Non-Photorealistic Rendering with Spot Colour
Final Year Project Report

Abstract

In this report the idea of spot colour has been used to produce image processing effects that will highlight key areas within an image, enhancing the impact that the region of interest has over the whole image.

Joseph Wilson-Hall

Contents

Table of terms 2
Acknowledgements 3
1. Introduction 4
2. Related Research 5
3. Project Background 9
3.1. Spot Colour 9
3.2. Region of Interest 9
3.3. Ground Truths 10
4. Research 11
4.1. Machine Learning 11
4.1.1. Convolutional Neural Network 12
4.1.2. Single-Shot Detector 13
4.1.3. SSD with MobileNetV3 13
4.1.4. RCNN 14
4.1.5. Mask-RCNN 15
4.2. COCO 16
4.2.1. COCO dataset 16
4.2.2. COCO SSD network MobileNetV3 17
4.2.3. COCO Mask-RCNN 17
4.3. Algorithms 18
4.3.1. Intersection over Union 18
4.3.2 Saliency 19
4.3.3. Pixel Neighbours 19
4.3.4. Adaptive Edge Detection 20
4.4. Dataset 21
5. Design 22
5.1. Overview Design Change 22
5.2. Agile Design Methodology 23
5.3. Source Code Layout 25
5.4. V1 26
5.5. V2 28
5.6. V3 31
5.7. V4 33
5.8. V5 36
6. Testing 37

6.1. Image Evaluation 38
6.2. Initial Testing 38
6.3. K-fold Cross-Validation 39
6.4. ANOVA 39
7. Conclusions 40
7.1. Results from Initial Testing 40
7.2. Results from Final Testing 44
7.2.1. 50-Fold Cross-Validation 44
7.2.2. ANOVA 51
Test between all groups of data 51
V1 : V5 51
V2 : V5 51
V3 : V5 52
V4 : V5 52
7.2.3. Hypothesis 54
7.3. Conclusion 61
7.4. Future Work 62
References 65

Table of terms

1. ROI: Region of Interest
2. SSD: Single Shot Detector
3. CNN: Convolutional Neural Networks
4. RCNN: Region-Based Convolutional Neural Networks
5. Mask RCNN: Mask Region-Based Convolutional Neural Networks
6. AI: Artificial Intelligence
7. COCO: Common Objects In Context
8. IoU: Intersection over Union
9. NPR: Non-Photorealistic Rendering
10. YOLO: You Only Look Once
11. ANOVA: Analysis of Variance
12. GUI: Graphical User Interface

Acknowledgements

I would like to thank all my friends and family that have motivated and supported me throughout this year. I would also like to thank my friends who took the time to participate in this project and create ground truths for my data. Finally, I would like to thank my supervisor Professor Paul Rosin who has guided me throughout this project and without his keen insight into this area of study I would not have been able to produce this project.

Thank you.

GitHub Repo: https://github.com/jwilson-hall/Spot-Colour

1. Introduction

Non-Photorealistic rendering with spot colour is the process of taking an image as input and applying a spot colour algorithm which will identify an object or region within the image. The algorithm will emphasize the region by converting everything but the selected region into greyscale.

This report focuses on research into different methodologies used to retrieve an ROI from an image without the prior context of what the image contains. The region is going to be used for spot colour which is why we are trying to identify a ROI. Identifying suitable regions for spot colour is not always black and white in terms of which region is best suited for this, which has been accounted for within the testing section of this report. Different people may find different aspects of an image intriguing which leads to different interpretations of which areas within an image would be best suited for spot colour.

The ideal result would be that the program returns a perfect mask of an object from an image that has an obvious region of interest. Throughout this study, different methodologies that attempt to solve this problem will be discussed, as well as methods that we have implemented to test on our own data sets. To test the methodologies that were implemented for the purpose of this research, ten images were collected and the focus group consisting of five people would then suggest where the ground truth/ region of interest should be for that image. This allowed for some variation in where the region of interest should be as different people may have different interpretations as to where the focus of the image is.

Ten images is not a large dataset for testing which is why this has been changed later on in the report. Twenty images were randomly selected images from the 2017 COCO unlabelled dataset were used as it was deemed that those ten images were suitable enough for spot colour. The reasoning behind having another ten images is to evaluate the accuracy of the methodologies and to make sure that how well each method works was tested fairly, while implementing different methods the first ten images were used to train and adjust variables in an attempt to improve the method. By adjusting the variables, the aim was to improve the overall precision across all images, not just the ten images that were used to train the method. Once each method had been tested, the methods were then tested on the COCO data set to

test the true accuracy of each method. This was done because while adjusting variables for one set of images might improve the precision for that particular set, it may not be so effective when run on a completely different data set.

The final goal of this project is to be able to process an image and identify the region that should be used for spot colour. Due to the difficulties that identifying a ROI pose, this project and any researched carried out in it can contribute to improving the chances of algorithms coming close to meeting the standards held by human interpretation. Using the detected region, the program will use the mask as a guide to determine if a pixel in an image should be in colour or in greyscale. This report will discuss related research to this area and what other methodologies have been able to produce and how they may compare to methods that have been implemented within this report. During this project five different methods have been produced and using intersection over union on ground truths, the effectiveness of each of these methods has been evaluated.

2. Related Research

This project draws inspiration from a few different papers with different approaches to solve the same problem of retrieving the region of interest within an image or frame of video footage. Related papers include; [[1],[7],[12],[20]].

The research paper written [20] called “Non-Photorealistic Rendering with Spot Colour” is the most similar to this project as they are both attempting to identify regions where it is appropriate to apply Spot Colour. The approach [20] used was quite different to any of the approaches used within this paper. Their algorithm takes an RGB image and a monochromatic NPR image as input and converts the RGB image into HSV (Hue, Saturation, Value) channels which is a colour representation model based on the human perception of colour.

The first step in the algorithm is converting the input image from RGB colour channels into HSV channels. Next a gaussian kernel with a variance of 4 is used to remove noise from colour channels. After the smoothing thresholding is applied to the hue in order to represent image with a smaller number of colours [20]. The image ends up consisting of three types of regions black, grey and white [20].

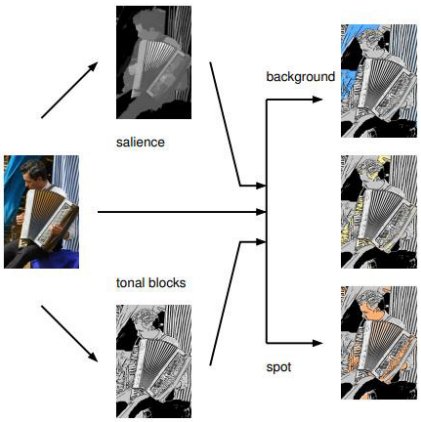
The spot colour algorithm is applied to the grey areas, meaning the threshold values are computed from hues corresponding to the grey area [20]. The hue layers are analysed in order to determine which is most appropriate for spot colour [20]. Circular statistics are applied instead of linear statistics, which applies to the smoothing and thresholding of the selected hue layer. Using the circular mean hue and mean saturation of the selected hue layer they apply the mean hue and saturation to mid tonal pixels belonging to that selected hue layer. Looking at **Figure 1**(b) the proposed pipeline can be seen [20].

The result is very similar to what this report aims to produce, however, the methods used and approach to this problem are very different. See Figure 1 for the results of the method.

Figure 1: (a) Example processing images from [20] (b) Example of proposed method from [20]

(a)

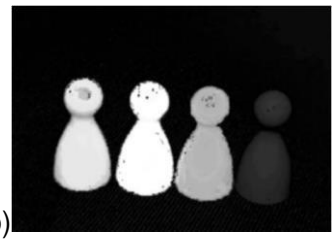
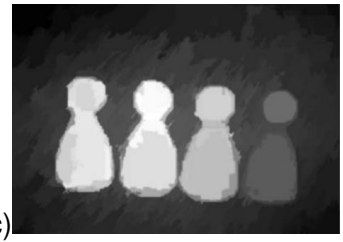
(b)



The previous method selected a hue layer to identify which region should be used for spot colour, this only accounts for a single channel which limits what an applicable region can be. For example, a particular ROI may have multiple colours within an object/area, which would be discarded using this method. After further research, the next step was looking into different methods to obtain the ROI from an image, this led to salient maps and how contrast within an image can be used to identify a ROI.

The research paper [7] called “Global Contrast Based Salient Region Detection” attempts to retrieve a region of interest from within any image without prior context, and this is similar in style to this project. The method uses a global contrast-based method that separates a large-scale object from its surroundings [7] In order to look at the global contrast, this method uses a histogram-based contrast to define the saliency values for the image using colour statistics like looking at the colour contrast of a pixel to all other pixels within the image [7]. Through this method, it can produce very accurate binary masks as for what someone might consider to be the region of interest within the image. The other Contrast-based method from this paper is a Region-based Contrast method which works producing finer detailed masks at the cost of being more computationally demanding to run. The paper gathered the largest publicly available image data set and compared their algorithm against eight other methods, which are considered to be ‘state-of-the-art’ methods [7]. Through their experiments, they found that the proposed methods outperformed the other algorithms in terms of precision and recall while still being simple and efficient. See Figure 2 for input image(a), example binary mask from Histogram based Contrast method(b) and Region-based Contrast method(c).

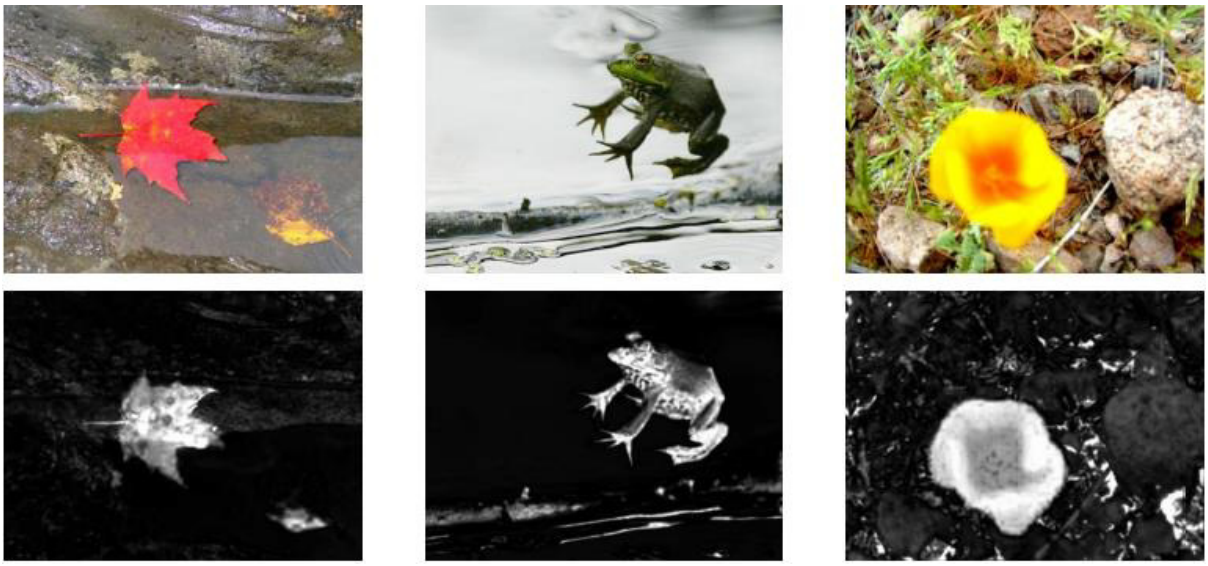
Figure 2: image(a), example binary mask from Histogram based Contrast method(b) and Region-based Contrast method(c) – from paper [7]

(a) (b)  (c) 

The research paper [1] called “Frequency-tuned salient region detection” is focused on retrieving the region of interest through frequency tuned salient regions, which means that they will estimate centre-surround contrast using colour and luminance features [1]. In the paper it is claimed that this approach offers three advantages over existing methods at the time of its publication - those three advantages being; uniformly highlighted salient regions with well-defined boundaries, full resolution and is computationally efficient [1]. The salient map produced by this method has many different applications within image processing. When this paper was produced it outperformed other ‘state-of-the-art’ methods in precision and recall, similar to [3]’

However, [3] is a more recent method and does outperform this method of producing a salient map.

Figure 3: Example processing Image from [1]



The relevant literature to this paper includes methods that focus on obtaining salient maps which are biologically motivated, purely computational or a mixture of these methods [[1], [7], [12], [17], [20]]. While this type of work is relevant there is also a lot of literature that takes the approach of using neural networks to obtain a region of interest from within an image which includes paper [11]. Using a mixture of these methods, this paper illustrates how a combination of these methods performs. I think one of the most effective and computationally efficient methods which I would have liked to implement would be [7], the drawback of any of these methods is that it requires an input image that it is suitable for this algorithm as some input images won’t have good results.

3. Project Background

In this section of the report the following sections explain key aspects project.

3.1. Spot Colour

Spot colour is the process of taking an image and identifying interesting objects or regions within the image to then exaggerate and bring more attention to by making the rest of the image greyscale, thus creating a large contrast between the ROI and the background of the image. There are many different motivations that people have when they attempt to use this process which include; making an image more interesting and artistic, commercially illustrative pieces and personal use to place emphasis on particular regions. This sort of method is very common among artists and photographers. It is common for many artists to use more bland and dull colours to emphasize the brighter and more interesting area within a painting. It is also common to find Spot Colour in many photographer's images, for example if you use Google images and type "selective colouring" you will find an enormous amount of Spot Colour images produced by many photographers. It is the contrast between the grey and colour that makes this style so eye catching and pleasing, due to its ability to easily emphasize or create a focus in the image. See **Figure 6** for an example of spot colour used in photography.

Figure 6: (a) Suitable spot colour Image (b) processed image with spot colour

(a) (b)

3.2. Region of Interest

The region of interest can be subjective when it comes to how different people view an image. In certain images where there is a distinct focus within the image like in **Figure 6. Figure 6** is a great example of an easily identifiable region of interest/spot colour region due to the background being blurred emphasizing the focus of the image.

