Image Segmentation Assignment

Mani Kumar R

April 16, 2025

Github: `https://github.com/Manikumarksr/Multi-class-image-segmentation`

wandb: `https://wandb.ai/ksrmanikumar-indian-institute-of-science/image_segmentation?nw=nwuserksrmanikumar`

1 Task 1: Dataset Preparation using Python

The dataset was sourced from the COCO (Common Objects in Context) 2014 validation set, which originally contains approximately 41,000 images and their annotations. The following resources were downloaded and utilized:

1. Images: `http://images.cocodataset.org/zips/val2014.zip`
2. Annotations: `http://images.cocodataset.org/annotations/annotations_trainval2014.zip`

From this dataset, I filtered images and annotations to include only the following five classes as specified:

- 1: Person
- 2: Bicycle
- 3: Car
- 4: Motorcycle
- 5: Airplane

This filtering process resulted in a subset of 8,000 relevant images. The image archive was extracted to a directory named val2014. I developed a Python script, **process_data.py**, to process the COCO annotations and generate multi-label segmentation masks. These masks assign pixel values from 0 to 5 indicating background, person, bicycle, car, motorcycle and airplane respectively.

The script parsed the JSON annotation file, identified instances of the specified classes, and combined their segmentation masks into a single multi-label mask per image, ensuring overlapping regions were handled by prioritizing higher class indices (though overlaps were rare in this subset). The processed images and masks are stored in the "**coco_val_subset**" directory.

Finally, I used another script, **split_data.py**, to split the dataset into training, validation, and test sets in an **8:1:1** ratio (6,400 training, 800 validation, and 800 test images).

2 Task 2: Train an Image Segmentation Model

Architecture Chosen: I selected **DeepLabV3 with a MobileNetV3 backbone** for this multi-label segmentation task.

Reasons:

1. Efficiency: MobileNetV3 is lightweight, making it suitable for the 6-hour computational constraint on a P100 GPU (provided via Kaggle).
2. Performance: It offers comparable segmentation performance to heavier models like UNet or DeepLabV3 with ResNet50, but with lower computational overhead, enabling faster training and potential deployment on edge devices.

2.1 Training Configuration:

1. **Loss Function: Cross-Entropy Loss**, adapted for multi-label segmentation by treating each pixel's class prediction as an independent classification task.
2. **Learning Rate: 0.0001** is chosen (Based on previous experience).
3. **Epochs: 30**, sufficient to observe loss convergence without exceeding time constraints.
4. **Batch Size: 4** with image size of **256x 256**.
5. **Optimizer: Adam**, with default parameters, for its robustness in segmentation tasks.
6. **Computational Resources:** Trained on a **Kaggle P100 GPU**, taking approximately 3.5 hours.
7. To improve the model's generalization and robustness, augmentations like **HorizontalFlip**, **VerticalFlip**, **Rotations**, and **Resize(256, 256)** were applied to the training images and masks.

2.2 Training Metrics:

1. Monitored using a public WandB dashboard: `https://wandb.ai/ksrmanikumar-indian-institute` `image_segmentation?nw=nwuserksrmanikumar`.

2. Key metrics included Dice Score, IoU, pixel accuracy, and class-wise pixel accuracy, logged for both training and validation sets across epochs.

The training process effectively reduced both training and validation losses, as visualized in the WandB dashboard, indicating successful learning.

2.3 Generalization on test data

The best-performing model (selected based on validation IoU after 30 epochs) was evaluated on the 800-image test set. Results are as follows:

1. **Overall Test IoU:** 0.524.
2. **Test IoU per Class:**
   (a) Background: 0.91
   (b) Person: 0.70
   (c) Bicycle: 0.26
   (d) Car: 0.41
   (e) Motorcycle: 0.42
   (f) Airplane: 0.43

Figure 1 below illustrates model predictions on test images, showing input images, ground truth masks, and predicted masks. The model excels at segmenting larger, frequent classes (e.g., person, car) but struggles with smaller, underrepresented classes (e.g., bicycle), as expected given the dataset's imbalance.

