Image segmentation and Filtering using Mean-Shift

BDA 492: Data Mining and its Applications
Assignment 2

Manav Prabhakar

July 2021

Abstract

Mean-shift, an old pattern recognition procedure, has been used here to analyze the complex multi-modal feature space, an image, and to delineate arbitrarily shaped clusters in it. We will discuss briefly about the mean - shift procedure and will then focus on implementing it for segmentation and then filtering of a set of images.

1 Introduction

Low-level computer vision tasks are intricate. The low-level stage is required to provide a reliable enough representation of the input and that the feature extraction process be controlled only by very few tuning parameters corresponding to the intuitive measures in the input domain. Feature space-based analysis of images is suitable for achieving this. Mean-shift and mode finding techniques, such as k-means and mixtures of Gaussians, model the feature vectors associated with each pixel (e.g., color and position) as samples from an unknown probability density function and then try to find clusters (modes) in this distribution. The rationale behind the density estimation-based non-parametric clustering approach is that the feature space can be regarded as the empirical probability density function of the represented parameter. Dense regions in the feature space, thus, correspond to the modes of the unknown density. We shall be using the mean shift procedure for Mode detection and clustering [1].

The key to mean shift is a technique for efficiently finding peaks in this high-dimensional data distribution without ever computing the complete function explicitly.

2 The Approach and Implementation

2.1 Kernel Density Estimate and Mean-Shift

Kernel Density Estimate is the most popular density estimation technique. For n data points in a d-dimensional space R^d, the multi-variate kernel density estimator with kernel K(x) and a symmetric positive definite $d \times d$ bandwidth matrix H, computed in the point x is give by

$$\hat{f}(x) = \frac{1}{n} \sum_{i=1}^{n} K_H(x - x_i) \tag{1}$$

$$\text{where } K_h(x) = |H|^{-1/2} K(H^{-1/2} x)$$

where

$$X = (x_1, x_2, ...x_d)^T$$

$X_i = (x_{i1}, x_{i2}, ...x_{id})^T$, $i = 1, 2, ..., n$ The multi-variate kernel can be generated from a symmetric uni-variate Kernel $K_i(x)$ in two different ways:

- $K^P(x)$: obtained from the product of the uni-variate kernels

$$K^P(x) = \prod_{i=1}^{d} K_1(x_i)$$

- $K^S(x)$: obtained by rotating $K_1(x)$ in R^d i.e. radially symmetric.

$$K^S(x) = a_{K,d} K_1(||x||)$$

The constant $a_{k,d}$ assures that $K^S(X)$ integrates to 1.

$$a_{k,d}^{-1} = \int_{R^d} K_1(||x||) dx$$

Using one bandwidth parameter h, we can rewrite our kernel density estimation from 2.1 as

$$\hat{f}(x) = \frac{1}{nh^d} \sum_{i=1}^{n} K_H \frac{x - x_i}{h} \tag{2}$$

The quality of a kernel density estimator is measured by the mean of square error between the density and its estimate, integrated over the domain of definition. We have used Gaussian Kernel, so

$$k_N(x) = \exp{(-\frac{1}{2} x)} \text{ , } x \geq 0 \tag{3}$$

yields the multi-variate kernel

$$K_N(x) = (2\pi)^{-d/2} \exp{(-\frac{1}{2} ||x||^2)} \tag{4}$$

Rewriting the well know KDE equation in the form of radially symmetric $K^S(x)$

$$\hat{f_{h,k}}(x) = \frac{c_{k,d}}{nh^d} \sum_{i=1}^{n} k(\|\frac{x - x_i}{h}\|)^2 \tag{5}$$

The modes are located among the zeros of the gradient. Mean shift algorithm facilitates the locating of these zeros without estimating the density.

$$\nabla \hat{f_{h,k}}(x) = \frac{2c_{k,d}}{nh^{d+2}} \sum_{i=1}^{n} (x - x_i) k'(\| \frac{x - x_i}{h} \|)^2 \tag{6}$$

Let

$$g(x) = -K'(x)$$

assuming that the derivative of the kernel k exists for all x$\epsilon[0, \inf)$, except for a finite set of points. Using g(x), the kernel G(x) is defined as

$$G(x) = c_{g,d} g(\|x\|^2)$$

where $c_{g,d}$ is the corresponding normalization constant. The kernel $K(x)$ is the shadow of $G(x)$. Introducing g(x) into the equation, we have

$$\nabla \hat{f}_{h,k}(x) = \frac{2c_{k,d}}{nh^{d+2}} \sum_{i=1}^{n} (x - x_i) g(\| \frac{x - x_i}{h} \|)^2$$

$$= \frac{2c_{k,d}}{nh^{d+2}} \left[\sum_{i=1}^{n} g(\| \frac{x - x_i}{h} \|)^2 \right] \left[\frac{\sum_{i=1}^{n} x_i g(\| \frac{x - x_i}{h} \|^2}{\sum_{i=1}^{n} g(\| \frac{x - x_i}{h} \|)^2} - x \right]$$

