Development and Comparison of Computer Vision Methods for Sea Turtle Segmentation

Xianghui Jiang
z5468921
University of New South Wales
z5468921@ad.unsw.edu.au

Zihang Cheng
z5502944
University of New South Wales
z5502944@ad.unsw.edu.au

Jinghan Wang
z5286124
University of New South Wales
z5286124@ad.unsw.edu.au

Yitong Li
z5504073
University of New South Wales
z5504073@ad.unsw.edu.au

Xinyi Wu
z5509160
University of New South Wales
z5509160@ad.unsw.edu.au

I. Introduction

The project's goal is to develop and compare various computer vision methods for instance segmentation of sea turtle images, a crucial task for wildlife research, such as population monitoring and behavior analysis. We use the SeaTurtleID2022 dataset, which comprises 8,729 high-resolution photographs of 438 unique sea turtles collected over 13 years. Each image includes annotations such as identity, observation time, and segmentation masks for different body parts, and all images are rescaled to a maximum of 2000 pixels.

Instance segmentation requires models to identify all objects in an image and generate pixel-level masks for each, ensuring accurate segmentation and labeling of different instances of the same category. Specifically, in this project, we will use various deep learning-based computer vision methods to segment the head, flippers, and carapace of each sea turtle. Additionally, we will evaluate the performance of each model using metrics such as Intersection over Union (IoU), Dice coefficient, and accuracy.

II. Literature Review

In the field of computer vision, instance segmentation and semantic segmentation have experienced remarkable development with the advancement of deep learning architectures. These architectural developments have not only improved the performance of visual tasks but also addressed core computer vision challenges including data dependency, model training efficiency, and cross-task transfer learning.

In 2015, Long, Shelhamer, and Darrell proposed Fully Convolutional Networks (FCN) [1], which transformed traditional classification networks into segmentation models by replacing fully connected layers with convolutional layers. Additionally, to address the reduction in image dimensions caused by convolution and pooling operations, upsampling methods were employed to restore the image dimensions.

In the same year, Ronneberger, Fischer, and Brox introduced the U-Net architecture [2], featuring a symmetric encoder-decoder structure. This design proved particularly effective in preserving fine details while maintaining global context. Its ability to perform precise segmentation with limited training data led to its widespread adoption across various domains.

In 2017, Lin et al. proposed the Feature Pyramid Network (FPN) [3] architecture, which addressed the challenge of detecting and segmenting objects at different scales by constructing feature pyramids with high-level semantics. FPN significantly improved small object detection performance through simple network structure modifications without increasing the computational complexity of the original model. The top-down pathway with lateral connections enables the model to maintain both semantic strength and fine details across different scales.

In the same year, He et al. introduced the Mask R-CNN [4] architecture, which extended the Faster R-CNN framework by adding a mask prediction branch to generate binary masks for each object. The architecture also introduced the RoIAlign layer, which preserved spatial information through precise bilinear interpolation, significantly improving instance mask prediction accuracy. This architecture has been widely used for instance segmentation and keypoint detection tasks, becoming one of the most widely applied instance segmentation models.

Chen et al. proposed DeepLabv3 [5], which enhanced segmentation accuracy and multi-scale object processing capabilities through innovative use of atrous convolution and Atrous Spatial Pyramid Pooling (ASPP). This architecture effectively captures multi-scale contextual information while maintaining computational efficiency.

Most recently, the combination of Mask2Former [6] and Swin Transformer [7] represents the latest advancement in instance segmentation. Mask2Former's universal framework flexibly adapts to various tasks including semantic segmentation, instance segmentation, and panoptic segmentation, while Swin Transformer's hierarchical window attention mechanism significantly improves model performance in complex multi-object scenes. Leveraging Swin Transformer's window shifting mechanism, the model can effectively capture image details and contextual information across multiple resolution hierarchies, resulting in clearer and more precise segmentation boundaries.

III. Methods

A. FCN

Fully Convolutional Networks (FCNs) hold a landmark significance in the field of computer vision, especially for image segmentation tasks. Since their introduction by Long et al. in 2014, FCNs have become a foundational architecture in the domain of image segmentation, continually driving innovation and development. The advent of this architecture marked a significant breakthrough in deep learning for pixel-level prediction tasks.

FCNs originated from an innovative modification of traditional convolutional neural networks (CNNs). Typically, the fully connected layers at the end of traditional CNNs output a fixed-dimensional vector, a structure that performs well in tasks like image classification but loses detailed spatial information of the image. This structural limitation fundamentally restricts their utility in pixel-level segmentation tasks where positional information needs to be preserved. FCNs address this limitation ingeniously by substituting fully connected layers with convolutional layers, enabling the network to process input images of any size and produce corresponding dense prediction outputs.

In the specific transformation process within an FCN, suppose there is an original network with a fully connected layer receiving a $7 \times 7 \times 512$ input. This layer can be equivalently transformed into a convolutional layer with a 7×7 kernel. This transformation process reorganizes the weight matrix of the fully connected layer into convolutional kernel parameters, maintaining the original feature extraction functionality while endowing the network with the capacity to handle variable-sized inputs. The transformed convolutional layer can perform sliding window operations over the input feature map, generating independent predictions for each local receptive field, thereby achieving dense prediction across the entire image. As Figure 1 illustrated, FCN-8s adopts VGG-16 as the base network, progressively downsampling a 224×224 input image to a size of 7×7 while increasing the feature channel count from 64 to 4096 [1].

