ELDA: Using Edges to Have an Edge on Semantic Segmentation Based UDA

Anonymous L3D-IVU submission

Paper ID 49

Abstract

In this paper, we introduce Edge Learning based Domain Adaptation (ELDA), a novel unsupervised domain adaptation (UDA) framework which incorporates edge information into its training process to serve as a type of domain invariant information. In our experiments, we quantitatively and qualitatively demonstrate that the incorporation of edge information is indeed beneficial and effective, as it enables ELDA to outperform the contemporary state-of-the-art methods on two commonly adopted benchmarks for semantic segmentation based UDA tasks. We further provide ablation analysis to justify the decisions of ELDA.

1. Introduction

A number of approaches have been explored to tackle the challenge of *unsupervised domain adaptation (UDA)*, including adversarial training [1–4], anchoring [5–8], and pseudo labeling (PL) [9–12], and have achieved remarkable adaptation performance. However, they rely solely on the semantic labels in the source domain and the raw input data in the target domain, which limit their performance and thus leave room for further improvements. In light of these shortcomings, another branch of work has incorporated domain invariant information into their training processes to help bridge the domain gaps. Domain invariant information possesses a favorable property: the concept it represents is general across different domains. This property makes it highly desirable for UDA tasks as it is robust against domain gaps. As a result, such property has inspired researchers to explore the usage of domain invariant information in their UDA methods, in which it is oftentimes embedded into the training objectives of some auxiliary tasks. A commonly adopted type of domain invariant information is depth, which contains clues relating to the distance of the surfaces of scene objects from a viewpoint. A number of methods have been proposed to leverage depth information [13–16]. Recently, [17] proposed to utilize self-supervised learning (SSL) techniques to retrieve depth information with the aim of assisting semantic segmentation based UDA tasks and achieved remarkable performance.

Unfortunately, the methods that utilized SSL to retrieve depth information have two crucial constraints: First, the computational cost of training an accurate auxiliary SSL-based depth estimation model is often expensive. A few researchers [17] even employed two separate depth estimation models in both the source and the target domains to ensure the quality of the generated depth estimation. This worsens the computational burden incurred and makes it less suitable for real world applications. Second, since SSL-based models have no access to ground truth labels, their performance is not comparable to physical sensors or supervised models [18] in terms of accuracy. In other words, their predictions might deviate from the ground truths and bring negative impacts on the training processes of semantic segmentation based UDA models.

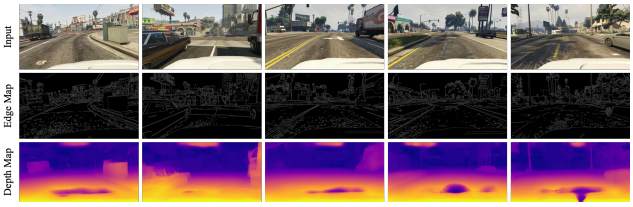

Figure 1. A example showing the differences of the depth maps extracted by [18] and the edges on the images from GTA5 [19] without the use of any ground truth labels. For the depth maps, nearer surfaces are brighter, while further surfaces are darker.

Being aware of the problems associated with using SSL based depth estimation to assist in the training processes of UDA models, we propose to replace it with edges, which is also a type of domain invariant information. The benefits are twofold. First, the computational cost of extracting edges from an input image is substantially lower than those of extracting a depth map from the same image using SSL. Edges in an image can be obtained by performing convolution using certain fixed kernels over the input image in one pass [20], while the extraction of a depth map usually requires a sophisticated SSL model [18]. Second, the quality of edges is typically much more consistent than that of depth, as depth estimation using only RGB images is an ill-posed problem [21], and is susceptible to the influences of noises, model architectures, and data distributions. On the contrary, edges are relatively consistent, and less likely to deviate from the ground truth. Fig. 1 depicts a motivational example of such characteristics, in which the object

boundaries are better captured in the extracted edge maps. In contrast, the depth maps are noisy and the object boundaries are much more ambiguous. The availability of such a high quality boundary information thus offers a promising way to enable a model to better adapt to a target domain.