The first term is proportional to the density estimate while the second term is the mean shift.

$$m_{h,G}(x) = \frac{\sum_{i=1}^{n} x_i g(\| \frac{x - x_i}{h} \|^2}{\sum_{i=1}^{n} g(\| \frac{x - x_i}{h} \|)^2} - x \tag{7}$$

This gives us the difference between the weighted mean using kernel G for weights and x, the center of kernel window. the gradient density can now be written as

$$\nabla \hat{f}_{h,k}(x) f = f_{h,G}(x) \frac{2\hat{c_{k,d}}}{h^2 c_{g,d}} m_{h,G}(x)$$

Implies

$$m_{h,G}(x) = \frac{1}{2} h^2 c \frac{\nabla \hat{f}_{h,K}(x)}{\hat{f}_{h,G}(x)}$$

This equation shows that the mean shift vector computed with kernel G is proportional to the normalized density gradient estimate obtained with kernel K. The mean shift vector thus, will always point towards the direction of maximum increase in density. The sequence of successive locations of the kernel G can be obtained from,

$$y_{j+1} = \frac{\sum_{i=1}^{n} x_i g(\| \frac{x - x_i}{h} \|^2}{\sum_{i=1}^{n} g(\| \frac{x - x_i}{h} \|)^2}$$

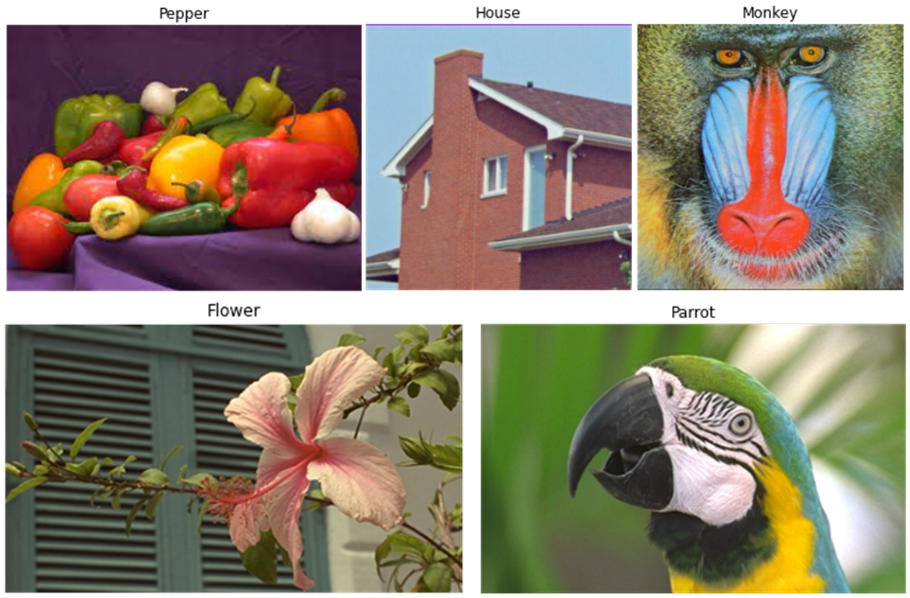

Figure 1: The images that have been used for implementing mean-shift segmentation and filtering

2.2 Mean-shift based segmentation

We have employed the following approach for implementing mean-shift segmentation on images.

- Convert the image to L*u*v space or alternately keep the original RGB colors and augment them with the pixel (x,y) locations.
- For every pixel (R,G,B,x,y), compute the weighted mean of its neighbors using a kernel. We have used a finite -radius Gaussian kernel.
- Weight the color and spatial scales differently that is the values of h_s, h_r and M will be different.
- Replace the current value with this weighted mean and iterate until either the motion is below a threshold or a finite number of steps have been taken.
- Cluster all final values (modes) that are within a threshold. In other words find the connected components.

Since, each pixel is associated with a final mean-shift (mode) value, this results in an image segmentation i.e., each pixel is labeled with its final component.

