Comparing Supervised vs. Unsupervised Image Segmentation Methods

Guang Wei Yu
University of Toronto
guangwei.yu@mail.utoronto.ca

Richard Zemel
Dept. of Computer Science
University of Toronto
zemel@cs.toronto.edu

Abstract

This project compares the supervised logistic regression segmentation algorithm against the unsupervised k-means clustering segmentation. We observed that the difference between either method is not very significant. When performed on the 100 test cases for BSD300, the supervised method on average achieved a precision rate of 0.47 and the unsupervised method achieved a precision rate of 0.41. The variance in performance for both method is quite high, indicating that the models performed well for some images but not for other. This effect seems more prevalent for the supervised result. The standard deviation values for the precision rate are $\sigma_{supervised} = 0.18, \sigma_{unsupervised} = 0.16$

1 Introduction

The goal of the segmentation problem is to represent images in a simpliefied and meaningful manner by partitioning them into multiple groups. Practical applications of image segmentation include medical imaging, video surveillance, as well as means of pre-processing for a magnitude of detection and recognition tasks.

Supervised segmentation algorithms use a priori knowledge involving the ground truth of a training set of images, while unsupervised algorithms is trained online during segmentation. This project aims to examine results from each of the categories of algorithms.

For supervised segmentation, we use logistic regression to learn a prior model for the features used in segmentation. For unsupervised algorithm, we will use a simple k-means algorithm to cluster features and examine results for various choice of k.

2 Background

2.1 Database

For this project, we use the Berkeley Segmentation Dataset 300 (BSD300) [2], which contains 300 images with numerous human-marked ground-truth segmentation labels. Of these, 200 are treated as training images and 100 are treated as testing images.

2.2 Superpixel Segmentation

We first pre-process the image by over-segmenting it using superpixel segmentation. The algorithm we used for this project is the simple linear iterative clustering (SLIC) method. This is an unsupervised algorithm that uses local k-means of predetermined $k = (\#$ of superpixels$)$ to over segment the image into superpixels. The initialization is a uniform grid structure to ensure that the resulting superpixels are relatively uniform.

The reulsting superpixel image is shown in fig. 1. We treat these superpixels as indivisible units in our segmentation problem, which becomes a clustering problem of these superpixels. We note that finer details in the segmentation boundary can be lost from the superpixel representation, but in the overall context of meaningful representation, we do not care about these details. This is shown in fig. 2 where in terms of context, the ground truth (left) and superimposed ground truth on superpixels (right) are almost identical.

We found that a superpixel segmentation of around 400 superpixels worked well in representing the ground-truth accurately and consistently.

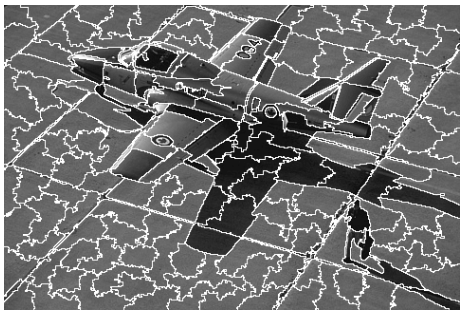
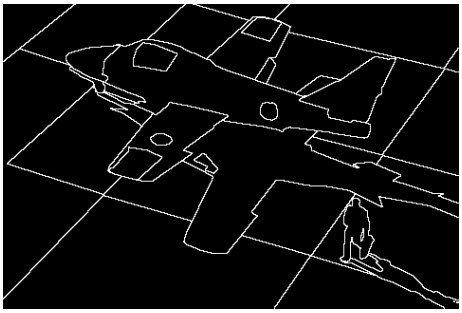
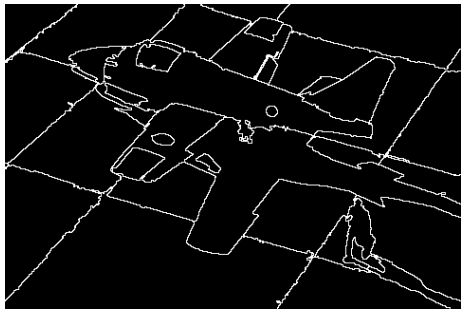


Figure 1: SLIC Superpixel Segmentation



(a) training labels



(b) superimposed labels

Figure 2: Training Lables Imposed on Superpixel Image

2.3 Feature Extraction

After the pre-processing, we extract features at the superpixel level. For this project, we focus on three basic features: the image intensity, image texture, and edge detection.

We extract image intensity at pixel level and average them to obtain superpixel intensity. These are used to produce birghtness similarity between same class and different class superpixels (for supervised algorithm) or features for clustering (for unsupervised algorithm).

