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

Ting-Hsuan Liao*[1]
tingforun@gapp.nthu.edu.tw
Huang-Ru Liao*[1]
mimiliao2000@gapp.nthu.edu.tw
Shan-Ya Yang†[1]
sonya.tw2000@gapp.nthu.edu.tw
Jie-En Yao†[1]
matt1129yao@gapp.nthu.edu.tw
Li-Yuan Tsao†[1]
abcdabcd10211@gapp.nthu.edu.tw
Hsu-Shen Liu†[1]
tenhoshi9023@gapp.nthu.edu.tw
Bo-Wun Cheng‡[1]
bobcheng15@gapp.nthu.edu.tw
Chen-Hao Chao‡[1]
lance_chao@gapp.nthu.edu.tw
Chia-Che Chang‡[2]
chia-che.chang@mediatek.com
Yi-Chen Lo‡[2]
yichen.lo@mediatek.com
Chun-Yi Lee[1]
cylee@gapp.nthu.edu.tw

[1] Elsa Lab,
Department of Computer Science,
National Tsing Hua University,
No. 101, Section 2, Kuang-Fu Road,
Hsinchu City 300,
Taiwan

[2] MediaTek, Inc.,
No. 1, Dusing 1st Road,
Hsinchu Science Park,
Hsinchu City 300,
Taiwan

*, †, and ‡ indicate equal contributions.

Abstract

Many unsupervised domain adaptation (UDA) methods have been proposed to bridge the domain gap by utilizing domain invariant information. Most approaches have chosen depth as such information and achieved remarkable successes. Despite their effectiveness, using depth as domain invariant information in UDA tasks may lead to multiple issues, such as excessively high extraction costs and difficulties in achieving a reliable prediction quality. As a result, we introduce Edge Learning based Domain Adaptation (ELDA), a 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, and enables ELDA to outperform the contemporary state-of-the-art methods on two commonly adopted benchmarks for semantic segmentation based UDA tasks. In addition, we show that ELDA is able to better separate the feature distributions of different classes. We further provide ablation analysis to justify our design decisions.

© 2022. The copyright of this document resides with its authors.
It may be distributed unchanged freely in print or electronic forms.

1 Introduction

Supervised learning for semantic segmentation has achieved unprecedented successes in the past few years. Albeit effective, training a semantic segmentation model in a supervised manner requires pixel-level labeling, which is often prohibitively expensive and time-consuming. Being aware of these undesirable drawbacks, recent studies have been attempting to make use of the existing labeled datasets or simulated environments to train semantic segmentation models, and then adapt the models to certain label-less targeted domains. This category of research direction is often referred to as semantic segmentation based UDA in the computer vision community. A number of approaches have been explored to tackle this challenge, including adversarial training [28, 30, 33, 34], anchoring [19, 22, 37, 38], and pseudo labeling (PL) [20, 27, 41, 42], 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 limits their performance and thus leaves 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 confronted by them. Domain invariant information possess 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 to help shrink the domain gap [6, 16, 25, 31]. The authors in [32] further utilized self-supervised learning (SSL) to retrieve depth information, with an 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 associated with training an accurate auxiliary SSL-based depth estimation model is often expensive. A few researchers [32] 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 such methods 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 (e.g., lidar, stereo camera, etc.) or supervised models [10] in terms of accuracy. In other words, their predictions might deviate from the ground truths, and hence might bring negative impacts on the training process of the semantic segmentation based UDA methods.

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 costs of extracting edges from an input image is substantially lower than those of extracting a depth map from the same image using SSL. Specifically, edges in an image can be obtained by performing convolution using certain fixed kernels over an input image in one pass [15], while the extraction of a depth map usually requires training and inferencing of a sophisticated depth model [10]. Second, the quality of edges is typically much more consistent than that of depth maps, as depth estimation using only RGB images is an ill-posed problem [9], 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

