Guided Collaborative Training for Pixel-wise Semi-Supervised Learning

Supplementary Material

Zhanghan Ke[1,2], Di Qiu[2], Kaican Li[2], Qiong Yan[2], and Rynson W.H. Lau[1]

[1] Department of Computer Science, City University of Hong Kong
kezhanghan@outlook.com, rynson.lau@cityu.edu.hk
[2] SenseTime Research
{kezhanghan,qiudi,likaican,yanqiong}@sensetime.com

Appendix A: Algorithm of C

In GCT, we use a classical image processing pipeline C to calculate the ground truth of the flaw detector F on the labeled subset by taking the task model prediction $T^k(x_l)$ and the corresponding label y as the input. C is composed of three operations:

1. $blur(inp,\ (height, width))$: Blur inp by a Gaussian kernel of given shape.
2. $dilate(inp,\ (height, width))$: Dilate inp for each local region of given shape.
3. $norm(inp)$: Normalize all pixels in inp to range between [0, 1].

We show the pseudo code of C in Python style as follows (assume the shape of $T^k(x_l)$ is $H \times W \times O$):

Algorithm 1 Image Process Pipeline C.

Require: Channel average coefficient μ ; Operations repeat times ν.

def $C(T^k(x_l),\ y)$:
    $F_{gt} = \mu \sum_o |T^k(x_l)^{(h,w,o)} - y^{(h,w,o)}|$
    $F_{gt} = blur(F_{gt},\ (\frac{H}{8}, \frac{W}{8}))$
    for i **in** $range(0,\ \nu)$:
        $F_{gt} = dilate(F_{gt},\ (3, 3))$
        $F_{gt} = blur(F_{gt},\ (\frac{H}{4}, \frac{W}{4}))$
    $F_{gt} = norm(F_{gt})$
    return F_{gt}

In our experiments, we set $\mu = \frac{1}{2}$ for semantic segmentation, and we set $\mu = \frac{1}{o}$ for other three tasks. We set $\nu = 10$ for real image denoising, $\nu = 5$ for night image enhancement, and $\nu = 1$ for other two tasks.

Appendix B: Architecture of Flaw Detector

The flaw detector F is a fully-convolutional neural network, which contains 8 convolutional layers with 4×4 kernels. The amount of kernels is increased from 64 to 512 in the first 7 layers and then decreased to 1 in the last layer. Each of the first 7 convolutional layers is followed by batch normalization [2] and leaky ReLU [4] with threshold of 0.2. The convolutional layers with stride=2 reduce the resolution of the feature maps. At the end of F, we add a bilinear interpolation operation to rescale the output to the size of the input. In all experiments of GCT, we optimize F by Adam [3] (with learning rate $1e^{-4}$). The architecture of F is as follow:

Layer	**Details**
Input	concatenate $T^k(x)$ and x as the input
Conv + BN + ReLU	out-channels=64, kernel-size=4, stride=2, padding=*same*
Conv + BN + ReLU	out-channels=128, kernel-size=4, stride=2, padding=*same*
Conv + BN + ReLU	out-channels=128, kernel-size=4, stride=1, padding=*same*
Conv + BN + ReLU	out-channels=256, kernel-size=4, stride=2, padding=*same*
Conv + BN + ReLU	out-channels=256, kernel-size=4, stride=1, padding=*same*
Conv + BN + ReLU	out-channels=512, kernel-size=4, stride=2, padding=*same*
Conv + BN + ReLU	out-channels=512, kernel-size=4, stride=1, padding=*same*
Conv	out-channels=1, kernel-size=4, stride=2, padding=*same*
Interpolation	out-shape=$H \times W$, mode=*bilinear*, align-corners=$True$

Appendix C: Training Details

We have experimented with several SSL methods, including (1) the consistent-based Mean Teacher (MT) [5]; (2) the self-supervised SSL (S4L) [6]; (3) the adversarial-based method proposed in [1] (AdvSSL); (4) the GCT framework proposed by us. Here are the definitions of the hyper-parameters for SSL in these methods:

Methods	**Hyper-Parameters for SSL**
MT [5]	λ_{MT} - coefficient for scaling the consistency constraint η_{MT} - epochs for ramping up the consistency constraint α_{MT} - moving average coefficient for ensembling the teacher model
S4L [6]	λ_{S4L} - coefficient for scaling the unsupervised rotation constraint
AdvSSL [1]	λ^l_{Adv} - coefficient for scaling the labeled adversarial constraint λ^u_{Adv} - coefficient for scaling the unlabeled adversarial constraint
GCT (Our)	λ_{fc} - coefficient for scaling the flaw correction constraint λ_{dc} - coefficient for scaling the dynamic consistency constraint η_{dc} - epochs for ramping up the dynamic consistency constraint ξ - flaw threshold for calculating the dynamic consistency constraint and combining the two SSL constraints

