MULTIPLE MACHINE LEARNING MODELS ON HIGH RESOLUTION IMAGES

JOSHUA LONG

A thesis submitted in partial fulfillment of the requirements for

Research Project COMP9596

School of Computer Science and Engineering
University of New South Wales

Submission Date:	June 11, 2019
Supervisor:	Assoc Prof. Raymond Wong
Assessor:	Prof. Wei Wang

MULTIPLE MACHINE LEARNING MODELS ON HIGH RESOLUTION IMAGES

Joshua Long

Supervisor: Raymond Wong

June 11, 2019

Abstract

Recently, deep learning models have joined the forefront in image recognition tasks, however, their reliance on large training datasets and deep network architectures makes them computationally expensive. This issue is furthered due to the increasing presence of high resolution images in the field of computer vision. Therefore, the need for optimising the application of deep learning models is becoming more essential. The aim of this project is to provide an implementation for image recognition that optimises runtime without significant reduction in accuracy. To achieve this, two contrasting segmentation methods capable of separating a high resolution image into smaller regions of interest (ROI) are proposed. The first will use local entropy and global thresholding while the other will use Canny edge detection and morphological dilation. A set of three image datasets consisting of both synthetic and real imagery with both homogeneous and non-homogeneous backgrounds will also be prepared. Extensive testing will then be conducted to determine optimal parameters and compare the implementation with baseline performance.

List of Figures

List of Tables

1 Introduction

1.1 Motivation

Deep learning has gained considerable popularity recently due to its outstanding performance in image processing. However, one of the drawbacks of deep learning is the inherently high costs on memory and computational power. These costs escalate further when considering the application of deep learning for processing high resolution images.

This research will focus on optimising runtime in the application of deep learning models for processing high resolution images. There are many areas that benefit from high resolution image recognition including remote sensing, printed circuit board (PCB) inspections, driverless vehicles and security. Taking a closer look at remote sensing, advances in camera technology for aerial imagery has enabled the ground surface to be studied in far greater detail. While the tiny dimensions of PCB units make high resolution necessary for the task of detecting defects and missing components. As the general trend for enhanced picture quality extends the domain of computer vision, optimising the application of deep learning models will become increasingly essential.

1.2 Approach

In order to reduce runtime, one of the key objectives of this work will be to segment the image by identifying smaller regions of interest (ROI) and eliminating other regions that are not required to be processed. To achieve this goal, the high resolution image will first undergo a preprocessing stage to obtain a downsized grayscale image. This smaller processed image will then be segmented by two contrasting approaches which will define ROIs and return their full scale position and location within the original image. The first segmentation design combines local entropy and global thresholding, while the second uses Canny edge detection and dilation. The two methods are referred to in this paper as the LEGT and CEDD methods for simplicity.

Upon receiving a list of the ROI positions and locations from the segmentation stage, each ROI image is cropped from the original using the provided dimensions. These ROI images are then passed on to a deep learning classification model. The two classification models applied in this project include AlexNet and DeepLab using

MobileNet as the backbone. For the testing phase, three datasets will be prepared consisting of one synthetic dataset and two genuine datasets comprised of real world images.

Chapter 2 of this report describes the method and includes detailed information on the preprocessing, segmentation and classification stages. Chapter 3 documents the results from an extensive number of tests on the prepared datasets that demonstrate the performance of the implementation in both synthetic and real high resolution images. Parameter choice for optimised performance and other factors that affect scalability are also investigated in this section. Chapter 4 summarises the results, discusses their implications and suggests future directions that require more research.

1.3 Related Work

One of the more significant advances that brought deep learning to the forefront of image processing was the introduction of AlexNet designed by Alex Krizhevsky in 2012 [1]. AlexNet was a deep convolutional neural network (CNN) that was trained to classify the 1.2 million images in the ImageNet dataset [2] into 1000 different classes. The architecture consisted of five convolutional layers followed by three fully-connected layers and made use of an efficient GPU implementation to speed up the convolution operation during training. AlexNet achieved record breaking results that were considerably better than the previous state of the art and has since brought much attention to deep learning in the field of computer vision.

Recently, many CNNs have focused on increased accuracy such as VGGNet [3] and ResNet [4], however, some focus has been placed on reducing architecture size to minimise the high demands of CNNs on time and memory. The development of SqueezeNet in 2016 [5] aimed to verify the proposition that for a given level of accuracy, there are typically multiple CNN architectures that achieve that accuracy level. SqueezeNet provided proof of this statement as it managed to achieve AlexNet level accuracy with a model only a fraction of the size and with 50x fewer parameters.

While CNN models were becoming a powerful tool in image classification, DeepLab introduced in 2017 [6] focused on the task of labelling all pixels within the image with a corresponding class, a process known as semantic image segmentation. DeepLab presented a method for learning multiple scale contextual features through the use of atrous convolutions with varying dilation rates and atrous spatial pyramid pooling. The technique allowed DeepLab to effectively capture objects and image context at multiple scales without downscaling the image. However, DeepLab's requirement to perform convolutions on a large number of high resolution feature maps led to high computational demands. In addition, the coarse sub-sampling of features during the atrous convolutions meant there was a loss of important information.

The introduction of RefineNet [7] intended to address these issues by providing a multi-path refinement network that exploits information available from each layer during the sub-sampling process. The approach proved that high resolution prediction was possible through fusing various levels of detail from different convolution stages without maintaining large intermediate feature maps. Another runtime and memory optimised model called SegNet [8], used an encoder-decoder architecture. The design stored the max-pooling indices from the feature maps in the down-sampling phase and used them to up-sample during the decoding phase eliminating the need for more expensive decoder training parameters.

In contrast, this project method takes the approach of reducing the input size through the use of image processing techniques prior to deep learning. As mentioned previously, it does this through locating ROIs within the image and applies the classification model to these smaller regions only. Therefore, the method can use any classification model architecture and performs best when applied to images with homogeneous backgrounds.

1.4 Key Concepts

1.4.1 Entropy

Entropy is a concept in information theory used to measure the amount of information produced by a stochastic source of data. It can be viewed as a measure of disorder or uncertainty as many random low probability events lead to high entropy while a set of less chaotic high probability events is low in entropy. In the case of high entropy, the randomness of the events transfers more information than a comparatively consistent dataset where there is little deviation from the expected highly probable events. Given that P_i is the probability of an event occurring, entropy can be calculated as follows:

$$S = -\sum_i P_i log P_i \tag{1}$$

In the context of images, local entropy relates to the likelihood of an event in a given neighbourhood. In [9] a definition of local entropy is provided whereby a neighbourhood Ω_k is defined by window size $M_k \times N_k$. It then states that if L is the maximal grayscale and n_j is the number of pixels with grayscale j in the neighbourhood, the entropy of Ω_k can be written as follows:

$$E(\Omega_k) = -\sum_{j=0}^{L-1} P_j log P_j \tag{2}$$

where

$$P_j = \frac{n_j}{M_k \times N_k} \tag{3}$$

Local entropy is a useful technique in determining ROIs within an image as it highlights information rich areas such as edges and foreground objects that contain more distinct features than the background. Figure 1 provides an example of local entropy and it's effect of illuminating the strawberries and plate that hold more texture and features than the plain background.

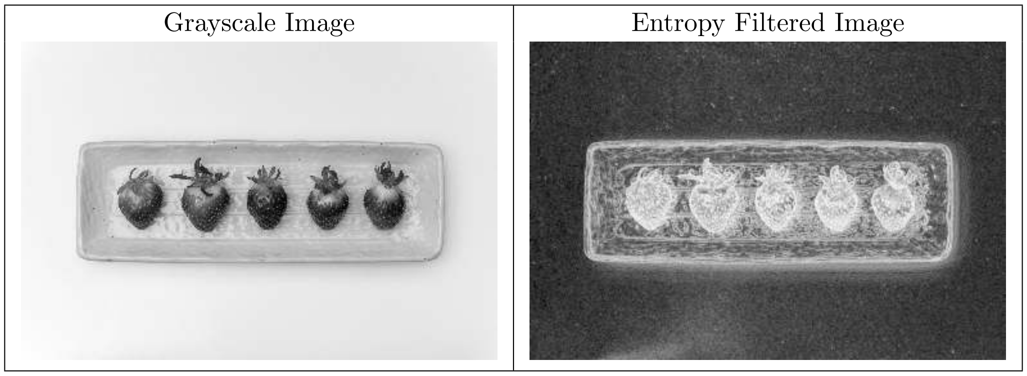


Figure 1: Local Entropy.

1.4.2 Global Thresholding

Global thresholding is based on the assumption that grayscale image pixel intensities form a bimodal histogram in which each peak represents one of two classes, background and foreground objects. The foreground objects can be extracted by simply applying a threshold and allocating all pixels with intensity above the threshold to the foreground object class and all others below the threshold to the background class.

