Berkeley Segmentation Dataset and Benchmark

Alejandro Trujillo
Universidad de los Andes
Bogotá, Colombia
af.trujilloa@uniandes.edu.co

Santiago Martínez
Universidad de los Andes
Bogotá, Colombia
s.martinez1@uniandes.edu.co

Abstract

Object segmentation in images is an important computer vision problem as it eases the issue of detection by making it unnecessary to use a sliding window technique in order to obtain semantic information of objects, because now it is only necessary to analyze segmented areas. BSDS (Berkeley segmentation dataset) is an approach of a database that uses a diverse set of images and human segmentations we call ground Truths to test algorithms against human performance in the problem of segmentation. We used this Dataset to compare a segmentation algorithm based on clustering developed by us against the one present in the benchmark gPb-ucm. Our best algorithm was the one that used GMM (Gaussian Mixture Model) for clustering using a feature space that included color (RGB) and spatial information. The information was normalized and "forced" to be in the same dynamic range. Its optimal F-measure was of 0,59 and the average performance over the dataset was of 0,4. Although this still is a very poor result in contrast to even worse methods than the gPb-ucm, we could improve our results by taking into account more information about the images like texture or local difference in color.

1. Introduction

In order to get a quantitative evaluation method when it comes to the segmentation problem, the University of California in Berkeley developed a dataset that contains images of varied origins, where every picture was segmented by humans serving as an annotation[1]. The goal of this laboratory is to use the BSDS500 to evaluate a segementation method developed by us and put it in contrast to the one made by the developers of BSDS500 (gPb-ucm). Functions obtained from the BSDS500 benchmark were utilized in order to evaluate our algorithm and compare them against each other and against gPb-ucm.

2. Methodology

2.1. Dataset Description

The data-base used for this experiment is the Berkeley Segmentation Dataset 500 (BSDS500), which is composed, as its name say, by 500 hundread images (200 for train and test and 100 for validation). All images in this database have either a landscape orientation (481x321) or Portrait orientation (321x421). All the images have a size of 37.7 Mb and the whole dataset consists of 250,9 Mb. The images are quite varied, as they include Landscapes, persons, animals, etc. The ground truth for this benchmark comes in a ".mat" format. Inside of it, one can find a cell array that, in each position, carries a structure which includes a binary matrix representing the segmentation made by the human corresponding to that cell and a matrix filled with integers which represents the regions (each number a different region) made by that same human. Each image was annotated by a different amount of humans. Given the following image:

Figure 1. Arbitrary Image in the train set

One finds regions made by 5 different humans as the ground truth. Here are the ground truths done in regions and Borders:

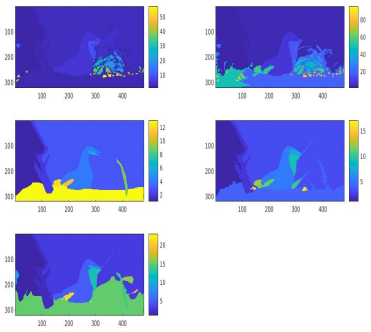

Figure 2. Regions Ground Truth

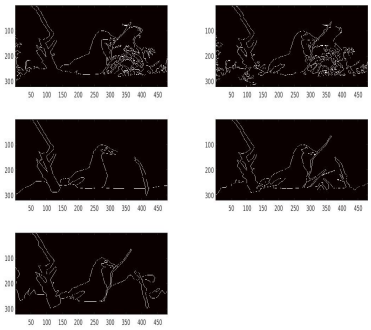

Figure 3. Borders Ground Truth

2.2. Evaluation Method

The evaluation method from this benchmark, in case of algorithms that produce regions is the following: Declaring the *overlap* between two regions R and R'

$$\omega(R, R') = \frac{|R \cap R'|}{|R \cup R'|} \tag{1}$$

The covering of a segmentation S by a segmentation S' is defined, as:

$$C(S' \to S) = \frac{1}{N} \sum_{R \in S} |R| \cdot \max_{R' \in S'} \omega(R, R') \tag{2}$$

where N denotes the total number of pixels in the image. likewise, the covering of a segmentation made by a computer S by a family of ground-truth segmentations Gi is defined by first covering S separately with each human segmentation Gi, and then averaging over the different results.[2]
After this, the Precision-Recall curve is computed by modifying a hyper-parameter (number of clusters in our case) and plotting it's results. We intent to compare our segmentation method to that of the BSDS benchmark. With the precission recall curve, we can get the F-Measure of our method, which is the harmonic mean of the precission and recall in a given point in the precission recal curve.

