Project 2_Update 2

1. Introduction

This project implements a semantic segmentation model based on the DeepLabV3 architecture, using PyTorch to segment and classify pixel-level features in campus environment images. The goal is to utilize a DeepLabV3 model with a ResNet-101 backbone to effectively segment and classify three classes: stairs, doors, and background. The custom dataset created for this project consists of images captured from different parts of the campus, each paired with a mask indicating the segmented regions of interest. Hyperparameter tuning and validation techniques were applied to optimize performance metrics like mAP, loss, and accuracy.

2. Method

Model Architecture

The architecture used for this project is DeepLabV3 with a ResNet-50 backbone, a widely used model for semantic segmentation tasks. Below is a detailed explanation of the components and modifications made to adapt the model for the task of segmenting stairs, doors, and background:

- **DeepLabV3 Overview:**
  - DeepLabV3 is a semantic segmentation architecture designed to address the challenges of segmenting objects at varying scales and handling complex object boundaries.
  - The key feature of DeepLabV3 is its Atrous Spatial Pyramid Pooling (ASPP) module
- **ResNet-50 Backbone:**
  - The ResNet-50 backbone is used for feature extraction. ResNet-50 is a convolutional neural network with 50 layers and employs residual connections to address the vanishing gradient problem in deep networks.
  - It outputs high-level features that the ASPP module further processes.
- **Key Components of DeepLabV3:**
  - **ASPP Module:**
    - The ASPP module is integrated into the PyTorch implementation of DeepLabV3.
    - It is responsible for extracting multi-scale features from the input by applying parallel atrous convolutions with different dilation rates.
- **Model Customization:**
  - The DeepLabV3 model was trained on COCO dataset (for 21 classes) and pascal dataset (for 20 classes), so modifications were made to adapt it to this project:
    - **Final Classifier:** The number of output classes in the final layer was changed to 3 (doors, stairs, background). The classifier's final layer, responsible for generating pixel-wise predictions, was replaced with a convolutional layer to match the three-class segmentation requirement.
    - **Auxiliary Classifier:** This secondary classifier, which provides an additional loss to improve convergence during training, was also updated to output three classes. This was done to reduce Overfitting.
    - **Freezing Pre-trained Layers:** To retain the general features learned during pretraining, all layers of the ResNet backbone were frozen, except the classifier and auxiliary classifier layers. This approach leverages transfer learning, allowing the model to adapt quickly to the new task. Fine-tuning only the task-specific layers ensures efficient learning for the custom dataset.

Loss Function

The chosen loss function for this task is Cross-Entropy Loss, a standard for multi-class classification problems, including semantic segmentation. Here's a detailed explanation:

1. **Purpose:**
   - Semantic segmentation requires a class prediction for each pixel in an image. Cross-Entropy Loss is well-suited for this task because it treats each pixel as an individual classification problem.
   - The model learns to assign the highest probability to the correct class (e.g., stairs, doors, or background) for every pixel.
2. **Why Cross-Entropy Loss?**
   - **Probability-Based Loss:**
     Instead of penalizing predictions based on raw class scores, Cross-Entropy Loss penalizes the model based on the predicted probabilities compared to the true class probabilities.
   - **Pixel-Wise Optimization**:
     By computing the loss for each pixel, it ensures that the model learns fine-grained segmentation for all parts of the image, including challenging regions like boundaries between stairs and doors.
   - **Focus on Probability Distribution**:
     It ensures the model not only predicts the correct class but does so with a higher confidence by optimizing the probability distribution.

3. Dataset

The dataset used for this project was created to segment doors, stairs, and background in indoor environments. Images of doors were captured from indoor environment of the campus, along with images of both, indoor and outdoor, stairs.

- **Description:**
  - Dataset consists of 200 annotated images for doors, and stairs, along with background.
- **Annotation tool:**
  The annotation process was conducted using Roboflow, which provided an efficient platform for generating accurate labels and masks.

  Mask Creation:
  - Masks were generated as pixel-wise annotations corresponding to three classes:
    - **Doors**: Class 1
    - **Stairs**: Class 2
    - **Background**: Class 0

  Built-in Dataset Splitting:
  - Roboflow's built-in feature was used to split the dataset into training, validation, and test sets:
    - **Training Set**: 80% of the dataset.
    - **Validation Set**: 10% of the dataset.
    - **Test Set**: 10% of the dataset.

