Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Prof. M. Pollefeyes

Filippo Ficarra: Lab 5 - Image Segmentation

fficarra@student.ethz.ch, 22–938–062.

19/11/2023 - 17:24h

1 Mean-Shift Algorithm

The Mean-Shift Algorithm aims to find modes in a given dataset. It is used to perform unsupervised clustering. The principal benefit of this algorithm is that we don't have to impose a fixed number of cluster a priori but it will just depend on a variable called "bandwidth".

1.1 Distance function

The first function completed is the distance function. This function it will further used in the gaussian kernel to compute the weight of each point. The following function simply computes the eucledian distance of the point x and all points in X.

Listing 1: Distance function

```
1    def distance(x, X):
2      return np.sqrt(np.sum((x - X) ** 2, axis=1))
```

1.2 Gaussian Kernel

The Gaussian kernel is used since it provides continuous and smooth distributions and are suitable to extract information also in noisy data.

Give a bandwith, the gaussian kernel is simply the following:

Listing 2: Gaussian Kernel

```
1    def gaussian(dist, bandwidth):
2      K = np.exp(-(dist ** 2) / (2 * bandwidth ** 2))
3      return K
```

1.3 Update point

Updating the point is the part in where the algorithm start to find the clusters. The update of a point $y^{(t+1)}$ is computed as following.

$$y_j^{(t)} \leftarrow \frac{\sum\limits_{i=1}^{n} X_i w_i(j)}{\sum\limits_{i=1}^{n} w_i(j)}$$

where the $w_i(j)$ are the weights for the j-th point computed with the gaussian function above.

1.4 Iterate the algorithm

The final step is to compute the update for each point and compute it for a number of iteration as we can show below:

Listing 3: Mean-shift

```
1    def meanshift_step(X, bandwidth=1):
2      y = np.copy(X)
3      for i in range(X.shape[0]):
4            dist = distance(y[i], X)
5            weight = gaussian(dist, bandwidth)
6            y[i] = update_point(weight, X)
7      return y
8
9    def meanshift(X):
10     for _ in range(20):
11           X = meanshift_step(X, bandwidth=bandwidth)
12     return X
```

Then the centroids are computed in this way:

Listing 4: Centroids

```
1    centroids, labels = np.unique((X / 4).round(), return_inverse=True, axis=0)
```

2 Results

First, the number of colors was 24 so we were constrained with the number of clusters. This has been noticed when using different values of the bandwidth in the gaussian kernels.

Bandwidth	Number of Clusters
1	282
2	44
2.5	25
3	15
4	9
5	5
6	4
7	3

Figure 1: Bandwidth vs. Number of Clusters

Lower bandwidth correspond to local understanding of data and so to a greater number of clusters, while higher bandwidth will have more general understanding of the data and tend to reduce the numnber of clusters. On the left a table with the bandwith and the corresponding number of clusters. Given this values, the bandwith suitable for this exercise were from 3 on. Below the images corresponding to bandwidth = 3,4,5,6,7.
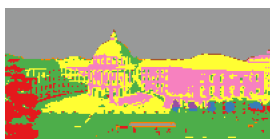

Figure 2: bandwidth = 3

Figure 3: bandwidth = 4

Figure 4: bandwidth = 5

Figure 5: bandwidth = 6

Figure 6: bandwidth = 7