Real-time Domain Adaptation in Semantic Segmentation

Federico Baldi s349417 | Davide Marzano s344814 | Ilaria Parodi s339184 | Lorenzo Spiriti s346950

Abstract

In this report, we address the challenges of real-time domain adaptation in semantic segmentation, with a focus on the domain shift between synthetic and real-world images. We begin by training two widely adopted architectures, DeepLabV2 and BiSeNet, on the Cityscapes dataset to establish a baseline, experimenting with both the standard cross-entropy loss and the median frequency weighted cross-entropy loss. Next, we analyze the generalization gap by training BiSeNet on the synthetic GTA5 dataset and evaluating its performance on the Cityscapes dataset. To reduce the domain gap, we explore commonly used data augmentation techniques, such as color jittering, random cropping and horizontal flipping, identifying the most effective combinations. In addition, we implement adversarial domain adaptation strategies to improve cross-domain performance. Finally, we investigate the impact of various segmentation loss functions, including focal loss, dice loss and their combination, evaluating how their integration with adversarial training and data augmentation techniques influences the model's generalization capability across domains. The code is available at https://github.com/fedeCode00/Sem_Seg_25.git.

1. Introduction

Semantic Segmentation is a fundamental computer vision task that assigns a class label to each pixel in an image, enabling fine-grained scene understanding. It plays a crucial role in several real-world applications, including autonomous driving, pedestrian detection and therapy planning. While deep learning models can achieve high accuracy when trained and tested within the same domain, maintaining a balance between accuracy and efficiency remains a key challenge for real-world deployment.

Training accurate segmentation models requires high-quality pixel-level annotations, which are expensive and time-consuming to obtain. To address this limitation, synthetic datasets have become increasingly popular due to their scalability and easier generation process. However, models trained on synthetic data often experience a significant drop in performance when evaluated on real-world images. This issue, known as domain shift, is caused by differences in appearance, textures, lighting and other visual discrepancies between synthetic and real domains. Addressing this domain gap is essential for the effective deployment of semantic segmentation systems in practice.

In this project, we focus on real-time semantic segmentation, aiming to achieve a balance between segmentation accuracy and inference speed. To establish a baseline, we trained DeepLabV2 and BiSeNet on the real-world Cityscapes dataset. We then assessed the impact of domain shift by training BiSeNet on the synthetic GTA5 dataset and evaluating it on Cityscapes.

To mitigate the resulting performance degradation, we applied data augmentation strategies and explored adversarial domain adaptation techniques to improve model generalization. Finally, we investigated the role of different segmentation loss functions in the domain adaptation setting. Specifically, we compared the performance of a pixel-level loss (Focal loss), a region-level loss (Dice loss) and a combination of both. The goal of this project is to evaluate how different strategies, such as architectural choices, data augmentation, loss functions and domain adaptation methods, can reduce the domain gap while maintaining real-time inference capabilities.

2. Methodology

2.1. Datasets

In this work, we used two different datasets: Cityscapes and GTA5.

Cityscapes. The Cityscapes dataset is a real-world benchmark for semantic segmentation of urban street scenes. It contains high resolution images (2048x1024) from 50 different European cities. The dataset is composed of 20,000 coarse-annotated data, that are typically used for pretraining and 5,000 fine-annotated images with pixel-level accurate segmentation masks. It features 19 semantic classes used for evaluation, the classes are divided into 7 broader categories: ground, construction, object, nature, sky, human and

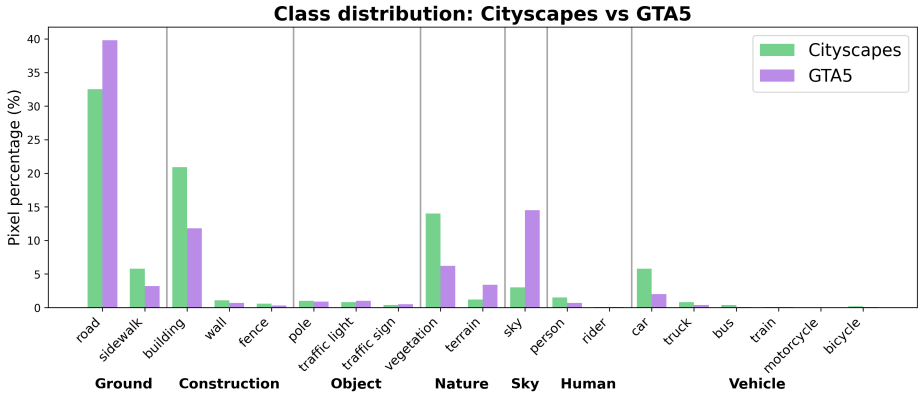


Figure 1. Comparison of pixel percentage distribution for each semantic class in Cityscapes and GTA5 datasets.

