

Semantic Segmentation Refinement by Monte Carlo Region Growing of High Confidence Detections^{*}

Philippe A. Dias¹[0000-0001-9427-7112] and Henry Medeiros¹[0000-0002-7704-5587]

Marquette University, Milwaukee WI 53233, USA
{philipe.ambroziodias,henry.medeiros}@marquette.edu

Abstract. The semantic segmentation produced by most state-of-the-art methods does not show satisfactory adherence to object boundaries. Methods such as fully-connected conditional random fields (CRFs) can significantly refine segmentation predictions. However, they rely on supervised parameter optimization that depends upon specific datasets and predictor modules. We propose an unsupervised method for semantic segmentation refinement that takes as input the confidence scores generated by a segmentation network and re-labels pixels with low confidence levels. More specifically, a region growing mechanism aggregates these pixels to neighboring areas with high confidence scores and similar appearance. To minimize the impact of high-confidence prediction errors, our algorithm performs multiple growing steps by Monte Carlo sampling initial seeds in high-confidence regions. Our method provides both running time and segmentation improvements comparable to state-of-the-art refinement approaches for semantic segmentation, as demonstrated by evaluations on multiple publicly available benchmark datasets.

Keywords: Segmentation refinement · instance semantic segmentation · unsupervised post-processing.

1 Introduction

The identification of the objects present in an image is a primary goal of computer vision. Such a determination can be carried out at different levels of granularity, which correspond to four well-known subproblems: image classification, object detection, semantic segmentation, and instance segmentation. These subproblems aim to achieve increasingly complex goals and have therefore been addressed with different levels of success.

In recent years, the combination of deep Convolutional Neural Networks (CNN) and increasingly larger publicly available datasets has led to substantial improvements to the state of the art in image classification [1, 2]. For segmentation tasks, however, the performance of conventional CNN architectures is

^{*} We acknowledge the support of USDA ARS agreement #584080-5-020, and of NVIDIA Corporation with the donation of the GPU used for this research.

limited by the typical downsampling (pooling) intrinsic to such networks. Downsampling is employed to learn hierarchical features, but it ultimately leads to imprecise, coarse segmentation in scenarios that require pixel-wise predictions. Strategies including *atrous* convolutions [3] and upsampling [4–6] have been proposed to address this limitation, but the segmentation they produce still tend not to be finely aligned with the boundaries of the objects. Post-processing approaches such as conditional random fields (CRFs) [3, 7] have been successful in segmentation refinement, but their performance depends on proper optimization of parameters for each specific dataset and predictor module being used.

In this paper, we propose the *Region Growing Refinement (RGR)* algorithm, an unsupervised and easily generalizable post-processing module that performs appearance-based region growing to refine the predictions generated by a CNN for semantic segmentation. Based on the classification scores available from the detector, our method first divides the image into three regions: high confidence background, high confidence object, and uncertainty region. The pixels within the uncertainty region, which are the ones that tend to be misclassified by CNN-based methods, are then labeled by means of region growing. We apply Monte Carlo sampling to select initial seeds from the high confidence regions, which are then grown according to a distance metric computed in the 5-D space of spatial and color coordinates. Our model has the advantage of not requiring any dataset-specific parameter optimization, working in a fully unsupervised manner.

We report experiments using different CNNs, datasets and baselines. We first employ the Fully Convolutional Instance-aware Semantic Segmentation (FCIS) algorithm [8] as a predictor module as well as the baseline for our performance evaluation. Since the ground truth annotations composing the MS COCO dataset contain non-negligible inaccuracies in terms of boundary adherence, we also assess RGR efficacy on the PASCAL VOC 2012 [9] validation set and on selected sequences from the DAVIS dataset [10]. Compared to the first two datasets, annotations provided for the DAVIS dataset are more fine-grained, with tighter boundary adherence. As a result, in addition to relatively small increases in segmentation accuracy for the MS COCO (+1.5% in AP) and the PASCAL datasets (+0.5% in $mAP\%$), we report significantly better results for the DAVIS sequences (+3.2% in $\mathcal{J}(IoU)\%$), which more realistically reflect the segmentation improvement observed through qualitative (visual) inspection.

We also compare the RGR algorithm against the state-of-the-art but supervised methods dense CRF [7] and DT-EdgeNet [11], for refinement of Deeplab [3] predictions. RGR provides both running time and segmentation refinement performance comparable to the optimized versions of both supervised methods, but without requiring neither dataset- nor model-specific fine-tuning.

2 Related Work

The relatively recent development of CNNs trained on large publicly available datasets represented an inflection point in image understanding research. Compared to the period when models based on hand-engineered features (e.g. HOG

[12], SIFT [13]) were the norm, currently the state of the art in object recognition tasks is being improved at a dramatically faster pace [2].

