Image Segmentation Using Conditional Random Fields

Roy Amante Salvador and Daryll Panaligan
Department of Computer Science
College of Engineering, University of the Philippines Diliman
Quezon City Philippines, 1101
(632) 434 3877

***Abstract*—In this paper, we propose the use of conditional random fields (CRFs) to address the challenge of image segmentation. As part of pre-processing the data, we perform oversegmention on the training images to represent them as a group of superpixels. By considering each superpixel as a vertex, we are able to utilize CRFs with improved computational efficiency compared to using individual pixels. We identified several unary features such as the superpixel color, histogram, and the histogram of oriented gradients. For the pairwise features, we considered the color difference, histogram difference, and texture similarity. We also discovered that the considering the location of the superpixels relative to the image had little effect in improving the performance of the model. By experimenting with the different combinations of unary and pairwise features for the model on the Weizmann Horse Dataset, we are able to develop a model that showed good accuracy.**

***Keywords*—Image Segmentation, Conditional Random Field (CRF), Structured Support Vector Machine (SSVM)**

I. Introduction

With the increasing demand for improvements in the field of computer vision, the problem of recognizing and labelling objects and properties in an image has been widely discussed. While it is a simple and natural task for humans, this task is a challenge for computers. Several practical real-time applications, such as object recognition systems in self-driving vehicles, require solutions to this challenge that exhibit exceptional performance in both speed and accuracy to ensure the reliability of the system.

In the problem of image segmentation, the task is to classify each pixel into the several classes that are present in the image. By doing so, we get the result of segmenting the image into each class and recognizing each segment as one of the corresponding classes. In the example of self-driving vehicles, this would enable its systems to classify objects in its vicinity as a person, a stationary object such as a pole, or another vehicle. This allows the system to create quick judgements based on its environment.

A common approach to the image segmentation problem are conditional random fields (CRFs). While originally used only in the field of text processing, it has been recognized that CRFs can provide a solution to the problem of image labelling and segmentation.

In this paper, we have a simplified version of image segmentation problem which only contains two classes, the foreground and the background. Using the Weizmann Horse Dataset for training and testing, we specifically segment the image into two classes, the horse and the landscape.

In performing image segmentation with CRFs, identifying the features to be used is an important task, as these features have a significant impact on the performance. We aim to identify both the unary and pairwise features to propose a CRF model that would allow us to achieve good performance with the given dataset.

II. Related Work

While several proposed solutions to the image segmentation problem utilize CRFs, studies such as [3] and [7] exist where other methods are used. In [3], a kernelized version of structured support vector machines (SSVM) learning was used. An SSVM is a variant of the SVM classifier where it allows the training of classifier for general structured output labels such as graphs, parsed trees, etc. By designing several novel non-linear kernel functions, they were able to propose a different supervised learning approach to the segmentation task. Using the Weizmann Horse Dataset, they were able to achieve an accuracy of 94.63%. This study utilized a type of feature called the histogram of oriented gradients (HOG), which we also utilized in this study. This is described in detail in section III-C1d.

Shotton et al. [7] proposed the use of *semantic texton forests*, which are variant of randomized decision forests used for both clustering and classification. In this study, they found that the CIELab color space generalized better than RGB in their method. In utilizing the CIELab color space, the results of our method have shown the opposite.

Meanwhile, Liu et al. [15] has shown a solution using CRFs. Instead of using hand-crafted features, as other previous studies have done, they utilized deep convolutional neural networks (CNNs) for their ability to learn features that accommodate within-class variance and at the same time possess discriminative information. From the results of this study, which yielded an accuracy of 95.7% on the same dataset, we've attempted the use of CNN features as unary potentials, and the use of SSVMs to learn the parameters of our CRF model.

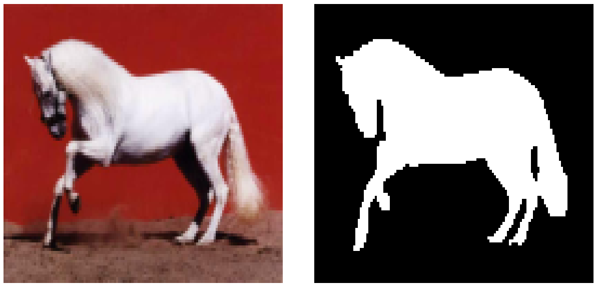

Fig. 1. Weizmann Dataset Horse and label image pair instance

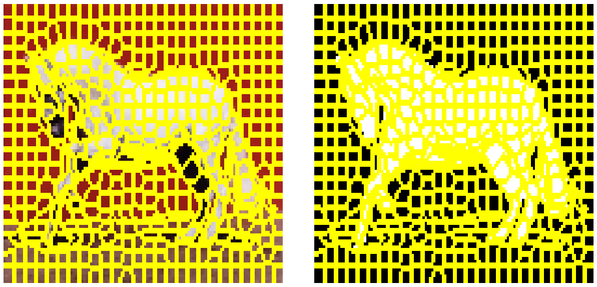

Fig. 2. Superpixels produced by Simple Linear Iterative Clustering (SLIC) Image on the right shows the superimposed superpixels on the label image

