Urban Scene Semantic Segmentation with Low-Cost Coarse Annotation

Anurag Das[1,2], Yongqin Xian[5], Yang He[6], Zeynep Akata[3,4], Bernt Schiele[1,2]
[1]MPI for Informatics, [2]Saarland Informatics Campus, [3]MPI for Intelligent Systems, [4]University of Tübingen, [5]ETH Zürich, [6]CISPA
{andas,schiele}@mpi-inf.mpg.de, yongqin.xian@vision.ee.ethz.ch, yang.he@cispa.saarland, zeynep.akata@uni-tuebingen.de

Abstract

For best performance, today's semantic segmentation methods use large and carefully labeled datasets, requiring expensive annotation budgets. In this work, we show that coarse annotation is a low-cost but highly effective alternative for training semantic segmentation models. Considering the urban scene segmentation scenario, we leverage cheap coarse annotations for real-world captured data, as well as synthetic data to train our model and show competitive performance compared with finely annotated real-world data. Specifically, we propose a coarse-to-fine self-training framework that generates pseudo labels for unlabeled regions of the coarsely annotated data, using synthetic data to improve predictions around the boundaries between semantic classes, and using cross-domain data augmentation to increase diversity. Our extensive experimental results on Cityscapes and BDD100k datasets demonstrate that our method achieves a significantly better performance vs annotation cost tradeoff, yielding a comparable performance to fully annotated data with only a small fraction of the annotation budget. Also, when used as pretraining, our framework performs better compared to the standard fully supervised setting.

1. Introduction

Deep learning has made substantial progress in the field of semantic segmentation thanks to the availability of large-scale datasets [8, 10, 21]. However, annotating those datasets for semantic segmentation requires carefully labeling every pixel in the images, which is time-consuming and expensive. This has motivated abundant attempts to reduce the annotation efforts by exploring weaker forms of supervision, for example image-level label [23, 14], bounding box [17], scribble [20] and points [1]. Those works mainly focus on the PASCAL VOC dataset [10] with only a few object instances per image, which is a relatively easy scenario. In real-world urban scenes, however, the density of traffic participants is significantly higher [8]. Reducing the annotation burden for such complex urban scenarios remains challenging and underexplored. Cityscapes [8] is one of the most popular datasets for urban scene segmentation, consisting of two kinds of pixel-level annotations (see Fig. 1): (1) Coarse annotation - annotators draw coarse polygons for labelling classes ignoring the finer details around class boundaries. The goal is to annotate as many pixels as possible within 7 min of annotation time per image [8], with the only condition that each polygon must have pixel labels from a single class. (2) Fine annotation - annotators draw fine polygons that align well with the object boundaries, which is much more time-consuming i.e., 90 minutes per image. Coarse annotations tradeoff finer details for lower annotation cost. For best performance, most existing works [22, 5, 45, 41, 44, 38, 46, 39, 13, 32, 43] rely on the full fine annotations, which is significantly expensive to annotate. The coarse data is sometimes used as additional training data to marginally boost the performance (less than 2%) [5, 41, 38]. However, relatively little is known about the potential value of using coarse annotations. In this work,

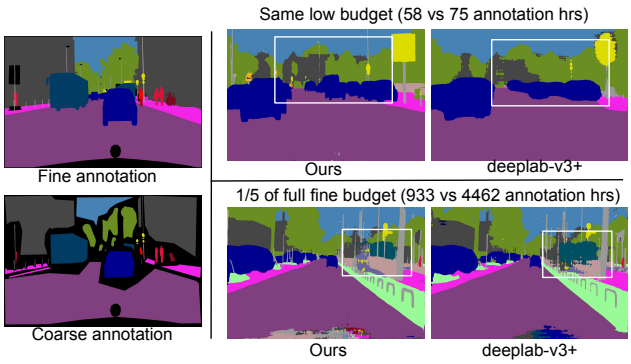

Figure 1: Left: Comparison between coarse and fine annotation. Coarse annotation trades off boundary accuracy for lower annotation cost. Right: Our method using coarse and synthetic data can predict better than DeepLab-v3+ trained on fine data in similar lower budget regime (see car and building in row 1), whereas with one-fifth of full fine budget our method perform comparable for most classes and even better for tail distribution classes (e.g. train in row 2).

we propose a novel hybrid supervision scheme to combine coarse data and synthetic data, aiming to reduce the annotation cost for urban scene segmentation without sacrificing final performance. While coarse data has the advantages of being significantly larger (over $10\times$) than fine data given a fixed annotation budget, it is notoriously difficult to achieve a good performance using coarse data alone, which partially explains the lacking literature in this area. One limitation is that the coarse data has a lot of unlabeled regions which might not contain sufficient supervision signals. We propose a coarse-to-fine self-training framework that generates pseudo labels with consistency constraints for unlabeled regions, gradually converting sparse coarse annotation into dense fine annotation. To circumvent the need of boundary information, we propose to leverage synthetic data which provides precise dense annotation and is free in terms of annotation cost. A boundary loss is applied on the synthetic data to encourage the network to focus on fixing boundary errors. To alleviate the domain gap, we further perform the cross-domain data augmentation that mixes the images from two domains. Finally, we retrain our network with pseudo labelled coarse data and synthetic data to refine the network.

Our work makes the following contributions. First, we emphasize the potential value of coarse annotation, which is significantly cheaper than fine annotation, but has been largely ignored as a primary source for training. Second, we develop a strong baseline for urban scene segmentation that uses hybrid supervision signals from coarse and synthetic data therefore substantially reducing the annotation cost. To the best of our knowledge, we are the first to combine coarse and synthetic data for urban scene segmentation. Finally, we show the trade-off between the annotation budget and performance on the challenging Cityscapes and BDD100k datasets. We empirically demonstrate that our method consistently outperforms its fine data counterpart when using comparable annotation budgets. Notably, we achieve competitive performance with only one-fifth and one-eighth annotation cost compared to using fine annotation on Cityscapes and BDD100k datasets respectively.

2. Related Work

Semantic segmentation with weak annotations. Prior works use image-level annotation [24, 23], point [1], scribble [20] and bounding box [17] annotations. Further, other works incorporate zero-shot learning [2, 36, 19, 9] to transfer knowledge from categories with annotations to the novel classes during test. However, prior works in this direction does not perform complex scene segmentation and only show results in segmenting few objects from images. Different from these works, we focus on challenging urban scene semantic segmentation with coarse annotation.We show a feasible solution to reach competitive results with coarse annotations, which only needs 7 minutes for each 1024×2048 image in average.

