Real-time Domain Adaptation in Semantic Segmentation

Ali Abbasi
Politecnico di Torino
ali.abbasi1@polito.it

Sarvnazsadat Roumianfar
Politecnico di Torino
sarvnazsadat.roumianfar@polito.it

Negin Meyhami
Politecnico di Torino
negin.meyhami@polito.it

Abstract

Several deep learning-based semantic segmentation approaches have been proposed within the last decade. Despite this, semantic segmentation faces several challenges, particularly in terms of efficiency and domain adaptability. Semantic segmentation as a supervised approach requires a large amount of pixel-wise labeled data which needs a huge human effort and besides that, it still faces the domain shift problem. In this regard, we consider real-time Domain Adaptation in Semantic Segmentation by exploiting PIDNet model.

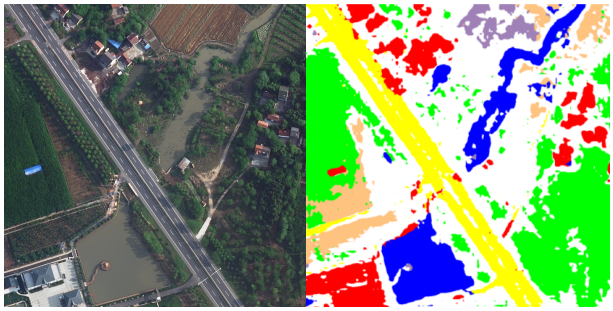


Figure 1. Left: image from LoveDA rural dataset. Right: its semantic label.

1. Introduction

Semantic segmentation [11] is a fundamental task in computer vision that enables image scene understanding at the pixel level, which is crucial to many real-world applications such as autonomous driving. However, deploying semantic segmentation models in real-world scenarios often needs not only high accuracy but also high-speed performance to process visual data in real-time.

Real-time semantic segmentation is crucial in applications where rapid decision-making is essential, such as autonomous vehicles navigating on roads. Achieving this balance between speed and accuracy is challenging, as high-performing models typically require significant computational resources, which can limit their deployment on resource-constrained devices. In addition, there is another problem to tackle which is Domain-shift problem, that can significantly reduce the accuracy of the model.

This paper focuses on Real-time Semantic Segmentation Networks with an emphasis on addressing the challenges related to domain shifts through Domain Adaptation. Domain adaptation is critical in scenarios where models trained on one domain (e.g., urban environments) and when applied to a different domain (e.g., rural areas) the performance drops due to variations in scene structure, textures, and environmental conditions.

To address these challenges, we use PIDNet [20], a state-of-the-art real-time semantic segmentation network based on different benchmarks, and adapt it for use with the LoveDA [18] dataset, which is a benchmark dataset that includes urban and rural scenes. Our approach in this paper includes:

- Implementing PIDNet on LoveDA dataset and evaluating segmentation accuracy, latency, and computational efficiency when trained and tested on domain-specific data without domain shift problem.
- Introducing the domain shift by validating our model on a different dataset with respect to the train data set and measuring the performance drop
- Implement domain adaptation techniques, such as data augmentation, adversarial training, and image-to-image translation, to improve generalization across domains.
- Further addressing the domain shift issue that causes lower results by experimenting with different loss functions aimed for this problem and combining them together.

By combining advanced model architectures with domain adaptation strategies, this project aims to bridge the gap between real-time performance and robust generalization across diverse domains.

The source code is available at https://github.com/aliabbasi2000/Real-time-Domain-Adaptation-in-Semantic-Segmentation.

2. Related Work

2.1. Semantic Segmentation

- **Classical Semantic Segmentation:** Before the advent of deep learning classical semantic segmentation methods relied on hand-crafted features, like edges or blobs that were manually designed and extracted from images, combined with flat classifiers [4] [17]. Flat classifiers are relatively simple models that make predictions based on the extracted features. These classifiers include Boosting, Random Forests, or Support Vector Machines [4]. In addition, techniques like superpixel-based methods [1] [17], Markov Random Fields (MRFs) [3], and Conditional Random Fields (CRFs) [9] were widely used to model spatial relationships between pixels and the semantics of the image as a kind of prior information among adjacent pixels. These methods were designed to leverage spatial relationships and color-based information to segment an image into meaningful regions. In MRFs, the image is modeled as a graph where each pixel is a node, and the edges represent spatial relationships between neighboring pixels. CRFs extended the idea of MRFs by modeling global conditional dependencies instead of just pairwise relationships. CRFs incorporated richer contextual information by using features such as texture, edge responses, and higher-order relationships. Fully connected CRFs further improved segmentation by allowing all pixels in the image to interact with one another. These models were particularly effective in refining coarse segmentations by aligning them with object boundaries. These classical methods were often limited by the expressive power of their features and were not able to fully utilize context information [4]. They also struggled when semantic information was necessary for pixel-wise segmentation, such as with similar objects occluding each other [17]. While these methods are not typically used for segmentation anymore, some graphical models, especially CRFs, are still utilized by state-of-the-art methods as post-processing (refinement) layers to improve semantic segmentation performance [17]. The goal of these CRFs is to smooth noisy segmentation maps by favoring same-label assignments to spatially proximal pixels [9].

