Article

Revisiting Consistency for Semi-Supervised Semantic Segmentation [†]

Ivan Grubišić [1] , Marin Oršić [2] and Siniša Šegvić [1,*]

[1] Faculty of Electrical Engineering and Computing, University of Zagreb, Unska 3, 10000 Zagreb, Croatia
[2] Microblink Ltd., Strojarska Cesta 20, 10000 Zagreb, Croatia
[*] Correspondence: sinisa.segvic@fer.hr
[†] This paper is an extended version of our paper published in the proceedings of the 17th International Conference on Machine Vision and Applications, MVA 2021, Aichi, Japan, 25–27 July 2021.

Abstract: Semi-supervised learning is an attractive technique in practical deployments of deep models since it relaxes the dependence on labeled data. It is especially important in the scope of dense prediction because pixel-level annotation requires substantial effort. This paper considers semi-supervised algorithms that enforce consistent predictions over perturbed unlabeled inputs. We study the advantages of perturbing only one of the two model instances and preventing the backward pass through the unperturbed instance. We also propose a competitive perturbation model as a composition of geometric warp and photometric jittering. We experiment with efficient models due to their importance for real-time and low-power applications. Our experiments show clear advantages of (1) one-way consistency, (2) perturbing only the student branch, and (3) strong photometric and geometric perturbations. Our perturbation model outperforms recent work and most of the contribution comes from the photometric component. Experiments with additional data from the large coarsely annotated subset of Cityscapes suggest that semi-supervised training can outperform supervised training with coarse labels. Our source code is available at https://github.com/Ivan1248/semisup-seg-efficient.

Keywords: semi-supervised learning; semantic segmentation; dense prediction; one-way consistency; deep learning; scene understanding

1. Introduction

Most machine learning applications are hampered by the need to collect large annotated datasets. Learning with incomplete supervision [1,2] presents a great opportunity to speed up the development cycle and enable rapid adaptation to new environments. Semi-supervised learning [3–5] is especially relevant in the dense prediction context [6–8] since pixel-level labels are very expensive, whereas unlabeled images are easily obtained.

Dense prediction typically operates on high resolutions in order to be able to recognize small objects. Furthermore, competitive performance requires learning on large batches and large crops [9–11]. This typically entails a large memory footprint during training, which constrains model capacity [12]. Many semi-supervised algorithms introduce additional components to the training setup. For instance, training with surrogate classes [13] implies infeasible logit tensor size, while GAN-based approaches require an additional generator [6,14] or discriminator [7,15,16]. Some other approaches require multiple model instances [17–20] or accumulated predictions across the dataset [21]. Such designs are less appropriate for dense prediction since they constrain model capacity.
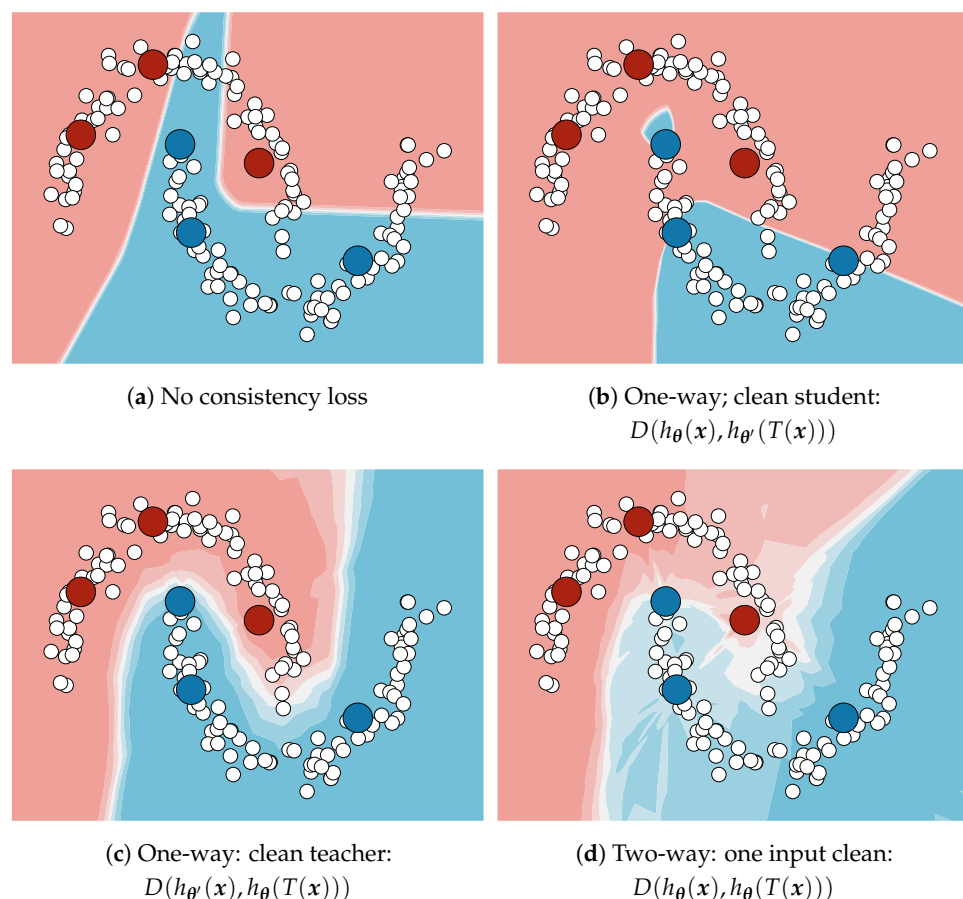
This paper studies semi-supervised approaches [3,5,18,21,22] that require consistent predictions over input perturbations. In the considered consistency objective, input perturbations affect only one of two model instances, while the gradient is not propagated towards the model instance which operates on the clean (weakly perturbed) input [4,5]. For

brevity, we refer to the two model instances as the perturbed branch and the clean branch. If the gradient is not computed in a branch, we refer to it as the teacher, and otherwise as the student. Hence, we refer to the considered approach as a one-way consistency with the clean teacher.

Let x be the input, T a perturbation to which the ideal model should be invariant, h_{θ} the student, and $h_{\theta'}$ the teacher, where θ' denotes a frozen copy of the student parameters θ. Then, one-way consistency with clean teacher can be expressed as a divergence D between the two predictions:

$$L_{\theta}^{ct}(x, T) = D(h_{\theta'}(x), h_{\theta}(T(x))) . \tag{1}$$

We argue that the clean teacher approach is a method of choice in case of perturbations that are too strong for standard data augmentation. In this setting, perturbed inputs typically give rise to less reliable predictions than their clean counterparts. Figure 1 illustrates the advantage of the clean teacher approach in comparison with other kinds of consistency on the Two moons dataset. The clean student experiment (Figure 1b) shows that many blue data points get classified into the red class due to teacher inputs being pushed towards labeled examples of the opposite class. This aberration does not occur when the teacher inputs are clean (Figure 1c). Two-way consistency [21] (Figure 1d) can be viewed as a superposition of the two one-way approaches and works better than (Figure 1b), but worse than (Figure 1c). In our experiments, D corresponds to KL divergence.

(**a**) No consistency loss

(**b**) One-way; clean student:
$D(h_{\theta}(x), h_{\theta'}(T(x)))$

(**c**) One-way: clean teacher:
$D(h_{\theta'}(x), h_{\theta}(T(x)))$

(**d**) Two-way: one input clean:
$D(h_{\theta}(x), h_{\theta}(T(x)))$

Figure 1. A toy semi-supervised classification problem with six labeled (red, blue) and many unlabeled 2D datapoints (white). All setups involve 20,000 epochs of semi-supervised training with cross-entropy and default Adam optimization hyper-parameters. The consistency loss was set to none (**a**), one-way with clean student (**b**), one-way with clean teacher (**c**), and two-way with one input clean (**d**). One-way consistency with clean teacher outperforms all other formulations.

One-way consistency is especially advantageous in the dense prediction context since it does not require caching latent activations in the teacher. This allows for better training in many practical cases where model capacity is limited by GPU memory [12,23]. In comparison with two-way consistency [20,21], the proposed approach both improves generalization and approximately halves the training memory footprint.

This paper is an extended version of our preliminary conference report [24]. It exposes the elements of our method in much more detail and complements them with many new experiments. In particular, the most important additions are additional ablation and validation studies, full-resolution Cityscapes experiments, and a detailed analysis of a large-scale experiment that compares the contribution of coarse labels with semi-supervised learning on unlabeled images. The new experiments add more evidence in favor of one-way consistency with respect to other consistency variants, investigate the influence of particular components of our algorithm and various hyper-parameters, and investigate the behavior of the proposed algorithm in different data regimes (higher resolution; additional unlabeled images).

The consolidated paper proposes a simple and effective method for semi-supervised semantic segmentation. One-way consistency with clean teacher [4,5,25] outperforms the two-way formulation in our validation experiments. In addition, it retains the memory footprint of supervised training because the teacher activations depend on parameters that are treated as constants. Experiments with a standard convolutional architecture [26] reveal that our photometric and geometric perturbations lead to competitive generalization performance and outperform their counterpart from a recent related work [25]. A similar advantage can be observed in experiments with a recent efficient architecture [27], which offers a similar performance while requiring an order of magnitude of less computation. To our knowledge, this is the first account of the evaluation of semi-supervised algorithms for dense prediction with a model capable of real-time inference. This contributes to the goals of Green AI [28] by enabling competitive research with less environmental damage.

This paper proceeds as follows. Section 2 presents related work. Section 3 describes the one-way consistency objective adapted to dense prediction, our perturbation model, and a description of our memory-efficient consistency training procedure. Section 4 presents the experimental setup, which includes information about datasets and training details, as well as the performed experiments in semi-supervised semantic segmentation. Finally, Section 5 presents the conclusion.

2. Related Work

Our work spans the fields of dense prediction and semi-supervised learning. The proposed methodology is most related to previous work in semi-supervised semantic segmentation.

2.1. Dense Prediction

Image-wide classification models usually achieve efficiency, spatial invariance, and integration of contextual information by gradual downsampling of representations and use of global spatial pooling operations. However, dense prediction also requires location accuracy. This emphasizes the trade-off between efficiency and quality of high-resolution features in the model design. Some common designs use a classification backbone as a feature encoder and attach a decoder that restores the spatial resolution. Many approaches seek to enhance contextual information, starting with FCN-8s [29]. UNet [30] improves spatial details by directly using earlier representations of the encoder in a symmetric decoder. Further work improves the efficiency with lighter decoders [23,31]. Some models use context aggregation modules such as spatial pyramid pooling [32] and multi-scale inference [31,33]. DeepLab [26] increases the receptive field through dilated convolutions and improves spatial details through CRF post-processing. HRNet [34] maintains the full resolution throughout the whole model and incrementally introduces parallel lower-

resolution branches that exchange information between stages. Semantic segmentation gains much from ImageNet pre-trained encoders [23,26].

2.2. Semi-Supervised Learning

Semi-supervised methods often rely on some of the following assumptions about the data distribution [35]: (1) similar inputs in high density regions correspond to similar outputs (smoothness assumption), (2) inputs form clusters separated by low-density regions and inputs within clusters are likely to correspond to similar outputs (cluster assumption), and (3) the data lies on a lower-dimensional manifold (manifold assumption). Semi-supervised methods devise various inductive biases that exploit such regularities for learning from unlabeled data.

Entropy minimization [36] encourages high confidence in unlabeled inputs. Such designs push decision boundaries towards low-density regions, under assumptions of clustered data and prediction smoothness. Pseudo-label training (or self-training) [37–39] also encourages high confidence (because of hard pseudo-labels) as well as consistency with a previously trained teacher. The basic forms of such algorithms do not achieve competitive performance on their own [40], but can be effective in conjunction with other approaches [4,41]. Pseudo-labels can be made very effective by confidence-based selection and other processing [37,38,42]. Note that some concurrent work [42] uses the term pseudo-label as a synonym for processed teacher prediction in one-way consistency, but we do not follow this practice.

Many approaches exploit the smoothness assumption by enforcing prediction consistency across different versions of the same input or different model instances. Introducing knowledge about equivariance has been studied for understanding and learning useful image representations [43,44] and improving dense prediction [45–47]. Exemplar training [13] associates patches with their original images (each image is a separate surrogate class). Temporal ensembling [21] enforces per-datapoint consistency between the current prediction and a moving average of past predictions. Mean Teacher [3] encourages consistency with a teacher whose parameters are an exponential moving average of the student's parameters.

Clusterization of latent representations can be promoted by penalizing walks which start in a labeled example, pass over an unlabeled example, and end in another example with a different label [48]. PiCIE [46] obtains semantically meaningful segmentation without labels by jointly learning clustering and representation consistency under photometric and geometric perturbations.

MixMatch [49] encourages consistency between predictions in different MixUp perturbations of the same input. The average prediction is used as a pseudo-label for all variants of the input. Deep co-training [19] produces complementary models by encouraging them to be consistent while each is trained on adversarial examples of the other one.

