Guided Collaborative Training for Pixel-wise Semi-Supervised Learning

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

Abstract. We investigate the generalization of semi-supervised learning (SSL) to diverse pixel-wise tasks. Although SSL methods have achieved impressive results in image classification, the performances of applying them to pixel-wise tasks are unsatisfactory due to their need for dense outputs. In addition, existing pixel-wise SSL approaches are only suitable for certain tasks as they usually require to use task-specific properties. In this paper, we present a new SSL framework, named Guided Collaborative Training (GCT), for pixel-wise tasks, with two main technical contributions. First, GCT addresses the issues caused by the dense outputs through a novel flaw detector. Second, the modules in GCT learn from unlabeled data collaboratively through two newly proposed constraints that are independent of task-specific properties. As a result, GCT can be applied to a wide range of pixel-wise tasks without structural adaptation. Our extensive experiments on four challenging vision tasks, including semantic segmentation, real image denoising, portrait image matting, and night image enhancement, show that GCT outperforms state-of-the-art SSL methods by a large margin. Our code available at: https://github.com/ZHKKKe/PixelSSL[i].

Keywords: Semi-Supervised Learning · Pixel-wise Vision Tasks

1 Introduction

Deep learning has been remarkably successful in many vision tasks. Nonetheless, collecting a large amount of labeled data for training is costly, especially for pixel-wise tasks that require a precise label for each pixel, *e.g.*, the category mask in semantic segmentation and the clean picture in image denoising. Recently, semi-supervised learning (SSL) has become an important research direction to alleviate the lack of labels, by appending unlabeled data for training. Many SSL methods have been proposed for image classification with impressive results, including adversarial-based methods [11,27,42,46], consistent-based methods [22,

[i] We implemented PixelSSL, a semi-supervised learning codebase for pixel-wise tasks.

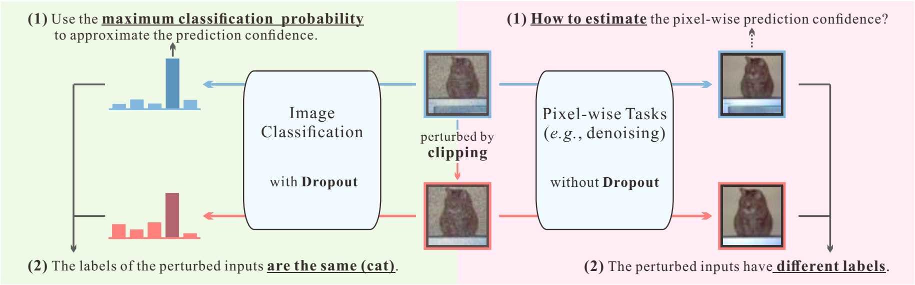

(1) Use the maximum classification probability to approximate the prediction confidence.
Image Classification
with Dropout
perturbed by clipping
(1) How to estimate the pixel-wise prediction confidence?
Pixel-wise Tasks (e.g., denoising)
without Dropout
(2) The labels of the perturbed inputs are the same (cat).
(2) The perturbed inputs have different labels.

Fig. 1. Difficulties of Pixel-wise SSL. The dense outputs in pixel-wise tasks causes unsatisfactory SSL performances since (1) it is difficult to estimate the pixel-wise prediction confidence and (2) existing perturbations designed for SSL are not suitable for dense outputs.

25,39,44], and methods that are combined with self-supervised learning [45,49]. In contrast, only a few works have applied SSL to specific pixel-wise tasks [7,19, 21,33], and they mainly focus on semantic segmentation.

In this work, we investigate the generalization of SSL to diverse pixel-wise tasks. Such generalization is important in order for SSL to be used in new vision tasks with minimal efforts. However, generalizing existing pixel-wise SSL methods is not straightforward since they are designed for certain tasks by using task-specific properties (Sec. 2.2), *e.g.*, assuming similar semantic contents between the input and output. Another possible generalization approach is to apply SSL methods designed for image classification to pixel-wise tasks. But there are two critical issues caused by the dense outputs, as illustrated in Fig. 1, leading to unsatisfactory performances of these methods on pixel-wise tasks.

First, dense outputs require pixel-wise prediction confidences (Sec. 2.3), which are difficult to estimate. Pixel-wise tasks are either pixel-wise classification (*e.g.*, semantic segmentation and shadow detection) or pixel-wise regression (*e.g.*, image denoising and matting). Although we may use the maximum classification probability to represent the prediction confidence in pixel-wise classification, it is unavailable in pixel-wise regression. Second, existing perturbations designed for SSL (Sec. 2.4) are not suitable for dense outputs. In pixel-wise tasks, strong perturbations in the input, *e.g.*, clipping in Mean Teacher [44], will change the input image and its labels. As a result, the perturbed inputs from the same original image have different labels, which is undesirable in SSL. Besides, the perturbations through Dropout [43] are disabled in most pixel-wise tasks. Although Dual Student [22] proposes to create perturbations through different model initializations, its training strategy can only be used in image classification.

To address the above two issues caused by dense outputs, we propose a new SSL framework, named Guided Collaborative Training (GCT), for pixel-wise tasks. It includes three modules – two models for the specific task (the task models) and a novel flaw detector. GCT overcomes the two issues by: (1) approximating the pixel-wise prediction confidence by the output of the flaw detector, *i.e.*, a flaw probability map, and (2) extending the perturbations used in Dual Student to pixel-wise tasks. Since different model initializations lead to inconsistent predictions for the same input, we can ensemble the reliable pixels, *i.e.*, the pixels with lower flaw probabilities, in the predictions. In addition, minimizing the flaw probability map should help correct the unreliable pixels in the predictions. Motivated by these ideas, we introduce two SSL constraints, a dynamic consistency constraint between the task models and a flaw correction constraint between the flaw detector and each of the task models, to allow the modules in GCT to learn from unlabeled data collaboratively under the guidance of the flaw probability map rather than the task-specific properties. As a result, GCT can be applied to diverse pixel-wise tasks, simply by replacing the task models without structural adaptations.

We evaluate GCT on the standard benchmarks for semantic segmentation (pixel-wise classification) and real image denoising (pixel-wise regression). We also conduct experiments on our own practical datasets, *i.e.*, the datasets with a large proportion of unlabeled data, for portrait image matting and night image enhancement (both are pixel-wise regression) to demonstrate the generalization of GCT on real applications. GCT surpasses start-of-the-art SSL methods [19, 44,49] that can be applied to these four challenging pixel-wise tasks. We envision that this work will contribute to future research and development of new vision tasks with scarce labels.

2 Related Work

2.1 SSL for Image Classification

Our work is related to two main branches of SSL methods designed for image classification. The adversarial-based methods [11,27,42,46] assemble the discriminator from GAN [14], and try to match the latent distributions between labeled and unlabeled data through the image-level adversarial constraint. The consistent-based methods [22,25,39,44] learn from unlabeled data by applying a consistency constraint to the predictions under different perturbations. Apart from them, some latest works combine self-supervised learning with SSL [45,49] or expand the training set by interpolating labeled and unlabeled data [5,6].