Semi-supervised semantic segmentation. studied for example in [15, 31, 6, 18, 4, 12], aims to leverage unlabeled data to improve the segmentation performance or reduce the annotation efforts. Generative adversarial training has been exploited to achieve this goal, either by applying a trained discriminator to provide training signals for unlabeled images [15] or generate labeled pairs from GANs [18]. In addition, other works propagate labels from a learned model to unlabeled images [4, 47, 12]. Even though the above approaches show success in reducing annotation efforts, they still need fine annotations to train a semantic segmentation model for urban scenes, while we only apply coarse annotation to obtain high-quality segmentation.

Self-training and pseudo labels Self-training aims to learn from unlabeled data and generate pseudo labels for supervised learning. Xie *et al*. [37] present Noisy Student Training, in which the student model is added with noise such as dropout, stochastic depth and data augmentation. Ghiasi *et al*. [11] introduce multi-task self-training (MuST) which uses several specialized teacher models trained on labeled data to create a multi-task pseudo labeled dataset. The dataset is used to train a single student model with multi-task learning. Zoph *et al*. [48] reveal that self-training is always helpful when using stronger data augmentation and in the case that pre-training is helpful, self-training improves upon pre-training. More generally, self-training has been exploited to improve the performance of semantic segmentation [4] significantly. Different to this work, we do not touch the fine annotated data to train our model and still obtain competitive performance at much lower annotation costs.

Semantic segmentation using synthetic data There are many successful datasets released for urban scenes [27, 35, 29]. However, most prior works focus on pretraining a model [27, 35], domain adaptation [33, 34, 3] or generalization [7, 42]. In this work, we combine real-world data with coarse annotations and synthetic data to train a semantic segmentation model for urban scenes. We make use of synthetic data to provide useful details and boundaries to networks, to predict pseudo labels for the unlabeled regions of coarsely annotated data. We show that just using coarse annotations along with synthetic data can perform comparable with a fraction of annotation budget of expensive fine annotations.

3. Coarse-to-Fine Self-Training Framework

Although coarse annotations could significantly reduce the labelling cost, it is notoriously difficult to achieve a good performance for urban scene segmentation using solely coarse data due to the missing boundary information. To this end, we propose a novel coarse-to-fine self-training

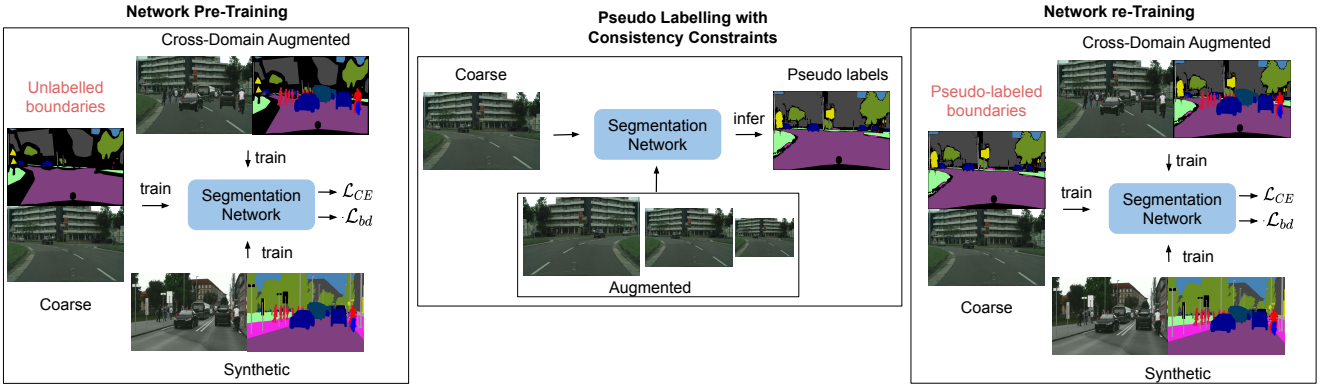

Figure 2: Our coarse-to-fine self-training framework. We propose to improve the coarse annotations by generating pseudo labels. In the Network Pre-Training phase, we use coarse data with unlabelled boundaries and augment synthetic data. We have $\mathcal{L}_{CE}$, classification loss for all data, with additional $\mathcal{L}_{bd}$, boundary loss only for fine detailed synthetic data. After Network Pre-Training, we generate pseudo labels for ignored boundary regions in coarse annotations, followed by Network Re-Training, where we replace the coarse annotation with improved coarse annotations with pseudo labeled boundaries and train iteratively.

framework that utilizes hybrid supervision from coarse and synthetic data and generates pseudo labels at unlabelled regions for re-training the network. Fig. 2 shows an overview of our coarse-to-fine self-training framework, consisting of three stages: 1) network pre-training using sparse coarse annotation and synthetic data, 2) pseudo labelling unlabeled regions in coarse data with consistency constraints, 3) network re-training with pseudo-labelled coarse data and synthetic data. The last two stages can also be performed iteratively to refine the network.

3.1. Segmentation network pre-training

The first stage of our framework is segmentation network pre-training where we learn a strong "teacher" network, e.g., DeepLab-v3+ [5], for pseudo labelling in the second stage. Let $D = (X_c, Y_c) \cup (X_s, Y_s)$ be our training set where X_c denotes real images with coarse annotations Y_c, and X_s denotes synthetic images with fine annotations Y_s. While coarse annotations Y_c are particularly sparse at object boundaries, synthetic annotations Y_s provide dense labels for every pixel in the synthetic images. The key insight of our approach is to capture real distributions from coarse data and learn boundary information from synthetic data. In the following, we describe our cross-domain data augmentation for reducing the domain gap, and the boundary loss for explicit boundary modeling.