Consistency losses may encourage trivial solutions, where all inputs give rise to the same output. This is not much of a problem in semi-supervised learning since there the trivial solution is inhibited through the supervised objective. Interestingly, recent work shows that a variant of simple one-way consistency evades trivial solutions even in the context of self-supervised representation learning [50,51].

Virtual adversarial training (VAT) [4] encourages one-way consistency between predictions in original datapoints and their adversarial perturbations. These perturbations are recovered by maximizing a quadratic approximation of the prediction divergence in a small L^2 ball around the input. Better performance is often obtained by additionally encouraging low-entropy predictions [36]. Unsupervised data augmentation (UDA) [5] also uses a one-way consistency loss. FixMatch [52] shows that pseudo-label selection and processing can be useful in a one-way consistency. However, instead of adversarial additive perturbations, they use random augmentations generated by RandAugment. Different from all previous approaches, we explore an exhaustive set of consistency formulations.

2.3. Semi-Supervised Semantic Segmentation

In the classic semi-supervised GAN (SGAN) setup, the classifier also acts as a discriminator which distinguishes between real data (both labeled and unlabeled) and fake data produced by the generator [14]. This approach has been adapted for dense prediction by expressing the discriminator as a segmentation network that produces dense C+1-way logits [6]. KE-GAN [53] additionally enforces semantic consistency of neighbouring predictions by leveraging label-similarity recovered from a large text corpus (MIT ConceptNet). A semantic segmentation model can also be trained as a GAN generator (AdvSemSeg) [7]. In this setup, the discriminator guesses whether its input is ground truth or generated by the segmentation network. The discriminator is also used to choose better predictions for use as pseudo-labels for semi-supervised training. s4GAN + MLMT [8] additionally post-processes the recovered dense predictions by emphasizing classes identified by an image-wide classifier trained with Mean Teacher [3]. The authors note that the image-wide classification component is not appropriate for datasets such as Cityscapes, where almost all images contain a large number of classes.

A recent approach enforces consistency between outputs of redundant decoders with noisy intermediate representations [54]. Other recent work studies pseudo-labeling in the dense prediction context [55–57]. Zhu et al. [55] observe advantages in hard pseudo-labels. A recent approach [20] proposes a two-way consistency loss, which is related to the Π-model [21], and perturbs both inputs with geometric warps. However, we show that perturbing only the student branch generalizes better and has a smaller training footprint. A concurrent work [58] successfully applies a contrastive loss [59,60] between two branches which receive overlapping crops, and proposes a pixel-dependent consistency direction. Mean Teacher consistency with CutMix perturbations achieved state-of-the-art performance on half-resolution Cityscapes [25] prior to this work. Different than most presented approaches and similar to [25,55,56,61], our method does not increase the training footprint [12]. In comparison with [55,56,61], our teacher is updated in each training step, which eliminates the need for multiple training episodes. In comparison with [25], this work proposes a perturbation model which results in better generalization and shows that simple one-way consistency can be competitive with Mean Teacher. None of the previous approaches addresses semi-supervised training of efficient dense prediction models. We examine the simplest forms of consistency, explain advantages of perturbing only the student with respect to other forms of consistency, and propose a novel perturbation model. None of the previous approaches considered semi-supervised training of efficient dense-prediction models, nor studied composite perturbations of photometry and geometry.

3. Method

We formulate dense consistency as a mean pixel-wise divergence between corresponding predictions in the clean image and its perturbed version. We perturb images with a composition of photometric and geometric transformations. Photometric transformations do not disturb the spatial layout of the input image. Geometric transformations affect the spatial layout of the input image and the same kind of disturbance is expected at the model output. Ideally, our models should exhibit invariance to photometric transformations and equivariance [44] to the geometric ones.

3.1. Notation

We typeset vectors and arrays in bold, sets in blackboard bold, and we underline random variables. $P[\underline{y}|\underline{x} = x]$ denotes the distribution of a random variable $\underline{y}|\underline{x}$, while $P(y|x)$ is a shorthand for the probability $P(\underline{y} = y|\underline{x} = x)$. We denote the expectation of a function of a random variable as e.g., $\mathbf{E}_{\underline{\tau}} f(\underline{\tau})$. We use similar notation to denote the average over a set: $\mathbf{E}_{x \in \mathbb{D}} f(x)$. We use the Iverson bracket notation: given a statement P, $[\![P]\!] = 1$ if P is true; 0 otherwise. We denote cross-entropy with $H_{\underline{y}}(y^*) := \mathbf{E}_{\underline{y} \sim \underline{y}^*} \ln p(\underline{y} = y)$, and entropy with $H(\underline{y})$ [62]. We use Python-like array indexing notation.

We denote the labeled dataset as $\mathbb{D}_l$, and the unlabeled dataset as $\mathbb{D}_u$. We consider input images $x \in [0,1]^{H \times W \times 3}$ and dense labels $y \in \{1 \ldots C\}^{H \times W}$. A model instance maps an image to per-pixel class probabilities: $h_\theta(x)_{[i,j,c]} = P(y_{[i,j]} = c | x, \theta)$. For convenience, we identify output vectors of class probabilities with distributions: $h_\theta(x)_{[i,j]} \equiv P[\underline{y}_{[i,j]} | x, \theta]$.

3.2. Dense One-Way Consistency

We adapt one-way consistency [4,5] for dense prediction under our perturbation model $T_\tau = T_\gamma^G \circ T_\varphi^P$, where T_γ^G is a geometric warp, T_φ^P a per-pixel photometric perturbation, and $\tau = (\gamma, \varphi)$ perturbation parameters. T_γ^G displaces pixels with respect to a dense deformation field. The same geometric warp is applied to the student input and the teacher output. Figure 2 illustrates the computational graph of the resulting dense consistency loss. In simple one-way consistency, the teacher parameters θ' are a frozen copy of the student parameter θ. In Mean Teacher, θ' is a moving average of θ. In simple two-way consistency, both branches use the same θ and are subject to gradient propagation.
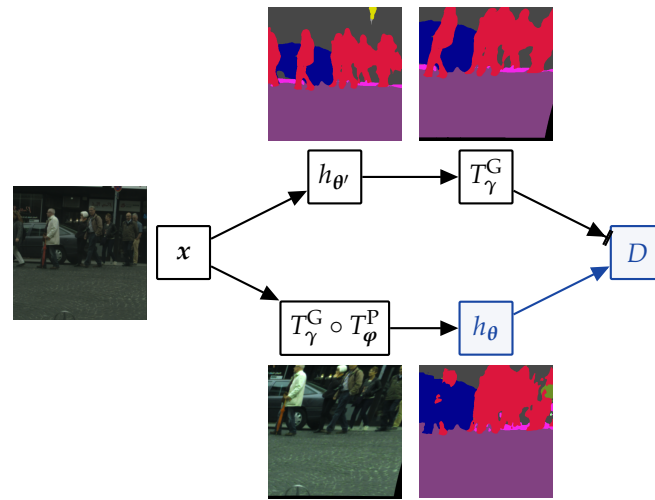
Figure 2. Dense one-way consistency with clean teacher. Top branch: the input is fed to the teacher $h_{\theta'}$. The resulting predictions are perturbed with geometric perturbations T_γ^G. Bottom branch: the input is perturbed with geometric and photometric perturbations and fed to the student h_θ. The loss D corresponds to the average pixel-wise KL divergence between the two branches. Gradients are computed only in the blue part of the graph.

A general semi-supervised training criterion $L(\theta; \mathbb{D}_l, \mathbb{D}_u)$ can be expressed as a weighted sum of a supervised term L_s over labeled data and an unsupervised consistency term L_c:

$$L(\theta; \mathbb{D}_l, \mathbb{D}_u) = \mathop{\mathbb{E}}_{(x,y) \in \mathbb{D}_l} L_s(\theta; x, y) + \alpha \mathop{\mathbb{E}}_{x \in \mathbb{D}_u} \mathop{\mathbb{E}}_\tau L_c(\theta; x, \tau). \tag{2}$$

In our experiments, L_s is the usual mean per-pixel cross entropy with L^2 regularization. We stochastically estimate the expectation over perturbation parameters τ with one sample per training step.

We formulate the unsupervised term L_c at pixel (i, j) as a one-way divergence D between the prediction in the perturbed image and its interpolated correspondence in the clean image. The proposed loss encourages the trained model to be equivariant to T_γ^G and invariant to T_φ^P:

$$L_c^{i,j}(\theta; x, \tau) = D\left(T_\gamma^G(h_{\theta'}(x))_{[i,j]}, h_\theta((T_\gamma^G \circ T_\varphi^P)(x))_{[i,j]}\right). \tag{3}$$

We use a validity mask $v^\gamma \in \{0,1\}^{H \times W}$, $v_{[i,j]}^\gamma = \left[\!\left[T_\gamma^G(\mathbf{1}_{H \times W})_{[i,j]} = 1 \right]\!\right]$ to ensure that the loss is unaffected by padding sampled from outside of $[1, H] \times [1, W]$. A vector produced by

$T_\gamma^G(h_\theta(x))_{[i,j]}$ represents a valid distribution wherever $v_{[i,j]}^\gamma = 1$. Finally, we express the consistency term L_c as a mean contribution over all pixels:

$$L_c(\theta; x, \tau) = \frac{1}{\sum(v^\gamma)} \sum_{i,j} v_{[i,j]}^\gamma L_c^{i,j}(\theta; x, \tau) . \tag{4}$$

Recall that the gradient is not computed with respect to θ'. Consequently, L_c allows gradient propagation only towards the perturbed image. We refer to such training as one-way consistency with clean teacher (and perturbed student). Such formulation provides two distinct advantages over other kinds of consistency. First, predictions in perturbed images are pulled towards predictions in clean images. This improves generalization when the perturbations are stronger than data augmentations used in L_s (cf. Figures 1 and 3). Second, we do not have to cache teacher activations during training since the gradients propagate only towards the student branch. Hence, the proposed semi-supervised objective does not constrain model complexity with respect to the supervised baseline.

We use KL divergence as a principled choice for D:

$$D(\underline{y}, \tilde{y}) := \underset{\underline{y}}{\mathbf{E}} \ln \frac{P(\underline{y} = y)}{P(\tilde{y} = y)} = H_{\tilde{y}}(\underline{y}) - H(\underline{y}) . \tag{5}$$

Note that the entropy term $-H(\underline{y})$ does not affect parameter updates since the gradients are not propagated through θ'. Hence, one-way consistency does not encourage increasing entropy of model predictions in clean images. Several researchers have observed improvement after adding an entropy minimization term [36] to the consistency loss [4,5]. This practice did not prove beneficial in our initial experiments.

Note that two-way consistency [20,21] would be obtained by replacing θ' with θ. It would require caching latent activations for both model instances, which approximately doubles the training footprint with respect to the supervised baseline. This would be undesirable due to constraining the feasible capacity of the deployed models [12,63].

We argue that consistency with clean teacher generalizes better than consistency with clean student since strong perturbations may push inputs beyond the natural manifold and spoil predictions (cf. Figure 1). Moreover, perturbing both branches sometimes results in learning to map all perturbed pixels to similar arbitrary predictions (e.g., always the same class) [64]. Figure 3 illustrates that consistency training has the best chance to succeed if the teacher is applied to the clean image, and the student learns on the perturbed image.
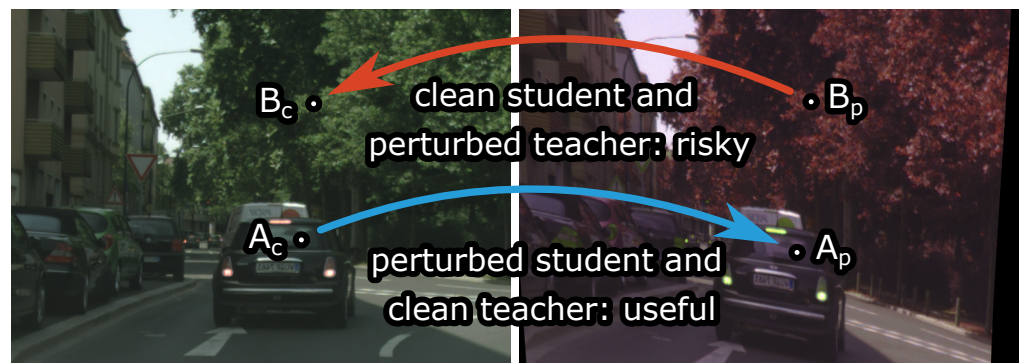
Figure 3. Two variants of one-way consistency training on a clean image (**left**) and its perturbed version (**right**). The arrows designate information flow from the teacher to the student. The proposed clean-teacher formulation trains in the perturbed pixels (A_p) according to the corresponding predictions in the clean image (A_c). The reverse formulation (training in B_c according to the prediction in B_p) worsens performance, since strongly perturbed images often give rise to less accurate predictions.

