Background-Aware Pooling and Noise-Aware Loss for Weakly-Supervised Semantic Segmentation
Supplement

Youngmin Oh Beomjun Kim Bumsub Ham*
School of Electrical and Electronic Engineering, Yonsei University

We present a detailed analysis for hyperparameters (Sec. 1), and provide a quantitative comparison between GAP and BAP (Sec. 2). We also show runtime comparisons of our pseudo label generator and other methods (Sec. 3). We then compare dot product based and cosine similarity based classifiers for semantic segmentation (Sec. 4), and describe training losses in detail (Sec. 5). We finally show more quantitative and qualitative results on PASCAL VOC 2012 [3] and MS-COCO [9] (Sec. 6).

1. Hyperparameters

Grid size (N). To analyze an effect of a grid size, we use different grid sizes for training a classification network using BAP and for generating pseudo segmentation labels (*i.e.*, u_0). Table 1(a) compares performance of pseudo labels on the PASCAL VOC 2012 [3] $train$ set. We can see that training with a single query (*i.e.*, $N = 1$) provides the worst result. A plausible explanation is that a definite background might contain inhomogeneous clutter, and thus a larger grid size can help the classification network to use diverse queries, acting as a regularizer. On the other hand, for generating pseudo labels, we empirically find that a smaller grid size tends to give better results. Accordingly, we set the grid size N to 4 for training the classification network, and to 1 for generating pseudo labels. We also show in Table 1(b) segmentation results, trained with corresponding pseudo labels in Table 1(a), on the PASCAL VOC 2012 val set. To this end, we use DeepLab-V1 [1, 2] with the standard cross-entropy loss. We can see that more accurate pseudo labels give better segmentation results.

Damping and balance parameters (γ and λ). We show in Fig. 1(a) an effect of damping parameters on confidence scores $\sigma(\mathbf{p}) = (\alpha(\mathbf{p}))^{\gamma}$, where

$$\alpha(\mathbf{p}) = \frac{D_{c^*}(\mathbf{p})}{\max_c(D_c(\mathbf{p}))}. \quad (1)$$

Note that confidence scores turn into binary ones with a large value of γ, similar to BCM [15]. We show in Fig. 1(b) the performance of DeepLab-V1 [1, 2] for different parameters γ and λ on the PASCAL VOC 2012 [3] val set. We select a pair of parameters which provides the best result (*i.e.*, $\gamma = 7$, $\lambda = 0.1$).

*Corresponding author.

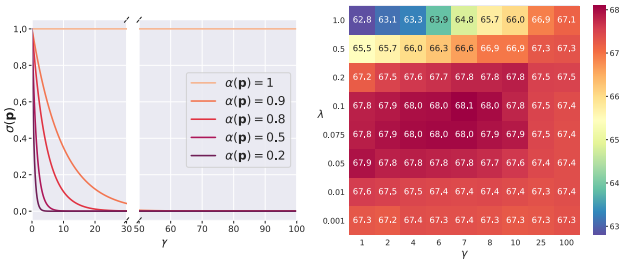
Table 1: (a) Comparison of our pseudo labels Y_{crf} using different grid sizes on the PASCAL VOC 2012 [3] $train$ set in terms of mIoU. (b) Comparison of mIoU scores using DeepLab-V1 [1, 2], trained with corresponding pseudo labels in (a), on the PASCAL VOC 2012 val set. The best performance is reported in bold and the second best is underlined.

Grid Size N		For Generating 1	2	3
For Training	1	75.26	75.28	75.24
	2	76.68	76.63	76.57
	3	78.36	78.24	78.18
	4	**78.70**	<u>78.50</u>	78.48
	5	78.30	78.10	78.12

(a)

Grid Size N		For Generating 1	2	3
For Training	1	64.88	64.81	64.83
	2	65.90	65.88	65.87
	3	67.66	67.67	67.69
	4	**67.82**	<u>67.79</u>	67.74
	5	67.73	67.65	67.66

(b)



(a) (b)

Figure 1: (a) Confidence scores σ with different damping parameters γ. (b) Comparison of mIoU scores for different hyperparameters (γ and λ) on the PASCAL VOC 2012 [3] val set. Best viewed in color.

2. Comparison of GAP and BAP

To further verify that BAP gives better CAMs [17] than GAP, we quantify the quality of CAMs for different pooling methods. To this end, we consider normalized CAM scores u_c (Eq. (5) of the main paper) as *True-*

Table 2: Quantitative comparison of CAMs using different pooling methods on the PASCAL VOC 2012 [3] *train* set.

Method	aero	bike	bird	boat	bot	bus	car	cat	cha	cow	tab	dog	hor	mbik	pers	plnt	she	sofa	trai	tv	mean
GAP	62.7	27.3	68.4	59.1	86.2	91.4	86.7	83.3	51.5	82.6	74.6	81.6	74.7	69.7	67.2	68.2	79.0	69.5	84.1	92.1	73.0
BAP	69.9	68.2	71.9	70.7	84.2	93.5	87.5	88.3	57.2	84.7	66.4	85.0	75.8	79.1	71.5	79.0	83.1	66.0	88.8	87.1	77.9

Table 3: Comparison of mIoU scores on the PASCAL VOC 2012 [3] *val* set.

