and to produce photorealistic results, a constraint is placed on the transformation that is applied to the input image, as opposed to directly on the output image. This is achieved by adding a term to the original Gatys equation in Figure 2.8. which penalises distortions. The approach builds on the Matting Laplacian from Levin et al.[25] The second core idea was to introduce guidance to the style transfer vs the semantic segmentation of the inputs to avoid a content mismatch problem.

When all of the components are added together the new equation taken from the paper[24] becomes:

$$\mathcal{L}_{total} = \sum_{l=1}^{L} \alpha_\iota \mathcal{L}_c^\iota + \Gamma \sum_{l=1}^{L} \beta_\iota \mathcal{L}_s^\iota + \lambda \mathcal{L}_m \tag{3.1}$$

where L is the number of convolutional layers and ι indicates the ιth layer in the deep neural network. Γ is the weight that controls the style loss, a value of 10^2 was used in the project. α_ι and β_ι are weights used to configure layer preferences. $\mathcal{L}_c^\iota$ is the content loss, $\mathcal{L}_s^\iota$ is the augmented style loss and $\mathcal{L}_m$ is the photorealism regularisation. Finally, λ is a weight that controls the photorealism regularisation, 10^4 was used in the project as shown in Figure 3.3.

(a) Input and Style (b) $\lambda = 1$ (c) $\lambda = 10^2$ (d) $\lambda = 10^4$, our result (e) $\lambda = 10^6$ (f) $\lambda = 10^8$

Figure 3.3: Effect of Varying λ in Equation 3.1.

3.2.2 Dataset Generation Process

The code for the Deep Photo Style Transfer[24] was made available but as it used the Torch framework, a TensorFlow implementation[26] was chosen instead. The generation of the dataset with augmented style took several steps. First a style needed to be chosen and a ground truth needed to be made for the reference style. The only parts of the original images that the style transfer was to be applied to was the peppers. The way the algorithm was implemented meant that there was a one-to-one mapping between the classes in the input image and the segmentation of the reference style. To achieve the correct result, a new segmentation map needed to be made for the original input images where the peppers had the same class label as the corresponding reference style and the rest of the pixels in the image were set to the background class.
The four images necessary for each style transfer were, the reference style image, the ground truth for the reference style, the image to be augmented and the new binary label for the image showing the location of the peppers. These images were input into the deep style algorithm along with an option to select where the neural style algorithm would be applied, a deep matting post-processing step or both together.
The style transfer took approximately 25 minutes per image and was run for several days on the GPU server using a python script. The output was saved to a file every 100 iterations. Once the style transfer was complete, some post processing was necessary to finish the dataset generation. As only the peppers were being appropriately stylised, it was desired to leave the background of the images the same as the originals. To do this, each pixel in the images were checked against their ground truth segmentation maps. If the pixel value was equal to the pepper class then the image would take the corresponding pixel value from the new stylised image, otherwise the pixel would take the value of the original image. The new images could now be used to fine-tune the DeepLab model using the original segmentation maps.

Augmented Model

Once the new dataset was created a new TFRecord was made with the new dataset parameters. The model was then fine-tuned using the new images in a similar manner as the PASCAL VOC fine-tuning. The model was trained using 125 images for 30,000 iterations on the GPU server taking approximately 8 hours. The model was evaluated using images of bell peppers from ImageNet and a frozen graph inference model was exported for use in the model demo.

3.3 Summary

This chapter discussed the process taken for implementing the baseline experiment along with applying style transfer and training the new model. Some issues were mentioned along with the training specifications and hyperparameters used. The Neural Style Transfer algorithm was described in relative detail as it contained improvements relevant to this project based on the Gatys paper discussed in the technical background chapter. The results from the experiments can be seen in the following chapter.

Chapter 4

Experiments and Results

This chapter first introduces the types of performance metrics that are used to determine the quality of the test results. Due to the nature of the project, a large portion of the results are highly visual, as a result there is a significant number of figures presented. It was decided to keep the discussion in a separate chapter as the majority of the points discussed are not directly related to the specific images presented in this chapter.

4.1 Performance Metrics

4.1.1 Intersection-Over-Union (IOU)

When evaluating the performance of a deep learning model, predictions are classified into four categories: true positives, true negatives, false positives and false negatives. With dense prediction tasks, such as semantic segmentation, it can be ambiguous as to what is classified as a true positive. As a result, there a different ways of evaluating segmentation techniques.
Multiple performance measures are available for evaluating semantic segmentation results, each with strengths and limitations. A comparison of various methods[27] including Overall Pixel (OP), Per-Class (PC) and Jaccard Index (JI) was made and concluded that the current standard is the Jaccard Index, also known as Intersection-Over-Union (IOU), as it addresses issues with OA and PC such as the bias in the presence of unbalanced classes and datasets with a large background class. A limitation of IOU is that while it can evaluate the amount of pixels that have been correctly labelled, it does not account of how accurate the segmentation boundaries are. The paper suggests matching the IOU with a boundary F1 contour matching score (BF) which indicates how well the predicted boundary of each class aligns with the true boundary. Unfortunately the paper that the baseline experiments are based on only uses IOU as a metric and does not combine it with a BF score. To compare results with the paper, the same performance measure must be used.
IOU, also known as the Jaccard Index similarity coefficient, is a method which quantifies the percentage of overlapped pixels between the target mask and the predicted output, i.e. the number of pixels common between the target and prediction masks divided by the total number of pixels present in both masks. It can be determined per class or as an average over all classes. The IOU is first calculated for each class separately and then averaged over all classes to provide the mean IOU score for the semantic segmentation prediction.

$$IOU = \frac{target \cap prediction}{target \cup prediction}$$

The equation as defined in both papers by Barth et al. is shown in 4.1. where the mean IOU per class equals the intersection of the semantic segmentation and ground truth divided by their union. A confusion matrix, also known as an error matrix, is a type of table layout used to visualise the performance of an

algorithm, typically a supervised one. Each row of the matrix represents the instances in a predicted class while each column represents the instances in an actual class. It is a special kind of contingency table with an actual and predicted dimension and identical sets of classes in both dimensions. To derive the equation, a pixel-level confusion matrix, C, is calculated for each image I in dataset D where $S^I_{gt}(p)$ is the ground truth label of pixel p in image I and $S^I_{ps}(p)$ is the predicted label. As a result, C_{ij} equals the count of pixels with ground truth i and prediction j. The total number of pixels labelled with class i in the ground truth image and total number of pixels with prediction j in the image are denoted by G_i and P_j respectively.

