COMP9517 Group Project

Xiaoqian Hu
z5544297

Dinan Jiang
z5514965

Elaine Zhang
z5135234

Reynor Lou
z5534515

Mianzhi Wu
z5463410

Abstract*—In this project, we will specify how to apply semantic segmentation for sea turtle recognition based on traditional and deep learning-based methods. While the goal of the project is to detect the flipper, carapace and head part of the turtle, significant differences have been found between traditional and deep learning methods. We aim to extend our research so that animal segmentation can be extensively utilized for wildlife research. The performance of the models on the Sea Turtle dataset are measured by the Intersection over Union (IoU) scores. Further discussions of the applicability and effectiveness of different methods will be presented for improving the wildlife semantic segmentation research.

***Keywords*—Semantic Segmentation, K-means clustering, Cunny edgy detection, Computer Vision, Deep learning, DeepLabV3, Swin Transformer, ResNet-50, MobileNetV3**

I. INTRODUCTION

Semantic Segmentation as an frequently-utilized technique, has its own challenges in the capability of segmenting an unknown image into different parts and objects. Without knowing the identity of the objects in a scene, segmentation helps us understand the real world and assist in the improvement of computer vision [1]. In terms of wildlife observation, where people need to recognize the animals from the nearby surroundings, precise segmentation plays an important role in understanding the animals' figures and behaviours. In our project, practical techniques are applied into segmenting the animal parts. More specifically, we develop and evaluate various methods for automated segmentation to identify the sea turtle, including the carapace, head and flipper from the body.

In this study, we mainly summarize our techniques into two categories, which are traditional baseline and deep learning. Traditional unsupervised methods including K-means and Canny edge detection, along with the supervised deep learning models, Swin Transformer, ResNet-50, MobileNetV3 have been employed onto a large amount of sea turtle photo dataset. As required, with the assistance of sea turtle photos from different angles and environments, we need to segment the three specified parts of turtles and evaluate the performance of each model. This amplifies our primary goal of the research, how the traditional approaches are different from complicated deep learning models specifically in terms of the precision. Through experimental test results, we find that significant difference has been found between the unsupervised and supervised learning methods in Intersection of Union (IoU). This observation provides us with insight into the practicability of applying different models into wildlife segmentation and the advantages of utilizing advanced deep learning models to improve the accuracy of complicated real world object segmentation.

II. LITUERATURE REVIEW

A. Research Background

Semantic segmentation, referred to the process of assigning semantic labels to pixels of an image, is widely used in the preprocessing step of system understanding in the object's surrounding scene [2]. Segmentation, as an essential part of computer vision applications, including wildlife observation, human being detection, etc., can extract the useful information of the observed object for further analysis.

In this research, we aim at the animal recognition and semantic segmentation of the sea turtle body into 3 parts, flipper, head and carapace. After applying traditional methods such as K-Means and edge detection, we found that unsupervised methods can achieve some segmentation, but they have difficulty dealing with noisy and complex background environments because they rely on hand-designed features and do not understand the global information of the image. The complexity of the underwater environment poses significant challenges for image segmentation. This implies that accurate segmentation of sea turtles in natural backgrounds is difficult due to lighting conditions, visual interference, and background features of water and sand that compare the surface and color of sea turtles. Therefore, the limitations of these methods motivate us to explore more robust solutions.

For this purpose, the project uses advanced deep learning models which are ResNet-50, MobileNetV3 and Swin Transformer. DeepLabv3 improves segmentation accuracy by combining ResNet-50 as a feature extractor, which can capture detailed features in complex backgrounds. Swin Transformer introduces sliding windows and self-attention mechanisms that reduce the amount of computation and can adapt to processing high-resolution images and complex scenes. Moreover, MobileNetV3 can boost the ability of features to reveal themselves through using a partial residual design [3]. These semantic segmentation methods based on deep learning can not only extract information from the pixel level but also capture spatial relationships in images, so they can effectively identify the detailed structure of turtles in complex environments through their robust structured model.

In order to simplify the manual process of segmenting animal body parts, our research compares the different segmentation methods to look for a balance between computational efficiency and performance accuracy. By providing the in-depth analysis of unsupervised and supervised techniques, we can summarize that under changeable lighting surroundings, deep leaning methods compared to traditional methods will assure a high precision of the segmentation task which is more suitable to our complex real-world research and application.