Retrieving a region of interest is a crucial step for many different computer vision applications, such as image editing software, pedestrian detection, parking occupancy detection, healthcare, x-ray analysis, crop and yield monitoring - plus many more applications, as it can be implemented into a variety of different situations to automate tedious processes for humans. The main objective when analysing an image is to retrieve the region that can be seen as the most 'important', telling a program to do this is rather complicated as it is open to human interpretation as to what is most important within a particular image. In this report, the focus is on retrieving an ROI for the purpose of making that region more interesting and artistic. While this is the main focus, some acknowledgements need to be made about what images are well suited for identifying a Region of interest.

3.3. Ground Truths

To effectively test each version of the pipeline it was important to come up with some sort of way to measure how close the method was to an ideal output, hence the implementation of a ground truth for each image. To make it more realistic there is some leeway given to the different versions of the pipeline as some images it may be more difficult to identify the ROI, by creating five ground truths for an image will take into account that different people and possibly AI models may identify different regions in an image to be the focal point. To create these five ground truths five people participated in identifying different regions within an image that they believe to be the region of interest. Comparing the output against the five ground truths obviously created a lot of variation as to how similar they are to each ground truth, so the ground truth that scored the highest in similarity was used for each image as it means that the output is the closest to that interpretation of the image. When the images are compared the binary masks that are used as a guide as to where the image should be in colour are compared as this is an easier comparison than using an RGB image. To compare the similarity between images 2 different methods were considered such as, mean squared error and intersection over union. Mean squared error will return a value that is closer to zero the closer the images are together, while this is a perfectly valid way to compare two images to judge the differences the ideal output would be a percentage value that represents how similar they are not how different they are which is why IoU is used and discussed later in this report.

4. Research

Before approaching the problem, initial research was critical to determine how the problem has been solved previously. The goal of this project was to implement some form of machine learning/ artificial intelligence to identify the spot colour region within an image, as set out in the initial plan for the project. Initial research led me to looking into salient maps and how they are used to identify regions of interest. While the methods proposed in the papers have a lot of promise for reaching a solution, they do not implement any kind of machine learning. Salient maps are used to identify regions of interest which doesn't always mean that it has identified a suitable region of interest for spot colour.

After reading papers [[1], [7], [20]], it became apparent that salient maps are effective and efficient at identifying ROI's however, this did not quite fit the purpose of the project so this was avoided in favour of more AI-based solutions to the problem. Using a combination of AI and salient maps would be an effective way to tackle this problem and so in retrospect, that would be the preferred method.

4.1. Machine Learning

Later research led me to explore SSD network with MobileNetV3, CNN, RCNN and Mask RCNN. These networks have been used to train models on large-scale datasets to identify specific and generic objects. CNN (Convolution Neural Network) and RCNN (Region-Based Convolutional Neural Network) are machine learning models that are used for computer vision tasks, typically for object detection.

After researching different network models to use for image processing, I decided that Mask RCNN and SSD MobilenetV3 would be the best fit to solve the problem that this report is working toward solving. In combination with these image processing models, research was also conducted into salient maps to determine if implementing some generic saliency methods would be beneficial in highlighting particular regions of interest; there was some degree of accuracy after initial testing, although it was not something that was prioritised as the goal of the project was to implement AI in order to reach a solution.

4.1.1. Convolutional Neural Network

Within the area of Machine Learning NN's and CNN's are used in all manner of technologies in today's world, the type of problems they can be used for including image processing, classification prediction and regression prediction problems. The most common use of CNN's can be found in image processing applications in order to complete specific tasks such as identifying words from images which many people take for granted now that it is commonplace with everyone's phones. An application of this can be found in Google's Google Lens application which processes images and is able to use context in the image to apply appropriate searches, one of which includes the automatic translation of words into the default language of the phone. [2].

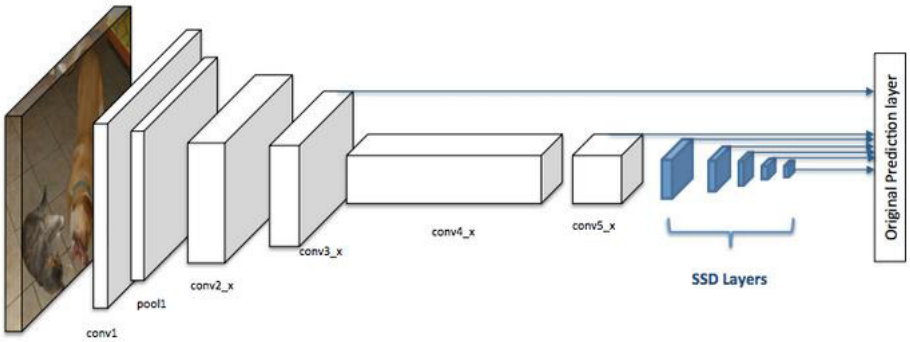
A CNN will typically take an image as input; however the CNN will view the different channels as a layer meaning that the an RGB image has a depth of 4. When images are viewed on a computer a human will only perceive the mix of these colours and as a single layer. A CNN can be split up into three layers which represent different steps in the process that an input image will go through, these include the convolutional layer, the activation layer, the pooling layer and the fully connected layer which will then provide the output of the network. The Convolutional layer can be considered to be the core of the network as this is where the most computationally heavy tasks are completed. Within the convolutional layer a 3d kernel filters are used to calculate the element-wise product of the image and then using the sum of those values per kernel to output a 2d matrix. The second layer of the CNN can be referred to as the activation layer, this layer is responsible for changing the summed values from the node into the activation of the or output for that input. To change the activation for a node the "rectified linear activation function" also known as ReLu, which is a piecewise linear function that will only output the input if it is positive, or it will output zero. The third layer within the CNN is the pooling layer which is used to progressively reduce the spatial size of the input in order to reduce the number of parameters and computation, a common approach to pooling is to use a 2 cross 2 max filter with a stride of 2. The final layer of the CNN is the fully connected layer which transforms the pooled feature map matrix into a single column which is the parsed to the to the fully connected layers meaning that

all the nodes in the preceding layer are connected to all the nodes in the succeeding layer which is then parsed an activation function like ReLu.

4.1.2. Single-Shot Detector

An SSD has two key components that make it so effective for image processing, this includes a backbone model and the SSD head. The backbone model is typically a pre-trained image classification network but excludes the fully connected classification layers which is used to extract features from input images. The SSD head is one or more convolutional layers that get added to the chosen backbone model which will usually output bounding boxes, object classification or both the bounding boxes and the object classification for that bounding box. See **Figure 7** below from [16] for an example of the architecture of a CNN with an SSD detector.

Figure 7: SSD Architecture Diagram from [16]

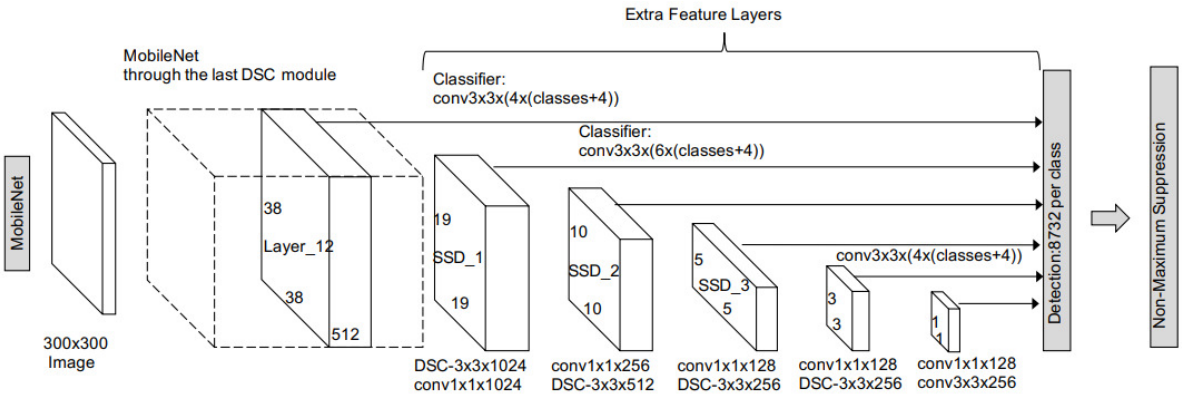


4.1.3. SSD with MobileNetV3

SSD with MobileNetV3 is an implementation of machine learning and is a very popular algorithm in object detection. The SSD network with MobileNetV3 that was implement has been trained using the COCO dataset allowing it to identify 91 different classes of objects from images. The base of the architecture consists of MobileNetV3 followed by several convolutional layers from the SSD network. This model takes an image as input and is capable of returning bounding boxes of a detected object within an image with the corresponding classification. The implementation of the MobileNetV3 makes it faster than other implementations like Fast RCNN and RCNN.

See **Figure 4** [22] for an example on the architecture for the SSD network with MobileNetV3.

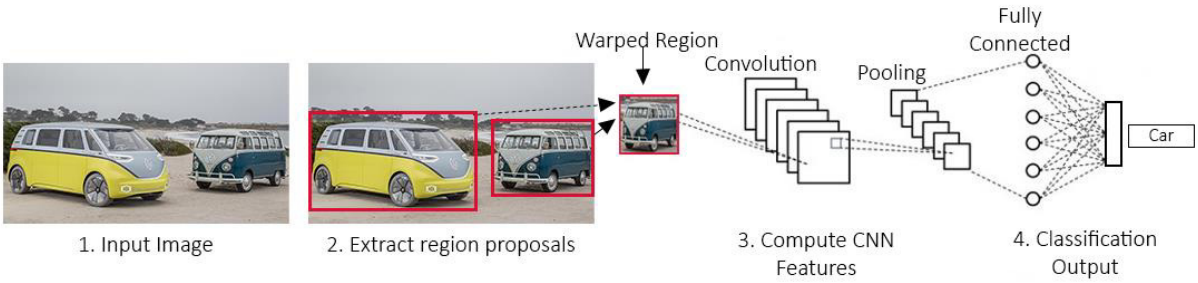
Figure 4: SSD MobileNetV3 Architecture Diagram from [22]



4.1.4. RCNN

RCNN is a CNN with a proposed region-based selection step that happens before the image is parsed to the CNN. See **Figure 5** for an example of what an RCNN model does when given an input image; based on what the network is used for, it may output a mask for a region or just classify a region within an image like the example in **Figure 5**.

Figure 5: RCNN Diagram



As shown in **Figure 5,** before a region is parsed to the CNN, the RPN (Region Proposal Network) identifies and extracts regions from an input image and resizes the proposed regions into a pre-determined input size for the network to process. This type of network is very effective for returning bounding boxes for regions of interest within images.

4.1.5. Mask-RCNN

Similar to RCNN, Mask RCNN is able to produce regions where the classified object is within the image. While it is similar to the RCNN model in terms of structure due to it being based on top of Faster RCNN its function is different, the aim of Mask-RCNN is to produce a mask/region where a detected object is located. The concept of producing a mask for any given image is known in the image processing field as Image segmentation. Image segmentation is when the model is able to distinguish between different objects in an image and highlight those selected regions where it can find these features. Why is Mask-RCNN so important? Mask-RCNN is able to outperform most other models available today in terms Accuracy and recall. See **Table 2** from [10] where the tests on Mask RCNN proved that it outperformed the base variants of all previous state-of-the-art models [10].

Table 2

	backbone	AP^{bb}	AP^{bb}_{50}	AP^{bb}_{75}	AP^{bb}_{S}	AP^{bb}_{M}	AP^{bb}_{L}
Faster R-CNN+++ [19]	ResNet-101-C4	34.9	55.7	37.4	15.6	38.7	50.9
Faster R-CNN w FPN [27]	ResNet-101-FPN	36.2	59.1	39.0	18.2	39.0	48.2
Faster R-CNN by G-RMI [21]	Inception-ResNet-v2 [41]	34.7	55.5	36.7	13.5	38.1	52.0
Faster R-CNN w TDM [39]	Inception-ResNet-v2-TDM	36.8	57.7	39.2	16.2	39.8	**52.1**
Faster R-CNN, RoIAlign	ResNet-101-FPN	37.3	59.6	40.3	19.8	40.2	48.8
Mask R-CNN	ResNet-101-FPN	38.2	60.3	41.7	20.1	41.1	50.2
Mask R-CNN	ResNeXt-101-FPN	**39.8**	**62.3**	**43.4**	**22.1**	**43.2**	51.2

Mask RCNN is used in two versions of the pipeline, and within one of those pipelines, it is used in conjunction with SSD MobilenetV3 to produce a more accurate bounding box. After researching different network models to use for image processing, I decided that Mask RCNN and SSD MobilenetV3 would be the best fit to solve the problem that this report is working toward solving. In combination with these image processing models, research was also conducted into salient maps to determine if implementing some generic saliency methods would be beneficial in highlighting particular regions of interest; there was some degree of accuracy after initial testing, although it was not something that was prioritised as the goal of the project was to implement AI in order to reach a solution.

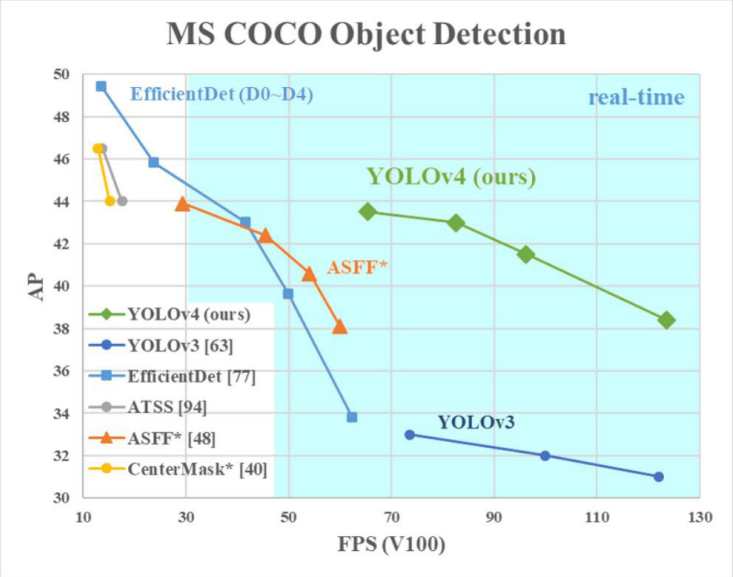
4.2. COCO

The COCO models were trained on over 300,000 images containing 2.5 million labelled instances and involved extensive crowd worker involvement [15] to collect, label, and segment these images for training the model. The COCO dataset has become the baseline standard for benchmarking models and real-time object detection performance analysis.

