ML HW6

1. Code with detailed explanations (30%)

Part 1 & 2 & 3 & 4:

Initialization:

In main function, we choose the image that we want to do the clustering, clustering method and corresponding parameters of the kernel and number of clustering **(Part 2)** and then we compute the Gram matrix by using computeGramMatrix function. The kernel is defined as follows:



一張含有 文字, 螢幕擷取畫面, 軟體, 字型 的圖片

自動產生的描述一張含有 文字, 螢幕擷取畫面, 軟體, 字型 的圖片

自動產生的描述

**Kernel kmeans:**

In main function:

一張含有 文字, 螢幕擷取畫面, 字型 的圖片

自動產生的描述

1. Initialization method and give the corresponding label to the datapoint

**(Part3)**Based on the init\_type, the initialization method is selected. If init\_type is set to "kmeans++", the algorithm uses the k-means++ method to determine the k cluster centers and assigns labels to data points based on their closest center. Otherwise, data points are randomly assigned to k clusters.

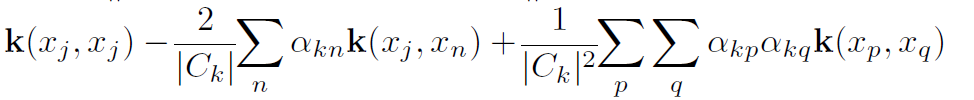
一張含有 文字, 螢幕擷取畫面, 字型 的圖片

自動產生的描述一張含有 文字, 螢幕擷取畫面, 字型, 軟體 的圖片

自動產生的描述

1. kmeans clustering

In each iteration, I calculate the mean of the data points within the same label group to serve as the center of that cluster. For each data point, I compute its distance to these centers and assign it the label of the closest center using the following formula:



If the number of label changes between iterations is small enough, the algorithm is considered to have converged, and the loop is terminated.

In kernel\_kmeans:

一張含有 文字, 螢幕擷取畫面 的圖片

自動產生的描述

1. visualization

After having each iteration labels, I use the labels to create the image of each iteration and save as gif

一張含有 文字, 螢幕擷取畫面, 字型 的圖片

自動產生的描述

一張含有 文字, 字型, 軟體, 螢幕擷取畫面 的圖片

自動產生的描述

**Ratio cut:**

In main function:

一張含有 文字, 螢幕擷取畫面, 字型 的圖片

自動產生的描述

The process is similar to unnormalized spectral clustering which is described below,

一張含有 文字, 螢幕擷取畫面, 字型, 文件 的圖片

自動產生的描述

Calculating L by using L=D-W, where D is diagonal matrix where dii = summation of ith row of W.

一張含有 文字, 螢幕擷取畫面, 字型 的圖片

自動產生的描述

The code of kmeans algorithm in unnormalized spectral clustering is like kmeans clustering. Doing the initialization according init\_type to assign the label to the datapoints. **(Part3)** If init\_type is set to "kmeans++", the algorithm uses the k-means++ method to determine the k cluster centers and assigns labels to data points based on their closest center. Otherwise, data points are randomly assigned to k clusters.

一張含有 文字, 螢幕擷取畫面, 字型 的圖片

自動產生的描述一張含有 文字, 螢幕擷取畫面, 軟體 的圖片

自動產生的描述

After initialization, in each iteration, I calculate the mean of the data points within the same label group to serve as the center of that cluster. For each data point, I computed its distance to these centers and assigned it the label of the closest center.

一張含有 文字, 螢幕擷取畫面, 軟體 的圖片

自動產生的描述

After having each iteration label, I use the labels to create the image of each iteration and save as gif. The code of the visualization is the same as kernel kmeans except adding the plot of eigenvector.

一張含有 文字, 螢幕擷取畫面, 軟體 的圖片

自動產生的描述

**👆(Part4 function:plot\_eigenspace)**

**Normalized Cut:**

In main function:

一張含有 文字, 螢幕擷取畫面, 字型 的圖片

自動產生的描述

The process is described as follows:

一張含有 文字, 螢幕擷取畫面, 字型, 數字 的圖片

自動產生的描述

Calculating L by using L=D-W, where D is diagonal matrix where dii = summation of ith row of W. Moreover, calculating the normalized L using D-1/2L D-1/2.

一張含有 文字, 螢幕擷取畫面, 軟體, 字型 的圖片

自動產生的描述

The kmeans function is the same as in the ratio cut, also the code of visualization is as same as in the ratio cut.

1. Experiments settings and results (30%) & discussion (20%)

**hyperparameters : γs=0.001, γc=0.001**

For all gif file, the naming rule is as follows:

**f”{clustering method}-{initialization type}-{number of clustering}”**

clustering method: kernel represent kernel kmeans, ratio represent ratio cut, normalized represent normalized cut

initialization type: kmean++ represent kmeans++ strategy, random represent random assign the label from 1 to number of clustering to each datapoints

number of clustering : 2 or 3

**Part1**

**Init\_type: random**

i.image1

|  |  |  |  |
| --- | --- | --- | --- |
| Final image | Kernel kmeans | Ratio cut | Normalized cut |
| 2 clusters |  |  |  |

ii.image2

|  |  |  |  |
| --- | --- | --- | --- |
| Final image | Kernel kmeans | Ratio cut | Normalized cut |
| 2 clusters |  | 一張含有 地圖, 紅色, 胭脂紅, 圖形 的圖片  自動產生的描述 | 一張含有 地圖, 紅色, 藝術 的圖片  自動產生的描述 |

Image 1 can clearly distinguish between the island and the ocean, likely because of distinct color or texture differences. However, in Image 2, the presence of numerous white dots on the tree causes kernel kmeans difficulties in distinguishing the background from objects like the tree and the animal while ratio cut and normalized cut perform better.