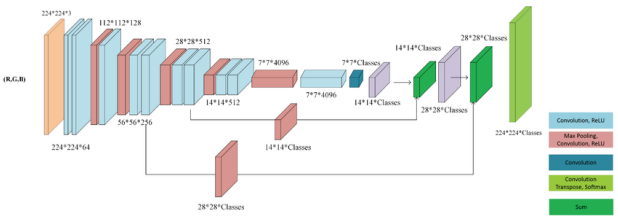

Fig. 1. Fully Convolutional Network - 8s [8]

A key innovation in the FCN architecture is its approach to handling multi-scale features. The network must upsample the low-resolution deep features progressively back to the original input size. This process primarily relies on transposed convolution (also known as deconvolution or upconvolution), spatially expanding the feature map through learnable parameters. As Figure 1 depicted, FCN-8s employs a three-step upsampling strategy: first upsampling the 7×7 feature map to 14×14, then expanding to 28×28, and finally restoring to the original resolution of 224×224. Research indicates that this progressive upsampling strategy is more effective than direct upsampling [9]. To enhance segmentation precision and edge detail, FCN incorporates a skip connection mechanism, fusing features from different depth levels. Specifically, features from shallower layers, pool3 (28×28) and pool4 (14×14), are fused with corresponding scale upsampling results, a multi-scale feature fusion strategy further developed in subsequent works like DeepLab [10].

The network's final output is accomplished through a convolutional layer with a number of output channels corresponding to the target classes, performing dense classification. Each pixel's feature vector is converted into a class probability distribution using the Softmax function. Notably, the feature extraction backbone of FCN offers considerable flexibility and scalability, allowing for the adoption of various deep neural network architectures. While the original FCN utilized the then-prevalent VGG network as the base architecture, the rapid development of deep learning has shown that more modern architectures like ResNet, with superior feature representation capabilities, can provide enhanced feature extraction performance. This modular design philosophy enables FCNs to continually adapt to technological advancements, maintaining their significant position in the field of image segmentation.

B. Unet

The main purpose of U-Net is to segment images using deep learning techniques. Its architecture consists of two parts: the first part is a down-sampling network, similar to a full convolutional network (FCN), which is used to extract features step by step; the second part is an up-sampling network, which is used to recover the resolution of the image step by step. In the up-sampling process, each layer increases the dimension of the feature map to generate higher resolution segmentation results, which is a very important requirement in image segmentation.

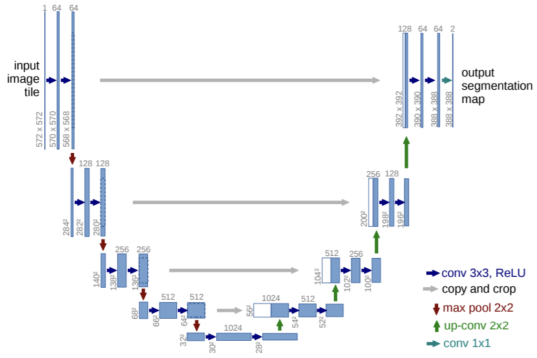

Fig. 2. Unet The architecture of UNet deep learning image segmentation model.

We adopt a U-Net-based segmentation model using ResNet-50 as the encoder backbone.The encoder part of the U-Net

architecture is responsible for extracting features, while the decoder part recovers the image resolution by upsampling layer by layer and generates accurate segmentation results. Specifically, ResNet-50 is pre-trained on the ImageNet dataset, and is therefore able to extract features efficiently and help the model converge quickly.

The output of the model has four channels representing the background, turtle head, flippers, and turtle armor (carapace). We use a combination of loss functions, including cross-entropy loss and Dice loss, to better optimize the segmentation quality. The cross-entropy loss measures the classification accuracy of each pixel.

C. FPN

The introduction of Feature Pyramid Networks (FPN) has marked a significant breakthrough in the fields of object detection and semantic segmentation. Traditional convolutional neural networks exhibit notable limitations in handling multi-scale object detection; however, FPN addresses this issue effectively by constructing a multi-scale feature pyramid structure.

The core innovation of FPN lies in its unique dual-path architecture. The bottom-up path retains the feature extraction capabilities of standard convolutional neural networks, while the top-down path introduces an innovative feature fusion mechanism. This design allows the network to utilize both the fine spatial details of lower layers and the rich semantic information of higher layers.

During the feature extraction phase, FPN employs a backbone network, such as ResNet or VGG, to generate multi-scale feature maps. These feature maps undergo successive downsampling, gradually reducing in spatial resolution while their receptive fields expand. Subsequently, through upsampling operations and lateral connections, FPN achieves effective integration of features across different levels. Specifically, each layer of features is adjusted through 1×1 convolution to ensure compatibility for feature fusion.

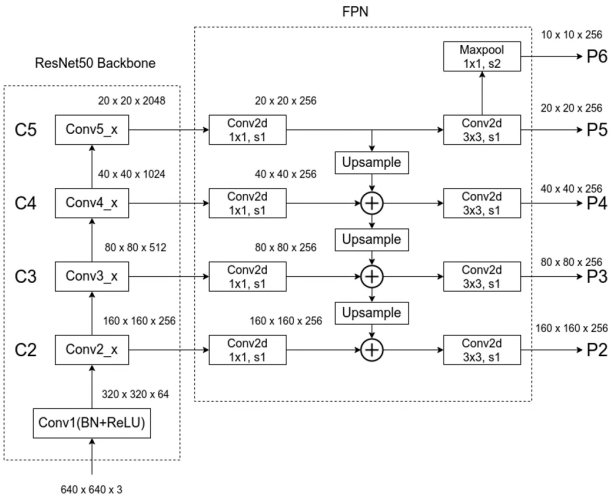

Fig. 3. FPN + Resnet 50 [11]

As Figure 3 illustrated, this employs ResNet50 as the backbone network for experimental validation. ResNet50 includes five main residual block groups (C1-C5), producing feature maps at varying spatial scales. The spatial resolutions of these feature maps are sequentially $320\times320\times64$, $160\times160\times256$, $80\times80\times512$, $40\times40\times1024$, and $20\times20\times2048$. By using 1×1 convolution, the channel counts of all layers are standardized to 256, laying the groundwork for subsequent feature fusion.