4.2.1. COCO dataset

COCO is one of the largest datasets and is commonly used within computer vision to identify segmentation regions and classification of objects within images. This dataset is commonly used to train models that are used to identify ROI's in an image. COCO provides over 91 categories for classification within over 300,000 images, hence why it is so commonly used in computer vision tasks. See **Table 1** from paper [4] for an example implementation of COCO being used as a benchmark to test the effectiveness of different models. Within the table notice the widely used model YOLOv3 and YOLOv4 [4] being compared against each other to evaluate how the newer model performs against the previous version.

Table 1 from paper [4]



4.2.2. COCO SSD network MobileNetV3

Within the methods produced in this report there are two different AI models that are used to help process images and find where the region of interest should be. The first model used, is the model that is trained on the large COCO dataset called “ssd_mobilenet_v3_large_coco_2020_01_14”, this model allows for regions to be identified using bounding boxes as well as providing classification. Classification is not necessary for this particular use case as we are only interested in identifying the region. Implementing this model into the pipeline as the first processing step was simple thanks to OpenCV’s implementation of importing DNN’s through the “dnn_DetectionModel” function which simply allows you to parse the location of the model and config path. Once the network is defined and config parameters have been set all that is left to do is to use the “detect” function while parsing an image and a confidence threshold for detecting objects. After using the “detect” function it returns classification ids with corresponding bounding boxes showing the region in which the object has been detected, using this allowed for determining where the focus of the image should be due to the bounding boxes being used to identify the ROI.

4.2.3. COCO Mask-RCNN

The second model used is the second version of Mask RCNN produced from the large COCO labelled dataset in 2018 which is capable of producing a bounding area where the object lies within an image as well as a classification, again the classification is not necessary for this use case. The bounding area produced from this model is limited to the maximum and minimum areas of the output mask, which is why for V5 of the pipeline this model has been used in conjunction with the SSD MobileNetV3. Implementing this model into the pipeline is fairly simple thanks to OpenCV again as its possible to convert into a usable network using the “readNetFromTensorflow” function which parses the model and config path. After it reads the necessary data it is possible to use the network to process images using the function “forward” which returns an output mask along with a classification id. Mask-RCNN is a powerful model and is very capable at identifying objects within the input image.

4.3. Algorithms

In this section I will be going over some of the common algorithms that were used within the program to explain how they work and why they were used.

4.3.1. Intersection over Union

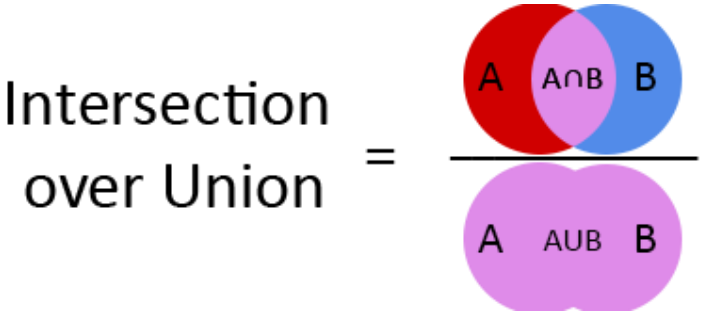
In order to evaluate the images IoU was the best way to do this as it returns a number between 0 and 1, this can be represented as a percentage, 1 being 100% and 0 being 0%. Typically, IoU is used to evaluate the performance of AI models such as CNN, RCNN, Mask RCNN and many others which is why it was chosen due to its wide use to evaluate the performance of these models.

Intersection over union is almost self-explanatory as for two images we are going to take the pixel count where the images intersect (are the same value for any x, y pixel) and divide that intersection by the total number of pixels, thus giving us a percentage value of how similar the images are. See **Figure 8** for an example of two binary masks that get compared, see (a) for the ground truth image and the (b) for the binary mask produced by V5 of the pipeline. These two images had a 90% intersection over union score. See **Figure 9** for a visual explanation of the IoU algorithm, A and B representing two different images/binary masks.

Figure 8: (a) ground truth binary truth (b) binary mask produced by V5

(a) (b)

Figure 9: Example intersection over union diagram



4.3.2 Saliency

Saliency is used to show how different an object is to it's surroundings and helps define where the focus is within an image. The salient space is typically represented using . Saliency allows for appropriate and efficient image processing to identify regions of interest within an image, similar to how this report looks at these problems it also tries to solve the same problem but with a different approach. Instead of training a model to identify features within an image, saliency takes a different approach which can be defined as a more mathematical due to how it approaches processing the image.

As mentioned previously in paper [7] which is about using global contrast to calculate where the salient region should be this is a more statistical/ mathematical approach to solving the problem. While this is a reliable way to solve the problem as mentioned this is only one way that salient space is used to segment the image into different regions. The saliency that was implemented in two of the pipelines is from the paper [17] which is an implementation of saliency that attempts to highlight people within images, this was used as it allowed for a simple use of saliency within the pipeline to build and improve the pipeline. This implementation of saliency focuses on highlighting humans in different poses, so does not work too well as a generic object detection however as it was simple to implement it meant there could be a baseline for the first version of the pipeline that could be built upon. Salient regions are usually defined as regions that could present the main meaningful or semantic contents from within an image. Most salient methods look at regions that have abrupt changes or unpredictable characteristics, which are then used in order to help identify areas within the image. Salient detection has many different approaches but it is common to find that they will be used in a way to try to interpret images as humans may recognize features within an image using colour and texture to differentiate between different regions within an input image.

4.3.3. Pixel Neighbours

The function is parsed a binary mask that has been pre-processed with an adaptive Canny edge detector, as the chosen area has detected an object/ROI the edge detector attempts to find the edges and this function will create an appropriate mask. In two of the methods produced in this report a function was produced in order to

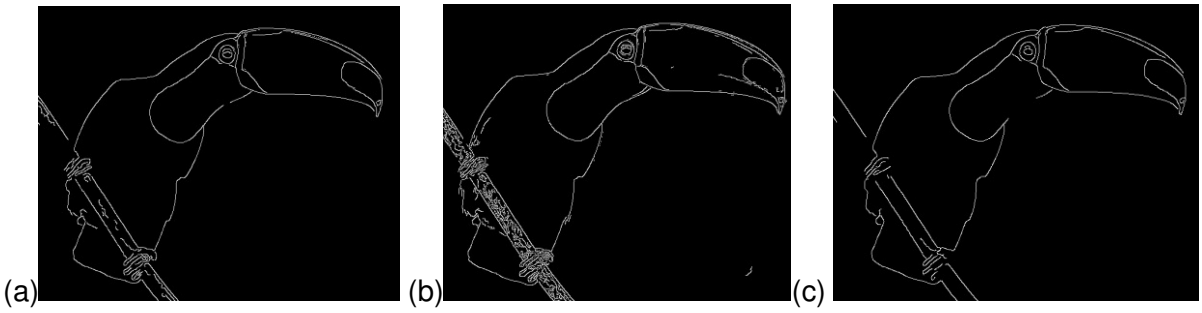
help determine if a black pixel should stay black. By reducing the search space using a model the amount of black pixels within the ROI is significantly reduced, this allows for one or several passes over the image to evaluate if a pixel meets the criteria to be white. The pixel is evaluated with a connectivity of 4 meaning that it looks up, down, left, and right of the selected pixel. The range of the search that is used is relative to the ROI size, with optimal parameters the range of the search is usually 1/10th of the ROI. If a black pixel is surrounded on all four sides then it means that it will be changed to white in the binary mask, after a few passes it creates an effective binary for the object. The limitation of this method is that it requires the AI model to locate an ROI/object in the image to reduce the search area, without reducing the search area the Canny edge detector would pick up too many edges and create too many lines which would essentially just create a white square.

4.3.4. Adaptive Edge Detection

As not all images are the same and may have had some different forms of pre-processing, therefore there needed to be some sort of measurement as to how many lines the canny edge detector has managed to retrieve. Instead of actually measuring the number of lines detected it is much easier to look at the ratio of white compared to black, due to the lines being extremely thin the method aims to get as close as possible to a 1/10 ratio of white to black as the area that is used is the region of interest.

In order to affect how many lines are detected using the Canny edge detector you can apply some blurring before the using the edge detector. Using various Gaussian kernel sizes, the amount of blur that is applied to an image can be adjusted to suit the requirements. When deciding which kernel size to use the image is processed using a 7x7 kernel, the ratio of white to black is then measured and different sizes of the kernel is then attempted to try to keep it within the bounds around 8% to 12%. See **Figure 10** for examples of different kernel sizes effecting the Canny edges.

Figure 10: (a) 7x7 kernel (b) 3x3 kernel (c) 11x11 kernel



4.4. Dataset

In the initial stages of the project, images were collected based on whether they seemed suitable for spot colour. The criteria set out for the images include; full RGB image, less than 2.5M pixel count and an obvious region of interest that could be identified by anyone. These images were used to help fine tune the different versions of the pipeline, after adjusting parameters within the different versions of the pipeline the IoU evaluation would be used to measure how much the accuracy has been improved from the different versions. The total size of the dataset used is 20 images, this is due to the five different manual identifications on the potential ROI's for an image. Each image has five versions of the ground truth which has been identified by five different people as their interpretation of the ROI for an image, so in total one hundred ground truths have been created for the total 20 images. Producing these different ground truths was a exceedingly time-consuming process as it required cooperation from the five people and the manual image processing within photoshop to create the desired binary mask that suited what was set out by the participants . As the pipelines have been fine tuned for this small dataset of ten images in order to test it and get some more real results ten images were randomly selected from the 2017 COCO unlabelled data set. The images were chosen by selecting 20 random images from the dataset and selecting ten of the images that were deemed to have an obvious ROI. As the different pipelines had not been fine-tuned for this set of images the results from the evaluation were a better representation of what someone might get from using this method on any image. Doing this provided some insight as to how the accuracy of the methods can vary quite a lot just by using a different dataset as the results did change a lot from the training set to the testing set.

Due to the datasets being so small it is hard to truly gauge how accurate the methods are, to solve this problem further testing has been done to try to validate the results of each method. To look at the variation of the results that can be generated a cross-validation test has been implemented for each method. Just with the small amount of data provided by the testing the difference is obvious so in order to collect more information from this small amount of data, implementing k-fold cross validation allows us to measure the overall accuracy of all the methods. K-fold cross validation will be discussed further in the report.

5. Design

In this section of the report, it will discuss the initial design that was set out in the planning process and how that plan changed once development began and research into how it can be accomplished. As well as the design of the project and how each section was going to be tested it will also cover the design methodology that has been used throughout this project.

5.1. Overview Design Change

The original design of this project was going to have two different versions of the pipeline. The first version was supposed to replicate a more traditional and mathematical approach to the problem using the saliency space to obtain the spot colour region. During the initial plan it was not obvious as to which implementation of saliency would be used in the first version of the pipeline. The second version of the pipeline was supposed to be the implementation of a model which was going to be trained using the pyTorch framework, at this point in the project it was unclear as to how implementing the model would be accomplished. pyTorch is an open source machine learning framework that allows for an easier way to implement machine learning through its library of useful functions. Later after researching into COCO allowed for the use of the COCO dataset to implement some form of machine learning into the pipeline although it was not in the original plan. The approach to only have two versions of the pipeline changed and instead, a more flexible approach was taken where multiple versions of the pipeline would be produced in the attempt to build upon the previous version and improve its overall accuracy through the initial testing that was laid out. Initial testing consisted of two different sets of images split into a training set and a testing/validation set, this split was

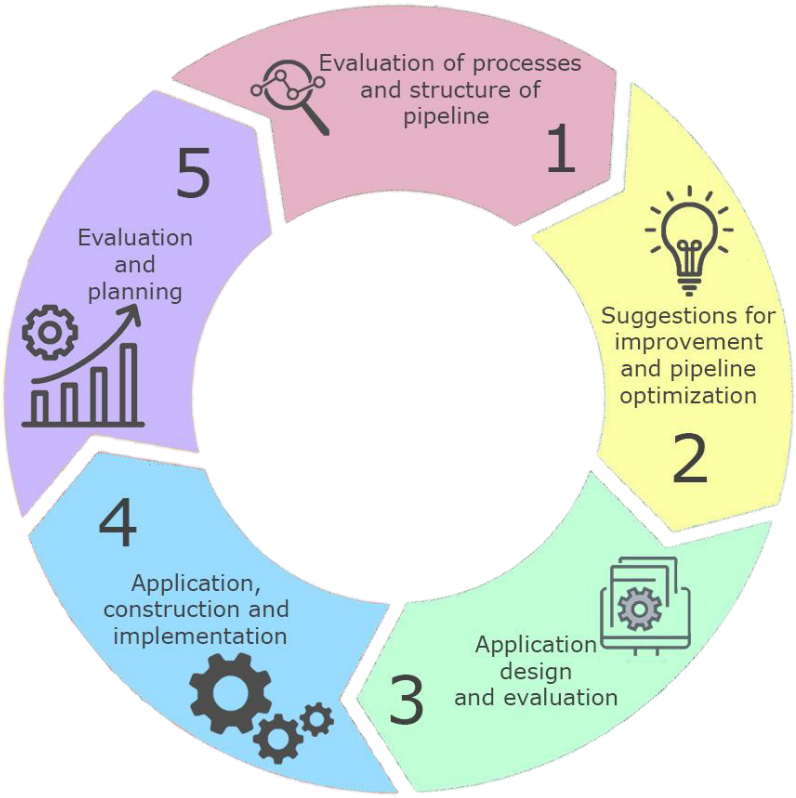
simply just 10 images for training and 10 for testing. The training set was used to fine tune each version of the pipeline while the testing set used to validate the optimal parameters that were used during the training set, while this does provide some insight into how effective each pipeline is the amount of data provided by the total 20 images makes it difficult to make definitive conclusions. Due to the small amount of data available additional testing was added to the project, this includes 50-fold cross validation and ANOVA testing which are discussed in more detail later in the report. As the initial plan has changed quite substantially the deliverables have also changed, when submitting this project there will now be five different versions of the pipeline instead of the initial two as well as this report. The focus of this project has remained the same and is intended to find the region best suited for the spot colour algorithm.

