EC-312 Digital Image Processing
Project Report
Analysis of Histopathological Images

Group Members

Maazin Zaidi	339132
Masab Bin Nasim	345908
Usama Sohail	343679
Sana Fatimah	359065

Supervised by Dr Usman Akram, PhD

Submission Date
June 4, 2023

Abstract

In this project, we were required to perform semantic segmentation on given images of skin carcinoma. Any given image could be classified into up to 12 different segments, namely: Glands (GLD), Inflammation (INF), Hair Follicle (FOL), Hypodermis (HYP), Reticular Dermis (RET), Papillary Dermis (PAP), Epidermis (EPI), Keratin (KER), Background (BKG), Basal Cell Carcinoma (BCC), Squamous Cell Carcinoma (SCC) & Intra-epidermal Carcinoma (IEC).

After segmentation, we were required to classify each image into one of three classes: BCC, SCC and IEC. We were instructed to verify our accuracy by applying a performance metric. In this case, we had to create a confusion matrix and find the accuracy, given by:

$$\text{Accuracy} = \frac{\text{True Positives} + \text{True Negatives} + \text{True Neutrals}}{\text{Total Samples}}$$

The given dataset had a total of 1500 images, from which 1400 were to be used for training and 100 for testing.

Contents

Chapter 1

Introduction

1.1 Libraries Used

The following libraries were used:

- os
- numpy (used as np)
- train_test_split (imported from sklearn.model_selection
- KMeans (imported from sklearn.cluster)
- math
- pandas (used as pd)
- matplotlib.pyplot (used as plt, used during rough work)

1.2 Extraction of Images and Masks

Using the same concept as we used in our lab work to create npy files, we hardcoded the root directory and then accessed each image by opening folders and appending the images to a predefined array of zeros called allImages. We did the same for all the masks.

While doing so, we also appended the corresponding labels - 0 for BCC, 1 for SCC, 2 for IEC. We then used the train_test_split function to split our images, masks and labels into training and testing groups of sizes 1400 and 100 respectively.

Chapter 2

Segmentation

Our main logic behind the segmentation part of the project involved shifting each image to its HSV colour space and extracting its Hue and Saturation values for segmentation. Similar to any other machine learning problem, both segmentation and classification processes were split into training and testing.

2.1 Training

1. Sharpen the current image by a factor found empirically (2).
2. Convert the image to its HSV colour space.
3. Using the given mask, find the hue and saturation values of each of the 12 segments. The segments in the image can be found by extracting the unique colours present in the image and mapping them to a pixel type from a pre initialised dictionary called pixelClasses.
4. Repeat step 2 for each image, updating the average hue and saturation of each segment in a maintained dictionary called "segmentAverages".

2.2 Testing

1. Sharpen the current image by the same factor as in training.
2. Convert the image to its HSV colour space.
3. Append the hue and saturation values to a feature vector.
4. Apply k means clustering on the feature vector.
   (a) Start with 12 clusters.
   (b) For each center provided by the kmeans clustering algorithm, find the closest from the segmentAverages dictionary. Append each to a list.
   (c) Loop through the list to check for any repeats, if there is any value repeating, repeat step 2 and 3 for one less clusters.
   (d) Once we have finalised the kmeans clustering and have the labels, using the classToPixel dictionary, we can then assign colours to all of the labels and finally have a segmented image.

Chapter 3

Classification

The classification algorithm was much easier and simpler compared to the segmentation algorithm.

3.1 Training

1. Identify the class of the mask using the label.
2. For each pixel type in the image (present in the list obtained in step 3 of training during segmentation) increment the number in the classPercentages dictionary. The dictionary maps each pixel type to a 1x3 array. Each element represents the class (BCC, SCC, IEC)
3. After training is done, divide each element of the array by the sum of that respective array and multiplying this value by 100. This provides us with the likelihood of each pixel showing up in a certain type of class.

3.2 Testing

1. Create an empty list called classProbability of size 1x3
2. Loop through the list of the segments present in the image obtained during segmentation.
   (a) Add the classPercentages of that pixel to the classProbability array respectively.
   (b) Do this for each item (pixel type) in the list
3. Divide this list by the total number of segments to obtain the likelihood of the image belonging to each of the classes.
4. Find the maximum percentage and append this to an array called maxProbabilities. Also append the ground truth label to the groundTruths array.

3.3 Exporting to Excel

Once the testing loop is finished, we create a pandas dataframe which contains the maximum probabilites and the ground truths. We then export this to excel so that we can import it in a different PyCharm file to apply our Performance Metrics

Chapter 4

Performance Metrics

4.1 Confusion Matrix

In order to create a confusion matrix and calculate the accuracy, we will create another PyCharm file and import our excel file, called "Results.xlsx".