Otsu's method is one of the more common clustering-based image thresholding techniques that searches for the optimal threshold value. The algorithm uses the two class assumption and iteratively searches for the optimum threshold such that the combined intra-class variance is minimal. This is calculated using the following formula:

$$\sigma_w^2(t) = w_0(t)\sigma_0^2(t) + w_1(t)\sigma_1^2(t) \tag{4}$$

The weights w_0 and w_1 are the probabilities of each class separated by the threshold t. The variances for each class are denoted by σ_0^2 and σ_1^2.

In reality, not all grayscale images form a bimodal histogram, in fact the image histogram can have multiple peaks and in such cases it is more appropriate to separate the image into multiple classes by multilevel thresholding. Yen's thresholding method [10] recognises that there is a discrepancy between the original image and thresholded image which decreases as number of classes increases. However, since increased number of classes raises the number of bits required to represent the image, Yen proposes a cost function to determine the most appropriate classification number.

$$C(k) = \rho(Dis(k))^{1/2} + (log_2(k))^2 \tag{5}$$

where k is the classification number, ρ is a positive weighting constant and $Dis(k)$ is the discrepancy between the thresholded image and original image.

1.4.3 Canny Edge Detection

The Canny edge detector uses a multi stage algorithm to detect a wide variety of edges within an image and is described in [11]. Figure 2 provides an example of Canny's algorithm applied to a simple image.

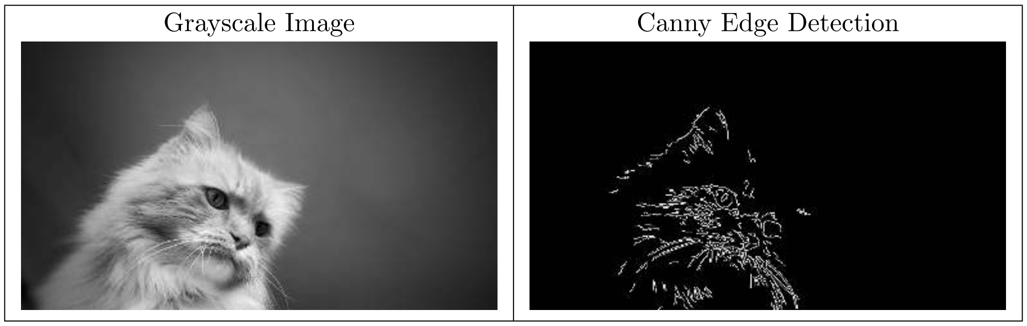


Figure 2: Canny Edge Detection.

1.4.4 Morphological Dilation

Morphological dilation is the process of expanding shapes contained within an image. The operation is performed using a structuring element that can be square or disk shaped. In binary images, this structuring element is then superimposed at each location on the image in which the central pixel is non zero.

1.4.5 Median Filtering and Contrast Stretching

Median filtering is a non-linear filtering technique that can be used to smooth and remove noise from an image. It achieves this by replacing each pixel in the image with the median of its neighbouring pixels.

Contrast stretching on the other hand aims to improve contrast by stretching the range of intensity values. In a typical grayscale image it may do this by first finding the minimum and maximum intensity values and then redistributing them across the entire intensity range adjusting minimum intensity pixels to 0 and maximum intensity pixels to 255.

2 Method

2.1 Image Preprocessing

The initial steps in preprocessing included downsizing the original image and then converting it to grayscale. This scaled down grayscale image formed the basis for further operations in the segmentation stage such as local entropy filtering, global thresholding and edge detection. Therefore, the grayscale image size was minimised to reduce the runtime of these subsequent operations while ensuring the size was great enough for ROIs to still be detected. For the testing phase of this project a scale of 8 was selected as the most appropriate constant for downsizing HD to Quad Ultra-HD images. This was applied by dividing both the width and height by 8 and obtaining a downsized image with an area equal to $\frac{1}{64}$ the original size.

Median filtering and contrast stretching were treated as optional steps in the preprocessing task. The median filter was used to see if it could reduce some of the textural distractions of the background while still preserving the transition edges between the ROIs and background. Contrast stretching has the effect of intensifying features within the image and was intended to make ROIs more separable from the background during segmentation. A block diagram in Figure 3 outlines the major steps performed in preprocessing the image prior to the segmentation stage.

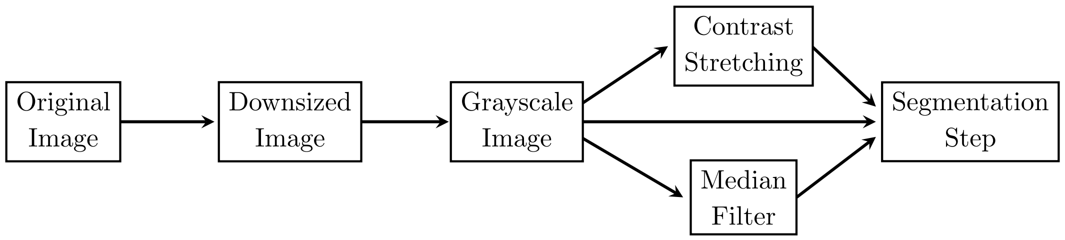


Figure 3: Image Preprocessing Block Diagram.

2.2 Image Segmentation

In order to separate the ROIs from the background, two different approaches have been designed and implemented. The first approach utilises local entropy and global thresholding while the other approach uses Canny's edge detector with morphological dilation of the edges. For simplicity these two methods will be referred to as the

LEGT and CEDD methods respectively.

2.2.1 Segmentation with Local Entropy and Global Thresholding (LEGT)

The first approach for image segmentation was to use local entropy and global thresholding. The grayscale image produced during the preprocessing stage was passed through a local entropy filter. The global threshold was then calculated by performing either Otsu's method or Yen's method on the entropy filtered image. The thresholded entropy filtered image was then obtained in binary form by applying the threshold to the entropy filtered image.

The final step was to perform connected component labelling to define the separate ROIs within the thresholded image. This produced bounding box positions and dimensions for each ROI within the downsized preprocessed image. All fully overlapped bounding boxes were discarded to avoid doubling up by classifying the same region. Each bounding box was also repositioned and rescaled up to match the original image in preparation for the final classification stage.

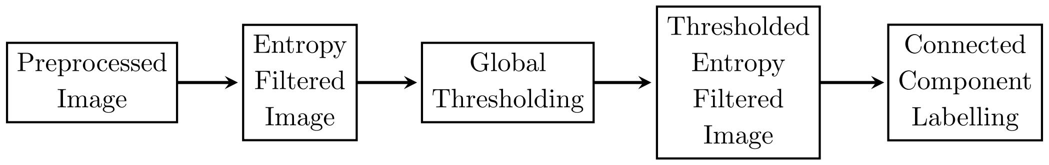

Preprocessed Image
Entropy Filtered Image
Global Thresholding
Thresholded Entropy Filtered Image
Connected Component Labelling

Figure 4: Segmentation by Local Entropy and Global Thresholding Block Diagram.

A visualisation of these steps is provided in Figure 5 which demonstrates separation of the plane from its background. It also provides the bounding box location for cropping the ROI from the original image in the final classification stage.

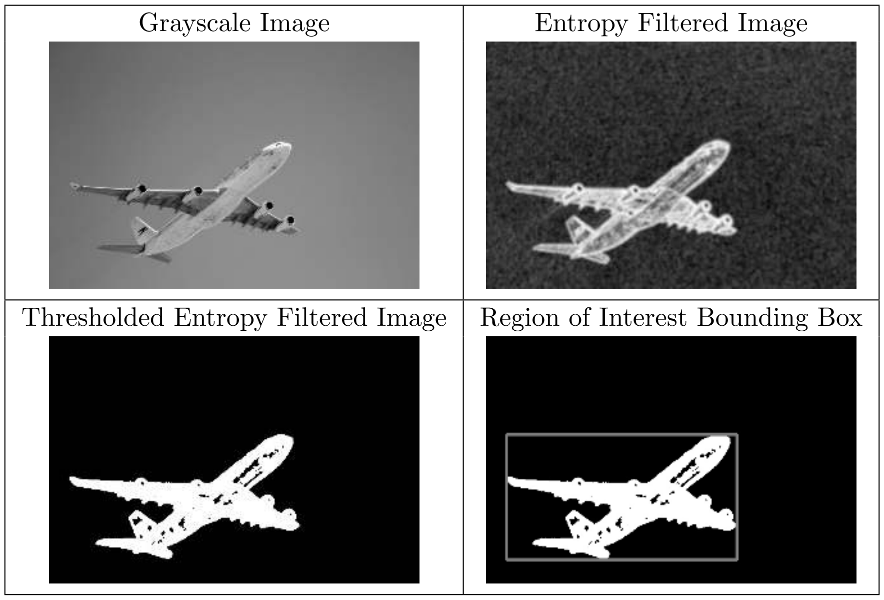

Grayscale Image
Entropy Filtered Image
Thresholded Entropy Filtered Image
Region of Interest Bounding Box

Figure 5: Region of Interest Extraction using Entropy and Otsu Thresholding.

