ECE 544NA Fall 2016 Assignment 3

Jifu Zhao Date: 10/21/2016

I. Pencil-and-Paper

1. E-Step

Since that $p_{X|\Theta(x_i|\theta)}$ is described as:

$$p_{X|\Theta(x_i|\theta)} = \sum_{h=1}^{m} w_h p_{V|H,\Theta}(x_i, h, \theta) \tag{1}$$

The corresponding log-likelihood is:

$$log\mathcal{L}(\Theta) = \sum_{i=1}^{n} log(\sum_{h=1}^{m} w_h p_{V|H,\Theta}(x_i, h, \theta)) \tag{2}$$

Now, suppose the hidden variable is y, where y could be 1, 2, $\cdots$, m. Then, we can calculate the expectation of the log-likelihood $E[log\mathcal{L}(\Theta)]$ as:

$$E[log\mathcal{L}(\Theta)] = \sum_{k=1}^{m}[\sum_{i=1}^{n} log(\sum_{h=1}^{m} w_h p_{V|H,\Theta}(x_i, h, \theta|y = k))] \cdot \gamma_i(k) \tag{3}$$

where $\gamma_i(k)$ is the posterior probability for $y = k$, and $\gamma_i(k)$ is defined as:

$$\gamma_i(k) = \frac{w_k \mathcal{N}(x_i; \mu_k, \Sigma_k)}{\sum_l w_l \mathcal{N}(x_i; \mu_l, \Sigma_l)} \tag{4}$$

Also, notice that

$$p_{V|H,\Theta}(x_i, h, \theta|y = k)) = \begin{cases} \mathcal{N}(x_i; \mu_k, \Sigma_k) & \text{if } h = k \\ 0 & \text{if } h \neq k \end{cases} \tag{5}$$

So,

$$\begin{aligned} E[log\mathcal{L}(\Theta)] &= \sum_{i=1}^{n}\sum_{k=1}^{m} log(w_k p_{V|H,\Theta}(x_i, h = k, \theta)) \cdot \gamma_i(k) \\ &= \sum_{i=1}^{n}\sum_{k=1}^{m} log(w_k \mathcal{N}(x_i; \mu_k, \Sigma_k)) \cdot \frac{w_k \mathcal{N}(x_i; \mu_k, \Sigma_k)}{\sum_l w_l \mathcal{N}(x_i; \mu_l, \Sigma_l)} \\ &= \sum_{i=1}^{n}\sum_{k=1}^{m} log(w_k \mathcal{N}(x_i; \mu_k, \Sigma_k)) \cdot \gamma_i(k) \end{aligned} \tag{6}$$

So, in E-Step, the most important thing is to calculate $\gamma_i(k)$ for each i and k, where

$$\gamma_i(k) = \frac{w_k \mathcal{N}(x_i; \mu_k, \Sigma_k)}{\sum_l w_l \mathcal{N}(x_i; \mu_l, \Sigma_l)} \tag{7}$$

2. M-Step

From above equation (5), our goal is the maximize the expectation of log-likelihood $E[log\mathcal{L}(\Theta)]$.

First, consider w_k. With the constraint of $\sum_k w_k = 1$, we have:

$$\frac{\partial}{\partial w_k}[\sum_{i=1}^{n}\sum_{k=1}^{m} log(w_k\mathcal{N}(x_i;\mu_k,\Sigma_k))\cdot\gamma_i(k) + \lambda(\sum_k w_k - 1)] = 0 \tag{8}$$

