Sparse Non-local CRF: Supplementary Materials

Olga Veksler
University of Waterloo, Canada
oveksler@uwaterloo.ca

Yuri Boykov
University of Waterloo, Canada
yboykov@uwaterloo.ca

Abstract

In the supplementary materials we discuss the advantages of using more than one quantization and give more implementation details and results for the applications. We also give and discuss failure examples.

1. One quantization vs. Two quantizatoins

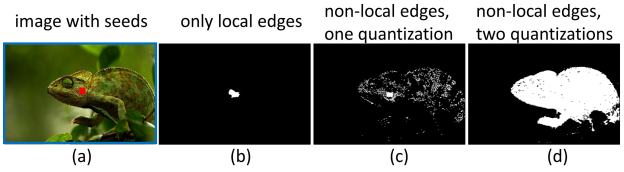

Figure 1. Illustrates the advantage of two quantizations over a single quantization. Segmentation is with seeds: red for the object, blue for the background.

Fig. 1(a) shows an image for segmentation with seeds, the red pixels are hard-constrained to the object, the blue to the background. If we use only local edges, we get segmentation in Fig. 1(b). There is no setting of w_{pq} for local edges that results in a better segmentation, as segmenting a textured patch around the seed is cheaper than segmenting the whole object. If we use a single quantization and only non-local edges, the result is in Fig. 1(c). Only the pixels that fall into the same bin as some object seed pixel are segmented as the object. Adding local edges worsens the result, as the local edges pull the isolated pixels in (c) to be labeled as the background. With two quantizations (second quantization is shifted relative to the first) we get almost the whole object segmented, Fig. 1(d). This is because pixels in the red square connect to pixels in color bins other than their own, and these, in turn, connect to other similar pixels.

2. Application: GrabCut

Here we give more details for the GrabCut application from Sec. 4.1 of the main paper.

2.1. Empty Solution Avoidance

We now give details of our empty solution avoidance technique. In principle, every image has its own optimal setting of the parameters that balance the unary and pairwise terms in Eq. (12) in the main paper. However, we have to chose parameters that generalize well for most images. Sometimes a parameter setting that generalizes well for most images works particularly badly for an individual image. For example, if too much weight is given to the pairwise terms, the result is an empty solution with everything assigned to the background. Some methods [1] develop heuristic solutions[1] to this problem.

We develop a principled approach that guarantees an empty solution is avoided, although the object can still be of small size. Our approach relies on the fact that we find a global optimum of our energy in Eq. (3) and can efficiently compute the energy of any labeling. Thus our approach cannot be applied straightforwardly for energies with dense CRF. Also note that our approach is applied after non-local random edges are selected, and they stay fixed for a given image during all the iterations of GrabCut algorithm. This is important (no change in edges) to guarantee an empty solution is avoided.

Let $\boldsymbol{x}^0$ be an empty labeling, i.e. $x_p^0 = 0$ for all p. Let $\hat{\boldsymbol{x}}$ be a reasonable non-empty solution. We want to ensure that the parameter setting is s.t. our energy in Eq. (3) is lower for a reasonable non-empty labeling than for an empty one, i.e. $E(\hat{\boldsymbol{x}}) < E(\boldsymbol{x}^0)$. If this holds, we proceed to optimization. If $E(\hat{\boldsymbol{x}}) > E(\boldsymbol{x}^0)$, we add a "balooning" term to our energy

$$E^b(\boldsymbol{x}) = \sum_{p \in \mathcal{P}} (1 - x_p). \tag{1}$$

with weight λ_b, where λ_b is just sufficiently large to have

$$E(\hat{\boldsymbol{x}}) + \lambda_b E^b(\hat{\boldsymbol{x}}) < E(\boldsymbol{x}_0) + \lambda_b E^b(\boldsymbol{x}_0), \tag{2}$$

This ensures that a reasonable solution $\hat{\boldsymbol{x}}$ is preferred to the empty solution. Observe that $E^b(\boldsymbol{x}_0) > E(\hat{\boldsymbol{x}})$, since E^b

[1] This heuristic is not discussed in the main paper but is implemented in their publically available code.

achieves its maximum at $\boldsymbol{x}_0$, and $\hat{\boldsymbol{x}}$ is not empty by assumption. Therefore a sufficiently large setting of λ_b is

$$\lambda_b = \frac{E(\hat{\boldsymbol{x}}) - E(\boldsymbol{x}_0)}{E^b(\boldsymbol{x}_0) - E^b(\hat{\boldsymbol{x}})} + \epsilon, \tag{3}$$

where ϵ is a small number.

