Berkeley Segmentation Dataset and Benchmark (BSDS500)

Mariajosé Serna Ayala
Universidad de los Andes
m.serna10@uniandes.edu.co

Henry Daniel Torres Acuña
Universidad de los Andes
hd.torres11@uniandes.edu.co

Abstract

*Images can be segmented into regions by applying a mathematical model able to group data points closer in a given representation space. In this set of experiments we use k-means with feature space representation L*a*b, and Gaussian Mixture Model (GMM) with feature representation space L*a*b+xy to segment all images from the database BSDS500 from Berkeley computer vision group. First we ran the segmentation algorithms for train and validation sets to evaluate the best set of number of clusters (k). Then, we ran the algorithms for the test set with the set of k with higher average precision (AP) and segmentation covering(SC). Finally, in order to improve the segmentations we apply a median filter to the resulted segmentations. The best result was GMM + median filter on La*b*+xy representation space with an* $SC = 0.62$ *and* $AP = 0.15$

1.. Introduction

Image segmentation is the task of finding pixels that belong to a same group, without specifically defining these clusters[1]. Therefore, the purpose is to divide an image into discrete regions such that the pixels have elevated intra-similarity in each region and high contrast between regions. This similarity is defined different depending on the model used to separate the pixels. Also this similarity is defined as the le between the feature representation of each pixel. We used *Lab* and *Lab+xy*, as space feature representation for each pixel.

K-means is a variance-based clustering algorithm, and also the most famous.This is because its easy implementation and its relative simple mechanism. It is implemented by applying the Lloyd's algorithm which initially randomly assigns a k number of centroids to the data. Next, by optimizing the Euclidian distance, it assigns every point to the closest centroid. Later, the centroids are recomputed, to a better adaptation to the data is has been assign to. And the second and third point are reiterated until the centroids stop changing. The one parameter that needs to be previously set is the number of clusters (k). Which is a not so obvious when you do not have previous information of the image. Thanks to the euclidian distance it tends to form circular shape (in the representation space) clusters, which makes it an inconvenient for clustering data or images with different ways of distribution. Between other famous clustering algorithms (Watershed transform, hierarchical segmentation and Gaussian mixture models) is by far the most simple, and it gives reasonable results with a good representation. In this laboratory we decided to use the information of all three L*a*b channels to represent each pixel, and then cluster with k-means.

The Gaussian Mixture Model (GMM) is a clustering method based on fitting a mixture of Gaussian distribution to each group of points in the representation space. The data points are assigned to the normal components that maximize the component posterior probability of the data. In this method the goal is to first estimate, and later optimize the parameters of each Gaussian distribution that represents a group. For this, it is used the *Mahalanobis metric*. These parameters are defined with Expectation Maximization (EM) algorithm. One of the benefits of using this clustering alternative is the fact that it outputs for every point a soft-assignment (probability) of belonging to on of the formed clusters (Gaussian distribution). Also, unlike k-means, it does not tend to create groups of similar sizes or forms. Therefore is much more flexible defining clusters with uneven sizes inside a single image. The drawback of this methods is that it intrinsically assume the data is normally distributed, which is often not an accurrate assumption. We used GMM representing each pixel by L*a*b channels and the pixel position in x and y (length and width) (L*a*b+xy).

2.. Materials and Methods

Database

The BSDS500 is an extension of BSDS300 dataset, it uses 200 of its images as train, 100 as validation and adds 200 new test images[3]. All images have corresponding

ground truth segmentations, done by five different subjects on average. The performance of algorithms developed over this dataset is evaluated by measuring Precision / Recall on detected boundaries and other three additional region-based metrics.

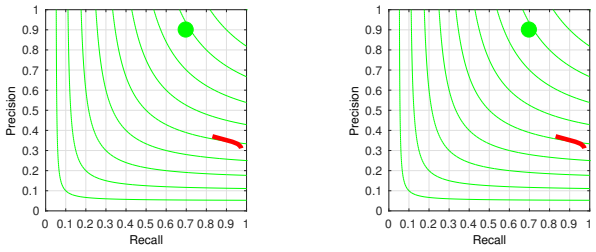
Train an validation images are supposed to be used for determining hyperparameters that the code may need. In our case, the only parameter we left undefined was k. But defining an optimal K for all images is difficult because every image has different number of objects (groups) inside it. Therefore, we defined a better set of k's between two different groups, each one with 6 different k's.