Method	*val*
Architecture: DeepLab-V1 [1, 2]	
SDI [7]	65.7
BCM [15]	66.8
Ours w/ NAL	
GAP	66.3
BAP	68.1
Architecture: DeepLab-V2 [2]	
SDI [7]	74.2
BCM [15]	70.2
Ours w/ NAL	
GAP	72.6
BAP	74.6

Table 4: Runtime comparison of our pseudo label generator and other segmentation methods. We report the average runtime on the PASCAL VOC 2012 [3] *val* set.

Times (s)	GrabCut [14]	MCG [11]	WSSL [10]	Ours
CPU	1.9	25.5	0.4	0.4
GPU	-	-	-	0.1
Total	1.9	25.5	0.4	0.5

Positives (TP_c) if they are activated inside ground-truth regions for a class c. Otherwise, we regard them as *False-Positives* (FP_c). We define the precision for the class c as: $\frac{\text{TP}_c}{\text{TP}_c+\text{FP}_c}$. We compare in Table 2 the per-class precision of CAMs obtained with different pooling methods. We can clearly see that BAP outperforms GAP, especially for *bike*, *boat*, and *plant* classes. A plausible reason is that GAP may hinder classifier weights since it aggregates a mixture of foreground and background features inside object bounding boxes. BAP overcomes this problem by computing a background attention map adaptively, encouraging the classifier weights to focus more on foreground features.

We also evaluate the classification networks using different pooling methods on the PASCAL VOC 2012 [3] *val* set. To this end, we consider ground-truth boxes as pre-obtained ones to crop each object. We find that the classification accuracy increases from 80.3% to 82.3% by using BAP instead of GAP. This suggests that BAP could be helpful in improving object classification/detection performance.

Table 3 shows mIoU scores of DeepLab-V1 and -V2 [1, 2], trained with different pseudo labels using GAP or BAP, on the PASCAL VOC 2012 *val* set. We can see that the models trained with the pseudo labels using GAP ('GAP') achieve competitive mIoU scores to BCM [15], demonstrating once again the effectiveness of our approach to using a classification network with bounding boxes.

Table 5: Comparison of mIoU scores using DeepLab-V1 [1, 2] on the PASCALS VOC 2012 [3] *val* set. For our NAL, we use two different classifiers. DP: a dot product based classifier. CS: a cosine similarity based classifier.

Method	*val*
SDI [7]	65.7
BCM [15]	66.8
Ours w/ NAL	
using DP	67.8
using CS ($\tau = 20$)	68.1

Table 6: Results of DeepLab-V2 with different initialization on the PASCAL VOC 2012 [3] *val* set. Numbers in brackets indicate fully-supervised performance. FS denotes performance relative to the fully-supervised one.

Method	w/ ImageNet		w/ MS-COCO	
	mIoU	FS	mIoU	FS
SDI [7]	69.4 (74.5)	93.2	74.2 (77.7)	95.5
BCM [15]	-	-	70.2 (74.5)	94.2
Ours w/ NAL	70.9 (73.4)	96.6	74.6 (77.5)	96.3

3. Runtime

We show in Table 4 a runtime comparison for generating pseudo labels. For fair comparison, we use a NVIDIA Titan RTX GPU with an Intel i5 3.7GHz for all experiments. For the result of MCG [11], we use a publicly available MATLAB implementation, and choose the best segment proposal for each bounding box. Our pseudo label generator mainly consists of two parts: (1) feature extraction with computing a unary term (*i.e.*, u_c and u_0); (2) DenseCRF [8]. The first part takes 0.1 seconds on the GPU and the second one takes 0.4 seconds on the CPU. WSSL also adopts DenseCRF to generate pseudo segmentation labels from object bounding boxes. Compared to WSSL, our pseudo label generator requires a negligible overhead (0.1 seconds), while it brings substantial mIoU gains. For example, our pseudo labels Y_{crf} achieve the mIoU gain of 9.0/8.1% over WSSL on the PASCAL VOC 2012 [3] *train*/*val* sets (see Table 1 in the main paper).

4. Classifier for semantic segmentation

Here we provide a detailed description of two different classifiers for semantic segmentation. First, a dot product (DP) based classifier computes a probability for a class c as follows:

$$H_c(\mathbf{p}) = \frac{e^{\phi(\mathbf{p})\cdot W_c}}{\sum_i e^{\phi(\mathbf{p})\cdot W_i}}. \quad (2)$$

Table 7: Per-class results of DeepLab-V1 [1, 2] in terms of mIoU on the PASCAL VOC 2012 [3] dataset: (a) the results on the *val* set, (b) the results on the *test* set. All numbers except for ours are taken from corresponding papers.

Method	bkg	aero	bike	bird	boat	bot	bus	car	cat	cha	cow	tab	dog	hor	mbik	pers	plnt	she	sofa	trai	tv	mean
SDI [7]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	65.7
BCM [15]	89.8	68.3	27.1	73.7	56.4	72.6	84.2	75.6	79.9	35.2	78.3	53.2	77.6	66.4	68.1	73.1	56.8	80.1	45.1	74.7	54.6	66.8
Ours w/ NAL	91.0	78.1	30.8	83.1	62.4	67.7	85.9	78.4	83.2	34.8	77.3	52.7	76.2	73.6	70.3	74.0	49.9	77.7	40.5	79.3	64.6	68.1

(a)

Method	bkg	aero	bike	bird	boat	bot	bus	car	cat	cha	cow	tab	dog	hor	mbik	pers	plnt	she	sofa	trai	tv	mean
SDI [7]	-	78.1	31.1	72.4	61.0	67.2	84.2	78.2	81.7	27.6	68.5	62.1	76.9	70.8	78.2	76.3	51.7	78.3	48.3	74.2	58.6	67.5
Ours w/ NAL	91.7	80.5	32.5	80.5	61.1	66.2	86.6	79.2	85.4	28.6	73.6	60.8	79.4	71.5	78.2	73.9	57.5	81.1	48.2	78.7	62.8	69.4

(b)

Table 8: Per-class results of DeepLab-V2 [2] in terms of mIoU on the PASCAL VOC 2012 [3] dataset: (a) the results on the *val* set, (b) the results on the *test* set. All numbers except for ours are taken from corresponding papers.