- **Real-time Semantic Segmentation:** Real-time semantic segmentation is a specific subfield of semantic segmentation that focuses on achieving high-speed performance while trying to keep it accuracy high. The ability to process images in real-time is critical for applications like autonomous driving, where rapid decision-making is essential. However, achieving this balance is challenging because semantic segmentation models are often computationally intensive.

  Early approaches to semantic segmentation were not optimized for real-time performance due to their high computational demands, like Fully Convolutional Networks (FCNs) [13]. More recent models like U-Net [12] and DeepLab [4] have tried to address this challenge, but they were not able to fully solve the problem. The achievable accuracy is still not usable on real-time systems.

  Recent advancements in real-time semantic segmentation have focused on designing lightweight efficient architectures. Techniques such as encoder-decoder architectures [2], dilated convolutions [5], and spatial pyramid pooling [8] are some of these models that are adapted for speed-optimization.

  PIDNet [20] is a significant leap forward in this domain. It is Specifically designed for real-time applications by effectively balancing the accuracy and speed which causes to achieve state-of-the-art results on benchmark datasets such as Cityscapes and CamVid. We have used this model in our work.

2.2. Domain Adaptation

Domain adaptation is an approach to handle the domain shift problem in semantic segmentation tasks. [22] There are various strategies to handle this challenge:

The First method which is widely used for this problem, is adversarial training techniques [6]. The method consists of two main components; The Generator and the Discriminator. The generator which acts like a feature extractor tries to fool the discriminator whose objective is to distinguish whether the image is from source domain or the target domain. The generator competes against the discriminator and in this competition, the generator aims to learn domain-invariant features. So it learns how to create features that are indistinguishable between the source and target domains.

The second method is image-to-image translation, that transforms the images by converting the style of source domain images to look like target domain images. By making two domain more visually closer, the model is able to adapt better to the target domain. [19].

3. METHOD

3.1. DeepLabV2

DeepLabV2 [4] is a semantic segmentation framework that extends Fully Convolutional Networks (FCNs) by incorporating several innovative techniques to address challenges like multi-scale object representation, boundary refinement, reduced feature resolution, and reduced localization accuracy. The DeepLab system repurposes networks trained on image classification for semantic segmentation

[4]. Below, its components and their contributions to the segmentation pipeline is described.

- **Atrous Convolution**, (also called dilated convolution [4]) used to expand the receptive field without increasing the number of parameters or computational cost. The standard convolution operation at location i in an image x is defined as:

$$y[i] = \sum_k x[i + k] \cdot w[k] \tag{1}$$

where k indexes over the kernel. Atrous convolution introduces a dilation rate r to the kernel, modifying the operation as:

$$y[i] = \sum_k x[i + r \cdot k] \cdot w[k] \tag{2}$$

This allows the network to capture larger context while maintaining the spatial resolution of feature maps.

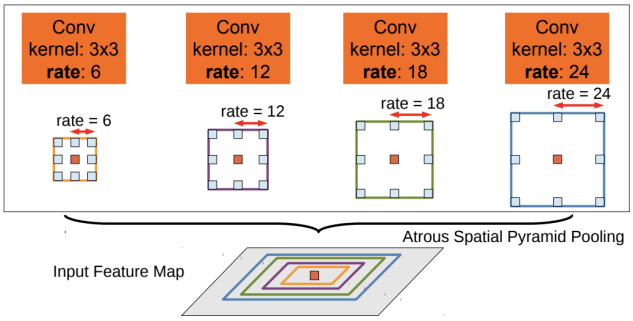


Figure 2. Atrous Spatial Pyramid Pooling (ASPP). To classify the center pixel (orange), ASPP exploits multi-scale features by employing multiple parallel filters with different rates. The effective Field-Of-Views are shown in different colors. The image is taken from [4]

- **Atrous Spatial Pyramid Pooling**, used to further enhance multi-scale feature aggregation [4]. ASPP consists of multiple parallel atrous convolution layers with different dilation rates. Let the output of the i-th atrous convolution in the ASPP module be f_i, and the module's output be y. ASPP computes $y = concat(f_1, f_2, ..., f_n)$ where n is the number of parallel atrous convolutions. The outputs of these layers are concatenated and passed through a 1×1 convolution to produce a fused feature map. ASPP captures both objects and image context at multiple scales [4].