B. Dataset

The specific task is to identify the head, flippers, and body of a sea turtle from the SeaTurtleID2022 dataset which contains 8,729 photos of 438 sea turtles across 13 years of observation data. The dataset also provides annotations that contain the segmentation mask of the body parts to be identified, providing rich information for individual identification and body part segmentation.

C. Related Technologies

a) K-Means clustering

Initially, the traditional K-Means clustering method has been used as a baseline model for the segmentation task. K-Means works by clustering pixel values into different groups without needing labeled data. Different from the other methods, K-means requires an initialized proper number of clusters, k^2 to generate different cluster result. It is most frequently regarded as a simple and computationally faster model than the complex multi-hierarchy clustering method to divide an image into several number of discrete regions [4]. However, limitations of the capabilities in correctly segmentation are still preserved in the cost of computational efficiency, which will be discussed later.

b) Canny edge detection

Another traditional method that has been applied to our research is Canny edge detection, combined with morphological operations to improve the performance of segmentation tasks. Canny edge detection is used to extract edges in images, and morphological operations help to eliminate noise and optimize edge results. Generally, Canny algorithms consist of four steps, initially smoothing by Gaussian convolution to reduce noise, and then using Sobel operator to detect edge strength and directions. The third step is using edge direction to process non-maximal suppression and eventually eliminating the broken edges [5]. Each step only needs to process each pixel in the image once, which improves the processing speed of the measure.

Moreover, Morphological operations use small structural elements to process the neighborhood around each pixel to assist with the Canny detector. This method has low computational cost and is suitable for real-time processing, but its effectiveness is limited compared to deep learning-based segmentation methods in complex scenes. We still conclude that the limitations of the detector override the efficiency and computation speed of the method.

c) Swin Transformer

Switching to the advanced deep learning models, we first reached a high segmentation accuracy using the Swin Transformer. The Swin Transformer is an improved version of the Vision Transformer (ViT) model, which introduces a sliding window mechanism to segment input images [6]. It divides the images into small windows and applies self-attention to each window. As a result, the time complexity is significantly lower than that of ViT. Swin Transformer is one of the latest models in computer vision and is more computationally efficient than the standard transformer, especially when working with high-resolution images. In this project, we use Swin-T as the backbone and FPN as the segmentation head.

One of the reasons we chose this model is the exceptional performance in handling complex images. We selected the top 30 worst-segmen ted images processed by UNet and found that when dealing with reflections or small objects, it was very challenging. Even humans cannot distinguish them clearly. We finally found Swin transformer can solve this problem well.

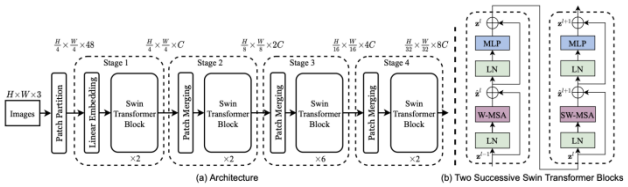

Fig 1. The architecture of Swin ransformer and ViT

d) ResNet-50

Another model combined with DeepLabv3 is ResNet-50 to deal with the problem of turtle segmentation task. DeepLabv3 is a model well defined for semantic segmentation, that is, the classification of each pixel in an image towards understanding its structure. In this case, we refer to the different parts of the turtle from the surrounding environment, such as head, flipper, and turtle. The feature extractor is ResNet-50, which is especially suitable for this task because of its deep residual learning architecture. The trade-off between the two is detailed feature extraction versus understanding of spatial relationships.