Method	bkg	aero	bike	bird	boat	bot	bus	car	cat	cha	cow	tab	dog	hor	mbik	pers	plnt	she	sofa	trai	tv	mean
SDI [7]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	74.2
BCM [15]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	70.2
Ours w/ NAL	93.1	83.8	33.1	87.4	64.1	78.3	93.2	84.4	89.3	39.3	84.2	61.9	85.2	80.1	76.9	79.4	57.5	84.6	51.9	85.8	74.7	74.6

(a)

Method	bkg	aero	bike	bird	boat	bot	bus	car	cat	cha	cow	tab	dog	hor	mbik	pers	plnt	she	sofa	trai	tv	mean
Ours w/ NAL	93.6	85.0	34.5	90.4	65.2	75.9	92.0	85.2	91.3	38.4	84.9	67.6	86.6	85.8	82.3	79.3	62.3	87.7	60.0	81.8	69.2	76.1

(b)

We omit bias terms to make the classifier weight W to be more representative for corresponding classes. The classifier weights depend on both magnitude and direction of the features ϕ, and thus this classifier is not robust to intra-class variations. It has recently been shown that a cosine similarity (CS) based classifier could outperform the DP based one [4, 12, 16]. Specifically, the CS based classifier modifies Eq. (2) as follows:

$$H_c(\mathbf{p}) = \frac{e^{\tau \frac{\phi(\mathbf{p})}{\|\phi(\mathbf{p})\|} \cdot \frac{W_c}{\|W_c\|}}}{\sum_i e^{\tau \frac{\phi(\mathbf{p})}{\|\phi(\mathbf{p})\|} \cdot \frac{W_i}{\|W_i\|}}}, \tag{3}$$

where we denote by τ a scale parameter. This encourages classifier weights and features to lie on a hypersphere, whose radius is τ. Since this classifier considers the angle between the feature ϕ and the classifier W_i only, we expect that it is more robust against intra-class variations than the DP based one.

We report in Table 5 mIoU scores of DeepLab-V1 [1, 2] using DP and CS based classifiers on the PASCAL VOC 2012 [3] *val* set. We can see that the CS based classifier outperforms the DP based one slightly. Accordingly, we exploit the CS based classifier with the scale parameter τ of 20. It is worth noting that our model using the DP based classifier still outperforms the state of the art [7, 15].

5. Training losses

We describe more details on training losses to deal with the unreliable regions $\sim\mathcal{S}$ (see Table 3 in the main paper). We define entropy regularization [5] as follows:

$$\mathcal{L}_{ER} = -\frac{1}{|\sim\mathcal{S}|} \sum_{\mathbf{p}\in\sim\mathcal{S}} \sum_c H_c(\mathbf{p}) \log H_c(\mathbf{p}), \tag{4}$$

where $|\cdot|$ indicates the number of pixels. This encourages a $(L+1)$-dimensional probability map H to have one clear peak at each position $\mathbf{p}$ for a confident prediction. The overall loss is defined as follows:

$$\mathcal{L} = \mathcal{L}_{ce} + \lambda \mathcal{L}_{ER}, \tag{5}$$

where we set λ to 0.1. Motivated by the work of [13], we also use a bootstrapping technique as follows:

$$\mathcal{L}_{BS} = -\frac{1}{|\sim\mathcal{S}|} \sum_{\mathbf{p}\in\sim\mathcal{S}} \sum_c Y_c(\mathbf{p}) \log H_c(\mathbf{p}), \tag{6}$$

where Y_c indicates the c-th element of a new target Y. Specifically, we define the new target vector at each position $\mathbf{p}$ as follows:

$$Y(\mathbf{p}) = \beta y(\mathbf{p}) + (1-\beta) H(\mathbf{p}), \tag{7}$$

where β is a balance parameter between pseudo ground-truth labels Y_{crf} and model predictions H. y is a $(L+1)$-dimensional one-hot vector, with a value of 1 at the c-th dimension, where $c = Y_{\text{crf}}(\mathbf{p})$. The bootstrapping technique is similar to an EM-like algorithm, where the new target vector is estimated in the E-step, and the model is optimized to better predict such a new target in the M-step. We adjust the balance parameter β from 1 to 0 by using the poly

schedule during training. The overall loss is then defined as follows:

$$\mathcal{L} = \mathcal{L}_{ce} + \lambda \mathcal{L}_{BS}, \tag{8}$$

where we also set λ to 0.1.

