Weakly Supervised Instance Segmentation by Learning Annotation Consistent Instances - Supplementary Material

Aditya Arun[1], C.V. Jawahar[1], and M. Pawan Kumar[2]

[1] CVIT, KCIS, IIIT Hyderabad
[2] OVAL, University of Oxford

1 Learning Objective

In this section we provide detailed derivation of the objective presented in the section 4.2 of the main paper.

Given the loss function Δ (equation (9) of the main paper) which is tuned for the task of instance segmentation, we compute the diversity terms as given in the equation (7) of the main paper. Recall that the diversity for any two distributions is the expected loss of the samples drawn from the two distributions. For the prediction distribution Pr_p and the conditional distribution Pr_c, we derive the diversity between them and their self diversities as follows.

Diversity between prediction network and conditional network: Following equation (7) of the main paper, the diversity between prediction and conditional distribution can be written as,

$$DIV_{\Delta}(\mathrm{Pr}_p, \mathrm{Pr}_c) = \mathbb{E}_{\mathbf{y}_p \sim \mathrm{Pr}_p(\mathbf{y}|\mathbf{x};\boldsymbol{\theta}_p)}[\mathbb{E}_{\mathbf{y}_c \sim \mathrm{Pr}_c(\mathbf{y}|\mathbf{x},\mathbf{a};\boldsymbol{\theta}_c)}[\Delta(\mathbf{y}_p, \mathbf{y}_c)]]. \tag{1}$$

We then write the expectation with respect to the conditional distribution (the inner distribution) as expectation over the random variables $\mathbf{z}$ with distribution $\mathrm{Pr}(\mathbf{z})$ using Law of the Unconscious Statistician (LOTUS). The expectation over the random variable $\mathbf{z}$ with distribution $\mathrm{Pr}(\mathbf{z})$ is approximated by taking K samples from $\mathrm{Pr}(\mathbf{z})$,

$$DIV_{\Delta}(\mathrm{Pr}_p, \mathrm{Pr}_c) = \mathbb{E}_{\mathbf{y}_p \sim \mathrm{Pr}_p(\mathbf{y}|\mathbf{x};\boldsymbol{\theta}_p)}\Big[\frac{1}{K}\sum_{k=1}^{K} \Delta(\mathbf{y}_p, \mathbf{y}_c^k)\Big]. \tag{2}$$

We finally compute the expectation with respect to the prediction distribution as,

$$DIV_{\Delta}(\mathrm{Pr}_p, \mathrm{Pr}_c) = \frac{1}{K}\sum_{k=1}^{K}\sum_{\mathbf{y}_p^{(i)}} \mathrm{Pr}_p(\mathbf{y}_p^{(i)};\boldsymbol{\theta}_p)\Delta(\mathbf{y}_p^{(i)}, \mathbf{y}_c^k). \tag{3}$$

Self diversity for conditional network: As above, using equation (7) of the main paper, we write the self diversity coefficient of the conditional distribution as

$$DIV_{\Delta}(\Pr_c, \Pr_c) = \mathbb{E}_{\mathbf{y}_c \sim \Pr_c(\mathbf{y}|\mathbf{x},\mathbf{a};\boldsymbol{\theta}_c)}[\mathbb{E}_{\mathbf{y}'_c \sim \Pr_c(\mathbf{y}|\mathbf{x},\mathbf{a};\boldsymbol{\theta}_c)}[\Delta(\mathbf{y}_c, \mathbf{y}'_c)]]. \tag{4}$$

We now write the two expectations with respect to the conditional distribution as the expectation over the random variables $\mathbf{z}$ and $\mathbf{z}'$ respectively. In order to approximate the expectation over the random variables $\mathbf{z}$ and $\mathbf{z}'$, we use K samples from the distribution $\Pr(\mathbf{z})$ as,

$$DIV_{\Delta}(\Pr_c, \Pr_c) = \frac{1}{K}\sum_{k=1}^{K}\frac{1}{K-1}\sum_{\substack{k'=1,\\ k'\neq k}}^{K}\Delta(\mathbf{y}_c^k, \mathbf{y}_c^{k'}). \tag{5}$$