2.2 SSL for Pixel-wise Tasks

Existing research on pixel-wise SSL mainly focuses on semantic segmentation. GANs dominate in this topic through the combination with the SSL methods derived from image classification. For example, Hung *et al.* [19] extract reliable predictions to generate pseudo labels for training. Mittal *et al.* [33] modify Mean Teacher [44] to a multi-label classifier and use it as a filter to remove uncertain categories. Besides, Lee *et al.* [18] and Huang *et al.* [26] study weak-supervised learning in the SSL context. However, these works require pre-defined categories, which is a general property of classification-based tasks. Chen *et al.* [7] apply

SSL in face sketch synthesis, which belongs to pixel-wise regression. It regards the pre-trained VGG [41] network as a feature extractor to impose a perceptual constraint on the unlabeled data. Unfortunately, the perceptual constraint can only be used in tasks that have similar semantic contents between the inputs and outputs. For example, it does not work on segmentation since the semantic content of the category mask is different from the input image.

2.3 Prediction Confidence in SSL

Prediction confidence is necessary for computing the SSL constraints, which consider the predictions with higher confidence values as the targets, *i.e.*, pseudo labels. Earlier works show that the averaged targets are more confident. For example, Temporal Model [25] accumulates the predictions over epochs as the targets; Mean Teacher [44] defines an explicit model by exponential moving average to generate the targets; FastSWA [4] further averages the models between epochs to produce better targets. Others [27,30,42] regard the maximum classification probability as the prediction confidence.

In pixel-wise SSL, the outputs of the discriminator are used to approximate the prediction confidence [19,33]. Instead, we propose the flaw detector to estimate the prediction confidence, with two key differences. First, the flaw detector predicts a dense probability map with location information while the discriminator predicts an image-level probability. Second, we use the ground truth of the labeled data to generate the targets of the flaw detector.

2.4 Perturbations in SSL

Many SSL methods heavily rely on perturbations for training. The consistent-based methods [25,38,44] utilize data augmentations to alter the inputs. To further improve the inconsistency, VAT [34] generates virtual adversarial noises while S4L [49] adds a rotation operation to the inputs. Others such as MixMatch [6] and ReMixMatch [5] generate perturbed samples by data interpolation. Apart from the perturbations in the inputs, Dropout perturbs the predictions through a random selection of nodes [37]. The models in Dual Student [22] have inconsistent predictions for the same input due to different initializations.

Since the perturbations from both data augmentations and Dropout are not suitable for dense outputs, GCT follows Dual Student in creating perturbations. However, unlike Dual Student, GCT learns from unlabeled data through the two SSL constraints based on the flaw detector, allowing GCT to be applicable to diverse pixel-wise tasks.

3 Guided Collaborative Training

3.1 Overview of GCT

In this section, we first present an overview of GCT. We then introduce the flaw detector and the two proposed SSL constraints. Fig. 2 shows the GCT framework.

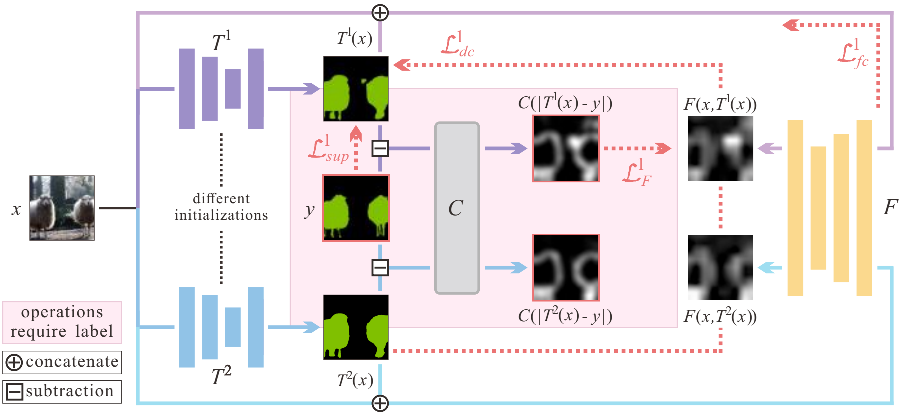


Fig. 2. The GCT Framework. It consists of two task models T^1, T^2 and a flaw detector F. Since T^1 and T^2 have different initializations, their predictions for the same x are inconsistent. These two task models learn from the unlabeled data through $\mathcal{L}_{dc}$ and $\mathcal{L}_{fc}$ under the guidance of the outputs of F. The ground truth of F is calculated on the labeled subset by an image processing pipeline C, which takes $T^1(x)$ (or $T^2(x)$) and y as the input. Here we take semantic segmentation as an example.

T^1 and T^2 are the two task models, which are referred to as T^k ($k \in \{1, 2\}$) in the following context. The architecture of T^k is arbitrary, and GCT allows the task models to have different architectures. The only requirement is that T^1 and T^2 should have different initializations to form the perturbations between them (which is the same as Dual Student). F denotes the flaw detector. In SSL, we have a dataset consisting of a labeled subset $\mathcal{X}_l$ with labels $\mathcal{Y}$ and an unlabeled subset $\mathcal{X}_u$. The inputs $\mathcal{X} = \mathcal{X}_l \cup \mathcal{X}_u$ for both T^1 and T^2 are exactly the same. Given an $x \in \mathcal{X}$, the GCT framework first predicts $T^k(x)$ of size $H \times W \times O$, where the value of O is defined by the specific task. Then, the concatenation of x and $T^k(x)$ is processed by F to estimate the flaw probability map $F(x, T^k(x))$ of size $H \times W \times 1$. The prediction confidence map can be approximated by $1 - F(x, T^k(x))$. We train GCT iteratively in two steps like GAN [14].

In the first step, we train T^k with fixed F. For the labeled data, the prediction $T^k(x_l)$ is supervised by its corresponding label y as:

$$\mathcal{L}^k_{sup}(x_l, y) = \sum_{h,w,o} \mathcal{R}(T^k(x_l)^{(h,w,o)},\ y^{(h,w,o)}), \tag{1}$$

where $\mathcal{R}(\cdot,\cdot)$ is a task-specific constraint, and (h, w, o) is a pixel index. To learn the unlabeled data, we propose a dynamic consistency constraint $\mathcal{L}_{dc}$ and a flaw correction constraint $\mathcal{L}_{fc}$, which are guided by the flaw probability map and will be described in Sec. 3.3 and Sec. 3.4, respectively. The final constraint for T^k is a combination of three constraints as:

$$\mathcal{L}^k_T(\mathcal{X}, \mathcal{Y}) = \sum_{\{x_l, y\}} \mathcal{L}^k_{sup}(x_l, y) + \sum_{x} \Big(\lambda_{dc}\, \mathcal{L}^k_{dc}(x) + \lambda_{fc}\, \mathcal{L}^k_{fc}(x)\Big), \tag{2}$$