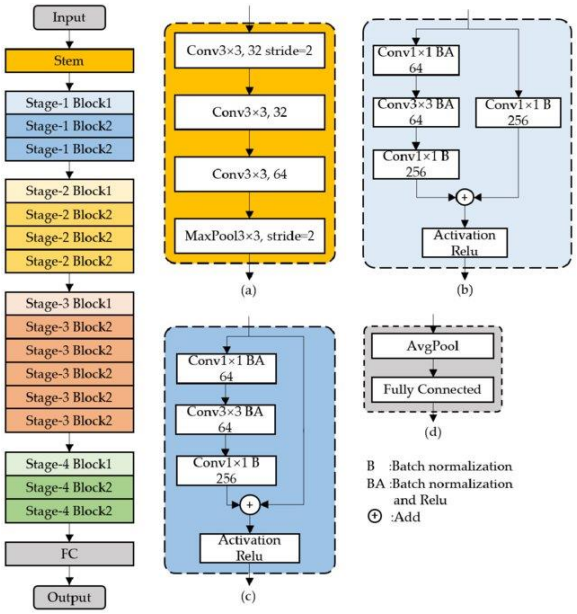

Fig 2. The architecture of ResNet-50

First ResNet-50 processes the image and extracts meaningful features from it through stacked convolutional layers. The residual connections, maintaining the quality of learning also prove to be of improved depths of models, which is essential to avoid problems such as vanishing gradients. DeepLabv3 using the features provided by ResNet-50 to perform segmentation. One of its important components is the Atrous Spatial Pyramid Pooling (ASPP), which helps the model to capture multi-scale information by looking at the image at different levels of detail. This is valuable in distinguishing the turtle's unique textures and shapes since it allows the model to understand both fine details such as the overall shape of the turtle.

The reason we are choosing DeepLabV3 with ResNet-50 is because it can reduce noise in complex nature environments, which contains generally sand and water within the sea amongst other elements. After efficient noise elimination, it achieves more accurate segmentation through focusing on just the essential features and discarding some background details.

e) MobileNetV3-Large

The final option for our experiment is applying another backbone MobileNetV3 onto our DeepLabV3 framework. As discussed earlier, DeepLabV3 dominates in its computational efficiency and performs well in semantic segmentation. Since MobileNet, developed by Howard et al. (2019), is a relatively new and efficient convolutional neural network. We apply model to reduce the number of parameters and ease the computation complexity while maintaining the robustness of feature extraction capability as a CNN model. In our experiment, a U-shaped semantic segmentation network was created to process the segmenting task. Within the MobileNetV3 blocks, there is a width scaling parameter to upscale the network to make it suitable to create networks of different scales [7].

Research compromising MobileNetV3 as its backbones sometimes include the application of other CNN models like Reset. The common advantages of these models are generally exploited from the speed and reduced complexity of computations, which triggers the interest in applying the MobileNetV3 to explore any difference in the performance of ResNet-50 backbone.

MobileNetV3 is also well-known for its lightweight architecture, which makes it a comparatively better choice for larger-scale dataset like sea turtle segmentation. Through applying the MobileNetV3, we are able to figure out the possibility of getting balance between high segmentation accuracy and strictly limited hardware usage, which will give us more chances in applying the techniques into practical animal segmentation and its relevant research.

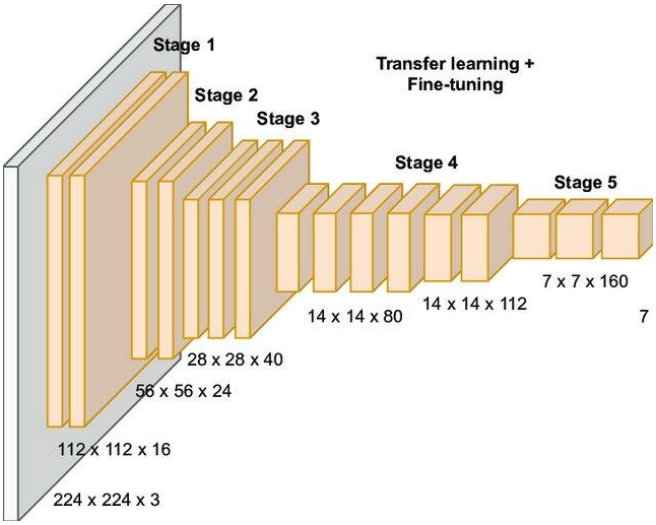

Fig 3. The architecture of MobileNetV3

III. METHOD

Upon the five introduced models in the literature review, we first applied the unsupervised traditional methods of K-means clustering and Canny edge detection to roughly realize the segmentation of sea turtle. The data preprocessing step was straightforward and unexpectedly, the results of traditional methods was not decent. The deep learning models then supplemented the poor performance of previous models. However, data augmentation and longer processing time for training the models were inevitable in exchange with their high performance and accuracy.

A. Data Preprocessing

Data preprocessing is a crucial step before getting into the pretrained model of deep learning where raw data is transformed into a suitable format for model training and validation. Since we are following the open-slitting requirement of the original dataset, data splitting is also important for segmentation task which will be later combined with data augmentation tools.

