E9-246 ADVANCED IMAGE PROCESSING

ASSIGNMENT 2: REPORT

NAME: HARSHIT DUBEY

SR. NO.: 04-03-06-10-51-23-1-23043

DEGREE: M.TECH AI

DEPARTMENT : COMPUTER SCIENCE AND AUTOMATION

Question 1:

Segmentation using N-Cut (2 Segments)

Brief of Implementation

Functions:

- *compute matrix:* Calculates the similarity matrix between each pixel pair based on spatial distance and color difference.
- *compute segment:* Segments the image using spectral clustering based on the similarity matrix.
- *get_colors:* Calculates the average color for each segment (foreground and background).
- *display_image*: Displays the segmented image and the similarity matrix.
- segmentation: Main function that resizes the image, calls other functions, and displays the results.

Steps:

- *Preprocessing:* The image is resized to a smaller size (64x64) for efficiency.
- *Similarity matrix:* The similarity between each pixel pair is calculated based on:
    - Spatial distance: Euclidean distance between pixel coordinates.
    - Color difference: Euclidean distance between pixel color values.
    - Threshold: Only pixels within a certain spatial distance are considered similar.
- *Segmentation:*
    - The similarity matrix and a degree matrix (diagonal matrix of row sums) are used to construct a graph representation of the image.
    - Spectral clustering is applied to the graph to divide pixels into two clusters (foreground and background).
- *Color calculation:* The average color for each cluster is calculated.
- *Visualization:* The segmented image and the similarity matrix are displayed.

Inputs:

- *image:* The input image.
- *sigma1:* Parameter controlling the influence of color difference on similarity.
- *sigmaX:* Parameter controlling the influence of spatial distance on similarity.
- *r:* Threshold for spatial distance in similarity calculation.

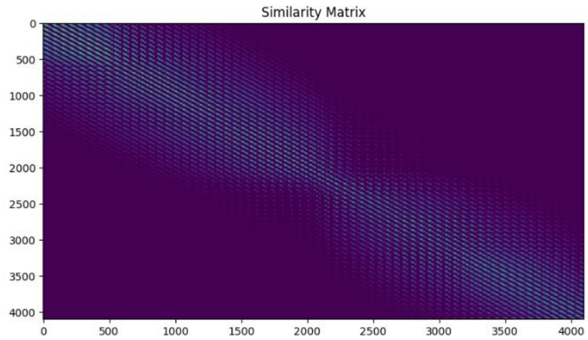
Output:

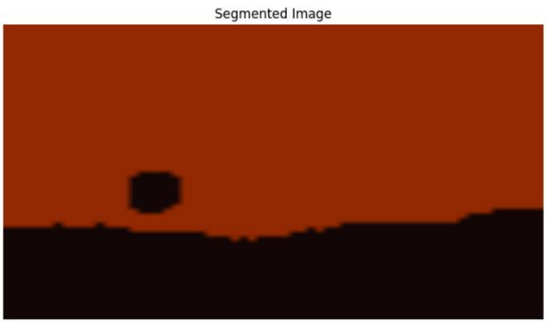
- A segmented image where each pixel is assigned to either foreground or background.
- The similarity matrix used for segmentation.

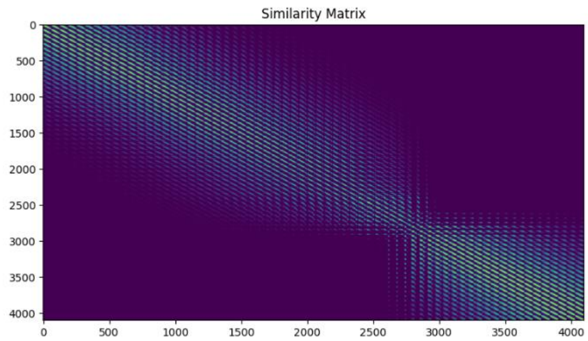
Similarity Measure 1: (RGB Values)

