


## Article

# PANDA: A Polarized Attention Network for Enhanced Unsupervised Domain Adaptation in Semantic Segmentation

Chiao-Wen Kao <sup>1</sup>, Wei-Ling Chang <sup>2,\*</sup>, Chun-Chieh Lee <sup>2</sup>  and Kuo-Chin Fan <sup>2</sup>

<sup>1</sup> Department of Applied Artificial Intelligence, Ming Chuan University, Taoyuan 320, Taiwan; chiaowen@mail.mcu.edu.tw

<sup>2</sup> Department of Computer Science and Information Engineering, National Central University, Taoyuan 320, Taiwan; jackclee@cc.ncu.edu.tw (C.-C.L.); kcfan@csie.ncu.edu.tw (K.-C.F.)

\* Correspondence: w410148@gmail.com

**Abstract:** Unsupervised domain adaptation (UDA) focuses on transferring knowledge from the labeled source domain to the unlabeled target domain, reducing the costs of manual data labeling. The main challenge in UDA is bridging the substantial feature distribution gap between the source and target domains. To address this, we propose Polarized Attention Network Domain Adaptation (PANDA), a novel approach that leverages Polarized Self-Attention (PSA) to capture the intricate relationships between the source and target domains, effectively mitigating domain discrepancies. PANDA integrates both channel and spatial information, allowing it to capture detailed features and overall structures simultaneously. Our proposed method significantly outperforms current state-of-the-art unsupervised domain adaptation (UDA) techniques for semantic segmentation tasks. Specifically, it achieves a notable improvement in mean intersection over union (mIoU), with a 0.2% increase for the GTA→Cityscapes benchmark and a substantial 1.4% gain for the SYNTHIA→Cityscapes benchmark. As a result, our method attains mIoU scores of 76.1% and 68.7%, respectively, which reflect meaningful advancements in model accuracy and domain adaptation performance.

**Keywords:** unsupervised domain adaptation; semantic segmentation; attention mechanism; feature extraction; self-training



**Citation:** Kao, C.-W.; Chang, W.-L.; Lee, C.-C.; Fan, K.-C. PANDA: A Polarized Attention Network for Enhanced Unsupervised Domain Adaptation in Semantic Segmentation. *Electronics* **2024**, *13*, 4302. <https://doi.org/10.3390/electronics13214302>

Academic Editor: M-Tahar Kechadi

Received: 30 September 2024

Revised: 25 October 2024

Accepted: 30 October 2024

Published: 31 October 2024



**Copyright:** © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

With the rapid advancement of machine learning technologies, deep learning has become a key solution to numerous computer vision challenges, typically requiring large amounts of labeled data for model training. Semantic segmentation, a critical task in this domain, assigns each pixel in an image to a specific semantic category, relying on pixel-level annotation. This technique is widely applied across various fields, playing a vital role in enhancing automation, accuracy, and efficiency. In autonomous driving, semantic segmentation enables vehicles to recognize essential elements such as roads, pedestrians, and traffic signs, facilitating intelligent navigation. In medical imaging, it assists doctors in accurately identifying and analyzing tissues or lesions, improving the precision and efficiency of diagnosis and treatment. In industry, it empowers robots to reliably distinguish different objects in their operational environment, aiding in more accurate task execution [1].

Unsupervised domain adaptation (UDA) is increasingly recognized as an effective method to reduce the burden of data annotation. Traditional supervised learning relies on labeled datasets for model training, but acquiring such labeled data is both costly and time consuming. To circumvent the laborious process of dataset annotation, UDA has emerged as a solution. It typically leverages annotated source-domain data for training, while using unlabeled target-domain data for testing. The objective of UDA is to ensure that neural networks trained on the source domain can generalize well and perform effectively on the target domain.

Despite recent advancements, UDA for semantic segmentation still faces several challenges and limitations. First, in practical applications, variations in data due to different environments and conditions often lead to differences in feature distribution between the source and target domains, a phenomenon known as domain shift. This domain shift can lead to reduced model performance, as the model may overfit to the source-domain features and fail to generalize to the target domain. Additionally, the lack of labeled data in the target domain makes it more difficult to extract effective features. Another challenge lies in maintaining the balance between achieving high accuracy in the source domain and transferring this accuracy effectively to the target domain without performance degradation.

Moreover, current UDA methods often struggle with scalability and adaptability to real-world scenarios, where conditions and data distributions are highly dynamic and unpredictable. Developing robust UDA methods that can handle such variability is crucial for practical deployment. To address these challenges, we propose Polarized Attention Network Domain Adaptation (PANDA), a novel approach incorporating Polarized Self-Attention (PSA) [2] for UDA semantic segmentation. By combining two PSA blocks with a standard convolution block, PANDA effectively extracts complex channel and spatial features while maintaining high resolution, enhancing the understanding of objects with varying sizes and shapes by capturing more informative features.

## 2. Related Work

### 2.1. Semantic Segmentation

Semantic segmentation plays a critical role in intelligent vehicle perception, offering pixel-level semantic insights that are essential for tasks such as drivable area detection and object landmark identification. With the advent of deep learning, there has been rapid progress in developing more sophisticated segmentation techniques. In recent years, various methods have been proposed to improve the performance of semantic segmentation [3]. For instance, prototype learning [4,5] focuses on establishing a latent feature space where predictions are derived by measuring the distance between the test anchor and the prototypes for each class. On the other hand, pixel-wise contrastive learning [6] is applied to strengthen the similarity between pixels of the same semantic category.

Despite these advancements, the high cost of acquiring annotated datasets for training remains a challenge, leading researchers to explore simulated data as a cost-effective alternative. However, the large differences between synthetic and real-world environments, as well as varying weather conditions [7,8], still make it hard for models to perform well in all situations.

### 2.2. UDA

In typical machine learning problems, it is often assumed that the training and test datasets follow similar distributions. However, real-world applications frequently encounter significant differences between these datasets. As a result, models trained on the training dataset may perform poorly on the test dataset. Transfer learning has emerged as a solution for mitigating distributional differences. Within this field, UDA has become crucial for tackling such challenges. Various studies have been explored for UDA in different tasks like image classification [9–12], object detection [13–17], and semantic segmentation [18–22].

UDA methods are generally categorized into three primary approaches that include discrepancy minimization, adversarial training, and self-training. Firstly, discrepancy minimization methods aim to narrow the gap between source and target domains by mapping their features into a shared space. Techniques like maximum mean discrepancy [23] and correlation alignment [24] are employed to minimize the distance between these domains. Secondly, adversarial training [25,26] utilizes generative adversarial network (GAN) to align features across domains. In this approach, a generator produces domain-invariant features, while a discriminator distinguishes between features from the source and target domains. Through this process, the generator learns to generate features that align with

the target domain. Finally, self-training employs a pretrained model to generate pseudo-labels for the target domain. Strategies like confidence thresholds [27] and pseudo-label prototypes [28] help mitigate the issues of pseudo-label drift.

Most state-of-the-art UDA methods rely on self-training. For instance, DAFormer [18] utilizes Transformer to capture dependencies over long ranges, while HRDA [19] integrates features from both high and low resolutions to capture contextual relationships across different scales. Masked Image Consistency (MIC) [20] encourages the model to learn contextual relationships by generating masked regions in target-domain images. However, accurately capturing fine-grained features remains a significant challenge in this field.

### 2.3. Attention Mechanism

The fundamental concept behind attention mechanisms is to assign varying levels of importance to different parts of input data, enabling models to selectively focus on relevant features essential for the task. Efficient Attention (EA) [29] modifies the order of multiplication to reduce the need for computing relationships between every position, achieving computational efficiency that scales linearly with input size. SENet [30] uses global average pooling to generate a channel representation, which is then processed through a fully connected layer to produce channel weights that are applied to the original feature map. GCNet [31] simplifies computation by calculating a general attention map for all positions in the input feature map, replacing the compression step in SENet [30] to reduce computation costs while preserving global information. CBAM [32] independently applies attention operations to both the channel and spatial dimensions, avoiding extensive convolutional operations typical in traditional attention mechanisms.

While these attention mechanisms maintain high resolution in the channel and spatial dimensions, they trade off with increased time complexity and sensitivity to noise. Although some alternative methods are more computationally efficient, they often compromise on preserving high-resolution features. To tackle challenges in complex semantic segmentation tasks, we employ Polarized Self-Attention (PSA) [2] to capture contextual information effectively, thereby enhancing adaptation between the source and target domains. As a result, PANDA preserves high-resolution features in both the channel and spatial dimensions, proving advantageous for practical applications in real-world scenarios.