The histogram in Figure 6 shows the ROI and background pixel intensities of this entropy filtered image and their separation into different classes by the global threshold.

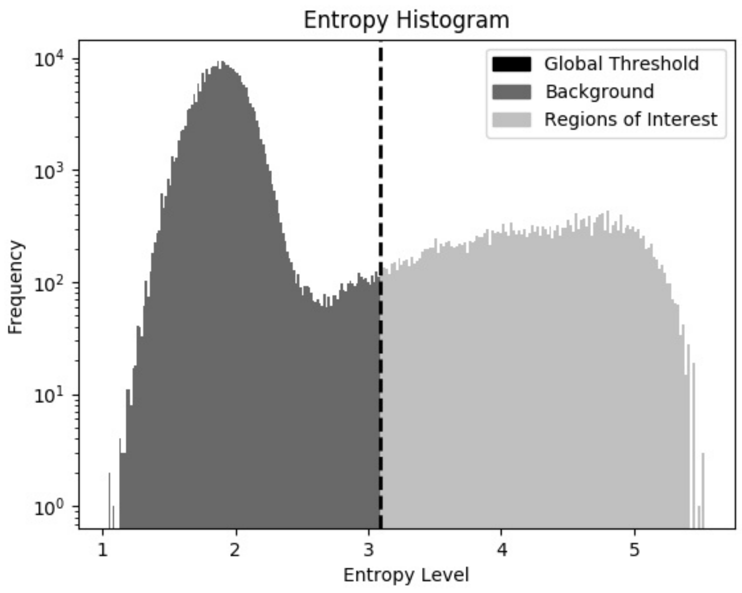


Figure 6: Local Entropy Histogram with Region of Interest Separated by Global Thresholding using Otsu's Method.

2.2.2 Segmentation with Canny Edge Detection and Dilation (CEDD)

The second approach used the Canny edge detection algorithm in order to find edges within the image and then define ROIs at the location of these edges. Since the Canny algorithm provided no guarantee of producing closed contours around each ROI the edges were dilated by a small amount with the intention of closing any gaps in the object boundaries. This reduced the likelihood of the ROI itself being segmented into smaller pieces. The final step of connected component labelling was identical to the previous approach in section 2.2.1.

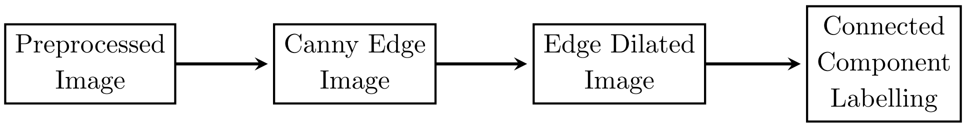


Figure 7: Segmentation by Canny Edge Detection and Dilation Block Diagram.

By dilating the edges slightly we avoid the issue of dividing the ROI into multiple pieces by closing gaps between edges surrounding the object. Figure 8 demonstrates the benefit of dilating the edges before performing the connected components operation.

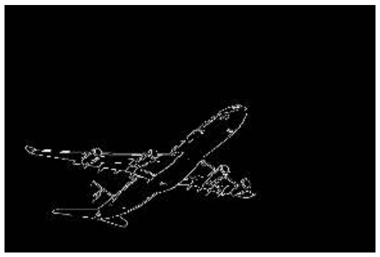
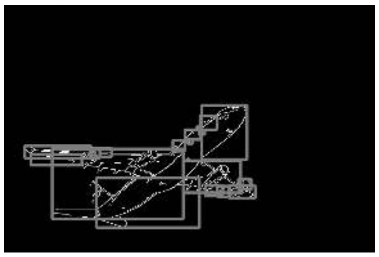
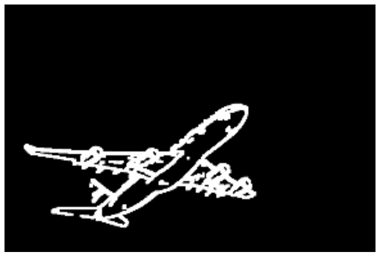
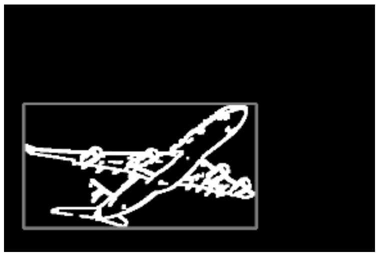
Canny Edge Image	Edge Connected Components
	
Edge Dilated Image	Dilated Edge Connected Components
	

Figure 8: Region of Interest Extraction using Canny Edge Detector and Dilation.

2.3 Image Classification

Upon completing the segmentation stage, a list of non fully overlapped rectangles defined by position and dimensions within the original image is passed as input to the classification stage. Each of these rectangles corresponds to a potential ROI within the image. Rectangles that are under a certain size and do not hold the potential to be successfully classified are discarded. The remaining rectangles are used to crop sections of the image referred to as ROI images which are then passed on to the deep learning classification model.

The two models used in the classification stage of this project included AlexNet [1] and DeepLab [6]. AlexNet was chosen due to its performance and popularity in the field of deep learning. On the other hand, DeepLab which implements a method for Semantic Image Segmentation presents the advantage of supporting variable input size. The ability to pass images of varying input size allowed more flexibility in resizing each ROI image during classification.

2.3.1 Classification with AlexNet

The AlexNet model pretrained according to the ImageNet Large Scale Visual Recognition Challenge 2012 (ILSVRC2012) by Berkeley Artificial Intelligence Research was used as the first model of choice for this project. During classification each of the input ROI images would undergo a resize operation to adjust its resolution to fit the input size of the AlexNet model. The AlexNet model would then run through several batches of images until all ROI images had been classified.

Once each of the ROI images had been classified an output probability threshold of 0.6 was applied to remove any weak classifications. In the case that a successful classification over this threshold was made, a bounding box with a classification label was placed in the original image. Figure 9 provides an example of AlexNet classifying 200 randomly placed items of fruit in a synthetic 8K image with an enlarged section for clarity.

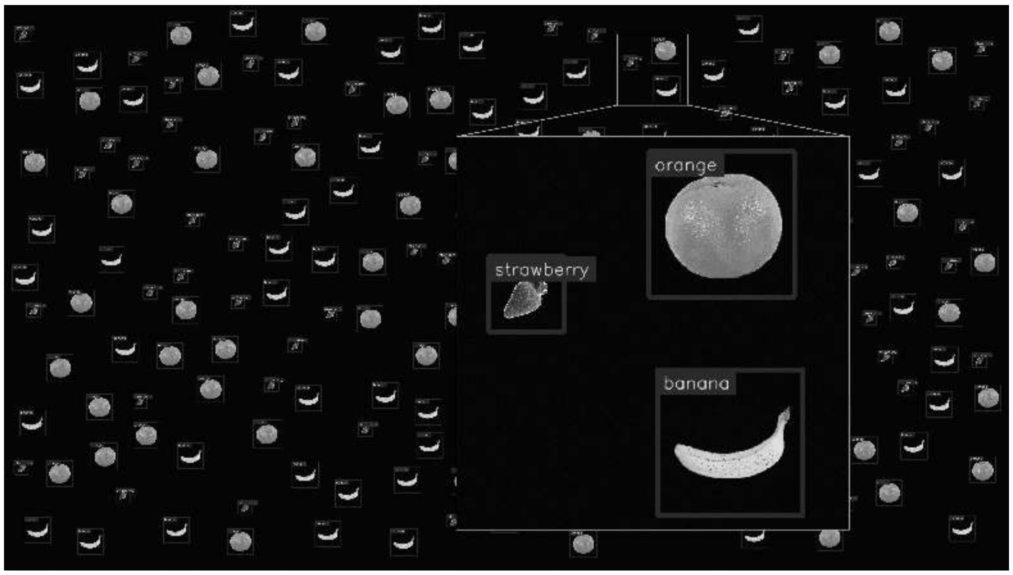


Figure 9: AlexNet Classification on Synthetic Image with 8K resolution.

One of the obvious drawbacks of this model was the restriction on input size. Since AlexNet was trained using colour images with 227×227 resolution, each potential ROI determined during the segmentation stage required the cropped image region to be rescaled to this input size. This had the disadvantage of losing accuracy in cases where the object occupied only a tiny portion of the ROI image as it became too small after adjusting the resolution. On the other hand, if the entire ROI image was smaller than the input size, it would be enlarged therefore requiring more unnecessary computational resources without improvements in accuracy.

Another disadvantage of AlexNet was that it only provides one label for each ROI image. Therefore in such cases where an ROI image contained multiple overlapping objects of different classes, all but perhaps one object classification would be lost. For this reason, DeepLab was chosen as the new model and was used as the main classification model during testing.

