Image Segmentation

Hrudaya Jinna

***Abstract*— In the field of computer vision, Image segmentation is the most significant task. In the case of SLIC, we cluster and segment comparable pixels, which may subsequently be used for further image processing. In addition, we segment super pixels in regions where we see similar pixels. In this work we discuss about different clustering algorithms as well as neural network analysis. Implemented the multiple threshold technique, followed by the K-means clustering method, SLIC and talked about comparison and then examine the outcomes of SLIC and K-means. In the last sections we will talk about multilayer feed forward neural networks and CNNs in relation to handwritten character recognition. In this final section, we will contrast the results obtained from a multilayer neural network with those obtained from a CNN.**

I. INTRODUCTION

An image is composed of pixels, each of which is segmented into a finite number of regions based on the similar characteristics that each image have. These regions are connected to one another because in an image all of the grey levels will be similar, and we will refer to these connected regions as image segments. For instance, if we take a pizza and think of it as a picture, we can break it up into eight separate pieces, just as an image may be broken up into a variety of different regions. Therefore, segmentation consists of nothing more than assigning some labels in order to identify certain aspects of an item or objects. Even if we are watching a cricket match on television, we will find ourselves looking at a lot of different images one after the other. And the brain remembers these images and tells us that we are watching a cricket game, which makes the segment significant in all of the circumstances where it is applied to real life.

In its most basic form, segmentation does not require any prior information to be performed. After we have segmented the image, the next step is to cluster it, which means to group the pixels as we normally would use DBSCAN (Density-based spatial clustering of applications with noise). In reality, there are many different clustering methods, and one of these is called DBSCAN. The usage of segmentation can be beneficial for a variety of reasons, including the enhancement of efficiency.

There are a variety of approaches to segmentation. Different types of segmentation include cluster-based segmentation, edge-based segmentation, threshold-based segmentation, and region-based segmentation. The subdivision of watersheds, etc. The use case and the work that needs to be implemented are taken into consideration in the choice of which method to adopt.

II. MULTIPLE-THRESHOLD TECHNIQUE

The Multi-Otsu threshold [1] is used to separate the pixels of an input image into a number of distinct classes, each of which is obtained based on the intensity of the gray levels contained within the image. The number of classes that are produced by Multi-Otsu is directly proportional to the number of thresholds that are calculated.

```
[ 79 125 170]
```

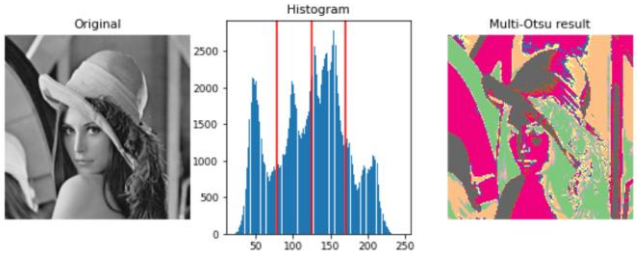

Fig: Histogram and Multi Otsu for three thresholds 79,125,170.

Whereas multi-Otsu is identical to Otsu, with the exception that we have the ability to define the number of thresholds. If we have two thresholds, then the image can be divided into three different sections and so on. Here, we've used Lenna as the Input Image. For the picture, we plotted grayscale values from 0 to 255 and did 3, 4, 5 thresholds, and the results are shown.

```
[ 74 113 145 180]
```

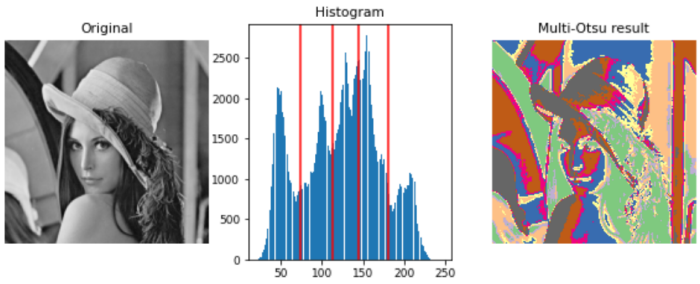

Fig: Histogram and Multi Ostu for four threshholds 74,113,145,180.