However, traditional CNN-based methods that are effective for image classification and object detection present limitations for segmentation tasks. The combination of max-pooling and *striding* operations constitute a standard strategy for learning hierarchical features, which are a determining factor in the success of deep learning models. They explore the transition invariance favored by image-level labeling tasks, i.e., the fact that only the presence of an object matters, not its location. Such premise, however, does not hold for pixel-dense classification tasks, which require precise object localization. As pointed out by Chen et al. in [3], such spatial insensitivity and image downsampling inherent to conventional CNNs are two major hindering factors for their performance in image segmentation. An especially noticeable effect is that these methods generate coarse segmentation masks with limited boundary adherence.

One particularly successful strategy for segmentation tasks is the concept of fully convolutional networks (FCNs) [4]. In FCNs, the traditionally fully connected layers at the end of conventional CNNs are replaced by convolutional layers, which upsample the feature maps to generate a pixel-dense prediction. Different upsampling or *decoding* approaches have been proposed, including the use of deconvolution layers [4, 14], and encoder-decoder architectures with skip-layer connections [5, 6, 15, 16]. Other methods focus on reducing the downsampling rate. The Deeplab architecture [3] is one such method in which the concept of dilated (or *atrous*) convolutions is successfully employed to reduce the downsampling rate from $1/32$ to $1/8$.

A variation of the semantic segmentation problem is instance segmentation, which requires detecting and segmenting individual object instances. Coarse segmentations significantly hamper this type of task, since neighboring instances of objects of the same class are frequently merged into a single segmentation. Dai et al. in [17] introduced the concept of instance-sensitive FCNs, in which a FCN is designed to compute score maps that determine the likelihood that a pixel belongs to a relative position (e.g. right-upper corner) of an object instance. FCIS [8] is an extension of that approach, which achieved the state-of-the-art performance and won the COCO 2016 segmentation competition.

In addition to adjustments in CNN architectures, multiple works have investigated the use of low-level information strategies to construct models for object detection and semantic segmentation. Girschick et al. in [18] introduced the model known as RCNN that performs object detection by evaluating multiple region proposals. In RCNN, the region proposals are generated using Selective Search [19], which consists of merging a set of small regions [20] based on different similarity measures, an approach closely related to the concept of superpixels.

Superpixels are perceptually meaningful clusters of pixels, which are grouped based on color similarity and other low-level properties. Stutz et al. [21] provide a review of state-of-the-art superpixel approaches. One of the most widely-used methods is the Simple Linear Iterative Clustering (SLIC) algorithm [22], a *k-means*-based method that groups pixels into clusters according to their proximity

in both spatial and color spaces. Semantic segmentation models using superpixels traditionally employ them as a pre-processing step. The image is first divided into superpixels, followed by a prediction step in which each element is individually evaluated using engineered hierarchical features [23] or CNNs [24, 25].

In addition to the generation of region proposals or pre-segmentation elements, local-appearance information has been also employed in post-processing steps to improve segmentations obtained with deep CNN models. One possible approach consists of adapting the GrabCut model [26], which is a well-known method for interactive segmentation based on the minimization of an energy function that models the background and foreground as Gaussian mixture models (GMM). Another such post-processing approach is to model the problem as a CRF defined over neighboring pixels or small patches. Krähenbühl et al. in [7] introduced an efficient algorithm for fully connected CRFs containing pairwise potentials that associate all pairs of pixels in an image. This algorithm is successfully exploited by the Deeplab model [3] to produce fine-grained segmentations.

Chen et al. introduced the DT-EdgeNet in [11], which is a computationally cheaper alternative that replaces the dense CRF with a domain-transform approach that refines the segmentation using a modern edge-preserving filtering method. Both approaches, however, have to be optimized in a supervised manner when applied to different datasets or combined with different predictors.

3 Proposed Approach

The method we propose for refinement of segmentation boundaries is a generic **unsupervised** post-processing module that can be coupled to the output of any CNN or similar model for semantic segmentation. Our RGR algorithm consists of four main steps: 1) identification of low and high confidence classification regions; 2) Monte Carlo sampling of initial seeds; 3) region growing; and 4) majority voting and final classification. The operations that comprise these steps are described in detail below. In our description, we make reference to Figure 1 and Algorithm 1, which list the operations performed by our method for each proposed detection in an image. If detections for multiple classes are present, the algorithm is executed on the score maps associated with each class, and the final classification is defined by computing the maximum likelihood across classes.

