Tree segmentation by combining CNNs with engineered features

Michael Scott
Computer Science
University of Cape Town
Western Cape, South Africa
sctmic012@myuct.ac.za

Charl Ritter
Computer Science
University of Cape Town
Western Cape, South Africa
rttcha002@myuct.ac.za

Fergus Strangways-Dixon
Computer Science
University of Cape Town
Western Cape, South Africa
strfer001@myuct.ac.za

CCS Concepts

• Machine Learning
• Image Processing
• Computer Vision

KEYWORDS

Image segmentation, Convolutional Neural Networks, U-Net, Fully Convolutional Network, Atrous Convolutional Network, Per-pixel Transforms, Small Width Feature Extractor, Large Feature Extractor.

1 PROJECT DESCRIPTION

Aerobotics is a company in Cape Town that makes use of drones to capture high resolution images of consumer's tree farms, in a variety of colour spaces. Pixel information is extracted from these images (RGB, Near Infrared, chlorophyll absorption and height) and a neural network is used to segment the image into predetermined classes (healthy tree, grass, dirt, diseased tree etc). This is done to provide consumers with analytics and information about their farm, so that they can make decisions based on their crop's health.

A neural network is a network of interconnected layers and operations, each with their own associated weights. These operations and weights are applied to pixel information to make a decision on their classification. They are then adjusted by the errors produced in pixel classification. The network can "learn" a set of rules to classify image pixels and improves these rules as more data is provided. The term for this is *semantic segmentation* i.e. self-learning segmentation [6, 7, 15]. Currently the Aerobotics machine learning system achieves very accurate crop segmentation accuracy. Unfortunately, their system is unable to accurately segment image boundaries and edges and this provides an opportunity to explore methods to address this issue.

To obtain better accuracy, our project aims to take a novel approach to improve Aerobotics' image segmentation strategy, by using hand engineered features as input for *enhanced neural network models*. This is due to the fact that hand engineered features are traditionally used in computer vision to eliminate background noise and emphasize certain characteristics of objects that are of interest [14].

These *enhanced models* are neural networks that have been specifically developed and tweaked to segment images. They have seen great success on extremely noisy datasets and have performed an order of magnitude better than their competitors.

With this in mind, Aerobotics requested that our team investigate selected *enhanced neural network models* as a mechanism to process hand engineered features. The overarching aim of which is to investigate the accuracy implications of processing hand engineered features against raw images in these *enhanced models*.

There following neural network models will be investigated:

- Fully Convolutional Neural Network: A neural network using a fully connected layer to improve segmentation [15]
- U-Net: A neural network that uses skip convolutions, upsampling and concatenation to achieve even better segmentation accuracy [6]
- Atrous: A neural network that uses specialised convolutions to capture spatial information [7]

The categories of hand engineered features explored in this project are as follows:

- Per-pixel transforms: These are modifications dedicated to transforming single pixels
- Small width feature extractors: These are modifications dedicated to transforming a small neighbourhood of pixels
- Large feature extractors: These are modifications dedicated to transforming a large neighbourhood of pixels [14]

We intend to use these features and neural networks in a four-step process to investigate our aim. The first step of which is to pre-process the images using our feature extractors (i.e. hand engineered features), eliminating background noise. The second is to configure the neural networks for training. The third step is to train models with pre-processed images as an input as well as with raw images as a base case. The final step is to compare the resultant accuracy of these networks using metrics discussed further in this proposal.

2 PROBLEM STATEMENT

Although neural networks achieve state of the art segmentation accuracy for many applications. There are still a host of minor issues which can cause parties to question their usefulness in select scenarios. This is due to the fact that neural networks still produce segmented objects that have inaccurate boundaries or have low quality resolution (when compared to the original image). Furthermore, these architectures are data intensive, requiring huge training samples in order to learn basic image features [7, 15]. This is despite many advances made in neural network architectures, which suggests that a different approach should be used to solve these challenges.

These challenges are of a particular concern to Aerobotics for two reasons. Revenue and customers may be lost because of

incorrect advice given to the consumer, based on inaccurate segmentation. Secondly, training these architectures require lots of annotated data, which is costly particularly in their niche application.

