6.867 Project - Milestone 4

Kimberly Villalobos Carballo, Timothy Leplae-Arthur, Sean Fraser

November 9th, 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].

Some of the initial results are shown in the diagram below:

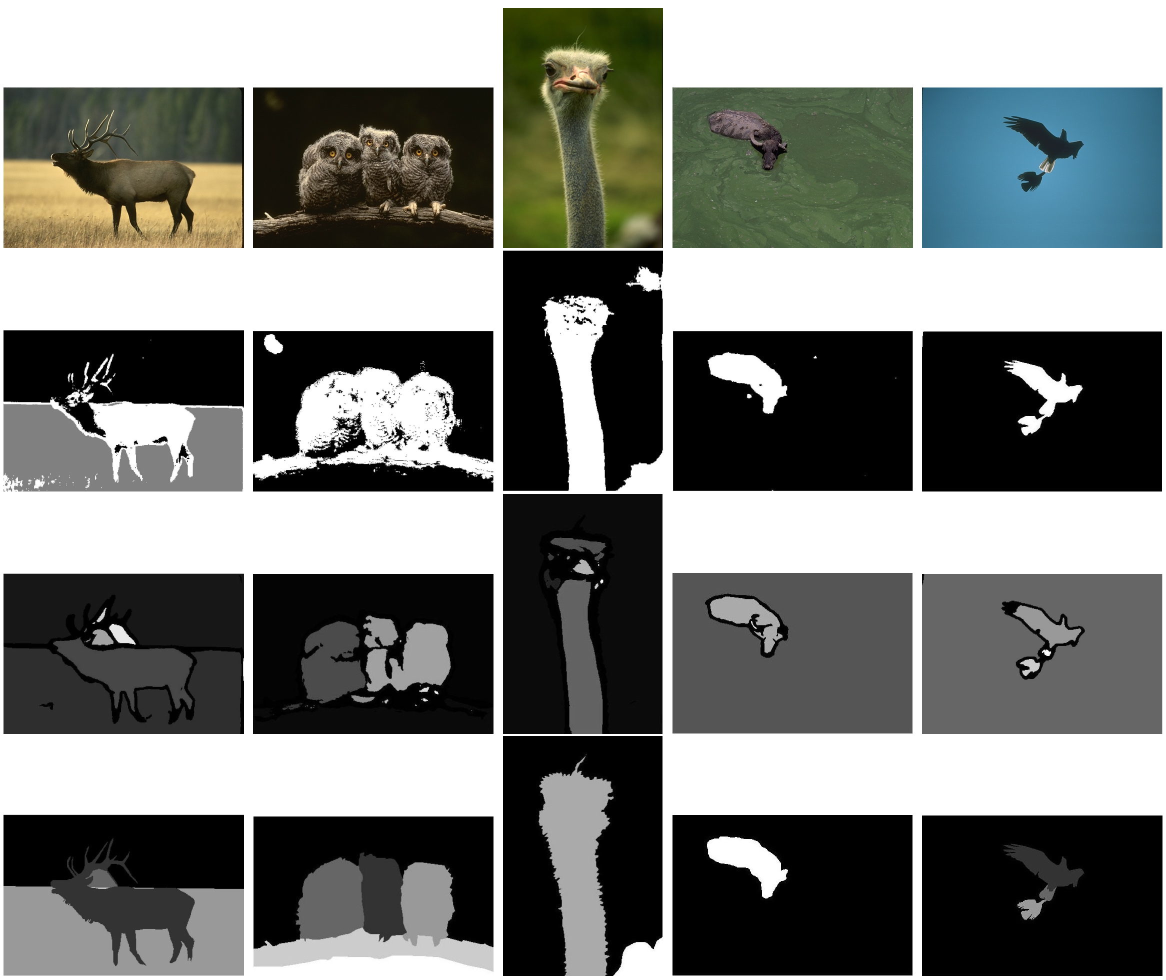

Figure 1: Comparison of the original image (1st row), Clustering based segmentation (2nd row), the CNN method (3rd row), and the ground truth images (4th row) for 5 different images from the dataset.

In the table below, the images are in the same order as above from left to right.

Table 1: Comparison of Results, using Rand Index and Varation of Information measures

Image	1	2	3	4	5
Clustering sumRI	0.910	0.670	0.777	0.892	0.990
Clustering sumVOI	0.818	2.099	1.219	0.336	0.114
CNN sumRI	0.912	0.928	0.843	0.884	0.974
CNN sumVOI	0.888	1.174	1.007	0.419	0.202

We produced the initial results according to the following procedure.

Edge Based Segmentation method using CNN:

To generate edge maps for our input images, we use the method of Holistically-Nested Edge Detection, and train a fully convolutional neural network on the BSDS500 dataset. Our CNN is composed of 5 components. Components 1 and 2 are composed of 2 subsequent convolutional layers, followed by a branching layer and a max pooling layer. Components 3, 4, and 5 are composed of 3 convolutional layers followed by a branching layer and a max pooling layer (except for component 5, which does not have a max pooling layer in the end). In each component, the branch layer allows for us to recreate the edge maps and explicitly see how the edge maps change after each component. As we expected, the edge maps of the outputs in each branch became increasingly smooth and defined. Using the outputted edge maps, we can run our inference algorithm to segment the foreground.

Seeing the initial edge maps, there are a couple improvements that we want to make that will lead to better segmentation. The first of these is picking the best output for our image. For inputs with simple foregrounds, the 4th outputted edge map (after the fourth component) may be the best, whereas for more challenging foregrounds, the 5th output may be best. The general heuristic that describes a good output is one that has defined, smooth thin edges. Running some images too deep into the CNN can thicken the edges which increase error after inference. Running some images too shallow in the CNN can output edges that are not defined enough.

Figure 2: Edge-map outputs of the CNN, from left to right: original, final output, 1st branch in CNN, 2nd, 3rd, 4th, 5th.

To perform the segmentation, we used the functions `evaluation_reg_image` from the BSDS500, which takes as input the boundaries/ edge-map and creates a segmented image to compare it to the ground truth.

Clustering Based method:

Using MATLAB Statistics and Image Processing packages, the images were 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.

In the above images (from left to right), the optimal k was 3 for the first one and 2 for the rest.

Pre- and Post-Processing:

For post-processing, we tried to smooth the images before performing the inference as outlined in the CNN method. This had minor to negligible improvements, which we have to fine-tune and re-assess for the next steps of the project.

Because the CNN uses convolutions of size 16, we cropped the images from 321 by 481 to 320 by 480. To improve our results we will also try Image Smoothing via L_0 Gradient Minimization, although these improvements may only be minor as well.

Evaluation of Results:

To compare the results of the Clustering and the CNN to the ground truth segmentations we used Rand Index (RI) and Variation of Information (VOI). The higher the probablistic Rand Index, and the lower the Variation of Information, the better the segmentation is. This is because the Rand Index measures the similarity between the samples, while the VOI measures the distance between the samples.

Therefore, looking at table 1 it is clear the the CNN performed better than the clustering, even at this early stage of the project.

We also compared the images using the Ground Truth Covering, which is the average ground truth covering. For the CNN it was 0.83 and for the Clustering it was 0.75. For these we used ODS and OIS measures (also included in the dataset).

Questions that remain include how to improve the results with the various pre-processing and post-processing techniques and fine-tuning, in addition to running our comparisons on the rest of the dataset. We also plan to vary the CNN to explore how different architectures affect overall accuracies.

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