vehicle. Regions marked with the void label, which correspond to uncertain or irrelevant areas, are excluded from evaluation. [3]
In our experiments, we used a subset of the fine-annotated split, consisting of 2,072 images, selected from specific cities.
GTA5. The GTA5 dataset is a synthetic dataset created from the Grand Theft Auto V video game. It includes 24,966 high-resolution images (1914×1052), each automatically annotated with pixel-level labels. All the images represent virtual urban scenes from a car's perspective. There are 19 semantic classes compatible with the ones of Cityscapes dataset, that makes it suitable for domain adaptation tasks. [6] For our analysis, we used a subset of 2,500 images from the full GTA5 dataset.

From the analysis of the pixel-wise distribution of semantic classes in both Cityscapes and GTA5 (Figure 1), significant differences in class frequency are observable. For instance, the 'road' and 'building' classes show high pixel percentages in both datasets, while others like 'train', 'motorcycle' and 'bicycle' are considerably less represented. Moreover, GTA5 has a higher percentage of 'road' and 'sky' pixels compared to Cityscapes, while Cityscapes exhibits a higher proportion of 'building' and 'vegetation' pixels. The overall class distribution is highly imbalanced in both datasets, with a few dominant classes occupying the majority of pixels and several others appearing only sparsely. This imbalance can bias model training, leading to poor performance on underrepresented classes incrementing the domain shift issue.

2.2. Network Architectures

For our project we employed two distinct network architectures: DeepLabV2 as a representative classical semantic segmentation model and BiSeNet as a real-time solution.

DeepLabV2. DeepLabV2 [2] is a semantic segmentation framework designed to improve the spatial accuracy and multi-scale understanding of deep convolutional models. Its architecture introduces three key components:
Atrous Convolution: also known as dilated convolution, is a mechanism that enables dense feature extraction by inserting holes in the convolutional kernels. This allows DeepLab to increase the receptive field of the network without reducing the spatial resolution or increasing the number of parameters, which is crucial for preserving fine spatial details.
Atrous Spatial Pyramid Pooling (ASPP): employs multiple parallel atrous convolution layers with varying dilation rates to capture multi-scale context and segment objects of varying sizes by aggregating features from different receptive fields.
Fully Connected Conditional Random Fields (CRFs): as a post-processing step, DeepLab integrates a dense CRF model to refine the raw segmentation output. This enhances the delineation of object boundaries by modeling long-range dependencies between pixels, effectively correcting the coarse predictions made by the DCNN.
Overall, DeepLab strikes a balance between segmentation accuracy and efficiency, achieving state-of-the-art performance on benchmarks such as PASCAL VOC 2012, thanks to its hybrid design that combines dense feature extraction, multi-scale representation and structured prediction.

BiSeNet. BiSeNet [8] is a real-time semantic segmentation architecture designed to decouple the modeling of spatial detail and semantic context through a two-branch structure.
Spatial Path (SP): a shallow, wide branch composed of three convolutional layers with stride 2, designed to preserve high-resolution spatial information at 1/8 of the input resolution. This path maintains fine-grained spatial cues often lost in traditional fast models.

Context Path (CP): a deep path designed for efficient semantic context extraction. It uses a lightweight backbone (e.g., Xception39) with aggressive downsampling and incorporates a global average pooling layer at the end to ensure a large receptive field. Each stage of this path is enhanced by *Attention Refinement Modules (ARM)*, which capture global contextual information through global average pooling and compute an attention vector to reweight the feature maps, refining semantic features.
Feature Fusion Module (FFM): merges Spatial Path and Context Path outputs through concatenation and batch normalization, followed by a lightweight attention mechanism. After global average pooling, the module computes importance weights that reweight the fused features based on the global context. This allows the model to effectively combine low-level spatial details with high-level semantic information. This architecture achieves a strong trade-off between accuracy and speed.

2.3. Evaluation Metrics

To evaluate the performance of the developed models, we used the following metrics.