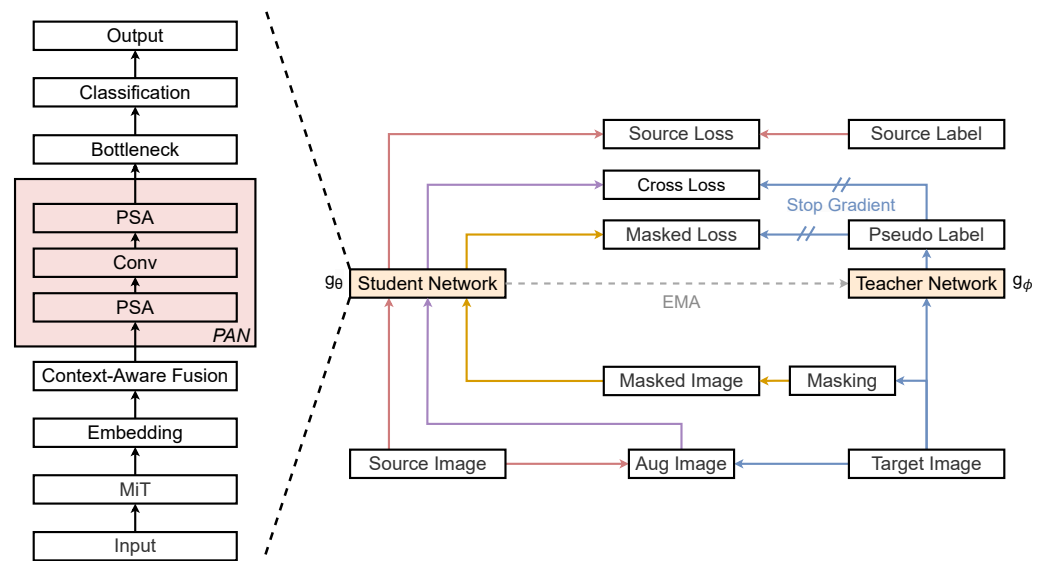
## 3. Proposed Approaches

### 3.1. Overview

PANDA is designed to capture richer contextual information within images. As shown in Figure 1, the PANDA model consists of three main components, which are source domain training, target-domain training, and cross-domain training. The abbreviations used in the paper are listed in the Abbreviations section.

PANDA includes a teacher model and a student model, both identical in structure. The teacher model generates pseudo-labels for the unlabeled target-domain data (blue), and the model is jointly optimized using the supervised source loss (red), the augmented cross-loss (purple), and the masked loss (yellow). Additionally, the parameters of the teacher model are updated using exponential moving average (EMA) (gray). Inspired by [33], we integrate Polarized Attention Network (PAN) into both the student network  $g_\theta$  and teacher network  $g_\phi$ , enabling the model to better capture key features within images. The network includes several key components: MiT Encoder [34], embedding layer, context-aware fusion layer, bottleneck module, and classification layer. In Section 3.3, we provide a detailed description of each component, explaining their functionalities and contributions to the overall network.

Firstly, for the source domain, we employ rare-class sampling (RCS) to increase the sampling probability of images containing rare classes. This ensures that the model can still effectively learn from classes with a limited number of training samples. The selected RCS images are fed into both the student network and ImageNet Encoder, where respective predictions are made, and the source loss and feature distance loss are computed.



**Figure 1.** Overview of the proposed PANDA architecture.

For the target domain, masked images are input into the student network to guide the learning process. This allows the model to confidently predict meaningful features, even under masked conditions. The predicted results, along with the pseudo-labels generated by the teacher network from the complete target-domain image, are used to compute the masked loss.

Lastly, in the cross-domain component, we adopt ClassMix [35] to combine the source and target domains images for generating mixed images. These mixed images provide a more comprehensive and diverse training set. The student network adopts the mixed images to make predictions, while the pseudo-labels and annotations from the source domain are similarly mixed and used to compute the mixed loss.

In conclusion, PANDA employs a combination of strategies tailored to source, target, and cross-domain training. This comprehensive approach allows the model to fully utilize the available information from each domain, improving its ability to generalize across varying conditions.

### 3.2. Self-Training for UDA

#### 3.2.1. Source-Domain Training

Given the source-domain images and their corresponding annotations, denoted as  $\mathcal{X}_S = \{(x_S^i, y_S^i)\}_{i=1}^{N_S}$ , the student network  $g_\theta$  is trained by calculating the source loss to achieve good performance on the target-domain image  $\mathcal{X}_T = \{x_T^i\}_{i=1}^{N_T}$ , which does not have annotations. Here,  $N_S$  and  $N_T$  represent the number of images in the source and target domains, respectively. However, models trained solely using the source-domain loss typically struggle to generalize to the target domain, leading to poor performance of the student network when applied to target-domain images. Additionally, adversarial training approaches [28,36] are often unstable and have been outperformed by self-training methods [18–20]. Therefore, we adopt a self-training approach by utilizing an additional teacher network  $g_\phi$  to generate reliable pseudo-labels for the target domain.

Due to the long-tail distribution characteristics of the dataset, there is a possibility that random sampling might only select samples related to rare classes in the later stages of model training. Such a delay could result in the model becoming biased toward common classes early on, making it difficult to relearn these rare classes with only a few available samples. To prevent the early neglect of rare classes during training, we adopt an RCS strategy based on DAFormer [18]. This strategy increases the sampling frequency of samples containing rare classes, to mitigate the impact of class imbalance on the model's



training process. Specifically, the sampling frequency  $f_c$  for each class  $c$  is computed based on the percentage of pixels in the dataset belonging to class  $c$ :

$$f_c = \frac{1}{N_S \cdot H \cdot W} \sum_{n=1}^{N_S} \sum_{k=1}^{H \times W} [y_S^{n,k,c}] \quad (1)$$

where  $H \times W$  denotes image height and width, and  $[\cdot]$  represents the Iverson bracket, which equals 1 if the condition inside the brackets is true and 0 otherwise. The sampling probability  $P(c)$  for class  $c$  is calculated using a Softmax function related to  $f_c$ , as shown in Equation (2). Classes with lower sampling frequencies have higher probabilities, with temperature  $T$  controlling the smoothness of the distribution. A higher  $T$  leads to a more uniform distribution, while a lower  $T$  results in more frequent sampling of rare classes with smaller  $f_c$ .

$$P(c) = \frac{e^{(1-f_c)/T}}{\sum_{c'=1}^C e^{(1-f_{c'})/T}} \quad (2)$$

In practice, a class  $c$  is randomly selected, and an image containing that class is randomly sampled from the subset of data containing that class for training. The selected source-domain image is then fed into the student network  $g_\theta$  to generate pixel-wise predictions. The source loss  $\mathcal{L}_S$  is subsequently computed by comparing these predictions with the ground truth using cross-entropy:

$$\mathcal{L}_S = - \sum_{k=1}^{H \times W} \sum_{c=1}^C y_S^{k,c} \log g_\theta(x_{RCS}^{k,c}) \quad (3)$$

To prevent the model from overfitting when processing source-domain images and to preserve useful features obtained through ImageNet pretraining, we utilize the feature distance (FD) proposed in DAFormer [18]. This method normalizes the L2 distance between the bottleneck features  $F_\theta$  of the student network  $g_\theta$  and the bottleneck features  $F_{ImageNet}$  of ImageNet:

$$d^k = \|F_{ImageNet}(x_{RCS}^k) - F_\theta(x_{RCS}^k)\|^2 \quad (4)$$

Given that ImageNet primarily trains on thing classes (objects with specific sizes or shapes such as vehicles, riders, etc.) rather than stuff classes (background regions without specific shapes such as sky, buildings, etc.), the loss is computed only for image regions containing the thing class  $C_{things}$ . Firstly, the labels  $y_S$  are downsampled to match the size of the bottleneck features  $H_F \times W_F$ , followed by global average pooling (GAP) over the channels for each category. If the pooled value exceeds a threshold proportion  $r_{ImageNet}$ , the category is retained to obtain the downsampled labels  $y_{S,small}$ :

$$y_{S,small}^c = \delta \left( \text{AvgPool}(y_S^c, \frac{H}{H_F}, \frac{W}{W_F}) \right) \quad (5)$$

$$\delta(x) = \begin{cases} 1, & \text{if } x > r_{ImageNet} \\ 0, & \text{otherwise} \end{cases} \quad (6)$$

Subsequently, to ensure that the feature map corresponds to the considered thing class, only the portion of  $y_{S,small}$  belonging to the thing class is retained to generate the binary mask:

$$M_{things}^k = \sum_{c'=1}^C y_{S,small}^{k,c'} \cdot [c' \in C_{things}] \quad (7)$$