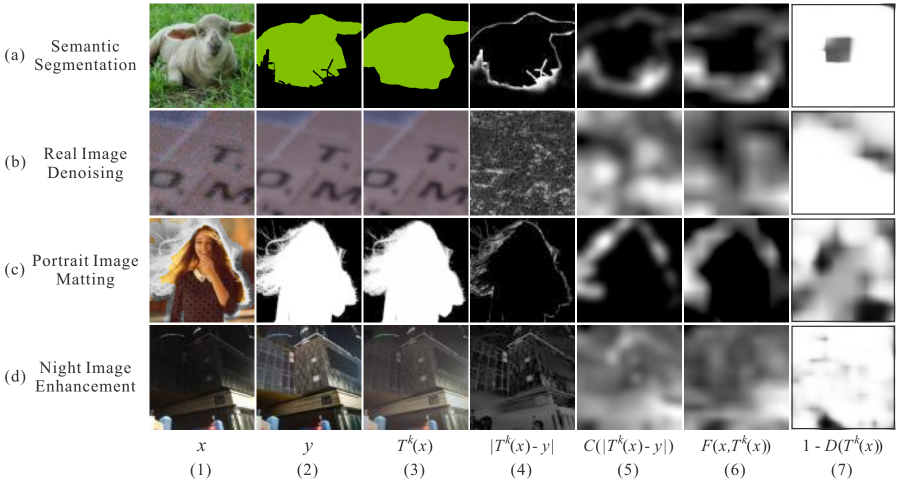


Fig. 3. Flaw Detector *vs.* **Discriminator.** The flaw detector F outputs $F(x, T^k(x))$ that highlights the flaw regions of $T^k(x)$ correctly. However, the fully convolutional discriminator D tends to activate all small errors. We show $1 - D(T^k(x))$ that activates the fake probability of each pixel, which has a similar meaning to the flaw probability. Since $|T^k(x) - y|$ is sparse and sharp, we use $C(|T^k(x) - y|)$ as the ground truth of F.

where $\{x_l, y\}$ is a pair of labeled data. λ_{dc} and λ_{fc} are hyper-parameters to balance the two SSL constraints.

In the second step, F learns from the labeled subset. We calculate the ground truth of F through a classical image processing pipeline C based on $T^k(x_l)$ and y. In our framework, F is trained by using Mean Square Error (MSE) as:

$$\mathcal{L}_F^k(\mathcal{X}_l, \mathcal{Y}) = \sum_{\{x_l, y\}} \Big(\frac{1}{2} \sum_{h,w} \Big(F(x_l, T^k(x_l))^{(h,w)} - C(\,|T^k(x_l) - y|\,)^{(h,w)} \Big)^2 \Big), \quad (3)$$

where $C(|T^k(x_l) - y|)$ is the ground truth of F, which will be discussed in Sec. 3.2.

3.2 Flaw Detector

On the labeled subset, the goal of the flaw detector F is to learn the flaw probability map $F(x_l, T^k(x_l))$ that indicates the difference between $T^k(x_l)$ and y, *i.e.*, the flaw regions in $T^k(x_l)$. One simple way to find the flaw regions is $|T^k(x_l) - y|$. However, it is difficult to learn many tasks since it is sparse and sharp (column (4) of Fig. 4). To address this problem, we introduce an image processing pipeline C that converts $|T^k(x_l) - y|$ to a dense probability map (column (5) of Fig. 4). C consists of three basic image processing operations: dilation, blurring and normalization[ii]. To estimate the flaw probability map $F(x_u, T^k(x_u))$ for the unlabeled data, we apply a common SSL assumption [51]: the distribution of

[ii] Refer to Appendix A in the Supplementary for the algorithm of C.

unlabeled data is the same as that of the labeled data. Therefore, F trained on the labeled subset should also work well on the unlabeled subset.

The architecture [(iii)] of the flaw detector is similar to the fully convolutional discriminator D in [19]. However, D averages all predicted pixels to get a single confidence value during training, as its target is an image-level real or fake probability. In pixel-wise tasks, the prediction is usually accurate for some pixels but not the others, and pixels of higher accuracy should have higher confidence. Using an average confidence to represent the overall confidence is not appropriate. For example, $T^1(x)$ may be more confident (more accurate) than $T^2(x)$ in a small local region although the average prediction confidence of $T^1(x)$ is lower than $T^2(x)$. Therefore, the per-pixel prediction confidence (from the flaw detector) is more meaningful than the average prediction confidence (from the discriminator) in pixel-wise tasks. Fig. 4 visualizes the results of F and D in the four validated tasks.

3.3 Dynamic Consistency Constraint

The two task models in GCT have inconsistent predictions for the same input x due to the perturbations between them. We use the dynamic consistency constraint $\mathcal{L}_{dc}$ to ensemble the reliable pixels in $T^1(x)$ and $T^2(x)$. Typically, the standard consistency constraint [25,44] is unidirectional, *e.g.*, from the ensemble model to the temporary model. Here, "dynamic" indicates that our $\mathcal{L}_{dc}$ is bidirectional and its direction changes with the flaw probability (Fig. 4(a)). Intuitively, if a pixel in $T^1(x)$ has a lower flaw probability, we treat it as the pseudo label to the corresponding pixel in $T^2(x)$. To assure the quality of the pseudo label, we introduce a flaw threshold $\xi \in [0, 1]$ to disable $\mathcal{L}_{dc}$ for the pixels that have higher flaw probability values than ξ in both $T^1(x)$ and $T^2(x)$. Through this process, there is an effective knowledge exchange between the task models, making them collaborators.

Formally, given a sample $x \in \mathcal{X}$, GCT outputs $T^1(x)$, $T^2(x)$, and their corresponding flaw probability maps $F(x, T^1(x))$, $F(x, T^2(x))$ through forward propagation. We first normalize the values in $F(x, T^k(x))$ to $[0, 1]$, and then set the pixels that are larger than ξ to 1 as:

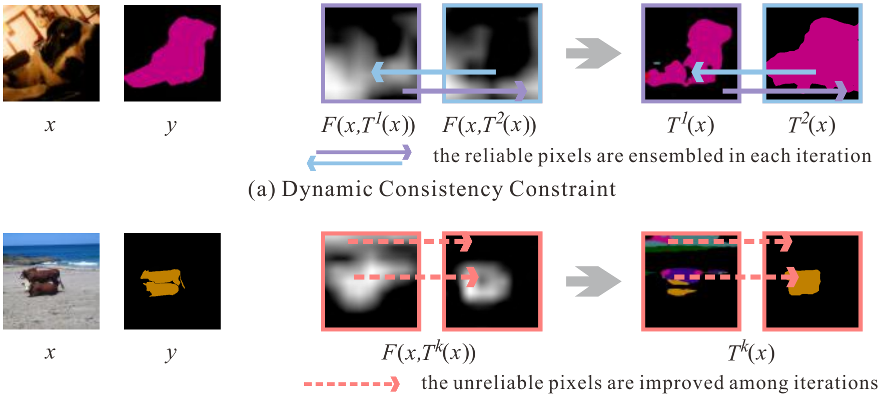
$$F(x, T^k(x))^{(h,w)} \leftarrow \max\left(F(x, T^k(x))^{(h,w)},\ \left\{F(x, T^k(x))^{(h,w)} > \xi\right\}_1\right). \tag{4}$$