On re-arranging the above equation, we get

$$DIV_{\Delta}(\Pr_c, \Pr_c) = \frac{1}{K(K-1)}\sum_{\substack{k,k'=1\\ k'\neq k}}^{K}\Delta(\mathbf{y}_c^k, \mathbf{y}_c^{k'}). \tag{6}$$

Self diversity for prediction network: Similar to the above two cases, using equation (7) of the main paper, we can write the self diversity of the prediction network as

$$DIV_{\Delta}(\Pr_p, \Pr_p) = \mathbb{E}_{\mathbf{y}_p \sim \Pr_p(\mathbf{y}|\mathbf{x};\boldsymbol{\theta}_p)}[\mathbb{E}_{\mathbf{y}'_p \sim \Pr_p(\mathbf{y}|\mathbf{x};\boldsymbol{\theta}_p)}[\Delta(\mathbf{y}_p, \mathbf{y'}_p)]]. \tag{7}$$

Note that the prediction distribution is a fully factorized distribution, and we can compute its exact expectation. Therefore, we compute the two expectations with respect to the inner prediction distribution as,

$$DIV_{\Delta}(\Pr_p, \Pr_p) = \mathbb{E}_{\mathbf{y}_p \sim \Pr_p(\mathbf{y}|\mathbf{x};\boldsymbol{\theta}_p)}\Big[\sum_{\mathbf{y'}_p^{(i)}}\Pr_p(\mathbf{y'}_p^{(i)};\boldsymbol{\theta}_p)\Delta(\mathbf{y}_p, \mathbf{y'}_p^{(i)})\Big]; \tag{8}$$

and the expectation with respect to the outer prediction distribution as,

$$DIV_{\Delta}(\Pr_p, \Pr_p) = \sum_{\mathbf{y}_p^{(i)}}\sum_{\mathbf{y'}_p^{(i)}}\Pr_p(\mathbf{y}_p^{(i)};\boldsymbol{\theta}_p)\Pr_p(\mathbf{y'}_p^{(i)};\boldsymbol{\theta}_p)\Delta(\mathbf{y}_p^{(i)}, \mathbf{y'}_p^{(i)}). \tag{9}$$

2 Optimization

2.1 Optimization over Prediction Distribution

As parameters $\boldsymbol{\theta}_c$ of the conditional distribution are constant, the learning objective of the prediction distribution is written as,

$$\boldsymbol{\theta}_p^* = \arg\min_{\boldsymbol{\theta}_p} DIV_{\Delta}(\Pr_p, \Pr_c) - (1-\gamma)DIV_{\Delta}(\Pr_p, \Pr_p). \tag{10}$$

This results in a fully supervised training of the Mask R-CNN network [3]. Note that the only difference between training of a standard Mask R-CNN architecture and our prediction network is the use of the dissimilarity objective function (10) above, instead of simply minimizing the multi-task loss of the Mask R-CNN.

The prediction network takes as the input an image and the K predictions sampled from the conditional network. Treating these outputs of the conditional network as the pseudo ground truth label, we compute the gradient of our dissimilarity coefficient based loss function. As the objective (10) above is differentiable with respect to parameters $\boldsymbol{\theta}_p$, we update the network by employing stochastic gradient descent.