- **Fully Convolutional Backbone**, meaning that DeepLabV2 uses a deep convolutional neural network, such as VGG-16 [14] or ResNet-101 [7], as a backbone pre-trained on image classification tasks. The fully connected layers in them are replaced with atrous convolution layers, allowing the network to produce dense feature maps. This modification ensures that spatial information critical for semantic segmentation is preserved throughout the network.

- **Fully Connected Conditional Random Fields**, used to refine coarse segmentation predictions and improve localization of object boundaries, as a post-processing step. CRFs enforce spatial and boundary consistency in the segmentation output and are applied to the score maps to smooth noisy segmentation maps by favoring same-label assignments to spatially proximal pixels [4]. The energy function for a CRF is defined as:

$$E(x) = \sum_i \psi_u(x_i) + \sum_{i<j} \psi_p(x_i, x_j) \tag{3}$$

where $\psi_u(x_i)$ is the unary potential from the network's predictions and$\psi_p(x_i, x_j)$ is the pairwise potential encouraging spatial smoothness and alignment with image boundaries. The CRF is optimized iteratively using mean-field approximation.

DeepLabV2 is trained using a pixel-wise cross-entropy loss. The loss function is calculated as the sum of cross-entropy terms for each spatial position in the CNN output map, where the loss for a single pixel p is defined as:

$$L_p = -\sum_{c=1}^{C} y_p^c \log(\hat{y}_p^c) \tag{4}$$

where C is the number of classes, y_p^c is the ground truth label for class c, and $\hat{y}_p^c$ is the predicted probability for class c.

3.2. PIDNet

PIDNet [20] is a real-time semantic segmentation network that performs well on the CamVid and CityScapes dataset. It can be directly used for the real-time applications, such as autonomous vehicle and medical imaging. In detail: PIDNet-S presents 78.6% mIoU with speed of 93.2 FPS on Cityscapes test set and 80.1% mIoU with speed of 153.7 FPS on CamVid test set. In addition, PIDNet-L is the most accurate (80.6% mIoU) among all real-time networks for cityscapes. [21]

Its architecture includes of three branches: Proportional (P), Integral (I), and Derivative (D), inspired by the PID controller.

The **P branch** keeps high-resolution details, that helps to get fine details in images, which is important to have accurate segmentation. The **I branch** looks at multi-scale context and gathers information from different image sizes. It helps the network understand larger structures.

The **D branch** aims at boundary detection and captures high frequency features to enhance segmentation on the edges of an object. Figure 3 shows the three-branch network architecture.

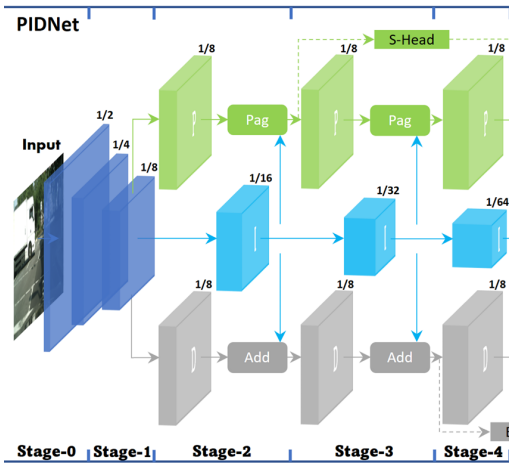


Figure 3. The basic architecture of Network. The image is taken from [20].

The loss function of PIDNet [20] helps to optimize semantic segmentation and boundary detection. It combines different loss components that helps to achieve better performance in real-time semantic segmentation tasks. The first component is semantic loss (l_0) that enhance early segmentation performance.

The second component is weighted binary cross-entropy loss (l_1) to detect boundaries more accurately by focusing on boundary pixels.

The third component is cross-entropy loss (l_2) to minimize the difference between predicted and true class labels. The final component is boundary-awareness cross-entropy loss (l_3) to improve coordination between segmentation and boundary detection. This loss is calculated as:

$$l_3 = -\sum_{i,c} [\mathbf{1} : \mathbf{b_i} > \mathbf{t}] \left(s_{i,c} \log \hat{s}_{i,c}\right) \tag{5}$$

Where t is a predefined threshold, and b_i, $s_{i,c}$, and $\hat{s}_{i,c}$ represent the output of the boundary head, the segmentation ground truth, and the prediction result of the i-th pixel for class c, respectively.