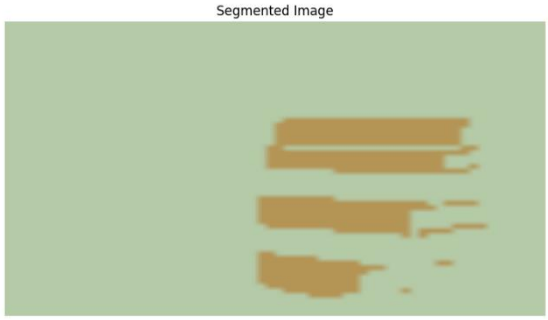
RGB Images

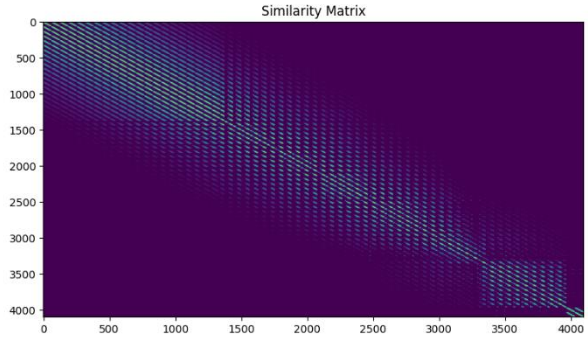
Segmented Images:

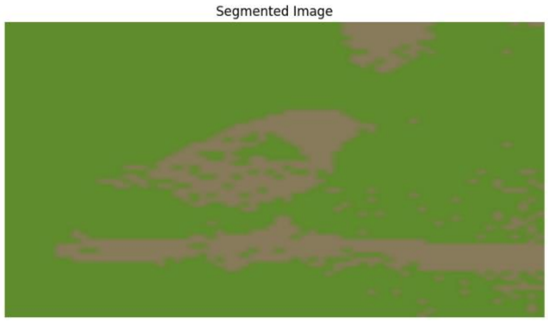


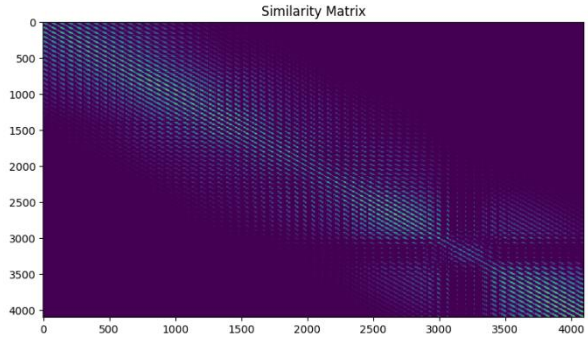




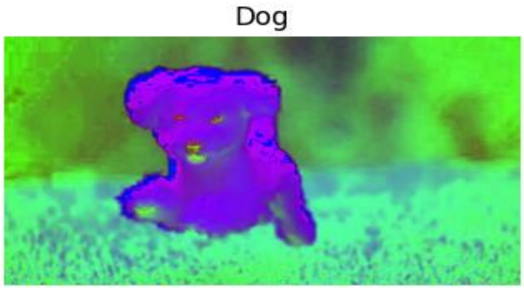


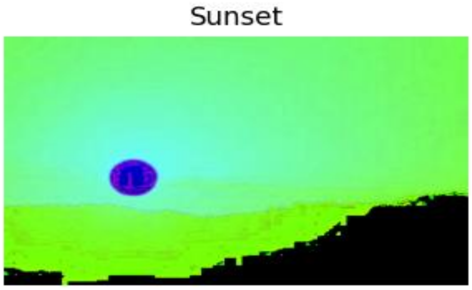


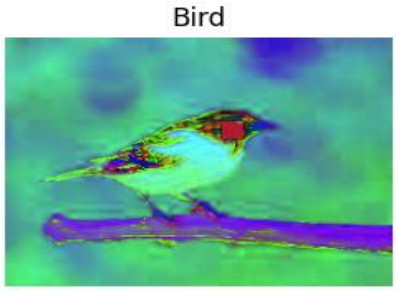




Similarity Measure 2: (HSV Values)

