Improving Semantic Segmentation via Efficient Self-Training

Yi Zhu, Zhongyue Zhang, Chongruo Wu, Zhi Zhang, Tong He, Hang Zhang, R. Manmatha, Mu Li, Alexander Smola

Abstract—Starting from the seminal work of Fully Convolutional Networks (FCN), there has been significant progress on semantic segmentation. However, deep learning models often require large amounts of pixelwise annotations to train accurate and robust models. Given the prohibitively expensive annotation cost of segmentation masks, we introduce a self-training framework in this paper to leverage pseudo labels generated from unlabeled data. In order to handle the data imbalance problem of semantic segmentation, we propose a centroid sampling strategy to uniformly select training samples from every class within each epoch. We also introduce a fast training schedule to alleviate the computational burden. This enables us to explore the usage of large amounts of pseudo labels. Our Centroid Sampling based Self-Training framework (CSST) achieves state-of-the-art results on Cityscapes and CamVid datasets. On PASCAL VOC 2012 test set, our models trained with the original train set even outperform the same models trained on the much bigger augmented train set. This indicates the effectiveness of CSST when there are fewer annotations. We also demonstrate promising few-shot generalization capability from Cityscapes to BDD100K and from Cityscapes to Mapillary datasets.

Index Terms—Semantic segmentation, semi-supervised learning, self-training, fast training schedule, cross-domain generalization

1 Introduction

Semantic segmentation is a fundamental computer vision task whose goal is to predict semantic labels for each pixel. Significant progress has been made in the last few years in part thanks to the collection of large and rich datasets with high quality human-annotated labels. However, pixel-by-pixel annotation is prohibitively expensive, e.g., labeling all pixels in a 1024 $\times$ 2048 Cityscapes image takes on average 1.5 hours [1]. Thus until now, we only have semantic segmentation datasets consisting of thousands or tens of thousands of annotated images [1], [2], [3], which are orders of magnitude smaller than datasets in other domains [4], [5], [6], [7].

Given the fact that learning with annotated samples alone is neither scalable nor generalizable, there is a surge of interest in using unlabeled data for semantic segmentation, such as video label propagation [8], [9], [10], [11], knowledge distillation [12], [13], adversarial learning [14], [15], consistency regularization [16], [17], and self-training [18], [19]. However, these methods often beat a competing baseline in their own settings but have difficulty achieving state-of-the-art performance on widely adopted benchmark datasets. There might be three possible challenges. First, most methods mentioned above are optimized with both labeled and unlabeled data in a single stage, and are likely limited by inaccurate predictions in the early training phase. Second, segmentation datasets are usually highly unbalanced in terms of number of pixels for each category. The dataset will become even more unbalanced when adding the unlabeled data. Without a proper way to handle data imbalance, the trained model may perform worse and does not generalize. Third, since segmentation model training is computationally heavy, much of the literature does not explore the usage of large amounts of unlabeled data. Hence, the performance improvement from using unlabeled data is marginal.

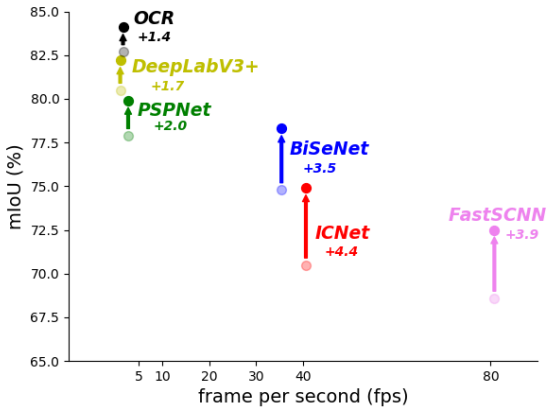


Fig. 1. Our self-training framework can enhance various methods without using additional labeled data or inference time. Each arrow shows the performance improvement obtained over the base model on the Cityscapes validation set.

In this paper, we would like to revisit semi-supervised learning in semantic segmentation. In order to address the first challenge, we propose to use a self-training framework based on [20], which is illustrated in Fig. 2. This framework has three stages: i) train a *teacher model* on a small set of labeled data ("real labels"), ii) generate "pseudo labels" on a large set of unlabeled data by using the teacher model, iii) a *student model* is trained using the combination of both real and pseudo labels. By training the teacher model with real labels alone in the first stage, the quality of our generated

- *Yi Zhu, Zhi Zhang, Tong He, R.Manmatha, Mu Li and Alexander Smola are with Amazon Web Services, Santa Clara, CA 95054. E-mail: {yzaws, zhiz, htong, manmatha, mli, smola}@amazon.com*
- *Zhongyue Zhang is with Snapchat. E-mail: {zzhang5}@snapchat.com*
- *Chongruo Wu is with UC Davis. E-mail: {crwu}@ucdavis.edu*
- *Hang Zhang is with Facebook. E-mail: {zhanghang}@fb.com*
- *This work was done while all the authors were at Amazon.*

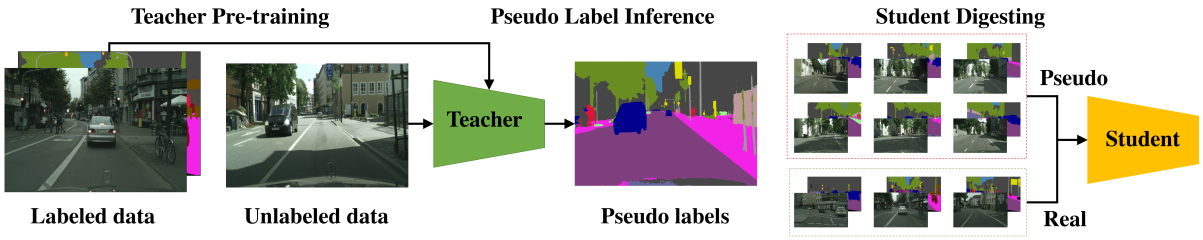


Fig. 2. Overview of our self-training framework. First, train a teacher model on labeled data. Second, generate pseudo labels on a large set of unlabeled data. Third, train a student model using the combination of both real and pseudo labels with the proposed centroid sampling strategy.

pseudo labels is guaranteed.

In terms of the second challenge, data imbalance, we introduce a centroid sampling strategy. The idea is to make sure that our model can see training examples from all classes with approximately equal chance in each epoch. Specifically, we store the class-level information in a dictionary and use it to flexibly adjust the sampling ratio from each class during training. In this manner, the data imbalance problem is alleviated.

Since we can generate unlimited pseudo labels as training samples, it significantly increases the training computational cost. A key reason that segmentation models are slow to train is because of high resolution input images. We thus propose a schedule that reduces the image resolution during training from time to time. A small resolution reduces both the computational cost per image and also allows a large batch for better system efficiency.

Equipped with Centroid Sampling, our Self-Training framework (CSST) is able to boost the performance of multiple models without using additional labeled data or inference time as shown in Fig. 1. In particular, our improved BiSeNet with CSST [21] even outperforms the original PSPNet [22] on the Cityscapes validation set. Note that BiSeNet is a real-time method with a ResNet18 backbone, while PSPNet is 15 times slower with a ResNet101 backbone. Furthermore, we can achieve 83.8 mIoU on the Cityscapes test set without using large-scale labeled dataset like Mapillary Vista [2] or Cityscapes Coarse [1]. These results clearly demonstrate the effectiveness of our framework in leveraging unlabeled data. Our contributions are summarized below:

- We introduce a self-training framework together with centroid sampling for semantic segmentation. Our method outperforms models pre-trained on large amounts of external labeled data.
- We introduce a fast training schedule that accelerates the model learning by 2x without losing accuracy. This helps us to explore the usage of large amounts of pseudo labels more efficiently.
- We extensively evaluate the proposed method to show its effectiveness on situations with more supervision (Cityscapes) and less supervision (CamVid), as well as challenging cross-domain generalization tasks from Cityscapes to Mapillary and from Cityscapes to BDD100K. We also evaluate on PASCAL VOC 2012 to show its effectiveness beyond street scene segmentation.

2 Related Work

Semantic segmentation. Starting from the seminal work of FCN [23], there has been significant progress in models for semantic segmentation [22], [24], [25], [26] . Recent work has focused more on exploiting object context by using attention [27], [28], [29], [30], [31], [32], designing more efficient networks [33], [34], [35], [36] and performing neural architecture search [37], [38], [39], [40].

Our work is different as we introduce a self-training framework for semantic segmentation to explore the benefit of using a large amount of unlabeled data. We demonstrate that our method is orthogonal to model development. We can improve a number of widely adopted models irrespective of the network architectures as shown in Fig. 1.

Semi-supervised learning. Recently we have witnessed the power of semi-supervised learning in the image classification domain. By using a large amount of unlabeled data, [41], [20] are able to achieve state-of-the-art performance on ImageNet. There are also numerous papers on semi-supervised semantic segmentation, such as video label propagation [8], [9], [10], [11], knowledge distillation [12], [13], adversarial learning [14], [15], consistency regularization [16], [17], etc.

In this paper, we revisit semi-supervised learning for semantic segmentation by using a teacher-student self-training framework. Our work is different from video label propagation methods [8], [9], [10], [11] because they use unlabeled data from the same domain, e.g., other unlabeled frames in the video sequence of the same dataset. We instead show that self-training works under relatively large domain shift, e.g., using unlabeled data from different datasets. Our work also differs from knowledge distillation [12], [13] since our student network may be bigger than the teacher. Compared to adversarial learning methods [14], [15] and consistency regularization methods [16], [17], our approach is simpler as we do not use an additional discriminator network or regularization techniques.

There are several concurrent papers [42], [43], [44], [45] adopting a similar self-training pipeline for semantic segmentation. However, they focus more on model development and do not discuss the impact brought by using pseudo labels. On the contrary, we study alternative formulations of self-training, propose a class-balanced data sampling strategy, experiment on large-scale pseudo labels, introduce a fast training technique and demonstrate strong cross-domain generalization performance.

Domain adaptation. Our cross-domain generalization task is related to unsupervised domain adaptation (UDA). The

goal of UDA is to learn a domain invariant feature representation to address the domain gap/shift problem. Numerous approaches have been proposed in this field, such as the dominating adversarial learning pipeline [46], [47], [48], [49], self-training [18], [19], [50], and consistency regularization [51], [52], [53].

