309551101郭育麟

HW06

1. Code with detailed explanations

***Part1:***

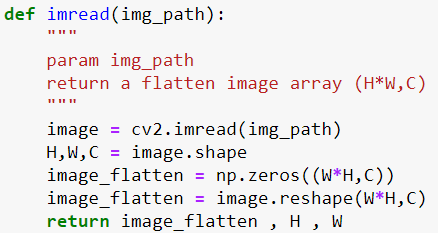
In this part , we need to make GIF images to show the clustering procedure of mine kernel k-means and spectral clustering(Ratio cut and Normalized cut) . And for every part’s spectral clustering , I also plot the eigenspace of graph Laplacian (Part4) . The reason why using ratio cut and normalized cut is avoid extremely solution , for example , there are only one datapoint in the subgraph , and all the others datapoint are in another subgraph , it can avoid this situation by using ratio cut or normalized cut .

**Kernel k means**

Following the steps

I :

load the image .



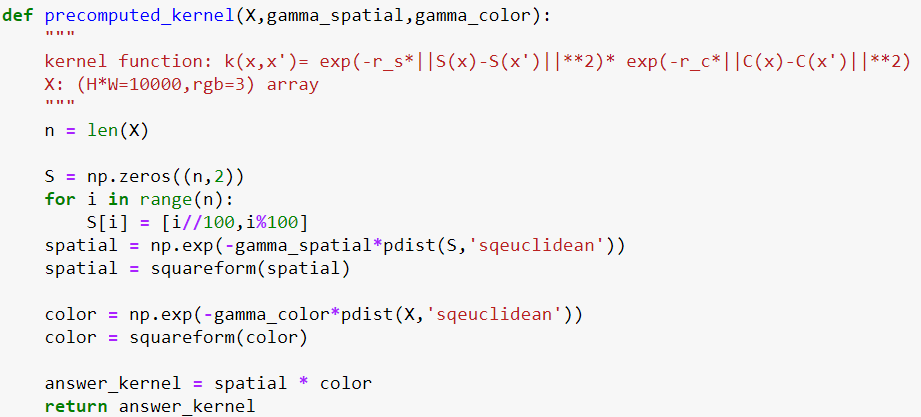
This function will load picture and return image(flatten) and the image’s height and width

II :

choose parameters “k” which represent “k-clustering” and the initialization of k-means clustering used in kernel k-means ( In this part I choose “random” , other option is left to part3 ) and gamma\_spatial and gamma\_color.

III :

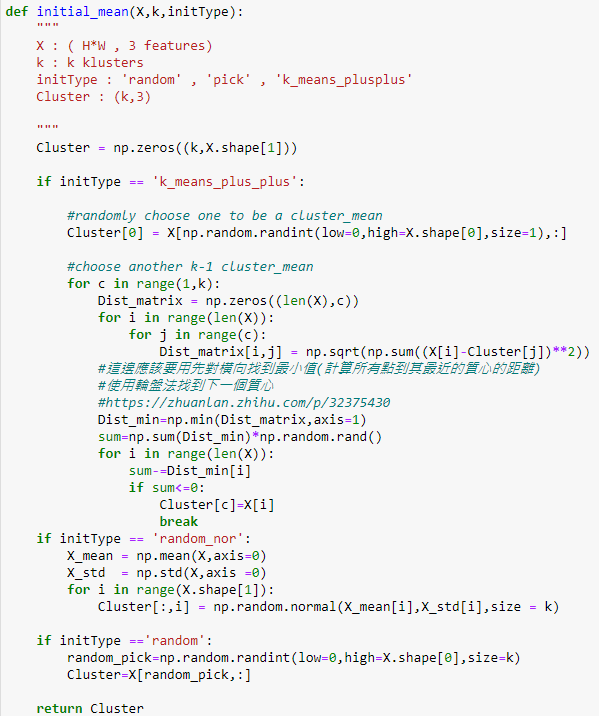
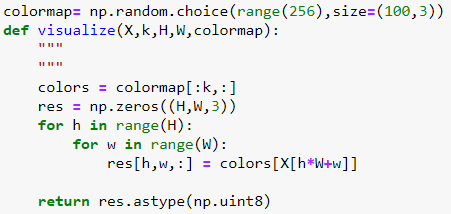
Calculate kernel function



The kernel function is

IV :

Used k-means algorithm .