1) Step 1 - Thresholding the image into three regions: Conventional models typically infer a final classification by pixel-wise thresholding the scores obtained for each class. Instead of relying on intermediate threshold values, our refinement module directly exploits the available classification scores to differentiate between three regions: a) high confidence foreground (or object) R_F ; b) high confidence background R_B ; and c) uncertainty region R_U . As defined in Eq. 1, these regions are identified using two high confidence thresholds τ_F and τ_B

$$\begin{aligned} R_F &= \{p_j | M(p_j) \geq \tau_F\}, \\ R_U &= \{p_j | \tau_B < M(p_j) < \tau_F\}, \\ R_B &= \{p_j | M(p_j) \leq \tau_B\}, \end{aligned} \tag{1}$$

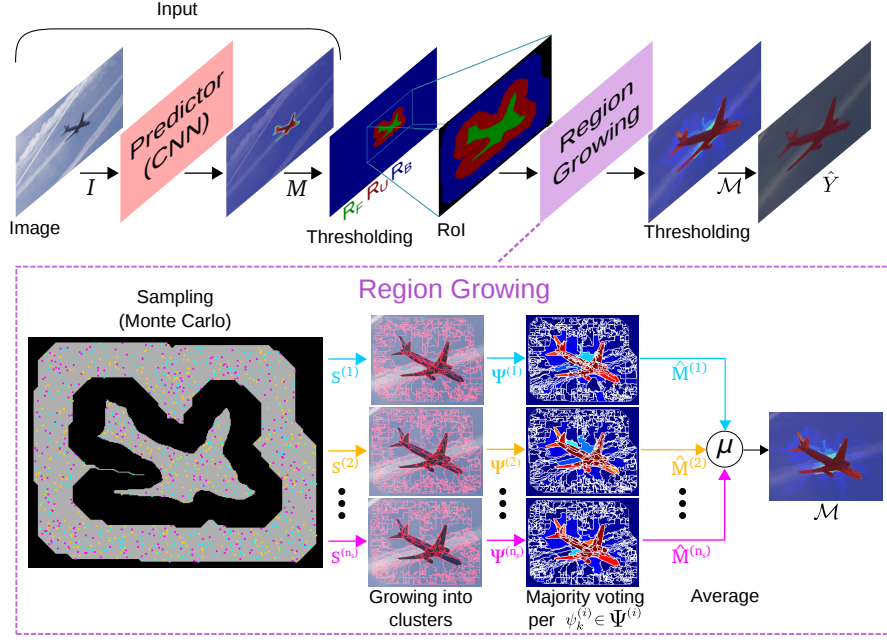


Fig. 1. Best viewed in color. Diagram illustrating the sequence of tasks performed by the proposed RGR model for segmentation refinement. Each task and its corresponding output (shown below the arrows) are described in Algorithm 1.

where p_j is the j -th pixel in the input image I , and $M(p_j)$ is its corresponding score in the detection confidence map M computed by the original predictor (usually a CNN). High confidence regions correspond to areas where pixels present scores near the extremes of the likelihood range. For normalized score maps, values lower than a threshold $\tau_B \sim 0.0$ identify pixels in the high confidence background, while values larger than $\tau_F \sim 1.0$ indicate high confidence foreground elements. To recover possible false negatives, morphological shrinking is performed on the background R_B boundaries. The fourth leftmost image in Figure 1 illustrates the division of the image into the three regions R_F , R_U , and R_B .

2) *Step 2 - Monte Carlo sampling of initial seeds*: Inspired by the notion of pixel affinity employed by many algorithms for superpixel segmentation, our method applies a region growing approach to classify pixels within the uncertainty zone. More specifically, we build on the simple non-iterative clustering (SNIC) algorithm [27].

SNIC selects seeds on a regular grid over the whole image. In contrast, our algorithm selects seeds by Monte Carlo sampling only high confidence regions. Such an adaptation provides three main advantages for segmentation refinement. First, our random selection of seeds allows clusters of flexible size. This is beneficial for: i) growing into larger regions that were missed by the predictor, and

Algorithm 1 RGR refinement algorithm**Input:** Image I , confidence map M **Output:** Refined semantic segmentation \hat{Y} of image I

- 1: Threshold M into three regions: background R_B , foreground R_F and uncertain zone R_U
- 2: Define a Region of Interest (RoI) according to Eq. 2
- 3: **for** $i = 1$ to n_s **do**
- 4: Form a set $S^{(i)}$ of initial seeds by uniformly sampling from $R_B \cup R_F$
- 5: Generate a set of clusters $\Psi^{(i)}$ by performing region growing using the SNIC algorithm with $S^{(i)}$ as input
- 6: For each generated cluster $\psi_k^{(i)} \in \Psi^{(i)}$, compute confidence map $\hat{M}^{(i)}$ according to Eq. 4
- 7: **end for**
- 8: Compute the pixel-wise average \mathcal{M} of $\hat{M}^{(i)}$, $i = 1, \dots, n$
- 9: Generate \hat{Y} by pixel-wise thresholding \mathcal{M}

ii) forming smaller clusters, rather than *leaking* into nearby pixels. Second, it enforces the classification of unlabeled pixels to derive from high confidence information. And third, at each Monte Carlo iteration, clusters are grown from different sets of randomly selected seeds. Combined with majority voting per cluster and pixel-wise averaging across iterations (detailed in Step 3), this procedure acts as a filter against false positives detected with high confidence by the predictor.