6. More segmentation results

PASCAL VOC. To test fully-supervised performance, we train DeepLab-V1 [1, 2] with ground-truth segmentation labels, achieving a mIoU score of 69.5% on the PASCAL VOC 2012 [3] *val* set. This is similar to the mIoU scores of 69.1% and 69.8% reported in [7, 15], respectively. We show in Table 6 mIoU scores of DeepLab-V2 [2] using different initialization. For comparison, we also provide fully-supervised performance. We can see that our approach gives better results than other methods.

Table 7 compares per-class mIoU scores of DeepLab-V1 on the PASCAL VOC 2012 dataset. We can clearly see that our approach outperforms other WSSS methods by a large margin on both *val* and *test* sets. For example, our approach gives better results than BCM and SDI [7] for 13 and 14 classes, respectively. This demonstrates the importance of high-quality pseudo labels. We also provide in Table 8 per-class mIoU scores of DeepLab-V2 on the same dataset. Compared to the results of Table 7, we can see that using a deeper CNN improves the mIoU performance for all classes, achieving a new state of the art.

We show in Fig. 2 visual examples of u_c, u_o, and pseudo segmentation labels. We can see that our approach using BAP generates high-quality background attention maps u_o, leading to better CAMs u_c. Also, we show that Y_{ret} is effective to identify unreliable regions. For example, the *person*'s legs in the fourth row are incorrectly labeled as a *cow* class, which can be marked by our approach. We show in Figs. 3 and 4 qualitative results of DeepLab-V1 and -V2 on the PASCAL VOC 2012 dataset, respectively. We can see that the networks trained with pseudo segmentation labels only (Weak) already yield satisfactory results, and exploiting a small number of ground-truth pixel-level labels (Semi) provides more accurate results with sharp object boundaries. The last rows for each figure show failure examples.

MS-COCO. We show in Table 9 AP scores of Mask-RCNN [6] on the MS-COCO [9] *test* set. We train Mask-RCNN with each of our pseudo labels, *i.e.*, Y_{crf} and Y_{ret}. The behavior of AP scores is similar to that of mIoU scores for semantic segmentation (see Tables 4 and 5 in the main paper). Compared to the results of Table 6 in the main paper, we find that exploiting both Y_{crf} and Y_{ret} gives better results especially for AP, AP_{50}, AP_{75}, AP_S, and AP_M. This again confirms two pseudo labels Y_{crf} and Y_{ret} are complementary to each other.

In Fig. 5, we show visual examples of pseudo segmentation labels on the MS-COCO *train* set. We find that pseudo labels of VOC-to-COCO show reasonable results even for unseen classes (*e.g.*, *giraffe* and *pizza* classes), demonstrating the effectiveness of our pseudo label generator. We also show in Fig. 6 instance segmentation results of Mask-RCNN on the MS-COCO dataset. We can see that the model trained with COCO-to-COCO gives better results than the one using VOC-to-COCO.

Table 9: Quantitative comparison of Mask-RCNN [6], trained with either Y_{crf} or Y_{ret}, on the MS-COCO [9] *test* set.

Method	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
VOC-to-COCO						
BAP: Y_{crf}	17.0	35.6	14.4	5.8	16.6	28.4
BAP: Y_{ret}	12.7	35.1	6.5	6.1	12.5	19.2
COCO-to-COCO						
BAP: Y_{crf}	20.7	41.9	18.4	8.0	20.2	32.2
BAP: Y_{ret}	17.5	43.5	11.5	8.9	16.7	24.9

References

[1] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Semantic image segmentation with deep convolutional nets and fully connected CRFs. In *ICLR*, 2015. 1, 2, 3, 4, 6

[2] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. *IEEE Trans. PAMI*, 40(4):834–848, 2018. 1, 2, 3, 4, 6, 7

[3] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (VOC) challenge. *IJCV*, 88(2):303–338, 2010. 1, 2, 3, 4, 5, 6, 7

[4] Spyros Gidaris and Nikos Komodakis. Dynamic few-shot visual learning without forgetting. In *CVPR*, 2018. 3

[5] Yves Grandvalet and Yoshua Bengio. Semi-supervised learning by entropy minimization. In *NeurIPS*, 2005. 3

[6] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, 2017. 4, 9

[7] Anna Khoreva, Rodrigo Benenson, Jan Hosang, Matthias Hein, and Bernt Schiele. Simple Does It: Weakly supervised instance and semantic segmentation. In *CVPR*, 2017. 2, 3, 4

[8] Philipp Krähenbühl and Vladlen Koltun. Efficient inference in fully connected CRFs with gaussian edge potentials. In *NeurIPS*, 2011. 2

[9] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 1, 4, 8, 9

[10] George Papandreou, Liang-Chieh Chen, Kevin P Murphy, and Alan L Yuille. Weakly-and semi-supervised learning of a deep convolutional network for semantic image segmentation. In *ICCV*, 2015. 2

[11] Jordi Pont-Tuset, Pablo Arbelaez, Jonathan T Barron, Ferran Marques, and Jitendra Malik. Multiscale combinatorial