$$F - Measure = \frac{2(Precision * Recall)}{Precision + Recall} \tag{3}$$

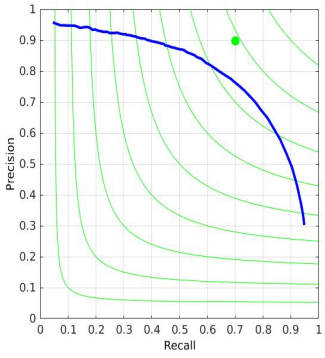

Figure 4. Precision-Recall curve for gPb-ucm in train

The graphic 4 show us the Precision-Recall curve for the gPb-ucm method, this curve tells us how the algorithm behaves with respect to the threshold given, as for lower thresholds, the algorithm will segment more borders.

2.3. Algorithm

Our Algorithm is a clustering based segmentation one, which hyper-parameter is the number of clusters. A python code was written in order to process and save information in *".mat"* files just as the BSDS500 benchmark requires it, as our segmentation method was developed in python and all the benchmarks were done in Matlab. This was done using an already developed code taken from *stackoverflow.com*[3] which used the scipy library to save data as a *.mat* and loading them in Matlab as type cell. The number of clusters used was varied from 2 to 202 in steps of 10. In order to enhance our algorithm, we ran some experiments on the train set using GMM-based clustering and varying the feature space, these can be seen in figure 5 and tables 1 and 2

2.4. Clustering Methods Used

The clustering methods used in this laboratory were K-means and GMM (Gaussian Mixture Model). These Methods were used because, doing a similar evaluation to the covering defined before, they performed considerably better than the watershed and hierarchical methods. Thus, we decided to use them. The K-means algorithm is a famous clustering method that, given a pre-defined number of clusters, divides the data comparing numerical information and minimizing the distance between each point of the cluster and its distance to the centroid of the cluster. It's an iterative

algorithm that does not necessarily find the global minimum of the energy function that defines the problem, as it starts with randomized centroids and stops iterating when there's no a significant change in the centroid's position. Due to its iterative nature, it's an algorithm that requires a considerable amount of memory and processing power. similar to the K-means algorithm, the GMM consists of adjusting an arbitrary number of gaussian distributions to the set of points in the selected feature space. The main advantage of GMM in comparison to the K-means algorithm is the possibility of clustering in different shapes, as K-means uses the euclidean distance which only permits circular clusters, while the Gaussian model is more flexible as it can take different shapes. Moreover, the GMM gives a "smooth" answer, as it's not a deterministic clustering, but gives a probability map of belonging to a given cluster, which gives more flexibility to the algorithm. In order to choose the number of clusters for the algorithm, we need to iterate over a wide range of these while evaluating each experiment and select the one with the best performance.

3. Results

3.1. Experiments

Firstly, We used the train data to iterate using the GMM method. the number of clusters selected was 9 (from 2 to 18), We didn't use a higher amount of clusters because the processing time augmented the more clusters we considered. The segmentation ran successfully on the server over the train data, it was necessary to run Matlab R2016 to be able to use the function as Matlab generates a mistake with the word "groundTruth" (since matlab 2017 it is a command), another problem was the type of "segs" files, the functions (allBench_fast) only worked with uint16, after solving these issues, the .txt files and benchmark ran successfully.