3.3. Photometric Component of the Proposed Perturbation Model

We express pixel-level photometric transformations as a composition of five simple perturbations with five image-wide parameters $\boldsymbol{\varphi} = (b, s, h, c, \boldsymbol{\pi})$. These perturbations are applied in each pixel in the following order: (1) brightness is shifted by adding b to all channels, (2) saturation is multiplied with s, (3) hue is shifted by addition with h, (4) contrast is modulated by multiplying all channels with c, and (5) RGB channels are randomly permuted according to $\boldsymbol{\pi}$. The resulting compound transformation $T_{\boldsymbol{\varphi}}^{\text{P}}$ is independently applied to all image pixels.

Our training procedure randomly picks image-wide parameters $\boldsymbol{\varphi}$ for each unlabeled image. The parameters are sampled as follows: $b \sim \mathcal{U}(-0.25, 0.25)$, $s \sim \mathcal{U}(0.25, 2)$, $h \sim \mathcal{U}(-36°, 36°)$, $c \sim \mathcal{U}(0.25, 2)$, and $\boldsymbol{\pi} \sim \mathcal{U}(\mathbb{S}_3)$, where $\mathbb{S}_3$ represents the set of all 6 3-element permutations.

3.4. Geometric Component of the Proposed Perturbation Model

We formulate a fairly general class of parametric geometric transformations by leveraging thin plate splines (TPS) [65,66]. We consider the 2D TPS warp $f \colon \mathbb{R}^2 \to \mathbb{R}^2$, which maps each image coordinate pair $\boldsymbol{q}$ to the relative 2D displacement of its correspondence $\boldsymbol{q}'$:

$$f(\boldsymbol{q}) = \boldsymbol{q}' - \boldsymbol{q} \,. \tag{6}$$

TPS warps minimize the bending energy (curvature) $\int_{\text{dom}(f)} \left\| \frac{\partial^2 f(\boldsymbol{q})}{\partial \boldsymbol{q}^2} \right\|_{\text{F}}^2 d\boldsymbol{q}$ given a set of control points and their displacements $\{(\boldsymbol{c}_i, \boldsymbol{d}_i) \colon i = 1 \ldots n\} \subset \mathbb{R}^2 \times \mathbb{R}^2$. In simple words, a TPS warp produces a smooth deformation field which optimally satisfies all constraints $f(\boldsymbol{c}_i) = \boldsymbol{d}_i$. In the 2D case, the solution of the TPS problem takes the following form:

$$f(\boldsymbol{q}) = \boldsymbol{A} \begin{bmatrix} 1 \\ \boldsymbol{q} \end{bmatrix} + \boldsymbol{W} \big[\phi(\|\boldsymbol{q} - \boldsymbol{c}_i\|) \big]_{i=1\ldots n}^{\mathsf{T}} , \tag{7}$$

where $\boldsymbol{q}$ denotes a 2D coordinate vector to be transformed, $\boldsymbol{A}$ is a 2×3 affine transformation matrix, $\boldsymbol{W}$ is a $2 \times n$ control point coefficient matrix, and $\phi(r) = r^2 \ln(r)$. Such a 2D TPS warp is equivariant to rotation and translation [66]. That is, $f(T(\boldsymbol{q})) = T(f(\boldsymbol{q}))$ for every composition of rotation and translation T.

TPS parameters $\boldsymbol{A}$ and $\boldsymbol{W}$ can be determined as a solution of a standard linear system which enforces deformation constraints $(\boldsymbol{c}_i, \boldsymbol{d}_i)$, and square-integrability of second derivatives of f. When we determine $\boldsymbol{A}$ and $\boldsymbol{W}$, we can easily transform entire images.

We first consider images as continuous domain functions and later return to images as arrays from $[0, 1]^{H \times W \times 3}$. Let $I \colon \text{dom}(I) \to [0, 1]^3$ be the original image of size (W, H), where $\text{dom}(I) = [0, W] \times [0, H]$. Then the transformed image I' can be expressed as

$$I'(\boldsymbol{q} + f(\boldsymbol{q})) = \begin{cases} I(\boldsymbol{q}), & \boldsymbol{q} \in \text{dom}(I), \\ \boldsymbol{0}, & \text{otherwise.} \end{cases} \tag{8}$$

The resulting formulation is known as forward warping [67] and is tricky to implement. We, therefore, prefer to recover the reverse transformation $\tilde{f}$, which can be conducted by replacing each control point $\boldsymbol{c}_i$ with $\boldsymbol{c}_i' = \boldsymbol{c}_i + \boldsymbol{d}_i$. Then, the transformed image is:

$$\tilde{I}(\boldsymbol{q}') = \begin{cases} I(\boldsymbol{q}' - \tilde{f}(\boldsymbol{q}')), & \boldsymbol{q}' - \tilde{f}(\boldsymbol{q}') \in \text{dom}(I), \\ \boldsymbol{0}, & \text{otherwise.} \end{cases} \tag{9}$$

This formulation is known as backward warping [67]. It can be easily implemented for discrete images by leveraging bilinear interpolation. Contemporary frameworks already include the implementations for the GPU hardware. Hence, the main difficulty is to determine the TPS parameters by solving two linear systems with $(n + 3) \times (n + 3)$ variables [66].

In our experiments, we use $n = 4$ control points corresponding to the centers of the four image quadrants: $(c_1', \dots, c_4') = \left(\left[\frac{1}{4}H, \frac{1}{4}W \right]^{\mathsf{T}}, \dots, \left[\frac{3}{4}H, \frac{3}{4}W \right]^{\mathsf{T}} \right)$. The parameters of our geometric transformation are four 2D displacements $\gamma = (d_1, \dots, d_4)$. Let f_γ denote the resulting TPS warp. Then, we can express our transformation as $T_\gamma^{\mathrm{G}}(x) = \mathrm{backward\_warp}(x, f_\gamma)$.

Our training procedure picks a random γ for each unlabeled image. Each displacement is sampled from a bivariate normal distribution $\mathcal{N}(\mathbf{0}_2, 0.05 \times H \times \mathbf{I}_2)$, where H is the height of training crops.

3.5. Training Procedure

Algorithm 1 sketches a procedure for recovering gradients of the proposed semi-supervised loss (2) on a mixed batch of labeled and unlabeled examples. For simplicity, we make the following changes in notation here: x_{l} and y_{l} are batches of size B_{l}, x_{u}, γ and φ batches of size B_{u}, and all functions are applied to batches. The algorithm computes the gradient of the supervised loss, discards cached activations, computes the teacher predictions, applies the consistency loss (3), and finally accumulates the gradient contributions of the two losses. Backpropagation through one-way consistency with clean teacher requires roughly the same extent of caching as in the supervised baseline. Hence, our approach constrains the model complexity much less than the two-way consistency.

Algorithm 1. Evaluation of the gradient of the proposed semi-supervised loss given perturbation parameters (γ, φ) on a mixed batch of labeled $(x_{\mathrm{l}}, y_{\mathrm{l}})$ and unlabeled (x_{u}) examples. CE denotes mean cross entropy, while KL_masked denotes mean KL divergence over valid pixels.

```
 1  # x_u, γ, φ, p_t, p_s, and v contain batches of B_u elements.
 2  # x_l, y, and p_l contain batches of B_l elements.
 3  procedure compute_loss_gradient(h, θ, x_l, y_l, x_u, γ, φ):
 4      θ' ← frozen_copy(θ) # simple one−way
 5
 6      # supervised loss
 7      p_l ← h_θ(x_l)
 8      L_s ← CE(y_l, p_l)
 9      g ← ∇_θ L_s # clears cached activations
10
11      # unsupervised loss
12      with no_grad(): # activations not cached here
13          p_t ← T_γ^G(h_θ'(x_u)) # clean teacher
14      p_s ← h_θ((T_γ^G ∘ T_φ^P)(x_u)) # perturbed student
15      v ← ⌊T_γ^G(1_{B_u×H×W})⌋ # validity mask
16      L_c ← α · KL_masked(p_t, p_s, v)
17      g ← g + ∇_θ L_c
18
19      return g
```
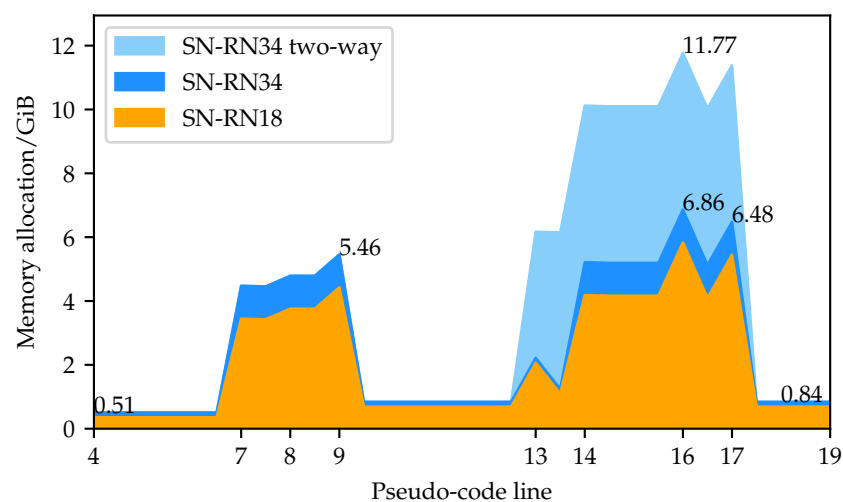

Figure 4 illustrates GPU memory allocation during a semi-supervised training iteration of a SwiftNet-RN34 model with one-way and two-way consistency. We recovered these measurements by leveraging the following functions of the `torch.cuda` package: `max_memory_allocated`, `memory_allocated`, `reset_peak_memory_stats`, and `empty_cache`. The training was carried out on a RTX A4500 GPU. Numbers on the x-axis correspond to lines of the pseudo-code in Algorithm 1. Line 9 backpropagates through the supervised loss and caches the gradients. The memory footprint briefly peaks due to temporary storage and immediately declines since PyTorch automatically releases all cached activations immediately after the backpropagation. Line 13 computes the teacher output. This step does not cache intermediate activations due to `torch.no_grad`. Line

16 computes the unsupervised loss, which requires the caching of activations on a large spatial resolution. The memory footprint briefly peaks since we delete perturbed inputs and teacher predictions immediately after line 16 (for simplicity, we omit opportunistic deletions from Algorithm 1). Line 17 triggers the backpropagation algorithm and accumulates the gradients of the consistency loss. The memory footprint briefly peaks due to temporary storage and immediately declines due to automatic deletion of the cached activations. At this point, the memory footprint is slightly greater than at line 4 since we still hold the supervised predictions in order to accumulate the recognition performance on the training dataset.

The ratio between memory allocations at lines 16 and 9 reveals the relative memory overhead of our semi-supervised approach. Note that the absolute overhead is model independent since it corresponds to the total size of perturbed inputs and predictions, and intermediate results of dense KL-divergence. On the other hand, the memory footprint of the supervised baseline is model dependent, since it reflects the computational complexity of the backbone. Consequently, the relative overhead approaches 1 as the model size increases, and is around 1.26 for SwiftNet-RN34.

Figure 4. GPU memory allocation during and after execution of particular lines from Algorithm 1 during the 2nd iteration of training. Our PyTorch implementations involve SwiftNet-RN18 and SwiftNet-RN34 models with one-way and two-way consistency, 768×768 crops, and batch sizes $(B_\mathrm{l}, B_\mathrm{u}) = (8, 8)$. Line 9 computes the supervised gradient. Line 13 computes the teacher output (without caching interemediate activations). Lines 16 and 17 compute the consistency loss and its gradient.

4. Results

Our experiments evaluate one-way consistency with clean teacher and a composition of photometric and geometric perturbations ($T_\gamma^G \circ T_\phi^P$). We compare our approach with other kinds of consistency and the state of the art in semi-supervised semantic segmentation. We denote simple one-way consistency as "simple", Mean Teacher [3] as "MT", and our perturbations as "PhTPS". In experiments that compare consistency variants, "1w" denotes one-way, "2w" denotes two way, "ct" denotes clean teacher, "cs" denotes clean student, and "2p" denotes both inputs perturbed. We present semi-supervised experiments in several semantic segmentation setups as well as in image-classification setups on CIFAR-10. Our implementations are based on the PyTorch framework [68].

4.1. Experimental Setup

Datasets.We perform semantic segmentation on Cityscapes [9], and image classification on CIFAR-10. Cityscapes contains 2975 training, 500 validation and 1525 testing images with resolution 1024×2048. Images are acquired from a moving vehicle during daytime

and fine weather conditions. We present half-resolution and full-resolution experiments. We use bilinear interpolation for images and nearest neighbour subsampling for labels. Some experiments on Cityscapes also use the coarsely labeled Cityscapes subset ("train-extra") that contains 19,998 images. CIFAR-10 consists of 50,000 training and 10,000 test images of resolution 32×32.