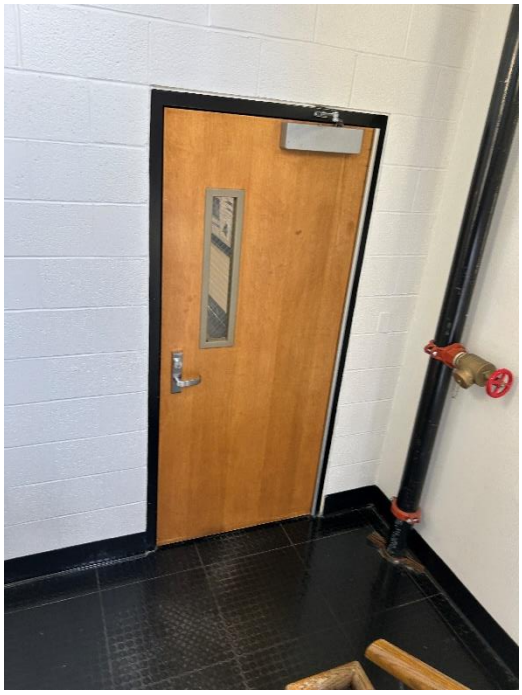
- **Data Augmentation:**
  Data augmentation was applied to enhance the diversity of the dataset and improve the model's generalization capability. Common augmentations included:
  - **Resizing**:
    - All images were resized to 512x512 pixels for uniform input size.
  - **Transformations**:
    - Random flipping, rotations, and brightness adjustments were applied to simulate diverse scenarios and prevent overfitting.

- **Normalization:**
  Images were normalized to align with the pretrained DeepLabV3 model's input requirements. This involves adjusting pixel intensity distributions using ImageNet mean and standard deviation values.
    - Used ImageNet mean and standard deviation: mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225].

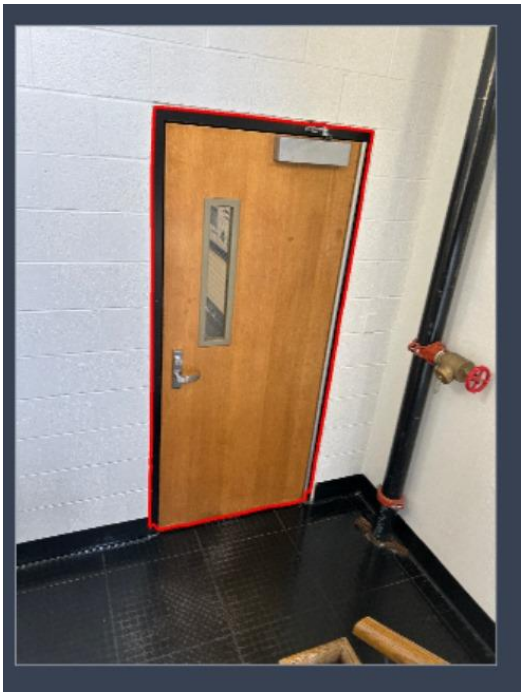
- **Mask Preparation**:
  Masks were converted to tensors after resizing, with pixel values representing the class labels (0 for background, 1 for doors, and 2 for stairs).

- **Data Structure:**
    - The images captured are in .JPG format and their respective masks are exported in .JSON format.

Image



Annotation



Dataset Link: [Dataset on SharePoint]