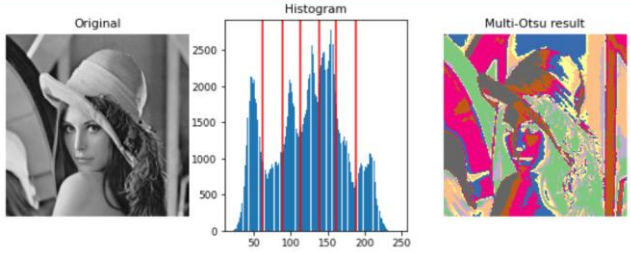

Fig: Histogram and Multi Otsu for five thresholds

III. K-MEANS CLUSTERING METHOD

K-means clustering is a clustering method for grouping objects into K groups based on the similar features and pixels. It segments the image from the background. It does by clustering the image as K-clusters, which are also known as K-centroids. In this algorithm first it creates K points in a feature space. In order to assign each pixel in an image to its nearest center, the distance between the center and each pixel is minimized. All of the pixels in the image are given to the centers that are closest to them by finding the shortest path between them and the centers. Most of the time, we use Euclidean distance to find the distance between them. Then the centers keep on updating by performing the means of all the pixels. Usually we keep assign to the centers and shift them until there are no shifts possible. However, our code will terminate these algorithms when they reach a particular threshold.

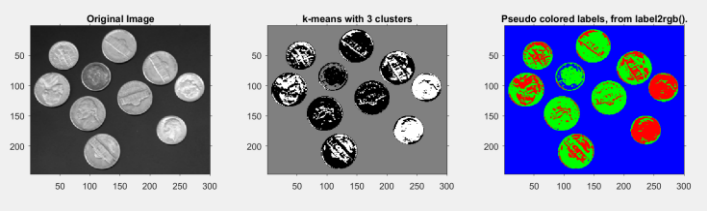

Fig: k-means with 3 clusters and Pseudo colored labels for the image.

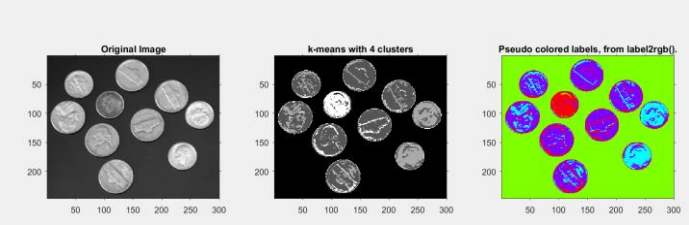

Fig: k-means with 4 clusters and Pseudo colored labels for the image

Here we have taken coins.png as an input and we performed K-means with 3, 4,5clusters in the figures and then plot the pseudo colored labels for all the clusters. In order to more clearly differentiate the classes, we have assigned each one a distinct color as part of our allocation process so that the user can easily recognize the differences between them.

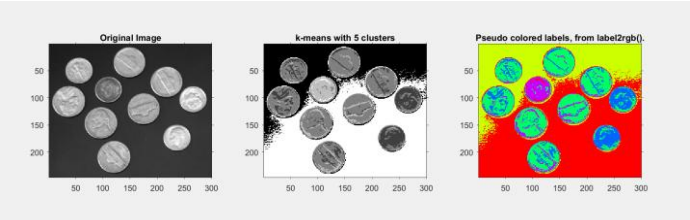

Fig: k-means with 4 clusters and Pseudo colored labels for the image

IV. SLIC IMPLEMENTATION

In his paper, Achanta [2] suggested using the best Super Pixels Approach instead of the methods that were already in use. The authors came up with an algorithm called SLIC, which stands for Simple Linear Iterative Clustering Super Pixels. This algorithm works best with both grayscale and color pictures. Here, I'd like to talk about the color picture shown in the figure below.

RGB2LAB helps the image of type uint8 to convert it to double and divides it by 255 to ensure that the RGB values are in the range 0-1 and that the modified image can have the appropriate negative values for a and b. (If the image is left in its original format of uint8, MATLAB will adjust the numbers such that they fall inside the range 0-255.)As a result, we now have a color space that can be said to be nominally perceptually uniform. As a result, we are able to measure the differences between colors by employing the Euclidean distance that exists between color coordinates. After conversion, the image has a mirrored appearance. To avoid losing too much information, we might switch to using signed shorts.

Fig: Original Image toysflash.png

In order to carry out the SLIC procedure, it is necessary for us to set four parameters.

1. Image- is the image that is going to be segmented.
2. k- Is the number of super pixels that you want to use. Take note that this is just a placeholder value; the real number of super pixels that are produced will almost always be higher, especially if parameter m is set to a low value.
3. W- Weighting factor between disparities in color and location in space. It's helpful to have values in the range of roughly 5 to 40. When you want Super Pixels to have more regular and smoother shapes, use a large number to reinforce them. You might try a value of 10 as a starting point.
4. Radius - Regions that are morphologically smaller than this are merged with the regions that are close to them. You could try the value 1 or 1.