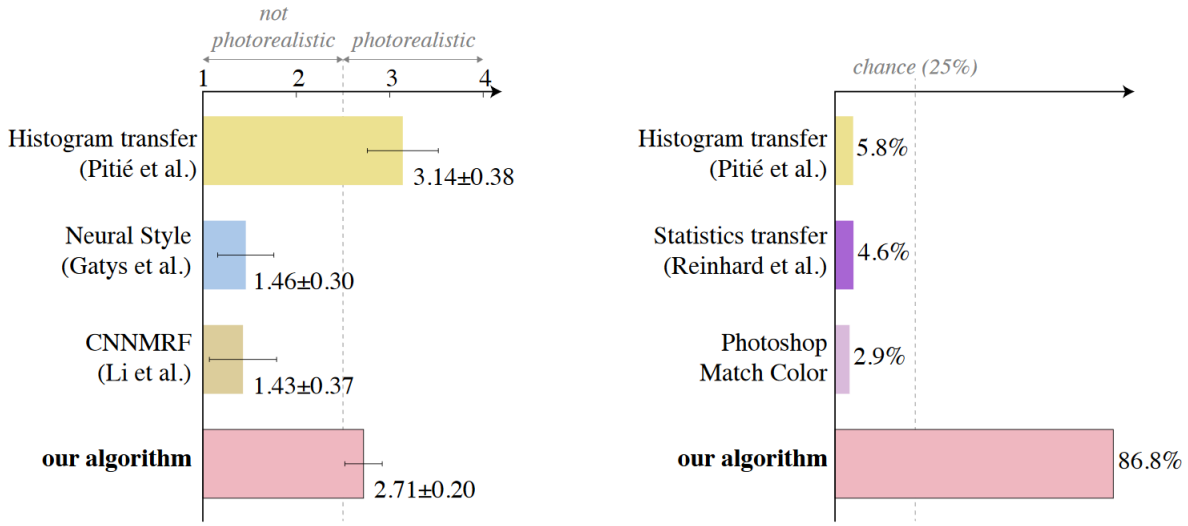
$$IOU = \frac{1}{L}\sum_{i=1}^{L}\frac{C_{ii}}{G_i + P_i - Cii} \tag{4.1}$$

$$C_{ij} = \sum_{I \in D}\left|\{p \in IS^I_{gt}(p) = i \wedge S^I_{ps}(p) = j\}\right| \tag{4.2}$$

$$G_i = \sum_{j=1}^{L} C_{ij} \quad , \quad P_j = \sum_{i=1}^{L} C_{ij} \tag{4.3}$$

4.1.2 User Studies

The Deep Photo Style Transfer paper used user studies to validate its work. The studies consisted of 40 users scoring the image results on a scale of 1 to 4, ranging from 'definitely not photorealistic' to 'definitely photorealistic'. The test contained 8 different scenes for each of the 4 methods, Histogram transfer, Neural Style, CNNMRF and the paper's algorithm, totalling 32 questions.



(a) Photorealism Scores

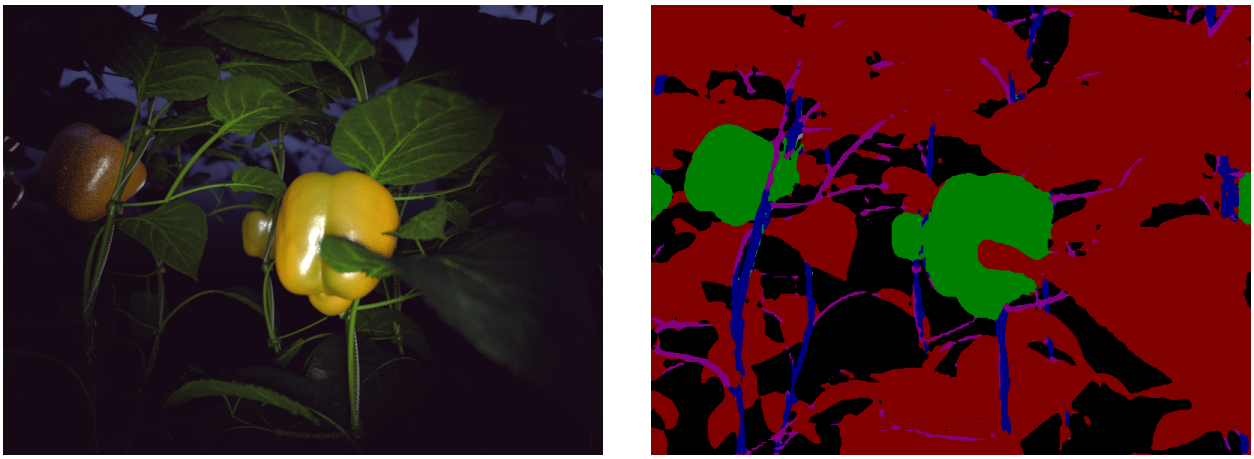
(b) Style Faithfulness Preference

Figure 4.1: User Study Results from Luan et al.

This performance test was not executed in this project for logistical reasons but it could be used in future research as a means of analysing image results. The studies are obviously subjective in nature but with enough user data, trends can be spotted.