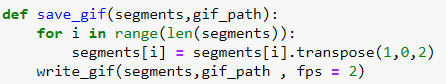
First , we need to use initial\_mean function to calculate initial k clustering location . In this part , I use random to create k clustering location . Then we need to calculate every pairs distance and find the minimum distance to classified category . For example , given a datapoint , we need to compute the distance between this datapoint and the current k clustering center , and choose the smallest distance to represent that this datapoint is classified the specific category among k . This step will continue n times , n represent n datapoints .

Second , we need to update new k clustering location . we find the same category datapoint in the previous step and add them and also do normalization so that we can get new k clustering location and difference between new clustering location and old clustering location .

Third , we use visualize function to color the same category datapoint in the current clustering state and return it . So that we could use plt.show to plot the picture . Furthermore , after I use visualize function , I append it to a list to make GIF .

V:

Make GIF images



In this function , I put segments(every output from function visualize) and convert to RGB so that can use write\_gif (from array2gif import write\_gif) to make GIF .

Main:



See result [[link](#part1_kmeans_result)]

**Spectral clustering – Ratio cut**

Unnormalized Laplacian L = D-W serve in the approximation of the minimization of RatioCut

Following the steps

I:

Same as k-means algorithm I [[link]](#StepI)

II:

Same as k-means algorithm II [[link]](#StepII)

III:

Same as k-means algorithm III [[link]](#StepIII)

IV:

Compute Laplacian matrix

D can seen as a degree matrix

Then use np.linalg.eig for eigenvalue decomposition for Laplacian matrix

V:

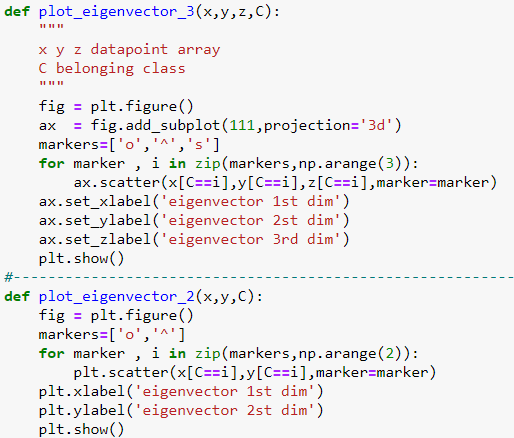
Sorting the eigenvalue and get the 2nd and 3rd (1st eigenvalue is 0 represent fully connected ) eigenvalue and its corresponding eigenvector . Use these eigenvector to execute k\_means funtcion .

(k-means algorithm is same as k-means algorithm IV [[link]](#StepIV))

VI:

Make GIF images and plot eigenvector

Make GIF images is same as k-mean algorithm V [[link]](#StepV)



In plot\_eigenvector\_3 , x , y , z represent eigenvector of the graph Laplacian , because I want to do 3-clustering , so pick x,y,z . if I only want to do 2-clustering , only pick x, y .

Main:



See result [[link](#part1_ratio_result)]

**Spectral clustering – Normalized cut**

Normalized Laplacian serve in the approximation of the minimization of Normalized Cut .

This is very similar as ratio cut , the main difference between ratio cut and normalized cut is , this means normalization . And also , the each row eigenvector also do normalization . Others are same as ratio cut .

Main:



See result [[link](#part1_normalized_result)]

***Part2:***

Try more clusters

Only change parameters “k”

K-means algorithm [[link](#part2_kmeans_result)]

Spectral clustering ratio cut [[link](#part2_ratio_result)]

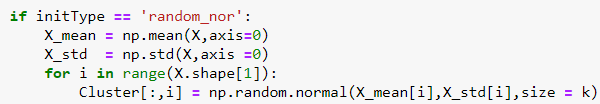
Spectral clustering normalized cut [[link](#part2_normalized_result)]

