University of New Haven

PROJECT 2

Deep Learning (DSCI-6011-01)

Team Members:

1. Lavanya Gurram

2. Srikanth Thota

3. Shaagun Suresh

1. Introduction

The primary objective of this project is to fine-tune the **EfficientNet-B0** network for the task of food image segmentation, specifically targeting three food categories: **grilled chicken**, **paneer**, and **eggplant**, along with the background. By leveraging a pre-trained EfficientNet-B0 architecture, the model is adapted for semantic segmentation, trained using pixel-wise labeled data, and optimized through hyperparameter tuning. This report details the network architecture, dataset processing, hyperparameter tuning, and results of training and validation.

2. Method

2.1 Network Architecture

The **EfficientNet-B0** network, originally designed for image classification, is modified for semantic segmentation by appending an upsampling module. The architecture includes:

- **EfficientNet-B0 Backbone**: Pre-trained on ImageNet, used for feature extraction.
- **Upsampling Layer**: Converts the extracted feature maps to a pixel-wise prediction of classes.
- **Output Layer**: Produces a segmentation map where each pixel is classified into one of four classes (background, grilled chicken, paneer, and eggplant).

Key architectural details:

- **Input Size**: 512x512 RGB images.
- **Frozen Backbone**: EfficientNet-B0 layers are frozen to retain pre-trained weights.
- **Trainable Layers**: Upsampling and output layers are trainable, designed specifically for this task.

2.2 Loss Function

The model is trained using a **Weighted Cross-Entropy Loss**, where weights are assigned to handle class imbalance. For this task:

- **Background**: 0.1
- **Grilled Chicken, Paneer, Eggplant**: 1.0 each

This loss function ensures that underrepresented classes contribute equally to the loss computation.

3. Dataset

3.1 Description

The dataset consists of **200 food images**, each annotated with pixel-wise masks representing the three classes:

- **Grilled Chicken** (Red): (255, 0, 0)
- **Paneer** (Green): (0, 255, 0)
- **Eggplant** (Blue): (0, 0, 255)

3.2 Annotation and Labeling

The annotation process began by using LabelMe, an annotation tool that allows for polygon-based labeling. Each food item—**Grilled Chicken**, **Paneer**, and **Eggplant**—was annotated using polygons to outline the precise shape of each object within the images. Labels were assigned to each food item directly within LabelMe, which generated JSON files containing the polygon coordinates and associated labels.

These JSON files were later processed using a Python script to convert the polygon-based annotations into pixel-wise masks. During this conversion, each label was mapped to a specific RGB color, creating color-coded masks:

- **Grilled Chicken**: Red (255, 0, 0)
- **Paneer**: Green (0, 255, 0)
- **Eggplant**: Blue (0, 0, 255)

By converting the JSON files into color-coded masks, the model was able to learn pixel-level details specific to each food item, providing a robust basis for accurate segmentation. The RGB colors were subsequently translated into class indices during data loading, making it easier for the model to interpret and predict pixel-wise classifications.

One drive link - Images and Annotation

3.3 Data Collection

Images were sourced from:

1. AI-generated supplementary images to increase diversity. – Stable Diffusion

3.4 Partitioning

The dataset is split into:

- **Training Set**: 80% (160 images)
- **Validation Set**: 20% (40 images)

3.5 Data Augmentation

To improve model generalization, the following augmentations were applied:

- **Random Horizontal Flip**: Probability = 0.5
- **Random Rotation**: ±15 degrees
- **Color Jitter**: Brightness, contrast, saturation, and hue adjustments.