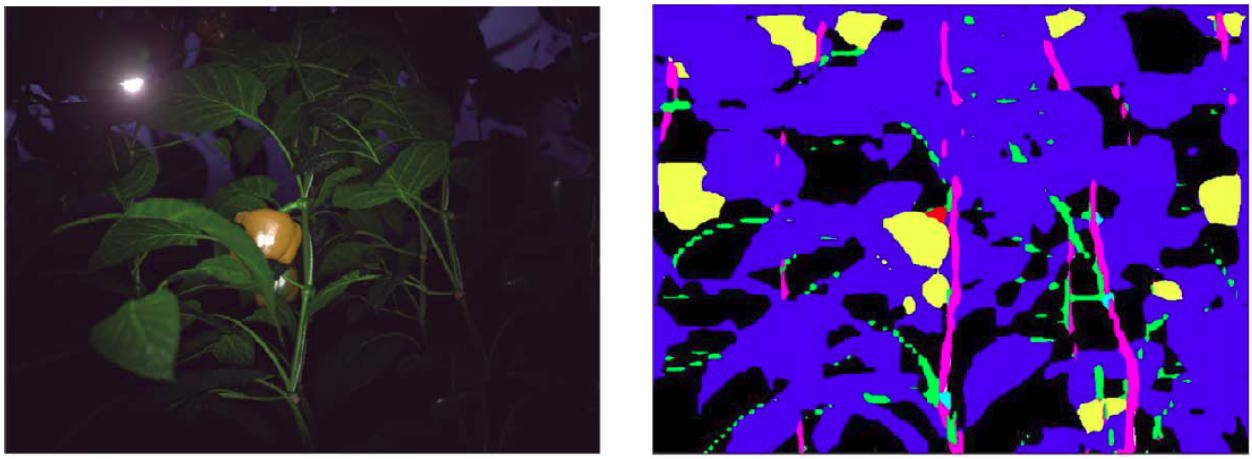
4.2 Baseline Results

After training the model with 8750 synthetic images for 30,000 iterations, results were achieved that matched the results from the baseline paper. Figure 4.2. shows an example of a segmentation performed on the test set. The segmentation closely matches the equivalent test from the Barth et al. paper as seen in Figure 4.3. Only a few segmentation examples were provided in the paper so the test images do not match.



(a) Synthetic Image (b) Semantic Prediction

Figure 4.2: Baseline Output Results



(a) Synthetic Image (b) Semantic Prediction

Figure 4.3: Barth et al. Output Results

The mean IOU was determined to be 0.38 across all classes. The baseline paper did not provide a single mean IOU value in their results but instead provided a plot of the class IOU values for each of their experiments. This can be seen in Figure 4.4. where A is the relevant label. From reading the IOU values from the chart and calculating the average, the mean IOU appears to be approximately 0.42, which is very close to the 0.38 achieved.

The goal of this baseline was to match the results of the original paper as closely as possible, once it was determined that the results were adequately similar, the baseline was set and further implementation could be performed.

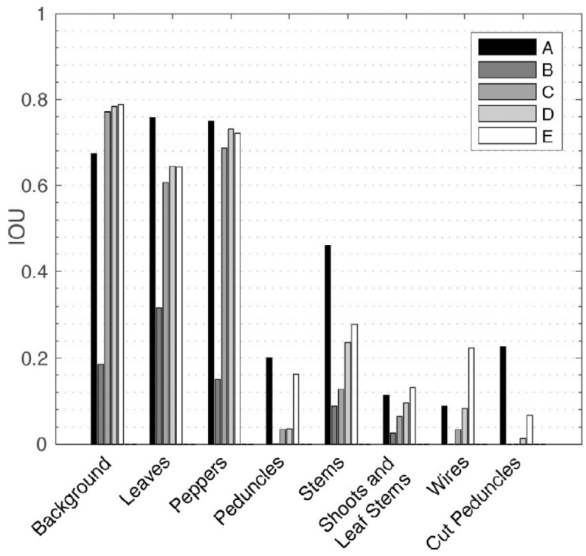


Figure 4.4: Average IOU per class for experiments A through E from Barth et al.

4.3 Style Transfer

The following set of Figures shows the different stages of applying a style transfer to the original capsicum annuum dataset. One of the issues mentioned in the Barth et al. paper was that due to the time of year the empirical images were taken in the greenhouse, all of the peppers are green and yellow. There was only a small range of ripeness available for study and as a result the empirical images mainly consist of yellow peppers. The synthetic dataset was created to have a mix of 50% green peppers and 50% yellow peppers. The test style that was chosen was that of a red bell pepper as seen in Figure 4.5. along with the corresponding ground truth which was manually created using Adobe Photoshop.

(a) Reference Style

(b) Style Segmentation

Figure 4.5: Style Reference

Figure 4.6. shows the output of style transfer model after several different number of steps. The only part of the stylisation that is relevant to the test is the bell pepper classes. Figure 4.7b. shows the location of the peppers, this segmentation map was used to extract the stylised peppers and overlay them onto the input image, Figure 4.7a. The final output, which would be used to fine-tune the model can be seen in Figure 4.8. This process was repeated until the style transfer dataset contained 125 images.