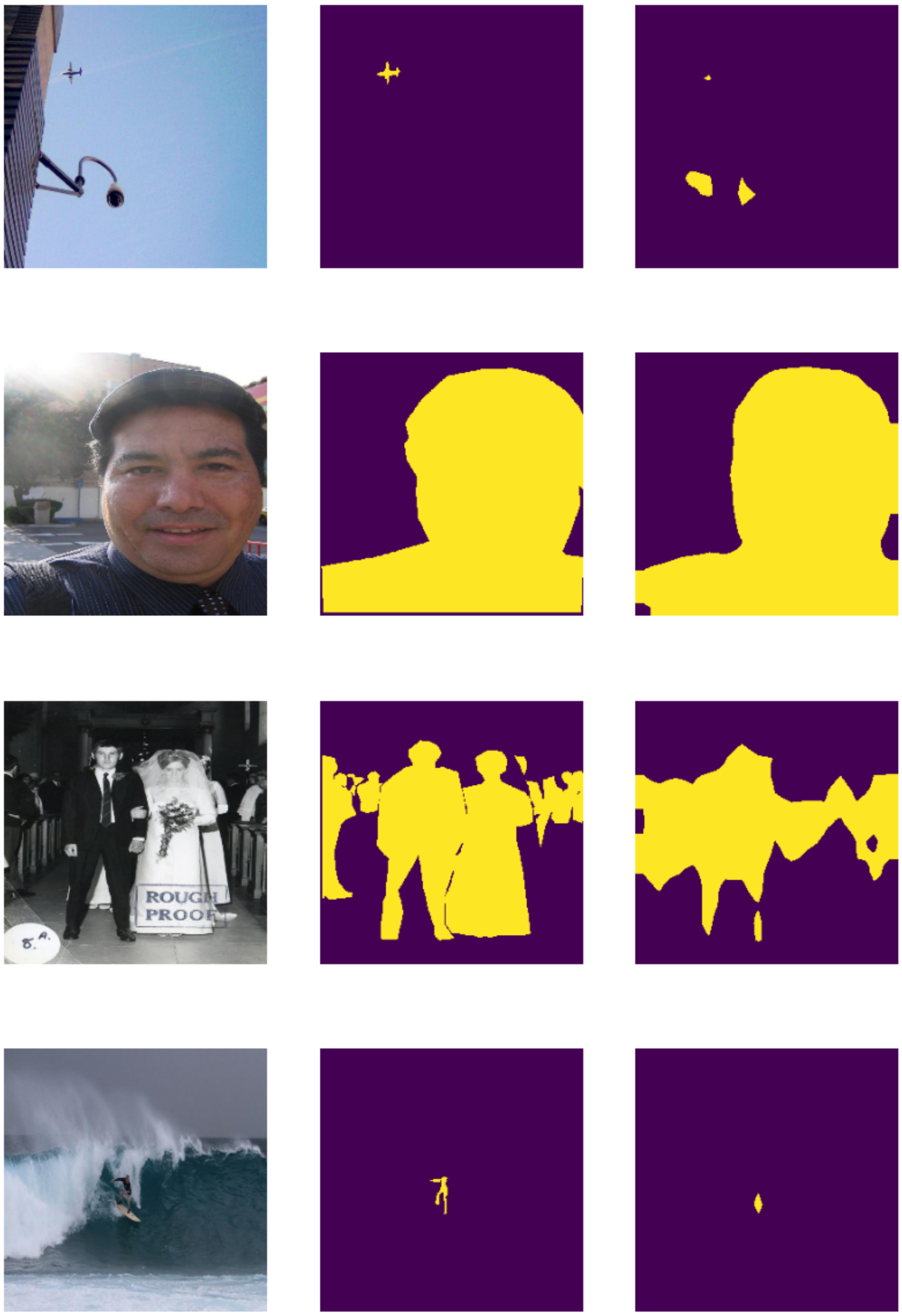

Figure 1: Illustration of model outputs on Test images

2.4 Comparison with Alternative:

I also trained **DeepLabV3 with a ResNet50** backbone for comparison. It achieved similar IoU scores (overall IoU $\simeq 0.53$) but required more computational power. Thus, MobileNetV3 was preferred for its efficiency without sacrificing significant performance.