Cross-domain augmentation. Inspired by DACS [33], we randomly sample class masks (with probability, $p = 0.5$) from synthetic labels and corresponding image segments and paste them onto real masks and images (see Fig.2 and sec.1 of supplement). We select real samples with probability, $p = 0.5$ from a given training batch and perform this augmentation using the synthetic data sampled uniformly from the whole synthetic dataset, to obtain a new set of training data (X_{aug}, Y_{aug}), i.e. $(X_{aug}, Y_{aug}) = (mask) \otimes (X_c, Y_c) + (1 - mask) \otimes (X_s, Y_s)$ where, $mask$ is sampled from synthetic label and $\otimes$ is elementwise multiplication. This simple strategy alleviates the domain gap and improves data diversity, which is particularly helpful in low-budget regimes. [33] solves unsupervised domain adaptation problem, where labels are not available for real images, whereas for our problem we have manually annotated coarse labels for the real images. Also, we sample synthetic data for augmentation from whole dataset whereas [33] sample synthetic data only from the training batch.

Classification loss. We adopt the standard cross-entropy loss ($\mathcal{L}_{CE}$) on the mixture of coarse data, synthetic data and augmented data, i.e., $(X_c, Y_c) \cup (X_s, Y_s) \cup (X_{aug}, Y_{aug})$.

Boundary loss. As coarse annotations lack proper boundaries, we propose to adopt a boundary loss on the synthetic data, encouraging the network to predict better boundaries. Suppose $\hat{y}_s$ is the predicted label mask of a synthetic image x_s with GT mask y_s, we compute the prediction boundary Γ_{pred} and the ground truth boundary Γ_{GT} using the following equation,

$$\Gamma_{pred} = ||\nabla \hat{y}_s||_2; \Gamma_{GT} = ||\nabla y_s||_2 \quad (1)$$

where ∇ is the gradient operator with central difference approximation. We estimate $\hat{y}_s$ with Gumbel Softmax trick [16] to make it differentiable for backpropagation. For both boundaries, we select pixels with representative boundaries by thresholding them (threshold=$1e^{-8}$ [32]). Assuming p^+_{GT} and p^+_{pred} as the corresponding boundary pixels for ground truth and segmentation prediction masks

after thresholding (ie. p^+_{GT}, p^+_{pred} are points where $\Gamma_{GT} > 1e^{-8}, \Gamma_{pred} > 1e^{-8}$ respectively), we calculate our boundary loss as:

$$\begin{aligned}\mathcal{L}_{bd} = \lambda_1 |\Gamma_{pred}(p^+_{GT}) - \Gamma_{GT}(p^+_{GT})| \\ + \lambda_2 |\Gamma_{pred}(p^+_{pred}) - \Gamma_{GT}(p^+_{pred})|\end{aligned} \tag{2}$$

where we set λ_1, λ_2 to be 0.5 in our experiments. This boundary loss has been used in [32] by applying it on finely annotated images. In contrast, we apply this boundary loss on synthetic data and we do not access any finely annotated data, which is a more challenging setting. This loss term complements the cross-entropy loss by enforcing boundary consistency. We do not apply the boundary loss on coarse data due to its inaccurate boundaries.

3.2. Iterative pseudo labelling and network re-training.

In the second stage, we use the model trained in the previous stage to generate pseudo labels for unlabelled pixels for the coarse training data. We adopt test time augmentation consistency [4, 25] for generating precise pseudo labels. Specifically, we use a combination of flip (flip, no flip) and resize (scale : 0.5, 1, 2.0) for augmentations. For any of the 6 combinations, if pseudo labels disagree, we mark the pixel as ignore. Further we also do confidence thresholding. If prediction confidence obtained after averaging the logits for 6 augmentations, is greater than the threshold (0.9), we accept the pseudo label, otherwise it is marked ignore.

After generating the pseudo labels, we move to the third stage, where we replace the unlabelled pixels in the coarse-annotated images with the pseudo labels. Note that after each iteration we only replace the previous labels from the ignore regions of the coarse data and keep the manually annotated coarse labels intact. Then we re-train the segmentation network using the new coarse data, original synthetic data and augmented data with the same loss functions defined in the first stage. These two stages can be repeated in an iterative manner to refine the network.

3.3. Tackling class imbalance with model-based sampling

Selection of training samples is not trivial. Specifically for lower annotation budgets, where tail distribution classes appear scarcely, it becomes more important to have such samples in training. Also, since we assume that initially, images are unlabelled, it becomes difficult to get training samples from tail end classes. There is an additional manual classification overhead cost for obtaining the class distribution in the training data. We present model-based sampling to ensure pixels from tail end classes are sufficiently present in the training dataset without any manual overhead cost.

Our model-based sampling method makes use of a model trained on an initial set of randomly sampled, 1,000 coarsely annotate images. Using this initial model we estimate the class distribution of available unlabeled images. With the estimated class distribution information, we make sure that we have sufficient samples from each class by sampling almost the same number of data samples having a particular class. We build incrementally on the training samples, i.e., for 2,000 training samples, we use the initial 1,000 training samples and add another 1,000 samples obtained with the help of the initial segmentation model. We compare the performance of our sampling technique with random sampling in Tab. 3 and observe that our model-based sampling can indeed increase performance significantly.

Algorithm 1: coarse-to-fine framework for generating pseudo labels

Data: Coarse data : (X_c, Y_c), Synthetic data : (X_s, Y_s)
Step 1 - Network Pre-Training :

- Do cross domain augmentation on coarse data to get (X_{aug}, Y_{aug}).
- Train segmentation model $f_\theta(x)$ on combined data $(X, Y) = (X_c, Y_c) \cup (X_s, Y_s) \cup (X_{aug}, Y_{aug})$

$$\theta^* = \arg\max_\theta \mathcal{L}(Y, f_\theta(X))$$

where, $\mathcal{L} = \mathcal{L}_{CE}(Y, f(X, \theta)) + \lambda \mathcal{L}_{bd}(Y_s, f(X_s, \theta))$

Step 2 - Pseudo Labelling with Consistency Constraints : Generate pseudo labels from trained network f_{θ^*} for ignored region (X_{ps}, Y_{ps}), following consistency rules.
Step 3 - Network Re-Training : Replace coarse data (X_c, Y_c) with pseudo labels (X_{ps}, Y_{ps}) and retrain iterating steps 2-3

Dataset	Type	train	val	time	% annotated
Cityscapes	Coarse	19998	NA	7 min	63.04
	Fine	2975	500	90 min	99.98
BDD100k	Coarse	4000	NA	7 min	69.81
	Fine	3000	1000	75 min	100
Synscapes	Synthetic	25000	NA	NA	100
GTA-5	Synthetic	24966	NA	NA	100

Table 1: Left: Datasets statistics. train: number of training images, val: number of validation images, time: time to annotate an image, "% annotated": average percentage of pixels annotated per image. Coarse annotation with less labeled pixels are significantly faster to annotate compared to fine annotations.