Mean Intersection over Union (mIoU) (%): measures the average overlap between the predicted and ground truth segmentation masks across all semantic classes. It is computed as the mean of the Intersection over Union (IoU) scores calculated individually for each class. It is defined as: [4]

$$\text{mIoU} = \frac{1}{k+1} \sum_{i=0}^{k} \frac{Y_{ii}}{\sum_{j=0}^{k} Y_{ij} + \sum_{j=0}^{k} Y_{ji} - Y_{ii}}$$

where:

- Y_{ii}: number of pixels correctly predicted as class i (True Positives),
- Y_{ij}: number of pixels belonging to class i but predicted as class j (False Negatives),
- Y_{ji}: number of pixels belonging to class j but predicted as class i (False Positives),
- $k+1$: total number of classes.

Latency: Measures the average inference time per image. It is a practical indicator of how suitable a model is for real-time applications:

$$\text{Latency} = \frac{\text{Total inference time}}{\text{Number of samples}}$$

Floating Point Operations (FLOPs): represent the total number of floating-point operations required to process a single input image during inference.A lower FLOP count indicates a more efficient architecture in terms of inference time and energy consumption, which is critical for deployment on edge devices or real-time systems.

Number of parameters: denotes the total number of trainable weights in the model, including both convolutional kernels and bias terms. While a larger parameter count may enhance the model's ability to capture complex spatial patterns, it also increases memory usage and the risk of overfitting. This metric is particularly important when considering model scalability and storage constraints.

Frames Per Second (FPS): is a measure of how many images per second the model is able to process in real time.

$$\text{FPS} = \frac{1}{\text{latency}}$$

2.4. Implementation Details and Training Settings

Different batch sizes were used throughout the experiments according to memory constraints. Initially, a batch size of 16 was adopted, which was reduced to 8 for adversarial training and further to 4 when experimenting with alternative loss functions. Input images were resized to 1024×512 pixels for Cityscapes and 1280x720 pixels for GTA5 to balance spatial resolution and computational efficiency. All models were trained for 50 epochs and the best checkpoint was selected based on the validation mIoU.
To optimize the segmentation networks, Stochastic Gradient Descent (SGD) was used with a momentum of 0.9. Weight decay was set to 5×10^{-4} for DeepLabV2 and 1×10^{-4} for BiSeNet.
The initial learning rate also varied across models: 1×10^{-3} for DeepLabV2 and 2.5×10^{-2} for BiSeNet. In both cases, a polynomial learning rate decay schedule was applied with power 0.9, as defined by the formula:

$$\eta = \eta_0 \cdot \left(1 - \frac{\text{iter}}{\text{max\_iter}}\right)^{0.9}$$

For adversarial training, the discriminator was optimized using the Adam optimizer with a learning rate of 1×10^{-4}, $\beta_1 = 0.9$ and $\beta_2 = 0.99$. The same polynomial decay schedule was applied to the segmentation network. These settings were adopted following Tsai et al. (Learning to Adapt Structured Output Space for Semantic Segmentation, CVPR 2018), which forms the basis of our approach. [7]
The adversarial loss term was weighted with a factor $\lambda_{\text{adv}} = 1 \times 10^{-3}$, as empirically shown in the same work to ensure stable adaptation in the output space.

3. Experiments and Results

3.1. Baseline on Target Domain

To establish a performance baseline and compare models in the absence of domain shift, we trained DeepLabV2 and BiSeNet on the Cityscapes dataset for 50 epochs.

DeepLabV2 was trained with a ResNet-101 backbone, while BiSeNet was trained with ResNet-18, both pre-trained on ImageNet. The results reported in Tab. 1 show the performance in the validation set. Figure 2 shows a qualitative comparison between the input image, the ground truth and the segmentation results obtained using DeepLabV2 and BiSeNet. Both models achieve comparable mIoU scores (55.01% for DeepLabV2 and 54.55% for BiSeNet), confirming their effectiveness on the target domain. However, BiSeNet significantly outperforms DeepLabV2 in terms of inference speed, with lower latency, computational cost and higher frame rate, making it a more suitable candidate for real-time applications. These results were obtained using standard cross-entropy loss. We also experimented with median frequency balancing loss, which consistently led to lower performance across both models. In particular, we observed mIoU scores of 53.71% for DeepLabV2 and 53.49% for BiSeNet. Median frequency balancing is a class weighting scheme that aims to compensate for class imbalance by assigning higher weights to underrepresented classes. For each class, the weight is computed as the ratio between the median of class frequencies and the class's own frequency. This encourages the model to focus more on rare classes during training. However, in our case, the resulting weights may have overcompensated the imbalance, leading to suboptimal optimization and reduced overall performance.