Our Monte Carlo approach consists of n_s iterations. At each iteration i , a set $S^{(i)}$ of initial seeds is defined by uniformly sampling the high-confidence area $R_H = R_B \cup R_F$. Let p_h represent pixels within the region R_H , where the index $h = 1, \dots, |R_H|$. Uniform sampling of seeds can thus be performed index-wise according to $h \sim U(1, |R_H|)$. We determine the number of seeds to be sampled $|S^{(i)}|$ based on the sampling domain area (i.e. $|R_H|$) and the desired average spacing between samples σ , i.e., $|S^{(i)}| = |R_H|/\sigma$. The spacing between seeds ensures the availability of paths through which all the initial centroids can propagate throughout the uncertainty region.

3) *Step 3 - Region Growing*: The dashed block at the bottom of Figure 1 illustrates the sequence of operations performed by RGR for region growing. To reduce the computation time, we restrict the region growing to a Region of Interest (RoI) around the uncertain zone R_U . Based on the spatial distance to R_U , the background R_B can be split into two regions: far background R_{fB} and near background R_{nB} . Since the far background is unlikely to influence the clustering of the uncertain region, R_{fB} can be ignored during region growing. Hence, we define the RoI for region growing as

$$RoI = R_{nB} \cup R_F \cup R_U. \quad (2)$$

Initial centroids are then grown according to an adaptation of the SNIC algorithm. As in SNIC, we measure the similarity between a pixel and a centroid as their distance in a five-dimensional space of color and spatial coordinates. Let

the spatial position of a pixel be represented by a vector $\mathbf{x} = [x \ y]^T$, while its color is expressed in the CIELAB color-space by $\mathbf{c} = [l \ a \ b]^T$. The distance $d_{j,k}$ between a pixel j and a centroid k is given by Eq. 3, where θ_s and θ_m are normalizing factors for spatial and color distances, respectively

$$d_{j,k} = \sqrt{\frac{\|\mathbf{x}_j - \mathbf{x}_k\|_2^2}{\theta_s} + \frac{\|\mathbf{c}_j - \mathbf{c}_k\|_2^2}{\theta_m}}. \quad (3)$$

Also as in SNIC, our region growing implementation relies on a priority queue, which is constantly populated with nodes that correspond to unlabeled pixels 4- or 8-connected to a region being grown. This queue is sorted according to the similarity between the candidate pixel and the average (centroid) of the growing region, given by Eq. 3. While the queue is not empty, each iteration consists of: 1) popping the first element of the queue, which corresponds to the unlabeled pixel that is most similar to a neighboring centroid k ; 2) annexing this pixel to the respective cluster $\psi_k^{(i)}$; 3) updating the region centroid; and 4) populating the queue with neighbors of this pixel that are still unlabeled.

We add a constraint to the original SNIC algorithm in order to reduce the incidence of false detections. A node is only pushed into the queue if its distance to the corresponding centroid is smaller than a certain value d_{Max} . This strategy ensures that an unlabeled pixel within R_U will only inherit information from high confidence pixels that are sufficiently similar to it. This creates the possibility of “orphan” elements, i.e., pixels for which no node is ever created. Such pixels are therefore classified as background. For each set of initial seeds $S^{(i)}$, the region growing process generates a cluster map $\Psi^{(i)}$ that associates each pixel to a respective cluster $\psi_k^{(i)}$.

4) *Step 4 - Majority voting and final classification*: Following the region growing process, RGR conducts a majority voting procedure to ultimately classify each generated cluster into foreground or background. As expressed in Eq. 4, a pixel p_j contributes a positive vote for foreground classification if its original prediction score $M(p_j)$ is larger than a threshold τ_0 . We compute the ratio of positive votes across all pixels $p_j \in \psi_k^{(i)}$ to generate a refined likelihood map $\hat{M}^{(i)}$ for each set of clusters according to

$$Y_k^{(i)} = \left\{ p_j \in \psi_k^{(i)} \mid M(p_j) > \tau_0 \right\}, \quad (4)$$

$$\hat{M}^{(i)}(p_j) = \frac{|Y_k^{(i)}|}{|\psi_k^{(i)}|}. \quad (5)$$

The likelihood maps $\hat{M}^{(i)}$ obtained from the n_s Monte Carlo samplings are averaged to generate a final pixel dense score map $\mathcal{M} = \frac{1}{n_s} \sum_{i=1}^{n_s} \hat{M}^{(i)}$. Finally, each pixel is classified into foreground if more than 50% of its average votes are positive. Otherwise, the region is labeled as background, that is,

$$\hat{Y}(p_j) = \mathbb{1}_{\mathcal{M}(p_j) > 0.5}, \quad (6)$$

where $\hat{Y}(p_j)$ is the final binary classification map and $\mathbb{1}$ is an indicator variable that assumes the value 1 when the corresponding condition is satisfied.