2.2.1 Setting up the images

Five input images have been used for implementing the approach (see Figure 1. For every image, we obtain an $(M \times N \times C)$ matrix where (M, N) denote the height and width of the image, while C represents the number of channels. Here C = 3, for RGB. We can perform segmentation based on just the RGB values, however, better results can usually be obtained by clustering in the joint domain of color and location. In this approach, the spatial coordinates of the image are concatenated with the color values and mean-shift clustering is applied in this five-dimensional space. Since location and color may have different scales, the kernels are adjusted separately, just as in the bilateral filter kernel. We have three values for each pixel that depict the intensity of the pixels So, the intensity of a pixel p can be defined as

$$I_p = (R_p, G_p, B_p)$$

. Similarly, we can also define the spatial position (S_p) of the pixel with the coordinates of the pixel.

$$S_p = (x_p, y_p)$$

We combine these values into a 5 dimensional vector. So, for each pixel we will have

$$V_p = (R_p, G_p, B_p, x_p, y_p)$$

We will have these values for each pixel for an image and will perform mean-shift segmentation on this set of 5 dimensional vectors to prevent clustering of small isolated pixels that happen to have the same color, which may not correspond to a semantically meaningful segmentation of the image.

2.2.2 Computing Weighted means with finite radius Gaussian Kernel

Let the image be of size (M, N), then we will have $(M \times N, 5)$ matrix. The kernel window size was fixed to around 10% of the smaller value of height or width of the image. So, if the image is of size $(230, 320)$, the window size shall be (23×23). We have computed for every pixel, so all images were padded with extra zeros based on the window size, such that the mean values can be calculated by taking very pixel as the centre of the kernel exactly once.

2.2.3 Tuning parameters and replacing current value with weighted means

Let $\hat{\theta}_i = [R_i, G_i, B_i, x_i, y_i]$ Then to update every pixel, we iterate using the following equation

$$\hat{\theta} = \frac{\sum_{i=1}^{n} x_i W(\|\frac{\hat{\theta}-x_i}{h}\|^2}{\sum_{i=1}^{n} W(\|\frac{\hat{\theta}-x_i}{h}\|)^2} \tag{8}$$

where

$$W_{h_s h_r}(x) = \frac{C}{(h_s)^2, (h_r)^2} k\left(\|\frac{S}{h_s}\|^2\right) k\left(\|\frac{I}{h_r}\|^2\right)$$

Here, V^S is the spatial part, V^I is the intensity/range part of the 5 dimensional feature vector V. $k(x)$ is the common profile used in both domains. We have used a Gaussian kernel as $k(x)$. h_s, h_r are employed kernel bandwidths. In practice, the user just needs to tune the bandwidth parameter $h = (h_s, h_r)$ to determine the resolution of mode detection. Alternatively, we can also derive the values for h_s and h_r using MSE. We maintain both the previous and the new values for RGB and x,y. For filtering purposes, only the RGB values are updated.

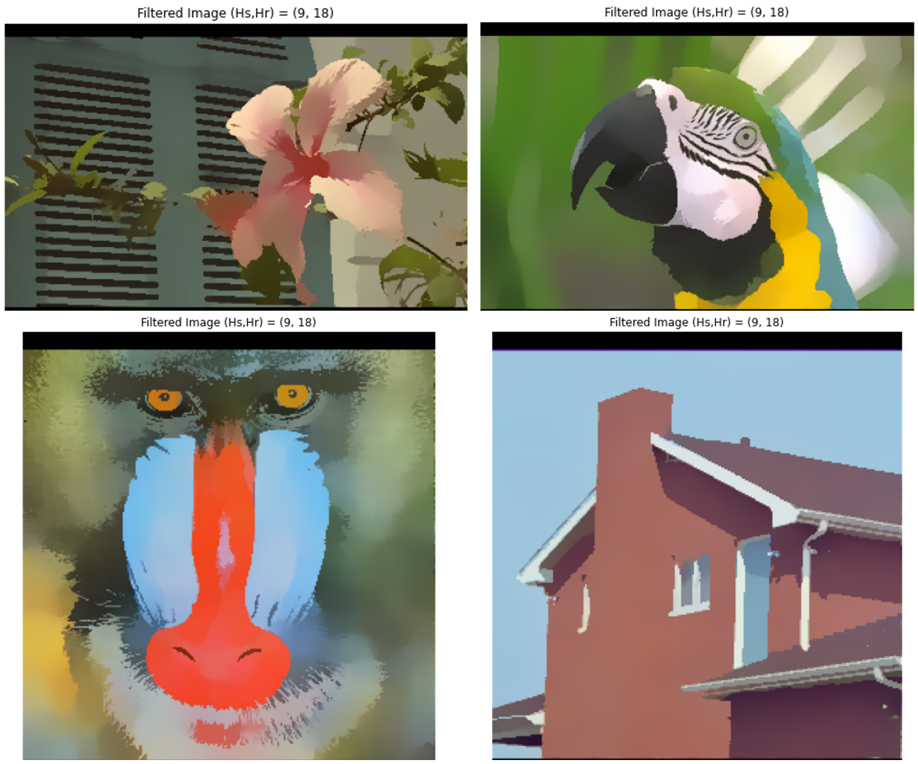

Figure 2: Filtering results

2.2.4 Clustering values within a threshold

Once the image has been filtered using mean shift, the next objective is to segment the image by clustering pixels with same modes. For this, we use both our new RGB values and new (x,y) values. The colors for the clusters can be arbitrarily chosen or can be the mean of the colors of all pixels in a particular cluster. The latter approach has been followed here.