The final loss function for PIDNet [20] is a weighted sum of all these components.

$$\text{Loss} = \lambda_0 L_0 + \lambda_1 L_1 + \lambda_2 L_2 + \lambda_3 L_3 \tag{6}$$

Where the parameters for these loss weights are set as $\lambda_0 = 0.4$, $\lambda_1 = 20$, $\lambda_2 = 1$, $\lambda_3 = 1$, and the threshold $t = 0.8$.

4. Experiment

4.1. Datasets

The dataset used in our experiments is LoveDA [18], a remote sensing land-cover dataset, that has 5987 images for urban and rural areas. The dataset includes seven land cover classes: background, building, road, water, barren, forest, and agriculture. For the purpose of our experiments, LoveDA-rural was taken as the target domain, while the source domain is LoveDA-urban.

4.2. Implementation Details

In our implementation, we begin by training the DeepLabV2 segmentation network with an R101 backbone on the LoveDA-rural dataset [18] to establish the upper bound performance. We used the PIDNet-S network, pre-trained on ImageNet [20]. The following configurations were applied during our experiment, if not specified differently:

- Epochs: 20
- Batch size: 8
- Image resize: (512, 512)
- Normalization with ImageNet mean ([0.485, 0.456, 0.406]) and standard deviation ([0.229, 0.224, 0.225])
- Input Channel num: 3
- Kernel size: 3*3

4.3. DeepLAB

We start by using DeepLabV2 as our classic semantic segmentation model, using ResNet-101 [7] as backbone. The model is pre-trained on ImageNet dataset and initialized with those weights. It is configured for 7-class segmentation, corresponding to the LoveDA [18] dataset's class structure. In this case we are using the Rural subset of LoveDA dataset, resizing to 384×384 pixels. The training procedure followed these specifications:

- Batch size: 4 for training, 2 for validation
- Optimizer: SGD with momentum
- Learning rate: 0.01
- Momentum: 0.9
- Weight decay: 0.0005
- Loss function: Cross-Entropy Loss

The goal of this step is to compare the performance of classic semantic segmentation networks with real-time semantic segmentation networks, without considering domain shift. We obtained 19% accuracy on rural dataset with DeepLabV2 which is a classic segmentation network. This low accuracy is due to DeepLab being weak at handling small objects because it does not use detailed low-level features, also for obtaining high accuracy it needs a large amount of data.

4.4. PIDNet

We trained the PIDNet-S model on the LoveDA-urban dataset [18]. The model was trained using cross-entropy loss with the Adam optimizer, a learning rate of 0.0001, and a weight decay of $1e{-}4$.In addition, to enhance generalization, we applied normalization. By validating it on the same trained data (without domain shift effect), the model achieved mIoU of 37.7% as is shown in Figure 4.

The advantages of PIDNet over DeepLabV2 is in higher accuracy and speed. Our results shows that PIDNet achieves higher validation mIoU, while also having lower latency. In terms of Inference latency, PIDNet performs segmentation on a single image in 10.7 ms, whereas the DeepLabV2 has the latency of 16.9 ms on the same image. In addition, Comparing the validating mIoU of PIDNet and DeepLabV2, demonstrates that the PIDNet has better accuracy in performing semantic segmentation. Overall, these results demonstrates that the PIDNet is more suitable for real-time applications.

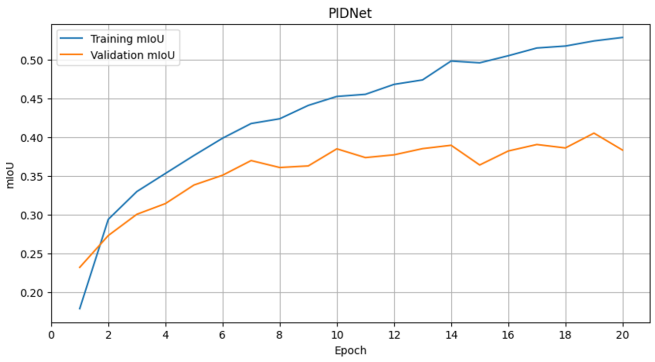


Figure 4. mIoU learning curve Without Domain Shift. Training and validation mIoU curves curves show steady improvement, indicating good generalization.

4.5. Domain Shift

By changing the validation domain from the urban dataset to the rural dataset, we introduce a domain shift effect in the validation process and aim to assess the model's ability to generalize and perform well on unseen data. Figure 5 illustrates the gap between training mIoU and validation mIoU caused by this domain shift. During early training (until epoch 9), the validation mIoU increases along with the training mIoU. This suggests that the model is learning useful pattern from data. However, after epoch 9, the Validation mIoU remains fluctuating while the training mIoU continues to rise. This gap clearly indicates that, after the 9th epoch, the model no longer generalizes well to unseen images and instead memorizes the training data rather than learning features which is a clear sign of overfitting.