Clustering methods

As mentioned before the combinations of space representation and clustering algorithms where: GMM in L*a*b+xy, and K-means in L*a*b. For this we created a MatLab function with input parameters: cluster algorithm and K. The advantage of these two clustering methods is that both of them only have one parameter to define (k). Therefore, we did not need to assume other parameters. Even though we needed to do add a parameter to the implementation of GMM in MatLab, in the function *fitgmdist*, a *Regularization value* of 0.01. This value is used to introduce additional information in order to solve an ill-conditioned problem. This was needed because in some images the segmentation was not possible, which sometimes can be due to a large different between the data.

For each image we segmented with 6 different values of K. First we tried with K =[3,4,5,6,7] for the train set with k-means. Then for the segmentation of the validation set with K-means, and of the train set with K-means and GMM, we used K = [3,5,7,9,11,13]. Even though there was not a big difference between these two sets of K. The real difference was between the number of images in each set (validation and train). Validation, as mentioned before has 100 images and train has 200 images.

As an intention to improve our segmentations results, we applied a medial filter to the segmented images to remove any unwanted region from the image. We can use different neighborhood of n × n (filter size). But generally neighborhood of n = 7 is used because large neighborhoods produce more severe smoothing[4]. In order to see if it did really have a positive impact over the segmentations quality, we implemented the filter to the validation segmentations set done with K-means, evaluated, and then compare it to the segmentation no filtered (Figure3b and 3a) The increase in both, area under the precision recall curve and the F-measure, proves that the implementation of a median filter to the final segmentations actually works. Therefore we used it for the test images too.



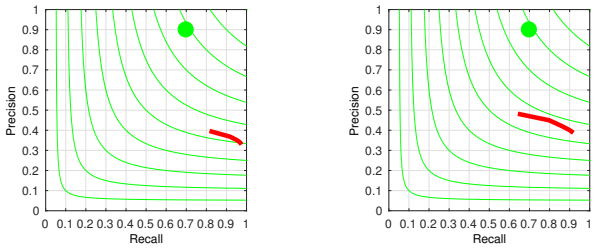
(a) (b)

Figure 1: Comparison between precision recall-curves for k-means segmentations in validation set. (a) Is for raw segmentations, $F - measure$: 0.51, $Area_PR = 0.05$. (b) Corresponds to the segmentations filtered with a median filter, $F - measure$: 0.54, $Area_PR = 0.12$.

Evaluation

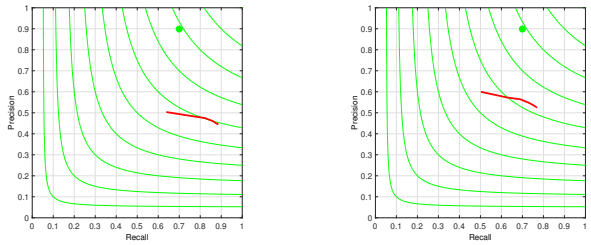
For the evaluation we used the library *bench fast* provided by the Berkeley computer vision group[3]. In this there are presented various benchmarks that are divides into two big groups: boundaries evaluation and quality of regions evaluation. There are implemented two different bechmarks for evaluation of boundaries, the average precision (AP) which is the area under the curve in a Precision-Recall graph and the F-Meassure that is the harmonic mean of the precision and recall values. For the quality of the segmented regions the library implements three different benchmarks: Variation of information, Rand Index and Segmentation Covering (SC).Since the segmentation covering benchmark explain with just one number the overlap between our segmented regions and the groundtruth, we decided to evaluate our regions using only this benchmark.

3.. Results



(a) (b)

Figure 2: Comparison between precision recall-curves for k-means segmentations. (a) Is for raw segmentations (no processing) and (b) corresponds to the segmentations filtered with a median filter.



(a) (b)

Figure 3: Comparison between precision recall-curves for GMM segmentations. (a) Is for raw segmentations (no processing) and (b) corresponds to the segmentations filtered with a median filter.