2.2 Optimization over Conditional Distribution

Similar to the prediction network, the conditional network is optimized by treating the parameters of the prediction network θ_p as constant. Its learning obective is given as,

$$\boldsymbol{\theta}_c^* = \operatorname*{arg\,min}_{\boldsymbol{\theta}_c} DIV_\Delta(\mathrm{Pr}_p, \mathrm{Pr}_c) - \gamma DIV_\Delta(\mathrm{Pr}_c, \mathrm{Pr}_c). \tag{11}$$

A non-differentiable training procedure: The conditional network is modeled using a Discrete Disco Nets which employs a sampling step from the scoring function $S^k(\mathbf{y}_c)$. This sampling step makes the objective function non-differentiable with respect to the parameters $\boldsymbol{\theta}_c$, even though the scoring function $S^k(\mathbf{y}_c)$ itself is differentiable. However, as the prediction network is fixed, the above objective function reduces to the one used in Bouchacourt *et al.* [2] for fully supervised training. Therefore, similar to Bouchacourt *et al.* [2] we solve this problem by estimating the gradients of our objective function with the help of temperature parameter ϵ as,

$$\nabla_{\boldsymbol{\theta}_c} DISC^\epsilon_\Delta(\mathrm{Pr}_p(\boldsymbol{\theta}_p), \mathrm{Pr}_c(\boldsymbol{\theta}_c)) = \pm \lim_{\epsilon \to 0} \frac{1}{\epsilon}(DIV^\epsilon_\Delta(\mathrm{Pr}_p, \mathrm{Pr}_c) - \gamma DIV^\epsilon_\Delta(\mathrm{Pr}_c, \mathrm{Pr}_c)) \tag{12}$$

where,

$$DIV^\epsilon_\Delta(\mathrm{Pr}_p, \mathrm{Pr}_c) = \mathbb{E}_{\mathbf{y}_p \sim \mathrm{Pr}_p(\boldsymbol{\theta}_p)} \left[\mathbb{E}_{\mathbf{z}^k \sim \mathrm{Pr}(\mathbf{z})} [\nabla_{\boldsymbol{\theta}_c} S^k(\hat{\mathbf{y}}_a) - \nabla_{\boldsymbol{\theta}_c} S^k(\hat{\mathbf{y}}_c)] \right], \tag{13}$$

$$DIV^\epsilon_\Delta(\mathrm{Pr}_c, \mathrm{Pr}_c) = \mathbb{E}_{\mathbf{z}^k \sim \mathrm{Pr}(\mathbf{z})} \left[\mathbb{E}_{\mathbf{z}^{k'} \sim \mathrm{Pr}(\mathbf{z})} [\nabla_{\boldsymbol{\theta}_c} S^k(\hat{\mathbf{y}}_b) - \nabla_{\boldsymbol{\theta}_c} S^{k'}(\hat{\mathbf{y}}'_c)] \right], \tag{14}$$

and,

$$\begin{aligned}
\hat{\mathbf{y}}_c &= \operatorname*{arg\,max}_{\mathbf{y} \in \mathcal{Y}} S^k(\mathbf{y}_c) \\
\hat{\mathbf{y}}'_c &= \operatorname*{arg\,max}_{\mathbf{y} \in \mathcal{Y}} S^{k'}(\mathbf{y}_c) \\
\hat{\mathbf{y}}_a &= \operatorname*{arg\,max}_{\mathbf{y} \in \mathcal{Y}} S^k(\mathbf{y}_c) \pm \epsilon \Delta(\mathbf{y}_p, \hat{\mathbf{y}}_c) \\
\hat{\mathbf{y}}_b &= \operatorname*{arg\,max}_{\mathbf{y} \in \mathcal{Y}} S^k(\mathbf{y}_c) \pm \epsilon \Delta(\hat{\mathbf{y}}_c, \hat{\mathbf{y}}'_c)
\end{aligned} \tag{15}$$

In our experiments, we fix the temperature parameter ϵ as, $\epsilon = +1$.