4. Experiments

In this section, we first describe our experimental setting, then we present our results and comparison with baselines, ablation studies with framework components followed by a qualitative analysis.

Datasets. We use Cityscapes [8] and BDD100k [40] datasets for coarse annotations, as well as Synscapes [35] and GTA-5 [28] datasets for synthetic annotation. Statistics

for these datasets are shown in Tab. 1. Only the Cityscapes dataset provides manually annotated coarse data. As BDD100k does not include coarse annotations, we divide the 7,000 training samples into 3,000 fine annotated samples and 4,000 coarse samples where the coarse annotations are simulated (see supplement Sec.1 and Fig.1 for details). We report results on the standard validation set on Cityscapes and BDD100k.

Cost of annotation The cost for fine annotation of Cityscapes images, where almost all pixels are annotated is around 90 minutes [8] per image including quality control. On the contrary, the cost of coarse annotations is just 7 minutes [8] per image. Further for BDD100k, fine annotations cost is around 75 min compared to just 7 min for coarse annotation per image. We get this annotation cost for BDD100k by manually annotating 10 samples each for coarse and fine annotations via labelme [30]. We consider the cost of synthetic data annotation to be free following [26], as it is generated from photorealistic rendering techniques.

Implementation details We use DeepLab-v3+ with Imagenet-pretrained Xception-71 as the backbone. We use SGD optimizer and "Poly" learning rate scheduler with power=2.0. We also set momentum to 0.9 and weight decay rate to 0.0001. We employ a crop size of 760×760 and a batch size of 12. For each round of training we train for 100 epochs. Further we perform 3 iterations of our self training framework. For evaluation, we use multiscale inferencing with scales $\{0.5,1,2\}$. We use the same hyperparameters for both BDD100k and Cityscapes.

4.1. Comparison with baselines

In this section, we first introduce our baselines followed by showing the performance vs annotation cost trade-off and comparing with the best results of DeepLab-v3+.

Baselines. We take DeepLab-v3+ [5] as the segmentation network and first compare with two popular supervision schemes adopted by most of existing works [5, 44, 41, 38]. The first baseline is trained with only fine data, denoted as DeepLab-v3+ (fine). The second baseline is trained on the combination of fine data and coarse data, denoted as DeepLab-v3+ (fine+coarse). Specifically, we first pretrain the network on the coarse data followed by fine-tuning it on fine data. Second, we also compare with another two intuitive baselines, DeepLab-v3+(Synscapes) and DeepLab-v3+(GTA) where model is trained with synthetic data.

Performance vs annotation cost trade-off. In this experiment, we compare with the baselines under different annotation costs on Cityscapes and BDD100k. For Cityscapes fine data, we draw 50, 100, 200, 400, 800, 1600 and 2975 images from the training set, taking 75, 150, 300, 600, 1200, 2400 and 4462 hours respectively to annotate. Similarly, for coarse data, we draw 500, 1000, 2000, 4000 and 8000 images from the training set, amounted to 58, 116, 233, 467, and 933 annotation hours respectively. Finally, for synthetic data, we randomly draw 500, 1000, 2000, 4000 and 8000 images where we assume the annotation is free. Note that all those training examples for Cityscapes are sampled incrementally based on our proposed model-based sampling strategy described in Sec. 3.3. For BDD100k, we follow the same sampling scheme as Cityscapes.

Fig. 3 shows the results on Cityscapes (left) and BDD100k (right) datasets. We have the following observations. In general, our method achieves the best annotation cost vs performance trade-off. The method ranking follows Ours(coarse +syn(Synscapes/GTA-5)) $>$ DeepLab-v3+ (fine+coarse) $>$ DeepLab-v3+ (fine) $>$ DeepLab-v3+(synthetic) where synthetic can be GTA-5 or Synscapes. While using coarse data for pretraining, i.e., DeepLab-v3+ (fine+coarse), indeed improves fine data alone, i.e., DeepLab-v3+ (fine), our proposed coarse-to-fine framework achieves a significant larger performance boost. On Cityscapes, our method with Synscapes obtains an impressive mIoU of 77.5% with 933 annotation hours, which amount to only one-fifth of the annotation cost (4462 hours) of using full fine data (mIoU is 77.4%). This is encouraging because no prior works report such competitive results without using any fine data.

In addition, we find that the performance gap is larger under a small annotation budget e.g., for Cityscapes experiment, ours achieves 67.5% mIoU with 58 annotation hours vs DeepLab-v3+ (fine)'s 37.2% mIoU with 75 annotation hours. Similarly for BDD100k, ours achieve 49.9% mIoU with 58 annotation hrs vs 25.9% mIoU with 62.5 annotation hrs. This implies that annotating diverse coarse examples is much more important than carefully labelling every pixel when the annotation budget is small, shedding light on annotating large-scale urban scene segmentation datasets. More importantly, these results empirically confirm the effectiveness of our proposed coarse to fine framework.

Comparing with the best results of DeepLab-v3+. We present the best results achieved by DeepLab-v3+ in Tab. 2 for both Cityscapes and BDD100k datasets. For Cityscapes, Compared to DeepLab-v3+ (fine) trained on full training set (Tab. 2), Ours (coarse+syn) is able to achieve a competitive result with only one-fifth of its cost (933 vs 4462 hours). For BDD100k, we achieve comparable performance with only one-eighth of cost wrt full fine training (466 vs 3750 hours). Interestingly, our method tends to perform better on the tail classes with a lower number of pixel instances. For example, our method substantially outperforms DeepLab-v3+ (fine) on train, traffic light, rider and motor for Cityscapes, and wall, bicycle, rider and motorcycle for BDD100k. This is attributed to the improved diversity in training samples from our cross-domain data augmentation. We also compare two pretraining strategies: (1) the model