Aerobotics, through internal investigations, have identified that using engineered features and modern neural network models might solve some of those problems. However, they want to investigate to what extent they do before going forward with any implementation.

In light of this, we have devised the following aims in order to address the above:

1. Investigate the feasibility of using feature extractors to decrease training time
2. Determine to what extent engineered features increase output accuracy (a 2% difference will be accepted as significant)
3. Test specific feature extractor and feature extraction combinations to determine the best combination to use as an input for a neural network
4. Investigate whether the performance of some feature extractors is dependant or independent of architecture

The aforementioned aims were then distilled into the following research questions:

1. Do traditional hand engineered features in tree crop segmentation, as inputs to select neural networks, significantly improve accuracy over their base neural network implementations.
2. Is the segmentation accuracy of certain hand engineered features dependant on the specific type of neural network it is used in or is their performance independent of neural network architecture?
3. Is there a significant positive or negative impact of the hand engineered features on the rate that these neural networks learn to segment tree crops?

3 RELATED WORK

As there are many ways to extract features from an image, the following literature was reviewed as a guideline for which combinations of feature extraction methods to perform.

In terms of colour. Wang et al., focused primarily on the L*a*b* colour space method, which is a small feature extractor. They used this method to identify trees in images by looking for colours matching the green spectrum. They found that the method was very fast but had some shortcomings when different plants that were next to each other, did not have a drastic enough difference in their shades of greens. This resulted in them being grouped together.[1] In a follow up paper Wang et al. separated individual trees from the surrounding landscape in images. For images with simple backgrounds, they proposed using Edge-detectors, however for images with more complex backgrounds they devised their own method, based on Edge-detectors coupled with colour space methods, of which they recommended the L*a*b* colour space method, and mathematical morphology based on their results [2].

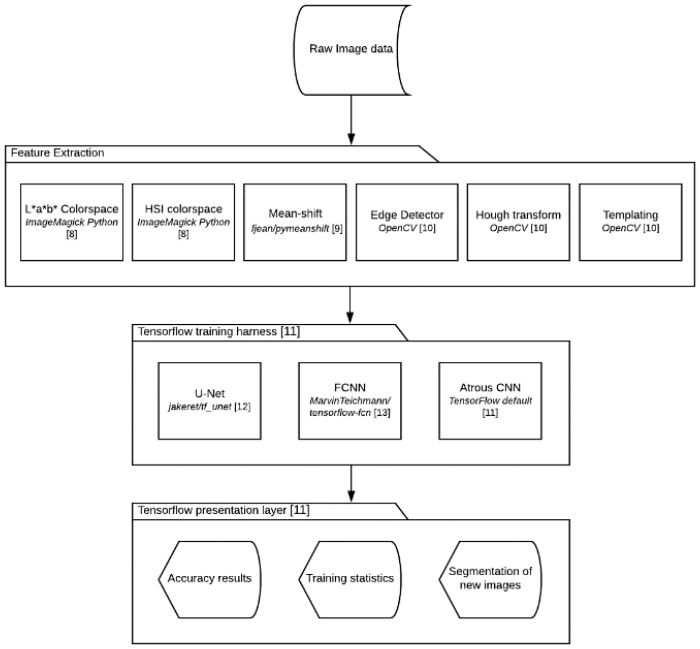
In the context of combining feature extractors to improve image segmentation, several papers proved useful. Zheng et al., firstly applied two per pixel feature extractor, namely HSI colour spaces RGB colour spaces, to the images. After this they applied a small feature extractor, namely Mean-shift, to the images. This separated the images into two parts, green and non-green vegetation. They tested this algorithm on a 100 images of different plant types, with different illuminations and soil types and achieved a median of mis-segmentation of 4.2% between green and non-green vegetation.[3] In the paper by Tao et al., they also use the Mean-shift feature extractor. They however apply this feature extractor first to the images and then they apply the Normalized cuts method. By applying the Mean-shift method first, it allows the Normalized cuts method to be applied to the resulting segments of the Mean-shift, rather than directly to the image pixels. They found this combination requires significantly lower computational complexity and therefore, is feasible for real-time image processing, while yielding superior image segmentation performance.[4] Finally, in their paper, Tellaeche et al., used a combination of per pixel feature extractors and large feature extractors. They first apply an RGB colour space method to the images, that results in a binary image. This image shows crops as white and the soil, stones and residual as black. After the binary image is obtained, they apply a Hough transform to it, to detect the furrows of the crops in the image.[5] Tellaeche et al., did not comment on the performance of this combination on its own, but as a whole used with Support Vector Machines. They found that it performed well, although results may vary in over exposed images. The use of combining feature extractors to improve image segmentation was established to have a high possibility of leading to improved image segmentation, through the literature review as well.