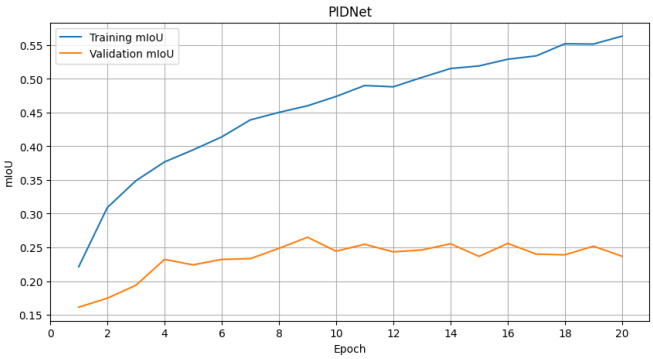


Figure 5. mIoU learning curve With Domain Shift. The training mIoU continues to increase, while the validation mIoU fluctuates after epoch 9, indicating potential overfitting and lack of generalizing to the new domain.

4.6. Data Augmentation

In the data augmentation part, we tested a range of augmentation techniques, including horizontal and vertical flips, random rotations, Gaussian blurring, brightness/contrast adjustments, and multiplicative noise. Each augmentation was applied during training with a probability of 50%. Finally, the entire block of augmentations were applied with a probability of 75% to avoid overfitting. These augmentations were selected to simulate real-world variations, such as changes in lighting, and noise, and trying to improve model's ability to generalize unseen data. Augmentation was performed at the training only, and evaluations on the validation set were done without improvement to get the actual measure of the model's performance. Out of all the combinations we tried, the best result was a combination of all of them resulted in a validation mIoU of 27.3%, as it is seen in Table 1, showing the effectiveness of the improvement in domain shift. Figure 6 shows the performance of the model on both the training and validation sets across each epoch.

4.7. Domain Adaptation

A) **Adversarial approach:** In order to tackle domain shift we use adversarial framework described here [16]. Adversarial approach algorithm consists of two modules: a

Augmentation	mIoU%	Road	Building	Water	Barren	Forest	Agriculture	Background
Domain Shift	23.6	19	30	24	8	15	20	48
Aug1:RandomRotate90	25.13	20.63	30.70	30.12	6.36	10.90	27.62	49.61
Aug2: GaussianBlur	24.72	22.73	32.00	32.39	9.00	7.31	19.65	50.00
Aug1 + Aug2	25.13	20.63	30.70	30.12	6.36	10.90	27.62	49.61
Aug4: Aug3 + HFlip + VFlip	26.69	21.66	39.15	32.69	9.21	8.76	25.90	49.45
Aug4 + RandomBrightnessContrast	27.36	24.15	32.71	25.59	8.53	13.48	36.21	50.85

Table 1. Class-wise IoU for Different Augmentations and Domain Shift

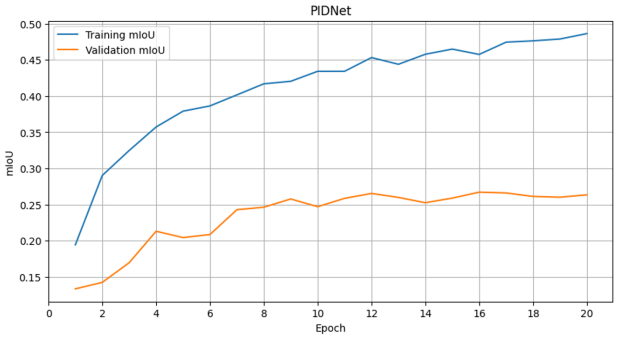


Figure 6. mIoU learning curve With Augmentation. Augmentation improves the result by reducing the gap between training and validation mIoU and helps the model to generalize better

segmentation network G and the discriminator D, where G predicts pixel-wise semantic labels, and D distinguish whether the segmentation output is from the source or target domain.

An adversarial loss on the target prediction allows the network to backpropagate gradients from D to G, encouraging G to generate similar segmentation distributions in the target domain to the source prediction.