Common setup. We include both unlabeled and labeled images in $\mathbb{D}_{\mathrm{u}}$, which we use for the consistency loss. We train on batches of B_{l} labeled and B_{u} unlabeled images. We perform $\lfloor |\mathbb{D}_{\mathrm{l}}|/B_{\mathrm{l}} \rfloor$ training steps per epoch. We use the same perturbation model across all datasets and tasks (TPS displacements are proportional to image size), which is likely suboptimal [69]. Batch normalization statistics are updated only in non-teacher model instances with clean inputs except for full-resolution Cityscapes, for which updating the statistics in the perturbed student performed better in our validation experiments (cf. Appendix B). The teacher always uses the estimated population statistics, and does not update them. In Mean Teacher, the teacher uses an exponential moving average of the student's estimated population statistics.

Semantic segmentation setup. Cityscapes experiments involve the following models: SwiftNet with ResNet-18 (SwiftNet-RN18) or ResNet-34 (SwiftNet-RN34), and DeepLab v2 with a ResNet-101 backbone. We initialize the backbones with ImageNet pre-trained parameters. We apply random scaling, cropping, and horizontal flipping to all inputs and segmentation labels. We refer to such examples as clean. We schedule the learning rate according to $e \mapsto \eta \cos(e\pi/2)$, where $e \in [0 \ldots 1]$ is the fraction of epochs completed. This alleviates the generalization drop at the end of training with standard cosine annealing [70]. We use learning rates $\eta = 4 \times 10^{-4}$ for randomly initialized parameters and $\eta = 10^{-4}$ for pre-trained parameters. We use Adam with $(\beta_1, \beta_2) = (0.9, 0.99)$. The L^2 regularization weight in supervised experiments is 10^{-4} for randomly initialized and 2.5×10^{-5} for pre-trained parameters [27]. We have found that such L^2 regularization is too strong for our full-resolution semi-supervised experiments. Thus, we use a $4\times$ smaller weight there. Based on early validation experiments, we use $\alpha = 0.5$ unless stated otherwise. Batch sizes are $(B_{\mathrm{l}}, B_{\mathrm{u}}) = (8, 8)$ for SwiftNet-RN18 [27] and $(B_{\mathrm{l}}, B_{\mathrm{u}}) = (4, 4)$ for DeepLab v2 (ResNet-101 backbone) [26]. The batch size in corresponding supervised experiments is B_{l}.

In half-resolution Cityscapes experiments the size of crops is 448×448 and the logarithm of the scaling factor is sampled from $\mathcal{U}(\ln(1.5^{-1}), \ln(1.5))$. We train SwiftNet for $200 \times \frac{2975}{|\mathbb{D}_{\mathrm{l}}|}$ epochs (200 epochs or 74,200 iterations when all labels are used), and DeepLab v2 for $100 \times \frac{2975}{|\mathbb{D}_{\mathrm{l}}|}$ epochs (100 epochs or 74,300 iterations when all labels are used). In comparison with SwiftNet-RN18, DeepLab v2 incurs a 12-fold per-image slowdown during supervised training. However, it also requires less epochs since it has very few parameters with random initialization. Hence, semi-supervised DeepLab v2 trains more than $4\times$ slower than SwiftNet-RN18 on RTX 2080Ti. Appendix A.2 presents more detailed comparisons of memory and time requirements of different semi-supervised algorithms.

Our full-resolution experiments only use SwiftNet models. The crop size is 768×768 and the spatial scaling is sampled from $\mathcal{U}(2^{-1}, 2)$. The number of epochs is 250 when all labels are used. The batch size is 8 in supervised experiments, and $(B_{\mathrm{l}}, B_{\mathrm{u}}) = (8, 8)$ in semi-supervised experiments.

Appendix A.1 presents an overview and comparison of hyper-parameters with other consistency-based methods that are compared in the experiments.

Classification setup. Classification experiments target CIFAR-10 and involve the Wide ResNet model WRN-28-2 with standard hyper-parameters [71]. We augment all training images with random flips, padding and random cropping. We use all training images (including labeled images) in $\mathbb{D}_{\mathrm{u}}$ for the consistency loss. Batch sizes are $(B_{\mathrm{l}}, B_{\mathrm{u}}) = (128, 640)$. Thus, the number of iterations per epoch is $\lfloor \frac{|\mathbb{D}_{\mathrm{l}}|}{128} \rfloor$. For example, only one iteration is performed if $|\mathbb{D}_{\mathrm{l}}| = 250$. We run $1000 \times \frac{4000}{|\mathbb{D}_{\mathrm{l}}|}$ epochs in semi-supervised, and 100 epochs in supervised training. We use default VAT hyper-parameters $\xi = 10^{-6}$,

$\epsilon = 10$, $\alpha = 1$ [4]. We perform photometric perturbations as described, and sample TPS displacements from $\mathcal{N}(\mathbf{0}, 3.2 \times \mathbf{I}_2)$.

Evaluation. We report generalization performance at the end of training. We report sample means and sample standard deviations (with Bessel's correction) of the corresponding evaluation metric (mIoU or classification accuracy) of 5 training runs, evaluated on the corresponding validation dataset.

4.2. Semantic Segmentation on Half-Resolution Cityscapes

Table 1 compares our approach with the previous state of the art. We train using different proportions of training labels and evaluate mIoU on half-resolution Cityscapes val. The top section presents the previous work [7,8,25,57]. The middle section presents our experiments based on DeepLab v2 [26]. Note that here we outperform some previous work due to more involved training (as described in Section 4.1). since that would be a method of choice in all practical applications. Hence, we get consistently greater performance. We perform a proper comparison with [25] by using our training setup in combination with their method. Our MT-PhTPS outperforms MT-CutMix with L2 loss and confidence thresholding when 1/4 or more labels are available, while underperforming with 1/8 labels.

The bottom section involves the efficient model SwiftNet-RN18. Our perturbation model outperforms CutMix both with simple consistency, as well as with Mean Teacher. Overall, Mean Teacher outperforms simple consistency. We observe that DeepLab v2 and SwiftNet-RN18 get very similar benefits from the consistency loss. SwiftNet-RN18 comes out as a method of choice due to about $12\times$ faster inference than DeepLab v2 with ResNet-101 on RTX 2080Ti (see Appendix A.2 for more details). Experiments from the middle and the bottom section use the same splits to ensure a fair comparison.

Table 1. Semantic segmentation performance (mIoU/%) on half-resolution Cityscapes val after training with different proportions of labeled data. The top section reviews experiments from previous work. The middle section presents our experiments with DeepLab v2. The bottom section presents our experiments with SwiftNet-RN18. We run experiments across 5 different dataset splits and report mean mIoUs with standard deviations. The subscript "~[25]" denotes training with L^2 loss, confidence thresholding, and $\alpha = 1$, as proposed in [25]. The best results overall are bold, and best results within sections are underlined.

Method	Label Proportion			
	1/8	1/4	1/2	1/1
DLv2-RN101 supervised [8,25]	56.2	60.2	64.6[1]	66.0
DLv2-RN101 s4GAN+MLMT [8]	59.3	61.9	–	65.8
DLv2-RN101 supervised [7]	55.5	59.9	64.1	66.4
DLv2-RN101 AdvSemSeg [7]	58.8	62.3	65.7	67.7
DLv2-RN101 supervised [57]	56.0	60.5	–	66.0
DLv2-RN101 ECS [57]	$\underline{60.3}$	$\underline{63.8}$	–	$\underline{67.7}$
DLv2-RN101 MT-CutMix [25]	$\underline{60.3}_{1.2}$	$\underline{63.9}_{0.7}$	–	$\underline{67.7}_{0.4}$
DLv2-RN101 supervised	$56.4_{0.4}$	$61.9_{1.1}$	$66.6_{0.6}$	$69.8_{0.4}$
DLv2-RN101 MT-CutMix~[25]	$\mathbf{63.2}_{1.4}$	$65.6_{0.8}$	$67.6_{0.4}$	$70.0_{0.3}$
DLv2-RN101 MT-PhTPS	$61.5_{1.0}$	$\mathbf{66.4}_{1.1}$	$\mathbf{69.0}_{0.6}$	$\mathbf{71.0}_{0.7}$
SN-RN18 supervised	$55.5_{0.9}$	$61.5_{0.5}$	$66.9_{0.7}$	$70.5_{0.6}$
SN-RN18 simple-CutMix	$59.8_{0.5}$	$63.8_{1.2}$	$67.0_{1.4}$	$69.3_{1.1}$
SN-RN18 simple-PhTPS	$60.8_{1.6}$	$64.8_{1.5}$	$\mathbf{68.8}_{0.7}$	$71.1_{0.9}$
SN-RN18 MT-CutMix~[25]	$61.6_{0.9}$	$64.6_{0.5}$	$67.6_{0.7}$	$69.9_{0.6}$
SN-RN18 MT-CutMix	$59.3_{1.3}$	$63.3_{1.0}$	$66.8_{0.6}$	$69.7_{0.5}$
SN-RN18 MT-PhTPS	$\underline{62.0}_{1.3}$	$\mathbf{66.0}_{1.0}$	$69.1_{0.5}$	$\mathbf{71.2}_{0.7}$

Now, we present ablation and hyper-parameter validation studies for simple-PhTPS consistency with SwiftNet-RN18. Table 2 presents ablations of the perturbation model, and

also includes supervised training with PhTPS augmentations in one half of each mini-batch in addition to standard jittering. Perturbing the whole mini-batch with PhTPS in supervised training did not improve upon the baseline. We observe that perturbing half of each mini-batch with PhTPS in addition to standard jittering improves the supervised performance, but quite less than semi-supervised training. Semi-supervised experiments suggest that photometric perturbations (Ph) contribute most, and that geometric perturbations (TPS) are not useful when there is 1/2 or more of the labels.

Table 2. Ablation experiments on half-resolution Cityscapes val (mIoU/%) with SwiftNet-RN18. Subscripts denote the difference from the supervised baseline. The label "supervised PhTPS-aug" denotes supervised training where half of each mini-batch is perturbed with PhTPS. The bottom three rows compare PhTPS with Ph (only photometric) and TPS (only geometric) under simple one-way consistency. We present means of experiments on 5 different dataset splits. Numerical subscripts are differences with respect to the supervised baseline.

Method	Label Proportion			
	1/8	1/4	1/2	1/1
SN-RN18 supervised	55.5	61.5	66.9	70.5
SN-RN18 supervised PhTPS-aug	$56.2_{+1.5}$	$62.2_{+0.7}$	$67.4_{+0.5}$	$70.4_{-0.1}$
SN-RN18 simple-Ph	$59.2_{+3.7}$	$64.9_{+3.4}$	$68.3_{+1.4}$	$71.8_{+1.3}$
SN-RN18 simple-TPS	$58.2_{+3.1}$	$63.4_{+1.9}$	$66.7_{-0.2}$	$70.1_{-0.4}$
SN-RN18 simple-PhTPS	$60.8_{+5.3}$	$64.8_{+3.3}$	$68.8_{+1.9}$	$71.1_{+0.6}$

Figure 5 shows perturbation strength validation using 1/4 of the labels. Rows correspond to the factor that multiplies the standard deviation of control point displacements s_G defined at the end of Section 3.4. Columns correspond to the strength of the photometric perturbation s_P. The photometric strength s_P modulates the random photometric parameters according to the following expression:

$$(\underline{b}, \underline{s}, \underline{h}, \underline{c}) \mapsto (s_P \cdot \underline{b}, \exp(s_P \cdot \ln(\underline{s})), s_P \cdot \underline{h}, \exp(s_P \cdot \ln(\underline{c})) \ . \tag{10}$$

We set the probability of choosing a random channel permutation as $\min\{s_P, 1\}$. Hence, $s_P = 0$ corresponds to the identity function. Note that the "1/4" column in Table 2 uses the same semi-supervised configurations with strengths $s_G, s_P \in \{0, 1\}$. Moreover, note that the case $(s_G, s_P) = (0, 0)$ is slightly different from supervised training in that batch normalization statistics are still updated in the student. The differences in results are due to variance—the estimated standard error of the mean of 5 runs is between 0.35 and 0.5. We can observe that the photometric component is more important, and that a stronger photometric component can compensate for a weaker geometric component. Our perturbation strength choice $(s_G, s_P) = (1, 1)$ is close to the optimum, which the experiments suggests to be at $(1, 0.5)$.