4 PROCEDURES AND METHODS

The proposed research project is split into two distinct components, namely the pre-processing for creation of engineered features, and testing the impact on accuracy that these features have on a variety of Convolutional Neural Networks (CNNs). We hypothesise manually highlighting features on behalf of a CNN will lead to an increase in the rate at which a model becomes accurate.

4.1 Design

Our overall structure for the software needed to test the hypotheses will be split into layers:



4.2 Development Tools

Our project will be developed in Python. This is because Python has many frameworks and libraries useful for our applications. The libraries used for image processing are ImageMagick and OpenCV as they have all the functionality we need to implement these our feature extractors. The library used for model training is the TensorFlow library. TensorFlow is a leading framework in the machine learning industry, therefore it has vast support and documentation allowing us to easily implement our design.

4.3 Implementation

The first component requires the creation of a driver class for the image feature extraction. This will be done as group. Implementing each of the actual transformations within this driver will be done individually, with each member implementing 2 of the 6 feature extractors.

As input for the next stage, one-many feature extractors will be applied to raw images, this will be guided by our own testing, our literature reviews and the related work section. These will then be fed into our architectures. In order to establish a base-accuracy to compare to, CNN models will also be trained on raw image data.

The second stage of testing these transformations with 3 different FCNN architectures will be performed individually. We will each be exploring one of: U-Nets, Atrous CNNs, and Fully Convolutional Neural Networks. All these architectures have implementations in Google's TensorFlow framework, which allows us to create a common training handler for them and perform minor tweaks per architecture.

The third stage will be to implement a presentation layer to collate and clearly show various statistical outputs measured from training. It will also be used to perform image segmentation comparisons.

5 RESULT EVALUATION

5.1 Evaluation Process

Testing the performance of our newly created CNN models will be crux of our implementation. We cannot justify the success of our feature extraction methods without objectively validating their accuracy improvements. We aim to create a reproducible testing process that will evaluate the improvement in accuracy of models using image feature extraction. We will each train and test our models first on unprocessed images to observe the base accuracy. Then we will train our other models on images with engineered features to validate the improvement in segmentation accuracy. Great care will be taken to ensure the raw and transformed images are in the same order and come from the same base image. As an extension, we hope to select a combination of these image transformations which will be guided by our own results as well as literature (some of which has been explored in related work). This combination will be similarly evaluated to determine its effectiveness.

An example of one test run would be:

1. Choose CNN architecture
2. Compute base accuracy on raw image data
3. Choose 1 or more image processing techniques
4. Compute accuracy using processed image data
5. Compare accuracy gains

During training the TensorFlow framework provides real time metrics for accuracy, allowing us to evaluate the segmentation accuracy, as well as the training time required to gain an acceptable level of accuracy. When we perform manual validation, we will compare the correct segmentation maps to the maps generated by the model trained on raw images, as well as processed images. Segmentation accuracy validation is represented as a percentage change measuring the predicted segmentation map to the correct one.

5.2 Measured Metrics

We will measure the base accuracy of the models from each architecture by training and evaluating them on the raw images received and comparing this accuracy to the accuracy of models trained and evaluated on images with the transforms applied. We will view any improvement of 2% or greater from the base accuracy as a significant improvement in segmentation accuracy.