We represent image texture by using historgrams of the intensities. We compute the pixel level histogram for each superpixel by dividing intensity range into 256 bins from 0 to 255 and sampling each superpixel to obtain a representation of the texture. Again, we compare the texture similarity

(characterized by χ^2 distance) and formulate difference measure (for unsupervised algorithm) or use as input to logistic function (for supervised algorithm).

To help segmentation, we also use edge detection filters. In particular, we take the response of the image convolved with the Gabor filter at pixel level. These responses are then averaged at the superpixel level to produce filtered response as our feature vector.

2.4 Comparison of Result

For meaningful segmentation result comparison, we look to the paper by Estrada and Jepson [1] on benchmarking image segmentation. The precision and recall scores are used to determine the quality of the segmentation algorithms for this project. These are given by (1) and (2) as

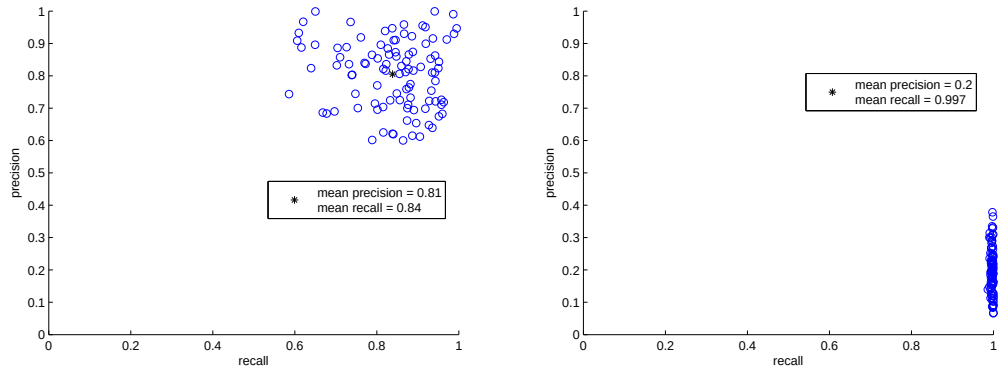
$$\text{Precision} = \frac{\text{Matched}(B_{source}, B_{target})}{|B_{source}|} \tag{1}$$

$$\text{Recall} = \frac{\text{Matched}(B_{target}, B_{source})}{|B_{target}|} \tag{2}$$

, where source and target indicate the classifier's segmentation and ground-truth respectively, B denote the boundary pixels, and $|B|$ denoting the number of boundary pixels.

The correspondence algorithm used to determine these matching scores are same as the paper, since the evaluation was done on the BSD as well with regard to human ground-truth boundaries. The score is computed based on bi-partite matching originally proposed by Martin [3] and improved in computation efficiency by Estrada and Jepson [1].

To generate some comparison data, we show the quantitative result of superpixel boundary (no meaningful segmentation apart from superpixel preprocessing) and human-label boundary (ground truth) under this method of evaluation in fig. 3. The ground truth has relative high precision, but the low recall indicate the discrepancy amongst each human labeler. The precision recall graph for the superpixel boundary result should indicate the precision rate without any algorithm. The successful algorithms should beat the average precision rate of 0.2 if they were to provide any use. The near perfect recall is simply telling us that the superpixel boundary themselves can capture any human labels extremely well, which confirms with what we have mentioned before.



(a) ground truth (b) superpixel boundary

Figure 3: Comparison Algorithm on Raw Data

3 Methods

To meaningfully compare between supervised and unsupervised segmentation, we will first apply the preprocessing described above to obtain superpixel representation of the image. We will then treat

the segmentation as clustering problem where instead of the segmentation boundary B, we solve for a clustering pattern of the superpixels.

We then convert the cluster labels to boundary maps, which will be the boundary B_{source} that we compare. This id done by shifting the image one pixel to the right and one pixel down, and compare the shifted labels to the original unshifted version to obtain maps of change in labels.

We compare the quality of B_{source} between ground-truth B_{target}, we adopt the quantitative measure mentioned in section 2.4.

3.1 Supervised Segmentation

In supervised segmentation, we further reformulate the superpixel clustering problem as binary classification problems of same vs different class for each superpixel (C_1 and C_0). This is repeated for all superpixel $\mathcal{S}$ where each superpixel $p_i \in \mathcal{S}$ is compared to all superpixels $q \in \mathcal{S}, q \neq p_i, \forall i$, avoiding repeated comparison of all previous p_i. For each unlabeled superpixel p_i, we begin by assigning p_i an arbitrary unique label t_p. We notate t_p, t_q as the temporary segment labels assigned to superpixel p, q respectively, and the set of all superpixel is $\mathcal{S}$. Then for all unlabeled superpixels q we treat the problem as a binary classification problem $S(p, q)$ where the result class $C_1 : t_p = t_q$ when p and q will belong to same segment, and $C_0 : t_p \neq t_q$ otherwise. We do this by modeling the binary classification poundary as posterior probability given by (3).

$$S(p,q) = \begin{cases} C_1 : t_p = t_q, & P(C_1|\mathbf{w}, \mathbf{F}(p,q)) \geq P(C_0|\mathbf{w}, \mathbf{F}(p,q)) \\ C_0 : t_p \neq t_q, & P(C_1|\mathbf{w}, \mathbf{F}(p,q)) < P(C_0|\mathbf{w}, \mathbf{F}(p,q)) \end{cases} \tag{3}$$

We repeat this problem as many times until no segments is left unlabeled. . The segmentation is then the boundary of each of these class labels.