$\{condition\}_1$ is a boolean-to-integer function, which outputs 1 when the *condition* is true and 0 otherwise. We define the dynamic consistency constraint for T^k as:

$$\begin{aligned} \mathcal{L}^k_{dc}(x) &= \frac{1}{2}\sum_{h,w}\Big(m^k_{dc}(x)^{(h,w)}\sum_{o}\Big(T^k(x)^{(h,w,o)} - T^{\tilde{k}}(x)^{(h,w,o)}\Big)^2\Big), \\ \text{where}\qquad m^k_{dc}(x)^{(h,w)} &= \left\{F(x, T^k(x))^{(h,w)} > F(x, T^{\tilde{k}}(x))^{(h,w)}\right\}_1. \end{aligned} \tag{5}$$

$\tilde{k}$ represents the other task model, *e.g.*, $\tilde{k} = 2$ when $k = 1$. If a flaw probability value in $F(x, T^{\tilde{k}}(x))$ is smaller than both ξ and the corresponding pixel in

[(iii)] Refer to Appendix B in the Supplementary for the architecture of the flaw detector.


x
y
F(x,T^1(x))
F(x,T^2(x))
T^1(x)
T^2(x)
the reliable pixels are ensembled in each iteration
x
y
F(x,T^k(x))
T^k(x)
the unreliable pixels are improved among iterations

(a) Dynamic Consistency Constraint

(b) Flaw Correction Constraint

Fig. 4. Proposed SSL Constraints. (a) The dynamic consistency constraint exchanges the confident knowledge between the task models. (b) The flaw correction constraint minimizes the flaw probability map for each task model.

$F(x, T^k(x))$, T^k will learn this pixel from $T^{\tilde{k}}$ through $\mathcal{L}^k_{dc}$. We use MSE since it is widely used in SSL and is general enough for many tasks. To prevent unreliable knowledge exchange at the beginning of training, we apply a cosine ramp-up operation with η epochs (from the standard consistency constraint) to $\mathcal{L}_{dc}$.

3.4 Flaw Correction Constraint

Apart from $\mathcal{L}_{dc}$, the flaw correction constraint $\mathcal{L}_{fc}$ attempts to correct the unreliable predictions of the task models (Fig. 4(b)). The key idea behinds $\mathcal{L}_{fc}$ is to force the values in the flaw probability map to become zero. We define $\mathcal{L}_{fc}$ for T^k (with F being fixed) as:

$$\mathcal{L}^k_{fc}(x) = \frac{1}{2}\sum_{h,w}\Big(m_{fc}(x)^{(h,w)}\Big(F(x, T^k(x))^{(h,w)} - 0\Big)^2\Big). \tag{6}$$

We use a binary mask $m_{fc}(x)$ to enable $\mathcal{L}_{fc}$ on the pixels without $\mathcal{L}_{dc}$, *i.e.*, the pixels with unreliable predictions in both task models:

$$m_{fc}(x)^{(h,w)} = \big\{F(x, T^1(x))^{(h,w)} > \xi \;\; \text{AND} \;\; F(x, T^2(x))^{(h,w)} > \xi\big\}_1. \tag{7}$$

We consider that the flaw detector F helps improve the task models through $\mathcal{L}_{fc}$. For a system containing only one task model and the flaw detector, the objectives can be derived from Eq. (3) and (6) as:

$$\begin{aligned} \min_F \; V_{GCT}(F) &= \frac{1}{2}\mathbb{E}_{\{x_l, y\}\sim P_{\mathcal{X}_l, \mathcal{Y}}}\,[(F(x_l, T^k(x_l)) - C(\,|T^k(x_l) - y|\,))^2], \\ \min_T \; V_{GCT}(T^k) &= \frac{1}{2}\mathbb{E}_{x\sim P_{\mathcal{X}}}\,[(F(x, T^k(x)) - 0)^2], \end{aligned} \tag{8}$$

where $\mathcal{X}_l$ and $\mathcal{X}$ have the same distribution. We simplify Eq. (8) by removing the pixel summation operation. In such situation, F learns the flaw probability map while T^k optimizes it with a zero label. If we assume that the training process converges to an optimal solution in iteration t^*, we have:

$$\lim_{t \to t^*} C(|T^k(x_l) - y|) = 0 \quad \Rightarrow \quad \lim_{t \to t^*} V_{GCT}(F) = V_{GCT}(T^k), \tag{9}$$

where t is the current iteration. Hence, the objective $V_{GCT}(F)$ changes during the training process and is equal to $V_{GCT}(T^k)$ when $t = t^*$. The alignment in the objectives indicates that F and T^k are collaborative to some degree.

To illustrate the difference between $\mathcal{L}_{fc}$ and the adversarial constraint, we compare Eq. (8) with the objectives of LSGAN [32]. If we modify LSGAN for SSL, its objectives should be:

$$\begin{aligned} &\min_{D} V_{LSGAN}(D) = \frac{1}{2}\mathbb{E}_{x \sim P_{\mathcal{X}}}\left[(D(T^k(x)) - 1)^2\right] + \frac{1}{2}\mathbb{E}_{y \sim P_{\mathcal{Y}}}\left[(D(y)) - 0)^2\right], \\ &\min_{T^k} V_{LSGAN}(T^k) = \frac{1}{2}\mathbb{E}_{x \sim P_{\mathcal{X}}}\left[(D(T^k(x)) - 0)^2\right], \end{aligned} \tag{10}$$

where D is the standard discriminator that tries to differentiate $T^k(x)$ and y. In contrast, T^k tries to match the distributions between $T^k(x)$ and y. Here we reverse the labels, *i.e.*, 1 for fake and 0 for real, to be consistent with Eq. (8). Since the targets of D are constants, we have:

$$\lim_{t \to t^*} V_{LSGAN}(D) \neq V_{LSGAN}(T^k), \tag{11}$$

which means that D and T^k are adversarial during the whole training process.

4 Experiments

In order to evaluate our framework under different ratios of the labeled data, we experiment on the standard benchmarks for semantic segmentation and real image denoising. We also experiment on the practical datasets created for portrait image matting and night image enhancement to demonstrate the generalization of GCT in real applications. We further conduct ablation experiments to analyze various aspects of GCT.

Implementation Details. We compare GCT with the model trained by the labeled data only (SupOnly) and several state-of-the-art SSL methods that can be applied to various pixel-wise tasks: (1) the adversarial-based method proposed in [19] (AdvSSL); (2) the consistent-based Mean Teacher (MT) [44]; (3) the self-supervised SSL (S4L) [49]. For AdvSSL, we remove the constraint that requires classification probability to make it compatible with pixel-wise regression. For MT, we use MSE for the consistency constraint. We do not add Gaussian noise as extra perturbations since it will degrade the performance. For S4L, a four-category classifier trained by Cross Entropy is added to the end of the task model to predict the rotation angles. (0°, 90°, 180°, 270°).