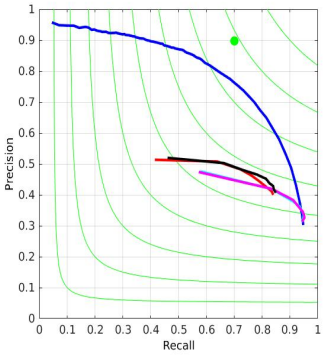

Figure 5. Plot evaluation gPb-ucm (blue line), GMM-Lab+xy (red line), GMM-rgb (cyan line), GMM-lab(magenta line), GMM-rgb+xy (black line), Evaluation over the Train Data

In figure 5 the Precision-Recall for the aforementioned experiments can be found. We used 4 different feature spaces and the GMM method. The best score showed was GMM-rgb+xy (RGB color space and spatial information) with near 0.5 precision and 0.4 recall, the precision seems constant until 0.65 recall, when it starts declining. Just looking at the graph we can already discard using feature spaces that only include color, as they perform considerably worse than those that include color and spatial information.

	F-Measure(ODS)	F-Measure(OIS)	APR	Best K/threshold
gPb-UCM	0.74	0.77	0.73	0.13
GMM-RGB+xy	0.58	0.64	0.21	6
GMM-RGB	0.56	0.61	0.13	4
GMM-lab	0.56	0.61	0.16	4
GMM-lab+xy	0.58	0.64	0.19	8

Table 1. F-Measures, Area under the curve and best k/threshold for the different methods

	Covering			RI		VI	
	ODS	OIS	Best	ODS	OIS	ODS	OIS
gPb-UCM	0.62	0.68	0.77	0.83	0.87	1.57	1.36
GMM-RGB+xy	0.45	0.53	0.63	0.73	0.79	2.21	2.03
GMM-RGB	0.42	0.48	0.53	0.70	0.78	2.26	2.22
GMM-lab	0.42	0.48	0.53	0.70	0.78	2.26	2.22
GMM-lab+xy	0.44	0.52	0.62	0.73	0.79	2.22	2.06

Table 2. Scores for the experiments and gPb-UCM in ODS, OIS (Train set)

The region Benchmark in the table 2 report the score, the first three columns show Covering used the optimal data scale (ODS), Optimal Image Scale (OIS) and the Best criteria, the three middle column represent Rand Index (RI) and three next the Variation of Information (VI). However we will focus on the covering as the other metrics are more complex to analyze and aren't as used nowadays in comparison to say the *Jaccard* index, which is analog to that of the overlap defined in section 2.2.

Only observing the ODS columns, the difference in segmentation is important, as the gPb-ucm score is over 0.62, our best method scores 0.45 (GMM-RGB+xy). This however is consistent with the evaluation method proposed in the last laboratory, where our best method was GMM using the feature space (RGB+xy). Looking at the F-Measure and the area under the curve, one can see that the most convenient feature space for this problem is thr RGB+xy one, as it yields a higher area under the curve in contrast to the lab+xy one, which implies a higher average precision for the algorithm, even though the difference is very subtle. Analyzing this results, we decided that the best feature space to use was RGB+xy. The final Evaluation consisted of using this feature space using the clustering methods K-means and GMM.

.

3.2. Final evaluation

For the final evaluation, The number of clusters was varied from 2 to 202 in steps of 10, in order to get a wide enough range for our Precision-Recall curve.

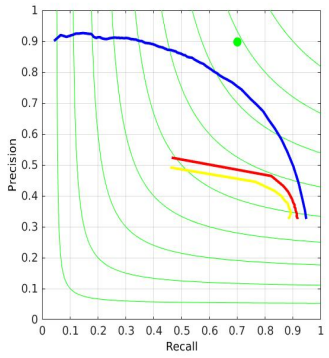

Figure 6. Plot evaluation folder val gPb-ucm vs GMM (red line) vs KM (yellow line)

	F-Measure(ODS)	F-Measure(OIS)	APR	Best K/threshold
gPb-UCM	0.74	0.77	0.73	0.13
GMM	0.59	0.62	0.22	12
K-Means	0.56	0.58	0.20	12

Table 3. F-Measures, Area under the curve and best k/threshold for the different methods

	Covering			RI		VI	
	ODS	OIS	Best	ODS	OIS	ODS	OIS
gPb-UCM	0.59	0.65	0.74	0.83	0.86	1.69	1.48
K-means	0.37	0.40	0.48	0.73	0.74	2.48	2.47
GMM	0.40	0.45	0.55	0.74	0.77	2.32	2.31