Our work differs from conventional UDA, particularly the self-training ones [18], [19], [50], in multiple ways. First, our settings are different. Most of the papers in the UDA literature adopt a synthetic-to-real setting (e.g., GTA5/SYNTHIA to Cityscapes), while we consider a real-to-real setting (e.g., Cityscapes to Mapillary/BDD100K). Second, our framework is more generic and simpler. We do not use adversarial learning or specially designed loss functions to reduce the domain gap, and yet achieve decent generalization performance. Third, we show promising results on a challenging but practical task where the target domain has new classes and we have only a few labeled samples in the target domain. Most papers in the UDA literature do not consider this scenario. They [46], [47], [18] often assume there is no labeled data in the target domain and only report performance when the source and target domain have the same number of classes.

3 Methodology

In this section, we introduce our efficient self-training method for semantic segmentation. We first describe our teacher-student framework in Sec. 3.1. With the large amount of pseudo labels, we propose a centroid data sampling technique in Sec. 3.2 to address the data imbalance problem. Following this, we design a fast training schedule in Sec. 3.3 to handle the large expanded training set.

3.1 Self-Training using Unlabeled Data

Due to the difficulty and ambiguity of annotating segmentation masks, we introduce a teacher-student framework to perform self-training on semantic segmentation. Our goal is to use a small set of labeled data and a large quantity of unlabeled data to improve both the accuracy and robustness of the trained model. In this way, the human labeling effort can be largely reduced.

We present an overview of our self-training framework in Fig. 2. Given a set of labeled samples $(\mathbf{x}, \mathbf{y}) \in (\mathcal{X}, \mathcal{Y})$, where $\mathbf{x} \in \mathbb{R}^{H \times W \times 3}$ is an image and $\mathbf{y} \in \mathbb{R}^{H \times W \times C}$ is the corresponding human-annotated segmentation mask. H, W and C indicate the image height, width and number of classes. We first train a teacher model $\mathcal{T}$ with standard cross-entropy loss, computed as the inner product of one-hot ground-truth and the predicted softmax probability,

$$L_{\mathcal{T}} = -\sum_{i=1}^{N} \mathbf{y_i} \log\left(p_{\mathcal{T}}(\mathbf{x_i})\right). \tag{1}$$

N denotes the number of labeled samples. $\mathbf{y_i}$ is one-hot encoding of the class labels, while $p_{\mathcal{T}}$ represent the softmax predictions from the teacher model containing the class probabilities. We adopt a number of training techniques in [11] to make the teacher model as good as possible.

We then use the teacher model to generate pseudo labels $\mathbf{y}'$ on a large set of unlabeled images $\mathbf{x}'$ where $(\mathbf{x}', \mathbf{y}') \in (\mathcal{X}', \mathcal{Y}')$. In terms of semantic segmentation, we prefer to use hard pseudo labels because this could save significant disk space and training time,

$$\mathbf{y}' \sim \underset{C}{\operatorname{argmax}}\, p_{\mathcal{T}}(\mathbf{x}'). \tag{2}$$

Here, $\mathbf{y}'$ is also one-hot encoding of the class labels as $\mathbf{y}$ (hard pseudo label), which is sampled from the distribution of $p_{\mathcal{T}}(\mathbf{x}')$. We show several visual examples of our generated pseudo labels in Fig. 3 for better illustration. Given a teacher model trained on the Cityscapes fine dataset, we directly use it to predict segmentation masks on the Cityscapes coarse dataset and the Mapillary Vista dataset [2]. As can be seen, the quality of our teacher-generated pseudo labels on the Cityscapes coarse dataset is close to that produced by human annotators, thanks to our well-trained teacher model. Even for the Mapillary Vista dataset, our teacher model is able to provide reasonable segmentation predictions despite the large domain gap. In addition, we have investigated on using a confidence threshold to filter the hard pseudo labels as in [54], however, we do not observe performance improvement.

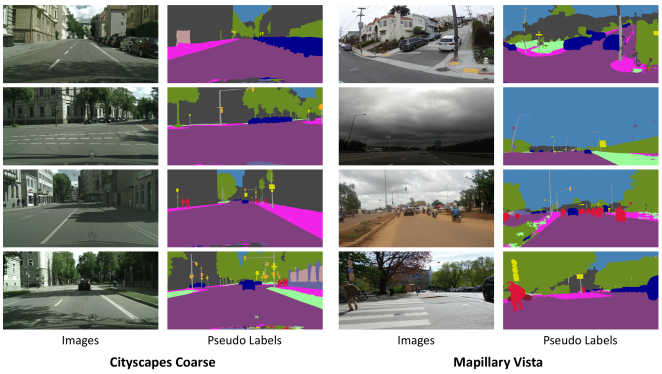


Fig. 3. Visualizations of our teacher-generated pseudo labels. Left: on Cityscapes coarse images. Right: on Mapillary images. Our teacher model is able to provide reasonable segmentation predictions.

Finally, we train a student model $\mathcal{S}$ using both real and pseudo labels. Similar to teacher training, we are effectively minimizing the cross-entropy,

$$L_{\mathcal{S}} = -\sum_{i=1}^{N} \mathbf{y_i} \log\left(p_{\mathcal{S}}(\mathbf{x_i})\right) - \sum_{j=1}^{M} \mathbf{y'_j} \log\left(p_{\mathcal{S}}(\mathbf{x'_j})\right). \tag{3}$$

M denotes the number of unlabeled samples. $p_{\mathcal{S}}$ represent the softmax predictions from the student model containing the class probabilities. By training on hard pseudo labels with cross-entropy loss, our self-training framework encourages the predicted class probabilities to be near one-hot, so the entropy of unlabeled data is minimized [55].

3.1.1 *Relationship to other self-training frameworks*

Our proposed method is a three-stage two-network pipeline, i.e., (1) train a teacher model $p_{\mathcal{T}}$, (2) generate hard pseudo labels and (3) train a separate student model $p_{\mathcal{S}}$. In this section, we would like to compare our method to several other self-training alternatives and discuss their relationships and differences. Experimental results will be presented in Sec. 4.3.3.

Consistency regularization Compared with our three-stage two-network pipeline, there is a simple form of joint teacher-student learning in the consistency regularization literature [56], [54], [18], [19]. To be specific, there is only one network p, and the network is optimized for both real and pseudo labels in one stage,

$$L_{consistency} = -\sum_{i=1}^{N} \mathbf{y_i} \log\left(p(\mathbf{x_i})\right) - \sum_{j=1}^{M} \mathbf{y'_j} \log\left(p(\mathbf{x'_j})\right). \quad (4)$$

Note that the pseudo labels $\mathbf{y'_j}$ are treated as hidden variables that can be learned throughout model training. This is different from our method because our pseudo labels are fixed once the teacher model converges in the first stage.

Teacher finetuning A separate student network may not be required in the last stage. We could continue to finetune the well-trained teacher model by using both real and pseudo labels. Mathematically, it has a similar formulation as Eq. (3), but without a new student network,

$$L_{finetune} = -\sum_{i=1}^{N} \mathbf{y_i} \log\left(p_{\mathcal{T}}(\mathbf{x_i})\right) - \sum_{j=1}^{M} \mathbf{y'_j} \log\left(p_{\mathcal{T}}(\mathbf{x'_j})\right). \quad (5)$$

This pipeline has a limitation, e.g., if a lightweight model is desired for edge deployment, this pipeline needs to train the lightweight model from the beginning. On the contrary, our method can directly start from stage three and may even benefit from teacher models trained in stage one due to knowledge distillation.

Soft pseudo labels Instead of using hard pseudo labels, we could directly use softmax predictions from the teacher model, i.e., soft pseudo labels. However, it is practically challenging for semantic segmentation due to the large computational cost. For example, saving the softmax predictions of all training images in the Cityscapes dataset requires $23500 \times 1024 \times 2048 \times 19 \approx 1$TB. This will significantly slow down data IO during training and obviously does not scale to scenarios with more data.

A straightforward solution is to save the top-k predictions instead of saving softmax scores for all classes. Then the student model can be trained similar to Eq. (3),

$$L_{soft} = -\sum_{i=1}^{N} \mathbf{y_i} \log\left(p_{\mathcal{S}}(\mathbf{x_i})\right) - \sum_{j=1}^{M} \mathbf{y'_{j-soft}} \log\left(p_{\mathcal{S}}(\mathbf{x'_j})\right). \quad (6)$$

Note that, the label $\mathbf{y'_{j-soft}}$ used is in softmax logits form, not in one-hot encoding form.

Model regularizer The benefit of using soft pseudo labels is avoid training the student model using overconfident teacher predictions. There is another alternative, confidence-regularized self-training (CRST) [19], which aims for the same goal by proposing both a label regularizer (LR) and a model regularizer (MR). We adopt the MR formulation because it is shown to achieve the best performance.

The intuition is to add a model regularizer in the loss function, which acts like an output smoothness encouraging term. To be specific, during student model training, we minimize

$$L_{\mathcal{S}} = -\sum_{i=1}^{N} \mathbf{y_i} \log\left(p_{\mathcal{S}}(\mathbf{x_i})\right) - \sum_{j=1}^{M} \mathbf{y'_j} \log\left(p_{\mathcal{S}}(\mathbf{x'_j})\right) - \sum_{j=1}^{M} \log p_{\mathcal{S}}(\mathbf{x'_j}). \quad (7)$$

We want to emphasize that the labels $\mathbf{y'_j}$ are still hard pseudo labels in the one-hot form. The added term $-\sum_{j=1}^{M} \log p_{\mathcal{S}}(\mathbf{x'_j})$ promotes output smoothness of the student model. This model regularizer term can effectively punish sharp predictions.

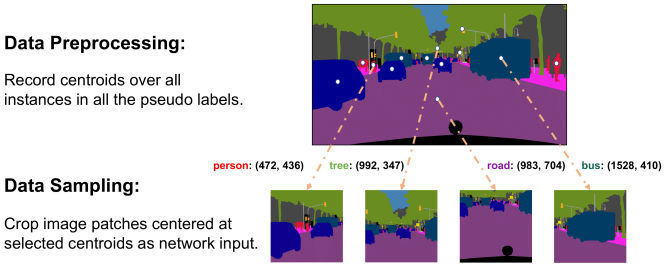


Fig. 4. Overview of centroid sampling. *Data preprocessing*: We first record the centroid of the areas containing the class of interest into a dictionary $\mathcal{D}$ before training, i.e., $\mathcal{D}$ = {bus: [(i,1528,410),$\cdots$], tree: [(i,992,347),$\cdots$], person: [(i,472,436),$\cdots$], road: [(i,983,704),$\cdots$], $\cdots$}, i denotes the index of the image. *Data sampling*: We then query training samples from $\mathcal{D}$ to achieve balanced training. Once we have the coordinates of selected centroids, we crop an image patch around each centroid and feed them to the network. Figure best viewed in color.