2.5 Further Improvements

The following histogram 2 shows that the **dataset imbalanced** representation more of category 1. Hence, the model cannot generalize well to other classes. This skewed distribution explains lower IoU scores for rare classes. Potential solutions are:

1. **Weighted Cross-Entropy Loss:** Assign higher weights to underrepresented classes.
2. **Focal Tversky Loss:** Focus on hard examples to improve minority class performance.
3. **Oversampling:** Increase instances of rare classes during training.

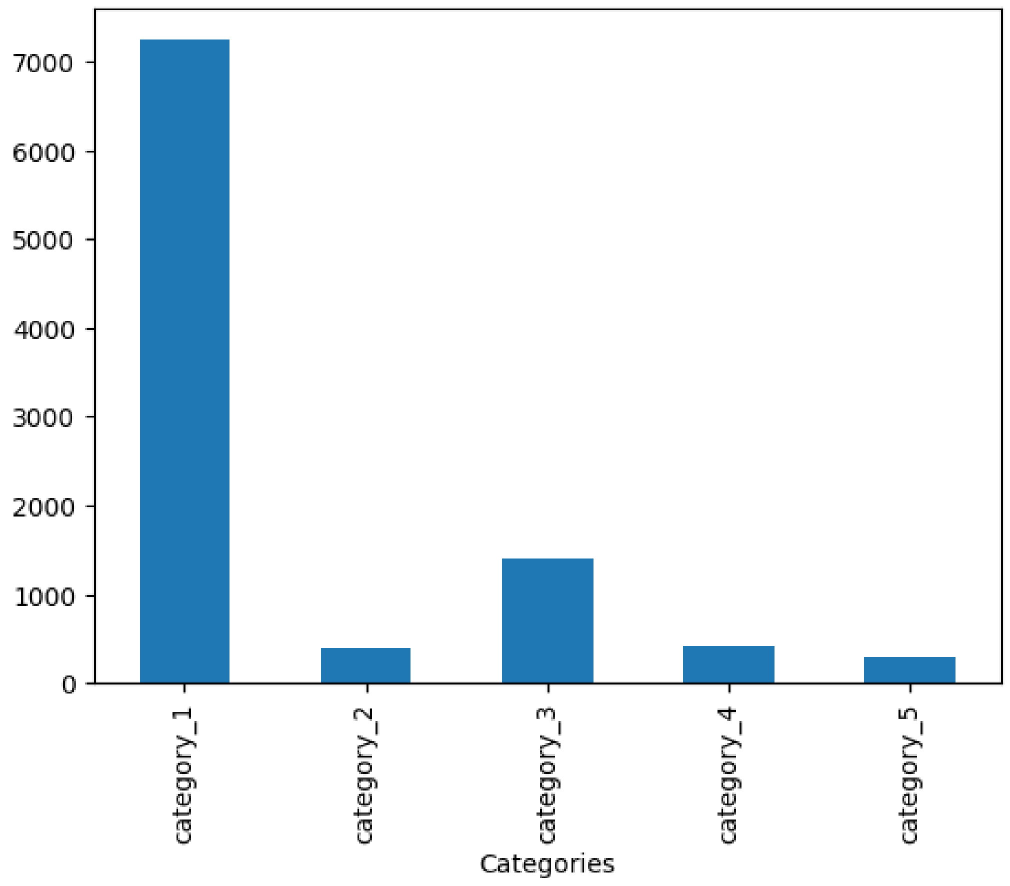

Figure 2: Histogram representing the distribution of various classes in the dataset.

3 Conclusion

This report demonstrates a complete pipeline for multi-label image segmentation using the COCO dataset and DeepLabV3 with MobileNetV3. The approach meets all assignment requirements, including dataset preparation, model training within computational limits, and clear evaluation using standard metrics. While the model performs well overall, addressing class imbalance and adding augmentation could further enhance results. The implementation is reproducible, with all code, weights, and metrics accessible via Github and WandB.