In order to validate the aforementioned motivations, and take full advantage of the high quality edge information, we propose **E**dge **L**earning based **D**omain **A**daptation, abbreviated as **ELDA**. ELDA utilizes edges as the domain invariant information by incorporating edge extraction into the training process of a UDA model as an auxiliary task. The experimental results show that ELDA is able to achieve the state-of-the-art performance on two commonly adopted benchmarks [19,22,23]. The contributions of this work are:

- We introduce the use of edge information as an auxiliary task for semantic segmentation based UDA, and develop an effective framework named ELDA, that is able to take advantage of the valuable edge information embedded in the input images of different domains.
- We highlight the cost and quality issues of UDA methods that leverage depth estimation as an auxiliary task.
- We validate ELDA on two common UDA benchmarks quantitatively and qualitatively, and show that it is able to achieve superior performance over the baselines.
- We demonstrate that by incorporating edge estimation into the framework as an auxiliary task ELDA is able to capture fine-grained features in the target domain.

2. Methodology

2.1. Problem Formulation

In UDA tasks, a model has access to a source dataset $X_s = \{x_s^1, ..., x_s^N\}$, $Y_s = \{y_s^1, ..., y_s^N\}$, and a target dataset $X_t = \{x_t^1, ..., x_t^M\}$, where N and M denote the number of instances from the source and the target domains, respectively. A tuple (x_s, y_s) represents an image-label pair from the source domain, and x_t denotes a target domain image. The objective is to train a model such that its predictions can best estimate the ground truth labels in the target domain.

2.2. Architecture Design of the ELDA Framework

Fig. 2 illustrates an overview of the ELDA framework.

2.2.1 Shared Domain Invariant Encoder (SDI-Enc).

In auxiliary-task learning, the concept of shared encoder architectures is usually adopted to extract common features so as to enhance performance [24]. Inspired by this, ELDA also employs the shared encoder technique [17] for capturing both edge and segmentation features.An input image from either the source domain or the target domain is fed into the shared encoder to extract a shared feature f_{shared}.

2.2.2 Task Specific Branch (TSB).

To enable f_{shared} to be further interpreted into specific feature embeddings that bear edge and segmentation meanings, two separate branches of TSBs, similar to those used in [17, 24], are utilized to generate initial edge and segmentation predictions. The two TSBs contain their separate encoders and decoders. The encoders are in charge of encrypting f_{shared} into task specific features f_{edge} and f_{seg}, which are later fed to the CM. On the other hand, the decoders are responsible for decoding f_{edge} and f_{seg} into $\hat{e}_s^{\text{init}}$ or $\hat{e}_t^{\text{init}}$ and $\hat{y}_s^{\text{init}}$ or $\hat{y}_t^{\text{init}}$, respectively, depending on the original domains of the input images, for updating SDI-Enc and the TSBs.

2.2.3 Correlation Module (CM).

With the goal of communicating information between the task specific latent embeddings f_{edge} and f_{seg}, we employ a correlation module [17, 25] into the ELDA architecture. This operation helps the model to preserve the essential features from the two TSBs. CM can be formulated as follows:

$$f_{\text{seg}}^{\text{mid}} = \text{Conv}(f_{\text{seg}}), \quad f_{\text{edge}}^{\text{mid}} = \text{Conv}(f_{\text{edge}}), \tag{1}$$

$$\begin{aligned} f_{\text{seg}}^{\text{cm}} &= f_{\text{seg}} + f_{\text{edge}}^{\text{mid}} * \text{Sigmoid}(\text{Conv}(f_{\text{edge}})), \\ f_{\text{edge}}^{\text{cm}} &= f_{\text{edge}} + f_{\text{seg}}^{\text{mid}} * \text{Sigmoid}(\text{Conv}(f_{\text{seg}})), \end{aligned} \tag{2}$$

where $\text{Conv}(\cdot)$ and $\text{Sigmoid}(\cdot)$ are the convolution and sigmoid functions, and $f_{\text{seg}}^{\text{cm}}$ and $f_{\text{edge}}^{\text{cm}}$ the output embeddings.