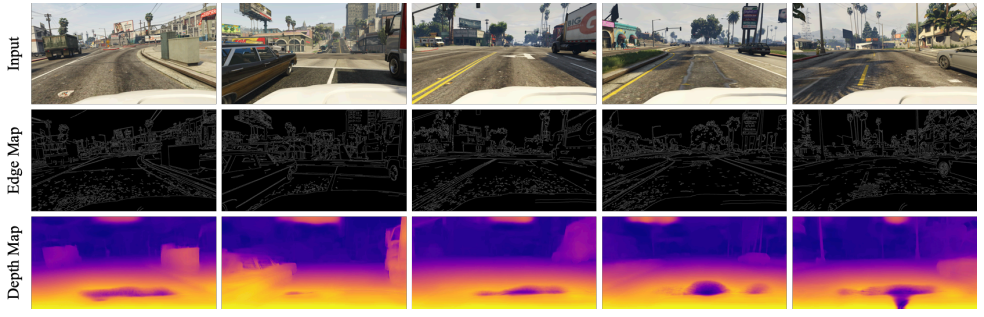

Figure 1: An example that depicts the differences between edges and depth maps extracted by [11] on the selected images from the GTA5 dataset [23] without the use of any ground truth labels. For the depth maps, nearer surfaces are brighter, while farther ones are darker.

are better captured in the extracted edge maps. In contrast, the depth maps are noisy and the object boundaries are relatively blurred. 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 motivation, 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 its training process as an auxiliary task. The experimental results show that without resorting to ensemble distillation methods [2, 36, 37] or transformer based architectures [13], ELDA is able to achieve the state-of-the-art performance on two commonly adopted benchmarks [2, 23, 24]. The contributions of this work are summarized as follows:

- We introduce the use of edge information as an auxiliary task for semantic segmentation based UDA, and develop an effective framework named ELDA, to take full advantage of the valuable edge information embedded in the images of both domains.
- We validate ELDA on two commonly adopted benchmarks quantitatively and qualitatively, and show that it is able to achieve superior performance to the previous methods.
- We demonstrate that by incorporating edge information into semantic segmentation based UDA, ELDA can capture fine-grained features in an unlabeled target domain.

2 Related Work

2.1 Unsupervised Domain Adaptation for Semantic Segmentation

A number of UDA methods have been proposed in an attempt to bridge the discrepancies between different domains. One category of these works adopted adversarial training process to learn representations of their target domains [28, 30, 31, 33, 34]. These frameworks often consist of a generator and a discriminator trained against each other in order to minimize the domain gap, and have shown significant improvements over those trained in the source domains only without the use of any adaptation technique. Another line of work has focused on self-training and data augmentation measures to tackle UDA problems. For those utilizing self-training, the concentration was mainly on preventing overfitting by using regularization [40, 42] or class-balancing [20, 41] when minimizing uncertainty in their

target domains. The authors in [27] extended the concept of self-training and proposed a data augmentation technique. It fine-tunes a model with mixed labels generated by combining ground truth annotations from a source domain and pseudo labels from a target domain. Recent researchers further employed ensemble learning to deal with the above challenge [2].

2.2 Auxiliary Tasks in Semantic Segmentation based UDA Methods

As mentioned in Section 1, a few recent researchers have turned their attention to incorporating domain invariant features into the training processes of their UDA methods. To achieve this objective, a commonly adopted method is to introduce them through auxiliary tasks. Among these tasks, depth estimation is the most widely used one. The main incentive behind this is that geometric and semantic information are highly correlated [6, 12, 16, 25, 31, 32]. This concept is first introduced in SPIGAN [16], which uses synthetic semantic segmentation and depth information as additional means of regularization for their style transfer model. The authors in DADA [31] proposed to shrink the domain gap by fusing segmentation and depth maps together during the adversarial training process. Moreover, CorDA [32] explicitly exploits a segmentation path and a depth estimation path in their UDA framework, where the latter path serves as an auxiliary task. These two paths are interleaved, in which the embeddings from them are fused together using attention layers such that two paths can mutually benefit from each other. In addition, CorDA leverages the prediction discrepancies from two depth decoders to assist in pseudo-label refinement for the segmentation path. Furthermore, GUDA [12] leverages more auxiliary paths, including depth estimation, surface normal, as well as self-supervised photometric loss, in their UDA model. However, the aforementioned methods require additional depth estimation models trained based on certain self-supervised learning approaches and may suffer from the issues mentioned in Section 1.

2.3 Usage of Edge Information in Other Computer Vision Domains

Edge detection has been utilized in a wide variety of computer vision research domains such as semantic segmentation [3, 14], object detection[10, 17], facial recognition[5], and representation learning [4]. Edges are low-level representations extracted from images that are able to reflect important information about the discontinuities in depth, surface orientation, material properties, as well as scene illumination in images. Besides its abundance in semantic information, edges from an image can be easily extracted using conventional computer vision approaches such as Sobel [15] and Canny edge extraction algorithms [1]. These algorithms are efficient and straightforward and do not require the use of any learnable parameter. Therefore they are considered readily available from almost all image domains, and meet the characteristics of domain invariant features described in Section 1. Despite these advantages, the use of edge information has not been explored in semantic segmentation based UDA tasks. In this work, we show that edge information can be used as a type of domain invariant feature and can indeed help boost the performance of the proposed ELDA model.

