SSU : Assignment 2
Image segmentation using clustering

Teymur Azayev

1 Overview

This document is a short report for the second assignment from the subject. It includes a theoretical overview of the task, the EM algorithm, including implementation and comparisons to a few baseline approaches.

2 Task

Given a small set of images $T^m \in \{x^1, ..., x^m\}$ we are asked to segment the images into background - foreground parts. This means clustering each pixel into one of the two categories. The image shapes are generated by a common shape model which is common across the whole dataset. The RGB color vectors are generated using the appearance model which is specific to each image and is conditioned on the global shape model.

3 Theory

As mentioned in the task description, each pixel is eventually generated by an appearance model which is conditioned on the shape model.

$$
\begin{aligned}
p_{u_i,\theta^l}(x_i^l, s^i) &= p_{u_i,\theta^l}(x_i^l|s_i)p_{u_i}(s_i) && (1)\\
&= p_{u_i,\theta^l}(x_i^l|s_i)\frac{e^{u_i s_i}}{1+e^{u_i s_i}} && (2)
\end{aligned}
$$

Where the prior is given by the shape model which was provided. This further gives the likelihood of a single pixel in a given image:

$$p_{u_i,\theta^l}(x_i^l, s) = \sum_{s_i \in \{0,1\}} p_{u_i,\theta^l}(x_i^l|s_i) p_{u_i}(s_i) \tag{3}$$

The above can be simply extended as a sum over all pixels to give us a likelihood of an image and then summed over all images to give us the log likelihood of the whole dataset.

$$L(u,\theta) = \frac{1}{m} \prod_{l \in T^m} \prod_{i \in D} \sum_{s_i \in \{0,1\}} p_{u_i,\theta^l}(x_i^l, s^i) \tag{4}$$

$$\mathcal{L}(u,\theta) = \frac{1}{m} \sum_{l \in T^m} \sum_{i \in D} \log \sum_{s_i \in \{0,1\}} p_{u_i,\theta^l}(x_i^l, s^i) \tag{5}$$

By using the bound:

$$\log \sum_{s_i \in \{0,1\}} p_{u_i,\theta^l}(x_i^l, s^i) \geq \sum_{s_i \in \{0,1\}} \alpha(s_i|x_i^l) p_{u_i,\theta^l}(x_i^l, s^i) - \tag{6}$$

$$\sum_{s_i \in \{0,1\}} \alpha(s_i|x_i^l) \log \alpha(s_i|x_i^l) \tag{7}$$

We get the objective function for the EM algorithm as:

$$(u^*, a^*, \theta^*) = \underset{u^*,a^*,\theta^*}{\operatorname{argmax}} \frac{1}{m} \sum_{l}^{m} \sum_{i \in D} \sum_{s_i \in \{0,1\}} [\alpha(s_i|x_i^l) \log p_{u_i,\theta^l}(x_i^l, s^i) - \alpha(s_i|x_i^l) \log \alpha(s_i|x_i^l)] \tag{8}$$

4 EM Algorithm

The EM algorithm works in two steps: The E (expectation) step assigns posterior values to the auxilliary variables and the M (Maximization) step maximizes the variables locally while keeping fixed auxilliary variables. It is essentially a blockwise coordinate ascent. Intuitively it can be though of as creating a tight lower bound by forcing the lower bound into an equality, and then maximizing the lower bound locally. This is performed until convergence.

To derive the E-step for the EM algorithm we can rewrite the lower bound as follows:

$$\log \sum_{s_i \in \{0,1\}} p_{u_i,\theta^l}(x_i^l, s^i) \quad \geq \quad \sum_{s_i \in \{0,1\}} \alpha(s_i|x_i^l) log \frac{p_{u_i,\theta^l}(x_i^l, s^i)}{\alpha(s_i|x_i^l)} \tag{9}$$

$$\log \sum_{s_i \in \{0,1\}} \alpha(s_i|x_i^l) \frac{p_{u_i,\theta^l}(x_i^l, s^i)}{\alpha(s_i|x_i^l)} \quad \geq \quad \sum_{s_i \in \{0,1\}} \alpha(s_i|x_i^l) log \frac{p_{u_i,\theta^l}(x_i^l, s^i)}{\alpha(s_i|x_i^l)} \tag{10}$$

$$f(E[X]) \quad \geq \quad E[fX] \tag{11}$$

Where

$$X \quad = \quad \frac{p_{u_i,\theta^l}(x_i^l, s^i)}{\alpha(s_i|x_i^l)} \tag{12}$$