To find a reasonable non-empty $\hat{\boldsymbol{x}}$, we sort pixels inside the box by their preference for the foreground, and hard-constrain the top r fraction of them to the foreground. In practice, we set $r = 1/6$. Then we perform optimization of Eq. (3) in the main paper, and the resulting solution is $\hat{\boldsymbol{x}}$.

The choice of λ_b in Eq. (3) guarantees an empty solution is avoided, but it may result in too much ballooning, i.e. the whole box can be assigned to the object. We avoid this as follows. Let x^1 be the labeling where all box pixels are assigned to the object, i.e. the labeling we wish to avoid. We want to ensure

$$E(\hat{\boldsymbol{x}}) + \lambda_b E^b(\hat{\boldsymbol{x}}) < E(\boldsymbol{x}_1) + \lambda_b E^b(\boldsymbol{x}_1), \tag{4}$$

Observe that $E^b(\boldsymbol{x}_1) = 0$. Therefore, to ensure λ_b is not too large, we need

$$\lambda_b < \frac{E(\boldsymbol{x}_1) - E(\hat{\boldsymbol{x}})}{E^b(\hat{\boldsymbol{x}})}, \tag{5}$$

If the value of λ_b we compute according to Eq. (3) also happens to satisfy Eq. (5), then we proceed with λ_b. Otherwise we set λ_b according to Eq. (5). This is because while Eq. (3) ensures there is no empty solution collapse, it does not give us the smallest λ_b that avoids empty solution. And, if, in fact, Eq. (5) is not satisfied, a solution that is too large, i.e. the whole box is segmented as an object is preferred to the reasonable solution $\hat{\boldsymbol{x}}$. To avoid this, λ_b needs to be reduced according to Eq. (5).

We now evaluate how well our sparse non-local CRF performs without our empty solution avoidance technique, to ensure that most of the progress is due to our model, and not due to avoiding empty solutions. In Tab. 1, we augment Tab. 3 from the main paper with three more lines: our results with no ballooning in case of Gaussian and distance weights, as well as sparse GrabCut$_2$ (our implementation of GrabCut) without ballooning. The results of GrabCut$_2$ and our sparse non-local CRF with Gaussian weights worsen just a little, whereas our method with distance weights worsens significantly, especially for ECSSD dataset. This means that distance edge weights are more susceptible to the empty solution problem, and also that our mechanism to avoid empty solution is effective. The last column of Tab. 1 gives the running time, in seconds, per image for our matlab implementation for the methods that we implement ourselves.

	GrabCut	MSRA1K	ECSSD	run time (sec)
sparse GrabCut$_1$	.909	.945	NA	-
sparse GrabCut$_2$	.897	.956	.868	1.2
dense GrabCut$_1$	**.932**	.959	.829	-
dense GrabCut$_2$	.872	.950	.837	1.4
ours (gauss)	.919	**.966**	**.892**	2.1
ours (dist)	.928	.961	.880	1.2
sparse GrabCut$_2$, no ballooning	.894	.956	.863	1.0
ours (gauss), no ballooning	.923	.959	.870	1.1

Table 1. Evaluation of our model for the GrabCut algorithm without the empty solution avoidance technique (no ballooning). The performance metric F_β score, higher is better.

2.2. Parameter Setting

Like in previous work, we tune parameters on GrabCut dataset and then apply them to the other datasets with no change. All images are scaled to 256×256. All methods are run for 10 iterations or until convergence, if it happens sooner. For the distance weights, the number of bins in color quantization is 32, number of quantizations is 2, we use 8 non-local edges per pixel (four from each quantization), in Eq. (8) in the main paper, $\sigma_{col} = 15$ and $\lambda_l = 100$, and in Eq. (10) in the main paper, $\sigma_{col} = 1$, $\lambda_{nl} = 250$. Such a small setting of $\sigma_{col} = 1$ means that only very similar pixels in non-local edges have significant weight. The large setting of λ_{nl} means that the non-local edges to these similar pixels are assigned a lot of weight. Distance between pixels connected by a non-local edge matters much less than with Gaussian weights. Which means that similar pixels in the image are connected by an edge with significant weight even if they are far apart.