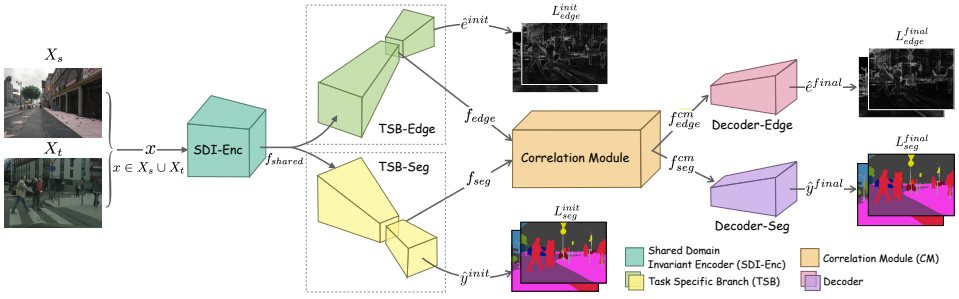

Figure 2: An illustration of the proposed ELDA framework.

3 Methodology

3.1 Problem Formulation

In UDA tasks, a model has access to a source dataset $X_s = \{x_s^1, ..., x_s^N\}$, their corresponding labels $Y_s = \{y_s^1, ..., y_s^N\}$, 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 target domains, respectively. Specifically, a tuple (x_s, y_s) represents an image-label pair from the source domain, and x_t represents a target domain image. The training objective is to train a model such that its predictions can best estimate the ground truth labels in the target domain. In other words, the mean intersection-over-union (mIoU) of the predictions from the model should be maximized. For the detailed notation used in our work, please refer to the notation table in the supplementary material.

3.2 Overview of the Proposed ELDA Framework

Fig. 2 illustrates an overview of the proposed ELDA framework. First, an input image from either the source domain (i.e., $x_s \in X_s$) or the target domain (i.e., $x_t \in X_t$) is first fed into a shared domain invariant encoder (SDI-Enc) to obtain the shared latent feature embedding (f_{shared}). Subsequently, f_{shared} is passed through two separate task specific branches (TSBs), to generate the task specific features, i.e., the edge (f_{edge}) and the semantic segmentation (f_{seg}) features, and the initial predictions of edges ($\hat{e}_s^{\text{init}}$ or $\hat{e}_t^{\text{init}}$) as well as semantic segmentation ($\hat{y}_s^{\text{init}}$ or $\hat{y}_t^{\text{init}}$). Both f_{edge} and f_{seg} are then propagated through a correlation module (CM) to exchange information between f_{edge} and f_{seg}. Then, the outputs of CM are forwarded to two distinct decoders to generate the final output predictions of edges ($\hat{e}_s^{\text{final}}$ or $\hat{e}_t^{\text{final}}$) and semantic segmentation ($\hat{y}_s^{\text{final}}$ or $\hat{y}_t^{\text{final}}$), respectively, where subscripts s and t denote the source and the target domains. The edge detection loss L_{edge} and the segmentation loss L_{seg} are computed to update the model's weights. In the following subsections, we explain the components of ELDA. In Section 3.3 we elaborate on the details of SDI-Enc, TSBs, and CM. In Section 3.4, we describe the formulations of the loss functions L_{edge} and L_{seg}.

3.3 Architecture Components of the ELDA Framework

3.3.1 Shared Domain Invariant Encoder (SDI-Enc)

In auxiliary task learning, shared encoders are usually adopted to extract common features so as to enhance performance and reduce inference cost [26]. ELDA employs the shared

encoder structure in [32] to extract f_{shared} for capturing both edge and segmentation features.

3.3.2 Task Specific Branch (TSB)

To enable f_{shared} to be further interpreted into specific feature embeddings that bear edge and semantic segmentation meanings, two separate branches of TSBs, similar to those used in [26, 32], 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. Meanwhile, the decoders are employed to decode f_{edge} and f_{seg} in to $\hat{e}_s^{\text{init}}$ or $\hat{e}_t^{\text{init}}$ and $\hat{y}_s^{\text{init}}$ or $\hat{y}_t^{\text{init}}$, respectively, depending on the original domain of the input image, for updating SDI-Enc and the TSBs. Please note that $\hat{e}$ represents the edge predictions, $\hat{y}$ denotes the semantic segmentation predictions, and the subscripts s and t stand for the source and the target domains.