3 Results and Analysis

The approach was tested by implementing it on the 5 input images (Figure 1).

We obtained satisfying results for both filtered (see Figure 2) and segmented images. (see Figure 3)

Further, we also implemented for different values of h_s and h_r (Figure 4). From the values and the results, we can see that as the value of h_r increases, there is smoothening of the image or the filter smoothes the image, while this does not happend when we increase h_s. For $h_s/h_r < 1$, the image seems to have a smooth filter The approach seems to work pretty well as we are able to attain suitable results after filtering and segmentation.

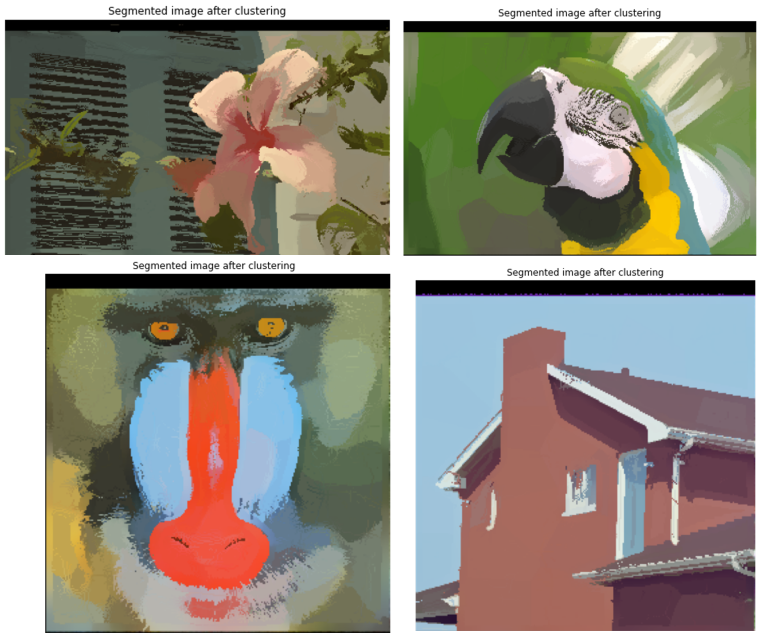

Figure 3: Segmentation results

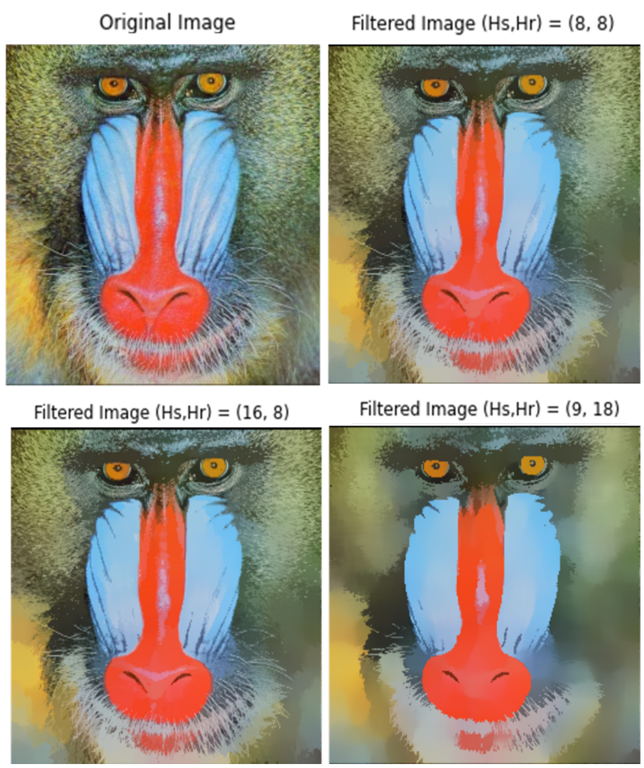

Figure 4: Filtered images after applying mean shift with different values of h_s and h_r

4 Acknowledgement

To Prof. Santosh Singh, Big Data Analytics Centre for providing an opportunity to understand, implement and analyze this pattern recognition technique (Mean -shift procedure) on images.

5 Conclusion

We were able to implement and understand the procedure of applying Mean shift technique for performing image filtering and segmentation.

References

[1] Dorin Comaniciu and Peter Meer; Mean shift: A robust approach toward feature space analysis,IEEE Transactions on Pattern Analysis and Machine Intelligence 2002 p603-619,