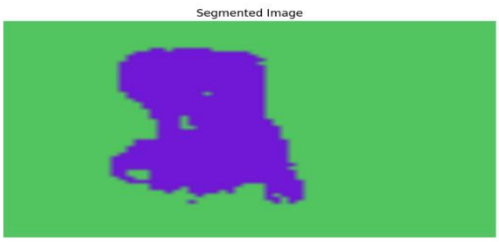


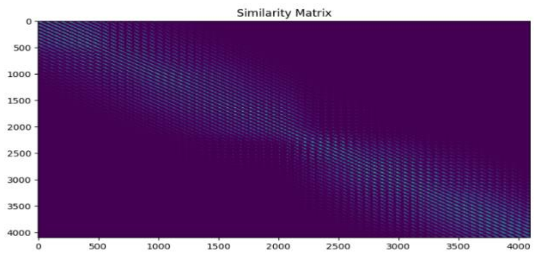


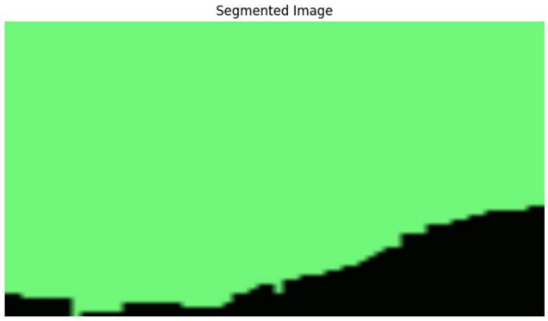


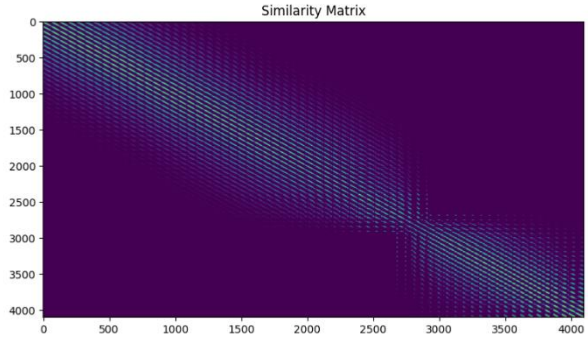
HSV Images

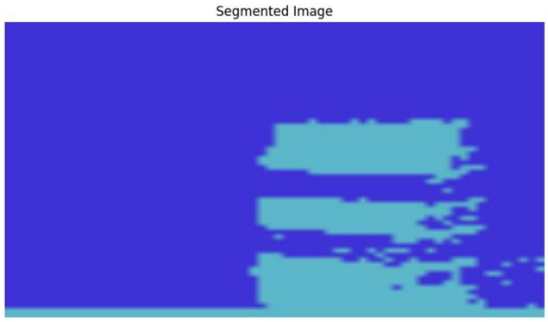
Segmented Images:

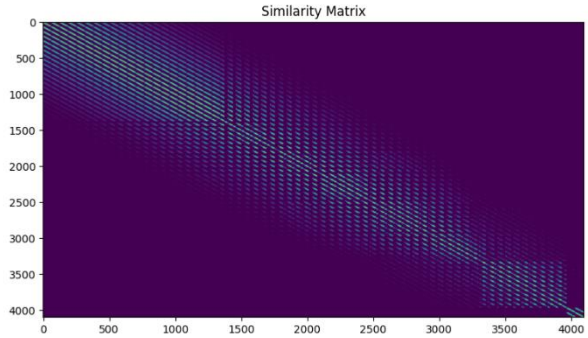


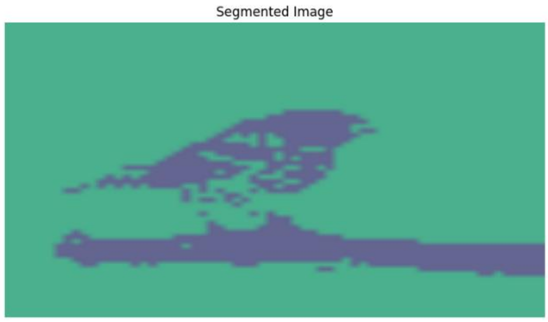


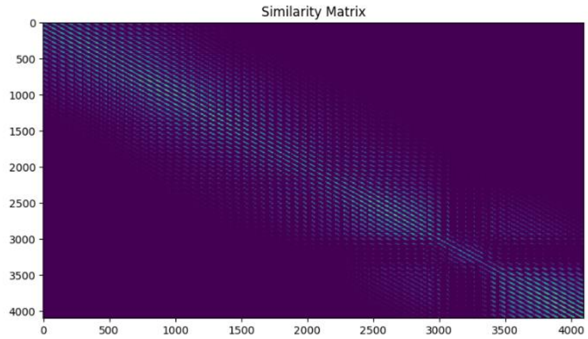








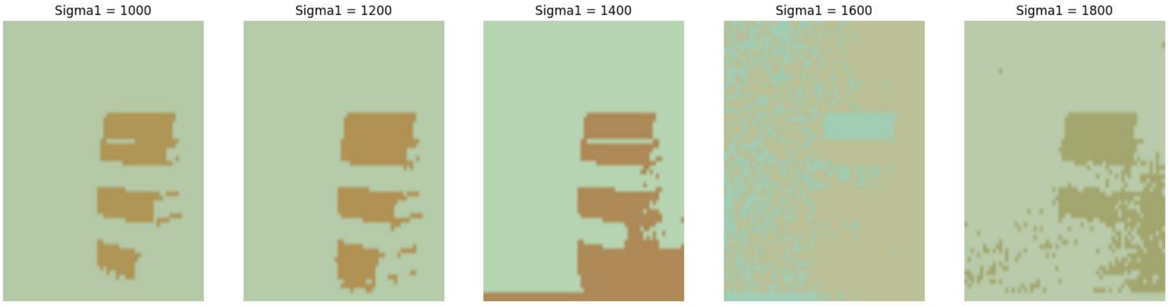




Analysis:

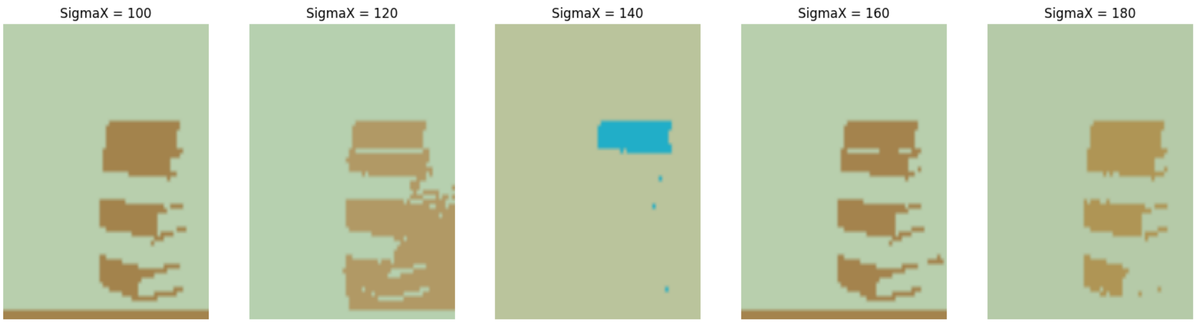
Impact on segmentation due to variation in sigma1 value:

- Higher sigma1:
  - Less weight to color differences.
  - Larger segments with potential merging of similar colors.
  - Less sensitive to noise or slight color variations.
- Lower sigma1:
  - Emphasizes color differences.
  - Smaller segments following object boundaries more closely.
  - More sensitive to noise or slight color variations.



Impact on segmentation due to variation in sigmaX value:

- *Higher sigmaX:*
  - Considers pixels further apart as similar.
  - Leads to larger segments.
  - May merge nearby objects.
- *Lower sigmaX:*
  - Emphasizes close proximity.
  - Leads to smaller segments.
  - Follows object boundaries more closely.



Impact on segmentation due to variation in r value:

- Higher values: larger, smoother segments with less sensitivity to noise.
- Lower values: smaller, detailed segments with more sensitivity to noise.