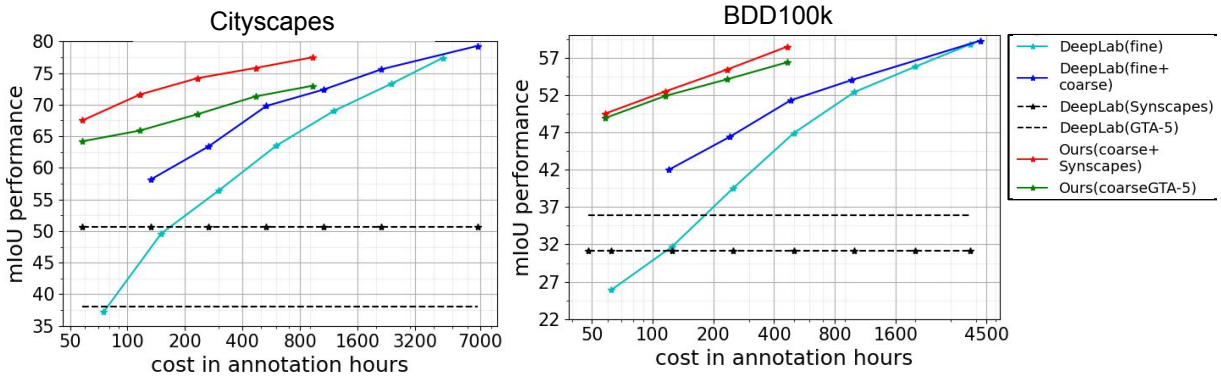

Figure 3: Annotation cost vs performance. Baselines: 1.DeepLab(fine): the standard training strategy with fine data, 2.DeepLab (fine+coarse): pretraining on coarse data following by fine-tuning on fine data, 3.DeepLab(Synscapes/GTA): standard training with synthetic Synscapes/GTA dataset, vs Ours (coarse+syn(Synscapes/GTA-5)): our method trained on the combination of coarse and synthetic data. DeepLab-v3+ is abbreviated as DeepLab.

Cityscapes																					
Method	budget	road	building	vege.	car	sky	sidewalk	fence	terrain	wall	pole	tra. sign	person	bus	truck	bicycle	train	tra. light	rider	motor.	mIoU
DeepLab-v3+ (fine)	4462	**98.0**	**92.2**	**92.2**	**94.9**	**94.8**	**84.5**	58.4	**62.7**	56.8	**61.2**	**76.1**	80.4	88.7	81.9	**75.2**	80.6	65.8	60.6	62.4	77.4
Ours (coarse + syn)	**933**	97.2	91.8	90.8	94.0	94.3	78.2	**60.7**	55.8	**58.8**	61.1	75.5	**80.5**	**92.0**	**82.1**	74.5	**85.6**	**69.8**	**64.7**	**65.5**	**77.5**
Pretrain (coarse)	6795	98.3	92.7	92.5	95.3	95.1	86.4	65.2	61.4	54.2	65.9	78.9	81.9	90.7	**86.3**	77.1	83.8	69.4	64.8	67.5	79.3
Pretrain (ours)	6795	**98.4**	**93.5**	**93.2**	**96.0**	**95.5**	**87.3**	**66.3**	**65.1**	**54.7**	**71.2**	**82.5**	**85.5**	**92.6**	84.5	**80.8**	**88.0**	**75.3**	**70.7**	**72.7**	**81.8**

BDD100k																					
Method	budget	road	car	sky	pole	vege.	building	tra. sign	sidewalk	tra-light	terrain	person	truck	fence	bus	wall	bicycle	rider	motor.	train	mIoU
DeepLab-v3+ (fine)	**3750**	**94.6**	**90.2**	**95.2**	**48.3**	**85.7**	**85.4**	49.2	**62.5**	48.3	44.3	64.3	**57.5**	45.7	**75.5**	27.1	47.8	46.3	48.8	0	**58.8**
Ours (coarse + syn)	466	93.0	87.7	91.0	40.2	82.9	82.8	**49.5**	60.5	**50.8**	**44.5**	**65.7**	51.4	**49.3**	74.8	**31.4**	**50.0**	**56.5**	**49.7**	0	58.5
Pretrain	4216	94.7	89.7	95.2	46.5	85.7	84.9	48.2	63.7	47.1	46.9	61.2	56.2	49.6	77.9	37.8	46.9	43.7	50.5	0	59.3
Pretrain (ours)	**4216**	**95.3**	**91.2**	**95.6**	**57.1**	**87.1**	**86.8**	**58.9**	**67.7**	**59.7**	**48.3**	**69.9**	**58.3**	**51.9**	**82.9**	**35.3**	**59.3**	**55.4**	**55.9**	**0.4**	**64.1**

Table 2: Comparing with the best results of DeepLab-v3+. We report per-class IoU as well as mean IoU (mIoU). The class names are sorted in decreasing order of the class-wise image distribution. We show results for Cityscapes (top block) and BDD100k (bottom block). For each, we have two comparisons - 1) our results without using fine data vs DeepLab-v3+ uses all available fine data; our method performs better on tail classes with overall comparable performance at one-fifth (Cityscapes) and one-eighth (BDD100k) budget. 2) Pretraining with our framework vs pretraining with coarse data. Our model serves as a better pretraining method than directly coarse data pretraining.

trained on coarse data following the conventional way [5], (2) the model trained on coarse and synthetic data using our framework. As shown in Tab. 2, pretraining with our framework leads to a significant improvement of 2.5 mIoU for Cityscapes and 4.8 mIoU for BDD100k datasets. These results imply that our method could serve as a promising pretraining strategy for the best performance.

4.2. Ablation study on model components and hyperparameters

In this section, we conduct the ablation studies on cross-domain augmentation, boundary loss, impact of self-training iterations and comparison with a semi-supervised learning method. The ablation experiments are conducted on Cityscapes with Synscapes being the synthetic dataset.