Our dataset in this project follows Common Objects in Context (COCO) format which is well known standard for object segmentation and detection tasks. To optimize the training and testing the data is split up into three subsets according to metadata_splits.csv:

-Training Set: used to train the model and consists most of the data: 5303 images.

-Validation Set: Adapt hyper-parameters and check model performance during training: 1118 images

-Test Set: the model is used for final validation (testing of model): 2308 images.

The primary goal of the project is to perform semantic segmentation. Therefore, the accurate mask is significant to achieve the objective, the segmentation focuses on three distinct classes, turtle, flipper and head, which respectively correspond to categories 1, 2 and 3 within the original annotation files. After loading all the images and converting them to RGB format, three masks are initialized for each image and each mask corresponds to one of the target classes. The Python library was used to read annotations from images and masks eventually were created and were combined between each class with logical method meaning that overlapping parts in same class were represented correctly.

B. Unsupervised Segmentation Method

a) K-Means clustering

For traditional methods, we started with K-means clustering by applying Gaussian blur to the image before down sampling to smooth it and reduce noise. After extracting the RGB values of each pixel along with its spatial coordinates, we used them as feature vectors. Then all features were normalized to a $[0, 1]$ range to ensure a balanced impact of color and spatial features during clustering.

Then we selected an appropriate number of clusters K to divide the pixels into K groups, and the clustering result was a 2D matrix where each pixel was assigned to a specific cluster. After converting the clustering result matrix into a grayscale image, where each cluster corresponds to a unique grayscale value, the grayscale image represents the initial segmentation result, where each grayscale value indicates a different region, as seen in fig 4.

Fig 4. Gray scale image from K-means clustering

The Canny edge detection algorithm, which will be explained more detailed further, was used to extract the boundaries of the target regions to identify clear contours of the segmented regions. Connected regions were retained as contour, and we assumed that the largest contour represented the primary object (the turtle).

Eventually, the retained contour was filled to create a binary mask as a black and white image viewed in fig 5, and the IoU was around 0.16, without being able to detect the flipper, carapace and head part separately.

Fig 5. Final segmentation of K-means clustering

b) Canny edge detection

We first pre-processed the image through converting the image to a floating-point type so that the average value of each channel can be calculated. Then the average luminance of each channel were created and adjusted to the same grayscale average to balance the colours in the image and correct the colour shift caused by lighting.

In the white-balanced image, the V channel in the HSV colour space was used for histogram equalization to improve the contrast of the image. We converted the image to HSV mode, and applied histogram equalization to the luma channel (V). Also, the image was converted back to BGR colour space to enhance the light and dark details of the image. A Gaussian filter was also utilized here to denoise the image and we then gained our process image as Fig 6.

Fig 6. Processed image for Cunny edge detector

The corresponding annotation data was derived from COCO where we generated binary masks for each part by setting the target regions of the corresponding category in the annotation to 1. To maintain the binary characteristic, all mask values were still truncated to the range of $[0, 1]$ and then resized to a uniform size for subsequent processing and evaluation.

We then computed the gradient of the image using the Sobel operator and used non-maximum suppression to retain local maxima. Dual threshold detections were applied to generate a preliminary edge map. In this project, 75 and 225 were used as the lower and upper thresholds to balance the sensitivity and accuracy of edge detection. After edge detection, since isolated noise points could still be within the image, morphological operations were used to remove these noises to be predicted later.

An edge detection algorithm again, was applied to the preprocessed images to generate predicted masks. The dimensions and resolution of the actual and predicted masks had to be consistent to eliminate evaluation bias. The IoU ratio for each target part was eventually calculated and stored as 0.21, which was slightly higher than the K-means clustering, but further lower than the supervised learning models.

Fig 7. Final segmentation of Canny edge detection

C. *Supervised Segmentation Method*

a) Swin Transformer

Swin transformer, as a vision transformer, utilized the shift window to process high-resolution images. The process started with RGB image input, which was divided into non-overlapping patches. Each patch was considered as a token and its features were formed by concatenating the raw pixel RGB values. In this project, we used Swin-S as backbone with 96 channels whose layer number was {2,2,18,2}. The first step was linear embedding, which transformed the features while keeping the token constant and maintaining a resolution of H/4 × W/4. Secondly, patch merging, which could connect features from adjacent $2x2$ patches and projected to 2C dimension, then down sampling the resolution to $H/8 \times W/8$. The procedures above could be repeated multiple times to achieve the output resolution of $H/32 \times W/32$ feature map.