So, we have:

$$\sum_{i=1}^{n}\frac{\gamma_i(k)}{w_k} + \lambda = 0 \tag{9}$$

Summing it over k from 1 to m and with the equation that $\sum_k \gamma_i(k) = 1$, we can have:

$$\lambda = n \tag{10}$$

So, we have:

$$w_k^{new} = \frac{1}{n}\sum_{i=1}^{n}\gamma_i(k) \tag{11}$$

Now, let's consider μ_k and Σ_k. Following the steps in *A gentle tutorial of the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden Markov models*, we can simply get the following result:

$$\mu_k^{new} = \frac{\sum_{i=1}^{n} x_i\gamma_i(k)}{\sum_{i=1}^{n}\gamma_i(k)} \tag{12}$$

$$\Sigma_k^{new} = \frac{\sum_{i=1}^{n}(x_i - \mu_k^{new})\cdot(x_i - \mu_k^{new})^T\cdot\gamma_i(k)}{\sum_{i=1}^{n}\gamma_i(k)} \tag{13}$$

So, Equation (10), (11) and (12) are the main steps for M-Step.

Following Equation (6), (10), (11) and (12), we can iterate through E-Step and M-Step until reaching some stop criteria.

II. Code-from-Scratch

1. Methods

(1). The overall structure of the EM part is shown in Figure 1, which is named EM.py.

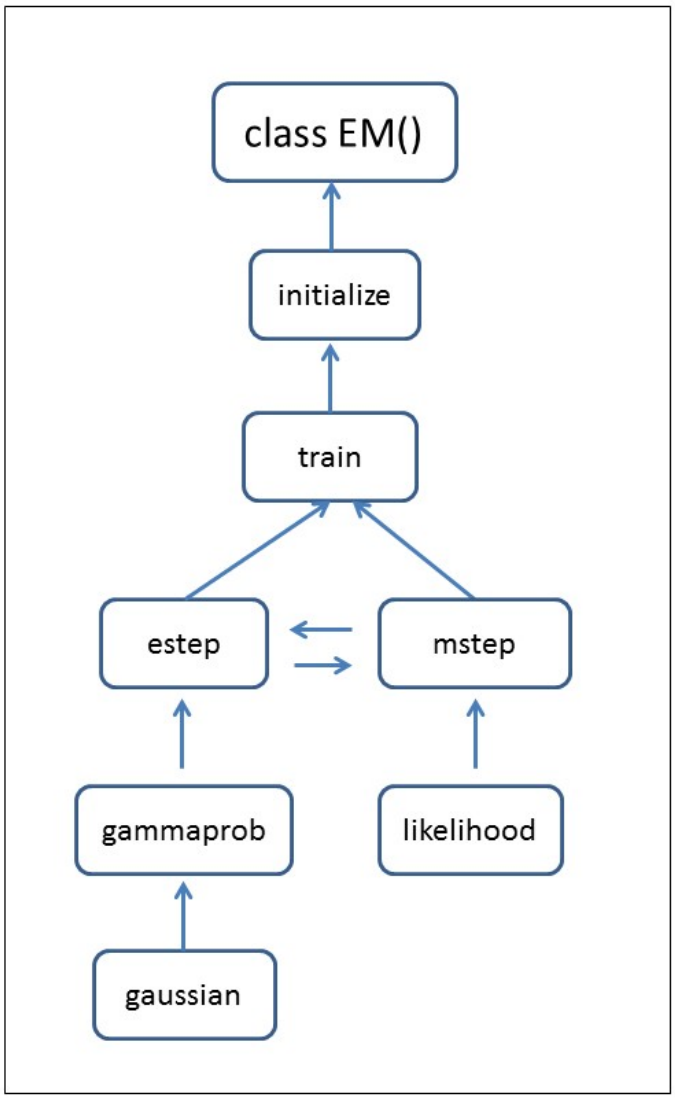

Figure 1: Algorithm Structure

The whole program has initialize, train, estep, mstep, gammaprob, gaussian, likelihood part (get_label and get_params are not shown in this figure). These functions are based above each other.

(2). For the EM algorithm, the final goal is to determine m Gaussian mixtures that can best represent the original data.

Suppose that we are going to train a GMM model with m mixtures, and the original data has n samples and each has d features. Then, there are m weights, m means and each has d dimensions.