3.2 Fighting Data Imbalance with Centroid Sampling

A big challenge in semi-supervised semantic segmentation is how to deal with the data imbalance problem. Such imbalance often comes from two aspects. First, segmentation datasets are highly imbalanced in terms of the number of pixels for each category because real-world data follows a long-tail distribution. For example, in street scene segmentation datasets, we always see more pixels for classes such as road and sky, than classes such as train or motorcyle. Second, due to object shape and class confusion, different categories have different level of difficulties to be segmented. Especially for unlabeled data, a biased teacher tends to produce more label predictions with high confidence on easy classes, but fewer label predictions with low confidence on hard classes. This will aggravate the data imbalance problem. Hence, a better way to control the usage of pseudo labels in terms of handling imbalance is essential to make self-training work in the semantic segmentation domain.

Inspired by [25], [11], we introduce a centroid sampling strategy as shown in Fig. 4. The idea is to make sure that our model sees training examples from all classes with approximately equal chance in each epoch. To be specific, centroid sampling has two stages: data preprocessing and data sampling.

Data preprocessing: We first record the centroid of areas containing the class of interest into a dictionary $\mathcal{D}$ before training. A centroid is the arithmetic mean position of all

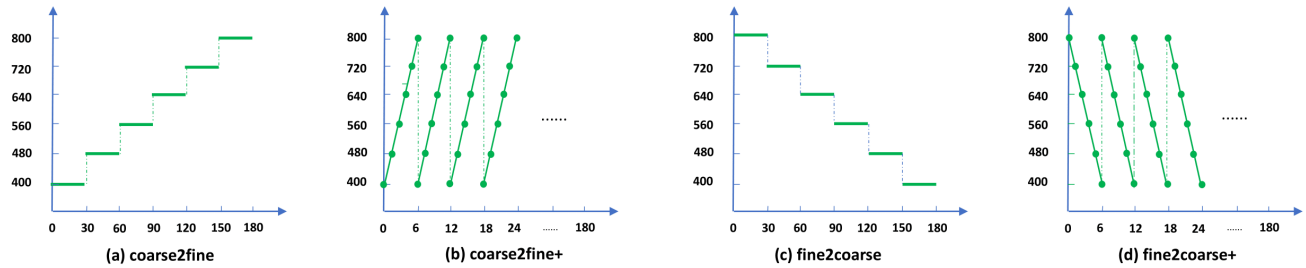


Fig. 5. Overview of proposed fast training schedules. (a) coarse2fine schedule uses small crop size during early training epochs and gradually increases the crop size. (b) coarse2fine+ iterates the crop size every epoch to maximize the scale variation during model training. (c) and (d) fine2coarse and fine2coarse+ are simply the reverse process of (a) and (b), respectively. x-axis: epoch number. y-axis: crop size. See texts in Sec. 3.3 for more details.

the points within an object (i.e., a connected component of a specific class in the annotated segmentation mask). For example in Fig. 4, the centroid of a car instance will be a point (white circle) roughly in the middle of the car. Note that, we could have multiple centroids in one image depending on how many semantic connected components exist in that image. In the end, the dictionary $\mathcal{D}$ will have C keys, where C is the number of classes. Each key contains a list with coordinates of all the centroids in the dataset belonging to that class.

Data sampling: Then we can query training samples using the class-level information from $\mathcal{D}$ to achieve balanced training. For example, if we have a total of N training samples, we hope to see a training instance from each class about N/C times in one epoch. During training, we simply select N/C centroids from each key in $\mathcal{D}$ to build an uniform epoch. If the key is an underrepresented class, which does not have N/C centroids, we will oversample existing ones. Once we have the coordinates of these centroids, we crop an image patch around each centroid and feed them to the network. As we can see in Fig. 4, these image patches will contain at least one class of interest as the centroid point. Note that in order to keep the training stable, we use large resolution image patches (e.g., 800×800) as a network input, which will potentially include other classes. We find this beneficial since our goal is to make sure the model can see training examples from all classes uniformly in each epoch, not to force the distinctiveness of each patch.

Our proposed centroid sampling technique is different from the class-uniform sampling method introduced in [11] because they only select underrepresented classes. We instead use all the classes because we believe it is better to train the model following the underlying data distribution. In addition, [11] generates fewer centroids per image because they do tile-based sampling, while we perform instance-based sampling.

3.3 Fast Training Schedule for Large Datasets

Once we have the pseudo labels and a principled way to handle data imbalance, it is time to kickoff training. However, training a semantic segmentation model consumes lots of computing resources. Due to the large crop size, we can only use a small batch size to fit into the GPU memory. Hence, even training on a medium-scale dataset using a high-end 8-GPU machine takes days to finish. Now if we increase the dataset size by 10 times or more, the training will take weeks to complete. This is part of the reason why self-training hasn't been investigated thoroughly in the field of semantic segmentation.

Given that the slowness is caused by using a small batch size, can we reduce the crop size during training to increase the batch size? Some studies [22], [24] have shown that reducing crop size hurts performance. Thus, trading crop size for batch size is not a good idea for semantic segmentation. This is because segmentation is a per-pixel dense prediction problem and a small crop size will lead to the problem of losing global context. However, what if we can design a training schedule that iterates between a small crop size and a larger crop size, so that the training time can be reduced without losing segmentation accuracy?

In this work, inspired by the coarse-to-fine concept in computer vision [57], we design several training schedules to speed up the experiments. Our goal is to achieve faster training without losing accuracy. We introduce four learning schedules, namely coarse2fine, fine2coarse, coarse2fine+ and fine2coarse+. To be specific, we use a Cityscapes experiment as an illustrative example, with 800×800 input crop size and 180 training epochs. As can be seen in Fig. 5, (a) coarse2fine means we first use small crop sizes in the early epochs, then change to larger and larger sizes for the rest of the training. Each crop size stays constant for several epochs, e.g., we change the crop size every 30 epochs. (b) coarse2fine+ means we iterate the crop size every epoch to maximize the scale variation during model learning. fine2coarse and fine2coarse+ are simply the reverse process of (a) and (b), respectively. We will show the performance of each learning schedule in Sec. 4.3.5, and demonstrate that the coarse2fine+ schedule is the best candidate. It is able to speed up the training by up to 2x without performance degradation.

We would like to point out that our fast training schedule is a general technique. It is not only suitable for the self-training framework with huge number of pseudo labels, but also applicable to standard supervised learning on large-scale datasets such as Mapillary [2] and BDD100K [3].

4 Experiments

We first describe the datasets and the implementation details. In Sec. 4.3, we perform ablation studies on Cityscapes and compare to the recent literature. Then in Sec. 4.4, we use

CamVid to demonstrate the effectiveness of our approach when there are few real labels. In addition, we evaluate on PASCAL VOC 2012 to validate our framework beyond street scene segmentation in Sec. 4.5. Finally in Sec. 4.6, we evaluate our model on two cross-domain generalization tasks, one is from Cityscapes to BDD100K, the other is from Cityscapes to Mapillary. For all the experiments, we use the standard mean Intersection over Union (mIoU) metric to report segmentation accuracy.

TABLE 1
Effectiveness of self-training and centroid sampling (CS).

(a) Self-training with pseudo labels

	Real	Pseudo	w/o CS	w CS
Teacher	3K	-	78.1	78.8
Student	3K	3K	78.6	79.3
Student	3K	6K	79.0	79.9
Student	3K	12K	79.2	80.0
Student	3K	40K	78.8	**80.2**

(b) Duplicate real labels

	Real	w/o CS	w CS
Teacher	3K	78.1	78.8
Teacher	6K	78.5	79.2
Teacher	9K	78.6	79.1
Teacher	15K	78.5	**79.3**
Teacher	43K	78.6	79.1

4.1 Datasets

Cityscapes [1] contains 5K high quality annotated images, split into 2975 training, 500 validation, and 1525 test images. The dataset defines 19 semantic labels and a background class. Cityscapes also has 20K coarsely annotated samples. In this paper, we refer to the 20K samples as the Cityscapes coarse dataset, and only use these images as unlabeled data (without their labels). **CamVid** [58] defines 32 semantic labels, however, most papers only focus on 11 of them. It includes 701 densely annotated images, split into 367 training, 101 validation and 233 test images. **PASCAL VOC 2012** [59] contains 20 object categories and one background class. Its original dataset has 1464 training, 1449 validation and 1456 test images. The dataset is augmented by 9K extra annotations provided by [60], resulting in 10582 training images. Here, we use the 9K images in the augmented set as unlabeled data to generate pseudo labels for self-training. **BDD100K** [3] has the same 19 semantic labels as Cityscapes, but is collected mainly in US. The dataset is split into 7K training, 1K validation and 2K test images. **Mapillary Vista** [2] is a recent large-scale benchmark with global reach and includes more variety. The dataset has 18K training, 2K validation and 5K test images. We use its research edition which contains 66 classes. We will refer to it as Mapillary for short in the rest of the paper.

4.2 Implementation Details

We employ the SGD optimizer for all the experiments. We set the initial learning rate to 0.02 for training from scratch and 0.002 for finetuning. We use a polynomial learning rate policy [61], where the initial learning rate is multiplied by $(1 - \frac{\text{epoch}}{\text{max\_epoch}})^{\text{power}}$ with a power of 0.9. Momentum and weight decay are set to 0.9 and 0.0001 respectively. Synchronized batch normalization [22] is used with a default batch size of 16. When using our fast training schedule, the batch size can increase to 64 when the crop size is smaller. The number of training epochs is set to 180 for Cityscapes, BDD100K and Mapillary, 90 for CamVid, and 50 for PASCAL VOC 2012. The crop size is set to 800 for Cityscapes, BDD100K and Mapillary, 640 for CamVid and 480 for PASCAL VOC 2012. For data augmentation, we perform random spatial scaling (from 0.5 to 2.0), horizontal flipping, Gaussian blur and color jittering (0.1) during training. We adopt the OHEM loss following [62]. For all the ablation experiments, we run the same training recipe five times and report the average mIoU.