Finally, the binary mask  $M_{things}$  is applied to calculate the average distance  $d^k$  for the thing classes, which represents the feature distance loss:

$$\mathcal{L}_{FD} = \frac{\sum_{k=1}^{H_F \times W_F} d^k \cdot M_{things}^k}{\sum_{k=1} M_{things}^k} \quad (8)$$

### 3.2.2. Target-Domain Training

To enhance the model's ability to learn contextual relationships in the target domain, we use MIC [20], which incorporates random masks into target-domain image to infer results close to pseudo-labels from information surrounding the masks. The mask generation involves randomly sampling a patch mask  $M$  from a uniform distribution:

$$M_{ib+1:(i+1)b, jb+1:(j+1)b} = [v < r^{Mask}], \quad v \sim U(0, 1) \quad (9)$$

where  $i$  and  $j$  are patch indices with  $i \in [0 \dots H/b - 1]$  and  $j \in [0 \dots W/b - 1]$ ,  $b$  is the patch size, and  $r^{mask}$  represents the mask ratio. The masked target image is obtained through element-wise multiplication of the mask and the target-domain image:

$$x_{Mask} = M \odot x_T \quad (10)$$

To ensure that the model can accurately reconstruct labels even in the absence of complete target-domain images and utilize contextual information effectively, we use the masked loss  $\mathcal{L}_M$  to maintain consistency between the outputs of the student network  $g_\theta$  and the teacher network  $g_\phi$ . The prediction of  $x_{Mask}$  is compared with pseudo-label  $p_T$  using cross-entropy:

$$L_M = - \sum_{k=1}^{H \times W} \sum_{c=1}^C q_T p_T^{(k,c)} \log g_\theta(x_{Mask}^{k,c}) \quad (11)$$

where  $p_T$  represents the pseudo-label, as defined in Equation (12), derived from the class with the maximum Softmax value in the predictions obtained by inputting the target-domain image into the teacher network  $g_\phi$ . Importantly, gradients are not propagated back to the teacher network.

$$p_T = [c = \arg \max_c g_\phi(x_T)] \quad (12)$$

Since the pseudo-labels may be incorrect, their quality is weighted based on the confidence estimate  $q_T$ . This is computed as the proportion of pixels where the maximum Softmax value in the prediction exceeds the threshold  $\tau$ , defined as Equation (13). As training iterations proceed, the generated pseudo-labels become more accurate, Softmax values increase,  $q_T$  grows larger, and  $\mathcal{L}_M$  imposes a stronger constraint on the model, facilitating the learning of more precise contextual information.

$$q_T = \frac{\sum_{k=1}^{H \times W} [\max_{c'} g_\phi(x_T^{k,c'}) > \tau]}{H \cdot W} \quad (13)$$

Typically, the teacher network is designed as an EMA teacher [37] to enhance prediction stability. The weights of the teacher network  $g_\phi$  are updated as the exponential moving average of the weights of the student network  $g_\theta$  with a smoothing factor  $\alpha$ :

$$\phi_{t+1} \leftarrow \alpha \phi_t + (1 - \alpha) \theta_t \quad (14)$$

where  $t$  denotes a training step.  $\phi$  and  $\theta$  represent the weights corresponding to the teacher network and the student network, respectively. Implementing a temporal ensemble with past student models, the EMA teacher improves the reliability and temporal stability of the pseudo-labels. As it updates based on  $g_\theta$ , the teacher progressively enhances its feature learning capacity.

### 3.2.3. Cross-Domain Training

Following DACS [36], we use color jitter, Gaussian blur, and ClassMix [35] to produce the augmented image. From the RCS image  $x_{RCS}$ , we randomly select a half-set of classes  $H$ . Then, we generate a binary mask:

$$M_{Mix} = \begin{cases} 1, & \text{if } y_S \in H \\ 0, & \text{otherwise} \end{cases} \quad (15)$$

Then, we apply  $M_{Mix}$  to the corresponding pixels of the target image  $x_T$  to create the mixed image  $x_{Mix}$ , defined in Equation (16). To generate labels corresponding to  $x_{Mix}$ , we blend the pseudo-label  $p_T$  with source label  $y_S$  in the same manner to obtain mixed labels  $y_{Mix}$ , as shown in Equation (17).

$$x_{Mix} = M_{Mix} \odot x_{RCS} + (1 - M_{Mix}) \odot x_T \quad (16)$$

$$y_{Mix} = M_{Mix} \odot y_S + (1 - M_{Mix}) \odot p_T \quad (17)$$

The student network is trained using  $x_{Mix}$ , which enhances the learning of domain-robust features. The cross-loss  $L_C$  is used to further train the student  $g_\theta$  on the source domain:

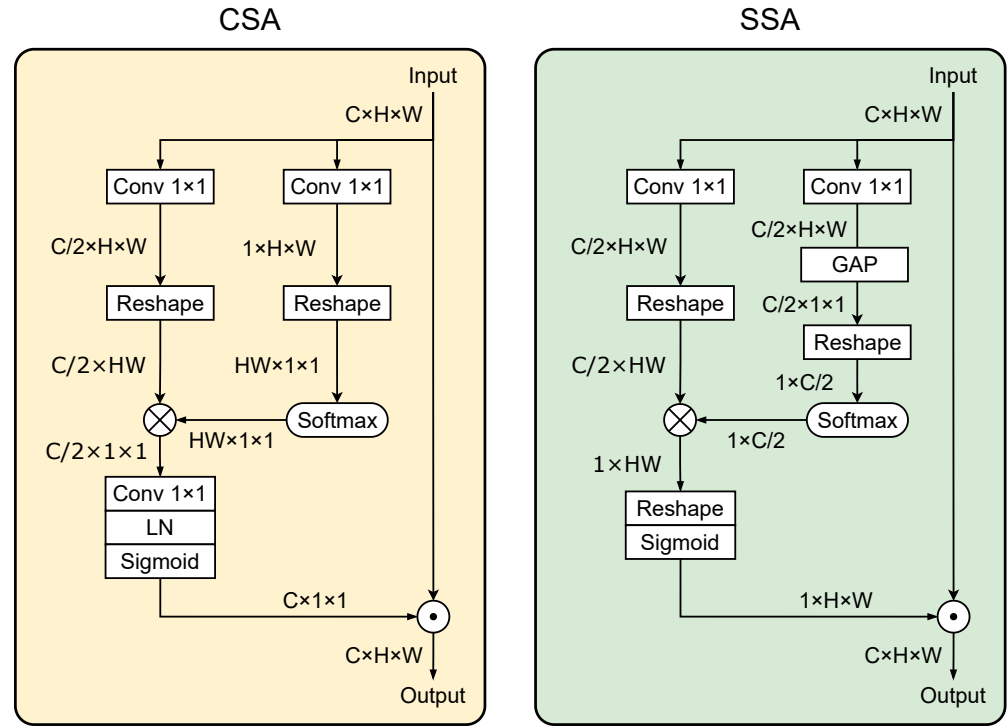
$$L_C = - \sum_{k=1}^{H \times W} \sum_{c=1}^C y_{Mix}^{(k,c)} \log g_\theta(x_{Mix}^{k,c}) \quad (18)$$

The overall PANDA loss  $L$  is calculated as the weighted sum of the individual loss components, expressed as  $L = L_S + \lambda_{FD} L_{FD} + L_M + L_C$ , which incorporates the supervised source-domain loss  $L_S$ , feature distance loss  $L_{FD}$ , masked loss  $L_M$ , and cross-domain loss  $L_C$ . In the training process, the source-domain loss, mask loss, and cross-loss are all computed using cross-entropy, ensuring consistent learning across both the source and target domains. Additionally, the feature distance loss employs distance averaging and introduces a balancing weight  $\lambda_{FD}$  to adjust its impact within the overall loss function. This component preserves effective features learned from ImageNet to guide model learning. The total loss allows PANDA to progressively adapt to different domain data distributions, thereby improving its ability to adapt to the unlabeled target domain over time.

### 3.3. Polarized Attention Network

According to Figure 1, the network comprises key components such as MiT Encoder [34], embedding layer, context-aware fusion layer, pan, bottleneck module, and classification layer. The multi-level features from MiT Encoder are first transformed into a consistent channel dimension through the embedding layer, then upsampled to match the target dimensions. These features are concatenated along the channel dimension and input into the context-aware fusion layer, which fuses the multi-level features. The output is then passed to PAN, which consists of two PSA blocks and a convolutional layer. The bottleneck module reduces the dimensionality, and final predictions are generated through the classification layer.