For Gaussian weights, we use 2 quantizations, 64 bins in color quantization, the number of non-local edges per pixel is 2 (one from each quantization), in Eq. (8) in the main paper, $\sigma_{col} = 15$ and $\lambda_l = 100$, and in Eq. (9) in the main paper, $\sigma_{col} = 40$, $\sigma_{pos} = 5$, $\lambda_{nl} = 30$. Here σ_{col} for non-local edges is much higher than with distance weights. This means that most pixels that fall into the same color bin have equally strong weight components in terms of color. However, the distance component now is Gaussian kernel with width 5 (all images are normalized with coordinate range from 0 to 100), so distant pixels in the same color bin have much weaker weights compared to the distance weights above.

2.3. Failure Examples

Some illustrative failure examples are in Fig. 2. We show the input image (bounding box is omitted), the ground truth, the results with sparse CRF (our implementation), results from dense Cut [1], our own implementation of GrabCut with dense CRF, and our results with Gaussian and distance weights. In the first row both for Gaussian and distance weights, the darker zebra stripes are similar to some por-

tions of the background and they are missed by our methods, but not by sparse CRF, as it has a strong prior for shorter object boundaries. In the second row, the toy has similar pants to the background, so our methods fail to segment it, unlike sparse CRF. However the toy arms are segmented by our method unlike the segmentation result of sparse CRF. In the third row, the result with Gaussian weights is highly accurate, but the result of distance weights joins most of the sign to the background due to strong color similarity. With Gaussian weights, there are much fewer strong connections between the sign and the background, so the sign gets fully segmented. In the last row, our result with Gaussian weights adds a shadow from the bike to the bike itself, due to the tendency of our model to join nearby fine spurious detail.

3. Application: OneCut

We now give more details and results for OneCut [5] application from Sec. 4.2 of the main paper. The advantage of OneCut over GrabCut is that the energy function in OneCut can be optimized exactly and there is no need to iterate. The setting is, again, segmenting an object given its bounding box.

An input image is color quantized with fixed bin width in each color channel. Let $s_j(\boldsymbol{x})$ be the number of pixels in bin j that have label 1, and define

$$E^{cs}(\boldsymbol{x}) = \sum_{j\in\{1,\dots,m\}} \min\{s_j(\boldsymbol{x}), n_j - s_j(\boldsymbol{x})\}. \quad (6)$$

E^{cs} is called a color separation term since it encourages pixels in the same color bin to be assigned to the same label. Also let E^l be the standard sparse CRF with local connections

$$E^l(\boldsymbol{x}) = \sum_{(p,q)\in\mathcal{N}_l} e^{-\frac{\|C_p - C_q\|^2}{2\sigma_{col}^2}} \cdot [x_p \neq x_q], \quad (7)$$

and $E^b(\boldsymbol{x})$ be the ballooning term in Eq. (1). OneCut optimizes the following energy

$$E(\boldsymbol{x}) = E^l(\boldsymbol{x}) + \beta E^{cs}(\boldsymbol{x}) + E^b(\boldsymbol{x}), \quad (8)$$

where C_p is the color of pixel p.

We show in Sec. 3.4 of the main paper that our sparse non-local CRF can be used to optimize the energy in Eq. (8) but with a different color separation term. In particular, if we set $w_{pq} = 1$ for all non-local pairwise interactions in Eq. (4), then our color separation term is modelled through the expected non-local energy as

$$\bar{E}^{nl}(\boldsymbol{x}) = 2 \cdot \sum_{j\in\{1,\dots,m\}} \frac{s_j(\boldsymbol{x})(n_j - s_j(\boldsymbol{x}))}{n_j}, \quad (9)$$

where n_j is the number of pixels in bin j.

For our approximation of OneCut (which, more properly called, is a version of OneCut with a differnt consistency term), we replace E^{cs} in Eq. (6) by our non-local connections with $w_{pq} = 1$.

Unlike [5], who are restricted by their construction to the color separation term based only on cardinality, i.e. the number of pixels separated in color bin j, in our approach, we can implement color separation terms that have cost that depends on the color of pixels split across the bin, thus encouraging any splits to happen across less similar colors.

We experiment with color consistency term with Gaussian weights on the pixel colors

$$E^{csG}(\boldsymbol{x}) = \sum_{(p,q)\in\mathcal{N}_{nl}} e^{-\frac{\|C_p - C_q\|^2}{2\sigma_{cs}^2}} \cdot [x_p \neq x_q], \quad (10)$$

Thus our generalized OneCut energy is as in Eq. (8) but with $E^{cs}(\boldsymbol{x})$ replaced by Eq. (10).

