1. Introduction

The safety and accuracy of autonomous vehicle navigation in natural environments rely heavily on their ability to recognize and respond to various scenarios and objects they may encounter. Due to the complexity, unscriptural and unstructured formation found in natural environments, fine-grained semantic segmentation of images captured by the vehicle’s cameras is essential for scene understanding. The pixel data extrapolated from the images captured will allow the determination of its natural surroundings including and not limited to terrain data, road conditions and surrounding vegetation. This project aims to develop and compare different computer vision methods for semantic segmentation in these challenging natural environments.

Given the complexity and variability of natural terrains, our approach focuses on developing a model that is capable of accurate segmentation. We utilised two advanced deep learning methods(U-net and Deeplabv3) along with implementing traditional computer vision techniques. These methods both aim to maximise segmentation accuracy and improve the vehicle's ability to navigate safely through diverse and unpredictable natural landscapes. As such, these methodologies hold promise for enhancing autonomous navigation, contributing to the broader application of autonomous systems in various challenging environments.

We started by gaining a comprehensive understanding of the WildScenes dataset's characteristics. The dataset contains 9,306 2D images of size 2,016 x 1,512 pixels capturing various natural environments. Due to its considerable size, processing the entire dataset would be computationally intensive and time-consuming. Therefore, we strategically selected a representative subset of images from the dataset through data sampling. This subset maintains the diversity and complexity of the original dataset, ensuring that our analysis and model training remain robust while optimising resource usage and processing time.

2. Literature Review

Semantic segmentation is the process of pixel-level image recognition, labelling each pixel according to the object class it belongs to. Unlike image classification, which assigns a single label to an entire image, semantic segmentation involves dense pixel-wise classification based on the model.

Through our research, we conclude that DeepLabv3 is one of the models that can significantly advance the performance of semantic segmentation tasks. The primary contributions of DeepLabv3 include improvements in Atrous Spatial Pyramid Pooling (ASPP) and the use of cascaded atrous convolutions. These enhancements build on previous works like DeepLabv2 and the atrous convolution techniques, aiming to refine the ResNet model. The ASPP in DeepLabv3 integrates image-level feature connections, a 1x1 convolution, and three 3x3 atrous convolutions with varying rates. This configuration allows the model to capture multi-scale contextual information effectively. Each parallel convolution layer is followed by batch normalisation, which improves stability and performance(Chen L C., 2018). On the other hand, DeepLabv3 employs a cascaded ResNet module with atrous convolutions at different rates. This approach is applied directly to intermediate feature maps rather than the final confidence maps, enhancing the model’s ability to capture detailed features at multiple scales(Chen L C., 2017).

In addition to DeepLabv3, U-Net has also been found helpful in this task. U-Net is constructed by a dual-path architecture, consisting of a contraction path (encoder) and an expanding path (decoder). These components enable U-Net to effectively capture context and achieve precise localization. The encoder path is a traditional stack of convolutional and max pooling layers designed to downsample the input image and extract high-level features. The decoder path, symmetrical to the encoder, uses transposed convolutions to upsample the features and refine the segmentation output. This design allows U-Net to be an end-to-end fully convolutional network (FCN), capable of accepting images of any size without the need for dense layers. Building on previous works in convolutional networks, U-Net's innovative approach has been widely adopted and adapted for various segmentation challenges, showcasing its versatility and robustness in different applications (Ronneberger O., 2015).

3. Methods

3.1 Data Sampling

There are numerous methods for data sampling that could be utilised for our application. However, the chosen sampling technique provides a more unified and effective approach. This method involves filtering through the original 9,301 images using colour information to establish 19 total classes, allowing us to identify and enumerate various artefacts. As a result, we created a frequency distribution table, shown in Figure 1, to illustrate the distribution of these classes.

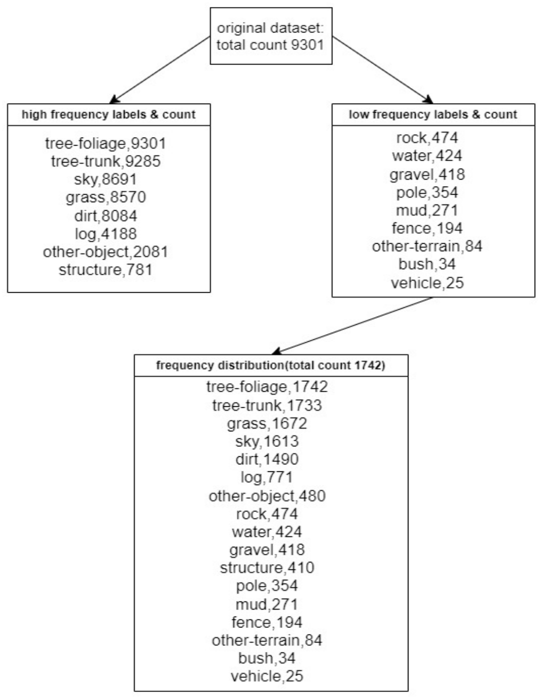


Figure 1. Data types distribution

As per Figure 1, the amount of images containing different colour information and classes vary incredibly with some only appearing a miniscule amount of times (low frequency labels- less than 500 times) to appearing in a majority of images (high frequency labels).This promotes the fact that some artefacts are considered unique and require additional awareness. We identified 9 classes within the low-frequency labels that we focused on primarily due to their uniqueness. These classes, appearing less frequently, require special processing to ensure accurate classification and recognition. By concentrating on these low-frequency items, we can better understand and classify these distinct classes, ultimately enhancing the performance of our segmentation model.