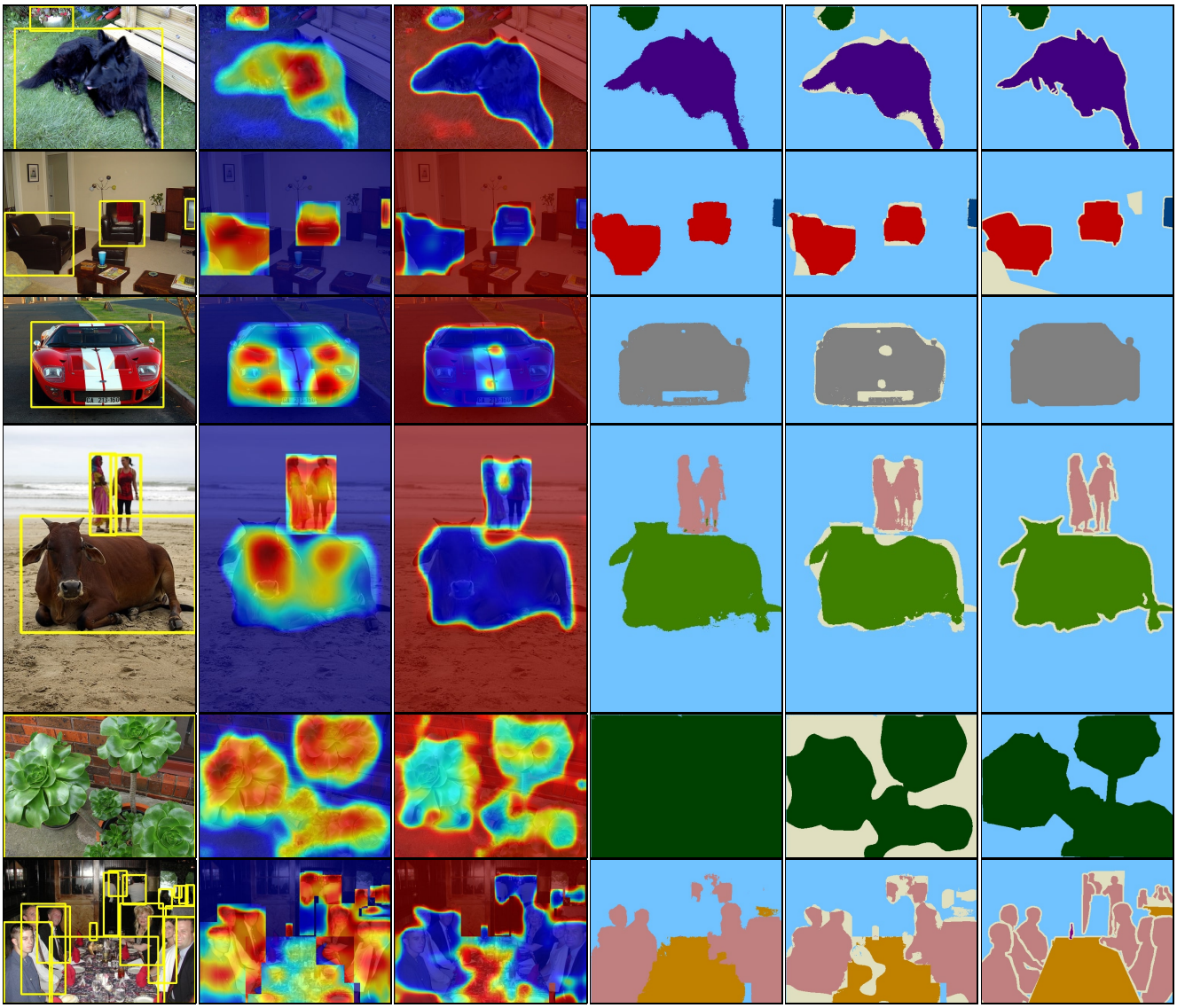


Figure 2: Visual examples of u_c, u_0, and corresponding pseudo segmentation labels on the PASCAL VOC 2012 [3] $train$ set. We overlap u_c over different classes for visualization. Ours: our pseudo labels Y_{crf}. Ours*: the same ones but with an indication of unreliable regions using Y_{ret}. Best viewed in color.

grouping for image segmentation and object proposal generation. *IEEE Trans. PAMI*, 39(1):128–140, 2016. 2

[12] Siyuan Qiao, Chenxi Liu, Wei Shen, and Alan L Yuille. Few-shot image recognition by predicting parameters from activations. In *CVPR*, 2018. 3