As before, let $\boldsymbol{x}^1$ be a labeling that is 1 inside the bounding box and 0 outside the bounding box, and $\boldsymbol{x}^0$ be a labeling that is 0 everywhere. In [5], for each individual image, they set

$$\beta = \beta' \frac{E^b(\boldsymbol{x}^0)}{E^{cs}(\boldsymbol{x}^1)}, \quad (11)$$

and search for a suitable β' setting on the GrabCut dataset experimentally. If one ignores E^l and if $\beta' < 1$, such choice of β leads to ensuring that $E(\boldsymbol{x}^1) < E(\boldsymbol{x}^0)$. We choose the same approach, except we replace E^{cs} in Eq. (11) by E^{csG}.

We implement our approximation to OneCut and generalized OneCut. We use 8 random neighbors per pixel, as with OneCut we found larger non-local neighborhoods work better, see Tab. 1 in the main paper. We used 256×256 images for this evaluation. For OneCut (our implementation), we used $\sigma_{col} = 15, \lambda_l = 50$ for the local edges, and $\beta' = 0.9$. For our approximation to OneCut, we used $\sigma_{col} = 6, \lambda_l = 135$ for the local edges, and $\beta' = 0.9$. For our generalized OneCut, we used $\sigma_{col} = 15, \lambda_l = 50$ for the local edges, $\sigma_{col} = 2$ for the non-local edges in Eq. (10), and $\beta' = 0.9$.

The results of the original OneCut [5] (our re-implementation of it), our approximation of OneCut, and our generalization of OneCut and are in Tab. 2 for several bin sizes. Our generalization works better for GrabCut dataset, and on other datasets, the performance is comparable for 64 bins, and much better as the number of bins decreases. OneCut (and ours OneCut approximate) are much more sensitive to the number of bins per channel compared to our generalized OneCut. This is because as the number of bins decreases, any particular bin is likely to contain pixels from different objects, but the cost of breaking it depends only on the partition size under OneCut (and its approximation). However, pixels that do belong to one object tend

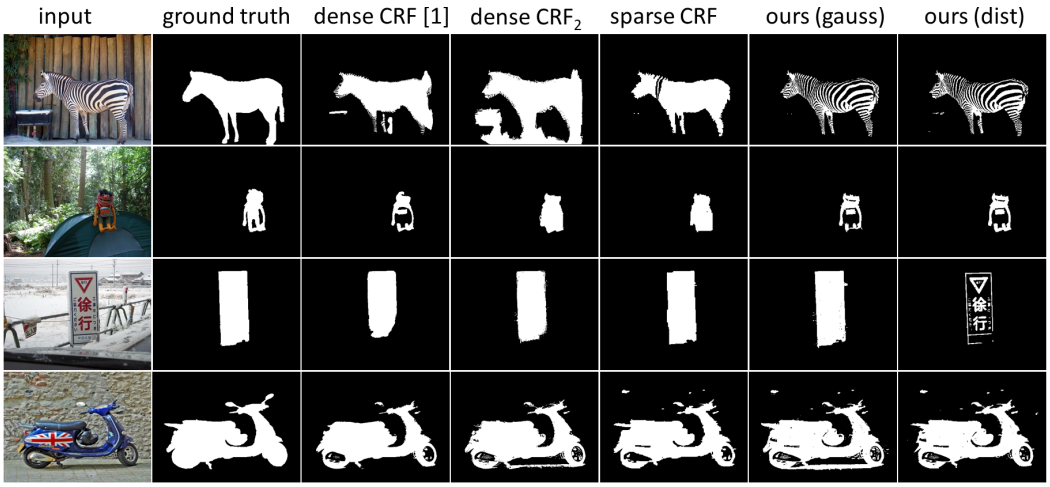

Figure 2. Failure examples for the GrabCut application, from left to right: input image; ground truth; GrabCut with sparse CRF our implementation; results from dense Cut [1]; our own implementation of GrabCut with dense CRF; our sparse non-local CRF with distance edge weights; our sparse non-local CRF with Gaussian edge weights.

	OneCut				Ours (approximate OneCut)				Ours (generalized OneCut)			
bins per channel	8	16	32	64	8	16	32	64	8	16	32	64
GrabCut	.797	.847	.867	.897	.799	.850	.871	.899	.875	.892	.903	**.908**
MSRA1K	.942	.945	.948	.948	.949	.951	.951	**.949**	.942	.946	.949	**.949**
ECSSD	.685	.753	.797	.818	.694	.762	.804	**.824**	.810	.810	.810	.813
running time (sec)	0.96	0.50	0.60	1.22	0.94	0.86	0.80	0.72	0.5	0.6	0.80	0.78

Table 2. Evaluation of OneCut [5], our approximation of it, and our generalizaiton of it for several different bin sizes. The performance metric F_β score, higher is better.

to have more similar colors. For our generalized OneCut, as the cost of splitting the bin does depend on pixel colors, it works better for separating bins according to the object membership. The running times are in the last row of Tab. 2.

