Instance Segmentation using Mask R-CNN

Chu-Chun, Chi

May 3, 2025

1 Introduction

https://github.com/chuchunchi/HW3_Instance_Segmentation

This report summarizes an instance segmentation approach using the Mask R-CNN architecture to detect and segment histopathological cell instances from TIFF images. The core objective is to detect multiple cell types, delineate their masks, and localize them with bounding boxes. The method builds on top of the well-established Mask R-CNN model [2], with adaptations in the input pipeline, training augmentations, and additional experiments to improve segmentation quality.

2 Method

2.1 Dataset and Preprocessing

We use a custom dataset composed of multi-class histopathological images with corresponding binary masks for each class. The dataset loader (see `CellDataset`) reads TIFF images and applies per-class mask extraction using connected component labeling from `skimage`. Degenerate masks (zero-area) are discarded. Each image is normalized to $[0, 1]$ and converted to 3-channel RGB if needed.

2.2 Data Augmentation

We adopt strong augmentations via Albumentations [1], including horizontal/vertical flips, 90-degree rotation, random jittering, and noise:

```
A.Compose([
  A.HorizontalFlip(p=0.5),
  A.VerticalFlip(p=0.5),
  A.RandomRotate90(p=0.7),
  A.Rotate(limit=20, p=0.3),
  A.ColorJitter(0.2, 0.2, 0.2, 0.05, p=0.5),
  A.OneOf([A.GaussNoise(), A.GaussianBlur()], p=0.15),
  ToTensorV2(transpose_mask=True),
])
```

Listing 1: Training Data Augmentation Pipeline

2.3 Model Architecture

We adopt Mask R-CNN [2], a two-stage instance segmentation framework that extends Faster R-CNN by adding a third branch for predicting segmentation masks on each Region of Interest (RoI), in parallel with the existing branch for classification and bounding box regression.

The model components include:

- **Backbone (ResNet-50):** Extracts low- to high-level image features using a deep convolutional network pretrained on ImageNet.
- **Neck (Feature Pyramid Network - FPN [3]):** Combines feature maps from different backbone layers to enable multi-scale object detection. This is crucial for detecting small cells in histopathology images, as it retains fine-grained spatial information from early layers and merges it with deeper semantic features.
- **Region Proposal Network (RPN):** Generates candidate object bounding boxes (called anchors), which are then refined in the second stage.
- **RoIAlign:** Accurately aligns extracted features with each proposed region, preserving spatial alignment required for mask prediction.
- **Heads:**
  - `Box Head:` Performs classification and bounding box regression.
  - `Mask Head:` Predicts a binary mask for each RoI using a small FCN applied to each feature map, yielding one mask per detected instance.

FPN is especially beneficial in medical image analysis, as it enhances the model's sensitivity to small and diverse cell types without incurring significant computational cost.

2.4 Training Setup

- **Optimizer:** SGD [5] or AdamW [6] (with cosine scheduler)
- **Batch size:** 2
- **Epochs:** 40
- **Learning Rate:** 5e-3
- **Validation Split:** 10%
- **Losses:** Classification, bbox regression, and mask loss (total loss as sum)

Mixed precision training [7] and gradient clipping [8] (max_norm=1.0) are enabled to stabilize training.

13	chuchun1231	1	2025-05-02 01:03	279804	313551057	0.3958

Figure 1: Performance on Codabench

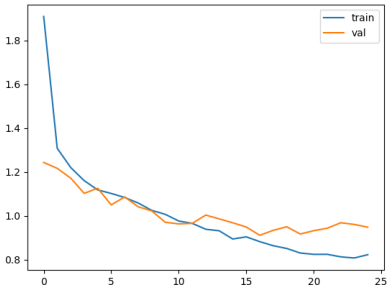

Figure 2: Training vs Validation Loss Curve

3 Results

3.1 Inference Results

Inference is performed on test TIFF images, and results are exported in COCO-style [4] JSON with RLE mask encoding. Performance snapshot:

- Training Loss: 0.8720
- Validation Loss: 0.9286
- Test Score: 0.3958

The performance on Codabench is shown in Figure 1 and the training curve for 25 epochs is shown in Figure 2

3.2 Prediction Format Validation

To ensure the correctness of the final output format, we used a validation script to check the structure and integrity of the predictions:

```
python src/validate_format.py --pred test-results.json
```

Listing 2: Validation Script Execution

The script confirmed that all predictions are valid and correctly structured for submission. Below are the key statistics from the validation output:

- **Total predictions:** 6843
- **Unique image IDs:** 101

- **Unique category IDs:** [1, 2, 3, 4]
- **Score range:** [0.050, 0.997]

This step confirms both semantic correctness (class consistency) and syntactic compliance (COCO-style JSON), ensuring that the test output meets evaluation requirements.

3.3 Visualization of Predictions

To qualitatively evaluate the model, we implemented a visualization script that overlays predicted masks and bounding boxes on the original TIFF images. This allows intuitive inspection of segmentation accuracy, instance separation, and class correctness.

We assign distinct colors to different categories and use alpha blending to highlight masks without obscuring the original image. Bounding boxes and category labels are rendered for each instance.