Figure 6 shows our validation of the consistency loss weight α with SN-RN18 simple-PhTPS. We observe the best generalization performance for $\alpha \in [0.25 \ldots 0.75]$. We do not scale the learning rate with $(1 + \alpha)^{-1}$ because we use a scale-invariant optimization algorithm.
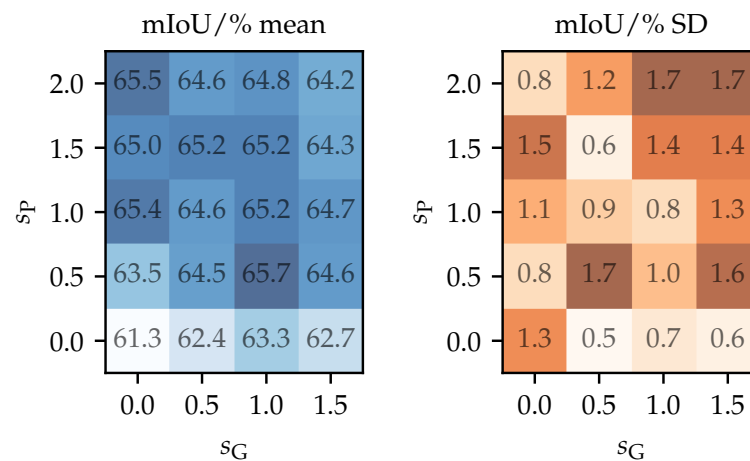
Figure 5. Validation of perturbation strength hyper-parameters on Cityscapes val (mIoU/%). We use 5 different subsets with 1/4 of the total number of training labels. The hyper-parameters s_P (photometric) and s_G (geometric) are defined in the main text. SD denotes the sample standard deviation.
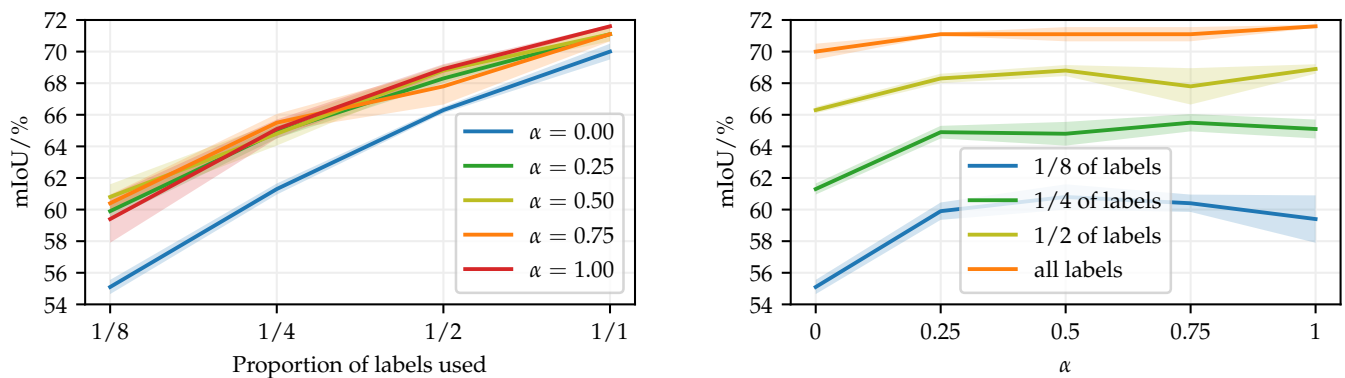

Figure 6. Validation of the consistency loss weight α on Cityscapes val (mIoU/%). We present the same results in two plots with different x-axes: the proportion of labels (**left**), and the consistency loss weight α (**right**).

Appendix B presents experiments that quantify the effect of updating batch normalization statistics when the inputs are perturbed.

Figure 7 shows qualitative results on the first few validation images with SwiftNet-RN18 trained with 1/4 of labels. We observe that our method displays a substantial resilience to heavy perturbations, such as those used during training.

	input	ground truth	simple-PhTPS	supervised
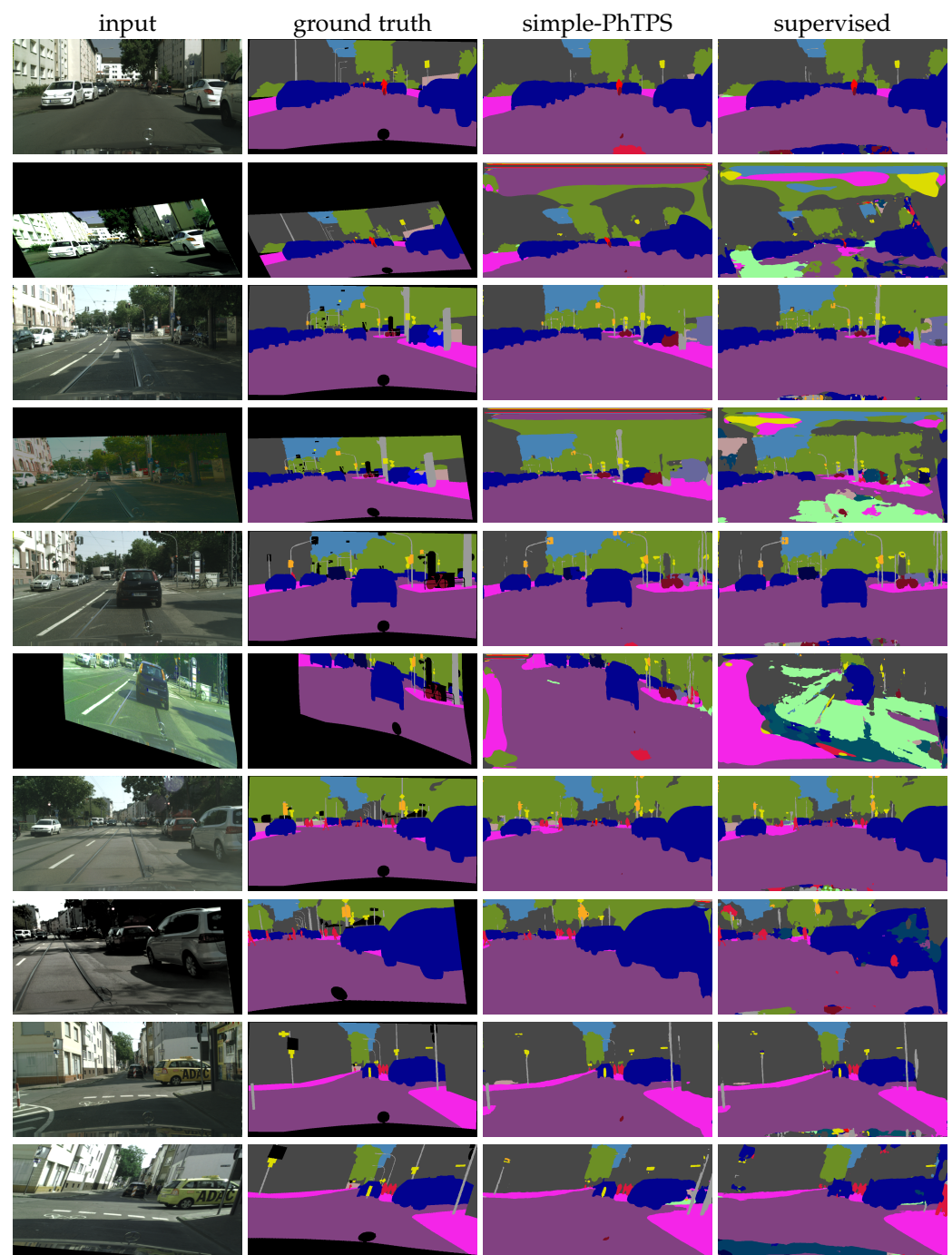
Figure 7. Qualitative results on the first few validation images with SwiftNet-RN18 trained with 1/4 of half-resolution Cityscapes labels. Odd rows contain unperturbed inputs, and even rows contain PhTPS perturbed inputs. The columns are (left to right): ground truth segmentations, predictions of simple-PhTPS consistency training, and predictions of supervised training.

4.3. Semantic Segmentation on Full-Resolution Cityscapes

Table 3 presents our full resolution experiments in setups such as in Table 1, and comparison with previous work, but with full-resolution images and labels. In comparison with KE-GAN [53] and ECS [57], we underperform with 1/8 labeled images, but outperform with 1/2 labeled images. Note that KE-GAN [53] also trains on a large text corpus (MIT ConceptNet) as well as that ECS DLv3$^+$-RN50 requires 22 GiB of GPU memory with batch size 6 [57], while our SN-RN18 simple-PhTPS requires less than 8 GiB of GPU memory with

batch size 8 and can be trained on affordable GPU hardware. Appendix A.2 presents more detailed memory and execution time comparisons with other algorithms.

We note that the concurrent approach DLv3$^+$-RN50 CAC [58] outperforms our method with 1/8 and 1/1 labels. However, ResNet-18 has significantly less capacity than ResNet-50. Therefore, the bottom section applies our method to the SwiftNet model with a ResNet-34 backbone, which still has less capacity than ResNet-50. The resulting model outperforms DLv3$^+$-RN50 CAC across most configurations. This shows that our method consistently improves when more capacity is available.

We note that training DLv3$^+$-RN50 CAC requires three RTX 2080Ti GPUs [58], while our SN-RN34 simple-PhTPS setup requires less than 9 GiB of GPU memory and fits on a single such GPU. Moreover, SN-RN34 has about $4\times$ faster inference than DLv3$^+$-RN50 on RTX 2080Ti.

Table 3. Semi-supervised semantic segmentation performance (mIoU/%) on full-resolution Cityscapes val with different proportions of labeled data. We compare simple-PhTPS and MT-PhTPS (ours) with supervised training and previous work. DLv3$^+$-RN50 stands for DeepLab v3$^+$ with ResNet-50, and SN for SwiftNet. We run experiments across 5 different dataset splits and report mean mIoUs with standard deviations. Best results overall are bold, and best results within sections are underlined.

Method	Label Proportion			
	1/8	**1/4**	**1/2**	**1/1**
KE-GAN [53]	66.9	70.6	72.2	75.3
DLv3$^+$-RN50 supervised [57]	63.2	68.4	72.9	74.8
DLv3$^+$-RN50 ECS [57]	67.4	70.7	72.9	74.8
DLv3$^+$-RN50 supervised [58]	63.9	68.3	71.2	76.3
DLv3$^+$-RN50 CAC [58]	**69.7**	<u>72.7</u>	−	<u>77.5</u>
SN-RN18 supervised	$61.1_{0.4}$	$67.3_{1.1}$	$71.9_{0.1}$	$75.4_{0.4}$
SN-RN18 simple-PhTPS	$66.3_{1.0}$	$71.0_{0.5}$	<u>$74.3_{0.7}$</u>	<u>$75.8_{0.4}$</u>
SN-RN18 MT-PhTPS	<u>$68.6_{0.6}$</u>	<u>$72.0_{0.3}$</u>	$73.8_{0.4}$	$75.0_{0.4}$
SN-RN34 supervised	$64.9_{0.8}$	$69.8_{1.0}$	$73.8_{1.4}$	$76.1_{0.8}$
SN-RN34 simple-PhTPS	$69.2_{0.8}$	$73.1_{0.7}$	**$76.3_{0.7}$**	**$77.9_{0.2}$**
SN-RN34 MT-PhTPS	**$70.8_{1.5}$**	**$74.3_{0.5}$**	$76.0_{0.5}$	$77.2_{0.4}$

Finally, we present experiments in the large-data regime, where we place the whole fine subset into $\mathbb{D}_l$. In some of these experiments, we also train on the large coarsely labeled subset. We denote the extent of supervision with subscripts "l" (labeled) and "u" (unlabeled). Hence, C_u in the table denotes the coarse subset without labels. Table 4 investigates the impact of the coarse subset on the SwiftNet performance on the full-resolution Cityscapes val. We observe that semi-supervised learning brings considerable improvement with respect to fully supervised learning on fine labels only (columns F_l vs. $F_l \cup C_u$). It is also interesting to compare the proposed semi-supervised setup ($F_l \cup C_u$) with classic fully supervised learning on both subsets (F_l, C_l). We observe that semi-supervised learning with SwiftNet-RN18 comes close to supervised learning with coarse labels. Moreover, semi-supervised learning prevails when we plug in the SwiftNet-RN34. These experiments suggest that semi-supervised training represents an attractive alternative to coarse labels and large annotation efforts.

Table 4. Effects of an additional large dataset on supervised and semi-supervised learning on full-resolution Cityscapes val (mIoU/%). Tags F and C denote fine and coarse subsets, respectively. Subset indices denote whether we train with labels (l) or one-way consistency (u).

Method	F_l	(F_l, F_u)	$(F_l, F_u \cup C_u)$	(F_l, C_l)
SN-RN18 simple-PhTPS	$75.4_{0.4}$	$75.8_{0.4}$	$76.5_{0.3}$	$\mathbf{76.9}_{0.3}$
SN-RN18 MT-PhTPS		$75.0_{0.4}$	$75.5_{0.3}$	
SN-RN34 simple-PhTPS	$76.1_{0.8}$	$77.9_{0.2}$	$\mathbf{78.5}_{0.4}$	$77.7_{0.4}$

4.4. Validation of Consistency Variants

Table 5 presents experiments with supervised baselines and four variants of semi-supervised consistency training. All semi-supervised experiments use the same PhTPS perturbations on CIFAR-10 (4000 labels and 50,000 images) and half-resolution Cityscapes (the SwiftNet-RN18 setups with 1/4 labels from Table 1). We investigate the following kinds of consistency: one-way with clean teacher (1w-ct, cf. Figure 1c), one-way with clean student (1w-cs, cf. Figure 1b), two-way with one clean input (2w-c1, cf. Figure 1d), and one-way with both inputs perturbed (1w-p2). Note that two-way consistency is not possible with Mean Teacher. Moreover, when both inputs are perturbed (1w-p2), we have to use the inverse geometric transformation on dense predictions [20]. We achieve that by forward warping [72] with the same displacement field. Two-way consistency with both inputs perturbed (2w-p2) is possible as well. We expect it to behave similarly to 1w-2p because it could be observed as a superposition of two opposite one-way consistencies, and our preliminary experiments suggest as much.