Now, using the sklearn.metrics library, we can create a confusion matrix and calculate the accuracy of our testing images. The confusion matrix is given below:

		PREDICTED		
		BCC	**SCC**	**IEC**
ACTUAL	**BCC**	22	6	6
	SCC	20	10	6
	IEC	14	9	7

Finally, we can calculate the accuracy by summing up the diagonals and diving that value by the total number of samples.

$$\text{accuracy} = \frac{22 + 10 + 7}{22 + 6 + 6 + 20 + 10 + 6 + 14 + 9 + 7} = \frac{39}{100} = 0.39$$

This shows us that our classification algorithm is 39% accurate.

Chapter 5

Flow Diagram

The flow diagram was made on a website called miro.com and is shown below.

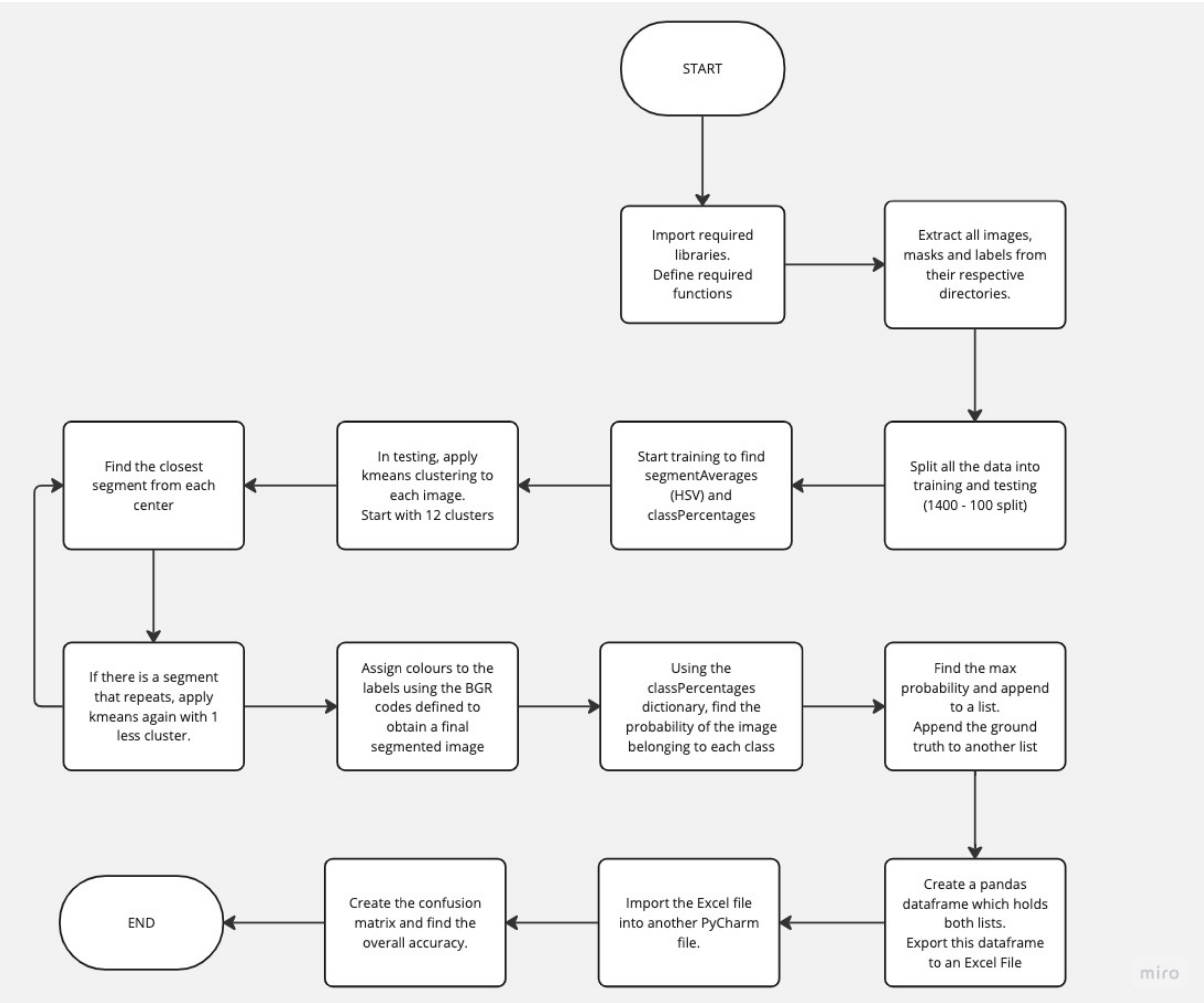

Figure 5.1: Flow Diagram

Chapter 6

Sample Outputs

The images below show the ground truths along with the segmented images.