To identify the correct region to use the spot colour algorithm in we need to identify the region of interest as this is going to be the focal point of the image. To accomplish this when the image is processed a NN is used to help identify objects within the image which become the region of interest. The decision to use identifiable objects in the image as the ROI was decided upon due to the type of images that the AI models are designed to detect. Given any image the object detector is bound to identify key objects within the image which is why it was chosen to identify the ROI within the images.

5.2. Agile Design Methodology

A design methodology is an important aspect to a project as it sets up the structure and timeline for the project. All companies that produce software and web applications will follow a design methodology as it helps enforce good procedures and practices during development . Choosing the right methodology is quite important as this will affect the whole development process throughout the rest of the project, after analysing the requirements and structure of the project it was decided that an agile approach would work best. Agile design methodology is an iterative and evolutionary which is allows for constant improvements to each version of the pipeline throughout the development process. Managing the pipelines using agile development is the most optimal way to handle the project as it allows for constant improvement on each version of the pipeline, using the evaluation methods and then adjusting the pipeline to improve the evaluation score of the method. Many studies

have been conducted on design methodologies and found that using a methodology increases the rate of success for that project by around 20%.



5.3. Source Code Layout

jwilsonhall final fixes of files before submission 2ac3854 16 hours ago 55 commits

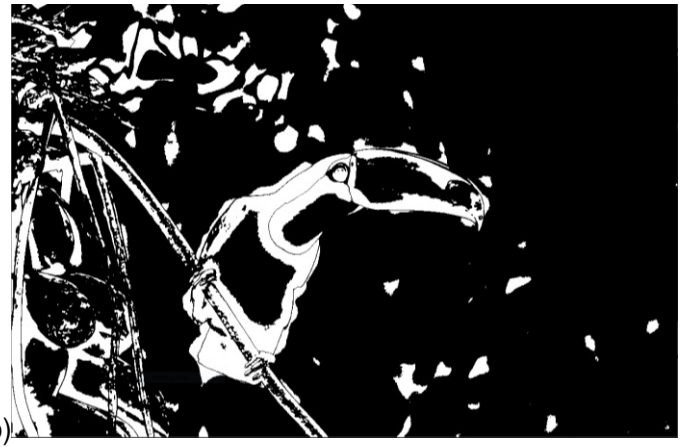
Ground_Truth	re-ran tests for v5 as there was an error in the code	last month
cross_validation	finsihed writing test for v4 pipeline	last month
dnn	added comments and ran training and test data	last month
hypothesis	final fixes of files before submission	16 hours ago
language_performance	final fixes of files before submission	16 hours ago
test_data	finsihed writing test for v4 pipeline	last month
test_output	re-ran tests for v5 as there was an error in the code	last month
train_data	removed unnecessary files	last month
train_output	re-ran tests for v5 as there was an error in the code	last month
.gitignore	changes to files	10 days ago
Cross_Validation.py	final fixes of files before submission	16 hours ago
IoU_Evaluation.py	final fixes of files before submission	16 hours ago
LICENSE	added LICENSE AND README	2 months ago
README.md	updated python setup for venv and requirements file	last month
example.py	created example file to produce necessary images for writeup	17 days ago
frozen_inference_graph.pb	removed unnecessary files	last month
log1.txt	Completed cross validation test for v1 of pipeline	last month
log2.txt	Completed cross validation test for v2 of pipeline	last month
log3.txt	Completed cross validation test for v3 of pipeline	last month
log4.txt	ran tests on v4 and finished additions to run v5	24 days ago
log5.txt	updated log5 for 50fold cross validation	9 days ago
log5combined.txt	corrections and added cross val for v5	10 days ago
log6.txt	corrections and added cross val for v5	10 days ago
python_setup.ps1	changed setup powershell script	24 days ago
requirements.txt	updated python setup for venv and requirements file	last month
ssd_mobilenet_v3_large_coco_2020_0...	removed unnecessary files	last month
v1.py	changes to files	10 days ago
v2.py	update Evaluation function paths	last month
v2Validation.py	final fixes of files before submission	16 hours ago
v3.py	final fixes of files before submission	16 hours ago
v3Validation.py	changed setup powershell script	24 days ago
v4.py	final fixes of files before submission	16 hours ago
v4Validation.py	corrections and added cross val for v5	10 days ago
v5.py	final fixes of files before submission	16 hours ago
v5Validation.py	corrections and added cross val for v5	10 days ago

5.4. V1

The first approach to finding the region of interest within an image consisted of lots of research into saliency and the different methods in which can be implemented to create masks for the ROI. Due to this being the first version of the pipeline it is more focussed on being a baseline in which can be improved upon throughout the progress of the project. The simplest way to implement a basic use of saliency was to use OpenCV in conjunction with python to produce a salient space in which can be used with Otsu's thresholding algorithm to produce a binary mask for that image. This does produce a region of interest however it is bound to how the saliency is calculated, the implemented saliency in V1 can be referred to static saliency which is optimized for detecting humans in images. Once the method is parsed an image it returns a two-dimensional matrix representing the pixel values in floating point, in order to turn it back into a normal image the floating point values are multiplied by 255. Multiplying the floating points in the matrix by 255 make it so that the pixel is represented by an integer value between 0-255. Each pixel is represented by an integer value, 0 meaning that the pixel is black and 255 means that the pixel is white, anywhere between these values is grey. Using OpenCV's implementation of Otsu's thresholding method the saliency map is used to create a binary mask, representing where regions of the original input image should be in colour as that is the identified spot colour regions. A copy of the original image is converted to greyscale, the binary mask that has been produced is then used as a guide and places the corresponding greyscale pixel into the RGB image where it identifies a black pixel. This will then give the processed spot colour image. See the results of this process at **Figure 11**.

Figure 11: (a) Saliency map (b) Otsu Threshold Image (c) Output Image

(a)

(b) 

(c)

While the dataset does not consist of many people it is a simple implementation that can be built upon and improved, initial testing with this method performed as expected from this method, giving it a maximum average of 36% with optimized parameters.

5.5. V2

The second method is similar to the first as it uses the same type of saliency however, this version of the pipeline is more focused on finding the best way to retrieve the region of interest. After research into what the best way is to find the region of interest the method that was chosen was to use an object detector in order to identify the region of interest within an image. Using an object detector to identify the ROI was chosen due to the fact that if an image does have a distinct ROI, it is likely that the ROI can be classified as an object such as a car, horse, person or bicycle. As the model is trained to generalize it is also possible to identify other objects even without classification, while this is technically a false positive, in this use case we do not care about the classification just the ability to identify the ROI from an image. Classifications that could cross over to other objects include things such as: Tiger being identified as a zebra or lizards identified as an elephant. The threshold for identifying a region of interest was set to 45% as this is a nice balance between accuracy and identifying areas within the input image. The model identifies specific regions of interest within an image while also allowing the model to locate regions of interest that are outside its classification due to the lower confidence threshold. Due to the lower threshold value images will often have multiple detections the bounding box that will be used is calculated by looking at the minimum and the maximum values of all the boxes, anything outside of each bounding box is discarded when the saliency algorithm is used which creates localized areas of each detection. A great example of each of these detections can be seen in **Figure 12**.

Figure 12: example thresholded salient map of multiple detections of cars.

See **Figure 13** for example image where two Komodo Dragons have been identified as an elephant with a 70% confidence value.

Figure 13: (a) Output Image (b) Otsu Threshold Image

(a) (b)

(a) shows example output of identified region of interest despite incorrect classification. (b) thresholded salient map to show bounding box areas.

Using this ROI that has been identified in the first step of this pipeline, the static saliency algorithm is used to highlight areas within the bounding box in order to mask the object that has been detected. While this is a more effective method that the previous it is aimed toward finding the ROI that we can use and minimize the amount of unimportant areas within an image. Finding these regions allows for the removal of areas in the mask that were highlighted that should not have been, comparing **Figure 11 b** and **Figure 14 b** is a great example of how finding the ROI in the image will reduce the amount of noise that is detected and thus improving the accuracy. In order to get more accuracy more research into creating accurate masks needed to be done as locating the ROI using the SSD MobileNetV3 was effective, the next step was coming up with a method to create a better mask within the bounding box that is identified.

After using the bounding box to determine which regions should have a mask the thresholded salient map is used as a guide to determine where the spot colour region should be.

See the results of this process at **Figure 14**.

Figure 14: (a)Saliency map (b) Otsu Threshold Image (c) Output Image

(a)

(b)

(c)

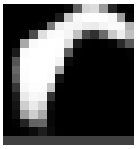
This method was a good improvement and provided many insights into methods that would be later used in other pipelines to improve the accuracy of other methods. During initial testing of this pipeline it was able to get an accuracy rating of 47% on the training set and on the testing set it got 41% which is still a good improvement over the previous method.

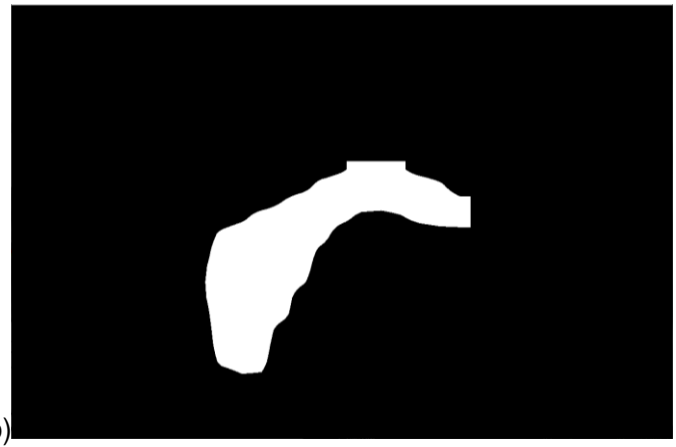
5.6. V3

The third version of the pipeline focussed on getting a mask that would be used to identify what regions would be used for spot colour. After further research into getting a mask from the region of interest within images Mask-RCNN seemed like a great option as it is very similar to the SSD MobileNetV3 network that was already implemented due to it being trained on the same dataset. This meant that similar results could be expected in terms of what sort of objects it should detect as it also meant it had the same classification list and threshold value for detecting objects within an image. To get a mask, this version implements Mask-RCNN which is an AI model that has been produced using the COCO dataset, this model is able to produce a bounding area, classification and a mask which represents where the object is detected within the bounding area. The first step of the pipeline was initializing the network using OpenCV, then parsing the network an input image which would be processed and return a mask and bounding boxes. Now that the network has returned the mask for the image, we have something that we can use for the spot colour region. Before we can use this mask we need to make some adjustments as the network will return a 15x15 image that needs to be scaled up so that it is usable as a mask, in order to do this we will resize the image back into the bounding box that it generated from the output of the network. When resizing an image, it is taken for granted with all the tools that are easily accessible today. When we need to resize a 15x15 image back into its original region we need to use interpolation in order to fill in the gaps by calculating what a value should be between a specified amount of known points. Bicubic interpolation will calculate the value of any given pixel by looking at 16 pixels which allows the out to be much smoother than if any other form of interpolation was used, it could be argued that bicubic interpolation is overkill for this task as bilinear interpolation with 4 pixels also does a good job but due to the scale and variety of scale at which the mask is upscaled the smoother output that bicubic is able to provide was preferred. After the resizing of the mask, thresholding is used to ensure that there are only black and white pixels and there have not been grey pixels as a result of the interpolation, this is an important step otherwise the later steps would break down as they have not been made to work with grey pixels. The new binary mask produced from this is then mapped to a black version of the input image as the mask resizing only applied to

the region of interest that it identified which is why it then needs to be added back to its original dimensions so that it can be used as a guide to determine where the region for spot colour is. See the results of this process at **Figure 15**.

Figure 15: (a) Output Mask before interpolation(scaled up x4.5) (b) Interpolated Binary-Mask (c) Output Image

(a) 

(b) 

(c)

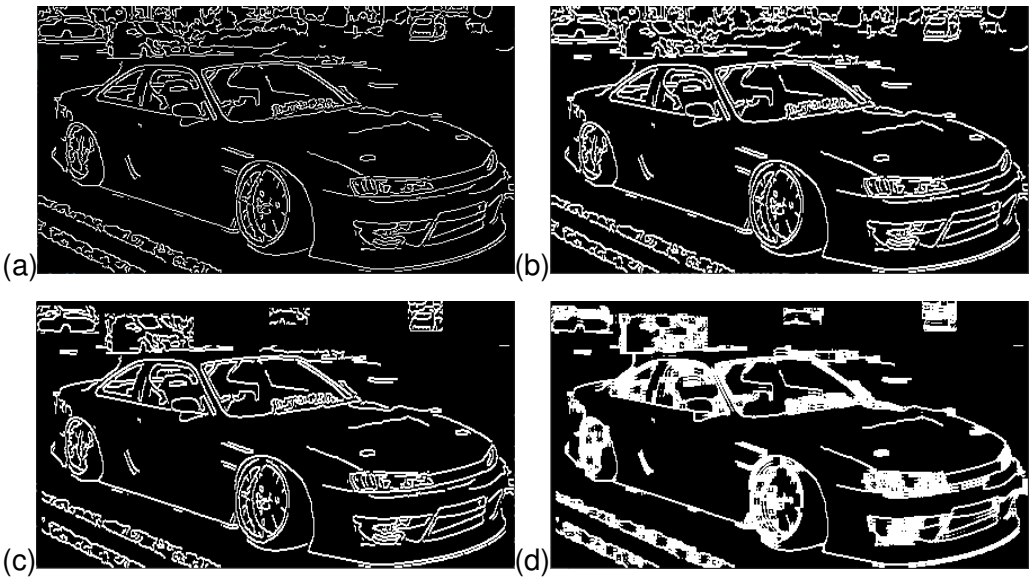
This method is a great improvement over the previous method and is able to identify the region of interest with the training and test image datasets. While the method was an improvement the results can definitely be improved which is why in the next method, the focussed moved to other methods for creating the binary mask after identifying the region of interest. The initial results from this method showed that it was able to produce an average max accuracy across the ground truths of 59% in

the training set and then 72% within the testing set. The scores of this method vary quite a lot between these two datasets which is why further testing is will be done to verify the results, this will be discussed later in the report.