2.3.2 Classification with DeepLab

The implementation of DeepLabv3+ with MobileNetv2 backbone used in this project can be found at [12][1] and was pretrained on the twenty classes of the Pascal Visual Object Classes 2012 challenge. As discussed previously in section 2.3 a list of ROI images that contained smaller pieces of the original was passed in as input to the

[1]Codebase: https://github.com/bonlime/keras-deeplab-v3-plus

classification stage. Once a full scale black image mask equal in size to the original image was created each ROI image was scaled down and classified. The resulting ROI mask was then rescaled back up and for each classified pixel in the ROI mask, the corresponding class label colour was placed on the full scale mask. Once classification was complete, the mask was overlaid on the original image.

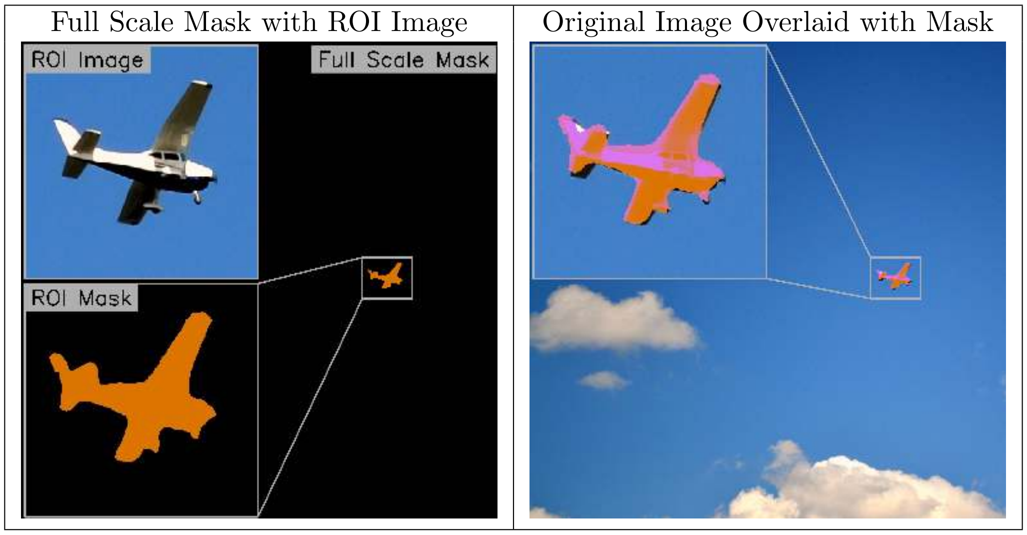

Full Scale Mask with ROI Image
Original Image Overlaid with Mask
ROI Image
Full Scale Mask
ROI Mask

Figure 10: DeepLab Classification with Mask Overlay on Original Image.

In the case of DeepLab, the variable input size presented the opportunity for fine tuning the amount each ROI image was rescaled. Due to the pixel labelling strategy of semantic image segmentation, it was also possible to locate and label multiple objects within one ROI image.

3 Results and Discussion

3.1 Image Datasets

During the testing phase of this project a series of experiments were performed on three different datasets. The datasets all contained images with one or more objects from the 20 classes of the PASCAL VOC 2012 challenge. The 20 classes are listed in Table 1. The details of the datasets used were as follows:

1. Synthetic 16K Dataset: A set of Quad Ultra-HD (15360×8640 pixels) images synthetically created with low entropy backgrounds and a number of classifiable objects placed at random. There were a total of 7 images containing between 1 and 150 objects selected from a set of 20 cropped classifiable objects.
2. HD Dataset: A set of random high definition (1280×720 pixels) images each with at least one classifiable object. The images tended to contain high entropy backgrounds that were not easily separable from the ROIs within the image. The dataset itself can be broken down into three different groups.
   (a) 104 HD images consisting of a single object.
   (b) 28 HD images consisting of multiple objects of the same class.
   (c) 16 HD images consisting of multiple objects of different classes. There were a total of 37 different objects within the 16 images.
3. 4K+ Dataset: A set of 69 images ranging from 4K (3840×2160 pixels) to 24 Mpx (6000×4000 pixels). The dataset contained a mixture of images containing a single classifiable object or multiple objects of the same or different classes. In total there were 82 different objects within the 69 images. To test the effectiveness of this project implementation, a fraction of the images were selected due to their relatively plain backgrounds. These images typically consisted of sky or lakes with the object such as a plane, boat or bird existing far in the distance.

Table 1: PASCAL VOC (2012) List of Twenty Classes Plus Background.

Background	Aeroplane	Bicycle	Bird	Boat	Bottle	Bus
Car	Cat	Chair	Cow	Dining Table	Dog	Horse
Motorbike	Person	Potted Plant	Sheep	Sofa	Train	Television

3.1.1 Assessing the Accuracy of Results

The assessment of results for each classification differed between the synthetic and real image datasets. For the Synthetic dataset, each image was accompanied by a ground truth mask which was compared pixel for pixel with the resulting mask from classification. The accuracy for each image was measured as the percentage of correctly classified pixels ignoring background pixels within the ground truth mask. The background pixels were ignored since they occupied the majority of the image area and would typically be the same even for poorly classified results.

In the case of the real image datasets (HD and 4K+), due to time restrictions on preparing the dataset, a simplified approach for scoring accuracy was used. After each classification, a list of objects classified within the image was obtained and compared with the expected list of objects. The accuracy was determined as the percentage of expected objects found by the classification model. For example in the case of the HD dataset groups a, b and c were scored out of 104, 28 and 37 respectively. Any extra classified objects that were not correct were discarded without penalty. While this testing method was not ideal it did provide an indication of whether the segmentation approaches were successful in locating objects amongst a diverse range of backgrounds from real world examples.

3.2 Factors Considered for Performance

In order to optimise performance, a range of factors have been explored for each of the segmentation methods. The major factors included:

1. Using all methods
    - maximum ROI image resolution (maxRes): The maxRes parameter provided an upper bound for the resolution of each ROI image passed into the classification model. During testing it was varied with the intention to find the minimal resolution that still achieved comparable accuracy to the baseline performance. Due to the varying aspect ratio of ROI images, the maxRes parameter was measured by area (for example maxRes = 200 refers to an upper bound ROI image size equal in area to a 200×200 square image).
    - Number of ROI images (nROI): It was expected that the number of ROI images processed at the classification stage would have a large influence on performance. A linear relationship between runtime and number of ROI images was expected.

2. LEGT method only.

    - Local entropy neighbourhood structuring element radius (Ωrad): The neighbourhood window used in local entropy was defined by a disk structuring element with a specified radius. It was expected that the size of the neighbourhood used to obtain the local entropy filtered image would affect the ability for global thresholding to isolate objects from their background.
    - Global thresholding: Comparison between using Otsu's method or Yen's method.
    - Contrast stretching: As an optional preprocessing step.
    - Median filtering: As an optional preprocessing step also with varied structuring element radius.

3. CEDD method only:

    - Edge dilation neighbourhood structuring element size (Ωn): The degree at which the edges were dilated was controlled by the size of the neighbourhood window. The structuring element used for dilation was square in shape and defined by a single dimension (for example $\Omega n = N$ defines a neighbourhood window that is $N \times N$ in size). It was expected that increased edge dilation limited the possibility of an object being divided into multiple ROI images. However, if the edges were dilated too much this could lead to overlap between two distinct objects and therefore increased chance of large ROI images with more than one object.

3.2.1 Measuring Baseline Performance

During testing the baseline performance was defined as taking the entire image and classifying it at multiple scales using the DeepLab model. As such, multiple runs were performed on each dataset in order to classify the images at different resolutions. This provided a fair performance comparison with the segmentation methods that also tended to shrink down ROI image size before classification by tuning the maxRes parameter.

3.3 Synthetic Dataset

3.3.1 Creating the Synthetic Dataset

As mentioned previously the synthetic dataset comprised a collection of 16K images with a varied number of classifiable objects placed on a low entropy dark background. In order to create the synthetic dataset, a set of 20 cropped object images with transparent backgrounds was collected. Figure 11 lists the 20 object images used in creating each synthetic image.