The formula of the adaptation task contains two loss functions from both modules:

$$\mathcal{L}(I_s, I_t) = \mathcal{L}_{\text{seg}}(I_s) + \lambda_{\text{adv}}\mathcal{L}_{\text{adv}}(I_t) \tag{7}$$

where $\mathcal{L}_{\text{seg}}$ is the cross-entropy loss using ground truth annotations in the source domain, $\mathcal{L}_{\text{adv}}$ is the adversarial loss that adapts predicted segmentations of target images to the distribution of source predictions, and λ_{adv} is the weight used to balance the two losses. Segmentation loss is defined as:

$$\mathcal{L}_{\text{seg}}(I_s) = -\sum_{h,w}\sum_{c\in C} Y_s^{(h,w,c)} \log(P_s^{(h,w,c)}), \tag{8}$$

where Y_s is the ground truth annotations for source images and $P_s = G(I_s)$ is the segmentation output. After that, the images in the target domain pass through G to obtain the prediction $P_t = G(I_t)$. The adversarial loss $\mathcal{L}_{\text{adv}}$ in (7) is defined as:

$$\mathcal{L}_{\text{adv}}(I_t) = \sum_{h,w} \log(D(P_t)^{(h,w,1)}). \tag{9}$$

This loss is designed to train the segmentation network and fool the discriminator by maximizing the probability of the target prediction being considered as the source prediction.

The discriminator is trained by giving it as input the segmentation softmax output $P = G(I)$, and using a cross-entropy loss $\mathcal{L}_d$ for the two classes (i.e., source and target). The loss can be written as:

$$\mathcal{L}_d(P) = -\sum_{h,w}(1-z)\log(D(P)^{(h,w,0)}) + z\log(D(P)^{(h,w,1)}), \tag{10}$$

where $z = 0$ if the sample is drawn from the target domain, and $z = 1$ for the sample from the source domain.

We use PIDNet as segmentation model (G) to train the discriminator, and the Adam optimizer with learning rate of $1e{-}4$, and the momentum of Adam optimizer is set as 0.9 and 0.99. for the discriminator we defined a fully-convolutional network with 5 convolutional layers, kernel size 4×4, and stride 2. Considering these configurations we got 28.07% of mIoU on 20th epoch. This result is near to the augmentation result indicating that this adversarial approach cannot offer improvement to our model.

One of the reasons that the adversarial approach struggles with improving the domain shift can be the fact that Urban and rural images have very different textures, color distributions, object layouts. Adversarial learning assumes that the target domain can align well with the source domain. In our case the domain gap is too large so the discriminator and generator might struggle to learn meaningful adaptations.

B) **Image-to-image translation approach:** In the image-to-image domain adaptation step, we implemented Domain Adaptation via Cross-domain Mixed Sampling [19]. We optimized the model using Stochastic Gradient Descent with a learning rate of 0.01, momentum set to 0.9, and a weight decay of $1e{-}4$ to avoid overfitting. The ReduceLROnPlateau scheduler is used to change the learning rate dynamically, that reduces the learning rate when a plateau in the training loss is found. If the training loss did not improve for 3 epochs in-row, the learning rate is reduced.

In this phase for generating pseudo-labels using the model trained on source domain, we performed a forward pass through the model, achieving class probabilities for

target-domain images. To convert the model's raw outputs to class probabilities, we used softmax function, which allows us to select the most confident predictions (with a threshold of 0.7) as pseudo-labels for the target domain. Next, we mixed source-domain images and labels with target-domain pseudo-labeled images using the ClassMix technique [19]. The results did not have significant improvement in validation performance, with mean IoU fluctuating around initial values and validation loss is still high. While some improvements were in first epochs, the model failed to generalize effectively. The reason could be imbalanced class performance, which may result in insufficient pseudo-labeled samples for weaker classes like barren. Another factor might be the domain shift in LoveDA [18], which is beyond color or texture differences, it includes structure variations and differences in layouts, and missing classes that image-to-image translation methods might not be able to effectively help the domain shift problem.

4.8. Extension

To further enhance the negative impact of domain shift we explored and implemented various techniques. First we started by changing the loss function and using the ones that is better suited for the issue of class imbalance, which is the main problem in LoveDA dataset [18]. Then we combined them with different weights to obtain better results. Detailed implementation and used loss functions are expressed below:

- **Focal Loss:** The focal loss [10] is a modification of the standard cross-entropy and addresses the problem of class imbalance by reducing the contribution of easy examples to the loss and focusing training on the hard examples by introducing a modulating factor. The focal loss is based on the cross-entropy (CE) loss for binary classification, which is defined as $-log(pt)$ where pt is the model's estimated probability for the correct class. The focal loss adds a modulating factor $(1-pt)^{\gamma}$ to the cross-entropy loss. The focal loss is defined as:

$$FL(pt) = -(1-pt)^{\gamma} * log(pt) \tag{11}$$

  This factor down-weights the loss assigned to well-classified examples (where pt is close to 1), and the loss is unaffected when an example is misclassified (pt is small). The γ parameter is a tunable focusing parameter that controls the rate at which easy examples are down-weighted. A higher γ value means that a classified example with high confidence will have little to no influence on the loss [10]. In our experiments we started by using the default value of 2 for γ, and obtained results similar to using cross-entropy loss, but gaining better results particularly on road, water and agriculture classes. We then raise γ to 4 but the results did not improve and we encountered fluctuations in the values of loss and mIoU.