Since Polarized Self-Attention (PSA) is the core module of the PAN architecture, we first provide a detailed explanation of PSA. PSA comprises two attention mechanism structures: channel-only self-attention (CSA) and spatial-only self-attention (SSA), as shown in Figure 2. CSA focuses on the importance of each channel, extracting critical channel-specific information, while SSA highlights spatial feature information, corresponding to the location of the target within the feature map.



**Figure 2.** The structure of PSA. Channel-only self-attention (CSA) is in the left half, and spatial-only self-attention (SSA) is in the right half. LN is layer normalization.

Given a feature map  $X \in \mathbb{R}^{C \times H \times W}$  as input, the channel-wise attention weight  $A^{ch}(X) \in \mathbb{R}^{C \times 1 \times 1}$  is calculated as follows:

$$A^{ch}(X) = F_{SG}[F_{LN}(W_z(\sigma_1(W_v(X)) \times F_{SM}(\sigma_2(W_q(X)))))] \quad (19)$$

where  $X$  represents the tensor derived from either the context-aware fusion layer or the standard convolution module;  $W_q$ ,  $W_v$ , and  $W_z$  are the  $1 \times 1$  convolutional layers;  $\sigma_1$  and  $\sigma_2$  are the tensor reshape operators;  $\times$  is a matrix dot-product operator;  $F_{SG}(\cdot)$  is a sigmoid operator;  $F_{LN}(\cdot)$  is a layer normalization; and  $F_{SM}(\cdot)$  is a Softmax operator. The output of CSA is  $Z^{ch} = A^{ch}(X) \odot X \in \mathbb{R}^{C \times H \times W}$ , where  $\odot$  is an element-wise multiplication operator.

In SSA structures, the spatial-wise attention weight  $A^{sp}(X) \in \mathbb{R}^{1 \times H \times W}$  is defined as follows:

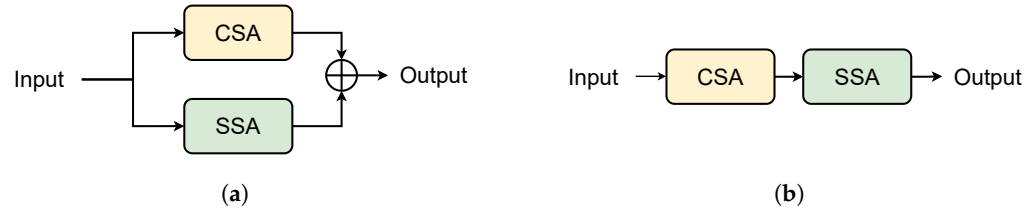
$$A^{sp}(X) = F_{SG}[\sigma_3(F_{SM}(\sigma_1(F_{GP}(W_q(X)))) \times \sigma_2(W_v(X)))] \quad (20)$$

where  $X$  denotes the tensor derived from either the context-aware fusion layer or the standard convolution module;  $W_q$  and  $W_v$  are the  $1 \times 1$  convolutional layers;  $\sigma_1$ ,  $\sigma_2$ , and  $\sigma_3$  are tensor reshape operators;  $\times$  is a matrix dot-product operator;  $F_{SG}(\cdot)$  is a sigmoid operator;  $F_{SM}(\cdot)$  is a Softmax operator; and  $F_{GP}(\cdot)$  is GAP. The output of SSA is  $Z^{sp} = A^{sp}(X) \odot X \in \mathbb{R}^{C \times H \times W}$ , where  $\odot$  is element-wise multiplication.

These two core structures, CSA and SSA, can be arranged in two layouts: parallel and sequential, as illustrated in Figure 3. In the sequential layout, there are two configurations: CSA $\rightarrow$ SSA and SSA $\rightarrow$ CSA. Pre-experiment results comparing these configurations on segmentation tasks for the GTA $\rightarrow$ Cityscape and SYNTHIA $\rightarrow$ Cityscape benchmarks indicate that the CSA $\rightarrow$ SSA configuration performs better on both benchmarks, as detailed in Table 1. Therefore, we adopt the CSA $\rightarrow$ SSA sequential layout for PAN in this work.

**Table 1.** Comparison of sequential layout configurations. The best performance in each column is shown in **bold**.

Method	GTA→Cityscape	SYNTHIA→Cityscape
MIC (baseline)	75.9	67.3
SSA→CSA	76.0	67.1
CSA→SSA	<b>76.2</b>	<b>68.1</b>

**Figure 3.** Illustration of the PSA block under different connection schemes: (a) Parallel layout and (b) sequential layout.

Hence, in the parallel layout,  $X$  is separately input into CSA and SSA, and their outputs are element-wise added to produce the parallel layout result, computed as Equation (21). In the sequential layout, it involves initially inputting into CSA, followed by passing the output of CSA to SSA, resulting in the sequential layout outcome, expressed as Equation (22).

$$PSA_p(X) = Z^{ch} + Z^{sp} = A^{ch}(X) \odot X + A^{sp}(X) \odot X \quad (21)$$

$$PSA_s(X) = Z^{sp}(Z^{ch}) = A^{sp}(A^{ch}(X) \odot X) \odot A^{ch}(X) \odot X \quad (22)$$

where  $+$  is an element-wise addition operator.  $PSA_p$  and  $PSA_s$  denote the way to fuse two structures in parallel and in sequence, respectively.

Inspired by [33], PAN employs two PSA blocks and a standard convolution module in both the student and teacher networks. We discuss the performance of different PSA layout combinations in Section 4.2. To capture more comprehensive features, the point-depth convolution used in [33] is replaced with a standard convolution module (denoted as Conv) to allow simultaneous capture of both spatial and channel features, resulting in richer feature representations. Specifically, taking the output of the first set of PSA as input, a  $3 \times 3$  convolution, denoted as  $W$ , is employed to extract features from the input. The extracted features undergo stabilization through batch normalization, represented as  $F_{BN}$ , which normalizes the features and aids in stabilizing the training process by reducing internal covariate shift. Following this, non-linear transformations are introduced using ReLU, denoted as  $F_{RL}$ , which enhances the network's ability to learn complex features by introducing non-linearity. This sequence of operations is crucial as it prepares the features to be fed into the second set of PSA, facilitating further refinement of features for downstream tasks. The output of PAN is computed as follows:

$$X_\mu = PSA(F_{RL}(F_{BN}(W(PSA(X))))) \quad (23)$$

Having established the PAN architecture, we now provide a detailed explanation of other critical layers in the network. These layers complement the PAN module by facilitating robust feature extraction. The embedding layer consists of four linear layers. The context-aware fusion layer, a variant of ASPP [38], differs by excluding GAP. The bottleneck module comprises a  $3 \times 3$  convolution, batch normalization, and ReLU, while the classification layer uses dropout and  $1 \times 1$  convolution. These components work in tandem with the PAN module to ensure efficient feature extraction and refinement, ultimately enhancing segmentation performance.



## 4. Experiments

### 4.1. Implementation Details

Our model is implemented in PyTorch 1.7.1+cu110, Python 3.8.5, and CUDA 11.4, running on a system with Ubuntu 18.04.5 LTS, a single NVIDIA Tesla V100 GPU with 32 GB memory, an Intel(R) Xeon(R) CPU E5-2698 v4 @ 2.20 GHz, and 256 GB RAM. We evaluate our proposed method on the two standard large-scale UDA segmentation benchmarks: GTA→Cityscapes and SYNTHIA→Cityscapes. The synthetic datasets GTA [39] and SYNTHIA [40] serve as the source domains, while Cityscapes [7] is used as the target domain. GTA [39] contains 24,966 synthetic images with a resolution of  $1914 \times 1052$ , and SYNTHIA [40] consists of 9400 synthetic images with a resolution of  $1280 \times 760$ . Cityscapes [7] provides 2975 training images and 500 validation images, each with a resolution of  $2048 \times 1024$ .

For evaluation, we employ the mean intersection over union (mIoU), as defined in Equation (24), which is a widely adopted metric for measuring segmentation performance.

$$\text{mIoU} = \frac{1}{n} \sum_{i=1}^n \frac{TP_i}{FN_i + FP_i + TP_i} \quad (24)$$

where  $TP$ ,  $FP$ , and  $FN$  are the number of true positive, false positive, and false negative pixels, respectively. Referencing [18–20], we evaluate on 19 classes for GTA [39] and 16 classes for SYNTHIA [40], ensuring compatibility with the Cityscapes dataset [7].