Intuition for the gradient computation: We now present an intuitive explanation of the computation of gradient, as given in equation (12). For an input $\mathbf{x}$ and two noise samples $\mathbf{z}^k, \mathbf{z}^{k'}$, the conditional network outputs two scores $S^k(\mathbf{y}_c)$ and $S^{k'}(\mathbf{y}_c)$, with the corresponding maximum scoring outputs $\hat{\mathbf{y}}_c$ and $\hat{\mathbf{y}}'_c$. The model parameters $\boldsymbol{\theta}_c$ are updated via gradient descent in the negative direction of $\nabla_{\boldsymbol{\theta}_c} DISC^{\epsilon}_{\Delta}(\Pr_p(\boldsymbol{\theta}_p), \Pr_c(\boldsymbol{\theta}_c))$.

- The term $DIV^{\epsilon}_{\Delta}(\Pr_p, \Pr_c)$ updates the model parameters towards the maximum scoring prediction $\hat{\mathbf{y}}_c$ of the score $S^k(\mathbf{y}_c)$ while moving away from $\hat{\mathbf{y}}_a$, where $\hat{\mathbf{y}}_a$ is the sample corresponding to the maximum loss augmented score $S^k(\mathbf{y}_c) \pm \epsilon\Delta(\mathbf{y}_p, \hat{\mathbf{y}}_c)$ with respect to the fixed prediction distribution samples $\mathbf{y}_p$. This encourages the model to move away from the prediction providing high loss with respect to the pseudo ground truth labels.
- The term $\gamma DIV^{\epsilon}_{\Delta}(\Pr_c, \Pr_c)$ updates the model towards $\hat{\mathbf{y}}_b$ and away from the $\hat{\mathbf{y}}_c$. Note the two negative signs giving the update in the positive direction. Here $\hat{\mathbf{y}}_b$ is the sample corresponding to the maximum loss augmented score $S^k(\mathbf{y}_c) \pm \epsilon\Delta(\hat{\mathbf{y}}_c, \hat{\mathbf{y}}'_c)$ with respect to the other prediction $\hat{\mathbf{y}}'_c$, encouraging diversity between $\hat{\mathbf{y}}_c$ and $\hat{\mathbf{y}}'_c$.

Training algorithm for conditional network: Pseudo-code for training the conditional network for a single sample from weakly supervised data is presented in algorithm 1 below. In algorithm 1, statements 1 to 3 describe the sampling process and computing the loss augmented prediction. We first sample K different predictions $\hat{\mathbf{y}}^k_c$ corresponding to each noise vector $\mathbf{z}^k$ in statement 2. For the sampled prediction $\hat{\mathbf{y}}^k_c$ we compute the maximum loss augmented score $S^k(\mathbf{y}_c) \pm \epsilon\Delta(\mathbf{y}_p, \hat{\mathbf{y}}_c)$. This is then used to find the loss augmented prediction $\hat{\mathbf{y}}_a$ given in statement 3.

In order to compute the gradients of the self diversity of conditional distribution, we need to find the maximum loss augmented prediction $\hat{\mathbf{y}}_b$. Here, the loss is computed between a pair of K different predictions of the conditional network that we have already obtained. This is shown by statements 4 to 7 in algorithm 1.

For the purpose of optimizing the conditional network using gradient descent, we need to find the gradients for the objective function of the conditional network defined in equation (11) above. The computation of the unbiased approximate gradients for the individual terms in the objective function is shown in statement 8. We finally optimize the conditional network by the employing gradient descent step and updating the model parameters by descending to the approximated gradients as shown in statement 9 of algorithm 1.

3 Experiments

In this section we present the implementation details of the prediction and the conditional network presented in the main paper. Next, we present details of our