3.3.3 Correlation Module (CM)

With an aim to communicate information between f_{edge} and f_{seg}, we employ a CM [8, 32] into ELDA. Specifically, within CM, the information in f_{edge} and f_{seg} are first filtered with sigmoid functions to re-weight the task specific intermediate embeddings $f_{\text{seg}}^{\text{mid}}$ and $f_{\text{edge}}^{\text{mid}}$ as:

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

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

$$f_{\text{edge}}^{\text{cm}} = f_{\text{edge}} + f_{\text{seg}}^{\text{mid}} * \text{Sigmoid}(\text{Conv}(f_{\text{seg}})), \tag{3}$$

where $\text{Conv}(\cdot)$ and $\text{Sigmoid}(\cdot)$ denote the convolution and sigmoid functions, and $f_{\text{seg}}^{\text{cm}}$ and $f_{\text{edge}}^{\text{cm}}$ are the outputs of CM. CM assists in preserving the essential features from both TSBs.

3.4 Loss Function Design

3.4.1 The Loss Function for Edge Estimation (L_{edge})

In ELDA, the supervision targets for edges in both the source and the target domains are generated using the Canny edge extraction algorithm [1] denoted as $\text{C}(\cdot;\sigma)$, where σ is the parameter for controlling the smoothness of an edge map. 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 $\text{C}(\cdot;\sigma)$. Both $L_{\text{edge}}^{\text{init}}$ and $L_{\text{edge}}^{\text{final}}$ are derived by extending the DICE loss [21], as it is able to prevent imbalance between different classes. The expressions of $L_{\text{edge}}^{\text{init}}$ and $L_{\text{edge}}^{\text{final}}$ are formulated as:

$$L_{\text{edge}}^{\text{init}} = (1 - \text{D}(\text{C}(x_s;\sigma), \hat{e}_s^{\text{init}})) + (1 - \text{D}(\text{C}(x_t;\sigma), \hat{e}_t^{\text{init}})), \tag{4}$$

$$L_{\text{edge}}^{\text{final}} = (1 - \text{D}(\text{C}(x_s;\sigma), \hat{e}_s^{\text{final}})) + (1 - \text{D}(\text{C}(x_t;\sigma), \hat{e}_t^{\text{final}})), \tag{5}$$

$$\text{D}(e,\hat{e}) = \frac{2\sum_{i=1}^{P} e\hat{e}}{\sum_{i=1}^{P} e^2 + \sum_{i=1}^{P} \hat{e}^2}, \;\; 0 \leq \text{D} \leq 1, \tag{6}$$

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

3.4.2 The Loss Function for Semantic Segmentation (L_{seg})

In ELDA, the training targets for semantic segmentation in the source domain are the ground truth labels, while those in the target domain are the pseudo labels generated using ELDA's predictions. The loss for semantic segmentation $L_{\text{seg}} = L_{\text{seg}}^{\text{init}} + L_{\text{seg}}^{\text{final}}$ is then computed between the predictions of ELDA and the training targets by utilizing the cross-entropy (CE) operator $\mathrm{CE}(\cdot)$ [39]. The expressions of the loss components $L_{\text{seg}}^{\text{init}}$ and $L_{\text{seg}}^{\text{final}}$ are formulated as follows:

$$L_{\text{seg}}^{\text{final}} = \mathrm{CE}(y_s, \hat{y}_s^{\text{final}}) + \mathrm{CE}(y_t^{'}, \hat{y}_t^{\text{final}}), \;\; L_{\text{seg}}^{\text{init}} = \mathrm{CE}(y_s, \hat{y}_s^{\text{init}}) + \mathrm{CE}(y_t^{'}, \hat{y}_t^{\text{init}}), \tag{7}$$

where $\mathrm{CE}(y,\hat{y}) = -\sum_{i=1}^{P} y \log \hat{y}$, $y_t^{'}$ represents the pseudo-labels in the target domain, and y denotes the segmentation labels, which can be either y_s or $y_t^{'}$. Finally, $\hat{y}$ denotes the predicted segmentation maps. Please note that $\hat{y}$ can be any of $\hat{y}_s^{\text{init}}$, $\hat{y}_t^{\text{init}}$, $\hat{y}_s^{\text{final}}$, or $\hat{y}_t^{\text{final}}$.