For the binary classification problem of superpixel p and q, we focus on determining whether superpixel p and q belongs to same class, where we assign $t_p = t_q$. The class itself does not matter in this model, which means we only need a binary classifier per superpixel. We assign a cost function $f(p, q)$ to describe the cost it takes to classify p and q into same class. The cost (4) is a linear combination of feature vector $F(p, q)$ and trained weights w.

$$f(p,q) = \sum_j w_j F_j(p,q) \tag{4}$$

We apply logistic regression to the linear cost function as $y = \sigma(f(p, q))$ to model the posterior, where $\sigma(z) = \frac{1}{1+e^{-z}}$ is the sigmoid function. The linear binary classifier obtained models the posterior as (5) after simplification.

$$\begin{aligned} P(C_1|\mathbf{w}, \mathbf{F}) &= \frac{1}{1 + e^{f(p,q)}} \\ P(C_0|\mathbf{w}, \mathbf{F}) &= \frac{e^{f(p,q)}}{1 + e^{f(p,q)}} \end{aligned} \tag{5}$$

To train the weight $\mathbf{w}$ in our logistic function, we formulate the objective for the binary classification problem as the cross-entropy in (6) which we wish to maximize. The model classification boundary is now C_1, C_0 while the ground truth are either $t_p = t_q$ or $t_p \neq t_q$ depending on human labels.

$$l(\mathbf{w}) = \sum_n \left(\sum_{\substack{(p,q)\in\mathcal{S}_n \\ q\neq p \\ t_p = t_q}} \log P(C_1|\mathbf{w}, \mathbf{F}(p,q)) + \sum_{\substack{(p,q)\in\mathcal{S}_n \\ q\neq p \\ t_p \neq t_q}} \log P(C_0|\mathbf{w}, \mathbf{F}(p,q)) \right) \tag{6}$$

We maximize (5) by setting its derivative with respective to $\mathbf{w}$ to 0 and solve for $\mathbf{w}$. We do this using gradient descent to update the weight iteratively given the large data set $\mathcal{S}$. We note that here we compare the superpixels p, q within each image only, where $\mathcal{S}_n$ is the set of superpixels in image n.

The training is done simultaneously for the 200 training images given by BSD300. We specify 400 superpixels per image. If we trained every superpixel pair, this would amount to a training data set of 32×10^6 which is too much. Instead, we train every segment pair for each image, using an average of the superpixel feature vectors belonging to each segment. This results in a total of 98671 segment pairs for the training case.

3.2 Unsupervised Segmentation

The unsupervised segmentation algorithm studied in this project is a k-mean clustering algorithm with redescending distance measure as robust error estimators. The robust estimators allows better treatment of outliers which would be significant for a general image.

We cluster all superpixel $p_j, j = 1, ..., 400$ into k clusters and label them $t_i, i = 1, ..., k$. The k-mean clustering algorithm then seeks to minimize the robust cost function (7).

$$\begin{aligned} l(\mathbf{w}) &= \sum_{i=1}^{k} \sum_{p_j \in t_i} \rho\left(\|\mathbf{x}_j - \boldsymbol{\mu}_i\|^2\right) \\ \rho(e^2) &= \frac{e^2}{\sigma_k^2 + e^2} \end{aligned} \tag{7}$$

For this project, we will use the Geman-McClure norm with the k-mean clustering algorithm and $k = 3, \sigma_k^2 = 10$. We noticed that the k-mean alogirhtm is extremely sensitive to choice of k such that although there were approximately 10 different segments in each image, we had to pick much smaller value for k.

We proceed to extract the cluster boundaries as the segmentation boundary to be tested as before.

4 Result and Analysis

The results are from the 100 test images from BSD300 are plotted together on a precision vs. recall scatterplot in fig. 4. We see that the average precision rate is around 0.4 for both algorithm which is higher than the 0.2 rate for raw superpixel precision, but still much lower than human result. The precision for supervised result are slightly higher at 0.47 on average compared to the 0.41 of unsupervised. The quality of segmentation result varies significantly from picture to picture, and seems more so for the supervised case compared to the unsupervised k-mean, likely due to the more complicated learning involved. The standard deviation for supervised algorithm precision rate is $\sigma_{supervised} = 0.18$, and for unsupervised is $\sigma_{unsupervised} = 0.16$. Furthermore, the result is far from ideal benchmark results found in paper by Estrada and Jepson [1] which seems to be around a precision of 0.6.