Effect of cross domain augmentation. In our method, we perform the cross-domain augmentation by copy-pasting

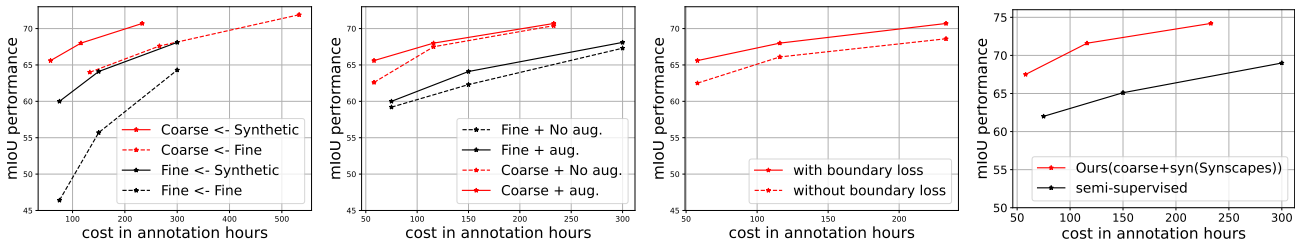

Figure 4: Left to Right: 1) Synthetic vs Fine for cross domain augmentation. 2) Importance of cross domain augmentation. 3) Importance of boundary loss. 4) Comparison with a semi-supervised learning baseline [4] on Cityscapes.

Coarse Samples	500	1K	2K	4K	8K
Iteration 0	65.6	68.0	70.7	73.4	74.5
Iteration 1	67.1	**71.6**	**74.2**	**75.8**	76.6
Iteration 2	**67.5**	70.4	73.4	74.4	**77.5**

Table 3: Self-training iterations using a mixture of coarse annotated data and synthetic data. Performance in different self-training iterations of our framework. Our framework achieves optimal performance in iteration 1 for most experiments itself and we do not need to perform more iterations.

Coarse Samples	1K	2K	4K	8K
Model-based	57.4	**62.3**	**66.4**	**68.4**
uniform sampling	57.4	59.7	64.5	66.5

Table 4: Model-based class balanced sampling vs random sampling. Our model based iterative sampling generates diverse training samples compared to uniform sampling.

synthetic objects onto coarse images i.e., Coarse ← Synthetic. In this study, we ablate other augmentation choices given two domains of data i.e., Coarse ← Fine, Fine ← Synthetic, Fine ← Fine. As shown in Fig. 4 (left), applying augmentation from Synthetic to Coarse data works the best. We also perform ablation to study the importance of cross-domain augmentation for our self-training framework. We present ablation results for both fine and coarse data in Fig. 4 (mid). We observe improvement in performance by using cross-domain augmentation for both fine and coarse data. In particular, the improvement for the lowest budget case (i.e., 58 hours) using coarse data is siginificantly improved (by a mIoU of 3%) by mixing data from Synscapes, where it provides sufficient samples with important details not present in the original data with coarse annotation.

Effect of boundary loss. We present the ablation of the boundary loss in Fig. 4 (mid). Even though synthetic data provides necessary boundary information, we can observe that training a network with only cross-entropy loss performs worse than additionally applying the boundary loss. Specifically, we observe performance gains of 3.1, 1.9 and 2.1 mIoU compared with the non-boundary version for annotation budgets of 58, 116 and 233 hrs, which corresponds to using 500, 1000 and 2000 images with coarse annotation.

Number of self-training iterations. The last two stages of our coarse-to-fine self-training framework can be trained iteratively. In Tab. 3, we present the performance improvement at different iterations with 500, 1000, 2000, 4000 and 8000 coarse samples. Iteration 0 corresponds to network pre-training stage, as explained in Algorithm 1. After iteration 0, we generate pseudo labels for the ignored regions in the coarse annotated images and create new GT for the next iteration. There are clear improvements from iteration 0 to 1, as the model touches newly annotated pixels which are ignored in the iteration 0. At iteration 2, the performance may slightly decrease in some cases due to the inevitable errors in pseudo labels.

Which examples should be labeled? Coarse annotation is not expensive to acquire, but it is interesting to know which examples to label earlier. We gradually expand the training examples from 1000 to 8000. At every step, we sample new coarse examples to train a model as discussed in Sec. 3.3. We compare our sampling strategy with uniform sampling in Tab. 3. Apparently, we can see the effectiveness of the model-based sampling and better results can be obtained.

Comparison with semi-supervised learning. We also compare our learning framework with a semi-supervised learning method [4] in Fig. 4(right). We adapt the codes provided by [4] for this comparison. For this experiment, we use the same coarse data samples for a given budget as our other experiments. We treat the coarse data samples as unlabelled data by not using its annotation mask. Similary, We use same the fine data samples as used in our baselines for a given budget.Our framework achieves 67.5% in a budget of 58 annotation hours vs 62.0% by semi-supervised approach with 75 hrs annotation budget (see 4(right)). Similarly, our framework with a budget of 116 and 233 annotation hours outperforms the semi-supervised approach by a gap of 2.4% and 1.4% mIoU with budgets 150 and 300