4 Experiments

To the best of our knowledge, the COCO dataset is the largest publicly available dataset for semantic/instance segmentation, with a total of 2.5 million labeled instances in 328,000 images [6]. However, as discussed in Section 4.1, COCO’s ground truth annotations contain imprecisions intrinsic from its labeling procedure. To minimize the influence of dataset-specific errors, we assess the performance of our algorithm on: i) the COCO 2016 validation set; ii) the PASCAL VOC 2012 dataset; and iii) selected video sequences of the DAVIS dataset.

Since RGR is an unsupervised post-processing refinement module, it can be coupled to any semantic segmentation network. In Section 4.1, we evaluate its performance in comparison to SNIC superpixels for refinement of FCIS predictions. In Section 4.2, RGR, CRF, DT-EdgeNet and GrabCut are compared for refinement of Deeplab predictions.

4.1 Comparison Against Superpixel Refinement

We denote the combination of the publicly available FCIS model and the refinement module RGR as FCIS+RGR. Since RGR works in a region growing manner that is inspired by the concept of superpixel segmentation, our performance analysis also includes FCIS+SNIC, a naive refinement method that consists of performing majority voting within superpixels generated with SNIC.

Following a grid-search executed on the PASCAL VOC dataset, for all our experiments FCIS+SNIC employs SNIC with $\theta_m = 0.1$ and superpixel size of $100px$. RGR thresholds were defined as follows. First, τ_0 corresponds to the original detector optimal threshold. For FCIS this value is 0.4, as reported in [8]. To identify the high confidence foreground, we empirically selected a high confidence threshold τ_F corresponding to $1.5 \times \tau_0$, hence 0.6. As for the background detection, we set the high confidence lower threshold to $\tau_B = 0.0$.

Table 1. Comparison between results obtained by FCIS, FCIS+SNIC and FCIS+RGR on the COCO 2016 (val), the PASCAL VOC 2012 (val), and the DAVIS datasets.

	COCO 2016						VOC 2012	DAVIS	
	AP (%)	AP_{50} (%)	AP_{75} (%)	AP_S (%)	AP_M (%)	AP_L (%)	mAP (%)	$\mathcal{J}(IoU)$ (%)	\mathcal{F} (%)
FCIS	35.1	60.1	36.5	9.8	38.4	59.8	70.6	71.2	69.9
FCIS + SNIC	34.9	59.0	36.4	9.3	38.2	59.6	70.6	72.8	70.4
FCIS + RGR	36.9	60.6	39.3	11.4	40.7	60.5	71.1	74.4	72.7

COCO 2016 Segmentation Dataset. Based on the COCO official guidelines, we evaluate the performance obtained by FCIS, FCIS+SNIC, and FCIS+RGR on the validation set composed of 40k images. Table 1 summarizes the performance of the three methods according to the standard COCO metrics, including average precision (AP) averaged over 0.5 : 0.95 intersection over union (IoU) thresholds and at different scales. While increasing the number of true positive detections (+0.3% in AR) for all the scenarios, the naive FCIS+SNIC approach also decreases the AP by introducing a larger number of false positives.

Our refinement model, RGR, on the other hand, increases the baseline FCIS overall performance by 1.8%. Compared to the improvement of 0.5% in AP_{50} , the increase of 2.8% in AP_{75} demonstrates that RGR is particularly successful for cases where the detections obtained from the input CNN are accurately located.

Figure 2 presents the average precision obtained for each of the 80 COCO categories. Since its region growing is based on local affinity characteristics, it is natural that the RGR refinement is especially effective for objects with more uniform appearance (e.g. airplane, frisbee). Nevertheless, these results also demonstrate the robustness of the refinement provided by RGR, since no object category shows a noticeable decrease in average precision.

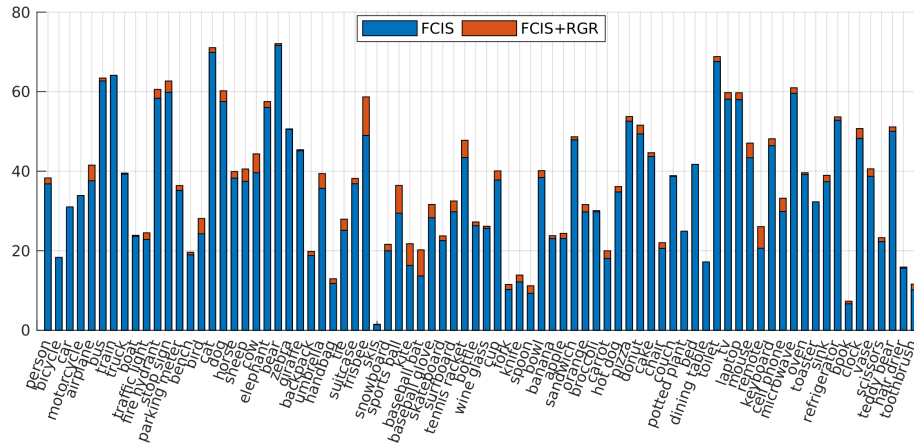


Fig. 2. AP obtained by FCIS and FCIS+RGR for each of the 80 COCO categories.