Experimental Setup. We notice that existing works of pixel-wise SSL usually report a fully supervised baseline with a lower performance than the original paper due to inconsistent hyper-parameters. In image classification, a similar situation has been discussed by [35]. To fairly evaluate the performance of SSL, we define some training rules to improve the SupOnly baselines. We denote the total number of trained samples as $N = S * T * b$, where S is the training epochs, T is the number of iterations in each epoch, and b is the batch size, which is fixed in each task. For the experiments performed on the standard benchmarks:

(1) We train the fully supervised baseline according to the hyper-parameters from the original paper to achieve a comparable result. The same hyper-parameters (except S) are used in (2) and (3).
(2) We use the same S as in (1) to train the models supervised by the labeled subset (SupOnly). Although T decreases as the labeled data reduces, to prevent overfitting, we do not increase N by training more epochs.
(3) We adjust S to ensure that N in SSL experiments is the same as (1). In SSL experiments, each batch contains both labeled and unlabeled data. We define "epoch" as going through the unlabeled subset for once. Meanwhile, the labeled subset is repeated several times inside an epoch.

By following these rules, the SupOnly baselines obtain good enough performance and do not overfit. The models trained by SSL methods have the same computational overhead, *i.e.*, the same N, as the fully supervised baseline. For experiments on the practical datasets, we first train S epochs for the SupOnly baselines. Afterwards, we train the SSL models with the same S. We use the grid search to find suitable hyper-parameters for all SSL methods. [iv][v].

4.1 Semantic Segmentation Experiments

Semantic segmentation [9,10,29] takes an image as input and predicts a series of category masks, which link each pixel in the input image to a class (Fig. 4(a)). We conduct experiments on the Pascal VOC 2012 dataset [12], which comprises 20 foreground classes along with 1 background class. The extra annotation set from the Segmentation Boundaries Dataset (SBD) [16] is combined to expand the dataset. Therefore, we have 10,582 training samples and 1,449 validation samples. During training, the input images are cropped to 321 × 321 after random scaling and horizontal flipping. Following previous works [19,33], we use DeepLab-v2 [9] with the ResNet-101 [17] backbone as the SupOnly baselines and as the task model in SSL methods. The same configurations as the original paper of DeepLab-v2 are applied, except the multi-scale fusion trick.

For SSL, we randomly extract 1/16, 1/8, 1/4, 1/2 samples as the labeled subset, and use the rest of the training set as the unlabeled subset. Note that the same data splits are used in all SSL methods. Table 1 shows the mean

[iv] Refer to Appendix C in the Supplementary for more training details.

[v] Refer to Appendix D in the Supplementary for visual comparisons.

Table 1. Results of Semantic Segmentation. We report mIOU (%) on the validation set of Pascal VOC 2012 averaged over 3 runs. The task model is DeepLab-v2.

Methods	1/16 labels	1/8 labels	1/4 labels	1/2 labels	full labels
SupOnly	64.55	68.38	70.69	73.56	75.32
MT [44]	66.08	69.81	71.28	73.23	75.28
S4L [49]	64.71	68.65	70.97	73.43	75.38
AdvSSL [19]	65.67	69.89	71.53	74.48	**75.86**
GCT (Our)	**67.19**	**72.14**	**73.62**	**74.82**	75.73

Table 2. Results of Real Image Denoising. We report PSNR (dB) on the validation set of SIDD averaged over 3 runs. The task model is DHDN.

Methods	1/16 labels	1/8 labels	1/4 labels	1/2 labels	full labels
SupOnly	37.52	38.16	38.74	39.14	39.38
MT [44]	37.73	38.22	38.64	39.08	39.43
S4L [49]	37.81	38.32	38.88	39.21	39.16
AdvSSL [19]	37.85	38.28	38.83	39.18	39.47
GCT (Our)	**38.13**	**38.56**	**38.96**	**39.30**	**39.51**

Intersection-over-Union (mIOU) on the PASCAL VOC 2012 dataset with pretraining on the Microsoft COCO dataset [28]. GCT achieves a performance increase of 1.26% (under 1/2 labels) to 3.76% (under 1/8 labels) over the SupOnly baselines. Moreover, our fully supervised baseline (75.32%) is comparable with the original paper of DeepLab-v2 (75.14%), which is better than the result reported in [19] (73.6%). Therefore, all SSL methods only have slight improvement under the full labels.

4.2 Real Image Denoising Experiments

Real image denoising [3,15,50] is a task that devotes to removing the real noise, rather synthetic noise, from an input natural image (Fig. 4(b)). We conduct experiments on the SIDD dataset [1], which is one of the largest benchmarks on real image denoising. It contains 160 image pairs (noisy image and clean image) for training and 40 image pairs for validation. We split each image pair into multiple patches with size 256 × 256 for training. The total training samples is about 30,000. We use DHDN [36], a method that won the second place in the NTRIE 2019 real image denoising challenge [2], as the task model since the code for the first place winner has not been published. The peak-signal-to-noise-ratio (PSNR) is used as the validation metric.

In image denoising, even small errors between the prediction and the ground truth can result in obvious visual artifacts. It means that the reliable pseudo labels are difficult to obtain, *i.e.*, this task is difficult for SSL. We notice that the task models with the same architecture in GCT have similar predictions.

Table 3. Results of Portrait Image Matting and Night Image Enhancement. We report PSNR (dB) on the validation set of the practical datasets averaged over 3 runs. In the table, "L" means labeled data while "U" means unlabeled data.

	Portrait Image Matting		Night Image Enhancement
Methods	100L + 3,850U	100L + 7,700U	200L + 1,500U
SupOnly	25.39	25.39	18.72
MT [44]	26.60	27.63	19.93
S4L [49]	26.87	28.24	19.63
AdvSSL [19]	26.52	27.57	19.59
GCT (Our)	**27.35**	**29.38**	**20.14**

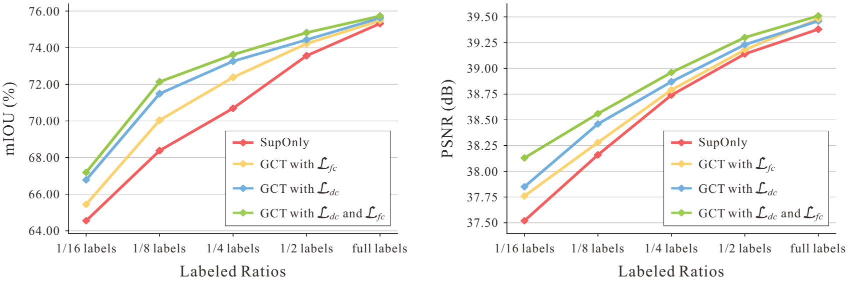
Therefore, the perturbations from different initializations are not strong enough. To alleviate this problem, we replace one of the task models with DIDN [48] that won the third place in the NTRIE 2019 challenge. We still use DHDN for validation.