5.7. V4

The fourth approach to finding the optimal region for spot colour was again focussed on coming up with a method that would produce a reliable mask as identifying the specific region of interest was taken care of by the SSD network. Approaching this problem to produce a mask took many different attempts but using the Canny edge detector with a variable blur seemed like an optimal way to get a mask in terms of accuracy. In order to get a reasonable number of edges detected from an input ROI defining what an optimal number of lines is an important step. In order to determine how many lines were detected a simple approach was taken which consisted of looking at the ratio of white to black. After testing different amounts of gaussian blurring on the training dataset a ratio of around 1/10 produced enough edges to use within the next processing steps. In order to get as close to this ratio as possible the input ROI is blurred and the ratio is calculated and based on this result the original input image will be blurred less or more based on how close it is to the determined optimal ratio. Before finding the optimal number of lines within the image the ROI will be created by using the conjunction of the bounding boxes to create a smaller image representing the ROI. When the Canny edge detector is used over this ROI anything that is outside the bounding boxes that detected the objects within the ROI are removed as this is seen as unnecessary noise generated from the edge detector. A great example of this can be seen in **Figure 16**.

Figure 16: (a) Shows Region of Interest after adaptive canny edge detector (b) shows dilation after Canny edges (c) shows the removal white pixels outside the bounding boxes within the ROI (d) Shows function that fills in the small gaps between edges



After the removal of the extra edges that are detected within the ROI the image is then processed further in order to fill in the empty space. A worked range is calculated based on the maximum length of the ROI which is calculated by taking the max length and dividing it by 3, 3 was chosen as a parameter due to how optimal the results were during testing. The worker range is used to replace black pixels with white pixels when the black pixel is surrounded on four sides. This is repeated for every black pixel within the ROI. The previous step is repeated 3 times which fills in the majority of the shape creating the mask region. In this version of the pipeline, it is focussed on retrieving a single binary mask which is why the next step in the pipeline attempts to remove and floating islands that may have been created in the previous steps. To remove these islands that have been created the program takes advantage of OpenCV's find contours function to outline the single main polygon within the image, this outline is then stored and the white islands are removed before the stored polygon is put back into the image leaving a single object mask as output for the pipeline. See **Figure 17** for output of image that has multiple detected objects within the region of interest. See **Figure 18** for example output for comparison against previous pipelines.

Figure 17: (a) Result of looking for pixels that are surrounded on 4 sides (b) Result of removing islands (c) Final output using binary mask

(a)

(b)

(c)

Figure 18: (a)Produced binary mask (b) Output image example

(a) (b)

This method was a dramatic improvement over the previous versions and with initial testing got much higher accuracy scores, the results of this could vary a bit and there was clearly still some room for improvement. The initial testing provided insight into how the score can vary quite a lot when it has been fine tuned for a set of images, initial testing scores showed that the method was able to produce output images with 80% accuracy for identifying the spot colour region in the training set. The testing data set got an accuracy score of 63% which was a lot lower than was expected compared to the results of the tuning.

5.8. V5

Version five of the pipeline builds on what has already been produced in version four and three, by attempting to merge these methods the end goal is to improve the reliability of the method as parameters that could be tuned in the previous version meant that there was a substantial amount of variation between the training and testing data set. Mask-RCNN has been implemented into this pipeline with the intention of being used to validate which pixels should be in white in order to stop areas that have been picked up by the canny edge detector which should increase the overall accuracy and reliability of the method. The SSD network was used to identify the region of interest as the RCNN network would often cut off objects within the ROI. Two images are created from these networks, one which is the same as V4 of the pipeline which is the ROI with the canny edge detector and the second image is the mask which has been put into another image with the same dimensions as the detected region from the SSD network. Iterating over the image the image it will look for pixel neighbours similar to the previous version of the pipeline, but it will also take into account how many pixels of the mask it is able to reach as well, this allows the

method to verify that the selected pixel is within a given range of the detected object and thus creates a more accurate binary mask. Compared to V4, V5 allows for multiple detections within the image rather than just one single ROI. It does this by looking at the largest connected region within the image, then it removes any islands that are smaller than half its size as this is considered to be noise. See **Figure 19** for example output of the method.

Figure 19 : (a) binary mask (b) output image of method

(a) (b)

This method was an improvement over the previous version and with initial testing got a lower score however further testing proved that it was a much more reliable method. In the initial testing this method with achieved 77% accuracy which is lower than the previous versions however it scored higher on the testing dataset compared against the previous method with an accuracy of 75% which is a much more consistent accuracy score. In order to prove these method results we need to look into more testing that will give more reliable data.

6. Testing

Throughout this project testing has been changed in order to accommodate for the small size of the datasets that are being used in this project. Initial testing consisted of two datasets which were the training and testing datasets. The next set of testing that was used to create more data from the smaller dataset, this is called K-fold Cross-Validation although in this use case it is 50-fold Cross-Validation. The final set of testing is called ANOVA, meaning the analysis of variance. These testing methods will be discussed in this section.

6.1. Image Evaluation

In order to evaluate an image you need to have a set of data that is going to represent the ground truth when attempting to quantify how accurate the output image is. In this report we have already discussed Intersection over Union however this is where it has been implemented. As previously stated, intersection over union takes 2 binary masks as input and uses the IoU method to return a percentage value which represents how similar this masks are. Due to spot colour being open to interpretation in this report we have five ground truths per image which have been created by five different people identifying the region of interest within the dataset. When evaluating an output image we will take the binary mask produced and compare it to the five ground truths for that image, taking the maximum value returned by the IoU function as this represents the region that was identified.

6.2. Initial Testing

Initial testing for this report does not have a sufficient amount of data to be able to draw conclusions from which is why the testing stage of the project has changed dramatically in order to account for the shortcomings of the dataset. However the initial testing did provide some insight into the project. Initial testing had ten images used for training and optimizing the different versions of the pipeline, this allowed for improved results and increased the accuracy of each method. As each method had been trained on this set it would not be a fair evaluation to draw conclusions from, which is why there was another dataset for testing each method results from both datasets can be seen in **Table 3** and **Table 5**. Looking at the difference in these tables it is clear that for some of the methods there is quite a large variation in results which is why a more robust method of testing has been implemented called K-fold Cross-Validation.

6.3. K-fold Cross-Validation

Cross validation is a resampling procedure which is used in order to help evaluate the accuracy of machine learning models when there is a limited dataset. The “k-fold” aspect to this testing means the number of iterations that it will repeat the Cross-Validation process, in this instance “50-fold Cross-Validation” has been implemented. The 50-fold Cross-Validation process follows this procedure:

1. Randomizing the 20 images into a 70/30 split (14/6)
2. Generate parameters for pipeline
3. Test every set of parameters on training set
4. Evaluate each set of parameters
5. Select highest performing parameters
6. Use parameters on the test dataset
7. Evaluate performance on test dataset
8. Repeat from step one k times

Cross-Validation is primarily used in machine learning as a measure of accuracy of the model when used on unseen data. This validation is an important step to measure the accuracy of a method. It will allow for more data from smaller datasets such as this one used in this report. Due to the small dataset a higher k value was opted for in order to make sure that the results that we get can help draw conclusions about the performance of each of the method.

6.4. ANOVA

ANOVA testing stands for the analysis of variance, this is a statistical method that aims to find out if results from an experiment are significant. It works by analysing the change in variance between each group of data. Using ANOVA we look at the spread of data away from the mean within the data groups, if there is a lot of variance within groups then there is more chance that randomly selecting a sample from the group will be different to the mean due to probability. ANOVA will also allow us to consider the sample size, the larger the dataset that is being used the less likely that randomly selecting a sample will be an outlier in the data. ANOVA Is calculated by comparing two types of variance from the data, we will compare the variance within each data group and also the variance between different samples. This testing provides several different figures which are then used to understand if

the data has a correlation between groups and if the data is statistically significant. By comparing the F statistic to the F crit value we can evaluate how statistically significant the test is. Looking at the how statistically significant the test is we can get a better evaluation and understanding how each method will perform on different data.

7. Conclusions

7.1. Results from Initial Testing

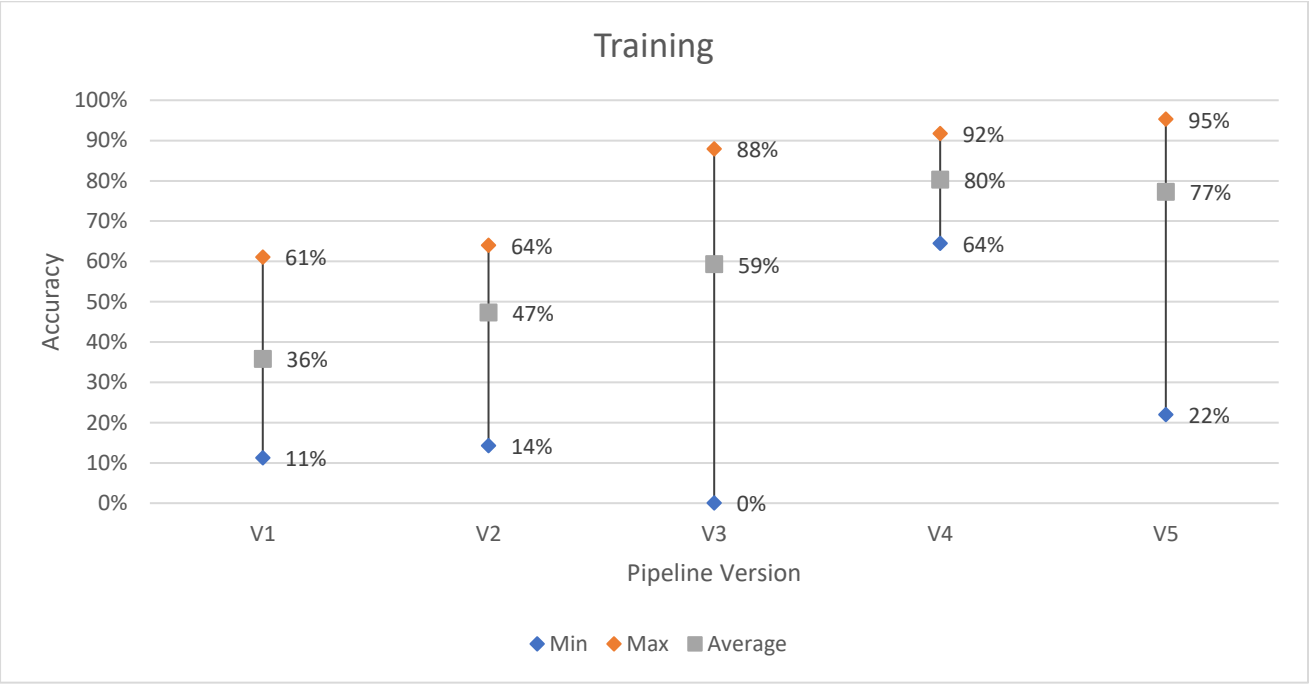


Table 3

	v1	v2	v3	v4	v5
Image 1	52%	58%	0%	84%	81%
Image 2	34%	55%	86%	84%	86%
Image 3	11%	14%	0%	64%	47%
Image 4	28%	41%	87%	79%	89%
Image 5	28%	60%	86%	82%	92%
Image 6	35%	36%	79%	80%	88%
Image 7	61%	64%	16%	69%	22%
Image 8	26%	49%	69%	92%	90%
Image 9	46%	57%	83%	79%	82%
Image 10	37%	39%	88%	90%	95%
Average of accuracy	36%	47%	59%	80%	77%

Table 4

	V1	V2	V3	V4	V5
Min	11%	14%	0%	64%	22%
Max	61%	64%	88%	92%	95%
Average	36%	47%	59%	80%	77%