In the feature fusion stage, the network adopts a layer-by-layer upsampling strategy. Specifically, the feature map from layer C5 is first upsampled to a resolution of 40×40, and then element-wise added to the processed feature map of layer C4. The merged features are then smoothed through a 3×3 convolution to ensure coherence of the features. This process is propagated downward layer by layer until the feature fusion of all levels is completed. Notably, the network also performs additional downsampling on the C5 features to generate the P6 feature map, further expanding the receptive field range.

This multi-level feature representation offers significant advantages: lower-level features preserve high spatial resolution and rich detail information, facilitating precise localization and detection of small objects; higher-level features contain more abstract semantic information, offering stronger classification capabilities and noise robustness. This complementarity enables FPN to effectively handle object detection tasks across different scales.

In practical applications, FPN demonstrates remarkable adaptability and scalability. For object detection tasks, it can be seamlessly integrated with a Region Proposal Network (RPN) and the Fast R-CNN detection head. For semantic segmentation tasks, the different scale feature maps are upsampled to a uniform resolution. This is typically achieved through bilinear interpolation or transpose convolution. Subsequently, these features can be merged through element-wise addition or feature concatenation. The merged features are then refined through additional convolutional layers, and finally, a 1×1 convolution is used to generate class predictions.

D. Mask R-CNN

Mask R-CNN is a significant deep learning model in the field of computer vision. It is an extension of Faster R-CNN, with an additional branch designed to predict masks for detected objects. Fig. 4 shows that the architecture of Mask R-CNN includes all the components of Faster R-CNN (highlighted in the red box) and introduces a mask prediction branch (highlighted in the yellow box). Mask R-CNN has been widely applied across multiple domains, featuring a simple architecture while delivering remarkable results, capable of generating high-quality pixel-level segmentation masks for detected objects.

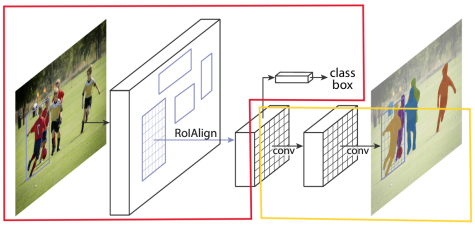

Fig. 4. Mask R-CNN Architecture with the red box indicating Faster R-CNN and the yellow box representing the newly introduced mask prediction branch.

In this project, we focused on instance segmentation of sea turtle images. We used ResNet-50 and Feature Pyramid Network (FPN) as the backbone network for Mask R-CNN. This backbone combines the strengths of ResNet, which introduces residual blocks to make the network deeper and more stable, and FPN, which fuses features of different scales to generate feature maps with rich semantic information.

To enhance the diversity of training data and prevent the model from overfitting to specific orientations or poses, we introduced horizontal flipping as a data augmentation technique. This augmentation increased the robustness of the model by ensuring that it learned to generalize better to various object orientations. During the training process, we encountered gradient explosion issues. To address this, we implemented loss monitoring and gradient clipping when updating weights. Additionally, we reduced the learning rate to improve the stability of the model.

The use of horizontal flipping for data augmentation and gradient clipping for weight updates improved the robustness and stability of the Mask R-CNN model in our instance segmentation task.

E. DeepLabV3

DeepLabV3 is a powerful deep learning model for semantic segmentation tasks, particularly effective for pixel-level classification, it is good at identifying object boundaries in complex backgrounds. The backbone of DeepLabV3could be ResNet-50 or ResNet-101, which is used to extract basic features. Then followed by the ASPP module, which uses multiple parallel void convolution layers and global pooling layers. The voidness of the voidness convolution is gradually increased in the ASPP module. Such Settings can help capture context information at different scales. After the ASPP output, DeepLabV3 adds a 1x1 convolution layer to integrate the feature map into the classification results for each pixel.

Because DeepLabV3 uses void convolution to increase the receptive field, it can better capture objects at different scales. Hollow convolution can also retain more context information without increasing the amount of computation, and is suitable for resolving small details in larger backgrounds, so this model performs well in tasks that require accurate resolution of object boundaries.

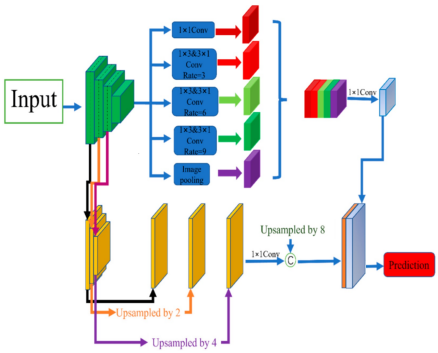

Fig. 5. The DeepLabV3 Plus model.

F. Mask2Former + Swin Transformer

Mask2Former + Swin Transformer is an advanced combination for image segmentation tasks, which leverages the powerful Mask2Former framework with Swin Transformer to achieve impressive results, particularly in instance and semantic segmentation tasks.

Mask2Former is a transformer-based segmentation framework that enjoys many key innovations. For example, it introduces a masked attention mechanism, suggesting the model effectively manages classifications between target and backdrop parts; the mask prediction module provides the masks and class labels of different instances to precisely recognize and separate multiple objects in complex backgrounds; its multi-task segmentation capability is designed as a universal segmentation model. Swin Transformer is a kind of Transformer based on windows with efficient global feature modeling designed for visual tasks. Compared to some traditional vision Transformers like ViTs, Swin Transformer uses a hierarchical, windowed attention mechanism for improved computational efficiency and multi-scale feature extraction.

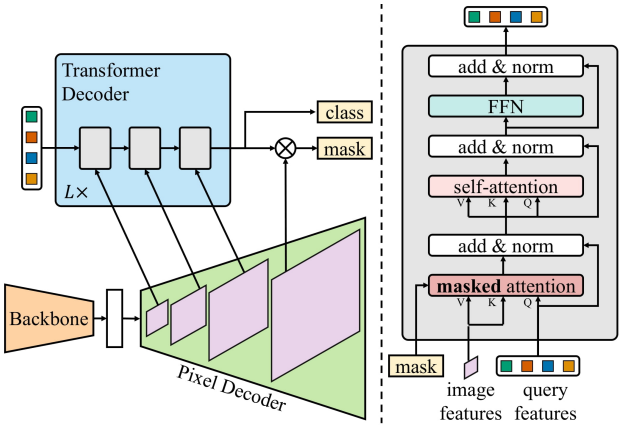

Fig. 6. The Architecture of Mask2Former + Swin Transformer.

