VRDL HW3: Instance Segmentation Report

Yi-Hsiang Ho, 111550106

1. Introduction

This task involves performing instance segmentation on RGB medical images using Faster R-CNN [1]. The dataset has 209 images with masks for training and 101 images for testing. The core idea of this work is to freeze part of the network architecture as well as apply multiple data augmentations, e.g., color jitter, random transformation, and non-linear distortion. ResNet-50 [2], which is in the original implementation, is chosen to be the backbone of the model, while most layers are frozen except the last layers of the backbone. GitHub repository is available here.

1.1. Instance Segmentation

Instance segmentation is a computer vision task that identifies and delineates each distinct object in an image at the pixel level. Unlike semantic segmentation, which classifies all objects of the same class as one, instance segmentation differentiates between individual objects of the same type. It combines object detection and segmentation to provide precise object boundaries and counts. Fig. 1 shows an example of instance segmentation.

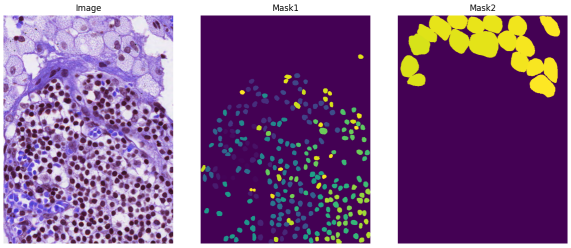

Figure 1. **Visualization of instance segmentation.** The image comes from the provided dataset. The left image is the original image, and the right two images are the different classes in the image. The colors represent different instances of the same class.

1.2. Mask R-CNN

Mask R-CNN [1] is an advanced deep learning model for instance segmentation that extends Faster R-CNN [3] by adding a branch for predicting segmentation masks. While Faster R-CNN detects object bounding boxes and classifies them, Mask R-CNN also generates a pixel-level mask for each detected object. It achieves this by adding a parallel fully convolutional network (FCN) [4] head to predict masks and using RoIAlign instead of RoIPool for better spatial alignment. This makes Mask R-CNN more accurate for precise object delineation.

1.3. Data Augmentation

Data augmentation is a technique used in computer vision to improve model generalization by applying transformations like rotation, cropping, and distortion to training images, especially when the dataset is small. It helps to create variations of the training data, making the model more robust to different conditions and reducing overfitting. Furthermore, Albumentations [5] says that grid distortion and elastic transform, two types of non-linear distortions, are helpful for medical images. Such a method may be suitable for this task.

2. Method

2.1. Data Preprocessing and Augmentation

The training dataset contains 209 images with masks. It is split into training and validation set with a ratio of 8:2.

Some data augmentations are applied to the training set. These include:

- Random horizontal flip
- Random color jitter
- Random affine transformation
- Elastic Transform
- Random Gaussian blur or Gaussian noise

The elastic Transform is a non-linear distortion that randomly deforms the image by applying a random displacement field. It is implemented by the `albumentations` library.

2.2. Model Architecture

The backbone remains the same as the original implementation, which is ResNet-50. I've also tried ResNet-101 and ResNeXt-50 [6], but unlike previous tasks, they do not improve the performance in this task. For the experiment result and analysis, please refer to Sec. 3.

The backbone is frozen except for the last layer since the customized dataset is small compared to the ImageNet dataset, which is used to pretrain the backbone. It may be

hard to train the whole backbone on a small dataset. The last layer is unfrozen to still allow the model to learn the features of the dataset.

The architecture of RPN remains unchanged. After region proposals are generated and refined (by NMS) by RPN, they are projected onto the shared feature map and passed through a RoI Align layer to produce fixed-size feature vectors. These vectors are then fed into two branches: one for object detection and one for instance segmentation. The object detection branch is like Faster R-CNN, while the instance segmentation branch is a fully convolutional network (FCN) that predicts a binary mask for each object.

2.3. Optimization

For optimization, AdamW [7] is used with an initial learning rate of 5e-5. A cosine annealing scheduler [8] dynamically adjusts the learning rate throughout training. The training process is conducted with a batch size of 2 over 50 epochs. The dataset contains roughly 168 training images, 41 validation images, and 101 test images. The best-performing model is selected based on AP50, which is calculated from COCOeval. The model is trained on a single NVIDIA GeForce RTX 4090 GPU in about 2 hours.

2.4. Hyperparameters

The hyperparameter details are listed below:

- Learning rate: 5e-5
- Optimizer: AdamW
- Scheduler: CosineAnnealingLR
- Batch Size: 2
- Epochs: 50

3. Results

In this section, I compare the performance of each component in the model. The details of each method are listed:

- Res50: ResNet-50 pretrained on ImageNet without any tricks but only freeze the whole backbone.
- Aug.: The same setting as Res50, but perform data augmentation excluding elastic transform.
- Elastic: Perform the same data augmentation but including elastic transform.
- Partial: Partial fine-tuning ResNet-50, which means unfreezing the last layer of the ResNet-50.

The results are shown in Tab. 1. The validation AP50 curve and training loss curve are shown in Fig. 2 and Fig. 3, respectively. The training loss is calculated by summing the loss of the object detection and instance segmentation branches. Even though some methods do not surpass Res50 in validation and public testing, they all perform better than Res50 in private testing. Thus, they are all selected in the final model.

Moreover, I find that adding elastic transform may not improve the performance much. It has less than 0.01 improvement in private testing and even worse in validation and public testing. This is different from what Albumentations says. I guess it is because the object in the dataset is small and shape-dependent. The elastic transform may adjust the object too much, or even not proper for this dataset.