- **Adaptive Focal Loss:** Adaptive Focal Loss [10] is a modification of the Focal Loss that solves the imbalanced class problem. It works by adding a modulating factor to the standard cross-entropy loss that reduces the weighting of the loss assigned to correctly classified instances. This modification allows the model to target training at a limited range of more difficult examples and prevents the large number of simple negatives from dominating the detector in training. Adaptive Focal Loss uses an adjustable focusing parameter, γ, that controls the rate at which easy examples are down-weighted. The formula for Adaptive Focal Loss is:

$$L_{\text{focal}} = -\sum_{p=1}^{N} \alpha_t (1-\hat{y}_p^{y_p})^{\gamma} \log(\hat{y}_p^{y_p}) \tag{12}$$

  where $\hat{y}_p^{y_p}$ is the predicted probability of the correct class for pixel p, and α_t is the adaptive weight for class t, which increases for more difficult examples.

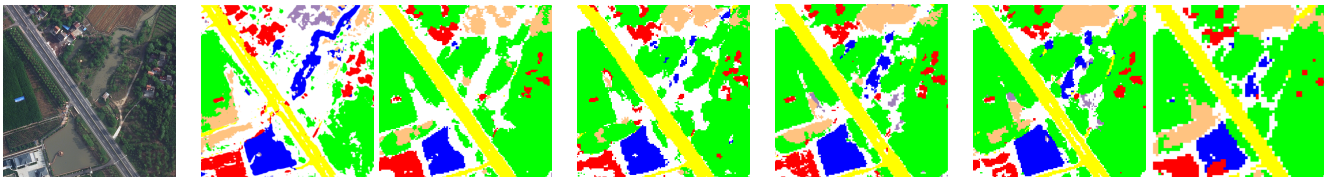
- **Dice Loss:** The Dice loss, also referred to as the Dice score coefficient (DSC) [15], measures the similarity between two sets, and it is calculated by comparing the overlap between the predicted segmentation and the ground truth segmentation. It is more robust to class imbalance compared to the cross-entropy loss and is a suitable loss function for segmentation problems that have an unbalanced ratio of foreground to background. By using dice loss in our model, we obtained slightly lower overall results, but it caused improvements in predicting building and barren classes.

- **Combined Loss:** While different losses like Focal Loss [10] helps by giving hard-to-classify samples more weight, it does not specifically balance the effect of rare classes. To reduce the imbalance between classes, we calculated static class weights from class distribution in the training dataset, and added them to the Cross-entropy Loss, making sure that underrepresented classes like barren, that have less pixels in the dataset, will receive more focus during training.

  By combining the Cross-entropy loss with Adaptive Focal Loss and Dice Loss, we managed to handle different types of segmentation challenges. The Adaptive Focal Loss focuses on hard-to-classify examples, the Dice Loss [15] optimizes the overlap between predicted and true class regions. Also the static weights for the classes make sure that the less represented classes get enough consideration in the training. With

Loss Function	mIoU%	Road	Building	Water	Barren	Forest	Agriculture	Background
Cross-entropy loss	23.6	19	30	24	8	15	20	48
Focal loss $\gamma = 2$	23.4	21.1	27.7	26.8	4.6	8.3	27.6	47.9
Focal loss $\gamma = 4$	21	23.5	25.8	36.1	10.2	9.6	38.2	38.4
Dice loss	21.3	18.9	32.3	25.5	11.6	12	1.6	47.2
Combined losses	29.9	24.6	29.2	45.1	10.5	16	33.4	50.5

Table 2. Class-wise IoU for Different Loss Functions



(a) Image (b) DeepLabV2 (c) PIDNet (d) Domain Shift (e) Augmentation (f) Adversarial (g) CombinedLoss

Figure 7. Comparison of predictions using different methods.

this method, the model is able to accurately segment regions, and provide better performance for different classes. article amsmath

The combined loss function is defined as:

$$L_{\text{combined}} = \lambda_{\text{ce}} L_{\text{ce}} + \lambda_{\text{focal}} L_{\text{focal}} + \lambda_{\text{dice}} L_{\text{dice}} \quad (13)$$

Where: λ_{ce}, λ_{focal}, and λ_{dice} are the weighting factors for each individual loss.