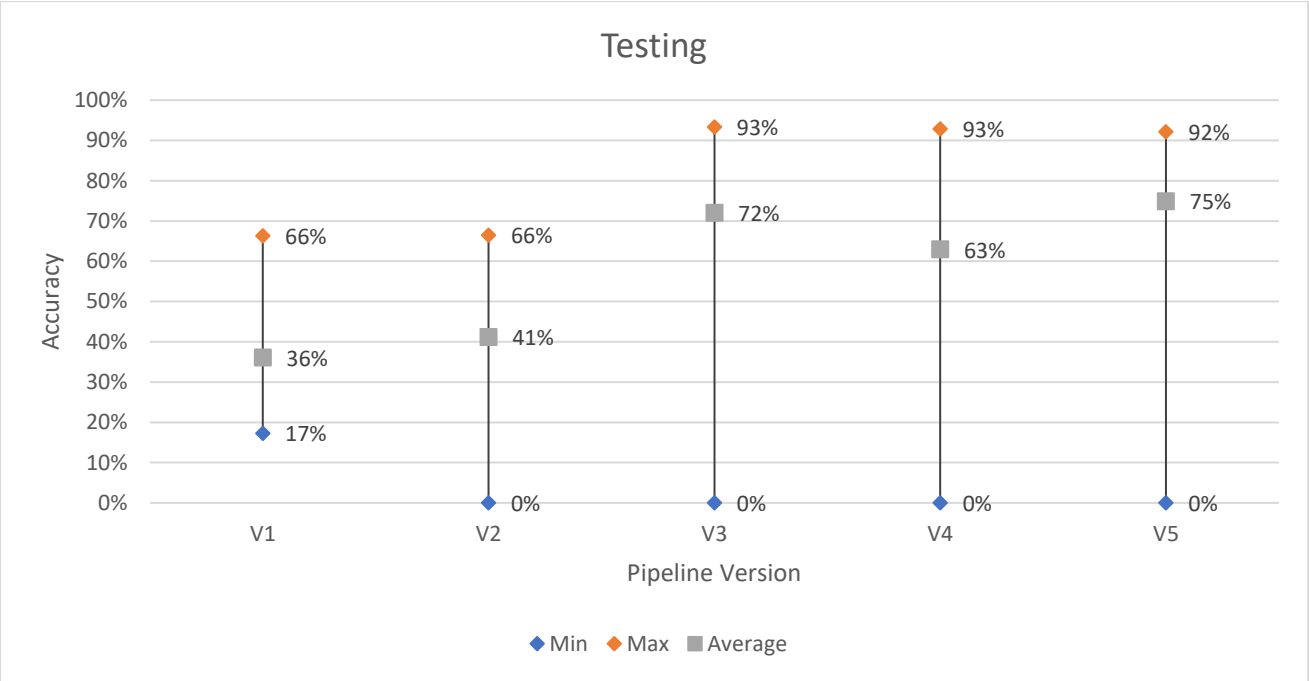


Table 5

Column1	v1	v2	v3	v4	v5
Image 1	46%	36%	86%	65%	84%
Image 2	66%	66%	76%	84%	84%
Image 3	28%	46%	85%	85%	90%
Image 4	41%	46%	76%	59%	76%
Image 5	17%	31%	89%	23%	83%
Image 6	31%	39%	73%	85%	83%
Image 7	34%	50%	62%	78%	72%
Image 8	19%	0%	0%	0%	0%
Image 9	36%	44%	93%	93%	92%
Image 10	44%	54%	81%	58%	84%
Average of accuracy	36%	41%	72%	63%	75%

Table 6

	V1	V2	V3	V4	V5
Min	17%	0%	0%	0%	0%
Max	66%	66%	93%	93%	92%
Average	36%	41%	72%	63%	75%

Image 8 can be seen in **Figure 20**, while in this image it is clear that there is a region of interest that could be used for spot colour. The models used struggle to generalize this region an assign it to a classified object, which means that the area will not be used for spot colour. If a generic object like a tin can or oil drum was used it would be possible to identify this object in the image like it has done with previous images when it identifies objects using a different classification but this is clearly an outlier in the data which reduces the overall average of the methods. Testing the methods with a larger dataset or a different dataset would allow the method to be tested more fairly and would minimize the impact that this outlier will have on the overall results of each method.

Figure 20: Image 8 from test dataset

The removal of this outlier would increase the average accuracy of each method's performance as can be seen in **Table 7**.

Table 7

Column1	v1	v2	v3	v4	v5
Image 1	46%	36%	86%	65%	68%
Image 2	66%	66%	76%	84%	84%
Image 3	28%	46%	85%	85%	90%
Image 4	41%	46%	76%	59%	76%
Image 5	17%	31%	89%	23%	83%
Image 6	31%	39%	73%	85%	83%
Image 7	34%	50%	62%	78%	72%
Image 9	36%	44%	93%	93%	92%
Image 10	44%	54%	81%	58%	84%
Average of accuracy	38%	46%	80%	70%	81%

7.2. Results from Final Testing

In this section it will discuss the results from the final testing stages of the project, in this testing 50-Fold Cross-Validation was used in order to evaluate each version of the pipeline.

7.2.1. 50-Fold Cross-Validation

For each version of the pipeline different parameters were randomized in order to select the optimal parameters that were going to be used on the testing dataset for that iteration of Cross-Validation.

<u>Parameters randomized for V1:</u>

- Thresholding method
  - Otsu's thresholding algorithm
  - Triangle thresholding algorithm
- Thresholding Type
  - Binary
  - To Zero

<u>Parameters randomized for V2:</u>

- Confidence of SSD Network
  - 0.1, 0.25, 0.4, 0.55, 0.7, 0.85
- Thresholding method
  - Otsu's thresholding algorithm
  - Triangle thresholding algorithm
- Thresholding Type
  - Binary
  - To Zero

<u>Parameters randomized for V3:</u>

- Confidence of Mask-RCNN
  - 0.1, 0.25, 0.4, 0.55, 0.7, 0.85
- Interpolation Type
  - Bilinear
  - Bicubic
  - Area – Resampling using pixel area relation

- Lanczos – Interpolation over 8x8 neighborhood
- Exact – Bit exact bilinear interpolation

<u>Parameters randomized for V4:</u>

- Short Pixel Range
  - 55, 60, 65, 70, 75, 80, 85, 90, 95, 100
- Long Pixel Range
  - 3, 6, 9, 12, 15, 18, 21, 24, 27, 30
- Confidence of SSD Network
  - 0.1, 0.25, 0.4, 0.55, 0.7

<u>Parameters randomized for V5:</u>

- Confidence of SSD Network
  - 0.1, 0.25, 0.4, 0.55, 0.7
- Short Pixel Range
  - 33, 6, 9, 12, 15, 18, 21, 24, 27, 30