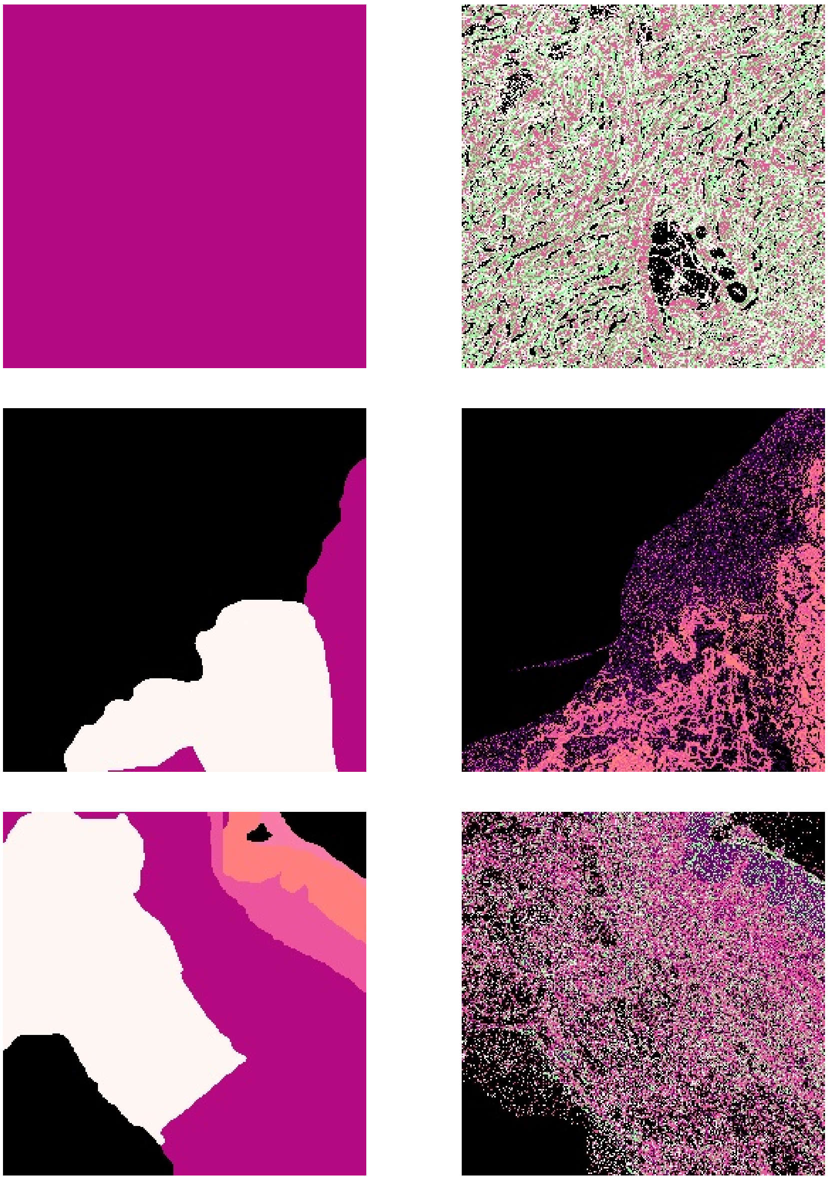

Figure 6.1: Ground Truths & Segmented Images

Chapter 7

Conclusion

In this project, our primary objective was to perform semantic segmentation and classification of skin carcinoma images with precision and accuracy. We created a confusion matrix to assess the accuracy of our classification. The dataset provided consisted of 1500 images, with 1400 allocated for training and 100 for testing.

While we made significant efforts to achieve accurate segmentation and classification, our results, as indicated by the confusion matrix, fell short of our desired level of accuracy. This suggests that there is room for improvement in our approach. One possible area for enhancement lies in refining the segmentation algorithm. By exploring advanced techniques or considering state-of-the-art approaches, we could potentially achieve better segmentation results, leading to improved classification accuracy.

Despite not attaining the desired accuracy levels in this project, it serves as a valuable learning experience, highlighting the importance of continuous improvement in the field of image analysis and medical diagnostics. Future work could focus on exploring innovative segmentation algorithms, optimizing hyperparameters, or incorporating additional features to enhance the performance of the system. With ongoing advancements in the field, there is a promising opportunity to refine the segmentation and classification processes and improve overall accuracy in skin carcinoma analysis.