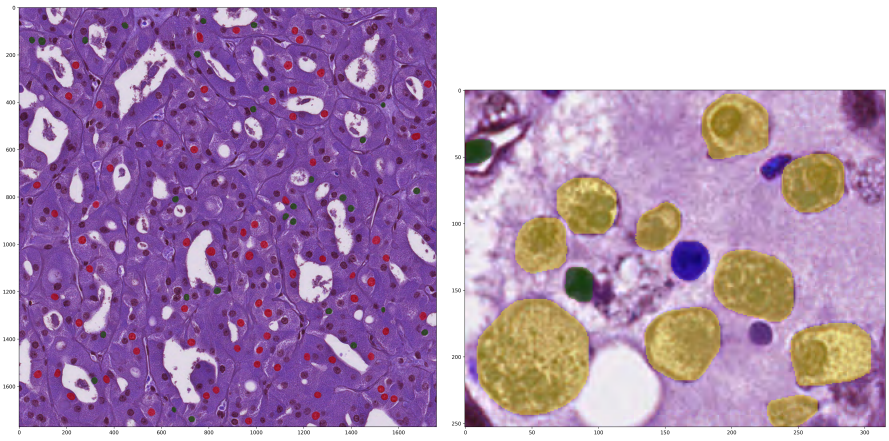

Figure 3: Examples of predicted instance masks and bounding boxes overlaid on input images

These visualizations demonstrate the model's ability to accurately delineate individual cell instances, even in cluttered regions.

4 Additional Experiments

We conducted two experiments to explore ways to enhance model performance, focusing on architecture and optimization choices.

4.1 Experiment 1: Model Selection – `maskrcnn_resnet50_fpn` vs `maskrcnn_resnet50_fpn_v2`

Hypothesis: The v2 variant, introduced in Torchvision 0.13+, includes improvements such as updated normalization layers and better backbone tuning. We hypothesize that it may lead to better mask quality and faster convergence.

Setup: Both models were trained for 25 epochs with identical augmentation, batch size, learning rate, and optimizer (AdamW). Loss curves and final test performance were recorded.

Results: See Table 1.

4.2 Experiment 2: Optimizer Comparison – SGD vs AdamW

Hypothesis: Adaptive optimizers like AdamW [6] may provide better performance in instance segmentation due to stable convergence and effective regularization. However, SGD with momentum has been shown to generalize better in many detection tasks.

Setup: Using `maskrcnn_resnet50_fpn`, we trained two versions of the model for 25 epochs: one using SGD and the other using AdamW. The scheduler (CosineAnnealingLR) and all other settings were kept constant.

Results: See Table 1.

4.3 Implications

From the experimental results in Table 1, we observe the following:

- The `maskrcnn_resnet50_fpn_v2` model with AdamW achieves the lowest training loss (0.8515), indicating improved fitting during training. However, its validation loss (0.9696) is slightly worse than that of the v1 + AdamW variant (0.9286), suggesting potential overfitting or less stable generalization despite architectural improvements in the v2 variant.
- In terms of test prediction score (AP50), `maskrcnn_resnet50_fpn (AdamW)` outperforms both SGD and FPNv2 configurations, reaching an AP50 of 0.40. This supports the hypothesis that adaptive optimizers like AdamW offer more stable convergence and better performance for instance segmentation tasks, especially on medical images with high inter-class imbalance and fine-grained structure.
- The original FPN model (`maskrcnn_resnet50_fpn`) with SGD achieves the highest validation loss and the lowest AP50 score (0.36), further indicating that the AdamW optimizer is generally more effective in this setting.

These findings suggest that while newer architectural variants such as FPNv2 can improve training dynamics, their real-world performance gains may be limited without careful tuning or regularization. Meanwhile, optimizer choice plays

a more significant role in both convergence and generalization in this instance segmentation task.

Table 1: Training and evaluation metrics for different model and optimizer choices (25 epochs)

Model Variant	**Optimizer**	**Final Train Loss**	**Final Val Loss**
maskrcnn_resnet50_fpn	SGD	0.9297	0.9957
maskrcnn_resnet50_fpn	AdamW	0.8720	0.9286
maskrcnn_resnet50_fpn_v2	AdamW	0.8515	0.9696

Model Variant	**Test Prediction Score (AP50)**
maskrcnn_resnet50_fpn (SGD)	0.36
maskrcnn_resnet50_fpn (AdamW)	0.40
maskrcnn_resnet50_fpn_v2 (AdamW)	0.36

References

[1] Alexander Buslaev, Vladimir I Iglovikov, Eugene Khvedchenya, Alex Parinov, Mikhail Druzhinin, and Alexander A Kalinin. Albumentations: Fast and flexible image augmentations. *Information*, 11(2):125, 2020.

[2] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[3] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017.

[4] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. *European conference on computer vision*, pages 740–755, 2014.

[5] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations (ICLR)*, 2017.

[6] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019.

[7] Paulius Micikevicius, Sharan Narang, Jonah Alben, Greg Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, et al. Mixed precision training. In *International Conference on Learning Representations*, 2018.

[8] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *International Conference on Machine Learning*, pages 1310–1318, 2013.