Segmentation using N-Cut (more than 2 Segments)

Brief of Implementation

- ***Constructing Similarity and Degree Matrices:***
    - The compute_matrix function remains unchanged, calculating similarity and degree matrices based on spatial proximity and color similarity.
- ***Spectral Clustering with Multiple Eigenvectors:***
    - The compute_segment_xtra function now uses k=4 in eigs to compute four leading eigenvectors representing four segments.
    - It assigns pixels to segments based on comparisons with mean values of the second and third eigenvectors.
- ***Color Calculation and Visualization:***
    - The get_colors_xtra function finds coordinates of each segment and calculates mean colors for each.
    - The display_image_xtra function replaces pixel values with corresponding segment colors and displays the segmented image.
- ***Main Function:***
    - The segmentation_xtra function resizes the image, calls other functions to perform segmentation and visualization, and returns the segmented image.

Segmented Images:

Difference between 2 segment and more than 2 segment implementations :

Feature	Two Segments	More than Two Segments
Eigenvectors used	First two eigenvectors	k eigenvectors (k > 2)
Segment assignment	Based on mean of second smallest eigenvector	Comparisons with means of additional eigenvectors
Number of segments	Fixed at 2	Variable, depending on k

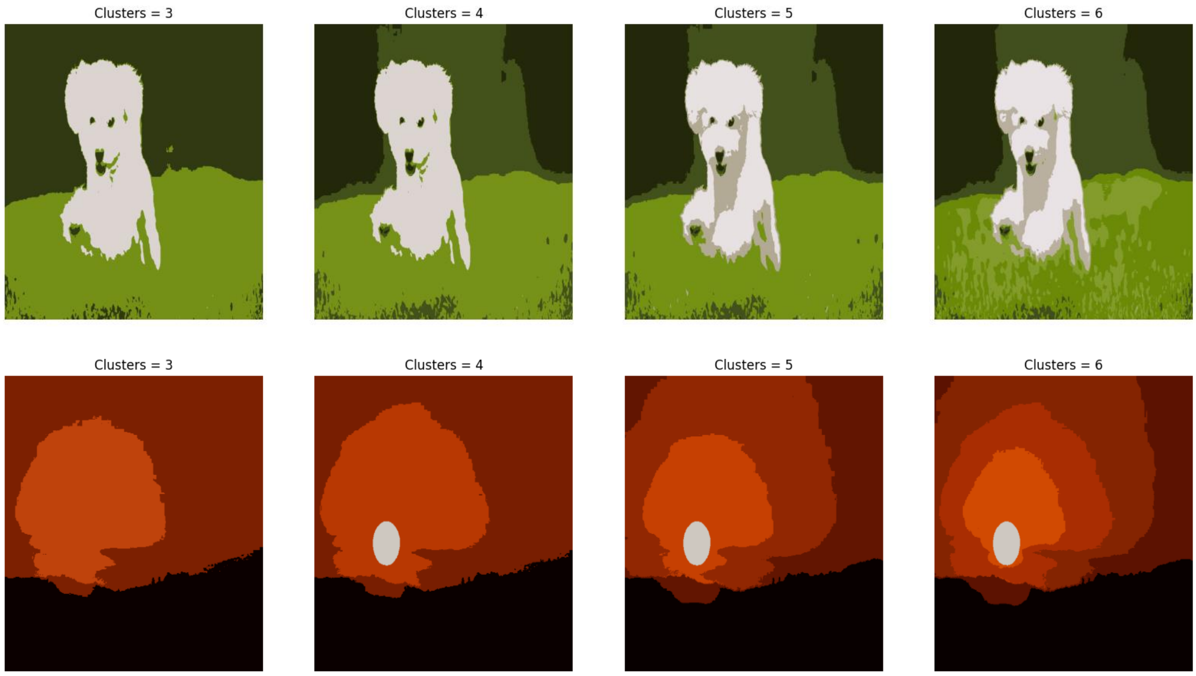
NOTE: The choice of `k` for more than two segments impacts the complexity and granularity of the segmentation.