Also, there are m covariance matrix, each of which has d^2 dimensions. So, the number of trainable parameters are:

$$m + md + md^2 = m(1 + d + d^2)$$

***Note**: In our case, to avoid some computation problems, although the origin input image is in RGB format, whose range is from 0 to 255. In the following program, we first change the input image value range from [0, 255] to [0, 1].

2. Results

(1). In the first part, the original image is shown in Figure 2.

Figure 2: Original Dog Figure

First, choosing m = 3, we can get three Gaussian Mixture model, the corresponding weights, mean and covariance matrix is shown below:

$$w_1 = 0.29569$$

$$w_2 = 0.31897$$

$$w_3 = 0.38534$$

And the mean vector is:

$$\vec{\mu_1} = [0.24586,\ 0.22526,\ 0.07403]^T$$

$$\vec{\mu_2} = [0.76036,\ 0.59326,\ 0.41495]^T$$

$$\vec{\mu_3} = [0.44742,\ 0.50353,\ 0.00079]^T$$

The corresponding covariance matrix is:

$$\Sigma_1 = \begin{bmatrix} 0.05137 & 0.04574 & 0.01579 \\ 0.04574 & 0.04544 & 0.01191 \\ 0.01579 & 0.01191 & 0.00864 \end{bmatrix}$$

$$\Sigma_2 = \begin{bmatrix} 0.02177 & 0.02319 & 0.02469 \\ 0.02319 & 0.02691 & 0.03041 \\ 0.02469 & 0.03041 & 0.03740 \end{bmatrix}$$

$$\Sigma_3 = \begin{bmatrix} 0.00741 & 0.00399 & 0.00002 \\ 0.00399 & 0.00441 & 0.00000 \\ 0.00002 & 0.00000 & 0.00001 \end{bmatrix}$$

***Note**: During the training process, we notice that the initialization has a big effect on the final output. In other words, we have chances to reach some local maximum rather than the global maximum. So, every time we run the code, we could get total different result. A good initialization is to find some good points using K-means method, however, in this work, this method is not implemented.

(2). Choosing m to be 3, 5 and 10, through some fixed random initialization, we can reconstruct the original image. The result is shown in Figure 3, Figure 4 and Figure 5.