2.3. Loss Function Design

2.3.1 The Loss Function for Edge Estimation.

In ELDA, the supervision targets for edges in both the source and the target domains are generated using the Canny edge extraction algorithm [26], denoted as an operator $\text{C}(\cdot)$. The edge loss $L_{\text{edge}} = L_{\text{edge}}^{\text{init}} + L_{\text{edge}}^{\text{final}}$ is then computed between the edge predictions from ELDA and the edges generated by $\text{C}(\cdot)$. Both $L_{\text{edge}}^{\text{init}}$ and $L_{\text{edge}}^{\text{final}}$ are derived by extending the DICE loss [27], as it is able to prevent data imbalance between edges and background. The expressions of $L_{\text{edge}}^{\text{init}}$ and $L_{\text{edge}}^{\text{final}}$ are formulated as the following:

$$\begin{aligned} \text{L}_{\text{edge}}^{\text{init}} &= (1 - \text{D}(\text{C}(x_s), \hat{e}_s^{\text{init}})) + (1 - \text{D}(\text{C}(x_t), \hat{e}_t^{\text{init}})), \\ \text{L}_{\text{edge}}^{\text{final}} &= (1 - \text{D}(\text{C}(x_s), \hat{e}_s^{\text{final}})) + (1 - \text{D}(\text{C}(x_t), \hat{e}_t^{\text{final}})), \end{aligned} \tag{3}$$

$$\text{D}(e, \hat{e}) = \frac{2\sum_{i=1}^{P} e\hat{e}}{\sum_{i=1}^{P} e + \sum_{i=1}^{P} \hat{e}}, \quad 0 \le D \le 1, \tag{4}$$

where P represents the number of pixels in an image, $\text{D}(\cdot)$ denotes the DICE loss operator, $e \in \{0, 1\}$ represents the edges generated by $\text{C}(\cdot)$, and $\hat{e} \in [0, 1]$ denotes the edges predicted by ELDA. $\hat{e}$ can be any of $\hat{e}_s^{\text{init}}$, $\hat{e}_t^{\text{init}}$, $\hat{e}_s^{\text{final}}$, or $\hat{e}_t^{\text{final}}$.

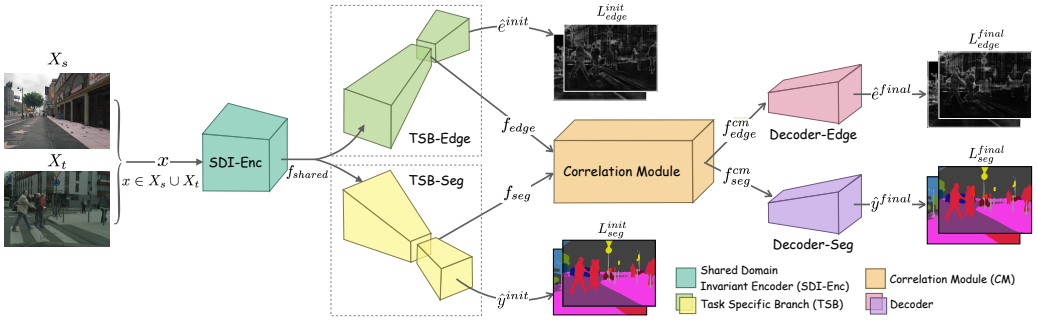

Figure 2. An overview of the proposed ELDA framework.

2.3.2 The Loss Function for Semantic Segmentation.

The loss for semantic segmentation $L_{\text{seg}} = L_{\text{seg}}^{\text{init}} + L_{\text{seg}}^{\text{final}}$ utilizes the cross-entropy (CE) operator $\text{CE}(\cdot)$ [28]. The expressions of its components are formulated as follows:

$$\begin{aligned} L_{\text{seg}}^{\text{final}} &= \text{CE}(y_s, \hat{y}_s^{\text{final}}) + \text{CE}(y_t^{'}, \hat{y}_t^{\text{final}}), \\ L_{\text{seg}}^{\text{init}} &= \text{CE}(y_s, \hat{y}_s^{\text{init}}) + \text{CE}(y_t^{'}, \hat{y}_t^{\text{init}}), \end{aligned} \tag{5}$$

$$\text{CE}(y, \hat{y}) = -\sum_{i=1}^{P} y \log \hat{y}, \tag{6}$$

where $y_t^{'}$ represents the pseudo-labels in the target domain, and y denotes the segmentation labels, which can be either y_s or $y_t^{'}$. On the other hand, $\hat{y}$ denotes the predicted segmentation maps, and can be any of $\hat{y}_s^{\text{init}}$, $\hat{y}_t^{\text{init}}$, $\hat{y}_s^{\text{final}}$, or $\hat{y}_t^{\text{final}}$.

