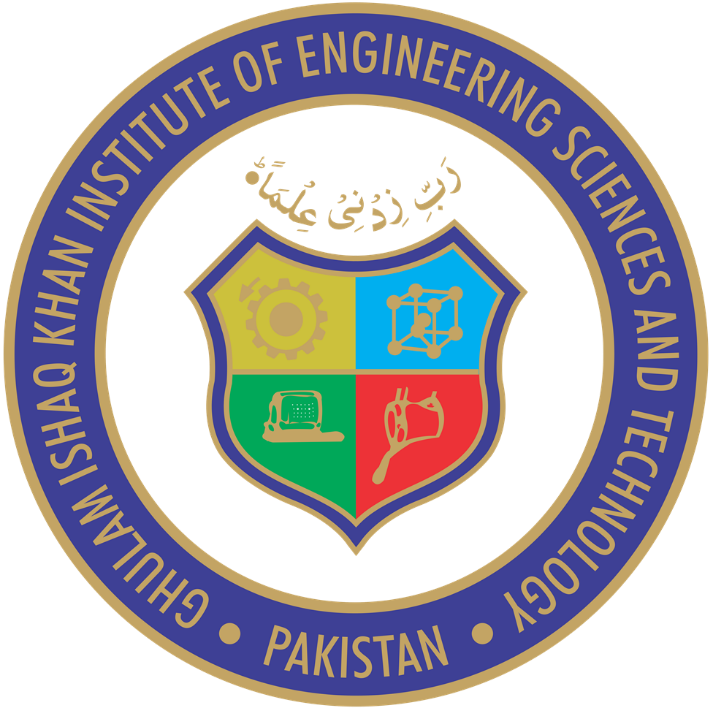
**CS-221**

**Data Structures and Algorithms**

**Semester Project Report**



**Members:**

* Mohammad Omar Khan **(2021305)**
* Muhammad Umer Tayyab **(2021306)**
* Muhammad Abdullah **(2021317)**
* Zaid Dandia **(2021719)**

**Function to calculate the correlation matrix**

An empty correlation matrix is declared with a size of N x N (where N is the number of rows/records). Then 2 outer loops are run for the number of rows/records and an inner loop for the number of columns. Using this loop the Pearson correlation is calculated for every element with every other element of the dataset and stored into the correlation matrix.

**Function to calculate the discretized similarity matrix**

Following the correlation matrix, the similarity matrix is calculated. The mean of every row is calculated using the built in numpy mean() function, which comes out to be a numpy array of size N. Each value in the correlation matrix is compared against the mean of that value’s column; if found to be less than the mean, it is changed to 1 in the similarity matrix, or else it is changed to 0. This matrix of 0’s and 1’s is our discretized similarity matrix. It is then displayed and saved using Open CV’s imshow() function and imwrite() function respectively.

**Function to show the discretized image in color (green)**

Consequently, each value in the correlation matrix is then multiplied by 255 to show a shade of color. An array of size N x N is initialized with an array of length 3 in each cell of the array to contain the value of the colors, called the image. Then a loop is run for the whole length of this new array and in the length 3 array’s first index (which corresponds to green) the corresponding value of the correlation matrix is copied. This is then again displayed and saved using Open CV’s imshow() function and imwrite() function respectively.

**Task 1:**

First, the desired file is read using the open() function, then using Numpy it is reshaped (using the reshape() function) into the dimensions given in the rows and columns in the starting of the file. Functions to calculate the correlation matrix, discretized similarity matrix, and show the discretized image in green are run.

**Task 2:**

A copy of the original data’s numpy array is made and shuffled using the shuffle() function of Numpy. After this the function to calculate correlation matrix, discretized matrix, and show discretized image in color are run for this shuffled/permuted data.

Afterwards, a signature array is declared with a size of N. The signature for each row is calculated by multiplying the sum of the values in the row with the mean of the values in the row. After this, the permuted data is sorted in accordance with the signature value. This is done using bubble sort where the condition of comparison is run on the signature array and if found to be true, that row of the signature array is shifted as well as the corresponding row in the permuted data matrix. Finally, functions to calculate correlation matrix, discretized similarity matrix, and show discretized image in color are run.

**Task 3:**

Finally, a weighted graph is calculated. This is done by getting an input from the user of a threshold value. Once entered, the edges having a value less than it are removed (made equal to 0). The diagonal value is also changed to 0, this is because every diagonal will try to make a neighbor with itself, and this will disrupt the clustering technique. After this the sum of the rows is computed and subsequently printed. The highest weight node is calculated, and its adjacent / neighboring nodes are made into a cluster. The neighboring nodes and the highest weight node are made 0, as this cluster has been extracted. All this while a count of neighbors and number of clusters is kept and incremented whenever a new one of either is found.

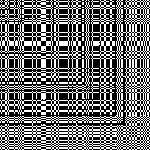
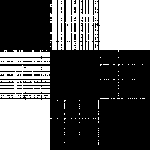
**Comparison of Task 2 and 3:**

Both tasks 2 and 3 are regarding image recovery, however task 2 uses Signature technique while task 3 uses clustering technique. The signature in task 2 is calculated by multiplying the mean and sum of the row and sorting it by the value of its signature and then displaying. The cluster in task 3 is calculated by computing the largest weighted node by the use of threshold value and its adjacent nodes made 0.