We extract 1/16, 1/8, 1/4, 1/2 labeled image pairs randomly for SSL. As shown in Table 2, our fully supervised baseline achieves 39.38dB (PSNR), which is comparable with the top-level results on the SIDD benchmark. Although SSL shows limited performance in this difficult task, GCT surpasses other SSL methods under all labeled ratios. Notably, GCT improves on PSNR by 0.61dB with 1/16 labels (only 10 labeled image pairs) while the previous SSL methods improve on PSNR by 0.33dB at most.

4.3 Portrait Image Matting Experiments

Image Matting [40,47] predicts a foreground mask (matte) from an input image and a pre-defined trimap. Each pixel value in the matte is a probability between $[0, 1]$. We focus on the matting of portrait images here, which has important applications on smartphone, *e.g.*, blurring the background of an image. In Fig. 3(c), the trimap is merged into x for visualization by setting the pixels inside the unknown region of the trimap to gray. Since there are no open-source benchmarks, we first collected 8,000 portrait images from Flickr. We then generate the trimaps from the results of a pre-trained segmentation model. After that, we select 300 images with fine details and label them by Photoshop (~20min per image). Finally, we combine 100 labeled images with 7,700 unlabeled images as the training set, while the remaining 200 labeled images are used as the validation set. For each labeled image, we generate 15 samples by random cropping and 35 samples by background replacement (with the OpenImage dataset [24]). For each unlabeled image, we generate 5 samples by random cropping. The structure of our task model is derived from [47], which is a milestone in image matting.

In this task, we verify the impact of increasing the amount of unlabeled data on SSL by experimenting on two configurations. With 100 labeled images, (1) we randomly select half (3,850) of unlabeled images for training, and (2) we use



(a) Semantic Segmentation Results on Pascal VOC

(b) Real Image Denoising Results on SIDD

Fig. 5. Ablation of the Proposed SSL Constraints. We compare the performance of $\mathcal{L}_{dc}$ and $\mathcal{L}_{fc}$ on (a) the Pascal VOC benchmark and (b) the SIDD benchmark. The results of SupOnly (red) and GCT with two SSL constraints (green) are the same as (a) Table 1 and (b) Table 2.

all (7,700) unlabeled images for training. As shown in Table 3, GCT yields an improvement over the SupOnly baselines by 1.96dB and 3.99dB for 3,850 and 7,700 unlabeled images respectively. This indicates that the SSL performance can be effectively improved by increasing the amount of unlabeled data. In addition, doubling the amount of unlabeled images achieves a more significant improvement (2.03dB) with GCT, compared with existing SSL methods.

4.4 Night Image Enhancement Experiments

Night Image Enhancement [8,13] is another common vision application. This task adjusts the coefficients of the channels in a night image to show more details (Fig. 4(d)). Our dataset contains 1,900 night images captured by smartphones, of which 400 images are labeled using Photoshop (∼15min per image). We combine 200 labeled images with 1,500 unlabeled images for training and use another 200 labeled images for testing. We use horizontal flipping, slight rotation, and random cropping (to 512 × 512) as data augmentations during training. We regard HDRNet [13] as the task model. Since the dataset is small, we experimented with only one SSL configuration (Table 3). Similar to the experiments in the other three tasks, GCT outperforms existing SSL methods.

4.5 Ablation Experiments

We conduct ablation studies to analyze the proposed SSL constraints, the hyperparameters in GCT, and the combination of the flaw detector and Mean Teacher.

Effect of the SSL Constraints. By default, GCT learns from the unlabeled data through the two SSL constraints simultaneously. In Fig. 5, we compare the experiments of training GCT with only one SSL constraint on the benchmarks for semantic segmentation and real image denoising. The results demonstrate

Table 4. Ablation of Hyper-Parameters. We report mIOU (%) on the Pascal VOC benchmark with 1/8 labels. The result under $\xi = 0.4$ or $\eta = 3$ is the same as Table 1.

flaw threshold ξ					
0.0	0.2	0.4	0.6	0.8	1.0
70.04	70.92	72.14	**72.43**	71.96	71.49

ramp-up epochs η				
0	1	3	5	10
71.34	72.03	**72.14**	72.06	71.95

that both $\mathcal{L}_{dc}$ and $\mathcal{L}_{fc}$ are effective. GCT with $\mathcal{L}_{dc}$ boosts the performance impressively, proving that the knowledge exchange between the two task models is reliable and effective. Meanwhile, the curve of GCT with $\mathcal{L}_{fc}$ indicates that the flaw detector also plays a vital role in learning the unlabeled data. Moreover, combining $\mathcal{L}_{dc}$ and $\mathcal{L}_{fc}$ allows GCT to achieve the optimal performance.

Hyper-parameters in GCT. We analyze the two hyper-parameters required by GCT (mentioned in Sec. 3.3), the flaw threshold ξ and the cosine ramp-up epochs η of $\mathcal{L}_{dc}$, on the Pascal VOC benchmark for semantic segmentation with 1/8 labels. Table 4 (left) shows the results under different ξ, which controls the combination of the two SSL constraints. Specifically, only $\mathcal{L}_{fc}$ is applied when $\xi = 0.0$, and only $\mathcal{L}_{dc}$ is applied when $\xi = 1.0$. Our experiments show that ξ can be set roughly, *e.g.*, $\xi \in [0.4, 0.8]$ is suitable for semantic segmentation. The cosine ramp-up with η epochs prevents exchanging unreliable knowledge due to the non-convergent flaw detector in the early training stage. The results in Table 4 (right) indicate that GCT is robust to η, even though the cosine ramp-up is necessary for the best performance.

Combination of the Flaw Detector and MT. The consistency constraint in MT is applied from the teacher model to the student model. However, the teacher model may be worse than the student model on some pixels, which may cause a performance degradation. To avoid this problem, we use the flaw detector to disable the consistency constraint when the flaw probability of the teacher's prediction is larger than the student's prediction. Under 1/8 labels, this method improves the mIOU value of MT from 69.81% to 70.47% on Pascal VOC and improves the PSNR value of MT from 38.22dB to 38.42dB on SIDD.

5 Conclusions

We have studied the generalization of SSL to diverse pixel-wise tasks and indicated the drawbacks of existing SSL methods in these tasks, which to the best of our knowledge is the first. We have presented a new general framework, named GCT, for pixel-wise SSL. Our experiments have proved its effectiveness in a variety of vision tasks. Meanwhile, we also note that SSL still has limited performance for tasks that require highly precise pseudo labels, such as image denoising. A possible future work is to investigate this problem and explore ways to create more accurate pseudo labels.

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 $\mu$ ; Operations repeat times $\nu$.
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 [20] and leaky ReLU [31] 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 [23] (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) [44]; (2) the self-supervised SSL (S4L) [49]; (3) the adversarial-based method proposed in [19] (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 [44]	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 [49]	λ_{S4L} - coefficient for scaling the unsupervised rotation constraint
AdvSSL [19]	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 [44]	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 [49]	λ_{S4L}	0.10	1.00	1.00	1.00
AdvSSL [19]	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	0.60	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.