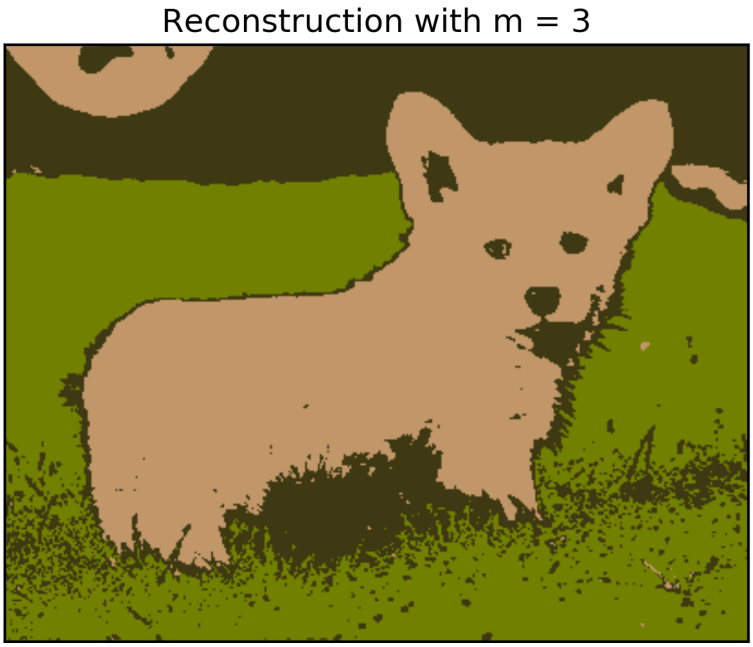

Figure 3: Dog with 3 Gaussian Mixtures

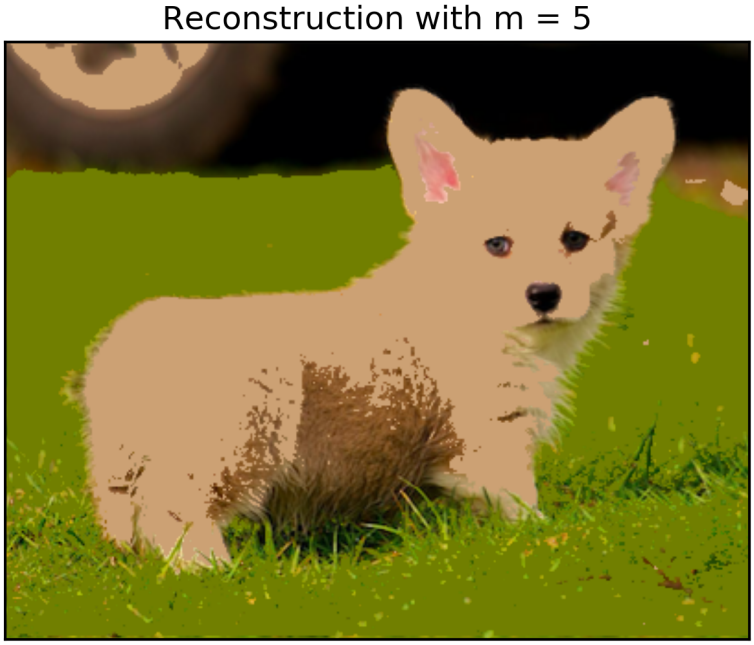

Figure 4: Dog with 5 Gaussian Mixtures

Figure 5: Dog with 10 Gaussian Mixtures

(3). In this part, we choose one figure that contains mountains to test our model. The original image is shown in Figure 6.

Figure 6: Original Mountain Figure

First, choosing m = 3, we can get three Gaussian Mixture model, the corresponding weights, mean and covariance matrix are shown below:

$$w_1 = 0.19563$$

$$w_2 = 0.43580$$

$$w_3 = 0.36857$$

And the mean vector is:

$$\vec{\mu_1} = [0.49253,\ 0.58691,\ 0.72089]^T$$

$$\vec{\mu_2} = [0.25903,\ 0.35042,\ 0.24338]^T$$

$$\vec{\mu_3} = [0.61135,\ 0.61422,\ 0.61133]^T$$

The corresponding covariance matrix is:

$$\Sigma_1 = \begin{bmatrix} 0.03400 & 0.02849 & 0.02296 \\ 0.02849 & 0.02511 & 0.02183 \\ 0.02296 & 0.02183 & 0.02151 \end{bmatrix}$$

$$\Sigma_2 = \begin{bmatrix} 0.01594 & 0.01579 & 0.01530 \\ 0.01579 & 0.01677 & 0.01532 \\ 0.01530 & 0.01532 & 0.01949 \end{bmatrix}$$

$$\Sigma_3 = \begin{bmatrix} 0.04029 & 0.03880 & 0.03899 \\ 0.03880 & 0.03778 & 0.03822 \\ 0.03899 & 0.03822 & 0.03962 \end{bmatrix}$$

Now, choosing m to be 3, 5 and 10, through some fixed random initialization, we can reconstruct the original image. The result is shown in Figure 7, Figure 8 and Figure 9.