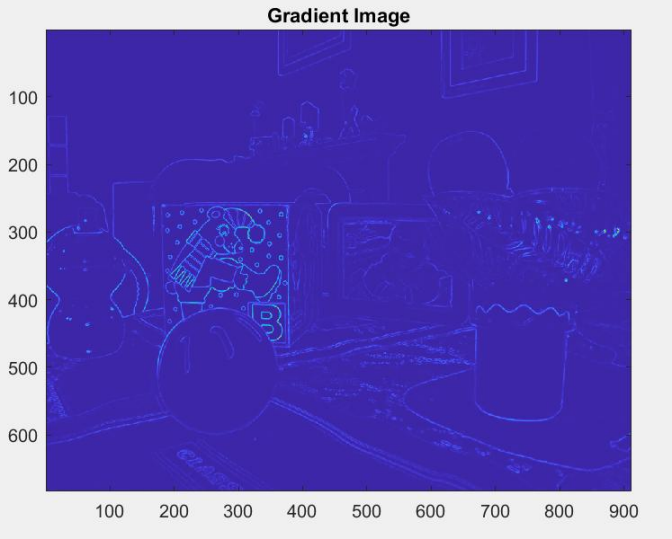

Fig: Gradient Image

Fig: Image when super pixels are 300

Fig: Image when super pixels are 900

Fig: Image when super pixels are 1500

We have taken an input image 'toysflash.png' and perform super pixels with 300,900,1500 respectively as indicated in the images above with m set to '15' and the number of iterations set to '5'.

The distance between each pixel in the sub image and the center of the cluster is calculated. Cluster center for each pixel in the sub image, and take the value with the biggest difference as the 'edge distance'.

In comparision to SLIC with K-means and multi threshold the image in SLIC has better clustering when compared to other techniques.

V. COMPARISION OF SLIC AND K-MEANS RESULTS

Both the k-means algorithm and the simple linear iterative clustering are image clustering algorithms.

The performance of the K-means algorithm can be improved by considering the color of each pixel and its position. Then, a color image is represented in a 5D space, which could be represented as (R, G, B, X, Y).

K-means and SLIC first performs clusters and then segment the regions based on the clusters. So we need to choose K first before performing the algorithm.

The major advantage is both algorithms are fast and easy to implement.

The drawbacks of K-means clustering are we need to select initial centers which are quite complex otherwise results in bad clusters and influence the run time of the algorithm. And also the clusters have spherical shapes only.

The drawbacks of SLIC are it is necessary for us to recalculate the positions of the centroids such that they are moved to the points on the graph that has the lowest gradients. Since it does not enforce connectivity, an additional step is required in order to check that each pixel has been reassigned correctly.

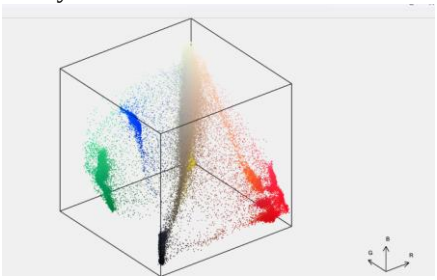

Fig: Image of color classes for the input Image

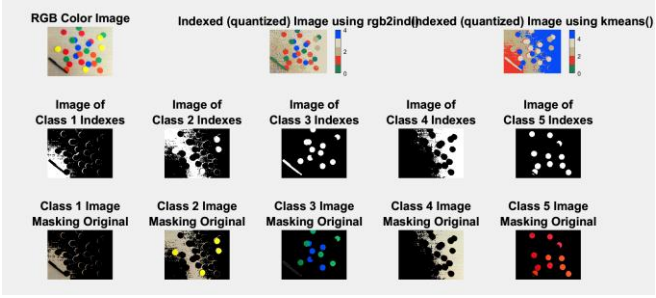

Fig: K-means for the colored image for k=5 where RGB color image ,Image using K-means(), Image of Class 1 to 5 Indexes, Class 1 to 5 Masking Original Images and K-means Images