4.3 Cityscapes

In this section, we break our framework apart and show the contributions from each one, namely self-training, centroid sampling and fast training. We also show comparisons to recent semi-supervised and supervised segmentation approaches on the Cityscapes dataset.

4.3.1 Effectiveness of self-training

Our baseline model (DeepLabV3+ with ResNeXt50 backbone) trained on the 3K Cityscapes fine annotations has an mIoU of 78.1% [11]. We adopt this model as the teacher to start self-training due to its good speed-accuracy tradeoff. We then use this model to generate pseudo labels on the Cityscapes coarse dataset and the Mapillary dataset. Note that the images in both datasets are annotated but we ignore the labels and treat them as unlabeled data. This leads to a total of 40K pseudo labels.

As can be seen in Table 1(a), using pseudo labels often improves the segmentation accuracy. With more and more pseudo labels, the performance keeps increasing, whether using centroid sampling or not. This suggests the potential of self-training for semantic segmentation: we can improve a strong baseline without using additional labeled data.

4.3.2 Effectiveness of centroid sampling

Here, we would like to show the performance improvement by handling data imbalance using centroid sampling. The results can be seen in Table 1(a), and we have several observations. First, using centroid sampling is always better than not using it. With all the 40K pseudo labels, the model trained with centroid sampling reaches an mIoU of 80.2%, which is 1.4% better than without it. Second, centroid sampling is essential to achieving good results in the situation that pseudo labels dominate the training set (e.g., 40K pseudo labels), otherwise the performance starts to drop ($79.2\% \rightarrow 78.8\%$ in the column for w/o CS). This is because we can avoid the biased teacher issue in self-training as mentioned in Sec. 3.2. Third, centroid sampling is a generic method to handle data imbalance, thus we can use it for teacher training as well. We see that there is an improvement of 0.7% ($78.1\% \rightarrow 78.8\%$).

However, one may argue that using pseudo labels effectively increase the training iterations within one epoch, and longer training times have been proven to be beneficial for dense prediction tasks such as semantic segmentation because it can refine the boundaries [34], [26]. To cancel out the effect of longer training, we design another group of experiments by duplicating the real labels. As may be seen in Table 1(b), increasing the number of iterations by duplication is also helpful, but not as good as using self-training (79.3% vs. 80.2%). In addition, the performance from duplication saturates quickly, but self-training continues to improve with more pseudo labels, which shows its potential to scale.

TABLE 2
Comparison to self-training alternatives.

Method	Network	Label	mIoU
Consistency regularization	single model	hard	78.9
Teacher finetuning	single model	hard	79.2
Soft pseudo labels	teacher + student	soft	79.6
Model regularizer	teacher + student	hard	79.1
Ours	teacher + student	hard	**80.2**

At this point, we have demonstrated the effectiveness of our proposed self-training framework and centroid sampling strategy. Both methods contribute to the improved performance on semantic segmentation.

4.3.3 Comparison to self-training alternatives

We perform ablation studies of the self-training alternatives mentioned in Sec. 3.1.1. We see in Table 2 that our three-stage two-network pipeline achieves the best performance.

Consistency regularization performs the worst possibly because the model is learning from inaccurate predictions in the early training phase. On the contrary, we first train a teacher model with decent accuracy and use it to generate pseudo labels.

In terms of teacher finetuning, after a close inspection on the training accuracy curve, we observe that teacher finetuning could sometimes be stuck at local minima, i.e., the accuracy does not improve anymore at the very beginning of the finetuning process (5th epoch). On the contrary, our two-network design is able to benefit from all the data by training the student model from scratch. Its performance continues to improve throughout the training. In addition, we would like to point out that with our approach, the student model may have a different network architecture which does not have to be the same as that of the teacher's. In the next section, we use a single teacher but train several student models with various backbones and network architectures. Our self-training framework consistently improves all of them, which demonstrate its generalizability.

Compared to soft formulations (i.e., soft pseudo labels and model regularizer), using hard pseudo labels seems a better choice for semantic segmentation task. A potential explanation is that soft labels may cause ambiguity both inside an object and around object boundaries, which is harmful for structured prediction tasks like semantic segmentation. This observation also agrees with [63] that dense per-pixel prediction problems favor hard labels.

4.3.4 Generalizing to other students

One advantage of our self-training framework is model-agnostic, i.e., the student model does not have to be the same as the teacher. It is just a way to increase the number of training samples by utilizing unlabeled data, and improve the accuracy and robustness of model itself. To be specific, we use the pseudo labels generated by our teacher (DeepLabV3+ with ResNeXt50 backbone), and train on various students (1) PSPNet with ResNet101 backbone [22]); (2) DeepLabV3+ with WideResNet38 backbone [65]; (3) OCR with HRNet-W48 backbone [28]; (4) ICNet with ResNet50 backbone [64]; (5) BiSeNet with ResNet18 backbone [21]; and (6) FastSCNN [34]. The first three are widely adopted as the best-performing segmentation models, and the last three are popular real-time fast segmentation models.

TABLE 3
Our self-training method can improve the student models irrespective of backbones and network architectures without using additional labeled samples. # Real: the number of real labels used in training

Method	Backbone	# Real	Val mIoU	Test mIoU
PSPNet [22]	ResNet101	3K	77.9	80.2
PSPNet w/ CSST	ResNet101	3K	**79.9**	**81.9**
DeepLabV3+ [11]	WideResNet38	3K	80.5	80.8
DeepLabV3+ w/ CSST	WideResNet38	3K	**82.2**	**83.3**
OCR [28]	HRNet-W48	3K	82.7	81.8
OCR w/ CSST	HRNet-W48	3K	**84.1**	**83.8**
ICNet [64]	ResNet50	3K	70.5	69.5
ICNet w/ CSST	ResNet50	3K	**74.9**	**74.5**
BiSeNet [21]	ResNet18	3K	74.8	74.7
BiSeNet w/ CSST	ResNet18	3K	**78.3**	**77.1**
FastSCNN [34]	-	3K	68.6	68.0
FastSCNN w/ CSST	-	3K	**72.5**	**72.3**

As shown in Table. 3, our self-training method can improve the student model irrespective of the backbones and network architectures on both validation and held-out test set. Our enhanced BiSeNet model achieves 78.3% mIoU on the Cityscapes validation set with an fps of 35.4. It outperforms the original PSPNet (77.9%) as well as being 15 times faster. In addition, our trained FastSCNN model achieves an mIoU score of 72.5% on the Cityscapes validation set, with only 1.1M parameters, and inference can be run at a speed of 81.5 fps. We believe this is a strong result for real-time semantic segmentation as compared to the recent literature [39].

4.3.5 Effectiveness of fast training

As mentioned previously, the surge in the amount of training samples leads to longer training times which requires a faster training schedule. We introduce four of them in Sec. 3.3 and report their performance in Table 4. Our baseline uses a crop size of 800 throughout the training, and achieves 80.2% mIoU (on 3K real labels and 40K pseudo labels). This experiment takes about 9 days to complete. Next, we simply switch to our proposed fast learning schedules. We see that by using coarse2fine+, we are able to match the baseline's performance with 1.7x speed up. This means the training time is effectively shortened to 5 days. In addition, our technique is orthogonal to mixed precision training [82]. By using fp16, we can further reduce the training time by 2x to 2.5 days. We believe the speed up will make research on semi-supervised semantic segmentation more accessible and scalable.

Lastly, we would like to point out that, the speed up is model dependent. Heavier models tend to consume more compute time given large resolution input. As can be seen in Table 5, we can enjoy 2x speed up when training with heavier models, e.g., using WideResNet38 or HRNet as the network backbone. Detailed training settings for all models in Table 5 can be found in Table 6. Other hyperparameters are the same for all models, e.g, SGD optimizer with a

TABLE 4
Fast training schedules.

Baseline	80.2
coarse2fine	79.6
fine2coarse	79.9
coarse2fine+	**80.2**
fine2coarse+	80.0
speed up	1.7x

TABLE 5
Model-dependent speed up.

Method	Backbone	Speed up
FastSCNN	-	1.5x
ICNet	ResNet50	1.6x
BiSeNet	ResNet18	1.7x
DeepLabV3+	ResNeXt50	1.7x
PSPNet	ResNet101	1.9x
DeepLabV3+	WideResNet38	2x
OCR	HRNet-W48	2x

TABLE 6
Training details of various models in Table 5

	FastSCNN	ICNet	BiSeNet	PSPNet	DeepLabV3+	OCR
Backbone	-	ResNet50	ResNet18	ResNet101	WideResNet38	HRNet-W48
Learning rate	0.045	0.01	0.025	0.01	0.02	0.01
Weight decay	$4e^{-5}$	$1e^{-4}$	$1e^{-4}$	$1e^{-4}$	$1e^{-4}$	$5e^{-4}$
Crop size	1024	769	1024	769	800	769
Epochs	1000	120	120	180	180	180
Batch size	16	16	16	16	8	8

TABLE 7
Comparison to recent semi-supervised methods.

Method	Venue	Cityscapes	CamVid
Baseline [66]	PAMI16	71.0	63.4
CRST [19]	ICCV19	72.7	67.2
HLCon [16]	ICCV19	73.2	68.1
CCT [67]	CVPR20	71.9	66.5
DST-CBC [44]	Arxiv20	72.8	67.9
PseudoSeg [68]	ICLR21	72.9	68.9
CSST (ours)	-	**74.1**	**69.6**

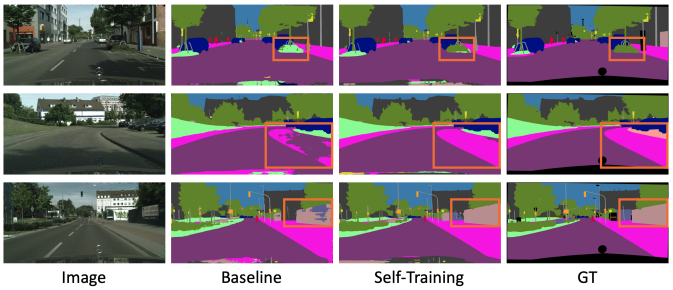


Fig. 6. Visual comparisons on Cityscapes images. We demonstrate that self-training can effectively handle class confusion, such as tree and vegetation (row 1), road and sidewalk (row 2), wall and fence (row 3). Black regions in ground truth represent background class.

momentum of 0.9, Poly learning rate policy with a power of 0.9.

