6.867 Project - Milestone 3

Kimberly Villalobos Carballo, Timothy Leplae-Arthur, Sean Fraser

October 26th, 2017

For our final project we are interested in exploring image segmentation techniques. Specifically, we wish to compare and analyze two different methods for image segmentation [4]: Edge Detection and Clustering. For these implementations we are using the Berkeley Segmentation Dataset and Benchmarks 500 (BSDS500) which consists of 500 natural images. For each image, several people were asked to draw a contour map separating different objects based on their own understanding[1].

Regarding the implementation/ software infastructure and algorithmic environment, the procedure we wish to carry out is below:

Edge Based Segmentation method using CNN:

Following the idea from Wang [5], we will train a CNN that takes an image as input and outputs the edge map of the corresponding edge map via classification on patches of size between $17 * 17$ and $21 * 21$. For each image patch, it is located on the edge map and the value at the center is the ground truth label, and these patch-label pairs, is what we try to classify and therefore minimize the loss on. For this part we will implement it according to the specific instructions outlined in the paper defining the layers/ architecture of the CNN. We will implement it from scratch using keras/tensorflow, and we will try a few loss functions such as the softmax classifier with cross-entropy loss or hinge loss etc.

Then we will iterate over neighboring pairs and triplets of pixels to transform the edge map into an adjacency matrix with positive entries for pairs of pixels that belong to the same segment, and negative entries for pairs of pixels in different segments.

Next, we will use Correlation Clustering(CC) functional to find the optimal segmentation sets (the partition with the highest CC score). For this part we will use the OpenGM package.

Finally, we will map the pixels from each partition set back to the image to get the segmented image.

Clustering Based method:

First the images are converted from RGB to $L*a*b$ color space using `makecform` and `applycform`. After that transformation all the color information is encoded in the a and b component of each pixel, and we can calculate the distance between any two of them by using the euclidean distance. Then for a specific k the centers are computed and for each pixel we assign the nearest center using the matlab implementation of k-means. This method focuses on minimizing the distance between pixels in the same cluster, and maximizing the distance between pixels in different clusters. The initial centers are chosen randomly and the optimal value for the number of clusters, k, will be the one that minimizes the loss function given by the sum of square distances between the points and their centers. For this section we will use matlab stats and image processing toolboxes.

Some questions that remain are post-processing techniques to improve the accuracy of the edge

detection and segmentation, some of which are described in Wang [5], for the CNN implementation.

Division of work: We will roughly split it so that two people are working on the CNN (Sean, Tim), as this requires some implementation from scratch and less documented packages, and one person working on the clustering / K-means method (Kim). However all three of us will work together on all parts of the project to some extent.

References

[1] Arbelaez, P.; Maire, M.; Fowlkes, C.; Malik, J. (2011) "Contour Detection and Hierarchical Image Segmentation" *IEEE TPAMI* 33.5: 898-916

[2] El-Sayed, M.; Estaitia, Y.; Khafagy M. (2013) "Automated Edge Detection Using Convolutional Neural Network" *International Journal of Advanced Computer Science and Applications* 4.10

[3] Xu, L.; Lu, C.; Xu, Y.; Jia, J. (2011) "Image Smoothing via L_0 Gradient Minimization" *ACM Trans. Graph.* 30.6

[4] Kaur, D.; Kaur, Y. (2014) "Various Image Segmentation Techniques: A Review" *International Journal of Computer Science and Mobile Computing* 3.5: 809-814

[5] Wang, R. (2016) "Edge Detection Using Convolution Neural Network" *Advances in Neural Networks - ISNN* 20: 741