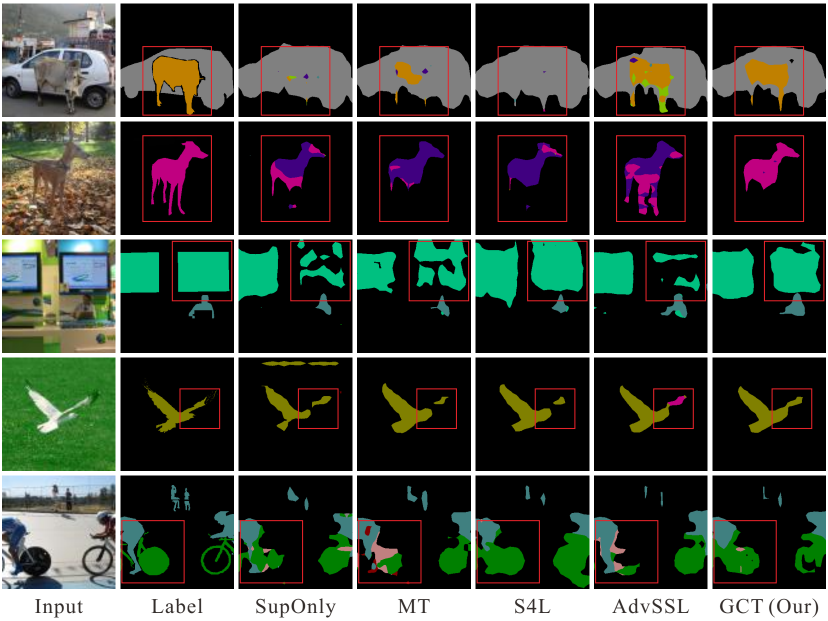


Fig. 1. Semantic Segmentation. Comparisons on the PASCAL VOC dataset using 1/8 labeled data.

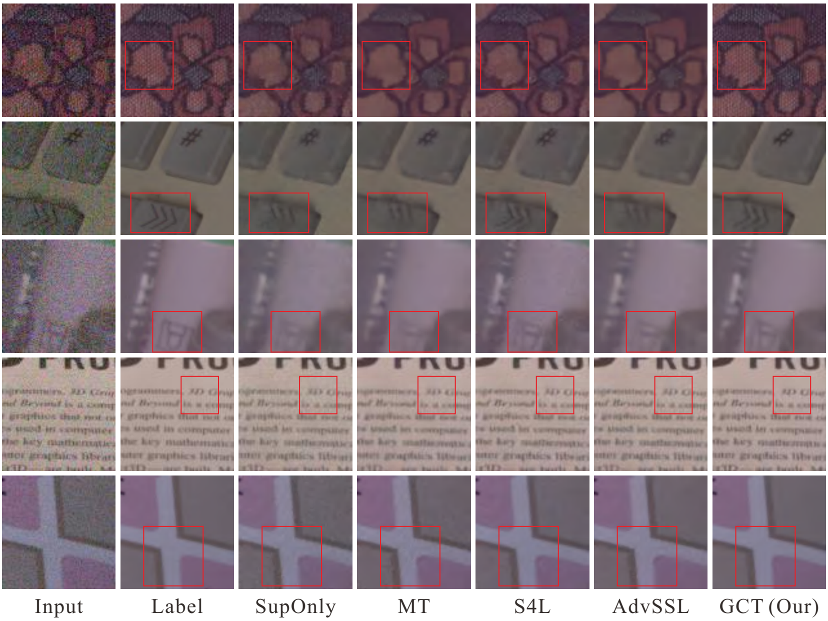


Fig. 2. Real Image Denoising. Comparisons on the SIDD dataset using 1/8 labeled data.

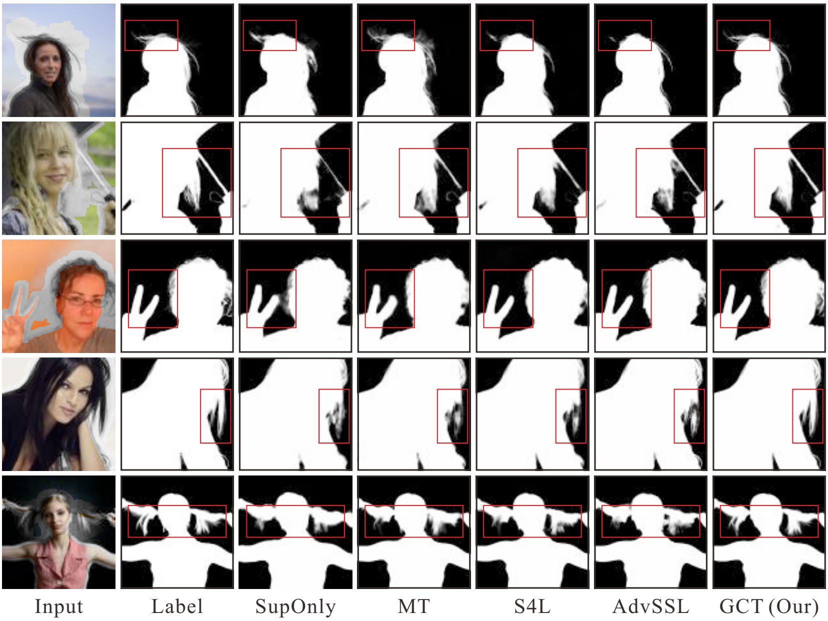


Fig. 3. Portrait Image Matting. Comparisons on our dataset using 100 labeled data and 3850 unlabeled data.

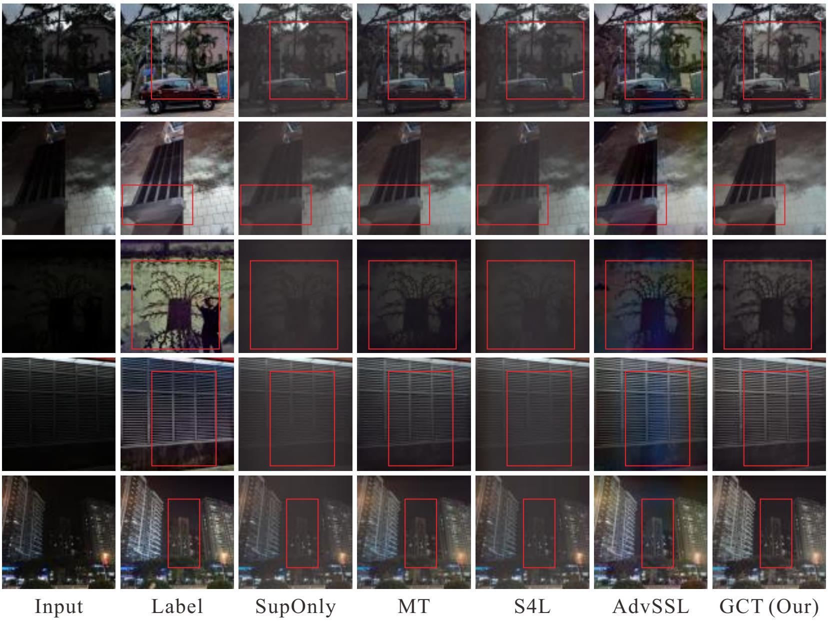


Fig. 4. Night Image Enhancement. Comparisons on our dataset using 200 labeled data and 1500 unlabeled data.