5. Additional Consideration

One of the most effective solutions against domain shift problem is to use style transfer based preprocessing before training. This method consists of adapting the visuals of the source domain photos to those of the target domain using a modification of a CycleGAN [23] model. This approach aids in reducing the domain shift and facilitating model generalization by transforming the images of the source and target domains to match the visual features of both domain, in this case the color, texture, and lighting.

The strength of CycleGAN for the task at hand is that it can do unpaired image-to-image translation, which means it can learn to translate images from one domain to another even if there are no corresponding images in the other domain. This property is especially useful in cases when it is hard to obtain image pairs that are matched. In the same way style transfer preprocessing has been effectively applied in areas like autonomous driving and remote sensing where the model might be working in different environmental conditions urban/rural which could drastically change model performance metrics.

6. Conclusion

In conclusion, our project aimed at real-time domain adaptation in semantic segmentation using PIDNet [20], a state-of-the-art real-time segmentation model. We started with the analysis of classic and modern segmentation networks and comparing DeepLabV2 [4] and PIDNet. Our experiments shows that PIDNet performs better than DeepLabV2 in both accuracy and inference speed. Our study emphasizes the effect of domain shift by validating our models on a different domain. The results shows that the model faces some challenges in generalizing, with significant decrease in mIoU when shifting from urban to rural environments. We experimented with various domain adaptation strategies, including data augmentation, adversarial training, and image-to-image translation [19]. Among all, data augmentation had more improvement in generalization, while image-to-image translation techniques were not effective in reducing the domain gap. Finally, we experimented with different loss functions to address class imbalance challenge, and we were able to improve segmentation performance for most of the classes. By combining Cross-Entropy Loss with static class weights, Adaptive Focal Loss [10], and Dice Loss [15], we managed to balance class representation and improve segmentation results.

References

[1] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2274–2282, 2012. 2

[2] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation, 2016. 2

[3] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1222–1239, 2001. 2

[4] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Deeplab: Semantic im-

age segmentation with deep convolutional nets, atrous convolution, and fully connected crfs, 2017. 2, 3, 8

[5] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):834–848, 2018. 2

[6] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks, 2016. 2

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. 3, 4

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9):1904–1916, 2015. 2

[9] Philipp Krähenbühl and Vladlen Koltun. Efficient inference in fully connected crfs with gaussian edge potentials, 2012. 2

[10] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection, 2018. 7, 8

[11] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3431–3440, 2015. 1

[12] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi, editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pages 234–241, Cham, 2015. Springer International Publishing. 2

[13] Evan Shelhamer, Jonathan Long, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(4):640–651, 2017. 2

[14] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2015. 3

[15] Carole H. Sudre, Wenqi Li, Tom Vercauteren, Sebastien Ourselin, and M. Jorge Cardoso. *Generalised Dice Overlap as a Deep Learning Loss Function for Highly Unbalanced Segmentations*, page 240–248. Springer International Publishing, 2017. 7, 8

[16] Yi-Hsuan Tsai, Wei-Chih Hung, Samuel Schulter, Kihyuk Sohn, Ming-Hsuan Yang, and Manmohan Chandraker. Learning to adapt structured output space for semantic segmentation. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7472–7481, 2018. 5

[17] Irem Ulku and Erdem Akag°nd°z. A survey on deep learning-based architectures for semantic segmentation on 2d images. *Applied Artificial Intelligence*, 36(1):2032924, 2022. 2

[18] Junjue Wang, Zhuo Zheng, Ailong Ma, Xiaoyan Lu, and Yanfei Zhong. Loveda: A remote sensing land-cover dataset for domain adaptive semantic segmentation, 2022. 1, 4, 5, 7

[19] Juliano Pinto Lennart Svensson Wilhelm Tranheden, Viktor Olsson. Dacs: Domain adaptation via cross-domain mixed sampling. *ArXiv*, 2007.08702, 2020. 2, 6, 7, 8

[20] Jiacong Xu, Zixiang Xiong, and Shankar P. Bhattacharyya. Pidnet: A real-time semantic segmentation network inspired by pid controllers. *ArXiv*, 2206.02066v3, 2023. 1, 2, 3, 4, 8

[21] XuJiacong. Pidnet, 2022. 3

[22] Samuel Schulter Kihyuk Sohn Ming-Hsuan Yang Manmohan Chandraker Yi-Hsuan Tsai, Wei-Chih Hung. Learning to adapt structured output space for semantic segmentation. *ArXiv*, 1802.10349, 2018. 2

[23] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks, 2020. 8