The following stage was the essential part of Swin Transformer model. Self-attention used uses a kind of shifted window to reduce computation time significantly. Within

local window, self-attention could make it more suitable for high-resolution images. Besides, features within each window interacted with one another, but different windows were isolated from each other. This allowed them to interact with other features and shared window information by shifted window. Compared to the unsupervised model we could see a significant improvement in successful segmentation of the sea turtle body part as well as a high mean IoU of 0.86. A predicted example was presented as follows:

Fig 7. Predicted Segmentation of Swin Informer

b) ResNet

After getting the pre-processed data, the consequent important steps for training models were data augmentation and transformation. The data augmentation step introduced variability of training data and enabled the model to learn invariant features and could reduce the risk of overfitting.

All images and masks were had to be resized first to ensure consistency in input sizes. Also, images were normalized using the mean and standard deviation values $(0.485, 0.456, 0.406)$ and $(0.229, 0.224, 0.225)$, promoting faster convergence during training. Horizontal flapping, rotation and colour jittering were also applied to the training dataset and the data loader was consequently created.

Upon applying the preprocess and augmentation steps, we now generated the dataset for data loader to import separately for training, validation and testing. A custom function **TurtleSegDataset** was also created to enable customize batch size, shuffling and number of worker.

After all the images were loaded, we applied Binary Cross Entropy with logits loss which was well-suited for multi-label segmentation tasks. AdamW was used as the optimizer, which combined Adam optimizer and weight decay for regularization and a learning rate scheduer was also assigned to gradually reduce the learning rate following a cosine curve. After training 25 epochs, we had to validate and test on the remaining data. Undoubtedly, the mean IoU for our ResNet-50 model also reach really high to 0.87 and predicted segmentation was precise and successful.

Fig 8. Predicted Segmentation of ResNet-50

c) MobileNetV3

MobileNetV3, as the backbone of our DeepLabV3 model, would remain the same structure as ResNet-50 did. Data was still needed to be processed and augmented to generate the data loader. Little difference in data augmentation had been found, for instance, using Random Crop and Brightness Contrast.

After setting up the model architecture, which was more lightweighted compared to ResNet-50, we applied the same cross entropy loss function, the optimizer and the learning rate scheduler which were frequently utilized in these deep learning models. 25 Epochs of training were fulfilled to be in accordance with the training epoch number of ResNet-50 and Swin transformer to compare the mean IoU in our later part. Generally, MobileNetV3, as a decent deep learning model, was also able to predict the sea turtle mask with a high IoU of 0.84.

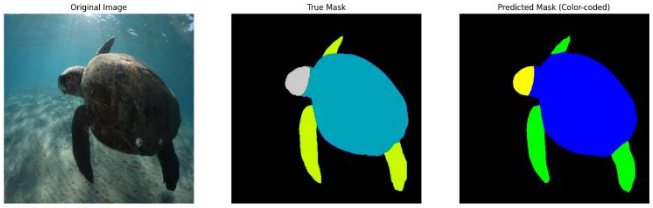

Fig 9. Predicted Segmentation of MobileNetV3

IV. Experimental Results

Basically 5 models have been implemented onto our sea turtle body part segmentation, and our main measurements were relying on the Intersection of Union (IoU), which also is known as the Jaccard Similarity Coefficient (JSC). Through measuring the IoU, we would be able to measure the overlap between the annotated masks and the predeicted segmented regions. Moreover, we calculated the IoU each for the three categories of carapace, turtle and flippers separately, and a mean IoU over the test set for each category representing for the overall performance of the segmentation task. A table of the IoU measurements of 5 methods was presented as below.

	mIoU	Carapace	Turtle	Flipper
Unsupervised methods:				
K-means clustering	0.1587	/	/	/
Canny edge detector	0.1272	0.2163	0.0925	0.0728
Supervised methods:				
Swin transformer	0.8692	0.9306	0.8497	0.8272
ResNet-50	0.8715	0.9715	0.9615	0.8423
MobileNetV3	0.8380	0.9020	0.7961	0.8158

A. Unsupervised Segmentation Method

It was clear that the unsupervised models could not successfully detect the body part of the sea turtle and demonstrated a poor performance of mean IoU around 0.15, , which is significantly lower than the performance of the deep learning methods. Although Canny edge detector improved the method by using Morphological operations could detect the carapace, turtle and flipper part of the turtle, but still resulted in low performance of IoU generally.