(a) Step 100

(b) Step 200

(c) Step 300

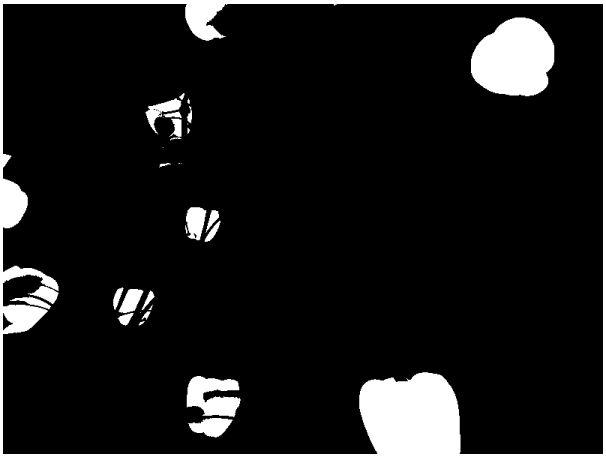
(d) Step 400

(e) Step 2000

(f) Step 4000

Figure 4.6: Stylisation Steps

(a) Input Image



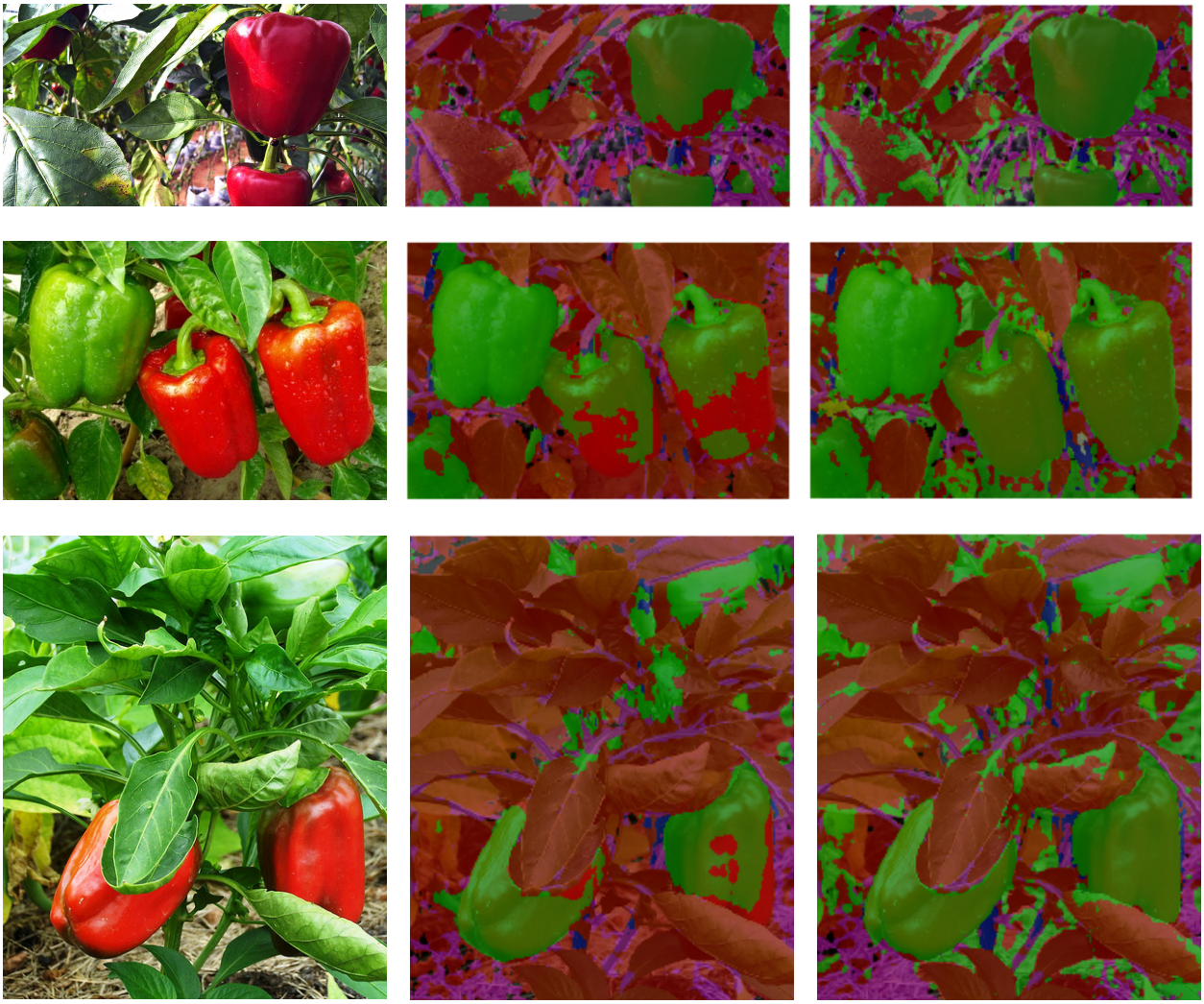
(b) Style Transfer Output

Figure 4.7: Neural Style Transfer

Figure 4.8: New Training Data

4.4 ImageNet Tests

After training the model with the synthetic dataset several images from ImageNet were used to determine the ability of the model to detect bell peppers from outside the test set. Without having the images labelled they can only be visually inspected for quality.



(a) Input Image (b) Baseline Segmentation (c) Style Segmentation

Figure 4.9: ImageNet Results

4.5 Summary

This chapter consisted of a large amount of visual examples from the different stages of the project implementation. The next chapter will discuss the results and address the shortcomings of the implementation. Some references will be made back to this chapter but most of the discussion is isolated from the exact examples presented here.

Chapter 5

Discussion

This chapter is a discussion on the successes and failures of the project implementation along with an analysis on the performance and suggestions of areas that could be improved.

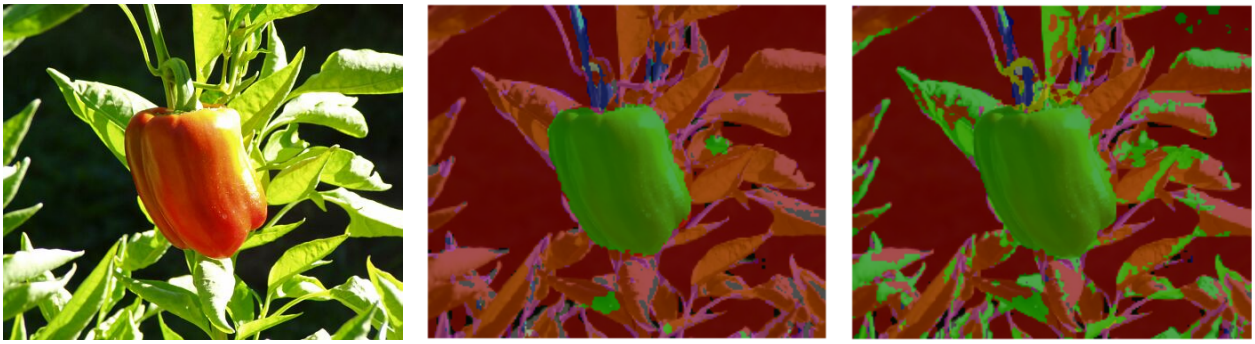
5.1 Model Performance

5.1.1 Matching Barth et al. Results

The mean IOU achieved in the baseline experiment was determined to be 0.38 across all classes. The approximate mean IOU from the Barth et al. results was calculated to be 0.42. While the IOU values are similar, both of the IOU results appear to be relatively low with some classes having an IOU of less than 0.2. The low IOU results in the baseline paper were addressed and were not considered to be a large issue. The results were linked to both the quality of the annotations in the ground truth, particularly in dark regions where classes that were hard to see were simply labelled as background and as a result were detected as false positives where if the labelling was accurate they would have been true positives. This annotation bias resulted in a lower reported mean IOU. The other issue is that elongated parts, such as stems were not connected which can be seen in examples such as Figure 4.3b. This is most likely due to the á trous algorithm upscaling a sparse, low resolution input feature map. Segmentations were therefore not completely overlapping with the ground truth. The original paper determined that good recall was observed for each class. This means that when the result was positive, the model performed well at predicting the output as positive. The downside of having good recall is that precision can be low as a result. This effect was exaggerated in the stylisation model.