A qualitative evaluation on images from GrabCut dataset are in Fig. 3 for the case of 64 bins per channel. On some images, the performance is almost identical, on some, our generalized OneCut preserves the details better.

OneCut does not work as well as GrabCut, especially for ECSSD dataset. Examining the results, it is mostly due to many images completely labelled as the background. The parameter setting of the ballooning term in Eq. (11) is not always successful.

4. Application: Salient Object Segmentation

We now give more details and results for salient object segmentation application from Sec. 4.3 of the main paper. In Sec. 4.1 we describe the regularized loss approach to weakly supervised salient object segmentation developed in [7]. In Sec. 4.2 we describe our changes to [7] and give more experimental results.

4.1. Sparse Local CRF for Weak Supervision

In [7] they develop a method for image tag weakly supervised single class semantic segmentation, i.e. the dataset contains a single object class of interest. Image tag weak supervision means that there is no pixel precise ground truth, but only the knowledge that the class of interest is present in the image. This approach is naturally applicable to salient object segmentation. The main tool in [7] is regularized loss, based on sparse CRF.

We now describe the approach in [7]. There are two classes: the object (class 1) and background (class 0). Let $\boldsymbol{x}$ be the output of CNN, where $\boldsymbol{x}$ has the same size as the input image. The output of CNN for pixel p is denoted by x_p. The last layer of CNN is softmax, so that $x_p \in (0, 1)$. The most important component of the regularized loss is sparse CRF

$$L_{crf}(\boldsymbol{x}) = \frac{1}{|\mathcal{P}|} \sum_{(p,q)\in\mathcal{N}} e^{-\frac{||C_p - C_q||^2}{2\sigma^2}} \cdot |x_p - x_q|, \quad (12)$$

where $\mathcal{P}$ is the set of all pixels in the image, $\mathcal{N}$ is the set of neighboring pixel pairs, C_p is the color of p. If CNN produces a sharp distribution, i.e. most x_p are close either to

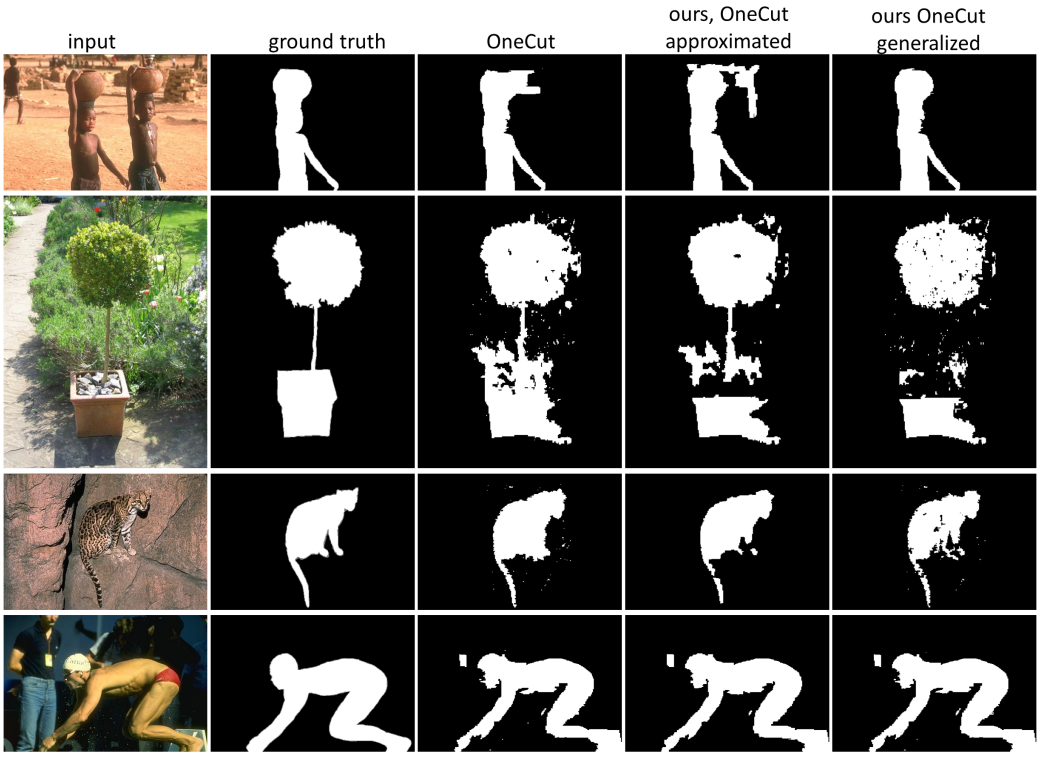

Figure 3. Qualitative comparison of OneCut, our approximation of it, and our generalization of it on some images from GrabCut dataset. Bounding boxes in the input image are ommitted.