2.3.3 The Total Loss.

The total loss of ELDA can thus be formulated as follows:

$$L_{\text{total}} = L_{\text{seg}} + \lambda L_{\text{edge}}, \quad \lambda \text{ is a balancing factor}, \tag{7}$$

where λ is set to 0.01 and 1 for the GTA5→Cityscapes and the SYNTHIA→Cityscapes benchmarks, respectively.

3. Experimental Results

3.1. Baselines and Evaluation Methods.

We evaluate and compare the experimental results of ELDA against the pure semantic segmentation-based UDA methods [5,6,9,11,12,29], as well as the methods that take advantage of additional information in the form of auxiliary tasks [13–15,17,30]. We evaluate ELDA and the above baselines on two common UDA benchmarks: GTA5 [19]→ Cityscapes [22] and SYNTHIA [23]→ Cityscapes. The models are trained using 2,975 images from Cityscapes, along with either 24,966 image-label pairs from GTA5 or 9,400 image-label pairs from SYNTHIA. They are then evaluated on the validation set of Cityscapes.

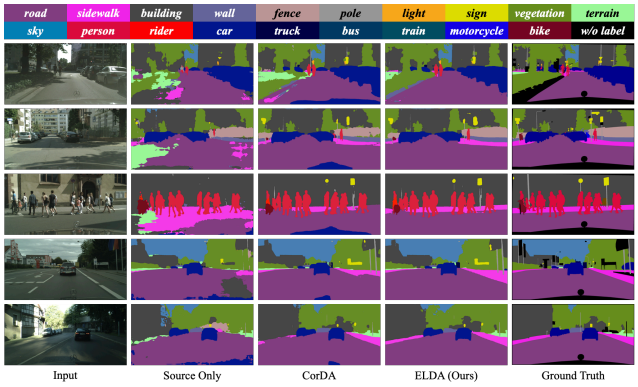

Figure 3. Semantic segmentation results on GTA5→Cityscapes.

3.2. Quantitative Results on the Benchmarks

Tables 1 and 2 compare the experimental results of our method against multiple recent state-of-the-art approaches. Please note that these baselines exclude the works that resort to ensemble distillation methods [6,31,32] or transformer based architectures [33] for fair comparisons. Table 1 reports the experimental results on the GTA5→Cityscapes benchmark. By leveraging edge prediction as an auxiliary task, ELDA achieves an mIoU of 57.3%, outperforming *source only* (i.e., the model only trained on the images from the source domain) and the current state-of-the-art CorDA [17] by 20% mIOU and 0.7% mIoU, respectively. Table 2 shows the results on the SYNTHIA→Cityscapes benchmark. ELDA is able to reach the state-of-the-art performance of 55.2% mIOU, and outperforms all the baselines. Please note that in this benchmark, the depth ground truth labels in the source domain are provided, and are leveraged by a few baseline approaches [13–15,17,30] in their auxiliary tasks. In contrast, ELDA is able to achieve superior performance to the baselines without the assistance of any extra labeled data.

3.3. Ablation Study

As described in Section 2, ELDA is made up of multiple key components such as SDI-Enc, TSB, and CM. To examine the contributions of them to the overall performance, we

