Homework 3:
Instance Segmentation with Mask R-CNN on Cell Dataset

Marie Picquet
Student ID: 313551818
National Yang Ming Chiao Tung University
Selected Topics in Visual Recognition
using Deep Learning (535521)

May 7, 2025

1 Introduction

The goal of this assignment is to develop an instance segmentation model using a convolutional neural network (CNN) capable of detecting and segmenting individual cells in microscopic images. The Mask R-CNN framework, equipped with a ResNet-50 backbone, was selected for its ability to localize objects and predict segmentation masks at the instance level [1].

Model training was conducted on Google Colab using GPU acceleration. Data preprocessing and postprocessing (RLE encoding) were implemented to meet the required format.

The complete code for this assignment is available at the following GitHub repository: `https://github.com/mariep00/535521-HW3.git`.

2 Method

Dataset

The provided dataset consists of 209 training/validation images and 101 test images, each showing cells of four distinct types. Ground-truth masks are provided per cell type in '.tif' format. Unique pixel values represent individual cell instances.

Image type	RGB (microscopy)
Segmentation type	Instance-level
Cell classes	4 types (class1 to class4)
Training/Validation set	209 labeled images
Test set	101 unlabeled images
Format	TIFF images + masks (one per class)

Table 1: Dataset summary

An example from the dataset is shown in Figure 1. The left side displays an RGB microscopy image, while the right side shows the corresponding instance segmentation mask. Each unique color in the mask represents an individual cell instance, and separate masks are provided for each of the four cell types.

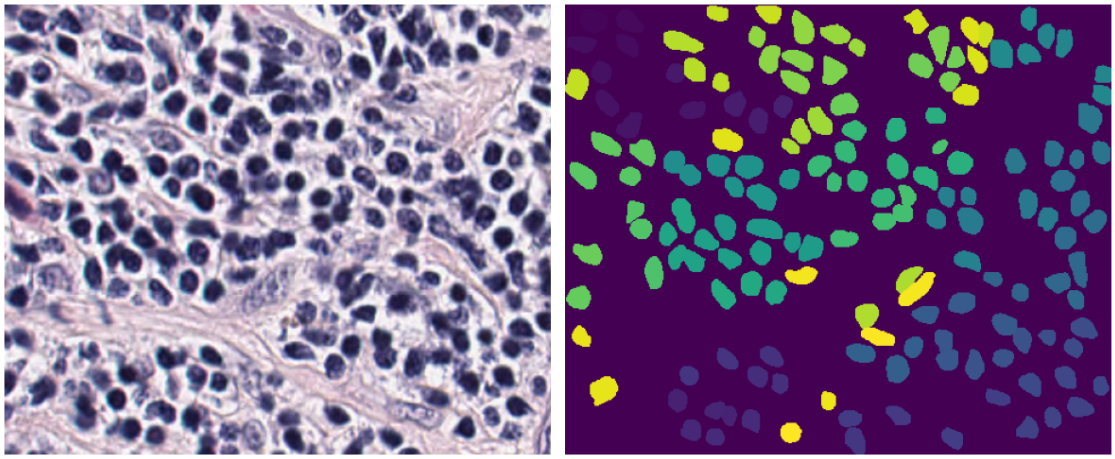

Figure 1: Example of an RGB microscopy image (left) and its corresponding instance segmentation mask (right). Each color represents a unique cell instance.

Data Preprocessing

Each '.tif' mask was parsed and encoded to distinguish individual instances per class using unique pixel values. Data was converted into the COCO format for training and submission.

Data Augmentation

Initial experiments incorporated several common augmentation strategies such as `RandomHorizontalFlip`, `ColorJitter`, and `RandomRotation` ($\pm 15°$), with the goal of improving model generalization.

However, results showed a loss in validation performance when these augmentations were used. A likely explanation is that the small dataset size and already low variation in cell morphology made the augmentations excessive or misaligned with the test distribution. Consequently, augmentation was excluded in the final training pipeline.

Further experiments are needed to carefully evaluate more targeted or conservative augmentation strategies tailored to medical image characteristics.

Model Backbone

The Mask R-CNN architecture was employed with a ResNet-50 backbone and Feature Pyramid Network (FPN). The model consists of:

- **Backbone:** ResNet-50 with pretrained ImageNet weights
- **Neck:** Feature Pyramid Network (FPN)
- **Heads:** Region Proposal Network (RPN), bounding box head, and segmentation mask head

The architecture is illustrated in Figure 2.