0 or to 1, then if two nearby pixels are not assigned to the same class, a loss related to image edge strength between pixels p and q is incurred. The lowest value of sparse-CRF is zero, incurred for a trivial solution: everything is classified as the object (or background). Thus one cannot train with sparse CRF alone.

The second component of regularized loss is volumetric loss. It penalizes empty (or full) solutions. Let $\bar{\boldsymbol{x}} = \frac{1}{|\mathcal{P}|}\sum_{p\in\mathcal{P}} x_p$, i.e. the normalized object size. There are two volumetric losses: batch and minimum volume. The batch volumetric loss L_b is defined for a batch of m images with outputs $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m$:

$$L_b(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m) = (\frac{1}{m}\sum_i \bar{\boldsymbol{x}}_i - 0.5)^2.$$

This loss encourages the average size of objects in a batch to be half of the image size.

The second volumetric loss is the minimum volume loss L_m. It acts on a single image output $\boldsymbol{x}$ and penalizes the result if the normalized object size is less than obj_{min}

$$L_m(\boldsymbol{x}) = \left[\bar{\boldsymbol{x}} < obj_{min}\right] \cdot \left(\bar{\boldsymbol{x}} - obj_{min}\right)^2, \qquad (13)$$

where $\left[\cdot\right]$ is 1 if the argument is true and 0 otherwise. Following [7], we set $obj_{min} = 0.15$.

For a dataset with one object class, the border pixels are likely to be background, and the center pixel is likely to be object. These priors are incorporated in positional loss. Let $\mathcal{B}$ be the set of pixels on the image border of width $w = 3$. Let $\mathcal{C}$ be the pixels in the central square of the image of size $c = 3$. The positional loss $L_p(\boldsymbol{x})$ is

$$L_p(\boldsymbol{x}) = (\frac{1}{|\mathcal{B}|}\sum_{p\in\mathcal{B}} x_p)^2 + (\frac{1}{|\mathcal{C}|}\sum_{p\in\mathcal{C}} x_p - 1)^2. \qquad (14)$$

The complete regularized loss for training is a weighted sum of components

$$L_{reg}(\boldsymbol{x}) = \lambda_{crf}L_{crf}(\boldsymbol{x}) + \lambda_b L_b(\boldsymbol{x}) + \lambda_m L_m(\boldsymbol{x}) + \lambda_p L_p(\boldsymbol{x}). \qquad (15)$$

In [7] they devise a two-stage strategy for training CNN with loss in Eq. (15): first annealing and then the normal stage. They also find that the annealing stage can be replaced by weight transfer. First, CNN is trained on Oxford-Pet [6] dataset, in both stages. Then, for any new dataset, the weights from CNN trained on OxfordPet are transferred, and training starts in the normal stage with $\lambda_{crf} = 100$, $\lambda_m = 5$, $\lambda_b = \lambda_p = 1$, $\sigma = 0.15$.

4.2. Sparse Non-local CRF for Weak Supervision

In our application, we replace the sparse CRF loss in Eq. (15) by our sparse non-local CRF.[2]. We evaluate both

[2]Note that in [7], they tested dense CRF for their regularized loss, but found that it is difficult to train with it, perhaps due to the difficulty of

	32 bins per channel			64 bins per channel		
number of edges	2	4	8	2	4	8
$r = 10/11$	.900	.899	.898	.894	.900	.900
$r = 2/3$	.987	.901	**.906**	.901	.900	.900
$r = 1/2$	.900	.900	.900	**.906**	.902	.901

Table 3. Ablation for different choices of bins per channel, number of edges, and r when training and evaluating on MSRAB dataset. The performance metric F_β score, higher is better.