Model	mIoU (%)	Latency	FLOPs	Params	FPS
DeepLabV2	55.01	14.24 ms	375.7 G	43.90 M	70.42
BiSeNet	54.55	4.11 ms	25.78 G	12.58 M	243.74

Table 1. Comparison of DeepLabV2 and BiSeNet on Cityscapes.

(a) Input Image (b) Ground Truth

(c) Prediction with DeepLabv2 (d) Prediction with BiSeNet

Figure 2. Qualitative comparison between input, ground truth and segmentation results from DeepLabv2 and BiSeNet.

3.2. Domain Shift Evaluation

To evaluate the impact of domain shift in semantic segmentation we trained BiSeNet on the synthetic GTA5 dataset and evaluated its performance on the validation split of the real-world Cityscapes dataset. A common challenge in semantic segmentation is the need for large amounts of pixel-wise annotated data, which is costly and time-consuming to collect. Synthetic datasets such as GTA5 provide automatically annotated images generated in virtual environments, but the models trained on synthetic data often perform poorly when tested on real-world images due to the domain gap. We obtained a mIoU of 14.57%, significantly lower than the value achieved when training on Cityscapes without domain shift.

3.3. Data Augmentation

To mitigate the performance degradation caused by domain shift, we applied data augmentation techniques to increase data diversity and reduce the visual gap between synthetic and real-world images by introducing controlled variability and simulating more realistic visual conditions. Specifically, we applied the following transformations to GTA5 images, each applied with a probability of 0.5:

- Horizontal flipping: encourages spatial invariance by exposing the model to mirrored views.
- Cropping: allows the model to focus on different regions within an image, promoting spatial robustness.
- Color jittering: simulates varying lighting conditions, enabling the model to better handle color variations.

These techniques are widely adopted in semantic segmentation to improve robustness and generalization [5].
Initially, each augmentation technique was tested individually over 10 epochs to assess its impact on training dynamics. We obtained the following mIoU scores: 20.76% for color jittering, 12.36% for random cropping and 13.56% for horizontal flipping. Based on this preliminary analysis, we conducted further experiments by training for 50 epochs the model with each of the following configurations: only color jittering, only horizontal flipping, a combination of both and the combination of jittering, flipping and cropping.

Table 2 reports the per-class IoU on the Cityscapes validation set for BiSeNet trained on GTA5 with the different data augmentations configurations. The baseline model achieves an overall mIoU of 14.57%, with strong performance on static classes like 'building' and 'vegetation'. Applying horizontal flip slightly improved segmentation for some classes, in particular an increase of nearly 10% of IoU can be observed for 'person' class. Color jittering alone achieves the best mIoU (19.31%), enhancing performance across most classes, especially 'road', 'sidewalk' and 'vegetation'. However, combining jittering and flipping did not

Model	mIoU(%)	Road	Sidewalk	Building	Wall	Fence	Pole	Light	Sign	Vegetation	Terrain	Sky	Person	Rider	Car	Truck	Bus	Train	Motorcycle	Bicycle
BiSeNet baseline	14.57	11.64	2.77	55.87	2.58	3.08	5.85	7.35	6.51	59.03	3.10	49.76	24.02	0.00	38.36	0.86	5.79	0.00	0.41	0.00
+ Flip	15.08	16.01	4.89	52.95	1.83	0.83	5.47	10.58	5.30	56.52	2.23	49.80	33.77	0.00	40.59	2.46	2.60	0.00	0.75	0.00
+ Jitter	19.31	31.14	20.57	53.06	4.47	1.44	12.85	14.76	6.30	70.81	9.13	58.02	37.78	0.00	37.92	1.38	1.78	0.00	5.42	0.00
+ Jitter + Flip	18.70	38.20	12.83	59.53	6.83	9.38	14.14	11.21	5.85	63.01	8.80	64.73	31.03	0.06	26.43	2.16	0.23	0.00	0.98	0.00
+ Jitter + Flip + Crop	14.63	28.61	9.59	64.41	5.82	1.95	2.46	1.02	2.75	49.09	4.43	57.48	26.75	0.00	19.05	4.57	0.00	0.00	0.00	0.00