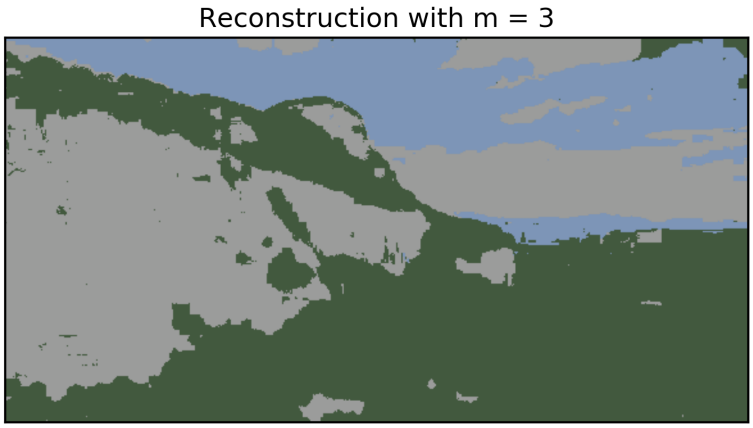

Figure 7: Mountain with 3 Gaussian Mixtures

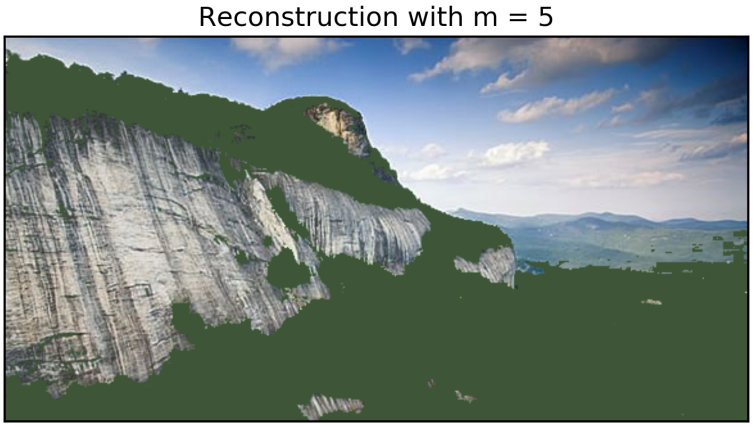

Figure 8: Mountain with 5 Gaussian Mixtures

Figure 9: Mountain with 10 Gaussian Mixtures

III. TensorFlow

1. Methods

In this section, we implemented the higher-level API from TenforFlow GMM. Since the original has already finished, we just simply import the module and build the GMM model through setting some parameters, such as num_clusters, random_seed, initial_clusters, covariance_type. In this part, we set the initial_clusters to be random and set covariance_type to be full.

$$gmm = GMM(*args)$$

Then, we can fit our data through

$$gmm.fit(x)$$

and then we can get the label through the function

$$gmm.predict(x)$$

2. Results

(1). In the first part, the original image is Figure 2, but we apply the GMM model from TensorFlow instead of our own model.

First, choosing m = 3, we can get three Gaussian Mixture model, the corresponding weights, mean and covariance matrix is shown below:

$$w_1 = 0.26408$$

$$w_2 = 0.23359$$

$$w_3 = 0.50233$$

And the mean vector is:

$$\vec{\mu_1} = [0.76676,\ 0.59889,\ 0.42117]^T$$
$$\vec{\mu_2} = [0.19277,\ 0.16881,\ 0.05494]^T$$
$$\vec{\mu_3} = [0.45933,\ 0.51962,\ 0.00615]^T$$

The corresponding covariance matrix is:

$$\Sigma_1 = \begin{bmatrix} 0.01856 & 0.01889 & 0.02230 \\ 0.01889 & 0.02332 & 0.02685 \\ 0.02230 & 0.02685 & 0.03730 \end{bmatrix}$$

$$\Sigma_2 = \begin{bmatrix} 0.03293 & 0.02596 & 0.00883 \\ 0.02596 & 0.02639 & 0.00610 \\ 0.00883 & 0.00610 & 0.00556 \end{bmatrix}$$

$$\Sigma_3 = \begin{bmatrix} 0.01044 & 0.00437 & 0.00023 \\ 0.00437 & 0.00520 & 0.00012 \\ 0.00023 & 0.00012 & 0.00132 \end{bmatrix}$$

(2). Choosing m to be 3, 5 and 10, through some fixed random initialization, we can reconstruct the original image. The result is shown in Figure 10, Figure 11 and Figure 12.