	32 bins per channel			64 bins per channel		
σ_{pos}	5	10	20	5	10	20
$r = 10/11$	.900	.891	.892	.901	.893	.890
$r = 2/3$	.901	.886	.893	**.902**	.893	.893
$r = 1/2$	**.902**	.890	.897	.900	.897	.889

Table 4. Ablation of different choices for the number of bins per channel, σ_{pos}, and r when training and evaluating on MSRAB dataset. The performance metric F_β score, higher is better.

the distance and Gaussian weights from Sec. 4.1 of the main paper. We use CNN architecture from [7][3], Unet [4] with ResNeXt [9] fixed features pretrained on Imagenet [2] and train 256×256 images. Like in [7], we start with weights pretrained on OxfordPet dataset. We train for 100 epochs.

We largely use the same parameter setting as in Sec. 2 for both distance and Gaussian weights, with two exceptions. First difference is that we use the same value of $\sigma_{col} = 0.15$ as in [7] for the color Gaussian component of all local and non-local w_{pq} terms, namely in Eqs. (8) to (10) in the main paper. The color channels are normalized for input to CNN, so 0.15 is roughly equivalent to 15 used with unnormalized color channels for the application in Sec. 2. The second difference concerns λ_l and λ_{nl} weights in Eqs. (8) and (10) of the main paper. We ensure that our total CRF weight is the same as the total CRF weight in [7], thus $\lambda_{crf} = 100$ in Eq. (15). Then we distribute a fraction r of it among the non-local edges, and a fraction $1-r$ among the local edges. We chose $r = 2/3$, which gives non-local portions of CRF twice more weight than the local edges. We also evaluate other choices for r in Tabs. 3 and 4.

The other weights are set as before. For the distance weights, the number of bins in color quantization is 32, number of quantizations is 2, we use 8 non-local edges per pixels (four from each quantization). For Gaussian weights, we use 2 quantizations, 64 bins in color quantization, the number of non-local edges per pixel is 2 (one from each quantization), and in Eq. (9) in the main paper, $\sigma_{pos} = 5$.

We use datasets DUTS [8] and MSRAB [3] for training, and our results and quantitative comparisons are reported in the main paper. Here we perform ablation studies on MSRAB dataset for different choices of number of bins, number of neighbors, and rate r of distribution of λ_{CRF} between the local and non-local weights. For distance and Gaussian weights, the results are reported in Tabs. 3 and 4, in terms of F_β score on the test fold of MSRAB dataset. The results are quite stable in terms of varying these parameters compared to the classical applications. This is probably because CNNs learn much more discriminitive features, compared to the simple color models used in classical applications.

We also compare CNN trained with weak supervision and CNN with the same architecture but trained with full ground truth in Tabs. 5 and 6 for MSRA and DUTS datasets, respectively. On MSRAB dataset, there is almost no gap between our classifiers and the classifier trained with ground truth, which implies that any further performance improvement would need a more more powerful CNN architecture. On DUTS dataset, there is a small but significant gap, which means it is still possible to improve performance by designing a better loss function, rather than switching to a different architecture. Interestingly, classifier trained with ground truth does not always has the best performance on other datasets.

We give more qualitative examples in Fig. 4, for quantitative comparisons to these methods see the main paper. In this figure we also compare to [10, 11], who use scribbles, much stronger forms of supervision, to sparse CRF [7], and to the same CNN architecture as ours, but trained on ground truth. Our methods preserve details much better than [7, 10, 11]. For some images, for example the bicycle in the forth row, some of the extracted details are more accurate than the ground truth annotations. Sometimes our method adds fine spurious details, like the shadow of the bicycle in row 4 or bowling alley lane edges in row 8. Some failure examples are in Fig. 5. Our method can have problems extracting very thin objects with washed out intensity edges, and can join brightly colored structures to the salient object. For other methods we compare to, these images are also challenging.

minimizing the loss function of their type. Using dense CRF in a loss function is different from using dense CRF as part of architecture. Since there is no pixel precise ground truth, having dense CRF in architecture does not help in designing a loss function useful for training in weakly supervised setting.

[3] https://github.com/morduspordus/SingleClassRL

References

[1] Ming-Ming Cheng, Victor Adrian Prisacariu, Shuai Zheng, Philip HS Torr, and Carsten Rother. Densecut: Densely connected crfs for realtime grabcut. In *Computer Graphics Forum*, volume 34, pages 193–201. Wiley Online Library, 2015. 1, 2, 4

[2] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009. 6

	MSRAB	ECSSD	DUTO	PascalS	THUR	SOD
ground truth training	**.909**	**.907**	.753	.828	.728	**.849**
ours (gauss)	.902	.898	.767	**.842**	.729	.812
ours (dist)	.907	.902	.771	.841	**.734**	.810

Table 5. MSRAB training dataset: Comparison to classifier with the same architecture that we use, but trained with full ground truth. Performance metrics are F_β (higher is better).