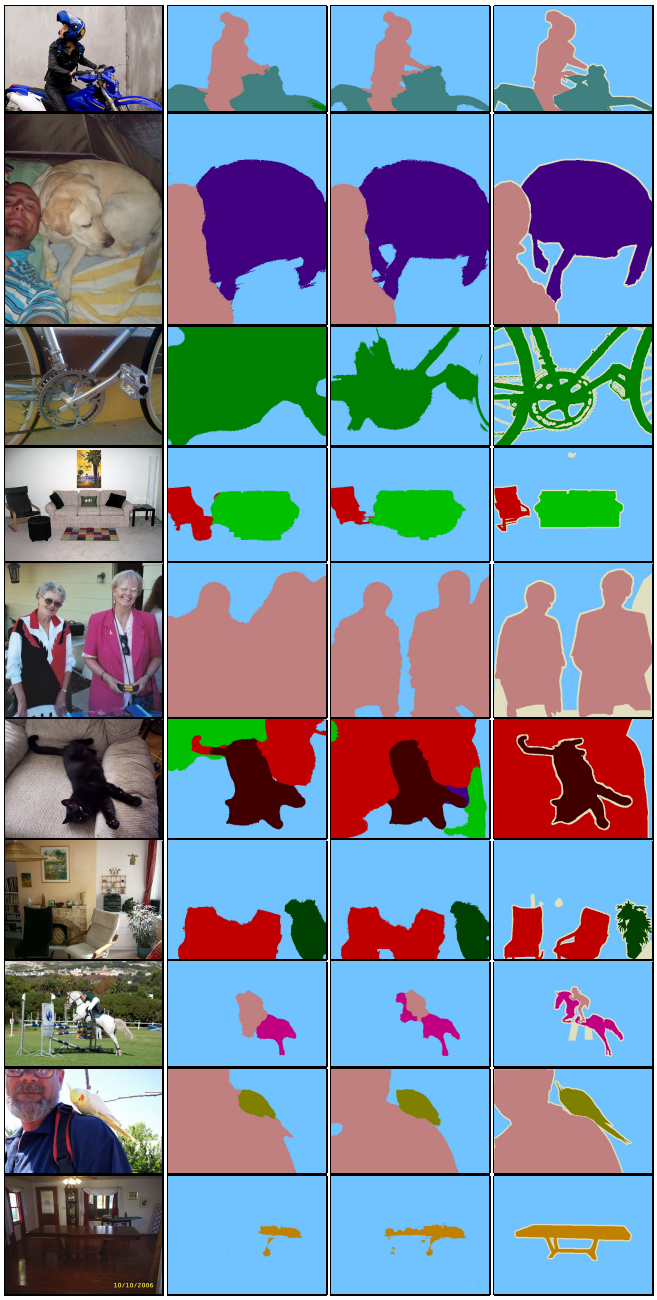
[13] Scott Reed, Honglak Lee, Dragomir Anguelov, Christian Szegedy, Dumitru Erhan, and Andrew Rabinovich. Training deep neural networks on noisy labels with bootstrapping. In *ICLR*, 2015. 3

[14] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. "GrabCut" interactive foreground extraction using iterated graph cuts. In *SIGGRAPH*, 2004. 2

[15] Chunfeng Song, Yan Huang, Wanli Ouyang, and Liang Wang. Box-driven class-wise region masking and filling rate guided loss for weakly supervised semantic segmentation. In *CVPR*, 2019. 1, 2, 3, 4

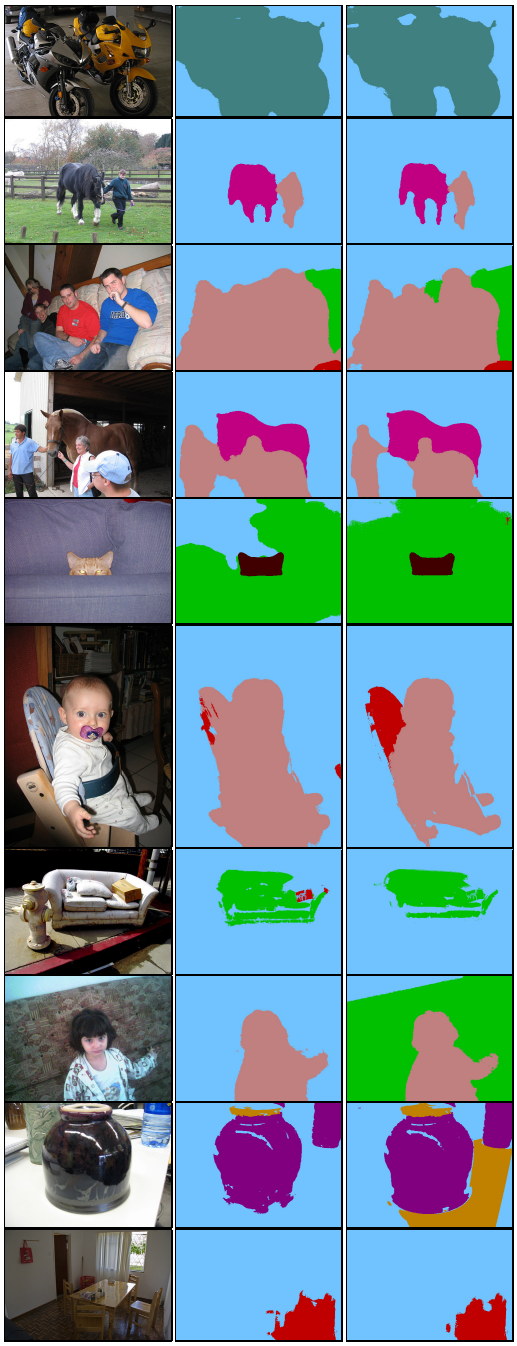
[16] Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. *ICML*, 2020. 3

[17] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *CVPR*, 2016. 1



Input image. Ours (Weak). Ours (Semi). Ground truth.

(a) Results on *val* set.