**Part2**

**Init\_type: random**

i.image1

|  |  |  |  |
| --- | --- | --- | --- |
| Final image | Kernel kmeans | Ratio cut | Normalized cut |
| 2 clusters | 一張含有 紅色, 地圖, 胭脂紅, 圖形 的圖片  自動產生的描述 | 一張含有 地圖, 紅色 的圖片  自動產生的描述 | 一張含有 地圖, 紅色, 胭脂紅 的圖片  自動產生的描述 |
| 3 clusters |  |  |  |

The reason that normalized cut in cluster 2 and 3 does not change too much is the initialization problem of using random initialization. Random initialization can lead to an imbalanced starting condition, where certain clusters are initially underpopulated or empty.

ii.image2

|  |  |  |  |
| --- | --- | --- | --- |
| Final image | Kernel kmeans | Ratio cut | Normalized cut |
| 2 clusters |  | 一張含有 地圖, 紅色, 胭脂紅, 圖形 的圖片  自動產生的描述 | 一張含有 地圖, 紅色, 藝術 的圖片  自動產生的描述 |
| 3 clusters |  |  |  |

The performance of 2 clusters seems better than in 3 clusters. Especially the part of the tree, the white noise may blend the features, reducing the contrast needed for effective clustering.

**Part3**

**Number of clusters:2**

i.image1

|  |  |  |  |
| --- | --- | --- | --- |
| Final image | Kernel kmeans | Ratio cut | Normalized cut |
| Random | 一張含有 紅色, 地圖, 胭脂紅, 圖形 的圖片  自動產生的描述 | 一張含有 地圖, 紅色 的圖片  自動產生的描述 | 一張含有 地圖, 紅色, 胭脂紅 的圖片  自動產生的描述 |
| Kmeans++ |  |  |  |

ii.image2

|  |  |  |  |
| --- | --- | --- | --- |
| Final image | Kernel kmeans | Ratio cut | Normalized cut |
| Random | 一張含有 紅色, 地圖 的圖片  自動產生的描述 | 一張含有 地圖, 紅色, 胭脂紅, 圖形 的圖片  自動產生的描述 | 一張含有 地圖, 紅色, 藝術 的圖片  自動產生的描述 |
| Kmeans++ |  |  |  |

The outcome for kmeans and random does not have significant difference. However, their initial image is very different from what you can see in the below:

一張含有 樣式, 圖案, 圖案 (服裝設計), 布 的圖片

自動產生的描述 一張含有 藝術, 紅色, 兒童藝術, 圖片 的圖片

自動產生的描述

the left image is the start image of random initialization,while the right image is using kmeans++.

Most of the time the convergence speed of kmean++ is faster than random initialization.

**Part4**

i.image1

I only use two eigenvector to show the result.

ratio cut:

|  |  |  |
| --- | --- | --- |
|  | Final image | Plot of eigenvector |
| Random  k=2 |  |  |
| Random  k=3 |  |  |
| Kmeans++  k=2 |  |  |
| Kmeans++  k=3 |  |  |

Normalized cut:

|  |  |  |
| --- | --- | --- |
|  | Final image | Plot of eigenvector |
| Random  k=2 |  |  |
| Random  k=3 |  |  |
| Kmeans++  k=2 |  |  |
| Kmeans++  k=3 |  |  |

ii.image2

ratio cut:

|  |  |  |
| --- | --- | --- |
|  | Final image | Plot of eigenvector |
| Random  k=2 |  |  |
| Random  k=3 |  |  |
| Kmeans++  k=2 |  |  |
| Kmeans++  k=3 |  |  |

Normalized cut:

|  |  |  |
| --- | --- | --- |
|  | Final image | Plot of eigenvector |
| Random  k=2 |  |  |
| Random  k=3 |  |  |
| Kmeans++  k=2 |  |  |
| Kmeans++  k=3 |  |  |

Points sharing the same label tend to be close to each other in the eigenspace. However, the eigenspace representations for ratio cut and normalized cut differ significantly.

1. Observations and discussion (20%)
   * 1. Compare the performance between different clustering methods.

In Image 1, the color distribution is relatively simple, resulting in similar performance across the models. However, in Image 2, the color distribution is more complex, with objects potentially having varying colors. As a result, kernel k-means struggles to perform well, while spectral clustering achieves better results.

* + 1. Compare the execution time of different settings.

Since the eigen decomposition in spectral clustering requires O(n^3), it generally takes longer than kernel k-means. The most time-consuming step in kernel k-means is computing the Gram matrix, which requires O(n^2). As a result, spectral clustering is computationally more expensive, especially for large datasets, due to its reliance on eigen decomposition.

* + 1. Anything you want to discuss.

If the difference between γs​ and γc​ is too large, the clustering performance will significantly degrade. Based on current testing, setting these parameters to be equal yields the best results.