3.4.3 The Total Loss (L_{total})

Based on the above derivations, the total loss can be formulated as $L_{\text{total}} = L_{\text{seg}} + \lambda L_{\text{edge}}$, where λ is a balancing factor whose value is provided in the supplementary material.

4 Experimental Results

4.1 Experimental Setup

We evaluate and compare the experimental results of ELDA against the pure semantic segmentation based UDA methods [18, 20, 27, 35, 37, 38, 40, 41], as well as the methods that take advantage of additional information in the form of auxiliary tasks [6, 12, 16, 25, 31, 32]. The goal of the comparisons against [27], which is a pure semantic segmentation based UDA method that bears a similar model architecture as ELDA, is to demonstrate that the additional edge information incorporated into the training process of ELDA is indeed beneficial. In addition, the goal of the comparisons against [32], which is one of the methods that utilize auxiliary tasks and is considered the state-of-the-art, is to show that edges work as well as, or even better than depth, when it comes to designing auxiliary tasks. We evaluate ELDA and the baselines on two commonly adopted benchmarks: GTA5 [23]→ Cityscapes [7] and SYNTHIA [24]→ Cityscapes. The detailed setups are offered in the supplementary material.

4.2 Quantitative Results on the Benchmarks

Tables 1 and 2 compare the results of ELDA against multiple baselines. Please note that these baselines do not include works that resort to ensemble distillation methods [2, 36, 37] or transformer based architectures [13] for fair comparisons. We also include the performance of ELDA trained solely in the source domains, denoted as *source only*, for reference. Table 1 reports the results evaluated on the GTA5→Cityscapes benchmark. By leveraging edge prediction as an auxiliary task, ELDA achieves an mIoU of 57.3%, outperforming *source only* by a margin of 20% mIOU. This demonstrates that the addition of edge information brings

GTA5 → Cityscapes																				
Method	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
Pure semantic segmentation based UDA methods																				
CBST[ECCV18]	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[CVPR19]	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
FDA[NeurIPS20]	92.5	53.3	82.4	26.5	27.6	36.4	40.6	38.9	82.3	39.8	78.0	62.6	34.4	84.9	34.1	53.1	16.9	27.7	46.4	50.45
PIT[ICCV21]	87.5	43.4	78.8	31.2	30.2	36.3	39.9	42.0	79.2	37.1	79.3	65.4	**37.5**	83.2	46.0	45.6	25.7	23.5	49.9	50.6
Uncertainty[ICCV21]	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[ICCV21]	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[ICRA21]	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*[CVPR21]	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
Semantic segmentation based UDA methods using auxiliary tasks																				
CorDA[CVPR21]	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 evaluated on the GTA5→Cityscapes UDA benchmark. Please note that the distillation stage of ProDA [37] is removed for a fair comparison.

SYNTHIA → Cityscapes																	
Method	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
Pure semantic segmentation based UDA methods																	
CBST[ECCV18]	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[CVPR19]	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
PIT[ICCV21]	83.1	27.6	81.5	8.9	0.3	21.8	26.4	33.8	76.4	78.8	64.2	27.6	79.6	31.2	31.0	31.3	44.0
Uncertainty[ICCV21]	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[ICCV21]	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[ICRA21]	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
Semantic segmentation based UDA methods using auxiliary tasks																	
SPIGAN[ICLR19]	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[IJCAI19]	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[ICCV19]	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[AAAI20]	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[CVPR21]	**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
CTRL[ICCV21]	86.4	42.5	80.4	20.0	1.0	27.7	10.5	13.3	80.6	82.6	61.0	23.7	81.8	42.9	21.0	44.7	45.0
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 evaluated on the SYNTHIA→Cityscapes UDA benchmark.

positive influences. Furthermore, ELDA indeed has an edge over DACS [27] and the current state-of-the-art CorDA [32] by 5.2% mIoU and 0.7% mIoU, respectively. Table 2 shows the results evaluated on the SYNTHIA→Cityscapes benchmark. The results display a similar trend as those on the GTA5→Cityscapes benchmark, in which ELDA is able to reach the state-of-the-art performance of 55.2% mIOU, outperforming 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 baselines [6, 12, 16, 25, 31, 32] in their auxiliary tasks. In contrast, ELDA is able to achieve superior performance without the use of any extra labeled data.