We train PANDA with a MiT-B5 encoder [34], initializing the backbone with ImageNet pretraining weights. In the UDA settings, we adopt the training parameters from MIC [20] and use the AdamW [41] optimizer. Specifically, we set the learning rate to  $6 \times 10^{-5}$  for the encoder and  $6 \times 10^{-4}$  for the decoder, applying a linear learning rate warmup during the first 1.5 k iterations. Afterward, a weight decay of 0.01 is adopted. We also use an EMA factor  $\alpha = 0.999$  and a quality threshold  $\tau = 0.968$ . During training, inputs are randomly cropped to a resolution of  $1024 \times 1024$ , with a batch of 2, for a total of 40 k iterations.

### 4.2. Variation in PSA Layout Combinations

In Table 2, we evaluate the influence of various PSA block configurations in the PAN architecture on the performance of UDA models. Six distinct configurations were explored, including single-PSA block models such as PANDA(P) and PANDA(S), as well as dual-PSA block models such as PANDA(P-P), PANDA(S-S), PANDA(S-P), and PANDA(P-S). Here, “P” represents a parallel layout, while “S” denotes a sequential layout.

The experimental results show that all PSA configurations outperform the baseline MIC method [20] in both GTA→Cityscapes and SYNTHIA→Cityscapes scenarios. Notably, the PANDA(S) model achieves the highest mIoU of 76.2% on the GTA→Cityscapes task, indicating that the sequential layout may allow for more refined stage-wise feature extraction. This suggests that sequential processing within a single PSA block is more effective than parallel processing for segmentation tasks, likely due to its ability to capture and refine feature information in multiple steps.

Moreover, dual-block models consistently outperform single-block configurations. Among the dual-block models, PANDA(P-S), which first implements a parallel layout followed by a sequential layout, demonstrates the best overall performance. It achieves an mIoU of 68.7% on the SYNTHIA→Cityscapes task and maintains competitive results on the GTA→Cityscapes task. The success of PANDA(P-S) lies in its ability to combine the broad feature extraction capabilities of the parallel block with the focused refinement afforded by the sequential block. This balance proves particularly advantageous, allowing the model to excel across diverse tasks.

Based on these observations, the PANDA(P-S) configuration, in particular, offers an optimal compromise by leveraging the strengths of both parallel and sequential attention, making it the preferred setup for further experiments and analysis.

**Table 2.** Performance comparison of different PSA arrangements. The best performance in each column is shown in **bold**.

Method	GTA→Cityscapes	SYNTHIA→Cityscapes
MIC [20]	75.9	67.3
PANDA(P)	76.1	67.8
PANDA(S)	<b>76.2</b>	68.1
PANDA(P-P)	76.0	68.5
PANDA(S-S)	76.0	67.7
PANDA(S-P)	76.1	68.0
PANDA(P-S)	76.1	<b>68.7</b>

#### 4.3. Comparison of Different Convolution Modules

In Table 3, we compare the standard convolution module inside PANDA with point-depth convolution used in [33]. The standard convolution module employs complete convolutional kernels to process all input channels, enabling it to capture richer features and contextual information. In contrast, point-depth convolution decomposes the convolution operation into pointwise and depthwise convolutions, reducing computational costs but potentially losing detail in feature extraction. On GTA→Cityscapes, PANDA with standard convolution achieves an mIoU of 76.1%, while the model utilizing point-depth convolution maintains the same performance, comparable to MIC [20]. Similarly, on SYNTHIA→Cityscapes, the improvement achieved using the standard convolution module significantly outperforms point-depth convolution. The mIoU improves by 1.4%, reaching 68.7% mIoU, compared to a 0.3% mIoU increase with point-depth convolution. The results indicate that the standard convolution module is more effective at capturing both detailed and global features in complex scenes for high-precision segmentation tasks. This might be due to the standard convolution module considering interactions between neighboring pixels during convolution, thereby preserving more channel and spatial information to enhance detailed feature extraction.

**Table 3.** Performance comparison of internal convolution modules.

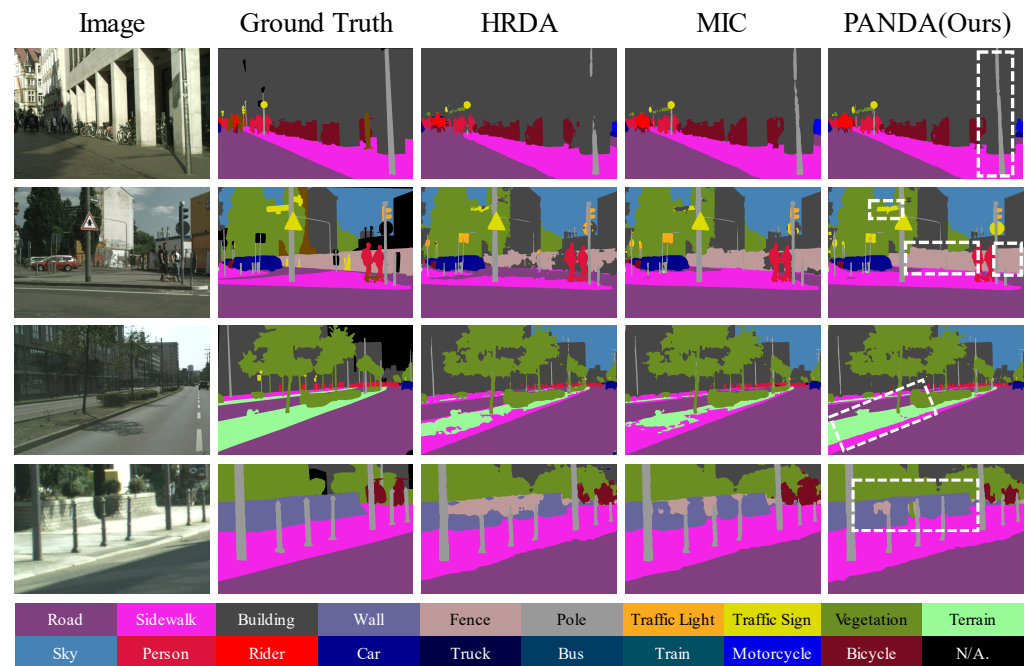
Method	Point-Depth Conv	Standard Conv	GTA→Cityscapes	SYNTHIA→Cityscapes
MIC [20]			75.9	67.3
PANDA	✓		75.9	67.6
PANDA		✓	76.1	68.7

#### 4.4. Comparison with State-of-the-Art Methods

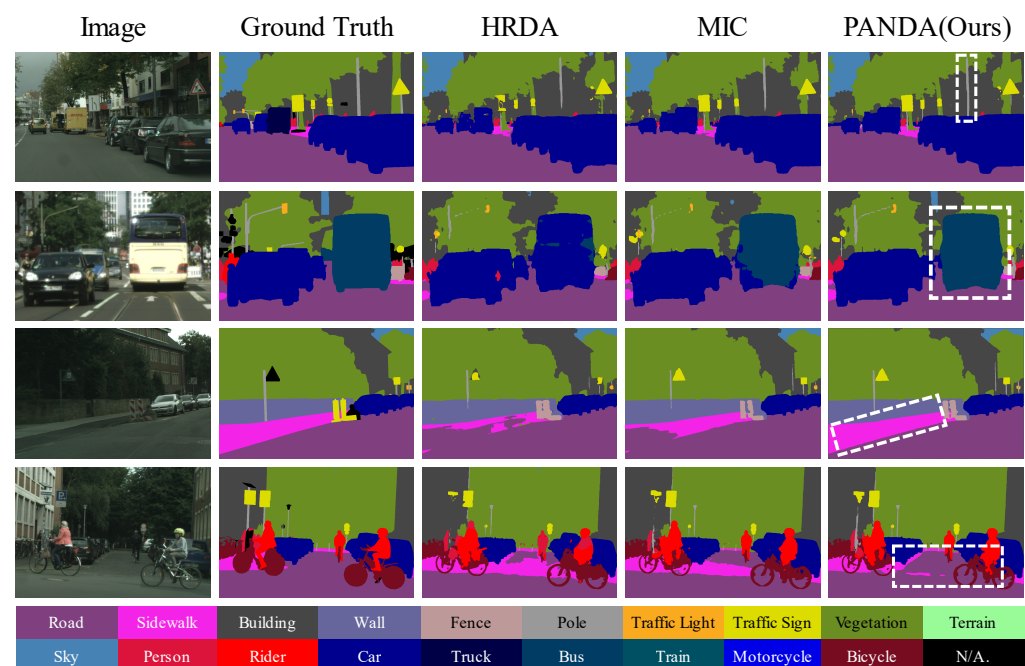
We evaluate the performance of our proposed method, PANDA, and compare it with various UDA methods. As detailed in Table 4, PANDA demonstrates notable improvements, achieving a +0.2% mIoU gain over MIC [20] on the GTA→Cityscapes task and a +1.4% mIoU gain on the SYNTHIA→Cityscapes task, reaching 76.1% and 68.7% mIoU, respectively. In a detailed class-wise IoU comparison, PANDA outperforms MIC [20] in 10 out of 19 categories for the GTA→Cityscapes task and in 9 out of 16 categories for the SYNTHIA→Cityscapes task. The most significant improvements are observed in categories such as *wall*, *pole*, and *traffic sign* on GTA→Cityscapes (see Figure 4); and *road*, *sidewalk*, and *bus* on SYNTHIA→Cityscapes (see Figure 5). The white dashed boxes highlight areas with significant improvements compared to other methods.