The model model builds upon the Mask2Former model and incorporates the advantages of the Swin Transformer. The architecture consists of three main parts, as Figure 6 illustrated: first, the original image is passed through the Swin Transformer as the backbone network to extract feature

maps; then, these feature maps go through a pixel decoder, which processes them to produce high-resolution feature maps; finally, these refined feature maps are passed to a Transformer decoder, where the decoder utilizes class labels and masks for fine-grained feature learning, eventually producing output with class labels and masks. This enables us to obtain optimized high-resolution feature maps, which contribute to more accurate image segmentation.

IV. Experimental Results

A. Data Introduction

The dataset comprises 8,729 high-resolution photographs documenting 438 unique individual sea turtles. All images were systematically collected over an extensive period of 13 years (2010-2022) in Laganas Bay, Zakynthos Island, Greece, with all images scaled to a maximum dimension of 2000 pixels. Following the split_open criteria specified in the metadata_splits file, the dataset is partitioned as shown in Figure 7. The training set contains 5,303 images, representing 60.8% of the total dataset, providing a robust foundation for model learning. The validation set contains 1,118 images (12.8%), offering a balanced subset for model tuning and evaluation. The remaining 2,308 images (26.4%) constitute the test set, providing a comprehensive basis for performance assessment.

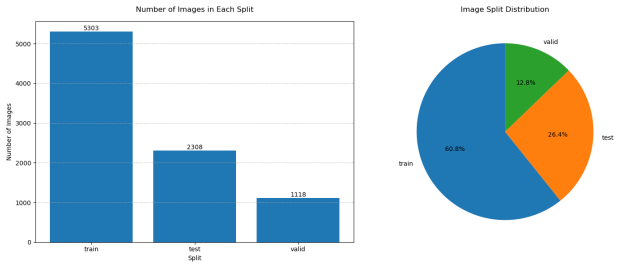

Fig. 7. Distribution of dataset splits

The dataset adopts the widely-used COCO (Common Objects in Context) format for annotations, which are stored in structured JSON files. For the annotation section, we primarily utilize the following key information:

1) Each image is assigned a unique identifier, from which we mainly use:
   - File name and path
   - Image dimensions (height and width)
2) The dataset defines three primary categories:
   - Turtle body
   - Head region
   - Flipper regions

   These three categories will be utilized in our subsequent experiments for classification.
3) For each identified object in an image, instance annotations include:
   - Precise boundary box coordinates (bbox)
   - Binary segmentation masks
   - Category identification

As shown in Figure 8, we present sample images with their corresponding annotations and masks.

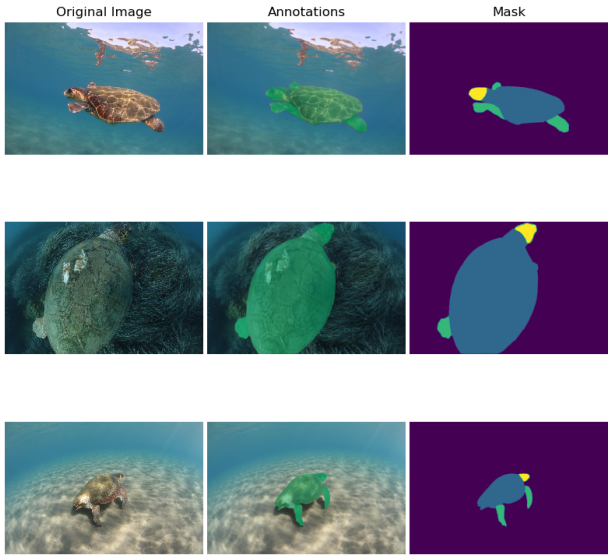

Fig. 8. Examples of image annotations and masks

In addition, in the experiment, it was found that the picture with ID 3793 in the training set sample and the pictures with ID 7772 and 8436 in the test set sample were not actually labeled. We will remove these three samples in the model.

B. Evaluation Metrics

In our experiments, we employ three standard metrics to evaluate the segmentation performance: Intersection over Union (IoU), Dice coefficient, and Accuracy. These metrics provide complementary perspectives on segmentation quality.

IoU, also known as the Jaccard index, measures the overlap between the predicted segmentation mask and the ground truth mask. For each class c, the IoU is calculated as:

$$IoU_c = \frac{|P_c \cap G_c|}{|P_c \cup G_c|}$$

where P_c represents the predicted segmentation mask and G_c represents the ground truth mask for class c. The mean IoU (mIoU) for each category (head, flippers, and turtle) is computed as:

$$mIoU_c = \frac{1}{N} \sum_{i=1}^{N} \frac{|P_c^i \cap G_c^i|}{|P_c^i \cup G_c^i|}$$

where N is the total number of images in the test set, and i denotes the i-th image.