III. Methodology

A. Dataset

The Weizmann Horse Dataset [4] consists of 328 side-view color images of horses which were randomly collected from the World Wide Web. The images are quite challenging as they contain horses with very high intra-class variability in terms of color and texture (e.g. cow-like) with different postures (running, jumping, standing, eating, etc.) on highly cluttered and varying backgrounds. Each image in the dataset was manually segmented and has a corresponding label image where each pixel is annotated as horse pixel or a non-horse pixel.

Figure 1 shows a sample instance in the dataset. White pixels comprise the horse region while black pixels comprise the background. We used Shotton's [6] version of the dataset which normalised the images, reducing their sizes to somewhere between 100x100 and 200x200 pixels. The dataset was split evenly, with the training and test sets each containing 50% of the samples.

B. Oversegmentation

Oversegmentation is a pre-processing step where objects for segmentation are further segmented themselves into sub-components. A widely used oversegmentation method in the field of computer vision is to divide images into superpixels, which are logical grouping of pixels in the grid based on perceptual and semantic meaning. While there are several unnecessary boundaries generated by this method, most of the of the significant boundaries are found. This means that a small number of pixels are lost when mapping pixels to superpixels. Another advantage of representing an image with superpixels versus the individual pixels is its computational efficiency when used with graphical models such as Conditional Random Fields (CRF).

We use the Simple Linear Iterative Clustering (SLIC) [1] algorithm for oversegmentation. SLIC generates superpixels at a lower computational cost while achieving a segmentation quality of about the same or better than some state-of-the-art graph-based and gradient-based methods in 2012. The algorithm clusters pixels in the combined five-dimensional CIELab color and image plane space using a special case of *k-means* to efficiently generate compact, nearly uniform superpixels. It scales up linearly in computational cost and memory usage as it runs in $O(n)$ time complexity.

We segment each horse image to around 500 superpixels and project the generated boundaries to the corresponding label image. Figure 2 shows the resulting superpixels created by SLIC. A region adjacency graph is then generated for each instance. We consider each superpixel as a vertex, and adjacent superpixels (vertices) are joined by an edge. We label each vertex as a horse or non-horse vertex based on the average intensity values of all the pixels in the superpixel from the label image. If its Euclidean distance is closer to a white pixel, then we tag it as a horse vertex, otherwise, we tag it as non-horse vertex. Although there's a loss of pixelwise accuracy due to the grouping of pixels, the resulting pixelwise accuracy of the superpixeled segmentation from the ground truth is almost always greater than 95%.

C. Feature Extraction

For each vertex and edge of the region adjacency graph, we attempted several combinations of the following features. The patches centered at the centroids of the vertices are extracted and features are derived from this patch except for the superpixel color. We've also tried patch sizes of *32x32* and *96x96* for the computation of histogram differences.

1) Unary Features:

a) Superpixel Color: This is the average of the intensity values of all the pixels in the superpixel. Aside from using the standard RGB color space, we also used the CIELab color space based on the assumption that it may provide illumination invariance from its lightness of color component.

b) Histogram: The histogram of intensity values of the extracted patch. Each channel is represented by 12 bins and normalized using the L2 norm.

c) CNN Features: Features produced by a Network-In-Network model [14] trained with the CIFAR-10 [9] dataset, which contains horses as one of the object classes, were tested. Similar to Liu et al.'s work [15], no fine tuning was performed to see how object recognition in deep network learning transfers to image segmentation. The extracted patch is fed to the network and the outputs of the last layer are obtained as features.

d) Histogram of Oriented Gradients (HOG): Computed from the grayscale values of the extracted patch. The patch is divided into blocks, composed of *3x3* cells made up of *8x8* pixels each, which represent larger regions in the image. For each cell, a local 1-D histogram of gradient or edge orientations over all the pixels in the cell are accumulated. The combined cell level histogram is the orientation histogram representation of the cell. The gradient of the cells are partitioned into 9 orientation bins, which are then normalized across blocks. An energy measure is accumulated over the cells in the block. This value is then used to normalize each cell in the block. At this point, we now have the histogram of oriented gradient (HOG) descriptors. The collection of HOG descriptors from all the blocks of a dense overlapping grid of blocks covering the window are accumulated and flattened into a feature vector.