We observe that 1w-ct outperforms all other variants, while 2w-c1 performs in-between 1w-ct and 1w-cs. This confirms our hypothesis that predictions in clean inputs make better consistency targets. We note that 1w-p2 often outperforms 1w-cs, while always underperforming with respect to 1w-ct. A closer inspection suggests that 1w-p2 sometimes learns to cheat the consistency loss by outputting similar predictions for all perturbed images. This occurs more often when batch normalization uses the batch statistics estimated during training. A closer inspection of 1w-cs experiments on Cityscapes indicates the consistency cheating combined with severe overfitting to the training dataset.

Table 5. Comparison of 4 consistency variants under PhTPS perturbations: one-way with clean teacher (1w-ct), one-way with clean student (1w-cs), two-way with one input clean (2w-c1), and one-way with both inputs perturbed (1w-p2). Algorithms are evaluated on CIFAR-10 test (accuracy/%) while training on 4000 out of 50,000 labels (CIFAR-10, 2/25) and half-resolution Cityscapes val (mIoU/%) while training on 1/4 of labels from Cityscapes train with SwiftNet-RN18 (CS-half, 1/4).

Dataset	Method	sup.	1w-ct	1w-cs	2w-c1	1w-p2
CIFAR-10, 4k	WRN-28-2 simple-PhTPS	$80.8_{0.4}$	$\mathbf{90.8}_{0.3}$	$69.3_{4.2}$	$72.9_{2.6}$	$73.3_{7.0}$
CIFAR-10, 4k	WRN-28-2 MT-PhTPS	$80.8_{0.4}$	$\mathbf{90.8}_{0.4}$	$80.5_{0.5}$	-	$73.4_{1.4}$
CS-half, 1/4	SN-RN18 simple-PhTPS	$61.5_{0.5}$	$\mathbf{65.3}_{1.9}$	$1.6_{1.0}$	$16.7_{3.0}$	$61.6_{0.5}$
CS-half, 1/4	SN-RN18 MT-PhTPS	$61.5_{0.5}$	$\mathbf{66.0}_{1.0}$	$61.5_{1.4}$	-	$62.0_{1.1}$

4.5. Image Classification on CIFAR-10

Table 6 evaluates the image classification performance of two supervised baselines and 4 semi-supervised algorithms on CIFAR-10. The first supervised baseline uses only labeled data with standard data augmentation. The second baseline additionally uses our perturbations for data augmentation. The third algorithm is VAT with entropy minimization [4]. The simple-PhTPS approach outperforms supervised approaches and VAT. Again, two-way consistency results in the worst generalization performance. Perturbing the teacher input results in accuracy below 17% for 4000 or less labeled examples, and is not displayed. Note that somewhat better performance can be achieved by complementing consistency with

other techniques that are either unsuitable for dense prediction or out of the scope of this paper [5,49,69].

Table 6. Classification accuracy [%] on CIFAR-10 test with WRN-28-2. We compare two supervised approaches (top), VAT with entropy minimization [4] (middle), and two-way and one-way consistency with our perturbations (bottom three rows). We report means and standard deviations of 5 runs. The label "supervised PhTPS-aug" denotes the supervised training, where half of each mini-batch is perturbed with PhTPS.

Method	Number of Labeled Examples			
	250	**1000**	**4000**	**50,000**
supervised	$31.8_{0.6}$	$59.3_{1.4}$	$81.0_{0.2}$	94.7
supervised PhTPS-aug	$48.7_{0.9}$	$67.2_{0.5}$	$81.7_{0.2}$	95.0
VAT + entropy minimization	$41.0_{2.5}$	$73.2_{1.5}$	$84.2_{0.4}$	$90.5_{0.2}$
1w-cs simple-PhTPS	$27.7_{5.9}$	$51.7_{3.5}$	$69.3_{4.2}$	$91.6_{1.5}$
2w-c1 simple-PhTPS	$30.3_{1.8}$	$54.8_{1.5}$	$72.9_{2.6}$	$95.9_{0.2}$
1w-ct simple-PhTPS	$\mathbf{68.8}_{5.4}$	$\mathbf{84.2}_{0.4}$	$\mathbf{90.6}_{0.4}$	$\mathbf{96.2}_{0.2}$

5. Discussion

We have presented the first comprehensive study of one-way consistency for semi-supervised dense prediction, and proposed a novel perturbation model, which leads to the competitive generalization performance on Cityscapes. Our study clearly shows that one-way consistency with clean teacher outperforms other forms of consistency (e.g., clean student or two-way) both in terms of generalization performance and training footprint. We explain this by observing that predictions in perturbed images tend to be less reliable targets.

The proposed perturbation model is a composition of a photometric transformation and a geometric warp. These two kinds of perturbations have to be treated differently, since we desire invariance to the former and equivariance to the latter. Our perturbation model outperforms CutMix both in standard experiments with DeepLabv2-RN101 and in combination with recent efficient models (SwiftNet-RN18 and SwiftNet-RN34).

We consider two teacher formulations. In the simple formulation, the teacher is a frozen copy of the student. In the Mean Teacher formulation, the teacher is a moving average of student parameters. Mean Teacher outperforms simple consistency in low data regimes (half resolution; few labels). However, experiments with more data suggest that the simple one-way formulation scales significantly better.

To the best of our knowledge, this is the first account of semi-supervised semantic segmentation with efficient models. This combination is essential for many practical real-time applications where there is a lack of large datasets with suitable pixel-level groundtruth. Many of our experiments are based on SwiftNet-RN18, which behaves similarly to DeepLabv2-RN101, while offering about $9\times$ faster inference on half-resolution images, and about $15\times$ faster inference on full-resolution images on RTX 2080Ti. Experiments on Cityscapes coarse reveal that semi-supervised learning with one-way consistency can come close and exceed full supervision with coarse annotations. Simplicity, competitive performance and speed of training make this approach a very attractive baseline for evaluating future semi-supervised approaches in the dense prediction context.

Author Contributions: Conceptualization, M.O., I.G. and S.Š.; methodology, I.G., M.O. and S.Š.; software, I.G. and M.O.; validation, I.G. and M.O.; formal analysis, I.G.; investigation, I.G., M.O. and S.Š.; resources, S.Š.; data curation, I.G. and M.O.; writing—original draft preparation, I.G., M.O. and S.Š.; writing—review and editing, I.G. and S.Š.; visualization, I.G., M.O. and S.Š.; supervision, S.Š.; project administration, S.Š.; funding acquisition, S.Š. All authors have read and agreed to the published version of the manuscript.

Data Availability Statement: Data are available in a publicly accessible repository that does not issue DOIs and are provided by third parties. The datasets that we use are available at https://www.cityscapes-dataset.com/ and https://www.cs.toronto.edu/~kriz/cifar.html (accessed on 8 Jan 2023). Source code that enables reproducibility of our experiments is available at https://github.com/Ivan1248/semisup-seg-efficient.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

DLv2	DeepLab v2
DLv3$^+$	DeepLab v3$^+$
CS	Cityscapes
MT	Mean Teacher
PhTPS	Our composition of photometric and geometric perturbations
pp	Percentage point
RN	ResNet
simple-X	Simple one-way consistency with clean teacher, with perturbation model X
1w-ct	One-way consistency with clean teacher
1w-cs	One-way consistency with clean student
2w-c1	Two-way consistency with one input perturbed
1w-p2	One-way consistency with both inputs perturbed
SD	Sample standard deviation with Bessel's correction
SN	SwiftNet
TPS	Thin plate spline
WRN	Wide ResNet

Appendix A. Additional Algorithm Comparisons

Appendix A.1. Hyper-Parameters

Tables A1 and A2 review hyper-parameters of consistency-based semi-supervised algorithms from Tables 1 and 3.

Table A1. Overview of hyper-parameters of consistency-based algorithms for semi-supervised semantic segmentation. We denote the setup that we use in our experiments including supervised training with "our". We use the following symbols: y is the teacher prediction, $\tilde{y}$ is the student prediction, hard is a function that maps a vector representing a distribution to the closest one-hot vector ($\text{hard}_{[c]} = [\![c = \arg\max_k y_{[k]}]\!]$), e is the proportion of completed epochs, α is the consistency loss weight, and η is the base learning rate. The if-clause in the "Consistency loss" column represents confidence thresholding, i.e., determines whether the pixel is included in loss computation.

Model	Method	Crop Size	Jitter. Scale	Iterations	Epochs	B_l	B_u	Consistency Loss	α	Learning Rate Schedule
DLv2	MT-CutMix [25]	321	[0.5...1.5]	40,000	135 *	10	10	$\|\text{hard}(y) - \tilde{y}\|_2^2$ if $\max_c y_{[c]} > 0.97$	0.5	$\eta(1-e)^{0.9}$
DLv2	MT-CutMix~[25]	321	[0.5...1.5]	37,100	100	4	4	$\|\text{hard}(y) - \tilde{y}\|_2^2$ if $\max_c y_{[c]} > 0.97$	1	$\eta(1-e)^{0.9}$
DLv2	ours	448	[0.5...2]	74,200	200	4	4	$D(y, \tilde{y})$	0.5	$\eta(1-e)^{0.9}$
SN	ours	448	[0.5...2]	74,200	200	8	8	$D(y, \tilde{y})$	0.5	$\eta\cos(e\pi/2)$
SN	ours	768	[0.5...2]	92,750	250	8	8	$D(y, \tilde{y})$	0.5	$\eta\cos(e\pi/2)$
DLv3$^+$	CAC [58]	720	[0.5...1.5]	92,560	249 *	8	8	see [58]	$[\![e > 5/80]\!] \times 0.1$	$\eta(1-e)^{0.9}$
DLv3$^+$	ours	720	[0.5...1.5]	92,560	249 *	8	8	$D(y, \tilde{y})$	0.5	$\eta(1-e)^{0.9}$

* Some authors [25,58] use the word "epoch" to refer to a fixed number of iterations — 1000 and 1157 iterations, respectively.

Table A2. Optimizer hyper-parameter configurations for Cityscapes semantic segmentation experiments, represented in a PyTorch-like style.

Model	Method	Optimizer Configuration	
		Main	Backbone (Difference)
DLv2	MT-CutMix [25]	SGD, lr = 3×10^{-5}, momentum = 0.9, weight_decay = 5×10^{-4}	
DLv2	MT-CutMix~[25]	SGD, lr = 3×10^{-5}, momentum = 0.9, weight_decay = 5×10^{-4}	
DLv2	ours	Adam, betas = $(0.9, 0.99)$, lr = 4×10^{-4}, weight_decay = 1×10^{-4}	lr = 1×10^{-4}, weight_decay = 2.5×10^{-5}
SN	ours	Adam, betas = $(0.9, 0.99)$, lr = 4×10^{-4}, weight_decay = 2.5×10^{-5}	lr = 1×10^{-4}, weight_decay = 6.25×10^{-6}
DLv3$^+$	CAC [58]	SGD, lr = 1×10^{-1}, momentum = 0.9	lr = 1×10^{-2}, weight_decay = 1×10^{-4}
DLv3$^+$	ours	SGD, lr = 1×10^{-1}, momentum = 0.9	lr = 1×10^{-2}, weight_decay = 1×10^{-4}

Appendix A.2. Time and Memory Performance Characteristics

Table A3 shows memory requirements and training times of methods from Tables 1 and 3. The times include data loading and processing, and do not include the evaluation on the validation set. The memory measurements are based on the `max_memory_allocated` and `reset_peak_memory_stats` procedures from the `torch.cuda` package. Note that some overhead that is required by PyTorch is not included in this measurement (see PyTorch memory management documentation for more information: https://pytorch.org/docs/master/notes/cuda.html (accessed on 8 Jan 2023). Some algorithms did not fit into the memory of GTX 2080Ti. The memory allocations are higher than in Figure 4 because the supervised prediction, perturbed inputs, and perturbed outputs are unnecessarily kept in memory.

For DeepLabv3$^+$-RN50, we use the number of iterations, batch size, and crop size from [58]. Note, however, that the method from [58] has the memory requirements of two-way consistency because of per-pixel directionality.