5.1.2 Effect of Stylisation on Generalisation

The aim of applying the neural style transfer to the original dataset was to improve the models ability to generalise to new images. This was tested by running both models on some images from ImageNet that contained bell peppers. The baseline model performed better than expected on the ImageNet images which left less room for improvement. Some of the results of this can be seen in Figure 4.9. While only one style was applied and the set of images that were used for testing was relatively small, some trends can be noticed. The first of which is the increase in false positive detections of bell peppers. When comparing the baseline segmentation results with the style segmentation results, it is clear that there is more 'noise' in the style results. This seems to occur on leaves that have a large amount of shine or glare and could be due to the style reference image containing a bell pepper with a significant amount of shine. This is more evident in Figure 5.1. below which contains a pepper amongst brightly lit leaves.



(a) Input Image (b) Baseline Segmentation (c) Style Segmentation

Figure 5.1: ImageNet Failure Case

Based on the small set of test results, the detection of the bell peppers seems to have improved. In the baseline tests some parts of the peppers were not fully detected, in the stylised test results the full peppers were detected.
Without any further modification, determining which model to use would come down to the application. If it is important that the full pepper is detected, then the model with the neural style transfer fine-tuning appears to be a better choice but if having a large amount of false positives for pepper detection is an issue than the original model is currently better. With the limited testing it is not possible to determine if the hypothesised solution was successful. The results do however indicate that the approach may yield positive results if the appropriate style references were chosen and style transfer as an augmentation technique can address some of the issues with time of day, weather and seasonal changes in an agricultural setting.

5.2 Model Shortcomings

5.2.1 Insufficient Testing

The main issue with analysing and discussing the results of this project is that there has not been enough testing to determine if the method is a viable one. The development of the model was only completed at a late stage of the project and testing took a significant amount of time. Generating the 125 images using the neural style transfer method took over 60 hours on the GPU server. The other issue with the results is that the ImageNet images that were used to test the ability of the models to generalise were not semantically labelled. This greatly reduced the quantisable results such as IOU measurements and reduced much of the analysis to being subjective. Manually labelling images would be the approach to take with this and having a large dataset of images with ground truths would open doors to more detailed analysis.

5.2.2 Overhead of Dataset Generation

One key consideration when investigating the approach of data augmentation via neural style transfer is that it greatly increases the time of the training stages of the model. While inference using the model is not affected, generating each augmented image took approximately 25 minutes on the Nvidia Titan GPU. 4000 iterations were taken when applying the styles and the difference between step 2500 and 4000 was not significant. This means that similar results could have been achieved in a two thirds of the time. While this is a large time saving, the time to generate a single training image is significant and the model also needs to be fine-tuned once the new data has been created.

(a) Step 2500

(b) Step 4000

Figure 5.2: Change in Image Quality over 1500 Steps

5.3 Technical Issues and Implementation Improvements

The original Barth et al. paper specified a base learning rate of 0.00005 with a batch size of 10. When these hyperparameters were attempted on the DeepLab model running on the GPU server, a OOM (out of memory) error occurred in TensorFlow. The follow-up paper by the same authors modified these hyperparameters to have a base learning rate of 0.0001 and a batch size of 4. This is what was chosen in the project implementation as it did not result in an OOM error.
The learning rate hyperparameter controls how quickly or slowly the network learns a problem. The amount of change to a model during each step in the optimisation search process is controlled by the learning rate and it is one of the most important hyperparameters for performance.

5.4 Thoughts on Barth et al. Approach

The aim of the Barth et al. paper was to produce a methodology which would reduce the current bottleneck of the reliance on manually annotated images that deep learning models require, especially convolutional neural networks. The paper, along with the follow-up paper, provided evidence for the hypothesis that only a small manually annotated empirical dataset was necessary for fine-tuning along with bootstrapping a synthetic dataset. The baseline result of their paper, which was training and testing on the synthetic dataset archived a mean IOU over all classes of 0.53. When the model was fine-tuned with empirical data and tested on an empirical dataset it achieved a mean IOU of 0.40. The question was raised as to the tradeoff in time investment between creating a synthetic dataset and manually annotating additional empirical data. The paper argues that once a model has been created for one plant, it is easier to create additional models.

5.5 Training Time vs Inference Speed

Something to consider is the application of the models. An example of a use of semantically segmented food produce is in robotic crop harvesting. As the training will not be occurring in real time, the training time and dataset generation time is not significant once it does not involve human interaction. While the training and dataset generation took a large amount of time, the inference model returns a prediction effectively immediately. If the case of robotic harvesting this may be the sole additional requirement along with prediction accuracy.

5.6 Summary

This chapter discussed the performance results of the project implementation along with thoughts on the efficacy of the neural style transfer method. Shortcomings with the implementation are also discussed along with some technical issues. The next chapter delves into the ethical problems that are present in deep learning with a particular focus on how they relate to this project.

Chapter 6

Ethics

As machine learning and artificial intelligence has grown in recent years, ethical concerns are being brought up and discussed more and more. An open letter on artificial intelligence[28] calling for research on the social impacts of AI was signed by Stephen Hawking, Elon Mush and many experts in the field in 2015. This project focuses on a relatively specific deep learning problem, semantic segmentation and its application to agriculture but the underlying technology has applications in many areas. In this chapter, ethical issues will be brought up which are directly related to the problem area of this project and there will also be some brief discussion about the more general short and long term ethical issues in artificial intelligence.

6.1 AI Automation

"Investments in AI should be accompanied by funding for research on ensuring its beneficial use, including thorny questions in computer science, economics, law, ethics, and social studies, such as: How can we grow our prosperity through automation while maintaining people's resources and purpose?" [29]

More and more jobs are becoming automated by machines. The automaton of manual labour has been occurring since the industrial revolution but with the rise of machine learning, the issue has received increased discussion. Automation leads to higher productivity with lower production costs. In the case of food production this means that more food could be grown at a cheaper rate. A potential ethical issue with automation in the food industry is that less farming jobs would be available as machines could perform the tasks of farmers. As automation is a tool and not a behaviour, the question becomes down to whether the creation and use of automation software is ethical. When automation is used correctly, it can improve working life and replace monotonous tasks. If automation is used as an assistant as opposed to a replacement, ethical issues do not arise.

6.1.1 Robot Tax

The growing concern that automation will replace workers and cause significant job loss has led to a legislative strategy known as a robot tax. The goal of the tax is to disincentivise the replacement of workers by machines and to create a social safety net for those who are displaced. The tax has already been introduced in some form in South Korea where tax breaks were reduced for investments in robotics.
The proposed robot tax has been highly controversial with arguments for and against. Some experts say that the tax is necessary so that government spending can continue even as taxable income from human workers decreases. Some well known people have voiced their opinion on the matter with Stephen Hawking critising owners of machines for initiating a lobby against wealth redistribution. Bill Gates came out in favour of the tax after Elon Musk stated that a universal basic would offset the employment effect of robots. The big

arguments against the tax are that it would be very difficult to enforce as the definition of a robot is very malleable and autonomous elements are already present in many areas and that having the tax is the same as having a penalty on innovation.

