University *of*
New Haven

PROJECT 2 UPDATE#1

Deep Learning (DSCI-6011-01)

Team Members:
1. Lavanya Gurram
2. Srikanth Thota
3. Shaagun Suresh

Project 2: Custom Food Segmentation with EfficientNet

1. Introduction

This project addresses the challenge of segmenting food items within images into distinct categories, specifically identifying "grilled chicken," "paneer," and "eggplant" using deep learning. Using a pre-trained EfficientNet-B0 model modified for semantic segmentation, this project explores strategies to accurately classify food items within a complex background. The report details the dataset preparation, training methodology, evaluation, and results achieved, with an emphasis on how well the model differentiates between the three food categories and the background.

2. Dataset Description

The dataset consists of 200 high-resolution images, each accompanied by a corresponding annotated mask that labels specific areas in the image as one of three classes:

- **Grilled Chicken** (red)
- **Paneer** (green)
- **Eggplant** (blue)

Each image and mask pair is stored in separate folders to facilitate loading and preprocessing. The images are in PNG format, and each pixel in the annotation mask corresponds to one of the four categories, represented by distinct RGB color values. This color-coded structure allows for straightforward mapping from RGB values to class indices during training.

One drive link - Images and Annotation

3. 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.

4. Data Partitioning

The dataset is divided to ensure the model has a balanced set for training and evaluation:

- **Training Set**: 80% (160 images) used to train the model.
- **Validation Set**: 10% (20 images) used to monitor model performance and prevent overfitting.
- **Test Set**: 10% (20 images) held back for final evaluation to measure generalization capability.

This partitioning strategy allows the model to learn effectively while also providing reliable metrics to evaluate its performance on unseen data.

5. Data Normalization and Transformation

To ensure consistency in the input images, each image and mask undergoes the following transformations:

- **Resize**: Resized to 512x512 pixels to ensure a uniform input size for the model.
- **Normalization**: Images are normalized to standardize pixel values, which helps improve model stability and performance during training.

6. Model Architecture and Methodology

The **EfficientNet-B0** architecture is utilized, with modifications to support segmentation tasks. Key features include:

- **EfficientNet Backbone**: Provides rich feature extraction, capturing spatial patterns in the image.
- **Upsampling Layer**: Adds a layer that outputs predictions at the original image resolution (512x512), allowing pixel-wise classifications.
- **Freezing Backbone Layers**: EfficientNet-B0's pre-trained layers are frozen, retaining learned feature extraction weights, while the final classifier layers are trained for segmentation.

The model outputs a segmentation map with class probabilities, where each pixel is classified into one of the four categories.

Loss Function

The model uses a Cross-Entropy Loss function to handle the classification task, assigning each pixel to one of the three classes. This loss function is particularly suitable for segmentation tasks with multiple classes

7. Evaluation Metrics

The performance of the model is evaluated using Cross-Entropy Loss on both training and validation datasets. The average loss for each dataset is calculated, providing insight into the model's learning progress and generalization ability.

8. Results and Analysis

The model was trained and evaluated on the training and validation datasets, with average training and validation losses recorded as follows:

- **Average Training Loss**: 1.491
- **Average Validation Loss**: 1.508

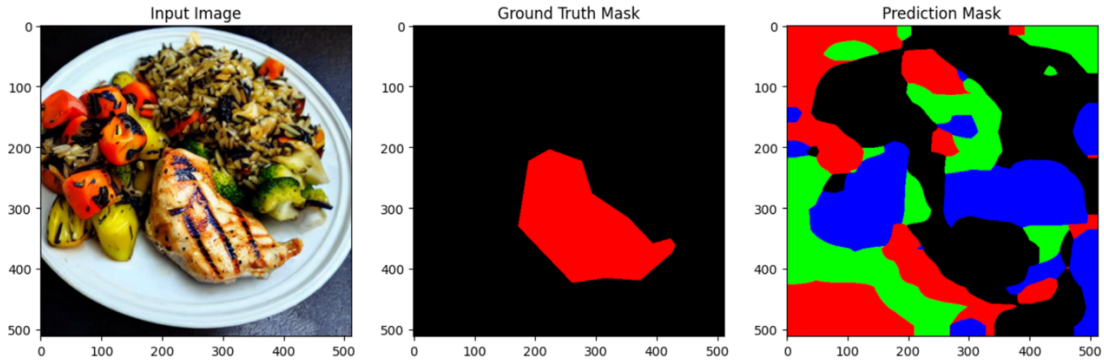
These values indicate that the model has learned to some extent, but there is room for improvement in further reducing the loss to enhance segmentation accuracy.