And the expectation denotes the sum over $s_i \in \{0, 1\}$ of the product of $\alpha(s_i|x_i^l)$ and X.
We know that for the inequality to hold with equality the expectation of X must equal to X itself which means that

$$\frac{p_{u_i,\theta^l}(x_i^l, s^i)}{\alpha(s_i|x_i^l)} \quad = \quad const \tag{13}$$

This means that

$$\alpha(s_i|x_i^l) \quad \propto \quad p_{u_i,\theta^l}(x_i^l, s^i) \tag{14}$$

We also know that since $\alpha(s_i|x_i^l)$ is a probability distribution it should sum up to 1.

$$\alpha(s_i|x_i^l) \quad = \quad \frac{p_{u_i,\theta^l}(x_i^l, s^i)}{\sum_{s_i} p_{u_i,\theta^l}(x_i^l, s)} \tag{15}$$

$$\alpha(s_i|x_i^l) = \frac{p_{u_i,\theta^l}(x_i^l, s^i)}{p_{u_i,\theta^l}(x_i^l)} \tag{16}$$

$$\alpha(s_i|x_i^l) = p_{u_i,\theta^l}(s^i|x_i^l) \tag{17}$$

We can see that the E-step is simply assigning the posterior to the hidden variables

For the M-step we will simply manipulate the objective function to get maximization functions for each variable separately.
For the u variables we can expand the most inner sum and maximize over the individual images. This is because the parameter u_i for a certain pixel is shared across the images but is different for each pixel. We get:

$$u^* = \underset{u^*}{\operatorname{argmax}} \frac{1}{m} \sum_{l=1}^{m} \alpha(s_i = 1|x_i^l) \log(\frac{e^{u_i}}{1+e^{u_i}}) + \alpha(s_i = 0|x_i^l) \log(\frac{1}{1+e^{u_i}}) \tag{18}$$

$$= \underset{u^*}{\operatorname{argmax}} \frac{1}{m} \sum_{l=1}^{m} \alpha(s_i = 1|x_i^l)(u_i - \log(1+e^{u_i})) - \alpha(s_i = 0|x_i^l) \log(1+e^{u_i}) \tag{19}$$

$$= \underset{u^*}{\operatorname{argmax}} \frac{1}{m} \sum_{l=1}^{m} \alpha(s_i = 1|x_i^l)(u_i - \log(1+e^{u_i})) - (1 - \alpha(s_i = 1|x_i^l) \log(1+e^{u_i}) \tag{20}$$

$$= \underset{u^*}{\operatorname{argmax}} \frac{1}{m} \sum_{l=1}^{m} \alpha(s_i = 1|x_i^l) u_i - \log(1+e^{u_i}) \tag{21}$$

Where equation 21 is our final result. In the process we have ignored constant terms which will fall out of the process when deriving the equation. As for the concavity of equation 21, we can see that $\alpha(s_i = 1|x_i^l)u_i$ is a linear function which leaves us to verify whether $log1 + e^{u_i}$ is also a concave/convex function. We can prove it using one of the definitions of concavity: $f((\alpha-1)x + \alpha y) > (\alpha-1)f(x) + \alpha f(y)$. Verifying one such point is enough to induce the condition across the whole domain only if the function f is monotonic, which in our case it is. To avoid unneccessary cluter of basic calculus the verification of the condition is ommitted

here and only the result is stated that the condition holds with the opposite equality, meaning that the function is convex. The whole equation is then concave and has a unique global maximum.

For parameters θ_1, θ_0 we do a similar manipulation. This time we maximize through all the pixels in an image. We do this separately for each image because these parameters are specific to each image.

$$\theta_1^* = \underset{\theta_{1*}}{\operatorname{argmax}} \sum_{i \in D} \alpha(s_i = 1|x_i^l) \log p_{\theta^l}(x_i^l|s^i = 1) p_{u_i}(x_i^l) \tag{22}$$

$$= \underset{\theta_{1*}}{\operatorname{argmax}} \sum_{i \in D} \alpha(s_i = 1|x_i^l) \log p_{\theta^l}(x_i^l|s^i = 1) \tag{23}$$

Performing analogically for θ_0 we get

$$\theta_0^* = \underset{\theta_{0*}}{\operatorname{argmax}} \sum_{i \in D} \alpha(s_i = 0|x_i^l) \log p_{\theta^l}(x_i^l|s^i = 0) \tag{24}$$