2) Pairwise Features:

a) Color Difference: For RGB color space, the Euclidean distance is computed between the average RGB values of the adjacent superpixel pairs. For CIELab color space, the color difference is defined by CIEDE2000 [5].

b) Histogram Difference: The Bhattacharyya distance is used to measure the difference of the histograms of two connected vertices.

c) Texture Similarity: The histogram of the Local Binary Pattern (LBP) feature is a good measure to classify shape and texture [2]. The Kullback-Leibler-Divergence is used to provide a score for similarity of the LBP histogram distributions of the two nodes.

D. Conditional Random Fields

The segmentation problem is formulated as a labelling problem of each superpixel as part of the horse or part of the background. Following the seminal work of Lafferty [12], the conditional probability of the labels given the set of superpixels representing the image is formulated as a conditional random field (CRF) given by:

$$P(Y|X) = \frac{e^{-E(Y,X)}}{Z}$$

Where Y is the set of binary labels corresponding to the set of superpixels X. E is the energy function factorized by various feature functions and Z is the normalizing function.

The labelling problem of the superpixel as a horse or a non-horse is optimized by a Maximum a Posteriori (MAP) approach where the best set of labels Y^* are obatained by minimizing the energy function E.

$$Y^* = \arg\min_{Y} E(Y,X)$$

The energy function is made up of unary ψ_u and pairwise potentials ψ_p.

$$E(Y,X) = \sum_{i}^{n} \psi_u(y_i, X) + \sum_{(i,j)\in C} \psi_p(y_i, y_j, X)$$

where $y_i \in Y$ is a node in the conditional random field representing the label of a superpixel where $n = |Y|$. C is the set of generated edges which connect two superpixels i and j.

To learn the parameters of the CRF model, we've used the structured prediction learning framework *pystruct* [16]. It uses SSVMs to learn the coefficients of CRF potentials. During inference, we make a prediction by finding the maximizing structured label y of the following linear function:

$$f(x) = \arg\max_{y\in Y} \theta^T \psi(x,y)$$

where ψ is the joint feature function of x and y, and θ are the parameters of the model. *Pystruct* assists with encoding the structure of our problem in a joint feature function ϕ and inferring the parameters θ from training data. We've chosen the Block-Coordinate Frank-Wolfe Optimization inference algorithm for SSVM [11] to learn the parameters because of its ability in obtaining reasonable solutions with fast inference [16].

IV. Experiment Results

To quantify the results of how the CRF model preforms, we use accuracy, precision, and recall, which are given by the following:

$$Accuracy = \frac{TN + TP}{TN + FN + TN + FP}$$

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

where TP is the number of horse superpixels correctly identified, TN is the number of superpixels correctly identified as background, FP is the number of superpixels incorrectly identified as a horse and FN is the number of superpixels incorrectly identified as background.

Table I shows the summary of the performance of the combination of features we've used in training the CRF. All models have been trained with 1000 iterations.

We first started with just the average color of all the pixels within a superpixel both in RGB and in CIELab color spaces. Tests show that the model has a tendency to predict superpixels as horse when the the image is in RGB than in CIELab. Even though CIELab better represents lightness information, it didn't surpass the accuracy of the model using RGB. We also tried to see, in CIELab space, if considering the position of the superpixels by adding the relative position from image center as a unary feature and the relative positions of their centroids as a pairwise feature will affect the performance of the model. Surprisingly, there was no significant increase in performance, hence we chose to retain the image in RGB color space without position information. This yielded an accuracy of just above 80%, but showed poor precision and recall which are both below 70%.

Extraction of patches of size 32 by 32 centered at the centroids of the superpixels are then done to derive intensity histogram and pairwise histogram difference. This has increased all metrics by a good margin. Adding either features of the CNN trained with CIFAR-10 dataset or HOG features as a node feature further improved performance. We've decided to use HOG over the CNN features as it performed far better, with both precision and recall reaching above 80%. Unfortunately, we were unable to find the preprocessing code used by the