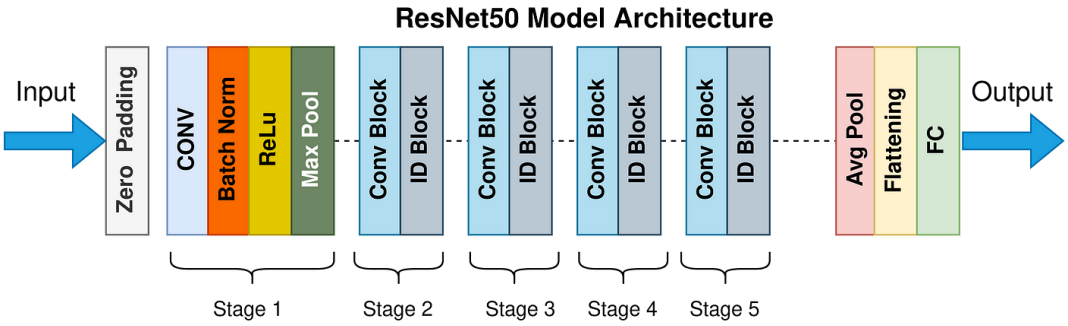

Figure 2: ResNet-50 backbone used within the Mask R-CNN architecture. The full instance segmentation model also includes a Feature Pyramid Network (FPN), Region Proposal Network (RPN), and dedicated heads for bounding box and mask prediction.

Training Hyperparameters

Optimizer

The model was trained using Stochastic Gradient Descent (SGD) with momentum = 0.9 and weight decay = 0.0005.

Learning Rate

A learning rate of 0.005 was initially used during training. However, to stabilize performance and avoid overfitting, a `ReduceLROnPlateau` scheduler was applied, which decreases the learning rate when the validation AP plateaued. This helped the model converge more smoothly by adapting the learning rate based on actual performance [4].

Batch Size

A batch size of 4 was selected to balance GPU memory constraints and convergence stability.

Epochs

The model was trained for 20 epochs. Best model checkpoints were saved.

Loss Function

Mask R-CNN uses a multi-task loss function combining classification loss, bounding box regression loss, and mask loss:

$$\mathcal{L} = \mathcal{L}_{cls} + \mathcal{L}_{box} + \mathcal{L}_{mask}$$

Cross-entropy loss is used for classification and mask predictions. Smooth L1 loss is used for bounding box regression.

3 Results

Figure 3 shows the training loss over 10 epochs, computed using only the training dataset. The loss decreases consistently, suggesting that the model is effectively learning from the data. However, this

plot does not include validation loss, so it does not provide insight into the model's generalization capability or overfitting behavior.

Despite this, model validation was carried out indirectly through submission to the Codabench platform, where performance is evaluated on a hidden test set. Using Average Precision (AP) at an IoU threshold of 0.5 (AP50), the final submission achieved an AP50 of **31%** on the private leaderboard. This result reflects the model's instance segmentation performance across all four cell types.

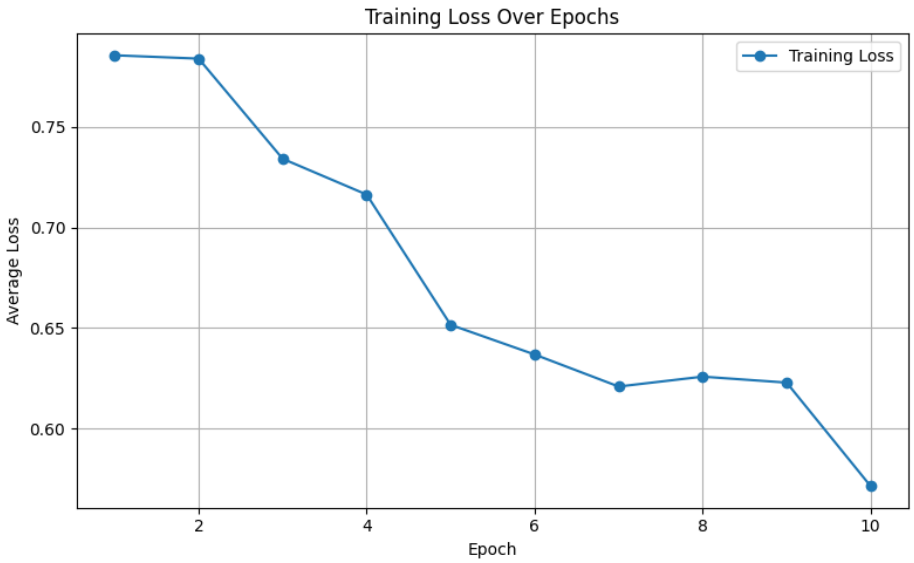

Figure 3: Training loss over epochs, computed on the training dataset only.

We also visually inspected the predictions and found that most cell instances were well-localized, with smooth and accurate mask boundaries.

References

[1] He, K., Gkioxari, G., Dollár, P., and Girshick, R. (2017). *Mask R-CNN*. Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2017.

[2] He, K., Zhang, X., Ren, S., and Sun, J. (2015). *Deep Residual Learning for Image Recognition*. arXiv:1512.03385.

[3] COCO Dataset API. `https://github.com/cocodataset/cocoapi`

[4] PyTorch Documentation. *ReduceLROnPlateau*. `https://pytorch.org/docs/stable/generated/torch.optim.lr_scheduler.ReduceLROnPlateau.html`