Table 2. mIoU(%) and per-class IoU (%) on Cityscapes for BiSeNet trained on GTA5 with various data augmentations.

further improve performance, suggesting that the benefits of individual augmentations may not be additive and could even interfere with each other. The inclusion of cropping further reduces the mIoU to 14.63%. This drop, consistent with the low score observed during the initial 10-epoch experiment, suggests that random cropping may disrupt the spatial structure of the scenes, making it harder for the model to recognize certain classes, especially small or spatially dependent ones. Overall, these results highlight color jittering as the most effective augmentation technique in this situation. By introducing realistic lighting variations, it significantly improves cross-domains generalization.

3.4. Domain Adaptation

To reduce the domain gap between synthetic and real images, we implemented an adversarial domain adaptation strategy following the approach proposed in [7]. We used GTA5 as the labeled source domain and Cityscapes as the unlabeled target domain. In particular, we adopted a single-level adversarial learning approach, where domain adaptation is performed only on the final segmentation output (i.e., softmax prediction), without applying any adaptation at intermediate feature levels.
The model consists of two main components:
Segmentation Network (G): A convolutional network that produces pixel-wise class probabilities. We employed the BiSeNet architecture, combined with the data augmentation configuration that achieved the highest mIoU score, ensuring consistency across experimental conditions.
Discriminator (D): A fully convolutional network that takes the softmax output and predicts whether it originates from the source or target domain. The discriminator has been implemented as described in the referenced paper.

The training follows a minimax formulation, where the discriminator is trained to distinguish the domain of the predictions, while the segmentation network is trained to fool D by producing domain invariant outputs.
The training process begins by feeding both the source image I_s and the target image I_t into the segmentation network, resulting in predictions P_s and P_t. The prediction $P_s$is compared to the ground truth and the segmentation loss is computed as follows:

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

where $P = G(I) \in \mathbb{R}^{H\times W\times C}$ is the segmentation softmax output. This loss guides the segmentation network to generate more accurate predictions on the source domain.
Additionally, to encourage alignment between target and source domain predictions, we compute the adversarial loss by passing the target prediction P_t through the discriminator:

$$\mathcal{L}_{adv} = -\sum_{h,w} \log D(P_t)^{(h,w,1)}$$

Minimizing this loss promotes the generation of target domain predictions that closely resemble source domain outputs, therefore increasing their likelihood of being classified as source by the discriminator.
The discriminator instead is trained to distinguish whether a prediction originates from the source or target domain. It's loss is defined as:

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

where $z = 1$ for source predictions and $z = 0$ for target predictions.
The full training objective is:

$$\min_G \max_D \ \mathcal{L}_{seg} + \lambda_{adv}\mathcal{L}_{adv} + \mathcal{L}_d$$

where λ_{adv} balances the contribution of the adversarial term.

Table 3 shows that the adversarial approach achieves an mIoU equal to 24.30%, leading to a substantial improvement compared to the score reached with only the augmentation techniques. This result indicates that domain adaptation through adversarial training is effective in reducing the domain gap between synthetic and real-world data. In particular, the adversarial model achieved achieved strong performance on large classes as 'building', 'vegetation' and

Model	mIoU(%)	Road	Sidewalk	Building	Wall	Fence	Pole	Light	Sign	Vegetation	Terrain	Sky	Person	Rider	Car	Truck	Bus	Train	Motorcycle	Bicycle
BiSeNet (Adv)	24.30	55.03	18.04	76.23	16.44	14.47	22.12	15.22	7.08	69.46	8.83	71.98	39.78	1.16	28.77	6.42	4.18	0.18	6.20	0.09

Table 3. mIoU(%) and per-class IoU (%) on Cityscapes for BiSeNet trained on GTA5 using adversarial domain adaptation.

'sky', suggesting that domain alignment was especially effective for global scene components. However, the performance remains low on small or less frequent classes such as 'train', 'rider' and 'bicycle', highlighting the necessity of further improvement in handling fine-grained details and underrepresented categories.