For the four validated tasks, we use grid search to find the suitable hyper-parameters for SSL. The final settings for the experiments are as follows:

Methods		Semantic Segmentation	Real Image Denoising	Portrait Image Matting	Night Image Enhancement
MT [5]	λ_{MT}	1.00	1.00	1.00	1.00
	η_{MT}	3	5	5	5
	α_{MT}	0.99	0.99	0.99	0.99
S4L [6]	λ_{S4L}	0.10	1.00	1.00	1.00
AdvSSL [1]	λ^{l}_{Adv}	0.01	0.001	0.01	0.001
	λ^{u}_{Adv}	0.001	0.001	0.01	0.001
GCT (Our)	λ_{fc}	1.00	0.10	1.00	0.10
	λ_{dc}	100	1.00	100	1.00
	η_{dc}	3	5	5	5
	ξ	0.40	0.60	0.40	0.60

Appendix D: Visual Comparisons

Here we provide visual comparisons of the SSL results for four validated tasks. The red bounding box in the figure highlights some main differences in the outputs. As shown below, GCT surpasses existing SSL methods in visual effects.

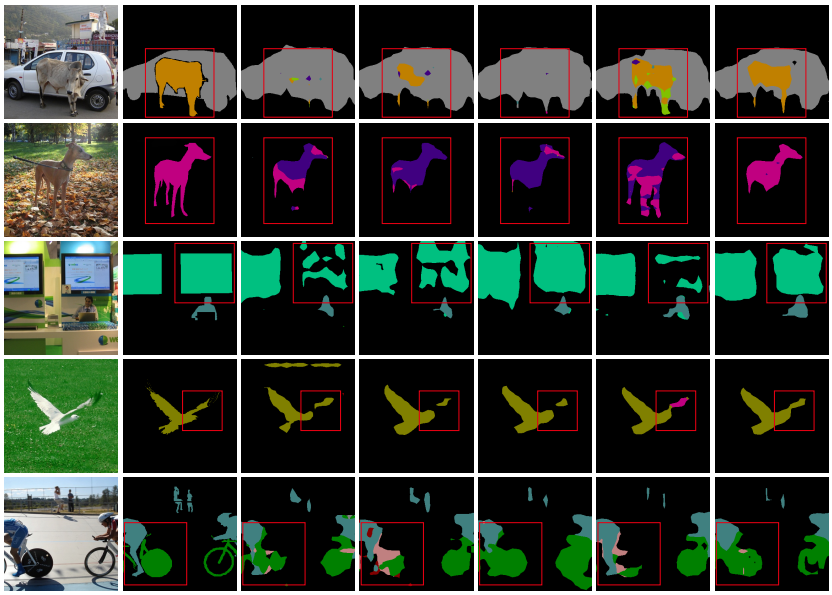

Fig. 1. Semantic Segmentation. Comparisons on the PASCAL VOC dataset using 1/8 labeled data.

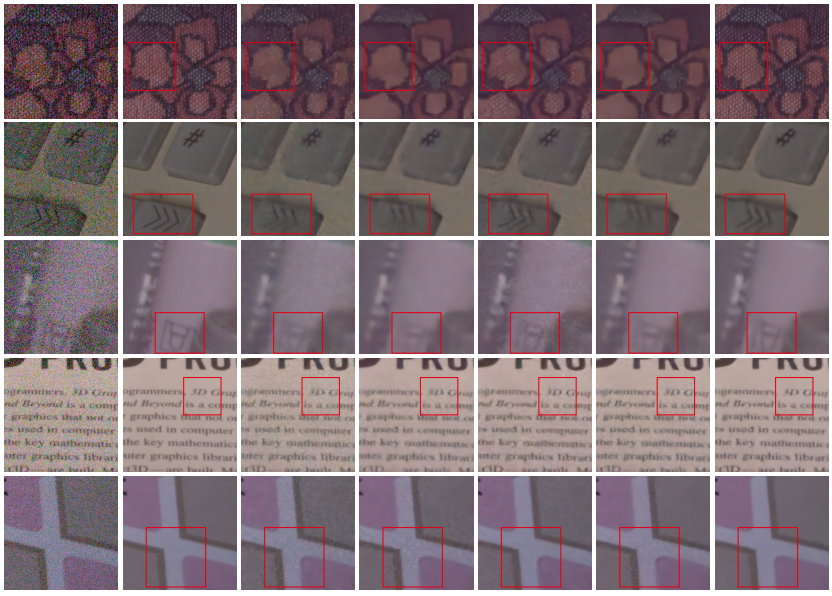

Fig. 2. Real Image Denoising. Comparisons on the SIDD dataset using 1/8 labeled data.