TABLE I. SUPERPIXELWISE PERFORMANCE OF DIFFERENT FEATURES ATTEMPTED FOR CRF LEARNING

Unary Feature(s)	Pairwise Feature(s)	Patch Size	Training Set			Test Set		
			Accuracy	Precision	Recall	Accuracy	Precision	Recall
RGB	RGB Euclidean Distance	-	81.68%	64.08%	69.24%	80.74%	59.76%	67.75%
CIELab	Delta CIE2000	-	80.56%	67.76%	66.03%	79.07%	63.12%	64.79%
CIELab Relative Position from Image Center	Delta CIE2000 Relative Position of Centroids	-	80.78%	67.74%	66.44%	79.21%	62.76%	64.55%
RGB Histogram	RGB Euclidean Distance Histogram Difference	32	83.44%	70.67%	70.79%	82.06%	67.34%	69.02%
RGB Histogram CNN Features	RGB Euclidean Distance Histogram Difference	32	87.29%	77.24%	78.27%	84.85%	71.65%	73.94%
RGB Histogram Histogram of Oriented Gradients (HOG)	RGB Euclidean Distance Histogram Difference	32	90.25%	82.87%	82.60%	89.22%	80.50%	81.26%
RGB Histogram Histogram of Oriented Gradients (HOG)	RGB Euclidean Distance Histogram Difference	96	92.29%	85.03%	85.57%	90.99%	82.01%	83.53%
RGB Histogram Histogram of Oriented Gradients (HOG)	RGB Euclidean Distance Histogram Difference Texture Similarity	96	95.96%	91.73%	92.37%	92.45%	84.84%	85.69%

TABLE II. AVERAGE PIXELWISE PERFORMANCE

	Accuracy	Precision	Recall	F1 Score
Training Set	94.55%	89.55%	89.03%	88.87%
Test Set	91.61%	84.12%	83.21%	82.72%

pretrained CNN model. Furthermore, we did not perform any fine tuning. These might have contributed to the CNN features' lower performance.

By optimizing the model, we found that using a larger patch size of *96x96* and adding texture similarity score by utilizing LBP histograms as pairwise features greatly improved the performance. For the training set, all metrics exceeded 90%, while test set accuracy is around 92% with both precision and recall at around 85%. This is the best performing model we've found and the one we've used to measure the actual segmentation performance. Due to dense computation of the histograms in a larger patch size, the downside of using this model is that the whole segmentation pipeline becomes slow, taking 10-20 seconds per image. This makes this CRF model unsuitable for real-time image segmentation.

We then tag each pixel with the label of its superpixel and compare it pixelwise with the ground truth image. This measures the overall segmentation quality of our method since representing the image with superpixels introduces errors pixelwise from the ground truth even when the CRF model correctly classifies all superpixels in the image. Aside from accuracy, precision and recall, we've also measured pixelwise performance with the F1 score, given by:

$$F1 = 2\ \frac{Precision \cdot Recall}{Precision + Recall}$$

Table II shows the segmentation quality of our method. Overall, we've achieved a good average F1 score for all the images in both the training and test sets, reaching around 88.72% and and 82.72% respectively. We also note that we've achieved visually decent segmentation. Figure 3 shows the resulting segmentation of our method on some of the test samples which achieved an F1 score higher than 90%.

TABLE III. COMPARISON WITH STATE-OF-THE-ART METHODS

Method	S_a	S_o
Training Set	94.55%	80.37%
Test Set	91.61%	71.26%
Levin and Weiss [13]	95.50%	-
Cosegmentation [8]	80.10%	-
Bertelli et al. [3]	94.60%	80.10%
Kuettel et al. [10]	94.70%	-
Liu et al. [15]	95.70%	84.40%

To compare with state-of-the-art methods for this dataset, we quantify by the pixelwise accuracy S_a and the foreground intersection over union score S_o. Table III shows our method is still far compared to contemporary state-of-the-art methods especially in terms of S_o score. Nonetheless we consider our work as a good starting point in understanding the capability of Conditional Random Fields (CRF) with Image Segmentation.

V. CONCLUSION

We have presented a model that segmented images and classified each segment into a horse or non-horse segment. By utilizing oversegmentation into superpixels, and identifying both unary and pairwise features, were able to create a CRF model that provides good results in terms of metrics such as accuracy, precision, recall, and the F1 score. Although we were able to achieve an acceptable level of performance, it should be emphasized that utilizing some features came at the cost computational performance which makes it unsuitable for use with real-time systems. Advances in the computations used to obtain these features may prove beneficial to this study.