B. Supervised Segmentation Method

The Supervised deep learning models all performed well, showing a final mean IoU all over 0.80. Among three methods, DeepLabV3 and ResNet-50 as its backbone had the highest performance, while the MobileNetV3 gave a

comparatively lower predicted mean IoU considering the lightweight nature of the model.

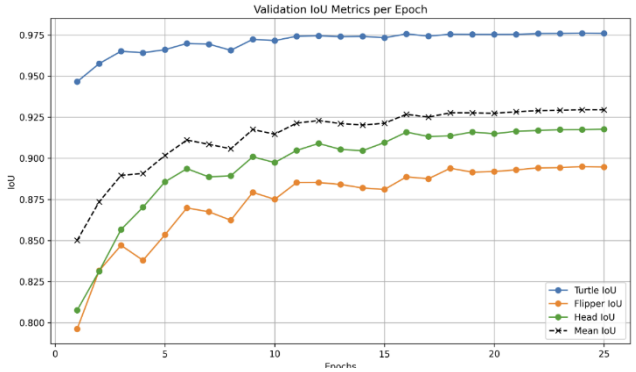

Fig 10. IoU metrics per Epoch on ResNet-50

We derived the performance graph from ResNet-50 which gave us a clearer insight into the performance of the metrics. Here, carapace consistently showed the highest IoU score reaching near 0.975 at epoch 25, while the head score finally landed around 0.83, indicating a lower capability of the model to identify the head part. All three deep learning model had the same characteristics of lower result in head class, which will discussed further in the later part.

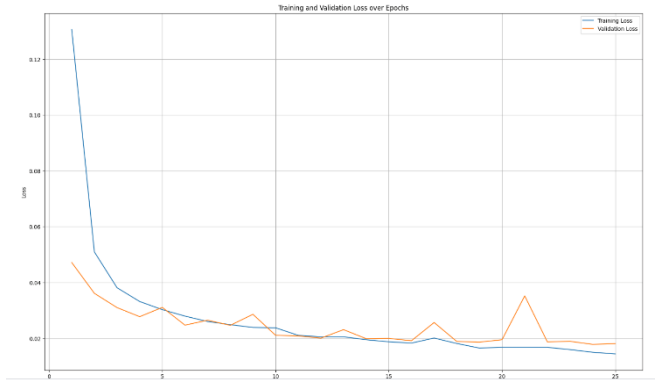

Fig 10. Training and Validation Loss per Epoch on ResNet-50

From the training and validation loss curve, we could summarize that the loss decreased sharply during the first 5 epochs, implying a high speed of the models being able to learn the sea turtle patterns. The remaining of the loss became stable with slight fluctuations in validation loss of head and flipper class. A smooth decrease in the loss curve could be a good sign of effective convergence with slight overfittings, implying that the model was able to perform the well on the validation dataset as well.

V. Discussion

Our discussion will first compare the unsupervised methods and supervised methods through talking about the advantages and disadvantages. Also, an analysis of the poorer predicted performance of head class will also be discussed.

A. Unsupervised Segmentation Method

a) K-means clustering

The low complexity of K-Means brings the advantage of not requiring extensive computational resources for training. This makes it a preferred method for many initial image segmentation tasks, especially in resource-constrained scenarios. Moreover, for low-resolution images, K-Means is highly computationally efficient. With fewer pixels, the clustering process can be completed quickly, and the segmentation results are sufficient to meet the basic needs of low-resolution scenarios.

While this approach is simple, limitations of the model have been found during the process of segmentation. Different from advanced models, K-Means relies purely on pixel features like color and position, so it struggles to distinguish between objects with similar colors, like turtles and the water or sand around them. Also, even small variations in pixel values can lead to fragmented and noisy segmentation results. This implies that K-means focuses only on the cluster's pixels locally, without including the global environment, which makes it difficult in recognizing the overall shape of turtles. Compared to deep learning models, which extract high-level features and are able to understand global context, K-Means is much more limited when it comes to handling complex natural environments.

b) Cunny edge detection

Although the Canny edge detection is also simple and effective, it has significant limitations in complex natural environments. First, both methods are very sensitive to image noise and quality. If the input image contains a lot of noise or illumination changes, the results of Canny edge detection may be unstable and inconsistent. Despite the pre-processing of Gaussian filtering, it is still difficult to eliminate the interference of water waves and shine in underwater environments.