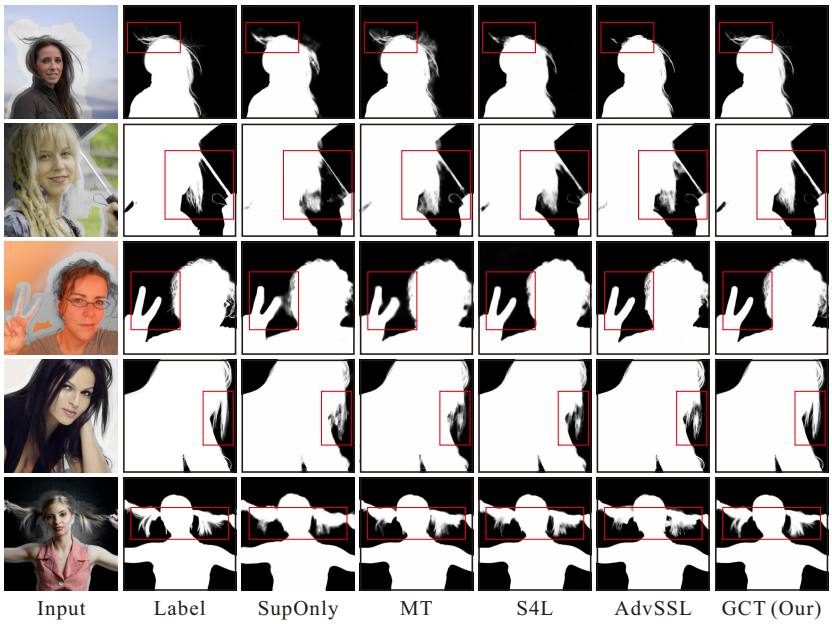

Fig. 3. Portrait Image Matting. Comparisons on our dataset using 100 labeled data and 3850 unlabeled data.

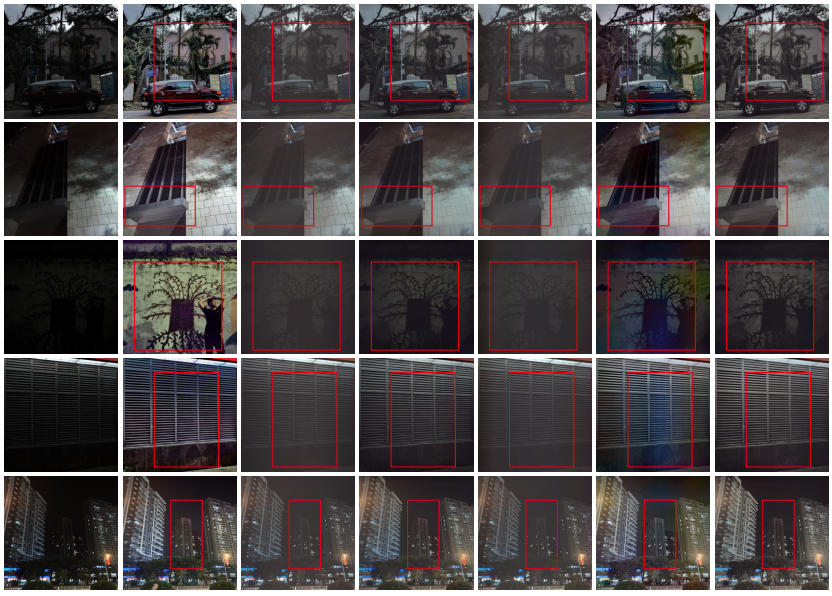

Fig. 4. Night Image Enhancement. Comparisons on our dataset using 200 labeled data and 1500 unlabeled data.

References

1. Hung, W.C., Tsai, Y.H., Liou, Y.T., Lin, Y.Y., Yang, M.H.: Adversarial learning for semi-supervised semantic segmentation. In: BMVC (2018)
2. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: ICML (2015)
3. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: ICLR (2014)
4. Maas, A.L., Hannun, A.Y., Ng, A.Y.: Rectifier nonlinearities improve neural network acoustic models. In: ICML (2013)
5. Tarvainen, A., Valpola, H.: Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In: NeurIPS (2017)
6. Zhai, X., Oliver, A., Kolesnikov, A., Beyer, L.: S4l: Self-supervised semi-supervised learning. In: ICCV (2019)