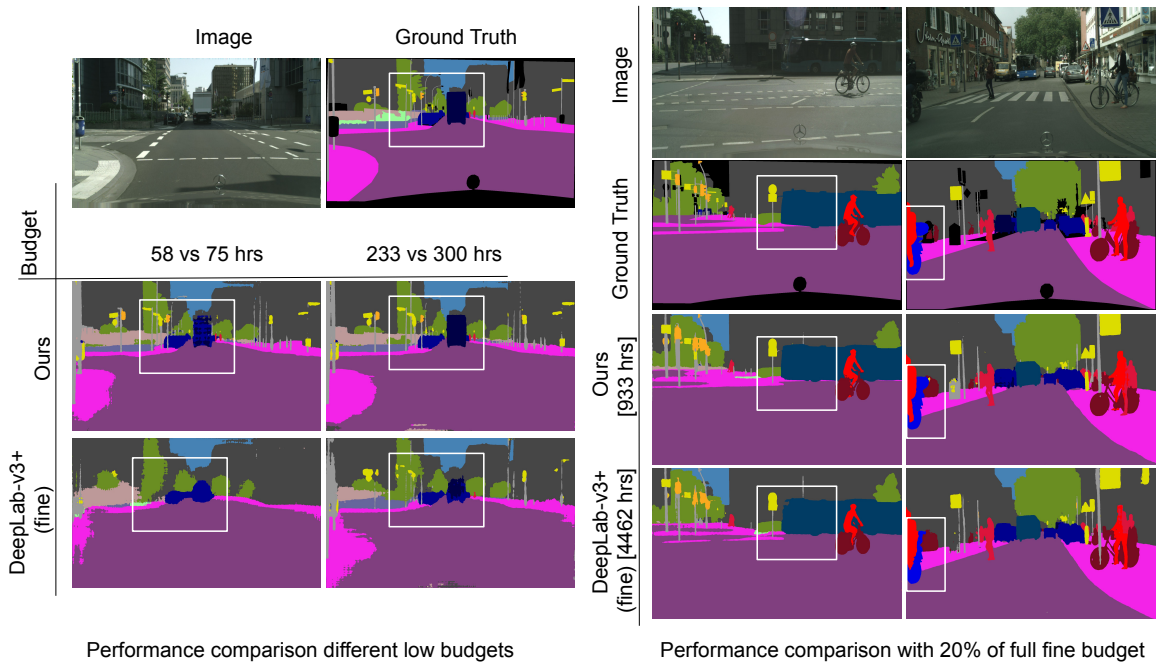

Figure 5: Qualitative comparison of ours vs DeepLab-v3+(fine) as baseline on Cityscapes dataset. Left: Qualitative performance for an image on different annotation budgets i.e. 58(ours) vs 75(baseline) and 233(ours) vs 300(baseline) annotation hours. Right: Qualitative comparison with 20% of full fine budget (933 vs 4462 annotation hours). Even with 5 $\times$ cheaper budget, our coarse-to-fine framework performs comparable with full fine annotation budget. Further, on some tails end classes, it even performs better(e.g., motorbike, bus, as highlighted). Region of interests is present in white bounding boxes.

hours respectively. This shows the importance of coarse annotation, which along with synthetic data can be efficiently used to obtain competitive performance.

4.3. Qualitative results

We visualize the qualitative comparison of the baseline with our framework in Fig. 5. In the left, we provide a comparison with the baseline trained with the fine annotated data at the budgets of 75 and 300 hours, which is slightly more than our versions at 58 and 233 hours. We observe that the performance of small objects and the region with rich details are improved. For instance, we highlight the truck and the traffic-light are predicted correctly from our model with 233 hours, while the model using fine annotation cannot recognize them well. Furthermore, our model with 58 hours can still recognize the majority of those objects, while the comparing method with 75 hours directly ignore those objects even though its training images have labels for every pixel. Also, baseline fails to predict traffic-light for both budgets, while our method succeeds. In the right, we provide a comparison with full fine annotation budget. Our framework with only one-fifth of full budget (933 vs 4462) is able to perform equally well with baseline as can be observed qualitatively. Moreover, the qualitative results also confirm that our framework performs better on tail classes (see Tab. 2). For instance bus and motorcycle (right, column1 and column 2 respectively) compared to baseline.

5. Conclusion

In this work, we argue that coarsely annotated data has been largely ignored as a primary training source. Therefore, we propose a new supervision scheme based on coarse data and synthetic data, significantly reducing annotation time. We develop a strong baseline that efficiently learns from coarse and synthetic data by combing self-training, a boundary loss and cross-domain data augmentation. We conduct extensive experiments to evaluate our method on Cityscapes and BDD100k datasets with two different synthetic datasets i.e., Synscapes and GTA-5. The experimental results show that our method achieves the best performance vs. annotation cost trade-off when compared to the standard supervision schemes with fine annotated data. More importantly, our method achieves competitive performance for Cityscapes compared to the state of the art with only one-fifth of its annotation budget. We hope our method inspires more future works along this challenging but rewarding research direction.

References

[1] Amy Bearman, Olga Russakovsky, Vittorio Ferrari, and Li Fei-Fei. What's the point: Semantic segmentation with point supervision. In *ECCV*, 2016.

[2] Maxime Bucher, Tuan-Hung Vu, Matthieu Cord, and Patrick Pérez. Zero-shot semantic segmentation. *Advances in Neural Information Processing Systems*, 32:468–479, 2019.

[3] Wei-Lun Chang, Hui-Po Wang, Wen-Hsiao Peng, and Wei-Chen Chiu. All about structure: Adapting structural information across domains for boosting semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1900–1909, 2019.

[4] Liang-Chieh Chen, Raphael Gontijo Lopes, Bowen Cheng, Maxwell D Collins, Ekin D Cubuk, Barret Zoph, Hartwig Adam, and Jonathon Shlens. Naive-student: Leveraging semi-supervised learning in video sequences for urban scene segmentation. In *European Conference on Computer Vision*, pages 695–714. Springer, 2020.

[5] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 801–818, 2018.

[6] Xiaokang Chen, Yuhui Yuan, Gang Zeng, and Jingdong Wang. Semi-supervised semantic segmentation with cross pseudo supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

[7] Sungha Choi, Sanghun Jung, Huiwon Yun, Joanne T Kim, Seungryong Kim, and Jaegul Choo. Robustnet: Improving domain generalization in urban-scene segmentation via instance selective whitening. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11580–11590, 2021.

[8] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016.

[9] Anurag Das, Yongqin Xian, Yang He, Bernt Schiele, and Zeynep Akata. $(sp)^2$net for generalized zero-label semantic segmentation. In *DAGM German Conference on Pattern Recognition*, pages 235–249. Springer, 2021.

[10] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 2010.

[11] Golnaz Ghiasi, Barret Zoph, Ekin D. Cubuk, Quoc V. Le, and Tsung-Yi Lin. Multi-task self-training for learning general representations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 8856–8865, October 2021.

[12] Yang He, Wei-Chen Chiu, Margret Keuper, and Mario Fritz. Std2p: Rgbd semantic segmentation using spatio-temporal data-driven pooling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4837–4846, 2017.

[13] Qibin Hou, Li Zhang, Ming-Ming Cheng, and Jiashi Feng. Strip pooling: Rethinking spatial pooling for scene parsing. In *CVPR*, pages 4003–4012, 2020.

[14] Zilong Huang, Xinggang Wang, Jiasi Wang, Wenyu Liu, and Jingdong Wang. Weakly-supervised semantic segmentation network with deep seeded region growing. In *CVPR*, pages 7014–7023, 2018.

[15] Wei-Chih Hung, Yi-Hsuan Tsai, Yan-Ting Liou, Yen-Yu Lin, and Ming-Hsuan Yang. Adversarial learning for semi-supervised semantic segmentation. *arXiv preprint arXiv:1802.07934*, 2018.