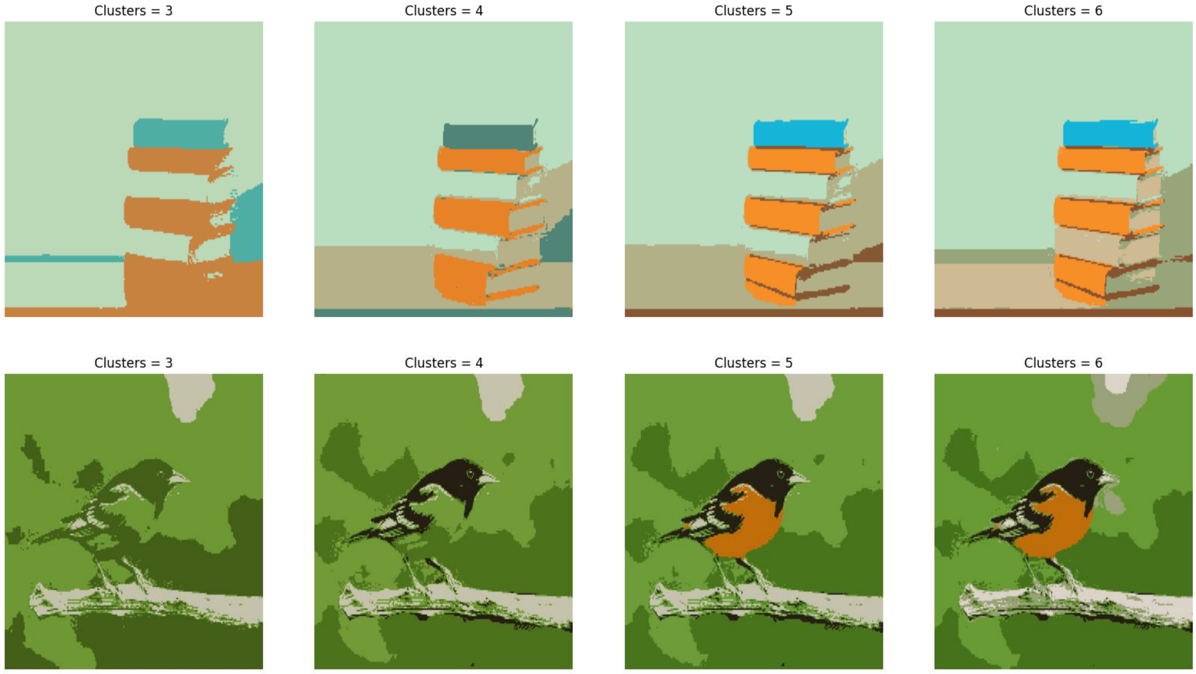
Segmentation using K-Means:

Brief of Implementation

- The input image is reshaped into a 2D array with each row representing a pixel and the three columns representing the Red, Green, and Blue (RGB) values of that pixel.
- ***K-means Clustering:*** For each cluster value from 3 to 6, a K-means clustering is performed on the flattened image. The n_clusters parameter is set to the current cluster value and n_init is set to ‘auto’. The K-means model is then fitted to the flattened image.
- ***Segment the Image:*** The segmented image is created by replacing each pixel with its corresponding cluster center. This segmented image is then reshaped back to the original image shape and normalized by dividing by 255.
- The segmented image is appended to the list of resultant images. For each cluster value from 3 to 6, the corresponding segmented image is displayed

Segmented Images:





Analysis:

Changing the value of k (the number of clusters) in the k-means segmentation algorithm affects the granularity of the segmentation and the level of detail captured.

- With fewer clusters (k=3), the image is segmented into larger, more general regions. This is a good option if you only need a basic understanding of the image's content.

- With more clusters (k=4, 5, 6), the image is segmented into smaller, more detailed regions. This is a good option if you need to capture more fine-grained details in the image.

However, there is a trade-off between the number of clusters and the quality of the segmentation. With too few clusters, the segmentation may be too coarse and miss important details. With too many clusters, the segmentation may be over-fitted to the noise in the image and create inaccurate or nonsensical segments.

The optimal value of k for a particular image will depend on the specific task you are trying to accomplish.

Question 2:

Segmentation using FCN:

Brief of Implementation

The code implements two variants of Fully Convolutional Networks (FCNs) based on the ResNet-18 architecture for image segmentation. The FCNs are trained on a dataset and evaluated for pixel-wise accuracy and mean Intersection over Union (IoU) on the test set.