Algorithm 1: *Conditional network training algorithm*

```
Input  : Training input $(x, a) \in \mathcal{W}$, and prediction network outputs $y_p$
Output: $\hat{\mathbf{y}}_c^1, \ldots, \hat{\mathbf{y}}_c^K$, sample $K$ predictions from the model
1 for $k = 1 \ldots K$ do
2     Sample noise vector $\mathbf{z}^k$, generate output $\hat{\mathbf{y}}_c^k$:
          $\hat{\mathbf{y}}_c^k = \arg\max_{\mathbf{y} \in \mathcal{Y}} S^k(\mathbf{y}_c)$
3     Find loss augmented prediction $\hat{\mathbf{y}}_a^k$ w.r.t. output from prediction network
      $\mathbf{y_P}$:
          $\hat{\mathbf{y}}_a^k = \arg\max_{\mathbf{y} \in \mathcal{Y}} S^k(\mathbf{y}_c) \pm \epsilon\Delta(\mathbf{y}_p, \hat{\mathbf{y}}_c^k)$
4 Compute loss augmented predictions:
5 for $k = 1, \ldots, K$ do
6     for $k' = 1, \ldots, K, k' \neq k$ do
7         Find loss augmented prediction $\hat{\mathbf{y}}_b^k$ w.r.t. other conditional network
          outputs $\hat{\mathbf{y}}_c^k$:
              $\hat{\mathbf{y}}_b^{k,k'} = \arg\max_{\mathbf{y} \in \mathcal{Y}} S^k(\mathbf{y}_c) \pm \epsilon\Delta(\hat{\mathbf{y}}_c^k, \hat{\mathbf{y}}_c')$
8 Compute unbiased approximate gradients for $DIV_\Delta^\epsilon(\mathrm{Pr}_c, \mathrm{Pr}_c)$ and
  $DIV_\Delta^\epsilon(\mathrm{Pr}_c, \mathrm{Pr}_c)$ as:
      $DIV_\Delta^\epsilon(\mathrm{Pr}_p, \mathrm{Pr}_c) = \frac{1}{K} \sum_{k=1}^{K} \Big[\nabla_{\boldsymbol{\theta}_c} S^k(\hat{\mathbf{y}}_a) - \nabla_{\boldsymbol{\theta}_c} S^k(\hat{\mathbf{y}}_c)\Big],$
      $DIV_\Delta^\epsilon(\mathrm{Pr}_c, \mathrm{Pr}_c) = \frac{2}{K(K-1)} \sum_{\substack{k,k'=1 \\ k' \neq k}}^{K} \Big[\nabla_{\boldsymbol{\theta}_c} S^k(\hat{\mathbf{y}}_b) - \nabla_{\boldsymbol{\theta}_c} S^{k'}(\hat{\mathbf{y}}_c')\Big].$
  Update model parameters by descending to the approximated gradients:
      $\boldsymbol{\theta}_c^{t+1} = \boldsymbol{\theta}_c^t - \eta \nabla_{\boldsymbol{\theta}_c} DISC_\Delta(\mathrm{Pr}_p(\boldsymbol{\theta}_p), \mathrm{Pr}_c(\boldsymbol{\theta}_c))$
```

ResNet based architecture and the detailed class-specific results on the Pascal VOC 2012 data set.

3.1 Implementation Details

We use the standard Mask R-CNN as the prediction network and adapt the U-Net architecture for the conditional network, as shown in figure 1 of the main paper. For a fair comparison, the prediction network, we use ImageNet pre-trained ResNet-50 architecture for experiments with image-level annotation and a pretrained ResNet-101 architecture for the bounding box annotations.

Similar to [4], the U-Net architecture is modified by adding a noise sample as an extra channel after the deconvolutional layers as shown in figure 1 of the main paper. A 1×1 convolution is applied to bring the number of channels back to the original dimensions (32 channels). The segmentation region proposal masks taken from MCG [1] is then multiplied element-wise with the features from all the channels. This allows us to extract features only from the segmentation proposal. A 1×1 convolution is applied again to make the number of channels equal to the number of classes. This is followed by a global average pooling layer which gives us, for each of the segmentation proposals, a vector of dimensions equal to the number of classes. This vector for each of the segmentation proposal is passed to the inference algorithm which in turn provides the output segmentation masks corresponding to the image-level annotations. For all our experiments we choose K=10 for the conditional network and use the Adam optimizer. For all the other hyper-parameters we use the same configuration as described in [4]. For the prediction network, we use default hyper-parameters described in [3].