[16] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

[17] Anna Khoreva, Rodrigo Benenson, Jan Hendrik Hosang, Matthias Hein, and Bernt Schiele. Simple does it: Weakly supervised instance and semantic segmentation. In *CVPR*, 2017.

[18] Daiqing Li, Junlin Yang, Karsten Kreis, Antonio Torralba, and Sanja Fidler. Semantic segmentation with generative models: Semi-supervised learning and strong out-of-domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

[19] Peike Li, Yunchao Wei, and Yi Yang. Consistent structural relation learning for zero-shot segmentation. *Advances in Neural Information Processing Systems*, 33, 2020.

[20] Di Lin, Jifeng Dai, Jiaya Jia, Kaiming He, and Jian Sun. Scribblesup: Scribble-supervised convolutional networks for semantic segmentation. In *CVPR*, 2016.

[21] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*. Springer, 2014.

[22] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015.

[23] Seong Joon Oh, Rodrigo Benenson, Anna Khoreva, Zeynep Akata, Mario Fritz, Bernt Schiele, et al. Exploiting saliency for object segmentation from image level labels. In *CVPR*, 2017.

[24] George Papandreou, Liang-Chieh Chen, Kevin Murphy, and Alan L Yuille. Weakly-and semi-supervised learning of a dcnn for semantic image segmentation. In *ICCV*, 2015.

[25] Giuseppe Pastore, Fabio Cermelli, Yongqin Xian, Massimiliano Mancini, Zeynep Akata, and Barbara Caputo. A closer look at self-training for zero-label semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2693–2702, 2021.

[26] Sujoy Paul, Yi-Hsuan Tsai, Samuel Schulter, Amit K Roy-Chowdhury, and Manmohan Chandraker. Domain adaptive semantic segmentation using weak labels. In *European conference on computer vision*, pages 571–587. Springer, 2020.

[27] Stephan R Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for data: Ground truth from computer games. In *European conference on computer vision*, pages 102–118. Springer, 2016.

[28] Stephan R. Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for data: Ground truth from computer games. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *European Conference on Computer Vision (ECCV)*, volume 9906 of *LNCS*, pages 102–118. Springer International Publishing, 2016.

[29] German Ros, Laura Sellart, Joanna Materzynska, David Vazquez, and Antonio M Lopez. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3234–3243, 2016.

[30] Bryan C Russell, Antonio Torralba, Kevin P Murphy, and William T Freeman. Labelme: a database and web-based tool for image annotation. *IJCV*, 77(1-3):157–173, 2008.

[31] Nasim Souly, Concetto Spampinato, and Mubarak Shah. Semi supervised semantic segmentation using generative adversarial network. In *Proceedings of the IEEE international conference on computer vision*, pages 5688–5696, 2017.

[32] Towaki Takikawa, David Acuna, Varun Jampani, and Sanja Fidler. Gated-scnn: Gated shape cnns for semantic segmentation. In *ICCV*, pages 5229–5238, 2019.

[33] Wilhelm Tranheden, Viktor Olsson, Juliano Pinto, and Lennart Svensson. Dacs: Domain adaptation via cross-domain mixed sampling. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1379–1389, 2021.

[34] Yi-Hsuan Tsai, Wei-Chih Hung, Samuel Schulter, Kihyuk Sohn, Ming-Hsuan Yang, and Manmohan Chandraker. Learning to adapt structured output space for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7472–7481, 2018.

[35] Magnus Wrenninge and Jonas Unger. Synscapes: A photorealistic synthetic dataset for street scene parsing. *arXiv preprint arXiv:1810.08705*, 2018.

[36] Yongqin Xian, Subhabrata Choudhury, Yang He, Bernt Schiele, and Zeynep Akata. Semantic projection network for zero-and few-label semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8256–8265, 2019.

[37] Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V. Le. Self-training with noisy student improves imagenet classification. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[38] Minghao Yin, Zhuliang Yao, Yue Cao, Xiu Li, Zheng Zhang, Stephen Lin, and Han Hu. Disentangled non-local neural networks. In *ECCV*, pages 191–207. Springer, 2020.

[39] Changqian Yu, Jingbo Wang, Changxin Gao, Gang Yu, Chunhua Shen, and Nong Sang. Context prior for scene segmentation. In *CVPR*, pages 12416–12425, 2020.

[40] Fisher Yu, Wenqi Xian, Yingying Chen, Fangchen Liu, Mike Liao, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving video database with scalable annotation tooling. *arXiv preprint arXiv:1805.04687*, 2(5):6, 2018.

[41] Yuhui Yuan, Xiaokang Chen, Xilin Chen, and Jingdong Wang. Segmentation transformer: Object-contextual representations for semantic segmentation. In *ECCV*, volume 1, 2021.

[42] Xiangyu Yue, Yang Zhang, Sicheng Zhao, Alberto Sangiovanni-Vincentelli, Kurt Keutzer, and Boqing Gong. Domain randomization and pyramid consistency: Simulation-to-real generalization without accessing target domain data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2100–2110, 2019.

[43] Fan Zhang, Yanqin Chen, Zhihang Li, Zhibin Hong, Jingtuo Liu, Feifei Ma, Junyu Han, and Errui Ding. Acfnet: Attentional class feature network for semantic segmentation. In *ICCV*, pages 6798–6807, 2019.

[44] Xiong Zhang, Hongmin Xu, Hong Mo, Jianchao Tan, Cheng Yang, Lei Wang, and Wenqi Ren. Dcnas: Densely connected neural architecture search for semantic image segmentation. In *CVPR*, pages 13956–13967, 2021.

[45] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *CVPR*, pages 2881–2890, 2017.

[46] Hengshuang Zhao, Yi Zhang, Shu Liu, Jianping Shi, Chen Change Loy, Dahua Lin, and Jiaya Jia. Psanet: Point-wise spatial attention network for scene parsing. In *ECCV*, pages 267–283, 2018.

[47] Yi Zhu, Karan Sapra, Fitsum A Reda, Kevin J Shih, Shawn Newsam, Andrew Tao, and Bryan Catanzaro. Improving semantic segmentation via video propagation and label relaxation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

[48] Barret Zoph, Golnaz Ghiasi, Tsung-Yi Lin, Yin Cui, Hanxiao Liu, Ekin Dogus Cubuk, and Quoc Le. Rethinking pre-training and self-training. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020*, 2020.