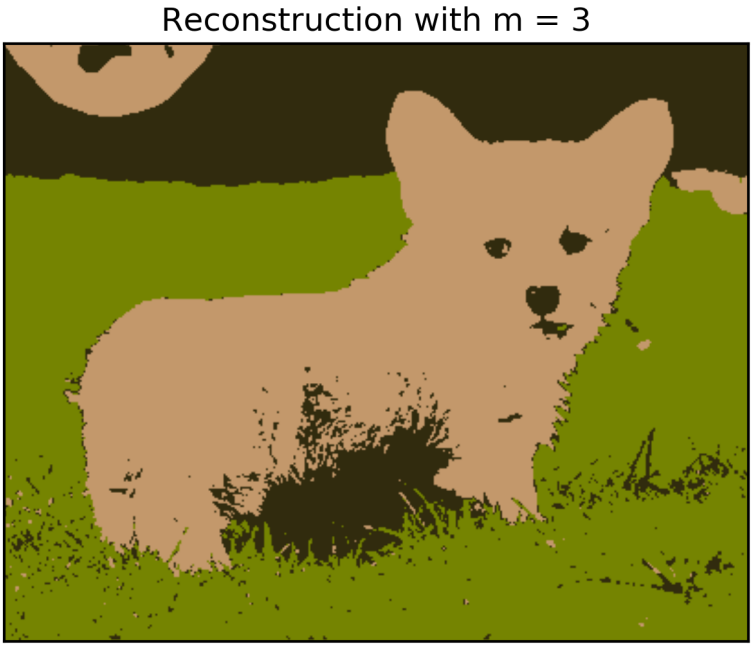

Figure 10: Dog with 3 Gaussian Mixtures

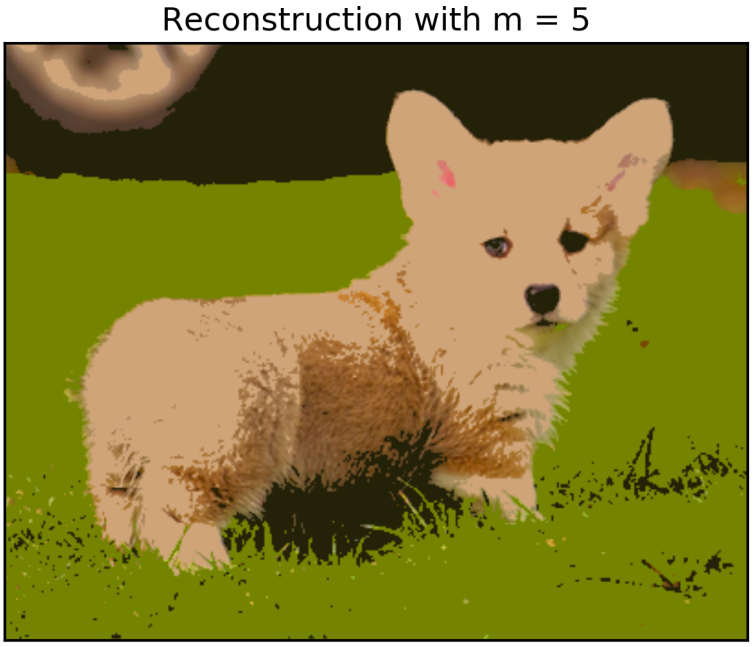

Figure 11: Dog with 5 Gaussian Mixtures

Figure 12: Dog with 10 Gaussian Mixtures

(3). In this part, we choose one figure that contains mountains to test our model. The original image is shown in Figure 6, but we apply the GMM model from TensorFlow instead of our own model.