Second, neither Canny edge detection nor K-Means methods understands the global context. The former mainly relies on gradient information, while the latter is based on colour and position. This local perspective makes it difficult to perceive the overall shape of the image, especially for complex images. The performance is reduced when identifying background turtles.

Moreover, the traditional method relies heavily on manually set parameters and is not adaptable enough. Canny's morphological operation needs to adjust the size and shape of the structural element according to the image characteristics, otherwise, it is easy to appear too smooth or inconsistent edges. Therefore, when dealing with changes in lighting, background, and objects, neither Canny edge detection nor K-Means can automatically adapt to the needs of dynamic scenes. These limitations make it difficult to achieve high-precision segmentation in complex natural environments.

B. Supervised Segmentation Method

a) Swin Transformer

One merit of self-attention is global information aggregation. It can focus on global context, making associations with distant targets. However, it also has some drawbacks, one of them is high structural complexity. Unlike traditional CNN models, it cannot be implemented by simply stacking convolutional and pooling layers. The other shortage is high computational cost. It always takes much longer time to train or reason the model. For the small datasets, transformer-based models often do not perform as well as traditional convolutional neural networks.

b) ResNet-50

Conventional segmentation methods like thresholding, edge detection or classical machine learning algorithms such as Support Vector Machine rely heavily on engineered features and intuitive rules. They are also frequently less robust to more complicated backgrounds. On the other hand, DeeplabV3 are based on deep learning to learn feature representation at multiple levels directly from raw pixel data and ResNet-50 is able to extract a high level of detailed and abstract features. Finally, the encoder-decoder architecture further refines segmentation boundaries by fusing low-level spatial information with high-level semantic feature [8]. As a result, the segmentation masks that will be created are more accurate than conventional ones. The model also benefits from the heavy application of data augmentation.

ResNet-50 as the highest performance methods of the five models, limitations were still found in poorer ability to recognize small class like head for wildlife segmentation. However, it is also the same problem encountered by the other two deep learning methods. We will discuss the reason and the potential improvements at the end of this section.

*c) MobileNet*V3

MobileNetV3, though comparatively performed lower than the other two methods, indicating the nature of its lightweight and efficient structure. As it is newly-developed model, it is designed to minimize the computational cost while maintaining the performance. Considering for the balance between efficiency and relatively good predictions, the MobileNetV3 would be suitable with limited computational power and restricted hardware environments. Also, the depth-wise separable convolutions in the structure assist in reducing the number of parameters which enhance the convenience of operation compared to the other standard convolutions.

C. *Reason of inferior performance and future work*

We analysed the 30 worst performing results and found that the models struggled with predicting very small objects. Some of the worse performing graph were showed as below.

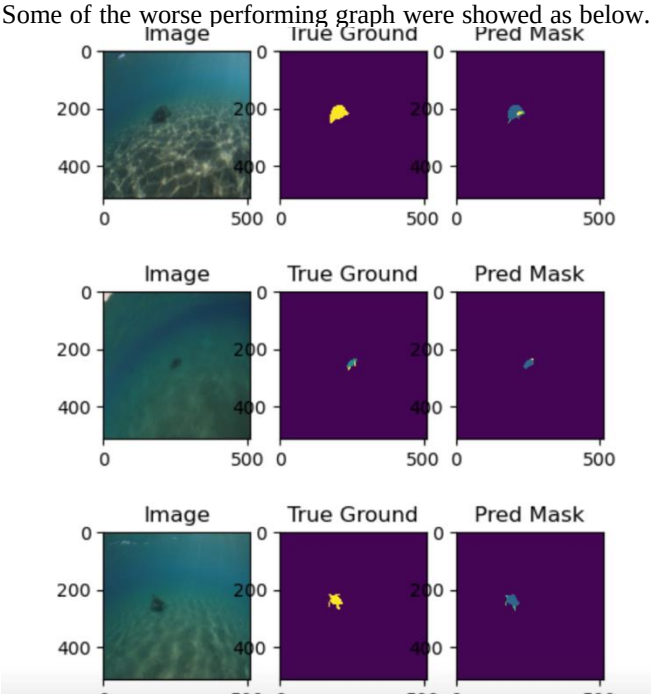

Fig 11. Worse performing images and results from Swin transformer

We could summarize that even if the model achieves high accuracy in this task, it has a challenging time with precisely segmenting small objects of turtle or flipper instances. The reason could be due to the limited resolution that fewer pixels with higher accuracy from a few turtles may mean less spatial information available for segmenting correctly. This lack of detail can hinder the model to detect small objects. Also, the challenging environment, for instance, variability in lighting conditions and the camouflage and shadow which are inevitable nature features when analysing sea turtle, could hinder from the accurate segmentation.