As for future work, we are interested in checking the performance of the model with several other datasets. This would allow us to further obtain the relative performance of the model compared to several baselines. Furthermore, we would also like to explore fine-tuning pre-trained deep convolutional

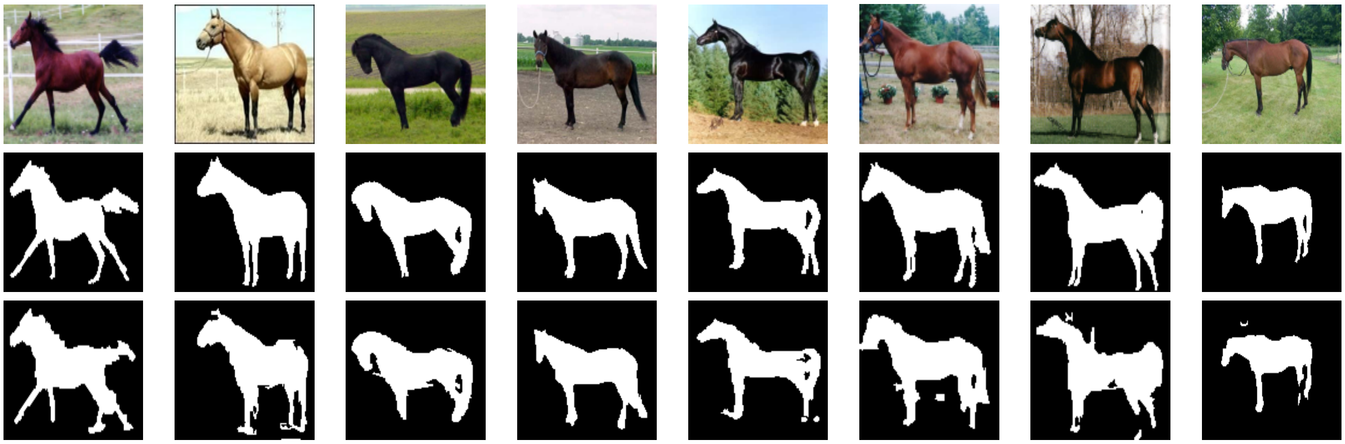

Fig. 3. Example segmentation results from test set. First row contains the original image. Second row contains the ground truth. Third row contains the segmentation produced by our method.

neural networks transferred to image segmentation as we believe they will yield higher performance.

Acknowledgment

We would like to thank Dr. Prospero Naval for allowing us to work on this topic. This is done as a Mini Project Requirement for CS 284 Machine Learning course.

References

[1] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Susstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(11):2274–2282, November 2012.

[2] Timo Ahonen, Abdenour Hadid, and Matti Pietikinen. Face recognition with local binary patterns. In *In Proc. of 9th Euro15 We*, pages 469–481, 2006.

[3] L. Bertelli, Tianli Yu, D. Vu, and B. Gokturk. Kernelized structural svm learning for supervised object segmentation. In *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '11, pages 2153–2160, Washington, DC, USA, 2011. IEEE Computer Society.

[4] Eran Borenstein, Eitan Sharon, and Shimon Ullman. Combining top-down and bottom-up segmentation. In *Proceedings of the 2004 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'04) Volume 4 - Volume 04*, CVPRW '04, pages 46–, Washington, DC, USA, 2004. IEEE Computer Society.

[5] Edul N. Dalal Gaurav Sharma, Wencheng Wu. The ciede2000 color-difference formula: Implementation notes, supplementary test data, and mathematical observations. Technical report, University of Rochester, 2004.

[6] Roberto Cipolla Jamie Shotton, Andrew Blake. Contour-based learning for object detection. In *International Conference on Computer Vision*, January 2005.

[7] Roberto Cipolla Jamie Shotton, Matthew Johnson. Semantic texton forests for image categorization and segmentation. In *Proc. IEEE CVPR*, June 2008.

[8] A. Joulin, F. Bach, and J. Ponce. Discriminative clustering for image co-segmentation. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[9] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

[10] D. Kuettel and V. Ferrari. Figure-ground segmentation by transferring window masks. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 558–565, June 2012.

[11] Simon Lacoste-Julien, Martin Jaggi, Mark Schmidt, and Patrick Pletscher. Block-coordinate Frank-Wolfe optimization for structural SVMs. In *ICML*, 2013.

[12] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[13] Anat Levin and Yair Weiss. Learning to combine bottom-up and top-down segmentation. *International Journal of Computer Vision*, 81(1):105–118, 2009.

[14] Min Lin, Qiang Chen, and Shuicheng Yan. Network in network. *CoRR*, abs/1312.4400, 2013.

[15] Fayao Liu, Guosheng Lin, and Chunhua Shen. CRF learning with CNN features for image segmentation. *CoRR*, abs/1503.08263, 2015.

[16] Andreas C. Müller and Sven Behnke. pystruct - learning structured prediction in python. *Journal of Machine Learning Research*, 15:2055–2060, 2014.