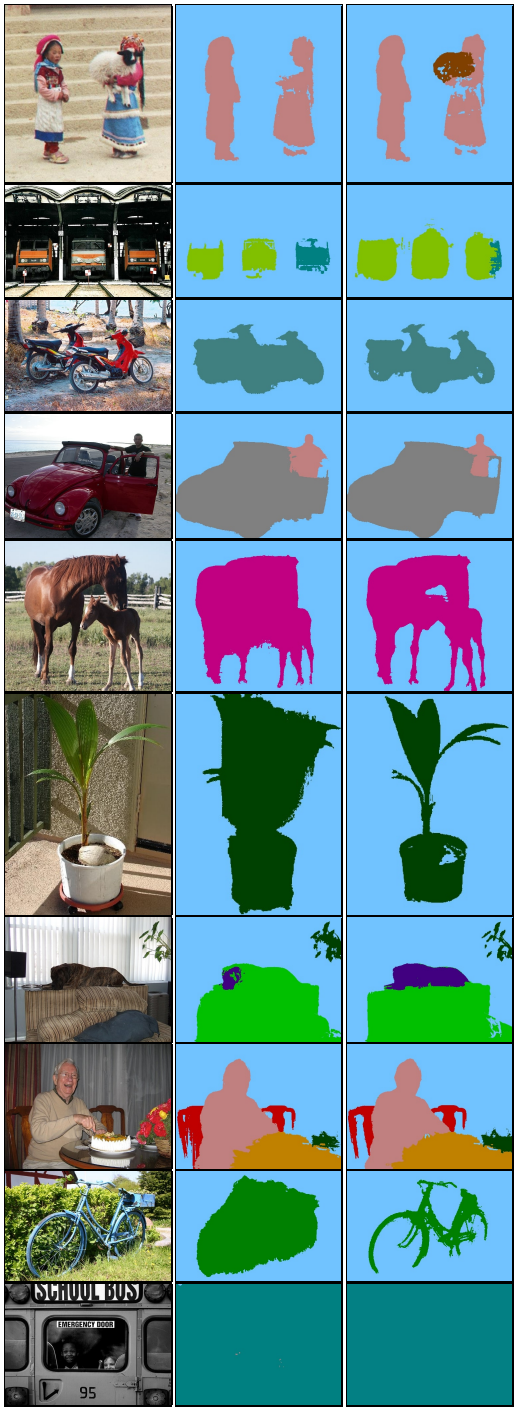
Input image. Ours (Weak). Ours (Semi).

(b) Results on *test* set.

Figure 3: Qualitative examples of our final model using DeepLab-V1 [1, 2] on the PASCAL VOC 2012 [3] dataset. Best viewed in color.

Input image. Ours (Weak). Ours (Semi). Ground truth.

(a) Results on *val* set.



Input image. Ours (Weak). Ours (Semi).

(b) Results on *test* set.

Figure 4: Qualitative examples of our final model using DeepLab-V2 [2] on the PASCAL VOC 2012 [3] dataset. Best viewed in color.