4.3.6 Comparison to semi-supervised methods

Despite the sizable work on semi-supervised semantic segmentation, most of them work with different network architectures (e.g., DRN [83], DeepLabv2 [66], etc.) on different dataset settings (e.g., synthetic-to-real [18], intra-dataset split [44], etc.). We choose five recent approaches for comparison, CRST [19], HLCon [16], CCT [67], DST-CBC [44] and PseudoSeg [68]. In particular, CRST, HLCon, DST-CBC and PseudoSeg are self-training based, which are closely related to our work.

In order to make fair comparisons, we use their official released code with minimal changes and report the performance in Table 7. All methods in the table use DeepLabv2 network architecture with ResNet101 backbone. They are trained on 3K Cityscapes real labels and 40K pseudo labels. As can be seen in Table 7, our proposed method outperforms recent semi-supervised approaches on the Cityscapes validation set under the same setting. This again demonstrates the effectiveness of our self-training framework, in particular the centroid sampling strategy because all the methods are trained on the same data.

4.3.7 Comparison to state-of-the-art

In the end, we would like to compare to recent state-of-the-art approaches on the test set of Cityscapes. For the test submission, we train our model using the best recipe suggested above, with several modifications. We use OCR model with a HRNet-W48 [74] backbone for both teacher and student, and adopt a standard multi-scale strategy following [22], [25] to perform inference on multi-scaled (0.5, 1.0 and 2.0), left-right flipped and overlapping-tiled images. As we can see in Table 8, our self-training method (CSST) achieves an mIoU of 83.8% using only the Cityscapes 3K real labels, outperforming all prior methods in the top of Table 8 that use the same training data. We are also better or competitive to recent approaches in the bottom of Table 8 that pretrained on large-scale labeled datasets (i.e., Mapillary). In particular for an apple-to-apple comparison, we even outperform [28] that use the same network but pretrained on 40K extra labeled training samples. This result is inspiring because it indicates that we may not need a ultra large-scale labeled dataset to achieve good performance. Especially for autonomous driving, we have unlimited videos recorded during driving but not the resources to label them. With this self-training technique, we may generalize the model to various cities or situations with images alone.

In addition, we want to point out that previous approaches adopt other effective techniques for semantic segmentation, such as attention mechanism [29], [30], [32], [28], improved boundary handling [73], [11], [84], better network architecture [85], [37], [38] or multi-tasking framework [69], [26], [80], [86]. Our self-training framework is orthogonal to all these techniques, and can incorporate them to further improve the performance.

Finally, we show several visual examples in Fig. 6, comparing predictions from the baseline model and our CSST enhanced model. We can see that using unlabeled data is useful because it can help to alleviate class confusion, such as between tree and vegetation. In terms of failure cases, we show four challenging scenarios. In Fig. 7 (a), the bicycle is overlapping with a car. It is hard to correctly tell them apart from such a long distance. In Fig. 7 (b), our model predicts a reflection in the mirror as a person. This is an interesting result because our prediction should be considered correct in terms of appearance without reasoning about context. In Fig. 7 (c), it is very hard to tell whether it is a car or bus when the object is far away, especially when there is strong illumination. In Fig. 7 (d), this is an ambiguous situation because the handbag is neither a person or car. It should be a background class.

TABLE 8
Per-class comparison with top-performing methods on the test set of Cityscapes. Top: methods trained using only 3K Cityscapes training set. Bottom: methods trained with extra labeled data such as Mapillary and Cityscapes coarse set. Our CSST only uses 3K labeled data from Cityscapes training set and unlabeled images from Mapillary and Cityscapes coarse set, thus the labeling effort is greatly reduced.

Method	road	swalk	build.	wall	fence	pole	tlight	tsign	veg.	terrain	sky	person	rider	car	truck	bus	train	mcycle	bicycle	mIoU
DepthSeg [69]	98.5	85.4	92.5	54.4	60.9	60.2	72.3	76.8	93.1	71.6	94.8	85.2	68.9	95.7	70.1	86.5	75.5	68.3	75.5	78.2
PSPNet [22]	98.6	86.2	92.9	50.8	58.8	64.0	75.6	79.0	93.4	72.3	95.4	86.5	71.3	95.9	68.2	79.5	73.8	69.5	77.2	78.4
AAF [70]	98.5	85.6	93.0	53.8	58.9	65.9	75.0	78.4	93.7	72.4	95.6	86.4	70.5	95.9	73.9	82.7	76.9	68.7	76.4	79.1
PanoDeepLab [26]	98.7	87.2	93.6	57.7	60.8	70.8	78.0	81.2	93.8	**74.1**	95.7	88.2	76.4	96.0	55.3	75.1	79.6	72.1	74.0	79.4
DenseASPP [71]	98.7	87.1	93.4	60.7	62.7	65.6	74.6	78.5	93.6	72.5	95.4	86.2	71.9	96.0	78.0	90.3	80.7	69.7	76.8	80.6
SPG [72]	98.8	87.6	93.8	56.5	61.9	71.9	80.0	82.1	94.1	73.5	**96.1**	88.7	74.9	96.5	67.3	84.8	81.8	71.1	79.4	81.1
BFP [73]	98.7	87.0	93.5	59.8	63.4	68.9	76.8	80.9	93.7	72.8	95.5	87.0	72.1	96.0	77.6	89.0	86.9	69.2	77.6	81.4
DANet [29]	98.6	86.1	93.5	56.1	63.3	69.7	77.3	81.3	93.9	72.9	95.7	87.3	72.9	96.2	76.8	89.4	86.5	72.2	78.2	81.5
HRNetv2 [74]	98.8	87.9	93.9	61.3	63.1	72.1	79.3	82.4	94.0	73.4	96.0	88.5	75.1	96.5	72.5	88.1	79.9	73.1	79.2	81.8
ACFNet [32]	98.7	87.1	93.9	60.2	63.9	71.1	78.6	81.5	94.0	72.9	95.9	88.1	74.1	96.5	76.6	89.3	81.5	72.1	79.2	81.8
EMANet [75]	98.7	87.3	93.8	63.4	62.3	70.0	77.9	80.7	93.9	73.6	95.7	87.8	74.5	96.2	75.5	90.2	84.5	71.5	78.7	81.9
ACNet [30]	98.7	87.1	93.9	61.6	61.8	71.4	78.7	81.7	94.0	73.3	96.0	88.5	74.9	96.5	77.1	89.0	89.2	71.4	79.0	82.3
Auto-DeepLab-L [37]	98.8	87.6	93.8	61.4	64.4	71.2	77.6	80.9	94.1	72.7	96.0	87.8	72.8	96.5	78.2	90.9	88.4	69.0	77.6	82.1
Gated-SCNN [76]	98.7	87.4	94.2	61.9	64.6	72.9	79.6	82.5	94.3	74.3	96.2	88.3	74.2	96.6	77.2	90.1	87.7	72.6	79.4	82.8
GALD-Net [77]	98.8	87.6	94.2	64.6	66.5	72.8	79.0	82.2	94.2	73.1	96.0	88.3	75.2	96.5	79.3	89.7	87.5	73.8	80.0	83.1
VPLR [11]	98.8	87.8	94.2	64.1	65.0	72.4	79.0	82.8	94.2	74.0	96.1	88.2	75.4	96.5	78.8	94.0	91.6	73.7	79.0	83.5
DecoupleSegNet [78]	98.8	87.8	94.4	66.1	64.8	72.3	78.8	82.6	94.2	74.0	96.1	88.7	75.9	96.6	80.2	93.8	91.6	74.3	79.5	83.7
OCR [28]	98.8	88.3	94.3	66.9	66.7	73.3	80.2	83.0	94.2	74.1	96.0	88.5	75.8	96.5	78.5	91.8	90.1	73.4	79.3	83.7
Axial-DeepLab-XL [79]	98.9	88.3	94.5	69.1	67.8	74.5	80.3	83.9	94.1	72.8	96.1	89.3	78.2	96.4	76.4	93.0	91.3	75.7	76.9	84.1
EfficientPS [80]	98.8	88.2	94.3	67.6	67.7	73.4	80.2	83.3	94.3	74.4	96.0	88.7	75.3	96.6	83.5	94.0	91.6	73.5	79.7	84.2
DCNAS [81]	98.8	87.8	94.2	64.1	65.0	72.4	79.0	82.8	94.2	74.0	96.1	88.2	75.4	96.5	78.8	94.0	91.6	73.7	79.0	84.3
PanoDeepLab [26]	98.8	88.4	94.4	64.3	68.3	75.3	81.0	84.2	94.2	73.7	96.1	89.7	78.6	96.7	82.2	93.7	90.2	76.4	79.8	84.5
CSST (ours)	**98.8**	**87.9**	**94.2**	**65.0**	**66.5**	**73.4**	**80.2**	**82.9**	**94.1**	73.4	95.6	**89.0**	**75.5**	**96.8**	**81.1**	**93.5**	**90.5**	**74.1**	**80.1**	**83.8**

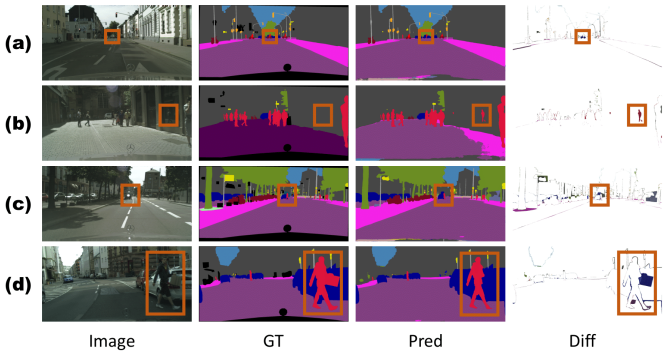


Fig. 7. Visualizations of challenging failure cases: (a) objects overlapping, (b) reflection in the mirror, (c) objects far away with strong illumination, and (d) annotation ambiguity. Diff indicates the difference between our class predictions and the ground truth.

TABLE 9
Comparison to top-performing methods on PASCAL VOC 2012 test set. VOC_TRAIN is the original train set with 1.5K samples. VOC_AUG is the augmented set with 9K samples, we do not use its labels for CSST.