4.3 Qualitative Results on the GTA5→Cityscapes benchmark

Fig. 3 presents the predictions from *source only*, CorDA [32], and ELDA on some images selected from the GTA5→Cityscapes benchmark. It is observed that predictions from ELDA are of better quality as they are less fragmented and have clearer boundaries as compared to those predicted by *source only* and CorDA. For example, ELDA is able to clearly mark the boundaries between sidewalks and roads, while CorDA and *source only* tend to produce uncertain and sub-optimal boundaries. Fig. 4 further shows how ELDA is able to produce impressive details in its predictions through approximating the training target of edges generated by $C(\cdot;\sigma)$. The incorporation of high quality edge information allows ELDA to even capture small and subtle features of the image, like the hollowness at the center of bike wheels. In contrast, the predictions from CorDA fail to capture a similar degree of details, which is due to the use of relatively ambiguous depth information during its training process.

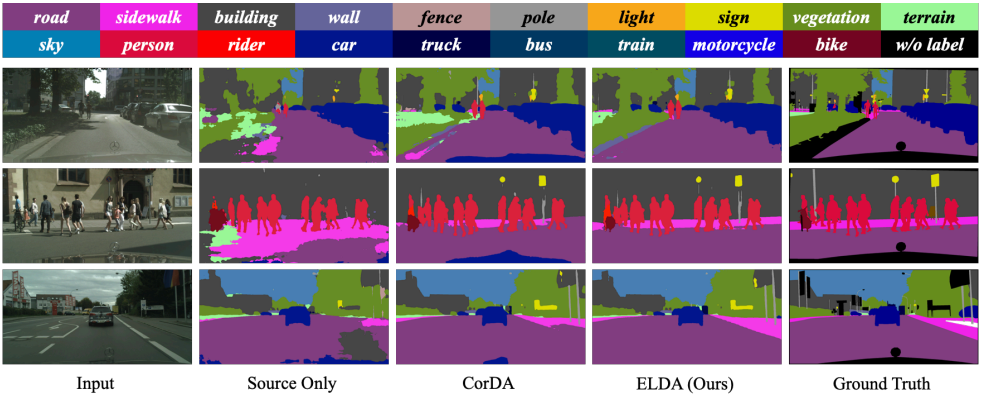

Figure 3: A comparison of the semantic segmentation predictions from different methods.

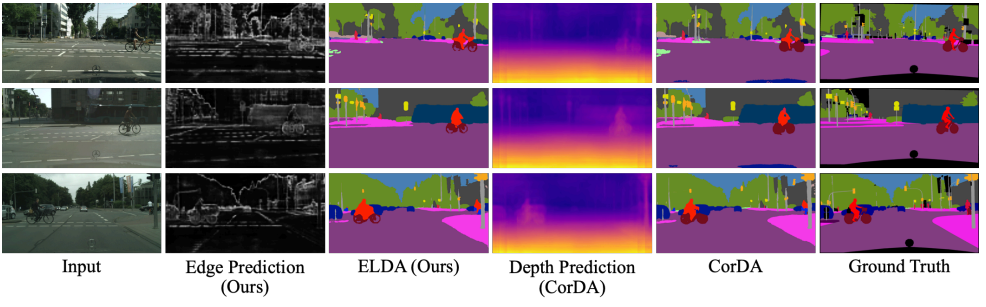

Figure 4: A comparison of the predictions evaluated by ELDA and CorDA.

4.4 Ablation Study

As described in Section 3, ELDA is made up of multiple key components such as SDI-Enc, TSB, and CM. To examine how each of them contributes to the overall performance, we compare the performances between (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 [27]). The experimental results presented in Table 3 show that with the addition of each component, the performance grows, indicating that the adoption of these components in ELDA indeed has positive performance impacts and thus validates our design choices.

4.5 Visualization of the Features in the Target Domain

To verify that ELDA is able to learn a more discriminative and thus descriptive feature space, we visualize the target domain features of *source only*, DACS, CorDA, and ELDA in Fig. 5 using t-SNE [29]. Please note that for ease of viewing and observation, we only display the target domain features of several semantic classes, including sidewalk, fence, car, truck, and bicycle. It is observed that the target domain features captured by *source only* does not have clear boundaries between different classes, reflecting its sub-optimal performance in the target domain. In contrast, the features captured by ELDA, DACS [27], and CorDA [32] display clearer boundaries, translating to their superior performance in the target domain.