3.2 ResNet based architecture for the conditional network

In section 6.4 of the main paper, we study the effect of an alternative architecture for the conditional network. In what follows, we provide the details of this ResNet based conditional network.

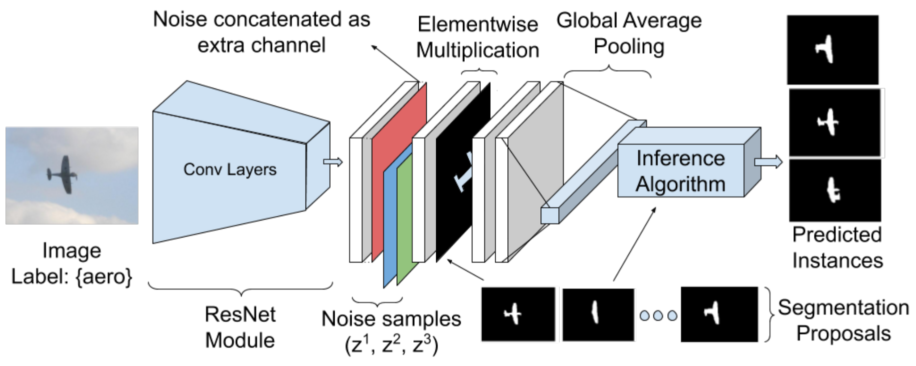

Fig. 1. *ResNet based conditional network*

The architecture for the ResNet based conditional network is shown in figure 1. The image is first passed through the ResNet module to obtain low-resolution high-level features. For the experiments where we use only the image-level annotations, a ResNet-50 module is employed and where we use the bounding-box level annotations, a ResNet-101 module is used. A noise filter is appended as an extra channel followed by a 1×1 convolutional filter, which brings the number of channels back to the original dimensions. The segmentation proposal masks are then multiplied element-wise to obtain segmentation proposal specific

Table 1. Per class result for $\text{mAP}^r_{0.5}$ metric on Pascal VOC 2012 data set for methods that are trained on using image-level supervision $\mathcal{I}$ and bounding box annotations $\mathcal{B}$

Method	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	pson	plant	sheep	sofa	train	tv	mAP
Ours (ResNet-50) $\mathcal{I}$	74.2	52.6	68.6	44.1	25.0	63.4	35.9	72.6	18.2	47.1	24.6	63.5	53.7	67.3	40.9	29.4	42.8	39.6	69.5	61.2	49.7
Ours $\mathcal{I}$	75.5	53.6	69.9	45.3	26.7	64.3	37.4	73.7	19.3	48.7	25.3	64.6	55.0	68.3	42.1	30.8	44.2	40.5	70.6	62.2	50.9
Ours (ResNet-101) $\mathcal{B}$	77.9	62.6	73.8	49.0	35.9	72.6	45.8	78.4	29.7	55.7	31.9	70.6	61.3	73.6	49.2	39.9	50.8	47.9	76.5	69.6	57.7
Ours $\mathcal{B}$	79.1	63.9	75.1	49.3	36.5	73.1	46.4	78.8	30.1	56.4	32.1	71.3	61.6	74.8	49.5	40.2	51.1	48.3	77.2	69.9	58.2

features. Next, a 1×1 convolutional is applied to make the number of channels equal to the number of classes. Finally, a global average pooling is applied to obtain a vector whose dimensions is equal to the number of classes in the data set. This vector is then passed through the inference algorithm to obtain the final predicted samples. As mentioned in section 6.4 of the paper, the results obtained using this ResNet based conditional network architecture are called as Ours (ResNet-50) and Ours (ResNet-101).

Note that, the U-Net based conditional network provides a higher resolution image features as compared to its ResNet based counterparts. These are then used to obtain the individual features of the segmentation mask proposals. The higher resolution features thus provide richer per-mask features. These are especially useful for smaller objects and cluttered environment where context resolution is important. The superior results of our method when using a U-Net based conditional network empirically verify this claim.