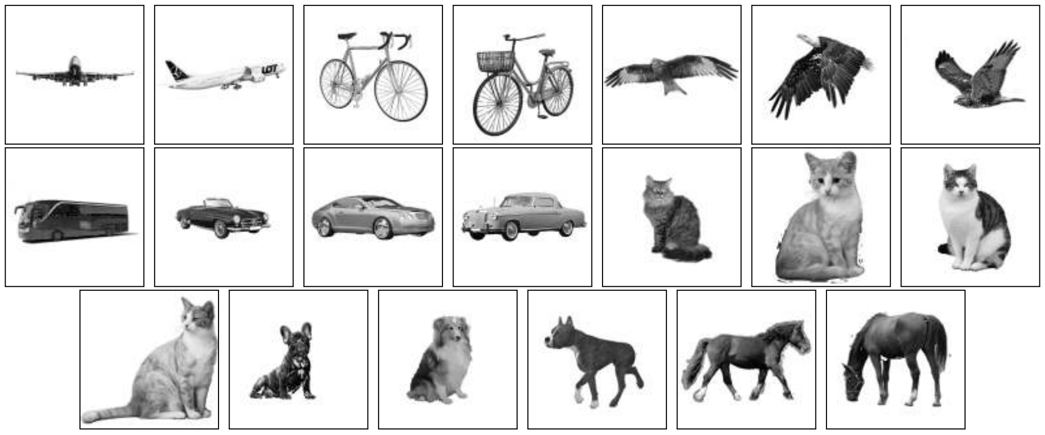


Figure 11: Cropped Object Images used to Construct Synthetic Dataset.

The process for creating each synthetic image and its associated mask was as follows:

1. Create a 16K dark image with a slight amount of background noise. To simulate the slightly noisy background a pixel intensity between 0 and 10 was chosen at random.

2. Create a 16K black image with no noise to use as the full scale ground truth mask.

3. For each of the cropped images, create a corresponding ground truth mask by first constructing an equal size black image. Then for each non background pixel in the cropped image label the corresponding pixel in the mask with the associated class label.

4. Place detectable objects from the cropped image set onto the synthetic image. If there are more than 20 objects to be placed, cycle through the list from the beginning repeatedly.

5. For each cropped image placement, copy the corresponding ground truth mask created in step 3 onto the 16K ground truth mask at the same position for assessment of classification accuracy later.

Following this process a total of 7 images were created each with a different number of objects ranging from 1 object to 150 objects.

3.3.2 Testing the Synthetic Dataset

During testing of the synthetic dataset, four main objectives were considered:

1. How well do the two segmentation methods perform compared to the baseline performance.
2. What is the optimal maxRes value to achieve good accuracy while minimising runtime.
3. How do each of these methods scale with increasing number of objects or nROI present within the image.
4. How does the variance and mean of the local entropy filtered image relate to the minimum resolution required for classification.

3.3.2.1 Comparison with Baseline Performance On Synthetic Image

A runtime comparison of the two segmentation approaches with the baseline performance on a synthetic image containing 20 objects is provided in Figure 12. It is clear that under the controlled background conditions of this synthetic image, both approaches perform far better than the baseline performance. In this case, segmentation using CEDD proved to be the most effective approach but only by a small margin. For perspective, the baseline performance of classifying the entire image at 4K resolution took 101 seconds and achieved 65.8% accuracy. Both CEDD and LEGT methods scored over 70% accuracy in approximately 20% of the time taking 18.5 and 21.6 seconds respectively.

AlexNet classification was also tested using each respective segmentation approach for analysing runtime. In this case, each of the ROI images was fitted to the fixed 227×227 pixel resolution. For comparison, AlexNet required 11.3 and 14.7 seconds to classify the same image with the LEGT and CEDD methods respectively. However, the accuracy was much lower with only 2 of the 20 objects recognised in both cases. This is likely due to the fact that the synthetic image was prepared using cropped images consisting of objects from the pre-trained DeepLab class labels.

Therefore, these results are reported only for runtime comparison between using DeepLab or AlexNet on top of each segmentation method.

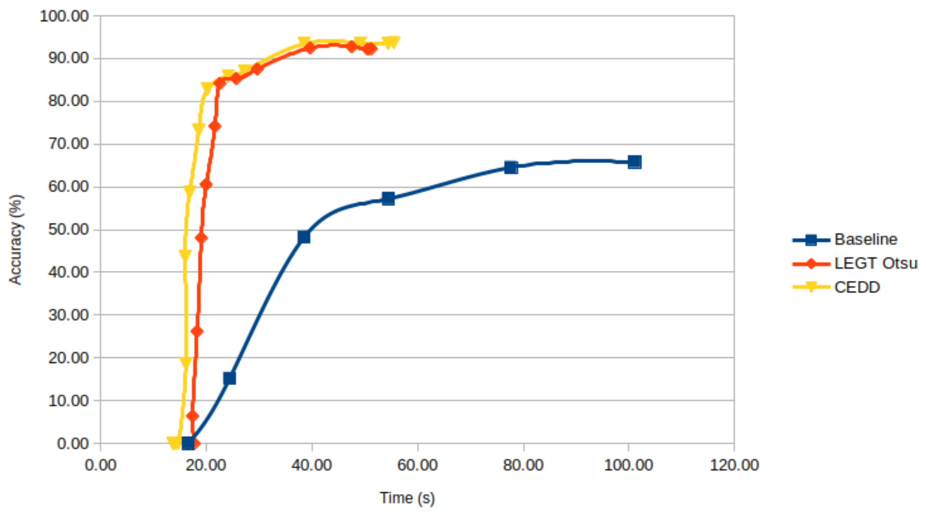


Figure 12: Comparison of DeepLab Classification on Synthetic Image.

In this test, the LEGT method used a neighbourhood window radius of $\Omega rad = 14$ for local entropy and used Otsu's method for thresholding. Segmentation using CEDD used a neighbourhood window size of $\Omega n = 14$ for the dilation operation. Each segmentation method was tested a number of times with increasing maxRes values which closely resembled the runtime. For baseline performance, the process described in section 3.2.1 was followed, however, due to memory restrictions could only be tested up to 4K resolution.

3.3.2.2 Optimal maxRes Choice for Fast Runtime and High Accuracy

The same test results are presented in Figure 13, however in this case accuracy is compared to maxRes value for the CEDD and LEGT methods. As mentioned previously, there was a close relationship between maxRes and runtime in which higher maxRes leads to longer runtime. From inspecting the graph two points of interest can be seen. The first point is located at the top of the steep increase in accuracy at a maxRes value of 200 for both methods. The other can be seen looking beyond this point where the accuracy increased slowly and reached a maximum at a maxRes value of 400. Choosing either of these maxRes values would be most appropriate for classifying this image. The choice between them would depend on the tradeoff between speed and accuracy as running the model with a maxRes value of 200 is faster but slightly less accurate.

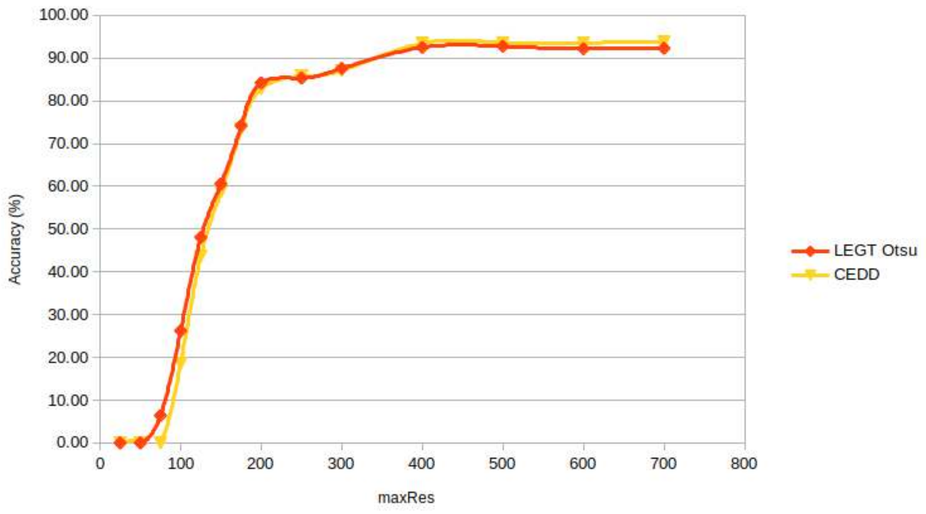


Figure 13: Relationship Between maxRes Size and Accuracy.

Fine tuning of the maxRes value using a real dataset would likely depend on the nature of the images. In particular, a dataset with high entropy backgrounds containing grass or trees for example would likely not perform well with lower maxRes values. This is because the ROI images would not isolate the classifiable objects as effectively and would therefore need higher resolution to classify correctly.

3.3.2.3 Scalability of Segmentation Methods with Number of Objects

One of the important features of the segmentation methods used in this project is that their runtime depends on the number of ROI images processed during classification. Since the DeepLab model was run on each ROI image, an increasing number of objects within an image was expected to lead to an increasingly long runtime. To test this relationship, 7 synthetic images were created each with 1, 5, 10, 20, 50, 100 and 150 randomly placed objects. The testing results proved that there was a linear relationship between the number of objects and runtime shown in Figure 14. This was the case when using both DeepLab and AlexNet for classification.