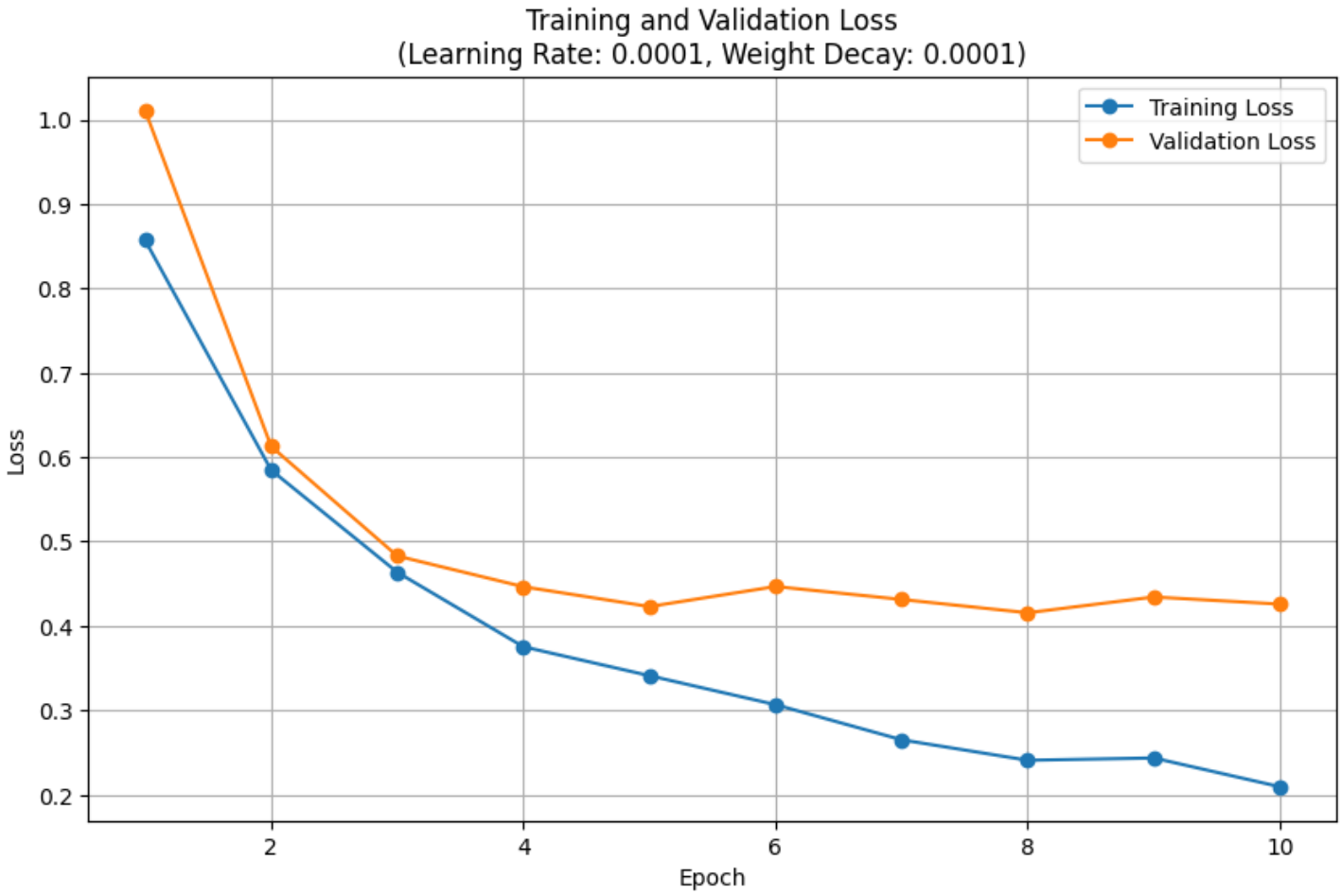
4. Experiments and Results

- **Hyperparameter Tuning:**
  The training process involved experimenting with different hyperparameter configurations to achieve the best segmentation performance. The selected hyperparameters and their ranges are as follows:
- **Learning Rate**:
    - Explored values: 1×10^{-3},1×10^{-4}
- **Weight Decay**:
    - Explored value: 1×10^{-4}
    - Regularization was applied to avoid overfitting.
- **Batch Size**:
    - Fixed value: 8 (as per the given instructions).
- **Number of Epochs**:
    - Experimented with up to 10 epochs for faster convergence.

- **Training and Validation Performance:**
  - Loss trends:
    - **1st hyperparamter set:** LR = 1×10^{-4} **,** Weight decay = 1×10^{-4}
      - Initial training loss: 0.8572; Final training loss: 0.2094.
      - Initial validation loss: 1.011; Final validation loss: 0.4259.
    - **2nd hyperparamter set:** LR = 1×10^{-3} **,** Weight decay = 1×10^{-4}
      - Initial training loss: 0.5355; Final training loss: 0.2004.
      - Initial validation loss: 0.6191; Final validation loss: 0.4546.

  Best hyperparamter value: **LR = 1×10^{-4} , Weight decay = 1×10^{-4}**

Training vs Validation Loss Plot:



The loss decreases steadily, indicating effective learning without significant overfitting.

- **Pre-Training Performance:**
  - **Training Loss:** 1.0559.
  - **Validation Loss:**1.0489.
- **Post-Training Performance:**
  - **Training Loss:** Reduced to 0.2094.
  - **Validation Loss:** Stabilized at 0.4259.
  - **mIoU:** Improved significantly to ~63%.
  - **Pixel-Wise Accuracy:** Increased to ~83%.
  - **Mean Accuracy:** ~81%

- **Visualizations:**
  - **Sample predictions compared to ground truth:**
    - Original Image | Ground Truth Mask | Predicted Mask.
    - Visualizations confirm improvement in pixel classification.