The Dice coefficient, also known as the F1 score for binary cases, provides another measure of overlap between prediction and ground truth. It is calculated as:

$$Dice_c = \frac{2|P_c \cap G_c|}{|P_c| + |G_c|}$$

The mean Dice coefficient for each category is computed as:

$$mDice_c = \frac{1}{N}\sum_{i=1}^{N}\frac{2|P_c^i \cap G_c^i|}{|P_c^i| + |G_c^i|}$$

Pixel-wise accuracy measures the proportion of correctly classified pixels across all classes. For a binary segmentation task (foreground vs. background), it is calculated as:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

where:

TP (True Positives): Pixels correctly classified as the target class

TN (True Negatives): Pixels correctly classified as background

FP (False Positives): Pixels incorrectly classified as the target class

FN (False Negatives): Pixels incorrectly classified as background

For our multi-class scenario, we calculate category-wise accuracy:

$$Accuracy_c = \frac{TP_c + TN_c}{TP_c + TN_c + FP_c + FN_c}$$

where the subscript c denotes the specific category (head, flippers, or turtle). These metrics complement each other in evaluating segmentation performance:

IoU is sensitive to both over- and under-segmentation Dice coefficient gives more weight to true positives Accuracy provides an overall measure of pixel-wise classification correctness

Each metric ranges from 0 to 1, where 1 represents perfect segmentation. In our experiments, we calculate these metrics separately for each of the three categories (head, flippers, and turtle) to provide a comprehensive evaluation of the segmentation performance.

C. Unet Results

The model is trained using the Adam optimizer with an initial learning rate of 0.001. To prevent too large a learning rate from leading to gradient explosion, we also use gradient pruning and dynamic learning rate adjustment (ReduceLROnPlateau) strategies. The learning rate is automatically reduced by half when the validation loss no longer decreases over multiple epochs. The training and validation sets are loaded via DataLoader with a batch size of 32 to ensure the stability and speed of the training process.

In order to evaluate the performance of the model on the test set, we calculated the Intersection over Union (IoU) and the mean Intersection over Union (mIoU) for each category (head, flipper, turtle armor). Specifically, the model was first used to predict the images on the test set, and then the IoU for each category was calculated using the compute iou() function. in this way, we were able to quantify the segmentation effect of the model on each category.

```
Intersection over Union (IoU):
IoU for turtle: 0.728
IoU for flippers: 0.677
IoU for head: 0.603
Overall mean IoU (mIoU): 0.669
```

Fig. 9. Performance metrics across different categories showing IoU values.

In this experiment, we recorded how the loss (Loss) and intersection ratio (IoU) of the model varied with the training epoch on both the training and validation sets. The following figure shows the curves of the model's loss values on the training and validation sets as a function of the number of training rounds (epoch). Overall, the trend of training loss and validation loss shows that the model is converging in most cases, but further attention needs to be paid to the fluctuation of validation loss to avoid overfitting problems.

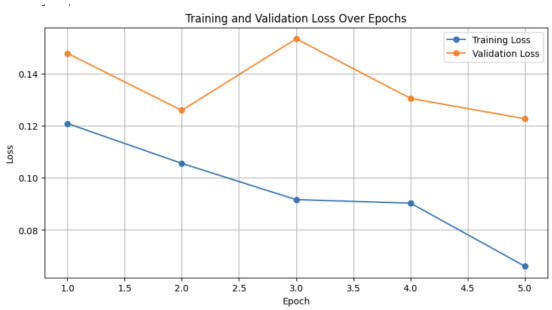

Fig. 10. Shows the curves of the model's loss values on the training and validation.

The following figure shows the variation of IoU with training rounds on the training and validation sets. The fluctuations in the validation IoU may imply that the model's ability to generalize to the validation set is less than optimal at some stages. This suggests that we may need more data augmentation techniques or try other means of regularization to improve the stability of the model.

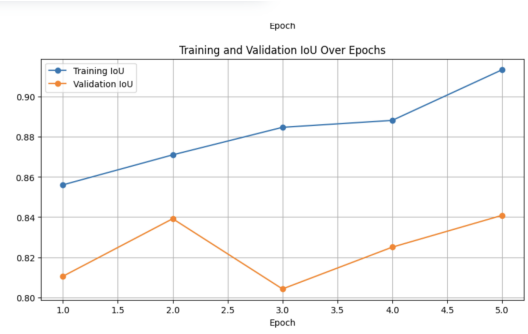

Fig. 11. Shows the variation of IoU with training rounds on the training and validation sets.

Overall, the U-Net model used in this experiment combined with the ResNet-50 encoder achieved better results in the turtle segmentation task, especially in the more obvious turtle armor

region. Although there is still room for improvement in the model's performance on small regions (e.g., flippers), we can further improve the segmentation effect through appropriate data enhancement and hyperparameter tuning. Future work can try to combine the attention mechanism or deeper feature fusion to improve the segmentation accuracy in complex backgrounds.

D. Deeplabv3 Result

The model demonstrates solid performance, with an overall Mean IoU of 0.7563 and Mean Accuracy of 0.8289, indicating reliable segmentation across classes. The model performs well in turtle class and achieves the high accuracy (IoU: 0.7936, Accuracy: 0.8933). However, the pformance is lower for the flipper (IoU: 0.6219, Accuracy: 0.7274) and head (IoU: 0.6220, Accuracy: 0.7009), It may indicate that the model is slightly deficient in accuracy in identifying and segmenting flippers, possibly because of the complexity of Flipper boundaries or shapes. The performance of the model in the segmentation of the head has room for improvement, which may be affected by the details of the head area in the image or the occlusion.

Overall, The segmentation of the model is best in the Turtle class, and slightly worse in the Flipper and Head class. In the future, the segmentation accuracy of the lower class can be improved by improving the model structure

```
turtle:
    Mean IoU:0.7936
    Mean Accuracy:0.8933
flipper:
    Mean IoU:0.6219
    Mean Accuracy:0.7274
head:
    Mean IoU:0.6220
    Mean Accuracy:0.7009

Average Test Metrics:
    Average Test IoU: 0.7563
    Average Test Accuracy:0.8289
```

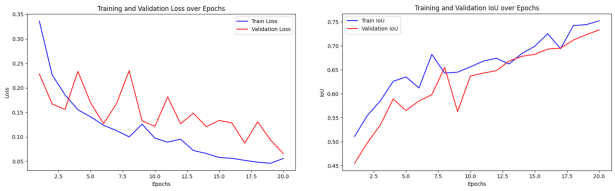

Fig. 12. Evaluation of training and valid Loss,mIou of Deeplabv3

As Figure 14 depicted,The training loss decreased steadily, model performance on the training set was continuously improved. The verification loss fluctuates, but decreases as a whole, indicating that the model has a good generalization effect. The IoU of training and verification gradually increased, and the accuracy of model prediction improved.In the last few rounds, the two lines are close, indicating that the model is not obviously overfitting and the generalization is good.