Figure 21

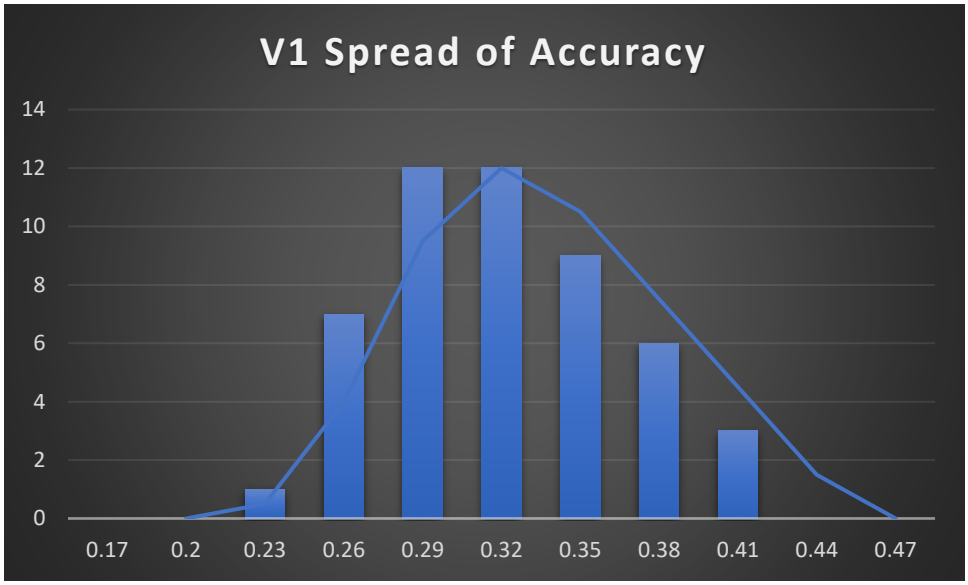


Figure 22

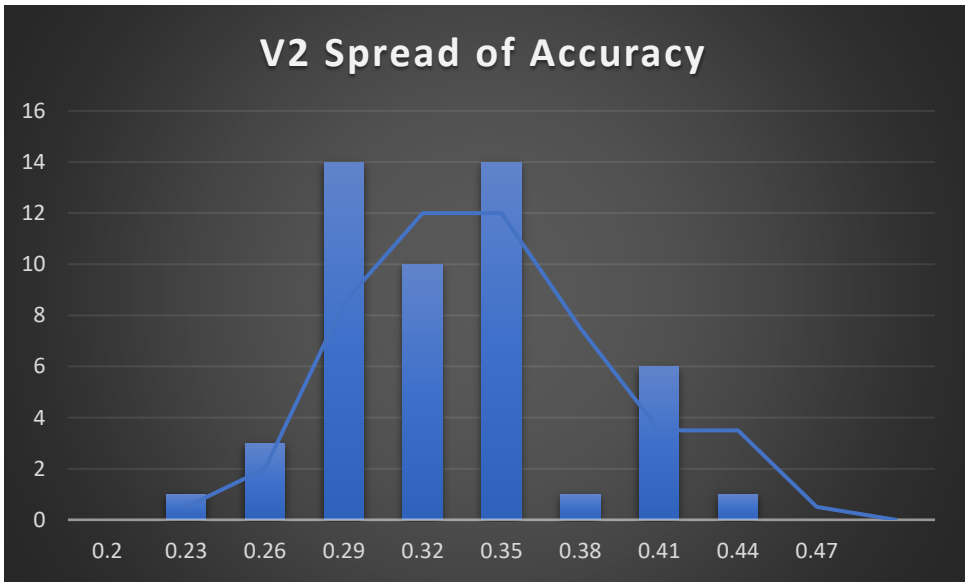


Figure 23

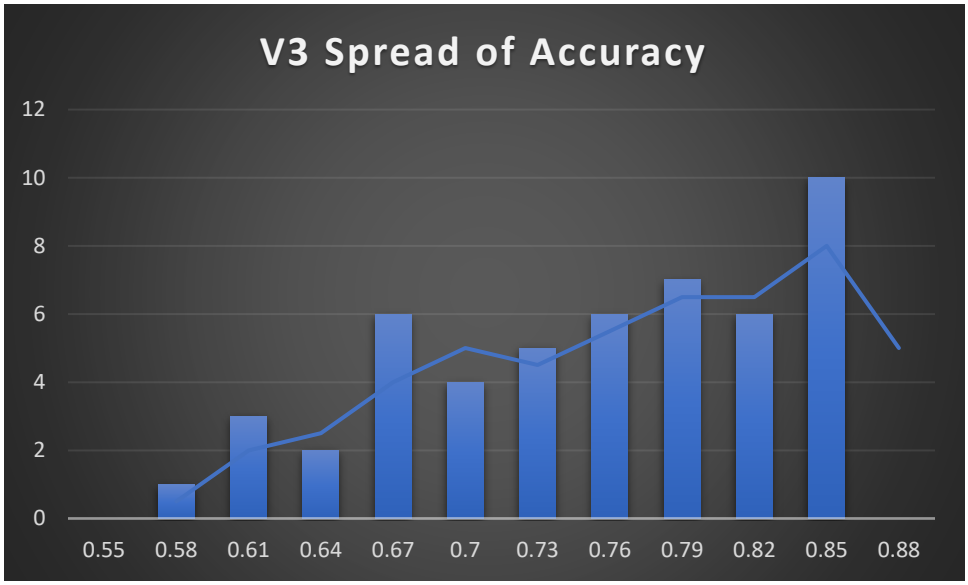


Figure 24

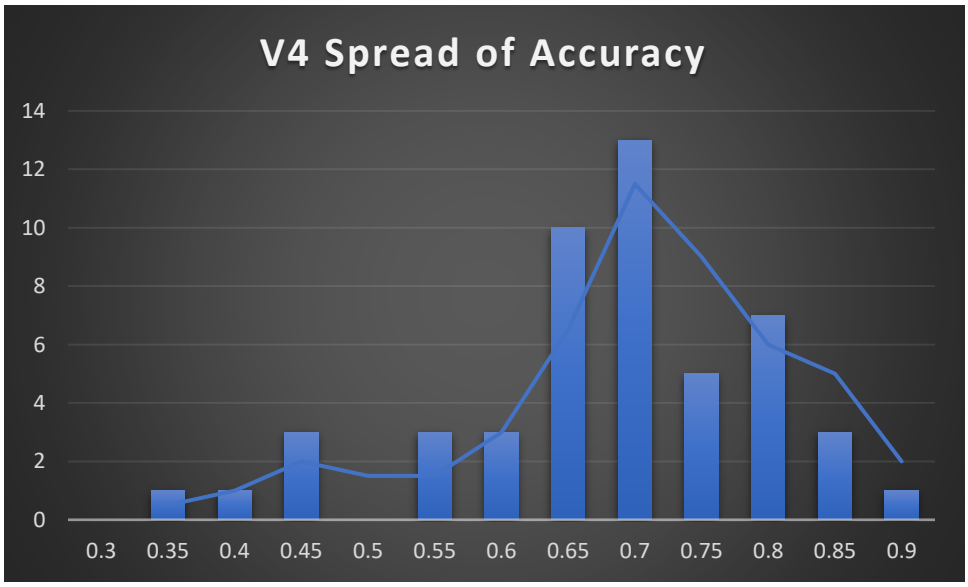


Figure 25

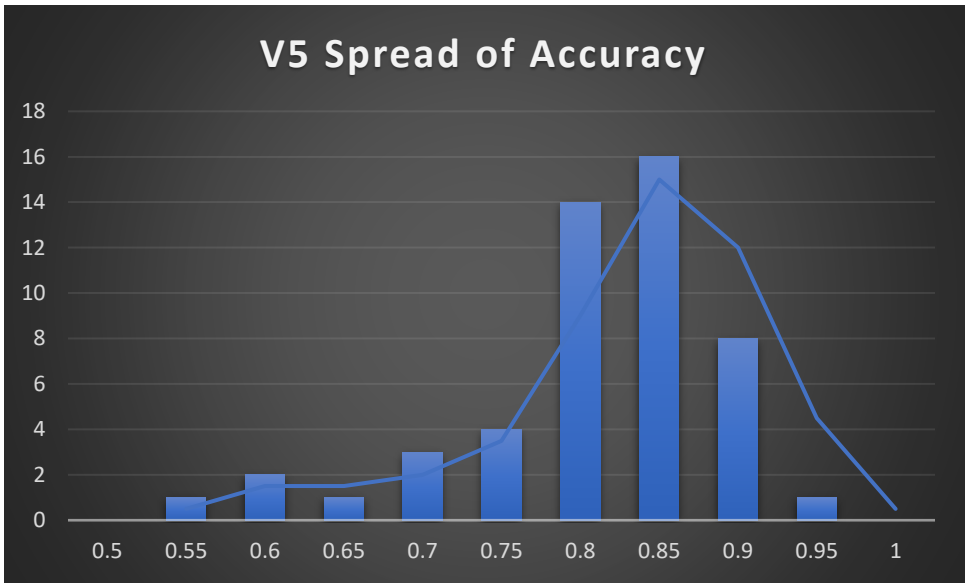


Figure 26

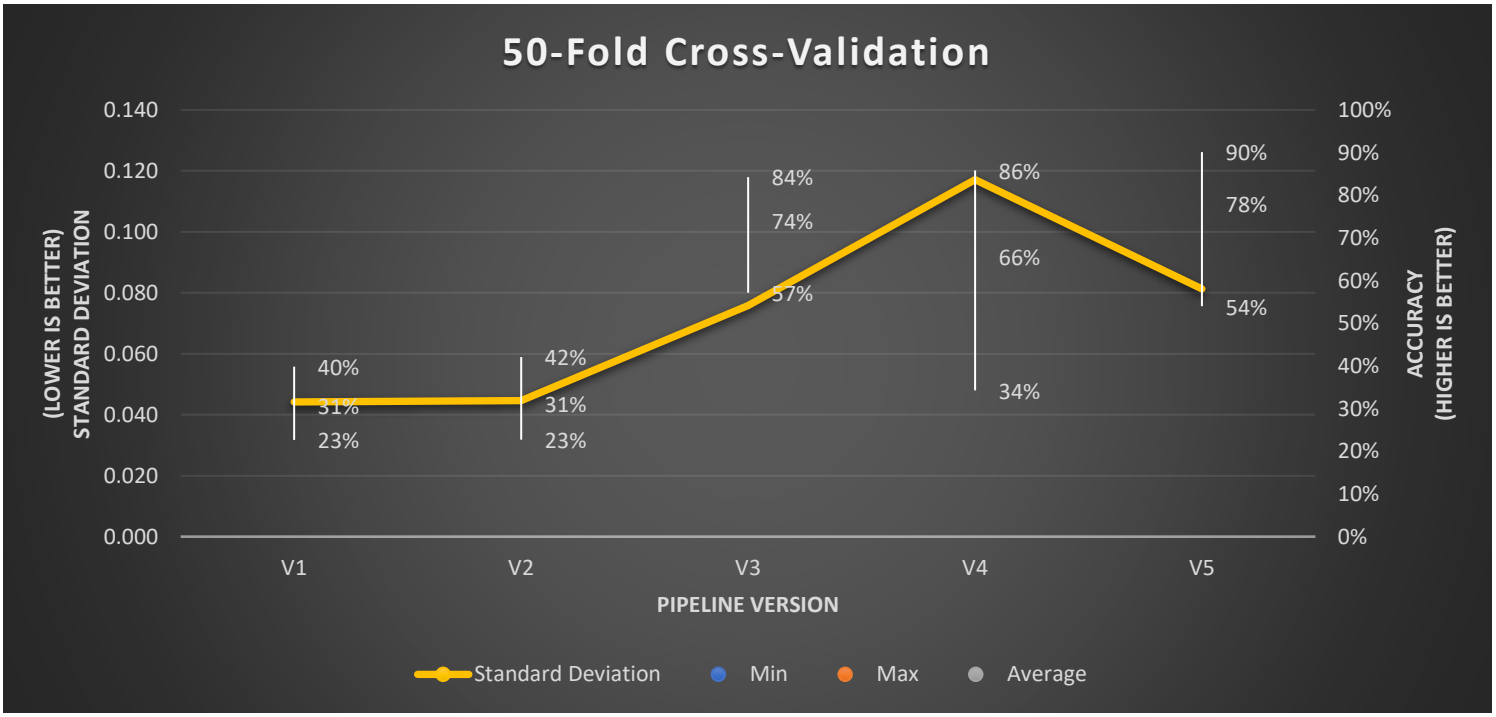


Table 8

Column1	V1	V2	V3	V4	V5
Min	23%	23%	57%	34%	54%
Max	40%	42%	84%	86%	90%
Mean	31%	31%	74%	66%	78%
Standard Deviation	0.0442	0.0446	0.0759	0.1171	0.0813

This graph shows the average accuracy of each version of the pipeline over 50 iterations of cross validation, this also shows standard deviation of each version of the pipeline through the 50-fold cross validation. To get the average of each version the pipeline will randomly select 14 images which will be used as the training set and process the set for maximum number of variations of parameters and select the parameters which performed the best, these parameters are then used on the remaining 6 images which are a part of the testing set. Using IoU to evaluate the performance of the parameters we will get an average accuracy score which represents the average accuracy that the parameters were able to achieve on the testing dataset.

As can be seen in **Figure 26** the last three iterations of the pipeline are relatively close in performance when comparing the average accuracy of each method however, this is not the only metric that should be considered when we want to try to evaluate the accuracy. When evaluating how effective a method is, it is important to consider how much variation there is within the data. Looking at V4 and V5 (**Table 8**) initially the data suggested that V4 and V5 will perform similarly in terms of the average accuracy however looking at the variation of the data (Standard Deviation) we can see that there is a significant difference between what the methods are capable of producing. Using 50-Fold Cross-Validation we are able to use the data it has produced to look how each method has performed on average and how much variance there is within the data. Any image processing method will always have some sort of variance in terms of what it is capable of producing on any given image. Standard deviation is able to measure how much the data varies in relation to the mean of the data, in this case the average performance of each method.

V1 and V2 are similar in terms of performance and average which is surprising when looking at the initial testing, which suggested that V2 was around 5-10% more accurate on average when compared to the previous version. This testing shows how similar these methods are within the pipeline.

Looking at **Figure 26** we can see that the best performing pipeline is V5 as this version has the highest average accuracy while also having the lowest standard deviation, having a lower standard deviation for a pipeline means that the majority of the time when an image is parsed to the pipeline there will be less variation in the results. V5 of the pipeline also has the highest max accuracy which means that on some images it is more likely to produce a higher accuracy than other versions of the pipeline.

7.2.2. ANOVA

Test between all groups of data

Table 9

Groups	Count	Sum	Average	Variance
v1	50	15.29191637	0.305838327	0.001953611
v2	50	15.62933394	0.312586679	0.001990099
v3	50	36.97699159	0.739539832	0.005758032
v4	50	32.8132796	0.656265592	0.013718702
v5	50	38.98429006	0.779685801	0.006606118

Table 10

Source of Variation	SS	df	MS	F	P-value	F crit
Between Groups	10.77837858	4	2.694594645	448.701821	1.9264E-111	5.187178737
Within Groups	1.471301558	245	0.006005312			
Total	12.24968014	249				

V1 : V5

Summary

Table 11

Groups	Count	Sum	Average	Variance
v5	50	38.98429006	0.7796858	0.006606118
v1	50	15.29191637	0.3058383	0.001953611

ANOVA

Table 12

Source of Variation	SS	df	MS	F	P-value	F crit
Between Groups	5.613286	1	5.6132857	1311.556835	1.53E-58	12.96584828
Within Groups	0.419427	98	0.0042799			
Total	6.032712	99				

V2 : V5

Summary

Table 13

Groups	Count	Sum	Average	Variance
v5	50	38.98429006	0.7796858	0.006606118
v2	50	15.62933394	0.3125867	0.001990099

ANOVA

Table 14

Source of Variation	SS	df	MS	F	P-value	F crit
Between Groups	5.45454	1	5.4545398	1269.055803	6.88E-58	12.96584828
Within Groups	0.421215	98	0.0042981			
Total	5.875754	99				

V3 : V5

Summary

Table 15

Groups	Count	Sum	Average	Variance
v5	50	38.98429006	0.7796858	0.006606118
v3	50	36.97699159	0.7395398	0.005758032

ANOVA

Table 16

Source of Variation	SS	df	MS	F	P-value	F crit
Between Groups	0.040292	1	0.0402925	6.517628996	0.012224	12.96584828
Within Groups	0.605843	98	0.0061821			
Total	0.646136	99				

V4 : V5

Summary

Table 17

Groups	Count	Sum	Average	Variance
v5	50	38.98429006	0.7796858	0.006606118
v4	50	32.8132796	0.6562656	0.013718702

ANOVA

Table 18

Source of Variation	SS	df	MS	F	P-value	F crit
Between Groups	0.380814	1	0.3808137	37.4727738	1.92E-08	12.96584828
Within Groups	0.995916	98	0.0101624			
Total	1.37673	99				

Using ANOVA testing the aim was to determine if the data collected from cross-validation was statistically significant, meaning is there a difference between the relationship between of each group of data and whether the results are likely due to chance/probability or to some factor of interest.

Looking at **Table 10** you can see the results from the initial ANOVA test which exhibits how statistically significant the test was. Looking at the F value compared to the F crit value it shows that the initial test was statistically significant and the results within the data aren't due to chance and there is a clear distinguishing factor of interest within the different versions of the pipeline. As the P-value from **Table 10** is below the threshold of 0.0005 it means that there is a difference between the means of the data and there isn't a correlation between the groups. The average performance of each pipeline can be seen in **Table 10** and it makes it clear which versions of the pipeline are most effective at identifying the correct regions of interest with a given image, another observation can be made with the sum total of all the scores. Looking at the sum scores of the pipeline it is clear that V5 was the best performer however V3 is not that far behind V5. V3 and V5 performed similarly to each other as V5 attempts to build on top of what V3 was able to produce by implementing features from V3 and V4 to make a more reliable estimation of the binary mask. V4 is also similar to V5 in terms of the method however it uses the Mask-RCNN algorithm to verify that the area it is turning into a binary mask is the correct region, the features from V4 that are implemented into V5 clearly show that it helps it outperform V3 however it only does so by a smaller margin which can be seen in the average and the sum columns within **Table 10**. The results from this table show that the data is statistically significant however it does not specify between which groups hence, why further ANOVA tests have been conducted to compare the performance of each pipeline against V5.

Looking at the results from the ANOVA test from V1 and V2 against V5 in **Table 11,12,13 & 14**, the results are very similar due to how similar the methods of V1 and V2. From these results the F value is much larger than the F crit value which demonstrates how significant this test is, showing that there is no correlation between this data and that the results from the methods are not a result of chance/probability but a factor of interest within V5. The P-value also demonstrates that there is no correlation between V1 and V2 when compared against V5.

Running the ANOVA test on V4 and V5 showed interesting results as V5 takes features of V4 and attempts to build and improve upon them. V5 uses the same adaptive edge detection algorithm to help verify which regions of the binary mask should be white. The results from the ANOVA test in **Table 18** indicate that the test was statistically significant between these two methods. The P-value indicates that there is no correlation between the means of the data collected from the 50-fold Cross-Validation. Before using the ANOVA test it is possible to draw some conclusions about the comparison between these two methods such as looking at the average performance or the total sum from **Table 17**.

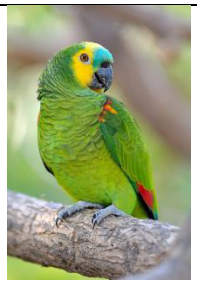
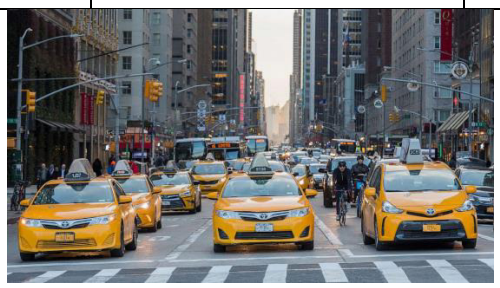
Comparing the data between V3 and V5 is a more interesting test as the results are especially similar, this is due to the fact that V5 builds on top of V3 implementation of Mask-RCNN. While the results from the ANOVA test show that the test was not statistically significant and using the P-value measurement the argument can be made that there is a correlation between the data, this is due to V5 only being marginally better than V3 in terms of the results its able to produce. In **Table 15** it is clear that the average and the sum of V3 and V5 are similar but on both V5 is slightly higher. In **Table 16** if the F value is compared to the F crit value we can determine that the ANOVA test between these two groups of data was not statistically significant. Looking at the P-value we can also see that it is more than the 0.0005 threshold, meaning that the data within the two groups are corelated to one another. This is not too surprising as V5 uses the same Mask-RCNN model to help find the region of interest, however V5 does outperform V3 and in order to prove this a final hypothesis has been created to distinguish the difference between these two pipelines.

7.2.3. Hypothesis

To determine which version of the pipeline is better and to also identify weaknesses within the methods a final test has been conducted. The hypothesis is that V5 will outperform V3 in selected images and V3 versions will also fail to identify some flowers as there is not a classification that is similar enough to identify flowers that will allow it to identify the correct regions. V5 incorporates more generic object detection using bounding boxes so it should be able to identify some regions of interest within a larger variety of images. 6 images were collected which have obvious regions of interest as can be seen in **Table 19**. V3 is better isolating multiple

objects within an image whereas V5 is likely be unable to separate between objects creating one large mask connecting all the identified objects together. The first 6 images selected are optimal images to be used on both of these methods as they have a simple identifiable spot colour region that can be identified. The hypothesis is that images with a singular object/region of interest are better suited to V5 method whereas V3 is better at distinguishing between objects within an image.

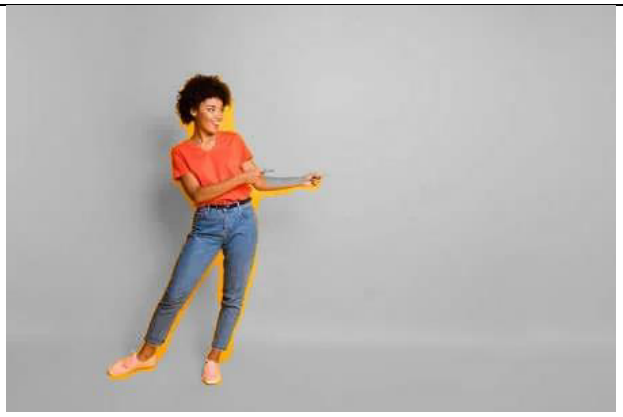
Table 19

V3 uses Mask-RCNN in order to create a binary mask of the ROI for each image, this is very effective however due to the output of the network being a 15x15 image it is limited by how accurate it can be when interpolated back to the correct size. In the output images of V3 you will notice that the line that represents the edge of the object is inside the object, V5 attempts to build on this by finding those edges and filling them in. See **Table 20** for output from V3 and V5 being compared. A single ground truth was created for each image to evaluate the performance empirically.