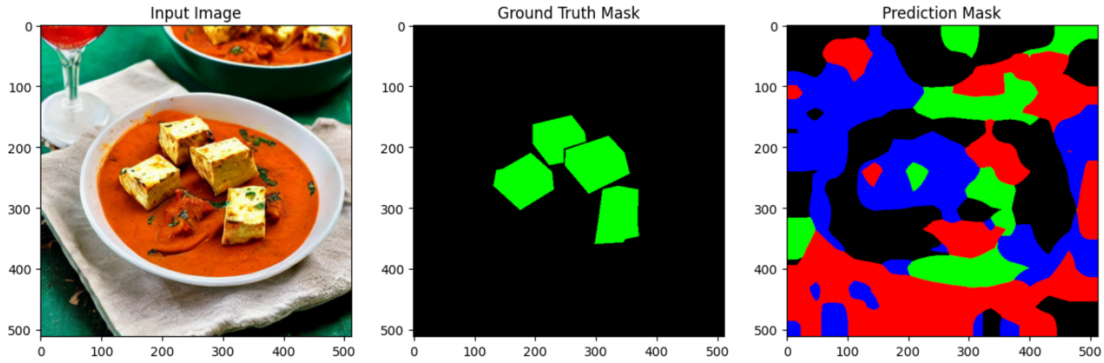
The model's training and validation losses are very close (1.491 for training and 1.508 for validation), indicating that the model is not significantly overfitting or underfitting.

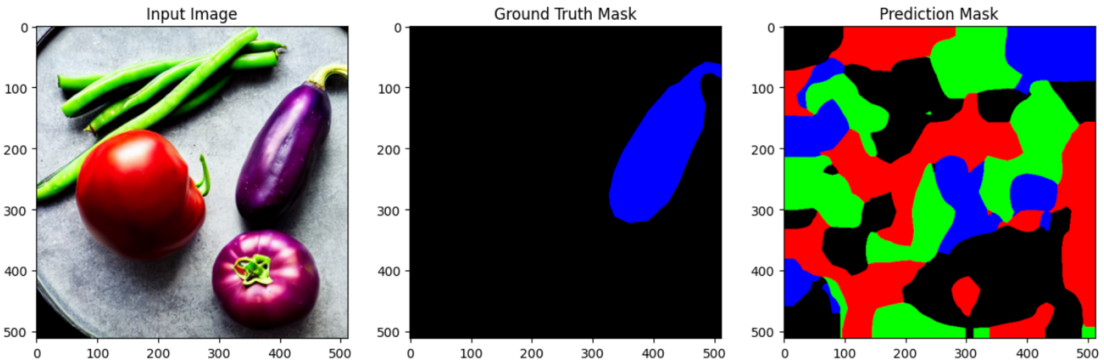
In overfitting, the training loss would be much lower than the validation loss, showing that the model performs well on training data but poorly on unseen data. In underfitting, both training and validation losses would be high, meaning the model struggles to capture the data patterns well enough for effective learning.