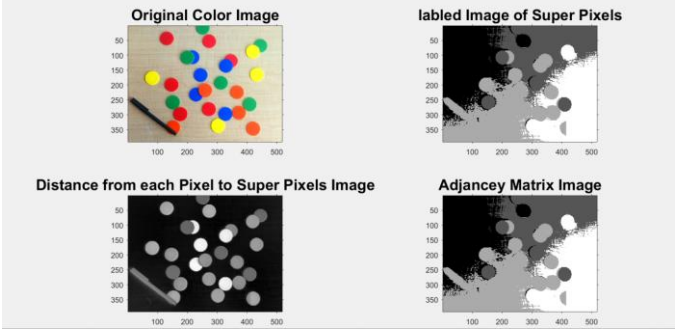

Fig: Images of Original Color Image 'ColoredChips.png',

Labeled image of Super Pixels, Distance from each Pixel to Super Pixel Image, Adjacency Matrix Image

In the above GUI we can observe that all the circled balls and pen are clearly clustered.

Computation Cost	SLIC	K-Means
Clustering	Least complex	High Complex
Performance	Slower	Higher
Choosing K	Good for super pixel value=900	Good for k=5
Time Complexity	Slow O(Iterations*Clusters*Instances*Dimensions	Fast O(Instances)

Table 01: Comparision of SLIC and K-Means Clustering Algorithms

VI. MULTI LAYER FEED FORWARD NEURAL NETWORK

A feed-forward neural network, also known as a FNN, is a type of neural network that has several layers of neurons in which each neuron in one layer is connected to all of the neurons in the following layer.

We intend to take into account images of handwritten numerals and construct a neural network with the purpose of classifying these pictures as "0," "1," etc., up to "9." In other words, we want to develop a model that has the inputs "0," "1,"..., "9" and the output labels "0," "1,"..., "9." These values will be acquired from the image. The image is a matrix of pixels with a specific size (28 by 28), and the grayscale is represented by values that range from 0 to 255, with 0 representing black and 255 representing white.

```
        _, predicted = torch.max(outputs.data, 1)
        n_samples += labels.size(0)
        n_correct += (predicted == labels).sum().item()

    acc = 100.0 * n_correct / n_samples
    print(f'Accuracy of the network on the 10000 test images: {acc} %')
```

```
Epoch [1/2], Step [100/600], Loss: 0.3376
Epoch [1/2], Step [200/600], Loss: 0.3198
Epoch [1/2], Step [300/600], Loss: 0.2100
Epoch [1/2], Step [400/600], Loss: 0.1439
Epoch [1/2], Step [500/600], Loss: 0.2643
Epoch [1/2], Step [600/600], Loss: 0.1198
Epoch [2/2], Step [100/600], Loss: 0.1288
Epoch [2/2], Step [200/600], Loss: 0.1128
Epoch [2/2], Step [300/600], Loss: 0.0613
Epoch [2/2], Step [400/600], Loss: 0.1235
Epoch [2/2], Step [500/600], Loss: 0.0511
Epoch [2/2], Step [600/600], Loss: 0.0786
Accuracy of the network on the 10000 test images: 96.97 %
```

Fig: Output for Multi Layer feed forward Neural Network with Accuracy 96.97%

Parameter	**Value**
Input Layer	
No of Input Layers	4
No of input Neurons	784
Activation Function	RELU
Hidden layer	
No of Hidden Layers	2
No of Hidden Neurons	500
Activation Function	RELU
Output Layer	
No of Output Layers	1
No of Output Neurons	10 neurons for 10 classes
Activation Functions	SOFTMAX
Learning rate	0.001
Learning rule	Momentum
Epochs	10
Accuracy Rate	96.99

Table 02: The results of different parameters that are calculated in multi layer feed forward neural network.

We will be using 4 input layer with activation function Relu and 2 hidden layers with Relu Activation Function with output SoftMax Activation Function. The Accuracy for the algorithm is 96.99% which is shown in the above table.

VII. CNN FOR HANDWRITTEN CHARACTER RECOGNITION

An example of a deep learning algorithm is the convolutional neural network where in simple terms deep refers to the number of layers that are present. In CNN it reuses the weights of the neurons.