3.2 Data Preprocessing

Grayscale mask preprocessing was utilised by first converting the images to grayscale. By accomplishing this, the images are simplified, and the computational load/time is reduced. It also allowed for focus to be shifted onto intensity variations rather than colour. Each pixel in the grayscale image represents the intensity value, which is essential for the analysis of specific artefacts. Then, the cross-entropy loss function was implemented to the data preprocessing in line with multi-class segmentation and loss function selection. The time spent on training was then further minimised by resizing the images to 128x128x3 using RGB channels and integrating the weight and height image CNN layer. For label preprocessing, each label image was encoded as a specific colour map with corresponding colour palettes. The dimensions of the encoded label images were set as 128x128x1, allowing for the original image to be compared with these colour palettes and categorising it into the 19 classes. After generating the predicted mask, we used the same encoding method to decode it into RGB channels. The PIL package was implemented as an import function to process data to avoid incorrect data preprocessing.

3.3 Hyperparameters

Iterative training was implemented to optimise the normal hyperparameters (learning rate and epoch size). A learning rate of 0.001 was selected to ensure the fastest, most stable approach was utilised. Given the dataset size of 1,742 samples, and after splitting the data into training and validation sets with an 80/20 ratio, the training set was comprised of 1,393 samples. To further optimise the batch normalisation the batch size was chosen such that each batch calculation is efficient and balanced. As such, the batch size was the divisor of the training set to avoid batches with a single remainder sample. This could disrupt the normalisation process and reduce model performance. Considering the need for computational efficiency and the constraints imposed by batch normalisation, a batch size of 10 was selected. This choice ensures that the batches are evenly divided, allowing for optimal utilisation of computational resources and stable training dynamics. The selected batch size strikes a balance between processing time and the requirements of batch normalisation, ensuring that the training process remains both effective and efficient.

3.4 Deeplabv3 + ResNet

Deeplabv3 followed the traditional encoder-decoder structure to complete the semantic segmentation problem (Chen L C., 2018). In the downsampling procedure, it will go through 5 convolution layers to get one result layer. In the decoder procedure, using this generated layer to concatenate the original image after one convolution layer to complete the first upsampling procedure. The following procedure is to get the original size of image to generate a predicted mask.

3.5 U-Net

The U-Net method also utilised a ResNet structure for its encoder, taking advantage of pre-train weights to extract rich feature representations. The encoder path consists of 5 stages, starting with an initial sequence of three convolutional layers followed by progressively deeper convolutional blocks from the ResNet architecture. This down sampling process captures high-level features and reduces spatial dimensions at the same time. Similarly to the encoder structure, the decoder path utilises upsampling layers to progressively restore the spatial dimensions of the feature maps. At each up-sampling stage, the features are concatenated with corresponding encoder features via skip connections, ensuring that spatial information is preserved throughout the network. Specifically, the decoder employs upsampling layers to increase the resolution of the feature maps, followed by convolutional blocks that refine these features.
We introduced a loss function that integrates Dice loss and Cross-Entropy loss to enhance segmentation performance. This approach leverages the benefit from both loss, promotes accurate pixel classification and ensures good overlap between predicted and true segmentation. During the forward pass, the function adjusts the dimensions of the target labels and converts to the appropriate data type. The losses are then computed for given prediction label images and original true labels and return the sum of both losses. This ensures that the model benefits from the pixel-wise accuracy and interspace overlap accuracy from both losses. The class that is responsible for calculating the dice loss ensures that the target labels have the correct dimensions and converts them to a one-hot encoded format to match the shape of the prediction label images. The prediction label images are then processed by using the softmax function to obtain probability distribution. The Dice coefficient is then calculated by measuring the overlap between the prediction label image and the original label image. The Dice loss is derived from this coefficient and encourages the model to maximise overlap.