An additional metric we wish to evaluate will be the accuracy over time for each model and feature extraction. We shall capture accuracy throughout intervals of the training process, for example at 25%, 50%, and 75% of the dataset.

6 KEY SUCCESS FACTORS

Key success factors are measures that should be reached in order accomplish one or more desirable goals. To achieve the goals, set by this research project, it is expected that the project achieves all or most of the following key success factors:

- Finding competent implementations of a FCNN model, a U-net model and an Atrous model.
- Building different feature extractors that perform well with the chosen CNNs.:
- Whether using engineered features as inputs to CNNs can improve their segmentation performance for tree data.
- Whether there are trends associated with certain CNN and feature extractor combinations
- What combinations of engineered features and CNNs perform the best.
- That the percentage increase in performance, between the CNNs using the different transformations and the CNNs using raw data, is a worthwhile increase.

7 ANTICIPATED OUTCOMES

7.1 Expected Impact

The results of this research project will benefit Aerobotics directly and will indirectly benefit any other parties doing similar research or feasibility studies. Currently, we hypothesise that the use of engineered features as inputs for various CNNs will improve segmentation performance on tree data. It was, however, observed that segmentation performances between different unsupervised architectures had an almost negligible difference in the project preceding this one. Similarly, many research projects dedicate a huge amount resources into developing new CNNs or engineered feature models, only to experience a very small increase in either efficiency, speed or both. Thus, although it is expected that we will improve the segmentation performance for tree data of CNNs, the margin of this improvement and how worthwhile the improvement will be is uncertain. The outcome of this project will allow any groups wishing to do research in this field to decide whether they wish to: adopt this technology into their projects, invest more resources into better researching a certain aspect of the project or to discontinue research in this field in favour of another. It will also display the effectiveness or ineffectiveness of combining feature extractors to improve image segmentation, allowing other researchers to build on this premise and develop better feature extractor methods or combinations thereof.

7.2 System Outcomes

One system outcome is the feature extractors that will be built for this project. For this project there will be six different feature extractors, two methods from each of the three types of feature extractors, namely per pixel feature extractors, small feature extractors and large feature extractors. The two per pixel feature extractors are, L*a*b* colour spaces and HSI colour spaces, the two small feature extractors are, Edge-detectors and Mean-shifts, and the two large feature extractors are, Hough transforms and Templating.

Another system outcome is the Convolutional Neural Networks that will be used for this project. There are three CNNs that will be used, namely a Fully Convolutional Network, a U-net and an Atrous Convolution Network.

The last system outcome is the accuracy tests. Firstly, a ground truth accuracy test will be performed on the CNNs, with raw image data. After this, multiple accuracy tests will be performed, one for every type of feature extractor and one for every chosen combination of the feature extractors. Every one of these tests will also be split into four tests, each testing different sizes of the data set used to train the systems. This split will be on 25%, 50%, 75% and 100% of the dataset to measure the amount of data required to achieve industry-standard accuracy for each combination of transformation and CNN model.

8 ETHICAL, PROFESSIONAL AND LEGAL ISSUES

We do not foresee any ethical issues as we are not researching humans or animals, using personal data, or generally interacting with people outside of the project team. As this project was initially put forward by Aerobotics, and they are providing the training data for the CNN models, they will expect some form of acknowledgement in our final paper. We do not foresee them making claims to intellectual property, as the engineered features were not novel ones provided by Aerobotics, and the platform we design to test the hypothesis will be our own, and our supervisor's, intellectual property.

We will also be extending software licensed under GPL-3.0 (github.com/jakeret/tf_unet & github.com/fjean/pymeanshift). This means our sections handling mean-shift transformations, as well as the U-Net implementation will need to be released under the same licence.

9 PROJECT PLAN

9.1 Risks

See appendix A.

9.2 Timeline

See appendix B and C.