Overview

- ***Dataset Preparation:*** The code starts by mounting Google Drive to access the dataset stored there. Then, it unzips the dataset and defines functions to create the dataset, load images and masks, and prepare them for training.

- ***Model Architecture:*** Two variants of the FCN based on ResNet-18 are defined:
    - *ResNet_segmentor:* This variant includes skip connections, allowing information from earlier layers to be fused with later layers during upsampling.
    - *ResNet_segmentor_noskip:* This variant does not include skip connections.

- ***Training:*** Both models are trained using the provided dataset. The training loop iterates over batches of images and masks, computes the loss, and updates the model parameters using backpropagation.

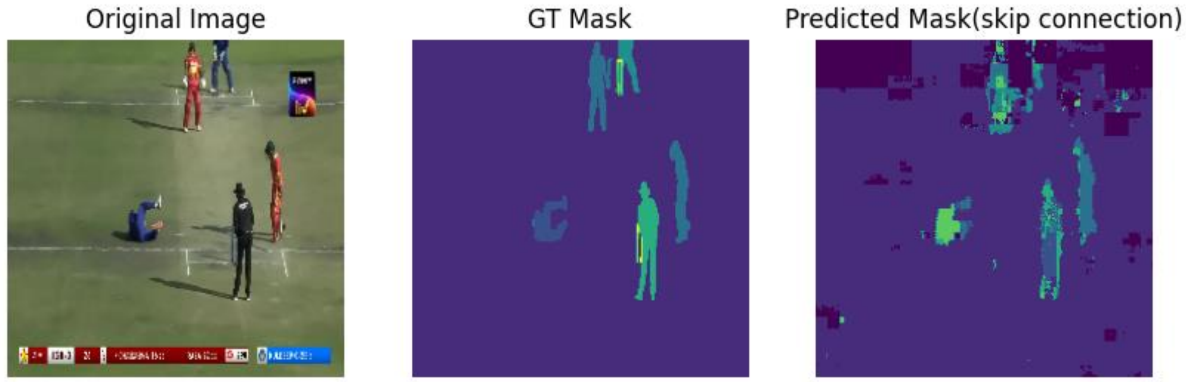
- ***Evaluation:*** After training, the models are evaluated on the test set. The pixel-wise accuracy and mean IoU are calculated for both models.

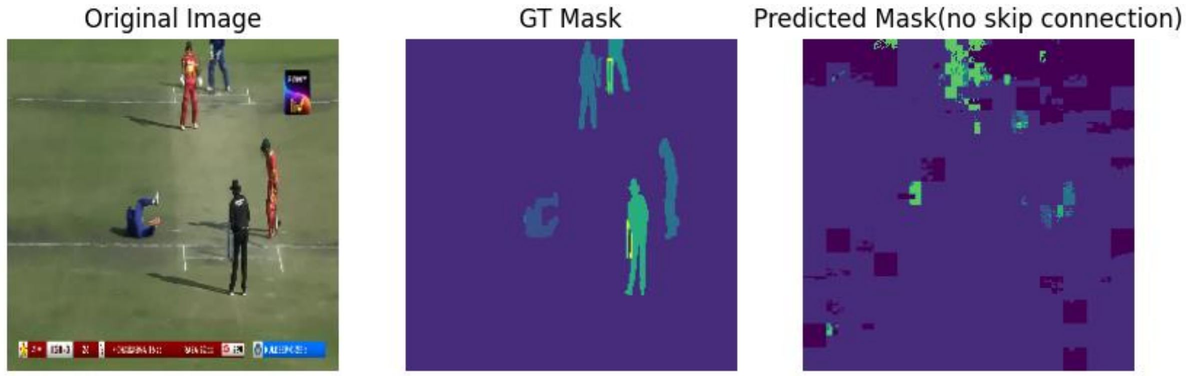
- ***Visualization:*** Sample images from the test set along with their ground truth masks and predicted masks are displayed for both models.

- ***Loss Plotting:*** Finally, the loss values during training for both models are plotted to compare their training performance.