Method	Edge Aux.	Correlation Module	mIoU
DACS [27]			52.1
ELDA (SDI-Enc+TSB)	✔		55.5
ELDA (SDI-Enc+TSB+CM)	✔	✔	**57.3**

Table 3: An ablation analysis for validating the effectiveness of each component.

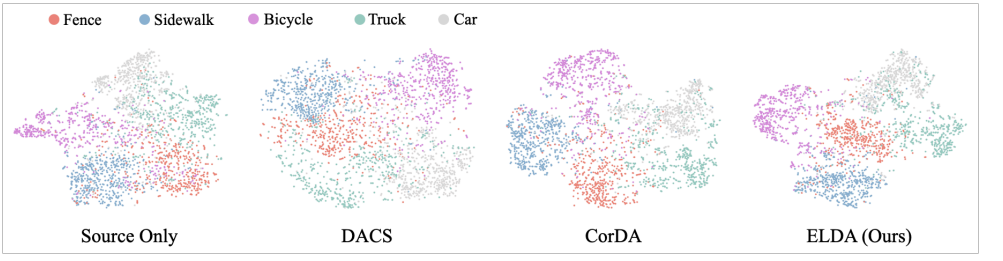

Figure 5: A visualization of the feature space using t-SNE for four semantic classes.

5 Conclusion

In this work, we proposed an effective framework, called ELDA, for performing semantic segmentation based UDA. ELDA utilizes the highly available and high quality edge information as the domain invariant information by incorporating edge extraction into its training process 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 showed that ELDA is able to achieve the state-of-the-art performance as compared to the baselines. In addition, we presented an ablation study and feature analysis in the target domain to validate our design choices. As the proposed framework is able to leverage precious edge information to enhance its adaptation performance, it thus offers a different direction to further enhance the performance of semantic segmentation based UDA models.

6 Acknowledgments

This work was supported by the National Science and Technology Council (NSTC) in Taiwan under grant numbers MOST 111-2223-E-007-004-MY3 and MOST 111-2628-E-007-010. The authors acknowledge the financial support from MediaTek Inc., Hsichu City, Taiwan. The authors would also like to acknowledge the donation of the GPUs from NVIDIA Corporation and NVIDIA AI Technology Center (NVAITC) used in this research work. The authors thank National Center for High-Performance Computing (NCHC) for providing computational and storage resources. Finally, the authors would also like to thank the time and effort of the anonymous reviewers for reviewing this paper.

References

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

[2] 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] Liang-Chieh Chen, Jonathan T. Barron, George Papandreou, Kevin Murphy, and Alan L. Yuille. Semantic image segmentation with task-specific edge detection using cnns and a discriminatively trained domain transform, 2016.

[4] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations, 2020.

[5] Xiaoming Chen and Wushan Cheng. Facial expression recognition based on edge detection. *International Journal of Computer Science Engineering Survey*, 6:1–9, 04 2015.

[6] 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.

[7] 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.

[8] 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.

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

[10] Vittorio Ferrari, Loic Fevrier, Frederic Jurie, and Cordelia Schmid. Groups of adjacent contour segments for object detection. *IEEE transactions on pattern analysis and machine intelligence*, 30(1):36–51, 2007.

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

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

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

[14] Zeyu Hu, Mingmin Zhen, Xuyang Bai, Hongbo Fu, and Chiew-lan Tai. Jsenet: Joint semantic segmentation and edge detection network for 3d point clouds. *Lecture Notes in Computer Science*, page 222–239, 2020. ISSN 1611-3349.

[15] 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.

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

[17] Joseph J Lim, C Lawrence Zitnick, and Piotr Dollár. Sketch tokens: A learned mid-level representation for contour and object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3158–3165, 2013.

[18] Fengmao Lv, Tao Liang, Xiang Chen, and Guosheng Lin. Cross-domain semantic segmentation via domain-invariant interactive relation transfer. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4333–4342, 2020.

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

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

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

[22] 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.

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

[24] 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.

[25] 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.

[26] 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.

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

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

[29] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008.

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

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

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

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

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

[35] Yanchao Yang and Stefano Soatto. Fda: Fourier domain adaptation for semantic segmentation. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4084–4094, 2020.

[36] 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.

[37] 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.

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

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

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

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

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