6.2 Transparency

An issue that has been prevalent since companies began to develop software is that unless their code is open source, it cannot be publicly scrutinised. If an algorithm makes a decision that affects someone, it may not be possible to see how the decision was made. This becomes more of an issue in deep learning as the inner workings of the algorithms are obscure even to their creators. This is due to the nature of the hidden layers in neural networks. Deep learning algorithms can be built using just a few hundred lines of code but as the logic is learned from training data, the inner workings are not reflected in the source code. Most algorithms are created and developed by for-profit companies and are viewed as highly valuable intellectual property which must remain in a 'black box'.

6.2.1 Asilomar AI Principles

Two of the principles in the list of 23 principles developed in conjunction with the 2017 Asilomar conference held by the Future of Life Institute are in relation to transparency. *"7. Failure Transparency: If an AI system causes harm, it should be possible to ascertain why."* and *"8. Judicial Transparency: Any involvement by an autonomous system in judicial decision-making should provide a satisfactory explanation auditable by a competent human authority."* [29] It may be difficult to follow these principles when Deep Learning is involved.

6.2.2 General Data Protection Regulation

One of the important sections in the GDPR act is the right to an explanation. This could address the issue of transparency by mandating that users are able to get the data behind the algorithmic decisions. It tackles the problem of intentional concealment by corporations and has already been introduced in the United States where insurance companies are required to explain how rate and coverage decisions are made. Some claim that the right to explanation is at best unnecessary and at worst harmful as it threatens innovation. The field of Explainable AI aims to provided better explanations that can be understood more easily while attempting to reduce the negative effects of transparency.

6.2.3 xAI

Explainable Artificial Intelligence (xAI) refers to techniques in which AI algorithms can be easily understood by humans, as opposed to the 'black box' concept where even designers cannot explain how a result was achieved. Transparency is usually associated with a tradeoff between how accurate and how transparent it is, the tradeoffs grow larger as the system increases in internal complexity. There are several approaches to providing explainability but they each have their downsides. Some of the approaches include using simpler models at the expense of accuracy, providing intermediate model states in vision models, for example showing which features are extracted at each level and modifying inputs to expose how the model processes changes.[30]

6.2.4 Open Source

Open Source provides a natural foundation for use as a vessel for transparency. The open source model is a software development model that encourages collaboration in a decentralised repository. It began as a response to the limitations of proprietary code and ensures that source code is available for general public

use. Open source software can be used by anyone and for anything, for good or for bad. For example, Project Maven is a Pentagon project which uses machine learning to analyse drone footage which uses Googles open source TensorFlow library.

6.3 Agricultural Ethics

When considering the impact of the increased use of technology in Agriculture, there are positive and negative ethical consequences. On the positive side, artificial intelligence can be used as a tool by farmers and agricultural companies in a similar way to technology developed during the industrial revolution and the increase in production could reduce food costs while increasing yield. This could increase the standard of living of agricultural workers and improve the state of the international food shortage crisis.
On the negative side, automation and technology could quickly become so proficient that the number of agricultural workers could drop dramatically as discussed in the AI Automation section above. Another possible effect is that the ability to quickly determine the quality of produce could lead to large amounts of food waste due to produce appearing misshapen or the wrong size. This is already a large issue with a study from the University of Edinburgh estimating that more than a third of farmed fruit and vegetables never make it to shop shelves as the not meet supermarket and consumers standards of how they should look.[31]

6.4 Other Uses of Semantic Segmentation

While this project focuses on semantic segmentation in an agricultural setting, similar models could be used without modification and trained using datasets that would be relevant to other areas such as autonomous vehicles. The DeepLab V3 Xception model which was used in this project has been widely used and trained with the Cityscapes[32] dataset that is contained in the repositories Model Zoo to performs segmentation for autonomous vehicles. The significance of this is that it is possible for a project to be used and implemented in a context in which it was not intended. This brings up ethical issues with regard to the safety of these models being used in a context other than Agriculture in which human life is at risk if poor predictions are made.

Dataset Bias

Deep learning models learn representations of the data they are provided with, as a result the models can learn biases due to the content of the dataset. In agriculture this does not cause too much of a concern as biased data does no pose an ethical problem, the model may simply perform poorly and lack the ability to generalise. It is however a more general issue with semantic segmentation and other classifications and could prove to be dangerous in contexts such as autonomous driving. For example, if a model was trained using only a certain range of images and all of the testing was performed on a similar set, the accuracy and performance of the model may be exaggerated compared to real world results which could pose a threat to life.

6.5 Summary

This chapter discussed some of the ethical concerns with deep learning and semantic segmentation in the context of the project and in a wider setting. The area of ethics in artificial intelligence in a rapidly growing area of study as machine learning becomes more and more incorporated in practical settings. The next chapter concludes the project with a brief overview along with some thoughts on areas that the project could be taken in the future based on research carried out during the course of the project.

Chapter 7

Conclusions and Further Research

The goal of this project was to explore semantic segmentation in the context of Agriculture. After the initial stage of research, a baseline model was developed and a hypothesis was defined. The hypothesis was to determine if the issue of changing environmental conditions in agricultural settings could be addressed by using neural style transfer as a data augmentation technique to make the baseline model more robust and improve its generalisation. Insufficient testing was completed to determine if the hypothesis was confirmed but early indications show that it could be a viable technique with some alterations outlined below as part of further research in the area. All of the code developed over the course is available on GitHub and all of the style transfer and model training/testing can be performed on different datasets.

7.1 Generative Adversarial Networks

A generative adversarial network (GAN) is a type of machine learning model in which two neural networks contest with each other in a zero-sum game. GANs were introduced in a paper[33] by Goodfellow et al. It is a type of unsupervised learning technique used to generate images that look authentic to observers and have realistic characteristics. The model consists of a generative network which generates new candidate images and a discriminator network which evaluates the quality of the candidates.