First, choosing m = 3, we can get three Gaussian Mixture model, the corresponding weights, mean and covariance matrix are shown below:

$$w_1 = 0.36099$$

$$w_2 = 0.20816$$

$$w_3 = 0.43084$$

And the mean vector is:

$$\vec{\mu_1} = [0.70019,\ 0.71285,\ 0.72229]^T$$
$$\vec{\mu_2} = [0.46636,\ 0.54795,\ 0.63692]^T$$
$$\vec{\mu_3} = [0.23894,\ 0.31741,\ 0.23316]^T$$

The corresponding covariance matrix is:

$$\Sigma_1 = \begin{bmatrix} 0.02382 & 0.01941 & 0.01793 \\ 0.01941 & 0.01833 & 0.01684 \\ 0.01793 & 0.01684 & 0.02003 \end{bmatrix}$$

$$\Sigma_2 = \begin{bmatrix} 0.02526 & 0.01735 & 0.00951 \\ 0.01735 & 0.01567 & 0.01217 \\ 0.00951 & 0.01217 & 0.02121 \end{bmatrix}$$

$$\Sigma_3 = \begin{bmatrix} 0.01769 & 0.01662 & 0.01141 \\ 0.01662 & 0.01926 & 0.01151 \\ 0.01141 & 0.01151 & 0.01468 \end{bmatrix}$$

Now, choosing m to be 3, 5 and 10, through some fixed random initialization, we can reconstruct the original image. The result is shown in Figure 13, Figure 14 and Figure 15.

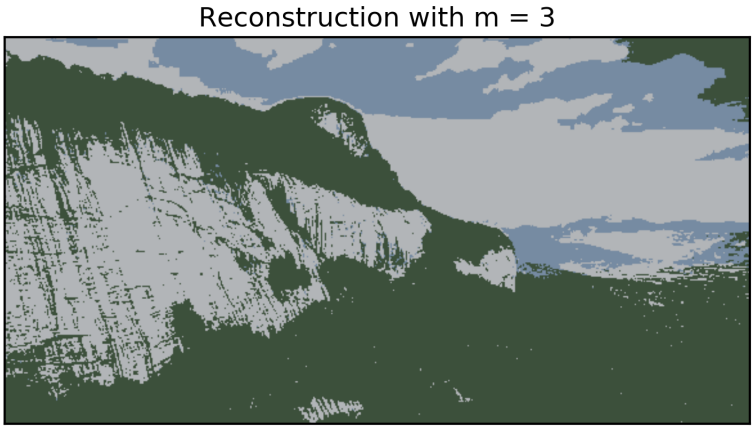

Figure 13: Mountain with 3 Gaussian Mixtures

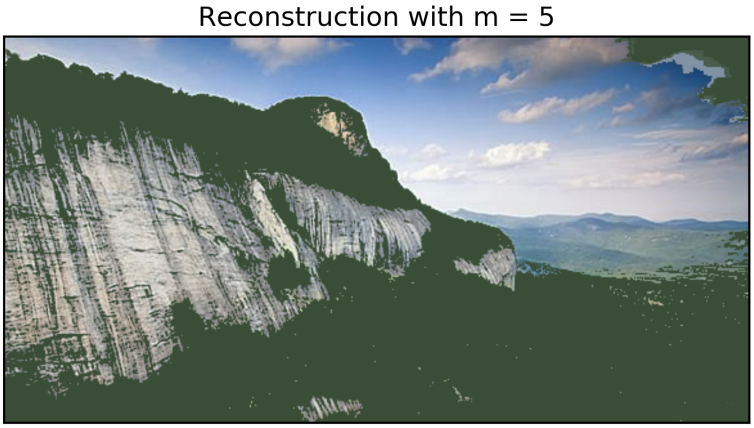

Figure 14: Mountain with 5 Gaussian Mixtures

Figure 15: Mountain with 10 Gaussian Mixtures