9.3 Deliverables

- Literature Review 02 May
- Project Proposal 23 May
- Revised Proposal 10 June
- Paper Plan 27 June
- Background 19 July
- First Implementation 19 July
- Final Prototype 29 July
- Implementation and Testing 02 August
- Outline of Complete Paper 09 August
- Final Draft of Paper 16 August
- Poster 23 September
- Web Page 30 September
- Reflection Paper 07 October

9.4 Milestones

- Presentation of Proposal 27 - 29 May
- Feasibility Demonstration 15 - 19 July
- Final Paper Submission 26 August
- Final Code Submission 02 September
- Final Project Demonstration 02 - 16 September

9.5 Resources Required

We will require the training data from Aerobotics. This will be in the form of pre-labelled images from drone mounted sensors, layered with the segmentation map. We may need to improve the accuracy of the maps provided using Fiji. Fiji is an open source image processing software package that has native support for segmentation map creation and adjustment. We will also need to use the Python implementations of ImageMagick and OpenCV to handle the image feature extractions. We will use the TensorFlow framework to handle all aspects of the machine learning layer. TensorFlow is created by Google, and provides comprehensive support for model design, training and usage. There are various GitHub repositories provided highlighted in the Procedures section, we will need to use these to provide the foundations of our models. The repositories are all licenced with Open Source licences (GPL-3.0 & MIT). As we are using a package licensed with GPL-3.0, those sections of the platform will need to be open sourced under the same licence as well. These sections will be the mean-shift transformation, and the U-Net implementation. Our only costly requirement may be access to a GPU cluster for training our models, this will only be necessary if the data set provided by Aerobotics to too large to effectively train on our own personal PCs.

9.6 Work Allocation

As mentioned above, there are 3 layers to our experimental system. For layer 1 - image processing, we will create a basic handler/API to process images together. The implementation of each of the transformations and feature extractions will be individually. Fergus will implement the per-pixel transformations (HSI and L*a*b* colour space transforms). Charl will implement the local-area feature extractors (Mean-Shift and edge detectors). Michael will implement the large-scale feature extractors (Hough transforms and Templating).

Layer 2 is responsible for the CNN models. Together we will create a simple interface for interacting with the training harness. We will implement training procedures for each of the CNN models specified individually. Fergus will implement the framework around the U-Net model, Charl will implement the FCNN model, and Michael will implement the Atrous model. This will result in a program able to import images of all forms from layer 1 and train a certain model on them.

Layer 3 is the presentation layer. As TensorFlow provides a common interface for monitoring and using models, this will be done together, and will provide feedback for an end user wishing to segment new images.

REFERENCES

[1] Xiaosong Wang, Xinyuan Huang, and Hui Fu. 2010. A Colour Texture Segmentation Method to Extract Tree Image in Complex Scene. 2010 International Conference on Machine Vision and Human machine Interface (2010). DOI:http://dx.doi.org/10.1109/mvhi.2010.138

[2] Xiao Song Wang, Xin Yuan Huang, and Hui Fu. 2009. The Study of Colour Tree Image Segmentation. 2009 Second International Workshop on Computer Science and Engineering (2009), 303 307. DOI:http://dx.doi.org/10.1109/wcse.2009.818

[3] Zheng, L., Zhang, J. and Wang, Q., 2009. Mean-shift-based color segmentation of images containing green vegetation. Computers and Electronics in Agriculture, 65(1), pp.93-98. DOI:https://www.sciencedirect.com/science/article/pii/S0168169908001828

[4] Tao, W., Jin, H. and Zhang, Y., 2007. Color Image Segmentation Based on Mean Shift and Normalized Cuts. IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics), 37(5), pp.1382-1389. DOI:https://ieeexplore-ieeeorg.ezproxy.uct.ac.za/stamp/stamp.jsp?tp=&arnumber=4305291

[5] Alberto Tellaeche, Gonzalo Pajares, Xavier P. Burgos-Artizzu, and Angela Ribeiro. 2011. A computer vision approach for weeds identification through Support Vector Machines. Applied Soft Computing 11, 1 (2011), 908–915. DOI: http://dx.doi.org/10.1016/j.asoc.2010.01.011

[6] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam 2017. Rethinking Atrous Convolution for Semantic Image Segmentation. CoRR abs/1706.05587 (2017).