PANDA's superior performance can be attributed to two key factors. Firstly, its enhanced feature extraction capabilities allow it to capture finer details and more intricate patterns in specific categories, such as *pole* and *traffic sign*, which are often challenging due to their complex and subtle shapes. PANDA excels in learning and differentiating these variations. Secondly, PANDA leverages context-aware mechanisms that utilize surrounding environmental information to improve object identification and classification. This is particularly beneficial for categories like *road* and *sidewalk*, where understanding the

broader scene context is crucial for accurate segmentation. Together, these improvements allow PANDA to handle categories that were traditionally challenging due to variability and contextual dependencies.



**Figure 4.** Qualitative comparison of PANDA with previous methods on GTA→Cityscapes.



**Figure 5.** Qualitative comparison of PANDA with previous methods on SYNTHIA→Cityscapes.

Additionally, class imbalance in the datasets presents a challenge during model training, as it can cause the model to develop a bias toward high-frequency categories, thereby reducing its ability to recognize rare categories. To address this, PANDA not only improves the overall mIoU but also maintains a strong capacity to recognize rare categories. Specifically, when rigorously considering PANDA's performance in surpassing both MIC [20] and HRDA [19], excluding high-frequency categories (such as *road*, *building*, *sidewalk*, *sky*,

and *vegetation*), PANDA improved the recognition rate for 70% of the categories in the GTA→Cityscapes task and for 55% of the categories in the SYNTHIA→Cityscapes task.

Further analysis, considering cases where PANDA outperforms either MIC [20] or HRDA [19], shows that the recognition rate increases to 95% for the GTA→Cityscapes task and to 88% for the SYNTHIA→Cityscapes task across all categories. These results highlight PANDA’s significant advantage in addressing class imbalance, demonstrating strong generalization capabilities, particularly for challenging and rare categories. It is also important to note that while the performance on certain categories, such as *wall*, is slightly lower compared to the baseline, this is primarily due to the lower occurrence frequency of this category in the SYNTHIA dataset [40], which may have resulted in insufficient training data for the model. Overall, PANDA exhibits excellent performance across the majority of categories, particularly with significant improvements in the recognition of rare categories.

**Table 4.** Comparison with state-of-the-art methods for UDA. We present pre-class IoU and mIoU. The best performance in every column is shown in **bold**.

Method	Road	SW	Build.	Wall	Fence	Pole	TL	TS	Veg.	Terrain	Sky	Person	Rider	Car	Truck	Bus	Train	M.Bike	Bike	mIoU
GTA→Cityscapes																				
ADVENT [26]	89.4	33.1	81.0	26.6	26.8	27.2	33.5	24.7	83.9	36.7	78.8	58.7	30.5	84.8	38.5	44.5	1.7	31.6	32.4	45.5
DACS [36]	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 [28]	87.8	56.0	79.7	46.3	44.8	45.6	53.5	53.5	88.6	45.2	82.1	70.7	39.2	88.8	45.5	59.4	1.0	48.9	56.4	57.5
DAFormer [18]	95.7	70.2	89.4	53.5	48.1	49.6	55.8	59.4	89.9	47.9	92.5	72.2	44.7	92.3	74.5	78.2	65.1	55.9	61.8	68.3
HRDA [19]	96.4	74.4	91.0	61.6	51.5	57.1	63.9	69.3	91.3	48.4	94.2	79.0	52.9	93.9	84.1	85.7	75.9	63.9	67.5	73.8
MIC [20]	<b>97.4</b>	<b>80.1</b>	91.7	61.2	56.9	59.7	66.0	71.3	<b>91.7</b>	51.4	<b>94.3</b>	79.8	56.1	<b>94.6</b>	85.4	<b>90.3</b>	<b>80.4</b>	<b>64.5</b>	<b>68.5</b>	75.9
PANDA	97.3	79.0	<b>91.8</b>	<b>63.4</b>	<b>58.4</b>	<b>62.0</b>	<b>66.6</b>	<b>73.4</b>	91.4	<b>52.7</b>	93.3	<b>80.8</b>	<b>58.0</b>	94.4	<b>85.7</b>	86.6	80.2	64.3	66.9	<b>76.1</b>
SYNTHIA→Cityscapes																				
ADVENT [26]	85.6	42.2	79.7	8.7	0.4	25.9	5.4	8.1	80.4	-	84.1	57.9	23.8	73.3	-	36.4	-	14.2	33.0	41.2
DACS [36]	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
ProDA [28]	87.8	45.7	84.6	37.1	0.6	44.0	54.6	37.0	88.1	-	84.4	74.2	24.3	88.2	-	51.1	-	40.5	45.6	55.5
DAFormer [18]	84.5	40.7	88.4	41.5	6.5	50.0	55.0	54.6	86.0	-	89.8	73.2	48.2	87.2	-	53.2	-	53.9	61.7	60.9
HRDA [19]	85.2	47.7	88.8	<b>49.5</b>	4.8	57.2	65.7	60.9	85.3	-	92.9	79.4	52.8	89.0	-	64.7	-	63.9	<b>64.9</b>	65.8
MIC [20]	86.6	50.5	<b>89.3</b>	47.9	7.8	59.4	66.7	<b>63.4</b>	87.1	-	<b>94.6</b>	<b>81.0</b>	<b>58.9</b>	90.1	-	61.9	-	67.1	64.3	67.3
PANDA	<b>91.7</b>	<b>59.8</b>	89.1	45.0	<b>9.4</b>	<b>61.8</b>	<b>68.1</b>	62.3	<b>88.5</b>	-	94.4	80.9	58.5	<b>90.2</b>	-	<b>67.3</b>	-	<b>67.8</b>	63.8	<b>68.7</b>

#### 4.5. Ablation Study

In this section, we investigate the effects of various attention mechanisms on model performance and compare their effectiveness with that of PANDA. Specifically, we evaluate the performance of PAN when substituted with several attention mechanisms, including EA [29], SENet [30], GCNet [31], and CBAM [32]. To enhance understanding, we begin by clarifying the columns presented in Table 5, which provides an ablation study on the effects of these attention mechanisms. The columns represent key performance metrics, such as mIoU, as well as computational costs, including the number of parameters, memory usage, and throughput associated with each method. The channel resolution and spatial resolution columns indicate the dimensions of the feature maps processed by each attention mechanism, where a larger resolution typically allows for more detailed information. The non-linearity column specifies the activation functions used, which can impact the model’s ability to capture complex patterns. The GTA→CS (mIoU) and SYN→CS (mIoU) columns report mIoU performance on the respective tasks, serving as a measure of segmentation accuracy. The param (M) column denotes the number of parameters in millions, providing insight into the model’s complexity. The memory (GB) column indicates the amount of GPU memory consumed during inference, which is crucial for understanding resource requirements. Finally, the throughput (img/s) column reflects the model’s processing speed, representing how many images can be processed per second.

According to Table 5, none of these four attention mechanisms yield significant performance improvements on the GTA→Cityscapes task. The observed enhancement in PANDA’s performance can be attributed to PSA [2], which retains a larger number of channels ( $\frac{C}{2}$ ) and spatial dimensions ( $[W, H]$ ). Furthermore, unlike the other methods, PSA utilizes a combination of Softmax and sigmoid as non-linear functions, which helps approximate more realistic and refined output results.