***Part3:***

Try different initial of kernel k-means method

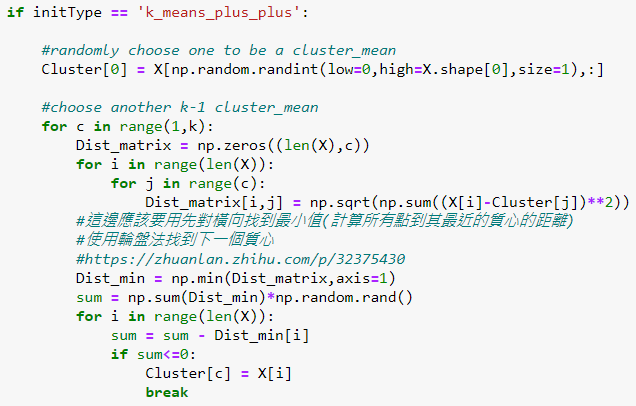
There are two extra method to initialize the kernel k-means

(1) Random-normalized



In this random\_normalized , I calculate the whole data point mean and variance , and random generate k data point as a clustering location by given mean and variance , I believe that this could convergence faster .

(2) k-means++



In k-means++ , random choose a datapoint as a clustering location center , and compute every datapoint to its distance , then if have more than 2 clustering center , find the minimum distance between a datapoint to some clustering center . Last , using roulette wheel section , choose next clustering center , this for-loop will continue k (the number of clustering) times .

Result [[link](#Part3_result)]

1. Experiments settings and results & discussion

**Part1**

**K-means algorithm**

|  |  |  |
| --- | --- | --- |
|  | Image1 | Image2 |
| Initial |  |  |
| Result |  |  |
| GIF | “image1\_2Clusters\_random\_kmeans” | image2\_2Clusters\_random\_kmeans |

We can see that the result is not ideal , it can not present the initial graph feature , so I think 2-clustering is not enough , maybe more clustering .

**Spectral clustering ratio cut**

|  |  |  |
| --- | --- | --- |
|  | Image1 | Image2 |
| Result |  |  |
| Eigenvector |  |  |
| GIF | image1\_2Clusters\_random\_ratio | image2\_2Clusters\_random\_ratio |

We can see that this result is better than k-means algorithm , it can see approximately contour . And I think the eigenvector of graph Laplacian can have the same coordinates within the same cluster , it separates two category from a certain threshold .

**Spectral clustering normalized cut**

|  |  |  |
| --- | --- | --- |
|  | Image1 | Image2 |
| Result |  |  |
| Eigenvector |  |  |
| GIF | image1\_2Clusters\_random\_Normalized | image2\_2Clusters\_random\_Normalized |

And by using normalized cut , the result seems like ratio cut . However , the eigenvector can see the difference better than ratio cut .

**Part2:**

**K-means algorithm**

|  |  |  |
| --- | --- | --- |
| Image1 | K=3 | K=4 |
|  |  |  |
|  | image1\_3Clusters\_random\_kmeans | image1\_4Clusters\_random\_kmeans |
|  | K=5 | K=6 |
| Result |  |  |
|  | image1\_5Clusters\_random\_kmeans | image1\_6Clusters\_random\_kmeans |

This result shows that maybe should not use more clustering . The more clustering I use , the graph is more complexity . So it might use 3-clustering in this case .

|  |  |  |
| --- | --- | --- |
| Image2 | K=3 | K=4 |
|  |  |  |
|  | image2\_3Clusters\_random\_kmeans | image2\_4Clusters\_random\_kmeans |
|  | K=5 | K=6 |
|  |  |  |
|  | image2\_5Clusters\_random\_kmeans | image2\_6Clusters\_random\_kmeans |

This result shows the same situation to image1 , I think that maybe use 3-clusering is the best choice .