[7] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-Net Convolutional Networks for Biomedical Image Segmentation. Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015 9351 (2015), 234–241. DOI:http://dx.doi.org/10.1007/978-3-319-24574-4_28

[8] Anon. ImageMagick - Python. Retrieved May 21, 2019 from https://wiki.python.org/moin/ImageMagick

[9] Frédéric Jean. 2015. fjean/pymeanshift. (August 2015). Retrieved May 21, 2019 from https://github.com/fjean/pymeanshift

[10] Olli-Pekka Heinisuo. opencv-python. Retrieved May 21, 2019 from https://pypi.org/project/opencv-python/

[11] Anon. TensorFlow. Retrieved May 21, 2019 from https://www.tensorflow.org/

[12] Joel Akeret. 2019. jakeret/tf_unet. (May 2019). Retrieved May 21, 2019 from https://github.com/jakeret/tf_unet

[13] Marvin Teichmann. 2017. MarvinTeichmann/tensorflow-fcn. (December 2017). Retrieved May 21, 2019 from https://github.com/MarvinTeichmann/tensorflow-fcn

[14] Yu-Jin Zhang. 2006. An Overview of Image and Video Segmentation in the Last 40 Years. Advances in Image and Video Segmentation (2006), 1–16. DOI: http://dx.doi.org/10.4018/978-1-59140-753-9.ch001

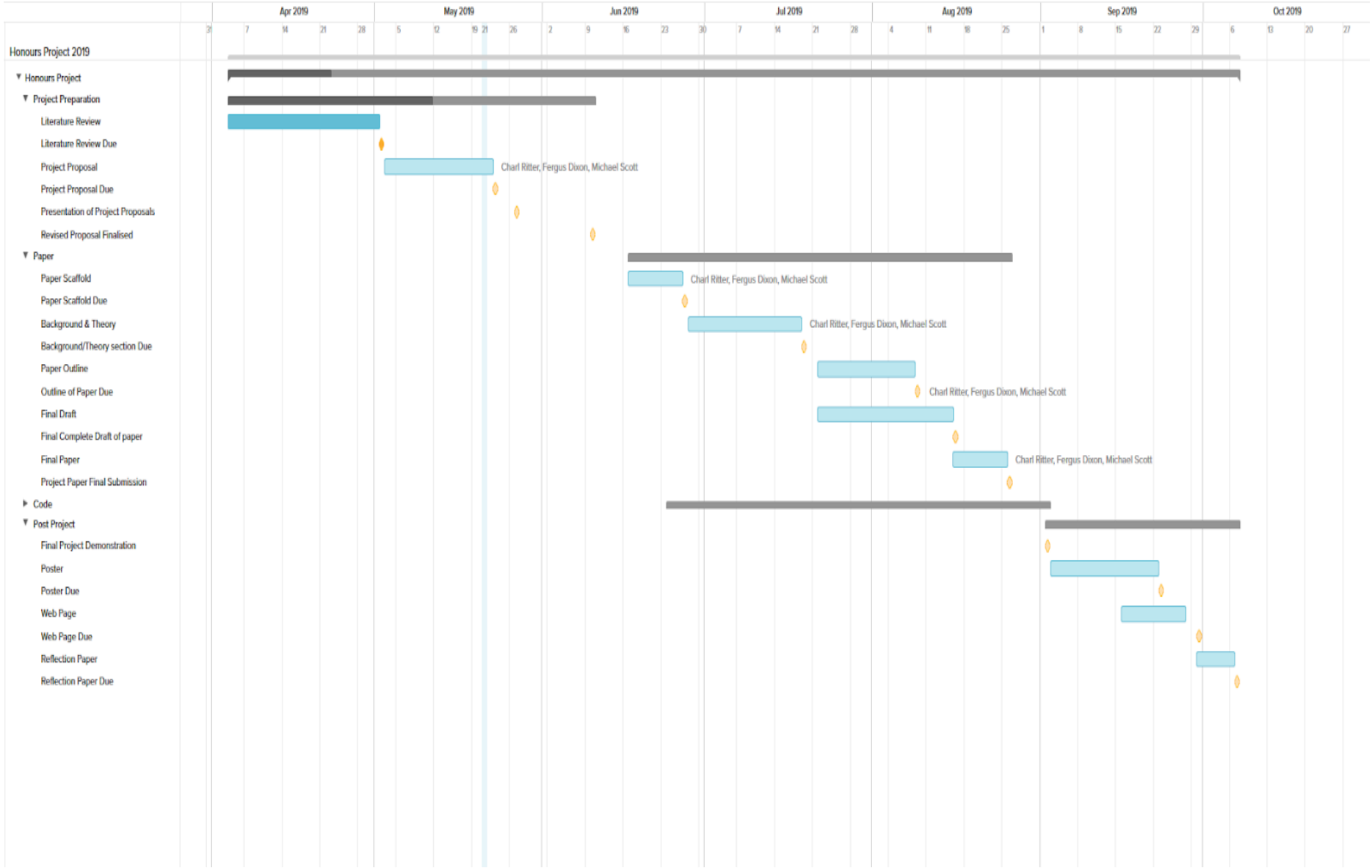
[15] Jonathon Long, Evan Shelhammer, Trevor Darrell. 2015. Fully convolutional networks for semantic segmentation. The IEEE Conference on Computer Vision and Pattern Recognition (CPVR) (2015), 3431-3440. DOI: https://doi.org/10.1109/cvpr.2015.7298965