E. FCN Result

The model demonstrates solid performance, with an overall Mean IoU of 0.8742 and Mean Accuracy of 0.9327, indicating reliable segmentation across classes. Specifically, the turtle class achieves the highest accuracy (IoU: 0.9225, Accuracy: 0.9713), reflecting the model's effectiveness in capturing prominent features. Performance is slightly lower for the flipper (IoU: 0.8351, Accuracy: 0.8940) and head (IoU: 0.8649, Accuracy: 0.9215), suggesting minor challenges in boundary definition for these finer regions. Overall, the model shows strong potential for accurate segmentation, with room for improvement in smaller, detail-rich areas.

```
FCN (Resnet101)
    Mean IoU: 0.8742
    Mean Accuracy: 0.9289

    Mean IoU of turtle: 0.9225
    Mean Accuracy of turtle: 0.9713

    Mean IoU of flipper: 0.8351
    Mean Accuracy of flipper: 0.8940

    Mean IoU of head: 0.8649
    Mean Accuracy of head: 0.9215
```

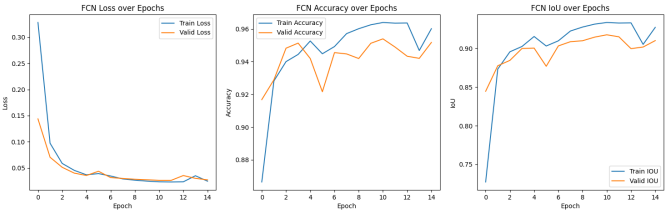

Fig. 13. Evaluation of training and validation mIoU, Loss, Accurancy of FCN(Resnet101)

As Figure 13 depicted, the FCN model demonstrates effective learning with both training and validation loss decreasing and stabilizing after five epochs, indicating good convergence. The AdamW optimizer, paired with adaptive learning rate adjustment from the ReduceLROnPlateau scheduler, maintains a balance between learning speed and generalization. Consistent improvement in accuracy and IoU, with minimal gap between training and validation, suggests strong generalization. Data augmentations, including resizing, flipping, brightness/contrast adjustment, and rotation, further enhance robustness, while normalization ensures stable inputs.

F. FPN Result

The FPN model with a ResNet152 backbone achieves solid segmentation performance, with a Mean IoU of 0.8696 and Mean Accuracy of 0.9258 across all classes, indicating effective overall feature capture and boundary delineation. The

turtle class has the highest metrics (IoU: 0.9202, Accuracy: 0.9653), showing the model's capability to accurately segment the primary object. The flipper and head classes perform slightly lower, with the flipper achieving an IoU of 0.8216 and accuracy of 0.8897, and the head achieving an IoU of 0.8670 and accuracy of 0.9223. These results suggest minor challenges in segmenting the finer details of smaller regions like the flipper. Overall, the FPN (ResNet152) demonstrates reliable segmentation performance, though there is room for improvement in more complex, detailed areas.

```
FPN (Resnet152):
    Mean IoU: 0.8745
    Mean Accuracy: 0.9265

    Mean IoU of turtle: 0.9242
    Mean Accuracy of turtle: 0.9708

    Mean IoU of flipper: 0.8317
    Mean Accuracy of flipper: 0.8899

    Mean IoU of head: 0.8677
    Mean Accuracy of head: 0.9187
```

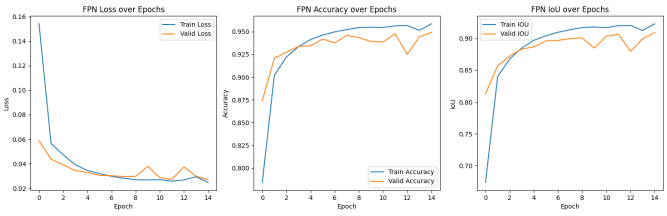

Fig. 14. Evaluation of training and validation mIoU, Loss, Accurancy of FPN(Resnet152)

As Figure 14 depicted, the FPN model with a ResNet152 backbone follows the same preprocessing and optimization setup as the FCN model. It exhibits a stable training process, with loss decreasing and accuracy and IoU improving steadily, indicating effective learning and convergence. This setup enables the FPN model to achieve strong segmentation performance with good generalization.

G. Mask R-CNN Results

The model demonstrates robust performance across all three categories, as shown in Fig. 11. The turtle category achieves the highest performance with metrics approaching 95%, while flipper and head categories maintain strong performance between 85-90%. The model achieves impressive average metrics with an IoU of 87.71%, Dice coefficient of 92.49%, and accuracy of 94.97%, indicating stable and efficient segmentation capabilities.

```
turtle:
    Mean IoU: 0.9455
    Mean Dice Coefficient: 0.9707
    Mean Accuracy: 0.9764

flipper:
    Mean IoU: 0.8495
    Mean Dice Coefficient: 0.9066
    Mean Accuracy: 0.9580
head:
    Mean IoU: 0.8362
    Mean Dice Coefficient: 0.8973
    Mean Accuracy: 0.9146

Average Test Metrics:
    Average Test IoU: 0.8771
    Average Test Dice Coefficient: 0.9249
    Average Test Accuracy: 0.9497
```

The training process exhibits ideal convergence behavior. The loss demonstrates rapid initial descent from 0.40, particularly in the first three epochs, before steadily decreasing to 0.19 by the ninth epoch. This smooth decay indicates stable training progression without overfitting symptoms.

The validation mIoU shows consistent improvement correlating with loss reduction, as depicted in Fig. 15. The metric improved from 0.75 to 0.88, with significant gains in the first five epochs. Maintaining a fixed learning rate of 0.0001 proved effective, facilitating steady optimization until reaching optimal performance in the ninth epoch.