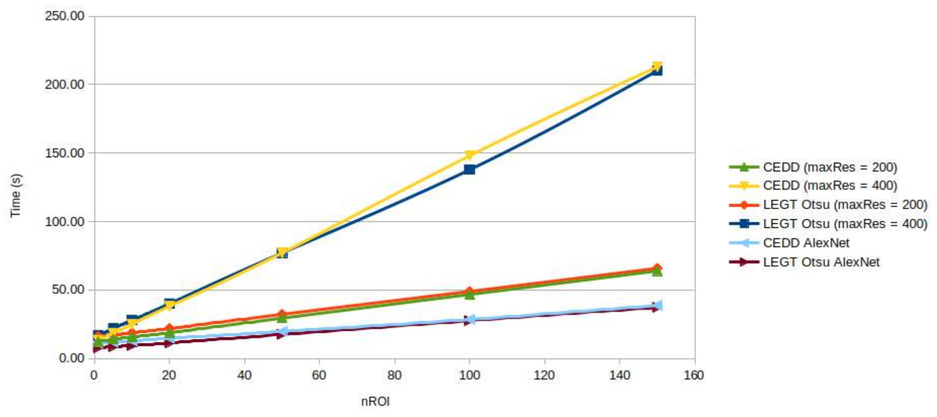


Figure 14: Scalability of Segmentation with DeepLab and AlexNet Classification with Increased Number of Objects.

3.3.2.4 Relationship Between Local Entropy and Minimum Resolution

Another interesting relationship to explore was the influence of local entropy on the minimum resolution required for successful classification. To explore this theory, the 20 cropped images used in forming the synthetic dataset were each tested with a range of values for maxRes. For each image, the minimum maxRes value that led to a successful classification was recorded. These minimum maxRes values were then plotted against the variance and mean of the corresponding entropy filtered image as shown in Figure 15.

Through observation of the graphs we can see that images containing higher mean and variance in their corresponding entropy filtered image tended to be successfully classified at a lower resolution. This is an important relationship as it provides a useful criteria for how much the resolution of each ROI image can be reduced to before classification. Although this relationship provides a promising technique for improving performance, the amount of image data used in this test was small and the background of each image is unrealistic. In order to further explore this relationship a more controlled environment tested on actual ROI images from high resolution samples would provide stronger evidence that this relationship exists and can be exploited. Such a task could not be completed in the given timeframe but presents an opportunity for future work.

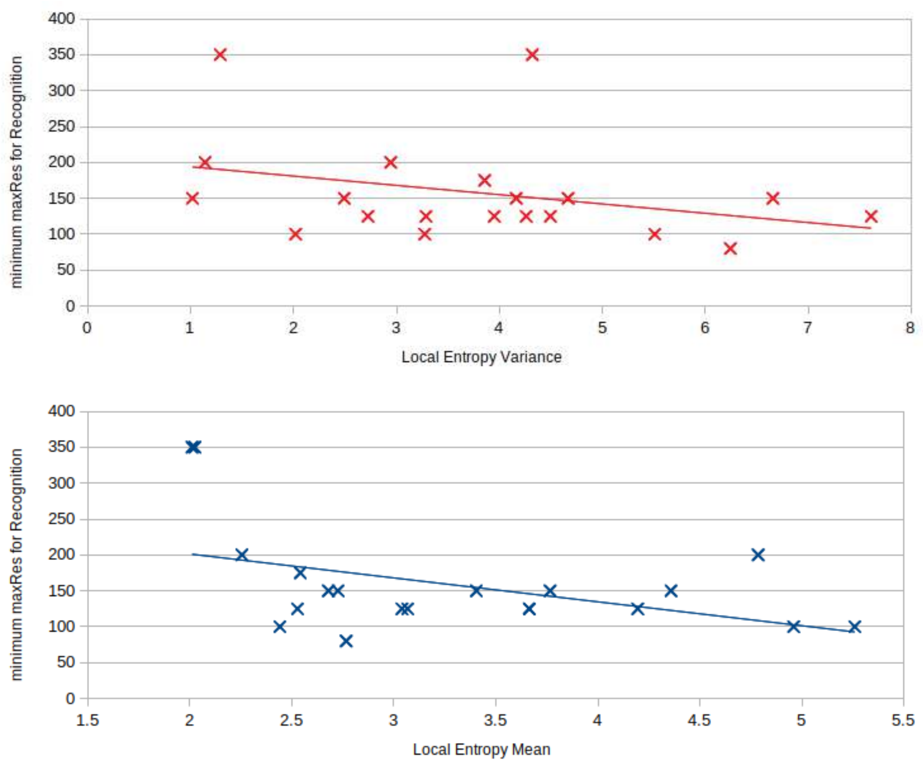


Figure 15: Relationship Between Local Entropy and Minimum Resolution for Successful Classification: Top - Variance of Local Entropy Filtered Image with Minimum maxRes Value, Bottom - Mean of Local Entropy Filtered Image with Minimum maxRes Value.

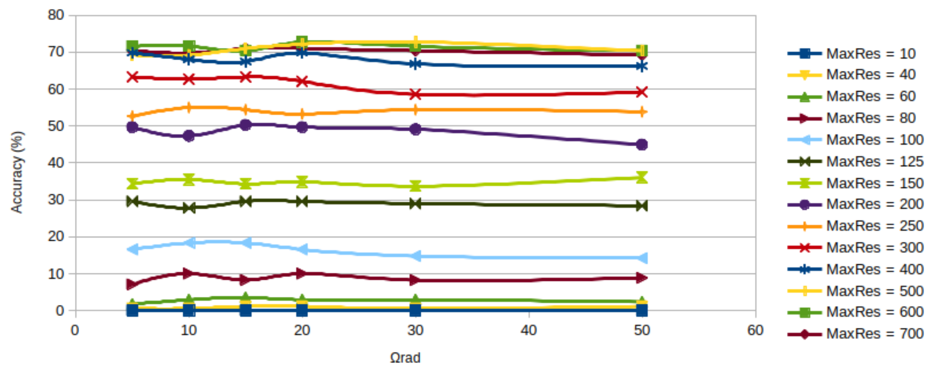
3.4 HD and 4K+ Datasets

While the results of testing the project implementation on synthetic images were promising, it was also important to see how it performed on real images with realistic backgrounds. During testing of the two real image datasets, the main objectives were as follows:

1. Determine optimal parameters and factors to maximise accuracy in classification for each segmentation method.
2. Compare segmentation methods with baseline performance.

3.4.1 LEGT with Otsu's Thresholding Method

The neighbourhood window radius (Ωrad) was expected to play a role in the ability for ROI images to be effectively isolated from their background by global thresholding. However, as depicted in Figure 16 the results were pretty similar with varying values of Ωrad. This is likely due to the assumption of Otsu's method that assumes the image can be defined by a bimodal histogram which can be separated into two classes. Since many of the images in this dataset contained items such as trees in the background, this was not the reality and the thresholding technique tended to include large sections of the background in the ROI images. Nonetheless, selecting values for Ωrad in the range of 12 to 16 tended to show slightly higher results provided a relatively large value for maxRes was selected.



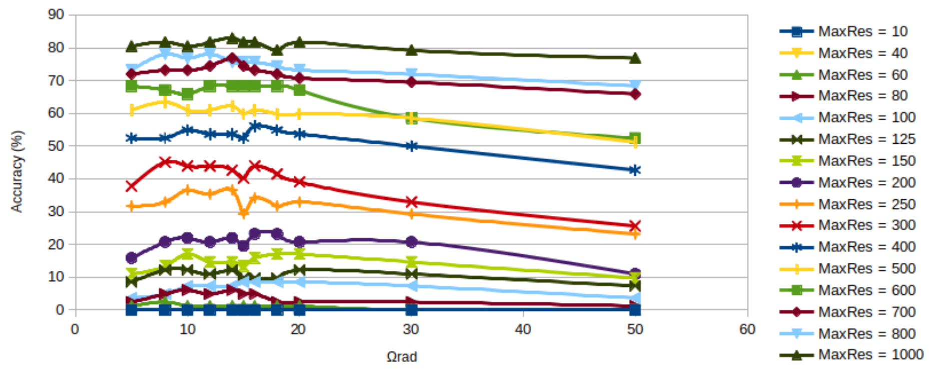
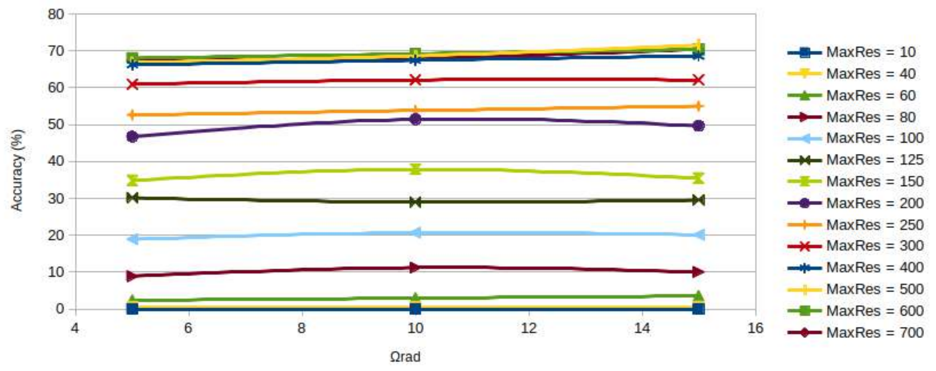


Figure 16: DeepLab Classification with LEGT using Otsu Thresholding: Top - HD Dataset with varying values of Ωrad, Bottom - 4K+ dataset with varying values of Ωrad.