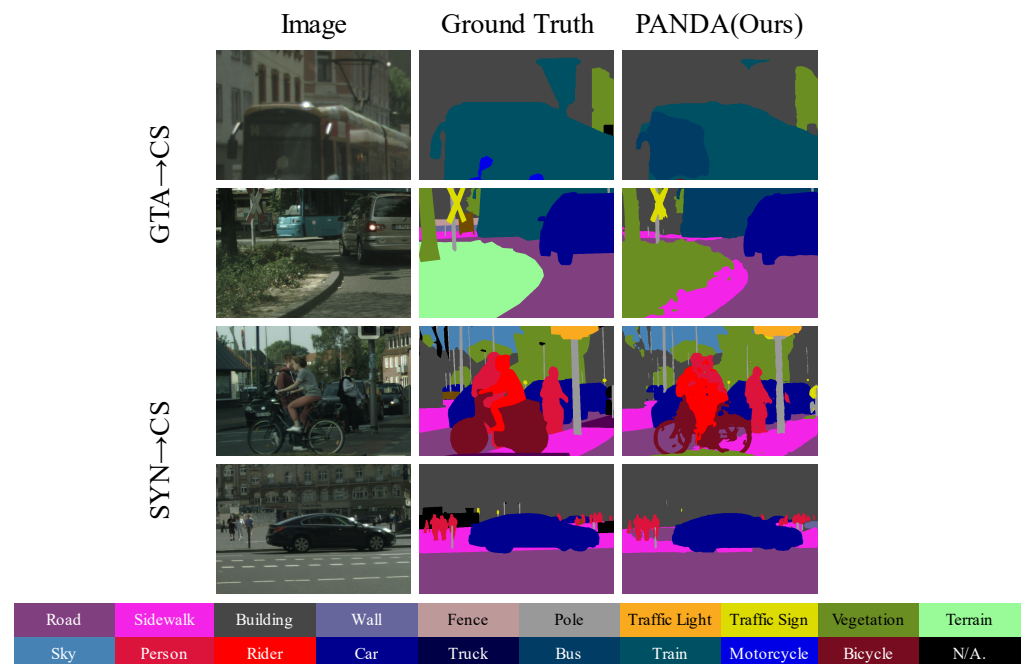
For the SYNTHIA→Cityscapes task, while the alternative attention mechanisms yield slight improvements, PANDA achieves a more substantial enhancement in segmentation performance compared to these mechanisms. Notably, PANDA increases the total number of parameters by only 1.69% compared to traditional attention mechanisms, indicating its efficiency in maintaining model complexity. Additionally, PANDA leads to a 2.41% reduction in GPU memory consumption, demonstrating its advantage in resource usage over conventional attention methods. Moreover, PANDA exhibits a 7.25% decrease in training speed relative to these approaches, emphasizing its effectiveness in balancing performance and computational overhead. Overall, these experiments highlight that PANDA is more suitable for UDA semantic segmentation tasks than previous attention mechanisms, balancing improved performance with reduced computational overhead.

**Table 5.** Ablation study on the effects of PSA and different attention blocks.

Method	Channel Resolution	Spatial Resolution	Non-Linearity	GTA→CS (mIoU)	SYN→CS (mIoU)	Param (M)	Memory (GB)	Throughput (img/s)
MIC [20]	-	-	-	75.9	67.3	85.69	22.60	0.98
+EA [29]	≪C	≪min(W, H)	Softmax	-	-	103.5	30.88	-
+SENet [30]	C/4	-	ReLU + Sigmoid	75.6	68.1	95.65	25.51	0.65
+GCNet [31]	C/4	-	ReLU + Softmax	74.8	67.9	96.18	25.55	0.73
+CBAM [32]	C/16	[W,H]	Sigmoid	75.6	68.4	95.39	26.02	0.70
PANDA	C/2	[W,H]	Sigmoid + Softmax	76.1	68.7	99.33	26.34	0.64

#### 4.6. Failure Case Analysis

In this section, we provide three representative failure cases of PANDA. As presented in Figure 6, PANDA confuses semantically similar objects such as *train* vs. *bus* and *terrain* vs. *vegetation*. In addition, it is extremely challenging to distinguish partially occluded objects (e.g., differentiating between *person* and *rider*, or identifying *sidewalk* behind *car*). In the future, we will therefore focus on addressing these issues.



**Figure 6.** Failure cases of segmentation results on GTA→Cityscapes (rows 1 and 2) and SYNTHIA→Cityscapes (rows 3 and 4).



## 5. Conclusions

In this study, we introduced PANDA, a novel model for unsupervised domain adaptation (UDA) in semantic segmentation that effectively integrates advanced design elements to enhance feature extraction capabilities. Our experiments revealed that PANDA significantly outperforms existing state-of-the-art methods, including MIC, achieving mIoU scores of 76.1% on GTA→Cityscapes and 68.7% on SYNTHIA→Cityscapes, reflecting respective gains of +0.2% and +1.4%.

The detailed analysis of various attention mechanisms revealed that traditional models, such as EA, SENet, GCNet, and CBAM, failed to achieve significant improvements on the GTA→Cityscapes. In contrast, PANDA employs Polarized Self-Attention, which effectively preserves a greater number of both channel and spatial dimensions that are essential for capturing intricate and contextually relevant features. Furthermore, the unique combination of Softmax and sigmoid non-linear functions within the PSA architecture optimizes feature selection, leading to more refined output predictions.

Additionally, when evaluating model complexity and computational efficiency, PANDA shows a minor increase in total parameter count. However, it offsets this by reducing GPU memory consumption and accelerating training times, making it not only highly accurate but also efficient for real-time deployment scenarios. This dual benefit is particularly advantageous in applications requiring both high performance and low latency. While acknowledging the model's limitations in handling rare classes in certain instances, PANDA demonstrates considerable success in addressing class imbalance and boosting recognition across the majority of categories. This indicates that the model's design choices, such as PSA and advanced feature fusion, significantly contribute to its robustness. Looking ahead, our future work will focus on refining PANDA's ability to handle rare categories more effectively, thereby improving its applicability to highly imbalanced datasets.

In summary, PANDA offers a balanced and effective solution for addressing the challenges of cross-domain learning tasks, particularly in handling class imbalances and enhancing the recognition of rare or challenging categories. Its performance demonstrates resilience and adaptability, reinforcing its potential as a viable unsupervised domain adaptation model for both academic research and practical applications in semantic segmentation.

**Author Contributions:** Conceptualization, C.-W.K., W.-L.C. and C.-C.L.; methodology, W.-L.C. and C.-W.K.; software, W.-L.C.; validation, W.-L.C. and C.-W.K.; formal analysis, W.-L.C. and C.-W.K.; investigation, W.-L.C.; resources, C.-W.K., C.-C.L. and K.-C.F.; data curation, W.-L.C.; writing—original draft preparation, W.-L.C. and C.-W.K.; writing—review and editing, C.-W.K. and W.-L.C.; visualization, W.-L.C.; supervision, C.-W.K. and C.-C.L.; project administration, C.-W.K. and C.-C.L.; funding acquisition, K.-C.F. and C.-W.K. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was funded by the National Science and Technology Council, Taiwan, R.O.C., grant numbers NSTC 113-2221-E-008-102-MY3 and NSTC 112-2222-E-130-001.

**Data Availability Statement:** The original data presented in the study are openly available in [7,39,40].

**Conflicts of Interest:** The authors declare no conflicts of interest.

## Abbreviations

The following abbreviations are used in this manuscript:

UDA	Unsupervised domain adaptation
PANDA	Polarized Attention Network Domain Adaptation
PSA	Polarized Self-Attention
mIoU	Mean intersection over union
GTA	Grand Theft Auto
GAN	Generative adversarial network
HRDA	High-Resolution Domain-Adaptive
MIC	Masked Image Consistency

EA	Efficient attention
SENet	Squeeze-and-excitation network
GCNet	Global Context Network
CBAM	Convolutional Block Attention Module
EMA	Exponential moving average
PAN	Polarized Attention Network
MiT	Mix Transformer
RCS	Rare Class Sampling
FD	Feature distance
GAP	Global average pooling
DACS	Domain Adaptation via Cross-domain Mixed Sampling
ASPP	Atrous Spatial Pyramid Pooling
CSA	Channel-only Self-attention
SSA	Spatial-only Self-attention
Conv	Convolution
SW	Sidewalk
Build.	Building
TL	Traffic light
TS	Traffic sign
Veg.	Vegetation
M.Bike	Motorcycle
SYN	SYNTHIA
CS	Cityscapes