3.5. Impact of Segmentation Loss Functions

In the last step of our analysis, we experimented with different segmentation loss functions to evaluate their impact on domain adaptation performance. To analyze how different aspects of the segmentation task influence domain adaptation, we tested three distinct loss functions: a pixel-wise loss (Focal Loss), a region-based loss (Dice Loss) and a combination of the two. For Focal and Dice loss functions, we considered both a standard and a weighted version to assess the impact of handling class imbalance. In the weighted case we adopted median frequency balancing. Pixel-level loss functions in semantic segmentation focus on each individual pixel to achieve high classification accuracy. These losses compute the discrepancy between predicted pixel values and ground truth labels independently for each pixel. Among these, Focal Loss is a modification of cross-entropy that balances the influence of easy and hard samples by assigning higher weight to hard-to-classify examples and lower weight to well-classified ones, controlled by a focusing parameter γ:

$$L_{\text{focal}} = -\sum_{i=1}^{N}(1-p_{t,i})^{\gamma}\log(p_{t,i})$$

where $p_{t,i}$ is the predicted probability for the correct class of sample i and γ reduces the contribution of well-classified samples. [1]

Region-level loss functions adopt a broader perspective, prioritizing the overall accuracy of object segmentation rather than individual pixels. Their goal is to ensure that the predicted segmentation mask closely matches the ground truth mask in shape and extent. Dice Loss measures the similarity between predicted and target masks by evaluating their overlap.

$$L_{\text{dice}} = 1 - \frac{2\sum_{i=1}^{N} p_i g_i + \epsilon}{\sum_{i=1}^{N} p_i + \sum_{i=1}^{N} g_i + \epsilon}$$

where p_i is the predicted probability for the class, g_i is the binary ground truth indicator (1 if the pixel belongs to the class, 0 otherwise) and ϵ is a smoothing term to stabilize division. [1]

Finally, we combined the two losses into a mixed loss function that balances their complementary strengths, using a parameter $\alpha \in [0, 1]$ to weight the relative importance of Dice Loss and Focal Loss:

$$L_{\text{combined}} = \alpha \cdot L_{\text{dice}} + (1-\alpha) \cdot L_{\text{focal}}$$

This combined loss leverages Dice Loss's ability to optimize global region coverage and Focal Loss's effectiveness in handling class imbalance and hard examples, improving the robustness of segmentation under domain shifts.

In Table 4, it can be noticed that each loss function exhibits distinct behavior in the presence of domain shift. Focal Loss without class weights achieves the highest overall mIoU of 26.17% and consistently performs well across both frequent and rare classes. The weighted Focal Loss shows better performance in underrepresented categories such as 'traffic sign', 'train' and 'bus', suggesting that the weighting scheme helps the model focus more on rare classes. However, this comes at the cost of reduced performance on some frequent classes like 'road' and 'vegetation', indicating a trade-off introduced by the weighting.

Dice Loss, which emphasizes region-level similarity, achieves strong segmentation results on large and contiguous regions such as 'road', 'building' and 'vegetation'. However, it consistently struggles with smaller and fine-grained classes like 'rider' and 'bicycle'. The weighted Dice Loss variant achieves the second-highest overall mIoU of 25.97%, with notable improvements in some foreground classes such as 'person' and 'truck'. This suggests that weighting can guide the model towards more discriminative boundaries for relevant but underrepresented categories. Nonetheless, the performance remains unstable for very small classes like 'train', 'bus' and 'bicycle'.

We chose to combine the weighted Dice loss, which showed better results than its unweighted version, with the unweighted Focal loss with three different values of α equal to 0.3, 0.5 and 0.7 to explore different trade-offs between region-level and pixel-level supervision. Among these combinations, the best mIoU is achieved with $\alpha = 0.7$, where the model benefits from the stronger influence of the Dice component. This configuration leads to improved segmentation in large, contiguous regions. However, the performance on small or rare classes remains limited, confirm-