3.4.2 LEGT with Yen's Thresholding Method

A similar set of tests was run using LEGT with Yen's thresholding method. In this case there was a slightly more pronounced relationship between accuracy and neighbourhood window radius. During testing of the 4K+ dataset shown at the bottom of Figure 17, values of 14 and 15 for Ωrad obtained the highest results in most cases while low values performed poorly by comparison.



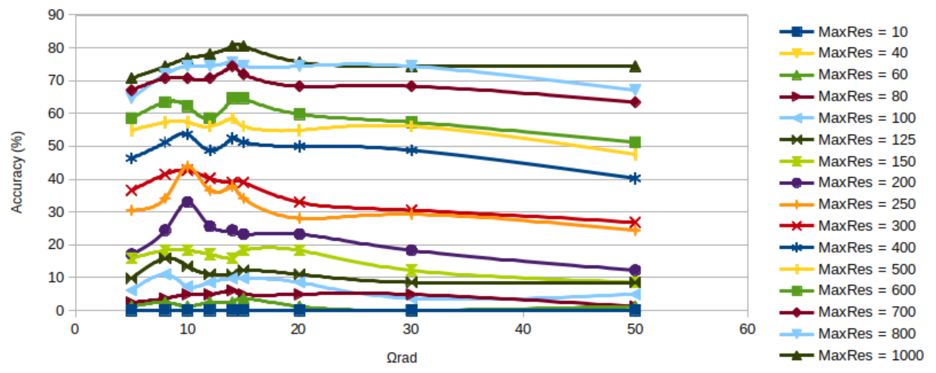
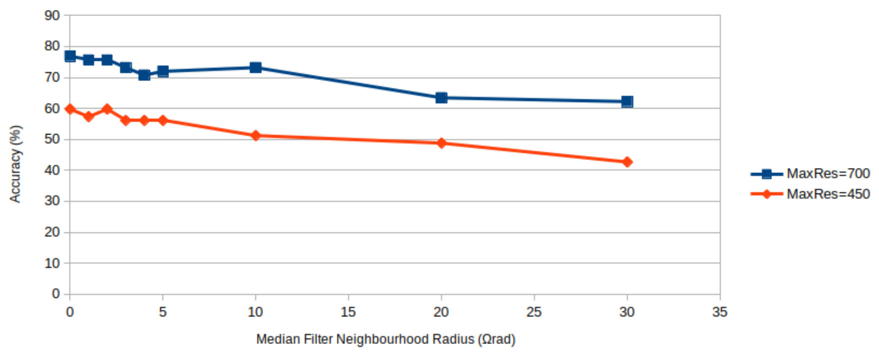


Figure 17: DeepLab Classification with LEGT using Yen Thresholding: Top - HD Dataset with varying values of Ωrad, Bottom - 4K+ dataset with varying values of Ωrad.

3.4.3 LEGT with Median Filtering and Contrast Stretching

Median filtering and contrast stretching did not provide good results when applied as additional preprocessing steps. The results in Figure 18 reflect a steady decrease in both cases where median filter was used with increasing structuring element radius (Ωrad) and greater contrast stretching was applied.

The degree of contrast stretching applied was controlled by selecting the the minimum and maximum pixel intensities at a particular percentile. For example contrast stretching with a percentile choice of 5 indicates that the maximum and minimum pixel intensities were selected from the distribution at the 95th and 5th percentile respectively. The selected pixels intensities were then stretched to the limiting intensities of 0 and 255 with all remaining pixels between the 5th and 95th percentile distributed across this range.



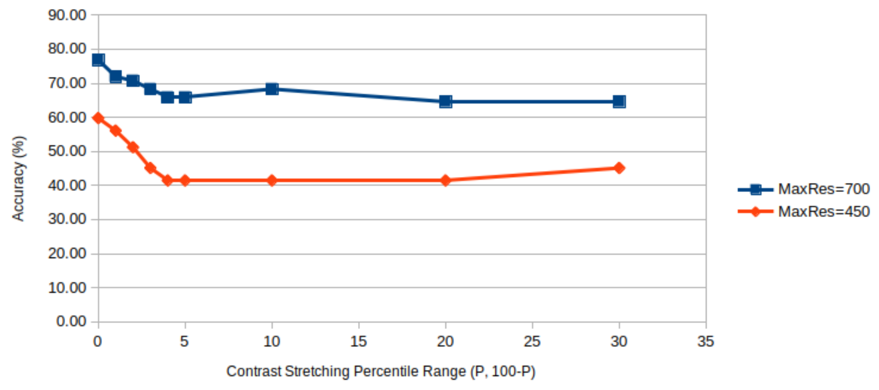


Figure 18: DeepLab Classification with LEGT using Median Filtering and Contrast Stretching: Top - 4K+ Dataset with Median Filtering Applied, Bottom - 4K+ dataset with Contrast Stretching Applied.

3.4.4 CEDD with Varying Degree of Dilation

The main parameter explored in testing the CEDD method accuracy was the neighbourhood window size Ωn used in the dilation operation. As mentioned previously, a value for $\Omega n = N$ defined a square neighbourhood window of size $N \times N$. By modifying the size of the Ωn parameter, the degree of dilation could be controlled.

From the results shown in Figure 19 the degree of dilation did have an impact on accuracy, particularly in the case of the 4K+ dataset. Specifically, values of Ωn in the range of 14 to 20 typically performed well. The reason for this is likely attributed to the fact that the Canny edge detection algorithm does not guarantee closed edges around each object. Therefore, with less dilation there was a tendency for the object to be separated into several ROI images, this situation was demonstrated previously in Figure 8.

On the other hand, there is a decrease in accuracy as the degree of dilation becomes too high. This could be caused by one of two factors. Firstly, greater dilation leads to larger ROI images that contain smaller objects relative to their ROI image size. Secondly, overlap was more likely between one object and another distinct object or background feature which also leads to large ROI images with comparably small objects.

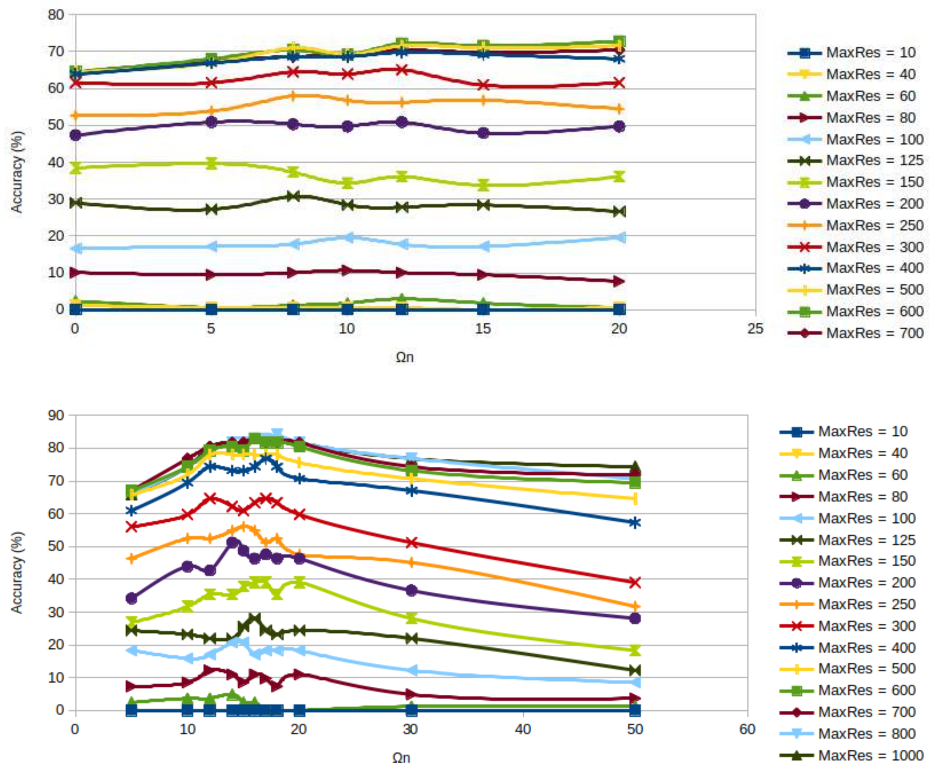


Figure 19: DeepLab Classification with CEDD: Top - HD Dataset with varying values of Ωrad, Bottom - 4K+ dataset with varying values of Ωrad.