3.3 Class specific results on VOC 2012 data set

We present the per-class result for our method on the Pascal VOC 2012 data set in table 1. The first two rows correspond to the result where our method was trained only using the image-level annotations. The last two rows correspond to the results where our methods were trained using the bounding box annotations.The qualitative results for each class is presented in figure 2.

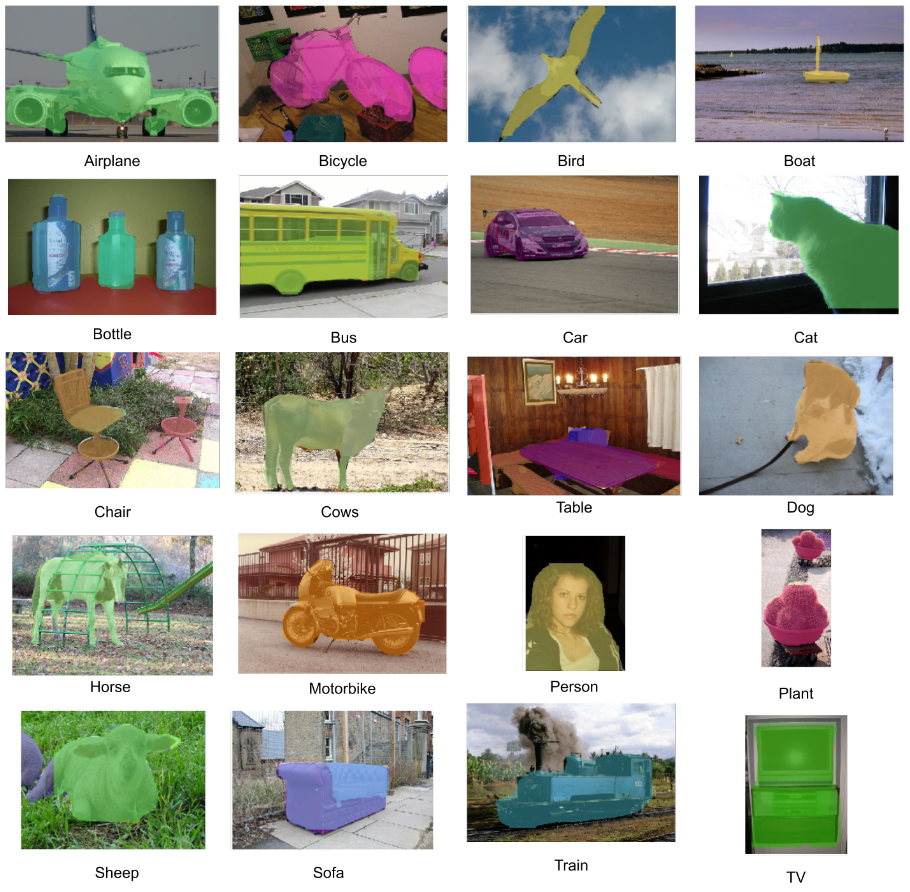

Fig. 2. *Qualitative results of our proposed approach on VOC 2012 validation set.*

References

1. Arbeláez, P., Pont-Tuset, J., Barron, J., Marques, F., Malik, J.: Multiscale combinatorial grouping. In: CVPR (2014)
2. Bouchacourt, D.: Task-Oriented Learning of Structured Probability Distributions. Ph.D. thesis, University of Oxford (2017)
3. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask R-CNN. In: ICCV (2017)
4. Kohl, S., Romera-Paredes, B., Meyer, C., De Fauw, J., Ledsam, J.R., Maier-Hein, K., Eslami, S.A., Rezende, D.J., Ronneberger, O.: A probabilistic u-net for segmentation of ambiguous images. In: NIPS (2018)