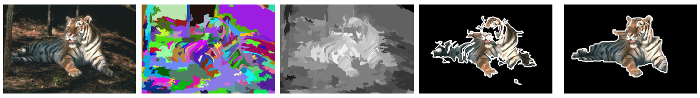
Saliency Segmentation based on Learning and Graph Cut Refinement

Paria Mehrani
pmehrani@uwo.ca
Olga Veksler
olga@csd.uwo.ca

Department of Computer Science
The University of Western Ontario
London, Canada



(a) original image (b) superpixels (c) confidences (d) classification (e) GC refinement

Figure 1: Summary of our approach.

(a) ground truth (b) saliency map (c) classifier result (d) GC refinement without confidences (e) GC refinement with confidences

Figure 2: Excluding vs. including confidences in Graph-cut segmentation

Saliency detection is a well researched problem in computer vision. In previous work [1, 3, 4], most of the effort is spent on manually devising a saliency measure which usually consist of three main steps. First, low level features, inspired by behavior and architecture of the early primate visual system, are extracted. Then for each feature, a saliency map is computed and finally, the saliency maps for each feature are normalized and combined, usually linearly.

A significant amount of effort have been spent by the previous approaches to design features that are relevant for saliency detection. Instead we propose a simple supervised learning algorithm that learns to detect saliency based on generic features often used in computer vision.

Most of the approaches produce a real valued saliency map. A pixel-accurate segmentation of a salient object, which is sometimes required, can be obtained by applying a fixed threshold on the saliency map. Another approach is to use an image segmentation algorithm to partition the image into regions and then apply a threshold on the mean segment saliency. None of these approaches produce accurate segmentations of the salient object. To avoid simply thresholding the saliency map while obtaining a pixel-accurate segmentation of the salient object, we refine the initial segmentation provided by the learned classifier by performing binary optimization based on graph-cut [2].

The overview of our method is in Figure 1. First an input image 1(a) is undersegmented into image regions 1(b), typically called superpixels. Next we extract features based on color, locations, size and texture and normalize feature maps. Our normalization measures the strength/weakness of features in one superpixel with respect to the other superpixels in the same image. After the features are normalized, we use a boosting algorithm with small decision tress for training to output a confidence for each superpixel separately 1(c). The confidence map is also taken to be our saliency map. We threshold at zero to get classification of the image into salient object and background, shown in 1(d).

The results obtained with a saliency classifier, even when reasonable, contain holes and inaccurate boundaries. This is especially true since we classify on superpixel level. Superpixel boundaries often do not coincide with salient object boundaries. To improve the boundary and coherence of classifier segmentation, we use binary graph cut optimization to refine the classifier results. The problem is formulated as binary salient object/background segmentation. Our energy function is of the standard form used in graph-cuts framework:

$$E(f) = \sum_{p\in\mathscr{P}} D_p(f_p) + \lambda \sum_{\{p,q\}\in\mathscr{N}} V_{\{p,q\}}(f_p, f_q). \tag{1}$$

In the equation above, $\mathscr{P}$ is the set of image pixels, f_p is the binary label assigned to pixel p, f is collection of pixel-label assignments, $\mathscr{N}$ is a 4-neighborhood system, D_p is the data term, and $V_{\{p,q\}}(f_p, f_q)$ is the smoothness term for two neighboring pixels.

The smoothness term is the one from [2] which is

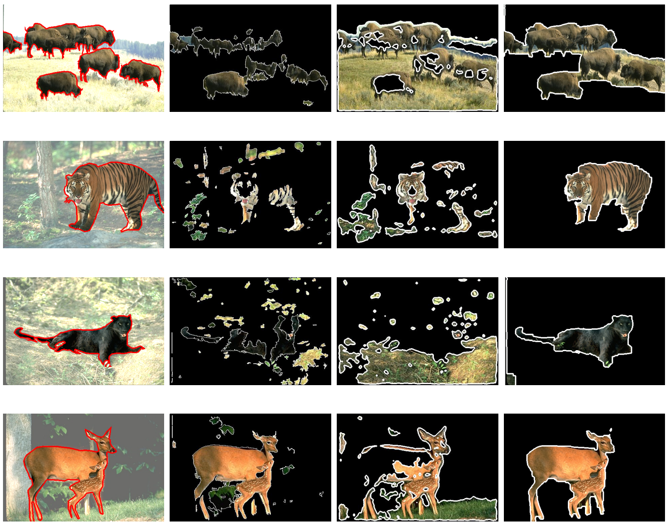
$$V_{\{p,q\}}(f_p, f_q) \propto \exp(-\frac{\Delta I^2}{2\sigma^2}) \cdot \delta(f_p \neq f_q) \tag{2}$$

Where ΔI denotes the intensity difference of two neighboring pixels, σ^2 is the variance of intensity difference of pixels, and $\delta(\cdot)$ is 1 if its argument is true and 0 otherwise. This $V_{\{p,q\}}$ encourages the boundary between labels to align with significant image edges.

Our data term consists of two parts. First of all, the classifier segmentation (Fig. 1(d)) is used to construct a color appearance model for the salient object and the background. This appearance model can help to fill the missing holes by reassigning pixels similar in appearance to the salient object (or background) to the appropriate label.

If we use only the appearance models for graph cut segmentation, the final segmentation can completely disregard the classifier results. Therefore, to make sure that the refinement with graph cuts does not differ drastically from the classifier results, in the second part of our data term we include the magnitude of the saliency map. Figure 2 illustrates graph-cuts optimization with and without the magnitude of saliency map term.

We evaluate our algorithm on three different datasets. In addition, we directly compare the performance of our algorithm with Achanta [1], Itti [4], and Hou [3]. Our algorithm out-performs the three algorithms for all the three datasets, i.e. our algorithm error rate is **20.12%** on Berkeley Segmentation Dataset compared to the error rates **25.47%**, **26.64%**, and **30.63%** obtained from Itti's, Hou's, and Achanta's algorithms respectively. Figure 3 shows some examples of successful salient object segmentation of our method, and compare them to Achanta's [1]and Hou's [3] methods.



(a) ground truth (b) Achanta (c) Hou (d) our result

Figure 3: Comparison of the performance for different methods.

[1] R. Achanta, S. Hemami, F. Estrada, and S. Süsstrunk. Frequency-tuned salient region detection. In *Conference on Computer Vision and Pattern Recognition*, 2009.

[2] Yuri Boykov and Gareth Funka-Lea. Graph cuts and efficient n-d image segmentation. *International Journal Computer Vision*, 70(2): 109–131, 2006.

[3] X. Hou and L. Zhang. Dynamic visual attention: searching for coding length increments. In *Advances in Neural Information Processing Systems*, pages 681–688. 2009.

[4] Laurent Itti, Christof Koch, and Ernst Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(11):1254–1259, 1998.