3.6 Normalization

The images were normalized to match the pre-trained EfficientNet's input requirements:

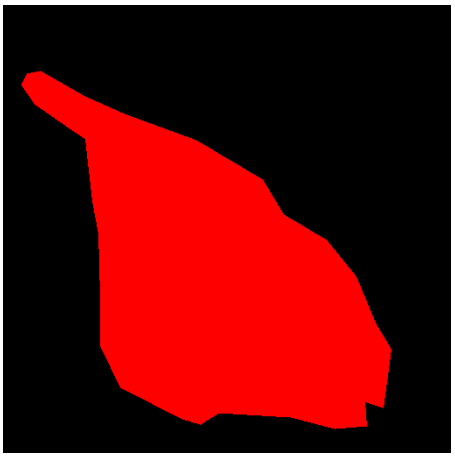
- **Mean**: [0.485, 0.456, 0.406]
- **Standard Deviation**: [0.229, 0.224, 0.225]

3.7 Sample Image and Annotation

Below is an example of a dataset sample:

- **Input Image**: 512x512 RGB.
- **Annotation**: JSON files converted into pixel-wise masks.

INPUT **MASK**



3.8 Data Structure

Each sample includes:

- **Input Image**: Tensor of shape [3, 512, 512].
- **Annotation Mask**: Tensor of shape [512, 512], with pixel values as class indices.

4. Experiments and Results

4.1 Hyperparameter Tuning

Several hyperparameters were tuned to achieve the best results:

- **Batch Size**: 8
- **Learning Rate**: Initially set to 0.001, reduced by half every 5 epochs.
- **Optimizer**: **AdamW** for efficient parameter updates.
- **Scheduler**: Learning rate scheduler with step size = 5 and gamma = 0.5.
- **Number of Epochs**: 30

Hyperparameter	**Range**	**Best Value**
Learning Rate	0.0001–0.001	0.001
Batch Size	4–16	8
Optimizer	SGD, Adam, AdamW	AdamW
Scheduler Step	3–10	5

4.2 Training and Validation Performance

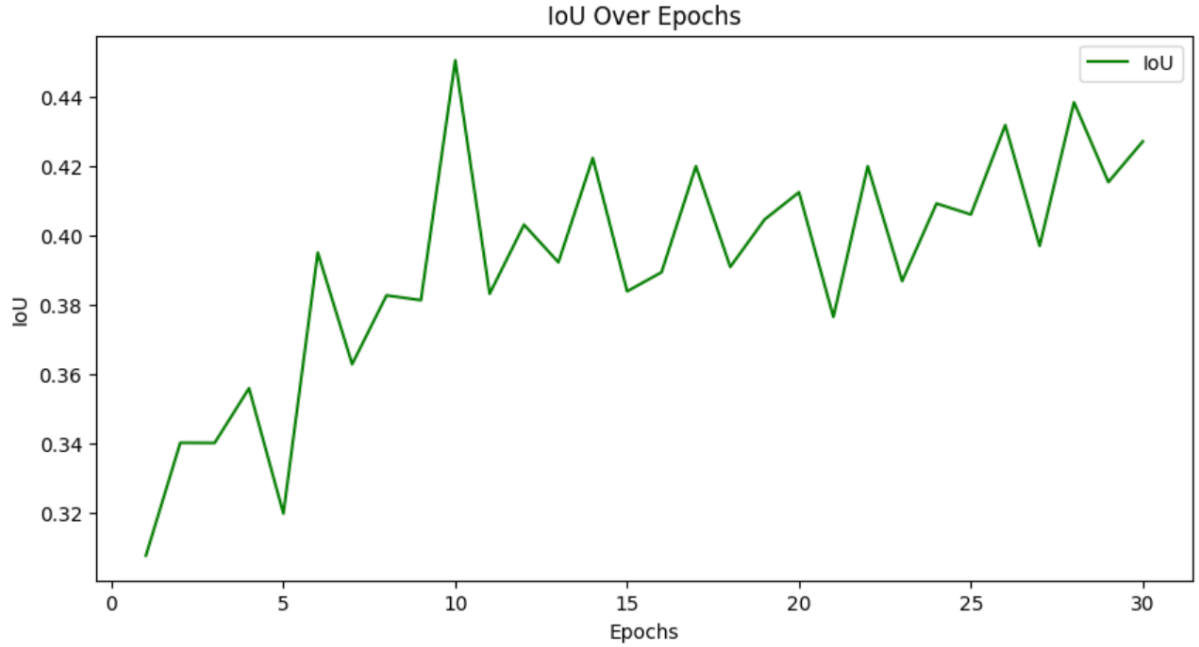
The model was trained for 30 epochs. Below are the results:

- **Best IoU**: 0.4504
- **Training Loss**: 0.3427
- **Validation Loss**: 0.5013

Plots:

1. **Training and Validation Loss**:

2. **IoU Over Epochs**:



4.3 Comparison of Pre- and Post-Training Performances

Pre-Training Performance:

- The model produced nearly random outputs before fine-tuning.
- There was little to no segmentation capability, as the model had not been specifically trained for the food segmentation task.

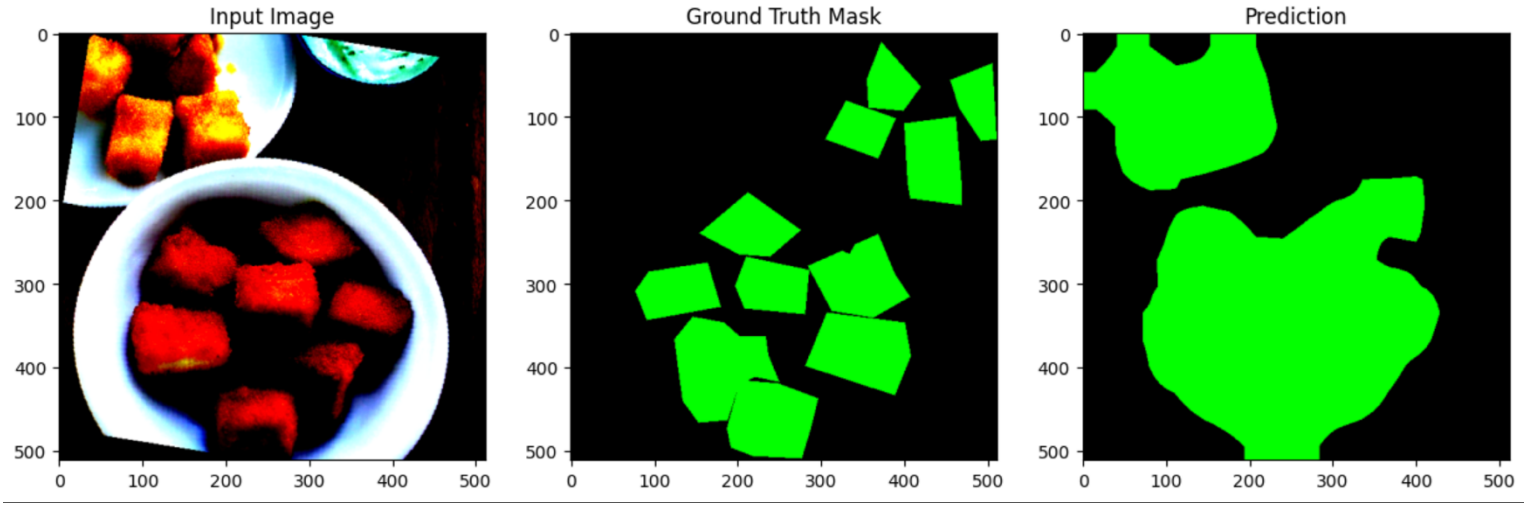
Post-Training Performance:

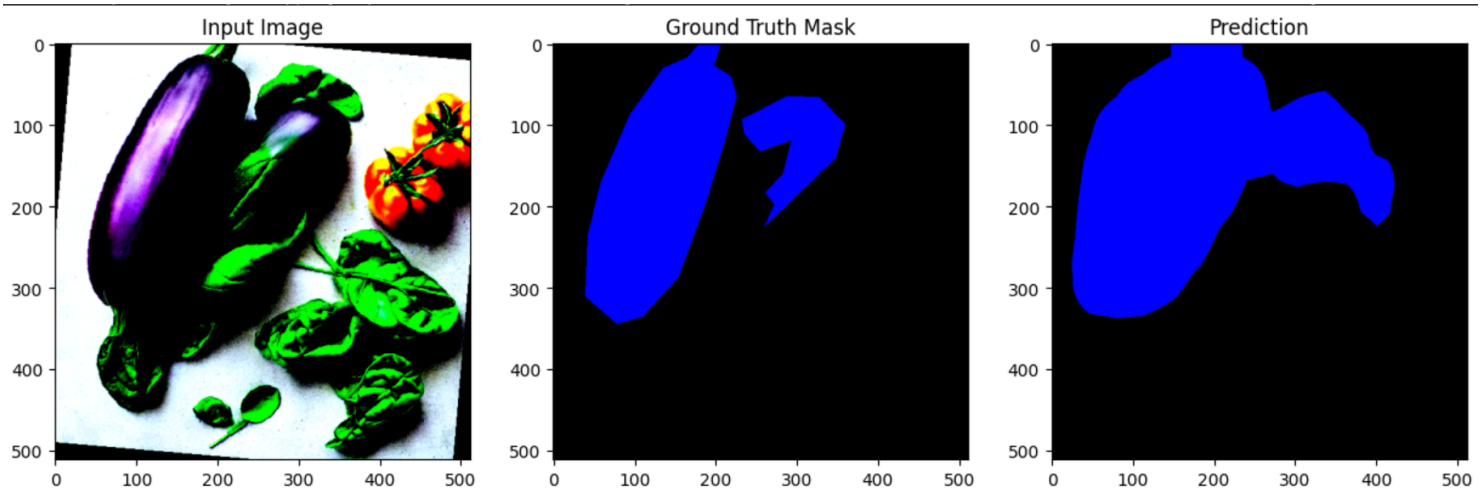
- After training, the model demonstrated noticeable improvement in segmentation capability.
- **Key Metrics:**
    - Best Intersection over Union (IoU): Improved to **0.4504**.
    - Training Loss: Reduced to **0.3427**.
    - Validation Loss: Reduced to **0.5013**.
- Improved ability to differentiate between the three food categories: grilled chicken, paneer, and eggplant.

Training Dynamics:

- **Training Loss:** Steadily decreased across epochs, indicating that the model effectively learned from the data.
- **Validation Loss:** Fluctuated slightly but showed overall improvement, suggesting reasonable generalization.
- **IoU Scores:** Increased from an initial **0.3076** in the first epoch to **0.4504** in the best epoch.

Visualization:





4.4 Analysis of Result

- **Improvement in Performance:**

  - The fine-tuning process significantly enhanced the model's ability to predict segmentation masks, evident from the improvement in IoU and reduction in loss values.
  - The model learned to classify pixels into correct categories for most regions of interest, demonstrating the effectiveness of the EfficientNet-B0 backbone for segmentation.

- **Strengths of Post-Training Model:**
  - The model generalized well without significant overfitting, as indicated by the relatively close training and validation losses.
  - It effectively segmented larger and distinct objects with clear boundaries.
- **Challenges Observed:**
  - The model struggled with small objects and overlapping categories, likely due to:
    - Limited dataset size (200 images may not fully capture the variability in food textures and colors).
    - Similarity in texture or color among the classes (e.g., grilled chicken and paneer).
  - Over-segmentation and under-segmentation were observed, where the model either mislabeled large areas or failed to label certain portions correctly.
- **Room for Improvement:**
  - IoU of 0.4504 is modest, indicating potential for further enhancement.
  - Addressing these challenges may require:
    - Increasing dataset size and diversity
    - Experimenting with data augmentation strategies to simulate more variations.
    - Refining the model architecture
- **Visualization Insights:**
  - Visualizations highlight that while predictions improved, the segmentation maps are often fragmented or contain misclassifications.
  - The model shows potential for improvement in delineating finer details and reducing class confusion.

5. Conclusion

This project successfully fine-tuned the **EfficientNet-B0** model for food segmentation. The model achieved a best IoU of **0.4504** with promising segmentation capabilities for the given categories.