Fig. 15. Evolution of validation mIoU and learning rate across training epochs.

H. Mask2Former + Swin Transformer Result

The model achieved an overall Mean IoU of 0.910 and an accuracy of 0.950, indicating strong performance in segmentation tasks, with effective distinction between categories. Specifically, The turtle class demonstrates the highest accuracy (IoU: 0.9329, Accuracy: 0.9742), indicating the model's strong capability to capture key features of prominent objects. The performance is slightly reduced for the flipper (IoU: 0.8531, Accuracy: 0.9013) and head (IoU: 0.8610, Accuracy: 0.9287), which suggests minor difficulties in delineating boundaries for these more intricate regions. Overall, the model exhibits promising accuracy in segmentation tasks, though there is still scope for enhancing performance in areas with smaller, more detailed structures.

```
Mask2Former + Swin Transformer:
    Mean IoU: 0.9104
    Mean Accuracy: 0.9502

    Mean IoU of turtle: 0.9329
    Mean Accuracy of turtle: 0.9742

    Mean IoU of flipper: 0.8610
    Mean Accuracy of flipper: 0.9287

    Mean IoU of head: 0.8531
    Mean Accuracy of head: 0.9013
```

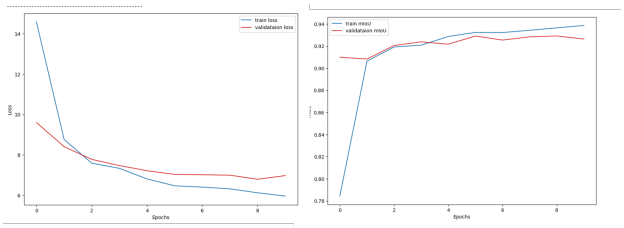

Fig. 16. Evaluation of training and valid Loss,mIou of Mask2Former + Swin Transformer

As Figure 16 shown, in the left plot, both training and validation losses decrease rapidly in the initial epochs, indicating that the model effectively learned relevant features early on. As epochs increase, the loss reduction slows down and stabilizes, reflecting model convergence. It's noteworthy that training loss decreases slightly faster than validation loss, but they converge closely towards the end, indicating that the model does not exhibit significant overfitting. In the right plot, both training and validation IoU show improvement with more epochs, indicating an increase in segmentation accuracy. In the early epochs, the IoU scores rise sharply, showing that the model quickly learns classification boundaries. Then, both training and validation IoU stabilize around 0.92, demonstrating good segmentation performance. Overall, the close alignment between training and validation losses, along with the high IoU for both, suggests that the model has strong generalization ability and segmentation accuracy.

V. Discussion

A. Summary

In this experiment, we performed example segmentation of sea turtle images, comparing six deep learning segmentation models: U-Net, Mask2Former + Swin Transformer,Mask R-CNN, FCN, FPN, and DeepLabV3. each of these models was applied to the SeaTurtleID2022 dataset with the goal of segment the head, flippers, and armor of a sea turtle. The performance, strengths and weaknesses of these models are discussed in detail below, along with a comprehensive analysis of their results.

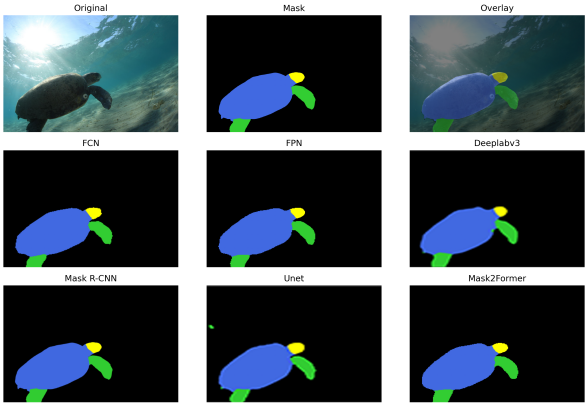

Fig. 17. Figure 132 Sample output

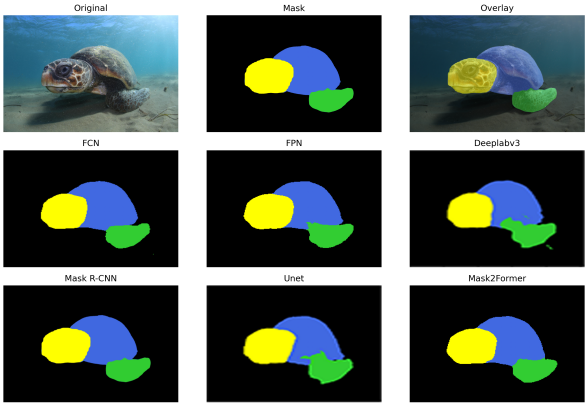

Fig. 18. Figure 5264 Sample output

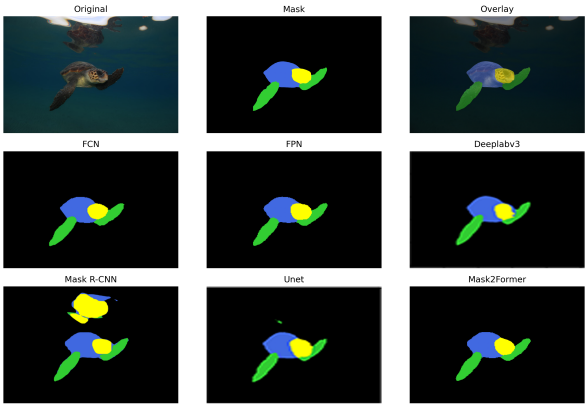

Fig. 19. Figure 7885 Sample output

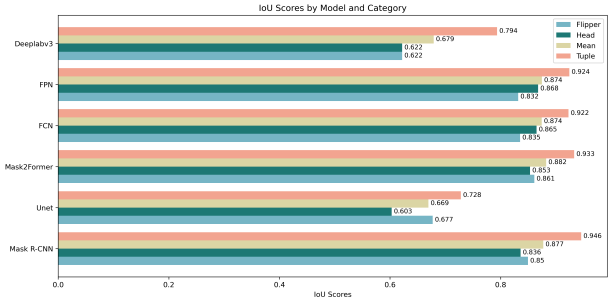

Fig. 20. IoU Scores by Model

B. Architectural and Performance Comparison

As Figure 20 depicted, This investigation conducted a systematic evaluation of six representative deep learning segmentation models. These models exhibited distinct variations in their architectural design and performance metrics.