A closer visual inspection of the output labels also shows that the metrics obtained might be misleading. Despite the extensive efforts described in [8] to create such a massive dataset, for some instances, the ground truth segmentation annotations lack accurate adherence to real object boundaries. As a consequence, improvements obtained through RGR refinement are not reflected in the final metrics for a significant number of images. Figure 3 provides examples of the imprecisions in ground truth annotations, collected from different object classes.



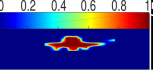


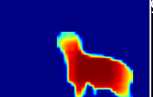


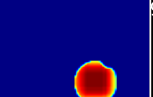


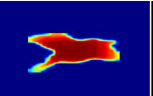
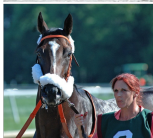

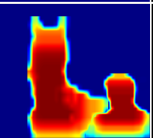


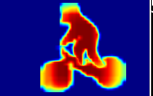


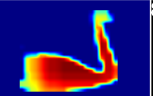
Image	GT	FCIS scores	FCIS	FCIS+SNIC	FCIS+RGR
			57.7%	46.8%	50.9%
			90.0%	89.7%	90.6%
			95.8%	94.8%	91.4%
			98.9%	96.9%	99.7%
			89.0%	89.3%	90.4%
			55.8%	58.6%	60.4%
			84.7%	86.5%	91.4%

Fig. 3. Examples of detections on the COCO (three top-most rows), PASCAL (fourth and fifth rows) and DAVIS (two bottom-most rows) datasets. From left to right: original image, ground truth, FCIS scores, FCIS detection, FCIS+SNIC, FCIS+RGR. The obtained AP (COCO), mAP (PASCAL) and IoU (DAVIS) are displayed above each corresponding detection. Ground truth annotations in the first and second rows are also examples of COCO annotations with poor boundary adherence.

While for the *airplane* example the segmentation obtained using RGR is clearly better in qualitative terms, its overlap (IoU) with the ground truth is almost 7.0% lower than the one associated with the FCIS output.

PASCAL VOC 2012 Segmentation Dataset. Table 1 also contains a summary of the results obtained by FCIS, FCIS+SNIC and our method FCIS+RGR on the PASCAL VOC 2012 validation set. The segmentation refinement generated using RGR provides a final mAP slightly better (+0.5%) than both FCIS and the refined version using naive superpixel-wise majority voting.

Since boundaries constitute a small fraction of the total image pixels, such a small difference in performance does not properly reflect the higher boundary adherence provided by RGR refinement. Thus, we follow the strategy presented

a contour accuracy metric \mathcal{F} , which corresponds to a F-measure computed based on the contour precision and recall. Segmentations refined using RGR yielded an increase of 2.8% in \mathcal{F} , confirming its ability to improve boundary adherence. Figure 4 presents the results obtained for each video sequence, with FCIS+RGR providing improvements in the range between 1.1% and 6.9% depending on the sequence. The fact that larger quantitative improvements are obtained for the DAVIS sequences corroborates our argument that the annotations available for the COCO and PASCAL datasets provide limited information regarding boundary accuracy/adherence of segmentation methods.

4.2 Comparison Against CRF, DT-EdgeNet and GrabCut

We also compare the performance of our method against the state-of-the-art dense CRF [7], which has been successfully exploited by Deeplab models for semantic segmentation refinement, as well as its alternative DT-EdgeNet [11] and the GrabCut model [26]. We adopt the publicly available model of Deeplab-LargeFOV as the base model for our comparison. This model is pre-trained on MS COCO and fine-tuned on the augmented *trainval* PASCAL VOC 2012 dataset.

We perform our evaluation for the refinement of the segmentations produced by the Deeplab-LargeFOV model on the PASCAL validation set. In this case, we configure RGR to use a different τ_F sampled from the distribution $U(0.5, 0.9)$ in each MC region growing iteration. The other thresholds remain fixed at $\tau_B = 0.0$ and $\tau_0 = 0.5$.

Quantitative results are summarized in Table 2, and qualitative examples are available in the supplementary materials. GrabCut demonstrated limited ability to refine the CNN predictions, which is expected since: i) it assumes that all pixels initialized as background/foreground are correct, and ii) the formulation using GMMs shows limited performance when the background/foreground appearance varies significantly. In contrast, by growing from a much higher number of seeds, RGR forms multiple clusters that significantly differ in terms of appearance but can share the same semantic labels.