References

1. Abdelhamed, A., Lin, S., Brown, M.S.: A high-quality denoising dataset for smartphone cameras. In: CVPR (2018)
2. Abdelhamed, A., Timofte, R., Brown, M.S.: Ntire 2019 challenge on real image denoising: Methods and results. In: CVPRW (2019)
3. Anwar, S., Barnes, N.: Real image denoising with feature attention. In: ICCV (2019)
4. Athiwaratkun, B., Finzi, M., Izmailov, P., Wilson, A.G.: There are many consistent explanations of unlabeled data: Why you should average. In: ICLR (2019)
5. Berthelot, D., Carlini, N., Cubuk, E.D., Kurakin, A., Sohn, K., Zhang, H., Raffel, C.: Remixmatch: Semi-supervised learning with distribution matching and augmentation anchoring. In: ICLR (2020)
6. Berthelot, D., Carlini, N., Goodfellow, I.G., Papernot, N., Oliver, A., Raffel, C.: Mixmatch: A holistic approach to semi-supervised learning. In: NeurIPS (2019)
7. Chen, C., Liu, W., Tan, X., Wong, K.: Semi-supervised learning for face sketch synthesis in the wild. In: ACCV (2018)
8. Chen, C., Chen, Q., Xu, J., Koltun, V.: Learning to see in the dark. In: CVPR (2018)
9. 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. TPAMI (2017)
10. Chen, L.C., Zhu, Y., Papandreou, G., Schroff, F., Adam, H.: Encoder-decoder with atrous separable convolution for semantic image segmentation. In: ECCV (2018)

11. Dai, Z., Yang, Z., Yang, F., Cohen, W.W., Salakhutdinov, R.R.: Good semi-supervised learning that requires a bad gan. In: NeurIPS (2017)
12. Everingham, M., Eslami, S.M.A., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The pascal visual object classes challenge: A retrospective. IJCV (2015)
13. Gharbi, M., Chen, J., Barron, J.T., Hasinoff, S.W., Durand, F.: Deep bilateral learning for real-time image enhancement. In: SIGGRAPH (2017)
14. Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A.C., Bengio, Y.: Generative adversarial nets. In: NeurIPS (2014)
15. Guo, S., Yan, Z., Zhang, K., Zuo, W., Zhang, L.: Toward convolutional blind denoising of real photographs. In: CVPR (2019)
16. Hariharan, B., Arbelaez, P., Bourdev, L., Maji, S., Malik, J.: Semantic contours from inverse detectors. In: ICCV (2011)
17. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)
18. Huang, Z., Wang, X., Wang, J., Liu, W., Wang, J.: Weakly-supervised semantic segmentation network with deep seeded region growing. In: CVPR (2018)
19. 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)
20. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: ICML (2015)
21. Kalluri, T., Varma, G., Chandraker, M., Jawahar, C.V.: Universal semi-supervised semantic segmentation. In: ICCV (2019)
22. Ke, Z., Wang, D., Yan, Q., Ren, J., Lau, R.W.: Dual student: Breaking the limits of the teacher in semi-supervised learning. In: ICCV (2019)
23. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: ICLR (2014)
24. Kuznetsova, A., Rom, H., Alldrin, N., Uijlings, J.R.R., Krasin, I., Pont-Tuset, J., Kamali, S., Popov, S., Malloci, M., Duerig, T., Ferrari, V.: The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale (2018)
25. Laine, S., Aila, T.: Temporal ensembling for semi-supervised learning. In: ICLR (2017)
26. Lee, J., Kim, E., Lee, S., Lee, J., Yoon, S.: Ficklenet: Weakly and semi-supervised semantic image segmentation using stochastic inference. In: CVPR (2019)
27. LI, C., Xu, T., Zhu, J., Zhang, B.: Triple generative adversarial nets. In: NeurIPS (2017)
28. Lin, T.Y., Maire, M., Serge Belongie, J.H., Perona, P., Ramanan, D., Dollr, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: ECCV (2014)
29. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. TPAMI (2016)
30. Luo, Y., Zhu, J., Li, M., Ren, Y., Zhang, B.: Smooth neighbors on teacher graphs for semi-supervised learning. In: CVPR (2018)
31. Maas, A.L., Hannun, A.Y., Ng, A.Y.: Rectifier nonlinearities improve neural network acoustic models. In: ICML (2013)
32. Mao, X., Li, Q., Xie, H., Lau, R.Y.K., Wang, Z.: Least squares generative adversarial networks. In: ICCV (2017)
33. Mittal, S., Tatarchenko, M., Brox, T.: Semi-supervised semantic segmentation with high- and low-level consistency. TPAMI (2019)
34. Miyato, T., Maeda, S.i., Ishii, S., Koyama, M.: Virtual adversarial training: a regularization method for supervised and semi-supervised learning. TPAMI (2018)

35. Oliver, A., Odena, A., Raffel, C., Cubuk, E., Goodfellow, I.: Realistic evaluation of semi-supervised learning algorithms. In: NeurIPS (2018)
36. Park, B., Yu, S., Jeong, J.: Densely connected hierarchical network for image denoising. In: CVPRW (2019)
37. Park, S., Park, J., Shin, S., Moon, I.: Adversarial dropout for supervised and semi-supervised learning. In: AAAI (2018)
38. Qiao, S., Shen, W., Zhang, Z., Wang, B., Yuille, A.L.: Deep co-training for semi-supervised image recognition. In: ECCV (2018)
39. Rasmus, A., Berglund, M., Honkala, M., Valpola, H., Raiko, T.: Semi-supervised learning with ladder networks. In: NeurIPS (2015)
40. Shen, X., Tao, X., Gao, H., Zhou, C., Jia, J.: Deep automatic portrait matting. In: ECCV (2016)
41. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: ICLR (2014)
42. Springenberg, J.T.: Unsupervised and semi-supervised learning with categorical generative adversarial networks. In: ICLR (2015)
43. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: A simple way to prevent neural networks from overfitting. JMLR (2014)
44. Tarvainen, A., Valpola, H.: Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In: NeurIPS (2017)
45. Tran, P.V.: Exploring self-supervised regularization for supervised and semi-supervised learning. arXiv preprint arXiv:1906.10343 (2019)
46. Wang, Q., Li, W., Van Gool, L.: Semi-supervised learning by augmented distribution alignment. In: ICCV (2019)
47. Xu, N., Price, B.L., Cohen, S., Huang, T.S.: Deep image matting. In: CVPR (2017)
48. Yu, S., Park, B., Jeong, J.: Deep iterative down-up cnn for image denoising. In: CVPRW (2019)
49. Zhai, X., Oliver, A., Kolesnikov, A., Beyer, L.: S4l: Self-supervised semi-supervised learning. In: ICCV (2019)
50. Zhang, K., Zuo, W., Zhang, L.: Ffdnet: Toward a fast and flexible solution for cnn based image denoising. TIP (2018)
51. Zhu, X.: Semi-supervised learning literature survey (2006)