3.6 Evaluation Metrics

To evaluate the performance of this method, we use the Intersection over Union (IoU) metric, which measures the similarity between the predicted mask and the ground truth. The IoU is calculated using the Jaccard index, illustrated in Figure 2. The mean IoU is derived by averaging the IoU values across all classes.

$$Jaccard = \frac{Intersection(A, B)}{Union(A, B)}$$

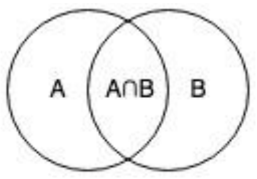


Figure 2. Jaccard Index

In this context, TP (True Positive), FP (False Positive), and FN (False Negative) denote the number of pixels that are correctly predicted, incorrectly predicted, and missed, respectively. C represents the total number of classes, and IoU_c is the IoU for class c (M. D. Sulistiyo et al., 2018).

$$IoU_c = \frac{TP_c}{TP_c + FP_c + FN_c}$$

$$mean\ IoU = \frac{1}{C}\sum_{c} IoU_c$$

3.7 Demo Software Implementation

Due to submission file size limitations, the training checkpoint file has been uploaded to Google Drive. When running the demo software, it will automatically download the best model checkpoint file. Once the download is complete, it can import any images from the WildScenes2D dataset to check the predicted mask. The software interface, implemented using Python's Tkinter, includes a 'Clear' button that allows clearing the current result and importing a new image for prediction.

4. Results

4.1 Deeplabv3 + ResNet

In Figures 4 and 5, the training and validation losses stabilise after 12 epochs. The difference between the training loss and validation loss remains around 0.17, suggesting a minor overfitting issue. As the training loss decreases, the training IoU increases, indicating that the cross-entropy loss effectively captures the necessary information for multi-class segmentation problems.

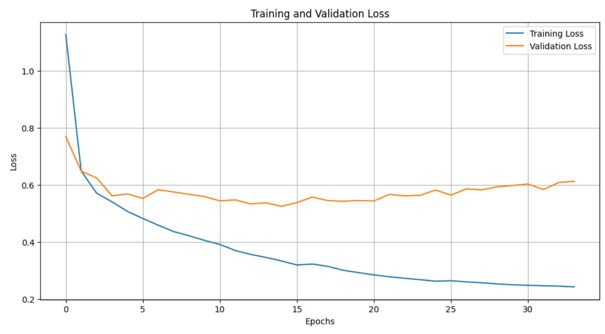


Figure 4. Training and Validation Loss for Multi-Classes

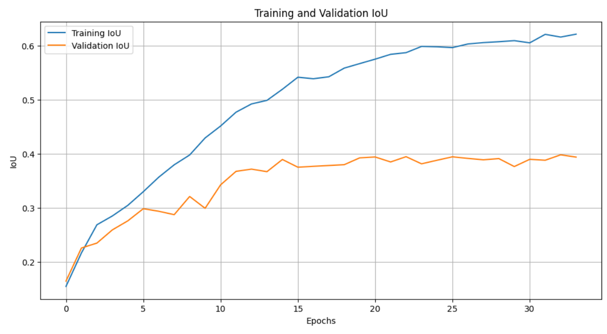
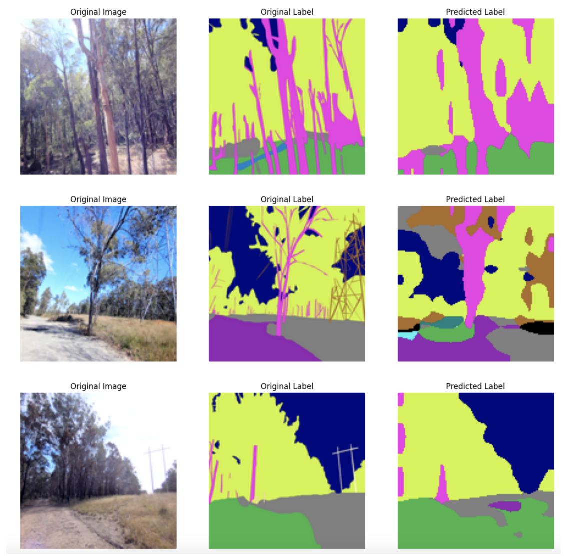


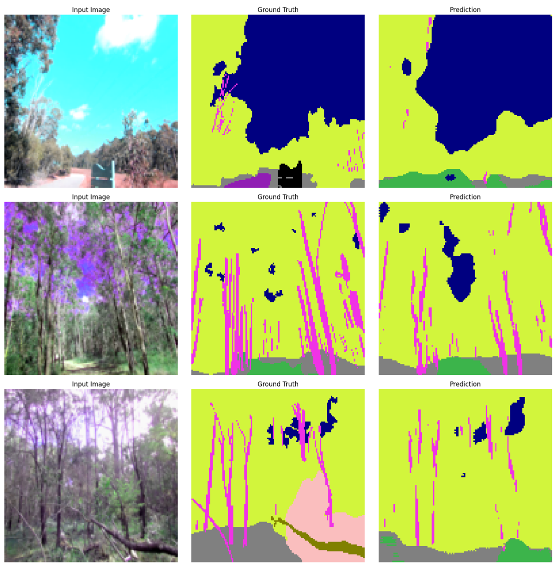
Figure 5. Training and Validation IoU

The effect plot for the DeepLabv3+ResNet model results shows three images from the test loader. The model performs well in predicting large areas accurately; however, thin trees and tiny leaves lead to significant overlaps, indicating challenges in segmenting finer details.



4.2 U-Net

- Pytorch



- Tensorflow

Reference:

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