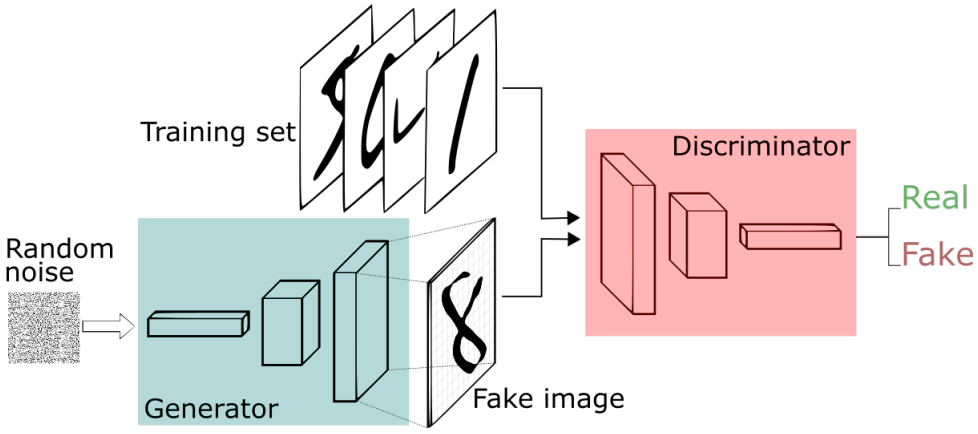


Figure 7.1: Generative Adversarial Network Architecture

This approach of generating images was initially investigated as a way of augmenting the dataset to cater for changing environmental conditions. It was later decided that neural style transfer would be a more appropriate approach for the project but the possibility of using a GAN is still there. GANs have some

issues such as the difficulty in training them but they can produce some amazing results. Figure 7.2. shows an example taken from the CycleGAN paper. It is still an early area of research but future developments could make the approach more desirable.

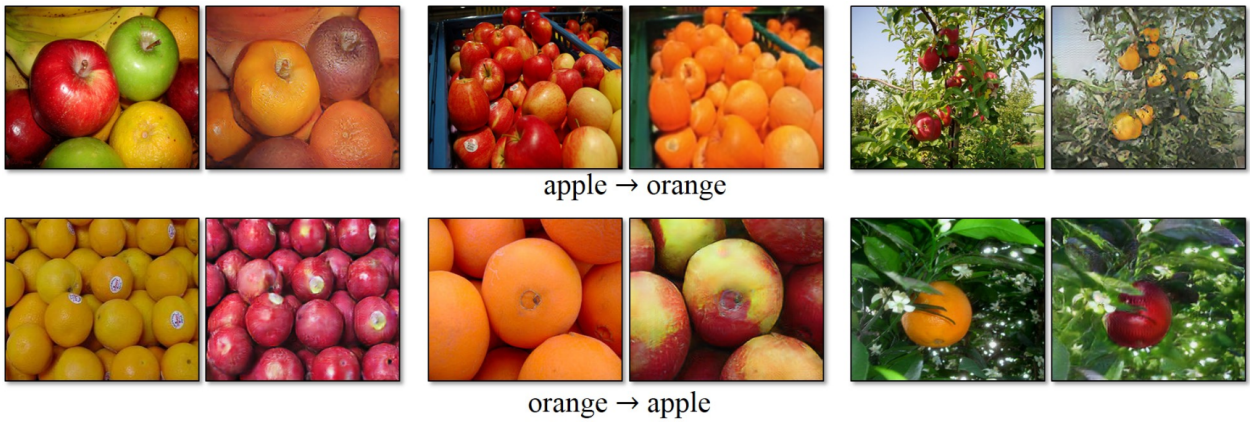


Figure 7.2: CycleGan Examples

7.2 Style Randomisation

Most data augmentation techniques include geometric distortions which are successful at teaching a model to be rotation and scale invariant. The issue with semantic segmentation in agriculture is that the variances between produce tends to be colour, texture and complex illumination based. A paper by Jackson et al.[34] introduced an idea of randomising texture, contrast and colour while preserving shape and semantic content for the purpose of data augmentation. A series of label-preserving transformations are applied to improve the quality of the dataset. The paper states that the original neural style transfer algorithm requires a lot of computation which makes it an unideal solution of data augmentation. This can be seen in this project where generating a single example took almost 30 minutes. A new style transfer algorithm called Fast Neural Style Transfer has dramatically sped up the time taken to perform style transfer with only a single forward pass through a network.

Figure 7.3: Style Augmentation with Randomised Texture, Colour and Contrast while Preserving Shape

7.3 Label Quality and Importance

The capsicum annuum dataset that was used in this project contains images labelled with 9 different classes. In the context of robotic harvesting, it may not be necessary to accurately label each different class. For example, it is possible that only the pepper and any protruding leaves or stems that are in the way of the

harvester need to be classified. This approach would not only speed up model inference as less classes would need to be determined, but it could also lead to the possibility of making datasets that are coarsely labelled with areas of increased quality, such as the bell peppers. This would involve using some modified training techniques such as conditional random fields and loss correction.
A similar idea is the one of importance-aware semantic segmentation. This idea was introduced in the context of autonomous driving in a paper by Chen et al.[9] where an Importance-Aware Loss (IAL) was designed which emphasises critical objects in autonomous driving, such as pedestrians, while putting less focus on areas such as the background sky. The IAL operates under a hierarchical structure with classes located in different levels depending on their importance. This idea could be incorporated in robotic harvesting to speed up processing time.
Another possibility is the use of saliency map detection[35] to detect the important regions in an image or a combination of saliency and semantic segmentation.

7.4 Inference Model Demo

A Colab notebook was created at the end of the project to demonstrate and compare the baseline and stylised models. The demo first loads a frozen inference model based on which one was selected. Image urls can be input or one of the several sample images can be selected. The demo then makes a prediction and the displays the input image, the segmentation map and the segmentation map overlayed on the input image. The colours of each class are shown in a legend beside the images. Figure 7.4. shows what the demo looks like and gives an example of it in use.