Method	VOC_TRAIN	VOC_AUG	Test mIoU
PSPNet [22]	✓	✓	82.6
PSPNet + CSST	✓		**83.1**
DeepLabv3 [88]	✓	✓	82.6
DeepLabv3 + CSST	✓		**82.9**
EncNet [89]	✓	✓	82.9
EncNet + CSST	✓		**83.3**

4.4 CamVid

A main goal of semi-supervised learning is to achieve good accuracy with less supervision. In this section, we would like to study the performance of our self-training framework when there are fewer real labels.

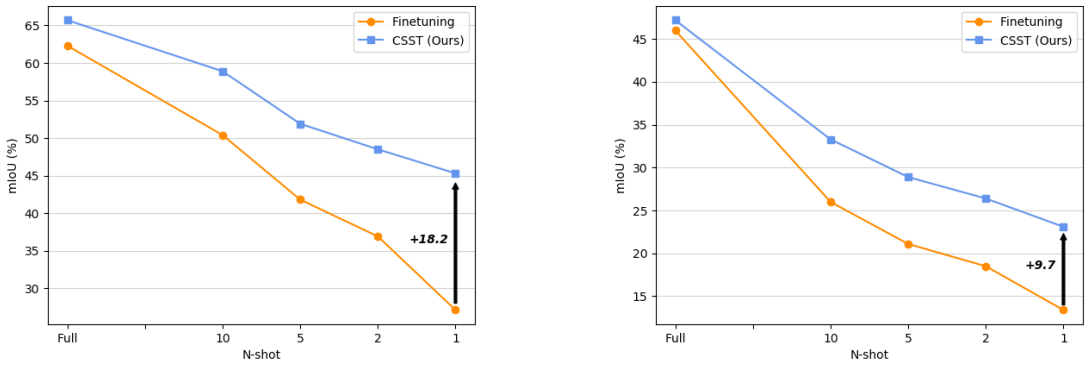
We use the CamVid dataset due to its relatively small scale (i.e., 367 training images). For fair comparison to recent semi-supervised methods, similar to our experiments on Cityscapes, we use DeepLabv2 network with ResNet101 backbone as the model. In terms of pseudo labels, we use the 367 training images to train a teacher network and then generate 40K pseudo labels using this teacher model.

As we can see in Table 7, our self-training framework works the best compared to recent approaches on CamVid. We improve 6.2% upon the baseline by using unlabeled data. We also find that our method has a bigger performance gain on the fewer-label scenario (CamVid) than the more-label scenario (Cityscapes), which suggests the practicality of our method when the target application has few annotations.

Furthermore, in order to compare to state-of-the-art performance, we switch to a heavier model using OCR with HRNet-W48. We are able to achieve 80.6% mIoU without pretraining on Cityscapes or Mapillary, which is comparable to previous methods [87], [11], [78] that are pretrained on extra labeled data.

4.5 PASCAL VOC 2012

We mainly focus on street scene segmentation datasets in this paper. Here, we also evaluate the performance of our proposed method on PASCAL VOC 2012 dataset [59], which is one of the standard benchmarks for generic semantic segmentation. Recall in Sec. 4.1, the original PASCAL VOC 2012 dataset contains 1.5K (train), 1.5K (val), and 1.5K (test) images. Due to the small size of the train set, most methods pretrain the model on an augmented set with 9K training samples from [60] and then finetune it on the train+val set. Instead of pretraining on the augmented set, we only use the additional 9K images as unlabeled data to generate pseudo labels. We compare our method to recent approaches in Table 9 on the test set of PASCAL VOC 2012. We can see that, our CSST trained with 1.5K samples even outperforms the models that are trained with 10.5K real labels. The same observation holds for PSPNet, Deeplabv3 and EncNet. This indicates that CSST is robust with less supervision and may generalize better when there is domain shift.



(a) Cityscapes to BDD100K (b) Cityscapes to Mapillary

Fig. 8. Cross-domain generalization experiments. Full indicates using the complete original training set. Our method outperforms conventional finetuning approach by a large margin, particularly in few-shot scenarios.

4.6 Cross-Domain Generalization

Using the self-training framework, we already showed that we can improve the accuracy of semantic segmentation on the current dataset by leveraging a large set of external unlabeled data. However, a segmentation model trained on one domain may not generalize to another. For instance, two driving datasets collected in different locations may be significantly different in terms of traffic, lighting and viewpoint. A conventional approach would be to train a model on the source domain, and finetune it on the new domain. However, this approach requires a large amount of annotations from the target domain to achieve good performance. Here, we would like to show that our self-training method can effectively mitigate the domain gap by learning the prior knowledge of data distribution from pseudo labels generated on the target domain.

4.6.1 Cross-domain generalization with same categories

We first describe the task of cross-domain generalization. We set the Cityscapes dataset as the source domain, and BDD100K dataset as the target domain. BDD100K has the same 19 categories as Cityscapes, but it is still a challenging situation because the data of BDD100K is collected in United States. On the other hand, the data in Cityscapes is collected from German cities. In terms of few-shot learning, we evaluate on 1-shot, 2-shot, 5-shot and 10-shot scenarios. We compare two settings, (1) a conventional finetuning approach: a model trained on Cityscapes, and then finetuned on the given BDD100K labeled samples; (2) our self-training approach: a model trained on both Cityscapes and pseudo labels from BDD100K, and then finetuned on the given BDD100K labeled samples. For simplicity, we do not use OHEM loss and other non-default training options, to better emphasize the contributions from model initialization. Our goal is to see whether self-training can help improve a model's generalization capability across domains.

The results can be seen in Fig. 8 (a), and we have several observations. First, the model trained using our self-training framework achieves the best performance across all scenarios compared to the finetuning approach. This indicates that the self-training method can largely reduce the human labeling effort when generalizing to new domains. Second, in terms of using the full training set, our approach also performs better. This maybe due to the fact that self-training serves as effective data augmentation and provides a better model initialization. Third, given fewer and fewer training samples, the gap between the finetuning model and our self-training model becomes larger. Even in the case of 1-shot learning, our method is able to achieve decent performance of 45.3% mIoU. We significantly outperform the finetuning approach, an improvement of 18.2% ($27.1\% \rightarrow 45.3\%$).

4.6.2 Cross-domain generalization with new categories

Note that the target domain could have new categories which makes the task even more difficult, as the model needs to learn knowledge from new scenarios with just a few labels.

Here, we set Cityscapes dataset as the source domain, and Mapillary dataset as the target domain. We choose Mapillary due to its large size and variability. In addition, Mapillary has many more semantic categories than Cityscapes, 66 compared to 19. Hence, most classes are considered new, never seen by the model trained on the source domain, which makes the problem very challenging.

To be specific, we conduct four steps. We first train a teacher model on Cityscapes with 19 classes. We then use the teacher model to generate pseudo labels on Mapillary. Note that, pseudo labels will only have 19 categories regardless of where the images are from, because they are generated by this teacher model. Then we combine the real and pseudo labels to train a student model. The classifier of the student model is still a 19-way classifier. Finally, we finetune the student model on Mapillary. Only in the last step, the classifier is 66-way because we are using labeled data from Mapillary. Hence, we are able to evaluate on the full 66 classes of Mapillary.

The results are shown in Fig. 8 (b). In the case of 1-shot learning, our method is able to outperform the finetuning approach with an improvement of 9.7% ($13.4\% \rightarrow 23.1\%$). Hence, the results suggest that our self-training technique can generalize well to various locations with just images and a few annotations from the new scene.

5 Conclusion

In this work, we introduce a self-training framework for semi-supervised semantic segmentation. Together with the proposed centroid sampling strategy, we properly handle the data imbalance problem and achieve state-of-the-art performance on Cityscapes and CamVid datasets. In particular, our method even outperforms models pre-trained on large amounts of external labeled data. We also have the same observation on PASCAL VOC 2012 dataset. Furthermore, our self-training framework can generalize to various students and significantly boost their performance without using additional labeled data or inference time. It is therefore more flexible in real-world applications. We also demonstrate its effectiveness on two challenging cross-domain generalization tasks. Our method significantly outperform conventional finetuning in the few-shot scenarios.

We want to emphasize that our proposed techniques are not specifically designed for semi-supervised semantic segmentation. For example, the fast training schedule can be used in supervised learning when there is a huge dataset. Centroid sampling can be used for real labels or even in other tasks beyond segmentation. We hope our framework can facilitate research in the direction of using large-scale unlabeled data for computer vision.

References

[1] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The Cityscapes Dataset for Semantic Urban Scene Understanding," in *CVPR*, 2016.

[2] G. Neuhold, T. Ollmann, S. R. Bulò, and P. Kontschieder, "The Mapillary Vistas Dataset for Semantic Understanding of Street Scenes," in *ICCV*, 2017.

[3] F. Yu, W. Xian, Y. Chen, F. Liu, M. Liao, V. Madhavan, and T. Darrell, "BDD100K: A Diverse Driving Video Database with Scalable Annotation Tooling," *arXiv preprint arXiv:1805.04687*, 2018.

[4] C. Sun, A. Shrivastava, S. Singh, and A. Gupta, "Revisiting Unreasonable Effectiveness of Data in Deep Learning Era," in *ICCV*, 2017.

[5] D. Mahajan, R. Girshick, V. Ramanathan, K. He, M. Paluri, Y. Li, A. Bharambe, and L. van der Maaten, "Exploring the Limits of Weakly Supervised Pretraining," in *ECCV*, 2018.

[6] A. Kuznetsova, H. Rom, N. Alldrin, J. Uijlings, I. Krasin, J. Pont-Tuset, S. Kamali, S. Popov, M. Malloci, T. D. Alexander Kolesnikov, and V. Ferrari, "The Open Images Dataset V4: Unified Image Classification, Object Detection, and Visual Relationship Detection at Scale," *IJCV*, 2020.

[7] S. Abu-El-Haija, N. Kothari, J. Lee, P. Natsev, G. Toderici, B. Varadarajan, and S. Vijayanarasimhan, "YouTube-8M: A Large-Scale Video Classification Benchmark," *arXiv preprint arXiv:1609.08675*, 2016.

[8] S. K. Mustikovela, M. Y. Yang, and C. Rother, "Can Ground Truth Label Propagation from Video help Semantic Segmentation?" in *ECCV Workshop*, 2016.