5 Implementation

We will now take a slightly closer look at the E and M steps of the EM algorithm. As mentioned in the previous section, the E step is simple and involves simply assigning the posterior probability to the auxilliary variables.

$$\alpha(s_i|x_i^l) = p_{u_i,\theta^l}(s^i|x_i^l) = \frac{p_{\theta^l}(x_i^l|s) p_{u_i}(s)}{p_{\theta_1^l}(x_i^l|s=1) p_{u_i}(s=1) + p_{\theta_0^l}(x_i^l|s=0) p_{u_i}(s=0)} \tag{25}$$

The M-step, however is slightly more complicated and involves maximizing for the u and θ variables individually. We can take partial derivatives of the maximization equations with respect to the parameters that are being maximized to get the solutions.

$$\frac{\partial}{\partial u} \sum_{l=1}^{m} \alpha(s_i = 1|x_i^l) u_i - \log(1 + e^{u_i}) = 0 \tag{26}$$

$$\sum_{l=1}^{m} \alpha(s_i = 1|x_i^l) - \frac{e^{u_i}}{1+e^{u_i}} \quad = \quad 0 \tag{27}$$

Taking the maximization for a single image we have

$$\alpha(s_i = 1|x_i) \quad = \quad \frac{e^{u_i}}{1+e^{u_i}} \tag{28}$$

$$u_i^l = \log(\frac{\alpha(s_i = 1|x_i)}{1-\alpha(s_i = 1|x_i)}) \tag{29}$$

But since we have l images then the results would be also summed over l images.

The maximizations for the θ parameters are simple estimations of the mean and covariance of a multivariate gaussian. We take the partial derivatives for each θ_i individually with respect to the means and the covariances.
For the means this $in\ general$ decomposes into a simple average

$$\hat{\mu} = \frac{1}{N}\sum_{n=1}^{N} x_n \tag{30}$$

For the covariance matrices:

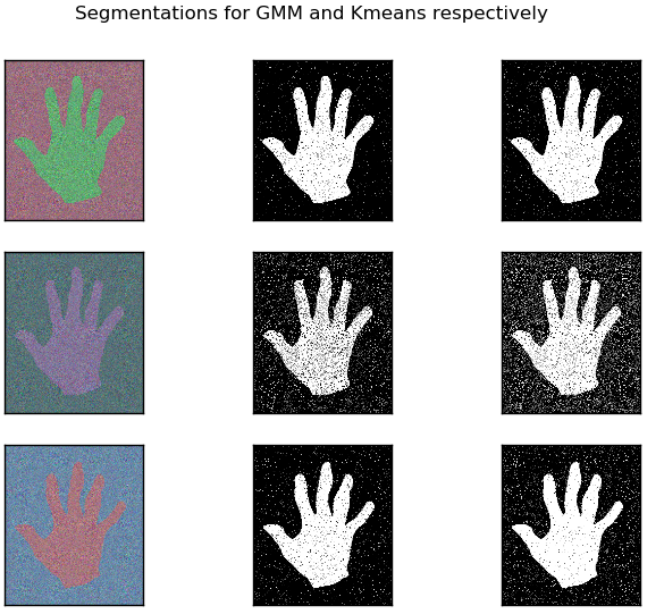
$$\hat{\Sigma} = \frac{1}{N}\sum_{n=1}^{N} (x-\hat{\mu})(x-\hat{\mu})^T \tag{31}$$

These are of course not in ready form and need to be summed over the appropriate α and normalized so that they can be used in the EM algorithm.

Results:

In the following figure we show resultant segmentations of a few baseline approaches. In these cases we ignore the shape model and simply

cluster the RGB vectors of each pixel in a given image into two groups. As the baseline approaches we use a k-means algorithm with two means and a Gaussian mixture model with full covariance matrix type and two mixture components. Both models are initialized using some appropriately cropped parts of the image which are assumed to be foreground and background respectively. If not initialized properly the segmentations of foreground and background are sometimes reversed.



The GMM approach gets an accuracy of 0.878 while the k-means reaches only 0.835. Both classifiers are evaluated on the whole dataset. The GMM is more accurate because it is a more powerful model and uses a full covariance matrix and has more parameters.

EM algorithm implementation:

Due to time constraints the implementation in python is only partially complete which means that it is not runnable.

6 Conclusion

This concludes the report on this task.