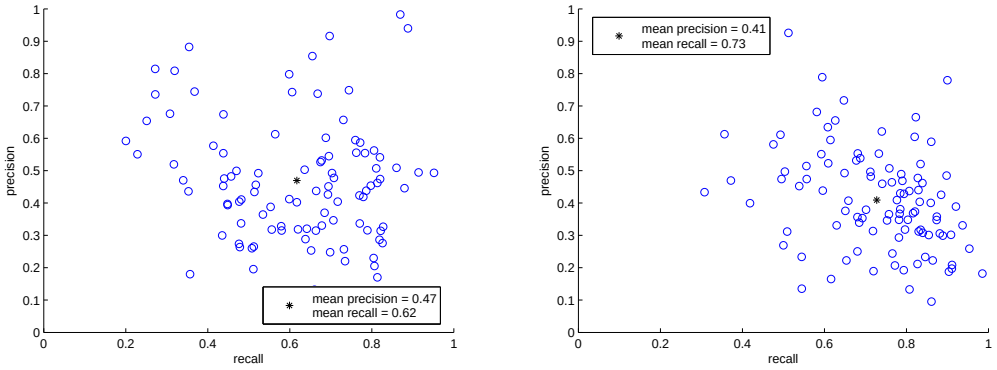
In addition to quantitative comparison, we also examine a sample of results visually for the segmentation algorithms in figs. 5 to 7. We can see that both algorithm can capture definitive segmentation boundaries that are immediately obvious to a human. It seems the unsupervised kmeans algorithm like to capture more details and come up with more discontinuous segments than the supervised results, which can be good or bad. However, even in visual inspection of the results, we cannot conclusively say which algorithim outperforms the other one.

References

[1] Francisco J. Estrada and Allan D. Jepson. Benchmarking image segmentation algorithms. *International Journal of Computer Vision*, 85(2):167–181, 2009.

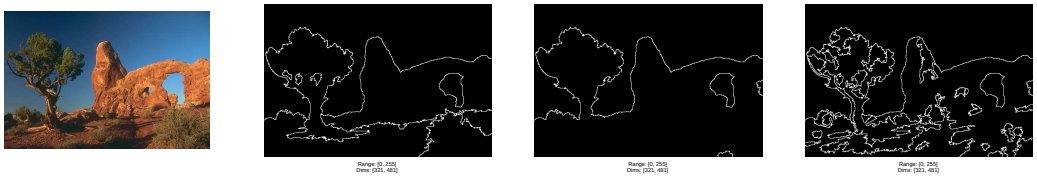
[2] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proc. 8th Int'l Conf. Computer Vision*, volume 2, pages 416–423, July 2001.

[3] David Royal Martin. *An Empirical Approach to Grouping and Segmentation*. PhD thesis, EECS Department, University of California, Berkeley, Aug 2003.



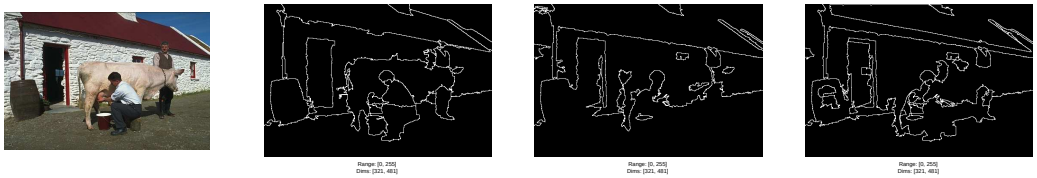
(a) supervised result (b) unsupervised result

Figure 4: Training Lables Imposed on Superpixel Image



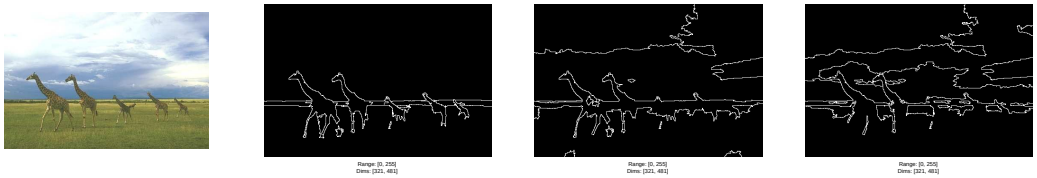
(a) original image (b) ground truth (c) supervised result (d) unsupervised result

Figure 5: Visual Segmentation Results 1



(a) original image (b) ground truth (c) supervised result (d) unsupervised result

Figure 6: Visual Segmentation Results 2



(a) original image (b) ground truth (c) supervised result (d) unsupervised result

Figure 7: Visual Segmentation Results 3