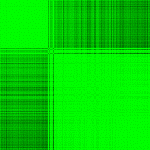
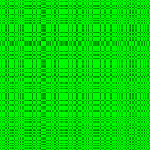
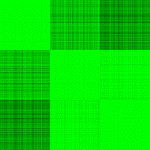
**Images:**

The following images are form the Iris Dataset

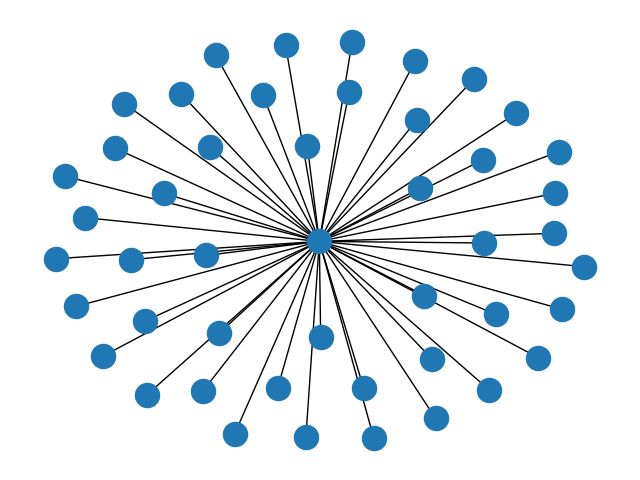
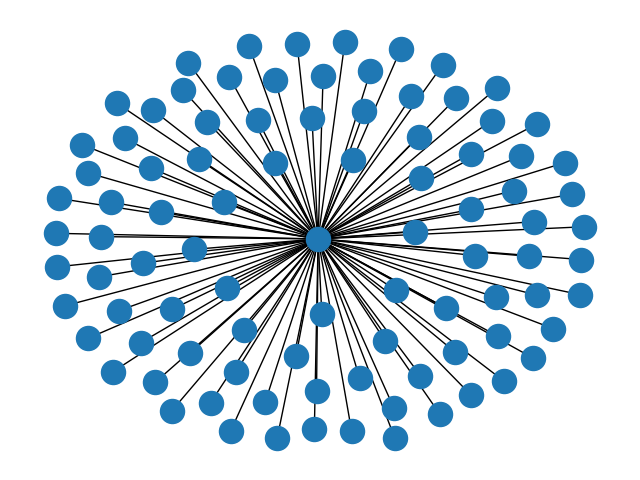
**Black & White Recovery Black & White Permuted Black & White Discretized**

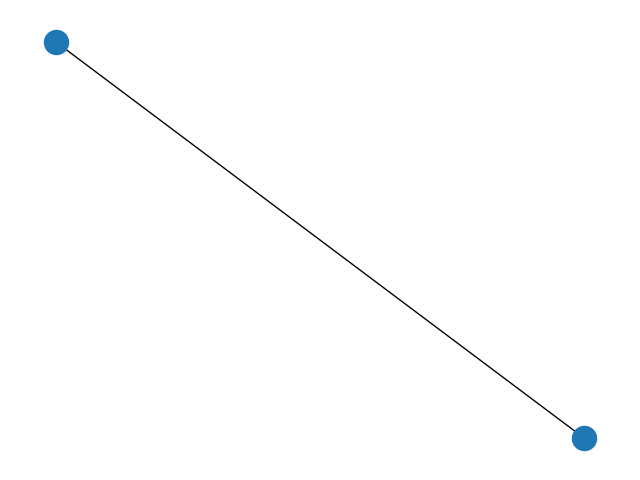
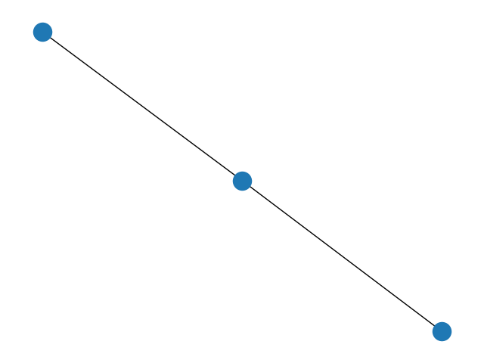
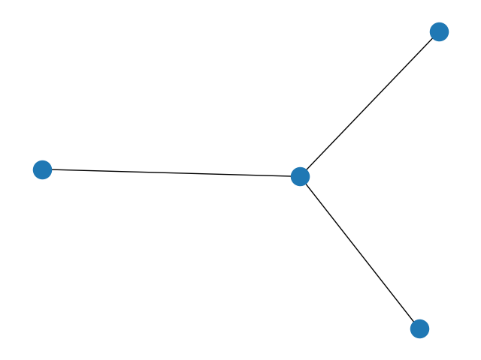
  

**Green Recovery Green Permutation Green Discretized**

The following are clusters generated from a threshold value of 0.98:





**Work Distribution:**

Mohammad Omar Khan (2021305): Task 1, Task 2, Task 4, GUI

Mohammad Umer Tayyab (2021306): Task 1, Task 2, Task 4, GUI

Muhammad Abdullah (2021317): Task1, Task2, Task3

Zaid Dandia (2021719): Task1, Task2, Task3