**Spectral clustering ratio cut**

|  |  |  |
| --- | --- | --- |
| Image1 | K=3 | K=4 |
|  |  |  |
|  | image1\_3Clusters\_random\_ratio | image1\_4Clusters\_random\_ratio |
|  | K=5 | K=6 |
|  |  |  |
|  | image1\_5Clusters\_random\_ratio | image1\_6Clusters\_random\_ratio |

By applying ratio cut , the result shows that it do the better performance than k-means . However , it still has the same problem , the more clustering I use , the graph is more complexity . And can see that when I choose k=5 , some clustering only has digits numbers , it reveal that these clustering subgraph are useless .

|  |  |  |
| --- | --- | --- |
| Image2 | K=3 | K=4 |
|  |  |  |
|  | image2\_3Clusters\_random\_ratio | image2\_4Clusters\_random\_ratio |
|  | K=5 | K=6 |
|  |  |  |
|  | image2\_5Clusters\_random\_ratio | image2\_6Clusters\_random\_ratio |

Also it has not the situation like image1 in more clustering , but it still make the picture more unclear .

**Spectral clustering normalized cut**

|  |  |  |
| --- | --- | --- |
| Image1 | K=3 | K=4 |
|  |  |  |
|  | image1\_3Clusters\_random\_Normalized | image1\_4Clusters\_random\_Normalized |
|  | K=5 | K=6 |
|  |  |  |
|  | image1\_5Clusters\_random\_Normalized | image1\_6Clusters\_random\_Normalized |

By testing with many clustering and many method , I can conclude that the best choice is k = 3

|  |  |  |
| --- | --- | --- |
| Image2 | K=3 | K=4 |
|  |  |  |
|  | Image2\_3Clusters\_random\_Normalized | Image2\_4Clusters\_random\_Normalized |
|  | K=5 | K=6 |
|  |  |  |
|  | Image2\_5Clusters\_random\_Normalized | Image2\_6Clusters\_random\_Normalized |

***Part3:***

**k-means**

|  |  |  |
| --- | --- | --- |
| k = 3 | Random-normalized | k-means++ |
| Image1 |  |  |
|  | image1\_3Clusters\_random\_nor\_kmeans | image1\_3Clusters\_k\_means\_plus\_plus\_kmeans |
| Image2 |  |  |
|  | image2\_3Clusters\_random\_nor\_kmeans | image2\_3Clusters\_k\_means\_plus\_plus\_kmeans |

The google says that it can faster convergence or benefit to convergence by adopting different kernel k-means , but in my practice , the k-means++ convergence speed is similar as original method , I think that the reason is the original picture is not too big , so the convergence speed is almost the same . And in ranomd\_normalized , the convergence speed maybe get worse than original , furthermore , It sometimes got a worst performance .

**Spectral clustering ratio cut**

|  |  |  |
| --- | --- | --- |
| K=3 | Random-normalized | k-means++ |
| Image1 |  |  |
|  |  |  |
|  | image1\_3Clusters\_random\_nor\_ratio | image1\_3Clusters\_k\_means\_plus\_plus\_ratio |
| Image2 |  |  |
|  |  |  |
|  | image2\_3Clusters\_random\_nor\_ratio | image2\_3Clusters\_k\_means\_plus\_plus\_ratio |

In the previous , I conclude that k=3 is the perfect choice . Image1’s the data points within the same cluster have the same coordinates in the eigenspace of graph Laplacian . However , the performance is not better by observing the graph in Iamge2 . The whole datapoint are all together , it can not easy to observe.

**Spectral clustering Normalized cut:**

|  |  |  |
| --- | --- | --- |
| K=3 | Random-normalized | k-means++ |
| Image1 |  |  |
|  |  |  |
|  | image1\_3Clusters\_random\_nor\_Normalized | image1\_3Clusters\_k\_means\_plus\_plus\_Normalized |
| Image2 |  |  |
|  |  |  |
|  | image2\_3Clusters\_random\_nor\_Normalized | image2\_3Clusters\_k\_means\_plus\_plus\_Normalized |
|  |  |  |

By applying normalized cut , the image1 still separate perfectly , furthermore , the image2 not like the previous graph , it can separate perfectly too , this could easy to observe and let the result be more precise .

1. Observations and discussion

The result shows that the spectral clustering get a better performance than k-means algorithm . And using spectral clustering can reduce dimension from n to k dimension , this could reduce the burden and let computational efficiency . And how to choose and be found . For example , when slowly increase k and find that the eigenvalue suddenly bigger , maybe it should choose k-1 category as clustering .