Machine Learning - Handin 4

Marc Skodborg, 201206073
Stefan Madsen, 201206043
Simon Fischer, 201206049
Group 8

December 2015

1 Status

We think the implementation works as intended, but for Expectation Maximization we were unable to reproduce the results from the book on the two dimensional IRIS data, yet it performs roughly as good as k-means. Also we occasionally get a divide by zero error.

2 Discuss plots of algorithm implementation

We are not entirely sure what and how we are supposed to plot to illustrate the algorithm running, so we did not.

If we initialize the EM algorithm with the best centers found by K-means, we get the following two F1-measures on the 4-d IRIS data:

$$\text{K-means F1} = 0.875981810666$$
$$\text{EM F1} = 0.868589411073$$

And for the 2-d IRIS data using the same approach:

$$\text{K-means F1} = 0.891774891775$$
$$\text{EM F1} = 0.891200870393$$

In both cases, we get a slightly worse clustering by EM, although practically an identical one. We are not certain this is the desired behaviour, or whether our EM-implementation is somehow faulty, which would also explain why our results differ from the book's when run on the same data.

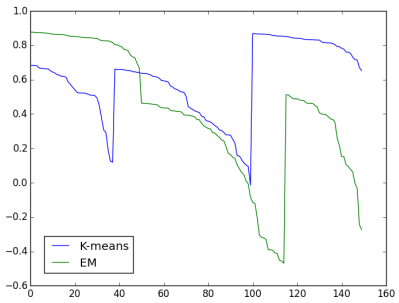
3 Discuss evaluation measures F1 and silhouette

The silhouette coefficient, as seen plotted in Figure 1 for two runs of the clustering, shows the relative performance of the two algorithms. The silhouette

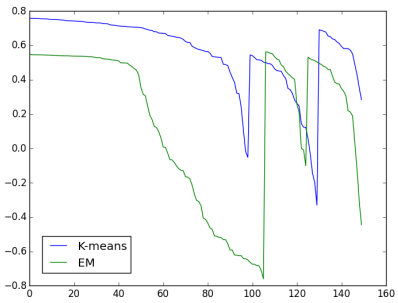
coefficient measures how good each point is clustered by comparing the average distances to points in the same cluster, and to points in the closest cluster. Values closer to +1 indicates average distances closer to points in the same cluster, i.e. a good clustering, whereas -1 indicates that a point is closer to points assigned a different cluster, a bad clustering of the point.

As we see in the first run in Figure 1a, the EM algorithm clustered points into the first cluster better than K-means did, as the first third of the green graph has higher and more even values than the blue one. Although it clustered the points in the first cluster better than K-means, it performs bad on the second and third cluster, with the graph being much more of a flat curve rather than the steep downwards curve of a good clustering as seen in EM for the first cluster. The EM-algorithm even dips below 0.0 for some points, indicating that they are in fact closer to one of the other two clusters and may have been clustered wrong. As the EM only performed better on clustering points into the first cluster, and K-means performs much better than EM on the two last clusters, we see a higher value for the F-measure in K-means, indicating a better overall clustering from this algorithm.

In the second run in Figure 1b, once again the K-means performs better than the EM algorithm. especially the first cluster is clustered great by K-means compared to EM, where the graph for EM has a very long downhill slope, spanning roughly 50 points, a third of the data set. As the K-means this time also dips into negative values, it is not as great a clustering as last run, which is also reflected by the F-measure of K-means in the two runs.



(a) K-means F=0.891774891
EM F=0.85677749



(b) K-means F=0.66110342
EM F=0.530583426

Figure 1: Silhouette plots for two runs

4 Image compression

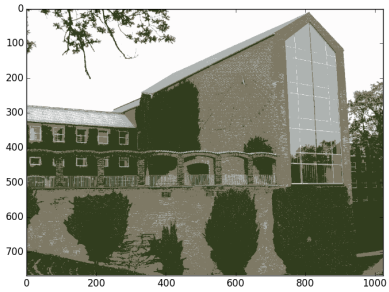
One way to compress an image would be to reduce the number of colours used. To achieve this you can use clustering to find, in our examples as seen in Figure 2, the four most representative colours, then it is just a matter of assigning colour

to each pixel according to clustering.

We are not sure how images are stored in memory, but we imagine that each pixel could take up less space if it is just an index into a table containing 4 different colours to make up the image, rather than a table of several million different colours, hence more bits are needed to index into the table, generating larger file sizes.

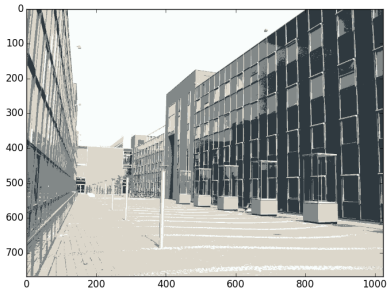
The quality of the image will be worse as we sacrifice a lot of possible colours to capture details in the image.

(a) Original



(b) Compressed to four colours

(c) Original



(d) Compressed to four colours

Figure 2: Image compression