[9] P. Luc, N. Neverova, C. Couprie, J. Verbeek, and Y. LeCun, "Predicting Deeper into the Future of Semantic Segmentation," in *ICCV*, 2017.

[10] I. Budvytis, P. Sauer, T. Roddick, K. Breen, and R. Cipolla, "Large Scale Labelled Video Data Augmentation for Semantic Segmentation in Driving Scenarios," in *ICCV Workshop*, 2017.

[11] Y. Zhu, K. Sapra, F. A. Reda, K. J. Shih, S. Newsam, A. Tao, and B. Catanzaro, "Improving Semantic Segmentation via Video Propagation and Label Relaxation," in *CVPR*, 2019.

[12] J. Xie, B. Shuai, J.-F. Hu, J. Lin, and W.-S. Zheng, "Improving Fast Segmentation With Teacher-student Learning," in *BMVC*, 2018.

[13] Y. Liu, C. Shun, J. Wang, and C. Shen, "Structured Knowledge Distillation for Dense Prediction," in *CVPR*, 2019.

[14] N. Souly, C. Spampinato, and M. Shah, "Semi and Weakly Supervised Semantic Segmentation Using Generative Adversarial Network," in *ICCV*, 2017.

[15] W.-C. Hung, Y.-H. Tsai, Y.-T. Liou, Y.-Y. Lin, and M.-H. Yang, "Adversarial Learning for Semi-Supervised Semantic Segmentation," in *BMVC*, 2018.

[16] S. Mittal, M. Tatarchenko, and T. Brox, "Semi-Supervised Semantic Segmentation with High- and Low-level Consistency," *arXiv preprint arXiv:1908.05724*, 2019.

[17] G. French, T. Aila, S. Laine, M. Mackiewicz, and G. Finlayson, "Semi-Supervised Semantic Segmentation Needs Strong, High-Dimensional Perturbations," *arXiv preprint arXiv:1906.01916*, 2019.

[18] Y. Zou, Z. Yu, B. V. K. V. Kumar, and J. Wang, "Domain Adaptation for Semantic Segmentation via Class-Balanced Self-Training," in *ECCV*, 2018.

[19] Y. Zou, Z. Yu, X. Liu, B. V. Kumar, and J. Wang, "Confidence Regularized Self-Training," in *ICCV*, 2019.

[20] Q. Xie, M.-T. Luong, E. Hovy, and Q. V. Le, "Self-Training with Noisy Student Improves ImageNet Classification," *arXiv preprint arXiv:1911.04252*, 2019.

[21] C. Yu, J. Wang, C. Peng, C. Gao, G. Yu, and N. Sang, "BiSeNet: Bilateral Segmentation Network for Real-time Semantic Segmentation," in *ECCV*, 2018.

[22] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid Scene Parsing Network," in *CVPR*, 2017.

[23] J. Long, E. Shelhamer, and T. Darrell, "Fully Convolutional Networks for Semantic Segmentation," in *CVPR*, 2015.

[24] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation," in *ECCV*, 2018.

[25] S. R. Bulò, L. Porzi, and P. Kontschieder, "In-Place Activated BatchNorm for Memory-Optimized Training of DNNs," in *CVPR*, 2018.

[26] B. Cheng, M. D. Collins, Y. Zhu, T. Liu, T. S. Huang, H. Adam, and L.-C. Chen, "Panoptic-DeepLab," *arXiv preprint arXiv:1910.04751*, 2019.

[27] Y. Yuan, L. Huang, J. Guo, C. Zhang, X. Chen, and J. Wang, "OCNet: Object Context Network for Scene Parsing," *arXiv preprint arXiv:1809.00916*, 2018.

[28] Y. Yuan, X. Chen, and J. Wang, "Object-Contextual Representations for Semantic Segmentation," *arXiv preprint arXiv:1909.11065*, 2019.

[29] J. Fu, J. Liu, H. Tian, Y. Li, Y. Bao, Z. Fang, and H. Lu, "Dual Attention Network for Scene Segmentation," in *CVPR*, 2019.

[30] J. Fu, J. Liu, Y. Wang, Y. Li, Y. Bao, J. Tang, and H. Lu, "Adaptive Context Network for Scene Parsing," in *ICCV*, 2019.

[31] H. Zhao, Y. Zhang, S. Liu, J. Shi, C. C. Loy, D. Lin, and J. Jia, "PSANet: Point-wise Spatial Attention Network for Scene Parsing," in *ECCV*, 2018.

[32] F. Zhang, Y. Chen, Z. Li, Z. Hong, J. Liu, F. Ma, J. Han, and E. Ding, "ACFNet: Attentional Class Feature Network for Semantic Segmentation," in *ICCV*, 2019.

[33] Z. Huang, X. Wang, L. Huang, C. Huang, Y. Wei, and W. Liu, "CCNet: Criss-Cross Attention for Semantic Segmentation," in *ICCV*, 2019.

[34] R. P. K. Poudel, S. Liwicki, and R. Cipolla, "Fast-SCNN: Fast Semantic Segmentation Network," *arXiv preprint arXiv:1902.04502*, 2019.

[35] H. Li, P. Xiong, H. Fan, and J. Sun, "DFANet: Deep Feature Aggregation for Real-Time Semantic Segmentation," in *CVPR*, 2019.

[36] H. Wu, J. Zhang, K. Huang, K. Liang, and Y. Yizhou, "FastFCN: Rethinking Dilated Convolution in the Backbone for Semantic Segmentation," in *arXiv preprint arXiv:1903.11816*, 2019.

[37] C. Liu, L.-C. Chen, F. Schroff, H. Adam, W. Hua, A. Yuille, and L. Fei-Fei, "Auto-DeepLab: Hierarchical Neural Architecture Search for Semantic Image Segmentation," in *CVPR*, 2019.

[38] L.-C. Chen, M. D. Collins, Y. Zhu, G. Papandreou, B. Zoph, F. Schroff, H. Adam, and J. Shlens, "Searching for Efficient Multi-Scale Architectures for Dense Image Prediction," in *NeurIPS*, 2018.

[39] V. Nekrasov, H. Chen, C. Shen, and I. Reid, "Fast Neural Architecture Search of Compact Semantic Segmentation Models via Auxiliary Cells," in *CVPR*, 2019.

[40] Y. Zhang, Z. Qiu, J. Liu, T. Yao, D. Liu, and T. Mei, "Customizable Architecture Search for Semantic Segmentation," in *CVPR*, 2019.

[41] I. Z. Yalniz, H. Jégou, K. Chen, M. Paluri, and D. Mahajan, "Billion-Scale Semi-Supervised Learning for Image Classification," *arXiv preprint arXiv:1905.00546*, 2019.

[42] A. Tao, K. Sapra, and B. Catanzaro, "Hierarchical Multi-Scale Attention for Semantic Segmentation," *arXiv preprint arXiv:2005.10821*, 2020.

[43] L.-C. Chen, R. G. Lopes, B. Cheng, M. D. Collins, E. D. Cubuk, B. Zoph, H. Adam, and J. Shlens, "Naive-Student: Leveraging Semi-Supervised Learning in Video Sequences for Urban Scene Segmentation," in *ECCV*, 2020.

[44] Z. Feng, Q. Zhou, G. Cheng, X. Tan, J. Shi, and L. Ma, "Semi-Supervised Semantic Segmentation via Dynamic Self-Training and Class-Balanced Curriculum," *arXiv preprint arXiv:2004.08514*, 2020.

[45] B. Zoph, G. Ghiasi, T.-Y. Lin, Y. Cui, H. Liu, E. D. Cubuk, and Q. V. Le, "Rethinking Pre-training and Self-training," in *NeurIPS*, 2020.

[46] J. Hoffman, E. Tzeng, T. Park, J.-Y. Zhu, P. Isola, K. Saenko, A. A. Efros, and T. Darrell, "CyCADA: Cycle-Consistent Adversarial Domain Adaptation," in *ICML*, 2018.

[47] Y.-H. Tsai, W.-C. Hung, S. Schulter, K. Sohn, M.-H. Yang, and M. Chandraker, "Learning to Adapt Structured Output Space for Semantic Segmentation," in *CVPR*, 2018.

[48] T.-H. Vu, H. Jain, M. Bucher, M. Cord, and P. Perez, "ADVENT: Adversarial Entropy Minimization for Domain Adaptation in Semantic Segmentation," in *CVPR*, 2019.

[49] Q. Zhang, J. Zhang, W. Liu, and D. Tao, "Category Anchor-Guided Unsupervised Domain Adaptation for Semantic Segmentation," in *NeurIPS*, 2019.

[50] Q. Lian, F. Lv, L. Duan, and B. Gong, "Constructing Self-Motivated Pyramid Curriculums for Cross-Domain Semantic Segmentation: A Non-Adversarial Approach," in *ICCV*, 2019.

[51] Y.-C. Chen, Y.-Y. Lin, M.-H. Yang, and J.-B. Huang, "CrDoCo: Pixel-Level Domain Transfer With Cross-Domain Consistency," in *CVPR*, 2019.

[52] C.-Y. Lee, T. Batra, M. H. Baig, and D. Ulbricht, "Sliced Wasserstein Discrepancy for Unsupervised Domain Adaptation," in *CVPR*, 2019.

[53] X. Yue, Y. Zhang, S. Zhao, A. Sangiovanni-Vincentelli, K. Keutzer, and B. Gong, "Domain Randomization and Pyramid Consistency: Simulation-to-Real Generalization Without Accessing Target Domain Data," in *ICCV*, 2019.

[54] K. Sohn, D. Berthelot, C.-L. Li, Z. Zhang, N. Carlini, E. D. Cubuk, A. Kurakin, H. Zhang, and C. Raffel, "FixMatch: Simplifying Semi-Supervised Learning with Consistency and Confidence," in *NeurIPS*, 2019.

[55] D.-H. Lee, "Pseudo-Label:The Simple and Efficient Semi-Supervised Learning Method for Deep Neural Networks," in *ICML Workshops*, 2013.

[56] T. Miyato, S. ichi Maeda andMasanori Koyama, and S. Ishii, "Virtual Adversarial Training: A Regularization Method for Semi-Supervised Learning," *PAMI*, 2018.

[57] C.-Y. Wu, R. Girshick, K. He, C. Feichtenhofer, and P. Krähenbühl, "A Multigrid Method for Efficiently Training Video Models," in *CVPR*, 2020.