Method	Val	Test pub.	Test priv.
Res50	0.4699	0.3640	0.3108
Aug.	0.4555	0.3532	0.3663
Elastic	0.4392	0.3383	0.3671
Partial	**0.4912**	0.3271	0.3608
Partial + Aug.	0.4906	**0.3606**	0.3827
Partial + Elastic	0.4855	0.3391	**0.3853**

Table 1. **The AP50 results of different methods.** "Val" refers to validation AP50. "Test pub." and "Test priv." refer to public and private test set AP50, respectively. Res50 is treated as the baseline. If the value is higher than Res50, it is highlighted in blue, or it will be in red. The highest values in each column are also highlighted in bold.

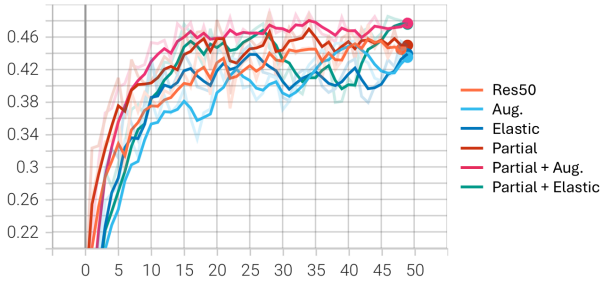

Figure 2. **Validation AP50 curve of different method.**

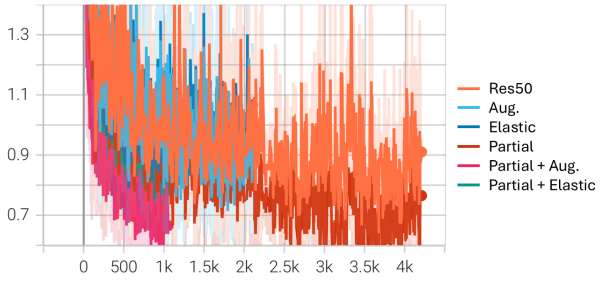

Figure 3. **Training loss curve of different method.**

Other Experiments

The Selection of Backbone

I've conducted experiments with different backbones, including ResNet-50, ResNet-101 and ResNeXt-50. The entire backbone is frozen, treating these models as feature extractors. The results are shown in Fig. 4. While ResNet-101 and ResNeXt-50 are more powerful than ResNet-50, they do not improve the performance in this task. I guess it is because the pretrained weights of ResNet-50 are specially trained on instance segmentation task. While ResNet-101 and ResNeXt-50 are not, they are just trained on their original image classification task. Therefore, utilizing ResNet-50 with well-pretrained weights is more suitable for this task.

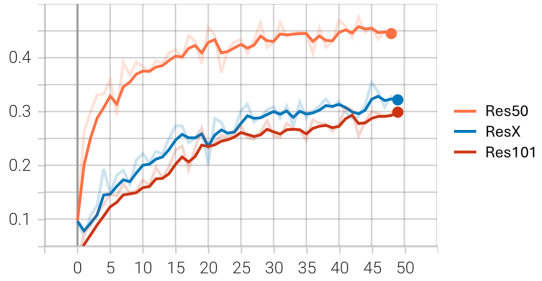

Figure 4. **The AP50 results of different backbones.**

Restart Strategy

In previous homework, restart strategy always improves the performance of the model. However, in this task, it does not always work. I applied it to Mask R-CNN with backbone as ResNet-50 but it turns out a degradation of the performance. The experiment result is shown in Tab. 2. The model is trained with the same hyperparameters and data augmentations, the whole backbone is frozen as well. The only difference is the restart strategy. My guess is that the dataset is too small, once the model is restarted, it may not have enough data to learn again.

Method	Val	Test pub.	Test priv.
Res50	**0.4699**	**0.3640**	0.3108
Res50 w/ restart	0.4411	0.3471	**0.3142**

Table 2. **The AP50 results of applying restart.** The definition of the term is the same as Tab. 1. The highest values in each column are highlighted in bold.

Freezing Model

In this task, only about 180 images are used for training. The backbone is pretrained on ImageNet, which contains about 1.2 million images. Fine-tuneing the whole backbone may not work well due to the small dataset. In my guess, I think leveraging the knowledge from ImageNet is more suitable for feature extraction. Therefore, I test the performance of full fine-tuning, freezing the backbone, and partial fine-tuning (unfreeze the last layer). The experiment result is shown in Tab. 3. It shows that the partial fine-tuning works the best, while full fine-tuning has a significant poor performance.

Method	Val	Test pub.	Test priv.
Full fine-tuning	0.3460	**0.3640**	0.3108
Freezing	0.4699	-	-
Partial fine-tuning	**0.4912**	0.3271	**0.3608**

Table 3. **The results of freezing the model.** The highest values in each column are highlighted in bold. "Freezing" has significantly worse AP50, so I do not evaluate it on the test set.

References

[1] Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. Mask r-cnn. In *ICCV*, 2014. 1

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 1

[3] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NeurIPS*, 2015. 1

[4] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015. 1

[5] Alexander Buslaev, Vladimir I. Iglovikov, Eugene Khvedchenya, Alex Parinov, Mikhail Druzhinin, and Alexandr A. Kalinin. Albumentations: Fast and flexible image augmentations. *Information*, 11(2), 2020. 1

[6] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *CVPR*, 2017. 1

[7] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2017. 2

[8] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. In *ICLR*, 2017. 2