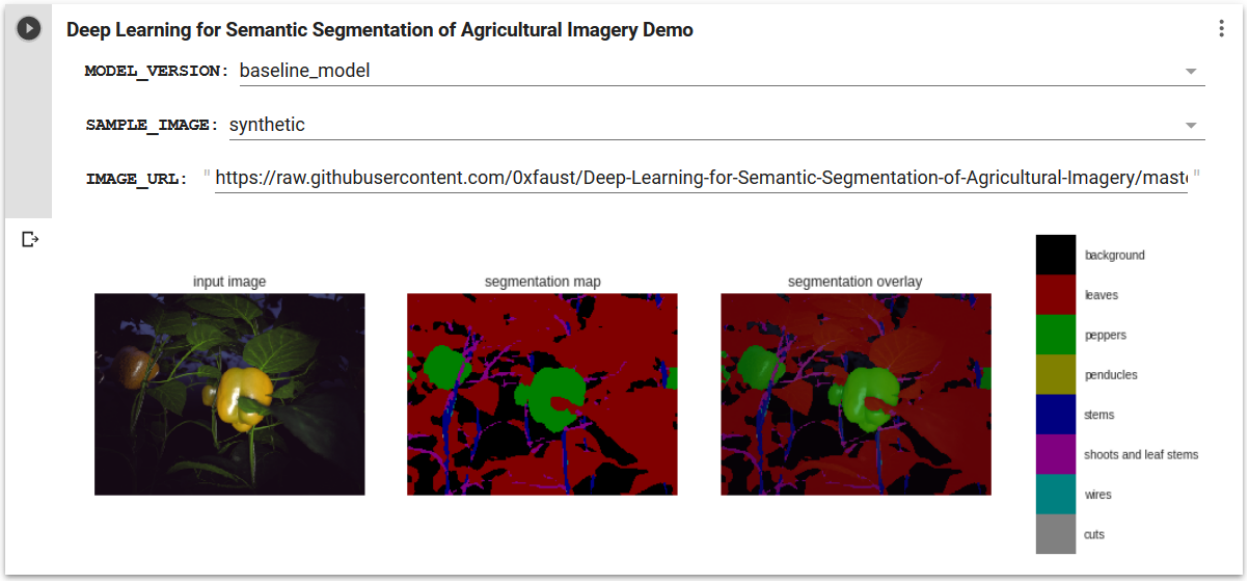


Figure 7.4: Project Demo

7.5 Summary

This chapter concluded the project report and briefly mentioned some possible future paths of development. The references and an outline of the code repository *(https://github.com/0xfaust/Deep-Learning-for-Semantic-Segmentation-of-Agricultural-Imagery)* can be found in the following sections.

Bibliography

[1] R. Barth, J. IJsselmuiden, J. Hemming, and E. V. Henten, "Data synthesis methods for semantic segmentation in agriculture: A capsicum annuum dataset," *Computers and Electronics in Agriculture*, vol. 144, pp. 284 – 296, 2018.

[2] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, 2014.

[3] H. Noh, S. Hong, and B. Han, "Learning deconvolution network for semantic segmentation," *CoRR*, vol. abs/1505.04366, 2015.

[4] E. Shelhamer, J. Long, and T. Darrell, "Fully convolutional networks for semantic segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 4, pp. 640–651, April 2017.

[5] L. A. Gatys, A. S. Ecker, and M. Bethge, "Image style transfer using convolutional neural networks," pp. 2414–2423, 2016.

[6] Food and A. O. of the United Nations (FAO), "How to feed the world in 2050," 2009.

[7] A. Kamilaris and F. X. Prenafeta-Boldú, "Deep learning in agriculture: A survey," *Computers and Electronics in Agriculture*, vol. 147, pp. 70 – 90, 2018.

[8] S. W. Chen, S. S. Shivakumar, S. Dcunha, J. Das, E. Okon, C. Qu, C. J. Taylor, and V. Kumar, "Counting apples and oranges with deep learning: A data-driven approach," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 781–788, April 2017.

[9] B. Chen, C. Gong, and J. Yang, "Importance-aware semantic segmentation for autonomous vehicles," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–12, 2018.

[10] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," 2009.

[11] G. van Rossum, "Python 2.7.16," Python Software Foundation, 1990.

[12] Pypa, "Virtualenv 16.4.3," virtualenv.org, 2011.

[13] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[14] D. Schlegel, "Deep machine learning on gpu," *University of Heidelber-Ziti*, vol. 12, 2015.

[15] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*, vol. 61, pp. 85 – 117, 2015.

[16] H. Abdullahi, R. Sheriff, and F. Mahieddine, "Convolution neural network in precision agriculture for plant image recognition and classification," 2017.

[17] A. K. Mortensen, M. Dyrmann, H. Karstoft, R. N. Jørgensen, and R. Gislum, "Semantic segmentation of mixed crops using deep convolutional neural network," 2016.

[18] M. Dyrmann, H. Karstoft, and H. S. Midtiby, "Plant species classification using deep convolutional neural network," *Biosystems Engineering*, vol. 151, pp. 72 – 80, 2016.

[19] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results," http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

[20] M. Cogswell, F. Ahmed, R. B. Girshick, L. Zitnick, and D. Batra, "Reducing overfitting in deep networks by decorrelating representations," *CoRR*, 2015.

[21] L. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *CoRR*, 2016.

[22] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *ECCV*, 2018.

[23] Y. Bengio, "Practical recommendations for gradient-based training of deep architectures," *CoRR*, vol. abs/1206.5533, 2012.

[24] F. Luan, S. Paris, E. Shechtman, and K. Bala, "Deep photo style transfer," 2017.

[25] A. Levin, D. Lischinski, and Y. Weiss, "A closed form solution to natural image matting," 2006.

[26] Y. Liu, "deep-photo-style-transfer-tf," https://github.com/LouieYang/deep-photo-styletransfer-tf, Alibaba-Zhejiang University Joint Research Institute of Frontier Technologies, 2017.

[27] G. Csurka, D. Larlus, and F. Perronnin, "What is a good evaluation measure for semantic segmentation?" *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, 01 2013.

[28] "Research priorities for robust and beneficial artificial intelligence," *Open Letter*, 2015.

[29] "Asilomar ai principles," *Asilomar Conference*, 2017.

[30] D. Mascharka, P. Tran, R. Soklaski, and A. Majumdar, "Transparency by design: Closing the gap between performance and interpretability in visual reasoning," 2018.

[31] U. of Edinburgh, "A third of fruit and veg crop too ugly to sell," 2018.

[32] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[33] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," 2014.

[34] P. T. Jackson, A. Atapour-Abarghouei, S. Bonner, T. Breckon, and B. Obara, "Style augmentation: Data augmentation via style randomization," 2018.

[35] E. Mohedano, K. McGuinness, X. Giró i Nieto, and N. E. O'Connor, "Saliency weighted convolutional features for instance search," 2017.