Table 20

No.	V3	V5
1		
2		

3		
4		

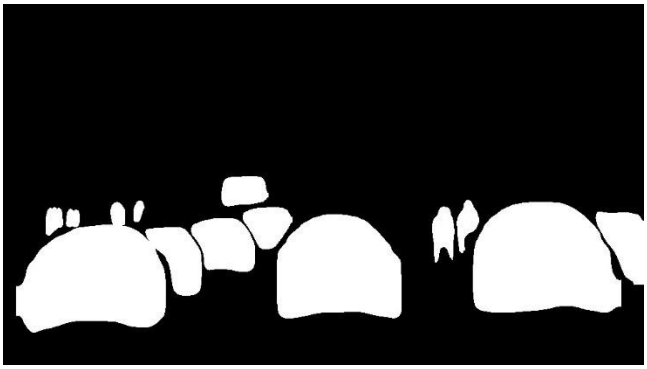
5		
6		

7		

Looking at image 7 in **Table 20** you can see that a lot of the details in image 7 have been put in colour in V5 whereas V3 has effectively isolated the multiple objects within the image. See **Figure 27** for a clear binary mask to compare V3 and V5. Looking at **Figure 27** it is clear that V5 struggled and essentially merged the individual masks together to create a large blob. Using **Table 20** we can make the observation that both the methods are helped when the particular region of interest is in more focus than the other parts of the image, this type of effect is helpful as when the edge detector is used it will identify more lines that are relevant to the ROI. This will happen because when there is a blur the edge detector will struggle to identify contours and highlight those edges, meaning that more lines are defined around the object in focus. From the ANOVA tests comparing V3 and V5 it identified that emperically there wasn't a statistically significance between the two methods and that the data was linked however looking at the results in the table it is clear that there is a variation in results between the methods.

Figure 27

(a)



(b)

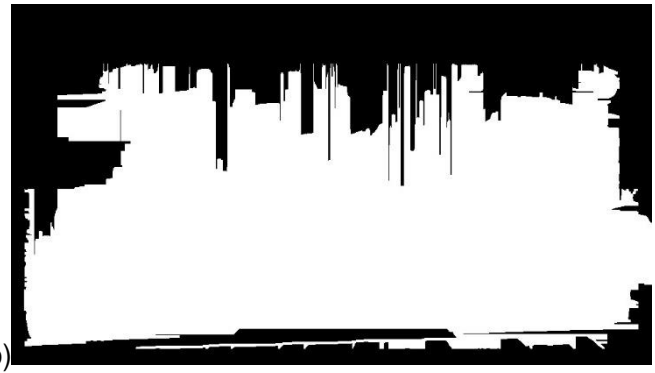


Figure 27: (a) binary mask produced by V3 on image 7 (b) binary mask produced by V5 on image 7

Table 21

Hypothesis Results	V3	v5
Image 1	83%	87%
Image 2	87%	95%
Image 3	86%	94%
Image 4	69%	81%
Image 5	88%	95%
Image 6	0%	77%
Image 7	77%	33%
Average	69%	88%

A single ground truth was created for each image in order to measure the accuracy from V3 and V5 however, **Table 21** does not prove anything as the type of images selected skewed in favour of V5 due to there only being one ROI within the majority of images. However, the hypothesis about how the pipelines perform was successful.

7.3. Conclusion

Throughout the development of this project there have been many aspects that have changed from the initial plan. Initially the project's end goal was to produce two versions of the pipeline one of which would be a non-AI driven method which would be compared against an AI method. These two methods were supposed to be able to identify a region of interest in which to apply the spot colour Algorithm. This aspect of the project changed into what I have delivered within this report. five versions of the pipeline have been produced and explained within this report. The five versions each make an attempt to improve on the previous methods and take those good aspects to produce more reliable and accurate binary masks. The initial plan was quite different to the final implementation as later research would lead to a better understanding of the project and what was the best way to implement this project. Some shortcomings of the project include the following, small dataset, reliability, computationally intensive and accuracy. These shortcomings are something that I would have liked to improve upon if I were to continue to develop the project more.

The project was able to produce five different versions of the pipeline, which gradually improve one after the other. Through Cross-Validation the accuracy of these pipelines has been evaluated which demonstrate what they are capable of. Due to the limited dataset that was used with Cross-Validation further testing has been conducted in order to verify the reliability of those results. The results show that the ANOVA testing was able to prove that V5 was statistically more reliable on V1, V2 and V4 however it was unable to prove that it was better than V3. On average V5 was 4% more accurate than V3, the ANOVA test between V5 and V3 also showed that the test was not statistically significant as the F value was smaller than the F crit value. Looking at the P-value of that test we can also see that it is above the P threshold of 0.0005 meaning that there is a correlation between the data, due to this and the fact that F crit value is more than the F value we can determine that this test is not statistically significant. In order to distinguish between these two methods further testing must be done in order to determine which of these methods is better at identifying the ROI.

The hypothesis section is supposed to prove that V5 is a better solution to identifying the spot colour region than V3. If we define identifying the spot colour region as a single area within the image. Then through this definition we have accomplished this.

Through the hypothesis test it was possible to confirm that V5 objectively performed better at identifying the ROI within images however, through observations we can see that image's with multiple sperate objects/ROI's V3 will outperform V5. Due to the how the mask is created for V3 it will consistently leave a gap between the inside and edge of the identified spot colour region (see **Figure 28**).

Figure 28

Despite the diversion away from the initial plan set out at the beginning of this project, I believe this work has been successful because I have been able to produce a method that is capable of identifying the correct spot colour region. As well as producing a spot colour method there has also been a lot of testing conducted to be able to verify the capabilities of the methods produced within this project. If this project were to be continued or repeated the approach would change and would focus on providing a more methodical way to test the pipelines to allow for more reliable results when finding the accuracy of the pipeline. In the section below it will elaborate on the ideas for expanding this project.

7.4. Future Work

Throughout the implementation of the project, different methods of obtaining the spot colour region have been considered. Some of the methods that were considered would have been effective and efficient however the goal of this project was to produce an AI driven solution to identifying the spot colour region, hence the produced methods.

Implementing the global contrast saliency method from [7] would have been a great implementation and it would have been interesting to see how the method performs when using the images within the dataset used in this project. This is the most interesting method that if there was more time within the project implementing this method would be a great solution to the problem set out in this report.

Other improvements could have been performed on some the methods produced within this report. There are many other optimisations that could have been performed to improve V5 of the pipeline, as a weakness was identified within V5's ability to handle multiple regions it should have been stream lined to use the most optimal region using the confidence threshold to identify which corresponding region should be used for spot colour. The lack of flexibility when it comes to if object detections is an issue for all versions of the pipeline, which is why an adaptive object detection method would be a great way to identify important regions. An adaptive object detection could be accomplished a few different ways, running the network multiple times until it is able to detect anything significant or what might be more optimal is setting the confidence threshold at around 1% which would allow for the selection of the most significant regions out of everything that was detected within the image. Another improvement that would potentially solve the multiple object detection of V5 is the ability to identify a suitable range for the pixel neighbour algorithm based on the number of object detections. Further testing and optimizations would be needed to be able to implement this effectively.

A major limitation of the pipelines is the fact that Python has poor performance when it comes to for loops and matrix operations, the most optimal way to use Python is to implement everything using libraries that are written in C like OpenCV and Numpy. It is not always possible to use these libraries which is why a more efficient programming language could have been used for this project, this could include Java, javascript or C++ as these languages are relatively faster than Python. While implementing a more efficient programming language, there is also further optimizations that are needed on the creation of the binary mask. See **Table 22** proof of programming language performance.

Table 22

Programming language	Time (s)
Python	12.0380
Java	3.753
JavaScript	1.321
C++	1.402

Table 22: Performance of language when calculating 5,761,455 prime numbers.

The major limitation of this project is the size of the dataset used to evaluate each method. Ideally the dataset would be a minimum of 1000 images with 1 or more ground truths available per image. This larger dataset would validate the result and it would mean that the further tests and experiments that were ran would not be necessary to be able to conclude the effectiveness of each method.

Finally depending on how these methods are used and implemented it is probable that they would benefit from some sort of user control/configuration if minor adjustments needed to be made to some of the parameters to create a more desirable binary mask. A simple GUI with variables that could be adjusted and seeing a live preview of what those adjustments do would be the best way to implement V5.

Continuing the development of this project would potentially allow for more accurate methods to be produced. Producing these methods allow for the automatic spot colour identification and processing, these algorithms could be utilised by artists and photographers to automatically find these regions of interest or spot colour regions. While this could be utilised for different reasons it also could be used in a way that allows for some user interaction to fine tune the mask finding algorithm for more accuracy and adaptability. Spot colour is the main motivation behind the algorithms developed in this project there is also the possibility for a wider scope of applications where creating a more accurate mask is very important. Improving the accuracy of the methods would possibly benefit many of areas within computer vision that require accurate masks. While these could benefit many different areas which weren’t the target of this project, more development is definitely required as there are many optimizations that would be necessary to make these methods more viable. Catering to these professions definitely need an easier way to be able to interact with the process which is why an intuitive GUI would be beneficial. This project could be beneficial for those mentioned areas which is why they should be considered with further development of this project.

References

[1]. Achanta, R., Hemami, S., Estrada, F. and Susstrunk, S. 2009. Frequency-tuned salient region detection. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition. IEEE. Available at: https://doi.org/10.1109/CVPR.2009.5206596?locatt=mode:legacy.

[2]. Albawi, S., Mohammed, T.A. and Al-Zawi, S. 2017. Understanding of a convolutional neural network. In: 2017 International Conference on Engineering and Technology (ICET)., pp. 1–6.

[3]. Ansari, M.A., Kurchaniya, D. and Dixit, M. 2017. A Comprehensive Analysis of Image Edge Detection Techniques. International Journal of Multimedia and Ubiquitous Engineering 12(11), pp. 1–12.

[4]. Bochkovskiy, A., Wang, C.-Y. and Liao, H.-Y.M. 2020. YOLOv4: Optimal Speed and Accuracy of Object Detection. arXiv [cs.CV]. Available at: http://arxiv.org/abs/2004.10934.

[5]. Bradski, G. 2000. The openCV library. *Dr. Dobb's Journal: Software Tools for the* Professional. Available at: https://elibrary.ru/item.asp?id=4934581.

[6]. Bradski, G. and Kaehler, A. 2008. Learning OpenCV: Computer vision with the OpenCV library. Available at: https://books.google.ca/books?hl=en&lr=&id=seAgiOfu2EIC&oi=fnd&pg=PR3&ots=hVI0chiCSa&sig=AkxfYB9sMzofuM2lPNObB21X9qM.

[7]. Cheng, M.-M., Mitra, N.J., Huang, X., Torr, P.H.S. and Hu, S.-M. 2015. Global Contrast Based Salient Region Detection. IEEE transactions on pattern analysis and machine intelligence 37(3), pp. 569–582.

[8]. Dang Ha The Hien 2017. A guide to receptive field arithmetic for Convolutional Neural Networks. Available at: https://blog.mlreview.com/a-guide-to-receptive-field-arithmetic-for-convolutional-neural-networks-e0f514068807

[9]. Girshick, R., Donahue, J., Darrell, T. and Malik, J. 2013. Rich feature hierarchies for accurate object detection and semantic segmentation. arXiv [cs.CV]. Available at:

http://arxiv.org/abs/1311.2524.

[10]. He, K., Gkioxari, G., Dollár, P. and Girshick, R. 2017. Mask R-CNN. arXiv [cs.CV]. Available at: http://arxiv.org/abs/1703.06870.

[11]. Howard, A. et al. 2019. Searching for MobileNetV3. arXiv [cs.CV]. Available at: http://arxiv.org/abs/1905.02244.

[12]. Howard, A.G. et al. 2017. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. arXiv [cs.CV]. Available at: http://arxiv.org/abs/1704.04861.

[13]. Ji, Q. et al. 2013. Fig. 3. A demonstration of ROI extraction from a saliency map. (a) The. Available at: https://www.researchgate.net/figure/A-demonstration-of-ROI-extraction-from-a-saliency-map-a-The-original-image-b-the_fig3_257344403

[14]. Jiang, H. and Nachum, O. 26--28 Aug 2020. Identifying and Correcting Label Bias in Machine Learning. In: Chiappa, S. and Calandra, R. eds. Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics. Proceedings of Machine Learning Research. PMLR, pp. 702–712.

[15]. Lin, T.-Y. et al. 2014. Microsoft COCO: Common Objects in Context. arXiv [cs.CV]. Available at: http://arxiv.org/abs/1405.0312.

[16]. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y. and Berg, A.C. 2015. SSD: Single Shot MultiBox Detector. arXiv [cs.CV]. Available at: http://arxiv.org/abs/1512.02325.

[17]. Montabone, S. and Soto, A. 2010. Human detection using a mobile platform and novel features derived from a visual saliency mechanism. Image and vision computing 28(3), pp. 391–402.

[18]. Ren, S., He, K., Girshick, R. and Sun, J. 2015. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. arXiv [cs.CV]. Available at: http://arxiv.org/abs/1506.01497.

[19]. Rosin, P.L. et al. 2020. NPRportrait 1.0: A Three-Level Benchmark for Non-Photorealistic Rendering of Portraits. arXiv [cs.CV]. Available at:

http://arxiv.org/abs/2009.00633.

[20]. Rosin, P.L. and Lai, Y.-K. 2013. Non-photorealistic rendering with spot colour. In: Proceedings of the Symposium on Computational Aesthetics. unknown, pp. 67–75.

[21]. Solanki, K. 2020. Object Detection using SSD MobileNet with TensorFlow. Available at: https://kamleshs.medium.com/object-detection-using-ssd-mobilenet-with-tensorflow-1ee6e45a378b

[22]. Zhang, Y. 2017. CS341 Final Report: Towards Real-time Detection and Camera Triggering. Available at: https://www.semanticscholar.org/paper/01b527641d1f23cddac932d9fe20b44ec39114a3