GTA5 → Cityscapes																				
Method	Aux.	Road	SideW	Build	Wall	Fence	Pole	Light	Sign	Veg	Terrain	Sky	Person	Rider	Car	Truck	Bus	Train	Motor	Bike	mIoU
Source only		70.1	18.4	66.1	12.8	17.4	22.1	30.8	16.1	79.1	14.4	71.3	57.1	23.7	77.5	29.5	37.0	4.9	29.6	31.5	37.3
CBST [9]		91.8	53.5	80.5	32.7	21.0	34.0	28.9	20.4	83.9	34.2	80.9	53.1	24.0	82.7	30.3	35.9	16.0	25.9	42.8	45.9
CAG-UDA [5]		90.4	51.6	83.8	34.2	27.8	38.4	25.3	48.4	85.4	38.2	78.1	58.6	34.6	84.7	21.9	42.7	**41.1**	29.3	37.2	50.2
Uncertainty [29]		90.4	31.2	85.1	36.9	25.6	37.5	**48.8**	48.5	85.3	34.8	81.1	64.4	**36.8**	86.3	34.9	52.2	1.7	29.0	44.6	50.3
IAST [11]		93.8	57.8	85.1	39.5	26.7	26.2	43.1	34.7	84.9	32.9	88.0	62.6	29.0	87.3	39.2	49.6	23.2	34.7	39.6	51.5
DACS [12]		89.9	39.7	87.9	30.7	39.5	38.5	46.4	52.8	88.0	44.0	88.8	**67.2**	35.8	84.5	45.7	50.2	0.0	27.3	34.0	52.1
ProDA* [6]		91.5	52.4	82.9	**42.0**	35.7	40.0	44.4	43.3	87.0	43.8	79.5	66.5	31.4	86.7	41.1	52.5	0.0	**45.4**	53.8	53.7
CorDA [17]	✓	94.7	63.1	87.6	30.7	40.6	40.2	47.8	51.6	87.6	47.0	**89.7**	66.7	35.9	**90.2**	48.9	**57.5**	0.0	39.8	**56.0**	56.6
ELDA (Ours)	✓	**94.9**	**64.1**	**88.2**	35.0	**44.7**	**40.3**	47.0	**54.6**	**88.7**	**47.4**	88.9	67.0	31.1	**90.2**	**53.7**	56.0	0.0	41.7	55.5	**57.3**

Table 1. The quantitative results on the GTA5→Cityscapes UDA benchmark. Column *Aux.* indicates the usage of any auxiliary task. Please note that the distillation stage of ProDA [6] is removed for a fair comparison. All the numbers are presented in percentage (%).

SYNTHIA → Cityscapes																		
Method	Aux.	Road	SideW	Build	Wall	Fence	Pole	Light	Sign	Veg	Sky	Person	Rider	Car	Bus	Motor	Bike	mIoU
Source only		51.8	17.0	73.0	7.1	0.2	25.4	9.4	10.2	70.7	84.0	55.6	13.7	68.0	2.9	8.5	16.1	32.1
CBST [9]		68.0	29.9	76.3	10.8	1.4	33.9	22.8	29.5	77.6	78.3	60.6	28.3	81.6	23.5	18.8	39.8	42.6
CAG-UDA [5]		84.7	40.8	81.7	7.8	0.0	35.1	13.3	22.7	84.5	77.6	64.2	27.8	80.9	19.7	22.7	48.3	44.5
Uncertainty [29]		87.6	41.9	83.1	14.7	1.7	36.2	31.3	19.9	81.6	80.6	63.0	21.8	86.2	40.7	23.6	53.1	47.9
IAST [11]		81.9	41.5	83.3	17.7	4.6	32.3	30.9	28.8	83.4	85.0	65.5	30.8	86.5	38.2	**33.1**	52.7	49.8
DACS [12]		80.6	25.1	81.9	21.5	2.9	37.2	22.7	24.0	83.7	90.8	67.6	38.3	82.9	38.9	28.5	47.6	48.3
SPIGAN [14]	✓	71.1	29.8	71.4	3.7	0.3	33.2	6.4	15.6	81.2	78.9	52.7	13.1	75.9	25.5	10.0	20.5	36.8
GIO-Ada [15]	✓	78.3	29.2	76.9	11.4	0.3	26.5	10.8	17.2	81.7	81.9	45.8	15.4	68.0	15.9	7.5	30.4	37.3
DADA [13]	✓	89.2	44.8	81.4	6.8	0.3	26.2	8.6	11.1	81.8	84.0	54.7	19.3	79.7	40.7	14.0	38.8	42.6
GUDA [30]	✓	88.1	53.0	84.0	22.0	1.4	**39.6**	28.2	24.8	82.7	81.5	65.5	22.7	**89.3**	**50.5**	25.1	**57.5**	51.0
CorDA [17]	✓	**93.3**	**61.6**	85.3	19.6	**5.1**	37.8	36.6	42.8	84.9	90.4	69.7	41.8	85.6	38.4	32.6	53.9	55.0
ELDA (Ours)	✓	92.6	56.6	**85.5**	**24.2**	2.1	37.6	**38.1**	**43.1**	**85.7**	**91.5**	**69.8**	**42.0**	87.2	47.6	20.0	50.1	**55.2**

Table 2. The quantitative results on the SYNTHIA→Cityscapes UDA benchmark. All the numbers are presented in percentage (%).