Table 4. Scores for our two final methods vs gPb-UCM in ODS, OIS (Test set)

In figure 6 We can see the gPb-UCM algorithm beats both our methods. This is because the probability of border and threshold approach gives a lot more flexibility to their curve, and their algorithm doesn't require clustering methods using one single feature space but uses information directly from the image like oriented histograms for multiple feature spaces like texture and color. We can also see that the GMM clustering based algorithm is better than the one that uses K-Means, this is due to the higher flexibility this clustering method has. In tables 3 and 4, one can see the performance of the three algorithms in question in more detail, which confirms the behaviour seen in figure 6.

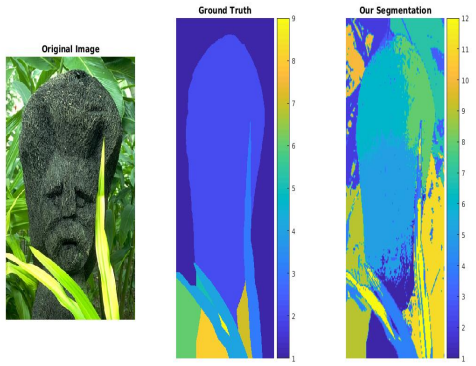

Figure 7. Example of our Segmentation

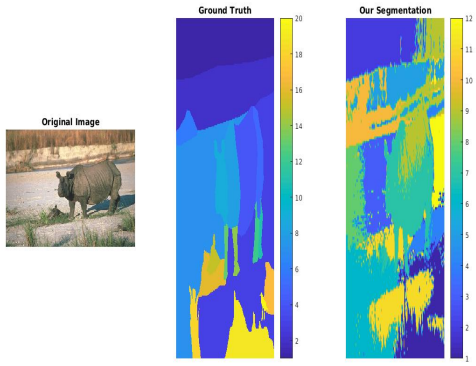

Figure 8. Example of our Segmentation

In figures 7 and 8, one can see qualitatively how our algorithm performs on 2 random images of the test set, Both segementations were done with GMM, feature space RGB+xy and k=12, which was the combination with the highest score.

4. Conclusions

The methods of GMM and K-Means, have an F-measure of 0.59 and 0.56 respectively over the test images, so comparing it to the BSDS segmentation method (gPb-UCM), which F-measure was of 0.74, they're both inferior methods. The BSDS benchmark allows us to compare segmentation algorithms with those that produce borders instead of regions in a fair way, generalizing the problem of segmentation. As expected, the best method was the gPb-UCM, and, in our case, it was the GMM based one. There is no significant difference between the recall-precision curves in the evaluation and test stages for each method, even though the range of clusters used was varied. In order to really obtain the optimal hyperparameter k, it would be necessary to use smaller steps and a wider range. In order to improve our

best method, we could use a more powerful feature space, including texture, local difference in color, etc. However, considering the segmentation problem a clustering one isn't the best aproach as the state of the art methods use deep learning and neural networks to achieve a higher performance.[4]

5. References

[1] David Martin, Charless Fowlkes, Doron Tal, Jitendra Malik, A Database of Human Segmented Images and its Applications to Evaluating Segmentation Algorithms and Measuring Ecological Statistics, University of California, 2001.

[2]P. Arbelez, M. Maire, C. Fowlkes and J. Malik, "Contour Detection and Hierarchical Image Segmentation", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 33, no. 5, pp. 898-916, 2011. Available: 10.1109/tpami.2010.161.

[3]T. Papastylianou, "Using scipy.io.savemat to Save Nested Lists", Stack Overflow, 2019. [Online]. Available: https://stackoverflow.com/questions/38960464/using-scipy-io-savemat-to-save-nested-lists. [Accessed: 12- Mar- 2019].

[4]K. Maninis, J. Pont-Tuset, P. Arbelaez and L. Van Gool, "Convolutional Oriented Boundaries: From Image Segmentation to High-Level Tasks", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 40, no. 4, pp. 819-833, 2018. Available: 10.1109/tpami.2017.2700300.