3.4.5 Comparison with Baseline Performance on HD Dataset

For comparing performance, each method was tested on each dataset with multiple different maxRes values as in the case of the synthetic dataset. In this case, the CEDD method offered the best performance although the margin was quite small. This provides some evidence that even in the presence of distracting features from the background, the CEDD method can still divide the image up into relatively small ROI images that capture the objects.

On the other hand, the LEGT method produced similar results to the baseline performance using either Otsu or Yen thresholding. This is likely due to the fact that the high entropy backgrounds have caused the thresholded entropy filtered image to contain large sections of the background as well. As such, the segmentation stage using LEGT tended to form one single ROI image that tended to be close if not equal to the full image size.

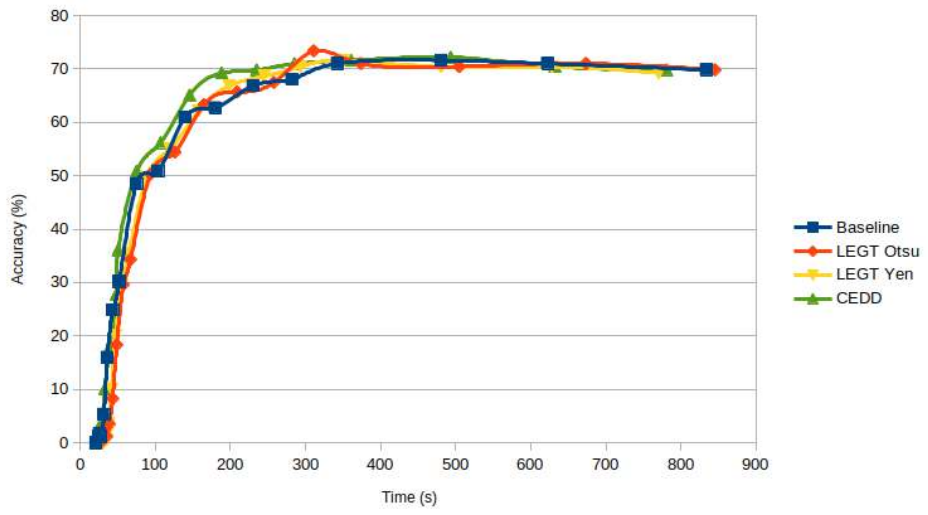


Figure 20: Comparison of DeepLab Classification on HD Dataset.

3.4.6 Comparison with Baseline Performance on 4K+ Dataset

In the case of the 4K+ dataset, the performance of the CEDD method was better by a significant margin. If we compare runtimes to obtain an accuracy of 83% the CEDD method required 9 minutes and 23 seconds which is approximately 22% of the time required by the baseline performance at almost 43 minutes.

A similar trend can be found for the LEGT method when compared with the HD dataset as it demonstrated similar results to the baseline performance. This suggests that the lower entropy backgrounds present in the 4K+ dataset were still not very effectively separated from the classifiable objects within each image.

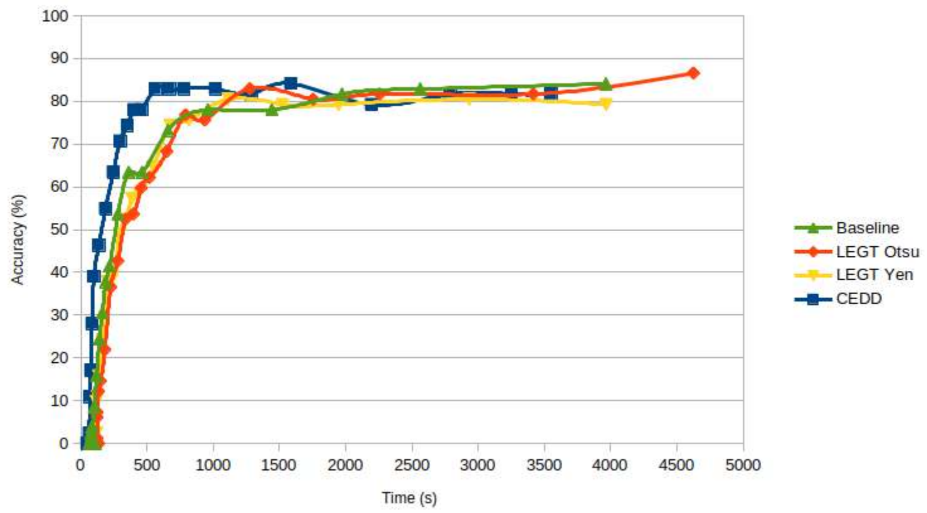


Figure 21: Comparison of DeepLab Classification on 4K+ Dataset.

4 Conclusion

In this project, a method for segmenting high resolution images into multiple regions of interest for runtime efficient classification has been implemented. To this end, two approaches for segmentation have been designed and tested including the local entropy with global thresholding (LEGT) method and the Canny edge detection with dilation (CEDD) method. For the task of classifying the smaller image segments, AlexNet and DeepLab have been used. In addition, three image datasets have been prepared including one synthetically created Quad Ultra-HD dataset and two genuine datasets (HD and 4K+) consisting of real world images. Using these datasets, extensive experimentation with varying parameters have been taken into consideration to determine their influence on performance. The key findings of this research project are presented below.

During testing, both methods performed well in cases where the background was homogenous, however only the CEDD method outperformed basline performance in images with more complex backgrounds. For a synthetic image containing twenty different objects, runtime was reduced down to approximately 20% of the baseline performance time using either the LEGT method or the CEDD method. During testing of the two real world image datasets, the LEGT method tended to select the entire image. In contrast the CEDD method still located ROIs successfully and in fact achieved similar accuracy in 22% of the time taken by baseline performance when classifying the 4K+ dataset.

Tests of multiple images from the synthetic datset proved a linear relationship between number of objects within an image and runtime. In addition to this a connection between the variance and mean of an entropy filtered image and the minimum resolution required for successful classification was demonstrated. For performance tuning of the LEGT method, a structuring element radius of 14 or 15 proved to be the most appropriate when defining the local entropy filtered image from real datasets. Similarly a square structuring element dimension of 14 to 20 provided the best results when dilating the edges using the CEDD method.

Overall both the LEGT and particularly the CEDD method achieved good results in less runtime than the baseline performance particularly if the background was homogeneous. In a real world setting global thresholding tended to include large regions of the background in the segmented ROIs. A possible direction for future research could explore other techniques in combination with the entropy filtered image such as the watershed algorithm. Another promising area to explore is the relationship between statistical measures of the entropy filtered image and minimum resolution required for classification. Although evidence of its existence is provided in this paper more thorough testing could clarify how it can be utilised to optimise performance.

References

[1] Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton. Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*, 25:1097–1105, 2012.

[2] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. *IEEE conference on computer vision and pattern recognition*, pages 248–255, 2009.

[3] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2015.

[5] Forrest Iandola, Song Han, Matthew Moskewicz, Khalid Ashraf, William Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and 0.5 mb model size. *arXiv preprint arXiv:1602.07360*, 2016.

[6] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):834–848, 2017.

[7] Guiosheng Lin, Anton Milan, Chunhua Shen, and Ian Reid. Refinenet: Multipath refinement networks for high-resolution semantic segmentation. *IEEE conference on computer vision and pattern recognition*, pages 1925–1934, 2017.

[8] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoderarchitecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(12):2481 – 2495, 2017.

[9] Chengxin Yana, Nong Sanga, and Tianxu Zhang. Local entropy-based transition region extractionand thresholding. *Pattern Rocognition Letters*, 24(16):2935–2941, 2003.

[10] Jui-Cheng Yen, Fu-Juay Change, and Shyang Chang. A new criterion for automatic multilevel thresholding. *IEEE Transactions on Image Processing*, 4(3):370–378, 1995.

[11] John Canny. A computational approach to edge detection. *Readings in computer vision*, pages 184–203, 1987.

[12] Emil Zakirov. Keras implementation of deeplab v3+ with pretrained weights. https://github.com/bonlime/keras-deeplab-v3-plus, 2019.

A Overview of Implementation

A.1 Source Code Location

https://github.com/joshlong90/Multiple-ML-Models-on-HiRes-Images

A readme file provides information on environment setup and running the program at this location.

A.2 File Structure

Table A.1: Source Code File Structure.

main.py	Contains main program that parses options and arguments.
imgproc.py	Contains image pre-processing methods.
imgproc.py	Contains image segmentation methods.
classifyDeepLab.py	Contains DeepLab classification and annotation methods.
classifyAlexNet.py	Contains AlexNet classification and annotation methods.