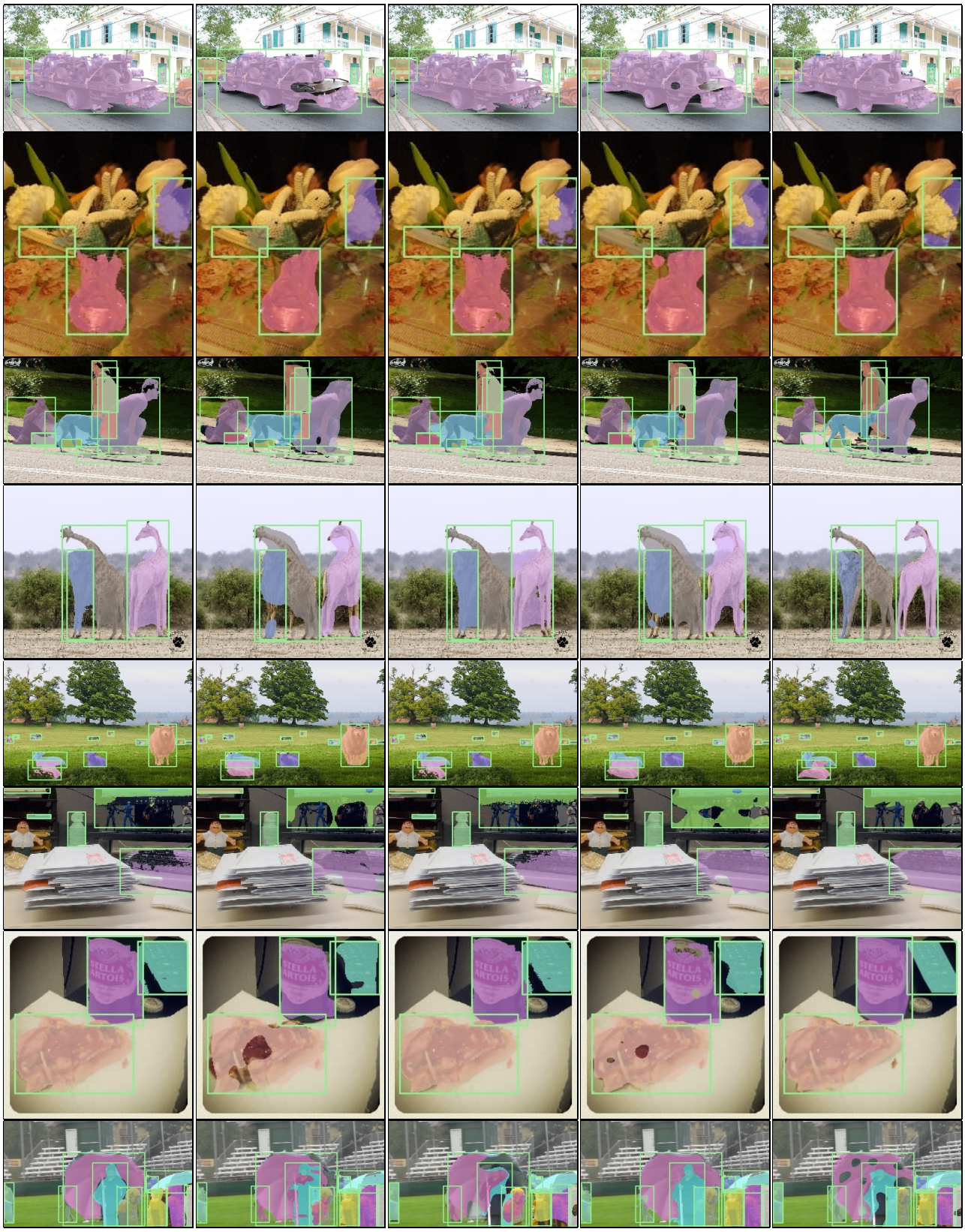
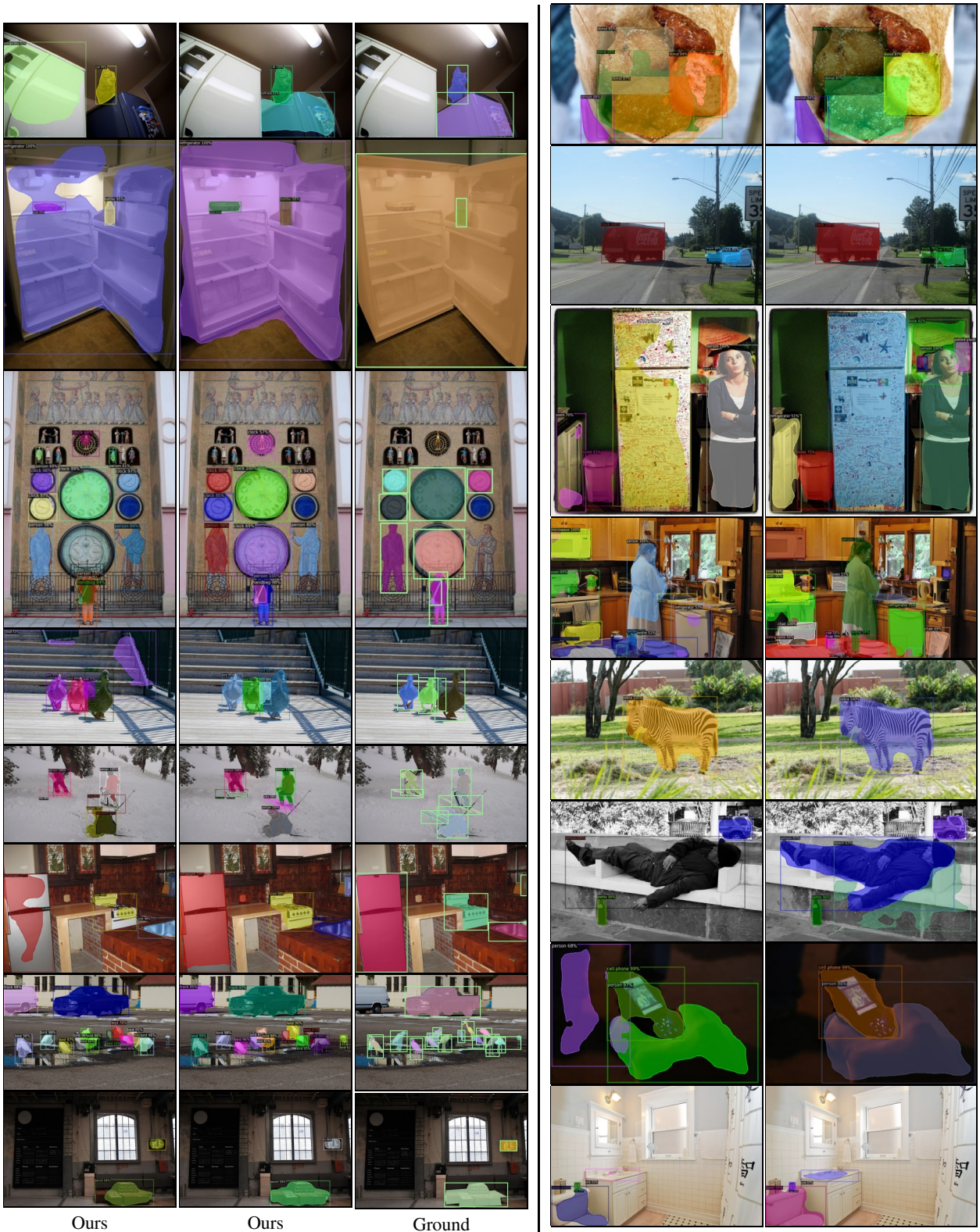


Figure 5: Visual examples of pseudo segmentation labels on the MS-COCO [9] *train* set. Note that VOC-to-COCO do not use any training samples of MS-COCO during training. Best viewed in color.



Ours (VOC-to-COCO). | Ours (COCO-to-COCO). | Ground truth.

(a) Results on *val* set.

Ours (VOC-to-COCO). | Ours (COCO-to-COCO).

(b) Results on *test* set.

Figure 6: Qualitative results of Mask-RCNN [6] on the MS-COCO [9] dataset. We train Mask-RCNN with two pseudo labels Y_{crf} and Y_{ret}, and compute the binary cross-entropy loss for the regions $\mathcal{S}$ only. Best viewed in color.