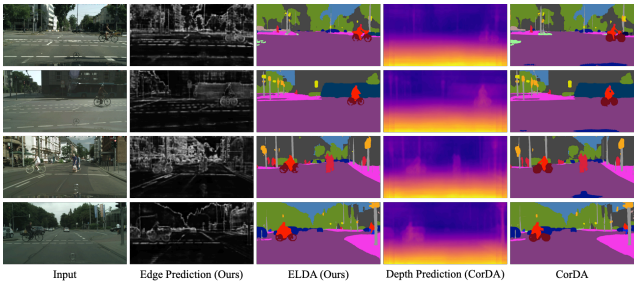

Figure 4. Examples of the edge predictions from ELDA, the depth predictions from CorDA, and the semantic segmentation results.

Method	Edge Aux.	Correlation Module	mIoU (%)
DACS [12]			52.1
ELDA (SDI-Enc+TSB)	✔		55.5
ELDA (SDI-Enc+TSB+CM)	✔	✔	**57.3**

Table 3. Validation of the effectiveness for each component.

compare the performances of (1) ELDA, (2) ELDA without CM (denoted as ELDA (SDI-Enc+TSB)), and (3) ELDA without SDI-Enc, TSB, and CM (which is essentially the DACS baseline [12]). The experimental results presented in Table 3 reveal that the performance grows with the addition of each component, suggesting that each components in ELDA indeed provides positive performance impacts.

3.4. Qualitative Results

Fig. 3 presents the segmentation predictions from *source only*, CorDA [17], and ELDA on a number of images selected from the GTA5→Cityscapes benchmark. It is observed that the predictions from ELDA are less fragmented and have clearer boundaries as compared to those of *source only* and CorDA. Fig. 4 further demonstrates how ELDA is able to deliver impressive details in its predictions through approximating the training target of edges generated by $C(\cdot)$. The incorporation of high quality edge information allows ELDA to even capture small and subtle features of the input images, such as the silhouettes of the spokes and axles at the center of bike wheels. In contrast, the predictions from CorDA fails to capture those details, which is mainly due to the use of the relatively inaccurate predictions from its SSL based depth model, as discussed in Section 1.

4. Conclusions

In this work, we proposed a novel UDA framework, called ELDA, to utilize the highly available and high quality edge information by incorporating edge extraction into the training process of ELDA as an auxiliary task. To validate the performance of ELDA, we evaluated it against a number of baselines on two commonly-adopted benchmarks, and quantitatively and qualitatively demonstrated that ELDA is able to achieve the state-of-the-art performance as compared to the contemporary UDA methods. As ELDA is able to leverage low-cost domain invariant edge information to enhance its adaptation performance, it thus offers a new avenue for future semantic segmentation based UDA models.

L3D-IVU
#49

References

[1] Tuan-Hung Vu, Himalaya Jain, Maxime Bucher, Matthieu Cord, and Patrick Pérez. Advent: Adversarial entropy minimization for domain adaptation in semantic segmentation, 2019. 1

[2] Yi-Hsuan Tsai, Kihyuk Sohn, Samuel Schulter, and Manmohan Chandraker. Domain adaptation for structured output via discriminative patch representations, 2019. 1

[3] Minghao Xu, Jian Zhang, Bingbing Ni, Teng Li, Chengjie Wang, Qi Tian, and Wenjun Zhang. Adversarial domain adaptation with domain mixup, 2019. 1

[4] Yuan Wu, Diana Inkpen, and Ahmed El-Roby. Dual mixup regularized learning for adversarial domain adaptation, 2020. 1

[5] Qiming Zhang, Jing Zhang, Wei Liu, and Dacheng Tao. Category anchor-guided unsupervised domain adaptation for semantic segmentation, 2019. 1, 3, 4

[6] Pan Zhang, Bo Zhang, Ting Zhang, Dong Chen, Yong Wang, and Fang Wen. Prototypical pseudo label denoising and target structure learning for domain adaptive semantic segmentation, 2021. 1, 3, 4

[7] Haoyu Ma, Xiangru Lin, Zifeng Wu, and Yizhou Yu. Coarse-to-fine domain adaptive semantic segmentation with photometric alignment and category-center regularization, 2021. 1