4.. Discusion

As it can be observed in the results section, the segmentation method that produces the best benchmarks results was GMM + median filter with an $SC = 0.62$ and $AP = 0.15$. Before the eployment of the median filter on the segmented masks we got a result of $SC = 0.61$ and $AP = 0.12$. The result of the average precision may arise as a low value, but this was an expected situacion since, because of to the computational restraints, it wasn't possible to evaluate a wider set of k's. Arguably, the improvement of the results observed when the mean filter is applied are due to the presence of salt and pepper noise in the segementation resultant noise. The median filter is suitable for the suppression of this noise because it eliminates the atypical values while preserving the boundaries. Also, it was observed that the predefined number of clusters affects directly the performance of the segmentation algorithm. Thus, the higher the number of clusters (k) the better the performance of our segmentation method on the benchamarks. It should be noted that it is necessary to performe further analysis with a wider set of k's in order to confirm this behavior.

The *bench fast* library allow us to position ourselves about the results of the Berkeley computer vision group. Although our results are more than average, there still aren't as excellent as theirs. There are many factors that influence the performance of the methods on the proposed benchmarks. First and foremost, the representation space they used to model each pixel is more robust than ours. For instance, while we only used Brightness information from channel L*, color information for channels a* and b* and spatial restrictions from coordinates x and y, they represented each pixel as a multiscale vector considering gradient information on brightness, color, texture channels and orientation as linear combination. Besides, they used the eigenvectors[3] information to add contour data on the gradient response over orientations for each eigenvector, thus highlighting a the boundaries of different regions of the image with each one.

The main problem with the algorithm developed was the feature representation space. We did not take into account that the coordinates of the representation L*a*b+xy needed to be normalized, same as the channels L, a and b. This is because otherwise the most relevant information is going to be the position. So, in k-means we used only L*a*b, because when we used not normalized L*a*b+xy representation it divided the image into groups mostly by its position. So we could have used a normalized L*a*b+xy representation space, and it would probably have increase both covering and the area under the curve. The most common error where presented by the k-means algorithm which segmented totally different object inside the same group, because it did not have much information than the color.

Certainly the presented algorithm has plenty of room for improvemt. A better representation space stands as a must have to achieve some progress in our performance. Regarding this, we propose to use additional information to the provided to La*b*+xy representation space. Texture information is crucial for humans to tell apart objects, therefore we propose the use of a texton map in which there are as many textons as the number of clusters provided by the user. Moreover, the introduction of a model that represents the probability that has a pixel to belong to a boundary. This can be achieve by applying to the original image a simple Canny Edge Detector and using the resulting mask to assing a boundary weight as new dimension of the representation vector. Finally, we propose to apply a typical normalization process to each feature on the representation space to equalize the wight that each dimension has on the clustering process.

5.. Conclusions

- Although the best result was obtained by applying GMM clustering method and the median filter on the L*a*b+xy representation space, GMM clustering method doesn't model accurately the pixel information since it assumes an intrinsic gaussian distribution of the data. GMM method clustered the given data better than k-means. However, this algorithm is computationaly greedy
- The application of the median filter eliminated the visible salt and pepper noise presented on the segmentation results. However, the price for this was a decrease on the recall of our segmentations.
- It is desirable to normalize all the dimensions that con-

form the representation vector for each pixel because it equalizes the influece of each feature on the clustering algorithm

- The two additional dimensions x and y gave very useful information about the space conformation of the pixels in the images. This allowed the clustering algorithm to mantain coherence when clasifying each pixel into groups. For instane, the conformational information can teach the algorithm to put in different cluster one pixel from a lamp and pixel from the orange shirt of the subject.

References

[1] R. Szeliski. Computer Vision: Algorithms and Applications. Chapter 5:Segmentations. 2011

[2] D. Martin and C. Fowlkes and D. Tal and J. Malik,A Database of Human Segmented Natural Images and its Application to Evaluating Segmentation Algorithms and Measuring Ecological Statistics, Proc. 8th Int'l Conf. Computer Vision, 2001

[3] P. Arbelaez, M. Maire, C. Fowlkes, J. Malik "Contour Detection and Hierarchical Image Segmentation". IEEE Trans. Pattern Anal. Mach. Intell.Washington, DC, may, 2011.

[4] N. Dhanachandra, K. Manglem, Y. Chanu, "Image Segmentation Using K -means Clustering Algorithm and Subtractive Clustering Algorithm" Procedia Computer Science, Volume 54, 2015, Pages 764-771, ISSN 1877-0509, http://dx.doi.org/10.1016/j.procs.2015.06.090. (http://www.sciencedirect.com/science/article/pii/S1877050915014143)