The combination of an optimized CRF with Deeplab-LargeFOV culminates in a mAP of 80.1%, while the alternative using optimal DT-EdgeNet provides 78.9%. Our fully unsupervised RGR algorithm yields 79.2% mAP on this same scenario, being hence competitive with both optimized supervised models but with the advantage of not requiring any dataset-specific fine-tuning.

The higher generalization capability of our method is highlighted by transfer learning experiments. We compared RGR and CRF for the refinement of FCIS detections in the scenario described in Section 4.1, i.e., on the same selected sequences of the DAVIS dataset but without performing any dataset-specific fine-tuning for any method. While RGR again improves segmentation quality by 3.2%, in this scenario CRF provides only 2.7%. This demonstrates the advantage of our unsupervised approach, which can be employed on different datasets without the need for fine-tuning. Albeit providing significant improvements in

Table 2. Comparison between different refinement methods for different networks.

VOC 2012		DAVIS	
	<i>mAP (%)</i>		<i>IoU (%)</i>
Deeplab	76.1	FCIS	71.2
Deeplab+GrabCut	77.9 (+1.8)	FCIS+GrabCut	71.2 (+0.0)
Deeplab+DT	78.9 (+2.8)	FCIS+DT	NA*
Deeplab+CRF	80.1 (+3.9)	FCIS+CRF	73.9 (+2.7)
Deeplab+RGR	79.2 (+3.1)	FCIS+RGR	74.4 (+3.2)

*implementation public available only for the Deeplab model.

segmentation performance, the dense CRF is strongly dependent on supervised optimization of hyper-parameters for specific datasets and predictor models.

4.3 Inference Time

Finally, we evaluate the inference time of RGR in comparison to dense CRF. As explained above, our algorithm consists of four main steps: 1) thresholding, 2) Monte Carlo sampling of seeds, 3) region growing, and 4) majority voting. Steps 1, 2 and 4 are currently implemented in MATLAB (R2017a), while the region growing step is implemented in C++ based on the SNIC algorithm. Runtime assessment was performed on an Intel XeonTM CPU E5-2620 v3 @ 2.40GHz (62GB). The average runtime per image in the PASCAL VOC dataset is ~ 0.5 sec with 3 Monte Carlo iterations. This is lower than the 0.8sec average inference time of CRF's publicly available implementation, as reported in [11].

A breakdown of the runtime analysis shows that the region growing step requires only ~ 0.2 sec per Monte Carlo (MC) iteration, which is comparable to the 0.18sec required by the CPU-based implementation of the domain transform method described in [11]. We highlight that the multiple MC iterations are easily parallelizable. As summarized in Figure 5, using MATLAB's Parallel Pool the runtime obtained for 10 MC iterations is less than 0.1sec higher than the 0.5sec/image obtained for 3 MC iterations, which corresponds to a performance improvement of 0.1% in the scenario described in Section 4.1.

4.4 Failure Cases

As demonstrated by the experimental results, the quality of the refinement obtained using RGR depends on the accuracy of the detector model used as input, especially in terms of localization. Although Monte Carlo sampling improves the robustness against high confidence false positives, errors might be propagated if the score maps collected from the detector module (CNN) contain regions with high concentrations of false positives. An example of such case is found in Figure 3. Given the high confidence of the false positives generated by the FCIS model for internal parts of the bicycle wheels, RGR is unable to correct these mistakes and hence these regions remain incorrectly segmented.

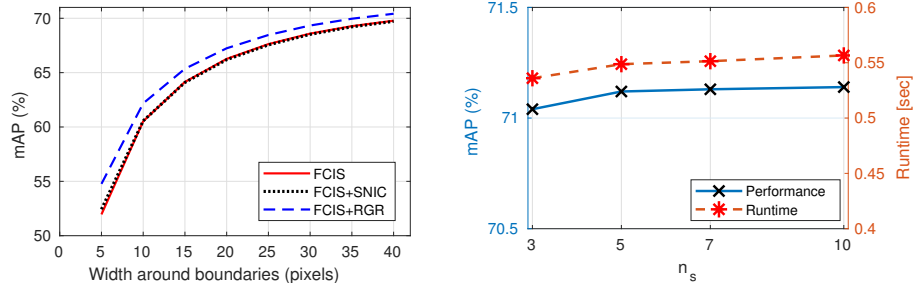


Fig. 5. *Left:* Mean average precision (mAP) on PASCAL on regions near object boundaries. *Right:* Runtime (orange) and performance (blue) of RGR according to the number of MC iterations (n_s) in the scenario described in Section 4.1.