## References

1. Toldo, M.; Maracani, A.; Michieli, U.; Zanuttigh, P. Unsupervised domain adaptation in semantic segmentation: A review. *Technologies* **2020**, *8*, 35. [[CrossRef](#)]
2. Liu, H.; Liu, F.; Fan, X.; Huang, D. Polarized self-attention: Towards high-quality pixel-wise regression. *arXiv* **2021**, arXiv:2107.00782.
3. Zhou, T.; Zhang, F.; Chang, B.; Wang, W.; Yuan, Y.; Konukoglu, E.; Cremers, D. Image Segmentation in Foundation Model Era: A Survey. *arXiv* **2024**, arXiv:2408.12957.
4. Wang, K.; Liew, J.H.; Zou, Y.; Zhou, D.; Feng, J. Panet: Few-shot image semantic segmentation with prototype alignment. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27 October–2 November 2019; pp. 9197–9206.
5. Zhou, T.; Wang, W.; Konukoglu, E.; Van Gool, L. Rethinking semantic segmentation: A prototype view. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 19–20 June 2022; pp. 2582–2593.
6. Zhou, T.; Wang, W. Cross-image pixel contrasting for semantic segmentation. *IEEE Trans. Pattern. Anal. Mach. Intell.* **2024**, *46*, 5398–5412. [[CrossRef](#)] [[PubMed](#)]
7. Cordts, M.; Omran, M.; Ramos, S.; Rehfeld, T.; Enzweiler, M.; Benenson, R.; Franke, U.; Roth, S.; Schiele, B. The cityscapes dataset for semantic urban scene understanding. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 27–30 June 2016; pp. 3213–3223.
8. Sakaridis, C.; Dai, D.; Van Gool, L. ACDC: The adverse conditions dataset with correspondences for semantic driving scene understanding. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Virtual, 11–17 October 2021; pp. 10765–10775.
9. Chen, L.; Chen, H.; Wei, Z.; Jin, X.; Tan, X.; Jin, Y.; Chen, E. Reusing the task-specific classifier as a discriminator: Discriminator-free adversarial domain adaptation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 7181–7190.
10. Wang, Q.; Meng, F.; Breckon, T.P. Data augmentation with norm-AE and selective pseudo-labelling for unsupervised domain adaptation. *Neural Netw.* **2023**, *161*, 614–625. [[CrossRef](#)] [[PubMed](#)]
11. Zhu, J.; Bai, H.; Wang, L. Patch-Mix Transformer for Unsupervised Domain Adaptation: A Game Perspective. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 18–22 June 2023; pp. 3561–3571.
12. Singh, I.P.; Ghorbel, E.; Kacem, A.; Rathinam, A.; Aouada, D. Discriminator-free unsupervised domain adaptation for multi-label image classification. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, Waikoloa, HI, USA, 3–8 January 2024; pp. 3936–3945.
13. Mattolin, G.; Zanella, L.; Ricci, E.; Wang, Y. ConfMix: Unsupervised Domain Adaptation for Object Detection via Confidencebased Mixing. In Proceedings of the 2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), Waikoloa, HI, USA, 2–7 January 2023; pp. 423–433.

14. Kennerley, M.; Wang, J.G.; Veeravalli, B.; Tan, R.T. 2pcnet: Two-phase consistency training for day-to-night unsupervised domain adaptive object detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 17–23 June 2023; pp. 11484–11493.
15. VS, V.; Oza, P.; Patel, V.M. Towards online domain adaptive object detection. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, Waikoloa, HI, USA, 2–7 January 2023; pp. 478–488.
16. VS, V.; Oza, P.; Patel, V.M. Instance Relation Graph Guided Source-Free Domain Adaptive Object Detection. In Proceedings of the 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada, 17–24 June 2023; pp. 3520–3530.
17. Pu, B.; Wang, L.; Yang, J.; He, G.; Dong, X.; Li, S.; Tan, Y.; Chen, M.; Jin, Z.; Li, K.; et al. M3-UDA: A New Benchmark for Unsupervised Domain Adaptive Fetal Cardiac Structure Detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle WA, USA, 17–21 June 2024; pp. 11621–11630.
18. Hoyer, L.; Dai, D.; Van Gool, L. Daformer: Improving network architectures and training strategies for domain-adaptive semantic segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 9924–9935.
19. Hoyer, L.; Dai, D.; Van Gool, L. HRDA: Context-aware high-resolution domain-adaptive semantic segmentation. In Proceedings of the Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, 23–27 October 2022; pp. 372–391.
20. Hoyer, L.; Dai, D.; Wang, H.; Van Gool, L. MIC: Masked image consistency for context-enhanced domain adaptation. In Proceedings of the 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada, 18–22 June 2023; pp. 11721–11732.
21. Chen, M.; Zheng, Z.; Yang, Y.; Chua, T.S. Pipa: Pixel-and patch-wise self-supervised learning for domain adaptative semantic segmentation. In Proceedings of the 31st ACM International Conference on Multimedia, Ottawa, ON, Canada, 29 October–3 November 2023; pp. 1905–1914.
22. Zhao, X.; Mithun, N.C.; Rajvanshi, A.; Chiu, H.P.; Samarasekera, S. Unsupervised domain adaptation for semantic segmentation with pseudo label self-refinement. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, Waikoloa, HI, USA, 3–8 January 2024; pp. 2399–2409.
23. Long, M.; Zhu, H.; Wang, J.; Jordan, M.I. Deep transfer learning with joint adaptation networks. In Proceedings of the International Conference on Machine Learning, Sydney, NSW, Australia, 6–11 August 2017; pp. 2208–2217.
24. Sun, B.; Saenko, K. Deep coral: Correlation alignment for deep domain adaptation. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 8–16 October 2016; pp. 443–450.
25. Hoffman, J.; Tzeng, E.; Park, T.; Zhu, J.Y.; Isola, P.; Saenko, K.; Efros, A.; Darrell, T. Cycada: Cycle-consistent adversarial domain adaptation. *Proc. Int. Conf. Mach.* **2018**, *80*, 1989–1998.
26. Vu, T.H.; Jain, H.; Bucher, M.; Cord, M.; Pérez, P. Advent: Adversarial entropy minimization for domain adaptation in semantic segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 2517–2526.
27. Zou, Y.; Yu, Z.; Kumar, B.; Wang, J. Unsupervised domain adaptation for semantic segmentation via class-balanced self-training. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 289–305.
28. Zhang, P.; Zhang, B.; Zhang, T.; Chen, D.; Wang, Y.; Wen, F. Prototypical pseudo label denoising and target structure learning for domain adaptive semantic segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 12414–12424.
29. Shen, Z.; Zhang, M.; Zhao, H.; Yi, S.; Li, H. Efficient attention: Attention with linear complexities. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, Waikoloa, HI, USA, 3–8 January 2021; pp. 3531–3539.
30. Hu, J.; Shen, L.; Sun, G. Squeeze-and-excitation networks. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 7132–7141.
31. Cao, Y.; Xu, J.; Lin, S.; Wei, F.; Hu, H. GCNet: Non-Local Networks Meet Squeeze-Excitation Networks and Beyond. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision Workshop, Seoul, Republic of Korea, 27–28 October 2019; pp. 1971–1980.
32. Woo, S.; Park, J.; Lee, J.Y.; Kweon, I.S. Cbam: Convolutional block attention module. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 3–19.
33. Yu, Q.; Wei, W.; Pan, Z.; He, J.; Wang, S.; Hong, D. GPF-Net: Graph-polarized fusion network for hyperspectral image classification. *IEEE Trans. Geosci. Remote Sens.* **2023**, *61*, 1–22. [[CrossRef](#)]
34. Xie, E.; Wang, W.; Yu, Z.; Anandkumar, A.; Alvarez, J.M.; Luo, P. SegFormer: Simple and efficient design for semantic segmentation with transformers. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 12077–12090.
35. Olsson, V.; Tranheden, W.; Pinto, J.; Svensson, L. Classmix: Segmentation-based data augmentation for semi-supervised learning. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, Waikoloa, HI, USA, 4–8 January 2021; pp. 1369–1378.
36. Tranheden, W.; Olsson, V.; Pinto, J.; Svensson, L. Dacs: Domain adaptation via cross-domain mixed sampling. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, Waikoloa, HI, USA, 3–8 January 2021; pp. 1379–1389.

37. Tarvainen, A.; Valpola, H. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17), Long Beach, CA, USA, 4–9 December 2017; pp. 1195–1204.
38. Chen, L.C.; Zhu, Y.; Papandreou, G.; Schroff, F.; Adam, H. Encoder-decoder with atrous separable convolution for semantic image segmentation. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 801–818.
39. Richter, S.R.; Vineet, V.; Roth, S.; Koltun, V. Playing for data: Ground truth from computer games. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 11–14 October 2016; pp. 102–118.
40. Ros, G.; Sellart, L.; Materzynska, J.; Vazquez, D.; Lopez, A.M. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 3234–3243.
41. Loshchilov, I.; Hutter, F. Decoupled weight decay regularization. *arXiv* **2017**, arXiv:1711.05101.

**Disclaimer/Publisher's Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.