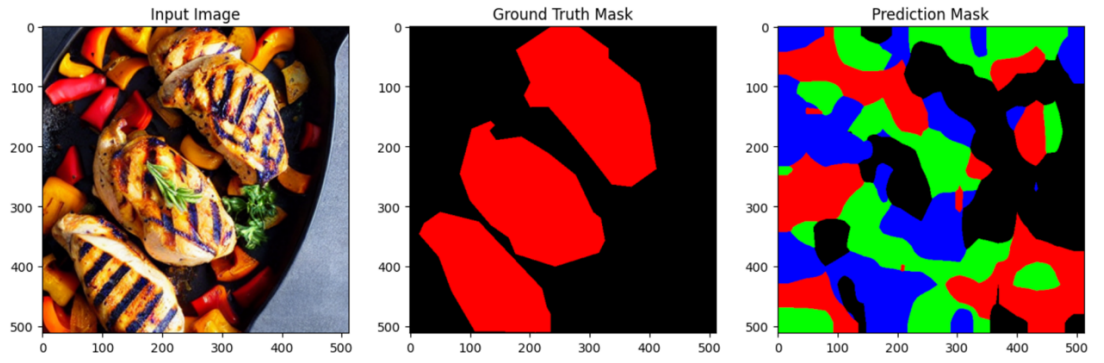
Since the training and validation losses in this case are almost equal, it suggests that the model is well-balanced between learning the patterns in the training data and generalizing to the validation data. However, the moderately high values for both losses imply that the model may not have fully learned the features needed for precise segmentation, which could be addressed by further tuning or more data.

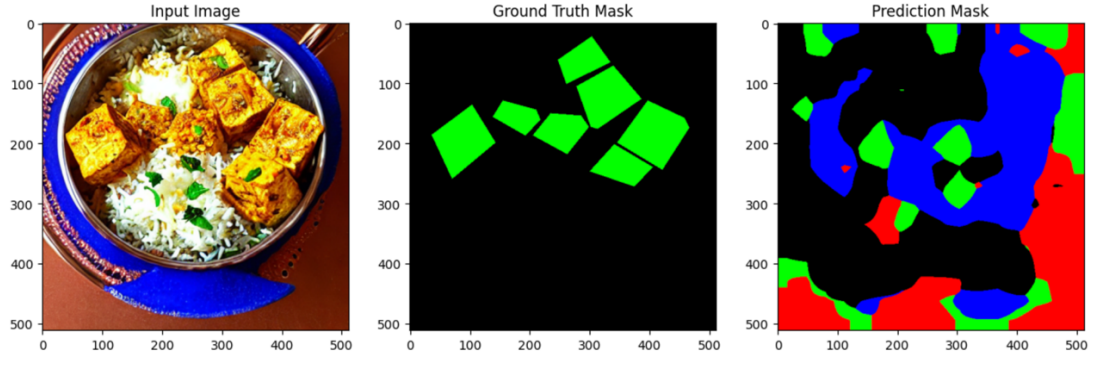
In summary, model is neither overfitted nor underfitted, but it may benefit from adjustments to improve accuracy.


Input Image
Ground Truth Mask
Prediction Mask


Input Image
Ground Truth Mask
Prediction Mask


Input Image
Ground Truth Mask
Prediction Mask


Input Image
Ground Truth Mask
Prediction Mask


Input Image
Ground Truth Mask
Prediction Mask

The visualizations reveal several insights into the model's current capabilities and limitations:

- **Inconsistent Segmentation**: The model struggles to consistently identify individual food items and often produces fragmented segmentation maps.
- **Class Confusion**: There is frequent misclassification between classes (e.g., grilled chicken identified as paneer or background), indicating that the model is finding it challenging to differentiate between the specific textures or colors of each class.
- **Over-segmentation and Under-segmentation**: In many cases, the model assigns incorrect labels to large portions of the image (over-segmentation) or misses parts of the target objects (under-segmentation), which suggests the need for more refined feature extraction.

9 . Conclusion

This project successfully implemented a modified EfficientNet-B0 model for food segmentation, achieving an average validation loss of **1.508**. While the model demonstrates basic segmentation abilities, significant improvements are needed for reliable application. Future work could focus on optimizing the model architecture, fine tune the model and exploring advanced techniques to enhance segmentation quality.