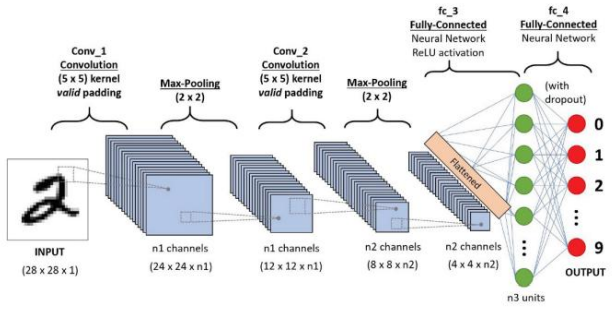

Fig [3]: A CNN classification for handwritten digit classification

The input image that we will be using is a two-dimensional Representation. The window of an image is going to be used. A feature is an abstract concept that can be retrieved, and a filter will be an example of a local feature in which the size will be reduced by us. We are using a window and applying a convolution at the location where we will be calling the feature maps. After that, we will be sub-sampling in the locations where the size reduces to obtain Pooled feature maps.

Data Set: Here we consider MNIST data set with images of 50000 with each input size 28X28.

Fig:Data Set for the first 25 images.

Parameter	Value
Input Image Size	28X28 for 50000 images
Input Layer	
No of Input Layers	2
1st Layer	Conv2d layer which convolves the image using 32 filters using 32 filters each of size (3*3)
2nd layer	MaxPooling2D layer which picks the max value out of a matrix of size (3*3
Activation Fuction	RELU
No of Input Neurons	784
Hidden layer	
1st hidden layer	size 64
Activation Function	RELU
2nd hidden layer	Size 32
Activation Function	RELU
No of Hidden Neurons	2048
Output Layer	
No of Output Neurons	10 neurons for 10 classes
Activation Function	Softmax Function
Optimizer	adam
Learning Constant	0.01
Learning rule	Momentum
Epochs	5
CPU	50 sec
Accuracy Rate	98.98
Kernel Size for all convolutional layers	3x3 a stride of 1

Table 03: The results of different parameters that are calculated for CNN.

	precision	recall	f1-score	support
0	0.97	1.00	0.98	980
1	0.98	1.00	0.99	1135
2	0.99	0.98	0.98	1032
3	1.00	0.98	0.99	1010
4	0.99	0.98	0.99	982
5	0.98	0.99	0.99	892
6	0.99	0.97	0.98	958
7	0.96	0.99	0.97	1028
8	0.98	0.98	0.98	974
9	0.99	0.96	0.97	1009
accuracy			0.98	10000
macro avg	0.98	0.98	0.98	10000
weighted avg	0.98	0.98	0.98	10000

Fig:The Output Results with 98% accuracy.

For the MNIST data set which we have used we got an Accuracy of 98.98%.

VIII. COMPARISON OF MULTI LAYER FEED FORWARD NETWORK WITH CNN

When we performed same data set and performed handwritten character recognition we found that CNN found better results when compared to feed forward neural network because the accuracy is 96.99% for multi layer feed forward network whereas for CNN its 98.99%.When it comes to finding a pattern in the data [3, 6] that is provided, the FNN form has shown itself to be more durable, as well as less sensitive, in comparison to the CNN form. Additionally, it has shown itself to generally perform at least slightly better.

When we look at the illustration below and read what it says, we see the number 8. Whereas, according to the algorithm, it is trained to produce the result of "9," which we obtained.

```
plt.imshow(X_train[80], cmap="gray")
print(y_train[80]);
```

```
9
```

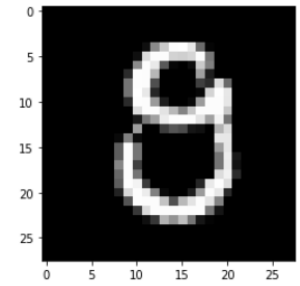

Fig: The difference between visual understanding (‘8’) and the recognition of handwritten digit is (‘9’) different.

One problem for both the algorithms is images are that they usually have resolutions that are higher than 100x100 pixels. This is too high for most modern hardware to handle. To deal with this, the images are often shrunk down to a more manageable size, where the layout and structure of the characters can still be seen but most of the individual characters can't.

The results for CNN and feed forward neural network are different for certain values of input.

IX. CONCLUSION

Here, we have used clustering and neural networks to implement the image segmentation algorithm. Also, analyzed clustering algorithms like K-means and SLIC and observed at the pros and cons of each one. We have also performed k=3, 4, 5 to the Otsu multi threshold algorithm. Then, handwritten character recognition was done on multi-layer feed forward and CNN and different parameters were used to compare the results.

REFERENCES

[1] Liao, Ping-Sung et al. “A Fast Algorithm for Multilevel Thresholding.” *J. Inf. Sci. Eng.* 17 (2001): 713-727.

[2] https://pubmed.ncbi.nlm.nih.gov/22641706/

[3] https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53