5 Conclusion

We have presented RGR, an effective unsupervised post-processing algorithm for segmentation refinement. Traditionally, the final classification step of existing methods for semantic segmentation consists of thresholding score maps obtained from CNNs. Based on the concepts of Monte Carlo sampling and region growing, our algorithm achieves an adaptive thresholding by exploiting high confidence detections and low-level image information. Our results demonstrate the efficacy of RGR refinement, showing increased precision on three different segmentation datasets. Our algorithm provides segmentation improvements competitive with existing state-of-the-art refinement methods, but with the advantage of not requiring any dataset- or model-specific optimization of parameters.

Moreover, we highlight limitations of existing datasets in terms of boundary adherence of available ground truth annotations. This is an intrinsic limitation of annotations procedures that rely on approximating segmentations as polygons. In such cases, the quality of the segmentation is proportional to the number of vertices selected by the user, but hand selecting more points increases the labeling time per object. As future work, we consider exploiting RGR for improving annotations available for existing datasets or designing an alternative annotation tool. Finally, we hypothesize that the performance of existing CNNs for semantic segmentation could be improved by training with RGR as an additional step before computing losses. Such an arrangement could lead to a detector module that optimally interacts with RGR, identifying keypoints for the refinement process.

References

1. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet Classification with Deep Convolutional Neural Networks. *Advances In Neural Information Processing Systems* (2012) 1–9

2. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2016) 770–778
3. Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **40** (2018) 834–848
4. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Volume 07-12-June. (2015) 3431–3440
5. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer (2015) 234–241
6. Lin, G., Milan, A., Shen, C., Reid, I.: RefineNet: Multi-Path Refinement Networks for High-Resolution Semantic Segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2017)
7. Krähenbühl, P., Koltun, V.: Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials. In: Advances in neural information processing systems. (2011) 109–117
8. Li, Y., Qi, H., Dai, J., Ji, X., Wei, Y.: Fully Convolutional Instance-aware Semantic Segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2017) 2359–2367
9. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. (<http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html>)
10. Perazzi, F., Pont-Tuset, J., McWilliams, B., Gool, L.V., Gross, M., Sorkine-Hornung, A.: A Benchmark Dataset and Evaluation Methodology for Video Object Segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2016) 724–732
11. Chen, L.C., Barron, J.T., Papandreou, G., Murphy, K., Yuille, A.L.: Semantic image segmentation with task-specific edge detection using CNNs and a discriminatively trained domain transform. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2016) 4545–4554
12. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Volume 1., IEEE (2005) 886–893
13. Lowe, D.G.: Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision* (2004)
14. Noh, H., Hong, S., Han, B.: Learning deconvolution network for semantic segmentation. In: IEEE International Conference on Computer Vision (ICCV). Volume 2015 Inter. (2015) 1520–1528
15. Badrinarayanan, V., Kendall, A., Cipolla, R.: Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2017)
16. Hariharan, B., Arbeláez, P., Girshick, R., Malik, J.: Hypercolumns for object segmentation and fine-grained localization. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Volume 07-12-June. (2015) 447–456
17. Dai, J., He, K., Li, Y., Ren, S., Sun, J.: Instance-sensitive fully convolutional networks. In: European Conference on Computer Vision (ECCV), Springer (2016) 534–549

18. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2014) 580–587
19. Uijlings, J.R.R., Van De Sande, K.E.A., Gevers, T., Smeulders, A.W.M.: Selective search for object recognition. *International Journal of Computer Vision* **104** (2013) 154–171
20. Felzenszwalb, P.F., Huttenlocher, D.P.: Efficient graph-based image segmentation. *International Journal of Computer Vision* **59** (2004) 167–181
21. Stutz, D., Hermans, A., Leibe, B.: Superpixels: An evaluation of the state-of-the-art. *Computer Vision and Image Understanding* (2017)
22. Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., Süsstrunk, S.: SLIC superpixels compared to state-of-the-art superpixel methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **34** (2012) 2274–2281
23. Gould, S., Rodgers, J., Cohen, D., Elidan, G., Koller, D.: Multi-class segmentation with relative location prior. *International Journal of Computer Vision* **80** (2008) 300–316
24. Farabet, C., Couprie, C., Najman, L., LeCun, Y.: Learning hierarchical features for scene labeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **35** (2013) 1915–1929
25. Mostajabi, M., Yadollahpour, P., Shakhnarovich, G.: Feedforward semantic segmentation with zoom-out features. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (2015) 3376–3385
26. Rother, C., Kolmogorov, V., Blake, A.: Grabcut: Interactive foreground extraction using iterated graph cuts. In: *ACM Transactions on Graphics (TOG)*. Volume 23., ACM (2004) 309–314
27. Achanta, R., Sabine, S.: Superpixels and Polygons using Simple Non-Iterative Clustering. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (2017) 4651–4660
28. Ghiasi, G., Fowlkes, C.C.: Laplacian pyramid reconstruction and refinement for semantic segmentation. In: *European Conference on Computer Vision (ECCV)*, Springer (2016) 519–534