	MSRAB	ECSSD	DUTO	PascalS	THUR	SOD	DUTS
ground truth training	.873	.918	.789	**.874**	.738	**.870**	**.867**
ours (gauss)	.875	.905	.781	.859	.731	.845	.827
ours (dist)	**.878**	.909	**.790**	.862	**.745**	.840	.839

Table 6. DUTS training dataset: Comparison to classifier with the same architecture that we use, but trained with full ground truth. Performance metrics are F_β (higher is better).

[3] Tie Liu, Jian Sun, Nan-Ning Zheng, Xiaoou Tang, and Heung-Yeung Shum. Learning to detect a salient object. In *Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2007. 6

[4] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention*, pages 234–241, 2015. 6

[5] Meng Tang, Lena Gorelick, Olga Veksler, and Yuri Boykov. Grabcut in one cut. In *ICCV*, 2013. 3, 4

[6] Andrea Vedaldi. Cats and dogs. In *Conference on Computer Vision and Pattern Recognition*, 2012. 5

[7] Olga Veksler. Regularized loss for weakly supervised single class semantic segmentation. In *European Conference on Computer Vision*, pages 348–365. Springer, 2020. 4, 5, 6, 8

[8] Lijun Wang, Huchuan Lu, Yifan Wang, Mengyang Feng, Dong Wang, Baocai Yin, and Xiang Ruan. Learning to detect salient objects with image-level supervision. In *CVPR*, 2017. 6

[9] Saining Xie, Ross B. Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Conference on Computer Vision and Pattern Recognition*, pages 5987–5995, 2017. 6

[10] Siyue Yu, Bingfeng Zhang, Jimin Xiao, and Eng Gee Lim. Structure-consistent weakly supervised salient object detection with local saliency coherence. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 3234–3242. AAAI Press, 2021. 6, 8

[11] Jing Zhang, Xin Yu, Aixuan Li, Peipei Song, Bowen Liu, and Yuchao Dai. Weakly-supervised salient object detection via scribble annotations. In *Conference on Computer Vision and Pattern Recognition*, pages 12546–12555, 2020. 6, 8

Figure 4. Qualitative comparison. From left to right: input image, ground truth, method trained with scribbles [11], method trained with scribbles [10], sparse CRF [7], our result with distance weight, our results with Gaussian weights, the same CNN architecture as we use, but trained with full ground truth.

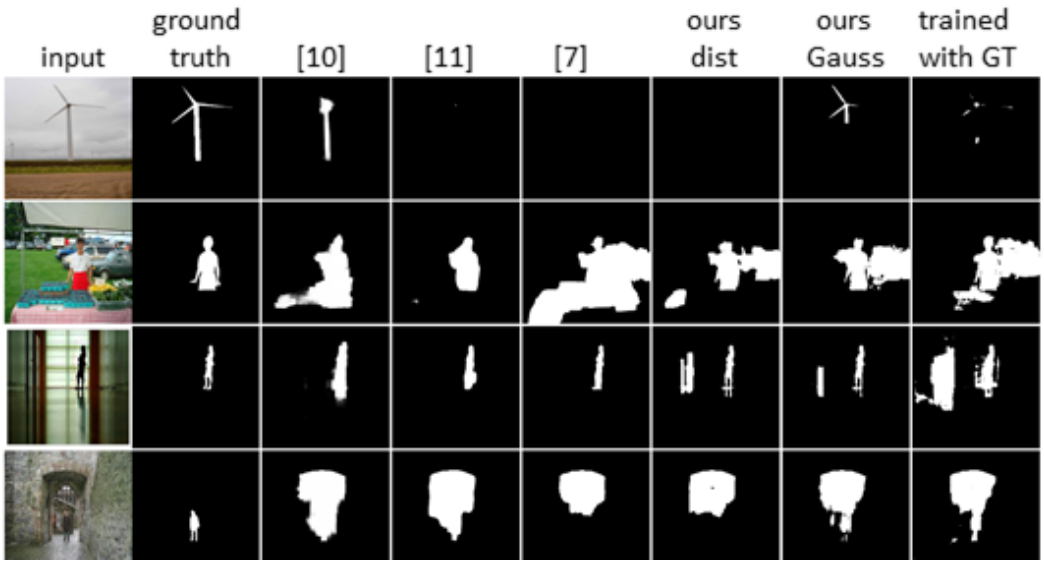

Figure 5. Some failure examples on the same methods as in Fig. 4.