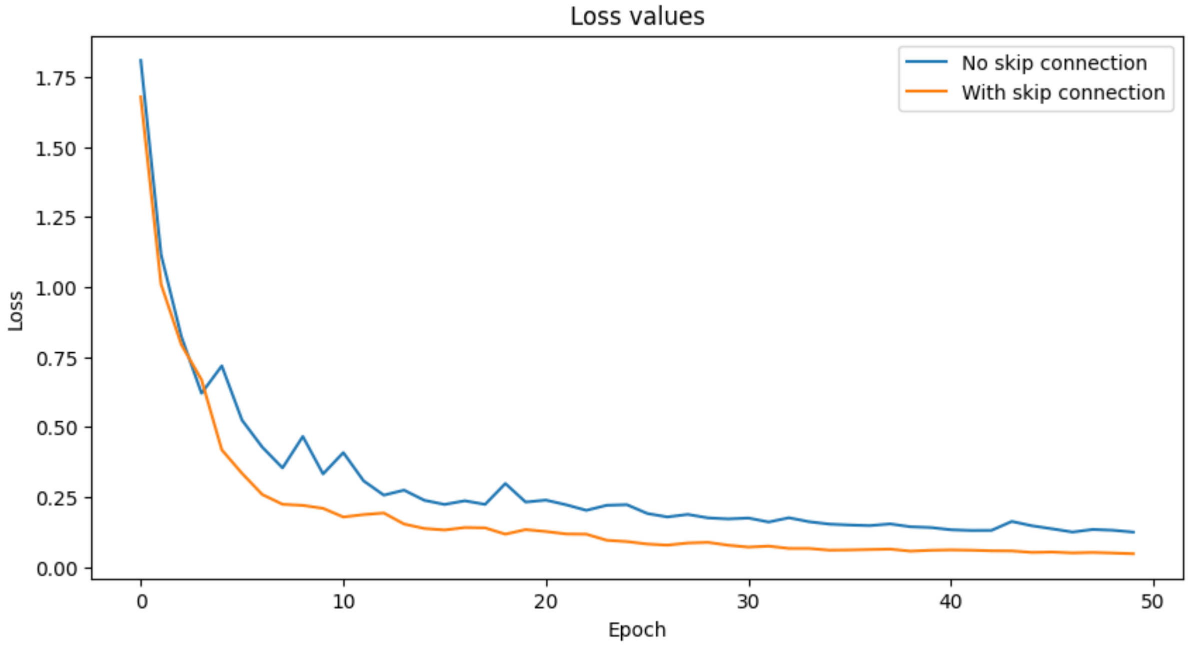
Overall, the code provides a comprehensive implementation of training and evaluation for ResNet-18 based FCNs for image segmentation, comparing the performance of models with and without skip connections.

Outputs:





Analysis:



Based on the results from the two variants of Fully Convolutional Networks (FCN) - with skip connections and without skip connections, following points can be concluded:

- Segmentation Quality: The FCN with skip connections provides a more accurate and detailed segmentation. The “Predicted Mask (skip connection)” closely resembles the “Ground Truth (GT) Mask”, indicating a high degree of accuracy. On the other hand, the “Predicted Mask (no skip connection)” is less accurate, with more noise and less detail.

- Loss over Epochs: The graph depicting loss values over epochs confirms the above observation. The loss for the FCN with skip connections (represented by the orange line) decreases more rapidly and reaches a lower final value compared to the FCN without skip connections (represented by the blue line). This indicates better performance of the FCN with skip connections.

Also based on the parameters calculated, we can make the following conclusions:

Per Pixel Accuracy: The FCN with skip connections has a higher per pixel accuracy (0.9649) compared to the FCN without skip connections (0.9224). This means that the FCN with skip connections is more accurate at predicting the class of each pixel in the image.

Mean Intersection over Union (IoU): A higher mean IoU indicates a better quality of segmentation. The FCN with skip connections has a higher mean IoU (0.3252) compared to the FCN without skip connections (0.2042), indicating that it provides a better quality of segmentation.

In conclusion, the FCN with skip connections outperforms the FCN without skip connections in both per pixel accuracy and mean IoU. Hence we can conclude that skip connections improve the performance of FCNs in image segmentation tasks. They allow the network to leverage information from multiple resolutions, thereby capturing both high-level semantic information and low-level detailed information. This results in more accurate and detailed segmentations.