Table A3. Half resolution Cityscapes (top section) and Cityscapes (bottom section) maximum memory allocation and training time on two GPUs.

| | | | | | | | Duration /min | |
Model	Method	Crop Size	Iterations	B_l	B_u	Memory /MiB	A4500	2080Ti
DLv2-RN101	MT-CutMix [25]	321	40,000	10	10	16,289	1067	–
	MT-CutMix~[25]	321	37,100	4	4	7037	794	1314
DLv2-RN101	supervised	448	74,300	4	–	6611	338	602
	MT-PhTPS	448	74,300	4	4	7021	816	1397
SN-RN18	supervised	448	74,200	4	–	1646	119	161
	simple-PhTPS	448	74,200	8	8	2398	228	279
	MT-PhTPS	448	74,200	8	8	2456	234	297
SN-RN18	supervised	768	92,750	8	–	4444	321	432
	simple-PhTPS	768	92,750	8	8	6683	732	963
	MT-PhTPS	768	92,750	8	8	6727	768	965
SN-RN34	supervised	768	92,750	8	–	5500	422	570
	simple-PhTPS	768	92,750	8	8	7737	994	1268
	MT-PhTPS	768	92,750	8	8	7818	1013	1276
DLv3+-RN50	supervised	720	92,560	8	–	11,645	1229	–
	simple-PhTPS	720	92,560	8	8	13,384	1884	–
	CAC [58]	720	92,560	8	8	25,165 †	>3000 *	–

† Estimated by running on NVidia A100. * The original implementation requires 36,005 MiB. Approximately 10.6 GiB can be saved by accumulating gradients as in Algorithm 1.

Table A4 shows the numbers of model parameters, and Table A5 shows inference speeds of models from Tables 1 and 3.

**Table A4.** Number of model parameters.

Model	Number of Parameters
DeepLabv2-RN101	43.80×10^6
DeepLabv3$^+$-RN50	40.35×10^6
SwiftNet-RN34	21.91×10^6
SwiftNet-RN18	11.80×10^6

Table A5. Model inference speed (number of iterations per second) on three different GPUs and two input resolutions. Inputs are processed one by one, without overlap in computation. The measurements include the computation of the cross-entropy loss, but do not include data loading and preparation.

|	1024×2048			512×1024		
	A4500	2080Ti	1080Ti	A4500	2080Ti	1080Ti
DeepLabv2-RN101	5.1	3.0	1.5	19.6	12.2	6.3
DeepLabv3+-RN50	16.1	9.6	5.2	54.2	30.7	23.5
SwiftNet-RN34	39.2	30.5	23.6	93.4	86.1	73.5
SwiftNet-RN18	56.5	45.3	34.6	139.5	115.8	98.4

Appendix B. Effect of Updating Batch Normalization Statistics in the Perturbed Student

Batch normalization layers estimate population statistics during training as exponential moving averages. Training on perturbed images can adversely affect the suitability of these estimates. Consequently, we explore the design choice to update the statistics only on clean inputs (when the supervised loss is computed), instead of both on clean and on perturbed inputs. Note that this configuration difference does not affect parameter optimization because the training always relies on mini-batch statistics.. Figure A1 shows the effect of disabling the updates of batch normalization statistics when the model instance (student) receives perturbed inputs in our semi-supervised training (one way consistency with clean teacher). The experiments are conducted according to the corresponding descriptions in Section 4. In case of half-resolution Cityscapes, disabling updates in the perturbed student (blue) increased the validation mIoU by between 0.3 and 1.4 pp, depending on the proportion of labels. However, in case of full-resolution Cityscapes, an opposite effect occured—mIoU decreased by between 0.1 and 1.1 pp. In CIFAR-10 experiments, the effect is mostly neutral.

(**a**)

Figure A1. *Cont.*

(b)

(c)
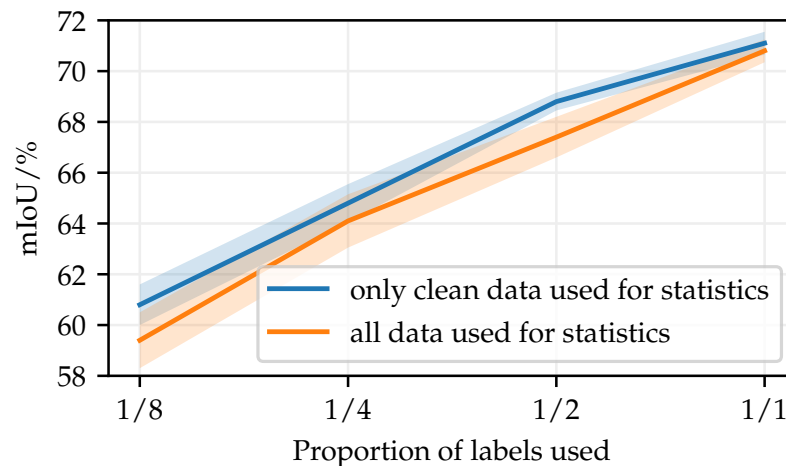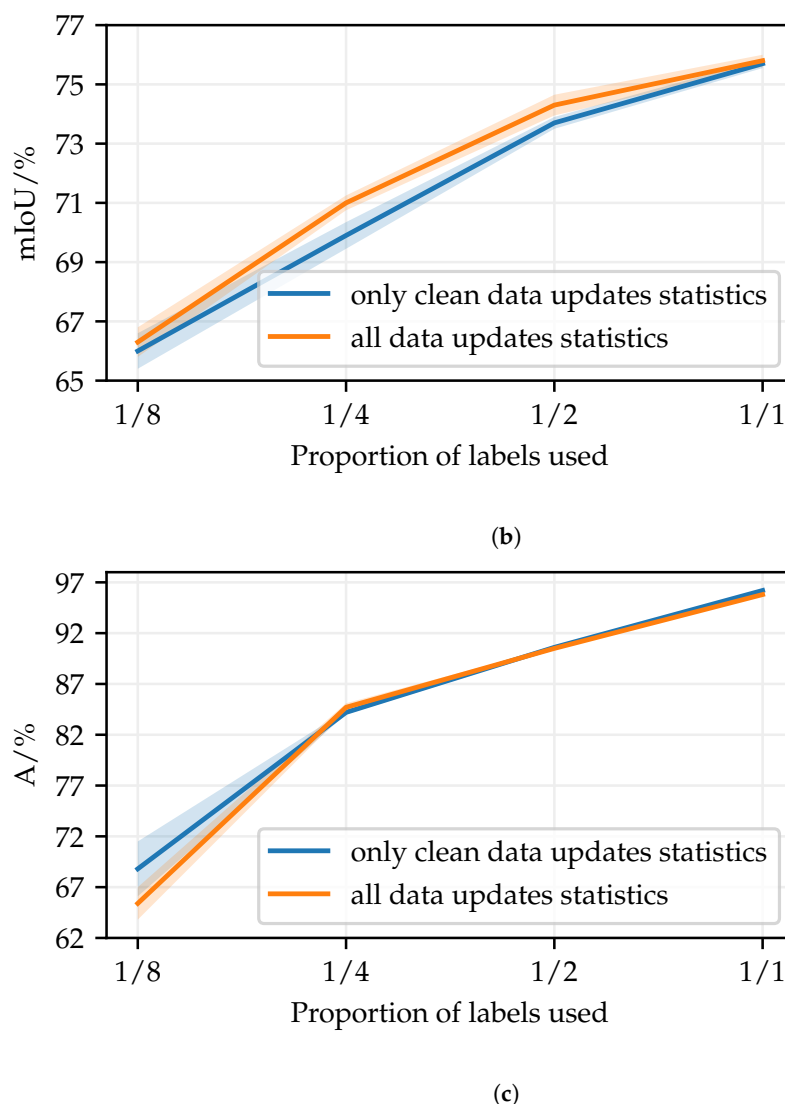
Figure A1. Effect of updating batch normalization statistics in the perturbed student. (**a**) Half-resolution Cityscapes val. (**b**) Cityscapes val. (**c**) CIFAR-10 validation set.

References

1. Kolesnikov, A.; Zhai, X.; Beyer, L. Revisiting Self-Supervised Visual Representation Learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, 16–20 June 2019; pp. 1920–1929.
2. Lee, J.; Kim, E.; Lee, S.; Lee, J.; Yoon, S. FickleNet: Weakly and Semi-Supervised Semantic Image Segmentation Using Stochastic Inference. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, 16–20 June 2019; pp. 5267–5276.
3. Tarvainen, A.; Valpola, H. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; pp. 1195–1204.
4. Miyato, T.; Maeda, S.; Koyama, M.; Ishii, S. Virtual Adversarial Training: A Regularization Method for Supervised and Semi-Supervised Learning. *IEEE Trans. Pattern Anal. Mach. Intell.* **2019**, *41*, 1979–1993. [CrossRef] [PubMed]
5. Xie, Q.; Dai, Z.; Hovy, E.H.; Luong, T.; Le, Q. Unsupervised Data Augmentation for Consistency Training. In Proceedings of the Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, Virtual, 6–12 December 2020; Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., Lin, H., Eds.; Curran Associates Inc.: Red Hook, NY, USA, 2020.
6. Souly, N.; Spampinato, C.; Shah, M. Semi Supervised Semantic Segmentation Using Generative Adversarial Network. In Proceedings of the IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, 22–29 October 2017; pp. 5689–5697.

7. Hung, W.; Tsai, Y.; Liou, Y.; Lin, Y.; Yang, M. Adversarial Learning for Semi-supervised Semantic Segmentation. In Proceedings of the BMVC, Newcastle, UK, 3–6 September 2018; p. 65.

8. Mittal, S.; Tatarchenko, M.; Brox, T. Semi-Supervised Semantic Segmentation with High- and Low-level Consistency. *IEEE Trans. Pattern Anal. Mach. Intell.* **2019**, *43*, 1369–1379. [CrossRef] [PubMed]

9. Cordts, M.; Omran, M.; Ramos, S.; Scharwächter, T.; Enzweiler, M.; Benenson, R.; Franke, U.; Roth, S.; Schiele, B. The Cityscapes Dataset. In Proceedings of the CVPRW, Boston, MA, USA, 7–12 June 2015.

10. Neuhold, G.; Ollmann, T.; Rota Bulò, S.; Kontschieder, P. Mapillary Vistas Dataset for Semantic Understanding of Street Scenes. In Proceedings of the ICCV, Venice, Italy, 22–29 October 2017; pp. 5000–5009.

11. Maggiori, E.; Tarabalka, Y.; Charpiat, G.; Alliez, P. Can semantic labeling methods generalize to any city? The inria aerial image labeling benchmark. In Proceedings of the 2017 IEEE International Geoscience and Remote Sensing Symposium, IGARSS 2017, Fort Worth, TX, USA, 23–28 July 2017; pp. 3226–3229.

12. Rota Bulò, S.; Porzi, L.; Kontschieder, P. In-Place Activated BatchNorm for Memory-Optimized Training of DNNs. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, 18–22 June 2018.

13. Dosovitskiy, A.; Springenberg, J.T.; Riedmiller, M.; Brox, T. Discriminative unsupervised feature learning with convolutional neural networks. In Proceedings of the Advances in Neural Information Processing Systems, hlMontreal, QC, Canada, 8–13 December 2014; pp. 766–774.

14. Salimans, T.; Goodfellow, I.J.; Zaremba, W.; Cheung, V.; Radford, A.; Chen, X. Improved Techniques for Training GANs. In Proceedings of the Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, Barcelona, Spain, 5–10 December 2016; pp. 2226–2234.

15. Tsai, Y.; Hung, W.; Schulter, S.; Sohn, K.; Yang, M.; Chandraker, M. Learning to Adapt Structured Output Space for Semantic Segmentation. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, 18–22 June 2018; pp. 7472–7481.

16. Gao, H.; Yao, D.; Wang, M.; Li, C.; Liu, H.; Hua, Z.; Wang, J. A Hyperspectral Image Classification Method Based on Multi-Discriminator Generative Adversarial Networks. *Sensors* **2019**, *19*, 3269. [CrossRef]

17. Rasmus, A.; Berglund, M.; Honkala, M.; Valpola, H.; Raiko, T. Semi-supervised Learning with Ladder Networks. In Proceedings of the Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, Montreal, QC, Canada, 7–12 December 2015; pp. 3546–3554.

18. Sajjadi, M.; Javanmardi, M.; Tasdizen, T. Mutual exclusivity loss for semi-supervised deep learning. In Proceedings of the Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, Barcelona, Spain, 5–10 December 2016; pp. 1163–1171.

19. Qiao, S.; Shen, W.; Zhang, Z.; Wang, B.; Yuille, A. Deep co-training for semi-supervised image recognition. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 135–152.

20. Bortsova, G.; Dubost, F.; Hogeweg, L.; Katramados, I.; de Bruijne, M. Semi-supervised Medical Image Segmentation via Learning Consistency Under Transformations. In Proceedings of the MICCAI, Shenzhen, China, 13–17 October 2019.

21. Laine, S.; Aila, T. Temporal Ensembling for Semi-Supervised Learning. In Proceedings of the 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, 24–26 April 2017.