[58] G. J. Brostow, J. Shotton, J. Fauqueur, and R. Cipolla, "Segmentation and Recognition Using Structure from Motion Point Clouds," in *ECCV*, 2008.

[59] M. Everingham, S. M. A. Eslami, L. V. Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes Challenge: A Retrospective," *IJCV*, 2014.

[60] B. Hariharan, P. Arbelaez, L. Bourdev, S. Maji, and J. Malik, "Semantic Contours from Inverse Detectors," in *ICCV*, 2011.

[61] W. Liu, A. Rabinovich, and A. C. Berg, "ParseNet: Looking Wider to See Better," in *ICLR*, 2016.

[62] Z. Wu, C. Shen, and A. van den Hengel, "High-performance Semantic Segmentation Using Very Deep Fully Convolutional Networks," *arXiv preprint arXiv:1604.04339*, 2016.

[63] T. He, Z. Zhang, H. Zhang, Z. Zhang, J. Xie, and M. Li, "Bag of Tricks for Image Classification with Convolutional Neural Networks," in *CVPR*, 2019.

[64] H. Zhao, X. Qi, X. Shen, J. Shi, and J. Jia, "ICNet for Real-Time Semantic Segmentation on High-Resolution Images," in *ECCV*, 2018.

[65] Z. Wu, C. Shen, and A. van den Hengel, "Wider or Deeper: Revisiting the ResNet Model for Visual Recognition," *arXiv preprint arXiv:1611.10080*, 2016.

[66] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs," *PAMI*, 2017.

[67] Y. Ouali, C. Hudelot, and M. Tami, "Semi-Supervised Semantic Segmentation With Cross-Consistency Training," in *CVPR*, 2020.

[68] Y. Zou, Z. Zhang, H. Zhang, C.-L. Li, X. Bian, J.-B. Huang, and T. Pfister, "Pseudoseg: Designing pseudo labels for semantic segmentation," in *ICLR*, 2021.

[69] S. Kong and C. Fowlkes, "Recurrent Scene Parsing with Perspective Understanding in the Loop," in *CVPR*, 2018.

[70] T.-W. Ke, J.-J. Hwang, Z. Liu, and S. X. Yu, "Adaptive Affinity Fields for Semantic Segmentation," in *ECCV*, 2018.

[71] M. Yang, K. Yu, C. Zhang, Z. Li, and K. Yang, "DenseASPP for Semantic Segmentation in Street Scenes," in *CVPR*, 2018.

[72] B. Cheng, L.-C. Chen, Y. Wei, Y. Zhu, Z. Huang, J. Xiong, T. S. Huang, W.-M. Hwu, and H. Shi, "SPGNet: Semantic Prediction Guidance for Scene Parsing," in *ICCV*, 2019.

[73] H. Ding, X. Jiang, A. Q. Liu, N. M. Thalmann, and G. Wang, "Boundary-Aware Feature Propagation for Scene Segmentation," in *ICCV*, 2019.

[74] K. Sun, Y. Zhao, B. Jiang, T. Cheng, B. Xiao, D. Liu, Y. Mu, X. Wang, W. Liu, and J. Wang, "High-Resolution Representations for Labeling Pixels and Regions," *arXiv preprint arXiv:1904.04514*, 2019.

[75] X. Li, Z. Zhong, J. Wu, Y. Yang, Z. Lin, and H. Liu, "Expectation-Maximization Attention Networks for Semantic Segmentation," in *ICCV*, 2019.

[76] T. Takikawa, D. Acuna, V. Jampani, and S. Fidler, "Gated-SCNN: Gated Shape CNNs for Semantic Segmentation," in *ICCV*, 2019.

[77] X. Li, L. Zhang, A. You, M. Yang, K. Yang, and Y. Tong, "Global Aggregation then Local Distribution in Fully Convolutional Networks," in *BMVC*, 2019.

[78] X. Li, X. Li, L. Zhang, C. Guangliang, J. Shi, Z. Lin, Y. Tong, and S. Tan, "Improving Semantic Segmentation via Decoupled Body and Edge Supervision," in *ECCV*, 2020.

[79] H. Wang, Y. Zhu, B. Green, H. Adam, A. Yuille, and L.-C. Chen, "Axial-DeepLab: Stand-Alone Axial-Attention for Panoptic Segmentation," in *ECCV*, 2020.

[80] R. Mohan and A. Valada, "EfficientPS: Efficient Panoptic Segmentation," *arXiv preprint arXiv:2004.02307*, 2020.

[81] X. Zhang, H. Xu, H. Mo, J. Tan, C. Yang, L. Wang, and W. Ren, "DCNAS: Densely Connected Neural Architecture Search for Semantic Image Segmentation," in *CVPR*, 2021.

[82] P. Micikevicius, S. Narang, J. Alben, G. Diamos, E. Elsen, D. Garcia, B. Ginsburg, M. Houston, O. Kuchaiev, G. Venkatesh, and H. Wu, "Mixed Precision Training," in *ICLR*, 2018.

[83] F. Yu, V. Koltun, and T. Funkhouser, "Dilated Residual Networks," in *CVPR*, 2017.

[84] Y. Yuan, J. Xie, X. Chen, and J. Wang, "SegFix: Model-Agnostic Boundary Refinement for Segmentation," *arXiv:2007.04269*, 2020.

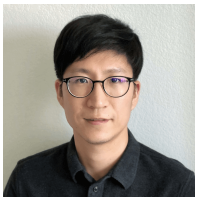
[85] Y. Li, L. Song, Y. Chen, Z. Li, X. Zhang, X. Wang, and J. Sun, "Learning Dynamic Routing for Semantic Segmentation," in *CVPR*, 2020.

[86] A. Valada, R. Mohan, and W. Burgard, "Self-Supervised Model Adaptation for Multimodal Semantic Segmentation," *IJCV*, 2019.

[87] H. Ding, X. Jiang, B. Shuai, A. Q. Liu, and G. Wang, "Semantic Correlation Promoted Shape-Variant Context for Segmentation," in *CVPR*, 2019.

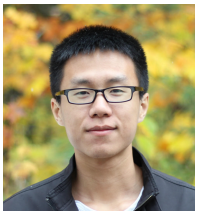
[88] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking Atrous Convolution for Semantic Image Segmentation," *arXiv preprint arXiv 1706.05587*, 2017.

[89] H. Zhang, K. Dana, J. Shi, Z. Zhang, X. Wang, A. Tyagi, and A. Agrawal, "Context Encoding for Semantic Segmentation," in *CVPR*, 2018.



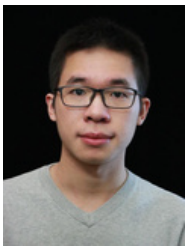
Yi Zhu is an Applied Scientist in Amazon AI. He received his B.E. in Electrical Engineering from Northwestern Polytechnical University (2011), M.S. in Electrical Engineering from University of Kansas (2014), and Ph.D. in Computer Science from University of California, Merced (2019). His research interests include video understanding, semantic segmentation and self-supervised representation learning.

Zhongyue Zhang is an Machine Learning Engineer working on deep learning and computer vision at Snap Inc. Before joining Snap, he worked at Amazon AI and Amazon Lab126. He received his bachelor degree in computer science from University of Washington in 2015.

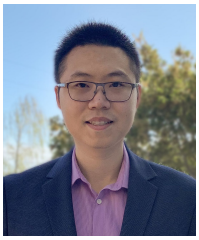


Chongruo Wu is currently a Ph.D. student at the University of California, Davis. He received a master's degree from the University of Michigan, and a bachelor's degree from Zhejiang University. His research interests include object detection and self-supervised learning.

Zhi Zhang is a Senior Applied Scientist in Amazon AI. He received the B.S. degree in electronic and information technology from Beijing Jiaotong University, Beijing, China, in 2012. He received the M.S./Ph.D. degrees in electrical and computer engineering from the University of Missouri, Columbia, MO, USA, in 2014 and 2018, respectively. His research interests include multimjpg edia content retrieval, image and video object detection and classification.

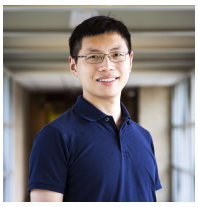


Tong He is an Applied Scientist in AWS Shanghai AI Lab. He received his B.E. in Statistics from Sun Yat-sen University (2013), and M.S. in Computer Science from Simon Fraser University (2016). His research interests include computer vision and graph neural networks.



Hang Zhang is a Research Scientist at Facebook. Before joining Facebook, he has worked at Amazon and NVIDIA. He received his PhD degree at Rutgers University in 2017 and his bachelor degree at Southeast University (Nanjing, China) in 2013.

R.Manmatha is a principal scientist working on computer vision at Amazon. Before joining Amazon he was a Research Associate Professor in the Department of Computer Science at the University of Massachussetts in Amherst. He also co-founded a mobile image search startup SnapTell which was acquired by Amazon. Manmatha's interests are in computer vision, document image analysis and information retrieval and has published in these areas. He is a former associate editor of IEEE PAMI and has served on conference committees in vision, document image analysis and information retrieval.



Mu Li is a Sr. principal scientist for machine learning at AWS. Before joining AWS, he was the CTO of Marianas Labs, an AI start-up. He also served as a principal research architect at the Institute of Deep Learning at Baidu. He obtained his PhD in computer science from Carnegie Mellon University. Mu's research has focused on large-scale machine learning. In particular, he is interested in the co-design of distributed systems and machine learning algorithms. He has been the first-author for computer science conference and journal papers on subjects that span theory (FOCS), machine learning (NeurIPS, ICML), applications (CVPR, KDD) and operating systems (OSDI).

Alex Smola received his Doctoral degree in computer science at the University of Technology Berlin in 1998. After that he worked at the Research School for Information Sciences and Engineering of the Australian National University. From 2004 to 2008, he was Senior Principal Researcher and Program Leader at National Information and Communication Technology Australia (NICTA). From 2008 to 2012 he worked at the Yahoo! Research Lab. He served as Adjunct Professor in the Department of Statistics at the University of Berkeley in 2012 and worked at Google Research thereafter. Moreover, from 2012-2017 he was full tenured professor at the Machine Learning Department of Carnegie Mellon University and cofounder of Marianas Labs. Since 2016 he works at Amazon Web Services, serving as Distinguished Scientist and VP. Alex Smola have published over 200 papers and 5 books, most recently the D2L.ai project which is ongoing.