APPENDICES

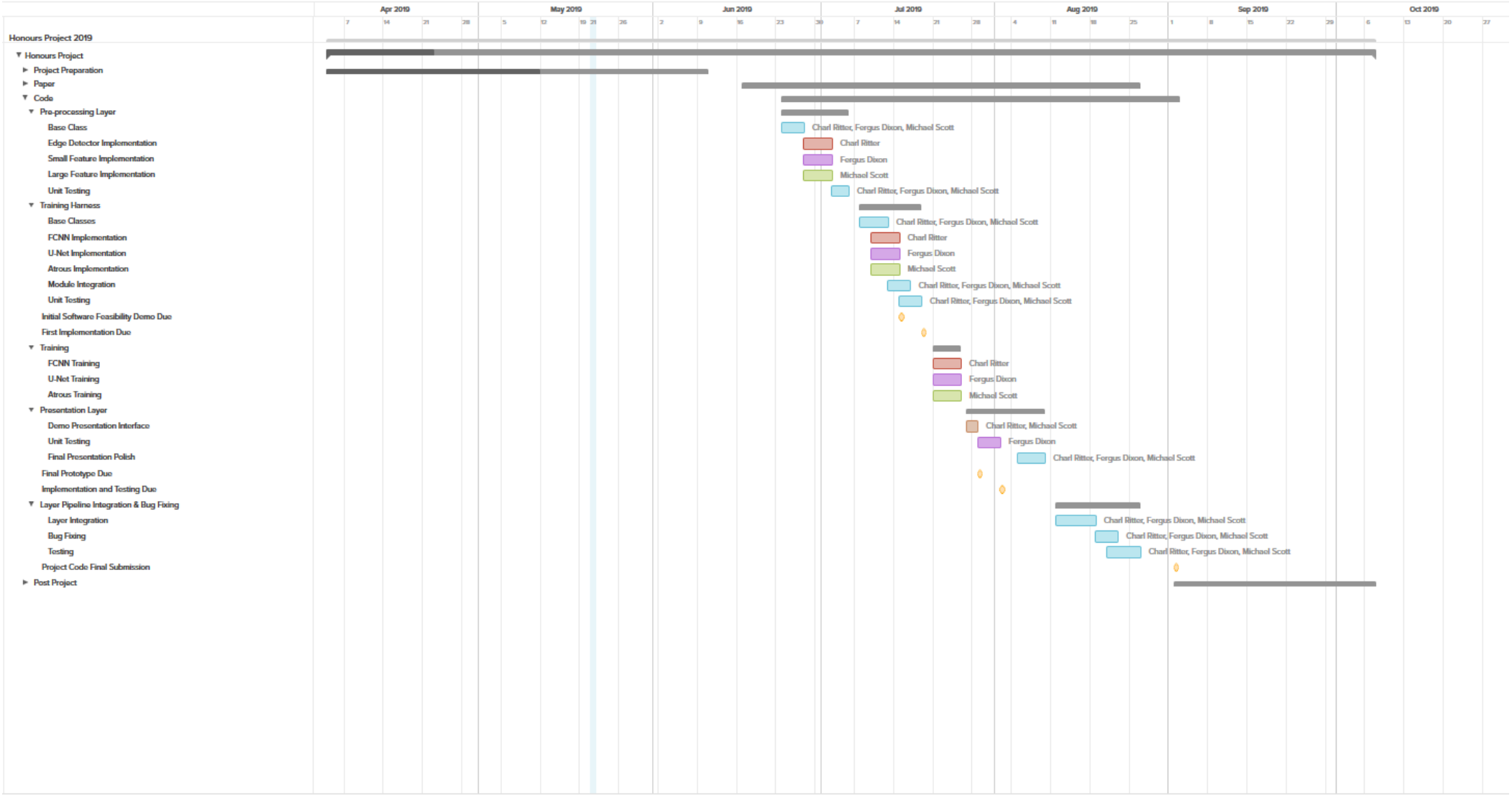
Appendix A:

Risk	Impact	Probability	Mitigation
Integration issues	High	Low	Research common frameworks to perform functionalities and maintain constant communication
Inaccurate ground truth data	Medium	High	Reannotate samples with Fiji. If data annotation is not usable, use an existing trained architecture to automatically reannotate
Long training times for neural networks	Medium	Medium	Ask permission to use a cluster to speed up model training
Number of models to train cause bottleneck	High	Low	Train models on many different clusters
Implementation difficulties with image extractors or neural network models	Medium	Low	Research alternative implementations of the models.
Feature extractor does not improve accuracy	Medium	Very low	Research image extractors that work well together and perform eye test on some of the samples that the feature extractor has processed
CNN model not implementable	Medium	Low	Discard CNN model and continue with other 2 functional models

Appendix B


Apr 2019
May 2019
Jun 2019
Jul 2019
Aug 2019
Sep 2019
Oct 2019
Honours Project 2019
Honours Project
Project Preparation
Literature Review
Literature Review Due
Project Proposal
Project Proposal Due
Presentation of Project Proposals
Revised Proposal Finalised
Paper
Paper Scaffold
Paper Scaffold Due
Background & Theory
Background/Theory section Due
Paper Outline
Outline of Paper Due
Final Draft
Final Complete Draft of paper
Final Paper
Project Paper Final Submission
Code
Post Project
Final Project Demonstration
Poster
Poster Due
Web Page
Web Page Due
Reflection Paper
Reflection Paper Due
Charl Ritter, Fergus Dixon, Michael Scott
Charl Ritter, Fergus Dixon, Michael Scott
Charl Ritter, Fergus Dixon, Michael Scott
Charl Ritter, Fergus Dixon, Michael Scott
Charl Ritter, Fergus Dixon, Michael Scott

Appendix C


Apr 2019
May 2019
Jun 2019
Jul 2019
Aug 2019
Sep 2019
Oct 2019
Honours Project 2019
Honours Project
Project Preparation
Paper
Code
Pre-processing Layer
Base Class
Edge Detector Implementation
Small Feature Implementation
Large Feature Implementation
Unit Testing
Training Harness
Base Classes
FCNN Implementation
U-Net Implementation
Atrous Implementation
Module Integration
Unit Testing
Initial Software Feasibility Demo Due
First Implementation Due
Training
FCNN Training
U-Net Training
Atrous Training
Presentation Layer
Demo Presentation Interface
Unit Testing
Final Presentation Polish
Final Prototype Due
Implementation and Testing Due
Layer Pipeline Integration & Bug Fixing
Layer Integration
Bug Fixing
Testing
Project Code Final Submission
Post Project
Charl Rittor, Fergus Dixon, Michael Scott
Charl Rittor
Fergus Dixon
Michael Scott
Charl Rittor, Michael Scott