22. Zheng, S.; Song, Y.; Leung, T.; Goodfellow, I.J. Improving the Robustness of Deep Neural Networks via Stability Training. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, 27–30 June 2016; pp. 4480–4488.

23. Krešo, I.; Krapac, J.; Šegvić, S. Efficient ladder-style densenets for semantic segmentation of large images. *IEEE Trans. Intell. Transp. Syst.* **2020**, *40*, 1369–1379.

24. Grubišić, I.; Oršić, M.; Šegvić, S. A baseline for semi-supervised learning of efficient semantic segmentation models. In Proceedings of the 17th International Conference on Machine Vision and Applications, MVA 2021, Aichi, Japan, 25–27 July 2021; pp. 1–5. [CrossRef]

25. French, G.; Laine, S.; Aila, T.; Mackiewicz, M.; Finlayson, G. Semi-supervised semantic segmentation needs strong, varied perturbations. In Proceedings of the BMVC, Virtual, 7–10 September 2020.

26. 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. *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**, *40*, 834–848. [CrossRef]

27. Oršić, M.; Krešo, I.; Bevandić, P.; Šegvić, S. In defense of pre-trained imagenet architectures for real-time semantic segmentation of road-driving images. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 12607–12616.

28. Schwartz, R.; Dodge, J.; Smith, N.A.; Etzioni, O. Green AI. *Commun. ACM* **2020**, *63*, 54–63. [CrossRef]

29. Long, J.; Shelhamer, E.; Darrell, T. Fully convolutional networks for semantic segmentation. In Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, 7–12 June 2015; pp. 3431–3440. [CrossRef]

30. Ronneberger, O.; Fischer, P.; Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Medical Image Computing and Computer-Assisted Intervention—MICCAI 2015, Proceedings of the 18th International Conference, Munich, Germany, 5–9 October 2015*; Navab, N., Hornegger, J., III, Wells, M.W., Frangi, A.F., Eds.; Lecture Notes in Computer Science; Proceedings, Part III; Springer: Cham, Switzerland, 2015; Volume 9351, pp. 234–241.

31. Zhao, H.; Qi, X.; Shen, X.; Shi, J.; Jia, J. ICNet for Real-Time Semantic Segmentation on High-Resolution Images. In Proceedings of the ECCV, Munich, Germany, 8–14 September 2018; Volume 11207, pp. 418–434.

32. Zhao, H.; Shi, J.; Qi, X.; Wang, X.; Jia, J. Pyramid Scene Parsing Network. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, 22–25 July 2017.

33. Oršić, M.; Šegvić, S. Efficient semantic segmentation with pyramidal fusion. *Pattern Recognit.* **2021**, *110*, 107611. [CrossRef]

34. Sun, K.; Xiao, B.; Liu, D.; Wang, J. Deep High-Resolution Representation Learning for Human Pose Estimation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, 18–20 June 2019.

35. Chapelle, O.; Schlkopf, B.; Zien, A. *Semi-Supervised Learning*, 1st ed.; The MIT Press: Cambridge, MA, USA, 2010.

36. Grandvalet, Y.; Bengio, Y. Semi-supervised Learning by Entropy Minimization. In *Proceedings of the Advances in Neural Information Processing Systems*; Saul, L.K., Weiss, Y., Bottou, L., Eds.; MIT Press: Cambridge, MA, USA, 2005; pp. 529–536.

37. Yarowsky, D. Unsupervised Word Sense Disambiguation Rivaling Supervised Methods. In Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics, Cambridge, MA, USA, 26–30 June 1995; Association for Computational Linguistics: Stroudsburg, PA, USA, 1995; pp. 189–196. [CrossRef]

38. McClosky, D.; Charniak, E.; Johnson, M. Effective Self-Training for Parsing. In Proceedings of the Human Language Technology Conference of the NAACL, Main Conference; New York, NY, USA, 4–9 June 2006; Association for Computational Linguistics: Stroudsburg, PA, USA, 2006; pp. 152–159.

39. hyun Lee, D. Pseudo-Label: The Simple and Efficient Semi-Supervised Learning Method for Deep Neural Networks. In Proceedings of the ICML 2013 Workshop: Challenges in Representation Learning (WREPL), Atlanta, GA, USA, 16–21 June 2013.

40. Oliver, A.; Odena, A.; Raffel, C.A.; Cubuk, E.D.; Goodfellow, I. Realistic Evaluation of Deep Semi-Supervised Learning Algorithms. In Proceedings of the Advances in Neural Information Processing Systems 32, NeurIPS 2018, Montréal, QC, Canada, 2–8 December 2018; Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., Garnett, R., Eds.; Curran Associates, Inc.: Red Hook, NY, USA, 2018; Volume 31.

41. Xie, Q.; Luong, M.; Hovy, E.H.; Le, Q.V. Self-Training with Noisy Student Improves ImageNet Classification. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, 13–19 June 2020; pp. 10684–10695.

42. Wang, Y.; Wang, H.; Shen, Y.; Fei, J.; Li, W.; Jin, G.; Wu, L.; Zhao, R.; Le, X. Semi-Supervised Semantic Segmentation Using Unreliable Pseudo-Labels. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, 18–24 June 2022; pp. 4238–4247.

43. Gerken, J.E.; Aronsson, J.; Carlsson, O.; Linander, H.; Ohlsson, F.; Petersson, C.; Persson, D. Geometric Deep Learning and Equivariant Neural Networks. *arXiv* **2021**, arXiv:2105.13926.

44. Lenc, K.; Vedaldi, A. Understanding Image Representations by Measuring Their Equivariance and Equivalence. *Int. J. Comput. Vis.* **2019**, *127*, 456–476. [CrossRef] [PubMed]

45. Wang, Y.; Zhang, J.; Kan, M.; Shan, S.; Chen, X. Self-supervised Equivariant Attention Mechanism for Weakly Supervised Semantic Segmentation. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, 13–19 June 2020.

46. Cho, J.H.; Mall, U.; Bala, K.; Hariharan, B. PiCIE: Unsupervised Semantic Segmentation using Invariance and Equivariance in Clustering. In Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2021, Virtual, 19–25 June 2021.

47. Patel, G.; Dolz, J. Weakly supervised segmentation with cross-modality equivariant constraints. *Med. Image Anal.* **2022**, *77*, 102374. [CrossRef]

48. Häusser, P.; Mordvintsev, A.; Cremers, D. Learning by Association—A Versatile Semi-Supervised Training Method for Neural Networks. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, 21–26 July 2017; pp. 626–635.

49. Berthelot, D.; Carlini, N.; Goodfellow, I.; Papernot, N.; Oliver, A.; Raffel, C.A. MixMatch: A Holistic Approach to Semi-Supervised Learning. In Proceedings of the Advances in Neural Information Processing Systems 33, NeurIPS 2019, Vancouver, BC, Canada, 8–14 December 2019; pp. 5049–5059.

50. Chen, X.; He, K. Exploring Simple Siamese Representation Learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2021, Nashville, TN, USA, 20–25 June 2021; Computer Vision Foundation/IEEE: New York, NY, USA, 2021; pp. 15750–15758.

51. Tian, Y.; Chen, X.; Ganguli, S. Understanding self-supervised learning dynamics without contrastive pairs. In Proceedings of the 38th International Conference on Machine Learning, ICML 2021, Virtual, 18–24 July 2021; Meila, M., Zhang, T., Eds.; Proceedings of Machine Learning Research; PMLR: Brookline, MA, USA, 2021; Volume 139, pp. 10268–10278.

52. Kurakin, A.; Li, C.L.; Raffel, C.; Berthelot, D.; Cubuk, E.D.; Zhang, H.; Sohn, K.; Carlini, N.; Zhang, Z. FixMatch: Simplifying Semi-Supervised Learning with Consistency and Confidence. In Proceedings of the Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, Virtual, 6–12 December 2020; Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., Lin, H., Eds.; Curran Associates Inc.: Red Hook, NY, USA, 2020.

53. Qi, M.; Wang, Y.; Qin, J.; Li, A. KE-GAN: Knowledge Embedded Generative Adversarial Networks for Semi-Supervised Scene Parsing. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, 18–20 June 2019; pp. 5237–5246.

54. Ouali, Y.; Hudelot, C.; Tami, M. Semi-Supervised Semantic Segmentation With Cross-Consistency Training. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, 13–19 June 2020.

55. Zhu, Y.; Zhang, Z.; Wu, C.; Zhang, Z.; He, T.; Zhang, H.; Manmatha, R.; Li, M.; Smola, A.J. Improving Semantic Segmentation via Self-Training. *arXiv* **2020**, arXiv:2004.14960.

56. Chen, L.; Lopes, R.G.; Cheng, B.; Collins, M.D.; Cubuk, E.D.; Zoph, B.; Adam, H.; Shlens, J. Naive-Student: Leveraging Semi-Supervised Learning in Video Sequences for Urban Scene Segmentation. In Proceedings of the Computer Vision—ECCV 2020—16th European Conference, Glasgow, UK, 23–28 August 2020; Vedaldi, A., Bischof, H., Brox, T., Frahm, J., Eds.; Lecture Notes in Computer Science; Proceedings, Part IX; Springer: London, UK, 2020; Volume 12354, pp. 695–714.

57. Mendel, R.; Souza, L., Jr.; Rauber, D.; Papa, J.; Palm, C. Semi-Supervised Segmentation based on Error-Correcting Supervision. In Proceedings of the Computer Vision—ECCV 2020—16th European Conference, Glasgow, UK, 23–28 August 2020.

58. Lai, X.; Tian, Z.; Jiang, L.; Liu, S.; Zhao, H.; Wang, L.; Jia, J. Semi-supervised Semantic Segmentation with Directional Context-aware Consistency. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, Virtual, 19–25 June 2021; Computer Vision Foundation/IEEE: New York, NY, USA, 2021.

59. van den Oord, A.; Li, Y.; Vinyals, O. Representation Learning with Contrastive Predictive Coding. *arXiv* **2018**, arXiv:1807.03748.

60. He, K.; Fan, H.; Wu, Y.; Xie, S.; Girshick, R. Momentum Contrast for Unsupervised Visual Representation Learning. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, 13–19 June 2020; pp. 9726–9735. [CrossRef]

61. Yang, L.; Zhuo, W.; Qi, L.; Shi, Y.; Gao, Y. ST++: Make Self-training Work Better for Semi-supervised Semantic Segmentation. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, 13–19 June 2022.

62. Olah, C. Visual Information Theory, 2015. Available online: https://colah.github.io/posts/2015-09-Visual-Information/ (accessed on 8 January 2023).

63. Huang, G.; Liu, Z.; Pleiss, G.; van der Maaten, L.; Weinberger, K.Q. Convolutional Networks with Dense Connectivity. *IEEE Trans. Pattern Anal. Mach. Intell.* **2022**, *44*, 8704–8716. [CrossRef] [PubMed]

64. Grill, J.B.; Strub, F.; Altché, F.; Tallec, C.; Richemond, P.; Buchatskaya, E.; Doersch, C.; Avila Pires, B.; Guo, Z.; Gheshlaghi Azar, M.; et al. Bootstrap Your Own Latent—A New Approach to Self-Supervised Learning. In Proceedings of the Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, Virtual, 6–12 December 2020; Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., Lin, H., Eds.; Curran Associates Inc.: Red Hook, NY, USA, 2020.

65. Duchon, J. Splines minimizing rotation-invariant semi-norms in Sobolev spaces. In *Constructive Theory of Functions of Several Variables*; Springer: Berlin/Heidelberg, Germany, 1977; pp. 85–100.

66. Bookstein, F.L. Principal Warps: Thin-Plate Splines and the Decomposition of Deformations. *IEEE Trans. Pattern Anal. Mach. Intell.* **1989**, *11*, 567–585. [CrossRef]

67. Szeliski, R. *Computer Vision: Algorithms and Applications*; Springer Science & Business Media: Berlin, Germany, 2010.

68. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In Proceedings of the Advances in Neural Information Processing Systems 33, NeurIPS 2019, Vancouver, BC, Canada, 8–14 December 2019; pp. 8024–8035.

69. Cubuk, E.D.; Zoph, B.; Shlens, J.; Le, Q. RandAugment: Practical Automated Data Augmentation with a Reduced Search Space. In Proceedings of the Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, Virtual, 6–12 December 2020; Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., Lin, H., Eds.; Curran Associates Inc.: Red Hook, NY, USA, 2020.

70. Loshchilov, I.; Hutter, F. SGDR: Stochastic Gradient Descent with Warm Restarts. In Proceedings of the 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, 24–26 April 2017.

71. Zagoruyko, S.; Komodakis, N. Wide Residual Networks. In Proceedings of the British Machine Vision Conference (BMVC) 2016, York, UK, 19–22 September 2016. [CrossRef]

72. Niklaus, S.; Liu, F. Softmax Splatting for Video Frame Interpolation. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, 13–19 June 2020.