[8] Munan Ning, Donghuan Lu, Dong Wei, Cheng Bian, Chenglang Yuan, Shuang Yu, Kai Ma, and Yefeng Zheng. Multi-anchor active domain adaptation for semantic segmentation, 2021. 1

[9] Yang Zou, Zhiding Yu, B. V. K. Vijaya Kumar, and Jinsong Wang. Domain adaptation for semantic segmentation via class-balanced self-training, 2018. 1, 3, 4

[10] Yang Zou, Zhiding Yu, Xiaofeng Liu, B. V. K. Vijaya Kumar, and Jinsong Wang. Confidence regularized self-training, 2020. 1

[11] Ke Mei, Chuang Zhu, Jiaqi Zou, and Shanghang Zhang. Instance adaptive self-training for unsupervised domain adaptation, 2020. 1, 3, 4

[12] Wilhelm Tranheden, Viktor Olsson, Juliano Pinto, and Lennart Svensson. Dacs: Domain adaptation via cross-domain mixed sampling, 2020. 1, 3, 4

[13] Tuan-Hung Vu, Himalaya Jain, Maxime Bucher, Matthieu Cord, and Patrick Pérez. Dada: Depth-aware domain adaptation in semantic segmentation, 2019. 1, 3, 4

[14] Kuan-Hui Lee, Germán Ros, Jie Li, and Adrien Gaidon. Spigan: Privileged adversarial learning from simulation. *ArXiv*, abs/1810.03756, 2019. 1, 3, 4

[15] Yuhua Chen, Wen Li, Xiaoran Chen, and Luc Van Gool. Learning semantic segmentation from synthetic data: A geometrically guided input-output adaptation approach. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 1, 3, 4

[16] Suman Saha, Anton Obukhov, Danda Pani Paudel, Menelaos Kanakis, Yuhua Chen, Stamatios Georgoulis, and Luc Van Gool. Learning to relate depth and semantics for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8197–8207, June 2021. 1

[17] Qin Wang, Dengxin Dai, Lukas Hoyer, Luc Van Gool, and Olga Fink. Domain adaptive semantic segmentation with self-supervised depth estimation, 2021. 1, 2, 3, 4

[18] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel Brostow. Digging into self-supervised monocular depth estimation, 2019. 1

[19] Stephan R. Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for data: Ground truth from computer games, 2016. 1, 2, 3

[20] N. Kanopoulos, N. Vasanthavada, and R.L. Baker. Design of an image edge detection filter using the sobel operator. *IEEE Journal of Solid-State Circuits*, 23(2):358–367, 1988. 1

[21] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network, 2014. 1

[22] 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, 2016. 2, 3

[23] 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 *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 2, 3

[24] Stamatios Georgoulis Simon Vandenhende and Luc Van Gool. Mti-net: Multi-scale task interaction networks for multi-task learning. In *European Conference on Computer Vision (ECCV)*, page 527–543, 2020. 2

[25] Xiaogang Wang Dan Xu, Wanli Ouyang and Nicu Sebe. Pad-net: Multi-tasks guided prediction-and-distillation net- work for simultaneous depth estimation and scene parsing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[26] John Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(6):679–698, 1986. 2

[27] Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. V-net: Fully convolutional neural networks for volumetric medical image segmentation, 2016. 2

[28] Zhilu Zhang and Mert R. Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. *CoRR*, abs/1805.07836, 2018. 3

[29] Zhedong Zheng and Yi Yang. Rectifying pseudo label learning via uncertainty estimation for domain adaptive semantic segmentation, 2020. 3, 4

[30] Vitor Guizilini, Jie Li, Rares Ambrus, and Adrien Gaidon. Geometric unsupervised domain adaptation for semantic segmentation, 2021. 3, 4

[31] Chen-Hao Chao, Bo-Wun Cheng, and Chun-Yi Lee. Rethinking ensemble-distillation for semantic segmentation based unsupervised domain adaption. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 2610–2620, June 2021. 3

[32] Kai Zhang, Yifan Sun, Rui Wang, Haichang Li, and Xiaohui Hu. Multiple fusion adaptation: A strong framework for unsupervised semantic segmentation adaptation. *CoRR*, abs/2112.00295, 2021. 3

[33] Lukas Hoyer, Dengxin Dai, and Luc Van Gool. Daformer: Improving network architectures and training strategies for domain-adaptive semantic segmentation, 2021. 3