The reason of lower IoU metrics for the head class is partly because of the small size of the head which could be hardly recognized if the turtle appears to be tiny in the original image. Also, smaller turtles could have been blended with their surrounding environment, so it is difficult to separate them, this kind of similarity may result in false detection or missed.

Some enhancements could be implemented for the future research. Through enhancing data quality and quantity, we could implement augmentation techniques that specifically target small objects such as context-specific augmentation, providing the model with more diverse examples and improve its ability to generalize. A hybrid model combination of both the Swin transformer and DeepLabV3 might further improve the performance of Swin transformer with a higher capability of capturing both local and global context for better segmentation. Also, Conditional random fields (CRF) is a bonus to our deep learning models if applied to refine the segmentation boundaries and eliminate the noise.

VI. CONCLUSION

Throughout the 5 different methods applied in our experiment, we are able to perform the task of segmenting the sea turtle in both traditional and deep learning methods. It is unexpected to see a significant low IoU in the traditional methods due to the inability to extract and analyse surrounding global information. In contrast, deep learning methods achieve outstanding semantic segmentation results in various complex background due to their feature learning capability. They not only learn edges and texture feature but also generalize well to unseen complex backgrounds. We can conclude that it is very good at handling noisy background through the integration of advanced network architectures.

Further discussion and comparison of deep learning models have also been presented to indicate a better model choice under certain circumstances considering of either computational constraints or higher performance value or even a balance of in between. Future improvements could be applied by getting more input images to be detected or extending our deep learning models with some specific modules. This research is aimed at implementing the semantic segmentation of sea turtle and we also believe that more techniques and advanced models for segmentation could be widely used in the wildlife research, medical imaging, autonomous driving and other research and daily-usage fields.

REFERENCES

Follow this format

[1] Y. Guo, Y. Liu, T. Georgiou, and M. S. Lew, “A review of semantic segmentation using deep neural networks,” International Journal of

Multimedia Information Retrieval, vol. 7, no. 2. Springer Science and Business Media LLC, pp. 87–93, Nov. 24, 2017. doi: 10.1007/s13735-017-0141-z.

[2] H. Yu et al., “Methods and datasets on semantic segmentation: A review,” Neurocomputing, vol. 304. Elsevier BV, pp. 82–103, Aug. 2018. doi: 10.1016/j.neucom.2018.03.037.

[3] B. S. S. Rao and B. S. Rao, “An Effective WBC Segmentation and Classification Using MobilenetV3–ShufflenetV2 Based Deep Learning Framework,” IEEE Access, vol. 11. Institute of Electrical and Electronics Engineers (IEEE), pp. 27739–27748, 2023. doi: 10.1109/access.2023.3259100.

[4] N. Dhanachandra, K. Manglem, and Y. J. Chanu, “Image Segmentation Using K -means Clustering Algorithm and Subtractive Clustering Algorithm,” Procedia Computer Science, vol. 54. Elsevier BV, pp. 764–771, 2015. doi: 10.1016/j.procs.2015.06.090.

[5] H. G. Kaganami and Z. Beiji, “Region-Based Segmentation versus Edge Detection,” 2009 Fifth International Conference on Intelligent Information Hiding and Multimedia Signal Processing. IEEE, Sep. 2009. doi: 10.1109/iih-msp.2009.13.

[6] Liu, Ze, et al. "Swin transformer: Hierarchical vision transformer using shifted windows." Proceedings of the IEEE/CVF international conference on computer vision. 2021.

[7] V. B. Yesilkaynak, Y. H. Sahin, and G. Unal, “EfficientSeg: An Efficient Semantic Segmentation Network,” 2020, arXiv. doi: 10.48550/ARXIV.2009.06469.

[8] Liu, M., & Yin, H. (2021). Efficient pyramid context encoding and feature embedding for semantic segmentation. Image and Vision Computing, 111, 104195. https://doi.org/10.1016/j.imavis.2021.104195