Mask R-CNN, functioning as a classical two-stage model, achieved the highest score of 0.9455 in tuple feature recognition. This superior performance can be attributed to several key design elements. The incorporation of the RoIAlign layer effectively addressed the spatial information loss associated with feature quantization. The multi-task learning framework enabled simultaneous detection and segmentation processes. Additionally, the Feature Pyramid Network's multi-scale feature fusion mechanism provided robust support for detailed feature extraction.

Mask2Former (SwinT) demonstrated balanced performance capabilities, achieving a high score of 0.861 in flipper recognition tasks. The hierarchical design of SwinTransformer enables feature capture at various scales, exhibiting enhanced adaptability compared to traditional CNN multi-scale processing. The self-attention mechanism facilitates improved handling of long-range dependencies, particularly effective for complex shape features. Furthermore, the introduction of multi-scale deformable attention modules enhanced the model's adaptive capabilities across different feature scales.

FCN (ResNet101) exhibited optimal performance in head feature recognition (0.8746), correlating with its end-to-end fully convolutional design. The deep network structure of ResNet101 provides substantial feature extraction capabilities. The skip connection design effectively mitigates the vanishing gradient problem in deep network training. Moreover, the fully convolutional design maintains spatial structural information from input images, crucial for precise semantic segmentation.

FPN (ResNet152) attained an average score of 0.8696, performing marginally below the top three models. Its top-down feature pyramid structure effectively integrates features across different hierarchical levels. The lateral connections preserve low-level detail features during upsampling processes. However, the absence of dedicated instance-aware mechanisms may constrain its performance in certain precise feature recognition tasks.

Unet and **Deeplabv3** demonstrated relatively lower performance, with average scores of 0.6693 and 0.6791 respectively. Despite Unet's proficiency in medical image segmentation, its encoder-decoder structure shows limitations in complex scenario processing. Deeplabv3, while incorporating dilated convolutions for expanded receptive fields, indicates that mere receptive field enlargement may be insufficient for enhancing detailed feature recognition capabilities.

C. Analysis of Performance Disparities

The performance variations among models manifest in several key aspects: Feature extraction capability: Models with deep backbone networks consistently demonstrate superior performance, indicating the crucial role of deep feature extraction in penguin feature recognition tasks. Multi-scale feature processing: Models incorporating feature pyramid structures or multi-scale feature fusion mechanisms exhibit enhanced adaptability, maintaining robust performance across evaluation metrics. Impact of attention mechanisms: Models implementing attention mechanisms show distinct advantages in processing long-range dependencies, particularly valuable for understanding structural feature relationships.

D. Hardware Resource Constraints

Computational resource limitations prevented certain deep learning models from achieving their full potential. Hardware constraints necessitated reduced input resolutions, potentially impacting feature capture capabilities. Novel models like SegGPT remained untested due to GPU memory constraints.

VI. Conclusion

A. Segmentation Model for Various Project Needs

Different models have their own most suitable applications. Mask R-CNN is generally used for tasks that require object detection and segmentation. Mask2Former + Swin Transformer is more adapt to multi-scale pro-cessing compared to traditional CNN, and it is the most advanced option for multi-scale segmentation involving objects of different sizes. FCN is especially suitable for Fine-grained segmentation tasks requiring high precision. FPN is used for multi-object segmentation, it could preserve low-level detail features during upsampling processes. DeepLabV3 performs well in large-scale semantic segmentation tasks, but enlarging receptive field is not sufficient for enhancing detailed feature recognition. UNet are suitable for simple segmentation with few details, so it performed limited in complex scenario.

B. Improvement

We recommend exploring model ensemble solutions combining Mask R-CNN's precise localization with Mask2Former's global modeling capabilities. Advanced backbone networks like ConvNeXt or novel Transformer architectures warrant investigation for enhanced feature extraction.

Vision Transformer-based segmentation models, including Swin-UNet and DETR, represent promising research directions. These architectures demonstrate potential in capturing both global and detailed information through hierarchical structures and self-attention mechanisms.

REFERENCES

[1] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3431–3440.

[2] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015*, 2015, pp. 234–241.

[3] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2117–2125.

[4] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2961–2969.

[5] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *arXiv preprint arXiv:1706.05587*, 2017.

[6] B. Cheng, I. Misra, A. G. Schwing, A. Kirillov, and R. Girdhar, "Masked-attention mask transformer for universal image segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022, pp. 1290–1299.

[7] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE International Conference on Computer Vision*, 2021, pp. 10 012–10 022.

[8] S. Piramanayagam, E. Saber, W. Schwartzkopf, and F. Koehler, "Supervised classification of multisensor remotely sensed images using a deep learning framework," *Remote Sensing*, vol. 10, p. 1429, 09 2018.

[9] H. Noh, S. Hong, and B. Han, "Learning deconvolution network for semantic segmentation," 2015. [Online]. Available: https://arxiv.org/abs/1505.04366

[10] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," 2018. [Online]. Available: https://arxiv.org/abs/1802.02611

[11] WZMIAOMIAO, "Detailed explanation of the fpn structure," Bilibili, 2021, occurrence time: 6:25. [Online]. Available: https://www.bilibili.com/video/BV1dh411U7D9/?spm_id_from=333.337.search-card.all.click&vd_source=5d1368fc302f3d2a03a943a8c9e58200