Model	mIoU(%)	Road	Sidewalk	Building	Wall	Fence	Pole	Light	Sign	Vegetation	Terrain	Sky	Person	Rider	Car	Truck	Bus	Train	Motorcycle	Bicycle
Focal	26.17	72.17	28.18	74.86	14.14	10.86	18.14	15.73	8.86	70.60	13.09	74.70	31.54	2.12	45.25	8.54	1.62	0.44	6.48	0.00
Focal (weighted)	24.60	60.78	22.63	70.22	12.12	11.28	19.35	16.51	14.04	59.66	8.14	67.22	35.42	2.03	33.48	9.26	15.49	4.41	3.48	1.96
Dice	24.36	68.29	14.28	69.75	14.88	8.73	19.80	16.95	5.08	67.51	7.73	58.68	42.84	0.60	50.80	12.24	0.00	0.00	4.65	0.00
Dice (weighted)	25.97	68.73	9.73	65.05	11.88	15.94	15.39	16.30	13.10	54.46	6.58	63.87	46.45	3.45	68.58	16.26	10.48	0.00	3.26	3.93
Focal + Dice ($\alpha = 0.5$)	22.99	46.89	15.91	73.15	15.27	9.60	20.19	17.20	5.19	68.30	11.22	70.22	33.78	4.49	22.76	7.82	2.11	3.14	9.46	0.19
Focal + Dice ($\alpha = 0.7$)	23.27	46.63	15.06	69.57	13.59	13.07	19.61	16.32	6.04	70.24	8.85	71.15	37.54	2.36	34.03	5.96	2.69	0.00	9.46	0.00
Focal + Dice ($\alpha = 0.3$)	22.06	45.66	13.21	72.37	12.84	9.12	17.09	9.85	2.40	69.80	9.08	71.05	26.72	6.88	22.23	11.04	2.85	6.42	9.86	0.65

Table 4. mIoU (%) and per-class IoU (%) on Cityscapes for BiSeNet trained on GTA5 with various loss functions for segmentation.

ing the challenge of relying on region-based supervision for fine-grained categories under domain shift.

The configuration with $\alpha = 0.5$ yields slightly lower overall performance, suggesting that an equal weighting between pixel-wise and region-wise losses may decrease their respective benefits without effectively addressing their weaknesses.

The setting with $\alpha = 0.3$, which emphasizes Focal Loss, achieves the lowest mIoU but shows relatively better results on some difficult foreground categories such as 'truck', 'rider' and 'motorcycle'. This suggests that a stronger emphasis on the pixel-level Focal Loss might help in handling class imbalance and sparsity, though at the cost of general segmentation performance.

In this situation, none of the combined variants surpasses the individual losses, highlighting the importance of tailoring the loss function to the specific challenges of domain adaptation and the characteristics of the target dataset.

4. Discussion and Findings

Our initial experiments established a performance baseline on the Cityscapes dataset, showing comparable mIoU scores for both DeepLabV2 (55.01%) and BiSeNet (54.55%). However, BiSeNet outperformed DeepLabV2 significantly in terms of inference speed, making it a more suitable option for real-time applications. This efficiency aligns with BiSeNet's architectural goal of decoupling spatial detail from semantic context, optimizing performance while maintaining speed. In contrast, the use of median frequency balancing loss consistently led to a drop in performance, suggesting that it may have overcompensated for class imbalance in this particular context.

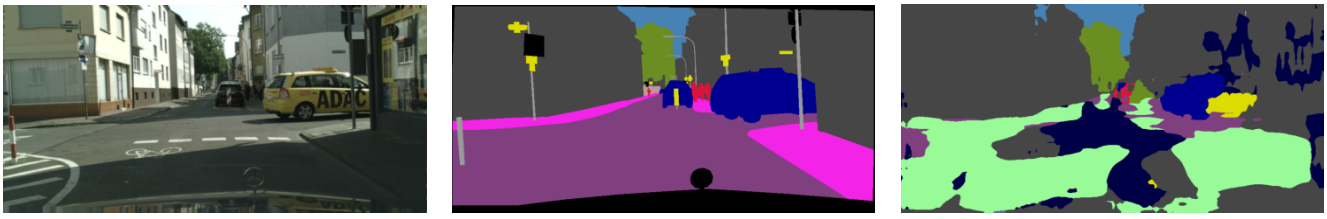
When evaluating BiSeNet trained on the synthetic GTA5 dataset and tested on Cityscapes, we observed a significant performance decline, with a mIoU of only 14.57%. This degradation underscores the importance of domain adaptation techniques when deploying models trained on synthetic data in real-world scenarios.

Our experiments revealed that color jittering alone was the most effective augmentation technique, achieving an mIoU of 19.31%. This improvement indicates that simulating varying lighting conditions and color variations plays a key role in enhancing the model's ability to generalize across domains.

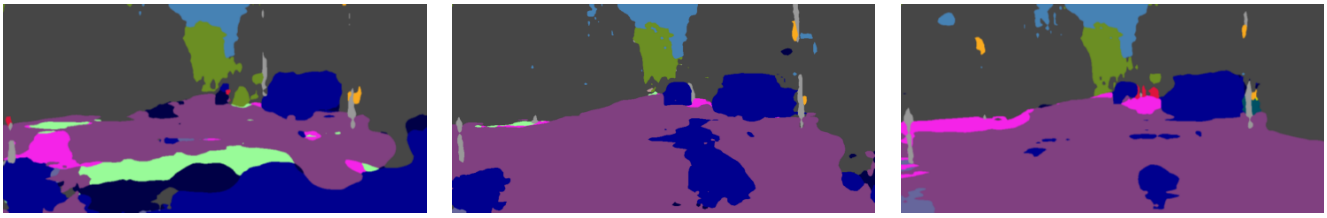
Furthermore, the implementation of an adversarial domain adaptation strategy resulted in a notable performance boost, increasing the mIoU to 24.30%. The adversarial model performed particularly well on larger, globally present classes, but struggled with smaller and less frequent classes. This limitation highlights an ongoing challenge in segmenting fine-grained details and handling highly imbalanced classes within complex scenes.

Finally, our analysis of different segmentation loss functions demonstrated that the standard Focal Loss improved the recognition of small objects, reaffirming its effectiveness in addressing class imbalance. In contrast, Dice Loss, which focuses on region-level similarity, performed well for large, continuous regions. However, combining the two losses did not lead to further improvements. This lack of progress may be due to the fact that each loss function targets different aspects of the segmentation task, whose effects may conflict during the optimization process.

In Figure 3, we present a qualitative overview of our semantic segmentation results. The initial prediction, without any domain shift mitigation, is highly fragmented. However, as data augmentation using jitter is applied, basic spatial coherence begins to emerge. Adversarial adaptation significantly improves the global structure of the segmentation and the model trained with Focal Loss, without class weighting, produces the most precise and detailed masks. This progression demonstrates the cumulative impact of our methodological improvements on the visual quality of the segmentation. Despite these advances, the models continue to struggle with accurately segmenting small and underrepresented classes. For example, categories such as 'poles', 'traffic lights' and 'traffic signs' are not well masked. This is likely due to the severe class imbalance present in the datasets, which biases the model towards more frequent categories. Nonetheless, further refinements are required to



(a) Input Image (b) Ground Truth (c) Prediction (No Adaptation)



(d) Prediction (Jittering) (e) Prediction (Adversarial) (f) Prediction (Focal Loss)

Figure 3. Qualitative comparison between input, ground truth and segmentation predictions under different settings.

fully recover the true mask.

5. Conclusion

Domain shift remains a major challenge when training semantic segmentation models on synthetic data and deploying them on real-world images. Visual discrepancies between the source and target domains often result in incorrect predictions and a substancial performance degradation. Augmentation techniques can enhance cross-domain generalization. However, their effectiveness in this situation was limited and insufficient to fully overcome the domain gap. Future work could explore more advanced augmentation strategies or style transfer techniques to improve the model's robustness across domains. Our results showed that combining adversarial domain adaptation with tailored loss functions can further improve cross-domain segmentation, especially for large and prominent scene elements. Nonetheless, accurately segmenting smaller or underrepresented classes remains difficult, especially in the absence of labeled target data. Addressing this limitation will be essential to developing more reliable and generalizable segmentation models for real-world applications.

References

[1] Reza Azad, Moein Heidary, Kadir Yilmaz, Michael Hüttemann, Sanaz Karimijafarbigloo, Yuli Wu, Anke Schmeink, and Dorit Merhof. Loss functions in the era of semantic segmentation: A survey and outlook, 2023. 6

[2] 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

[3] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3213–3223, 2016. 2

[4] Shijie Hao, Yuan Zhou, and Yanrong Guo. A brief survey on semantic segmentation with deep learning. *Neurocomputing*, 406:302–321, 2020. 3

[5] David Mash et al. Data augmentation in classification and segmentation: A survey and new strategies. *Journal of Imaging*, 9(2):46, 2023. 4

[6] Stephan R Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for data: Ground truth from computer games. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 102–118. Springer, 2016. 2

[7] 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, 2020. 3, 5

[8] Changqian Yu, Jingbo Wang, Chao Peng, Changxin Gao, Gang Yu, and Nong Sang. Bisenet: Bilateral segmentation network for real-time semantic segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 325–341, 2018. 2